

## Trabajo Fin de Máster

Guiado de personas ciegas mediante un  
simulador de visión protésica en entornos  
virtuales

---

Visual impaired guidance with Simulated  
Prosthetic Vision in virtual environments

Autor/es

Lorenzo Montano Oliván

Director/es

Alejandro Pérez Yus  
Jesús Bermúdez Cameo

ESCUELA DE INGENIERÍA Y ARQUITECTURA 2019



# Guiado de ciegos mediante un simulador de visión Protésica en entornos virtuales

## RESUMEN

La información del entorno es percibida por las personas a través de los sentidos. La vista es el sentido que proporciona una mayor riqueza de información acerca de la escena y de la ubicación de los objetos en la misma. Gracias a ella somos capaces de percibir y reconocer diferentes objetos y de navegar con facilidad por el entorno que nos rodea. Un gran número de personas se ven privadas de este sentido debido a las diferentes patologías existentes que degeneran gradualmente las vías visuales e incluso pueden conducir a la ceguera. Dependiendo de la gravedad de estas lesiones, es posible colocar una prótesis en alguna parte del sistema visual que permite percibir mediante estimulación eléctrica unos puntos de luz blanca conocidos como fosfenos. Con ellos se puede representar, por ejemplo, una información simplificada de la escena que ayude al sujeto a desplazarse sin colisionar.

Estas prótesis representan la información captada por una cámara, procesada con técnicas de visión artificial, para llevar a cabo la estimulación de los fosfenos. Sin embargo, el rango dinámico y la resolución visual proporcionados por estos implantes visuales es muy reducido en comparación con la visión humana por lo que es conveniente seleccionar la información que se representa. Por ello es importante desarrollar algoritmos que permitan extraer de la escena la información más relevante, con el fin de facilitar la navegación de las personas con discapacidad visual. Estos algoritmos pueden facilitar la percepción en general (representación icónica, información simplificada) pero también ayudar directamente en tareas concretas, como por ejemplo el guiado, a través de realidad aumentada. Para poder evaluar las distintas estrategias en una fase previa al ensayo en pacientes, se han desarrollado los llamados simuladores de visión protésica, los cuales facilitan la implementación de estos algoritmos y su evaluación en sujetos sanos.

En este proyecto se plantea por primera vez, hasta nuestro conocimiento, la integración de un sistema de guiado en un simulador de visión protésica basado en técnicas de navegación convencionales. El objetivo es facilitar la navegación de las personas con visión reducida, mediante la representación sobre la imagen de fosfenos de la dirección hacia la que se deben mover para alcanzar una ubicación deseada.

El sistema de guiado desarrollado se ha evaluado inicialmente en un entorno virtual, y posteriormente en un entorno real utilizando una cámara RGB-D. Para ello se han integrado técnicas de SLAM visual (Simultaneous Localization and Mapping) con técnicas de planificación de trayectorias sobre un mapa que se va actualizando durante el movimiento, representando en él los obstáculos detectados en tiempo real. La escena y la trayectoria planificada se proyectan sobre la imagen de fosfenos percibida por la persona.



# Visual impaired guidance with Simulated Prosthetic Vision in virtual environments

## Abstract

Information about the environment is perceived by people through their senses. Sight is the sense that provides the greatest wealth of information about the scene and the location of the objects in it. Thanks to it we are able to perceive and recognize different objects and easily navigate the environment around us. Many people are deprived of this sense due to the different existing pathologies that gradually degenerate the visual pathway and can even lead to blindness. Depending on the severity of these lesions, it is possible to place a prosthesis in some part of the visual system that allows perceiving white dots of light elicited by electrical stimulation, known as phosphenes. These dots can represent, for example, simplified information about the scene that helps the subject to move without colliding.

These prostheses represent the information captured by a camera, processed with computer vision techniques, to carry out the stimulation of phosphenes. However, the dynamic range and visual resolution provided by these visual implants are very small compared to human vision, so it is convenient to select the information to be represented. For this reason, it is important to develop algorithms that are able to extract the most relevant information from the scene, in order to facilitate the navigation for the visually impaired. These algorithms can help in general perception (iconic representation, simplified information) but also help directly in specific tasks, such as guiding through augmented reality. In order to be able to evaluate the different strategies in a pre-trial phase in patients, so-called prosthetic vision simulators have been developed to facilitate the implementation of these algorithms and their evaluation in healthy subjects.

In this project it is proposed, –to the best of our knowledge– for the first time, the integration of a guidance system in a prosthetic vision simulator based on well-known navigation techniques. The goal is to facilitate the navigation of visual impaired people, by representing the direction they must follow to reach a desired location with phosphenes.

The guidance system developed has been evaluated initially in a virtual environment, and then in a real setting using an RGB-D camera. For this purpose, visual SLAM (Simultaneous Localization and Mapping) techniques have been integrated with trajectory planning techniques on a map that is continuously updated with the obstacles detected while the user is moving, in real time. The scene and the planned trajectory are projected onto the phosphene image perceived by the person.



# Índice general

<b>Índice de figuras</b>	<b>5</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Estado de las prótesis visuales y SPV	3
1.2. Motivación	7
1.3. Objetivos y alcance	7
1.4. Entorno de trabajo	8
<b>2. Simulador de visión protésica</b>	<b>9</b>
2.1. Cámara RGB-D	9
2.2. Segmentación de las imágenes	10
2.3. Configuración del SPV	12
2.4. Generación de imágenes de fosfenos a partir de información 3D	15
2.5. Modos de representación	18
<b>3. Sistema de guiado</b>	<b>20</b>
3.1. Navegación	20
3.1.1. Localización en un mapa	21
3.1.2. Sistemas de referencia	21
3.1.3. Planificación de trayectorias	23
3.1.3.1. Planificador global	23
3.1.3.2. Planificador Local	23
3.1.3.3. Mapas de coste	23
3.2. Representación de la trayectoria con fosfenos	24
3.2.1. Camino de guiado	25
3.2.2. Brújula	26
<b>4. Implementación y entorno de simulación</b>	<b>28</b>
4.1. ROS (Robotic Operating System)	29
4.2. Entorno de simulación	31
4.2.1. Creación del escenario virtual	32
4.2.2. Creación del modelo humano de simulación	33
4.3. Localización y mapas	35
4.3.1. Entorno virtual	35
4.3.2. Escenario real	37
4.4. Navegación y SPV	38
<b>5. Experimentos</b>	<b>40</b>

5.1. Pruebas de navegación en un entorno virtual . . . . .	40
5.1.1. Pruebas de navegación con el camino de guiado . . . . .	41
5.1.2. Pruebas de navegación con la brújula . . . . .	42
5.1.3. Comparación de los dos métodos de guiado . . . . .	43
5.2. Pruebas de navegación en un entorno real . . . . .	44
<b>6. Conclusiones y trabajo futuro</b>	<b>46</b>
<b>Bibliografía</b>	<b>47</b>

# Índice de figuras

1.1. Ejemplo de Simulador de Visión Protésica. En la imagen (a) se puede ver la imagen capturada por una cámara, y en la (b) la imagen de fosfenos generada por un SPV que muestra una versión simplificada de la escena.	2
1.2. Ejemplo de configuración de un SPV desarrollado por Bermudez et al. [1]. Este dispositivo está compuesto por una cámara Microsoft Kinect 2 y unas gafas de realidad virtual Oculus Rift.	2
1.3. Prótesis Argus II: (a) Implante de una malla de electrodos en la retina. (b) La información de la escena es captada por la cámara integrada en las gafas para ser procesada y transformada en un patrón de iluminación de electrodos de la malla.	4
1.4. Prótesis Alpha-IMS: La prótesis cuenta con una matriz de 1.500 fotodiodos colocada detrás de la retina. Cada fotodiodo capta la luz, la amplifica y la envía al electrodo el cual genera el estímulo.	4
1.5. Comparación entre procesamiento tradicional de imagen y la extracción del suelo usando la información de profundidad.	5
1.6. Diferentes configuraciones del SPV usando diferente numero de fosfenos ( $N_p$ ) y campo de vista de la cámara simulada de fosfenos (viene dada por su distancia focal ( $f$ )). Se presenta la imagen RGB y 4 configuraciones diferentes: (a) $f= 525px$ , $N_p= 484$ ; (b) $f= 525px$ , $N_p= 1862$ ; (c) $f= 400px$ , $N_p= 1862$ ; (d) $f= 200px$ , $N_p= 1862$ .	6
1.7. Ejemplo de un Simulador de visión protésica, en el que se combina la visión residual del sujeto con la imagen de fosfenos generada en función de la iluminación de la escena. Las barras de escala indican el campo de vista en deg de ambas imágenes.	6
2.1. Cámara Asus Xtion Pro Live	9
2.2. Segmentación de la escena: (a) Imagen de la escena captada por la cámara RGB. (b) Estimación de las normales de la nube de puntos obtenida con la cámara de profundidad. (c) Resultado de la segmentación de los planos de la imagen con el region-growing (d) Imagen de fosfenos procesada por el SPV	10
2.3. Elección del plano del suelo, representado en verde, en función de unos umbrales que permiten diferenciarlo del resto de planos de la escena	11
2.4. Modos de distribución de los fosfenos en la imagen: Cuadrado y hexagonal	13
2.5. Imágenes del SPV con diferentes tamaño de fosfenos.	13
2.6. Imágenes con diferente número de fosfenos 1000 y 700 fosfenos.	14
2.7. Imágenes con un campo de vista de 58 % y 25 %.	14
2.8. Imágenes sin dropout frente a una con un 25 %.	15
2.9. Proyección de puntos entre espacios 2D y 3D.	17

2.10. Primer modo de representación del SPV: tablero de ajedrez. . . . .	18
2.11. Segundo modo de representación del SPV: representación de objetos. . .	19
3.1. Ejes de referencia del sistema. . . . .	22
3.2. Costmaps: (a) Entorno de simulación con obstáculos cercanos. (b) Visualización en RVIZ del mapa de coste local. . . . .	24
3.3. Método de camino de guiado: (a) Entorno de simulación Gazebo. (b) Imagen de RVIZ en la que se puede ver la trayectoria calculada representada como una línea roja en el mapa. (c) Imagen captada por la cámara RGB. (d) Imagen generada por el SPV en la que se ha integrado la trayectoria calculada sobre el plano del suelo. . . . .	25
3.4. Secuencia de imágenes con el sistema de guiado de brújula. En la fila superior se encuentran las imágenes captadas por la cámara y en la inferior las generadas con el SPV. . . . .	26
3.5. Movimiento de los marcadores de dirección respecto de la parte central de la imagen en función de el desvío en el seguimiento de la trayectoria. La línea de puntos roja representa la trayectoria calculada por el sistema de guiado. La línea morada es la pendiente de la trayectoria en la zona próxima al sujeto. . . . .	27
4.1. Esquema general de los paquetes utilizados. . . . .	29
4.2. Árbol de transformaciones del sistema. Con el se puede comprobar que todas las referencias están siendo publicadas y existe una relación entre ellas. . . . .	31
4.3. <i>Building editor</i> : Herramienta de creación de modelos de Gazebo. . . . .	32
4.4. Modelo de la planta de un hospital. . . . .	33
4.5. Modelo del robot con la cámara RGB-D integrada: (a) Modelo 3D del robot en Gazebo, (b) Ejes de referencia principales del robot. . . . .	34
4.6. Movimiento vertical de la cabeza durante la marcha. . . . .	34
4.7. Mapa 2D del entorno de simulación: (a) Imagen de la planta del entorno. (b) Mapa 2D capturado con la información del sensor de profundidad. . .	36
4.8. (a) Mapa 3D del escenario obtenido con ORB-SLAM2 (b) Mapa 2D del mismo escenario obtenido con cartographer [2]. . . . .	38
4.9. Esquema de paquetes utilizados por el sistema de navegación. . . . .	39
5.1. (a) Escenario en Gazebo, (b) Trayectoria global calculada por el planificador sobre el mapa. . . . .	40
5.2. Secuencia de imágenes de la trayectoria seguida con el camino de guiado: En la fila superior se pueden ver las imágenes captadas por la cámara RGB y en las demás las imágenes generadas por el SPV. En la segunda fila el campo de vista es de 58° y la imagen tiene 1000 fosfenos. En la tercera fila el campo de vista es de 25° y la imagen tiene 500 fosfenos. . .	41
5.3. Secuencia de imágenes de la trayectoria seguida con la brújula: En la fila superior se pueden ver las imágenes captadas por la cámara RGB y en las demás las imágenes generadas por el SPV. En la segunda fila el campo de vista es de 58° y la imagen tiene 1000 fosfenos. En la tercera fila el campo de vista es de 25° y la imagen tiene 500 fosfenos. . . . .	42
5.4. Trayectoria seguida por el sujeto con (a) el sistema de guiado de camino y (b) el sistema de guiado de brújula . . . . .	43

5.5. Planificación de la trayectoria sobre el mapa global 2D del entorno. . . .	44
5.6. (a) Nube de puntos 3D de la escena con la trayectoria representada con una línea verde y azul. (b) Imágenes generadas por el SPV con la trayectoria proyectada como una línea de puntos blanca. . . . .	45

# Capítulo 1

## Introducción

La vista es el sentido que proporciona más información del entorno que nos rodea. Gracias a ella somos capaces de reconocer objetos y personas, y nos proporciona la capacidad de navegar por el entorno con facilidad. La Organización Mundial de la Salud (OMS) [3] estima que aproximadamente 1.300 millones de personas viven con alguna forma de deficiencia de la visión, 217 millones tienen una deficiencia visual de moderada a grave y 36 millones son ciegas. El crecimiento y envejecimiento de la población aumentarán el riesgo de que más personas se vean afectadas por una deficiencia visual [4].

A lo largo de los últimos años se ha investigado sobre el desarrollo de dispositivos capaces de conseguir la percepción de imágenes formadas por puntos de luz, mediante la estimulación de diferentes partes de la vía visual, como son la corteza cerebral, el nervio óptico o la retina. Dicha estimulación permite percibir unos puntos brillantes de luz llamados *fosfenos*. Se requiere de un dispositivo que recoja información de la escena, la procese y la transforme en una señal de excitación. El dispositivo de captación de la escena suele ser una cámara convencional, con la cual se obtienen imágenes que se asemejan a lo que los humanos percibimos con nuestros ojos. Para el procesamiento de dichas imágenes ha de tenerse en cuenta el limitado nivel de detalle que puede transmitirse mediante los estímulos visuales descritos. Es por esto que se ha de tratar de buscar la manera de extraer la información más relevante del mundo que les rodea.

El conjunto de técnicas para el procesamiento y entendimiento de imágenes mediante un ordenador se enmarca en la disciplina conocida como visión por computador o visión artificial. Muchas de las aplicaciones desarrolladas mediante el uso de estas técnicas se centran en mejorar la calidad de vida de las personas. Este proyecto se enmarca en el desarrollo de aplicaciones para la ayuda de personas con algún tipo de lesión visual.

Existen sistemas de visión protésica que utilizan algoritmos de visión por computador para transformar imágenes obtenidas con una cámara en imágenes de baja resolución formadas por una malla de puntos representados en escala de grises. Debido a que la

información que puede transmitirse con este tipo de prótesis visuales es muy limitada, el objetivo de estos algoritmos es maximizar la información que se puede percibir en estas imágenes mediante representaciones icónicas. Para facilitar el desarrollo y evaluación de algoritmos y técnicas de visión por computador sobre pacientes sanos, sin la necesidad de realizar ensayos clínicos, se utilizan Simuladores de Visión Protésica (SPV). Los SPV son sistemas que generan imágenes sintéticas que simulan la información transmitida por un implante de prótesis visual a partir de imágenes o secuencias normalmente captadas con una cámara (Figura 1.1). Dichas imágenes sintéticas se pueden mostrar en una pantalla o en un sistema inmersivo de realidad virtual para darle mayor realismo (Figura 1.2).

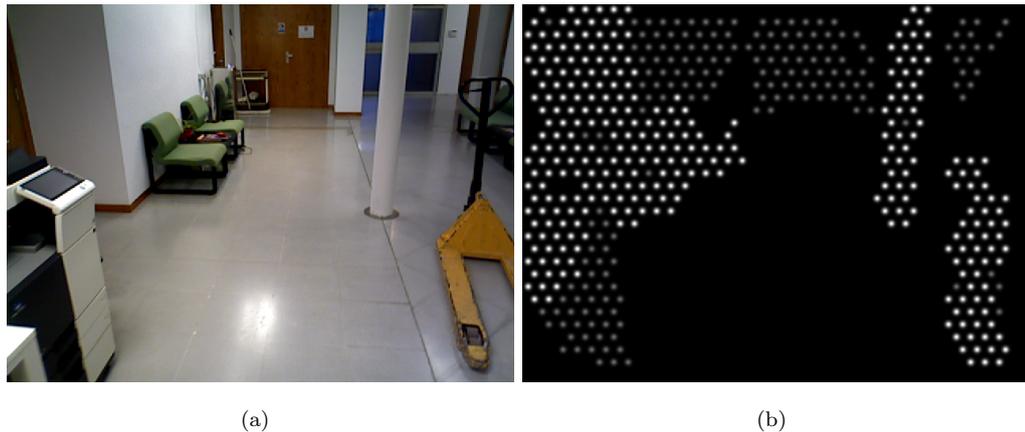


FIGURA 1.1: Ejemplo de Simulador de Visión Protésica. En la imagen (a) se puede ver la imagen capturada por una cámara, y en la (b) la imagen de fosfenos generada por un SPV que muestra una versión simplificada de la escena.



FIGURA 1.2: Ejemplo de configuración de un SPV desarrollado por Bermudez et al. [1]. Este dispositivo está compuesto por una cámara Microsoft Kinect 2 y unas gafas de realidad virtual Oculus Rift.

Otra rama de investigación va más allá y utiliza imágenes captadas en entornos simulados. El uso de este tipo de entornos virtuales es de gran utilidad debido a que permite realizar experimentos de forma cómoda y segura en los cuales se puedan evaluar eficazmente diferentes tipos de escenarios, sensores, tareas o algoritmos. Además, este tipo de experimentación permite recopilar datos de manera sistemática y desacoplar problemas

complejos como puede ser la localización precisa en interiores, con el fin de centrarse en los modos de representación.

En este proyecto se va a trabajar sobre un SPV que traduce las imágenes captadas por una cámara RGB-D en imágenes de fosfenos, utilizando representaciones icónicas de la realidad que permiten al sujeto desplazarse sin colisiones. En este trabajo se propone ir más allá de la simple representación de obstáculos, ampliamente tratada en el trabajo relacionado, añadiendo como información adicional la dirección que debe seguir el usuario para llegar a una ubicación concreta como puede ser el baño, la cafetería o el ascensor en un lugar del cual se dispone de un mapa. La persona será guiada en ese mapa hacia el destino deseado, aplicando métodos de navegación con los que se puedan generar trayectorias a partir de las que obtener una dirección de movimiento que plasmar en la imagen de fosfenos. En particular, se han desarrollado dos estrategias para transmitir esta información. Debido a que el usuario puede encontrar diferentes tipos de obstáculos a lo largo de la trayectoria, se le ha prestado atención a que el sistema sea capaz de detectarlos y corregir la trayectoria en tiempo real, para que ésta sea lo más segura posible. El sistema ha sido evaluado tanto con imágenes reales captadas por una cámara como en entornos de simulación.

## 1.1. Estado de las prótesis visuales y SPV

Algunas personas están privadas de la vista, uno de los sentidos que mas información aportan del entorno. Por ello el deseo de muchas personas es poder dotarles de este sentido. Existen diferentes líneas de investigación centradas en el estudio de las diferentes formas de estimulación de las vías visuales los cuales permiten percibir unos puntos brillantes de luz [5]. Estas prótesis consisten en una malla de electrodos que estimula la retina, el nervio óptico o la corteza cerebral, con el fin de que la persona pueda percibir una malla de fosfenos que se asemeja a una imagen de baja resolución.

La retinitis pigmentosa es una de las principales causas de pérdida visual que motivan la implantación de estas prótesis. Se trata de una degeneración progresiva de la retina (la estructura transparente y sensible a la luz situada en la zona posterior del ojo), que con el tiempo provoca una pérdida visual entre moderada y grave. Existen en el mercado diferentes prótesis visuales que permiten recuperar cierta visión funcional en pacientes afectados. Una de ellas, aprobada en Europa en marzo de 2011, es la Argus II de la empresa *Second Sight Medical Products, Inc.* [6]. La prótesis cuenta con un procesador, una cámara y una malla de 60 electrodos que producen un ángulo de visión de 20 grados y que se coloca cerca de la retina. La separación entre los centros de los electrodos es de 525  $\mu\text{m}$  (tanto en horizontal como en vertical), que corresponde a unos 1,88° de campo visual. La información del entorno es captada por la cámara, la cual se procesa para extraer la información más relevante, que finalmente será plasmada en la malla de

electrodos (Figura 1.3).

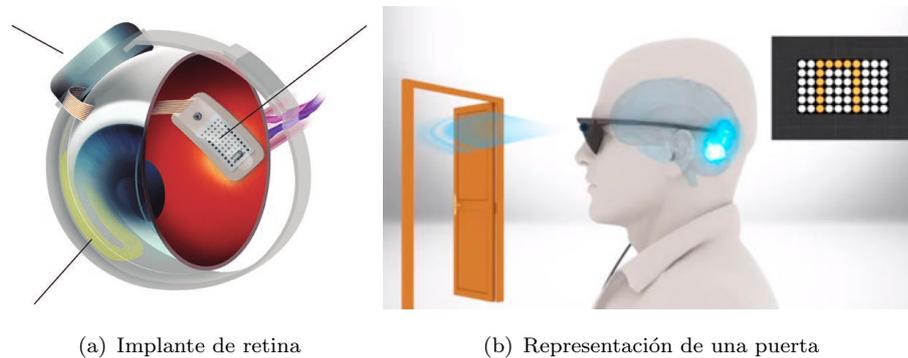


FIGURA 1.3: Prótesis Argus II: (a) Implante de una malla de electrodos en la retina. (b) La información de la escena es captada por la cámara integrada en las gafas para ser procesada y transformada en un patrón de iluminación de electrodos de la malla.

Otra de las prótesis más comunes, desarrollada por Retina Implant AG y aprobada también en la Unión Europea, es la Alpha-IMS[7]. Se trata de un implante subretinal de 1.500 fotodiodos, que capturan la luz y la convierten en corriente eléctrica para estimular las células ganglionares. Esta señal completa el resto del camino de la vía visual hasta llegar al cortex. En este caso no hay procesamiento de imagen, ya que directamente se sustituyen los fotorreceptores dañados de la retina por este sensor capaz de captar la luz. El implante tiene un tamaño de  $3 \times 3$  mm y los electrodos tienen un diámetro de  $50\mu\text{m}$ , con una distancia entre centros de electrodos de  $70\mu\text{m}$  que equivale a una resolución angular de  $0,25^\circ$ .

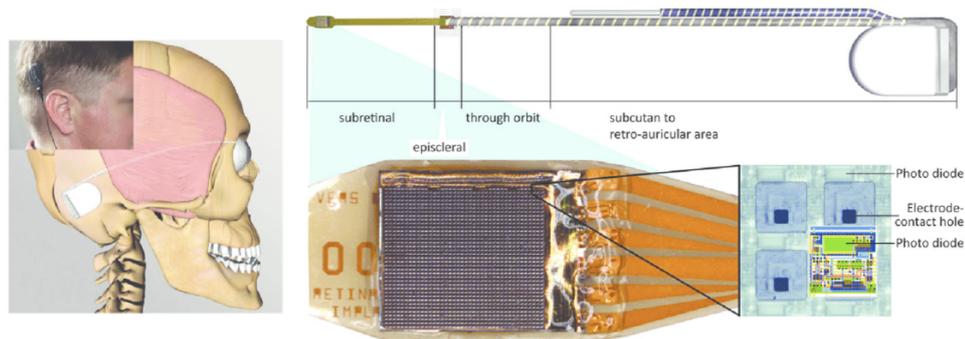


FIGURA 1.4: Prótesis Alpha-IMS: La prótesis cuenta con una matriz de 1.500 fotodiodos colocada detrás de la retina. Cada fotodiodo capta la luz, la amplifica y la envía al electrodo el cual genera el estímulo.

La manera más común para evaluar modos de representación mediante fosfenos es utilizando simuladores de prótesis visuales (SPV). Chen et al. [8] propone un modelo de estandarización con el fin de hacer los SPV más realistas, comparando las representaciones del simulador con la descripción obtenida de sujetos que han probado la prótesis

visual con el fin de ajustar parámetros como son el tamaño, el aspecto, la separación o la distribución espacial.

Existen diferentes líneas de investigación centradas en el procesamiento de la información obtenida por las cámaras utilizando técnicas de visión por computador, con el fin de representar con un número de píxeles determinado la mayor cantidad de información posible del entorno. La mayoría utilizan información extraída de cámaras convencionales [9]. Algunas están empezando a utilizar métodos más sofisticados, como el *deep learning* [10]. Por ejemplo, [11] lo usa para detectar información relevante de la estructura de la escena para destacarla.

Sin embargo, la propuesta de este proyecto se alinea más con los que utilizan cámaras RGB-D en vez de convencionales, debido a la gran utilidad que tiene la información de profundidad en este tipo de aplicaciones. Al dar información tridimensional de la escena delante de la persona, permite desarrollar métodos más seguros y fiables. Un ejemplo en el que se utiliza la profundidad de la escena para extraer información relevante de la misma, es el mostrado en la Figura 1.5 de Lui et al. [12], en el que se realiza una segmentación del suelo con el fin de detectar zonas libres de obstáculos por las cuales el sujeto pueda desplazarse. McCarthy et al. [13] presenta una representación que llama *profundidad aumentada*, donde píxeles de distinta intensidad representan los cambios de profundidad en la escena, y el suelo se elimina para destacar los objetos.



(a) Procesamiento de imagen tradicional



(b) Extracción del suelo con la información de rango

FIGURA 1.5: Comparación entre procesamiento tradicional de imagen y la extracción del suelo usando la información de profundidad.

En este proyecto se va a utilizar un SPV desarrollado por investigadores de la Universidad de Zaragoza, llamado RGBD-VIA [14] (Figura. 1.6). Dicho simulador, basado en imágenes captadas por una cámara RGB-D, ofrece la posibilidad de configurar diversos parámetros de la imagen de fosfenos tales como el tamaño, distribución espacial, aspecto, el campo de vista o el número de fosfenos. Además, cuenta con diferentes modos de representación de la escena, destacando una novedosa representación icónica del espacio libre de obstáculos que además transmite sensación de profundidad y avance sin depender de una resolución fotométrica muy elevada.

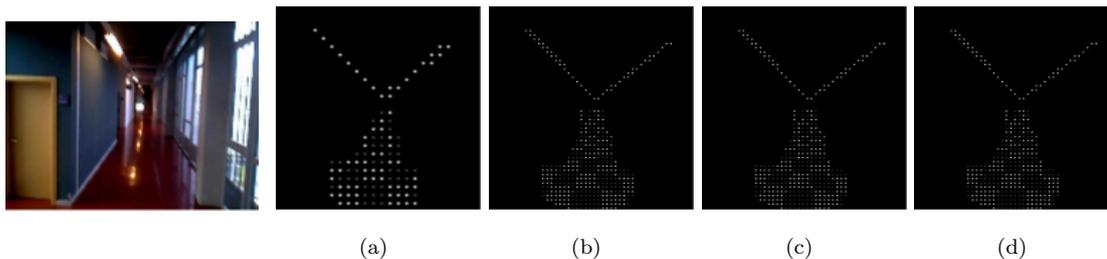


FIGURA 1.6: Diferentes configuraciones del SPV usando diferente numero de fosfenos ( $N_p$ ) y campo de vista de la cámara simulada de fosfenos (viene dada por su distancia focal ( $f$ )). Se presenta la imagen RGB y 4 configuraciones diferentes: (a)  $f= 525px$ ,  $N_p= 484$ ; (b)  $f= 525px$ ,  $N_p= 1862$ ; (c)  $f= 400px$ ,  $N_p= 1862$ ; (d)  $f= 200px$ ,  $N_p= 1862$ .

Otros investigadores también utilizan entornos simulados para realizar sus experimentos. Un ejemplo es el desarrollado por Zapf et al. [15], con el que hacen un estudio de la capacidad de movimiento en escenarios con peatones quietos y en movimiento. Las imágenes generadas por este simulador (Figura 1.7), combinan una simulación de visión residual (SRV) y visión periférica de  $102^\circ$  con el fin de comprobar si esta información es valiosa en tareas de navegación. En este proyecto se va a utilizar también este enfoque y se harán experimentos también en entornos simulados.

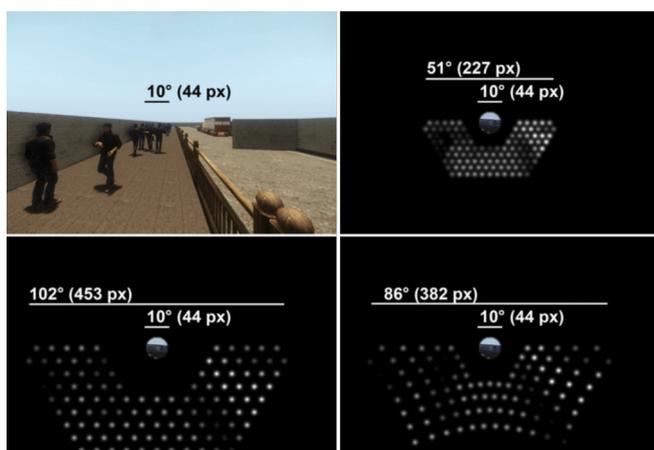


FIGURA 1.7: Ejemplo de un Simulador de visión protésica, en el que se combina la visión residual del sujeto con la imagen de fosfenos generada en función de la iluminación de la escena. Las barras de escala indican el campo de vista en deg de ambas imágenes.

## 1.2. Motivación

Un gran número de las aplicaciones de visión por computador se centran en mejorar la calidad de vida de las personas. Actualmente existen en el mercado dispositivos capaces de captar y procesar la información del entorno para estimular en la vía visual humana una imagen mallada del mismo. Son numerosos los trabajos en esta línea de investigación centrados en mejorar la movilidad de las personas mediante el procesamiento de la información de la escena. Sin embargo no hay ninguna aplicación centrada en el guiado de personas con algún tipo de lesión visual, que utilice técnicas de navegación para ayudarlas a desplazarse de forma segura de un lugar a otro evitando los obstáculos que se interponen en su camino.

Son numerosos los lugares públicos en los que se deben seguir una serie de indicaciones para alcanzar el objetivo deseado, por ejemplo, mediante carteles o líneas en el suelo. Un ejemplo en el que resultaría útil un sistema de guiado es un incendio, en el cual se requiere una evacuación rápida del lugar. Otro ejemplo podría ser un hospital, en el cual una persona con lesión visual no puede seguir las indicaciones marcadas en el suelo o en las paredes hasta llegar a la zona de atención deseada.

Las personas con lesiones visuales presentan grandes dificultades para seguir este tipo de indicaciones. Por ello la principal motivación de este proyecto es el desarrollar un sistema de guiado adaptable a la tecnología de prótesis visuales, que facilite y mejore la eficiencia y la seguridad con la que estas personas se desplazan por el entorno.

## 1.3. Objetivos y alcance

Este proyecto se enmarca en el desarrollo de aplicaciones para la ayuda de personas con algún tipo de lesión visual. El objetivo principal del mismo es desarrollar un sistema de guiado integrado en un simulador de visión protésica. Para abordar el problema de guiado, se ha planteado el uso de algoritmos de navegación convencionales sobre mapas previamente conocidos que se actualizan durante el movimiento y que permiten guiar y localizar al sujeto sobre el mismo. El objetivo principal es utilizar el sistema de guiado en un entorno virtual que permita evaluar las distintas propuestas de representación y asistencia en sujetos reales. El uso de simulación permite secuencias repetibles lo que da sentido estadístico a un futuro diseño de experimentos. Este planteamiento inicial se ha extendido para que el sistema se pueda usar en escenarios reales utilizando una cámara RGB-D. Ambos enfoques se integran en un diseño modular que, en el caso de imágenes reales, utiliza un sistema de SLAM para la localización.

## 1.4. Entorno de trabajo

Las herramientas con las que se ha desarrollado este proyecto son las siguientes:

- **RGBD-VIA**: Es un simulador de visión protésica desarrollado en la Universidad de Zaragoza, que permite generar imágenes de fosfenos con una cámara RGB-D [14].
- **Gazebo**: Es una herramienta que permite crear escenarios virtuales realistas tanto exteriores como interiores, en los que poder probar algoritmos y diseñar complejos modelos de robots.
- **ROS (Robotic Operating System)**: Es un conjunto de librerías y herramientas desarrolladas en su gran mayoría en C++ centradas en el desarrollo de aplicaciones robóticas.
- **RVIZ**: Herramienta de visualización integrada en las librerías de **ROS** con la que se pueden visualizar diferentes tipos de datos tales como nubes de puntos, escáneres láser, mapas, modelos robóticos o imágenes.
- **OpenCV y PCL** [16]: Conjunto de librerías de visión por computador en C++ para procesamiento de imágenes y nubes de puntos.
- **ORB-SLAM2** [17]: Es una librería de SLAM en tiempo real para cámaras monoculares, estéreo y RGB-D desarrollada en la Universidad de Zaragoza. Con ella se puede calcular la trayectoria de la cámara y realizar una reconstrucción 3D dispersa del entorno.

## Capítulo 2

# Simulador de visión protésica

Con el fin de simular el efecto que produce sobre una persona una prótesis visual, se ha utilizado un software capaz de transformar la información captada por una cámara RGB-D en una imagen mallada de baja resolución en las que se intenta representar la mayor cantidad de información de la escena posible. Esto es lo que se conoce como Simulador de Visión Protésica (SPV). En este capítulo se presenta el funcionamiento básico del SPV que se va a emplear en este trabajo [14] y sobre el cual se ha trabajado para incluir un sistema de guiado.

### 2.1. Cámara RGB-D

El dispositivo utilizado para captar las imágenes es una cámara RGB-D, la cual obtiene información del color de la escena utilizando una cámara convencional y un sensor de profundidad con el cual se obtiene información 3D de la misma. A partir de la información de profundidad de la escena, este tipo de cámaras proporcionan una nube de puntos de la misma, en la que cada punto contiene su posición 3D respecto a la cámara. Trabajar con las nubes de puntos proporcionadas por el sensor de profundidad, es un tanto diferente a trabajar con únicamente la información de color captada por la cámara, ya que al conocer la coordenada 3D de cada punto de la imagen, se puede obtener con mayor facilidad información geométrica de los objetos del entorno. En este proyecto se ha utilizado una librería llamada PCL (Point Cloud Library) [16] la cual permite realizar numerosas operaciones con este tipo de datos.



FIGURA 2.1: Cámara Asus Xtion Pro Live

La cámara utilizada en las simulaciones ha sido la ASUS Xtion Pro Live (Figura 2.1) debido a que se dispone de este modelo en el departamento. De esta forma tenemos la posibilidad de usarla tanto en simulación como en real y desarrollar métodos que funcionen indistintamente en un modo u otro. En las simulaciones se ha utilizado una cámara virtual a la cual se le pueden asignar los parámetros intrínsecos y extrínsecos de la cámara para que se asemeje en la medida de lo posible a la utilizada en la realidad.

## 2.2. Segmentación de las imágenes

Las imágenes basadas en fosfenos están formadas por una malla de puntos de baja resolución (tanto espacial como fotométrica), en la cual se intenta representar la información más relevante obtenida a partir de las imágenes captadas por una cámara. En este proyecto se ha utilizado el paquete SPV de [14], el cual viene con una serie de métodos implementados para extraer información de la cámara RGB-D y plasmarla en un sistema de visión protésica simulada de acuerdo con el tipo de representación de la escena elegida y los parámetros de configuración de la visualización. En esta sección se comenta cómo se divide la escena en distintos planos y objetos no planares, proceso conocido como *segmentación*.

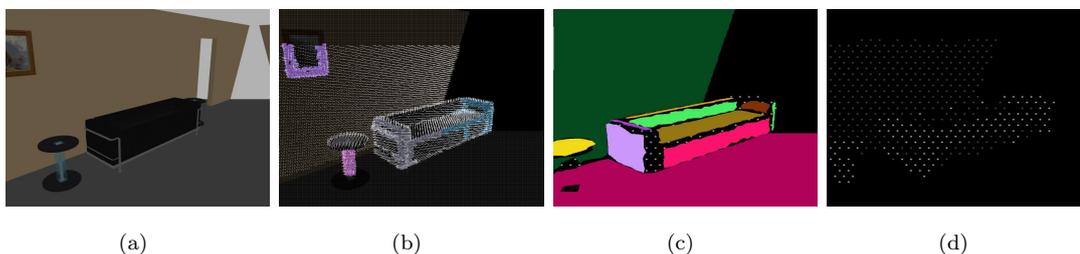


FIGURA 2.2: Segmentación de la escena: (a) Imagen de la escena captada por la cámara RGB. (b) Estimación de las normales de la nube de puntos obtenida con la cámara de profundidad. (c) Resultado de la segmentación de los planos de la imagen con el region-growing (d) Imagen de fosfenos procesada por el SPV

El primer paso es el **preprocesado** de la nube de puntos de profundidad con el fin de reducir la densidad de la misma para que sucesivas operaciones vean el coste computacional reducido al habernos deshecho de puntos que dan información redundante y poco relevante. Para ello se aplica una técnica llamada *filtro de voxel*, que consiste en subdividir el espacio 3D en cubos y seleccionar únicamente el centroide de los puntos contenidos en cada uno de ellas.

Para facilitar la comprensión de la escena, el siguiente paso consiste en **extraer el plano de suelo**. Esto nos va a permitir cambiar la referencia de la escena de manera que podamos razonar acerca de la posición y orientación de los distintos elementos de la misma para luego poder decidir cómo representarlos. Además, nos será útil para discernir

entre superficie sobre la cual se puede caminar y paredes y obstáculos, así como para facilitar la localización en un mapa para la navegación. El procedimiento usado para extraer el plano del suelo es el llamado RANdom SAmple Consensus (RANSAC) [18]. El plano elegido como plano de suelo (Figura 2.3) será aquel que tiene una orientación y distancia no muy diferente de la esperable considerando que la cámara va a estar en la cabeza y el individuo se encontrará mirando hacia abajo (aproximadamente a  $45^\circ$ ).

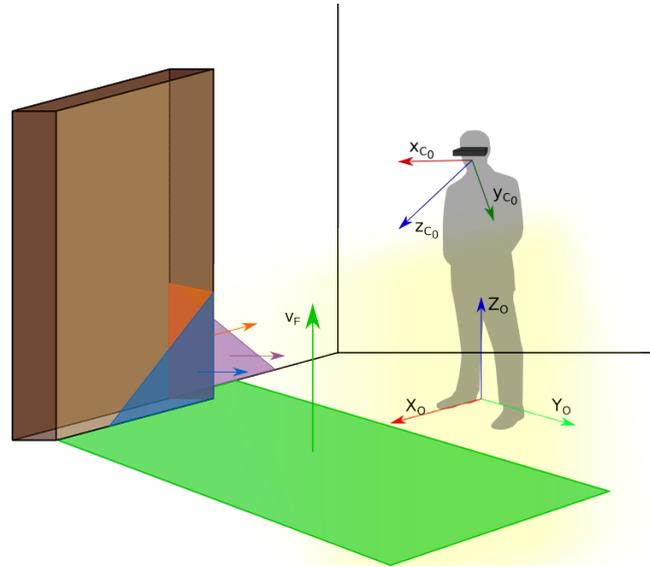


FIGURA 2.3: Elección del plano del suelo, representado en verde, en función de unos umbrales que permiten diferenciarlo del resto de planos de la escena

Los elementos en los cuales se va a segmentar la escena son *planos* y agrupaciones de puntos (denominadas *clusters*). Para extraer los **planos**, primero se realiza una estimación de las normales de cada punto basada en PCA (Principal Component Analysis) [19]. Se calculan los valores y vectores propios de una matrix de covarianza creada por los vecinos más cercanos de cada punto (Figura 2.2 (b)) y se escoge como normal el vector propio correspondiente al valor propio menor. Para extraer planos se utiliza un algoritmo llamado Region-Growing [20], integrado en la librería PCL. Este algoritmo divide el espacio en regiones cuyos puntos cumplen que la orientación de sus normales y la separación espacial de los mismos son similares (Figura 2.2 (c)).

La salida de este algoritmo es la ecuación del plano formado por los puntos de la región definido como  $Ax + By + Cz + D = 0$  donde  $(A, B, C, D)$  son los coeficientes del plano, siendo  $\mathbf{n}(A, B, C)$  el vector normal al plano y  $D$  la distancia al origen en la dirección de la normal. Posteriormente se realiza una **clasificación** de los planos según su posición y orientación con respecto al suelo. Se considerarán planos horizontales (suelo, mesas), verticales (paredes, puertas) y oblicuos (obstáculos).

Los puntos de la nube que no pertenezcan a ningún plano se procesarán en un algoritmo de extracción de **clusters**, el cual simplemente agrupa en clusters independientes

las agrupaciones de puntos próximos entre sí. Estos clusters pasan a ser directamente obstáculos sin más razonamiento (e.g. personas).

### 2.3. Configuración del SPV

Debido a los recientes avances en las pruebas clínicas realizadas con prótesis visuales, surge la necesidad de comprender mejor los aspectos psicofísicos y perceptivos de la visión protésica. En visión protésica, se conoce como *fosfenos*, a los puntos de luz percibidos a raíz de una estimulación eléctrica o mecánica de las vías visuales. Trabajos como [8] han recopilado información acerca de cómo se perciben y distribuyen dichos fosfenos y qué características particulares puede haber en su visualización en personas con implantes.

En este trabajo se va a utilizar el simulador desarrollado por [14], el cual recoge los modelos descritos y planteados por [8] para mostrar representaciones lo más realista posibles de lo que ven los pacientes. A lo largo de este trabajo nos referiremos comúnmente a este simulador como *SPV*. Este sistema tiene una serie de **parámetros** que pueden ser combinados entre sí para permitir simular distintos posibles dispositivos de visión protésica. Los parámetros más relevantes de configuración y una breve explicación acerca de los mismos son:

- **Forma:** El aspecto del fosfeno esta principalmente definido por su forma y color. El fosfeno es habitualmente descrito como un “pequeño punto de luz en la distancia”, y suele representarse como un círculo con los bordes difuminados. En este caso se han modelado con una distribución gaussiana:

$$f(x, y) = Ae^{-\left(\frac{(x-x_0)^2}{2\sigma_x^2} + \frac{(y-y_0)^2}{2\sigma_y^2}\right)} \quad (2.1)$$

donde  $A$  es la amplitud,  $(x_0, y_0)$  es el centro y  $\sigma_x$  y  $\sigma_y$  las desviaciones típicas en cada eje.

- **Color:** El color de los fosfenos se representa en escala de grises, donde se va a considerar que un fosfeno puede tomar distintos valores de intensidad según la corriente que se aplica a los electrodos. El número de niveles de intensidad es otro parámetro de configuración (que habitualmente se puede dejar en 8 [8]), aunque en este caso intentaremos crear representaciones que usen el mínimo de niveles posibles, dado que puede resultar complicado distinguir entre distintos niveles próximos entre sí y queremos métodos que no den lugar a equívoco. Es por esto que nos centraremos en representaciones icónicas con solo 3 niveles: blanco (corriente máxima), gris (corriente media) y negro (apagado).

- **Mapa:** La distribución espacial que adoptan los fosfenos en el campo de vista del paciente se denomina mapa de fosfenos. El simulador ofrece la posibilidad de cambiar la distribución entre las dos más comúnmente utilizadas en los SPV: cuadrada y hexagonal (Figura 2.4).

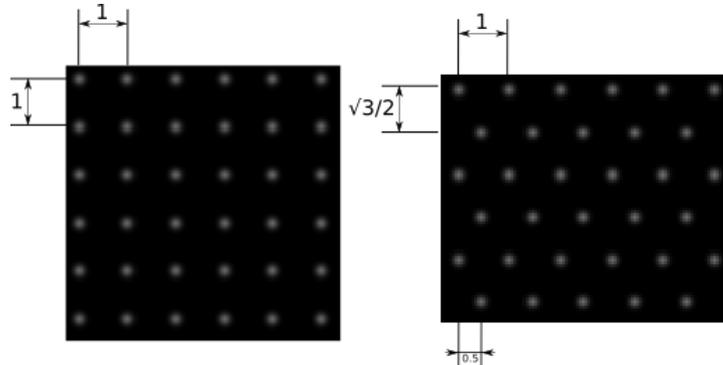


FIGURA 2.4: Modos de distribución de los fosfenos en la imagen: Cuadrado y hexagonal

- **Tamaño:** Con este simulador se puede aumentar o disminuir el tamaño de los fosfenos. Dicha tamaño se mide en grados de campo de vista y se ha observado que suele variar entre  $0.5^\circ$  y  $2^\circ$ .

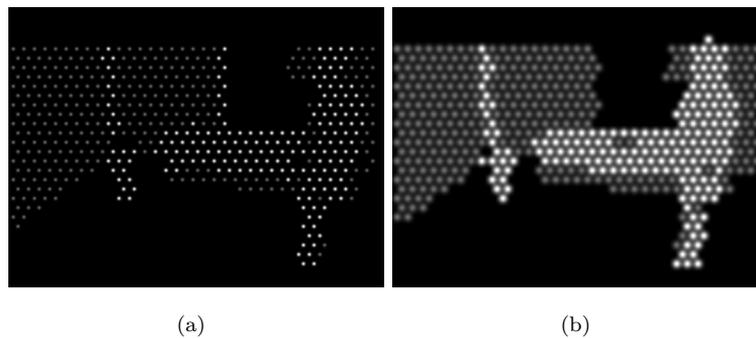


FIGURA 2.5: Imágenes del SPV con diferentes tamaño de fosfenos.

- **Número:** El simulador permite variar la cantidad de fosfenos de la imagen aumentando o disminuyendo la separación entre ellos. El número de fosfenos va a depender de la prótesis en sí, donde hemos visto que la Argus II (modelo más antiguo) tiene solo 60 fosfenos y el Alpha-IMS es capaz de llegar a 1.500. En nuestros experimentos nos moveremos entre 500 y 1000 fosfenos.

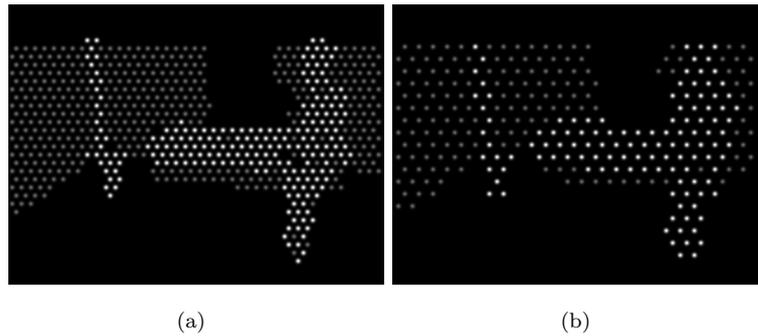


FIGURA 2.6: Imágenes con diferente número de fosfenos 1000 y 700 fosfenos.

- **Campo de vista:** Con este simulador podemos cambiar el campo de vista, y por tanto la amplitud de la escena que podemos representar con fosfenos. Con los sistemas de prótesis visuales actuales el campo de vista que puede alcanzarse es de aproximadamente  $20^\circ$  o  $30^\circ$ , muy por debajo de los aproximadamente  $210^\circ \times 150^\circ$  que tiene un humano sin deficiencias visuales.

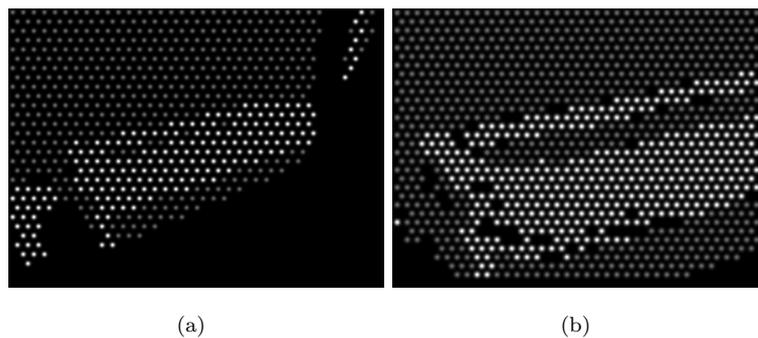


FIGURA 2.7: Imágenes con un campo de vista de 58% y 25%.

- **Ruido y Drop-out:** Normalmente no todos los fosfenos se iluminan debido a posibles daños en la vía visual o a un mal funcionamiento de la prótesis. Para modelar este problema, el simulador cuenta con un parámetro llamado *Drop-out* con el cual se puede conseguir que un porcentaje de fosfenos estén apagados de forma aleatoria. El simulador también cuenta con la posibilidad de añadir ruido gaussiano a la posición de cada fosfeno con el fin de simular que puede ser que haya un desplazamiento en la malla (Figura 2.8).

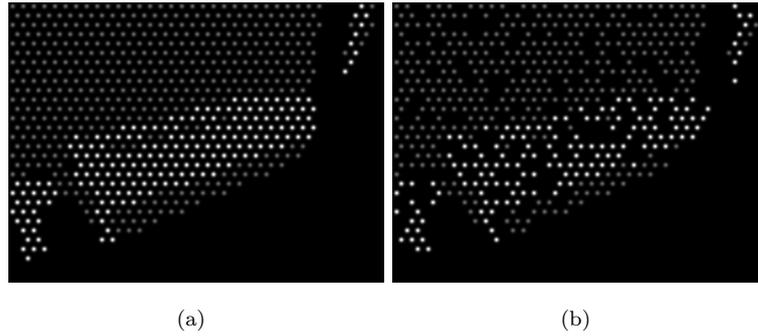


FIGURA 2.8: Imágenes sin dropout frente a una con un 25%.

## 2.4. Generación de imágenes de fosfenos a partir de información 3D

La salida del SPV ha de ser una imagen que simule lo que ven los pacientes con prótesis visuales. Para ello se crea una imagen de la resolución que se desee (según el tipo y resolución de pantalla), inicialmente con todos los píxeles en negro. Posteriormente, se calculan las posiciones  $\mathbf{p}$  de píxeles donde se situarán los fosfenos (los centros de los mismos) de manera que quepa el número de fosfenos establecido según la distribución establecida (rectangular o hexagonal). Por otro lado, se crea un fosfeno tipo completamente encendido con la forma y el tamaño que va a mostrarse en la imagen, y se calculan los niveles de intensidad intermedios estipulados entre el fosfeno completamente encendido (valor máximo blanco) y completamente apagado (completamente negro). En cada una de las posiciones  $\mathbf{p}$  se añadirá el fosfeno del nivel que determinará el modo de representación que esté actualmente siendo aplicado. Se contempla la posibilidad de que el ruido y el dropout altere la posición de los fosfenos en la imagen.

Nótese que hemos hablado de la visualización del SPV como una imagen. Dado que el tipo de elementos que vamos a querer representar en la imagen (planos y puntos) están en 3D en el mundo real, para poder proyectar entre imagen y mundo, haremos uso del modelo de la cámara *pinhole*. Con la proyección perspectiva, un punto  $\mathbf{X}$  en el espacio se corresponde con una posición en la imagen  $\mathbf{x}$  gracias a la matriz de proyección  $\mathbf{P}$  empleando coordenadas homogéneas.

$$\mathbf{x} = \lambda \mathbf{P} \mathbf{X} \quad (2.2)$$

Donde  $\lambda$  es el factor de escala y  $\mathbf{P}$  es la matriz de la cámara  $3 \times 4$ , que puede ser expresada como  $\mathbf{P} = [\mathbf{K} \mid \mathbf{0}] \mathbf{T}$  donde  $\mathbf{K}$  es la matriz intrínseca y  $\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$  representa la matriz extrínseca.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} f/d_x & 0 & c_x & 0 \\ 0 & f/d_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (2.3)$$

La matriz extrínseca de la cámara transforma los puntos tridimensionales del espacio en puntos tridimensionales en la referencia de la cámara. Explica la rotación ( $\mathbf{R}$ ) y traslación ( $\mathbf{t}$ ) para cada punto desde sus coordenadas originales hasta las coordenadas de la cámara. La matriz intrínseca  $\mathbf{K}$  contiene información sobre el centro de proyección  $(c_x, c_y)$ , el tamaño de los píxeles  $(d_x, d_y)$  y la distancia focal ( $f$ ). La distancia focal  $f$  es un parámetro de gran relevancia, puesto que de él depende el campo de vista de la cámara. O mejor dicho, se escoge una distancia focal  $f$  en función del campo de vista  $FOV_W \times FOV_H$  del sistema (parámetro de configuración del SPV) y la resolución de la imagen ( $W \times H$ ), relacionado a través de:

$$f_x = \frac{W}{2 \cdot \tan(FOV_W/2)} \quad (2.4)$$

donde  $f_x$  es la distancia focal en el eje  $x$ . Por simplicidad, en imágenes de fosfenos forzaremos  $f_x = f_y = f$ .

Para representar puntos 3D (y por tanto grupos de puntos o clusters) en imagen de fosfenos, utilizaremos la ecuación (2.3) para proyectar los puntos  $\mathbf{X}$  de la escena en píxeles de la imagen de fosfenos. Siendo conocidas las posiciones de los fosfenos en la imagen ( $\mathbf{p}$ ), se trata de localizar el fosfeno más próximo, procedimiento el cual puede acelerarse con una *look-up table*. La *look-up table* consiste en crear una matriz de la dimensión de la imagen en la que cada píxel indica el fosfeno más próximo (solo ha de computarse una vez para cada distribución de fosfenos). En la figura 2.9 se puede ver un ejemplo de la proyección con dos puntos.

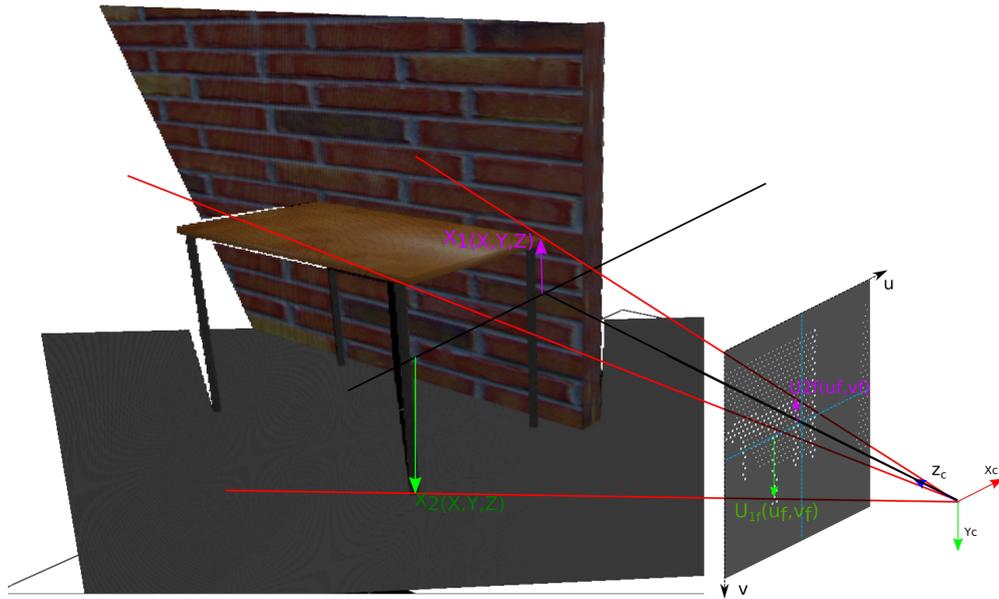


FIGURA 2.9: Proyección de puntos entre espacios 2D y 3D.

El otro elemento principal que vamos a querer representar son los planos 3D. Nótese que lo que nosotros hemos llamado *planos* es en realidad una *región plana*, que tiene ecuación del plano pero también vértices que la delimitan. Para calcular qué fosfenos han de activarse (es decir, cuales apuntan a un plano) lo haremos trazando rayos y calculando su intersección con los planos, con los siguientes pasos:

1. Se calculan los rayos de los fosfenos, dadas sus posiciones  $\mathbf{p}$ :

$$\mathbf{r} = \lambda \mathbf{K}^{-1} \mathbf{p} \quad (2.5)$$

donde en este caso  $\mathbf{r}$  denota la dirección de un rayo que emana de la imagen de fosfenos y no un punto en la escena.

2. Calcular la intersección 3D con el plano, considerando que tiene normal  $\mathbf{n}$  y distancia al origen  $D$ :

$$\mathbf{X} = -\frac{D \cdot \mathbf{r}}{\mathbf{n} \cdot \mathbf{r}} \quad (2.6)$$

donde  $\mathbf{X}$  es el punto 3D de intersección.

3. Comprobar que el punto de intersección está dentro de la región plana. Para ello se cambia a referencia 2D sobre el plano y se aplica un algoritmo que comprueba si un punto está dentro de un polígono. Previamente, es necesario sacar el polígono que rodea la región, lo cual puede hacerse aplicando un algoritmo de *Concave Hull*.

## 2.5. Modos de representación

El SPV viene con varios modos implementados para representar la información de la cámara RGB-D mediante fosfenos. En particular, destacamos dos de ellos, que utilizan directamente el procedimiento de representación de puntos y planos descrito en la Sección 2.4. En contraposición a representaciones clásicas (como la de transformación de la imagen a escala de grises de fosfenos) tienen la principal ventaja de que han extraído información directamente del 3D, pudiendo aportar información adicional (presencia de obstáculos, dirección de la escena) además de que son icónicas, no necesitando tanta resolución fotométrica y espacial para transmitir la información obtenida de manera eficiente y segura.

El primero de ellos, presentado en [14], consiste en principalmente representar el suelo como si fuese un tablero de ajedrez, con baldosas negras y blancas, de manera que se transmite sensación de profundidad y avance al moverte, sensaciones difíciles de transmitir con otras representaciones clásicas. Además, la rejilla y las líneas del horizonte del techo están orientadas según las direcciones de la escena. Se representa únicamente el polígono del suelo sobre el cual se puede caminar, es decir, que permita movimiento libre hasta encontrar un obstáculo que lo impida. El procedimiento de generación del polígono de suelo libre puede consultarse en [14]. Esta representación tiene como principales ventajas el explícitamente señalar hasta dónde se puede avanzar mientras que simultáneamente se transmite sensación de profundidad y de orientación en la escena.

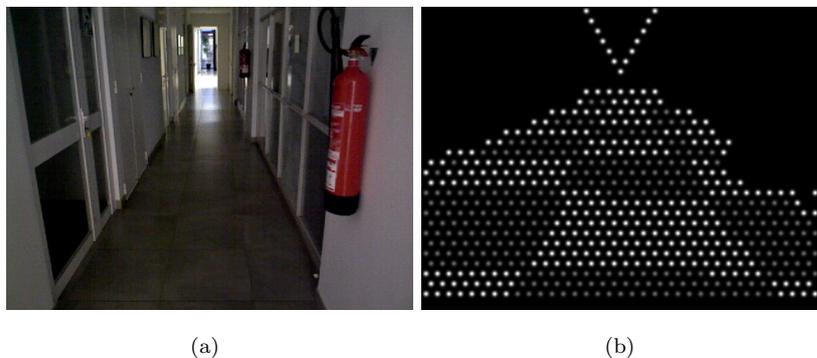


FIGURA 2.10: Primer modo de representación del SPV: tablero de ajedrez.

El segundo modo de representación proyecta en la imagen de fosfenos los objetos y paredes detectados mediante la segmentación de la escena. Los planos y agrupaciones de puntos segmentados se clasifican según su orientación, con el fin de iluminarlos de diferente forma. Las agrupaciones de puntos y planos horizontales se representan con el nivel de intensidad más alto, los planos verticales se pintan en gris y el suelo no se ilumina para diferenciarlo con claridad de elementos de la escena. Esta representación tiene la principal ventaja de permitir percibir formas y reconocer objetos de la escena, como personas, mesas o puertas, a la vez que el suelo y por tanto zona de movimiento

libre está implícitamente delimitada. Este será el modo de representación utilizado por defecto en el resto del trabajo.

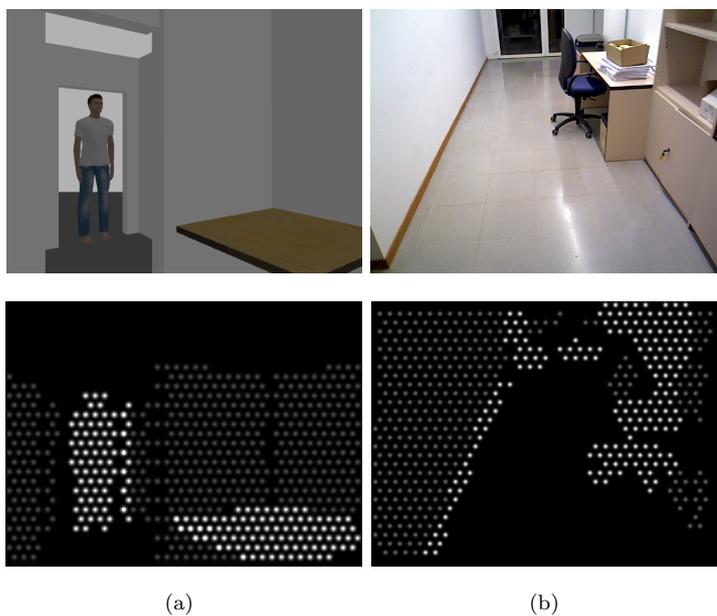


FIGURA 2.11: Segundo modo de representación del SPV: representación de objetos.

## Capítulo 3

# Sistema de guiado

El objetivo principal de este proyecto es desarrollar un sistema de guiado para ciegos integrado en un SPV. Para ello es necesario resolver dos grandes problemas: la obtención del camino que se ha de seguir para poder dirigir a la persona y el desarrollo de métodos para mostrar dicho camino en una representación apta para sistemas de visión protésica. Todo ello, además, ha de hacerse garantizando en todo momento la seguridad del individuo. Asumiremos mapa conocido, que permitirá calcular un camino para llegar a un objetivo, y se considerará la presencia de obstáculos que deberán ser evitados modificando la trayectoria.

En este capítulo se tratan ambos problemas. En primer lugar desarrollaremos el problema de navegación, dentro del cual se incluye la localización dentro del mapa y la planificación de la trayectoria a seguir. Puesto que se trata de un sistema complejo con muchas tareas que coordinar, dedicaremos especial atención a aclarar los sistemas de referencia involucrados. En segundo lugar, se proponen dos modos de representación de ayuda para la navegación con prótesis visuales que consiguen plasmar indicaciones visuales útiles para que el sujeto consiga llegar a su destino de forma segura y eficaz.

### 3.1. Navegación

La navegación es el conjunto de metodologías que hacen posible el movimiento de un robot, o en el caso que nos ocupa una persona, a través de un entorno de forma segura. Para que esto sea posible es necesario coordinar las diferentes tareas involucradas en este proceso. Las tareas más importantes son la localización del usuario en el mapa y la planificación de una trayectoria para llegar a un objetivo. En el caso de la robótica, la navegación culmina mandándole órdenes al robot para que éste se mueva en la dirección calculada. En el caso de personas buscaremos la manera de comunicar la información obtenida, en este caso mediante una prótesis visual, lo cual se trata en la Sección 3.2.

### 3.1.1. Localización en un mapa

Se conoce como localización al problema de determinar la posición y orientación de un sujeto relativa al mapa de un entorno conocido [21]. En este trabajo asumiremos que tenemos un mapa 2D (en planta) del edificio en el cual se va a navegar. En este contexto, hablando de navegación en interiores (donde el GPS no funciona), se han propuesto alternativas, por ejemplo, con RFID [22, 23] o Bluetooth [24]. En nuestro caso, dado que el elemento central de nuestro sistema es una cámara, se podrían usar marcadores [25] o incluso las propias señales dentro de un edificio [26] para ayudar en la localización.

Todas estas propuestas requieren escenarios específicamente preparados para esta tarea. Una de las grandes ventajas de trabajar en entornos simulados es que nos permite conocer la localización del usuario en el mapa en todo momento, por lo que en los experimentos de simulación este problema no es relevante. No obstante, para los experimentos en real utilizamos un sistema de SLAM [17] que nos permite localizarnos siempre y cuando previamente la escena haya sido mapeada. En la Sección 4.3.2 se detalla cómo se han implementado estos métodos.

Tanto el sistema de localización (con su mapa de localización 3D) como el de navegación (con su mapa 2D) deben de tener una referencia común que permita transformar la información de un sistema de referencia a otro, esto es, que permita expresar el mapa de un sistema en la referencia del otro. A continuación aclararemos los sistemas de referencia del sistema.

### 3.1.2. Sistemas de referencia

Un sistema de referencia es un conjunto de convenciones utilizadas para determinar la posición y orientación de un objeto en el espacio. Es necesario conocer respecto a que sistema de referencia están medidos los datos de las diferentes fuentes de información del sistema para poder realizar operaciones entre ellas. En este proyecto se ha definido una serie de ejes de referencia para hacer posible la localización de la cámara en el mapa.

En un sistema de coordenadas ortogonales existe un giro y una traslación que relaciona la medida de los dos sistemas de coordenadas. Esta operación se define con la matriz de rotación y traslación  $\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$  donde  $\mathbf{R}$  es la matriz de rotación y  $\mathbf{t}$  la de traslación.

La nomenclatura utilizada para describir estas matrices es  ${}^{REF1}\mathbf{T}_{REF2}$ , la cual define la rotación y traslación que hay que aplicarle a un punto cuyas coordenadas están en la  $REF2$  para proyectarlas en  $REF1$ . En la Figura 3.1 se representan los sistemas de referencia y las transformaciones utilizadas en este proyecto.

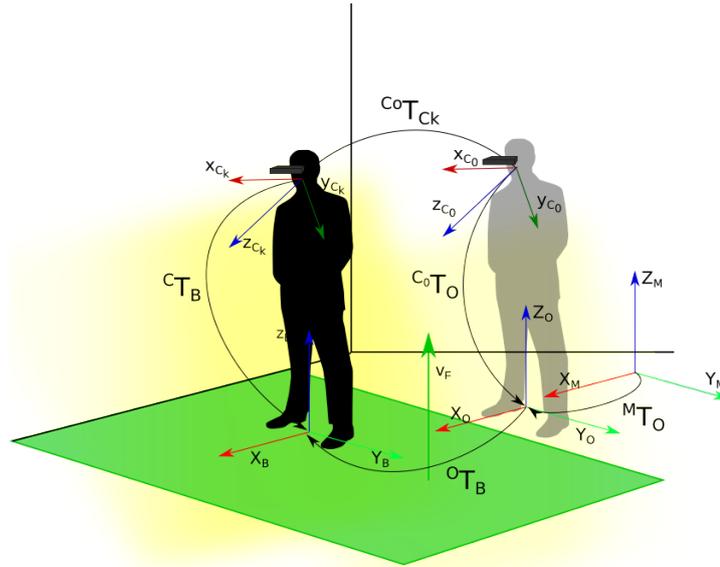


FIGURA 3.1: Ejes de referencia del sistema.

Denominaremos de forma general  $C$  a la referencia de la cámara, en particular  $C_t$  a la posición de la cámara en el instante  $t$ . Si tenemos un sistema de SLAM u odometría visual, denominaremos  $C_0$  a la referencia de la cámara en el instante en que se lanza el algoritmo ( $t = 0$ ) y  $C_k$  a la referencia de la cámara en el instante actual ( $t = k$ ). El sistema de SLAM u odometría nos irá proporcionando continuamente la transformación  ${}^{C_0}\mathbf{T}_{C_t}$ . En la Sección 2.2 describimos el cambio de referencia de cámara ( $C$ ) a suelo (*base*,  $B$ ), por tanto  ${}^C\mathbf{T}_B$ . Concretamente, la referencia de suelo  $B$  en el instante  $t = 0$  se llama *odom* ( $O$ ), y por tanto la transformación se llama  ${}^{C_0}\mathbf{T}_O$ .

La referencia del mapa 2D para la navegación se llama *map* ( $M$ ), y tendrá una posible traslación sobre el plano de suelo y rotación en  $z$  con respecto a  $O$ . La transformada  ${}^M\mathbf{T}_O$  sirve para relacionar el sistema de referencia del mapa ( $M$ ) y la referencia inicial de la cámara en el suelo ( $O$ ). Dicha transformación es fija y se supone conocida. Con este cambio de referencia a  $M$ , las coordenadas  $(x, y)$  nos dan información de la posición de los elementos sobre el plano de suelo, que a su vez es el plano 2D para la navegación.

La referencia  ${}^O\mathbf{T}_B$  define la proyección de la cámara sobre el plano de suelo en el instante actual con respecto a la referencia de suelo inicial y puede calcularse al conocerse  ${}^{C_k}\mathbf{T}_O$ . Con estas dos últimas transformaciones podemos obtener  ${}^C\mathbf{T}_B$ , que nos da la posición del suelo actual con respecto a la cámara actual. Dado que esta transformación puede observarse y calcularse según lo descrito en la Sección 2.2, esto puede utilizarse para corregir la posible deriva que el método de SLAM u odometría puede haber ido acumulando.

### 3.1.3. Planificación de trayectorias

Una vez comentada la localización en el mapa y las referencias del sistema, el siguiente paso es calcular la trayectoria a seguir. Se han utilizado dos planificadores para el cálculo de la trayectoria. El primero de ellos es un planificador global, el cual a partir de la información proporcionada por el mapa, es capaz de calcular el camino óptimo para alcanzar el objetivo. El segundo es un planificador local, encargado de recalculer la trayectoria en el caso de encontrar algún obstáculo durante el seguimiento de la trayectoria global.

Para hacer posible el funcionamiento de estos planificadores, es necesaria la creación de un mapa descriptivo del entorno en el que se recoja la información de los obstáculos de la escena. Para ello se utilizan los llamados mapas de coste. A continuación se describe brevemente el funcionamiento de los planificadores utilizados, así como de los mapas de coste.

#### 3.1.3.1. Planificador global

El planificador utilizado se basa en el algoritmo Dijkstra [27] para obtener el camino óptimo hasta el destino. Este algoritmo propone marcar inicialmente todos los estados como no visitados, e ir evaluando los costes para llegar a los estados vecinos desde el inicial. Es decir, sumar el coste del actual más el coste incremental para ir desde el actual hasta el vecino. En el caso de que uno de los vecinos ya hubiese sido evaluado anteriormente, el coste asociado sería el menor de ellos. La búsqueda finaliza cuando se visita el estado final.

#### 3.1.3.2. Planificador Local

El planificador utilizado es el DWA (Dynamic Window Approach) [28]. Este algoritmo utiliza la información de los obstáculos de la nube de puntos 3D proyectada sobre un mapa 2D, para calcular el camino más óptimo con el que esquivarlos. En particular, la nube de puntos se transforma con  ${}^M\mathbf{T}_{C_k}$  y se eliminan los puntos próximos al suelo ( $z \approx 0$ ) y aquellos muy elevados por encima de el usuario (e.g.  $z > 2m$ ). La nube resultante se proyecta sobre el plano 2D obviando la componente en  $z$ . Con los obstáculos ya en la referencia global, la trayectoria global se modifica en función de: la distancia a la que se encuentran los obstáculos, la distancia al objetivo y la alineación del sujeto respecto a este.

#### 3.1.3.3. Mapas de coste

Para tener una descripción del entorno por el que se va a navegar y conocer la ubicación de los obstáculos en el mismo, se divide en celdas a las que se les asigna un coste en

función del tamaño del obstáculo y la distancia hasta él. Esta información es utilizada por el planificador de trayectorias para saber que zonas del mapa son navegables y cuales son las mejores para alcanzar el objetivo y esquivar los obstáculos.

En este caso se cuenta con dos mapas de coste. El primero de ellos recoge la información del entorno proporcionada por un mapa capturado previamente. El segundo es un mapa local para llevar a cabo la evitación de obstáculos. Los costes de las celdas de este último vienen asignadas por la información de profundidad captada por la cámara, que es proyectada sobre el plano del suelo a modo de escáner láser 2D. Esto es posible debido a que previamente se han calculado las transformaciones  ${}^C\mathbf{T}_B$  y  ${}^O\mathbf{T}_B$ , que definen la localización de la cámara respecto al suelo.

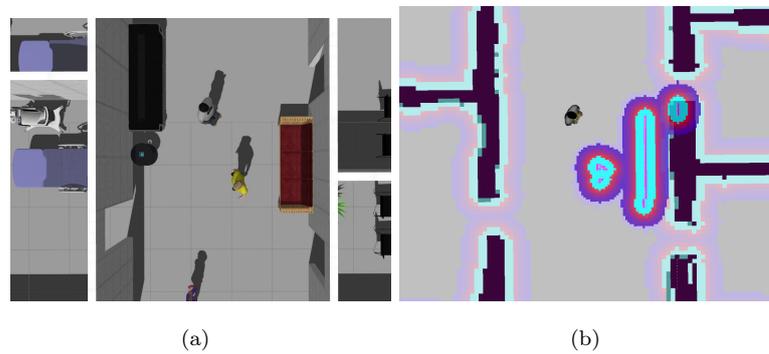


FIGURA 3.2: Costmaps: (a) Entorno de simulación con obstáculos cercanos. (b) Visualización en RVIZ del mapa de coste local.

En la Figura 3.2 se puede observar la representación de los obstáculos cercanos al sujeto en el mapa de coste local. Los puntos que definen el obstáculo, son proyectados sobre el mapa de coste, marcando la celda en la que se sitúa cada uno de ellos, y las celdas cercanas en función de un radio de inflado. Este define una distancia de seguridad para que la trayectoria calculada no pase demasiado cerca de los obstáculos.

### 3.2. Representación de la trayectoria con fosfenos

A continuación se presentan los dos métodos de guiado desarrollados para plasmar la dirección de movimiento en el SPV. Dado que en los sistemas de visión protésica la resolución espacial y fotométrica es muy baja, se ha tenido en cuenta la propuesta de representaciones simples que sean capaces de comunicar las pistas para la navegación sin obstruir sino más bien complementar la información útil que ya de por sí proporcionan los modos de representación existentes. Se ha prestado atención a que estas representaciones del camino sean claras, sin dar lugar a confusión, y sin ambigüedades.

### 3.2.1. Camino de guiado

El primero de ellos consiste en plasmar en el plano del suelo de la imagen, el camino que la persona debe seguir para alcanzar el objetivo. Para comenzar con la planificación de trayectorias y navegar con este método, se debe conocer donde se encuentra la ubicación a la que se quiere ir en el mapa. El planificador de trayectorias calcula una secuencia de puntos 2D en la referencia de coordenadas del mapa que lleva hasta el objetivo.

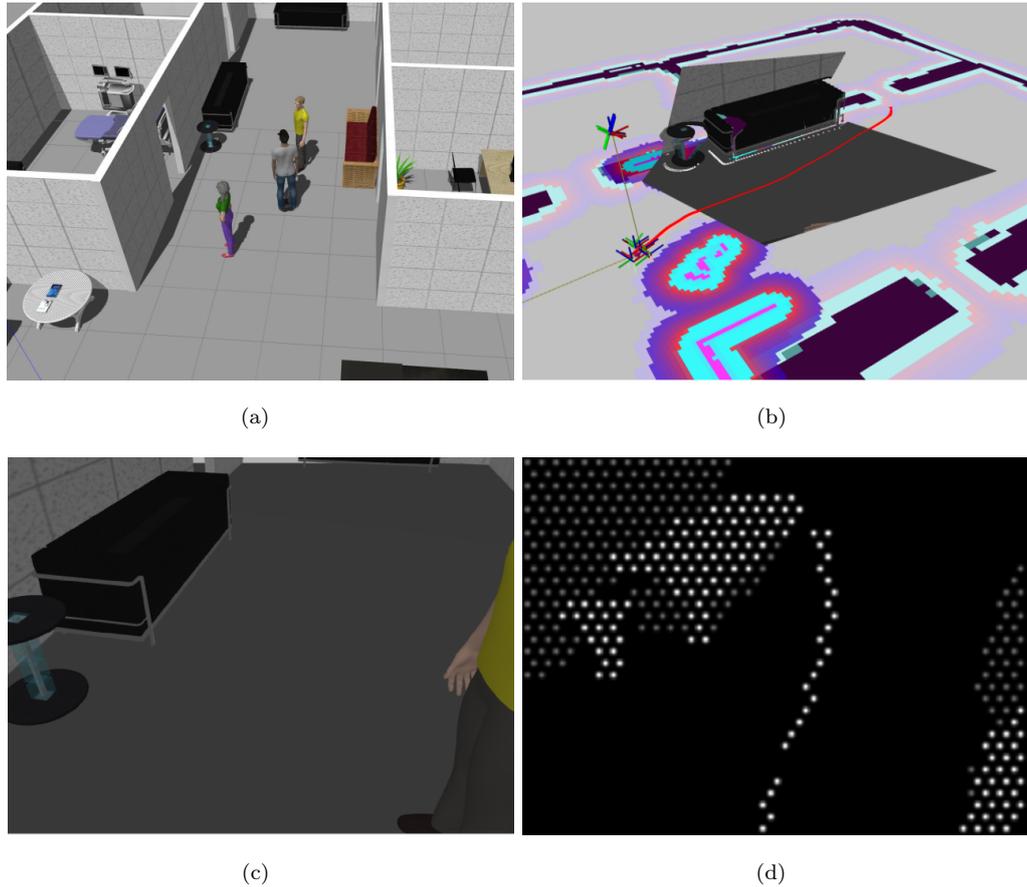


FIGURA 3.3: Método de camino de guiado: (a) Entorno de simulación Gazebo. (b) Imagen de RVIZ en la que se puede ver la trayectoria calculada representada como una línea roja en el mapa. (c) Imagen captada por la cámara RGB. (d) Imagen generada por el SPV en la que se ha integrado la trayectoria calculada sobre el plano del suelo.

Para representar la secuencia de puntos en la imagen (Figura 3.3), se han proyectado sobre la malla de fosfenos los puntos 3D mediante la matriz de proyección (2.3) y la matriz  ${}^M\mathbf{T}_C$  que define la localización de la cámara respecto al mapa. A estos puntos se les ha asignado un valor de intensidad alto para diferenciarlos claramente del resto de los objetos de la escena. Además, se ha optado por representarla en conjunto con el modo de representación de planos y objetos que por defecto deja el suelo con los fosfenos apagados y por tanto es más fácil distinguir claramente la línea de la trayectoria.

En la Figura 3.3 se puede observar un ejemplo de funcionamiento en un entorno de

simulación, donde el usuario planea ir a la habitación del fondo a la izquierda (Figura 3.3 (a)). En Figura 3.3 (b) se pueden ver los sistemas de referencia (cámara, base) del usuario así como la trayectoria representada en rojo que es la que ha de seguir, evitando obstáculos como es el caso de otro individuo. En Figura 3.3 (c) es la imagen a color que está actualmente viendo el sujeto, mientras que en Figura 3.3 (d) está la imagen del SPV de fosfenos con el modo de representación de planos y obstáculos y la línea de trayectoria fácilmente observable.

### 3.2.2. Brújula

El segundo método está basado en el sistema de guiado utilizado en algunos videojuegos, como los de simuladores de vuelo, en los que la dirección de movimiento a seguir para alcanzar un objetivo viene dada por una brújula virtual. En particular, se trata de unos indicadores con forma de flecha en la parte superior e inferior de la imagen que se mueven de forma simultánea respecto a la vertical e indican la desviación en orientación actual del usuario respecto de trayectoria. Para seguir la dirección calculada por el planificador de trayectorias de forma correcta, el sujeto debe corregir su orientación de tal forma que los marcadores se mantengan en la parte central de la imagen. Para representarlo con fosfenos, se han plasmado dos marcadores en forma de flecha, utilizando tres fosfenos blancos en la parte superior e inferior de la imagen.

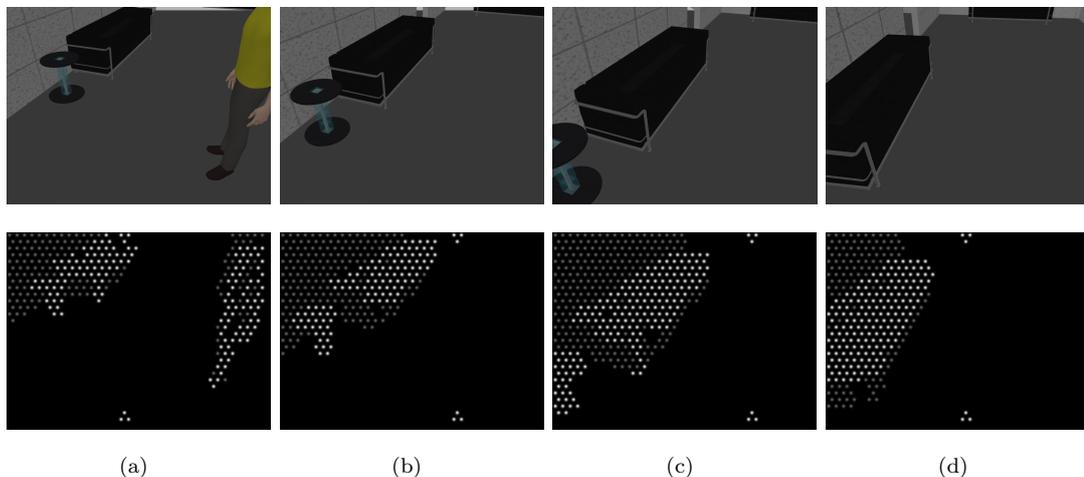


FIGURA 3.4: Secuencia de imágenes con el sistema de guiado de brújula. En la fila superior se encuentran las imágenes captadas por la cámara y en la inferior las generadas con el SPV.

En la Figura 3.4 se ha representando una secuencia de imágenes en los que se muestran los movimientos realizados por el sujeto para seguir la trayectoria calculada por el planificador. En ellas se puede apreciar el desvío de los marcadores hacia los laterales de la imagen en función de la orientación de la trayectoria con respecto al sujeto: conforme avanza, progresivamente va virando a la derecha para conseguir centrar los marcadores.

Nótese que, aun en el caso de que el modo de representación utilizado aparte de la brújula ya esté poniendo a esos fosfenos blancos, el disponer de flechas arriba y abajo ayuda a evitar que el usuario se pierda.

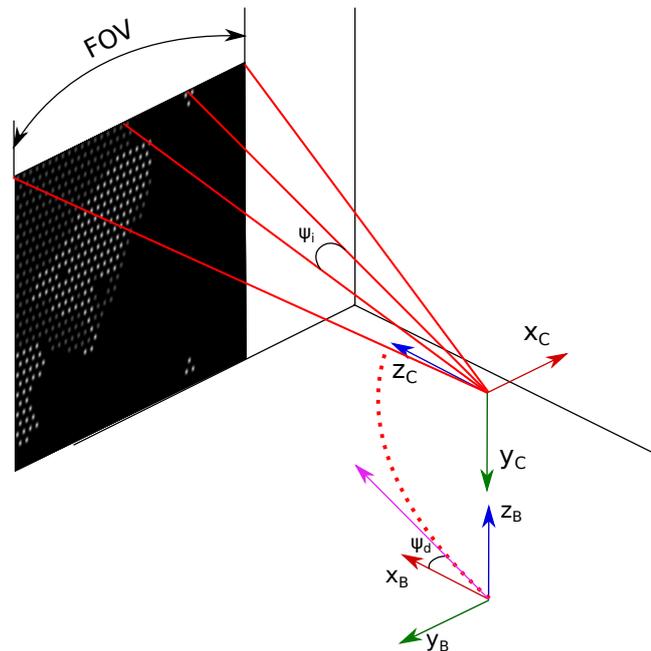


FIGURA 3.5: Movimiento de los marcadores de dirección respecto de la parte central de la imagen en función de el desvío en el seguimiento de la trayectoria. La línea de puntos roja representa la trayectoria calculada por el sistema de guiado. La línea morada es la pendiente de la trayectoria en la zona próxima al sujeto.

Para representar las flechas en la posición adecuada, en primer lugar se calcula el ángulo *yaw* ( $\Psi_d$ ) que indica la rotación en  $z_B$  que ha de tener el usuario para centrarse (es decir, tener su eje  $x_B$  alineado con la pendiente de la trayectoria). Para calcular la pendiente de la trayectoria se puede escoger un punto lo bastante próximo de la misma. De similar manera, se calculan los ángulos  $\Psi_i$  de cada rayo  $\mathbf{r}_i$  de los  $i$  fosfenos de la segunda fila situados en posiciones  $\mathbf{p}_i$  de la imagen de fosfenos. Se iluminará el fosfeno con ángulo  $\Psi_i$  más próximo a  $\Psi_d$ . El resto de componentes de la flecha y los equivalentes de la parte inferior son fácilmente computables al conocerse el número de fosfenos y el tipo de mapa.

## Capítulo 4

# Implementación y entorno de simulación

El sistema de guiado para prótesis visuales presentado en este trabajo tiene numerosas tareas que tienen que estar funcionando simultáneamente: procesamiento de la nube de puntos, localización en un mapa, planificación de trayectorias y generación de la imagen de fosfenos con el modo de representación escogido y la trayectoria obtenida. Además, se ha enfocado el problema para que funcione tanto con imágenes reales tomadas con una cámara RGB-D como en un entorno de simulación. Todo esto supone un reto a nivel de implementación, pues los numerosos paquetes de software involucrados han de comunicarse entre ellos y mandarse la información necesaria de unos a otros a su debido tiempo. Para comunicar y programar los diferentes sistemas de este proyecto se ha utilizado ROS (Robotic Operating System), un conjunto de librerías y herramientas desarrolladas en su gran mayoría en C++ centradas en el desarrollo de aplicaciones robóticas. En ellas se pueden encontrar desde drivers para sensores hasta algoritmos de navegación y localización del estado del arte.

Un esquema general de cómo se comunican las distintas partes puede encontrarse en la Figura 4.1. En la parte superior se muestra el bloque de obtención de imágenes, sea con una cámara real o simulada. En la parte central se muestra: por un lado el bloque de localización que bien puede venir dada por una localización simulada en los entornos virtuales, o un método de localización como es ORB-SLAM2 y por otro lado el servidor de Transformadas encargado de hacer un seguimiento de las referencias de coordenadas del sistema. Por último en la parte inferior se encuentran los paquete de Navegación y el SPV.

A lo largo del capítulo se comenta más en detalle cómo se ha llevado a cabo la implementación de cada bloque y cómo se comunican unos con otros. En primer lugar, se proporcionarán unas nociones de ROS para entender el sistema de comunicaciones. A continuación se describirá el trabajo realizado con el entorno de simulación para crear

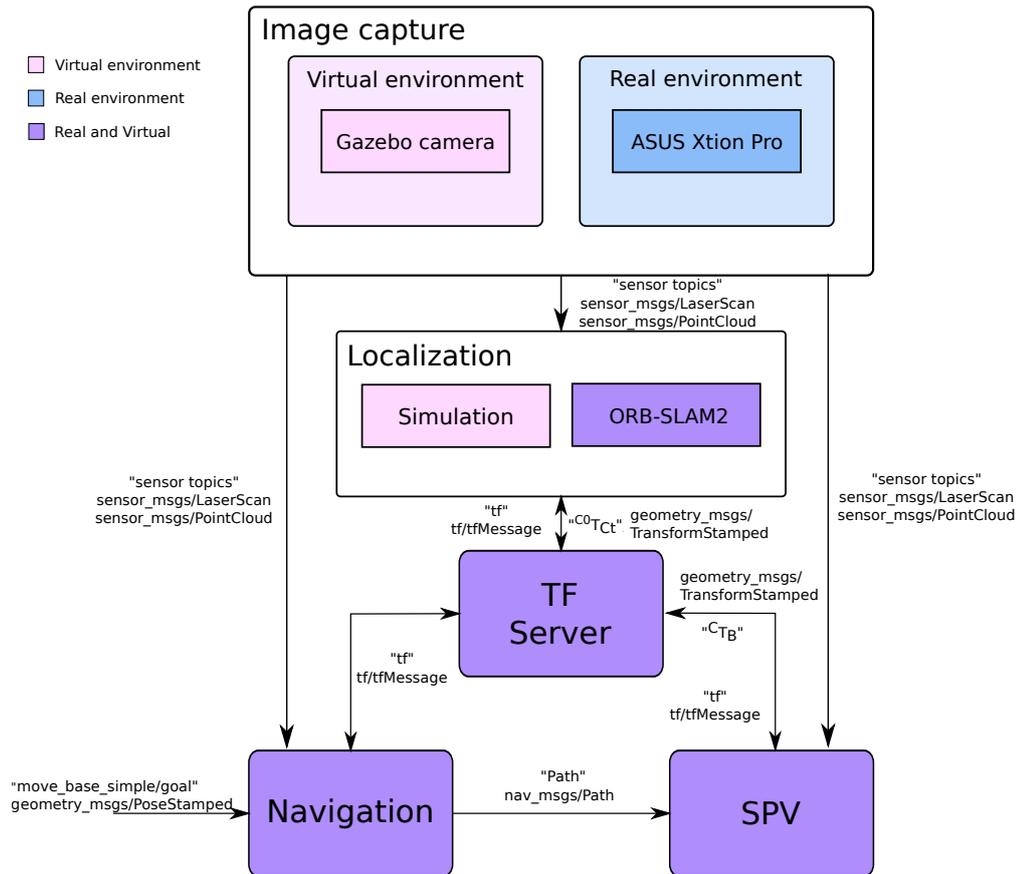


FIGURA 4.1: Esquema general de los paquetes utilizados.

entornos y modificar el robot para asemejarlo a una persona. Por último, se comentará cómo se ha implementado e integrado la localización, la navegación y el SPV para el sistema de guiado.

### 4.1. ROS (Robotic Operating System)

ROS es un entorno de trabajo orientado a aplicaciones robóticas que se ha utilizado en este trabajo puesto que es una herramienta muy apropiada para integrar distintos paquetes de software que tengan que comunicarse entre sí. Además, debido a la larga tradición de investigación en robótica, existen numerosos paquetes ya implementados que resuelven tareas de utilidad para este trabajo. Todo ello, además, se trata de software libre.

Se divide principalmente en dos partes: El sistema de ficheros y el de cómputo. El sistema de ficheros de ROS está principalmente organizado de la siguiente forma:

- **Paquetes:** Los paquetes son la unidad principal de organización del software en ROS. Un paquete puede contener la definición de procesos (nodos), diferentes li-

brerías, *datasets*, ficheros de configuración, etc. Por ejemplo, el SPV que utilizamos en este trabajo es un paquete.

- **Mensajes:** Es la forma de definir la estructura de datos utilizada en el sistema. Un mensaje puede estar formado por tipos de datos básicos (entero, doble, etc.) o por estructuras mas complejas como puede ser la definición de la posición y orientación de un robot. En particular, algunos tipos de mensajes utilizados en este proyecto son:
  - **sensor\_msgs:** Son los mensajes con datos que provienen de un sensor, como una cámara. Pueden ser, por ejemplo, una imagen, una nube de puntos (*sensor\_msgs/PointCloud*), un escáner láser.
  - **geometry\_msgs:** Mensajes con variables geométricas como pueden ser puntos, vectores o poses (transformaciones).
  - **nav\_msgs:** Se incluyen todos los mensajes que tienen que ver con la navegación, como los mapas de coste o la odometría (que incluye pose y *twist*).

El sistema de computo de ROS está organizado en procesos que se comunican con una unidad de control superior que conoce toda la información del sistema. Los principales conceptos que definen el sistema de cómputo de ROS son los siguientes:

- **Master:** El ROS Master es el encargado de monitorizar y registrar el nombre de todos los procesos activos en el sistema, así como el nombre de los mensajes que se están intercambiando.
- **Nodos:** Los nodos son procesos encargados de realizar las tareas de computación que les hayan sido asignadas. Por ejemplo en un sistema robótico, un nodo se encarga de leer la información de un sensor y de hacerla visible al resto de nodos del sistema, otro se encarga de planificar trayectorias y otro de localizar al robot en el mundo.
- **Servidor de parámetros:** Se encuentra localizado en el Master. En él se almacenan todos los parámetros de configuración del sistema.
- **Tópicos:** Los mensajes en ROS son tratados mediante un sistema de transporte basado en la subscripción y publicación. Un nodo publica mensajes sobre un tópico con un nombre que identifica el contenido del mensaje. Si un nodo está interesado en conocer está información se suscribe a ella para poder utilizarla. Por ejemplo, cuando se lanza el nodo de la cámara, se empiezan a publicar *sensor\_msgs/PointCloud* en el tópico */camera/depth\_registered/points*. El nodo del SPV se suscribe a dicho tópico para poder utilizar esa información en su ejecución.

Uno de los paquetes más importantes de ROS es `tf`. Sirve para llevar un seguimiento de las transformaciones del sistema y mantener la relación entre sistemas de referencia en un árbol de transformadas. Este árbol de transformadas puede ser consultado por cualquier nodo para conocer la posición relativa entre las distintas partes del sistema. Alternativamente, los nodos pueden publicar transformaciones para actualizar el árbol de transformadas. El paquete `tf` tiene un tipo de mensajes propios `tf/Message`.

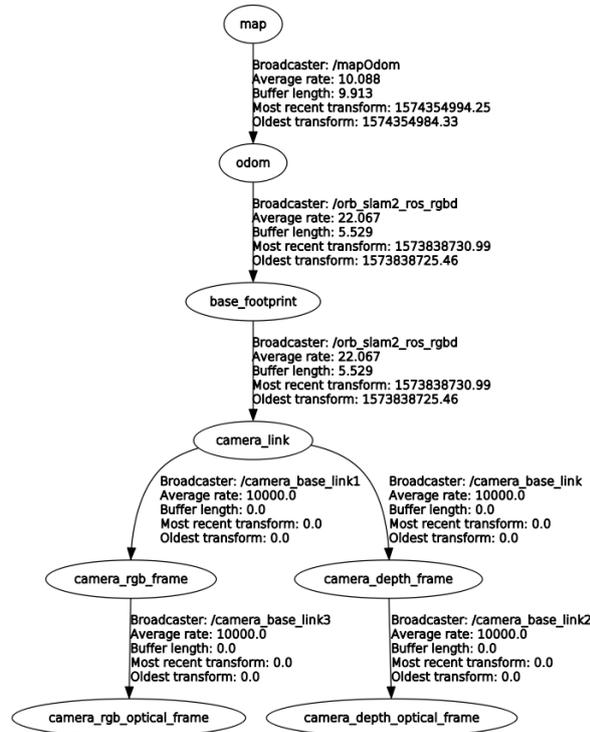


FIGURA 4.2: Árbol de transformaciones del sistema. Con el se puede comprobar que todas las referencias están siendo publicadas y existe una relación entre ellas.

En nuestro proyecto, el árbol de transformadas (Figura 4.2) adquiere una forma similar a la descrita en la Sección 3.1.2: `map` ( $M$ ) está conectado con `odom` ( $O$ ) y este a su vez con `base footprint` ( $B$ ) que se conecta con la cámara ( $C$ ) y esta respectivamente se divide en las referencias específicas de la misma (cámara de color, cámara de profundidad).

## 4.2. Entorno de simulación

El funcionamiento del sistema de guiado va a ser testeado principalmente en un entorno virtual. Se ha utilizado Gazebo como simulador [29]. Se trata de un simulador de software libre que ofrece la posibilidad de crear entornos ricos en detalles así como modelos de robots con masa, velocidad y rozamiento y otros atributos que permiten asemejar estos modelos a la realidad a la hora de desplazarse, colisionar o realizar otro tipo de acciones.

La principal ventaja de Gazebo con respecto a otros simuladores es que está perfectamen-

te integrado con ROS, utiliza el mismo tipo de comunicación a través de tópicos y por tanto facilita una integración cómoda con el resto de procesos. Además, las simulaciones de sensores producen mensajes del mismo tipo que los sensores reales, de manera que el mismo código puede ser usado indistintamente de cuál sea la fuente de información. Uno de los principales usos de Gazebo es el desarrollo de algoritmos de robótica en entornos de simulación controlados, de manera que puedan luego ser aplicados a sistemas reales sin tener que modificar el algoritmo en sí.

Un entorno completo de simulación en Gazebo es un conjunto de modelos y sensores. Estos modelos se pueden clasificar en dos bloques: el suelo y los edificios representan un conjunto de modelos estacionarios, mientras que los robots y otros objetos representan modelos dinámicos. Para este trabajo se han creado escenarios virtuales (modelos estacionarios), así como un modelo dinámico de humano con el fin de simular los movimientos que realizaría una persona. El objetivo de esta parte es poder evaluar el funcionamiento del simulador de visión protésica de la manera más rápida y cómoda como previo paso a llevarlo a un sistema real.

#### 4.2.1. Creación del escenario virtual

Un modelo en Gazebo, es cualquier objeto que tenga una representación física, desde geometrías simples hasta robots complejos. El modelo puede estar compuesto por al menos un sólido rígido al que se le pueden integrar articulaciones y sensores. En este caso de estudio, interesa crear un entorno que simule un edificio público, con varias habitaciones en el que poder guiar a una persona con discapacidad visual.

Gazebo cuenta con una herramienta de creación de entornos, llamada *Building editor*, que permite crear la estructura de las paredes que forman un edificio dentro del propio simulador.

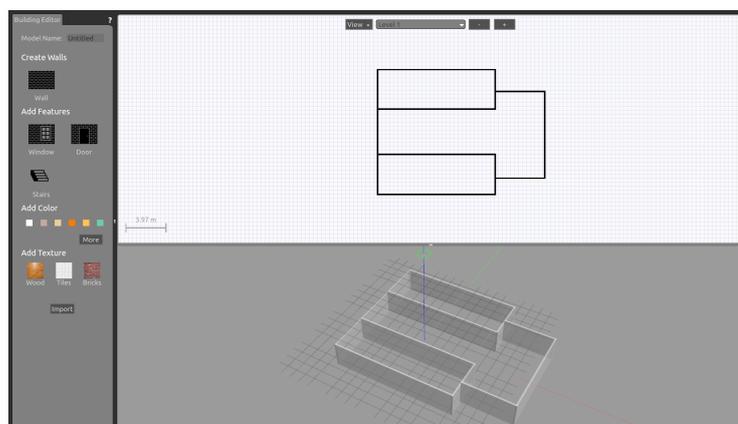


FIGURA 4.3: *Building editor*: Herramienta de creación de modelos de Gazebo.

Los muebles y objetos se han añadido posteriormente utilizando una librería de modelos

3D [30], que cuenta con numerosos objetos como sillas, mesas, ordenadores o libros con una gran resolución; y de ShapeNet [31], dataset del cual se han obtenido diferentes modelos 3D del mobiliario de un hospital y se han adaptado al formato utilizado por Gazebo para cargarlo. Este último paso es muy importante ya que, para que la física funcione en Gazebo, todos los objetos han de tener masa e inercia.



FIGURA 4.4: Modelo de la planta de un hospital.

#### 4.2.2. Creación del modelo humano de simulación

Los elementos que forman un modelo en Gazebo se denominan *links* y normalmente está definido por tres partes: la visual, la de colisiones y la inercia. La parte visual es la que se representa en el interfaz gráfico, en este caso se carga un modelo 3D con formato *.dae* del robot. La parte de colisiones, se suele definir con una geometría simplificada con la que el motor físico realiza las operaciones para ahorrar tiempo de cómputo en lugar de utilizar toda la malla 3D. Y por último en la parte de inercia se puede definir la masa y la matriz de inercia del elemento.

Una vez definidos todos los links que forman el robot, hay que definir la conexión existente entre cada uno de ellos utilizando los llamados *joints*. Esta propiedad permite definir la posición entre ambos elementos, así como los grados de libertad de la articulación que los une.

Debido a la complejidad de crear un modelo humano en este simulador desde cero, se ha creado uno que parte de la implementación de un robot diferencial llamado Turtlebot que además ya cuenta con una cámara RGB-D. Dicho modelo se ha modificado para darle una apariencia y una manera de moverse más parecida a la humana. En particular, se han hecho las siguientes modificaciones:

- Para añadirle apariencia humana, se ha añadido un nuevo *link* que va fijo unido a

la base del robot (*base footprint*) y que sólo aporta una malla de visualización de un humano (Figura 4.5 (a)).

- Para simular el punto de vista humano, se ha modificado la posición del *camera link* de forma que la cámara se sitúa a la altura de los ojos.
- Se ha modificado la *joint* del *camera link*, de modo que admite una rotación en el eje *y* que simula el movimiento de la cabeza de arriba a abajo. Para ello se le ha añadido una transmisión a la junta, de manera que se puede controlar la posición de la cabeza simplemente escribiendo en un tópic.
- Se ha añadido un movimiento de traslación en la cámara para simular el movimiento de la cabeza producido al andar. Para ello, se han añadido dos nuevos *links* con traslación entre ellos, que simulan movimiento de arriba a abajo y de izquierda a derecha.

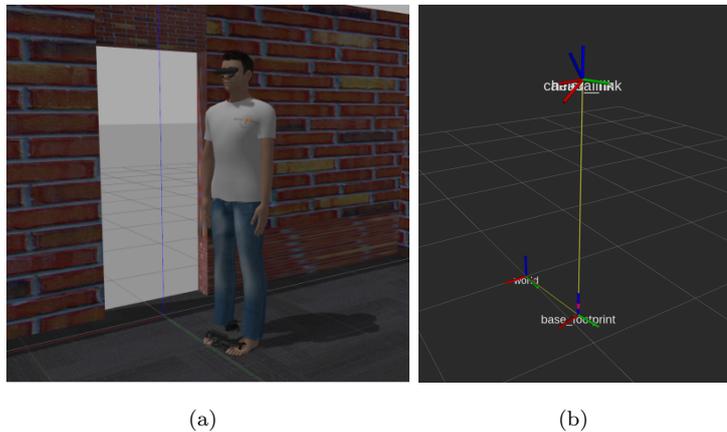


FIGURA 4.5: Modelo del robot con la cámara RGB-D integrada: (a) Modelo 3D del robot en Gazebo, (b) Ejes de referencia principales del robot.

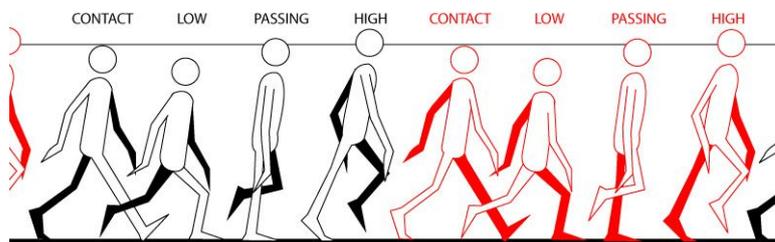


FIGURA 4.6: Movimiento vertical de la cabeza durante la marcha.

Para otorgar un movimiento realista, en este último punto se han utilizado los datos obtenidos en un estudio [32] en el que se pidió a varios sujetos andar hacia delante con la vista al frente para evitar movimientos con la cabeza.

La simulación de este movimiento, se ha realizado añadiendo unos ejes de referencia adicional al modelo del humano ligados a la cabeza, donde se encuentra situada la cámara. A esta referencia se le aplica un desplazamiento en los ejes **Y** y **Z** (Fig. 4.6), con el fin de producir un movimiento en las imágenes que simulen el movimiento de la cabeza al andar. El desplazamiento de estos ejes se ha modelado con la siguiente fórmula:

$$d(t) = a_0 + \sum_i^N a_i \sin(i\omega_o t + \phi_i); \omega_o = \frac{2\pi}{T} \quad (4.1)$$

Donde  $d(t)$  es la componente periódica del desplazamiento;  $a_0$  es el valor medio;  $a_i$  y  $\phi_i$  son la amplitud y la fase del armónico  $i$ -ésimo;  $T$  es el periodo de la marcha y  $N$  es el número de armónicos. En este caso el periodo de la marcha es proporcional a la velocidad con la que se mueve el robot.

En la Figura 4.5 (b), se muestra el esquema de las principales referencias asociadas al robot. Todas las referencias de coordenadas que definen la geometría del robot, están asociadas a la llamada *base footprint*, la referencia a la altura del suelo, a partir de la cual se puede obtener la localización del resto de referencias del robot.

Otra de las ventajas que supone Gazebo y el basar el modelo del humano en una Turtlebot es que viene implementado con un control de movimiento. Se trata de un robot diferencial, que puede moverse hacia delante y atrás y girar. Todo esto puede controlarse desde un mando o con el teclado, de modo que se pueda mover por el entorno creado. Se ha añadido también, para su manejo, el giro de la cámara para mirar hacia arriba y hacia abajo, de modo que puede observarse la escena con mayor libertad.

### 4.3. Localización y mapas

Para poder calcular las trayectorias que la persona ha de seguir para llegar a un objetivo, es necesario conocer el propio mapa y la localización del sujeto en el mismo. Este trabajo ha sido desarrollado pensando en que funcionase tanto en simulación (donde este problema se simplifica) como con imágenes reales. La tarea de localización se realiza de forma notablemente si el método está corriendo en simulación o en real.

A continuación se describe cómo se resuelven estos problemas en cada uno de los modos de funcionamiento.

#### 4.3.1. Entorno virtual

Una de las ventajas de trabajar en un entorno de simulación es que simplifica el problema de la localización, ya que el simulador permite conocer la localización de una referencia

de coordenadas determinada de manera exacta. De esta forma se facilita la evaluación de los algoritmos de guiado desarrollados en este proyecto.

En particular, en Gazebo existe el *plugin* Gazebo-ROS P3D que directamente publica en un tópic, la posición de cualquiera de los links del modelo con respecto del origen del mapa (referencia  $M$ ). En nuestro caso lo hacemos desde la referencia sobre el suelo, el link *base footprint*, lo cual nos da la transformación  ${}^M\mathbf{T}_B$  que también se publica mediante otro nodo en las tf para cerrar el árbol de transformadas desde  $M$  hasta  $C$ . De esta manera, tenemos a la persona en todo momento localizada en la referencia del mapa 2D sobre el que se va a navegar.

Un entorno creado según se ha descrito en la Sección 4.2.1 no nos proporciona ningún mapa 2D en planta para la navegación. Por lo tanto, lo tenemos que obtener nosotros mismos manualmente. Esto se ha realizado navegando con el modelo humano por el escenario y debido a que la localización en el mismo es exacta, se ha creado un nodo que genera un mapa del escenario mediante la proyección de la nube de puntos proporcionada por el sensor de profundidad sobre el plano del suelo a modo de escáner láser. En particular, primero la nube de puntos referenciada en la cámara se transforma a referencia absoluta del mapa con  ${}^M\mathbf{T}_C$  y se filtran los puntos próximos al suelo ( $z \approx 0$ ) y aquellos cuya coordenada  $z$  sea mayor que la altura de la persona para evitar que dichos puntos aparezcan en el mapa (ni suelo ni techo son obstáculos), y luego los puntos transformados se proyectan sobre el plano obviando la componente  $z$ . Este procedimiento para obtener el mapa 2D se hace antes de añadir obstáculos en la escena, para que así se pueda comprobar que la detección de obstáculos funciona también. En la Figura 4.7 se puede ver un ejemplo de mapa creado con este nodo, de uno de los escenarios virtuales

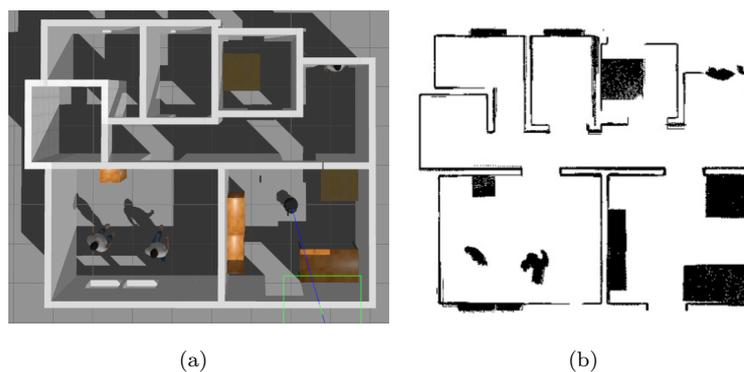


FIGURA 4.7: Mapa 2D del entorno de simulación: (a) Imagen de la planta del entorno. (b) Mapa 2D capturado con la información del sensor de profundidad.

### 4.3.2. Escenario real

Para resolver el problema de localización en un escenario real, lo haremos por medio de un sistema de SLAM (localización y mapeo simultáneos) y utilizaremos sólo la información que nos da la propia cámara RGB-D. En este proyecto se ha optado ORB-SLAM2 [17], una librería de SLAM en tiempo real para cámaras monoculares, estéreo y RGB-D, con la cual se puede calcular la trayectoria de la misma y una reconstrucción 3D dispersa del entorno. Esta librería se ha utilizado para generar un mapa 3D del escenario. Este mapa está formado por los puntos característicos de la escena identificados por el algoritmo, que son almacenados para localizarse respecto a ellos. Al estar utilizando el modo RGB-D, este mapa 3D está a escala.

Este tipo de algoritmos típicamente acumulan error durante la creación del mapa. Por ello se ha buscado que en el mapeo haya un *cierre de bucle*, es decir, se ha pasado por un lugar por el que ya se había pasado con anterioridad y el algoritmo lo ha identificado. Esto se hace con el fin de corregir el error acumulado en el mapa durante la trayectoria para que el mapa sea lo más fiel a la realidad posible, ya que un error de deriva muy elevado puede provocar fallos del método, como por ejemplo que el suelo pase a verse como un obstáculo al haberse desviado la posición relativa de la cámara con respecto al suelo.

Nótese que este mapa 3D no es el mismo que el mapa 2D necesario para la navegación. El uso de dos tipos de mapas diferentes se debe a que los planificadores y mapas de coste utilizados, están pensados para navegar con robots 2D sobre mapas de la planta del escenario. Por ello para poder integrar la localización proporcionada por este algoritmo en el sistema de guiado ha sido necesario definir las referencias de coordenadas (Sección 3.1.2) con las que proyectar la localización del sujeto y la información 3D de los obstáculos sobre los mapas 2D. Se estudió la posibilidad de combinar la información de la cámara RGB-D proyectada al suelo para la relocalización en el mapa 2D mediante un método de localización probabilístico, pero el escaso campo de vista y rango de la cámara RGB-D comparado con el de el láser 2D utilizado para construir el mapa, unido con el punto de vista diferente y en constante movimiento en tres dimensiones hacía que los métodos no consiguiesen funcionar debidamente.

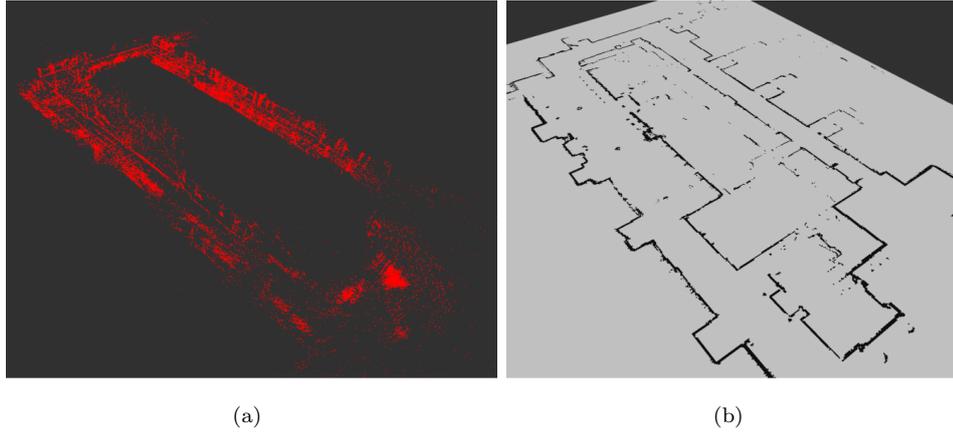


FIGURA 4.8: (a) Mapa 3D del escenario obtenido con ORB-SLAM2 (b) Mapa 2D del mismo escenario obtenido con cartographer [2].

Para obtener el mapa 2D de un escenario real necesario para la navegación se ha utilizado Cartographer. Se trata de un sistema de SLAM desarrollado por Google, que cuenta con una librería en ROS para la captura de mapas 2D. La creación de este mapa se ha llevado a cabo con un robot omnidireccional con 2 escáneres láser integrados. En la Figura 4.8 se muestran los mapas 3D y 2D de una escena real empleada en los experimentos.

El nodo de localización devuelve las transformaciones  ${}^O\mathbf{T}_B$  y  ${}^B\mathbf{T}_C$ , únicamente requiriendo como entradas las imágenes RGB y de profundidad, y la transformación inicial del suelo  ${}^O\mathbf{T}_{C_0}$ . En este momento la coincidencia de los mapas 2D con el inicio de la odometría,  ${}^M\mathbf{T}_O$ , ha de establecerse manualmente.

#### 4.4. Navegación y SPV

El sistema de navegación es el encargado de calcular y transmitir la trayectoria al SPV. El paquete de ROS utilizado en este proyecto como sistema de navegación es **Move base**. Éste es el encargado de calcular una trayectoria segura y libre de obstáculos a partir de la información de localización del sujeto, la nube de puntos de la escena proyectada sobre el plano del suelo y el mapa del entorno. Para ello utiliza los planificadores globales y locales descritos en la Sección 3.2 y los mapas de coste del entorno los cuales va actualizando con la información sensorial recibida.

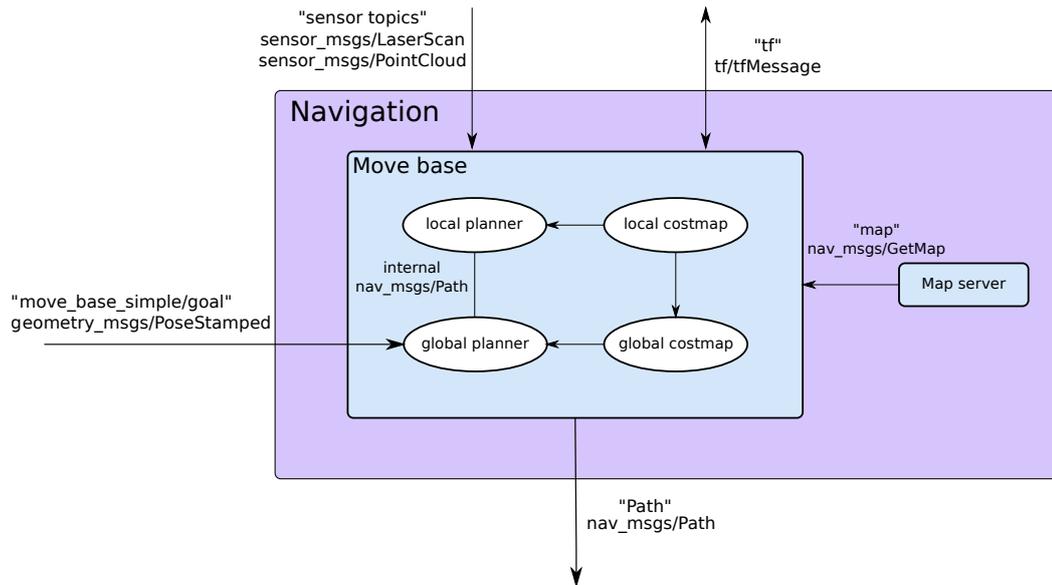


FIGURA 4.9: Esquema de paquetes utilizados por el sistema de navegación.

Un esquema de funcionamiento más detallado se encuentra en la Figura 4.9. La salida de este paquete es un vector de puntos 3D en la referencia de coordenadas del mapa que lleva hasta el objetivo marcado, el *path*. Para proyectar esta información sobre las imágenes del SPV, es necesario conocer la secuencia de transformaciones  ${}^M\mathbf{T}_C$  que permite proyectar estos puntos, en la referencia de coordenadas de la cámara. Las entradas de este sistema son la nube de puntos de los obstáculos convertida a escáner láser, el árbol de transformadas, el mapa 2D (para lo cual se utiliza el paquete *map server*) y una localización objetivo hacia la que tiene que buscar la trayectoria.

El SPV calcula la referencia del suelo respecto de la cámara y en el arranque del sistema para saber conocer la localización inicial de la misma, y se encarga de publicarla y mantenerla actualizada. El SPV recibe la trayectoria generada por el sistema de navegación y la proyecta con la matriz de proyección (Sección 2.4) sobre la imagen de fosfenos para iluminar los puntos correspondientes al camino.

## Capítulo 5

# Experimentos

Se han realizado pruebas de navegación con los sistema de guiado desarrollado tanto en entornos virtuales, como en un escenario real. A continuación se presentan algunas de las secuencias de imágenes obtenidas durante estas pruebas.

### 5.1. Pruebas de navegación en un entorno virtual

Para evaluar el funcionamiento del sistema de guiado, se han simulado en el entorno virtual en el que se ha desarrollado este proyecto, situaciones reales en las que una persona que sufre algún tipo de discapacidad visual, pudiese ser guiada hasta una ubicación deseada.

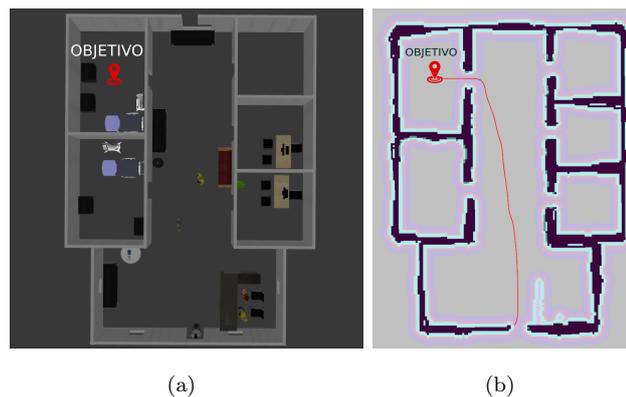


FIGURA 5.1: (a) Escenario en Gazebo, (b) Trayectoria global calculada por el planificador sobre el mapa.

Una de estas situaciones podría ser el guiado de una persona dentro de un hospital hasta la sala en la que ha sido citada. Para ello se ha diseñado un escenario que simula un hospital (Figura 5.1) en el que el sujeto debe alcanzar la localización de una sala.

El sujeto del experimento interactúa con el simulador a través de una pantalla y un teclado. En la pantalla puede ver la imagen de fosfenos pero no el resto de información que maneja el sistema. Durante el camino el sujeto se va encontrando personas u objetos, que se interponen en el camino planificado y por ello la ruta es recalculada con el fin de esquivarlos.

### 5.1.1. Pruebas de navegación con el camino de guiado

El objetivo de esta prueba es evaluar con diferentes configuraciones del SPV (diferentes números de fosfenos, campo de vista) el funcionamiento de este método y la claridad con la que se percibe la trayectoria en las imágenes del simulador. El sujeto debe llegar desde la puerta de entrada del hospital hasta la sala marcada en el mapa como objetivo (Figura 5.1).

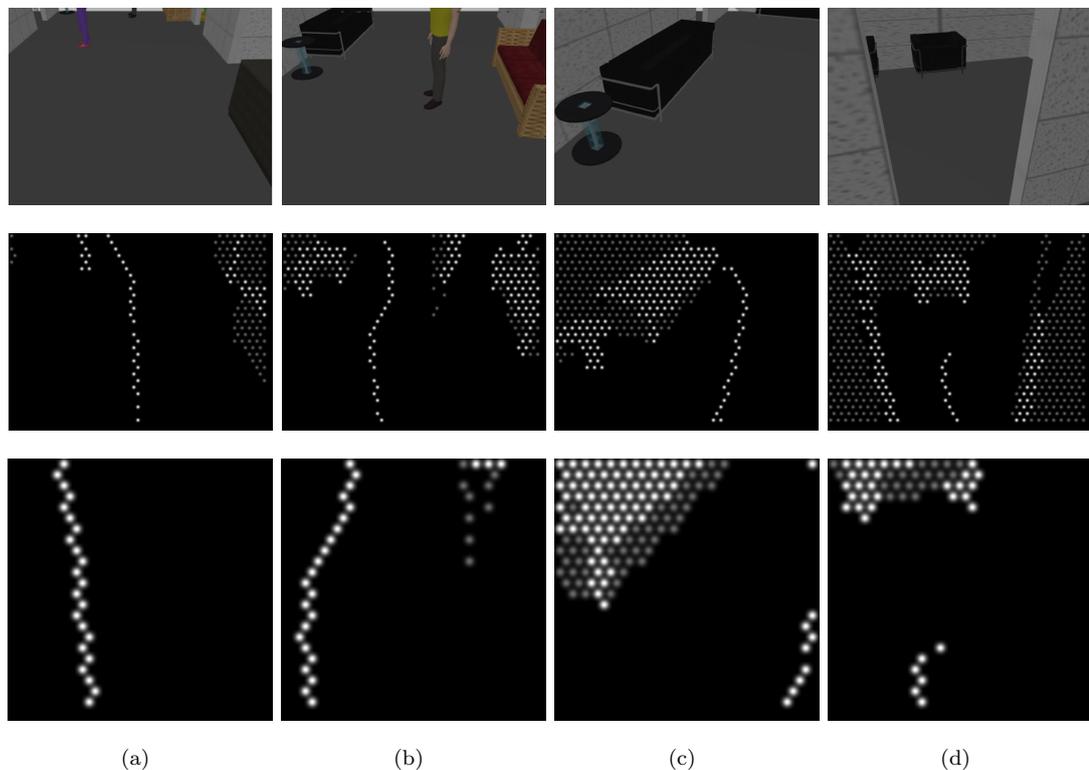


FIGURA 5.2: Secuencia de imágenes de la trayectoria seguida con el camino de guiado: En la fila superior se pueden ver las imágenes captadas por la cámara RGB y en las demás las imágenes generadas por el SPV. En la segunda fila el campo de vista es de  $58^\circ$  y la imagen tiene 1000 fosfenos. En la tercera fila el campo de vista es de  $25^\circ$  y la imagen tiene 500 fosfenos.

En la Figura 5.2 se presenta una secuencia de imágenes del mismo experimento realizado con dos configuraciones del SPV diferentes en las que se ha variado el campo de vista de la imagen y el número de fosfenos. La línea blanca representa la trayectoria que se debe seguir para alcanzar el objetivo. El planificador es capaz de recalcularla

trayectoria calculada inicialmente cuando se encuentra con un obstáculo cercano que no está registrado en el mapa con el fin de evitarlo.

En segunda fila de la Figura 5.2, las imágenes han sido generadas con el mismo campo de vista que tiene la cámara con la cual se ha grabado la secuencia. En ellas se pueden identificar algunos obstáculos con cierta claridad debido al amplio campo de vista de la imagen. En la tercera fila, en la que el campo de vista de la imagen se ha reducido a  $25^\circ$  y el número de fosfenos a 500. A pesar de ello debido a que el camino está representado sobre el plano del suelo con el fin de que no interfiera la identificación y representación del mismo con la de otros obstáculos de la escena, es posible seguir el camino para alcanzar el objetivo aunque se pierda información de la escena.

### 5.1.2. Pruebas de navegación con la brújula

Se ha repetido la misma secuencia descrita anteriormente, pero esta vez utilizando como método de guiado la brújula. El sujeto debe mantener los marcadores de dirección de la brújula centrados en la imagen para seguir correctamente el camino trazado por el planificador y evitar los obstáculos. En la Figura 5.3 se presenta una secuencia de imágenes obtenidas durante esta prueba.

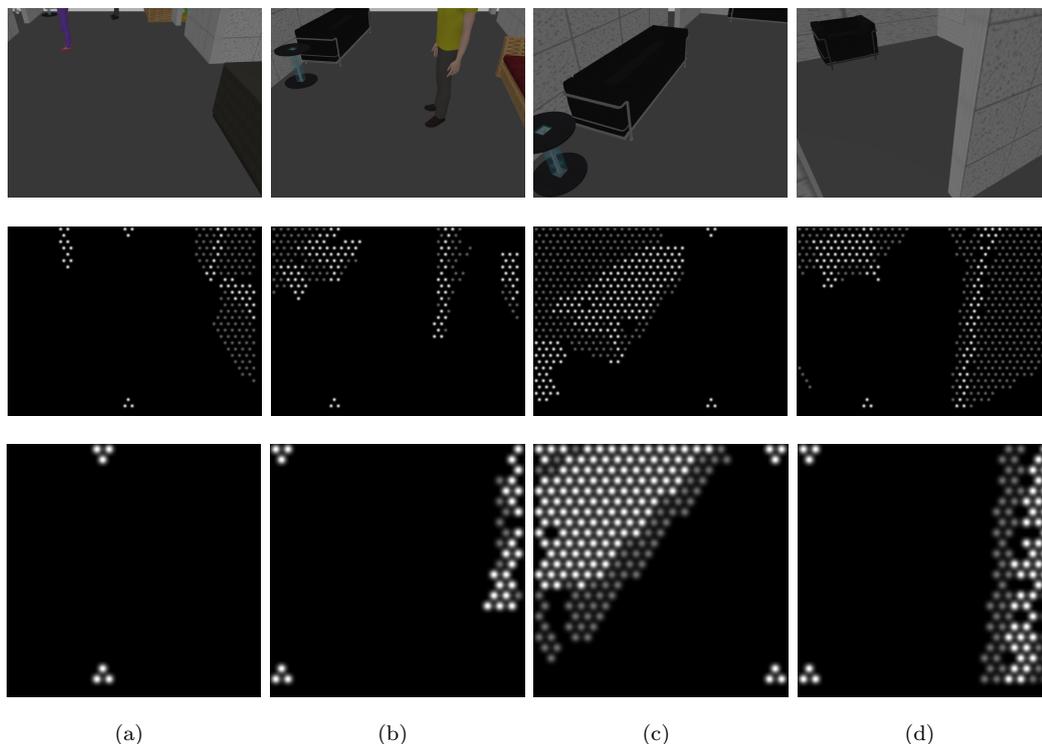


FIGURA 5.3: Secuencia de imágenes de la trayectoria seguida con la brújula: En la fila superior se pueden ver las imágenes captadas por la cámara RGB y en las demás las imágenes generadas por el SPV. En la segunda fila el campo de vista es de  $58^\circ$  y la imagen tiene 1000 fosfenos. En la tercera fila el campo de vista es de  $25^\circ$  y la imagen tiene 500 fosfenos.

La secuencia de imágenes presentada en la Figura 5.3 ha sido capturada en el mismo entorno simulación que las de la Figura 5.2 pero esta vez la información para el seguimiento de trayectorias viene proporcionada por el método de la brújula. Como se ha explicado anteriormente, para seguir la trayectoria correctamente el sujeto debe orientarse en la dirección que sitúe los marcadores de dirección, representadas como tres puntos blancos en la parte superior e inferior de la imagen, en la parte central de la misma. Por ejemplo, en la Figura 5.3 (b) los marcadores están en el lado izquierdo de la pantalla, lo que quiere decir que el sujeto debe girar en esa dirección para orientarse hacia el hueco que hay entre la persona y el sofá. En la Figura 5.3 (c) los marcadores están situados en el lado derecho de la imagen lo que indica que el sujeto debe de corregir su orientación girando en esa dirección para no colisionar con el sofá.

### 5.1.3. Comparación de los dos métodos de guiado

En la Figura 5.4 se representan las trayectorias seguidas por el sujeto utilizando ambos métodos de guiado. Con ambos métodos es capaz de seguir la trayectoria planificada. En el primero de ellos la trayectoria es menos oscilante ante los cambios de dirección debido a que el sujeto puede planificar la trayectoria en su horizonte de visión ya que esta está representada sobre la imagen. Sin embargo en el segundo caso la trayectoria es ligeramente más oscilante ya que es un método reactivo, en el que en cada momento se representa la dirección instantánea que debe tomar para seguir la trayectoria planificada que no se visualiza sobre la imagen explícitamente como en el anterior método. También cabe destacar que aunque la brújula de menos información, ocupa un menor número de fosfenos en la imagen y puede ser de mayor utilidad en el caso de que la prótesis utilizada tenga una resolución baja.

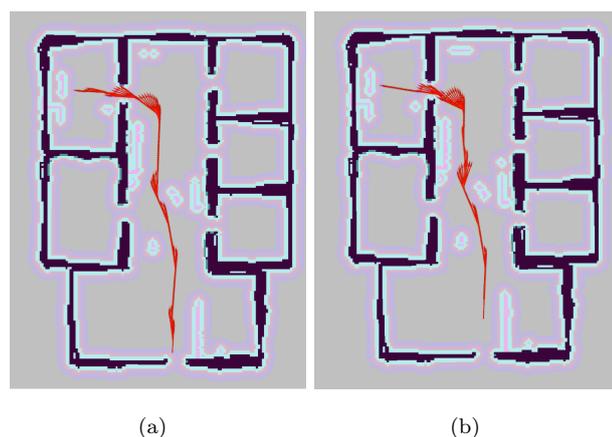


FIGURA 5.4: Trayectoria seguida por el sujeto con (a) el sistema de guiado de camino y (b) el sistema de guiado de brújula

## 5.2. Pruebas de navegación en un entorno real

Se han realizado pruebas en un escenario real con una cámara RGB-D. Para ello se ha construido un mapa 3D del entorno en el que se van a realizar los experimentos con el fin de poder localizar al sujeto en él. Además del mapa 3D, se ha generado un mapa 2D del mismo escenario con el que poder planificar trayectorias hasta un lugar conocido del mismo.

El experimento es el siguiente: se ha marcado como objetivo en el mapa (Figura 5.5) la puerta situada al final del pasillo. El sujeto debe caminar hacia dicho objetivo basándose en la información proporcionada por sistema de guiado del SPV.



FIGURA 5.5: Planificación de la trayectoria sobre el mapa global 2D del entorno.

En la Figura 5.6 se presenta una secuencia de imágenes capturadas durante el experimento. Se puede observar cómo la trayectoria inicialmente va en línea recta hacia la puerta, pero durante el trayecto hacia el objetivo, la persona se cruza y el planificador recalcula la trayectoria para no chocar con él. También se puede apreciar en las imágenes de la segunda fila, que debido a que el hueco entre la columna y la persona es demasiado estrecho, el planificador recalcula la trayectoria bordeando la columna y pasando por el espacio más amplio que encuentra, para posteriormente reconducirlo hacia el camino calculado inicialmente.

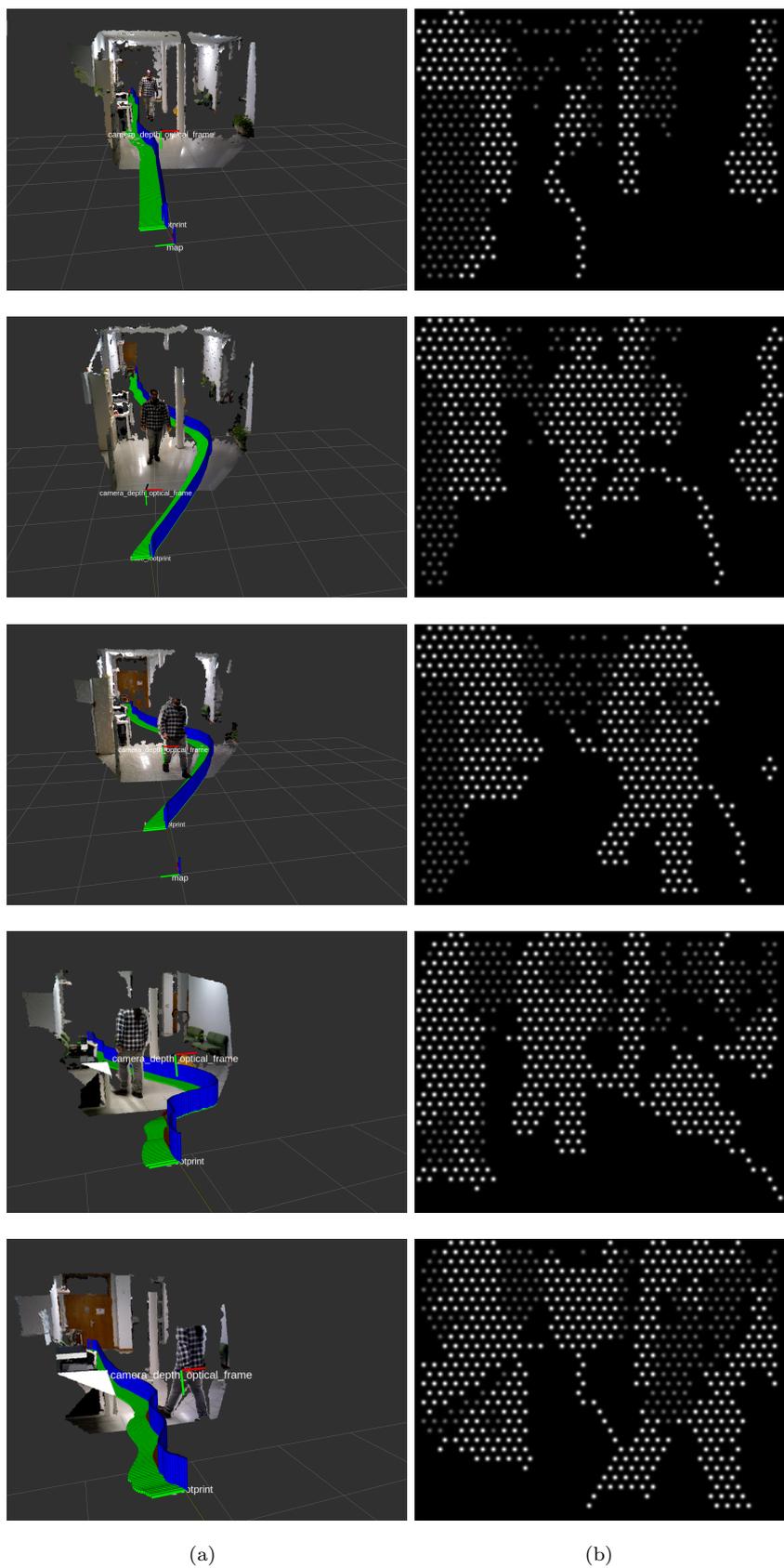


FIGURA 5.6: (a) Nube de puntos 3D de la escena con la trayectoria representada con una línea verde y azul. (b) Imágenes generadas por el SPV con la trayectoria proyectada como una línea de puntos blanca.

## Capítulo 6

# Conclusiones y trabajo futuro

En este trabajo fin de máster se ha desarrollado un sistema de guiado y asistencia para personas invidentes con prótesis visuales. El trabajo realizado se enmarca en un entorno de simulación de prótesis visuales que se va a usar para evaluar las distintas estrategias de representación y asistencia. El sistema de guiado propuesto funciona sobre el sistema operativo ROS. El sistema permite trabajar en un entorno de simulación (Gazebo) por un lado, lo que permite la repetitividad en el diseño de experimentos con sujetos reales y dotar de sentido estadístico a los resultados. Por otro lado, el sistema permite trabajar con sistemas de adquisición RGB-D reales replicando condiciones más realistas para la realización de demostradores que validarían una aplicación final en visión protésica. La principal contribución de este trabajo es un sistema de guiado con dos propuestas de representación de trayectorias: una trayectoria pintada sobre el suelo inspirada en las indicaciones típicas de hospitales y una brújula inspirada en los indicadores de simulación aérea. Para la correcta representación de estas trayectorias deben integrarse, en forma de realidad aumentada, en un sistema de representación icónica del entorno que incluye, suelos, obstáculos y diversas representaciones para transmitir profundidad. La modularidad del proyecto ha implicado un trabajo importante para la integración de los distintos sistemas involucrados que incluyen planificadores de trayectorias, sistemas de simulación para representación de fosfenos, algoritmos de detección de objetos y superficies planas y, en el caso de uso de cámaras reales, un algoritmo de SLAM (ORB-SLAM2). Además, se incluye un modelo de persona con prótesis y distintos escenarios modelados en Gazebo para su uso en experimentación médica.

Como trabajo futuro se contempla la realización de experimentos con sujetos reales para la validación y mejora de las distintas representaciones propuestas. El diseño de experimentos deberá seguir un esquema similar al usado en otras evaluaciones de fosfenos con sujetos reales [33]. El siguiente paso sería integrar la realimentación del sujeto en un sistema de realidad virtual que permitiese una inmersión mayor.

# Bibliografía

- [1] Jesus Bermudez-Cameo, Alberto Badias-Herbera, Manuel Guerrero-Viu, Gonzalo Lopez-Nicolas, and Jose J Guerrero. Rgb-d computer vision techniques for simulated prosthetic vision. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 427–436. Springer, 2017.
- [2] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2d lidar slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1271–1278. IEEE, 2016.
- [3] Organización mundial de la salud. URL <https://www.who.int/en/news-room/fact-sheets/detail/blindness-and-visual-impairment>.
- [4] The international agency for the prevention of blindness. URL <https://www.iapb.org/vision-2020/who-facts>.
- [5] Giles S Brindley and WS Lewin. The sensations produced by electrical stimulation of the visual cortex. *The Journal of physiology*, 196(2):479–493, 1968.
- [6] David D Zhou, Jessy D Dorn, and Robert J Greenberg. The argus® ii retinal prosthesis system: An overview. In *2013 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pages 1–6. IEEE, 2013.
- [7] Katarina Stingl, Karl Ulrich Bartz-Schmidt, Dorothea Besch, Angelika Braun, Anna Bruckmann, Florian Gekeler, Udo Greppmaier, Stephanie Hipp, Gernot Hörtdörfer, Christoph Kernstock, et al. Artificial vision with wirelessly powered subretinal electronic implant alpha-ims. *Proceedings of the Royal Society B: Biological Sciences*, 280(1757):20130077, 2013.
- [8] Spencer C Chen, Gregg J Suaning, John W Morley, and Nigel H Lovell. Simulating prosthetic vision: I. visual models of phosphenes. *Vision research*, 49(12):1493–1506, 2009.
- [9] Nick Barnes. An overview of vision processing in implantable prosthetic vision. In *2013 IEEE International Conference on Image Processing*, pages 1532–1535. IEEE, 2013.

- [10] Marco Leo, Antonino Furnari, Gerard G Medioni, Mohan Trivedi, and Giovanni M Farinella. Deep learning for assistive computer vision. *Computer Vision – ECCV 2018 Workshops*, 11134, 2018.
- [11] David Feng, Shaodi You, and Nick Barnes. Dsd: depth structural descriptor for edge-based assistive navigation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1536–1544, 2017.
- [12] Wen Lik Dennis Lui, Damien Browne, Lindsay Kleeman, Tom Drummond, and Wai Ho Li. Transformative reality: improving bionic vision with robotic sensing. In *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 304–307. IEEE, 2012.
- [13] Chris McCarthy, Janine G Walker, Paulette Lieby, Adele Scott, and Nick Barnes. Mobility and low contrast trip hazard avoidance using augmented depth. *Journal of neural engineering*, 12(1):016003, 2014.
- [14] Alejandro Perez-Yus, Jesus Bermudez-Cameo, Gonzalo Lopez-Nicolas, and Jose J Guerrero. Depth and motion cues with phosphene patterns for prosthetic vision. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 1516–1525, 2017.
- [15] Marc Patrick H Zapf, Mei-Ying Boon, Paul B Matteucci, Nigel H Lovell, and Gregg J Suaning. Towards an assistive peripheral visual prosthesis for long-term treatment of retinitis pigmentosa: evaluating mobility performance in immersive simulations. *Journal of neural engineering*, 12(3):036001, 2015.
- [16] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE international conference on robotics and automation*, pages 1–4. IEEE, 2011.
- [17] Raúl Mur-Artal and Juan D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5): 1255–1262, 2017. doi: 10.1109/TRO.2017.2705103.
- [18] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.
- [19] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2 (11):559–572, 1901.
- [20] Tahir Rabbani, FA Van Den Heuvel, and George Vosselman. Segmentation of point clouds using smoothness constraints. In *ISPRS commission V symposium: image engineering and vision metrology*, pages 248–253. International Society for Photogrammetry and Remote Sensing (ISPRS), 2006.

- [21] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.
- [22] Vladimir Kulyukin, Chaitanya Gharpure, John Nicholson, and Sachin Pavithran. Rfid in robot-assisted indoor navigation for the visually impaired. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 2, pages 1979–1984. IEEE, 2004.
- [23] José Faria, Sérgio Lopes, Hugo Fernandes, Paulo Martins, and João Barroso. Electronic white cane for blind people navigation assistance. In *2010 World Automation Congress*, pages 1–7. IEEE, 2010.
- [24] Dragan Ahmetovic, Cole Gleason, Chengxiong Ruan, Kris Kitani, Hironobu Takagi, and Chieko Asakawa. Navcog: a navigational cognitive assistant for the blind. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, pages 90–99. ACM, 2016.
- [25] Michael Zöllner, Stephan Huber, Hans-Christian Jetter, and Harald Reiterer. Navi—a proof-of-concept of a mobile navigational aid for visually impaired based on the microsoft kinect. In *IFIP Conference on Human-Computer Interaction*, pages 584–587. Springer, 2011.
- [26] Alejandro Rituerto, Giovanni Fusco, and James M Coughlan. Towards a sign-based indoor navigation system for people with visual impairments. In *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 287–288. ACM, 2016.
- [27] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.
- [28] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, 1997.
- [29] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2149–2154. IEEE, 2004.
- [30] Amir Rasouli and John K Tsotsos. The effect of color space selection on detectability and discriminability of colored objects. *arXiv preprint arXiv:1702.05421*, 2017.
- [31] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

- 
- [32] Aurelio Cappozzo. Analysis of the linear displacement of the head and trunk during walking at different speeds. *Journal of Biomechanics*, 14(6):411–425, 1981.
- [33] Melani Sanchez-Garcia, Ruben Martinez-Cantin, and Jose J Guerrero. Indoor scenes understanding for visual prosthesis with fully convolutional networks. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2019.