Universidad de La Laguna

Escuela de Doctorado y Estudios de Posgrado

TÍTULO DE LA TESIS DOCTORAL

Aplicación de técnicas de detección visual para la navegación en entornos con peatones

---

AUTOR/A

JAVIER	HERNANDEZ	ACEITUNO

DIRECTOR/A

LEOPOLDO	ACOSTA	SANCHEZ

CODIRECTOR/A

JOSE DEMETRIO	PIÑEIRO	VERA

DEPARTAMENTO O INSTITUTO UNIVERSITARIO

FECHA DE LECTURA

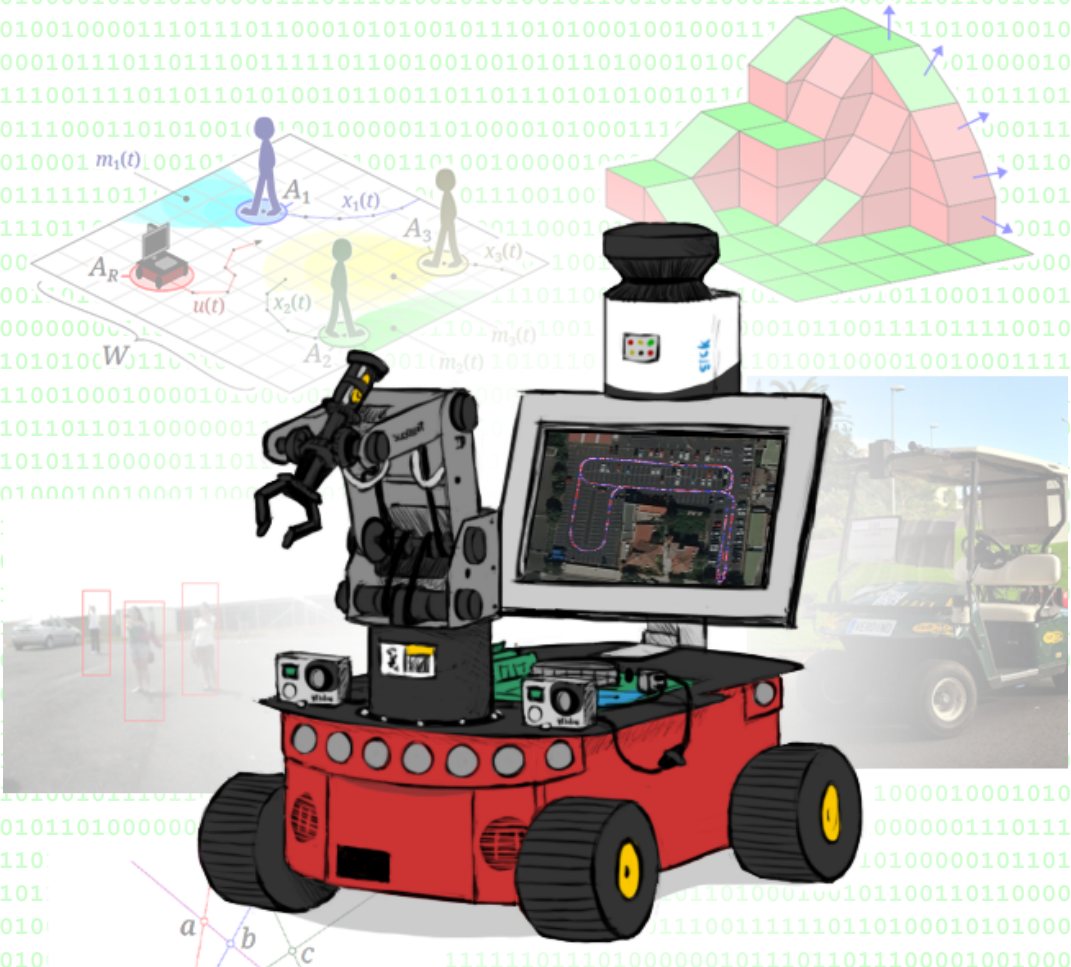
03/02/16

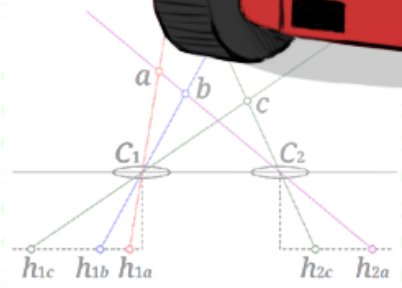
Universidad de La Laguna
Departamento de Ingeniería Informática y de Sistemas

Aplicación de técnicas de detección visual para la navegación en entornos con peatones

Javier Hernández Aceituno

2016





DIRECTORES:
Leopoldo Acosta Sánchez
José Demetrio Piñeiro Vera

Universidad de La Laguna

Escuela Técnica Superior de Ingeniería Informática
Departamento de Ingeniería Informática

Tesis doctoral

Aplicación de técnicas de detección visual para la navegación en entornos con peatones

Dirigida por:
Leopoldo Acosta Sánchez
José Demetrio Piñeiro Vera

Javier Hernández Aceituno

2015

Agradecimientos

Agradezco a mis directores de Tesis, Leopoldo Acosta Sánchez y José Demetrio Piñeiro Vera, su guía y tutela en este proyecto, todo el conocimiento que me han brindado y toda su paciencia e inestimable ayuda, sin las cuales este trabajo no habría visto nunca la luz.

Doy también gracias a Antonio Morell González, a Rafael Arnay del Arco, a Néstor Morales Hernández, a Jesús Javier Espelosín Ortega, a José Daniel Perea Ström, a Francisco Fumero Batista, a Iván Castilla Rodríguez, a Kelin Victoria Zúñiga Meneses, a Jonay Tomás Toledo Carrillo, a Eusebio Morell González, a Ángela Hernández López, a Dailos Reyes Díaz, a Juan Albino «Alexis» Méndez Pérez, a Isabel Sánchez Berriel y a todos los demás compañeros del Departamento de Ingeniería Informática y de Sistemas, por su valioso compañerismo y su apoyo técnico y moral.

Gracias en especial a mis padres y a mi hermana, por la infinita paciencia que me han tenido durante la realización de este trabajo de tesis doctoral.

Esta tesis ha sido realizada con el apoyo económico de una beca CajaCanarias correspondiente a la convocatoria del año 2011, de una ayuda del programa de Formación de Profesorado Universitario (FPU2012-3568) del Ministerio de Ciencia e Innovación, y de dos proyectos del Plan Nacional de I+D+i, SAGENIA (DPI2010-18349) y STIRPE (DPI2013-46897-C2-1-R).

Índice

Índice de tablas

Índice de figuras

Índice de algoritmos

Introducción

La navegación autónoma es uno de los elementos más llamativos del campo de la robótica. Por lo general, toda máquina capaz de actuar y tomar decisiones por sí misma requiere un sistema sensorial bien calibrado y una inteligencia artificial fiable, que aseguren su propia protección y la de los elementos de su entorno. Un vehículo autoguiado, además, debe velar por la integridad física de sus tripulantes y de los peatones que pueblen su área de trabajo.

Durante mi estancia en el Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, he tenido la oportunidad de trabajar y presentar públicamente en el prototipo VERDINO, un ambicioso proyecto de navegación autónoma, vigente desde hace ya más de diez años. Por supuesto, como estudiante de doctorado, mis aportaciones a este dispositivo han sido leves, pero quiero pensar que he influido positivamente en su desarrollo, al menos hasta cierto punto.

El funcionamiento de este vehículo se divide en dos etapas. Durante la primera, antes de ser capaz de navegar de forma independiente, se debe generar un mapa de la zona a recorrer, de forma que se puedan definir rutas eficientes. La codificación de estos mapas depende, por supuesto, de los sensores que se vayan a utilizar para identificar la posición relativa del dispositivo a localizar. El mapa de los nombres de las calles de una ciudad resultaría completamente inútil a un dispositivo incapaz de reconocer texto, así como una fotografía capturada por un satélite no ofrecería la menor información a un turista perdido.

En nuestro caso, el sistema de codificación más eficiente, respecto a la apariencia física del entorno del robot, es un mapa de profundidad. Existen multitud de dispositivos y algoritmos para obtener este tipo de datos, como se explicará en el texto de esta tesis, pero el principio básico de este enfoque es la traducción de los elementos estáticos de un área transitable a obstáculos fijos en un plano bidimensional. La principal ventaja de este tipo de datos es que es invariante a la iluminación y, dependiendo del sensor, inmune a transparencias.

La segunda etapa del proceso consiste en utilizar el mapa generado para guiarse dentro del área de trabajo. Conociendo la localización y orientación inicial del vehículo, el sistema de navegación del dispositivo genera una trayectoria tan eficiente y segura como sea posible, que lo conduzca hasta una posición objetivo dada por su tripulante, y la recorre de forma automatizada, sin necesidad de interacción humana.

Puesto que los entornos poblados son dinámicos por definición, estas rutas deben ser adaptables. En caso de observar un obstáculo imprevisto, como por ejemplo un peatón cruzando la calle, el robot deberá maniobrar para evitarlo o detenerse para no producir accidentes. Es por tanto vital que todo vehículo autoguiado cuente con un sistema sensorial eficiente y una capacidad de detección de obstáculos infalible, a fin de asegurar una navegación segura y libre de contratiempos.

El presente trabajo de tesis se divide en tres partes: la primera, que incluye los dos primeros capítulos, hace un estudio analítico de la capacidad para la detección visual de los sensores de estereovisión, telemetría y la fusión de ambos; la segunda, correspondiente a los capítulos 3 y 4, asciende un nivel de abstracción, analiza la precisión de algunas herramientas de detección de peatones y propone mecanismos de mejora basados en la utilización de secuencias de imágenes; la última, que comprende el resto del trabajo de tesis, emplea las técnicas desarrolladas en la navegación y generación automática de mapas de entornos dinámicos poblados.

Capítulo

1

Análisis matemático de la estereovisión variable

La estereovisión es una de las técnicas más útiles para reconstruir escenas tridimensionales a partir de imágenes planas. Para un montaje de dos cámaras enfocadas en la misma dirección y separadas una distancia conocida, se puede calcular la profundidad a la que se encuentran los objetos visibles (Fig. 1.1), utilizando la disparidad horizontal observada entre ellos (Lazaros et al., 2008).

Figura 1.1. Ejemplo de imágenes capturadas para estereovisión.

1.1. Cálculo de la profundidad mediante estereovisión

Para un punto $P = (x, y, z)$, cada cámara generará una proyección desplazada una distancia h desde su centro, que será proporcional a x (Fig. 1.2); conociendo la distancia focal f de la cámara, obtenemos la relación $x/y = h/f$. Por simplicidad, asumiremos que todas las cámaras utilizadas son iguales y que, por tanto, comparten un mismo valor de f.

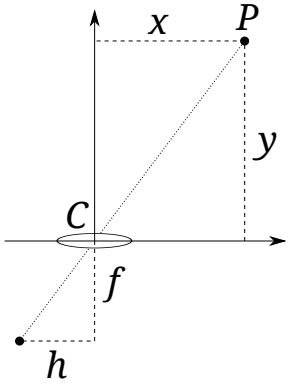


Figura 1.2. Proyección del punto P por la cámara C.

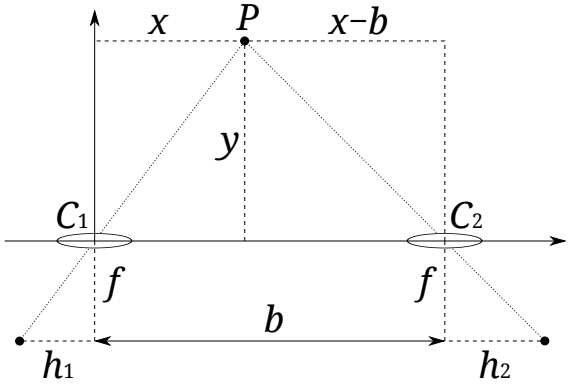


Figura 1.3. Proyección del punto P por las cámaras C_1 y C_2.

Esto no es suficiente para calcular la profundidad y del punto, ya que x no es un valor conocido. Sin embargo, para dos cámaras C_1 y C_2 separadas una distancia conocida b (Fig. 1.3), los valores h pueden utilizarse para calcular el valor y resolviendo el sistema de ecuaciones (1.1); denotaremos como $\langle h_1, h_2 \rangle$ al punto cuya posición puede obtenerse a partir de los valores h_1 y h_2:

$$\left.\begin{aligned} \frac{x}{y} &= \frac{h_1}{f} \\ \frac{x-b}{y} &= \frac{h_2}{f} \end{aligned}\right\} \Rightarrow (x, y) = \langle h_1, h_2 \rangle = \left(\frac{h_1 \cdot b}{h_1 - h_2}, \frac{f \cdot b}{h_1 - h_2} \right) \tag{1.1}$$

1.1.1. Ambigüedades

Una imagen contendrá por lo general muchos puntos cuya profundidad quiera conocerse y, si estos son similares entre sí, pueden producirse ambigüedades a la hora de identificar qué punto observado por una cámara corresponde a qué punto observado por la otra. Se conoce como *problema de la correspondencia* al proceso de emparejar correctamente los puntos visibles (Ogale and Aloimonos, 2005).

Un primer paso para resolver este problema consiste en aplicar la *restricción epipolar*, según la cual dos puntos observados sólo pueden corresponderse entre sí si se encuentran a la misma altura z respecto a la horizontal relativa de las cámaras (Fig. 1.4). Sin embargo, para imágenes que contengan varios objetos similares, esta restricción no es suficiente.

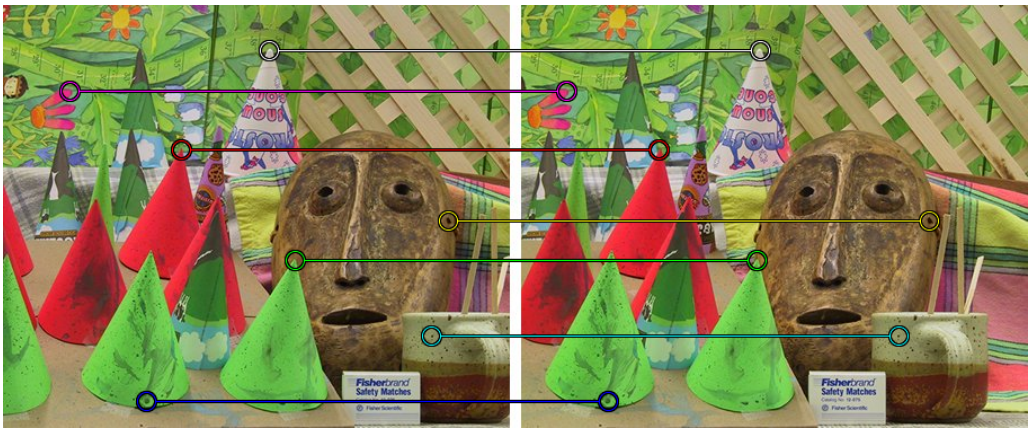


Figura 1.4. Demostración de la restricción epipolar.

Otra de las técnicas más habituales para resolver las ambigüedades es la adquisición de más de dos imágenes simultáneas de la escena para comparar por el proceso de reconstrucción. Para ello es necesario disponer de más de dos cámaras (estereovisión múltiple) o permitir que la configuración espacial del par de cámaras varíe (estereovisión variable).

Es este capítulo se estudia la eficiencia de los sistemas de estereovisión variable a la hora de resolver el problema de la correspondencia. Para ello, se describe matemáticamente la relación entre sus parámetros de configuración y la dificultad del proceso de emparejamiento de puntos. Este estudio sigue un enfoque estrictamente numérico, asumiendo que las cámaras de las que se dispone son estenopeicas con precisión exacta, y considerando dos configuraciones concretas: un sistema de estereovisión lineal clásico, con dos cámaras a distancia variable, y un par de cámaras a distancia fija con orientación variable.

1.2. Medida de la disparidad entre combinaciones de valores

Cuando se producen ambigüedades, cada cámara disponible (C_1 y C_2) llevará asociado un conjunto de valores de proyección h (H_1 y H_2 respectivamente) para una altura determinada. Los elementos de estos conjuntos deben emparejarse correctamente entre sí para generar, mediante el sistema de ecuaciones (1.1), las posiciones tridimensionales de los puntos. Si el emparejamiento es incorrecto, se obtendrá un conjunto de *puntos espurios* en posiciones diferentes a las de los puntos reales (Fig. 1.5).

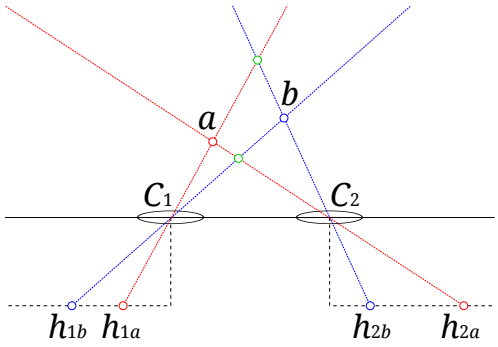


Figura 1.5. Ejemplo de ambigüedad para los puntos a y b. Los puntos espurios se muestran en verde.

Al variar la posición u orientación de una de las cámaras se obtendrá una tercera imagen, con su correspondiente conjunto de valores de proyección (H_3). Sabemos que, para todo par de conjuntos H, debe existir necesariamente al menos

una combinación de valores que produzca las posiciones correctas de los puntos. Tomando dos pares cualesquiera de conjuntos, podemos obtener los puntos reales como aquellos cuyas posiciones coinciden en ambos casos, para alguna de las combinaciones de valores.

Por ejemplo, en la figura 1.6 los puntos a y b producen tres conjuntos de valores: $H_1 = \{h_{1a}, h_{1b}\}$ para la cámara C_1, $H_2 = \{h_{2a}, h_{2b}\}$ para la cámara C_2 y $H_3 = \{h_{3a}, h_{3b}\}$ para el resultado de trasladar la cámara C_2 hacia la derecha. De combinar los conjuntos H_1 y H_2 se puede obtener los puntos $a = \langle h_{1a}, h_{2a}\rangle$ y $b = \langle h_{1b}, h_{2b}\rangle$, junto a dos puntos espurios, $\langle h_{1a}, h_{2b}\rangle$ y $\langle h_{1b}, h_{2a}\rangle$ (en verde en el gráfico), mientras que de la combinación de H_1 y H_3 se obtiene $a = \langle h_{1a}, h_{3a}\rangle$ y $b = \langle h_{1b}, h_{3b}\rangle$, junto a los puntos espurios $\langle h_{1a}, h_{3b}\rangle$ y $\langle h_{1b}, h_{3a}\rangle$ (en amarillo). Puesto que sólo las posiciones de a y b coincidirán en ambos casos, los puntos espurios pueden descartarse.

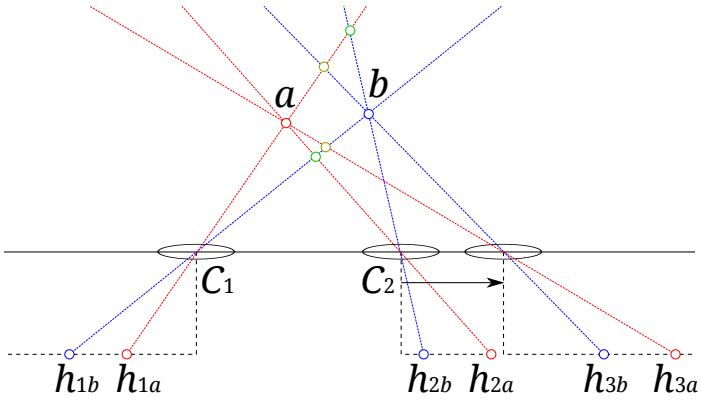


Figura 1.6. Resolución de la ambigüedad mediante estereovisión variable.

Sin embargo, si las posiciones de los puntos espurios son similares, esta discriminación podría inducir a error. Es por ello que deseamos situar las cámaras de tal forma que los puntos reales resulten inconfundibles, a pesar de los posibles errores de cálculo y del ruido de los dispositivos de entrada.

Se comienza pues por definir una medida de disparidad d entre dos puntos,

resultantes de la combinación de un valor de proyección de cada conjunto H, como la distancia euclídea entre los mismos:

$$d(h_x, h_y, h_z) = \left\| \langle h_x, h_y \rangle - \langle h_x, h_z \rangle \right\|_2 \tag{1.2}$$

El valor d depende del método utilizado para generar el conjunto de valores de proyección H_3. En las dos subsecciones siguientes se estudiará este valor para pares de cámaras a distancia variable (1.2.1) y para pares de cámaras con orientación variable (1.2.2).

Se define a continuación una medida de disparidad acumulada D, como la suma de los valores d de todas las combinaciones compatibles entre sí de tres conjuntos H — es decir, aquellas que no toman valores h repetidos. Por ejemplo, para dos conjuntos $H_A = \{h_{a1}, h_{a2}\}$ y $H_B = \{h_{b1}, h_{b2}\}$, se asume que el punto hipotético $\langle h_{a1}, h_{b2} \rangle$ no puede existir al mismo tiempo que los puntos $\langle h_{a1}, h_{b1} \rangle$ y $\langle h_{a2}, h_{b2} \rangle$, ya que se estarían reutilizando valores de proyección. Por tanto, el punto $\langle h_{a1}, h_{b2} \rangle$ sólo es compatible con el punto $\langle h_{a2}, h_{b1} \rangle$.

Si llamamos H_2^P y H_3^P a permutaciones concretas de los conjuntos H_2 y H_3, de forma que el valor $n-$ésimo de H_1 se empareje con los valores $n-$ésimos de H_2^P y H_3^P, la disparidad acumulada D para un caso específico de combinaciones de los conjuntos H se calcularía como:

$$D\left(H_1, H_2^P, H_3^P\right) = \sum_{i=1}^{|H_1|} d\left(h_{1i},\, H_{2i}^P,\, H_{3i}^P\right) \tag{1.3}$$

Este cálculo asume que los cardinales de los tres conjuntos de partida H son iguales; el caso contrario se estudiará en detalle en la subsección 1.2.3.

Puesto que se asume un cierto error de cálculo en las medidas reales, se describe un *valor umbral* Δ_D igual a la diferencia entre el valor D correspondiente a la combinación correcta de valores h y el menor valor D perteneciente a una combinación errónea. Las configuraciones óptimas de pares de cámaras son las que maximizan este valor umbral Δ_D, como se estudiará detalladamente en la

sección 1.3 de este capítulo.

1.2.1. Variación en longitud

El primer método que estudiamos, por el cual se produce una modificación en la posición relativa del par de cámaras, consiste en variar la distancia b entre ellas. Llamaremos Δ_b a la variación de este valor, entendida como la distancia recorrida por la cámara C_2 al alejarse de la cámara C_1. A partir del conjunto de valores de proyección resultante de la nueva posición H_3, se puede definir el cálculo de la posición de un punto a partir de la ecuación (1.1):

$$\langle h_1, h_2 \rangle = \left(\frac{h_1 \cdot b}{h_1 - h_2}, \frac{f \cdot b}{h_1 - h_2} \right) \Rightarrow \langle h_1, h_3 \rangle = \left(\frac{h_1 \cdot (b + \Delta_b)}{h_1 - h_3}, \frac{f \cdot (b + \Delta_b)}{h_1 - h_3} \right)$$

La distancia d entre dos puntos generados a partir de los valores h, para un sistema de estereovisión variable en longitud, se calcula entonces como

$$\begin{aligned}
d(h_1, h_2, h_3) &= \| \langle h_1, h_2 \rangle - \langle h_1, h_3 \rangle \|_2 \\
&= \sqrt{\left(\frac{h_1 \cdot b}{h_1 - h_2} - \frac{h_1 \cdot (b + \Delta_b)}{h_1 - h_3} \right)^2 + \left(\frac{f \cdot b}{h_1 - h_2} - \frac{f \cdot (b + \Delta_b)}{h_1 - h_3} \right)^2} \\
&= \sqrt{(h_1^2 + f^2) \cdot \left(\frac{b}{h_1 - h_2} - \frac{b + \Delta_b}{h_1 - h_3} \right)^2} \\
&= \sqrt{h_1^2 + f^2} \cdot \left| \frac{(h_1 - h_3)\, b - (h_1 - h_2)\, b}{(h_1 - h_2)\,(h_1 - h_3)} - \frac{\Delta_b}{h_1 - h_3} \right| \\
&= \left| \frac{\sqrt{h_1^2 + f^2}}{h_1 - h_3} \cdot \left(\frac{h_2 - h_3}{h_1 - h_2} \cdot b - \Delta_b \right) \right| \qquad (1.4)
\end{aligned}$$

Esta disparidad debe ser nula si se ha escogido la combinación correcta de valores, y no tendrá un resultado válido si h_1 es igual a h_2 ó h_3. Esto puede ocurrir porque la distancia entre dos de las cámaras sea nula ($b = 0$, $\Delta_b = 0$ ó

$\Delta_b = -b$) y ambas produzcan la misma información. Si esto es así, una de las cámaras es redundante y sus datos no contribuyen al resultado final.

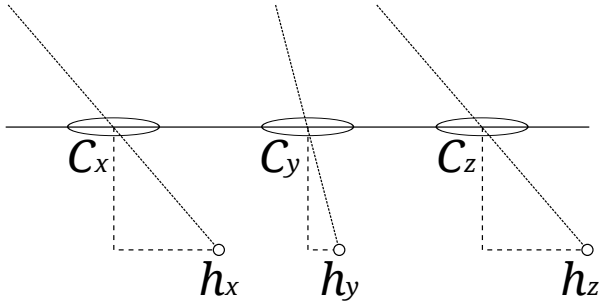


Figura 1.7. Ejemplo de rayos de luz paralelos ($h_x = h_z$) y divergentes ($h_x > h_y$).

En caso contrario (Fig. 1.7), si dos valores h_x y h_z coinciden, los rayos de luz que los generan son paralelos y, por tanto, el punto hipotético $\langle h_x, h_z \rangle$ estaría situado en el infinito. Si esto ocurre, la combinación de permutaciones de los conjuntos H utilizada para generarlo es incorrecta y puede descartarse por completo.

Supongamos ahora que los rayos de luz observados no son paralelos, sino divergentes. Esto se traduce numéricamente en que $h_x > h_y$, si la cámara C_x se encuentra a la izquierda de la cámara C_y, o $h_x < h_y$, si está a la derecha. En este caso, la coordenada y del punto $\langle h_x, h_y \rangle$ sería negativa, lo que lo situaría *detrás* de las cámaras. Dado que esto es imposible, el punto quedaría descartado junto a su combinación correspondiente de valores h.

Al descartarse puntos se hace innecesario calcular los valores de disparidad entre combinaciones de valores, lo que simplifica notablemente el problema. Puesto que la divergencia entre rayos de luz es más probable cuando disminuye la distancia entre cámaras, se deduce que, para cámaras estenopeicas ideales, reducir la separación del par es beneficioso a la hora de resolver el problema de la correspondencia.

1.2.2. Variación en orientación

El segundo método para variar la posición relativa del par de cámaras consiste en situar cada una en un extremo de un cuerpo rígido giratorio cuyo eje de rotación se encontraría en su centro. El radio de rotación por tanto sería $r = b/2$; llamaremos θ al ángulo de orientación.

Al aplicar una rotación se obtiene un conjunto de valores de proyección H_3, a partir de la cámara C_2 en su nueva posición. Sin embargo, el cálculo de los puntos visibles en este caso no es tan simple como para la variación en longitud.

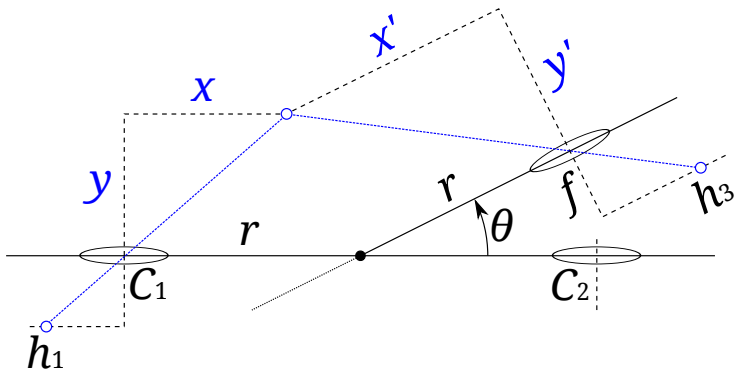


Figura 1.8. Resolución de la ambigüedad mediante variación en rotación.

Como muestra la figura 1.8, el principio básico de la proyección de puntos se cumple para la cámara rotada, $x'/y' = h_3/f$, pero es necesario adaptar las coordenadas de los puntos visibles al nuevo sistema:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x-r \\ y \end{bmatrix} - \begin{bmatrix} r \\ 0 \end{bmatrix} = \begin{bmatrix} y\sin\theta + (x-r)\cos\theta - r \\ y\cos\theta - (x-r)\sin\theta \end{bmatrix}$$

Aunque la posición relativa de las cámaras C_1 y C_2 cuando $\theta = 0$ coincide con

el caso anterior, y por tanto se cumple

$$(x,y) = \langle h_1, h_2 \rangle = \left(\frac{h_1 \cdot b}{h_1 - h_2}, \frac{f \cdot b}{h_1 - h_2} \right) = (h_1, f) \cdot \frac{b}{h_1 - h_2},$$

para calcular la posición de un punto a partir de la combinación de valores de los conjuntos H_1 y H_3, debemos recurrir a un nuevo sistema de ecuaciones:

$$\left.\begin{aligned} \frac{h_1}{f} &= \frac{x}{y} \\ \frac{h_3}{f} &= \frac{y\sin\theta + (x-r)\cos\theta - r}{y\cos\theta - (x-r)\sin\theta} \end{aligned}\right\} \frac{h_3}{f} = \frac{y\sin\theta + \left(\frac{h_1 y}{f} - r\right)\cos\theta - r}{y\cos\theta - \left(\frac{h_1 y}{f} - r\right)\sin\theta}$$

$$\frac{h_3}{f} = \frac{fy\sin\theta + h_1 y\cos\theta - fr\cdot(\cos\theta + 1)}{fy\cos\theta - h_1 y\sin\theta + fr\sin\theta}$$

$$f\cdot(fy\sin\theta + h_1 y\cos\theta - fr(\cos\theta+1)) = h_3\cdot(fy\cos\theta - h_1 y\sin\theta + fr\sin\theta)$$

$$y\cdot\left(f^2\sin\theta + h_1 f\cos\theta - fh_3\cos\theta + h_1 h_3\sin\theta\right) = fr\cdot(h_3\sin\theta + f(\cos\theta+1))$$

$$y = f\cdot\frac{r\cdot(h_3\sin\theta + f(\cos\theta+1))}{(f^2 + h_1 h_3)\sin\theta + (h_1 - h_3) f\cos\theta}$$

$$x = h_1\cdot\frac{r\cdot(h_3\sin\theta + f(\cos\theta+1))}{(f^2 + h_1 h_3)\sin\theta + (h_1 - h_3) f\cos\theta}$$

$$(x,y) = \langle h_1, h_3\rangle = (h_1, f)\cdot\frac{b\cdot(h_3\sin\theta + f(\cos\theta+1))}{2(f^2 + h_1 h_3)\sin\theta + 2(h_1 - h_3) f\cos\theta}$$

La disparidad entre dos puntos, calculados a partir de combinaciones de tres valores h, es:

$$\begin{aligned} d(h_1,h_2,h_3) &= \|\langle h_1,h_2\rangle - \langle h_1,h_3\rangle\|_2 \\ &= \sqrt{(h_1^2 + f^2)\cdot\left(\frac{b}{h_1 - h_2} - \frac{b\cdot(h_3\sin\theta + f(\cos\theta+1))}{2(f^2 + h_1 h_3)\sin\theta + 2(h_1 - h_3) f\cos\theta}\right)^2} \\ &= b\sqrt{h_1^2 + f^2}\left|\frac{1}{h_1 - h_2} - \frac{h_3\sin\theta + f(1+\cos\theta)}{2(h_1 h_3 + f^2)\sin\theta + 2(h_1 - h_3) f\cos\theta}\right| \end{aligned} \tag{1.5}$$

Las condiciones de divergencia también deben tenerse en cuenta en este caso. La relación entre los valores h_1 y h_3 es ahora más compleja, pero la restricción puede simplificarse sabiendo que y debe ser estrictamente mayor que cero para todas las configuraciones válidas.

1.2.3. Efecto de las diferencias entre cardinales

El cálculo del valor umbral Δ_D para dos o más puntos puede verse afectado por los cardinales de los conjuntos H. Una diferencia entre estos indica que se han producido *oclusiones*: existen puntos que se encuentran alineados respecto a alguna de las cámaras, ocultándose entre sí y produciendo un mismo valor de proyección h (Fig. 1.9).

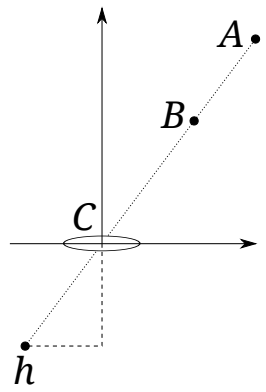


Figura 1.9. Ejemplo de oclusión. El punto A queda oculto a la cámara C tras el punto B.

Si el número de elementos de cualquiera de los conjuntos es 1, todos los puntos vistos desde la cámara correspondiente están alineados. En este caso sólo existirá una permutación posible de sus valores, y su combinación con cualquiera de los otros dos conjuntos H producirá la solución correcta.

De lo contrario, se debe aumentar artificialmente los conjuntos de menor cardinal, repitiendo algunos de sus elementos (Fig. 1.10). Puesto que es imposible conocer a priori qué puntos están produciendo oclusiones, todas las posibles repeticiones de valores h tendrán que ser estudiadas.

Se deberá generar las permutaciones de cada conjunto aumentado para resolver el problema de la forma habitual. Aunque este proceso aparentemente incrementaría la complejidad del problema, el número de permutaciones a analizar puede incluso reducirse para conjuntos aumentados.

Sea n el cardinal deseado para un conjunto H formado a partir de m valores

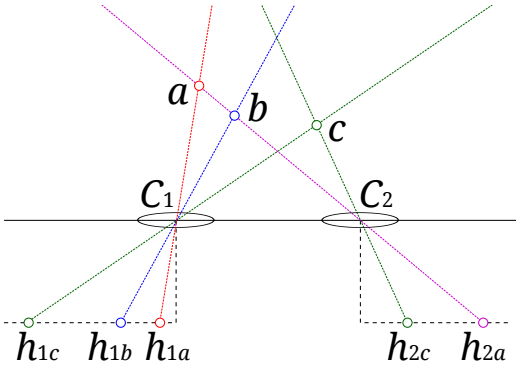


Figura 1.10. Ejemplo de diferencia de cardinales entre conjuntos H. Para generar los puntos a y b, el valor h_{2a} debe repetirse en el conjunto H_2 aumentado.

de proyección h diferentes. El número total de aumentos posibles de H es m^{n-m}, ya que se debe generar $n-m$ nuevos elementos a partir de m valores posibles.

Cabría pensar que un conjunto a aumentar produce $n!$ permutaciones por cada uno de sus m^{n-m} aumentos posibles, pero dado que los conjuntos aumentados contienen valores repetidos, muchas de sus permutaciones son superfluas.

Llamemos R_i al número de repeticiones del $i-$ésimo valor h, tal que $R_1 + R_2 + \cdots + R_m = n$. Cada valor aparece en el conjunto aumentado al menos una vez ($R_i \geq 1$) para no descartar puntos visibles. Para obtener el número máximo de repeticiones del $i-$ésimo valor, al que llamaremos M_i, se resta del cardinal deseado n el número de repeticiones de los valores anteriores (R_1 a R_{i-1}) y al menos una aparición de cada uno de los $m-i$ valores posteriores:

$$M_i = n - \sum_{j=1}^{i-1} R_j - (m-i).$$

Puesto que la suma del número de repeticiones de todos los elementos es igual a n, el $m-$ésimo elemento deberá ocupar todos los espacios que le queden disponibles, es decir, $R_m = n - (R_1 + R_2 + \cdots + R_{m-1}) = M_m$.

El número de permutaciones únicas de un conjunto aumentado de tamaño n viene dado por

$$PR_n^{R_1, R_2, \cdots, R_m} = \frac{n!}{R_1! R_2! \cdots R_m!}$$

De esto obtenemos que el número de permutaciones únicas para todos los posibles aumentos de tamaño n de un conjunto de tamaño m es

$$Q_n^m = \sum_{R_1=1}^{M_1} \sum_{R_2=1}^{M_2} \cdots \sum_{R_{m-1}=1}^{M_{m-1}} PR_n^{R_1,R_2,\cdots,R_{m-1},M_m} \tag{1.6}$$

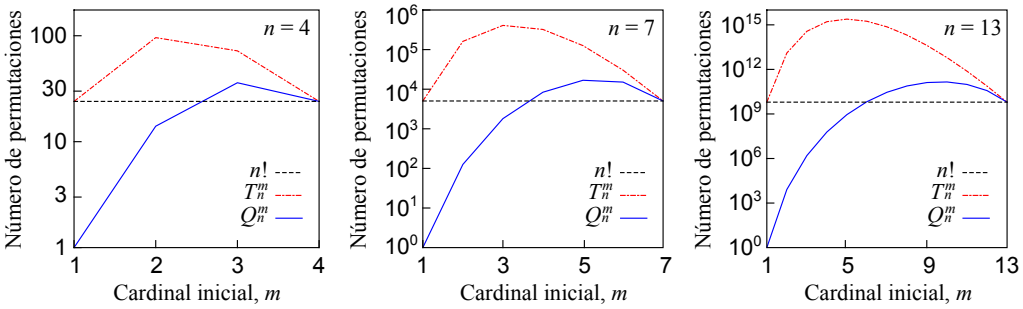


Figura 1.11. Número de permutaciones para variaciones de m y n, donde $T_n^m = n! \cdot m^{n-m}$.

Veamos como ejemplo el caso presentado en la figura 1.10. El conjunto de valores $H_2 = (h_{2a}, h_{2c})$ debe aumentarse para igualar el cardinal de $H_1 = (h_{1a}, h_{1b}, h_{1c})$ (es decir, $n = 3$ y $m = 2$). H_2 tiene $m^{n-m} = 2$ aumentos, $H_2^1 = (h_{2a}, h_{2a}, h_{2c})$ y $H_2^2 = (h_{2a}, h_{2c}, h_{2c})$, cada uno de los cuales tiene $n! = 6$ permutaciones posibles. Sin embargo, descartando las repeticiones, H_2^1 y H_2^2 tienen sólo $PR_3^{2,1} = 3$ y $PR_3^{1,2} = 3$ permutaciones únicas respectivamente:

$$H_2 = (h_{2a}, h_{2c}) \begin{cases} H_2^1 \rightarrow & (h_{2a}, h_{2a}, h_{2c}),\ (h_{2a}, h_{2c}, h_{2a}),\ (h_{2c}, h_{2a}, h_{2a}) \\ H_2^2 \rightarrow & (h_{2a}, h_{2c}, h_{2c}),\ (h_{2c}, h_{2a}, h_{2c}),\ (h_{2c}, h_{2c}, h_{2a}) \end{cases}$$

Por tanto, el número de permutaciones válidas del conjunto H_2 aumentado es

$$Q_3^2 = \sum_{R_1=1}^{3-(2-1)} PR_3^{R_1,(3-R_1)} = PR_3^{1,2} + PR_3^{2,1} = 6.$$

1.3. Resultados

A continuación analizamos la capacidad de un par de cámaras variable para evitar ambigüedades, como se explicó en la sección 1.2. Para ello definimos dos puntos P_1 y P_2, cuyas posiciones desean calcularse mediante estereovisión, y variamos su localización relativa de forma conjunta. Si denotamos por $P_0 = (x_0, y_0)$ al punto medio del segmento $\overline{P_1P_2}$, por δ a su longitud y por α al ángulo que forma con el eje x (Fig. 1.12), las posiciones de los puntos de interés son $P_1 = \left(x_0 + \frac{\delta}{2}\cos\alpha,\ y_0 + \frac{\delta}{2}\sin\alpha\right)$ y $P_2 = \left(x_0 - \frac{\delta}{2}\cos\alpha,\ y_0 - \frac{\delta}{2}\sin\alpha\right)$.

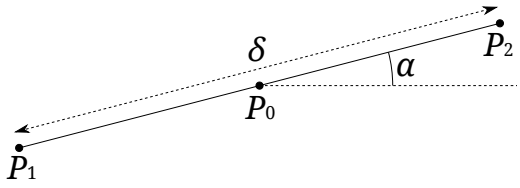


Figura 1.12. Parámetros de configuración de los puntos a localizar.

Para estudiar un par de cámaras variable, observamos el valor umbral Δ_D frente a variaciones de la longitud b y diferencia en longitud Δ_b o en orientación θ del par de cámaras, y de los parámetros de localización de los puntos de interés, x_0, y_0, δ y α. Por defecto, se toman los valores $b = 10$, $P_0 = (50, 50)$, $\delta = 5$ y $\alpha = 0$.

En las gráficas se muestran en blanco aquellas áreas en las que no existen ambigüedades. Como se aprecia en las figuras 1.13 a 1.16, esto ocurre siempre que la distancia entre dos cámaras cualesquiera es menor que la separación entre los puntos, es decir, $|b| < \delta$ y $|b + \Delta_b| < \delta$ (descartando $b = 0$, $\Delta_b = 0$ y $b + \Delta_b = 0$, que implicarían la superposición de dos de las cámaras).

Aunque la posición de los puntos no afecta al cálculo de su localización utilizando cámaras ideales, su orientación conjunta α sí modifica el valor umbral Δ_D, ya que altera la separación entre los puntos que es percibida por las cámaras (Fig. 1.17). Por ejemplo, para $\alpha = \tan^{-1}(y_0/x_0)$ los puntos quedan alineados respecto a la cámara 1, lo que eliminaría las ambigüedades como se explicó en la

sección 1.2.3. En cambio, $\alpha = \tan^{-1}(y_0/x_0) \pm 90°$ maximiza la separación observable entre puntos y, en consecuencia, también el valor umbral.

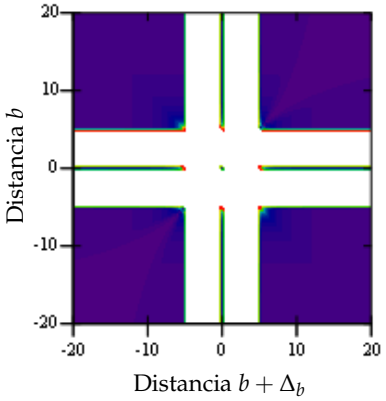


Figura 1.13. Valor umbral Δ_D para variaciones de la distancia entre las cámaras antes y después de la traslación.

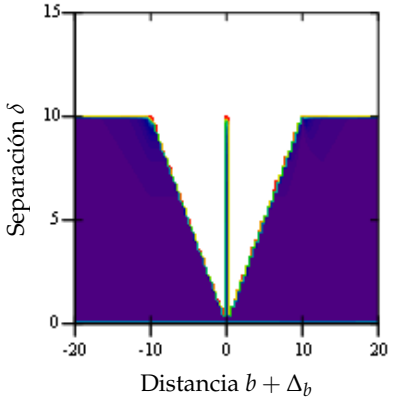


Figura 1.14. Valor umbral Δ_D para variaciones de la separación entre puntos y de la distancia entre las cámaras.

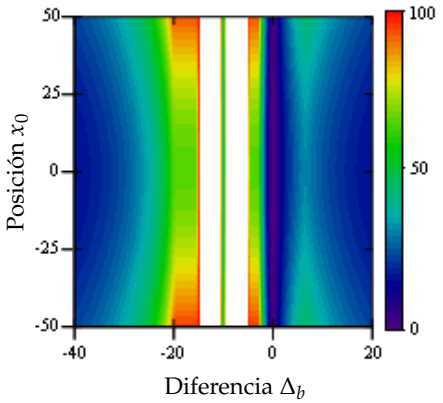


Figura 1.15. Valor umbral Δ_D para variaciones de la posición horizontal de los puntos y de la diferencia de distancia entre cámaras.

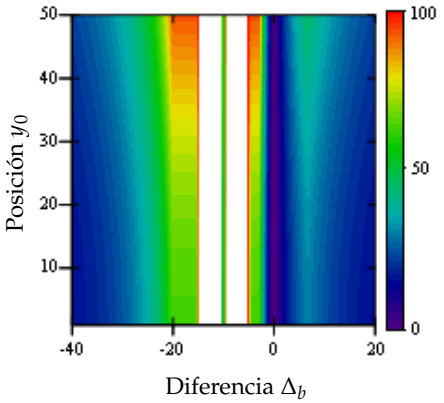


Figura 1.16. Valor umbral Δ_D para variaciones de la posición vertical de los puntos y de la diferencia de distancia entre cámaras.

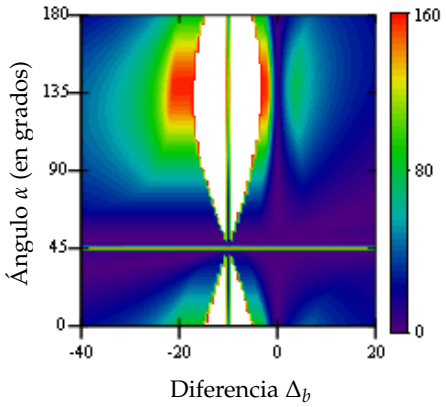


Figura 1.17. Valor umbral Δ_D para variaciones de la orientación de los puntos y de la diferencia de distancia entre cámaras.

La relación entre la separación aparente de los puntos y la distancia entre las cámaras se mantiene cuando varía la orientación del par estereovisor, como se puede observar en las figuras 1.18 y 1.19. Sin embargo, esta variación sí afecta al valor umbral, ya que orientaciones particulares de las cámaras pueden producir imágenes más claras de los puntos a analizar. Este efecto es más evidente cuando se estudia la orientación relativa de los puntos (Fig. 1.20).

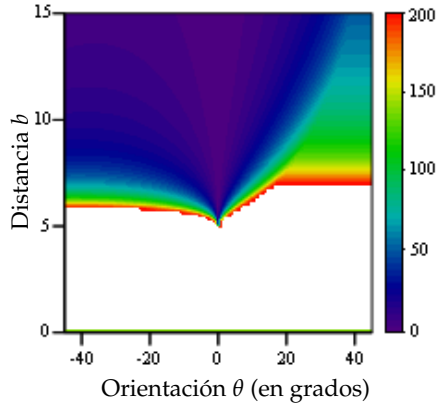


Figura 1.18. Valor umbral Δ_D para variaciones de la distancia y de la orientación relativa de las cámaras.

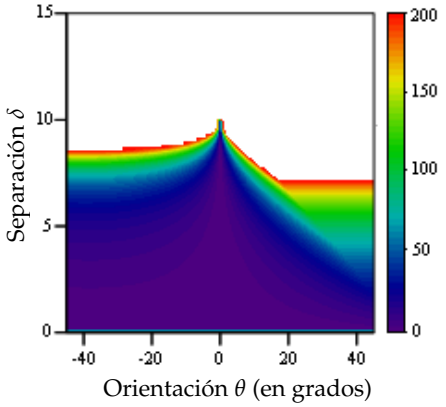


Figura 1.19. Valor umbral Δ_D para variaciones de la separación entre puntos y de la orientación relativa de las cámaras.

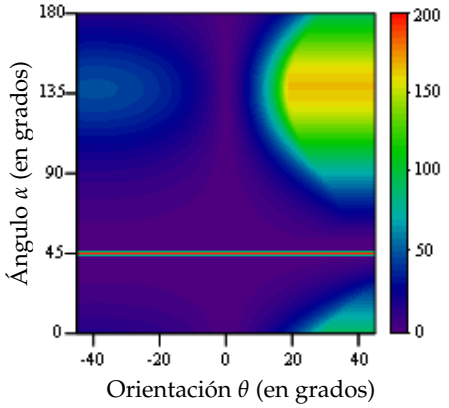


Figura 1.20. Valor umbral Δ_D para variaciones de las orientaciones relativas de los puntos y de las cámaras.

La posición de los puntos también influye de forma significativa en el valor umbral, para variaciones de la orientación de las cámaras. Esto se debe a que determinadas posiciones pueden provocar que, tras su rotación, el par estereovisor quede orientado directamente hacia los puntos de interés; del mismo modo, esta variación podría provocar que, para ciertas posiciones, los puntos quedasen fuera del encuadre de las cámaras (Figs. 1.21 y 1.22).

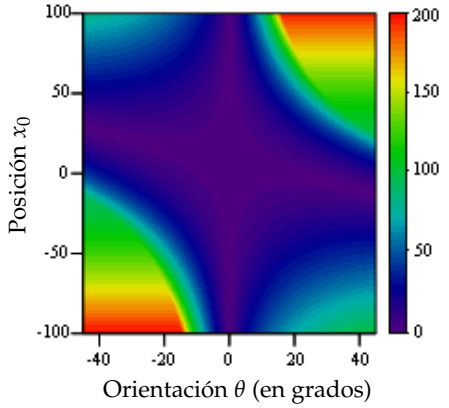


Figura 1.21. Valor umbral Δ_D para variaciones de la posición horizontal de los puntos y de la orientación relativa de las cámaras.

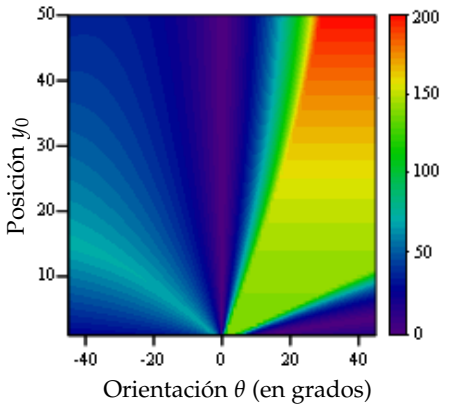


Figura 1.22. Valor umbral Δ_D para variaciones de la posición vertical de los puntos y de la orientación relativa de las cámaras.

Por lo general observamos que la distancia y_0 es proporcional al valor umbral Δ_D, ya que también lo es a la diferencia percibida entre las posiciones de los puntos por parte de las cámaras, antes y después de su rotación. Sin embargo, esta mejora está sujeta al valor de la componente x_0, cuyo signo debe coincidir con el de la rotación θ para que los puntos sean visibles en todo momento.

Capítulo

2

Fusión de estereovisión y telemetría

Los sistemas de estereovisión múltiples y variables, estudiados en el capítulo anterior, son la base de la mayoría de los métodos para calcular mapas de distancia a partir de cámaras convencionales. Sin embargo, estos sistemas se ven limitados por el error interno de las cámaras, que aumenta proporcionalmente a la distancia a los objetos. La fusión sensorial con un dispositivo de mayor precisión resulta pues beneficiosa.

Un *telémetro* es cualquier dispositivo capaz de medir distancias de forma remota. Aunque existen distintos métodos de medida que se pueden aplicar a esta tecnología (la estereovisión es, de hecho, un caso particular de telemetría), nuestra investigación se centra únicamente en los *telémetros láser*. Estos se basan generalmente en el principio del *tiempo de vuelo* para calcular las distancias a los objetos mediante la emisión de rayos de luz modulada (Iddan and Yahav, 2001).

Figura 2.1. Telémetro láser.

Esta tecnología ofrece una precisión mucho mayor que la de la estereovisión, debido a que la dispersión de los rayos emitidos es despreciable, dentro del rango de distancias indicado por el fabricante. Sin embargo, la resolución radial de los telémetros puede influir en la precisión de sus medidas para distancias grandes.

2.1. Resolución de ambigüedades mediante telemetría

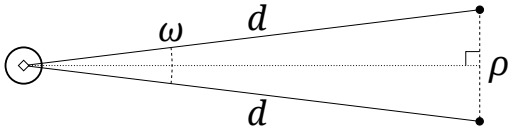


Figura 2.2. Resolución espacial y radial de un telémetro láser.

Si llamamos ω a la diferencia angular mínima entre dos rayos emitidos por un telémetro, la resolución espacial ρ del sensor a una distancia d (Fig. 2.2) puede calcularse mediante trigonometría, para valores pequeños de ω, como

$$\sin\frac{\omega}{2} = \frac{\rho/2}{d} \;\Rightarrow\; \rho = d \cdot 2\sin\frac{\omega}{2} \approx d \cdot \omega$$

Queda entonces demostrado que la resolución espacial de un telémetro es proporcional a su resolución radial y a la distancia a los objetos.

Un telémetro láser convencional devuelve un dato de profundidad z_α para cada ángulo de visión α, tal que todo punto visible podría calcularse como $(z_\alpha \cos\alpha,\, z_\alpha \sin\alpha)$. Sin embargo, debido a la limitación de la precisión radial, muchos posibles puntos de interés podrían quedar ocultos al sensor. Por ello, es común procesar la nube de puntos observable como un conjunto de pares de *radios* y *centroides*, que describen las localizaciones y tamaños aproximados de los objetos visibles (Fig. 2.3).

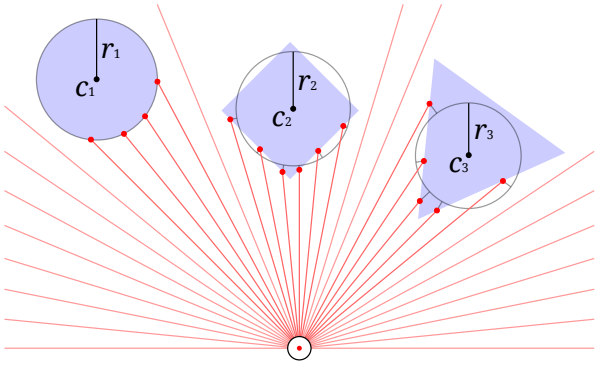


Figura 2.3. Ejemplo del cálculo de los radios y los centroides.

En este capítulo se presenta un modelo de fusión sensorial entre estereovisión y telemetría láser, publicado originalmente en Hernández-Aceituno et al. (2011), que toma como base toda la nomenclatura descrita en el capítulo 1.

El caso práctico para el que se desarrolló esta investigación, un sistema de visión para un prototipo de vehículo, debe interactuar con peatones y otros objetos de altura apreciable en un área generalmente llana. Esto permite simplificar la fusión sensorial asumiendo que las medidas de profundidad obtenidas para un único plano de barrido horizontal pueden extrapolarse a todo el volumen visible por las cámaras.

Los datos obtenidos a partir de un telémetro láser, una vez procesados, vendrán dados como un conjunto T de pares de la forma $\langle c, r \rangle$, donde $c = (c_x, c_y)$ es un centroide y r es su radio correspondiente (Fig. 2.4).

Como se vio en el capítulo anterior, para cada par de conjuntos de valores h se puede calcular varios grupos de puntos (Eq. (1.1) en la página 2), de los cuáles sólo uno será correcto. Los datos devueltos por el telémetro pueden utilizarse para determinar qué grupo es más aceptable, en función de la disparidad entre cada punto calculado $\langle h_1, h_2 \rangle$ y la circunferencia definida por cada par $\langle c, r \rangle$:

$$d(h_1, h_2, \langle c, r \rangle) = \left| \left(\frac{h_1 \cdot b}{h_1 - h_2} - c_x \right)^2 + \left(\frac{f \cdot b}{h_1 - h_2} - c_y \right)^2 - r^2 \right| \tag{2.1}$$

De nuevo, llamando H_1 y H_2 a los conjuntos de valores h devueltos por las cámaras C_1 y C_2 respectivamente, y H_2^P a una permutación específica de H_2 (de acuerdo a las restricciones de cardinal explicadas en la sección 1.2.3), podemos definir una medida de la disparidad acumulada para varios puntos visibles y pares centroide/radio.

La relación entre un punto y un par centroide/radio no es unívoca, ya que la circunferencia definida por un único par podría corresponder a varios puntos simultáneamente. Por lo tanto, para cada grupo de valores h de cada cámara deben considerarse todas las combinaciones con los pares devueltos por la tele-

metría, sin descartar las posibles repeticiones de permutaciones (Fig. 2.4).

$$D\Big(H_1, H_2^p, T\Big) = \sum_{i=1}^{|H_1|} \sum_{j=1}^{|T|} d\Big(h_{1i},\, H_{2i}^p,\, (c,r)_j\Big) \tag{2.2}$$

El valor umbral se redefine como la diferencia entre el valor de disparidad del grupo de puntos correcto y el de cualquier otra combinación de valores de proyección y pares de telemetría.

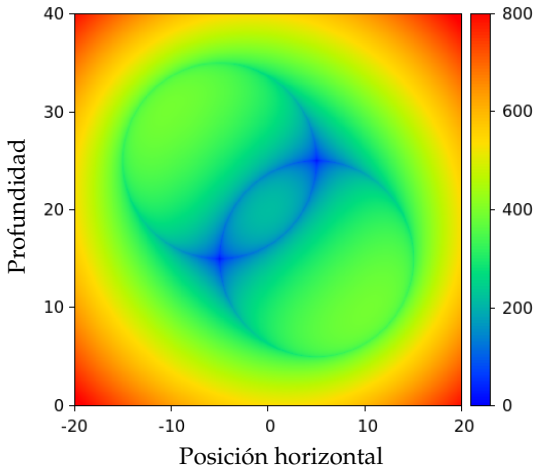


Figura 2.4. Valor de disparidad D para estimaciones de la posición de un punto en un plano horizontal, para los pares de telemetría $((-5,25),10)$ y $((5,15),10)$.

2.2. Caso práctico

El modelo de fusión desarrollado a partir de estas definiciones fue analizado en un entorno real, situando dos peatones en un entorno parcialmente controlado (el aparcamiento de la Escuela Técnica Superior de Ingeniería Informática de la Universidad de La Laguna) y usando el sistema sensorial del dispositivo VERDINO (apéndice A) como fuente de datos.

El par estereovisor reconoce los obstáculos visibles y se sirve de la salida del telémetro láser para resolver el problema de la correspondencia. En el ejem-

plo mostrado (Fig. 2.5), la estereovisión produce los pares de puntos $\{A_1, A_2\}$ y $\{B_1, B_2\}$, de los cuales sólo el primero es compatible con la información de telemetría (pares centroide/radio C_5 y C_6).

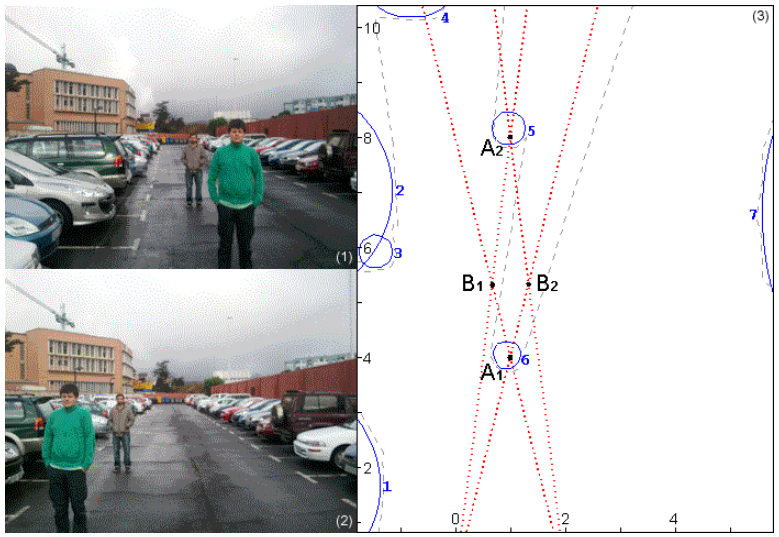


Figura 2.5. Datos de entrada (1,2) y de salida (3) del algoritmo de fusión sensorial. Las unidades de (3) se dan en metros a partir de la cámara 1. Los datos de estereovisión se muestran como líneas rojas, los de telemetría como circunferencias azules.

Para poder comparar la eficiencia de nuestro sistema de fusión sensorial con la de un montaje trinocular de cámaras, ambos métodos fueron estudiados en un entorno con dos peatones en distintas posiciones relativas, de manera similar a nuestros experimentos teóricos de la sección 1.3 en la página 14. La separación entre los peatones se mantuvo siempre menor que la distancia entre cámaras, para forzar la aparición de ambigüedades (Figs. 2.6 a 2.8).

Nuestros experimentos demuestran que la capacidad de resolución de ambigüedades, para una fusión sensorial entre estereovisión y telemetría láser, es muy superior a la de los sistemas de estereovisión variable y múltiple, especialmente a grandes distancias dentro del rango de detección del telémetro.

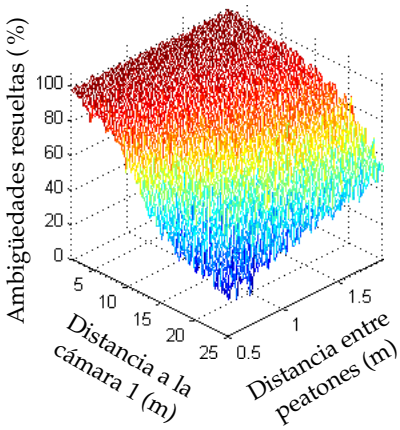


Figura 2.6. Porcentaje medio de ambigüedades resueltas correctamente por un sistema de estereovisión trinocular, para variaciones de la separación entre los peatones y de su distancia conjunta a las cámaras.

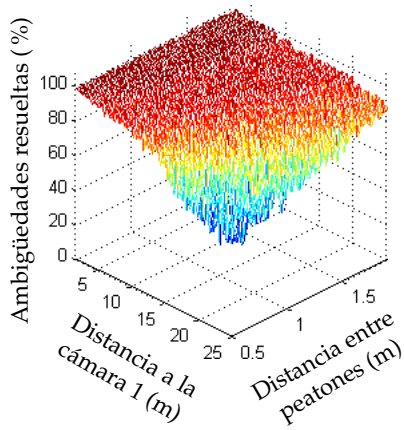


Figura 2.7. Porcentaje medio de ambigüedades resueltas correctamente por un sistema de fusión sensorial de estereovisión y telemetría, para variaciones de la separación entre los peatones y de su distancia conjunta a las cámaras.

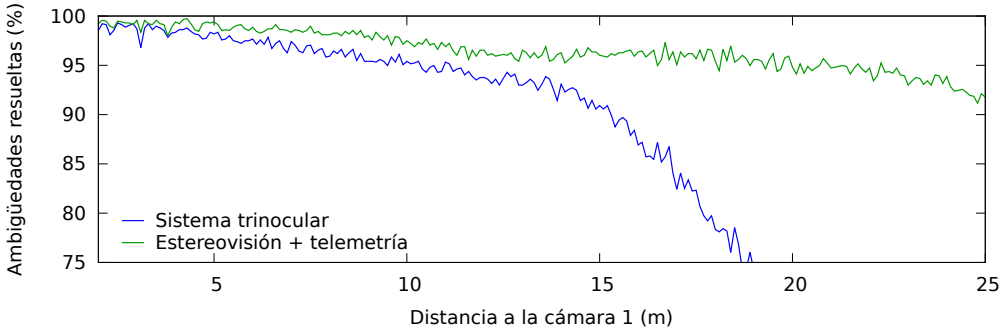


Figura 2.8. Porcentaje medio de ambigüedades resueltas correctamente por un sistema trinocular y por un sistema de fusión sensorial de estereovisión y telemetría, para variaciones de la distancia entre los peatones y las cámaras. La separación entre los peatones se mantiene fija a un metro.

Capítulo

3

Enfoque bayesiano de la detección por Viola-Jones

Para todo vehículo autónomo que pretenda garantizar un funcionamiento seguro en un entorno poblado, la detección de peatones es una tarea esencial. Sin embargo, su automatización para sensores de tipo visual es un proceso muy complejo, ya que exige la búsqueda, comprensión e interpretación de regiones concretas de su campo de visión. Uno de los algoritmos clásicos para resolver este problema es el desarrollado por Viola and Jones (2004).

3.1. Algoritmo de Viola-Jones

Este algoritmo de detección se basa en reconocer *características visuales* de los objetos que se desea localizar. Estas vienen definidas, para cada imagen a analizar, como sumas y restas de valores asociados a áreas rectangulares no rotadas, dados como el sumatorio del nivel de gris de sus píxeles (Fig. 3.1).

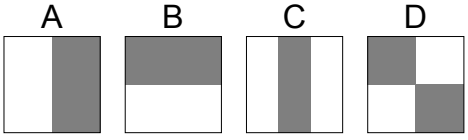


Figura 3.1. Características usadas en la detección de Viola-Jones. El valor de cada característica viene dada como el sumatorio de los píxeles del área blanca menos el sumatorio de los píxeles del área gris.

Se denomina *imagen integral* I de un mapa de bits M en escala de grises a aquella cuyos píxeles tienen como valor

$$I_{xy} = \sum_{i=0}^{x-1} \sum_{j=0}^{y-1} M_{ij}$$

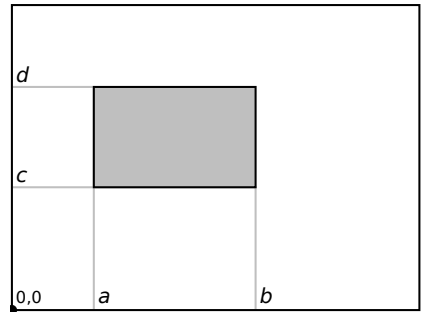


Figura 3.2. Ejemplo de área rectangular en una imagen integral.

Mediante el uso de este tipo de imágenes, el cálculo de las características de Viola-Jones se puede hacer en un tiempo constante. Por ejemplo, el sumatorio de los píxeles del rectángulo marcado en la figura 3.2 sería:

$$\begin{aligned}
\sum_{x=a}^{b-1} \sum_{y=c}^{d-1} M_{xy} &= \sum_{x=a}^{b-1} \left\{ \sum_{y=0}^{d-1} M_{xy} - \sum_{y=0}^{c-1} M_{xy} \right\} = \sum_{x=a}^{b-1} \sum_{y=0}^{d-1} M_{xy} - \sum_{x=a}^{b-1} \sum_{y=0}^{c-1} M_{xy} \\
&= \left(\sum_{x=0}^{b-1} \sum_{y=0}^{d-1} M_{xy} - \sum_{x=0}^{a-1} \sum_{y=0}^{d-1} M_{xy} \right) - \left(\sum_{x=0}^{b-1} \sum_{y=0}^{c-1} M_{xy} - \sum_{x=0}^{a-1} \sum_{y=0}^{c-1} M_{xy} \right) \\
&= I_{bd} - I_{ad} - I_{bc} + I_{ac}
\end{aligned}$$

Una vez definidas las características de los objetos que se desean detectar en las imágenes, se entrena un conjunto de *clasificadores* para reconocerlas, utilizando el meta-algoritmo de aprendizaje *AdaBoost* (Freund and Schapire, 1995). Para que el proceso de detección pueda ejecutarse en tiempo real se sigue una arquitectura de cascada: cada muestra se descarta tan pronto como uno de los clasificadores la rechaza:

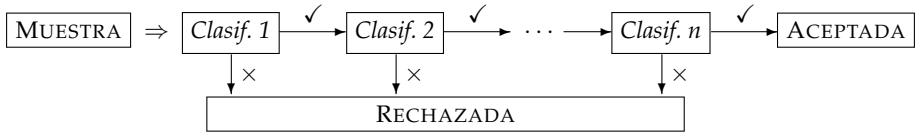


Figura 3.3. Estructura de cascada para n clasificadores.

En este capítulo se presenta un enfoque bayesiano de la herramienta de detección de Viola-Jones, mediante la interpretación probabilista de su resultado y el desarrollo de un método de convolución aproximada del mismo. Este trabajo fue publicado en Hernández-Aceituno et al. (2015b).

3.2. Modelo bayesiano

Sean x y z dos variables aleatorias tales que

- x expresa la existencia o ausencia de objetos de interés (en nuestro caso peatones) para cada píxel de una imagen.
- z muestra el valor correspondiente devuelto por la herramienta de detección de Viola-Jones aplicada a una imagen.

Es posible utilizar z para evaluar el grado de confianza de x (es decir, $p(x \mid z)$), mediante la aplicación del teorema de Bayes:

$$\underbrace{p(x \mid z)}_{\textit{a posteriori}} \propto \underbrace{p(z \mid x)}_{\textit{verosimilitud}} \cdot \underbrace{p(x)}_{\textit{a priori}}$$

Una de las utilidades de un modelo bayesiano es eliminar detecciones superfluas basándose en observaciones previas. Esta decisión puede resultar perjudicial a la hora de detectar peatones, ya que los falsos positivos son preferibles a los falsos negativos: ignorar una detección correcta involucra un peligro real, mientras que aceptar una detección errónea sólo produce una ruta menos eficiente.

Se propone por tanto una aplicación inversa del teorema de Bayes, que filtre la ausencia de objetos en lugar de sus detecciones. Para ello se recurre a los valores inversos de las variables presentadas: $p(\bar{x} \mid \bar{z}) \propto p(\bar{z} \mid \bar{x}) \cdot p(\bar{x})$. El cálculo de la verosimilitud $p(\bar{z} \mid \bar{x})$ y de la probabilidad a priori $p(\bar{x})$ se explica en las siguientes subsecciones.

3.2.1. Verosimilitud

En su comportamiento por defecto para una imagen dada, las detecciones generadas por el método de Viola-Jones son un conjunto de áreas rectangulares, dentro de las cuales se ha detectado un objeto de interés.

Es posible producir matrices binarias a partir de estas áreas, de modo tales que cada celda valga 1 si pertenece a una de ellas y 0 en caso contrario (Fig. 3.4). Llamaremos B_i a la matriz binaria correspondiente a la $i-$ésima área rectangular.

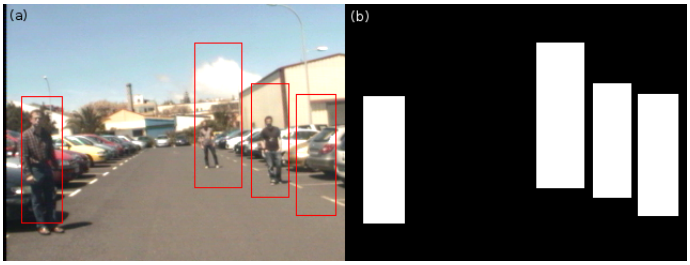


Figura 3.4. Ejemplo de ejecución básica de Viola-Jones (a) y unión de las matrices binarias correspondientes (b).

Para cada imagen, algunas de las áreas devueltas por el algoritmo pueden ser superfluas (falsos positivos), y otras pueden solaparse entre sí. Cuanto mayor es el número de áreas que se solapan sobre un determinado grupo de píxeles, mayor es la probabilidad de que estos contengan un objeto de interés.

El método de Viola-Jones permite el uso de una restricción de solapamiento mínimo: un área rectangular sólo se considera válida si resulta de la intersección de al menos tantas detecciones como indique un parámetro η especificado por el usuario (Fig. 3.5).

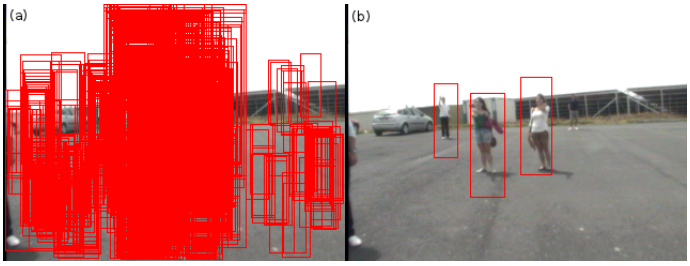


Figura 3.5. Ejemplo de ejecución de Viola-Jones para $\eta = 0$ (a) y para $\eta = 100$ (b).

En lugar de utilizar este parámetro, se propone generar una matriz de detecciones, tal que el valor de cada una de sus celdas sea igual al número de áreas rectangulares que contienen su píxel correspondiente (Fig. 3.2.1). Esta matriz equivale a la suma de las matrices binarias de todas las detecciones obtenidas.

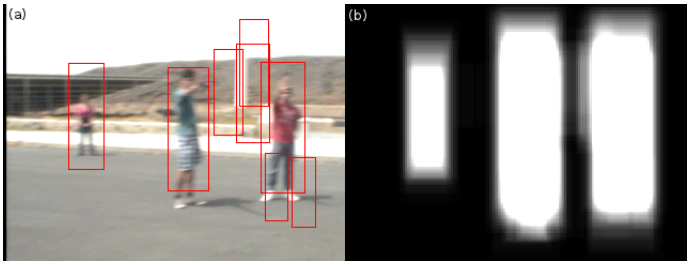


Figura 3.6. Ejemplo de Viola-Jones para $\eta = 3$ (a) y matriz de detecciones (b); las zonas más claras representan valores mayores.

La matriz de verosimilitud, correspondiente a la probabilidad de ausencia de objetos de interés en la imagen, es proporcional al valor opuesto de la matriz de detecciones; llamando N al número de detecciones obtenidas, esta puede calcularse como:

$$p\left(\bar{z} \mid \bar{x}\right) \propto N - \sum_{i=1}^{N} B_i$$

3.2.2. Probabilidad a priori

El uso del teorema de Bayes implica un paso de convolución: la función de probabilidad *a priori* debe calcularse a partir de la *a posteriori* previa.

Idealmente, la localización de cada objeto queda determinada por una distribución de probabilidad, generalmente normal. Sin embargo, las probabilidades resultantes de nuestros experimentos no se ajustan a este tipo de distribuciones.

Como sumatorios normalizados de funciones rectangulares binarias solapadas, cumplen ciertas características especiales: su distribución de probabilidad es simétrica, su valor mínimo es 0 y su valor máximo no aparece en un único punto, sino que se extiende sobre un área rectangular definida. Se diseñó una función de distribución que aproximase este comportamiento.

Matriz aproximada de detecciones

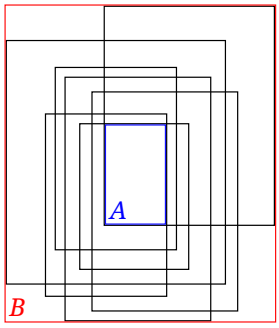


Figura 3.7. Áreas A y B y conjunto D (en negro).

Sea D el conjunto de detecciones, devueltas por el método Viola-Jones, que corresponden a un objeto de interés concreto. Dicho objeto se representará mediante la tupla $\langle |D|, A, B \rangle$, donde

- $|D|$ es el número de elementos de D,
- A es el área rectangular mínima que delimita la intersección de todos los elementos de D, y
- B es el área rectangular mínima que delimita la unión de todos los elementos de D (Fig. 3.7).

Usando estos datos, deseamos encontrar una función bidimensional que permita simular de manera aproximada el sumatorio de los elementos de D:

$$F\left(r, \langle |D|, A, B \rangle\right) = \begin{cases} 0 & \text{si } r \notin B \\ |D| & \text{si } r \in A \end{cases} \quad \text{para } r \in \mathbb{R}^2 \tag{3.1}$$

Si se limita el problema a una sola dimensión, los rectángulos A y B pueden considerarse como dos segmentos, a los que denotaremos por $\overline{a_1 a_2}$ y $\overline{b_1 b_2}$, respectivamente, considerando $b_1 < a_1 < a_2 < b_2$ (Fig. 3.8).

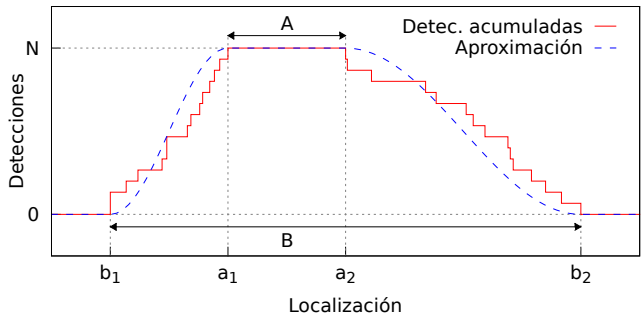


Figura 3.8. Ejemplo teórico del corte transversal de las áreas que definen un objeto de interés, junto a la aproximación de su sumatorio de detecciones.

Sea entonces la función f:

$$f\left(c, b_1, a_1, a_2, b_2\right) = \begin{cases} 0 & \text{si } c < b_1 \\ \frac{1}{2} - \frac{1}{2}\cos\left(\pi \cdot \frac{c - b_1}{a_1 - b_1}\right) & \text{si } b_1 \leq c < a_1 \\ 1 & \text{si } a_1 \leq c < a_2 \\ \frac{1}{2} - \frac{1}{2}\cos\left(\pi \cdot \frac{b_2 - c}{b_2 - a_2}\right) & \text{si } a_2 \leq c < b_2 \\ 0 & \text{si } b_2 \leq c \end{cases}$$

La forma de esta función se ajusta a la que se desea obtener (Fig. 3.8), pero se encuentra escalada en altura. Para dos dimensiones, la función que simula el sumatorio de las detecciones correspondientes a un objeto, y que cumple con la restricción (3.1), es:

$$F\left(\begin{bmatrix} r_{\mathrm{x}} \\ r_{\mathrm{y}} \end{bmatrix}, \langle |D|, A, B \rangle\right) = |D| \cdot f\left(r_{\mathrm{x}}, b_{\mathrm{izq}}, a_{\mathrm{izq}}, a_{\mathrm{der}}, b_{\mathrm{der}}\right) \cdot f\left(r_{\mathrm{y}}, b_{\mathrm{inf}}, a_{\mathrm{inf}}, a_{\mathrm{sup}}, b_{\mathrm{sup}}\right)$$

para $\begin{bmatrix} r_{\mathrm{x}}, r_{\mathrm{y}} \end{bmatrix}^T \in \mathbb{R}^2$, y donde a_{izq}, a_{der}, a_{inf} y a_{sup} son respectivamente los límites izquierdo, derecho, inferior y superior del área A, y b_{izq}, b_{der}, b_{inf} y b_{sup} son los límites correspondientes del área B.

Es posible generar una matriz de probabilidad a partir de las tuplas que definen los objetos de interés observados. Para M objetos:

$$P\left(x\right) \propto \sum_{i=1}^{M} F(r, \langle |D|, A, B \rangle_i), \text{ para } r \in \mathbb{R}^2 \tag{3.2}$$

Extracción de los objetos de la matriz de probabilidad a posteriori

Sea v_ω el valor de una celda ω perteneciente a una matriz de probabilidad *a posteriori*. Toda celda adyacente a ω se representará como $\mathcal{A}\omega$; si existe un camino de adyacencia de longitud q entre dos celdas λ y ω, se denotará por:

$$\lambda = \mathcal{A}^q\omega = \begin{cases} \omega & \text{si } q = 0 \\ \mathcal{A}^{q-1}(\mathcal{A}\omega) & \text{si } q > 0 \end{cases}$$

y diremos que λ es *alcanzable* desde ω. Por definición, ω es alcanzable desde ω.

Partiendo de estos conceptos, para extraer los datos de los objetos de interés contenidos en la matriz de probabilidad, se siguen los siguientes pasos:

1. Sea α la celda con mayor valor de la matriz: $\forall\omega, v_\alpha \geq v_\omega$. El valor v_α es proporcional al elemento $|D|$ del objeto con mayor número de detecciones.

2. Sea A el área rectangular mínima que contiene todas las celdas alcanzables desde α y con igual valor: $((\exists\, q \in \mathbb{N} \mid \omega = \mathcal{A}^q\alpha) \wedge (v_\omega = v_\alpha)) \Rightarrow \omega \in A$

3. Sea B el área rectangular mínima que contiene todas las celdas alcanzables desde α cuyo valor no sea 0 ni un mínimo local, de tal forma que se pueda asumir que toda celda $\lambda \notin B$ no corresponde al objeto actual: $((\exists\, q \in \mathbb{N} \mid \omega = \mathcal{A}^q\alpha) \wedge (v_\omega \neq 0) \wedge \neg\,(\forall\, \lambda = \mathcal{A}\omega \mid v_\omega < v_\lambda)) \Rightarrow \omega \in B$

4. Se obtiene la tupla $\langle v_\alpha, A, B\rangle$ y el objeto se elimina de la matriz de probabilidad. Estos pasos se repiten hasta que la matriz esté vacía.

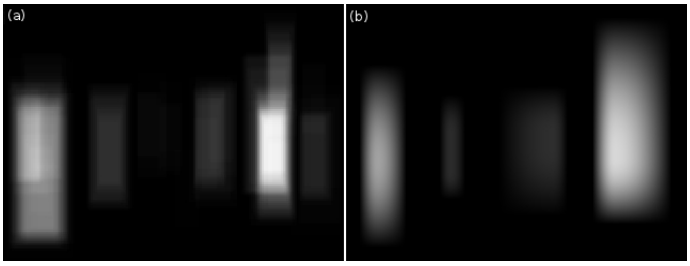


Figura 3.9. Generación aproximada de matriz de probabilidad: (b) se creó a partir de los objetos extraídos de (a).

Generación de matriz de probabilidad a priori

Cuando todos los objetos han sido extraídos, debe analizarse su relación respecto a las detecciones observadas hasta el momento durante la ejecución, a fin de predecir su movimiento. Para ello, las tuplas obtenidas en cada iteración se agrupan en *trayectorias*.

Cuando los objetos estudiados están claramente diferenciados, sus movimientos pueden analizarse y predecirse de forma individual; sin embargo, en nuestro caso las correspondencias entre los objetos de diferentes fotogramas son desconocidas. Para identificar la trayectoria más probable a la que añadir un objeto extraído, se utiliza una estimación de mínimos cuadrados.

En cada iteración, se eliminan aquellas trayectorias a las que no se ha agregado un nuevo elemento, ya que se asume que su objeto correspondiente ha abandonado el campo visual del vehículo. Se considera además que todo objeto aislado, para el que no se ha encontrado una trayectoria adecuada, acaba de aparecer por primera vez en la escena, y por tanto es el primer elemento de una nueva trayectoria (Fig. 3.10a).

Se procede entonces a estimar la siguiente posición de cada una de las trayectorias, aplicando una regresión lineal sobre los elementos de las tuplas almacenadas en cada una de ellas (Fig. 3.10b). El resultado de la predicción se usa para generar la matriz de probabilidad *a priori*, mediante la ecuación (3.2) aplicada a las posiciones futuras estimadas.

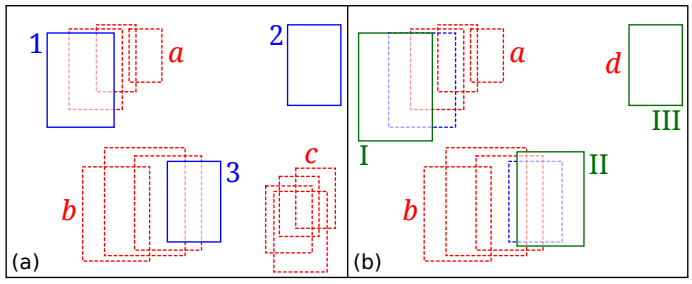


Figura 3.10. Inclusión de objetos (1, 2, 3) en trayectorias (a, b, c, d) y predicción de posiciones futuras (I, II, III).

3.3. Resultados

El método presentado fue aplicado sobre doce secuencias de imágenes, descritas en la tabla 3.1 y de las que se muestran ejemplos en la figura 3.11.

Tabla 3.1. Características de las secuencias de imágenes.

Secuencia de imágenes		Entorno	Trayectoria del robot	Movimiento de los peatones
(a)	*ETSII*	Urbano	Lenta, recta	Estático o errático
(b)	*ITER1*	Rural	Rápida, recta	Estático
(c)	*ITER2*	Rural	Rápida, giros	Estático
(d)	*BAHNHOF*	Urbano	Lenta, recta	Paralelo al robot
(e)	*JELMOLI*	Urbano	Rápida, giros	Varias direcciones
(f)	*SUNNY DAY*	Urbano	Rápida, recta	Paralelo al robot
(g)	*CAVIAR1*	Interior	Estática	Errático
(h)	*CAVIAR2*	Interior	Estática	Estático o errático
(i)	*CAVIAR3*	Interior	Estática	Estático o errático
(j)	*CAVIAR4*	Interior	Estática	Errático, en grupos
(k)	*DAIMLER*	Urbano	Rápida, giros	Varias direcciones
(l)	*CALTECH*	Urbano	Rápida, recta	Paralelo al robot

La secuencia *ETSII* fue tomada en los aparcamientos de la Escuela Técnica Superior de Ingeniería Informática de la Universidad de La Laguna, y las secuencias *ITER1* e *ITER2*, en el perímetro y el aparcamiento, respectivamente, de las instalaciones del Instituto Tecnológico y de Energías Renovables. Las tres fueron grabadas usando el sistema sensorial del dispositivo VERDINO (apéndice A).

BAHNHOF, *JELMOLI* y *SUNNY DAY* fueron descargadas de la página web del proyecto *Robust Multi-Person Tracking from Mobile Platforms* (Ess et al., 2007, 2008, 2009a,b) del Instituto Federal Suizo de Tecnología, y fueron grabadas usando un par de cámaras AVT Marlin F033C.

Figura 3.11. Ejemplo de fotogramas de las secuencias descritas en la tabla 3.1.

CAVIAR1 a *CAVIAR4* pertenecen al proyecto *Context Aware Vision using Image-based Active Recognition* (Fisher et al.) y fueron tomadas en un centro comercial de Portugal, usando una cámara fija. Las secuencias utilizadas corresponden a las vistas de pasillo de los vídeos *WalkByShop1* (*CAVIAR1*), *OneShopOneWait1* (*CAVIAR2*), *OneShopOneWait2* (*CAVIAR3*) y *ThreePastShop1* (*CAVIAR4*).

DAIMLER fue extraída de la base de datos de referencia para detección de peatones Daimler (Enzweiler and Gavrila, 2009), y *CALTECH* corresponde a la secuencia *V002* del conjunto de pruebas *seq06* de la base de datos Caltech (Dollár et al., 2009, 2012). Ambas secuencias fueron grabadas desde un vehículo en movimiento en un entorno de tráfico urbano.

Se ejecutaron diez pruebas sobre cada secuencia de imágenes, comparando los resultados de la ejecución del método presentado y del algoritmo de Viola-Jones sin modificar (Fig. 3.12). Como se explicó en la sección 3.2, el objetivo de nuestro trabajo es reducir la cantidad de falsos negativos devueltos por la herramienta original. Por ello, los métodos clásicos de análisis como las curvas ROC (*Receiver Operating Characteristic*) y DET (*Detection Error Tradeoff*), que dependen de la cantidad de falsos positivos, no muestran apropiadamente la mejora que introduce nuestro método. En su lugar, se muestra la relación media entre la tasa de falsos negativos de cada herramienta y los positivos reales, seleccionados manualmente en los fotogramas de partida.

Se observó que el enfoque bayesiano presentado siempre devuelve tasas de detección menos conservadoras que Viola-Jones, por lo que la tasa de falsos positivos se reduce para todas las secuencias de imágenes. La mejora es particularmente notable en las secuencias *ETSII*, *ITER1* y *2*, *CAVIAR1* a *4* y *DAIMLER*; esto se debe a que las capturas tienen buena visibilidad, lo que produce una mayor precisión en las detecciones del algoritmo original y, en consecuencia, un mayor margen de mejoría para nuestro método.

El resto de las secuencias tienen tasas de oclusión mayores e incluyen peatones en posturas y posiciones que dificultan su detección, lo que a su vez disminuye el potencial de un procesamiento bayesiano. Este efecto es especialmente claro en la secuencia *CALTECH*, que incluye muy pocos peatones visibles.

Por último se estudió el método de *Soft Cascade* (Bourdev and Brandt, 2005), basado en la asignación de pesos a las detecciones y cuyos resultados se dan sin solapamiento, lo que impide su procesamiento probabilista. Las pruebas preliminares demostraron que, debido a esta carencia, su precisión es visiblemente inferior a la de nuestro sistema, y por tanto no se incluyó en los experimentos.

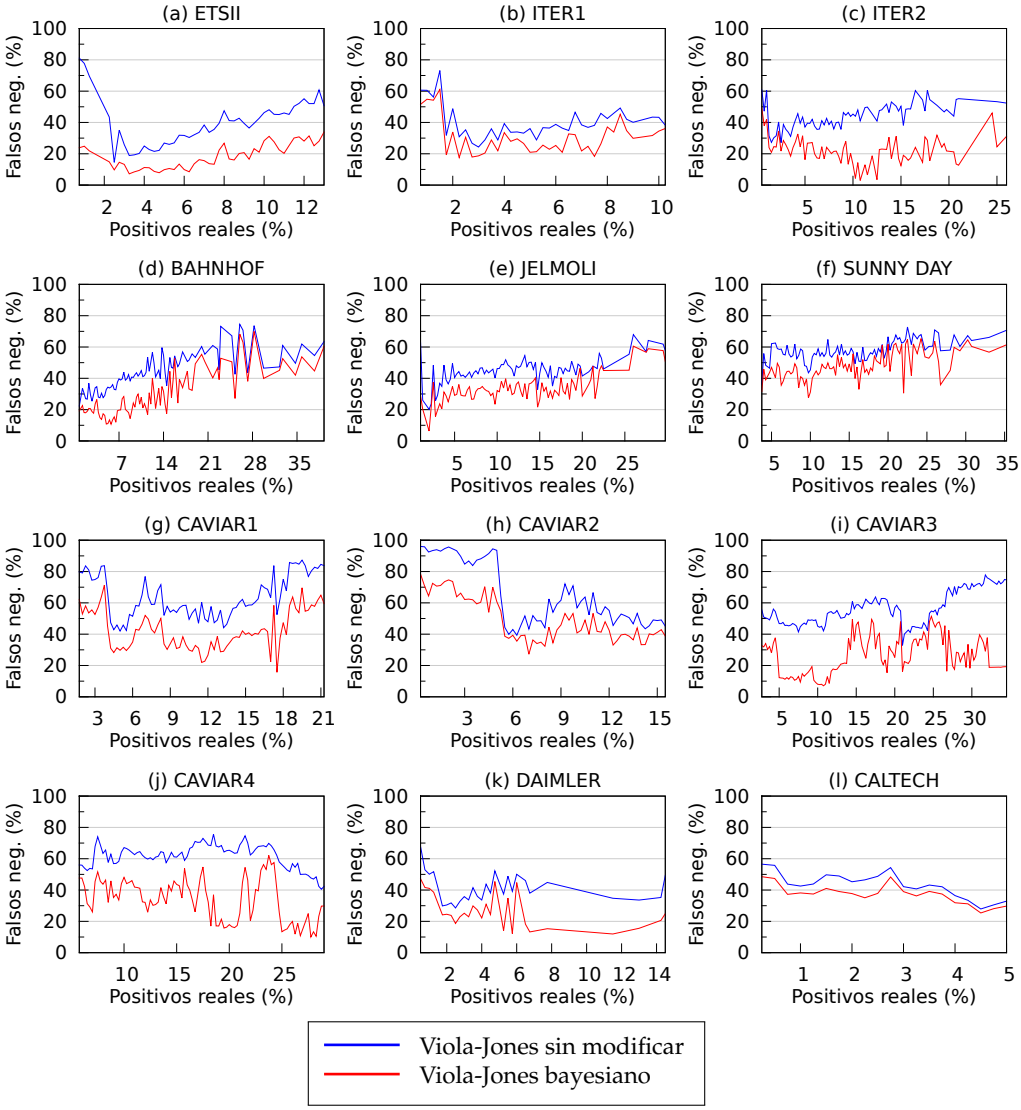


Figura 3.12. Valores medios de la ejecución para cada secuencia de imágenes.

Capítulo

4

Enfoque bayesiano de los histogramas de gradientes orientados

En el capítulo anterior se explicó un sistema de mejora de la detección visual, basado en el procesamiento bayesiano de la salida del algoritmo de Viola-Jones. Nada impide, sin embargo, aplicar el mismo mecanismo a otras herramientas de detección, o emplear modelos alternativos de distribución aproximada de áreas rectangulares acumuladas.

4.1. Histograma de gradientes orientados

Según Dalal and Triggs (2005), la apariencia y forma de un objeto dentro de una imagen puede ser descrita a partir de su distribución de gradientes de intensidad y de las direcciones de sus bordes. La implementación de este tipo de descriptores se basa en tres etapas de subdivisión.

El primer paso del algoritmo de detección mediante histogramas de gradientes orientados (HOG) es el cómputo de los gradientes de cada píxel. Para ello se filtra la imagen aplicando una máscara monodimensional derivativa, $[-1,\ 0,\ 1]$, tanto en horizontal como en vertical, y se calcula la pendiente asociada al resultado combinado, como una arcotangente.

El segundo paso consiste en crear histogramas de los gradientes obtenidos, para pequeños conjuntos rectangulares o circulares de píxeles contiguos, denomi-

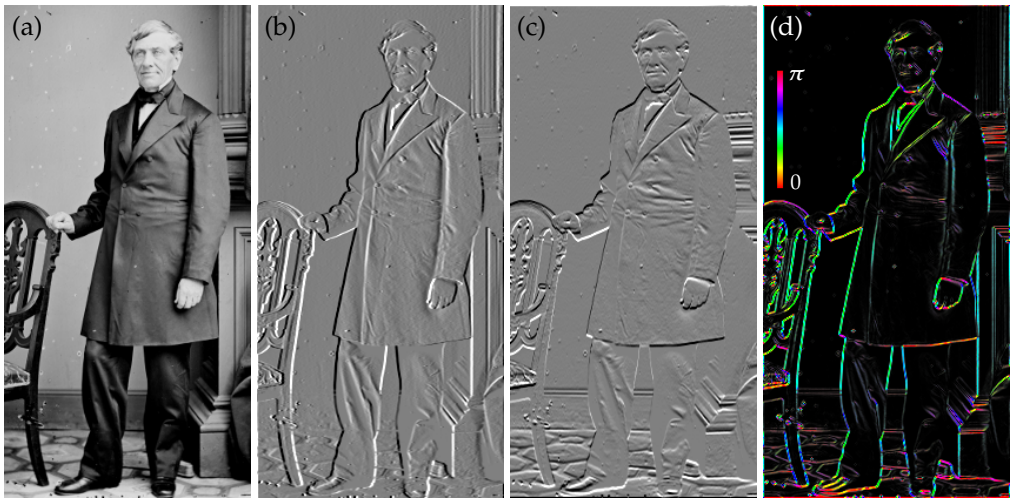


Figura 4.1. Filtros derivativos horizontal (b) y vertical (c) y valores de gradiente sin signo codificados por color (d) de una imagen (a).

nados *celdas*. El cálculo de los histogramas implica reducir el número de valores permitidos a un conjunto limitado de intervalos o *canales*, distribuidos uniformemente entre 0 y 2π ó entre 0 y π, dependiendo de si se considera el signo de los gradientes o no. En el trabajo original, se comprobó que el algoritmo se comporta de forma óptima para nueve canales sin signo.

En el tercer paso, para suprimir posibles errores debidos a cambios de iluminación o contraste, los módulos de los gradientes se normalizan para conjuntos de celdas contiguas, denominados *bloques*. Un descriptor HOG es, en resumen, el vector de los componentes de los histogramas normalizados de las celdas, para todos los bloques disponibles. Estos bloques pueden solaparse, así que una celda puede contribuir más de una vez al cálculo de un descriptor.

De nuevo, un bloque puede ser rectangular (R-HOG) o circular (C-HOG), y queda descrito por tres parámetros: el número de celdas por bloque, el número de píxeles por celda y el número de canales por histograma. En el estudio original, se observó que el algoritmo se comporta de forma más eficiente usando bloques de 3×3 celdas, compuestas por 6×6 píxeles, y 9 canales por histograma.

Una vez calculados los descriptores HOG, se utilizan como entrada para un sistema de reconocimiento basado en aprendizaje supervisado, que genera un clasificador SVM (*Support Vector Machine, Cortes and Vapnik, 1995)*. Cuando concluye el entrenamiento, el clasificador obtenido se utiliza como entrada para la función de detección por HOG, incluida en la librería de C++ de tratamiento de imágenes *OpenCV* (Bradski, 2000). Puesto que tanto los parámetros de entrada como el valor de salida de dicha función coinciden con los de Viola-Jones, estas pueden intercambiarse fácilmente en el algoritmo presentado en el capítulo 3.

4.2. Función de distribución aproximada de áreas acumuladas

En el capítulo anterior se desarrolló una función de distribución que permite modelar una matriz de detecciones, a partir de los datos de cada objeto de interés observado. Esta se basa en que el sumatorio de matrices binarias, definidas por las áreas rectangulares que identifican las detecciones, tiene un valor constante máximo en su área de intersección (denotada por A) e igual a cero fuera de su área de unión (denotada por B). Restringiendo la distribución descrita a una sola dimensión, dichas áreas se reducen a dos segmentos, $\overline{a_1 a_2}$ y $\overline{b_1 b_2}$, tal que $b_1 < a_1 < a_2 < b_2$.

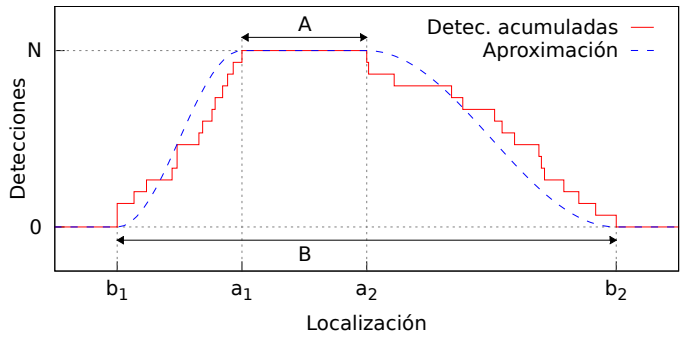


Figura 4.2. Función de distribución que modela una matriz de detecciones, junto a las áreas de intersección y unión que definen un objeto de interés.

$$\text{Sea la función } f(c, b_1, a_1, a_2, b_2) = \begin{cases} 0 & \text{si } c < b_1 \\ g\left(\dfrac{b_1 - c}{b_1 - a_1}\right) & \text{si } b_1 \leq c < a_1 \\ 1 & \text{si } a_1 \leq c < a_2 \\ g\left(\dfrac{c - b_2}{a_2 - b_2}\right) & \text{si } a_2 \leq c < b_2 \\ 0 & \text{si } b_2 \leq c \end{cases}$$

Se desea que $g(x)$ una los segmentos anterior y posterior de forma suave. Para ello, se imponen las restricciones $g(0) = 0$, $g(1) = 1$, $g'(0) = 0$ y $g'(1) = 0$. En el capítulo 3 se utilizó $g(x) = 1/2 - 1/2\cos(\pi x)$, pero es razonable pensar que otra función con características similares pueda cumplir el mismo papel.

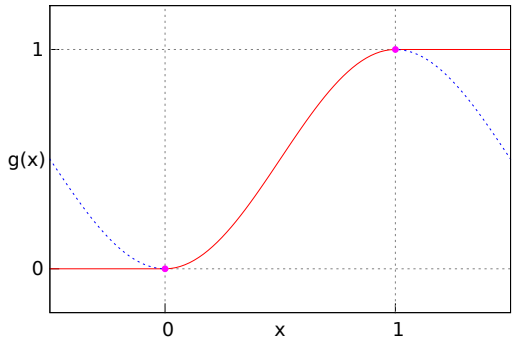


Figura 4.3. Forma aproximada de la función de unión.

Busquemos pues el polinomio $g(x)$ que cumpla con las cuatro restricciones impuestas. Puesto que podemos introducir cuatro incógnitas en nuestro problema, el grado máximo que podemos calcular es tres: $g(x) = ax^3 + bx^2 + cx + d$. Sabiendo que $g'(x) = 3ax^2 + 2bx + c$, se obtiene el sistema de ecuaciones:

$$\left.\begin{aligned} g(0) &= d = 0 \\ g(1) &= a + b + c + d = 1 \\ g'(0) &= c = 0 \\ g'(1) &= 3a + 2b + c = 0 \end{aligned}\right\} \Rightarrow \left.\begin{aligned} a &= -2 \\ b &= 3 \end{aligned}\right\} \Rightarrow g(x) = 3x^2 - 2x^3 \qquad (4.1)$$

Por tanto, otra posible función que completa $f(x)$ es $g(x) = 3x^2 - 2x^3$ y, como se vio en el capítulo anterior, el sumatorio de las $|D|$ detecciones correspondientes a un único objeto de interés se modela por:

$$F\left(\begin{bmatrix} r_x \\ r_y \end{bmatrix}, \langle |D|, A, B \rangle\right) = |D| \cdot f\left(r_x, b_{izq}, a_{izq}, a_{der}, b_{der}\right) \cdot f\left(r_y, b_{inf}, a_{inf}, a_{sup}, b_{sup}\right)$$

para $\left[r_x, r_y\right]^T \in \mathbb{R}^2$, y donde a_{izq}, a_{der}, a_{inf} y a_{sup} son respectivamente los límites izquierdo, derecho, inferior y superior del área de intersección A, y b_{izq}, b_{der}, b_{inf} y b_{sup} son los correspondientes al área de unión B.

Resulta curioso comprobar que, a pesar de ser curvas fundamentalmente diferentes, la ecuación $h(x) = 1/2 - 1/2\cos(\pi x)$ utilizada en el capítulo 3 es casi idéntica a $g(x) = 3x^2 - 2x^3$ en el intervalo $x \in [0, 1]$.

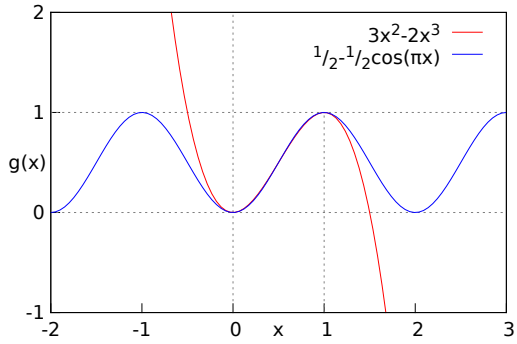


Figura 4.4. Comparación de las funciones de unión suave polinómica y trigonométrica.

Considerando una función $d(x) = \left(3^2 - 2x^3\right) - \left(1/2 - 1/2\cos(\pi x)\right)$, sabemos que la diferencia máxima entre $g(x)$ y $h(x)$ corresponderá a los valores de $d(x)$ tales que $d'(x) = 6x - 6x^2 - \pi/2\sin(\pi x) = 0$.

Esta ecuación no se puede resolver de forma analítica, pero por tanteo se puede obtener que $d'(0{,}278577) = 0$ y $d'(0{,}721423) = 0$, así que el valor de diferencia máxima para $0 \leq x \leq 1$ es $|d(0{,}278577)| = |d(0{,}721423)| \approx 10^{-2}$.

4.3. Resultados

El método de procesamiento bayesiano de la detección por HOG, que incluye la función de modelado de matrices de probabilidad basada en polinomios, fue aplicado sobre las doce secuencias de imágenes descritas en el capítulo 3: tres grabadas por el dispositivo VERDINO (*ETSII* e *ITER*, apéndice A), tres proporcionadas por el Instituto Federal de Tecnología de Zúrich (*ETHZ*, Ess et al., 2007, 2008, 2009a,b), cuatro pertenecientes al proyecto *CAVIAR* (Fisher et al.) y dos extraídas de las base de datos Daimler (Enzweiler and Gavrila, 2009) y Caltech (Dollár et al., 2009, 2012).

Se realizaron diez pruebas sobre cada secuencia de imágenes, comparando los resultados de la ejecución del método presentado y del algoritmo de detección por HOG sin modificar (Fig. 4.5). De nuevo, como se explicó en la sección 3.2, se muestra la relación media entre la tasa de falsos negativos de cada herramienta y los positivos reales, seleccionados manualmente en los fotogramas de partida. Se observa que el enfoque bayesiano presentado siempre produce tasas de detección menos conservadoras que la herramienta de HOG original, lo que reduce con éxito la cantidad de falsos positivos de todas las secuencias.

La precisión aumentó de forma particularmente notable para los conjuntos de imágenes de *ETHZ* y de *CAVIAR*, gracias a que tienen una mayor visibilidad. Esto provoca que las detecciones del método sin modificar mejoren, lo que da un mayor margen de ajuste al procesamiento bayesiano. Las secuencias *ETSII* e *ITER* muestran una mejora en precisión mediocre, debido a que la baja resolución de las cámaras del VERDINO dificulta el trabajo de los detectores de HOG. Lo mismo ocurre con las secuencias *CALTECH* y *DAIMLER*, por tener tasas de oclusión muy altas e incluir muy pocos peatones en poses reconocibles.

Se observa también que, para todas las secuencias de imágenes, la precisión del método de HOG bayesiano supera a la de Viola-Jones bayesiano. Dado que el post-procesamiento es casi idéntico en ambos, esta diferencia indica que la propia herramienta de base tiene una tasa de falsos negativos menor en el primer caso que en el segundo.

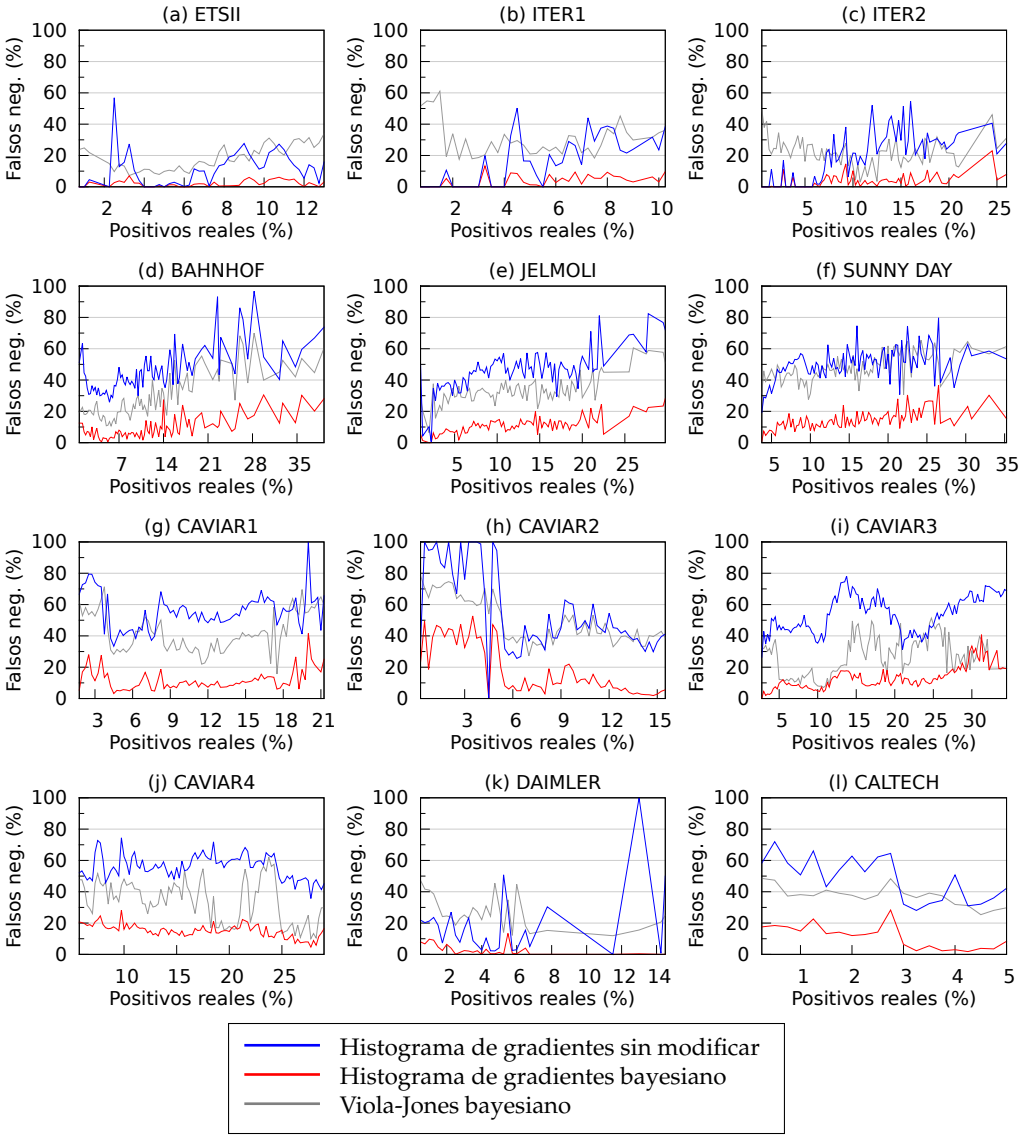


Figura 4.5. Valores medios de la ejecución para cada secuencia de imágenes.

Capítulo

5

Estados de colisión probable dependientes del tiempo

Para asegurar el correcto funcionamiento de un vehículo autónomo en un entorno no controlado, es necesario que las rutas a seguir sean completamente seguras, de modo que no se sufran daños ni se ponga en peligro a los peatones cercanos. Este cómputo requiere tomar decisiones en tiempo real, a la vez que se intenta alcanzar una posición objetivo de la forma más eficiente posible.

El concepto de *estado de colisión inevitable* o **ICS** (*Inevitable Collision State*, Fraichard and Asama, 2003) se define como toda configuración de un vehículo robótico autónomo para la cual una colisión con un obstáculo se vuelve inevitable. Esta definición es válida para entornos en los que los objetos se ajusten a trayectorias conocidas, pero no es una opción segura cuando los obstáculos son peatones.

Para corregir este problema, los *estados de colisión probable* o **PCS** (*Probabilistic Collision State*, Althoff et al., 2010) se basan en asociar a cada región del espacio transitable por un robot la probabilidad de contener un obstáculo. Esta se calcula basándose en la trayectoria observada para cada objeto, y depende tanto de la geometría del mismo como de la del robot.

5.1. Estados de colisión probable

En su publicación de los PCS, para evitar la utilización de intervalos de tiempo infinitos, Althoff et al. restringieron el movimiento del robot a un modelo de deceleración uniforme, de modo que su aceleración y su velocidad forman un ángulo constante y su producto escalar es negativo. Definieron además dos tipos de obstáculos: *pasivos*, si su comportamiento es independiente de la localización del robot, y *activos*, si son capaces de maniobrar para evitar colisiones.

Este modelo es insuficiente para una aplicación real, ya que se limita a movimientos de frenado y asume que todos los objetos se mueven de manera uniforme. Además, la suposición de que los obstáculos evitan activamente una colisión es peligrosa, ya que ningún robot debería esperar que su entorno varíe favorablemente.

Asimismo, los autores tratan todas las posibles localizaciones del área de trabajo simultáneamente, sin tener en cuenta una evolución temporal. Dado que la posición de los obstáculos varía durante el proceso, la trayectoria resultante siempre es incorrecta.

En este capítulo se describe, como método alternativo de generación de rutas, una modificación bayesiana dependiente del tiempo del sistema de PCS, basada en la predicción del movimiento de los obstáculos, que corrige los defectos del estudio de Althoff et al. (2010). Este trabajo fue publicado en Hernández-Aceituno et al. (2015a).

5.1.1. Definiciones

Sea W un espacio de trabajo, definido como una malla bidimensional, dentro del que existen tanto un robot como un número conocido N de obstáculos. $A_R \subset W$ es el área circular mínima que contiene al robot, cuyo radio se denota por r_R, y $A_i \subset W$ es la que contiene al $i-$ésimo obstáculo, cuyo radio se denota por r_i. Por defecto, en los experimentos mostrados en este capítulo, W es una malla de 50×50 celdas.

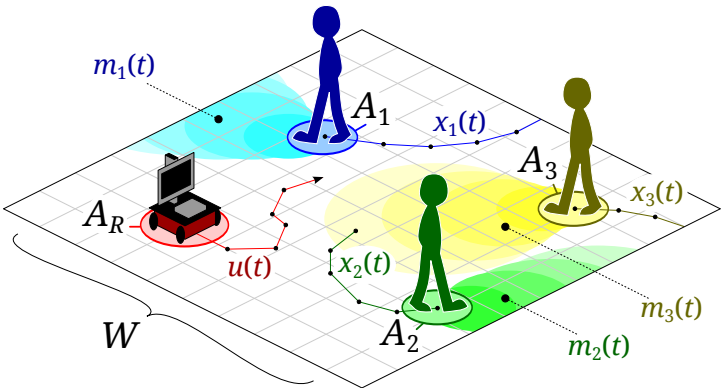


Figura 5.1. Nomenclatura utilizada en los cálculos.

Se asume que cada objeto debe ser observado durante un cierto intervalo de tiempo antes de que su comportamiento futuro pueda ser modelado con precisión. Se denota por $x_i(t)$ a la posición del $i-$ésimo objeto observada en el instante t de la ejecución. El intervalo de tiempo entre dos observaciones consecutivas se denotará por δ_t.

Al contrario que Althoff et al. (2010), no consideraremos intervalos temporales de longitud indefinida, sino ventanas de tiempo solapadas de tamaño fijo T, dentro de las cuales se estimarán las posiciones futuras de los obstáculos.

Puesto que las trayectorias de los obstáculos no se pueden predecir de forma exacta, se define para cada uno una *matriz de localización probable* $m_i(t)$, que asocia a cada celda de W la probabilidad de contener al $i-$ésimo objeto en el instante t de la ejecución.

Mediante la unión de las $m_i(t)$ se obtiene una *matriz global de colisiones* $M(t)$, que permitirá al robot diseñar una trayectoria segura $u(t)$ a través de W, para un intervalo de tiempo T (Fig. 5.1).

5.2. Matrices de localización probable

Para predecir las posiciones futuras de los obstáculos se recurre a un modelo bayesiano secuencial: se desea calcular la matriz de localización $m_i(t)$ a partir de

las posiciones observadas $\{x_i(0), x_i(1), \cdots, x_i(t)\}$ para el $i-$ésimo objeto desde el comienzo de la ejecución hasta el instante t; así pues:

$$p_i(t) \propto \underbrace{p(x_i(t) \mid m_i(t))}_{\textit{verosimilitud}} \times \int \underbrace{p(m_i(t) \mid m_i(t-1))}_{\textit{probabilidad a priori}} \times q_i(t-1)\ \mathrm{d}m_i(t-1),$$

$$\text{donde } q_i(t) = \underbrace{p(m_i(t) \mid x_i(0), x_i(1), \cdots, x_i(t))}_{\textit{probabilidad a posteriori en } t}$$

5.2.1. Verosimilitud

El cálculo de la verosimilitud depende del instante de tiempo considerado: toda observación anterior al momento actual t_A quedará determinada por los dispositivos de sensado, teniendo en cuenta sus errores de medida y posibles etapas de post-procesado de datos, mientras que las observaciones futuras sólo pueden modelarse como una función uniforme sobre todo el espacio de trabajo.

Suponiendo que las localizaciones observadas para un obstáculo determinado i en el momento t vienen dadas como una posición media $\mu_i(t)$ y una desviación típica $\sigma_i(t)$, la función de verosimilitud del modelo utilizado sería:

$$p(x_i(t) \mid m_i(t)) \sim \begin{cases} \mathcal{N}(\mu_i(t), \sigma_i^2(t)) & \text{si } 0 \leq t \leq t_A \\ \mathcal{U}(W) & \text{si } t > t_A \end{cases}$$

5.2.2. Probabilidad a priori

El comportamiento de cada objeto visible se modela como un movimiento bidimensional uniformemente acelerado. Recurriendo a su ecuación continua, se cumple:

$$x_i(t) = x_i(0) + v_i(0)\,(t \cdot \delta_t) + \frac{a_i}{2}\,(t \cdot \delta_t)^2 \tag{5.1}$$

Los valores de x_i no son posiciones concretas, sino distribuciones continuas de probabilidad, debido a que se infieren a partir de medidas con ruido; en consecuencia, lo mismo ocurre con los valores de v_i y a_i.

En este trabajo, la aceleración se modela como una función gaussiana de media μ_{ai} y σ_{ai}, calculada a partir de las variaciones de posición observadas por el sistema sensorial. Llamando $\Delta x_i(t) = x_i(t) - x_i(t-1)$ al desplazamiento puntual en t, a partir de la ecuación (5.1) obtenemos:

$$\begin{aligned}\Delta x_i(t) &= \overbrace{\left(x_i(0) + v_i(0)\, t\, \delta_t + \frac{a_i}{2} t^2 \delta_t^2\right)}^{x_i(t)} - \overbrace{\left(x_i(0) + v_i(0)\,(t-1)\,\delta_t + \frac{a_i}{2}\,(t-1)^2\,\delta_t^2\right)}^{x_i(t-1)} \\ &= v_i(0)\,\delta_t + \frac{a_i}{2}\,(2t-1)\,\delta_t^2\end{aligned}$$

Partiendo de esto, mediante una transformación lineal se puede obtener el modelo de probabilidad de la variación en posición:

$$a_i \sim \mathcal{N}\left(\mu_{ai},\, \sigma_{ai}^2\right) \;\Rightarrow\; \Delta x_i(t) \sim \mathcal{N}\left(v_i(0)\,\delta_t + \mu_{ai}\,\delta_t^2\left(t - \tfrac{1}{2}\right),\, \sigma_{ai}^2\,\delta_t^4\left(t - \tfrac{1}{2}\right)^2\right)$$

Se crea entonces una máscara de convolución que defina la región relativa del espacio a la que un objeto tiene mayor probabilidad de desplazarse, teniendo en cuenta su aceleración calculada, su posición y su velocidad acumulada. Aplicando esta a la matriz de probabilidad de localización *a posteriori* del objeto correspondiente, se obtiene la matriz *a priori* para el siguiente instante de tiempo.

Este cálculo consideraría tanto a los obstáculos como al robot como elementos puntuales. Para que este algoritmo funcione correctamente, es necesario tener en cuenta la geometría de todos los elementos. Esto se logra integrando la matriz de localización probable de cada objeto i en un área circular de radio $r_i + r_R$ — es decir, su propio radio más el del robot.

Llamando $\Delta x_i(t, w)$ a la celda de $\Delta x_i(t)$ correspondiente a la posición $w \in W$, la matriz de probabilidad *a priori* de la posición del $i-$ésimo objeto es:

$$p(m_i(t) \mid m_i(t-1)) = \int_{\|w-z\|_2 \leq r_i + r_R} \Delta x_i(t, z)\, \mathrm{d}z, \text{ para } w \in W$$

De este modo, la probabilidad de colisión con un objeto concreto es, al menos, la del punto más cercano del mismo con el que el robot podría impactar.

5.3. Matriz global de probabilidad de colisión

Cada posición de la matriz global $M(t)$ debe informar sobre la probabilidad de colisión combinada para todos los obstáculos y para cada instante t de la ventana temporal. Para ello se debe aplicar una unión, teniendo en cuenta que las localizaciones probables $m_i(t)$ de diferentes objetos se consideran independientes unas de otras, ya que no intentan esquivarse entre sí.

La regla de la unión para un número elevado de eventos es ineficiente, ya que implica calcular las intersecciones de todas las posibles combinaciones:

$$\begin{aligned} p\left(\bigcup_{i=1}^{n}\epsilon_i\right) = &\sum_{i=1}^{n} p(\epsilon_i) \\ &- \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} p\left(\epsilon_i \cap \epsilon_j\right) \\ &+ \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n} p\left(\epsilon_i \cap \epsilon_j \cap \epsilon_k\right) \\ &- \sum_{i=1}^{n-3}\sum_{j=i+1}^{n-2}\sum_{k=j+1}^{n-1}\sum_{l=k+1}^{n} p\left(\epsilon_i \cap \epsilon_j \cap \epsilon_k \cap \epsilon_l\right) \\ &+ \sum_{i=1}^{n-4}\sum_{j=i+1}^{n-3}\sum_{k=j+1}^{n-2}\sum_{l=k+1}^{n-1}\sum_{m=l+1}^{n} p\left(\epsilon_i \cap \epsilon_j \cap \epsilon_k \cap \epsilon_l \cap \epsilon_m\right) - \cdots \end{aligned}$$

Afortunadamente, las leyes de De Morgan permiten simplificar el cálculo:

$$p\left(\bigcup_{i=1}^{n}\epsilon_i\right) = p\left(\overline{\bigcap_{i=1}^{n}\overline{\epsilon_i}}\right) = 1 - \prod_{i=1}^{n} p(\overline{\epsilon_i}) = 1 - \prod_{i=1}^{n}\{1 - p(\epsilon_i)\}$$

Por tanto, $M(t) = \overline{\bigcap_{i=1}^{N}\overline{m_i(t)}}$.

Los datos correspondientes a una matriz de colisiones probables se representarán mediante un código de color, para mostrar su evolución temporal. Por ejemplo, aunque las colisiones entre obstáculos no se tienen en cuenta en este trabajo, en la figura 5.2 se produciría una superposición de los objetos 1 y 2 en el punto X (ya que los colores de sus trayectorias coinciden), pero no de los objetos 2 y 3 en el punto Y (dado que estos difieren).

A menos que se indique de forma explícita, por defecto los radios de integración $r_i + r_R$ en nuestros cálculos serán de una celda.

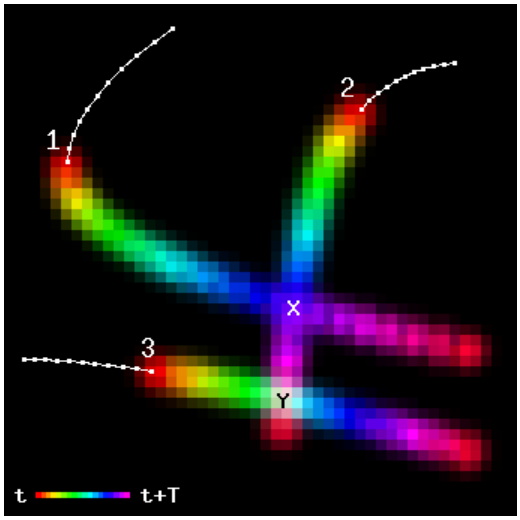


Figura 5.2. Matriz de colisiones $M(t)$ para tres obstáculos móviles. Se asume un error de medida nulo y se representan las trayectorias observadas mediante líneas blancas.

5.4. Algoritmo de Dijkstra dependiente del tiempo

Usando la matriz $M(t)$ se genera una trayectoria $u(t)$ a través de W, desde un punto inicial x_{ini} hasta un punto final x_{fin}, con una probabilidad de colisión mínima y respetando el modelo de movimiento del robot. Para ello, se utiliza una variante dependiente del tiempo del algoritmo de Dijkstra (Alg. 5.1).

Aunque el algoritmo presentado puede adaptarse a cualquier modelo cinemático, por simplicidad se simuló un vehículo no holonómico de tipo Dubins (1957), tal que $(\dot{x}, \dot{y}) = (v\cos\theta, v\sin\theta)$ y $\dot{\theta} \in [-\omega, \omega]$, donde v es la velocidad lineal y ω es la radial. En el ejemplo utilizado, $v = 1$ celda$/\delta_t$ y $\omega = (\pi/4)/\delta_t$.

El coste acumulado C_A de alcanzar al posición y desde la x viene dado por:

$$C_A(y) = C_A(x) + \|y - x\| \times M_y(t)\,, \tag{5.2}$$

donde $\|y - x\|$ es el coste asociado al movimiento, dado por el modelo cinemático específico del robot utilizado, y $M_y(t)$ es la probabilidad de colisión de la posición y en el instante t en que es alcanzada.

Algoritmo 5.1. Algoritmo de Dijkstra dependiente del tiempo

$C_A(x) \leftarrow \begin{cases} 0 & \text{si } x = x_{\text{ini}} \\ \infty & \text{si } x \in W - \{x_{\text{ini}}\} \end{cases}$ {C_A: Coste acumulado}

$t_{LL}(x) \leftarrow \begin{cases} 0 & \text{si } x = x_{\text{ini}} \\ \infty & \text{si } x \in W - \{x_{\text{ini}}\} \end{cases}$ {t_{LL}: Tiempo de llegada}

$V \leftarrow \{x_{\text{ini}}\}$ {V: Lista de localizaciones visitadas recientemente}

mientras $V \neq \emptyset$ **hacer**

- $x \leftarrow$ elemento de V con menor C_A $\{x \in V \mid C_A(x) \leq C_A(v)\ \forall v \in V\}$
- $V \leftarrow V - \{x\}$
- **para todo** $y \in W$ accesible desde x (según el modelo cinemático) **hacer**
  - $k \leftarrow C_A(x) + \|y - x\| \times M_y(t_{LL}(x) + \delta_t)$ {Ecuación (5.2)}
  - **si** $k < C_A(y)$ **ó** $\left(k \not> C_A(y)\ \textbf{y}\ t_{LL}(y) > t_{LL}(x) + \delta_t\right)$ **entonces**
    - $C_A(y) \leftarrow k$
    - $L_P(y) \leftarrow x$ {L_P: Localización precedente}
    - $t_{LL}(y) \leftarrow t_{LL}(x) + \delta_t$
    - $V \leftarrow V \cup \{y\}$

$z \leftarrow x_{\text{fin}}$ {Generación de la trayectoria}

mientras $u(0) \neq x_{\text{ini}}$ **hacer**

- $u(t_{LL}(z)) \leftarrow z$
- $z \leftarrow L_P(z)$

devolver trayectoria $u(t)$

Obsérvese, en la figura 5.3, que una trayectoria calculada mediante el algoritmo 5.1 *sólo* evita atravesar el camino de un obstáculo cuando esto le supone un riesgo de colisión.

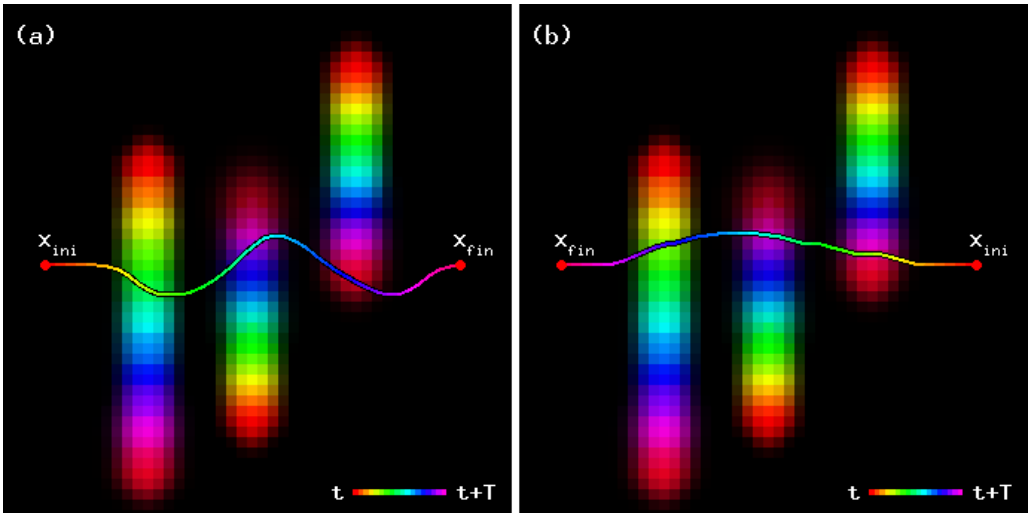


Figura 5.3. Trayectorias a través de un entorno concurrido ($T = 45\,\delta_t$).

La probabilidad total de colisión de una trayectoria del robot $p_C(u)$ se calcula como una unión, tal que el instante en que se alcanza cada posición afecta al resultado:

$$p_C(u) = \bigcup_{t=0}^{T_u} M_{u(t)}(t)$$

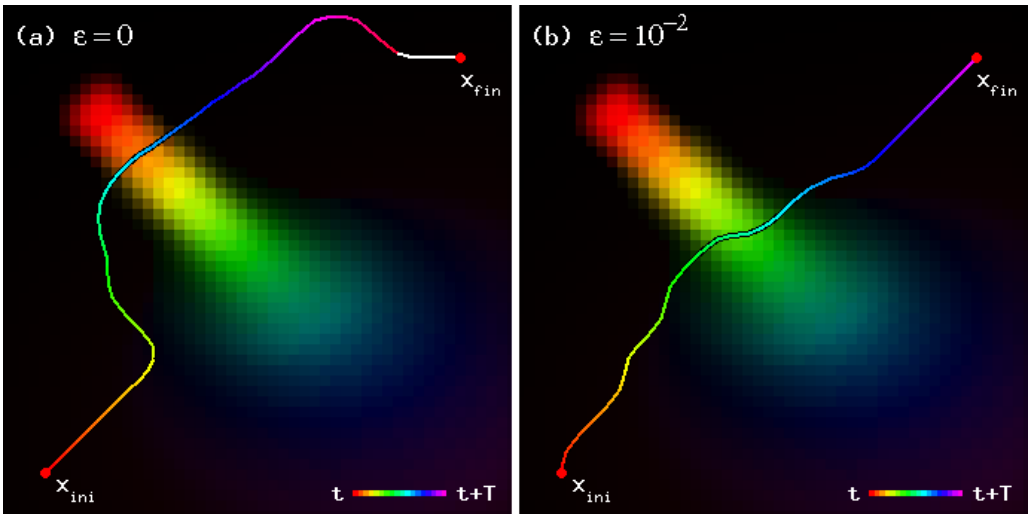
donde T_u es el tiempo necesario para que el robot llegue a su destino (equivalente a $t_{LL}(x_{\text{fin}})$ en el algoritmo presentado). La longitud espacial de una trayectoria se denotará por $\|u\|$ y se calculará como

$$\|u\| = \sum_{t=1}^{T_u} \|u(t) - u(t-1)\|_2$$

Podría considerarse que el método mostrado es similar al algoritmo *D* Lite* (Koenig and Likhachev, 2005), dado que el momento de llegada a una celda afecta a su coste, pero el enfoque presentado no requiere recalcular la ruta completa al detectar un obstáculo imprevisto. Esto se debe a que todos los pasos de la evolución de la matriz de colisión se tienen en cuenta simultáneamente durante el cómputo de una ruta. Sin embargo, en caso de ejecución secuencial, una ruta precalculada no puede ser reutilizada, ya que la matriz de colisiones se reconfigura completamente para cada iteración del algoritmo.

5.4.1. Restricción de mínima probabilidad

En ocasiones, la solución obtenida por el algoritmo puede ser excesivamente conservadora y tomar desvíos innecesarios. Se pueden producir trayectorias más razonables mediante la definición de una matriz de colisiones probables alternativa $M^{\varepsilon}(t)$, tal que $M^{\varepsilon}_x(t) = \max\{M_x(t), \varepsilon\}$ (Fig. 5.4).



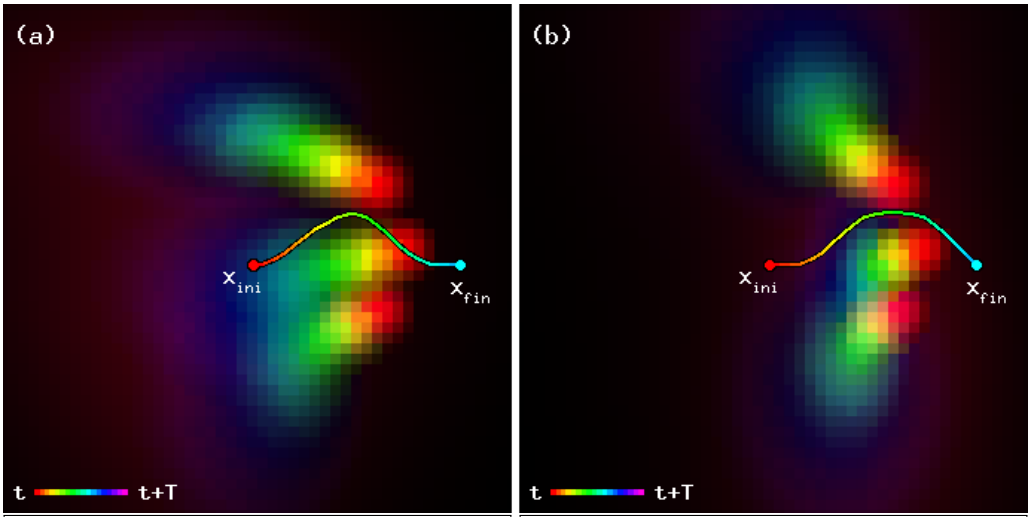
$p_C(u) = 1{,}0028 \times 10^{-3}$ $T_u = 63\,\delta_t$ $\quad \|u\| = 78{,}57$ celdas	$p_C(u) = 1{,}0346 \times 10^{-3}$ $T_u = 49\,\delta_t$ $\quad \|u\| = 61{,}26$ celdas

Figura 5.4. Restricción de mínima probabilidad sobre una trayectoria ($T = 50\,\delta_t$).

5.5. Resultados en simulación

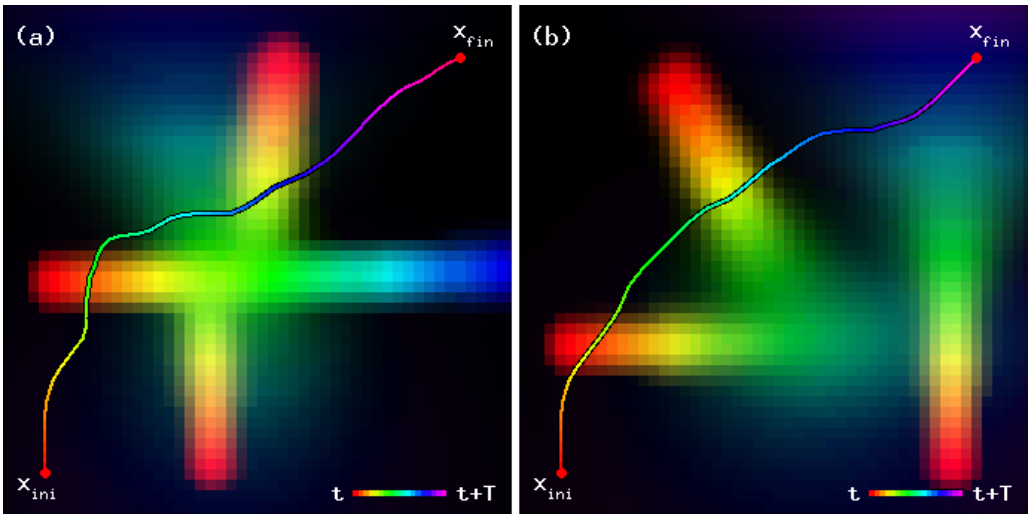
Para comparar los resultados del método presentado con los publicados por Althoff et al. (2010), se reprodujeron sus experimentos tanto para obstáculos pasivos como activos, pidiendo al robot trazar una trayectoria a través del área de trabajo (Fig. 5.5). Puesto que el estudio original no tenía en cuenta la evolución temporal de los objetos, nuestros resultados son muy diferentes: el robot es capaz de maniobrar en torno a los obstáculos, en lugar de limitarse a trayectorias de frenado, manteniendo una probabilidad de colisión muy baja.

Nuestro método también fue probado sobre 100 configuraciones más complejas generadas automáticamente, en las que los objetos fueron programados para seguir trayectorias curvas aleatorias que siempre se interpusiesen en el camino del robot (Tab. 5.1, Fig. 5.6).

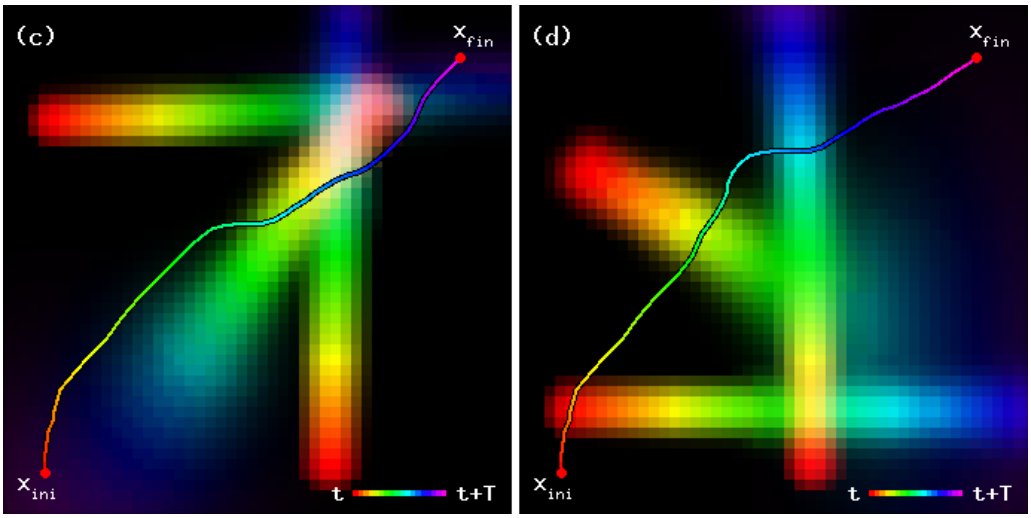


$p_C(u) = 6{,}1453 \times 10^{-3}$	$p_C(u) = 9{,}4258 \times 10^{-6}$
$T_u = 21\,\delta_t \qquad \|u\| = 24{,}14$ celdas	$T_u = 21\,\delta_t \qquad \|u\| = 24{,}14$ celdas

Figura 5.5. Ejemplos de Althoff et al. (2010) con obstáculos pasivos (a) y activos (b).



$p_C(u) = 5{,}0434 \times 10^{-3}$ $T_u = 58\,\delta_t$ $\quad \|u\| = 66{,}52$ celdas	$p_C(u) = 1{,}4190 \times 10^{-3}$ $T_u = 53\,\delta_t$ $\quad \|u\| = 63{,}60$ celdas



$p_C(u) = 1{,}1145 \times 10^{-2}$ $T_u = 51\,\delta_t$ $\quad \|u\| = 62{,}43$ celdas	$p_C(u) = 0$ $T_u = 54\,\delta_t$ $\quad \|u\| = 64{,}18$ celdas

Figura 5.6. Ejemplos de trayectorias con mínima probabilidad de colisión.

Tabla 5.1. Parámetros de configuración y resultados de las simulaciones.

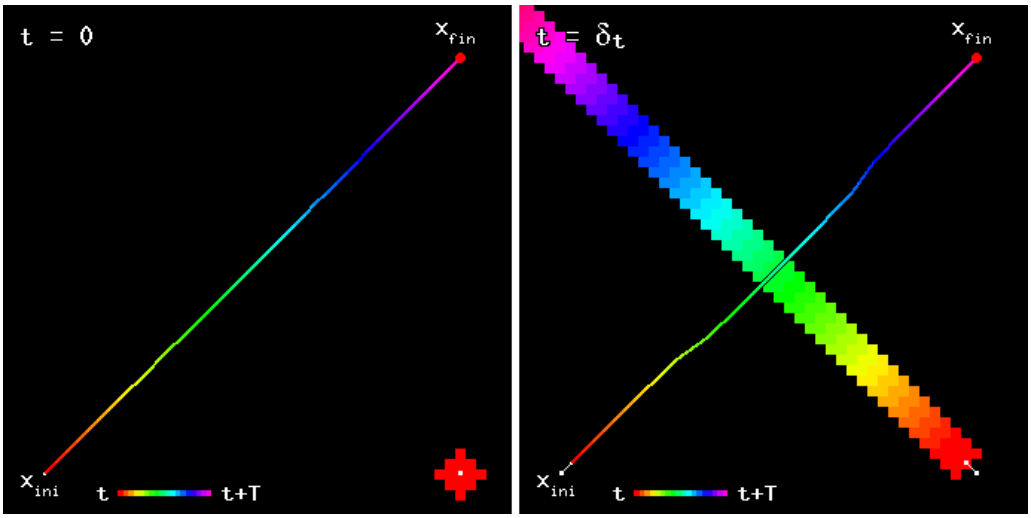
Número de objetos	$N = 3$
Tamaño de la ventana temporal	$T = 50\,\delta_t$
Restricción de probabilidad	$\varepsilon = 10^{-3}$
Radio de integración del objeto 1	$r_R + r_1 = 2$ celdas
Radio de integración del objeto 2	$r_R + r_2 = 1$ celda
Radio de integración del objeto 3	$r_R + r_3 = 1$ celda
Probabilidad media de colisión	$\overline{p_C(u)} = 1{,}548 \times 10^{-3}$
Tiempo medio de recorrido	$\overline{T_u} = 53{,}96\,\delta_t$
Longitud media de la trayectoria	$\overline{\|u\|} = 64{,}64$ celdas

Se obtuvieron probabilidades de colisión muy bajas para todos los casos, excepto aquellos en los que las trayectorias de los objetos ocupasen las posiciones inicial o final del robot, o bloqueasen completamente el espacio entre ambas.

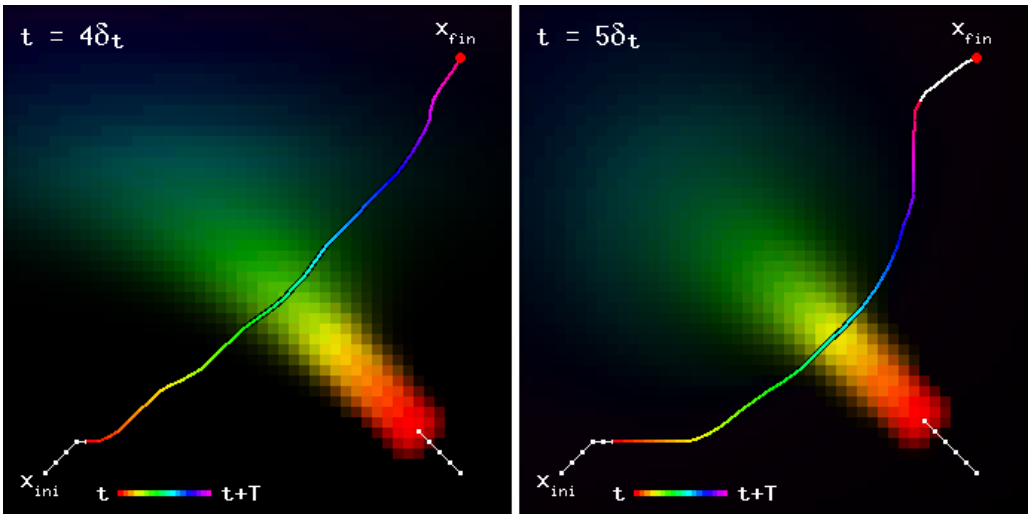
5.5.1. Ejecución secuencial

Aunque el algoritmo presentado produce trayectorias válidas, depende en gran medida de suposiciones respecto al comportamiento futuro de los peatones observados, sujetas a cambios constantes según avanza la ejecución. Se requiere un enfoque más complejo para poder aplicar este método casos reales.

Cada trayectoria generada se recorre sólo durante un cierto intervalo de tiempo, idealmente δ_t, mientras se observan los obstáculos y se calcula una nueva trayectoria (Fig. 5.7). Aunque los experimentos realizados asumen que el robot se mueve con velocidad constante, se pueden incluir maniobras de frenado o aceleración en el modelo de movimiento para generar trayectorias más realistas.

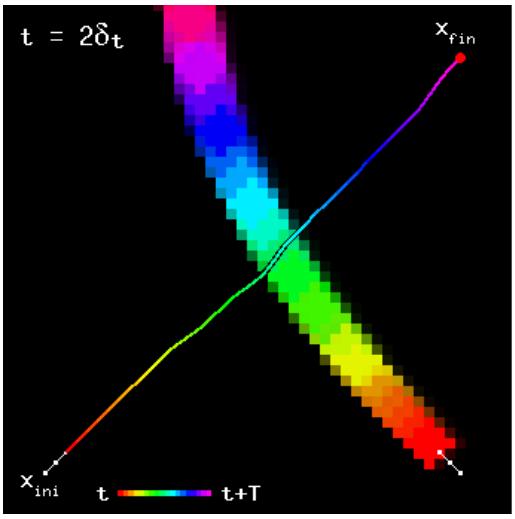


$p_C(u) = 0$ $T_u = 41\,\delta_t \qquad \|u\| = 56{,}57$ celdas	$p_C(u) = 0$ $T_u = 41\,\delta_t \qquad \|u\| = 55{,}74$ celdas

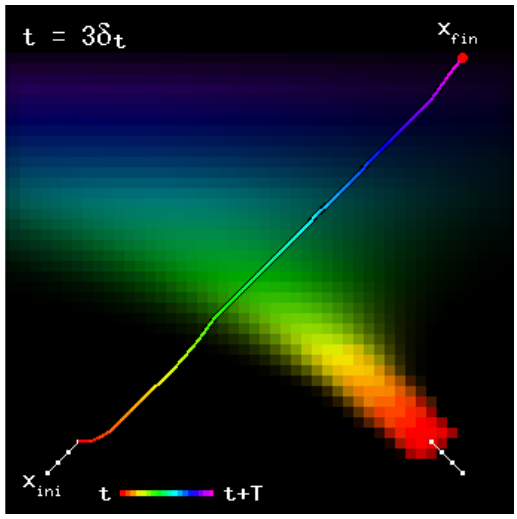


$p_C(u) = 4{,}4369 \times 10^{-2}$ $T_u = 43\,\delta_t \qquad \|u\| = 54{,}84$ celdas	$p_C(u) = 5{,}4746 \times 10^{-2}$ $T_u = 51\,\delta_t \qquad \|u\| = 59{,}11$ celdas

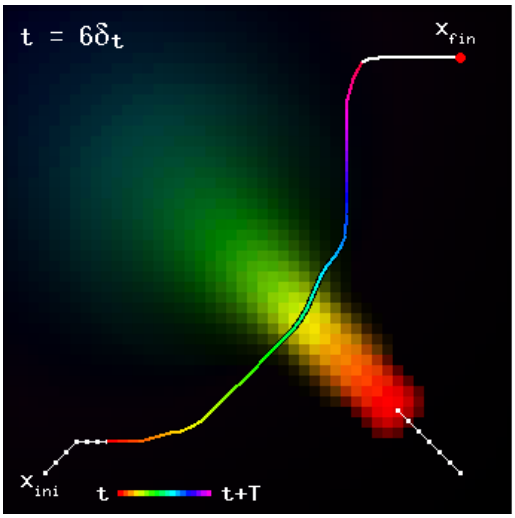
Figura 5.7. Ejemplo de cálculo secuencial de una trayectoria con un solo obstáculo.



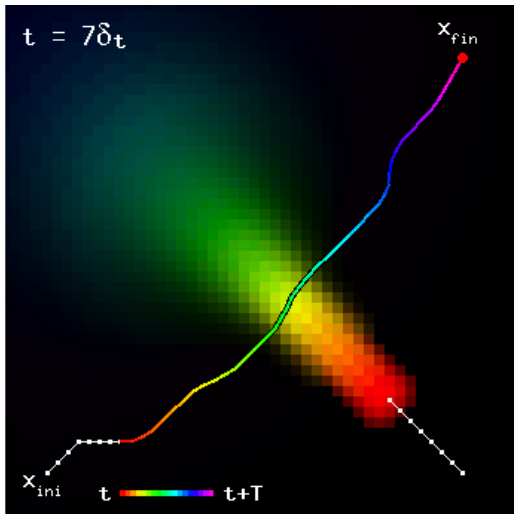
$p_C(u) = 0$
$T_u = 41\,\delta_t \qquad \|u\| = 54{,}91$ celdas



$p_C(u) = 8{,}2291 \times 10^{-2}$
$T_u = 40\,\delta_t \qquad \|u\| = 53{,}50$ celdas



$p_C(u) = 7{,}4170 \times 10^{-2}$
$T_u = 55\,\delta_t \qquad \|u\| = 61{,}04$ celdas



$p_C(u) = 5{,}3002 \times 10^{-2}$
$T_u = 42\,\delta_t \qquad \|u\| = 53{,}01$ celdas

$(T = 45\,\delta_t\,,\ \varepsilon = 10^{-3},\ r_R + r_1 = 2$ celdas$)$

5.6. Resultados experimentales

Tras verificar su eficacia teórica, nuestro algoritmo fue instalado en un dispositivo Pioneer (apéndice B), al que se ordenó realizar distintos recorridos dentro de un área de 4,25 $m \times 4{,}25\,m$ de un pasillo concurrido.

Las trayectorias de mínima probabilidad de colisión se calcularon de forma secuencial, usando una cuadrícula de 50×50 celdas, una ventana temporal de tamaño $T = 30\,s$ y un parámetro de restricción $\varepsilon = 0{,}1$. Se dio a cada peatón un radio de integración de un metro.

En la figura 5.8, la línea discontinua blanca corresponde a la trayectoria del robot y las de colores a los movimientos observados de los obstáculos.

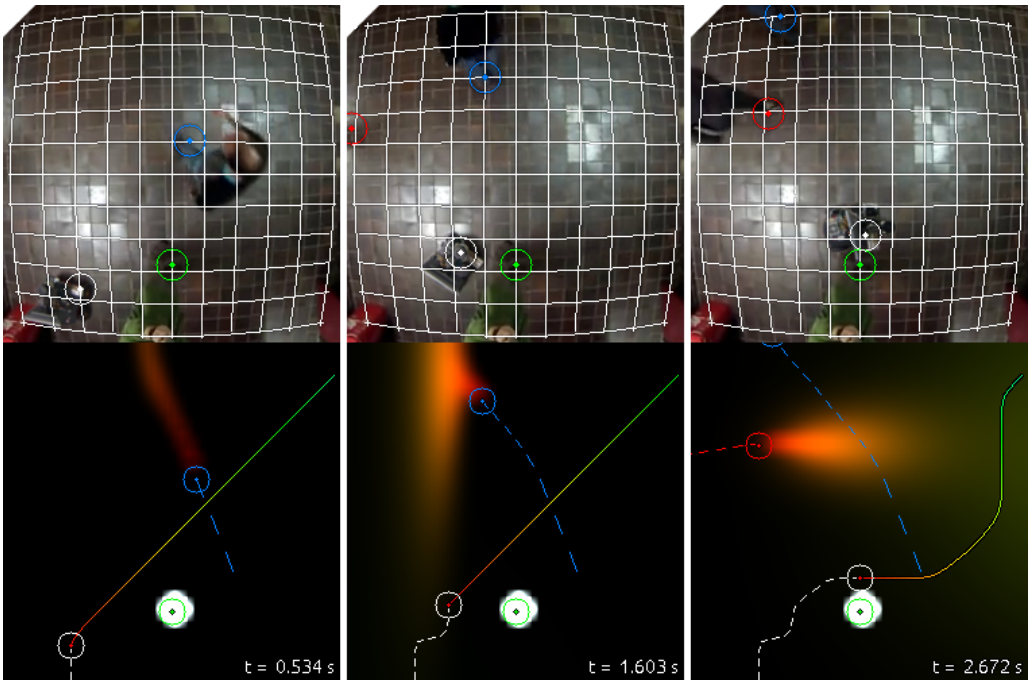


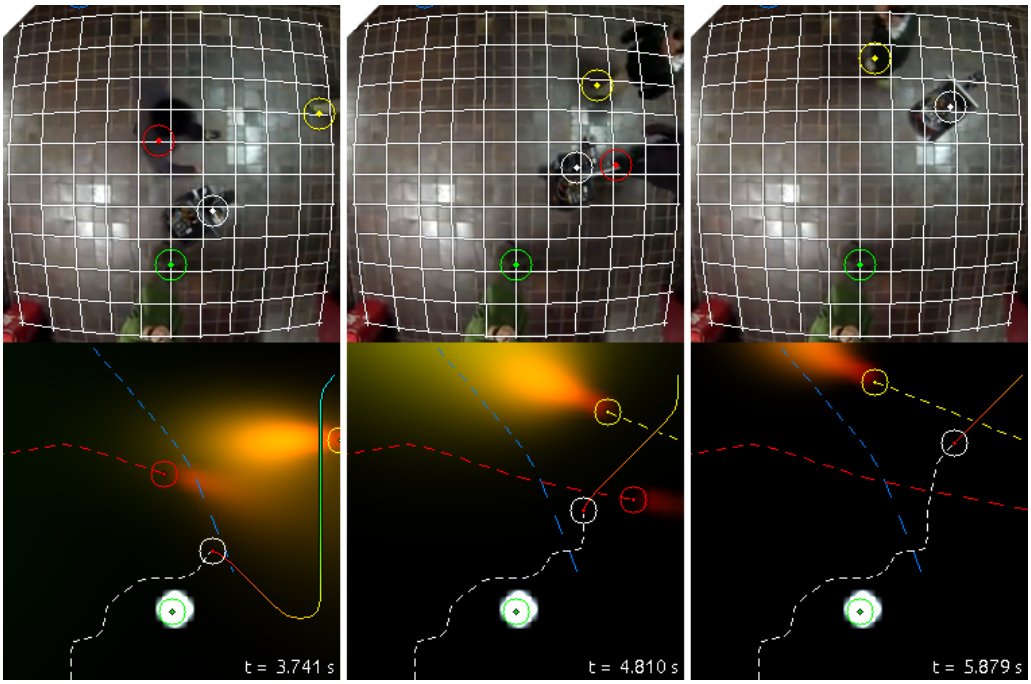
Figura 5.8. Ejemplo de ejecución secuencial en un entorno poblado.

Se definen dos nuevas variables:

- δ_u representa el incremento proporcional de la longitud de la trayectoria recorrida por el robot, respecto a la distancia mínima entre las posiciones de partida y objetivo,

$$\delta_u = \frac{\|x_{\text{fin}} - x_{\text{ini}}\|_2 - \|u\|}{\|x_{\text{fin}} - x_{\text{ini}}\|_2}$$

- Se denota por τ al tiempo de procesamiento de una iteración completa del algoritmo, incluyendo los cálculos tanto de la matriz de probabilidad de colisión como de la trayectoria óptima.



Para estudiar la eficiencia del método presentado, se calcularon los valores medios de δ_u, τ y $p_C(u)$ para 100 configuraciones diferentes de resolución del espacio de trabajo W, de tamaño de ventana temporal T y de valor mínimo de

Tabla 5.2. Incremento medio $\overline{\delta_u}$ de la distancia recorrida.

		Resolución del espacio de trabajo W (celdas)				
T	ε	20×20	30×30	50×50	80×80	120×120
$3\,\delta_t$	0	26,88 %	22,80 %	8,599 %	7,432 %	5,701 %
	0,05	13,34 %	12,89 %	9,583 %	8,045 %	4,886 %
	0,1	13,33 %	11,24 %	9,583 %	7,432 %	3,665 %
	0,5	3,339 %	2,985 %	2,697 %	2,534 %	2,443 %
$5\,\delta_t$	0	31,88 %	24,46 %	12,53 %	8,045 %	6,108 %
	0,05	15,84 %	17,85 %	9,583 %	8,044 %	4,072 %
	0,1	13,34 %	12,90 %	8,599 %	7,432 %	2,850 %
	0,5	3,340 %	2,985 %	2,671 %	2,533 %	2,443 %
$10\,\delta_t$	0	54,98 %	31,06 %	22,15 %	232,4 %	7,737 %
	0,05	26,87 %	22,80 %	19,42 %	9,269 %	6,515 %
	0,1	23,34 %	12,89 %	3,681 %	2,534 %	2,442 %
	0,5	3,330 %	2,984 %	2,697 %	2,533 %	2,443 %
$20\,\delta_t$	0	52,48 %	54,99 %	29,26 %	22,13 %	15,48 %
	0,05	26,88 %	22,80 %	21,39 %	8,045 %	2,442 %
	0,1	26,87 %	21,15 %	2,697 %	2,533 %	2,443 %
	0,5	26,88 %	2,984 %	2,696 %	2,534 %	2,442 %
$30\,\delta_t$	0	52,48 %	53,34 %	46,87 %	29,37 %	22,16 %
	0,05	26,88 %	22,80 %	23,36 %	11,72 %	2,442 %
	0,1	26,87 %	21,15 %	2,697 %	2,534 %	2,443 %
	0,5	3,339 %	2,985 %	2,696 %	2,533 %	2,443 %

probabilidad ε (tablas 5.2 a 5.4). Téngase en cuenta que los valores se calculan respecto a la trayectoria final del robot al terminar la ejecución.

Tabla 5.3. Tiempo de procesamiento medio $\overline{\tau}$ para una iteración del algoritmo.

		Resolución del espacio de trabajo W (celdas)				
T	ε	20×20	30×30	50×50	80×80	120×120
$3\,\delta_t$	0	2308 μs	6667 μs	25,49 ms	96,75 ms	306,0 ms
	0,05	1818 μs	7273 μs	27,88 ms	104,2 ms	332,9 ms
	0,1	2727 μs	6875 μs	27,69 ms	103,0 ms	331,4 ms
	0,5	2778 μs	5926 μs	25,56 ms	105,7 ms	335,7 ms
$5\,\delta_t$	0	2500 μs	6750 μs	27,64 ms	103,2 ms	320,3 ms
	0,05	2609 μs	7223 μs	29,23 ms	107,5 ms	340,9 ms
	0,1	2727 μs	7273 μs	27,84 ms	105,4 ms	341,4 ms
	0,5	2222 μs	6666 μs	26,00 ms	105,8 ms	346,0 ms
$10\,\delta_t$	0	2424 μs	6136 μs	28,15 ms	97,52 ms	350,0 ms
	0,05	2692 μs	7179 μs	30,16 ms	110,7 ms	355,5 ms
	0,1	2308 μs	7576 μs	28,70 ms	112,8 ms	360,7 ms
	0,5	2222 μs	6296 μs	26,44 ms	110,7 ms	361,2 ms
$20\,\delta_t$	0	2188 μs	6600 μs	28,06 ms	116,5 ms	398,8 ms
	0,05	2692 μs	7180 μs	30,00 ms	119,9 ms	390,8 ms
	0,1	2308 μs	7105 μs	30,67 ms	117,9 ms	390,6 ms
	0,5	2692 μs	6667 μs	27,33 ms	115,4 ms	389,2 ms
$30\,\delta_t$	0	2188 μs	6531 μs	27,38 ms	115,4 ms	415,2 ms
	0,05	2692 μs	7436 μs	29,39 ms	124,9 ms	413,1 ms
	0,1	2308 μs	7368 μs	30,22 ms	120,8 ms	411,5 ms
	0,5	2778 μs	5926 μs	27,11 ms	116,9 ms	409,3 ms

Tabla 5.4. Probabilidad media de colisión $\overline{p_C(u)}$

		Resolución del espacio de trabajo W (celdas)				
T	ε	20×20	30×30	50×50	80×80	120×120
$3\,\delta_t$	0	10^{-3}	10^{-3}	10^{-9}	10^{-9}	10^{-9}
	0,05	0,127	0,094	0,05	0,05	0,052
	0,1	0,173	0,1	0,099	0,099	0,1
	0,5	0,545	0,528	0,51	0,5	0,5
$5\,\delta_t$	0	10^{-6}	10^{-7}	10^{-9}	0,033	10^{-9}
	0,05	0,122	0,05	0,05	0,05	0,503
	0,1	0,187	0,115	0,101	0,099	0,1
	0,5	0,546	0,53	0,509	0,5	0,5
$10\,\delta_t$	0	10^{-6}	10^{-6}	10^{-7}	10^{-8}	10^{-9}
	0,05	0,05	0,05	0,05	0,05	0,504
	0,1	0,1	0,114	0,105	0,099	0,1
	0,5	0,543	0,531	0,509	0,5	0,5
$20\,\delta_t$	0	10^{-6}	10^{-6}	10^{-7}	10^{-9}	10^{-4}
	0,05	0,05	0,05	0,05	0,05	0,05
	0,1	0,1	0,103	0,108	0,099	0,1
	0,5	0,542	0,53	0,509	0,5	0,5
$30\,\delta_t$	0	10^{-6}	10^{-6}	10^{-7}	10^{-9}	10^{-4}
	0,05	0,05	0,05	0,05	0,05	0,05
	0,1	0,1	0,103	0,108	0,099	0,1
	0,5	0,543	0,53	0,508	0,5	0,5

Los datos experimentales demuestran que el tiempo de cómputo del algoritmo depende casi exclusivamente de la resolución del espacio de trabajo. Dado que cada iteración del proceso incluye tanto etapas de obtención de datos visuales como de envío de comandos al sistema motriz, este parámetro debe ajustarse con cuidado, de modo que puedan obtenerse trayectorias precisas en intervalos de tiempo pequeños.

Un incremento de este valor también produce trayectorias más cortas, debido a que un número elevado de celdas permite al algoritmo considerar movimientos más precisos sobre el área de trabajo. Sin embargo, el coste de esta precisión puede provocar que el robot se salte pasos de la ejecución y calcule trayectorias erróneas. La resolución óptima para nuestros experimentos se estimó en 50×50 celdas.

Los valores más altos de ε resultaron en trayectorias más cortas y probabilidades de colisión mayores; experimentalmente, se observó que el robot se comportaba de manera más imprudente y se acercaba más a los obstáculos. Sin embargo, un valor de $\varepsilon = 0$ puede producir rutas extremadamente ineficientes, con desvíos largos y maniobras de frenado innecesarias.

Se observó que un valor de $\varepsilon = 0{,}1$ daba resultados suficientemente buenos en nuestro caso de estudio. Obsérvese que, en nuestros cálculos *a posteriori* sobre el resultado final del algoritmo, la probabilidad de colisión de una trayectoria $p_C(u)$ a menudo se asemeja al valor del parámetro de restricción ε (Tab. 5.4).

Por último, no se observó que el tamaño de la ventana temporal T afectara al tiempo de procesamiento de forma significativa, ya que nuestro algoritmo explora todo el espacio de trabajo W independientemente de los datos disponibles de probabilidad de colisión. Sin embargo, valores elevados de T produjeron trayectorias más cortas con menor probabilidad de colisión, debido a que la precisión del método aumenta. Por tanto, se sugiere dar a este parámetro un valor mayor o igual al del tiempo estimado que el robot requiere para alcanzar su destino.

Capítulo

6

Detección de obstáculos mediante tiempo de vuelo en exteriores

En el capítulo 2 se describió el *tiempo de vuelo* como un método de detección de obstáculos mediante reflexión lumínica: un sensor basado en esta tecnología emite un pulso de luz, generalmente infrarroja, y calcula el tiempo transcurrido hasta la captura de su reflexión. Conociendo el valor de la velocidad de la luz c, la distancia s entre el sensor y un objeto detectado puede calcularse fácilmente como $s = c \times t/2$, donde t representa el tiempo transcurrido entre la emisión y la captura del pulso reflejado.

El uso de los telémetros láser planares en la detección de obstáculos ha sido ampliamente discutido en capítulos anteriores. Sin embargo, el campo de visión de este tipo de dispositivos está limitado a un único plano de barrido, lo que puede llevar a malinterpretaciones de los objetos observados y a ignorar obstáculos cuya altura es inferior a la posición del sensor. Aunque existen telémetros capaces de realizar barridos multiplanares, tales como los dispositivos *Velodyne* (Halterman and Bruch, 2010), su precio y el coste computacional que requieren resultan prohibitivos para un prototipo de bajo coste como VERDINO (apéndice A).

Los dispositivos *Kinect* de Microsoft son ampliamente utilizados en interiores para la captura de datos de profundidad, que adecuadamente procesados permiten superar los defectos de la telemetría láser.

La versión 2.0 del dispositivo *Kinect* (Fig. 6.1) contiene una cámara de tiempo de vuelo de 512 × 424 píxeles (Bamji et al., 2015), con un ángulo de visión de 70° en el eje horizontal y de 60° en el vertical, en tanto que la versión 1.0 cubría sólo 57° en el horizontal y 43° en el vertical. Esta mejora de la lente implica una reducción en la distancia mínima necesaria para obtener lecturas de profundidad: mientras que para la primera versión del sensor esta debía ser superior a 2,4 m, se observó experimentalmente que el nuevo modelo es capaz de percibir obstáculos en un rango de 50 cm a 4,2 m.

Figura 6.1. Sensor Kinect 2.0

El sensor *Kinect* supera en precisión tanto a los telémetros planares, gracias a su capacidad de captura en tres dimensiones, como a la estereovisión (capítulo 1), ya que esta se basa únicamente en información visual y se ve por tanto afectada de forma más agresiva por las variaciones de brillo y de textura de las superficies visibles.

Sin embargo, el alcance del tiempo de vuelo se ve gravemente mermado en exteriores, ya que la luz natural tiende a saturar los receptores lumínicos. Se propone entonces un algoritmo que procese y simplifique los datos de profundidad devueltos por un sensor tipo *Kinect*, tal que su instalación en el prototipo VERDINO sea viable.

6.1. Segmentación de obstáculos

La navegabilidad de un área de trabajo puede definirse a partir de los vectores normales de las superficies visibles que contiene. Lógicamente, el terreno sobre el que un vehículo puede navegar con seguridad debe incluir únicamente normales suficientemente verticales (Fig. 6.2).

Dado que se asume que el vehículo se mueve sobre el plano XY, el valor de la componente Z de los vectores normales de las zonas seguras ha de estar próximo

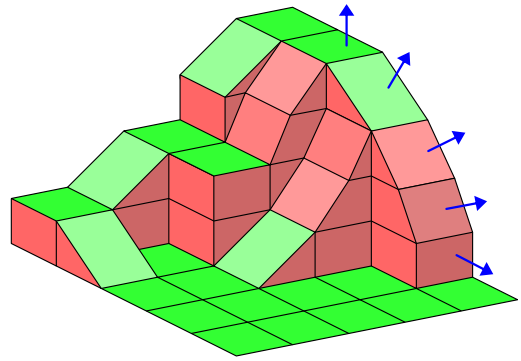


Figura 6.2. Distinción entre superficies navegables (en verde) y no navegables (en rojo) en función de sus vectores normales (en azul).

a 1 (asumiendo que dichos vectores son unitarios). Recíprocamente, los obstáculos se reconocen por poseer normales con un valor bajo en su componente Z.

La entrada del algoritmo de segmentación de obstáculos desarrollado es una nube de puntos tridimensional, generada por el sensor *Kinect* o a partir de la salida de un par estereovisor, que debe ser transformada al sistema de coordenadas del robot antes de poder ser manipulada. El número de muestras de esta nube se reduce realizando una intersección con una región tridimensional de control, lo que permite descartar aquellos puntos que resulten poco realistas o demasiado alejados del sensor como para que su posición sea fiable.

Así, todos los puntos espurios situados detrás del sensor o bajo tierra se eliminan de la nube; sin embargo, se tolera un margen de $-50\,cm$ en el eje vertical, para tener el cuenta la existencia de rampas descendentes navegables. Todos los puntos a más de $1\,m$ por encima del sensor se asumen superfluos, dado que se espera que todos los obstáculos de interés estén en contacto con el suelo. Toda medida alejada más de $20\,m$ a la izquierda, a la derecha o por delante del sensor también se ignora.

A continuación, las medidas que aparezcan aisladas se eliminan mediante un filtrado de *malla de vóxeles* (Rusu and Cousins, 2011). Este mecanismo reduce aún más el tamaño de la nube de puntos, a través de la aproximación de conjuntos de elementos cercanos entre sí al valor de sus respectivos centroides. Para el algoritmo presentado en este capítulo, el tamaño óptimo de las aristas de cada vóxel cúbico se fijó de manera empírica a $25\,mm$.

Partiendo de la nube resultante se puede aproximar el valor del vector normal correspondiente a cada punto, mediante una estimación por mínimos cuadrados de un plano tangente a la superficie definida por sus vecinos (Rusu, 2009). Así, para cada punto p_i se resuelve la igualdad $C \cdot v_j = \lambda_j \cdot v_j$, donde λ_j es el $j-$ésimo autovalor y v_j el $j-$ésimo autovector de la matriz C, calculada como

$$C = \frac{1}{n} \sum_{k=1}^{n} (p_k - \bar{p}) \cdot (p_k - \bar{p})^T,$$

donde n es el número de vecinos del punto p_i y $\bar{p}$ es el centroide de los mismos.

Finalmente, los puntos de la nube que no corresponden a obstáculos son descartados. Siendo θ el ángulo de inclinación de la pendiente máxima por la que el robot puede subir, toda zona cuyo vector normal tenga una componente Z mayor que $\cos(\theta)$ es navegable. Dado que el algoritmo de estimación no puede calcular el sentido de los vectores, se debe tener en cuenta el valor absoluto de dicha componente.

Siguiendo esta descripción, todo punto no navegable con un número suficientemente alto de vecinos no navegables es considerado parte de un obstáculo (Fig. 6.3). El proceso completo de segmentación de obstáculos puede resumirse como se muestra en el algoritmo 6.1.

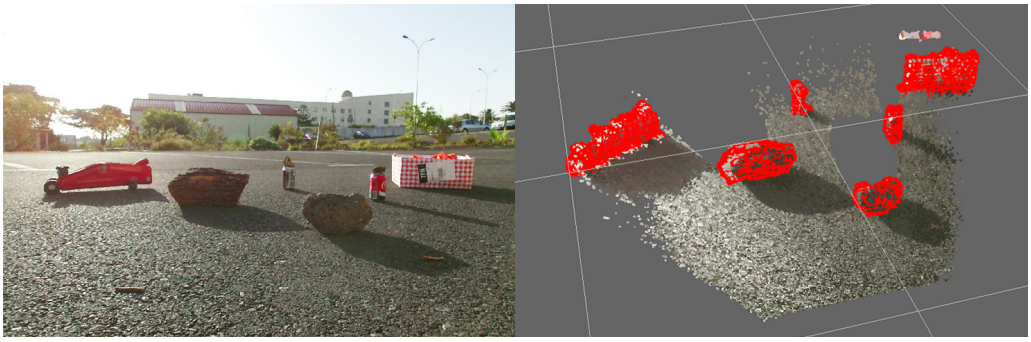


Figura 6.3. Datos visuales devueltos por un sensor Kinect (izquierda) y nube de puntos tridimensional correspondiente (derecha) con segmentación de obstáculos (en rojo).

Algoritmo 6.1. Segmentación de obstáculos

$PC_0 \leftarrow$ nube de puntos tridimensional observada
$PC_1 \leftarrow PC_0 \cap$ región admisible del espacio, relativa a la posición del robot
$PC_2 \leftarrow PC_1$ filtrada por una malla de vóxeles cúbicos de 25 mm de arista
Estimación de los vectores normales de los puntos de PC_2
$PC_3 \leftarrow \emptyset$
$N \leftarrow$ número mínimo de vecinos
$R \leftarrow$ radio máximo de búsqueda de vecinos
$\theta \leftarrow$ ángulo máximo de inclinación para superficies navegables
para todo $p_i \in PC_2$ **hacer**
  $z_i \leftarrow$ componente Z del vector normal asociado a p_i
  si $|z_i| < \cos(\theta)$ **entonces**
    $n \leftarrow 0$
    para todo $p_j \in PC_2$, donde $\left\|p_i - p_j\right\|_2 < R$ **hacer**
      $z_j \leftarrow$ componente Z del vector normal asociado a p_j
      si $|z_j| < \cos(\theta)$ **entonces**
        $n \leftarrow n + 1$
    si $n \geq N$ **entonces**
      $PC_3 \leftarrow PC_3 \cup \{p_i\}$
devolver PC_3

6.2. Integración en el mapa de costes

Para generar rutas seguras, el sistema de navegación del prototipo VERDINO hace uso de un *mapa de costes*, una malla bidimensional sobre la que se proyectan ortogonalmente los objetos observados, descartando la coordenada vertical de los puntos tridimensionales que los componen. Cada celda del mapa lleva asociado un coste, calculado en función de su proximidad a un obstáculo físico (Fig. 6.4).

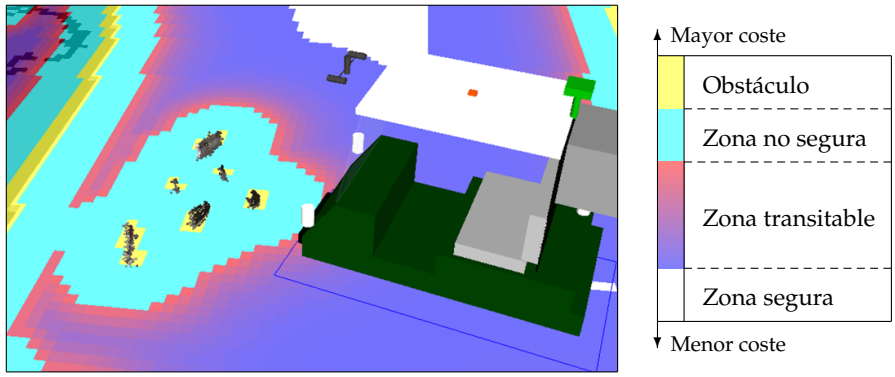


Figura 6.4. Ejemplo de mapa de costes usando segmentación de obstáculos.

El montaje físico de los sensores utilizados en este capítulo es el siguiente:

- Existen dos telémetros láser fijos en ambas esquinas frontales, orientados diagonalmente hacia el exterior, y un tercero en la parte posterior, alineado con el eje longitudinal del prototipo (Fig. 6.6). Cada uno tiene un ángulo de visión de 270° y recoge información de profundidad correspondiente al plano XY a 75 cm sobre el nivel del suelo.

- El dispositivo *Kinect* se instaló en la parte frontal del robot, a 25 cm sobre el suelo y apuntando directamente hacia delante. Se encuentra protegido de la radiación solar gracias a la carrocería del propio vehículo.

- Se instaló también una cámara estereoscópica de *PlayStation 4* (*PS4*, Fig. 6.5), encima del sensor *Kinect*, a 29 cm sobre el suelo y con igual orientación.

Se condujo una breve serie de experimentos que permitió descartar la telemetría del presente estudio. Los sensores basados en láser aportan medidas de profundidad muy precisas, pero no pueden aportar información tridimensional acerca del entorno, como lo hacen las nubes de puntos que generan la estereovisión y el tiempo de vuelo.

Figura 6.5. Cámara de PS4.

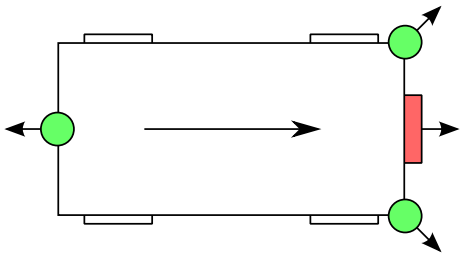


Figura 6.6. Situación y orientación en el robot de los telémetros (en verde) y de los sensores de tiempo de vuelo y estereovisión (en rojo).

Ante estructuras suficientemente complejas, como las que ilustran los siguientes ejemplos, esta diferencia puede llevar a equivocaciones respecto a la navegabilidad de una zona:

- En la figura 6.7, el haz de láser detecta una rampa ascendente frente al robot, pero es incapaz de identificarla como transitable. La segmentación de obstáculos, en cambio, la reconoce como suficientemente horizontal.

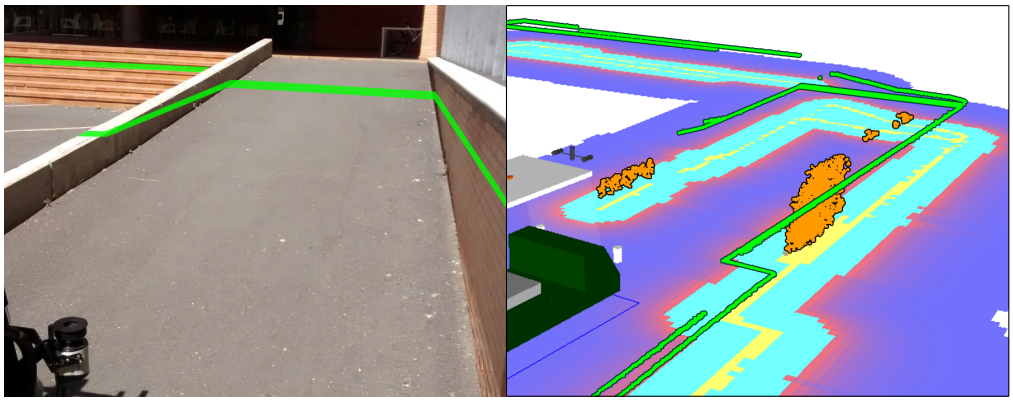


Figura 6.7. Comparación de las contribuciones de la telemetría (en verde) y del tiempo de vuelo (en naranja) al mapa de costes, cuando el robot observa una rampa.

- En la figura 6.8, el plano de telemetría impacta con unas escaleras a la altura del quinto escalón. El robot por tanto ignora un posible riesgo de colisión con los cuatro escalones inferiores, que son visibles al tiempo de vuelo y a la estereovisión.

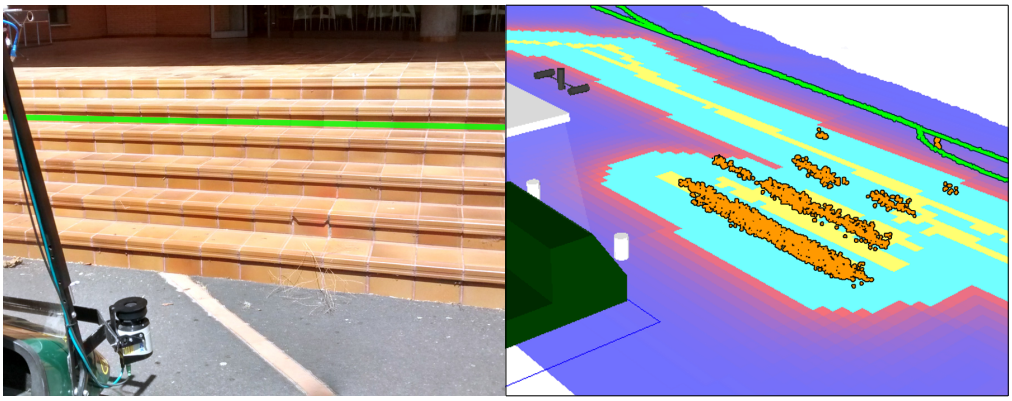


Figura 6.8. Comparación de las contribuciones de la telemetría (en verde) y del tiempo de vuelo (en naranja) al mapa de costes, cuando el robot observa una escaleras.

6.2.1. Resultados experimentales

Para comparar el resultado de aplicar una segmentación de obstáculos tanto al tiempo de vuelo como a la estereovisión, las imágenes devueltas por el par estereovisor de *PS4* son procesadas y transformadas en nubes de puntos mediante dos algoritmos diferentes: SBM (*Stereo Block Matching*, Konolige, 1998) y ELAS (*Efficient Large-Scale Stereo Matching*, Geiger et al., 2011).

En los experimentos realizados, el robot se hace circular por un entorno complejo en el que encuentra diferentes tipos de estructuras, tales como rampas o bordillos, mientras el algoritmo presentado le informa de los obstáculos detectados, a través del mapa de costes.

Para asegurar la fiabilidad del método presentado, para cada fuente de datos tridimensionales se evaluó la estabilidad de los mapas de costes producidos. La figura 6.9 muestra la variación media del valor de navegabilidad de las celdas para un cierto intervalo de tiempo, en un entorno estático. Este intervalo, de aproximadamente 5 segundos, corresponde al tiempo transcurrido hasta la estabilización del mapa de costes – es decir, el momento en el que todo objeto detectado forma parte del conjunto de observaciones integradas.

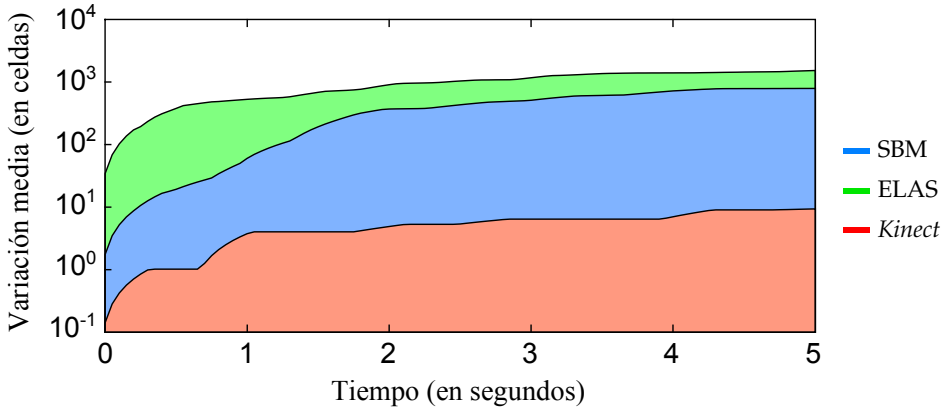


Figura 6.9. Variación media acumulada del mapa de costes.

La variación introducida por el sensor *Kinect* es, en media, varios órdenes de magnitud inferior a la de la estereovisión, independientemente del algoritmo utilizado. Esto se debe a que el par de cámaras de *PS4* provee información de un mayor número de elementos visuales que el sensor de tiempo de vuelo, pero los algoritmos que se encargan de su procesamiento no son suficientemente precisos. Esto provoca inconsistencias en la forma de los objetos y permite la integración de detecciones falsas en el mapa de costes.

En cambio, como se puede ver los ejemplos de la figura 6.10, todos los mapas producidos a partir del dispositivo *Kinect* representan los obstáculos físicos de forma más limpia y apropiada para su uso durante la etapa de navegación.

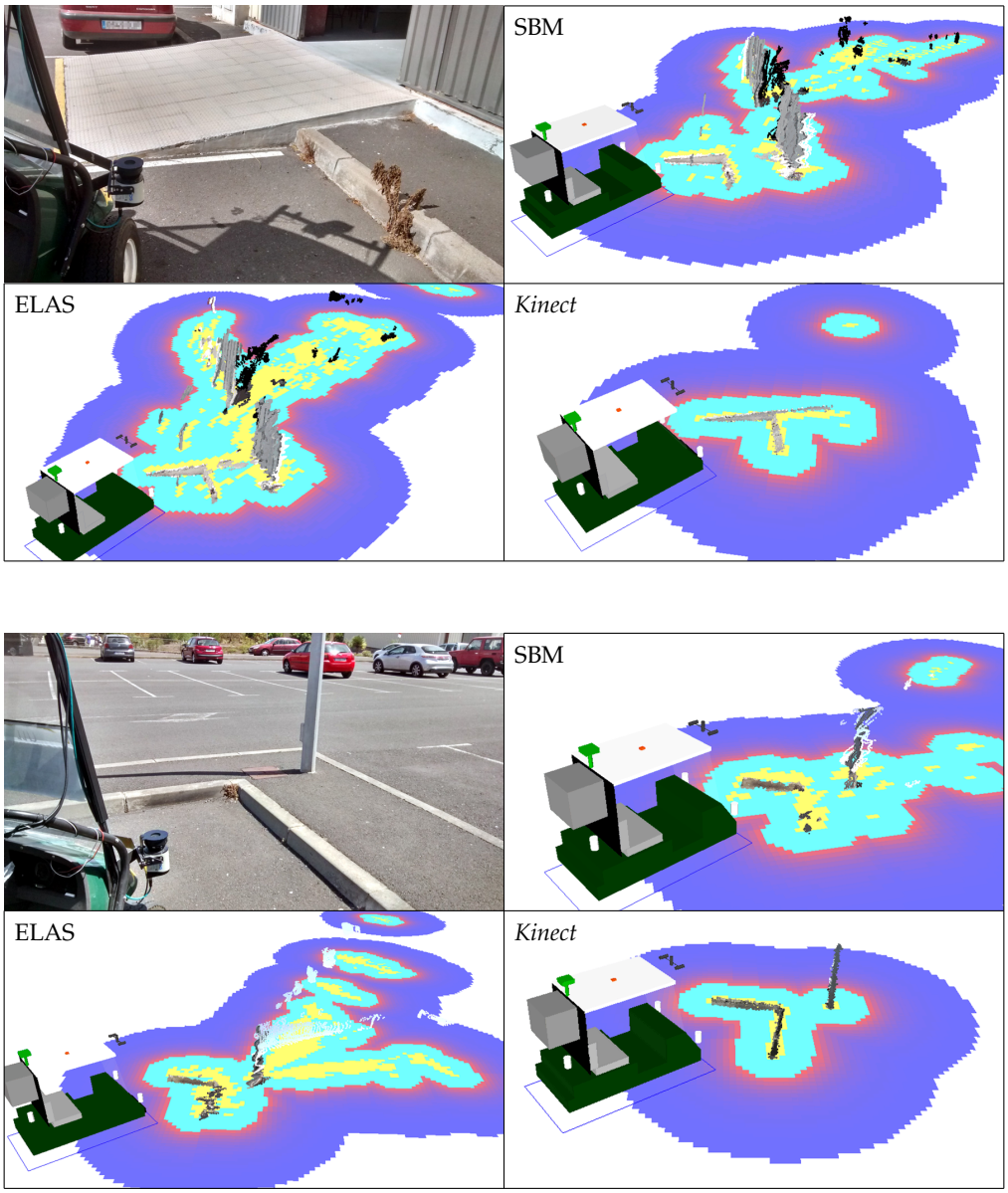
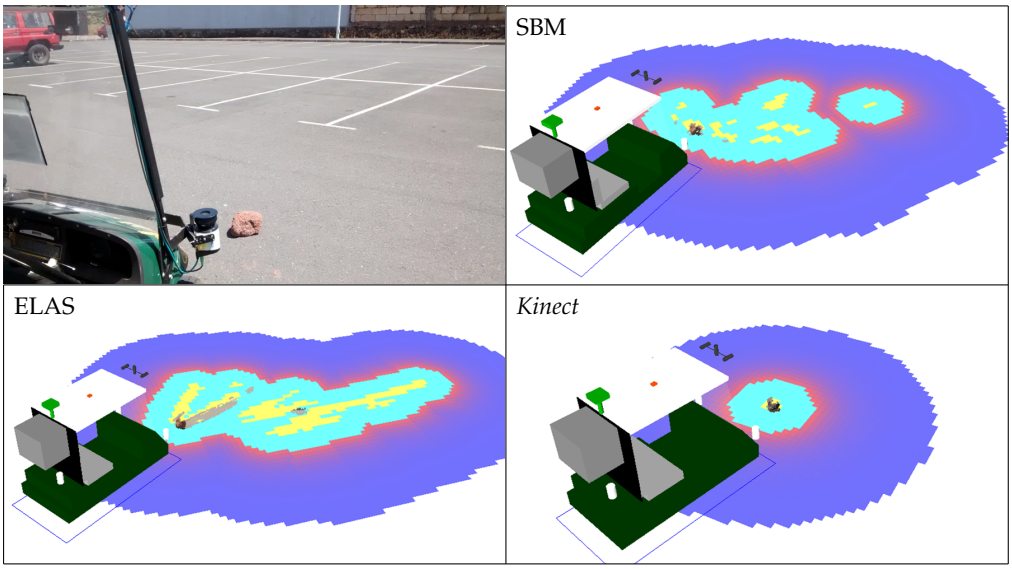


Figura 6.10. Comparación de las contribuciones de la estereovisión y del tiempo de vuelo al mapa de costes, ante una rampa (bloque superior), un bordillo (bloque inferior) y una zona despejada (siguiente página).



Sin embargo, se observó un pequeño defecto en la capacidad de detección del sensor de tiempo de vuelo en exteriores. Aunque su posición en el vehículo lo protege de la luz solar directa, la radiación infrarroja redirigida desde objetos reflectantes en su entorno puede producir medidas incorrectas aisladas, visibles como obstáculos flotantes.

Debido a su inverosimilitud física, estos fallos podrían ser detectados y eliminados fácilmente; sin embargo, dado que estas medidas falsas se derivan de obstáculos reales y aparecen a menor distancia de a la que realmente están, aceptarlas no supone un riesgo directo para el robot ni para su entorno.

Capítulo

7

Integración de GPS en localización y generación de mapas

La autolocalización es uno de los problemas más importantes relativos a la robótica móvil, especialmente en exteriores y áreas urbanas, donde la existencia de obstáculos dinámicos puede incrementar la incertidumbre de los algoritmos de localización. El conocimiento previo del entorno es por tanto crítico a la hora de planear trayectorias y ejecutar maniobras complejas sin sufrir colisiones.

Por otra parte, si el área a recorrer no ha sido explorada con anterioridad o está sujeta a cambios imprevistos, la generación automática de mapas puede ser necesaria. Habitualmente, los algoritmos de *localización y mapeo simultáneos* (SLAM) hacen uso de *filtros de partículas* para optimizar sus resultados.

Los filtros de partículas son una herramienta que permite estimar el estado de un sistema variable en el tiempo, como puede ser un robot, y se definen como conjuntos de *muestras* o *partículas* asociadas a *pesos*, tales que aquellas muestras que sean más coherentes con la realidad física percibida tendrán un mayor peso. La partícula con mayor probabilidad de corresponder al estado real del sistema constituye la *hipótesis* del filtro de partículas.

Se llama *remuestreo* al proceso por el cual se descartan aquellas muestras de menor verosimilitud para mejorar la calidad global del filtro; generalmente, el tamaño total del conjunto, que puede ser constante o variable, se compensa añadiendo nuevas partículas o replicando las de mayor peso.

Para evitar la necesidad de una inicialización manual de la posición del vehículo para cada ejecución, los mapas a utilizar precisan de referencias estáticas, dadas como una orientación espacial y la localización geográfica de un punto clave. Estas se obtienen a partir de sistemas de posicionamiento global (GPS) combinados con unidades de medida inercial (IMU).

En este capítulo se presenta un mecanismo de mejora de las estimaciones de posición para filtros de partículas de tamaño variable, mediante la integración de medidas de GPS. Para ello, la información de odometría se pre-corrige en secuencias cortas mediante comparaciones de barridos de telemetría (*scan-matching*), lo que produce estimaciones localmente consistentes de la posición del robot, y seguidamente se optimiza utilizando GPS. El sistema implementado sirvió de base práctica para la publicación Perea et al. (2013).

7.1. Descripción del algoritmo

El método de combinación utilizado es un ejemplo de *fusión de datos*, ya que se realiza sobre la salida de los sensores, en lugar de sobre las decisiones de procedimientos independientes (Kokar et al., 2004).

Llamando $A \subset \mathbb{R}^2$ al área de trabajo navegable, el conjunto de todas las posibles *poses* (posiciones y orientaciones) que puede adquirir el robot se denotará por $\Omega = \{\langle x, y, \omega \rangle\}$, con $\langle x, y \rangle \in A$ y $-\pi < \omega \leq \pi$. La notación $R = \langle p_R, m_R \rangle$ representa el estado real del robot, donde $p_R \in \Omega$ es su pose georreferenciada y $m_R \in \{0,1\}^{|A|}$ es el mapa binario de ocupación: las posiciones de A ocupadas por un obstáculo se marcarán con 1 y las libres con 0.

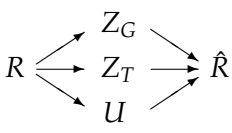


Figura 7.1. Arquitectura de fusión de datos.

Partiendo de esta nomenclatura, se describen tres estados relativos al sistema sensorial: U corresponde a la odometría, Z_T a la telemetría y Z_G al posicionamiento global. El estado de fusión $\hat{R}$ representa una estimación del estado R, obtenida a partir de U, Z_T y Z_G (Fig. 7.1).

7.1.1. Medidas odométricas

El estado del robot, en función de las medidas devueltas por su sistema de odometría, se define como $U = \langle u,\ f_U : \Omega^2 \to \Omega \rangle$, donde $u \in \Omega$ es la variación en posición y orientación, y la función f_U calcula una pose absoluta a partir de la última pose conocida y el movimiento relativo observado.

El modelo de proyección odométrica utilizado (Thrun et al., 2005) se muestra en el algoritmo 7.1, donde los valores α están relacionados con la covarianza del sensor odométrico del robot, y la función `muestra` (σ) devuelve un valor aleatorio sujeto a una distribución normal de media 0 y desviación típica σ.

Algoritmo 7.1. Proyección odométrica f_U.

Entrada: $p = \langle x, y, \omega \rangle \in \Omega \mid u = \langle \dot{x}, \dot{y}, \dot{\omega} \rangle \in \Omega$

$$\delta_{R1} \leftarrow \tan^{-1}(\dot{y}/\dot{x})$$
$$\delta_{R2} \leftarrow \dot{\omega} - \delta_{R1}$$
$$\delta_T \leftarrow \sqrt{\dot{x}^2 + \dot{y}^2} - \omega$$

$$\delta'_{R1} \leftarrow \delta_{R1} - \texttt{muestra}\left(\alpha_1 \cdot \delta_{R1}^2 + \alpha_2 \cdot \delta_T^2\right)$$
$$\delta'_{R2} \leftarrow \delta_{R2} - \texttt{muestra}\left(\alpha_1 \cdot \delta_{R2}^2 + \alpha_2 \cdot \delta_T^2\right)$$
$$\delta'_T \leftarrow \delta_T - \texttt{muestra}\left(\alpha_3 \cdot \delta_T^2 + \alpha_4 \cdot \delta_{R1}^2 + \alpha_4 \cdot \delta_{R2}^2\right)$$

$$x' = x + \delta'_T \cdot \cos(\omega)$$
$$y' = y + \delta'_T \cdot \sin(\omega)$$
$$\omega' = \omega + \delta'_{R1} + \delta'_{R2}$$

devolver $\langle x', y', \omega' \rangle \in \Omega$

Los valores α obtenidos empíricamente para los experimentos realizados en este capítulo son: $\alpha_1 = 0{,}2$; $\alpha_2 = 0{,}05$; $\alpha_3 = 0{,}1$ y $\alpha_4 = 10^{-3}$.

7.1.2. Medidas telemétricas

Una medida devuelta por un sistema de telemetría láser viene dada como un conjunto de obstáculos visibles, detectados mediante la reflexión de haces de luz infrarroja. Puesto que el ángulo de proyección θ de cada haz es conocido, y la distancia de impacto d puede calcularse a partir de la velocidad de la luz y el tiempo transcurrido entre la emisión y la recepción, el conjunto que define las medidas de telemetría de un sensor de N haces se denota por $\Lambda = \left\{ \langle \theta, d \rangle^N \right\}$. Tanto la resolución como los valores máximos y mínimos de θ y d vienen definidos por las características del dispositivo sensorial.

Denotando por $M = \left\{ (0,1)^{|A|} \right\}$ al conjunto de los posibles mapas de ocupación de A, el estado del robot según su sistema de telemetría viene dado por $Z_T = \langle z_T,\ m_T,\ h_T : \Omega \times \Lambda \to M,\ f_T : \Omega \times M \times \Lambda \to (0,1) \rangle$, donde $z_T \in \Lambda$ es la medida telemétrica más reciente y $m_T \in M$ es un submapa de ocupación.

La función h_T genera el submapa m_T a partir de la medida z_T y una pose $\langle x, y, \omega \rangle \in \Omega$ del robot. Para ello, llamando $m_T(a)$ al nivel de ocupación de m_T asociado a la posición $a \in A$, para cada haz $\langle \theta, d \rangle \in z_T$ se modifica el valor de $m_T\left(\langle x + d \cdot \cos(\omega + \theta), y + d \cdot \sin(\omega + \theta) \rangle\right)$ como corresponda, teniendo en cuenta tanto la resolución espacial de A como la precisión de las medidas del sensor telemétrico.

Aunque idealmente m_T debería coincidir exactamente con un fragmento del mapa real m_R, las limitaciones del sistema sensorial no permiten una certeza binaria de los datos calculados. Por tanto, el conjunto M abarca los mapas de ocupación que asocian, a cada posición de A, una probabilidad entre 0 y 1 de contener un obstáculo.

La función f_T calcula la verosimilitud de observar la medida telemétrica z_T desde cierta pose $p \in \Omega$, para un mapa determinado $m \in M$ calculado previamente, en función de su coherencia con los obstáculos visibles. Para ello, para cada haz láser de z_T se calculan dos posiciones: a_1, donde se produjo su reflexión, y a_0, el espacio libre más cercano.

Como muestra el algoritmo 7.2, el cálculo de a_0 es trivial: si se puede observar un obstáculo en la posición a_1, entonces necesariamente la posición inmediatamente anterior a_0 tiene que estar libre. Si los niveles de ocupación de estas posiciones en el mapa m, para un cierto valor umbral τ, no son coherentes con lo esperado, la medida z_T se asume errónea. El valor τ debe escogerse manualmente en función de la precisión del dispositivo de telemetría utilizado; en los experimentos presentados en este capítulo, se usó $\tau = 0{,}1$.

Algoritmo 7.2. Verosimilitud telemétrica f_T.

Entrada: $p = \langle x, y, \omega \rangle \in \Omega \mid m \in M \mid z_T \in \Lambda$
τ : umbral de ocupación
σ_T : desviación típica del error del dispositivo

$w_T \leftarrow 0$
para todo $\langle \theta, d \rangle \in z_T$ **hacer**
  $a_1 \leftarrow \langle x + d \cdot \cos(\omega + \theta),\ y + d \cdot \sin(\omega + \theta) \rangle$
  $a_0 \leftarrow \langle x + (d - \varepsilon)\cos(\omega + \theta),\ y + (d - \varepsilon)\sin(\omega + \theta) \rangle$
  si $m(a_0) < \tau < m(a_1)$ **entonces**
    $w_T \leftarrow w_T + \exp\left(-m(a_1)^2 / \sigma_T^2\right)$
devolver w_T

7.1.3. Medida de posicionamiento global

Una medida de un GPS viene dada, en general, como una pose aproximada $\mu \in \Omega$ y una matriz de covarianza Σ que define su error correspondiente; denotaremos al conjunto de datos este tipo como $\Gamma = \{\langle \mu, \Sigma \rangle\}$. El estado del robot de acuerdo a las medidas de su sensor de posición global se define entonces como $Z_G = \langle z_G,\ f_G : \Omega \times \Gamma \to \mathbb{R} \rangle$, con $z_G \in \Gamma$.

La función f_G calcula la verosimilitud de la medida de GPS $\langle \mu, \Sigma \rangle$, sujeta a una pose conocida $\langle x, y, \omega \rangle \in \Omega$. Sabiendo que tanto la posición como la orientación se ajustan a una función de probabilidad gaussiana, se obtiene:

$$f_G(\langle x, y, \omega \rangle, \langle \mu, \Sigma \rangle) = \frac{\sum_{k=-\infty}^{\infty} \exp\left(-\frac{1}{2} \begin{bmatrix} x - \mu_x \\ y - \mu_y \\ \omega - \mu_\omega + 2k\pi \end{bmatrix}^T \Sigma^{-1} \begin{bmatrix} x - \mu_x \\ y - \mu_y \\ \omega - \mu_\omega + 2k\pi \end{bmatrix} \right)}{\sqrt{(2\pi)^3 |\Sigma|}} \tag{7.1}$$

7.1.4. Fusión sensorial

El estado final del robot, definido como la fusión de los tres sensores, agrupa los datos y funciones de los estados U, Z_T y Z_G e incluye nuevos atributos y procedimientos, que describen el mecanismo de combinación a emplear a la hora de aproximar los valores del estado real R.

Para el caso que nos ocupa, llamaremos partícula a una tupla $\langle p, w, m \rangle$, tal que la pose $p \in \Omega$ situada en un submapa local $m \in M$ lleva asociado un peso $w \in \mathbb{R}$, proporcional a su probabilidad de corresponder a la pose real del robot p_R. Denotaremos por Φ al conjunto de los filtros de partículas de tamaño variable que contienen muestras de este tipo (Doucet et al., 2000; Murphy, 1999).

Partiendo de esta nomenclatura, el estado de fusión sensorial viene dado por:

$$\hat{R} = \Big\langle \overbrace{u_T,\ f_U}^{U},\ \overbrace{z_T,\ m_T,\ h_T,\ f_T}^{Z_T},\ \overbrace{z_G,\ f_G}^{Z_G},\ \hat{m}_R,\ f_R : M^n \to M,$$
$$D_R : f_U \times f_T \times f_G \to (\Phi \times \Omega \times M \times \Lambda \times \Gamma \to \Phi) \Big\rangle$$

donde $\hat{m}_R \in M$ es el mapa global acumulado de los obstáculos que se han observado durante la exploración del terreno; la función f_R calcula dicho mapa a partir de los submapas de ocupación generados en etapas anteriores de la ejecución, como se explicará en la subsección 7.1.5; y la función objetivo D_R, descrita

en el algoritmo 7.3, define cómo combinar los datos recogidos por los sensores de odometría, telemetría y posicionamiento global, para calcular el valor de verosimilitud asociado a cada pose y submapa local.

En cada paso de la ejecución, cada partícula del filtro se desplaza de acuerdo a la información odométrica recibida, se optimiza, mediante pequeñas variaciones de su pose, basándose en la información de telemetría y de posicionamiento global, y se recalcula su peso y su mapa local. Seguidamente, si se detecta que el filtro de partículas es poco eficiente, se realiza un remuestreo.

Algoritmo 7.3. Algoritmo de fusión D_R

Entrada: $F \in \Phi \mid u \in \Omega \mid z_T \in \Lambda \mid z_G \in \Gamma$
ρ : factor de remuestreo

$F' \leftarrow \emptyset$
para todo $\langle p, w, m \rangle \in F$ **hacer**
  $p' \leftarrow$ pose próxima a $f_U(p, u)$ que maximice $f_T(p', m, z_T) \times f_G(p', z_G)$
  $w' \leftarrow f_T(p', m, z_T) \times f_G(p', z_G)$
  $m' \leftarrow h_T(p', z_T)$
  $F' \leftarrow F' \cup \{\langle p', w', m' \rangle\}$
si $N_{\text{eff}}(F') < |F'| \times \rho$ **entonces**
  $\bar{w} \leftarrow \dfrac{1}{|F'|} \displaystyle\sum_{\langle p', w', m' \rangle \in F'} w'$
  Sea $\langle p'_i, w'_i, m'_i \rangle$ la $i-$ésima partícula de F'
  $\dot{w} \leftarrow$ número aleatorio entre 0 y $\bar{w}$
  $\dot{F} \leftarrow \emptyset$
  para i desde 1 hasta $|F'|$ **hacer**
    mientras $\displaystyle\sum_{j=1}^{i} w'_j > \dot{w}$ **hacer**
      $\dot{F} \leftarrow \dot{F} \cup \{\langle p'_i, w'_i, m'_i \rangle\}$
      $\dot{w} \leftarrow \dot{w} + \bar{w}$
  $F' \leftarrow \dot{F}$
} Remuestreo
devolver F'

La eficiencia del algoritmo de fusión depende en gran medida del número de partículas empleado y de la frecuencia de remuestreo. Ambos vienen determinados, siguiendo el método propuesto por Liu (1996), por el *número efectivo de muestras* del filtro, calculado como:

$$N_{\text{eff}}(F) = \frac{1}{\sum_{i=1}^{|F|} \tilde{w}_i^2} \tag{7.2}$$

donde $|F|$ es el número de partículas de $F \in \Phi$, y $\tilde{w}_i$ es el peso normalizado de su $i-$ésima muestra (Doucet et al., 2001). En el algoritmo presentado, cuando el porcentaje de muestras útiles es menor que un cierto umbral ρ, se realiza un remuestreo.

7.1.5. Generación de submapas

El proceso de remuestreo selecciona las partículas de mayor peso de un filtro, de forma cuasi-aleatoria, y las replica sustituyendo a las de menor verosimilitud. Esto implica que, para cualquier instante t de la ejecución, toda muestra disponible ϕ_t es una copia modificada de otra anterior, a la que se denotará por ϕ_{t-1}, y pertenece a una secuencia de copias que comienza al inicio de la exploración (Fig. 7.2). Obsérvese que, debido al mecanismo de remuestreo, dos o más partículas pueden compartir a sus predecesoras.

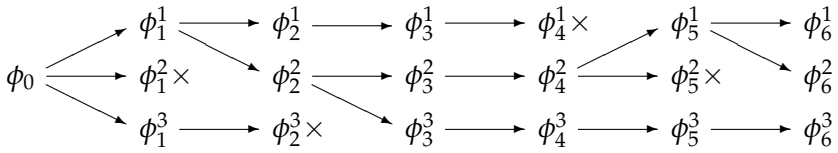


Figura 7.2. Secuencia de replicación y descarte de partículas.

Teniendo esto en cuenta, el mapa completo del espacio de trabajo puede calcularse, en cualquier momento de la ejecución, como la combinación de los submapas de la hipótesis del filtro de partículas y de todas sus predecesoras; ade-

más, como todos los submapas se almacenan georreferenciados, el resultado final también lo está. Llamando ϕ_t a la hipótesis en el instante t de la ejecución, y m_k al submapa correspondiente a la muestra ϕ_k, el mapa global acumulado de obstáculos observados puede calcularse como $f_R(m_t, m_{t-1}, \cdots, m_0) = \bigcup_{k=0}^{t} m_k$.

7.2. Experimentos

El algoritmo presentado fue probado experimentalmente utilizando el dispositivo VERDINO (apéndice A) en el aparcamiento de la Escuela Técnica Superior de Ingeniería Informática de la Universidad de La Laguna, de aproximadamente $90\,m \times 90\,m$, en el que se detectaron coches aparcados y en movimiento, así como peatones (Fig. 7.4); el robot generó un mapa de la zona usando tanto un filtro de partículas básico como la versión mejorada mediante la integración de un sistema de posicionamiento global, presentada en este capítulo (Fig. 7.5).

Durante las pruebas se detectaron ciertas dificultades concernientes a la navegación autónoma en exteriores. En primer lugar, los dispositivos de posicionamiento global sufren de un tipo de interferencia conocido como *multipath*, por la cual el receptor puede detectar la señal de los satélites emisores tras haberse reflejado en una superficie. Esto provoca que las medidas de distancia obtenidas sean erróneas y el cálculo de la posición del vehículo produzca resultados distintos a la realidad.

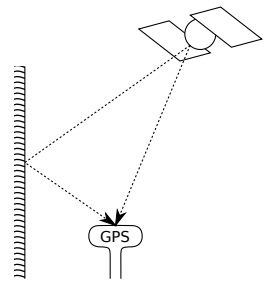


Figura 7.3. Ejemplo de multipath.

En el entorno donde se realizaron los experimentos existen además elementos que no se pueden detectar de forma precisa mediante un telémetro láser, tales como coches y vallas metálicas, por no presentar superficies verticales planas y continuas. El sistema de odometría también produjo errores que redujeron la precisión del mapeado, al sufrir leves deslizamientos al tomar las curvas.

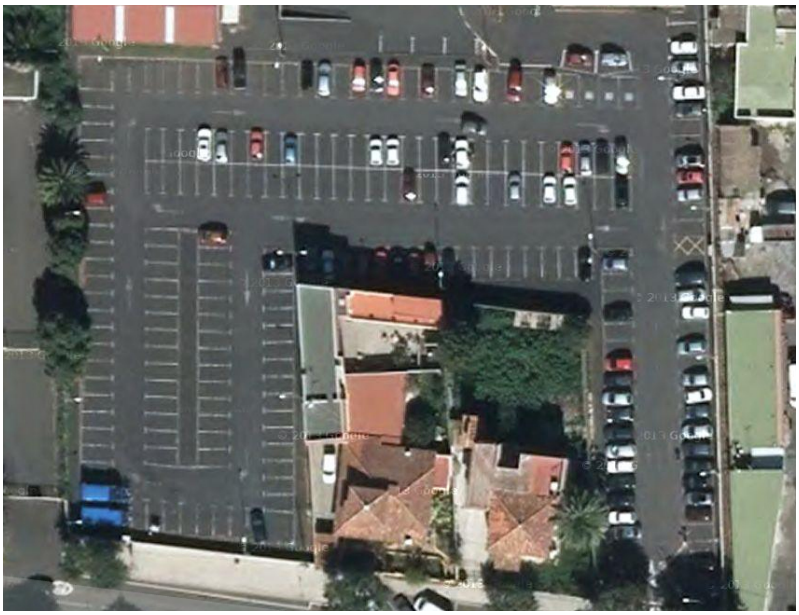


Figura 7.4. Mapa real del área de trabajo.

El resultado final, sin embargo, es visiblemente más preciso al integrar las medidas de posicionamiento global en el cálculo del peso de las muestras, gracias a que dicho sensor permite corregir los errores derivados de la telemetría, y viceversa. La figura 7.6 muestra un caso claro en el que, al no encontrar suficientes obstáculos estáticos próximos con los que localizarse, el robot pierde su ubicación si no dispone de un sistema de posicionamiento auxiliar en el que basarse. Por otra parte, los efectos negativos del *multipath* quedan también resueltos debido a la influencia de las medidas de telemetría (Fig. 7.7).

Gracias a esto, la hipótesis de localización del robot se mantiene estable y similar a las medidas de posicionamiento global durante toda la ejecución, exceptuando aquellas zonas en las que este sensor sufre interferencias, en cuyo caso el vehículo se guía utilizando información telemétrica.

En conclusión, mediante la integración de medidas de posicionamiento global en un algoritmo de localización y mapeo simultáneos, se observó una importante

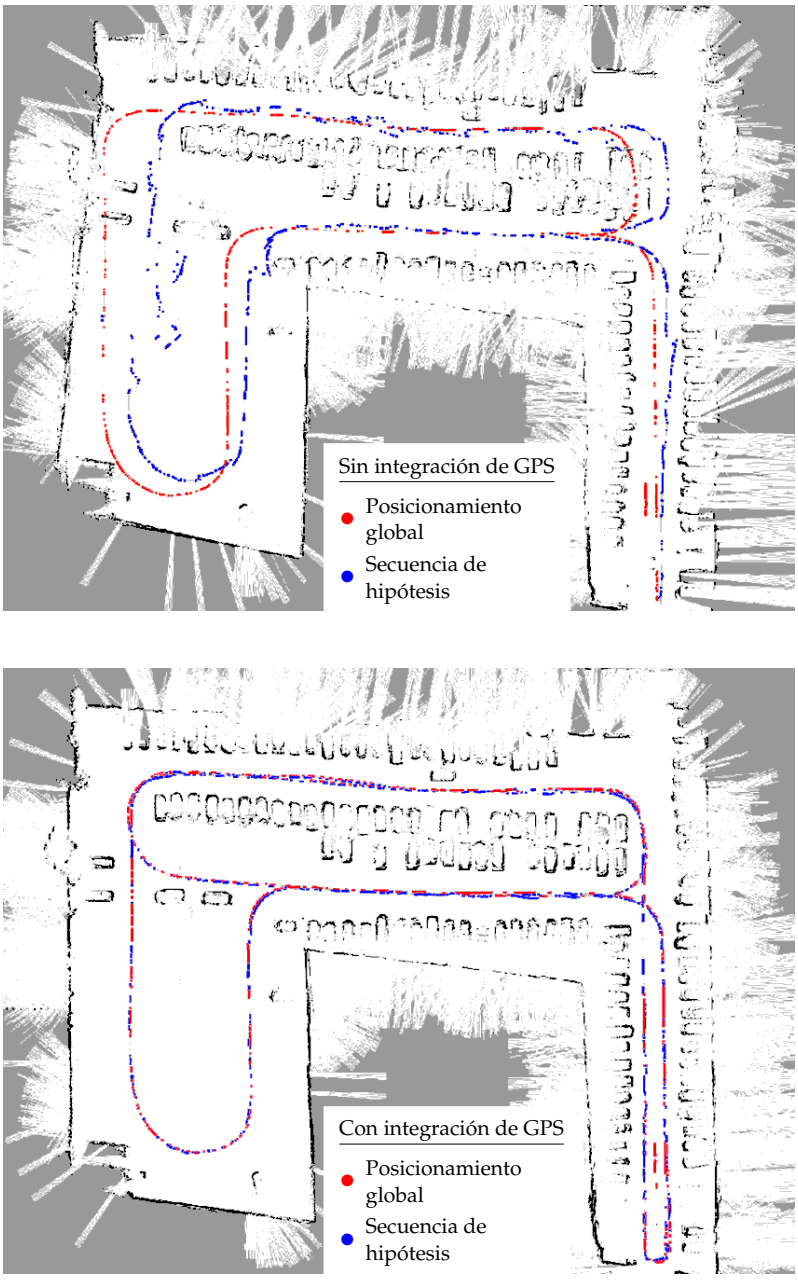


Figura 7.5. Mapas generados mediante SLAM

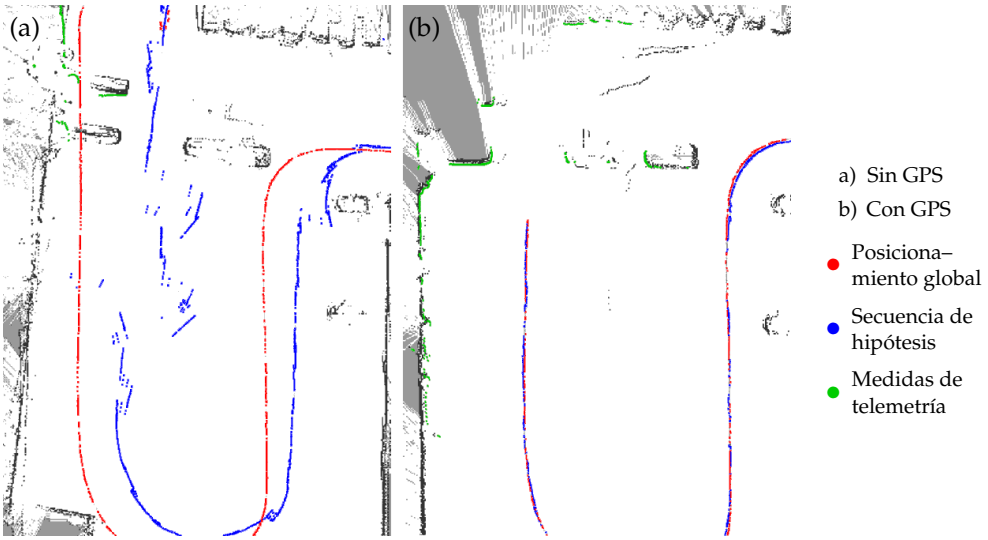


Figura 7.6. Corrección del error odométrico mediante GPS.

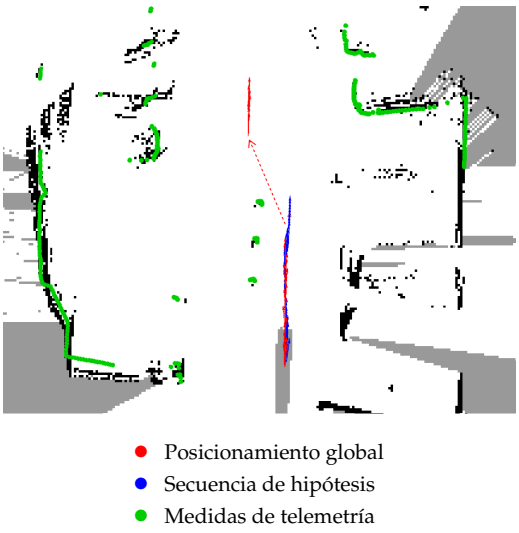


Figura 7.7. Corrección de multipath mediante telemetría.

reducción del error medio en la localización del robot a lo largo de su trayectoria, incluso disminuyendo drásticamente el número de partículas utilizado. La tabla 7.1 muestra los datos resultantes de uno de los experimentos realizados.

Tabla 7.1. Valor medio y desviación típica del error en las ejecuciones.

	Sin GPS	Con GPS
Nº partíc.	50	5
μ (metros)	3,7939	0,2125
σ (metros)	2,2861	0,6669

Conclusiones

En el transcurso del presente trabajo de tesis, se han abordado los temas de la fusión sensorial y procesamiento bayesiano, como mecanismos de mejora de la capacidad de detección de peatones por parte de un vehículo autónomo. También se han desarrollado mecanismos para predecir las posiciones futuras de los obstáculos dinámicos y métodos de generación automática de mapas georreferenciados, a fin de mejorar la seguridad de la navegación del dispositivo robótico.

En una primera etapa, se estudiaron dos formas diferentes de hacer variar la configuración de un par estereovisor, mediante alteraciones en su extensión y en su orientación, así como su fusión la fusión de un par fijo con un sistema de telemetría, y se analizó qué parámetros resultaban óptimos, a la hora de calcular de forma precisa las posiciones de los puntos observados. En esta investigación se definieron ecuaciones que permiten evaluar eficientemente las soluciones del problema de la correspondencia, y se demostró qué parámetros producen los mejores resultados.

Considerando cámaras estenopeicas ideales, un par estereovisor de longitud variable es más capaz de discernir correctamente la correspondencia entre puntos, cuanto menor sea la separación entre las cámaras. Un par giratorio tiene una mayor dificultad a la hora de ser configurado, dado que su ángulo óptimo de orientación depende directamente con la posición de los puntos a analizar.

La efectividad de un sistema de fusión entre estereovisión y telemetría fue evaluada y comparada con la de una disposición trinocular, y los resultados experimentales demostraron que el primero conlleva un aumento de precisión para rangos de distancia amplios.

A continuación se desarrollaron dos mecanismos de procesamiento bayesiano, tanto para el método de detección de peatones por Viola-Jones como para el basado en histogramas de gradientes orientados (HOG), y ambos fueron aplicados a un caso real, con obstáculos dinámicos a localizar y evitar por parte de un dispositivo autoguiado. Nuestros métodos describen una modificación estadística de la salida de las herramientas originales, que se combinan con dos formas diferentes de convolución aproximada de matrices bidimensionales de probabilidad con múltiples máximos locales.

Se demostró que estos algoritmos mejoran la precisión de los resultados, al restringir las matrices de detección devuelta por los métodos originales a las áreas dónde la aparición de objetos de interés es más probable, de acuerdo a los movimientos observados previamente.

A continuación, se realizó una mejora sobre un sistema de estados de colisión probable, mediante el cómputo de trayectorias dependientes del tiempo. Para ello se realizó una observación de las posiciones consecutivas de los obstáculos presentes en el área de trabajo, durante un cierto intervalo de tiempo, y se calculó sus localizaciones futuras en forma de regiones de probabilidad.

La trayectoria a seguir por parte del vehículo autodirigido se calculó utilizando una variante del algoritmo de caminos mínimos de Dijkstra, cuyos pesos varían en función del instante de tiempo en que se alcanza cada celda. Gracias a este mecanismo, el algoritmo original, restringido a maniobras de frenado, puedo ampliarse para seguir rutas complejas a través de entornos poblados, asegurando en todo momento una probabilidad mínima de colisión.

Por último, se propusieron métodos de integración de medidas de sistemas sensoriales, tanto de tiempo de vuelo como de posicionamiento global, en el mecanismo de navegación y generación automática de mapas de un dispositivo robótico, como forma de pesado de las muestras del filtro de partículas de

localización, y como técnica de georreferenciación de los mapas producidos.

Se observó que esta aportación permite una mejora en los cierres de bucle del sistema de mapeo, así como una reducción en el número y dispersión de las partículas a utilizar, lo que a su vez disminuye los requisitos de cómputo y aumenta la frecuencia de remuestreo. Esto produce mapas mucho más precisos y, por ser georreferenciados, fácilmente reutilizables.

Las herramientas diseñadas fueron implementadas e integradas en el sistema de control del dispositivo VERDINO, y se encuentran actualmente disponibles para su uso en sus actividades de exploración y navegación autónoma. Pero mi investigación en este ámbito está lejos de haber concluido. Existen multitud de técnicas de detección visual y de generación de mapas de obstáculos que pueden aplicarse a nuestro robot y para las cuales pueden desarrollarse mejoras.

Apéndice

A

Prototipo VERDINO

El prototipo VERDINO es un carro de golf EZ-GO TXT-2 modificado, un vehículo de dos plazas completamente eléctrico, equipado con sistemas computerizados de control de giro, frenado y tracción para la navegación autónoma.

Figura A.1. Prototipo VERDINO

Su conjunto sensorial consiste en un sistema de posicionamiento global (GPS) diferencial, basado en una estación móvil anexa al dispositivo y una estación fija en tierra cuya posición es conocida con mucha precisión. Esta última se usa para

estimar el error de medida introducido por cada satélite, y enviar las correspondientes correcciones a la estación móvil.

La orientación tridimensional del prototipo viene dada por una unidad de medida inercial (IMU); esta dispone de tres acelerómetros, tres giróscopos, tres magnetómetros y un procesador que calcula sus ángulos de Euler en tiempo real, basándose en la información de estos sensores. Se dispone asimismo de un sistema de odometría como herramienta de posicionamiento en caso de fallo del GPS.

El sistema de telemetría consiste en dos modelos láser Sick LMS221-30206, que ofrecen una resolución angular de 0,5°, un rango máximo de 80 m y errores sistemáticos y estadísticos en un rango de 1 a 20 m de $\pm 35\,mm$ y de $\pm 10\,mm$ respectivamente.

El prototipo también incluye una plataforma móvil frontal que sostiene un sistema de visión, que consiste en dos cámaras convencionales y termales. Esta plataforma es capaz de ajustar la altura y la rotación de las cámaras, para su uso en curvas y terreno irregular.

Las cámaras utilizadas en estereovisión son dos modelos Santachi DSP220x, con una resolución de 320×240 píxeles, tres canales de color, una distancia focal ajustable entre 3,9 y 85,8 mm y un ángulo de visión de 47°. Su precisión máxima a una distancia de d metros es por tanto $2{,}174 \times d$ mm/píxel.

Apéndice

B

Dispositivo Pioneer

Para la realización de esta tesis se utilizó un Adept Pioneer 3-AT equipado con un ordenador de abordo modificado. Este dispositivo todoterreno y holonómico tiene una velocidad lineal máxima de 0,7 m/s y una velocidad radial máxima de 140°/s, y contiene tres baterías «*hot-swap*» de 7,2 A cada una, un microcontrolador serial, entradas digitales y analógicas, actuadores independientes, ultrasonidos frontales y lectores de odometría. Sobre la base se instaló además un telémetro láser, idéntico a los descritos en el apéndice A.

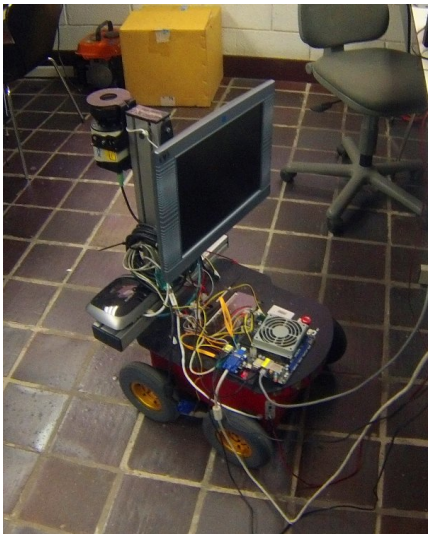


Figura B.1. Dispositivo Pioneer

El ordenador de abordo cuenta con un procesador Intel® Core™2 Quad con cuatro núcleos de 2.66 GHz, 4Gb de memoria RAM y un sistema operativo Ubuntu 12.04 de 64 bits con ROS Hydro Medusa instalado.

Bibliografía

Las siguientes referencias bibliográficas se presentan en orden alfabético por autor. Las referencias con más de un autor aparecen ordenadas en base al primero de los mismos.

D. Althoff, M. Althoff, D. Wollherr, and M. Buss. Probabilistic collision state checker for crowded environments. In *IEEE Int. Conf. Robotics and Automation*, pages 1492–1498, May 2010.

C. Bamji, P. O'Connor, T. Elkhatib, S. Mehta, B. Thompson, L. Prather, D. Snow, O. Akkaya, A. Daniel, A. Payne, T. Perry, M. Fenton, and V.-H. Chan. A 0,13 μm CMOS system-on-chip for a 512×424 time-of-flight image sensor with multi-frequency photo-demodulation up to 130 mhz and 2 gs/s ADC. *IEEE Journal of Solid-State Circuits*, 50(1):303–319, Jan. 2015.

L. Bourdev and J. Brandt. Robust object detection via soft cascade. In *IEEE Conf. Computer Vision and Pattern Recognition*, volume 2, pages 236–243, June 2005.

G. Bradski. The OpenCV library. *Dr. Dobb's Journal of Software Tools*, 2000.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conf. Computer Vision and Pattern Recognition*, volume 1, pages 886–893, June 2005.

P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *IEEE Conf. Computer Vision and Pattern Recognition*, pages 304–311, June 2009.

P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *PAMI*, 34, 2012.

A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proc. Conf. Uncertainty in Artificial Intelligence*, pages 176–183, Stanford, CA, USA, 2000.

A. Doucet, N. De Freitas, and N. Gordon, editors. *Sequential Monte Carlo methods in practice*. Springer Verlag, 2001.

L. E. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79(3):497–516, 1957.

M. Enzweiler and D. Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31(12):2179–2195, Dec. 2009.

A. Ess, B. Leibe, and L. van Gool. Depth and appearance for mobile scene analysis. In *IEEE Int. Conf. Computer Vision*, pages 1–8, Oct. 2007.

A. Ess, B. Leibe, K. Schindler, and L. van Gool. A mobile vision system for robust multi-person tracking. In *IEEE Conf. Computer Vision and Pattern Recognition*, pages 1–8, June 2008.

A. Ess, B. Leibe, K. Schindler, and L. van Gool. Moving obstacle detection in highly dynamic scenes. In *IEEE Int. Conf. Robotics and Automation*, pages 4451–4458, 2009a.

A. Ess, B. Leibe, K. Schindler, and L. van Gool. Robust multiperson tracking from a mobile platform. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31(10):1831–1846, Oct. 2009b.

R. Fisher, J. Santos-Victor, and J. Crowley. Context aware vision using image-based active recognition. Funded by the EC's Information Society Technology's programme project IST2001-37540. URL `http://homepages.inf.ed.ac.uk/rbf/CAVIAR/`.

T. Fraichard and H. Asama. Inevitable collision states. a step towards safer robots? In *IEEE/RSJ Int. Conf. Intell. Robots and Systems*, volume 1, pages 388–393, Oct. 2003.

Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In P. Vitányi, editor, *Computational Learning Theory*, volume 904 of *Lecture Notes in Computer Science*, pages 23–37. Springer Berlin / Heidelberg, 1995.

A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Asian Conf. Computer Vision*, pages 25–38. Springer, 2011.

R. Halterman and M. Bruch. Velodyne HDL-64E lidar for unmanned surface vehicle obstacle detection. In *Unmanned Systems Technology XII*, volume 7692, pages 0D–8, 2010.

J. Hernández-Aceituno, L. Acosta, and R. Arnay. Fusion of a variable baseline system and a range finder. *Sensors*, 12(1):278–296, 2011. URL `http://www.mdpi.com/1424-8220/12/1/278`.

J. Hernández-Aceituno, L. Acosta, and J. D. Piñeiro. Application of time dependent probabilistic collision state checkers in highly dynamic environments. *PloS ONE*, 10(3):e0119930, Mar. 2015a. URL `http://dx.doi.org/10.1371%2Fjournal.pone.0119930`.

J. Hernández-Aceituno, L. Acosta, and J. D. Piñeiro. Pedestrian detection in crowded environments through bayesian prediction of sequential probability matrices. *Journal of Sensors*, 2015b.

G. J. Iddan and G. Yahav. Three-dimensional imaging in the studio and elsewhere. *SPIE*, 4298:48–55, 2001.

S. Koenig and M. Likhachev. Fast replanning for navigation in unknown terrain. *IEEE Trans. Robotics*, 21(3):354–363, June 2005.

M. M. Kokar, J. A. Tomasik, and J. Weyman. Formalizing classes of information fusion systems. *Information Fusion*, 5(3):189–202, 2004.

K. Konolige. Small vision systems: Hardware and implementation. In Y. Shirai and S. Hirose, editors, *Robotics Research*, pages 203–212. Springer London, 1998.

N. Lazaros, G. C. Sirakoulis, and A. Gasteratos. Review of stereo vision algorithms: From software to hardware. *International Journal of Optomechatronics*, 2 (4):435–462, 2008.

J. S. Liu. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6(2):113–119, June 1996.

K. Murphy. Bayesian map learning in dynamic environments. In *Proc. Conf. Neural Information Processing Systems*, pages 1015–1021, Denver, CO, USA, 1999. MIT Press.

A. S. Ogale and Y. Aloimonos. Shape and the stereo correspondence problem. *International Journal of Computer Vision*, 65(3):147–162, 2005.

D. Perea, J. Hernandez-Aceituno, A. Morell, J. Toledo, A. Hamilton, and L. Acosta. MCL with sensor fusion based on a weighting mechanism versus a particle generation approach. In *16th IEEE Int. Conf. Intelligent Transportation Systems*, pages 166–171, Oct. 2013.

R. Rusu and S. Cousins. 3D is here: Point cloud library (PCL). In *IEEE Int. Conf. Robotics and Automation*, pages 1–4, May 2011.

R. B. Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Departamento de Informática, Technische Universität München, Alemania, Oct. 2009.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents. The MIT Press, 2005.

P. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57:137–154, 2004.