Universidad de Murcia

Facultad de Informática

Departamento de Ingeniería y Tecnología de Computadores

TESIS DOCTORAL

Una Arquitectura Eficiente de Percepción de Alto Nivel: Navegación Visual para Robots Autónomos en Entornos Estructurados.

Pedro Enrique López de Teruel Alcolea

Directores:

Alberto Ruiz García

José Manuel García Carrasco

Murcia, 2003

Resumen

Esta tesis describe el diseño e implementación de un robot móvil capaz de realizar una interpretación estructural del entorno usando únicamente información sensorial de tipo visual y propioceptivo. El comportamiento deseado es la navegación en tiempo real basada en esta interpretación, en lugar de la estrictamente reactiva a los estímulos inmediatos. Optamos por un criterio de diseño fundamentalmente predictivo: el sistema debe anticipar las consecuencias de sus acciones, mostrando una cierta comprensión predictiva de la escena en la que se mueve.

Se proponen soluciones para todos los niveles de la percepción. En las etapas inferiores se ha desarrollado una técnica de extracción de segmentos con información de color, de gran potencia expresiva y eficiencia computacional. Con objeto de inferir propiedades euclídeas del espacio se presenta una colección de métodos de autocalibración que, apoyándose en el formalismo de la geometría proyectiva, aprovechan también la información odométrica del agente autónomo. La percepción de alto nivel, finalmente, se resuelve mediante un ciclo de generación, seguimiento y corroboración de hipótesis modelizadoras del entorno, que son mantenidas en una representación interna estable y sintonizada con los movimientos del agente. La principal característica de este mecanismo es la interacción constante entre los procesos perceptivos ascendentes, guiados por los estímulos sensoriales, y los descendentes, guiados por los modelos previos. Todos estos elementos se integran en una arquitectura *hardware-software* eficiente, modular y flexible, en la que acción y percepción cooperan estrechamente para lograr la robustez y continuidad necesarias en la navegación.

Abstract

This thesis describes the design and implementation of a mobile robot which is able to perform a structural interpretation of indoor environments, using only visual and proprioceptive sensory information. The desired behaviour is real-time navigation based on this interpretation, instead of a reactive approach. We use a predictive design criterion: the system must anticipate the consequences of its actions, showing a certain predictive understanding of the scene in which it moves.

We propose solutions for all perception levels. First, we have developed a segment extraction technique augmented with colour information, of remarkable expressive power and computational efficiency. To infer euclidean properties of space we present a collection of autocalibration methods based on the projective geometry formalism, taking advantage of the odometric information of the autonomous agent. High level perception, finally, is solved through generation, tracking and confirmation of hypothesis about the environment, which are maintained in an stable internal representation tuned with the agent movements. This mechanism is characterized by a constant interaction between the bottom-up perceptive processes, guided by sensory stimuli, and the top-down ones, guided by the models. All these elements are integrated in an efficient, modular and flexible hardware-software architecture, in which action and perception cooperate to achieve the robustness and continuity needed by real world navigation.

A Paloma, el agua que nunca faltó en esta *travesía del desierto*.

A la memoria de mi padre.

Agradecimientos

A mis directores de tesis, los doctores A. Ruiz y J.M. García, por haberme guiado en esta investigación mostrando una paciencia y comprensión sin límites. Gracias por el conocimiento y la amistad que habéis compartido conmigo a lo largo de estos años.

A mi familia, que desde Lorca y Motril siempre estuvo ahí, y sin cuyo apoyo en los momentos difíciles esta tesis no habría visto la luz. Gracias por no haber dejado nunca de creer en mí.

A mis compañeros del Departamento de Ingeniería y Tecnología de Computadores, *los que están y los que han sido*, por hacer de cada día en la Facultad compartido con ellos un verdadero placer. Gracias por haberme animado y ayudado en multitud de ocasiones, y por entender así esta profesión.

A los compañeros de comidas y tertulias de café, Alberto, Juan, Lorenzo y Paloma, por hacer del descanso diario uno de los momentos más agradables y enriquecedores de la jornada.

A mis amigos de toda la vida, por recordarme que nada es tan valioso como una amistad sincera. Mención especial para *Rosen*, siempre dispuesto a ayudar en el *cacharreo*.

A la comunidad del software libre, por el extraordinario apoyo que prestan día a día al progreso científico y haber hecho posible la existencia de maravillas como *Linux*, *Latex*, *gcc* y tantas otras herramientas sin las cuales, definitivamente, esta tesis no hubiese sido tan divertida.

A Alberto, la persona que me enseñó a disfrutar de este trabajo y que desde el principio me brindó una amistad muy por encima de la relación profesional. Gracias por todo lo enseñado y compartido.

Y finalmente a Paloma, por quererme sin reservas y haberme acompañado en este viaje. Va por ti.

Pedro E.
Murcia, Junio de 2003

Índice general

Índice de figuras

Índice de tablas

Índice de algoritmos

CAPÍTULO I

"Picasso's Meninas stage proofs" (detalle), R. Hamilton, 1973

Introducción

Los dos proyectos que he indicado (un vocabulario infinito para la serie natural de los números, un inútil catálogo mental de todas las imágenes del recuerdo) son insensatos, pero revelan cierta balbuciente grandeza. Nos dejan vislumbrar o inferir el vertiginoso mundo de Funes. Éste, no lo olvidemos, era casi incapaz de ideas generales, platónicas. No sólo le costaba comprender que el símbolo genérico perro *abarcara tantos individuos dispares de diversos tamaños y diversa forma; le molestaba que el perro de las tres y catorce (visto de perfil) tuviera el mismo nombre que el perro de las tres y cuarto (visto de frente).*

J.L. BORGES, *Funes el memorioso.*

1.1. La percepción artificial

1.1.1. Generalidades

Recientemente ha aumentado el interés en las ciencias de lo artificial, impulsadas, entre otras causas, por el extraordinario avance tecnológico de la informática. En particular, a partir de la segunda mitad del siglo pasado una de las cuestiones que ha recibido más atención es la construcción de robots autónomos capaces de una interacción inteligente con su entorno. Para alcanzar este objetivo, es necesario resolver el problema de la *percepción artificial*. Éste es precisamente el campo de investigación de esta tesis.

Históricamente, el problema se ha estudiado desde muchos puntos de vista. Una de las primeras hipótesis era que los sistemas cognitivos podían modelarse perfectamente como sistemas de manipulación *simbólica* (Newell y Simon, 1976). La percepción se consideraba un simple módulo auxiliar que extraería los símbolos a partir de los estímulos sensoriales, y

quedaba por tanto relegada a un segundo plano. Los modelos propuestos utilizaban sistemas de inferencia basados en reglas, manipulaciones formales y métodos de búsqueda heurística. Pero, a menudo, lo que se llamaba "inteligencia artificial" no era más que una búsqueda de isomorfismos sintácticos entre estructuras predeterminadas por el diseñador.

Casi en paralelo surgieron las primeras técnicas de reconocimiento de patrones (*pattern recognition*). Éstas trabajan directamente sobre el espacio de estímulos sensoriales utilizando métodos estadísticos y de optimización (Nilsson, 1965; Devroye *et al.*, 1996). Se trata de detectar automáticamente ciertas propiedades relevantes del mundo externo a través de procesos de aprendizaje computacional, los cuales intentan descubrir regularidades significativas en las variables de entrada a partir de la observación de muchos casos particulares. Algunos modelos estaban inspirados en el funcionamiento del sistema nervioso (Rosenblatt, 1962; McCulloch y Pitts, 1943; Hebb, 1949). Pronto surgieron los modelos *conexionistas*, capaces de capturar relaciones más complejas mediante la autoorganización de un número elevado de elementos de proceso simples, asíncronos y con interacciones locales (Arbib, 1995; Churchland y Sejnowski, 1992). En contraste con los anteriores, todos estos modelos trabajan a un nivel subsimbólico, con representaciones distribuidas. En principio, pueden resolver incluso tareas en las que hay que inferir una cierta estructura composicional en los patrones sensoriales (LeCun *et al.*, 1998).

La mayoría de estas técnicas confían en el principio de inducción ingenuo de que si se resuelven bien muchos casos concretos, se resolverán bien muchos casos futuros. Más recientemente, la teoría matemática de la generalización ha matizado este principio. Una máquina inductiva probablemente generalizará si resuelve bien muchos más casos concretos que los que sería capaz de "aprender de memoria" (Vapnik, 1998). Esta teoría ha permitido construir máquinas extraordinariamente potentes, las SVM, (del inglés *support vector machines*) que controlan automáticamente su capacidad incluso en espacios de propiedades (implícitos) de dimensión arbitrariamente alta (Schölkopf *et al.*, 1999). Estas máquinas incluyen como caso particular a otros modelos anteriores y disponen de métodos de aprendizaje más eficientes.

El enfoque puramente inductivo, o empírico, es apropiado para abordar tareas de naturaleza estática, donde es suficiente asociar un espacio de entradas con otro de salidas, aunque sea a costa de "fuerza bruta muestral". A través de él se han alcanzado numerosos éxitos en aplicaciones de gran importancia práctica, como el reconocimiento acústico del habla, el reconocimiento óptico de textos escritos, etc. Sin embargo, los tipos de fallo que presentan revelan una frustrante ausencia de comprensión de ciertos aspectos claves del dominio de trabajo. En el caso concreto de los sistemas autónomos, además, este tipo de aprendizaje resulta claramente inviable, ya que es imposible recoger la virtualmente infinita variedad de situaciones que se pueden presentar en la interacción de un robot con un entorno mínimamente complejo.

Intentando superar estas limitaciones se han planteado alternativas que trasladan el énfasis hacia la relación de interdependencia mutua entre el agente y su entorno. Surgen así las aproximaciones *basadas en el comportamiento*, en las que se conectan los estímulos sensoria-

les directamente con las acciones sin necesidad de representaciones perceptuales explícitas (Brooks, 1991). Se confía en que la adecuada combinación de diferentes generadores de comportamientos simples produzca un comportamiento global complejo, no explícitamente preprogramado. No se utilizan en ningún momento abstracciones conceptuales, aunque éstas puedan resultar convenientes para describir el comportamiento observado. Se han conseguido resultados prometedores, incluso en robots humanoides, pero parece existir una barrera de complejidad que no es fácil atravesar.

Descendiendo aún más hacia los principios físicos, se ha propuesto también la hipótesis de que los sistemas cognitivos deben considerarse propiamente como *sistemas dinámicos* (Van Gelder, 1998). Según estos autores, no debe confundirse la evolución de un sistema natural con una mera simulación computacional de su comportamiento. En lugar de ello, se considera esencial la continuidad de las variables involucradas y su evolución temporal en un espacio de estados, de acuerdo con leyes dinámicas que es razonable expresar mediante ecuaciones diferenciales, en oposición a los algoritmos. Este enfoque tiende a rechazar las representaciones internas, estáticas, y prefiere manejar conceptos de la teoría de sistemas tales como trayectorias del espacio de estados, atractores, bifurcaciones, etc. La aproximación es atractiva, pero llevada al extremo –*'cognitive agents instantiate quantitative systems at the highest relevant level of causal organization' (Van Gelder)*– puede limitar en cierto sentido su ámbito de aplicación.

En conclusión, la percepción artificial se ha considerado desde trivial hasta no computable (Penrose, 1994), pasando por los que creen que es innecesario hacerla explícita o simplemente la reducen a un problema estadístico. Como veremos más adelante, hay incluso quienes señalan la importancia de la percepción de conceptos abstractos y son moderadamente optimistas respecto a la resolución del problema en el futuro (Hofstadter, 1995). Tales discrepancias nos advierten de que el problema podría estar incluso mal planteado. Uno de los expertos más entusiastas muestra incluso una cierta sensación de fracaso (Brooks, 2001):

> *'But despite of all this, both fields [AI and Alife] have been labelled as failures for not having lived up to grandiose promises. (…) At the heart of this disappointment lies the fact that neither AI nor Alife has produced artefacts that could be confused with a living organism for more than an instant. (…) we are not good at modeling living systems, at small or large scales. Something is wrong.'*

Pero, ¿qué puede estar mal? Brooks apunta varias posibilidades: hemos ajustado mal algunos parámetros, los modelos no son suficientemente complejos, nos falta potencia de cálculo, o simplemente, desconocemos algún concepto biológico fundamental que ni siquiera imaginamos…

1.1.2. El criterio predictivo

Preguntas como ¿qué es la percepción?, ¿qué produce?, ¿quién la utiliza?, etc., quizá sean prematuras dada la relativa juventud de este campo de investigación. Posiblemente en este

momento sea preferible replantear la cuestión de un modo algo más directo y operacional. ¿Qué tipo de procesos pueden contribuir a un comportamiento bien integrado con el entorno? Este punto de vista sugiere inmediatamente una idea que consideramos muy prometedora, a pesar de su simplicidad (Dennet, 1991):

> *'The key to control is the ability to* track *or even* anticipate *the important features of the environment, so all brains are, in essence,* anticipation machines.*'*

Es bueno predecir el futuro, al menos a corto plazo. Esta afirmación de sentido común es difícilmente rechazable. Nosotros añadiremos el siguiente matiz: la predicción de estados futuros del entorno realizada a partir de los estímulos sensoriales sólo puede contrastarse mediante la propia información sensorial. Esto significa que el intento de predicción explícita de *propiedades* del entorno podría ser incorrecto. Estas propiedades son infinitas y la selección del diseñador podría sesgar inconscientemente el diseño. Seguramente sea más fructífero tratar de anticipar directamente los estímulos sensoriales. Si esto se consigue a escalas de tiempo suficientemente largas, será difícil negar que se han captado propiedades relevantes del entorno, al menos desde el punto de vista que a nosotros nos interesa, el del propio agente.

En definitiva, dado un dominio de trabajo, y a un nivel determinado por la complejidad del mismo, el criterio descrito puede ayudarnos a juzgar el grado de *comprensión predictiva* del entorno alcanzada por un robot. En última instancia, por tanto, este criterio puede servir como guía para diseñar sistemas de percepción artificial.

1.1.3. Representaciones abstractas

Admitamos, pues, que anticipar los estímulos sensoriales es en general deseable para un agente autónomo. El paso siguiente, en nuestra opinión, es entonces casi obligado: necesitamos la mediación de ciertas *representaciones abstractas*. Hay varios argumentos que sostienen esta afirmación:

- En entornos de complejidad moderada, la evolución de la información sensorial es definitivamente inexplicable sin la ayuda de construcciones conceptuales completamente ajenas a –y no deducibles de– la dinámica local del espacio de estímulos. Pensemos, por ejemplo, en la aparición y desaparición de manchas de color en una imagen debidas a un objeto que gira en el espacio.

- El uso de abstracciones puede generar predicciones a una velocidad mucho mayor que la que permitiría el tratamiento de la masa sensorial original. De esta forma se reduce significativamente la latencia del ciclo entre la percepción y la acción, aumentándose así la "inmersión" del agente en el entorno. Esto es particularmente relevante cuando los recursos de procesamiento son limitados.

- El contexto espacio-temporal de la información sensorial determina en gran parte la capacidad de predicción. Por ejemplo, dependiendo de aspectos como la iluminación o los objetos vecinos, un objeto de color bien definido puede reflejar luz de prácticamente cualquier otro color. En el extremo opuesto, la dinámica del entorno podría también depender de diferencias arbitrariamente pequeñas en el espacio de estímulos.

Pero, más allá de los aspectos predictivos, hay otras razones de peso que justificarían el uso este tipo de representaciones en los sistemas de percepción artificial. Existen tareas esenciales para un agente autónomo que no parece sencillo resolver sin recurrir a este tipo de abstracciones conceptuales. Un ejemplo es el reconocimiento desde un punto de vista diferente de lugares previamente visitados, posiblemente con pequeños cambios. No es posible establecer una métrica apropiada para esto sobre el espacio de estímulos. Nos encontraríamos ante el *Funes borgiano* de la cita al comienzo del capítulo.

Ya en el ámbito de los sistemas naturales, un argumento adicional sería que muchas ilusiones ópticas no hacen si no evidenciar fallos de interpretación. Podríamos decir que las sensaciones básicas son detecciones de magnitudes elementales en la realidad, y que la percepción consiste en imponer estructura de interpretación al subconjunto de las sensaciones que requieren nuestra atención. Las ilusiones ópticas se basan a menudo en los fallos en la imposición de esta estructura (véase, por ejemplo, la imagen de la página 129). Esta dualidad entre información sensorial y organización coherente de los datos tiene también cierta tradición filosófica. Kant, por ejemplo, dividía la percepción en *sensibilidad* y *entendimiento*. Parafraseando uno de sus escritos, *'conceptos sin perceptos están vacíos; perceptos sin conceptos son ciegos'*.

Las representaciones, sin embargo, no pueden ser copias detalladas de la realidad. Caeríamos en una regresión infinita en la que aún tendría que percibirse lo importante de esa representación. La sensación de detalle efectiva sólo se debe a que estamos continuamente informados de cualquier cambio (O'Regan y Noe, 2001), y se reconstruye inmediatamente la interpretación de la masa sensorial que cae bajo el área de interés. Ver no es en crear una representación del mundo dentro del cerebro. Sólo consiste en tener la consciencia de que el mundo exterior es una *memoria externa*, accesible de modo inmediato a través de un cambio de atención (Simons y Chabris, 1999; Wolfe, 1999).

En conclusión, si queremos incorporar en los robots autónomos una cierta capacidad de predecir el resultado de las acciones posibles, necesitamos la mediación de ciertas representaciones abstractas, estructuradas. Dennet (1995) lo explica así:

> *'Skinnerian conditioning is a fine capacity to have, so long as you are not killed by one of your early errors. A better system involves* preselection *among all the possible behaviors or actions, weeding out the truly stupid options before risking them in the harsh world. (…) It [preselection] must come from a sort of* inner environment *–an inner something-or-other– that is structured in such a way that the surrogate actions it favors are more often than not the very actions the real world would*

> *also bless, if they were actually performed. In short, the inner environment, whatever it is, must contain lots of* information *about the outer environment and its regularities. Nothing else (except magic) could provide preselection worth having. Now, here we must be very careful not to think of this inner environment as simply a replica of the outer world, with all its physical contingencies reproduced. (…) The information about the world has to be there, but it also has to be structured in such a way that there is a nonmiraculous explanation of how it got there, how it is maintained, and how it actually achieves the preselective effects that are its* raison d'être.*'*

1.1.4. Percepción de alto nivel

Una de las características de la inteligencia es la capacidad para establecer analogías entre situaciones aparentemente distintas. Esto no puede resolverse mediante representaciones estáticas predeterminadas por el diseñador. Hofstadter (1995) sostiene que la clave de la inteligencia está en la *percepción de alto nivel*, concretamente en la capacidad de construir representaciones estructuradas y reconfigurables dinámicamente. Pero para poder realizar este tipo de percepción el procesamiento ascendente (*bottom-up*), dirigido por los datos, es insuficiente. Es necesario tener en cuenta el contexto, los objetivos, las expectativas, etc. Ese procesamiento debe estar, por tanto, complementado por un flujo descendente (*top-down*) que module el proceso perceptivo, dirigiéndolo a través de la propia representación. "Ver X" es en realidad "ver Y como X".

Sin embargo, la interpretación constante desde cero de la información sensorial en términos de representaciones estructuradas, composicionales, es computacionalmente intratable. Se corre el peligro de caer en la explosión combinatoria de las hipótesis candidatas para explicar las observaciones (Perlovsky, 1998). Las estructuras relacionales conducen frecuentemente a problemas de *graph matching*, un ejemplo típico de NP-completitud (Garey y Johnson, 1979). Necesitamos alguna estrategia para luchar contra este tipo de complejidad, especialmente si deseamos construir sistemas capaces de interactuar con el entorno en tiempo real.

Hofstadter plantea un modelo computacional basado en la interacción no determinista de numerosos microprocesos elementales, que constantemente construyen, destruyen, reagrupan y reordenan estructuras o hipótesis de interpretación tentativas. Los conceptos se manejan a un nivel subsimbólico, con numerosas relaciones entre ellos y capacidad de interactuar con conceptos circundantes. La vía de escape ante la explosión combinatoria es la llamada *presión cognitiva*, consistente en considerar simultáneamente diferentes hipótesis con una urgencia que depende de los hallazgos obtenidos en el propio proceso perceptual. Para tratar todas estas alternativas se propone utilizar una ejecución probabilística desviada racionalmente, mejor que una totalmente determinista que podría dejar caminos cerrados desde el principio.

Algunos sistemas inspirados en esta aproximación han tenido éxito en microdominios de cierta complejidad. Casi todos los estudios relacionados, sin embargo, se despreocupan

abiertamente de la percepción a más bajo nivel. La naturaleza inherentemente distribuida de la propuesta, además, no se adapta demasiado bien al paradigma computacional secuencial, por lo que, en principio, no se pueden satisfacer fácilmente las restricciones de operación en tiempo real. Como veremos a continuación, esta tesis describe un intento de aplicación de las ideas discutidas en este apartado a un dominio precisamente caracterizado por estos dos problemas.

1.2. Motivación

Nuestro objetivo es construir un sistema completo de percepción visual de alto nivel, integrado en un robot que se moverá en un entorno real parcialmente estructurado, concretamente el interior de edificios. El comportamiento del sistema deberá obedecer a las propiedades estructurales locales percibidas en el entorno, en contraste con aproximaciones de naturaleza más reactiva frente a los estímulos inmediatos. Para conseguirlo, proponemos una arquitectura de procesamiento inspirada en la interacción de los flujos perceptivos ascendentes y descendentes, al estilo de la propuesta de Hofstadter, pero simplificada y adaptada a las restricciones impuestas por un agente autónomo.

1.2.1. Percepción visual

Nos centraremos en la percepción visual por varios motivos. El primero, y quizá el más importante, es que se trata de la modalidad sensorial más potente en las especies naturales. El escaso esfuerzo consciente que el ser humano emplea en obtener una información detallada de su entorno a partir de las señales luminosas que llegan a su cerebro, puede resultar engañoso a la hora de evaluar la extrema complejidad intrínseca del proceso. Se trata, además, de un campo de investigación muy abierto y con soluciones satisfactorias sólo en dominios muy restringidos. Incluso en tareas relativamente bien definidas de reconocimiento o clasificación de objetos en condiciones más o menos controladas, se presentan multitud de dificultades. La enorme variedad de situaciones que se pueden presentar (entornos cambiantes, condiciones ambientales no controladas, iluminación variable, reflejos, etc.) impide atacar el problema mediante técnicas de reconocimiento estadístico y requiere la mediación de conceptos de alto nivel.

La visión tiene también la ventaja de que las propiedades del sensor se comprenden bien gracias a la reciente aplicación de la geometría proyectiva a la disciplina. Bajo condiciones muy poco restrictivas, es posible inferir la estructura espacial del entorno a partir de múltiples imágenes de éste. Sin embargo, el extraordinario avance experimentado en los últimos años por las técnicas de reconstrucción tridimensional no ha tenido una respuesta comparable en el desarrollo de métodos adaptados al tiempo real (Davison y Kita, 2001). Muchos de los métodos requieren procesamiento *off-line*, donde primero se adquiere la secuencia completa

de imágenes para después tratarlas sin ninguna clase de restricción temporal. Esta metodología resulta inviable en la construcción de sistemas autónomos, donde nos vemos obligados a realizar el procesamiento de la información visual en el mismo momento en que es adquirida.

La estrategia de reconstrucción más utilizada por este formalismo presenta la peculiaridad de que desaprovecha en primera instancia gran parte de la información de coherencia espacial de la imagen. Se parte de características aisladas (normalmente esquinas o segmentos extraídos de la secuencia de imágenes), para rellenar posteriormente la representación obtenida hasta conseguir una reconstrucción densa de la escena (Pollefeys, 2000). Pero sobre modelos de este estilo no es fácil inferir aspectos cualitativos de su estructura, y éstos pueden ser importantes para tomar decisiones no puramente reactivas en aplicaciones de navegación visual como la que nos ocupa.

La alternativa que estudiaremos en este trabajo, por ello, da en cierto modo la vuelta a este paradigma. Mediante procesos de interpretación que tienen en cuenta la consistencia global de la imagen trataremos de inferir previamente las propiedades tridimensionales de la escena, para después corroborarlas o refutarlas con el movimiento a partir de las restricciones multivista. En el ámbito de la robótica móvil, pensamos que es preferible extraer inicialmente la estructura 3D a gran escala para deducir a partir de ahí sus posibles detalles, que fabricar una copia detallada de la realidad para posteriormente interpretarla.

1.2.2. Enfoque ingenieril

A veces resulta enriquecedor para el investigador en percepción artificial realizar una suerte de "ingeniería inversa" sobre los mecanismos desarrollados por la naturaleza. Pero los detalles de implementación no son tan importantes como el logro del objetivo final. Se trata de averiguar las propiedades esenciales de la solución, más que los accidentes de implementación. De hecho, a menudo hay diferencias sustanciales entre los mecanismos perceptivos de las distintas especies, dependiendo de las condiciones evolutivas en cada caso. Por supuesto, es lícito emplear la solución natural como inspiración en los sistemas artificiales, pero tampoco debe suponer un límite si, en determinados aspectos, la implementación concreta adopta un enfoque abiertamente ingenieril.

1.2.3. Estructura del espacio

Como argumentamos anteriormente, las representaciones abstractas constituyen el ingrediente fundamental en la predicción eficiente de los estímulos. Pero los conceptos elementales que forman la base de estas estructuras no se pueden deducir únicamente de la dinámica del espacio sensorial. Se plantea entonces una decisión de diseño de gran importancia. ¿Cuáles son los componentes elementales adecuados, o conceptos primitivos, sobre los que se construyen las representaciones?

En los sistemas biológicos un proceso de prueba y error evolutivo ha contribuido a generar

los mecanismos que realizan este trabajo. Las soluciones obtenidas no son necesariamente óptimas en algunos sentidos, pero esencialmente resuelven el problema planteado de modo satisfactorio. En el diseño de sistemas artificiales, no obstante, es evidente de que no estamos todavía en condiciones de automatizar esa etapa. De alguna forma, el agente autónomo debe ser provisto de algún tipo de conocimiento previo, innato, una cierta expectativa del tipo de situaciones que puede encontrar en su mundo.

Afortunadamente, el criterio propuesto en la sección 1.1.2 nos permite adoptar en realidad cualquier concepto elemental, con la única condición de que mediante su organización estructural se consiga comprender, en el sentido predictivo, el mundo externo. Es evidente que si las regularidades básicas elegidas son frecuentes en el entorno, podrán conseguirse eficientemente representaciones concisas y de gran capacidad predictiva (por ejemplo, a más largo plazo). En caso contrario serán más complejas, costosas de mantener y tendrán menor capacidad de anticipación. En cualquier caso, para que la elección resulte aceptable, el comportamiento del agente no debería sufrir una degradación catastrófica ante las ausencias, incluso moderadamente frecuentes, de las regularidades esperadas en su dominio de actuación.

1.2.4. Acción y percepción

La percepción está directamente acoplada con las posibilidades de actuación del agente. No en vano, deseamos percibir estructuralmente el entorno para tomar decisiones de movimiento sobre éste. Pero observando el proceso en sentido contrario, la propia acción revierte también de modo significativo en los procesos perceptivos. La interacción dinámica con el medio puede servir para corroborar o refutar las hipótesis de estructuración que sobre él se lanzan inicialmente. Percibir, de hecho, es la mayor parte del tiempo comprobar que todo sigue en su sitio, como debería estar, y, sólo esporádicamente, tener capacidad de reacción frente a cambios no esperados o lugares en principio desconocidos.

De alguna manera, esta idea puede ayudar a resolver el problema de la explosión combinatoria que surgiría al tener que evaluar todas las posibilidades de interpretación a cada instante. En lugar de ello, simplemente tendremos que evaluar aquel subconjunto de la masa sensorial que no queda explicado por lo percibido anteriormente. Acción y percepción, pues, se alían en un bucle de sintonía fina que evita la constante construcción de estructuras nuevas, al tiempo que añade la consistencia y continuidad necesarias para un comportamiento eficaz. Los robots dotados de cámaras son, en este sentido, un tipo de sistema artificial muy apropiado para estudiar esta integración.

1.3. Objetivos

Estamos interesados en diseñar un robot que navegue en tiempo real utilizando únicamente la información sensorial visual y odométrica, descartando explícitamente otro tipo de

sensores como el sónar, GPS, láser, etc. En este sentido, incluso, el uso de una sola cámara (en lugar de otro tipo de dispositivos ópticos más potentes, como torretas estéreo o trinoculares, por ejemplo) centra la atención en el problema de interpretación que se pretende estudiar. El sistema debe ser completo, tratando desde el bajo nivel, que procesa directamente el estímulo sensorial crudo, hasta el alto, capaz de provocar acciones coherentes con la estructura percibida del entorno, mostrando cierta capacidad de anticipación sobre éste.

El robot operará en entornos estructurados de interiores, pero naturales (sin marcas artificiales de ayuda a la navegación), y que son, *a priori*, desconocidos. No obstante, incluiremos también la interpretación de señales externas, como modo de añadir más variabilidad al dominio y de enfatizar que a pesar de la deformación de perspectiva, se puede conseguir fácilmente su correcta interpretación a partir de la estructura del entorno inferida.

Creemos que el dominio es lo suficientemente variado como para plantear un desafío interesante. Deseamos que el robot interprete el entorno en términos de planos, en el sentido de que los perciba y organice de forma coherente tanto espacial como temporalmente. Deberá ser capaz de extrapolar de esta organización conceptos naturales del dominio tales como direcciones preferentes, espacios cerrados o situaciones familiares, por ejemplo. En cualquier caso, la hipótesis de entorno planar tampoco prohibe, como veremos, la existencia de otro tipo de objetos, estáticos o dinámicos, ajenos al modelo. Simplemente se asume que, en primera aproximación, el fondo de la escena está constituido fundamentalmente por planos. Sobre este fondo podría haber una serie de detalles considerados en principio de segundo orden y que, al menos bajo la hipótesis inicial, podrían ser aislados del fondo y situados de modo aproximado dentro de la escena sin producir una degradación del comportamiento.

En contraste con un comportamiento reactivo frente a los estímulos, trataremos de generar decisiones de locomoción de acuerdo con la estructura del entorno percibida mediante los mecanismos de interpretación. No estaremos tan interesados en los problemas de control del agente como en su uso subsidiario como apoyo a la percepción, siempre en un esquema de actuación local, inmediato. Cae también fuera del alcance de este trabajo, por tanto, la generación de comportamiento volitivo a más largo plazo.

Desglosando en objetivos más concretos, el robot diseñado debería mostrar las siguientes habilidades:

Reconstrucción 3D autocalibrada: El sistema debería ser capaz de realizar en tiempo real reconstrucciones tridimensionales euclídeas de entornos desconocidos, pero que se ajustan a un dominio de aplicación concreto, los entornos estructurados del interior de edificios. Esta capacidad no tendría que depender del conocimiento previo de las características del sensor. En lugar de ello, los parámetros de calibración de la cámara serán deducidos durante la misma navegación.

Interpretación de alto nivel: El robot deberá categorizar flexiblemente algunas situaciones

locales típicas del dominio (pasillos, esquinas, espacios abiertos, habitaciones, etc.), autoposicionándose en cada una de ellas. Dentro de estas situaciones, el sistema deberá también detectar y clasificar cierto tipo de señales visuales externas, para cuya interpretación será necesaria su contextualización en la escena en la que están ubicadas. Teniendo en cuenta este contexto, sería asimismo interesante que pudiese localizar otros elementos extraños en el entorno al objeto de tenerlos en cuenta como obstáculos.

Navegación interpretativa: La generación de órdenes de movimiento se realizará en base a la interpretación anterior. El modo de navegación, en principio, estará preprogramado en función de la situación, pero esto no supone ninguna limitación, y podría variar en función del objetivo. El sistema podría programarse para explorar el entorno (navegando siempre hacia lo más desconocido), o estar dirigido a la obtención de reconstrucciones completas (inspeccionando en detalle determinadas zonas de incertidumbre en la representación), por ejemplo. La interpretación, por tanto, debe apoyarse en el movimiento, pero sin depender del modo de navegación concreto. En última instancia, incluso, el robot debería igualmente ser capaz de interpretar la situación a través de un movimiento inducido, no controlado pero conocido a través de la odometría.

Anticipación en la percepción: La generación de hipótesis debe consumir la menor cantidad de recursos computacionales. La mayor parte del tiempo se empleará en mantener actualizada la hipótesis anterior, y sólo se añadirá algo nuevo a la representación cuando haya cambios importantes en la estructura de la escena.

Acumulación de conocimiento a través del movimiento: La interpretación local de cada escena se debe ir acumulando en una representación a largo plazo actualizada constantemente por la odometría. Esta representación podría ser una reconstrucción euclídea completa, con las situaciones tipo reconocidas superpuestas en los lugares correspondientes e interconectadas entre sí (mapa euclídeo-topológico).

Todas estas características deben integrarse en un prototipo funcional desarrollado sobre hardware estándar. Los algoritmos y soluciones propuestas para la implementación de los distintos procesos perceptivos se adaptarán para ejecutarse eficientemente sobre este hardware, en nuestro caso ordenadores personales basados en microprocesadores convencionales con tarjetas de adquisición de vídeo sin capacidad de procesamiento independiente. La plataforma de evaluación de la arquitectura será un robot móvil comercial, de la gama media/baja, dotado de un equipo de computación como el descrito, y conectado con el exterior a través de enlaces inalámbricos.

1.4. Plan de trabajo

El prototipo desarrollado para lograr estos objetivos se asentará sobre cuatro pilares básicos:

1. Un mecanismo versátil y eficiente de extracción de características de la imagen en el procesamiento de bajo y medio nivel, que asegure la potencia expresiva suficiente para atacar los problemas de los niveles posteriores sin afectar a los requerimientos de tiempo real de la navegación. La descripción de completa de este mecanismo se realizará en el segundo capítulo.

2. Un repertorio de técnicas de autocalibración de las cámaras que, apoyadas en la herramienta teórica de la geometría de múltiples vistas, sean capaces de funcionar durante la misma operación del robot, incluso en condiciones de difícil aplicabilidad de las técnicas clásicas. Las técnicas utilizarán únicamente la información odométrica proporcionada por los sensores de movimiento y las características extraídas del propio entorno de navegación, sin la ayuda de plantillas o cualquier otro tipo de conocimiento externo sobre éste. Al desarrollo de estas técnicas se dedica el capítulo tercero.

3. Un paradigma interpretativo de las escenas de entornos estructurados sobre las que navegará el agente autónomo, y que, a través de un esquema de generación, seguimiento, y corroboración o refutación de hipótesis conceptuales sobre la estructura de las mismas, permitan un cierto comportamiento deliberativo en la navegación, más allá de las conductas meramente reactivas. Las bases de este paradigma se explican en el capítulo cuarto.

4. Finalmente, el desarrollo de una arquitectura de acción y percepción que, integrando los componentes hardware y software necesarios en un esquema suficientemente modular y flexible, permita el funcionamiento robusto y eficiente del prototipo sobre entornos estructurados reales, mostrando las deseadas características de autonomía, anticipación y continuidad perceptiva en la navegación. El capítulo quinto describe las peculiaridades de esta arquitectura.

CAPÍTULO II

"Reptiles" (detalle), M.C. Escher, 1943

Niveles inferiores de la percepción visual

But life is short and information endless... Abbreviation is a necessary evil, and the abbreviator's business is to make the best of a job which, although intrinsically bad, is still better than nothing...
A. HUXLEY. *Brave New World Revisited.*

2.1. Introducción

Los sistemas de visión artificial son esencialmente intensivos en cómputo. En particular, las operaciones relacionadas con el procesamiento de bajo y medio nivel de la imagen consumen habitualmente una gran cantidad de recursos. No en vano, son éstas las etapas que tienen que enfrentarse con las grandes masas de datos sin procesar, directamente procedentes de los sensores visuales. La cantidad de información producida en tiempo real puede llegar a ser extraordinariamente voluminosa. La cámara de nuestro sistema, por ejemplo, proporciona unos 8 MB de información cambiante por segundo[1]. Evidentemente, si se pretende utilizar la interpretación visual para interactuar en tiempo real con el entorno, esta secuencia de imágenes debe ser tratada de modo muy eficiente desde el punto de vista computacional, tanto en términos de almacenamiento como, sobre todo, de procesamiento.

La implementación de los niveles inferiores de la percepción será, pues, clave en el desarrollo de la arquitectura. Estas etapas, encargadas de detectar las características de interés de la imagen eliminando la información que resulta redundante o incluso ruidosa, deberán cumplir en nuestro caso unas fuertes restricciones temporales de cara a mantener operativo el sistema autónomo. En este capítulo abordaremos este problema, siguiendo el esquema de discusión que se detalla a continuación. En primer lugar realizaremos una breve revisión de los

[1] Imágenes RGB de tamaño 288×384 píxeles a 25 *frames* por segundo (*fps*), con tres *bytes* por píxel.

métodos generales de segmentación de imágenes más utilizados, para particularizar después en aquellos más relacionados con nuestra línea de trabajo. Dado que una revisión exhaustiva sería prácticamente inviable, haremos énfasis en aquellas técnicas que, por su eficiencia computacional, son más populares entre la comunidad investigadora de robots guiados visualmente. Las ventajas e inconvenientes de cada paradigma nos servirán, en cierto modo, como justificación de nuestra aproximación, la cual será ampliamente descrita en un apartado posterior. Después se hará una defensa de la técnica propuesta resumiendo sus principales cualidades: la potencia expresiva de las características extraídas, la velocidad en el cómputo de las mismas y su adecuación al dominio de aplicación que nos ocupa, los entornos estructurados. El capítulo se cierra con una serie de estudios experimentales que ilustran dichas ventajas y validan el procedimiento descrito. El lector familiarizado con las técnicas de procesamiento de imagen de bajo nivel puede proceder directamente a la sección 2.4.

2.2. Aproximaciones a la segmentación de imágenes

La segmentación de imágenes consiste en la extracción de una determinada información de interés de la *imagen cruda*, entendida ésta como una señal, posiblemente multivaluada, definida sobre un dominio bidimensional discretizado en píxeles[2]. Evidentemente, y dependiendo del dominio de aplicación, caben multitud de interpretaciones para lo que significa "la información de interés". Ésta suele estar relacionada con la localización de elementos de la imagen que tienen alguna propiedad relevante para dicho dominio. Por ejemplo, si lo que queremos es extraer información tridimensional de una escena a partir de una secuencia de imágenes, puede interesarnos detectar y seguir a través de la misma un conjunto de puntos que nos permitan trabajar después con restricciones proyectivas, a partir de las cuales obtener la estructura de la escena, e incluso, colateralmente, el movimiento y las características intrínsecas del sensor óptico con el que se tomó la secuencia. En este tipo de aplicación, la segmentación adecuada consistiría en la localización con la mayor precisión de posible de dichas *esquinas* o "*puntos calientes*". Una situación bien distinta sería la planteada en una aplicación de seguimiento de un objeto de un determinado color, por ejemplo. En este caso, una segmentación por regiones de píxeles con colores más o menos compatibles con el objetivo resultaría más conveniente. Si, por el contrario, se intentase el reconocimiento del objeto por su forma, entonces la técnica más apropiada estaría entre aquellas capaces de extraer con la mayor regularidad y precisión posible el contorno del mismo. Vemos, por tanto, que el concepto

[2] Algunos autores reservan el término *segmentación* a la partición de una imagen en regiones de píxeles excluyentes, cubriendo o no la totalidad de la misma. Otros asocian dicha palabra a la detección de rectas (*segmentos*) en imágenes de borde. En general, nosotros utilizaremos el término *segmentar* en su sentido más amplio, el de detectar partes de interés dentro de la imagen. A dichas partes, que podrían ser puntos, rectas, zonas, contornos, etc., se les suele referir en la literatura con el nombre genérico de características (*features*). Si las características buscadas son zonas de píxeles contiguos, o agrupaciones en línea recta de píxeles de borde, en las acepciones específicas anteriormente comentadas, nosotros hablaremos de *segmentación regional* y *detección de segmentos*, respectivamente.

de segmentación tiene un sentido bastante amplio, y que el procedimiento concreto a aplicar dependerá en gran medida del uso posterior que quiera darse a la información extraída.

Intentar realizar aquí una revisión completa de la multitud de técnicas descritas en la literatura sobre el tema queda completamente fuera de nuestros objetivos. Tampoco faltan buenos artículos en los que éstas se compendian, y a los cuales se puede remitir el lector (Pal y Pal, 1993; Haralick y Shapiro, 1985). No obstante, sí que ofreceremos una perspectiva general y una clasificación de las mismas, al objeto de justificar nuestra elección final teniendo en cuenta el ámbito de aplicación que nos ocupa. Ilustraremos cada técnica con alguna referencia de ejemplo, siempre con la intención de clarificar la discusión, más que de enumerar la gran cantidad de variantes posibles. Finalmente, dentro de cada uno de los grupos destacaremos también aquellos procedimientos que, por sus cualidades, son susceptibles de ser aplicados en robots autónomos.

Comenzamos la clasificación dividiendo los métodos de segmentación en tres grandes grupos, según el tipo de datos de imagen sobre el que trabajan: *píxeles*, *bordes* o *regiones*.

2.2.1. Extracción de puntos de interés

Enmarcamos en este primer grupo aquellas técnicas que tratan de localizar los píxeles que resultan más salientes dentro de la imagen, atendiendo a la configuración espacial de niveles de luminosidad en su entorno cercano. Estos métodos son muy utilizados en tareas de reconstrucción tridimensional de escenas a partir de múltiples vistas 2D. Una referencia clásica es el artículo de Longuet-Higgins (1981). En general, éste es el preprocesamiento más empleado en la resolución de aquellos problemas de la visión por computador que se apoyan la geometría proyectiva como principal herramienta. Algunos ejemplos dignos de mención son la estimación de la matriz fundamental, la calibración de sensores, o la estimación de la traslación de la cámara y la estructura de la escena aprovechando las restricciones proyectivas existentes entre las distintas vistas. Una extraordinaria referencia en relación a estos temas es el libro de Hartley y Zisserman (2000). En general, la mayor ventaja de este tipo de métodos es que no suelen ser demasiado intensivos en cómputo, por lo que no ofrecen grandes inconvenientes en su aplicación bajo restricciones de tiempo real, tal y como exige la navegación visual.

Son varias las técnicas existentes para determinar estos puntos calientes, o esquinas (*corners*), como también se les conoce en la literatura. La gran mayoría trabajan intentando encontrar lugares de la imagen donde el gradiente es apreciable en al menos dos direcciones distintas. Quizá la más conocida sea el detector de Harris y Stephens (1988). Esta técnica trabaja computando la matriz de momentos del gradiente en el entorno local de cada píxel, para calcular posteriormente los dos valores propios de esta matriz. Si el valor propio menor es mayor que un cierto umbral, podemos deducir que en la zona hay al menos dos bordes con direcciones distintas, y que por tanto existe una esquina.

Otros métodos trabajan con algún tipo de filtro no lineal, como el de la mediana, por

ejemplo, aprovechando el hecho de que este operador de supresión de ruido tiene el efecto secundario de "redondear" las esquinas. Restando la imagen filtrada de la original, pues, pueden detectarse con relativa facilidad estos puntos de interés (Paler *et al.*, 1984). El inconveniente es la gran sensibilidad al posible ruido. Una técnica más moderna, también basada en una operación no lineal sobre la imagen, es el operador denominado SUSAN (del inglés *Smallest Univalue Segment Assimilating Nucleus*, o segmento monovaluado más pequeño similar al núcleo) (Smith y Brady, 1997). El propio nombre del método da una pista sobre su modo de funcionamiento. Se basa en la idea de encontrar, para cada punto de la imagen, el área formada por todos los píxeles del entorno local cuyos valores de gris distan del valor del píxel central (el *núcleo*) menos de una cantidad determinada[3]. Al área así determinada se le denomina USAN, es decir, segmento monovaluado similar al núcleo. Una vez determinada esta zona se calculan sus estadísticos de segundo orden, a partir de los cuales se pueden obtener el área, la posición media y la orientación aproximada de la zona. Si el área es grande y centrada en el núcleo, entonces se trata de una zona sin detalle. Si, por el contrario, es pequeña, pero con el centro alejado del núcleo, estamos en presencia de una esquina. Usando razonamientos similares sobre la forma y la posición del área el procedimiento puede servir otras tareas, como encontrar bordes, segmentar por regiones o suavizar imágenes, por ejemplo.

Las esquinas también pueden ser localizadas a partir de las secuencias de píxeles que marcan las siluetas de los objetos de la imagen. Una idea es extraer los puntos de mayor curvatura en estos contornos, utilizando por ejemplo técnicas de aproximación poligonal (Rosin, 1997b). Otra idea similar, que será la que nosotros explotaremos, es obtener la localización de las esquinas mediante la intersección de segmentos rectos cuyos extremos se tocan o están muy cercanos en la imagen. Los métodos de extracción de líneas necesarios para este tipo de aproximación serán comentados en un apartado posterior.

Un requerimiento muy importante en este tipo de técnicas es la precisión en la localización de los puntos de interés. La principal razón es que los posteriores procedimientos de reconstrucción tridimensional pueden ser muy sensibles a pequeñas variaciones en sus posiciones. De ahí que algunos de los algoritmos intenten incluso lograr precisiones a escala subpíxel, aunque esto último pueda conllevar un mayor costo computacional. Igualmente determinante resultará la capacidad de los algoritmos de lograr un buen porcentaje de aciertos en la detección, sin generar un alto número de falsos positivos. En tercer lugar, y puesto que el procesamiento posterior más habitual consiste en el seguimiento de las esquinas detectadas (*tracking*), o el establecimiento de correspondencias entre ellas, si vienen de cámaras distintas (*matching*), a veces resulta interesante que los métodos consideren también la estructura local de la imagen en torno a cada característica detectada. Una aproximación muy común consiste en utilizar los valores de gris de las ventanas de píxeles centradas en cada uno de los puntos extraídos. Con ellas se puede calcular una medida de correlación para cada par de esquinas

[3]La elección de este umbral estará íntimamente relacionada con la resistencia al ruido del algoritmo, es decir, su habilidad para no caer en falsos positivos.

candidatas a ser emparejadas entre las imágenes. Shi y Tomasi (1994), por ejemplo, extienden esta idea con un modelo de deformación afín entre las ventanas, dado que un modelo estrictamente traslacional puede no funcionar bien con secuencias largas. Esta última técnica, basada en el clásico trabajo anterior de Lucas y Kanade (1981), es refinada por Tommasini *et al.* (1998) para hacerla más robusta frente a falsos emparejamientos.

La eficiencia computacional hace a estos métodos muy atractivos en el ámbito de los sistemas autónomos guiados por cámaras. No obstante, dado que la búsqueda de precisión está a menudo penalizada con una pérdida de eficiencia, la mayoría de los trabajos descritos utilizan las versiones más simples de la detección de esquinas, a fin de poder trabajar a velocidades de varios *frames* por segundo. Bouguet y Perona (1995), por ejemplo, resuelven el problema de la estimación del movimiento de una plataforma con una sola cámara mediante el seguimiento de puntos en la imagen. Las esquinas se detectan y emparejan con el método de Lucas y Kanade, y después se siguen usando un filtro de Kalman extendido (Welch y Bishop, 1995) para aumentar la robustez. En un tipo de aplicación distinta, Beardsley *et al.* (1995) describen una plataforma con un par estéreo que hace reconstrucciones afines a partir del *matching* de esquinas entre imágenes. Estas reconstrucciones se usan posteriormente para navegar hacia espacio libre, o bien manteniendo el robot centrado entre dos obstáculos. El sistema presentado por Robert *et al.* (1997), también basado en el seguimiento de esquinas, tiene unas características muy similares. Davison extiende esta filosofía haciendo énfasis en la visión activa (Davison y Murray, 1998; Davison y Kita, 2001). Para ello, en la etapa de extracción de características se eligen cuáles de los puntos detectados son mejores candidatos a seguir para aumentar la precisión en la estimación de la posición del robot. Al mismo tiempo se va decidiendo sobre la marcha cuando hay que ir añadiendo nuevas características. Mediante el tratamiento de la incertidumbre con un filtro de Kalman extendido, el sistema también puede determinar qué puntos aportan más información a la hora de eliminar posibles ambigüedades.

A veces el objetivo de la localización y el seguimiento de puntos no es tanto la reconstrucción explícita de la escena como una aplicación más inmediata a la navegación. Por ejemplo, en los trabajos de Taylor *et al.* (1999) y Swain y Devy (1997) las esquinas se utilizan para cerrar un lazo de control que coloque al robot en un lugar determinado de la escena, suponiendo que se tiene información sobre la estructura de ésta última. La idea consiste en relacionar directamente el movimiento de la cámara con la variación de la proyección en imagen que experimentan aquellos puntos de la escena cuya posición 3D es conocida *a priori*. El enfoque es conocido con el nombre genérico de *visual servoing*. Este tipo de control directo es sobre todo utilizado en el control de robots manipuladores (Ruf y Horaud, 2000; Horaud *et al.*, 1998).

2.2.2. Extracción de regiones

La extracción de esquinas tiene la indiscutible ventaja de la eficiencia computacional. Sin embargo, como técnica de segmentación es quizá la que obtiene resultados menos expresi-

vos. Las esquinas, en sí mismas, aportan poca información más allá de su mera posición en la imagen. Aunque utilizando la geometría de múltiples vistas pueden practicarse reconstrucciones tridimensionales, las nubes de puntos 3D siguen careciendo, al menos *a priori*, de una estructura que permita interpretar la escena.

En este sentido, una segmentación regional, que trata de encontrar zonas relevantes de píxeles contiguos, tiene la ventaja de que a la posición de la característica segmentada se añade una cierta información de forma, color o textura. Esta expresividad adicional puede hacer más atractiva a esta alternativa, sobre todo si se piensa en el posterior procesamiento de más alto nivel, que puede incluir el reconocimiento de formas, la interpretación de situaciones, etc. Lamentablemente, el precio a pagar es de nuevo el incremento en el coste computacional. Las técnicas eficientes de segmentación por regiones suelen trabajar localmente, lo que puede producir problemas en la determinación precisa de las zonas más grandes, dificultando el trabajo de interpretación posterior. La única manera de solucionar este problema es tratar de modo más global las interacciones entre píxeles vecinos. Esto nos puede llevar a soluciones computacionalmente prohibitivas, sobre todo si el dominio de aplicación exige respuestas rápidas a imágenes constantemente cambiantes, como es nuestro caso.

Un primer grupo de estos métodos tiene como denominador común el umbralizado del valor de los píxeles de la imagen. Algunas de estas aproximaciones trabajan buscando valles sobre el histograma de grises, intentando estimar el valor de corte óptimo. Dada su sencillez, la técnica es bastante rápida, aunque puede tener serios problemas en imágenes con mucho detalle, donde la distribución no muestra modas bien definidas. Una posible solución es trabajar sobre histogramas locales, haciendo el umbralizado adaptativo a las distintas zonas de la imagen. Naturalmente, esto complica el procedimiento al multiplicarse las búsquedas de mínimos a muchas zonas, dependiendo de la precisión requerida. Un buen resumen de diversas variantes de estas técnicas puede encontrarse en (Davies, 1997).

Si se aprovecha el color, existe la posibilidad de aplicar técnicas de agrupamiento (*clustering*) sobre el espacio de valores que puede tomar cada píxel. El método puede funcionar bien sobre imágenes sin demasiada textura, y con colores bien definidos. Conociendo *a priori* los colores o texturas cuyas zonas se quieren extraer, se puede, por ejemplo, entrenar una red neuronal a tal efecto. Blanz y Gish (1990) describen un sistema de estas características, que una vez entrenado es capaz de segmentar imágenes en tiempo real. En (Ruiz *et al.*, 1998) describimos otro ejemplo de localización en imágenes de objetos de colores más o menos variables, como aplicación de un mecanismo eficiente de inferencia probabilística con tratamiento de la incertidumbre.

También es posible realizar la segmentación por color de modo no supervisado. Comaniciu y Meer (1997), por ejemplo, usan la técnica de *mean shift* (literalmente, desplazamiento hacia la media) para ir creando los *clusters* automáticamente. Se trata de un procedimiento no paramétrico que encuentra modas en una función de densidad multidimensional, en este caso, el espacio de color considerado (RGB, HLS, LUV, o cualquier otro). Está basado en la es-

timación local del gradiente de dicha densidad, cogiendo una muestra aleatoria y un núcleo (*kernel*) alrededor de ella. La media de las muestras englobadas dentro de dicho núcleo apunta en la dirección de una moda. Una vez localizada ésta, se localiza el *cluster* correspondiente, se eliminan todas sus muestras, y se repite el proceso con las restantes. El método funciona bastante mejor que el umbralizado simple, aunque para evitar artefactos en las regiones hay que tener en cuenta ciertas restricciones de coherencia espacial en el etiquetado. El método puede acelerarse submuestreando la imagen original, a costa de la definición en el segmentado.

El siguiente tipo de técnica intenta resolver las limitaciones de las anteriores tomando en cuenta las relaciones espaciales de cada píxel con sus vecinos a la hora de particionar la imagen. Es el llamado crecimiento de regiones, en el que píxeles de similar intensidad van agrupándose para formar zonas cada vez mayores. A menudo se debe iterar sobre la solución obtenida en pasos anteriores para lograr resultados aceptables, de nuevo afectando a la eficiencia del sistema. Por ejemplo, Mirmehdi y Petrou (2000) utilizan una técnica de este tipo, que segmenta adecuadamente incluso zonas texturizadas teniendo en cuenta medidas psicofísicas de apariencia de color. La relación espacial entre píxeles se tiene en cuenta a través de un tratamiento multiescala de la imagen, comenzando a mayor escala para resolver después sucesivamente los detalles. La velocidad de procesamiento, sin embargo, es excesivamente lenta como para permitir su uso en un sistema con restricciones de tiempo real.

Los métodos que obtienen mejores resultados son aquellos en los que los píxeles interactúan de un modo más global. Un modo de tener esto en cuenta es plantear el problema como una optimización sobre un grafo cuyos nodos son los píxeles, y cuyos arcos conectan píxeles relativamente cercanos en la imagen, hasta un determinado umbral de distancia. A cada uno de estos arcos se le asigna una medida que combina similitud (de nivel de gris, color, textura, etc.) con proximidad. El problema de optimización resultante se resuelve entonces usando técnicas clásicas de investigación operativa, como la minimización de una función de energía (Jermyn y Ishikawa, 1999), la partición de grafos mediante corte mínimo (Wu y Leahy, 1993; Shi y Malik, 2000), o similares. Cuanto mayor es el radio de conexión resultante, y por tanto el grado de conectividad del grafo, de mayor calidad resulta la solución obtenida, aunque, lógicamente, también se aumenta el coste computacional. Evidentemente, y salvo para simplificaciones muy grandes del problema, todas estas técnicas son aplicables sólo en segmentación de imágenes individuales, donde el tiempo de respuesta no es importante. Quedan descartadas, por tanto, para aplicaciones de navegación como la que nos ocupa.

En una línea parecida se hallan las segmentaciones regionales en las que se modela probabilísticamente la interacción entre píxeles vecinos. Es común, por ejemplo, tratar a la imagen como un campo de Markov aleatorio (MRF, *Markov Random Field*). Sobre esta estructura se itera buscando una configuración que minimice paulatinamente una función de energía calculada sobre el conjunto de píxeles, utilizando diferentes heurísticas para no caer en mínimos locales (Geman y Geman, 1984; Modestino y Zhang, 1992). La diferencia con las funciones de energía anteriores está en el proceso de minimización iterativo por convergencia, frente

a la solución en forma cerrada de los métodos basados en la investigación de operaciones. Sin embargo, estos procesos de relajación exigen tiempos de cómputo tan grandes que prácticamente anulan la posibilidad de emplearlos en robots. En realidad su uso más común es el análisis detallado de imágenes estáticas. Una posibilidad apuntada por algunos de estos autores sería desarrollar hardware dedicado que realizase la segmentación distribuyendo el trabajo entre múltiples unidades de proceso elementales, liberando así al procesador principal de esta tarea.

La segmentación por regiones no es tan utilizada en sistemas de navegación visual como la extracción de esquinas. La razón es que, como ya se ha comentado, lograr altas velocidades de procesamiento con estas técnicas es difícil, al menos para los métodos que pretenden obtener descripciones morfológicas de calidad. Aún así, algunos autores emplean este tipo de segmentación en aplicaciones en tiempo real. Bruce *et al.* (2000), por ejemplo, describen un sistema capaz de dividir en regiones disjuntas imágenes completas de tamaño mediano al ritmo de captura (30 *fps*) con una sola CPU estándar. La propuesta combina una técnica novedosa de umbralizado, basada en rápidas operaciones lógicas, con el crecimiento de regiones. Pero la aplicación es segmentar imágenes para robots que juegan al fútbol, en las cuales los objetos a seguir (la bola, el resto de robots, las porterías, etc.) son pintados con colores llamativos, fácilmente distinguibles del entorno y entre sí. Para escenas más realistas, se reconoce que la aproximación no es de mucha utilidad. Otra idea bastante simple y eficiente es la seguida por Howard y Kitchen (1997), que disponen de una cámara montada en un vehículo móvil con un ligero ángulo de picado sobre el suelo. Éste último se aisla del resto de la imagen utilizando información previa sobre su color, delimitando la zona correspondiente. De esta manera se consigue un sistema capaz de navegar hacia el espacio libre mientras construye un mapa del entorno, utilizando muy pocos recursos de cómputo.

Más habitual es la utilización de las regiones en las aplicaciones de seguimiento, donde no se intenta particionar toda la imagen, sino sólo localizar y seguir en el tiempo una determinada zona de interés dentro de una secuencia. Por ejemplo, Buluswar y Draper (1998) describen un sistema que reconoce colores en tiempo real, aprendiendo a partir de ejemplos, para luego clasificar los píxeles usando árboles de decisión. Los entornos de aplicación recomendados son la monitorización de vehículos en autovías y de objetivos militares. En la misma línea está el trabajo de McKenna *et al.* (1999), donde se emplea un modelo de mezcla de componentes gaussianas para los píxeles del objeto controlado. Esta técnica de modelado permite seguir objetos algo más complejos, compuestos incluso de varios colores. El sistema funciona a varios *frames* por segundo sobre una sencilla CPU de propósito general, a pesar de utilizar un filtro de Kalman como fuente adicional de robustez en el *tracking*. Comaniciu *et al.* (2000) muestran otra aplicación de seguimiento de personas en tiempo real, también con colores variables, pero en este caso trabajando sobre la distribución conjunta del espacio bidimensional de coordenadas de imagen y el color de los píxeles. De nuevo, el procedimiento empleado es el *mean-shift* anteriormente comentado. Waldherr *et al.* (1998) utilizan una técnica similar para

un robot que sigue personas utilizando el color y la posición adquiridos directamente de la imagen, a través de un modelo mixto que también se adapta a cambios de iluminación.

Por último, la segmentación por regiones suele también resultar útil cuando el objetivo es seguir pistas artificiales (*landmarks*) introducidas en el entorno para guiar al robot. Un ejemplo típico es el descrito por Beccari *et al.* (1997). Se trata de un robot que sigue líneas gruesas dibujadas en el suelo, al tiempo que interpreta señales de colores también resaltados. Tanto las señales como la línea a seguir son separadas del resto del entorno siguiendo una técnica rápida de umbralizado adaptativo.

2.2.3. Extracción de bordes

Quizá la alternativa más utilizada en visión artificial para segmentar una imagen sea la detección de bordes (*edge detection*). La principal razón es que esta solución constituye un adecuado compromiso entre la expresividad, de la que carecen los puntos, y la eficiencia, en la que muestra su debilidad la segmentación por regiones. En general, tanto el cómputo como la información redundante de la imagen se pueden reducir bastante si nos decidimos por el tratamiento de la misma utilizando sólo sus bordes más salientes. Es lógico, por tanto, que la literatura relacionada con el tema sea muy extensa.

La aproximación a la detección de bordes más utilizada está basada en la estimación del gradiente en cada punto de la imagen. Lo habitual es utilizar un par de máscaras lineales diseñadas para responder a cambios bruscos de la señal en dos direcciones perpendiculares. En este sentido son muy conocidos los operadores de Roberts, Sobel o Prewitt, referenciados en cualquier libro básico de procesamiento de imagen o visión por computador (véanse, por ejemplo, (Gonzalez y Woods, 1993) o (Shapiro y Stockman, 2000)). A partir de las respuestas de estos filtros pueden computarse la magnitud y la dirección del gradiente de la imagen en cada posición. Los bordes se localizan allí donde se detectan máximos locales en esta estimación. A veces, la propia dirección perpendicular al gradiente se usa para seguir el contorno de dichos bordes.

Una alternativa es usar máscaras que buscan la respuesta a una dirección o conjunto de direcciones en particular. Los filtros de Gabor (Granlund y Knutsson, 1995) son muy populares en este sentido, puesto que se puede parametrizar la sensibilidad del filtro no sólo a la orientación, sino también a la forma concreta que puede tomar el borde (rampa o pico) y a la cantidad de ruido esperado en la imagen de entrada.

Es también de gran importancia histórica el operador de Marr y Hildreth (1980). Este operador localiza los bordes allí donde se producen los cruces por cero de la laplaciana de la imagen original, previamente suavizada con un filtro gaussiano. Por ello, a este filtro se le conoce también con el nombre de LOG, del inglés *Laplacian of Gaussian*. Tiene la ventaja teórica de no necesitar el umbralizado requerido por los métodos de gradiente, pero, como contrapartida, la técnica es en realidad bastante sensible al tamaño de la máscara de filtrado

utilizada. En general, casi todos los métodos de detección de bordes tienen el problema de la estimación adecuada de umbrales, o, en su defecto, de algún otro parámetro que resulta clave en la precisión de la técnica de detección. Rosin (1997a) resume algunos de los intentos más interesantes en la automatización de estos procesos de umbralización.

Directamente relacionado con la localización de los bordes está el problema de su agrupamiento. Los fragmentos localizados por los filtros anteriores (frecuentemente denominados *edgels*) son obtenidos sólo localmente, y a menudo los bordes importantes aparecen en compañía de otros espurios, que deben ser rechazados. Sin embargo, tal y como observaron los psicólogos de la escuela Gestalt, los sistemas biológicos pueden agrupar perfectamente estos bordes en siluetas aisladas, aún en etapas previas al reconocimiento del objeto en cuestión. Algunos autores llaman *saliencia* a esta propiedad, que destaca algunos contornos de la escena en detrimento de otros bordes irrelevantes. Existen diversas propuestas de modelos computacionales que tratan de desarrollar esta habilidad. Uno de los más estimulantes es la red de saliencia propuesta por Shashua y Ullman (1991), estudiada en profundidad por Alter y Basri (1998). Esta red es, básicamente, un grafo-retícula superpuesto sobre una imagen de gradiente más o menos ruidosa, provista de información de orientación local, donde cada arco del grafo es sensible inicialmente a la magnitud del borde que hay debajo de él, y cada nodo tiene una capacidad elemental de proceso. La red ejecuta un proceso iterativo en el que la saliencia se va propagando entre arcos adyacentes, de modo estrictamente local en cómputo, pero con consecuencias globales conforme avanza el proceso iterativo. Las estructuras salientes van agrupando los bordes que las componen al tiempo que ven reforzada su saliencia. Por construcción, la red tiende a reforzar estructuras continuas y de suave curvatura. Aunque, para ser ejecutada en una sola CPU, la aproximación es computacionalmente muy costosa, la idea resulta interesante por ser estrictamente distribuida, y por tanto susceptible de ser implementada en hardware de modo masivamente paralelo.

En una línea similar se encuentran los agrupamientos basados en modelos neuronales, como el de Williams y Jacobs (1997b), para el que también se propone una arquitectura de cómputo distribuida (Williams y Jacobs, 1997a). Otros enfoques más algorítmicos de agrupamiento de contornos basados en la propagación de la saliencia a partir de medidas de orientación locales son los trabajos de Amir y Lindenbaum (1998) y Medioni *et al.* (2000). En general, sin embargo, todos estos esquemas son inviables en aplicaciones de tiempo real si no se dispone de dispositivos de procesamiento dedicados.

Son precisamente las razones de eficiencia las que hacen del operador de Canny (1986) el algoritmo por excelencia para localizar y agrupar los píxeles de borde en una imagen. En el diseño de este operador se intentaron optimizar simultáneamente tres criterios: buena respuesta (es decir, que si realmente hay un borde, éste se localice, y que no se den falsos positivos), buena localización (que el píxel de borde se sitúe en una posición lo más ajustada posible al borde real) y unicidad (evitar respuestas múltiples en un sólo borde). Realizando un completo análisis funcional se deriva una operación de filtrado óptima en el sentido de

los criterios anteriores. El filtro obtenido puede aproximarse mediante máscaras que calculan las derivadas en las direcciones X e Y de la imagen original, previamente suavizada con una máscara gaussiana. Se buscan entonces máximos locales del gradiente utilizando la información de magnitud y dirección de éste, con el fin de evitar múltiples respuestas. Después se agrupan los puntos obtenidos, siguiendo una trayectoria perpendicular a la dirección de máxima pendiente en cada borde. Un proceso de histéresis durante dicho seguimiento reduce la probabilidad de encontrar falsos bordes, sin perder la sensibilidad a trozos de contornos reales que puedan mostrar una respuesta localmente más débil. Una ventaja añadida del operador de Canny es que no sólo proporciona una imagen binaria con la localización de los bordes, sino que también se realiza un seguimiento de los mismos: la salida obtenida por el procedimiento está formada por cadenas de píxeles conectados posiblemente correspondientes a siluetas de objetos, y que por tanto pueden ser de utilidad en posteriores etapas de procesamiento, como el reconocimiento de formas, la extracción de segmentos, etc.

Así como en el ámbito de la segmentación por regiones se diseñaban procedimientos para seguir un objeto con unas determinadas propiedades de color y textura, también existen métodos dentro del procesamiento de bordes encaminados a localizar y seguir directamente las siluetas de determinados elementos de interés en la escena. Se trata de las técnicas de contorno activo, llamadas así porque utilizan un modelo de contorno que se adapta a los cambios de intensidad en la imagen para localizar un objeto de características establecidas. Si el modelo deformable no presupone nada sobre la estructura del objeto buscado, excepto quizá alguna restricción de regularización, se les suele dar el nombre de *snakes* (Kass *et al.*, 1988). Si, por el contrario, se pretende encontrar y seguir objetos de una forma determinada, se llaman *modelos deformables* (Jain *et al.*, 1998). En general, casi todas estas técnicas se apoyan en la minimización de una función de energía que tiene en cuenta tanto los bordes de la imagen como un término interno que tiende a suavizar la forma del contorno. Para los modelos deformables, además, se puede añadir una etapa bayesiana posterior que tenga en cuenta unas ciertas probabilidades de deformación impuestas *a priori* sobre el modelo a seguir (Jain *et al.*, 1996).

Por último, quizá quepan también en este apartado aquellos sistemas de visión que se apoyan el flujo óptico como etapa básica de preprocesamiento. En ellos se explota no sólo la variación espacial de la señal luminosa, sino también su evolución temporal. La información de interés aquí no es tanto la posición exacta en la imagen de los puntos de borde como el movimiento que experimentan cuando hay un desplazamiento relativo entre la cámara y la escena, o bien existe algún objeto móvil dentro de ésta última. Dicho movimiento es computable con una mínima robustez sólo en los puntos donde hay una cierta discontinuidad en la imagen, es decir, donde puede apreciarse algún borde más o menos local. Esto es lo que nos lleva a introducir esta técnicas en este punto de la discusión. Un trabajo clásico donde se describe este tipo de acercamiento es el de Verri y Poggio (1989). Algunos autores proponen incluso modelos inspirados biológicamente para estimar la información de profundidad a

partir del flujo óptico, al estilo de las redes de saliencia discutidas anteriormente para agrupar contornos. Por ejemplo, Worgotter y Cozzi (1999) describen un modelo neuronal con estas características, que usa elementos de proceso distribuidos sobre una retina y comunicados entre sí sólo localmente, de modo que el sistema es lo suficientemente eficiente como para trabajar en tiempo real, aún en una sola CPU. De todos modos, también se defiende la posibilidad de una implementación hardware para tratar imágenes de mayores resoluciones.

Son muchos los proyectos de robots guiados visualmente que emplean la detección de bordes como etapa de procesamiento de bajo nivel. En general, los operadores locales (basados en gradiente, filtros de orientación, o simplemente filtros paso-alto, por ejemplo) son muy atractivos en las aplicaciones en tiempo real, dada su facilidad de cómputo y la expresividad de la información obtenida. Ésta puede utilizarse posteriormente para encontrar primitivas de mayor nivel, tales como segmentos o contornos. Un par de ejemplos de uso en vehículos autónomos pueden ser los propuestos por Frizera *et al.* (2000) y Herman *et al.* (1997), donde simplemente se localizan y siguen los lados de un pasillo y una carretera, respectivamente. El objetivo es mantener una navegación centrada elemental, sin salirse de los caminos respectivos. A veces se aumenta la información de posición con otras pistas, tales como la posición de las puertas, por ejemplo (Lee *et al.*, 2000).

La utilización de modelos de contorno en el seguimiento de objetos en tiempo real es también relativamente usual sobre todo en robots manipuladores. Drummond y Cipolla (2002), por ejemplo, realizan el seguimiento visual de un modelo 3D conocido, como una pieza industrial, tomando como base la reproyección de sus bordes en la imagen (es lo que denominan *wireframe snake*, o contorno alámbrico activo). El sistema es capaz de posicionar y seguir con eficiencia y robustez la pieza mediante una sencilla técnica de búsqueda de la nueva silueta observada a partir de su posición en el *frame* anterior. Los mismos autores, en otro trabajo (Drummond y Cipolla, 1999), emplean también la técnica más tradicional de contornos activos bidimensionales (*snakes*) para controlar un brazo robótico con una cámara en el extremo que sigue piezas sobre las que hay dibujada una determinada señal. Finalmente, Isard y Blake (1998) amplían las técnicas de contorno activo para manejar situaciones con solapamiento, elementos de perturbación, etc. El método es computacionalmente costoso, puesto que mantiene una población grande de muestras como modelo no paramétrico de la densidad de probabilidad sobre el espacio de contornos (es un método de Montecarlo), pero aun así puede funcionar en tiempo real en una sola CPU.

Las técnicas de flujo óptico se utilizan también con cierta asiduidad en el diseño de prototipos de navegación visual, sobre todo en problemas de determinación del movimiento propio a partir de la estructura espacio-temporal de la secuencia de imágenes obtenida, sin hacer uso de ningún otro sensor propioceptivo de movimiento. En esta línea se encuentra el trabajo de Fermüller y Aloimonos (1997). Cierto tipo de robots reactivos utilizan también la divergencia local del flujo óptico para detectar obstáculos cercanos y evitarlos. Tal es el caso del sistema propuesto por Camus *et al.* (1999), que trabaja estimando el flujo sobre imágenes submues-

treadas, para conseguir velocidades de varios *frames* por segundo.

Una de las ventajas de trabajar con bordes es la relativa facilidad con la que a partir de ellos pueden extraerse estructuras más complejas que, siendo aún independientes del dominio concreto, proporcionan una información más descriptiva a las etapas posteriores de la aplicación. Un modelo muy utilizado a este nivel intermedio es el segmento, correspondiente a la agrupación rectilínea que puede formarse entre determinados bordes de la imagen. A este tipo de característica, por su gran popularidad entre la comunidad de visión por computador para agentes autónomos, y por ser la piedra de toque sobre la que propondremos nuestra arquitectura de percepción, dedicaremos la siguiente sección.

2.3. Detección de segmentos

Las escenas naturales son demasiado complejas para ajustarse con precisión a modelos geométricos excesivamente simples, como pueden ser los segmentos. No obstante, el fragmento de línea recta es un tipo de borde muy habitual en objetos fabricados por el hombre. En particular, esto es especialmente cierto en escenas de interiores de edificios y entornos estructurados en general. Puesto que es en este tipo de entornos donde, hoy por hoy, se mueven la mayoría de sistemas autónomos, el uso de este tipo de modelo está muy extendido entre la comunidad de visión para robots. Algunas de las ventajas más destacables del hecho de utilizar esta abstracción en tareas de percepción son las siguientes:

- Desde el punto de vista de la geometría proyectiva las líneas rectas son, junto con los puntos, los elementos más utilizados para imponer restricciones entre las múltiples vistas de una escena. Así, una gran parte de los métodos de calibración, reconstrucción tridimensional, o estimación del movimiento a partir de secuencias de imágenes utilizan este tipo de características como entradas de los diferentes algoritmos.

- Los segmentos pueden representarse de un modo bastante compacto. Una posibilidad es guardar simplemente las coordenadas de sus extremos, por ejemplo. Aunque, por supuesto, existen diversas representaciones alternativas, en general todas ellas requieren sólo unos pocos *bytes* de almacenamiento. Esto supone una ventaja frente a otras estructuras más complejas, como los contornos irregulares, por ejemplo.

- Prácticamente siempre se corresponden con entidades del mundo real. Esto es una ventaja frente a otras características más sencillas, como las esquinas, que a menudo no se corresponden a objetos concretos sino a intersecciones espurias de bordes que dependen del punto de vista desde el que se observa la escena.

- En relación directa con lo anterior, es más fácil detectar relaciones de oclusión, continuidad, o colinealidad entre segmentos que entre puntos aislados. Esto puede resultar de bastante utilidad en posteriores etapas de interpretación en el proceso perceptivo.

- Aunque en general los extremos de los segmentos puedan ser poco fiables tras el proceso de extracción, la determinación con mayor precisión de otro tipo de características, como las esquinas o los contornos, puede llevarse también a cabo a partir de la intersección o el agrupamiento de segmentos con extremos cercanos, por ejemplo.

- Finalmente, utilizando el algoritmo adecuado, ofrecen una buena relación entre la eficiencia de cómputo y la robustez en el proceso de localización.

Tal y como hicimos anteriormente para las distintas técnicas de segmentación, en este apartado revisaremos brevemente los principales métodos de detección de líneas rectas en imágenes. Una vez más, haremos énfasis en aquellos que, por su eficiencia computacional, pueden aplicarse bajo las restricciones de tiempo real exigidas por la navegación visual.

2.3.1. Transformada de Hough y sus variantes

Quizá el método más conocido para detectar bordes rectos en una imagen sea la transformada de Hough (1962) (en adelante HT), y sus distintas variantes (Leavers, 1993). La idea principal de la HT consiste en la transformación de los bordes del espacio de coordenadas de la imagen a otro espacio paramétrico donde se pueda identificar de modo más sencillo el tipo de estructura buscada[4]. Una vez realizada la transformación paramétrica, se buscan agrupaciones significativas de muestras en ese otro espacio, que presumiblemente se corresponderán con líneas localizadas sobre la imagen original.

Una parametrización muy utilizada consiste en caracterizar cada línea mediante su distancia perpendicular al origen de coordenadas, ρ, y el ángulo θ que forma el vector normal a la recta con el eje X. Todos los puntos (x, y) por los que pasa la recta, por tanto, cumplirán la ecuación $x\cos(\theta) + y\sin(\theta) = \rho$. El espacio (ρ, θ) es entonces discretizado, y cada punto de entrada realiza una votación sobre todas aquellas celdas correspondientes a configuraciones paramétricas de rectas compatibles con dicho punto (es decir, que pasen a través de él). Posteriormente se realiza un proceso de búsqueda de máximos en el espacio discretizado de parámetros utilizado en la votación. Estos máximos se corresponderán, teóricamente, con las líneas encontradas en la imagen original. Una parametrización alternativa, también bastante utilizada, es la llamada *foot of normal*. En ella, se caracteriza cada recta mediante el punto sobre el que cae la perpendicular a la misma que pasa por el origen (de ahí la denominación de *pie de la normal*). Esta otra forma tiene la ventaja de que el espacio de parámetros coincide con el de coordenadas de la imagen original (Davies, 1997).

La HT clásica tiene algunos problemas, entre los cuales podemos destacar los siguientes: la multiplicidad de votos asociados a cada punto de entrada, la dificultad de la elección del

[4]En nuestro caso, se trata de localizar líneas rectas, pero el método puede generalizarse para la búsqueda de otro tipo de modelos, como circunferencias, rectángulos, etc. Naturalmente, para cada tipo se elegirá una parametrización acorde con sus características geométricas.

grado de discretización del espacio de parámetros adecuado, la falta de robustez en la determinación de máximos locales en dicho espacio y, por último, la eficiencia computacional. Muchos autores han sugerido numerosas mejoras desde su aparición, tratando de vencer estos inconvenientes. He aquí un breve resumen de las mismas:

HT sobre el gradiente: Un modo de eliminar el voto múltiple consiste en utilizar no sólo la posición del punto de borde, sino también la dirección del gradiente en dicho punto. Así, cada votación se reduce a una sola celda del espacio paramétrico, correspondiente a la orientación detectada, o al menos a un subconjunto pequeño de las mismas, si se introduce un umbral de limitación sobre la precisión en la estimación del gradiente (O'Gorman y Clowes, 1976).

HT jerárquica: En esta variante se divide la imagen original en un mosaico de imágenes más pequeñas, sobre las que se aplica la HT clásica individualmente. Típicamente, como mucho deben aparecer dos líneas en cada elemento del mosaico. El esquema se aplica recursivamente, ascendiendo paulatinamente en la jerarquía mediante la agrupación de las soluciones para subimágenes contiguas. Las líneas de cada división se combinan entre sí usando también la HT. El proceso se repite hasta que se cubre la imagen completa. De este modo se facilita la búsqueda de máximos en el espacio de votación, al tiempo que se disminuye la complejidad algorítmica (Li *et al.*, 1986).

HT probabilística: Un modo de mejorar la eficiencia global es trabajar sólo sobre un subconjunto aleatorio de los puntos de entrada. Normalmente, la solución así obtenida es muy similar a la que se obtendría utilizando el conjunto completo, mientras que la carga computacional se reduce considerablemente (Kiryati *et al.*, 1991).

HT aleatoria: El problema de la búsqueda de máximos en el espacio paramétrico viene motivado, en gran parte, por la correspondencia *uno-a-muchos* (*one-to-many mapping*) que se aplica sobre los puntos de entrada al realizar la votación. Un modo de eliminarlo consiste en seleccionar aleatoriamente pares de puntos de entrada, en lugar de individualmente. Cada par de puntos forman una recta, y por tanto contribuyen con un sólo voto al espacio de parámetros (*one-to-one mapping*). El proceso repite iterativamente, hasta que una celda excede un umbral de votación determinado. En ese momento se guarda la recta encontrada, se eliminan los puntos correspondientes y se continúa con el proceso de extracción aleatoria de pares, hasta que el conjunto de entrada queda vacío (Xu y Oja, 1993).

HT conectiva: Otra forma de intentar el voto único es estimar la orientación local examinando pequeñas ventanas en torno a cada punto, como alternativa al uso de la información de orientación del gradiente. Sólo si se detectan muchos puntos alineados y conectados en esta ventana se añade el voto correspondiente al espacio de parámetros (Kalviainen y Hirvonen, 1995).

Sobre la base de estas ideas otros autores sugieren diversos refinamientos adicionales, encaminados a mejorar el rendimiento y la precisión de las distintas versiones de la transformada. Yang *et al.* (1997), por ejemplo, proponen un método híbrido (con elementos de las variantes conectiva, aleatorizada y jerárquica), que tiene también en cuenta el posible grosor variable de las líneas. Otro ejemplo es el descrito por McLaughlin y Alder (1998), que, como alternativa a la HT conectiva, proponen un método conocido con el nombre de *UpWrite*. Aquí también se computan modelos de alineamiento locales a cada píxel, pero en este caso caracterizándolos a través de la dirección del vector principal de la matriz de covarianza de la nube de puntos de su entorno. Estos modelos locales se pueden ir agrupando después si se encuentran alineados hasta un determinado grado de curvatura.

2.3.2. Métodos agregacionales y aproximaciones poligonales

En relación directa con la última aproximación, una alternativa a las transformadas del apartado anterior la constituyen los modelos agregacionales. Estos métodos se basan en el uso de algún procedimiento de agrupación de bordes elementales en contornos extendidos, que después son examinados para detectar segmentos sobre ellos. El propio operador de Canny, anteriormente descrito, puede utilizarse para realizar las agrupaciones. Naturalmente, estos contornos se pueden obtener también tras una segmentación por regiones de la imagen, con simples algoritmos de recorrido de los bordes de las zonas encontradas. Sea cual sea el método de extracción, una vez que se dispone de las secuencias ordenadas de puntos de borde se pueden analizar con métodos de aproximación poligonal para reducirlas a cadenas de segmentos individuales. Una buena revisión de este tipo de técnicas es la realizada por Rosin (1997b). Muchos de los procedimientos propuestos son muy eficientes computacionalmente, y por tanto implementables en sistemas con restricciones sobre el tiempo de ejecución (Pikaz y Dinstein, 1995; Yu y Yan, 1997). Un sistema completo de extracción de bordes, agrupación, y aproximación poligonal en tiempo real para robots autónomos es el descrito por Aste y Boninsegna (1993), por ejemplo.

2.4. Extracción eficiente de segmentos con información de color

Al igual que los métodos anteriores, las etapas de bajo y medio nivel de nuestra arquitectura de percepción estarán orientadas a la extracción de segmentos. Como característica distintiva de la propuesta aquí desarrollada, sin embargo, aumentaremos la natural expresividad geométrica de cada segmento con información sobre el color de la imagen a ambos lados del mismo. De este modo, a pesar del proceso de reducción de información inherente a todo proceso de extracción de características, se captura el grueso de la información fotométrica de la imagen, con las consecuentes ventajas añadidas derivadas de su aprovechamiento en etapas posteriores de la arquitectura.

La información estrictamente posicional de los segmentos se puede utilizar en la solución de problemas de estimación de índole geométrica, como la calibración del sensor, la reconstrucción tridimensional y la estimación del movimiento, por ejemplo. La de color, por otro lado, resultará de gran ayuda en la implementación de tales técnicas, puesto que tareas como el seguimiento de estructuras, la localización de contornos o incluso algunas relacionadas con la interpretación se verán muy simplificadas al disponerse de esta información adicional.

En los apartados que siguen describiremos el procedimiento completo de extracción de características propuesto. Cada etapa ha sido optimizada tomando en consideración las restricciones temporales bajo las que operará el agente autónomo, manteniendo siempre un compromiso aceptable entre la velocidad de procesamiento y la precisión en la localización de los segmentos. Este equilibrio, como veremos, jugará un papel fundamental en el sistema de navegación visual implementado. Un resumen del método presentado se puede consultar también en (López de Teruel *et al.*, 2003).

2.4.1. Preprocesamiento

El paso previo a la obtención del conjunto de segmentos consiste en la extracción de los bordes presentes en la imagen. Como ya se ha comentado en este mismo capítulo, el extractor de bordes más conocido es el operador de Canny, por su eficiencia, precisión y adaptabilidad a cambios locales de luminosidad, gracias a su proceso de histéresis. Las imágenes binarizadas obtenidas con esta técnica suelen tener bastante calidad, ya que respetan una cierta conectividad entre píxeles, conservan la unicidad de los bordes y filtran de modo eficiente los bordes falsos procedentes del ruido.

Nosotros, sin embargo, optaremos por una solución más simplificada, destinada a estimar únicamente la intensidad de borde aproximada en cada punto de la imagen. Al contrario que en el operador anterior, por ahora no nos ocuparemos de calcular de la orientación local, eliminar el ruido o conservar la conectividad. Todos estos requerimientos se cubrirán en la posterior etapa de agrupamiento, donde la cantidad de información a tratar quedará sensiblemente disminuida y, por tanto, se consumirán menos recursos de procesamiento. De esta forma podremos ahorrarnos las operaciones previas de filtrado gaussiano, estimación del gradiente, búsqueda de máximos locales e histéresis que, como vimos, resultaban necesarias para aplicar el detector de Canny. Alternativamente, computaremos una imagen de bordes de partida utilizando un simple filtro paso-alto de la imagen original, previamente convertida de color a niveles de gris ponderando adecuadamente los canales de rojo (R), verde (G) y azul (B) (Gonzalez y Woods, 1993). El filtro paso-alto puede ser implementado con una simple máscara $h_{3\times 3}$ como la mostrada en la figura 2.1.

En resumen, pues, nuestra etapa de preprocesamiento se reduce a estas operaciones, aplicadas sobre cada una de las posiciones (x, y) de la matriz de píxeles:

	−1	0	+1
−1	−1	−1	−1
0	−1	+8	−1
+1	−1	−1	−1

Figura 2.1: *Máscara $h_{3\times 3}$ utilizada en el filtro paso-alto.*

$$I^{gray} = 0.299\, I^R + 0.587\, I^G + 0.114\, I^B \tag{2.1}$$

$$I^{brd} = |h * I^{gray}| \tag{2.2}$$

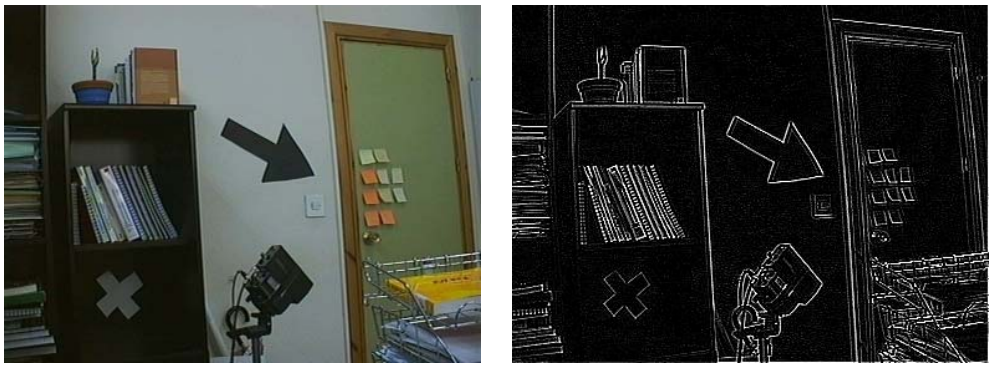
El símbolo $*$ representa el operador de convolución[5]. Si observamos los valores de la máscara h, comprobaremos que es similar a la empleada en el filtrado de la laplaciana de la gaussiana del operador del Marr-Hildreth, pero tomando en este caso el valor absoluto de la salida, en lugar de la más costosa búsqueda de los cruces por cero. Al utilizar un tamaño de máscara pequeño, la operación asociada es extremadamente eficiente desde el punto de vista computacional, puesto que pueden aprovecharse bastantes sumas parciales entre píxeles vecinos, y se requiere una sola pasada por la imagen. Lo mismo puede decirse de la obtención de la imagen de grises a partir de la de color RGB. La mayoría de bibliotecas de procesamiento de imagen contienen implementaciones eficientes de estas operaciones. En concreto, nosotros utilizamos las IPL (*Image Processing Libraries*) de Intel, las cuales explotan al máximo los recursos de los distintos procesadores de la familia IA-32, maximizando el paralelismo a nivel de instrucción en el uso de las unidades funcionales y el uso eficiente de la memoria caché[6].

En ocasiones puede resultar adecuado realizar un procesado previo de la imagen original con algún supresor de ruido. En concreto, nosotros utilizamos un filtrado de mediana, que, al contrario que otros filtros lineales (como el gaussiano, por ejemplo) tiene la interesante propiedad de no difuminar los bordes[7]. Esta operación está además implementada con gran eficiencia en la biblioteca IPL. En realidad este paso previo no es demasiado necesario ya que, como se verá, la posterior estimación de la orientación local de los bordes tiene un efecto regularizador colateral que la hace resistente al posible ruido. Aún así, su empleo puede resultar conveniente en determinados entornos en los que, como es nuestro caso, las paredes o el suelo pueden aparecer en la imagen con una cierta textura de grano relativamente apreciable.

[5]Este operador calcula cada punto de la imagen de bordes usando la expresión $I^{brd}_{x,y} = \left| \sum_{i=-1}^{i=1} \sum_{j=-1}^{j=1} h_{i,j}\, I^{gray}_{x+i,y+j} \right|$

[6]Las IPL están actualmente incluidas dentro de las más generales IPP (*Integrated Performance Primitives*), también de Intel. En el capítulo 5, donde trataremos la arquitectura hardware-software del sistema, comentaremos más en detalle las características de estas bibliotecas.

[7]Este filtrado funciona sustituyendo cada píxel de la imagen original con la mediana de los valores de una ventana local de tamaño dado $m \times m$. Típicamente, $m = 3$ o $m = 5$, por ejemplo.



(a) (b)

Figura 2.2: *Resultado del filtro paso-alto sobre una imagen de ejemplo: (a) Imagen original. (b) Imagen obtenida (un valor de gris más claro significa una mayor respuesta al borde).*

La imagen de bordes obtenida tras este preprocesamiento, aunque indudablemente peor que la de Canny desde el punto de vista cualitativo (no está conectada, está aún sin binarizar, y tiene los bordes menos localizados), será sin embargo suficiente para aplicar la siguiente etapa, en la que se agrupan los bordes utilizando la información de alineamiento local para extraer los segmentos deseados. La figura 2.2 muestra el resultado para una imagen de una escena interior de ejemplo. Nótese que se pierde toda la información direccional del gradiente, ya que sólo se hace una estimación aproximada de su magnitud. Como se puede observar, el filtro utilizado tiene también cierta tendencia a generar numerosos puntos de borde falsos, en forma de pequeñas "motas" en la imagen obtenida. Este efecto se atenúa notablemente si se ha utilizado el filtro de la mediana sobre la imagen de gris. En el siguiente apartado comprobaremos cómo, en cualquier caso, ninguna de estas cuestiones supone un problema grave para la etapa de procesamiento posterior.

2.4.2. Estimación de la orientación local y agrupamiento

Máscaras de saliencia

Una vez estimada la magnitud del gradiente, el siguiente paso consiste en la búsqueda de tramos de borde rectos, con el fin de obtener los segmentos. De nuevo es importante que el método realice esta búsqueda del modo más eficiente posible. Para ello, utilizaremos un conjunto de máscaras especialmente diseñado para detectar estructuras de alineación locales en torno a los bordes activos (López de Teruel y Ruiz, 1999a; López de Teruel y Ruiz, 1999b). Este conjunto debe computarse sólo durante la inicialización del sistema, para después ser utilizado durante todo el periodo de operación. Las denominaremos *máscaras de saliencia* puesto que, como se verá, tienen una respuesta mayor en los contornos más *salientes*, en un sentido similar al trabajo de Sashua y Ullman anteriormente comentado. En lo que sigue describiremos el procedimiento de generación y las ideas básicas subyacentes en la operación de tales máscaras.

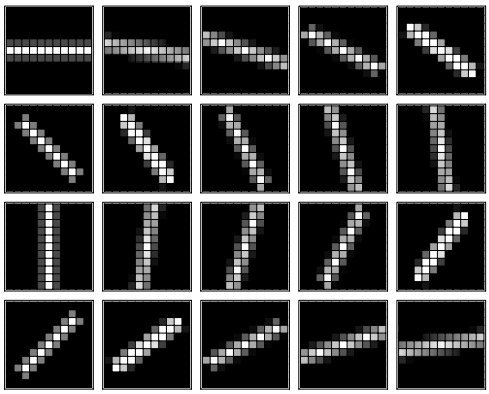


Figura 2.3: *Conjunto de máscaras de tamaño* 11×11*, correspondientes a* $A = 20$ *orientaciones discretas, radio* $R = 5$ *y anchura de borde de* $W_{edg} = 2.80$*.*

La figura 2.3 muestra un conjunto completo de máscaras de ejemplo. Cada conjunto es construido utilizando tres parámetros, que sirven para adaptar su funcionamiento a las distintas características de las imágenes de entrada. Dichos parámetros son los siguientes:

1. El radio en píxeles de cada máscara, R: Este parámetro mide el grado de *localidad* del estimador de orientación. A mayor radio, mayor es el área de imagen en el entorno del píxel actual que será utilizada para estimar la orientación del gradiente. Un radio más grande tendrá en general un beneficioso efecto regularizador, si bien puede presentar dificultades en la localización de segmentos cortos, al tiempo que aumentará el tiempo necesario para computar la respuesta de la máscara. Por contra, un radio excesivamente corto tenderá a resultar más impreciso en la estimación de la orientación, con el consiguiente peligro de fraccionamiento en los segmentos obtenidos. Valores razonables para R podrían ser 3, 4 ó 5 píxeles. Las máscaras de la figura 2.3, por ejemplo, fueron generadas con $R = 5$, correspondiente a la distancia máxima en X o Y desde el centro de la máscara hasta cualquier píxel en el borde de la misma, por lo que tienen un tamaño total de $(2R + 1) \times (2R + 1) = 11 \times 11$.

2. El número de orientaciones discretas que se considerarán, A: Dichas orientaciones quedarán distribuidas uniformemente en el rango $[0, 2\pi)$ radianes, a intervalos regulares de $2\pi/A$. En este caso, el valor más adecuado dependerá del radio de la máscara. Resulta razonable una elección de aproximadamente $A \approx 4R$. El conjunto de ejemplo fue generado utilizando $A = 20$ orientaciones distintas, y por tanto contiene ese mismo número de máscaras.

3. La anchura esperada de cada borde (o *edgel*), W_{edg}: Este valor marca la cantidad de píxeles activos que cabe esperar para un borde típico en la imagen paso-alto, en la dirección marcada por el gradiente (esto es, perpendicular al borde), y en unidades de píxel. El valor adecuado dependerá de la resolución de la imagen de entrada. Para imágenes como

	−5	−4	−3	−2	−1	0	+1	+2	+3	+4	+5
−5											
−4											
−3		2.2	0.1								
−2	4.4	6.6	4.4	2.2	0.1						
−1	0.1	2.2	4.4	6.6	4.4	2.2	0.1				
0			0.1	2.2	4.4	6.6	4.4	2.2	0.1		
+1					0.1	2.2	4.4	6.6	4.4	2.2	0.1
+2							0.1	2.2	4.4	6.6	4.4
+3									0.1	2.2	
+4											
+5											

Figura 2.4: *Valores concretos de la máscara de saliencia correspondientes a la cuarta orientación ($a = 3$, ángulo $3\pi/20$radianes $= 27^o$ respecto a la horizontal). Las posiciones i y j recorren los índices $-R, \ldots, R$, y los correspondientes valores $L^a_{i,j}$ aparecen multiplicados por* 10^2.

las que utilizaremos (mitad del tamaño PAL estándar, 288×384), un valor razonable puede estar en el rango $[1.5, 3.0]$. Para las máscaras concretas de la figura, por ejemplo, se utilizó $W_{edg} = 2.8$.

En la figura 2.4 puede apreciarse el valor numérico de los elementos para cada posición de la cuarta máscara de la fila superior de la figura 2.3, como ejemplo. Obsérvese que, en aquella figura, las celdas correspondientes a cantidades mayores se mostraban en gris más claro. Estos valores están normalizados de forma que sumen la unidad, aunque en la figura aparecen en forma de porcentaje. El procedimiento de construcción reparte el peso de los mismos de modo decreciente conforme nos alejamos del segmento ideal que pasa por el centro de la máscara en la orientación elegida, dependiendo de la anchura W_{edg}. Si llamamos $d^a_{i,j}$ a la distancia perpendicular de la posición (i, j) a dicho segmento ideal en la orientación a, la expresión empleada para el cómputo de los pesos (a los que denominaremos $L^a_{i,j}$, siguiendo el mismo convenio de índices) es:

$$L^a_{i,j} = \begin{cases} \dfrac{W_{edg}/2 - |d^a_{i,j}|}{\displaystyle\sum_{(l,m)\,|\,(d^a_{l,m} \le W_{edg}/2) \wedge (\sqrt{l^2+m^2} \le R)} (W_{edg}/2 - |d^a_{l,m}|)} & \text{si} \quad (d^a_{i,j} \le W_{edg}/2) \wedge (\sqrt{i^2+j^2} \le R). \\ 0 & \text{en caso contrario.} \end{cases} \tag{2.3}$$

Como se observa, los píxeles de la máscara cuyo valor es distinto de cero son sólo aquellos que distan menos de $W_{edg}/2$ del segmento que marca la orientación, y menos de R del centro. Esto último es necesario para hacer las máscaras isotrópicas, es decir, que no se favorezca la respuesta en ninguna orientación respecto a las demás al tener todas el mismo número de píxeles activos. El número de estos píxeles no nulos es relativamente pequeño con respecto al tamaño total de la máscara (35 de los 121 en nuestro ejemplo), e incluso puede reducirse si se

usan valores menores de W_{edg} para aumentar la eficiencia del procedimiento.

En general, valores más pequeños de R, A y W_{edg} generarán un conjunto de máscaras más eficiente, aunque también más impreciso. La elección correcta será aquella que obtenga un buen compromiso entre la velocidad de operación y la precisión alcanzada, teniendo en cuenta siempre la capacidad de procesamiento del hardware disponible.

Procedimiento de agrupamiento

Como ya se ha comentado, el cómputo de las máscaras se realiza sólo durante la puesta en marcha del sistema. Entonces se almacenan en memoria para ser utilizadas durante todo el periodo de operación del robot. Con las máscaras ya disponibles, el procedimiento de estimación de orientación y posterior agrupamiento funcionará del siguiente modo:

1. La imagen de bordes se recorre de izquierda a derecha, y de arriba a abajo, a la búsqueda de algún píxel de borde *activo*, esto es, cuyo valor $I^{brd}_{x,y}$ esté por encima de un determinado umbral τ_h. Una vez localizado éste, se aplica el conjunto de máscaras centradas sobre él, para obtener un conjunto de A medidas de respuesta a cada orientación. De todas ellas nos quedaremos solamente con la mayor, recordando el índice a^{max} correspondiente al ángulo en el que se alcanzó dicho máximo. Es decir:

$$a^{max} = \underset{a\in[0,A-1]}{argmax} \sum_{i=-R}^{i=R} \sum_{j=-R}^{j=R} L^{a}_{i,j} I^{brd}_{x+i,y+j} \tag{2.4}$$

   Obsérvese que, aunque el cómputo de cada respuesta individual es lineal, el posterior cálculo del máximo no lo es. Se consigue de esta forma el efecto regularizador del que hablábamos antes. A pesar de la posible existencia de puntos de borde aislados, incluso en las inmediaciones de contornos de objetos reales, esta no linealidad amortiguará el posible ruido en la estimación de la orientación. Esto supone una ventaja frente a otros métodos más sensibles a *outliers* en el entorno del píxel examinado, como el empleado en el método *UpWrite* mencionado en la revisión de la sección anterior.

2. Utilizando el valor obtenido se comienza un proceso de agrupamiento en la dirección correspondiente. Para ello, se utilizan unas tablas de acceso a píxeles adicionales, que pueden ser creadas al mismo tiempo que las máscaras de saliencia. Estas tablas marcan el camino a seguir en el agrupamiento, en la dirección indicada por a^{max}, a partir del píxel actual y en ambos sentidos. La figura 2.5 muestra el contenido concreto de una de ellas, correspondiente a la misma orientación que el ejemplo de la figura anterior. Los números enteros indican las posiciones que el procedimiento recorrerá, en relación al píxel central de la máscara, y con signos opuestos para sentidos opuestos del recorrido. La magnitud del número indica el orden de procesamiento. Así, si el píxel actualmente

	−5	−4	−3	−2	−1	0	+1	+2	+3	+4	+5
−5											
−4											
−3		−16	−13								
−2	−17	−14	−11	−9	−6						
−1	−15	−12	−10	−7	−4	−2	1				
0			−8	−5	−3	0	3	5	8		
+1					−1	2	4	7	10	12	15
+2							6	9	11	14	17
+3									13	16	
+4											
+5											

Figura 2.5: *Tabla de recorrido local de píxeles, en ambas orientaciones. Un valor $k > 0$ en la posición (i, j) significa que el píxel (i, j) se recorre en el $k-$ésimo lugar en el sentido positivo (hacia abajo y a la derecha). Si $k < 0$, entonces el píxel se recorre en la $|k|-$ésima posición, y en el sentido contrario. Las celdas no marcadas resultan en píxeles no visitados durante el recorrido.*

considerado es el (x, y), marcado con el 0 en la figura, la secuencia de posiciones recorridas será $(x + 1, y - 1)$, $(x, y + 1)$, $(x + 1, y)$, ... en sentido positivo, y $(x - 1, y + 1)$, $(x, y - 1)$, $(x - 1, y)$, ... en el contrario.

3. Mediante un proceso iterativo en ambas direcciones, se van capturando aquellos píxeles que tengan un valor de respuesta a borde mayor que otro umbral mínimo, τ_l. De este modo se acumula un conjunto de píxeles alineados que va creciendo por ambos extremos, y que se detiene en el momento en que no se puede seguir por falta de píxeles de borde activos en la dirección adecuada.

Para que el procedimiento sea mínimamente robusto hay que tener en cuenta que, debido a la discretización en píxeles de la imagen, es posible que la orientación obtenida inicialmente experimente pequeñas variaciones al movernos por el segmento. Por ello, es conveniente reestimar la dirección de máxima respuesta periódicamente, y continuar el procedimiento sólo si nos encontramos en unos márgenes de tolerancia determinados. Una posibilidad es hacerlo al llegar al borde de la tabla de recorrido (es decir, las posiciones 17 y -17 de la tabla de la figura 2.5). Si al reestimar la orientación la dirección de máxima respuesta ésta está aproximadamente alineada con la original (por ejemplo, en un rango de ± 1 orientaciones discretas) podemos continuar con el proceso de agrupamiento. En caso contrario, se termina el procedimiento por ese extremo.

Obsérvese que la ventaja del uso de las tablas de recorrido es de nuevo la aceleración del proceso. El número de operaciones necesarias se limita a la obtención de unos datos precomputados durante la puesta en marcha del sistema, en lugar de tener que recalcular constantemente la posición del siguiente píxel a examinar.

El proceso anterior finaliza cuando el grupo de píxeles de borde alineados no puede crecer por ninguno de sus extremos. En este punto, los últimos píxeles añadidos en cada sentido

podrían ejercer el papel de extremos del segmento obtenido. Sin embargo, podemos alcanzar precisión subpíxel con un coste adicional muy pequeño, proyectando dichos extremos sobre la dirección principal del conjunto de puntos capturado. Para ello, calculamos el centroide y la matriz de covarianza de dicho conjunto (al que llamaremos Δ^s, para un segmento s determinado):

$$\overline{\boldsymbol{x}}^s = (\overline{x}^s, \overline{y}^s) = \sum_{(x,y)\in\Delta^s} \frac{(x,y)}{|\Delta^s|} \tag{2.5}$$

$$\Sigma_s = \begin{pmatrix} \sigma^s_{xx} & \sigma^s_{xy} \\ \sigma^s_{xy} & \sigma^s_{yy} \end{pmatrix} = \sum_{(x,y)\in\Delta^s} \frac{((x,y)-(\overline{x},\overline{y}))^T \cdot ((x,y)-(\overline{x},\overline{y}))}{|\Delta^s|} \tag{2.6}$$

Estas operaciones pueden realizarse durante el mismo bucle de agrupamiento, mediante la acumulación de los valores parciales de x, y, x^2, y^2 y xy de los puntos capturados. Utilizando esta información, es sencillo proyectar perpendicularmente los extremos sobre la línea que pasa por el centroide y cuya dirección viene dada por el vector principal de mayor valor propio de la matriz de covarianza. Este vector tiene la siguiente forma:

$$\boldsymbol{v}^s = (v^s_y, v^s_x) = (1, \frac{\sigma^s_{xx} - \sigma^s_{yy} + \sqrt{(\sigma^s_{xx}+\sigma^s_{yy})^2 - 4(\sigma^s_{xx}\sigma^s_{yy} - {\sigma^s_{xy}}^2)}}{2\sigma^s_{xy}}) \tag{2.7}$$

Normalizando la expresión anterior para conseguir un vector unitario ($\overline{\boldsymbol{v}}^s = \boldsymbol{v}^s/\|\boldsymbol{v}^s\|$), y si llamamos $\boldsymbol{x}^s_+ = (x^s_+, y^s_+)$ y $\boldsymbol{x}^s_- = (x^s_-, y^s_-)$ a los extremos del segmento obtenidos por el procedimiento de captura, podemos calcular las proyecciones deseadas, $\boldsymbol{x}^s_1 = (x^s_1, y^s_1)$ y $\boldsymbol{x}^s_2 = (x^s_2, y^s_2)$, utilizando las siguientes expresiones:

$$\boldsymbol{x}^s_1 = \overline{\boldsymbol{x}}^s + ((\boldsymbol{x}^s_+ - \overline{\boldsymbol{x}}^s)\cdot\overline{\boldsymbol{v}})\overline{\boldsymbol{v}} \tag{2.8}$$

$$\boldsymbol{x}^s_2 = \overline{\boldsymbol{x}}^s + ((\boldsymbol{x}^s_- - \overline{\boldsymbol{x}}^s)\cdot\overline{\boldsymbol{v}})\overline{\boldsymbol{v}} \tag{2.9}$$

2.4.3. Postprocesamiento y muestreado de color

Una vez obtenidos los extremos realizaremos una etapa de postprocesamiento del segmento. Ésta consiste en el recorrido de la zona de imagen en las proximidades de la característica extraída, con un doble objetivo:

1. En primer lugar, marcar los píxeles de borde correspondientes como visitados, para evitar que vuelvan a ser procesados al buscar nuevos segmentos. Es importante realizar el marcado en este momento, en lugar de durante el propio proceso de captura, como podría parecer más intuitivo, ya que es muy posible que algunos puntos pertenecientes al modelo actual fuesen pasados de largo durante el agrupamiento (por ejemplo, por

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0														
1				0,0	0,1									
2			2,0	1,0	1,1	0,2	0,3							
3			**3,0**	2,1	2,2	1,2	1,3	0,4	0,5					
4		5,0	4,0	**3,1**	**3,2**	2,3	2,4	1,4	1,5	0,6	0,7			
5		6,0	5,1	4,1	4,2	**3,3**	**3,4**	2,5	2,6	1,6	1,7	0,8		
6			6,1	5,2	5,3	4,3	4,4	**3,5**	**3,6**	2,7	2,8	1,8	0,9	
7				6,2	6,3	5,4	5,5	4,5	4,6	**3,7**	**3,8**	2,9	1,9	
8						6,4	6,5	5,6	5,7	4,7	4,8	**3,9**		
9								6,6	6,7	5,8	5,9	4,9		
10										6,8	6,9			
11														

Figura 2.6: *Tabla de recorrido de píxeles para el entorno del segmento* $((x_1^s, y_1^s), (x_2^s, y_2^s)) = ((2,3),(11,8))$ *en una pequeña imagen de tamaño* 12×14. *Las dimensiones de la tabla resultante para la anchura elegida* $W_{seg} = 5.0$ *son* $r = 7$ *y* $c = 10$. *Un par* (i, j) *en la posición* (x, y) *significa que la posición* (i, j) *de la tabla apunta al píxel con coordenadas de imagen* (x, y). *Las celdas correspondientes al segmento principal se muestran en caracteres más oscuros.*

caer ligeramente alejados de la línea central).

2. En segundo lugar, etiquetar el segmento con información sobre los colores de las zonas de la imagen que hay a cada uno de sus lados, mediante el muestreo directo de los píxeles involucrados.

Para llevar a cabo estas tareas utilizaremos una nueva tabla, que en este caso contiene las posiciones de los píxeles que se encuentran en los alrededores del segmento considerado, recorriéndolo a lo largo para una anchura determinada. La figura 2.6 muestra un ejemplo concreto, para un segmento con extremos $((x_1^s, y_1^s), (x_2^s, y_2^s)) = ((2,3),(11,8))$, y una anchura de $W_{seg} = 5.0$ unidades de píxel. Los contenidos de dicha tabla se muestran en forma inversa, de modo que un par (i, j) en la posición (x, y) significa que el elemento en la fila i y columna j de la tabla apunta a la posición (x, y) de la imagen. Obsérvese que, por construcción, la tabla permite el recorrido ordenado de los píxeles del entorno del segmento, distinguiendo en todo momento en qué lado del segmento estamos según la fila en la que nos encontremos. Así, las posiciones en la mitad superior apuntarán a los píxeles del lado izquierdo, mientras que las de la mitad inferior harán lo propio sobre el lado derecho. Las celdas correspondientes al segmento principal, justo en la mitad, aparecen con índices más oscuros en la figura.

Debido a que las posiciones de los extremos son variables e impredecibles en una secuencia de imágenes arbitraria, está claro que estas otras tablas de recorrido no pueden ser precomputadas, sino que tienen que calcularse cada vez que se extrae un nuevo segmento. Es por tanto obvio que su procedimiento de construcción debe ser lo más rápido posible, ya que habrá que construir varias de ellas de cada *frame*. Para ello nos basaremos en el algoritmo

de Bressenham, un conocido procedimiento del ámbito de la computación gráfica empleado en la generación de líneas rectas sobre retículas de píxeles (Foley *et al.*, 1990). Los detalles de cómputo de la adaptación empleada quedan recogidos en el algoritmo 2.1.

En síntesis, el algoritmo realiza un pequeño número de operaciones enteras para generar la secuencia de posiciones que componen el "nervio central" del segmento (mostrado en negrita en el ejemplo de la figura 2.6). Más concretamente, el orden de ejecución es $O(max\{x_2 - x_1, y_2 - y_1\})$, es decir, lineal respecto a la máxima distancia horizontal o vertical entre los extremos. Posteriormente, mediante un sencillo procedimiento iterativo de réplica del recorrido a ambos lados simétricamente, se generan el resto de posiciones. En total, el número de operaciones necesarias es $O(r \times c)$, siendo r y c el número de filas y columnas de la tabla resultante. El valor $r \times c$ es equivalente al área en píxeles cubierta por el recorrido, esto es, la longitud del segmento por la anchura deseada. Una expresión alternativa para el orden de ejecución es pues $O(W_{seg} \times L)$. Para una imagen típica, el área total de píxeles cubiertos por las tablas de los segmentos obtenidos es en general bastante pequeña, por lo que el coste computacional será incluso menor que el de los filtros empleados en etapas anteriores. En los experimentos con imágenes reales, en efecto, el cálculo de estas tablas en ningún caso supera el 20 % del tiempo de procesamiento del procedimiento de extracción completo.

Estas tablas tienen dos ventajas de cara a la limpieza de los bordes y el muestreado del color. Por un lado, cubren el entorno de cada segmento de modo *denso*, esto es, sin dejar huecos de píxeles en su recorrido. Por otro, generan una *ordenación* de dicho entorno, de modo que es sencillo moverse por el mismo utilizando los índices resultantes. Así, es fácil muestrear el color correspondiente al lado izquierdo del segmento recorriendo todos aquellos píxeles de la imagen RGB original apuntados por las filas 0 a $\lfloor \frac{r}{2} \rfloor - 1$ de la tabla. Del mismo modo, podremos muestrear el lado derecho usando las filas $\lfloor \frac{r}{2} \rfloor + 1$ a $r - 1$. Si, por el contrario, queremos recorrer la banda central para marcar posibles píxeles cercanos al segmento encontrado que no fueron captados en el agrupamiento (y evitar así la aparición de características "parásitas" contiguas a la original), utilizaremos las filas centrales.

La extracción de color se realiza entonces calculando los valores medianos de las componentes roja (R), verde (G) y azul (B) de los píxeles situados a cada lado de la tabla. Dichos valores marginales (puesto que se obtienen para cada canal por separado) constituyen un sencillo estimador robusto de los colores de las dos regiones teóricamente separadas por el segmento tratado. La mediana resulta más robusta que la simple media, ya que en ésta última incluso el muestreo de unos pocos valores incorrectos (*outliers*) puede alterar bastante el vector de color obtenido. El matiz tiene su importancia, puesto que la existencia de dichos valores puede ser habitual en el procedimiento descrito. Esto último será especialmente probable en los extremos laterales de las tablas de acceso, donde podrían encontrarse píxeles pertenecientes a zonas distintas a las correspondientes al segmento actual.

A pesar de la sencillez del procedimiento, los *valores tipo* obtenidos resumen bien la información de color necesaria para prácticamente cualquier condición de la imagen de entrada.

ENTRADA:
- Extremos del segmento (x_1, y_1) y (x_2, y_2), y anchura deseada W_{seg}.

SALIDA:
- Tabla de posiciones en la imagen $(t^x_{i,j}, t^y_{i,j})$, de tamaño $r \times c$.

ALGORITMO:

Inicialización:

/* *Redondeo de posiciones, e incrementos en X e Y:* */
$(x_1, y_1) := ([x_1], [y_1])$. $(x_2, y_2) := ([x_2], [y_2])$.
$(\Delta x, \Delta y) := (x_2 - x_1, y_2 - y_1)$.
/* *Dimensiones de la tabla centrada en el segmento principal y simétrica tal que el área cubierta sea* * * *la mínima posible, pero mayor o igual que* $W\sqrt{\Delta x^2 + \Delta y^2}$ *(para cubrir el área deseada):* */
$(r, c) := (2\lceil \frac{W_{seg}\sqrt{\Delta x^2+\Delta y^2}}{2(\Delta x+1)} \rceil + 1, \Delta x + 1)$.

Segmento principal: /* *Algoritmo de Bressenham:* */
$e := 2\Delta y - \Delta x$. /* *Inicialización del error cometido.* */
$(x, y) := (x_1, y_1)$. /* *Primer píxel del segmento principal.* */
for $j := 0$ ***to*** $c - 2$ ***do***
  $(t^x_{\lfloor \frac{r}{2} \rfloor, j}, t^y_{\lfloor \frac{r}{2} \rfloor, j}) := (x, y)$. /* *Añadimos el píxel al segmento principal.* */
  if $e \geq 0$ ***then***
    $y := y + 1$. $e := e - 2\Delta x$. /* *Avance en la dirección no dominante.* */
  endif
  $x := x + 1$. $e := e + 2\Delta y$. /* *Avance en la dirección dominante.* */
endfor
$(t^x_{\lfloor \frac{r}{2} \rfloor, c-1}, t^y_{\lfloor \frac{r}{2} \rfloor, c-1}) := (x_2, y_2)$. /* *Ultimo píxel del segmento principal.* */

Entorno del segmento: /* *Réplicas sucesivas del segmento:* */
for $i := \lfloor \frac{r}{2} \rfloor + 1$ ***to*** $r - 1$ ***do***
  $\lambda_1 := (x_2 - x_1, y_2 - y_1) \cdot (t^x_{i-1,0} - x_1, t^y_{i-1,0} - y_1 + 1)$.
  $\lambda_2 := (x_2 - x_1, y_2 - y_1) \cdot (t^x_{i-1,c-2} - x_2, t^y_{i-1,c-2} - y_2 + 1)$.
  if $|\lambda_1| \leq |\lambda_2|$ ***then*** /* *Réplica del segmento completo:* */
    for $j := 0$ ***to*** $c - 1$ ***do*** /* *Derecha e izquierda:* */
      $(t^x_{i,j}, t^y_{i,j}) := (t^x_{i-1,j}, t^y_{i-1,j} + 1)$. $(t^x_{r-i-1,j}, t^y_{r-i-1,j}) := (t^x_{r-i,j}, t^y_{r-i,j} - 1)$.
    endfor
  endif else /* *Réplica parcial del segmento, con punto adicional:* */
    for $j := 1$ ***to*** $c - 1$ ***do*** /* *Réplica parcial derecha:* */
      $(t^x_{i,j}, t^y_{i,j}) := (t^x_{i-1,j-1}, t^y_{i-1,j-1} + 1)$.
    endfor
    $(t^x_{i,0}, t^y_{i,0}) := (t^x_{i,c-1} - x_2 + x_1, t^y_{i,c-1} - y_2 + y_1)$. /* *Punto adicional.* */
    for $j := 0$ ***to*** $c - 2$ ***do*** /* *Réplica parcial izquierda:* */
      $(t^x_{r-i-1,j}, t^y_{r-i-1,j}) := (t^x_{r-i,j+1}, t^y_{r-i,j+1} - 1)$.
    endfor
    /* *Punto adicional:* */
    $(t^x_{r-i-1,c-1}, t^y_{r-i-1,c-1}) := (t^x_{r-i-1,0} + x_2 - x_1, t^y_{r-i-1,0} + y_2 - y_1)$.
  endelse
endfor

Algoritmo 2.1: *Generación de una tabla para recorrer el entorno de un segmento. Por simplicidad en la descripción, el algoritmo trabaja sólo para segmentos en la dirección del primer octante (la X domina a la Y, esto es,* $\Delta x > 0$, $\Delta y > 0$ *y* $\Delta x > \Delta y$*). A partir del procedimiento descrito, las extensiones para funcionar con el resto de orientaciones resultarían inmediatas.*

Aunque la estimación de las medianas se hace de modo independiente para los tres canales, en la práctica es muy raro que esto introduzca artefactos en el color extraído. La razón estriba en la relativa regularidad de las nubes de puntos muestreados en el espacio RGB, motivada a su vez por la contigüidad espacial de los píxeles estudiados. De este modo se evita el uso de otro tipo de operador multivariable que, sin duda, supondría una sobrecarga computacional considerable. En la sección 2.5.1 comprobaremos la calidad del estimador mediante la recuperación de la imagen original a partir de los segmentos y etiquetas de color obtenidas con el procedimiento descrito. Se mostrarán entonces algunos ejemplos donde podrá apreciarse que la imagen reconstruida conserva la estructura cromática básica de la imagen original.

2.4.4. Algoritmo

A modo de resumen, el algoritmo 2.2 muestra el procedimiento completo de extracción de segmentos. Básicamente, éste se limita a recorrer la imagen buscando valores superiores a τ_h en la imagen paso-alto, para estimar posteriormente la orientación local del borde en los puntos encontrados utilizando el conjunto de máscaras de saliencia. Entonces se comienza una agrupación por ambos extremos hasta completar el segmento. Tras computar los extremos con precisión subpíxel, se genera una tabla de acceso similar a la de la figura 2.6, que se utiliza del modo anteriormente indicado para muestrear el color. Finalmente, todos los píxeles colocados en la zona central de la tabla se marcan como visitados para no volver a considerarlos durante el resto del proceso, y se continúa la búsqueda de nuevos segmentos hasta completar el recorrido de la imagen.

Como ya comentamos, es conveniente que las operaciones básicas de procesamiento de imagen (conversiones, filtros, etc.) se implementen utilizando una biblioteca de funciones optimizada, puesto que tienen un efecto determinante en el rendimiento global. Cabe destacar aquí que el filtro con las máscaras de orientación no supone una carga demasiado significativa dado que, aunque aplicar el conjunto completo de máscaras podría ser relativamente costoso, éstas tienen que aplicarse exclusivamente en los píxeles con respuesta de borde activa. En realidad, incluso, sólo en un subconjunto de los mismos, puesto que en el proceso de agrupamiento únicamente se vuelven a utilizar al llegar al extremo de los recorridos del tipo de la figura 2.5. En definitiva, las máscaras son sólo aplicadas en una proporción mínima de puntos de la imagen. A ello hay que añadir que cada máscara tiene también un elevado número de componentes nulos, lo que también acelera el proceso de cómputo.

2.5. Defensa de la técnica de segmentación propuesta

2.5.1. Poder expresivo

Los segmentos con información de color permiten capturar el grueso de la información de interés de una imagen, especialmente si ésta corresponde a un entorno estructurado, co-

ENTRADA:
- Imagen RGB (I^{rgb}) de tamaño $h \times w$ píxeles.
- Tamaño del filtro de la mediana (m) para el posible suavizado de la imagen.
- Umbrales alto (τ_h) y bajo (τ_l) de histéresis sobre la imagen paso-alto para la captación de píxeles.
- Umbral n_{min} de mínimo número de píxeles captados en el entorno local de una máscara para la continuación de un segmento.
- Anchura W_{seg} del recorrido de un segmento para el muestreo del color.
- Parámetros R, W_{edg} y A para las máscaras de saliencia y tablas de recorrido local de píxeles.

SALIDA:
- Conjunto de S segmentos $((x_1^s, y_1^s), (x_2^s, y_2^s))$ con información de color a cada lado, $(r_{izq}^s, g_{izq}^s, b_{izq}^s)$ y $(r_{der}^s, g_{der}^s, b_{der}^s)$, $s = 1, \ldots, S$.

ALGORITMO:

Cálculo de las máscaras: */* Necesario sólo durante la inicialización del sistema. */*
- Calcular las máscaras de saliencia L utilizando la ecuación 2.3 (ver figura 2.4).
- Calcular los recorridos locales orientados de píxel, ordenando las posiciones no nulas de las máscaras anteriores por distancia al centro, en direcciones contrarias ($+$ y $-$) (ver figura 2.5).

Preprocesamiento:
- Convertir la imagen I^{rgb} a niveles de gris, I^{gray}, usando la ecuación 2.1.
- Utilizar, si se considera necesario, el filtro de la mediana (tamaño $m \times m$) sobre I^{gray} para obtener I^{med}.
- Calcular la imagen de bordes I^{brd} usando el filtro paso-alto de la figura 2.1 sobre la imagen I^{gray} (o, en su caso, I^{med}) (ecuación 2.2).
- Marcar todos los píxeles de I^{brd} como "No visitados".
- Inicializar el contador de segmentos, $S := 0$.

Recorrido de la imagen de bordes:

for cada posición de imagen (x, y) ***do***

if $I_{x,y}^{brd} \geq \tau_h$ ***and*** (x, y) está marcado como "No Visitado" ***then***

- Incrementar el contador de segmentos, $S := S + 1$.

Orientación local y agrupamiento:

- Aplicar el conjunto de máscaras de orientación L sobre la posición (x, y) de la imagen paso-alto, I^{brd}. Sea a^{max} el índice de la máscara que obtiene la máxima respuesta (ecuación 2.4).

for *dir* ***in*** $\{+, -\}$ ***do*** */* Agrupamiento en ambas direcciones */*

- Inicializar orientación actual, $a^{act} := a^{max}$.
- Inicializar el extremo actual, $(x_{dir}, y_{dir}) := (x, y)$.

repeat

- Utilizando el recorrido precomputado para la orientación a^{act} (ver figura 2.5) y la dirección *dir*, centrados en el punto actual (x_{dir}, y_{dir}), ir capturando píxeles para los que $I^{brd} > \tau_l$, marcándolos como "Visitados".
- Actualizar el extremo con el último punto capturado, $(x_{dir}, y_{dir}) := (x^{ult}, y^{ult})$.
- Volver a aplicar el conjunto de máscaras L a I^{brd}, pero ahora sobre la posición (x_{dir}, y_{dir}). Sea a^{act} el nuevo índice (orientación) con máxima respuesta.

until (Número de puntos capturados en esta iteración $< n_{\min}$) ***or***
(Distancia entre a^{act} y $a^{max} > 1$)

endfor

Algoritmo 2.2: *Reducción de una imagen RGB a segmentos con información lateral de color (continúa en página siguiente).*

Refinamiento de los extremos:
- Calcular el centroide, la matriz de covarianza, y el vector propio principal del conjunto de puntos capturado para el segmento S actual, utilizando las ecuaciones 2.5, 2.6 y 2.7.
- Normalizar este último vector.
- Obtener con precisión subpíxel (x_1^s, y_1^s) y (x_2^s, y_2^s) proyectando los extremos (x_+, y_+) y (x_-, y_-) sobre la dirección principal, utilizando las ecuaciones 2.8 y 2.9.

Muestreado de color y postprocesamiento:
- Calcular la tabla de acceso (véase figura 2.6) correspondiente al segmento S actual, utilizando el algoritmo 2.1.
- Calcular la mediana de los canales RGB (por separado) de la imagen I^{rgb} para los píxeles cuya posición está indicada en las filas $0 \ldots \lfloor \frac{r}{2} \rfloor - 1$ de la tabla de acceso. El vector $(r_{izq}^s, g_{izq}^s, b_{izq}^s)$ obtenido es la información de color izquierda para el segmento S.
- Repetir el proceso anterior para obtener $(r_{der}^s, g_{der}^s, b_{der}^s)$, recorriendo las filas $\lfloor \frac{r}{2} \rfloor + 1 \ldots r - 1$ de la tabla.
- Marcar los píxeles de apuntados por las filas $\lfloor \frac{r}{2} \rfloor - \lfloor \frac{r}{6} \rfloor \ldots \lfloor \frac{r}{2} \rfloor + \lfloor \frac{r}{6} \rfloor$ como "Visitados".

endif
endfor

***Algoritmo 2.2 (continuación)**: Reducción de una imagen RGB a segmentos con información lateral de color (viene de la página anterior).*

mo las escenas interiores en las que se moverá nuestro robot. En este apartado justificaremos esta afirmación comprobando la capacidad informativa de los segmentos extraídos. Para ello, hemos desarrollado un sencillo algoritmo capaz de reconstruir una versión aproximada de la imagen original utilizando sólo la información obtenida por nuestro procedimiento. El método de recuperación se resume en el algoritmo 2.3.

Se trata de un simple procedimiento de difusión, semejante al empleado en simulaciones computacionales de la transmisión de calor sobre un cuerpo físico representado mediante una matriz de elementos discretos. La imagen es inicializada a partir del conjunto de segmentos, colocando en todos aquellos píxeles ubicados inmediatamente a la derecha e izquierda de cada uno los correspondientes valores RGB medianos. Dichos píxeles quedan fijos, de modo que su valor no se alterará a lo largo del proceso de difusión. Estos puntos serían análogos a las "fuentes de calor" de la simulación mencionada. El resto de píxeles de la imagen quedan sin inicializar. Entonces se entra en un bucle que va recalculando el valor de las componentes R, G y B de cada punto como la media aritmética de los valores de los vecinos a los que ya se ha dado algún valor. A su vez, el píxel actualizado se marca también como inicializado. Al cabo de unas cuantas iteraciones (dependiendo de la cantidad y distribución de los segmentos sobre la imagen) el proceso converge a una solución estable, en la cual las variaciones con respecto a la anterior comienzan a resultar despreciables. En ese momento puede detenerse la ejecución.

La figura 2.7 muestra el resultado del proceso de recuperación sobre la imagen utilizada como ejemplo anteriormente. A la izquierda (figura 2.7(a)) se muestra el conjunto de segmentos extraído, coloreados a ambos lados con los correspondientes colores medianos. Esta imagen es la utilizada como inicialización para el posterior proceso de difusión, que converge

ENTRADA:
- Conjunto de S segmentos $((x_1^s, y_1^s), (x_2^s, y_2^s))$ con información de color a cada lado, $(r_{izq}^s, g_{izq}^s, b_{izq}^s)$ y $(r_{der}^s, g_{der}^s, b_{der}^s)$, $s = 1, \ldots, S$.

SALIDA:
- Imagen RGB reconstruida, (I^{rgb}) de tamaño $h \times w$ píxeles.

ALGORITMO:

Inicialización:

- Marcar todos los píxeles de la imagen I^{rgb} como "No inicializados".

for s ***in*** $1, \ldots, S$ ***do*** /* *Para cada segmento:* */

- Generar la tabla $(t_{i,j}^x, t_{i,j}^y)^s$ con r^s filas y c^s columnas (usar $W_{seg} = 1.0$ para obtener $r^s = 3$).

for $j = 0$ ***to*** $c^s - 1$ ***do*** /* *Inicializamos los píxeles de cada lado con el color correspondiente:* */

if $0 \leq t_{0,j}^x < w$ ***and*** $0 \leq t_{0,j}^y < h$ ***then*** /* *Lado izquierdo.* */

$(I^r_{(t_{0,j}^x, t_{0,j}^y)^s}, I^g_{(t_{0,j}^x, t_{0,j}^y)^s}, I^b_{(t_{0,j}^x, t_{0,j}^y)^s}) = (r_{izq}^s, g_{izq}^s, b_{izq}^s)$.

endif

if $0 \leq t_{2,j}^x < w$ ***and*** $0 \leq t_{2,j}^y < h$ ***then*** /* *Lado derecho.* */

$(I^r_{(t_{2,j}^x, t_{2,j}^y)^s}, I^g_{(t_{2,j}^x, t_{2,j}^y)^s}, I^b_{(t_{2,j}^x, t_{2,j}^y)^s}) := (r_{der}^s, g_{der}^s, b_{der}^s)$.

endif

endfor

endfor

- Marcar todos los píxeles inicializados como "Fijos".

Iteración principal: /* *Proceso de difusión:* */

while not (convergencia hasta el grado deseado) ***do***

for cada posición de imagen (x, y) ***do*** /* *Trabajamos sobre una imagen intermedia* I'^{rgb}: */

if $I_{x,y}^{rgb}$ no está marcado como "Fijo" ***then***

$$(I'^r_{x,y}, I'^g_{x,y}, I'^b_{x,y}) := \frac{\sum_{k=x-1}^{k=x+1} \sum_{l=y-1}^{l=y+1} (I^r_{k,l}, I^g_{k,l}, I^b_{k,l})}{\sum_{k=x-1}^{k=x+1} \sum_{l=y-1}^{l=y+1} 1}, \quad I_{k,l}^{rgb} \neq \text{"No inicializado"}$$

- Marcar $I_{x,y}^{rgb}$ como "Inicializado".

endif

endfor

for cada posición de imagen (x, y) ***do*** /* *Pasamos de nuevo a la imagen principal:* */

$(I^r_{x,y}, I^g_{x,y}, I^b_{x,y}) := (I'^r_{x,y}, I'^g_{x,y}, I'^b_{x,y})$.

endfor

endwhile

Algoritmo 2.3: *Reconstrucción de una imagen a partir de un conjunto de segmentos etiquetados con información de color.*

hasta un nivel de precisión aceptable al cabo de unas 200 iteraciones. La imagen reconstruida se muestra en el centro (figura 2.7(b)). Por supuesto, la cantidad de iteraciones necesarias es variable en función tanto del tamaño de la imagen como de la complejidad de la misma, pero, típicamente, para imágenes de tamaño PAL/2 como las utilizadas en el sistema de visión

(a) (b) (c)

***Figura 2.7**: (a) Segmentos con información de color (imagen inicial del proceso de difusión). (b) Imagen recuperada (resultado del proceso de difusión). (c) Imagen JPEG con el mismo ratio de compresión.*

de nuestro robot, suele bastar con un número entre 200 y 400. El proceso de reconstrucción es, pues, bastante más lento que el proceso de extracción. De todos modos esto no supone un problema grave, puesto que su única utilidad radica en demostrar la potencia expresiva de los segmentos coloreados, y en realidad no es utilizado en ningún momento durante la navegación visual. Aún así, merece la pena hacer notar que si se dispone de una buena imagen inicial la convergencia es bastante más rápida. Por tanto, si se aplica el proceso de reconstrucción para una secuencia larga de imágenes, podemos emplear la reconstrucción de cada *frame* como solución inicial para el siguiente, aprovechando la misma idea de estándares de compresión de vídeo habituales como MPEG. De este modo, y fijando los píxeles correspondientes a los lados del nuevo repertorio de segmentos, el proceso de reconstrucción se ve significativamente acelerado.

Lo verdaderamente interesante de la imagen reconstruida es que, hasta un cierto grado de detalle, se conservan las estructuras geométrica y cromática básicas de la inicial, a pesar de la gran reducción en tamaño de la información manejada. Por ejemplo, la imagen original de la figura 2.7(a) tiene un tamaño de $288 \times 384 \times 3 = 324$ KB (tamaño PAL/2, con un byte por cada canal RGB). La información comprimida, por otro lado, está formada por un total de 261 segmentos, cada uno de los cuales está etiquetado con un par de valores RGB. Puesto que cada extremo del segmento puede codificarse utilizando sólo $\lceil \log_2(288 \times 384) \rceil = 17$ bits, y cada valor RGB, usando un byte por canal, con 24 bits, el conjunto completo de segmentos ocupa $261 \times ((2 \times 17) + 2 \times 24) = 21402$ bits $= 2.61$ KB. Esto supone un factor de compresión del 99.2 %. Por supuesto, este factor variará con el tamaño y complejidad de la imagen, pero, en general, la reducción en el espacio de almacenamiento empleado es de aproximadamente dos órdenes de magnitud. Con el fin de comparar con un método estándar de compresión, la figura 2.7(c) muestra la misma imagen comprimida con el formato JPEG hasta lograr un tamaño equivalente. En dicha imagen se observa que, si bien algunas zonas con mucho detalle quedan mejor registradas que con el algoritmo de reconstrucción propuesto, los bordes de los objetos más importantes (puerta, muebles, libros, cámara, etc.) quedan reflejados con mayor precisión en la reconstrucción basada en segmentos (figura 2.7(b)). La imagen JPEG muestra

también mayores problemas de *pixelización*, debido al nocivo efecto de la cuantización de la gama de valores de color provocado por el alto grado de compresión. Por el contrario, la imagen reconstruída muestra una transición mucho más suave en las zonas sin detalle, mientras que los bordes permanecen suficientemente nítidos y precisos.

En realidad, las diferentes características de las dos imágenes revelan de algún modo las diferentes filosofías de compactación subyacentes. La extracción de segmentos focaliza su atención en la conservación de la estructura geométrica de la imagen, a través de sus bordes rectos. La compresión JPEG, sin embargo, no toma en cuenta la estructura de la escena, limitándose a dividir la imagen en zonas rectangulares sobre las que aplica la transformada discreta del coseno, de la que luego se descartan las componentes de menos peso en la transformación (tantas más cuanto mayor sea el grado de compactación elegido) (Gonzalez y Woods, 1993). No obstante, está claro que es esto último lo que hace a la compresión JPEG mucho más general, puesto que es capaz de trabajar también con imágenes cuya estructura no puede ser capturada de modo adecuado utilizando únicamente segmentos. Tal puede ser el caso, por ejemplo, de escenas naturales en exteriores.

Pero es precisamente en la expresividad geométrica del modelo donde radica la mayor parte de su potencia. La utilización de la posición y orientación de los segmentos ofrece muchas facilidades de cara a la posterior tarea de reconstrucción tridimensional de la escena. Esto es especialmente cierto si la herramienta teórica empleada es la geometría proyectiva, en donde las primitivas más comúnmente utilizadas son las rectas y los puntos, mucho más manejables bajo el formalismo matemático subyacente que otros elementos más complejos, como curvas, superficies, etc. Los métodos existentes aprovechan principalmente las posiciones de dichas características para explotar las restricciones entre las múltiples vistas y resolver problemas tales como el seguimiento y emparejamiento de primitivas, la calibración de las cámaras, la reconstrucción del entorno o la estimación del movimiento, entre otros (Hartley y Zisserman, 2000). Aún así, el hecho de considerar también el color añade ciertamente algunas ventajas, incluso en este tipo de problemas. En el apartado que sigue enumeramos y comentamos algunas de estas ventajas.

2.5.2. Ventajas de la información de color

La información cromática adicional que proponemos puede resultar de cierta ayuda en múltiples ámbitos de la visión por computador y el procesamiento de imágenes. En esta sección haremos un breve recorrido por algunas de sus posibles aplicaciones.

Seguimiento de primitivas

Si bien es cierto que existen muchos métodos para el seguimiento de objetos en tiempo real basados en la textura y el color, lo más habitual es que trabajen sobre conjuntos de píxeles desestructurados, donde la información de interés es simplemente la posición aproximada

del objeto seguido en la imagen (véase, por ejemplo, el anteriormente citado ejemplo de McKenna *et al.* (1999)). Este tipo de procedimiento suele ser útil en tareas de monitorización, por ejemplo.

La inmensa mayoría de técnicas proyectivas, sin embargo, basan su operación en la localización y posterior emparejamiento de esquinas o segmentos entre imágenes de una misma escena tomadas desde distintos puntos de vista. En este otro tipo de aplicaciones es necesaria una localización más precisa de la característica a seguir. En el caso de las esquinas puede usarse el entorno local del punto localizado para realizar el posible emparejamiento (es el caso de los ya comentados métodos de Lucas y Kanade (1981), Shi y Tomasi (1994) y Tommasini *et al.* (1998)). En el de los segmentos, aunque también hay algunos trabajos relacionados con el uso del entorno de los mismos, la mayoría simplemente utilizan su posición y orientación, y confían en un estimador estadísticamente robusto para ir actualizando estos valores conforme cambia la imagen de entrada (por ejemplo, un filtro de Kalman en (Zhang, 1994), o el algoritmo EM (McLachlan y Krishnan, 1997) en (López de Teruel *et al.*, 2000)).

El tiempo real impone una fuerte restricción sobre el uso de algunas otras técnicas, que a menudo son intensivas en cómputo y por tanto se suelen usar sólo en aplicaciones de tipo *batch* (donde primero se graba la secuencia de imágenes completa para trabajar después sobre ella sin restricciones en el tiempo de ejecución). Un ejemplo de ello se puede encontrar en el interesante trabajo de Schmid y Zisserman (1997), donde se utiliza la restricción epipolar entre el par de imágenes para obtener correspondencias punto a punto entre segmentos candidatos al emparejamiento, para evaluar posteriormente una medida de similitud basada en la correlación local entre píxeles localizados de este modo. El problema es que el método necesita la estimación robusta de la matriz fundamental para cada par de imágenes, lo que dificulta su aplicabilidad en sistemas donde el tratamiento de la secuencia de imágenes debe realizarse *on-line*.

En nuestro caso, el doble vector de color extraído con el segmento es una información que, al tiempo que se obtiene y procesa sin apenas esfuerzo computacional añadido, permite restringir bastante el espacio de búsqueda. Se pueden diseñar así algoritmos robustos de emparejamiento y seguimiento más sencillos. Una posibilidad sería elaborar una medida de similitud entre estos vectores que favorezca el *matching* entre segmentos con colores compatibles, y penalice o incluso descarte aquellos otros con colores claramente discordantes. Esta medida, por supuesto, debería ser adecuadamente combinada con otra de distancia geométrica, que tenga en cuenta las respectivas posiciones y orientaciones. En el capítulo cuarto describiremos una aplicación de esta técnica en el sistema de visión de nuestro robot.

Localización de puntos por intersección

A menudo no es tan importante el hecho de encontrar muchos emparejamientos como hacerlo de manera robusta, aunque sólo sea para unas pocas características. Para obtener la

restricción epipolar entre dos imágenes calculando la matriz fundamental, por ejemplo, basta encontrar siete pares de puntos correspondientes entre dos imágenes [8]. Un modo de trabajar es ir cogiendo subconjuntos de unos pocos pares y calculando las distintas matrices fundamentales resultantes, para descartar aquellas que no son consistentes con un mínimo del resto de emparejamientos hasta un cierto nivel de tolerancia (se trata de una aplicación del conocido método RANSAC (Fischler y Bolles, 1981)). Utilizando nuestras primitivas, una alternativa sería localizar las esquinas mediante la intersección de segmentos no alineados cuyos extremos están cerca y que tienen colores compatibles entre sí, de acuerdo con su posición en la imagen. Cuanto más largos sean estos segmentos, más probabilidades hay de localizar los segmentos homólogos en la otra imagen y de que las correspondencias obtenidas sean correctas. Si, alternativamente, se dispone de tres o más vistas, entonces pueden utilizarse restricciones multivista que involucren directamente relaciones entre segmentos, sin tener que recurrir a la localización de puntos. El tensor trifocal, por ejemplo, puede estimarse directamente a partir de correspondencias entre segmentos, al contrario que la matriz fundamental.

Clasificación de objetos

Los valores de color se pueden utilizar también en tareas de clasificación, al estilo de las descritas por Buluswar y Draper (1998). Por ejemplo, si se buscan objetos de colores característicos en la escena, conocidos *a priori*, pueden usarse sencillos clasificadores basados en la información cromática para asociar directamente cada segmento a la clase correspondiente. Esta técnica puede llegar a ser especialmente útil en entornos relativamente controlados. Nuestro sistema autónomo, como se verá más adelante, utiliza la información de color de los segmentos extraídos en la imagen para clasificarlos como pertenecientes a tramos de pared, puertas, señales, obstáculos, etc., basándose en el conocimiento previo del color aproximado de estos elementos. Esta clasificación es utilizada posteriormente para realizar interpretaciones tentativas de la escena, que después serán corroboradas o descartadas según sean o no consistentes con el posterior movimiento de la cámara en relación a la escena. De nuevo, más adelante describiremos con detalle este mecanismo en el que se basa el motor interpretativo del robot.

Detección de contornos

A veces también resulta útil localizar la silueta de un determinado objeto aunque no se conozca su coloración. De este modo pueden aislarse objetos de interés en la imagen para su interpretación basándonos en su forma o posición, por ejemplo. Para ello, podemos buscar segmentos con extremos cercanos y colores similares con el fin de agruparlos en contornos sin que otros vecinos de colores discrepantes entorpezcan el proceso de búsqueda. De algún

[8] U ocho si se desea trabajar únicamente con restricciones lineales.

modo, se trata del proceso inverso a la aproximación poligonal, puesto que se parte del conjunto de segmentos aislados para obtener secuencias ordenadas de los mismos. Una vez más, esta posibilidad es aprovechada en nuestra arquitectura de percepción a la hora de localizar señales y obstáculos cuyo color es en principio desconocido.

Compresión de imágenes

Por último, tal y como describimos en la sección anterior, los segmentos coloreados constituyen un potente método de compresión, puesto que son capaces de sintetizar la información fotométrica de la imagen sin pérdida significativa de los detalles importantes. Existe un amplio rango de aplicaciones en las que, una vez extraído el conjunto de características, ya no se necesita la imagen original, que ocupa un tamaño en torno a cien veces mayor. Esto puede aprovecharse si se quiere, por ejemplo, enviar la secuencia de imágenes a través de un canal de transmisión con ancho de banda limitado, como la conexión inalámbrica que une al robot con el resto de la red, con el objeto de realizar el resto del procesamiento de modo externo. Ésta, de hecho, es una de las posibles configuraciones de cómputo que se utilizan en nuestro agente autónomo[9]. En este caso podríamos enviar a través del canal únicamente los segmentos con información de color extraídos, que consumirían menos ancho de banda que las imágenes completas originales. Otras aplicaciones podrían ser monitorizar remotamente la operación en tiempo real o, simplemente, salvar en disco secuencias largas de imágenes ocupando un tamaño mucho menor.

2.6. Resultados

En el último apartado de este capítulo mostraremos algunos ejemplos de funcionamiento del algoritmo 2.2, comparándolo con algunos métodos tradicionales de extracción de segmentos descritos en la literatura. La comparación se realizará atendiendo tanto a la calidad visual de los resultados como en términos de los respectivos tiempos de ejecución. Trataremos de justificar cómo en ambos aspectos nuestra propuesta posee ciertas ventajas significativas, las cuales la hacen muy atractiva para aplicaciones de procesamiento de imagen en tiempo real como la que nos ocupa.

2.6.1. Operación del algoritmo

En la figura 2.8 se muestran algunos ejemplos de operación sobre distintas imágenes de ejemplo. Con el fin de apreciar mejor la calidad visual de los resultados, se presenta sólo la información geométrica extraída por el procedimiento (posición de los segmentos). El color muestreado, por otro lado, queda implícitamente recogido en la figura 2.9, resultado de

[9]Describiremos ésta y otras posibilidades en mayor profundidad en el capítulo quinto, dedicado a la arquitectura hardware-software del sistema.

Figura 2.8: *Ejemplos de operación del algoritmo (se muestra sólo la posición de los segmentos, sin incluir la información de color) (a-c,d-f) Imágenes típicas recogidas por el robot durante periodos de navegación, con los correspondientes conjuntos de segmentos extraídos. (g-i,j-l) Resultados obtenidos sobre escenas más complejas, con mayor cantidad de detalles.*

aplicar el proceso de recuperación descrito en el algoritmo 2.3 a los conjuntos de segmentos obtenidos de las respectivas imágenes.

En las figuras 2.8(a-c,d-f) se recogen tres ejemplos de escenas típicas con las que trabaja el robot, correspondientes al entorno en el que éste opera. Como puede observarse, estas

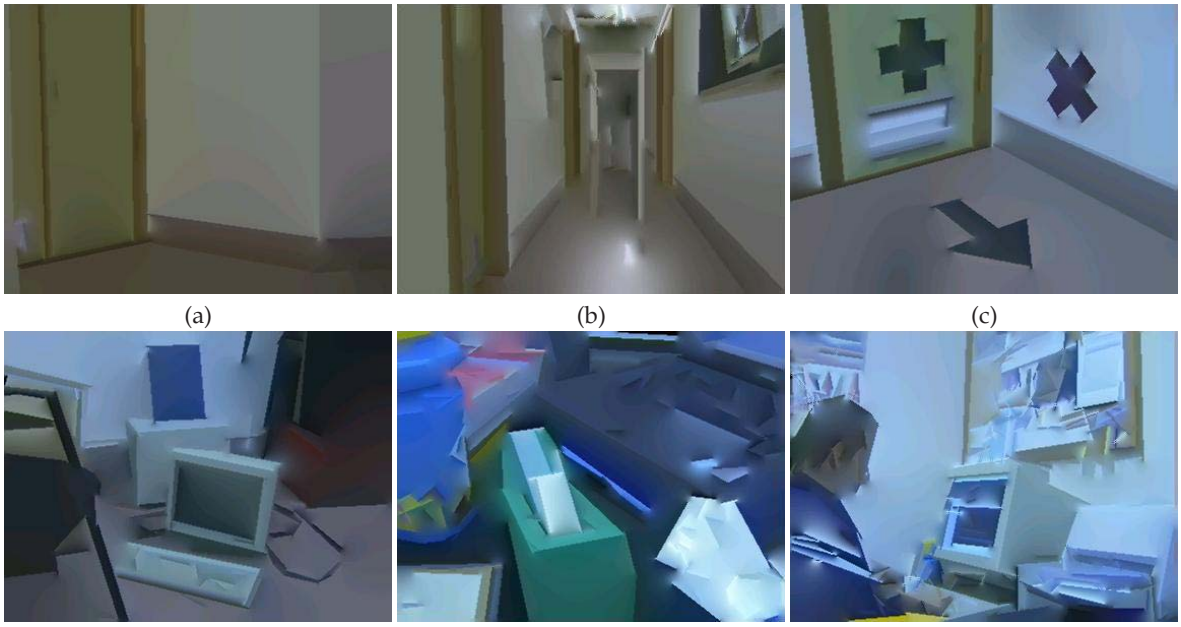


Figura 2.9: *Imágenes recuperadas a partir de la información de color (ver figura 2.8).*

imágenes quedan perfectamente caracterizadas mediante segmentos, puesto que los principales elementos de la escena están formados por paredes, puertas, señales y otros objetos a los que se ajusta perfectamente la primitiva. La información de color queda también recogida de modo bastante satisfactorio, como se puede apreciar en las correspondientes reconstrucciones de las figuras 2.9(a-c). En las figuras 2.8(g-i,j-l) se muestran otros ejemplos de complejidad creciente, donde empiezan a aparecer contornos que cada vez se ajustan menos a la restricción de linealidad. En estos casos el algoritmo aproxima los bordes por tramos usando segmentos más pequeños. Este efecto se aprecia muy claramente en el cable del ordenador de la figura 2.8(g,j), por ejemplo. Aún así, el procedimiento detecta bien los segmentos más largos, que son capturados sin ser afectados por el resto de detalles de la imagen.

Esta propiedad del algoritmo, derivada de su esquema de procesamiento inherentemente *local* a cada segmento (frente al procesamiento más *global* de aproximaciones como la transformada de Hough tradicional, donde cada punto podría en principio formar parte de cualquier segmento) hace al algoritmo más escalable frente a escenas más complejas, tanto en términos de tiempo de ejecución como de calidad de los resultados. A pesar de la lógica pérdida progresiva de detalles en las reconstrucciones (figuras 2.9(d-f)), es de destacar el hecho de que el grueso de la información fotométrica de las imágenes aún queda aceptablemente capturado.

2.6.2. Rendimiento

A lo largo de todo este capítulo se ha hecho énfasis en que el principal requerimiento que hacemos al proceso de extracción de primitivas es la eficiencia de cómputo. No en vano, debe-

mos recordar que éste es empleado en un sistema de percepción en tiempo real cuya fiabilidad y robustez depende directamente de la velocidad en el tratamiento de las imágenes. Un robot capaz de procesar mayor cantidad de imágenes por segundo podrá detectar, seguir y predecir a mayor velocidad los cambios en su entorno y, en definitiva, de interactuar con éste con mayor fiabilidad. Con el fin de evaluar el grado de obtención de este objetivo hemos realizado también un estudio del tiempo de ejecución del algoritmo sobre distintas plataformas de procesamiento y con distintos tamaños de imagen.

Para realizar dicho estudio en unas condiciones similares a las de funcionamiento real, se tomó una secuencia continua de 300 imágenes con la cámara montada en el robot durante un recorrido de éste en su entorno habitual de trabajo. Se pretendía recoger, de algún modo, una razonable variedad de escenas con las que el robot podría encontrarse durante un recorrido típico, incluyendo múltiples lugares y puntos de vista diferentes. Las figuras 2.8(a-b) corresponden a dos instancias concretas de dicha secuencia. A fin de realizar una comparativa lo más significativa posible, se realizaron pruebas de ejecución del algoritmo implementado sobre este conjunto de imágenes en cuatro versiones de procesadores de Intel de distintas generaciones.

En la figura 2.10 se resumen gráficamente los resultados obtenidos. La primera conclusión que cabe destacar es, en general, la relativa rapidez de cómputo en casi todas las plataformas. Se observa que, a partir del Pentium III a 1 GHz, pueden procesarse las imágenes a una velocidad superior al máximo ofrecido por el estándar PAL (30 *fps*) para imágenes RGB de 768×576 píxeles. Para plataformas menos potentes, como un Pentium III a 566 MHz, puede incluso alcanzarse una velocidad de procesamiento cercana a los 30 *fps*, si se submuestrea la imagen a un tamaño más pequeño, como 512×384 píxeles ($2/3$ del tamaño original). En procesadores bastante más antiguos, como el Pentium 266 MMX que podemos encontrar en el ya algo obsoleto PC de nuestro robot, todavía se obtiene una velocidad promedio de 140 ms/*frame* (equivalente a algo más de 7 *fps*) sobre imágenes de 384×288 (resolución PAL/2, que aún permite un adecuado nivel de detalle para nuestra aplicación de navegación en tiempo real).

Como puede apreciarse en las figuras 2.8(a-b), las imágenes de la secuencia pueden tener distintos grados de complejidad. Dado que el tiempo de ejecución depende directamente de ésta (lógicamente, imágenes con mayor número de segmentos y más detalles consumen mayor tiempo de procesamiento), resulta interesante mostrar, además del tiempo medio de ejecución para la secuencia, una banda de confianza de ± 2 desviaciones típicas para recoger de alguna manera la variabilidad de los resultados. El hecho a destacar en este caso es que, en general, el tiempo de ejecución es sensible al mayor o menor número de píxeles activos en la imagen de bordes, si bien, en condiciones normales, nunca suele alejarse del valor medio en más de un 50 %.

No obstante, para una escena fija el algoritmo muestra una fuerte linealidad en el tiempo de ejecución frente al número de píxeles (resolución) de la imagen. Esta última afirmación se

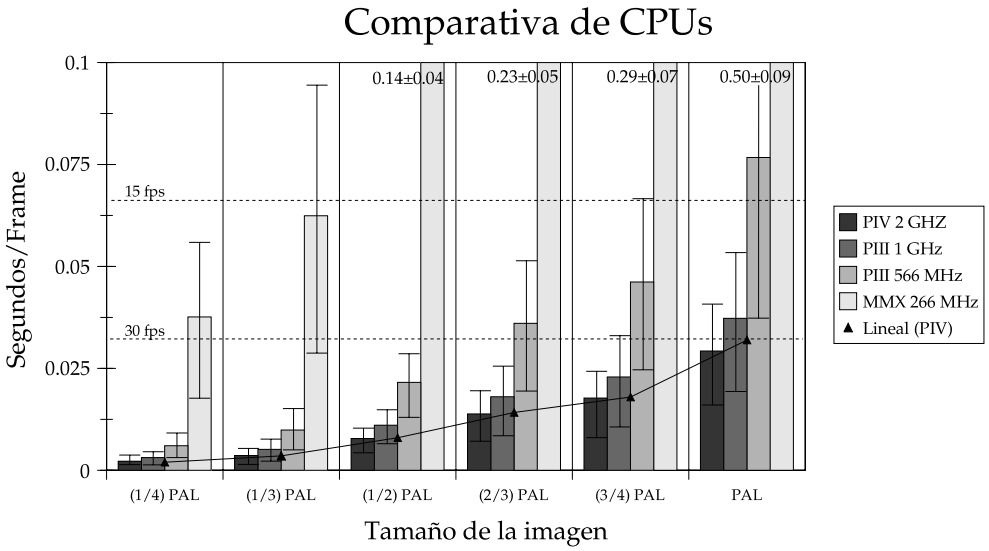


Figura 2.10: *Tiempos de ejecución por* frame *del algoritmo de detección de segmentos coloreados para diversos tamaños de imagen, sobre distintas CPUs. El valor representado es el tiempo medio de ejecución por* frame *sobre una secuencia continua típica de 300 imágenes, obtenida durante un recorrido real del robot. Para cada valor se muestra también una banda de confianza de* ± 2 *desviaciones típicas. Aunque el tiempo empleado en cada* frame *varía con la complejidad de la imagen (puesto que, obviamente, imágenes con mayor detalle son procesadas con mayor lentitud), el tiempo medio de ejecución es prácticamente lineal con el tamaño en píxeles de la imagen, como demuestran los valores predichos por la regresión lineal sobre los tiempos del Pentium IV, muy cercanos al tiempo de ejecución real.*

desprende directamente de un particular estudio elaborado sobre los tiempos de ejecución en el Pentium IV a 2 GHz. En primer lugar se realizó una regresión lineal sobre los tiempos obtenidos para todas las imágenes frente a los respectivos tamaños, con el fin de recoger estadísticamente la relación entre ambas variables. Posteriormente se utilizó el modelo lineal obtenido para predecir el valor del tiempo de ejecución esperado para cada tamaño. Los pequeños triángulos unidos por una línea continua marcan los valores predichos. Como puede apreciarse, dichos valores son muy cercanos al tiempo medio de ejecución (barras en gris más oscuro), lo que confirma la hipótesis expuesta.

2.6.3. Comparación con otros métodos

En esta sección, que cierra el capítulo, compararemos la eficiencia y los resultados de nuestro procedimiento con los obtenidos por otras conocidas técnicas de extracción de segmentos. Para realizar la comparativa elegimos sendos métodos representativos de los dos principales paradigmas comentados en la sección 2.3 (los basados en la transformada de Hough, por un lado,y los fundamentados en la aproximación poligonal, por otro). Como ejemplo del primer tipo elegimos la llamada *transformada de Hough probabilística progresiva* (PPHT), una variante relativamente moderna de la HT clásica adaptada para ser aplicada en entornos de tiempo real (Matas *et al.*, 2000). En cuanto a la aproximación poligonal, se usó el método de Douglas

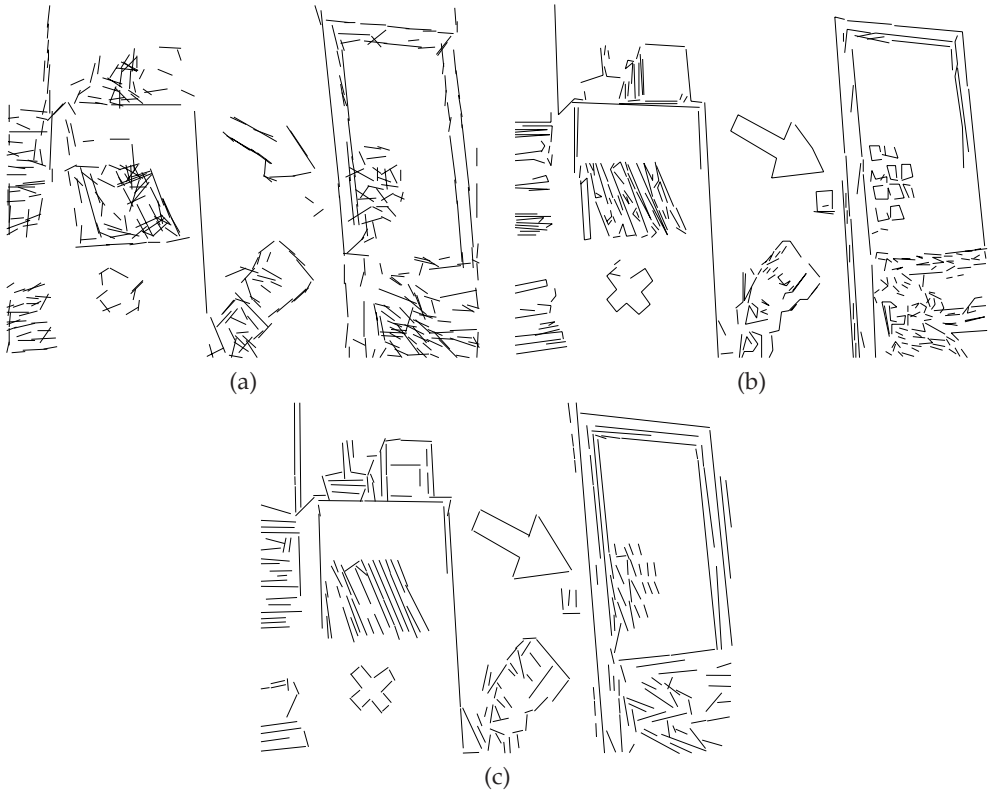


Figura 2.11: *Resultado de los tres algoritmos sobre la imagen de ejemplo de la figura 2.2: (a) Segmentos obtenidos por la PPHT. (b) Segmentos obtenidos por la aproximación poligonal de Douglas-Peuker (sobre una imagen de bordes computada con el operador de Canny). (c) Segmentos obtenidos por nuestro método.*

y Peuker (1973), basado en la eliminación iterativa de los puntos del contorno para ir obteniendo una polilínea cada vez más simple y de error mínimo. Ambos métodos están implementados en la OpenCV, una biblioteca de funciones de fuente abierta patrocinada por Intel que está haciéndose cada vez más popular dentro de la comunidad investigadora en visión artificial (Intel Corporation, 2000b), y que funciona a su vez sobre las eficiente IPL anteriormente mencionadas (al igual que nuestro método). De todos los métodos de extracción de segmentos de esta biblioteca, éstos son los más rápidos dentro de sus respectivas clases (variantes de la HT y aproximaciones poligonales). Ambos métodos trabajan sobre una imagen binaria de bordes, que se obtiene a partir de la original a través del operador de Canny.

La figura 2.11 muestra los resultados obtenidos para los tres procedimientos sobre una imagen de entrada de ejemplo. En general cabe destacar que, aún intentando un ajuste fino de sus parámetros (grado de discretización de la matriz de votación, espacio intersegmentos, etc.) la PPHT obtiene resultados claramente inferiores cualitativamente (figura 2.11(a)), con un conjunto de segmentos más fragmentado e impreciso. La aproximación poligonal de contornos (figura 2.11(b)), aunque manifiestamente mejor que la PPHT, queda aún algo más ruidosa que la conseguida por nuestro procedimiento (figura 2.11(c)). Una posible razón es

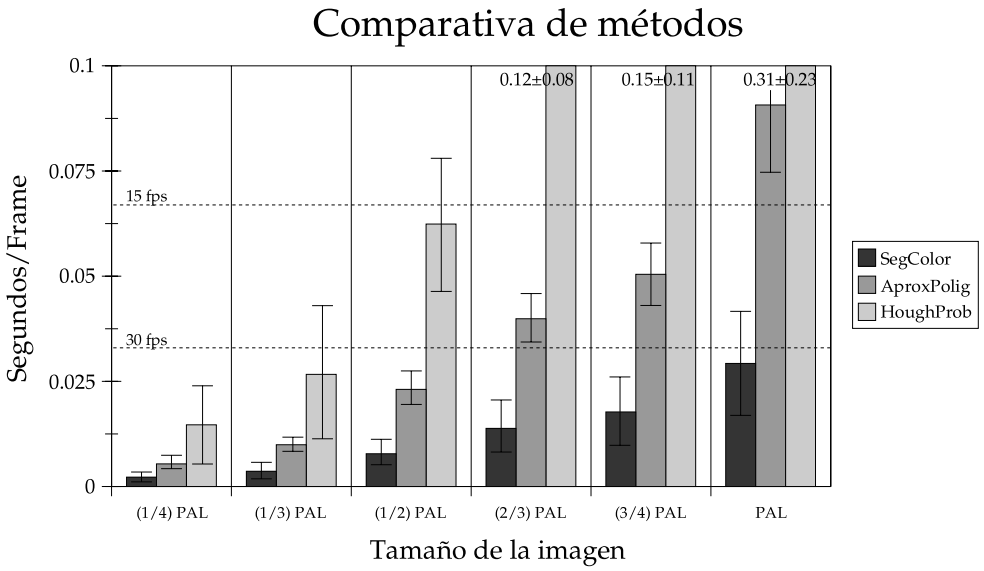


Figura 2.12: *Comparativa de tiempos de ejecución por* frame *para distintos tamaños de imagen y tres métodos distintos de detección (el algoritmo propuesto, un método de aproximación poligonal y una variante la transformada de Hough). De nuevo, se representa el tiempo medio de ejecución sobre la misma secuencia anterior, dentro de un margen de* ± 2 *desviaciones típicas.*

el carácter local del operador de Canny, frente al más global de las máscaras de orientación de tamaño 9×9 utilizadas por nuestra técnica. Otra razón añadida es que el operador de Canny carece de la etapa de "limpieza" en el postprocesamiento de cada segmento. Ambos factores dotan a nuestro acercamiento de un beneficioso efecto regularizador, que favorece a segmentos más largos frente a la captura de los detalles menos importantes. Obviamente, también aquí podríamos jugar con los parámetros del método de aproximación poligonal (como el tamaño de la máscara de suavizado *Gaussiano* del operador de Canny, el error tolerado en la aproximación, etc.) para intentar lograr una solución más regularizada. Sin embargo, esto también afectaría al tiempo de ejecución, así como a la cantidad de bordes detectados, que iría siendo menor al aumentar el tamaño del filtro. En el ejemplo mostrado se ajustaron manualmente todos estos parámetros para obtener una solución de compromiso aceptable, que alcanzase un grado de detalle similar al de nuestro procedimiento.

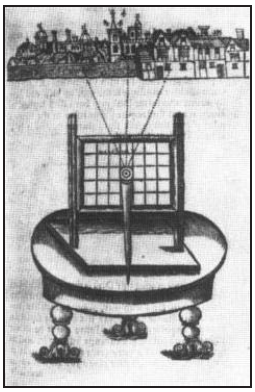
Finalmente, realizamos también un estudio comparativo de la velocidad de procesamiento de estos métodos frente al nuestro. La figura 2.12 resume los resultados de este otro estudio, realizado en un procesador Pentium IV a 2 GHz sobre la misma secuencia y tamaños de imagen que en la sección anterior. Como conclusión cabe destacar que, en todos los casos, el extractor de segmentos propuesto se ejecuta en menos de la mitad de tiempo que el método de aproximación poligonal, y casi diez veces más rápido que la PPHT, en término medio. Esto le permite ser el único que asegura una velocidad de procesamiento superior a la de captura (30 *fps*) para todos los tamaños de imagen.

2.7. Resumen

He aquí un resumen de las principales ideas que se han expuesto en este capítulo:

- Frente a otras alternativas como las regiones o las esquinas, se eligen los segmentos como primitiva perceptual de nivel inferior, por su adecuación a las escenas estructuradas en las que se moverá el robot, así como el buen compromiso alcanzado entre eficiencia de cómputo y capacidad descriptiva.
- La técnica de extracción propuesta se basa en un procesamiento eficiente de la imagen, en el que se tiene en cuenta la saliencia local de los bordes alineados para el posterior agrupamiento de píxeles en la formación de los segmentos.
- La información geométrica de cada segmento es ampliada con un muestreo robusto del color a ambos lados del mismo, utilizando un método de acceso eficiente a los píxeles de su entorno.
- Se demuestra la potencia expresiva de la primitiva desarrollando un procedimiento de recuperación de la imagen original que, partiendo únicamente de los segmentos con la correspondiente información de color, es capaz de recuperar una buena aproximación de la imagen original.
- Se sugieren algunas aplicaciones en distintos ámbitos de la visión artificial. En particular, la información de color añadida resultará especialmente útil en tareas claves dentro de la arquitectura de percepción, tales como la localización de contornos de objetos y señales, la creación inicial de interpretaciones tentativas de la escena y el seguimiento robusto de las correspondientes estructuras construidas.
- Los experimentos realizados demuestran la eficiencia, potencia expresiva y precisión de la técnica propuesta, cualidades todas ellas imprescindibles para poder abordar la percepción de alto nivel bajo la restricción de tiempo real exigida por la navegación.

CAPÍTULO III



"Perspectiva", J. Bate, 1635

Autocalibración a partir de odometría

La perspectiva, por consiguiente, debe ocupar el primer puesto entre todos los discursos y disciplinas humanas. En su dominio, la línea luminosa se combina con las variedades de la demostración y se adorna gloriosamente con las flores de la matemática, y más aún de la física. Sus resultados pueden detallarse analíticamente; pero me propongo encerrarlos en breves conclusiones, entretejiendo, según la modalidad de la materia tratada, demostraciones naturales y matemáticas, y deduciendo a veces los efectos de las causas, y otras veces las causas de los efectos.

LEONARDO DA VINCI, *Aforismos.*

3.1. Introducción

En los sistemas autónomos que interactúan con el entorno guiados únicamente a partir de la información visual, es fundamental disponer de mecanismos para extraer algún tipo de información estructural de la escena a partir de las imágenes obtenidas. En los últimos años, la investigación en visión por computador ha experimentado extraordinarios avances en el campo de la reconstrucción tridimensional de escenas a partir de una serie de imágenes tomadas desde distintos puntos de vista (Aström, 2000). Podemos suponer que las imágenes se toman simultáneamente desde múltiples cámaras, o bien sólo mediante una que se mueve a través de la escena, posiblemente cambiando durante la secuencia su configuración interna (por ejemplo, la distancia focal). Para escenas estáticas, no hay diferencias conceptuales importantes entre ambos casos a la hora de resolver el problema. En general, no obstante, nosotros nos centraremos en el segundo, puesto que la plataforma robótica móvil con la que trabajaremos es monocular, es decir, utiliza una única cámara como sensor óptico.

Las soluciones propuestas para estos problemas están basadas en la aplicación directa de la geometría proyectiva, una herramienta matemática que se ajusta muy bien a este campo de la visión artificial (Hartley y Zisserman, 2000; Faugeras, 2001). Dentro de las técnicas de reconstrucción basadas en este formalismo pueden alcanzarse distintos niveles en los tipos de reconstrucciones practicadas. Se suele hablar de reconstrucciones proyectivas, afines, similares o euclídeas[1] según el tipo de invariantes que se conservan en la reconstrucción con respecto a la escena original. Así, una reconstrucción proyectiva conserva sólo propiedades tales como la colinealidad, la convexidad, la incidencia o la tangencia, e invariantes como el *cross-ratio* de distancia entre puntos alineados, por ejemplo. Una afín, de nivel inmediatamente superior, comienza a respetar invariantes más fuertes, como el paralelismo entre líneas, ratios de áreas, o la situación de la línea en el infinito. Las reconstrucciones similares, adicionalmente, preservan ya distancias relativas y ángulos absolutos. Finalmente, en el nivel superior de la jerarquía, las reconstrucciones euclídeas logran obtener una escena cuya forma y medidas coinciden completamente con las reales, y son capaces de determinar la situación absoluta de las cámaras que tomaron las imágenes sin ningún tipo de ambigüedad.

En general, es poco habitual encontrar métodos que trabajen con este último tipo de reconstrucciones. Ello es debido a que, como es bien conocido, si no se dispone de algún otro dato externo (como el tamaño real de algún objeto, o la magnitud real del movimiento experimentado por la cámara), es imposible determinar la escala global de la imagen reconstruida. Muchas veces, por tanto, el límite máximo que se puede alcanzar en la jerarquía anteriormente comentada es la realización de reconstrucciones similares. Sin embargo, una reconstrucción similar puede no ser suficiente para un agente físico que interactúa con su entorno puesto que, en definitiva, éste necesitará antes o después trabajar con medidas reales si pretende moverse en el mismo con seguridad. Aunque, como veremos, es cierto que pueden realizarse ciertas tareas de navegación utilizando sólo propiedades similares o incluso afines (por ejemplo, navegar centrado a lo largo de un pasillo suficientemente ancho), está claro que la obtención de la escala global de la escena ofrece mayores ventajas dado que, en último término, el lazo de control del robot se cierra en términos de órdenes de movimiento absolutas, en coordenadas de mundo reales.

Haciendo uso de un simple par de imágenes, es perfectamente conocido que partiendo sólo de un número de puntos correspondientes entre las mismas puede practicarse una reconstrucción proyectiva de la escena, aún sin tener información previa de ningún tipo sobre su estructura o la calibración de las cámaras con las que fue tomada (Hartley *et al.*, 1992; Faugeras, 1992). Un modo de eliminar la ambigüedad de la reconstrucción proyectiva, y elevarla hasta una similar es trabajar con cámaras calibradas con anterioridad. En una aproximación

[1]Muchos autores denominan reconstrucciones *métricas* a aquellas que se diferencian de la escena real sólo en la escala global de la misma. Nosotros utilizaremos el término *similar* con este mismo sentido para recordar esta ambigüedad, evitando el término métrico, y dejando el de *euclídeo* para aquella reconstrucción que también obtiene el tamaño real de la escena y su situación con respecto al origen de coordenadas de interés, en nuestro caso el relativo al propio robot.

más atractiva, sin embargo, puede pensarse en determinar también de modo automatizado las características internas de las cámaras a partir de múltiples vistas, siempre utilizando sólo correspondencias de puntos o líneas entre imágenes. Este enfoque, denominado de autocalibración (Faugeras *et al.*, 1992), es una de las líneas de investigación más activas en este campo, puesto que ofrece una flexibilidad y autonomía mucho mayor que el enfoque precalibrado. Tanto es así que se han desarrollado técnicas, incluso, que resuelven el caso en que la cámara también varía algunos de sus parámetros intrínsecos durante la secuencia.

La autocalibración es de gran interés en los sistemas autónomos, puesto que ofrece la posibilidad de determinar las características de los propios sensores a través de la interacción con el medio. En este capítulo nos centraremos en el problema de calibrar una cámara montada a bordo de un robot, tanto en sus parámetros intrínsecos (dependientes de la cámara) como extrínsecos (situación relativa entre la cámara y la plataforma móvil). Quizá la ventaja más importante la introducirá el hecho de que, en nuestro caso, podamos controlar y conocer los movimientos realizados por el robot. Aunque, como veremos, en general esto no implica conocer el movimiento concreto realizado por la cámara, esta información extravisual será suficiente para lograr reconstrucciones euclídeas del entorno, eliminando todo factor de ambigüedad de escala. Esta última propiedad es muy deseable desde el punto de vista de la navegación, y será una característica central en la arquitectura de percepción propuesta.

3.2. Cámaras en vehículos móviles

En esta sección describiremos el modelo de cámara utilizado, considerando no sólo el clásico proceso proyectivo de formación de imagen, sino haciendo también explícita la posición relativa de la cámara en un vehículo móvil sobre el que podemos controlar el movimiento. Presentaremos, pues, en primer lugar una cámara general, que transforma coordenadas de objetos del mundo (en medidas reales) a coordenadas de imagen (en píxeles). Completamos entonces este esquema ampliándolo con la consideración de que la cámara está fijada en el sistema móvil de modo arbitrario, esto es, en posición general respecto a éste. Por último, estudiaremos también una serie de simplificaciones sucesivas que, en algunos casos, facilitarán las tareas de calibración y reconstrucción, discutiendo su aplicabilidad y viabilidad en distintas situaciones.

3.2.1. Modelo de cámara general

El modelo más ampliamente utilizado en geometría de múltiples vistas es la denominada cámara de tipo *pinhole*. Los elementos fundamentales de este modelo son el *plano de imagen* y el *centro de proyección*, o *centro óptico*, un punto situado a una distancia f de aquél, llamada *distancia focal*. Cualquier punto tridimensional $\boldsymbol{X}$ se proyecta entonces a la intersección $\boldsymbol{x}$ del plano de imagen con el rayo que une el centro de proyección con dicho punto. La línea per-

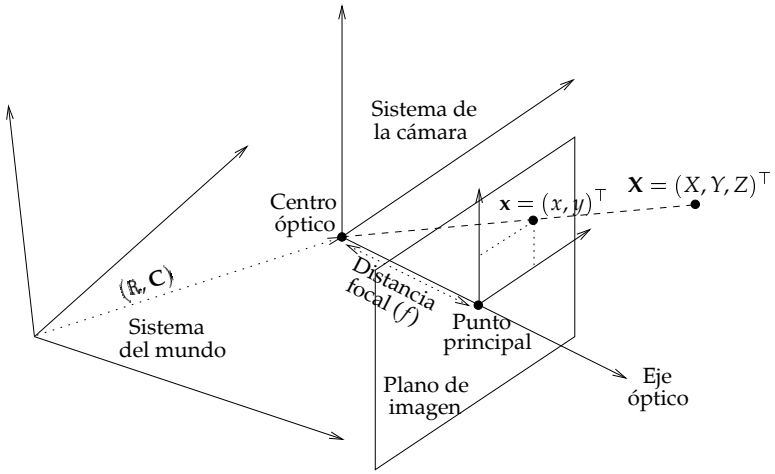


Figura 3.1: *Modelo de cámara tipo* pinhole.

pendicular al plano que pasa por el centro óptico se denomina *eje óptico*, y el correspondiente punto de corte es el llamado *punto principal*. En principio, los sistemas de coordenadas de la cámara y el mundo no tienen por qué coincidir, sino que puede haber un cierto desfase rotacional $\mathtt{R}$ y otro traslacional $\boldsymbol{C}$ entre ambos. El vector $\boldsymbol{C}_{3\times 1}$ representa la posición del centro óptico y $\mathtt{R}_{3\times 3}$ es la matriz rotación relativa entre ambos sistemas. Un esquema con todos estos elementos se muestra en la figura 3.1.

Algebraicamente, las relaciones entre las coordenadas de cámara $(X, Y, Z)^\top$ de $\boldsymbol{X}$ y las de imagen $(x, y)^\top$ de la correspondiente proyección $\boldsymbol{x}$ quedan recogidas por la siguiente ecuación:

$$(x, y)^\top = (f\frac{X}{Z}, f\frac{Y}{Z})^\top \tag{3.1}$$

La transformación entre las posiciones relativas al mundo y la cámara, por otro lado, realiza a través de la siguiente expresión (donde los subíndices *cam* y *wrl* indican los respectivos sistemas de coordenadas):

$$\boldsymbol{X}_{cam} = \mathtt{R} \cdot (\boldsymbol{X}_{wrl} - \boldsymbol{C}_{wrl}) \tag{3.2}$$

Por conveniencia, sin embargo, los vectores del modelo se suelen representar en forma homogénea, añadiendo una componente adicional. Para ello, dadas unas coordenadas tridimensionales $(X, Y, Z)^\top$ de un punto del espacio, se construye el vector homogéneo correspondiente $(X, Y, Z, 1)^\top$. La homogeneidad significa que la escala global del vector no importa, esto es, un vector $(X, Y, Z, W)^\top$ se considera equivalente a otro $(kX, kY, kZ, kW)^\top$, para cualquier $k \neq 0$. Esta independencia del módulo hace que cualquier punto del mundo siga teniendo tres grados de libertad, correspondientes a las tres dimensiones del espacio euclídeo.

Análogamente, las coordenadas de imagen se representarán como vectores de dimensión tres, también homogéneos, y por tanto con sólo dos grados de libertad.

Esta representación tiene la ventaja de que tanto las rotaciones, traslaciones y escalados como la proyección final sobre el plano de imagen pueden expresarse de modo lineal, en un formalismo donde cada una de las transformaciones indicadas se representa mediante una matriz, y cada uno de los puntos (del mundo o de la imagen) a través un correspondiente vector homogéneo. Una ventaja añadida es que los puntos infinitamente alejados del origen en una determinada dirección pueden representarse sin tener que recurrir a ningún tipo de notación especial. Así, el punto en el plano del infinito en la dirección 3D $(X, Y, Z)^\top$ se puede modelar simplemente añadiendo una cuarta coordenada nula, para obtener el vector homogéneo $(X, Y, Z, 0)^\top$. Algo similar puede hacerse con la dirección de imagen marcada por el píxel $(x, y)^\top$, para obtener el punto en la línea del infinito correspondiente, $(x, y, 0)^\top$. En cualquiera de los casos, el único vector homogéneo no permitido es el $\mathbf{0}$, es decir, aquel en el que todas sus componentes son nulas.

Una vez escogida esta representación, la transferencia entre puntos del mundo y la imagen puede modelarse usando una simple transformación proyectiva, que se representa algebraicamente mediante una matriz $\mathtt{P}_{3\times4}$. La imagen $\boldsymbol{x}_{3\times1}$ de un punto del mundo $\boldsymbol{X}_{4\times1}$ se obtiene a través del siguiente producto matriz-vector:

$$\boldsymbol{x} = \mathtt{P}\boldsymbol{X} \tag{3.3}$$

La matriz $\mathtt{P}$ recoge toda la información de posición relativa entre la cámara y el mundo y, en general, también de la posterior transformación de coordenadas de cámara a coordenadas de píxeles. Así, suele trabajarse con la siguiente descomposición de $\mathtt{P}$:

$$\mathtt{P} = \mathtt{K}\mathtt{R}[\mathtt{I}|-\boldsymbol{C}] \tag{3.4}$$

En esta expresión, $\boldsymbol{C}$ y $\mathtt{R}$ están definidos como antes, $\mathtt{I}_{3\times3}$ es la identidad y $\mathtt{K}_{3\times3}$ representa la matriz de parámetros intrínsecos de la cámara. Ésta última modela la transformación afín final experimentada por los píxeles una vez intersectados los rayos ópticos con el plano de imagen. Más en detalle, esta matriz tiene la siguiente forma:

$$\mathtt{K} = \begin{pmatrix} f_x & s & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.5}$$

Los parámetros f_x y f_y son las focales efectivas (en unidades de píxel) en las direcciones horizontal y vertical de la imagen, respectivamente. Estos valores son distintos para modelar un tamaño de píxel general, no necesariamente cuadrado. El llamado sesgo s (*skew*) modela la posible inclinación entre los ejes X e Y (habitualmente leve o despreciable). Finalmente, $(o_x, o_y)^\top$ son las coordenadas de imagen del punto principal de la cámara, es decir, el lugar

del plano de imagen sobre el que cae en perpendicular el centro óptico de la cámara.

Aunque tiene un total de doce elementos, P puede determinarse en general con once grados de libertad, dada la condición de homogeneidad. Estos grados se corresponden con los cinco parámetros intrínsecos de la matriz K, más los tres correspondientes al vector de posición $\boldsymbol{C}$, y otros tres de la matriz de rotación R. Estos últimos seis parámetros son los llamados extrínsecos, puesto que dependen únicamente de la posición y orientación de la cámara en la escena, y no de su configuración interna.

Bajo la representación homogénea, como ya se ha dicho, las ecuaciones de proyección resultantes para el modelo *pinhole* son lineales. Sin embargo, las cámaras con lentes reales pueden introducir algunos tipos de distorsión que no pueden recogerse dentro de este marco de linealidad. Si esta distorsión es importante (lo que suele ocurrir sobre todo cuando se trabaja con distancias focales cortas, con ángulos de visión mayores), puede ser necesario tomarla también en consideración. En ese caso, es habitual trabajar con las llamadas distorsiones *radial* y *tangencial*. La primera de ellas acerca o aleja los puntos de imagen $(x, y)^\top$ del llamado *centro de distorsión radial* $(r_x, r_y)^\top$ de modo proporcional al cuadrado de la distancia a éste (el radio de la circunferencia centrada en $(r_x, r_y)^\top$ que pasa por el punto $(x, y)^\top$). Dicho centro puede no coincidir con el centro óptico $(o_x, o_y)^\top$. La segunda deformación traslada la posición del punto en dirección perpendicular a dicho radio (es decir, tangente a la anterior circunferencia). Ambas deformaciones tienen constantes de proporcionalidad que varían de unas cámaras a otras, y son parámetros intrínsecos adicionales que también tienen que ser estimados durante el proceso de calibración. A veces, en modelos que tratan de ser más precisos, esta deformación se expresa en forma de polinomios de mayor orden, donde cada factor tiene una constante distinta, todas las cuales tienen que ser calibradas.

La no linealidad introducida en presencia de estas aberraciones hace que las líneas rectas de la escena no aparezcan como tales en la imagen. Afortunadamente, una vez conocidos los parámetros de la distorsión, es sencillo deshacer ésta inmediatamente sobre las características extraídas, de forma que tras esta primera rectificación de imagen puede continuarse el proceso bajo la hipótesis del modelo lineal propuesto. En el resto de esta tesis trabajaremos directamente sobre éste último, suponiendo que la distorsión es despreciable (lo cual es una aproximación aceptable para muchas cámaras) o bien que, en cualquier caso, de ser significativa sería corregida con anterioridad.

3.2.2. Modelo de cámara rígidamente acoplada a un móvil

Para el problema que nos ocupa resulta interesante hacer explícita la posición de la cámara respecto al sistema móvil, dado que es en éste último en el que podemos dar las órdenes de movimiento y tomar los valores de odometría. Supondremos un vehículo holonómico, esto es, que puede girar sobre sí mismo sin necesidad de trasladarse[2]. Esto significa que las

[2]La plataforma robótica utilizada como base de la arquitectura de percepción propuesta (un Pioneer 2 DX, de ActivMedia Robotics, cuyas características describiremos en mayor profundidad en el capítulo 5) posee esta

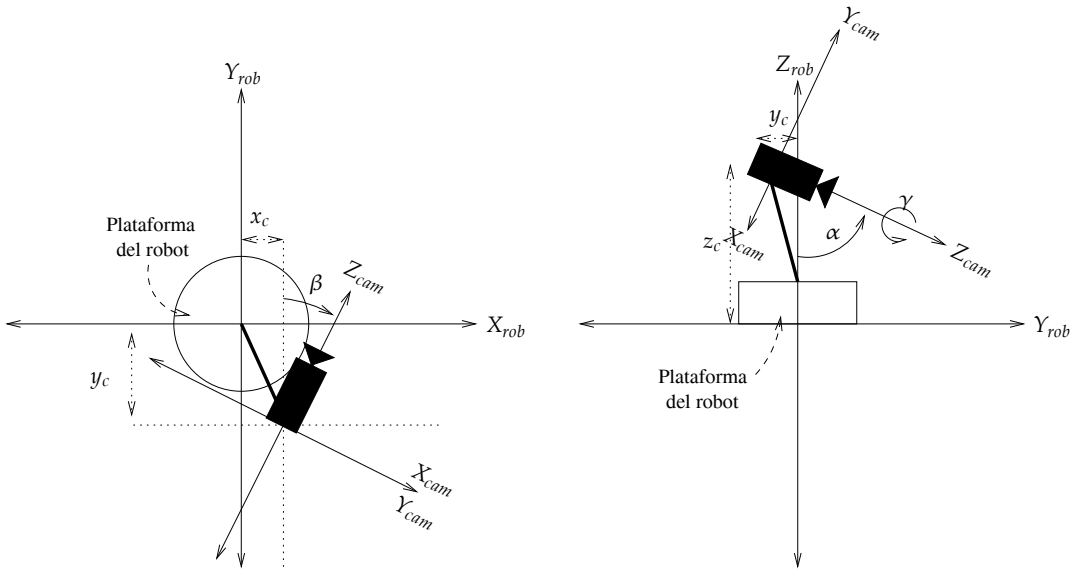


Figura 3.2: *Vistas superior y lateral de la situación relativa de la cámara respecto al robot. Obsérvense los ángulos de desfase α, β y γ entre los sistemas de cámara y de robot, así como la traslación relativa entre orígenes $(x_c, y_c, z_c)^\top$.*

órdenes de traslación sobre el plano del suelo, $(t_x, t_y)^\top$ y giro θ pueden proporcionarse por separado. En la figura 3.2 se muestra el esquema de posición general de cámara en un robot de estas características. El vector $(x_c, y_c, z_c)^\top$ es la posición del centro óptico de la cámara en el sistema de coordenadas de movimiento del robot, S_{rob}, con ejes X_{rob} (apuntando a la derecha de la dirección de avance sobre el suelo), Y_{rob} (apuntando hacia adelante) y Z_{rob} (en perpendicular al suelo, positivo para alturas sobre éste). El sistema de la cámara es S_{cam}, con ejes X_{cam}, Y_{cam} y Z_{cam}. Los tres grados de libertad de la rotación se modelarán a través de los ángulos α (inclinación de la cámara sobre el suelo, o *tilt*), β (rumbo relativo en el plano entre los sistemas S_{cam} y S_{rob}, o *pan*) y γ (giro de la cámara en torno a su eje óptico, o *roll*). El ángulo de *tilt* es cero para una cámara apuntando hacia el suelo, y crece al empezar a mirar delante del robot. El de *pan* crece en sentido antihorario sobre el suelo. El de *roll* es nulo para horizonte horizontal, y crece también en sentido antihorario mirando hacia la escena. En estas condiciones, la matriz $\mathtt{R}$ y el vector $\boldsymbol{C}$ de la ecuación 3.4 quedan como sigue:

$$\mathtt{R} = \begin{pmatrix} -\cos(\beta)\cos(\gamma) + \cos(\alpha)\sin(\beta)\sin(\gamma) & -\cos(\gamma)\sin(\beta) - \cos(\alpha)\cos(\beta)\sin(\gamma) & -\sin(\alpha)\sin(\gamma) \\ \cos(\alpha)\cos(\gamma)\sin(\beta) + \cos(\beta)\sin(\gamma) & -\cos(\alpha)\cos(\beta)\cos(\gamma) + \sin(\beta)\sin(\gamma) & -\cos(\gamma)\sin(\alpha) \\ \sin(\alpha)\sin(\beta) & -\cos(\beta)\sin(\alpha) & \cos(\alpha) \end{pmatrix} \tag{3.6}$$

$$\boldsymbol{C} = (x_c, y_c, z_c)^\top \tag{3.7}$$

particularidad.

Se puede comprobar que R es una rotación bien definida, con determinante unidad, y consistente con la definición de sistemas de coordenadas de robot y cámara bien orientados, en los que las direcciones de los ejes cumplen que $X \times Y = Z$ (*right-handed*).

3.2.3. Modelos simplificados

En ocasiones, puede ser aceptable realizar algunas restricciones sobre el modelo descrito, con el fin de facilitar los métodos de calibración o reconstrucción que sobre él quieran aplicarse posteriormente. En este apartado describimos algunas de estas simplificaciones, clasificándolas según trabajen sobre los parámetros intrínsecos o extrínsecos de la cámara, y discutiendo sobre las distintas condiciones de aplicabilidad de cada una.

Simplificaciones de cámara

El problema de realizar reconstrucciones a partir de imágenes en movimiento es esencialmente sensible al ruido. Algunos autores han enfatizado el problema de la baja resolución de las cámaras de computador, frente a las empleadas en fotogrametría, de mayor precisión (Daniilidis y Spetsakis, 1997). En estos últimos dispositivos puede tener más sentido realizar calibraciones exhaustivas. Sin embargo, en muchos ámbitos de aplicación de la visión artificial donde cierto tipo de errores pueden considerarse admisibles, es viable realizar ciertas suposiciones sobre las características de la cámara utilizada que se traducen en condiciones sobre la matriz de parámetros intrínsecos K (ecuación 3.5). La razón es que la deformación afín introducida por dicha matriz es quizá demasiado general, puesto que somete a consideración incluso la perpendicularidad de los ejes de coordenadas de la cámara.

Esta no perpendicularidad es prácticamente despreciable en casi todas las cámaras comerciales, así que en la mayoría de ocasiones podemos anular el parámetro del sesgo s, haciéndolo igual a cero. En general, muchos métodos de autocalibración realizan esta suposición como restricción adicional para disminuir en uno los cinco grados de libertad iniciales de K.

Más controvertida es la suposición de que el punto principal de la cámara está en el centro de la imagen, esto es, $o_x = o_y = 0$. Mientras que algunos autores reconocen que las reconstrucciones practicadas sobre esta simplificación son muy poco sensibles a la elección final de este punto (Bougnoux, 1998; Faugeras, 2001), otros opinan que, si bien la suposición de *skew* nulo es perfectamente aceptable, no lo es tanto la de punto principal "nominal", o fijado arbitrariamente en el centro de la imagen (Hartley y Zisserman, 2000). Recientemente, nosotros también hemos realizado algún estudio en este sentido (Ruiz *et al.*, 2002). En él se ilustra la dificultad teórica en el cálculo preciso de dicha posición, pero reconociendo el hecho de que en ciertas aplicaciones de geometría visual en entornos ruidosos (como la robótica móvil que nos ocupa, por ejemplo) su localización exacta es de poca relevancia en la práctica. La principal ventaja de este modelo es que la matriz de calibración queda diagonal ($\mathtt{K} = \mathrm{diag}(f_x, f_y, 1)$),

siempre que mantengamos que $s = 0$) [3].

Finalmente, puede resultar también razonable considerar que los píxeles que forman la imagen son perfectamente cuadrados. En estas condiciones, se puede considerar una focal f única, de modo que $f_x = f_y = f$. En cámaras de calidad esta propiedad suele cumplirse de modo muy aproximado, y en cualquier caso, si se calibra el ratio f_x/f_y con anterioridad, también puede realizarse la corrección adecuada en la etapa de adquisición, para llegar a la matriz de intrínsecos más simple posible, con un solo grado de libertad:

$$\mathtt{K} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.8}$$

Esta condición puede simplificar enormemente algunas técnicas de autocalibración (véase, por ejemplo, (Xu *et al.*, 2000)). Este modelo restringido también ha sido empleado como método de reconocimiento de figuras planas en perspectiva a través de cierto tipo de invariantes (Pizlo y Rosenfeld, 1992).

No obstante lo dicho, algunos autores insisten en defender la práctica de la geometría proyectiva sin restricciones de calibración, frente a los modelos de perspectiva reducidos (Gros *et al.*, 1997). El argumento principal es que no hay invariantes de utilidad en éstos últimos que no lo sean también en aquélla. También se defiende que algunos de los logros más importantes de la geometría proyectiva, como la visión tridimensional con cámaras no calibradas (aunque con ambigüedad proyectiva), o el cálculo sencillo de la geometría epipolar (sólo a partir de correspondencias de puntos, a través del cálculo de la fundamental), se han logrado teorizando sobre el modelo de cámara general. Muchos de estos avances han sido posibles gracias a la linealidad de las soluciones resultantes al generalizar la estructura de la $\mathtt{K}$, frente a las no linealidades impuestas a menudo por los modelos restringidos.

Como conclusión a este apartado diremos que, en general, el grado de aceptabilidad de cada tipo de restricción depende del ámbito concreto de aplicación y de la precisión requerida, así como la posible inviabilidad en la resolución del problema si no se realiza alguna simplificación de las comentadas.

Simplificaciones de posición relativa al vehículo

Veamos ahora las posibles simplificaciones sobre los parámetros extrínsecos que pueden resultar de interés. La primera es la que llamaremos hipótesis de *brazo nulo*. Esta hipótesis supone que $x_c = y_c = 0$, es decir, que la cámara no sólo está rígidamente unida al vehículo, sino que además experimenta una traslación de módulo igual a la que mide el robot, es decir,

[3] Si el punto principal no se encuentra en el centro de la imagen, pero se conoce *a priori* su situación en la misma (por ejemplo, a través de una calibración previa), aún pueden transformarse las coordenadas de píxel de las características medidas en la imagen mediante una simple traslación para conseguir la condición de diagonalidad de la matriz $\mathtt{K}$.

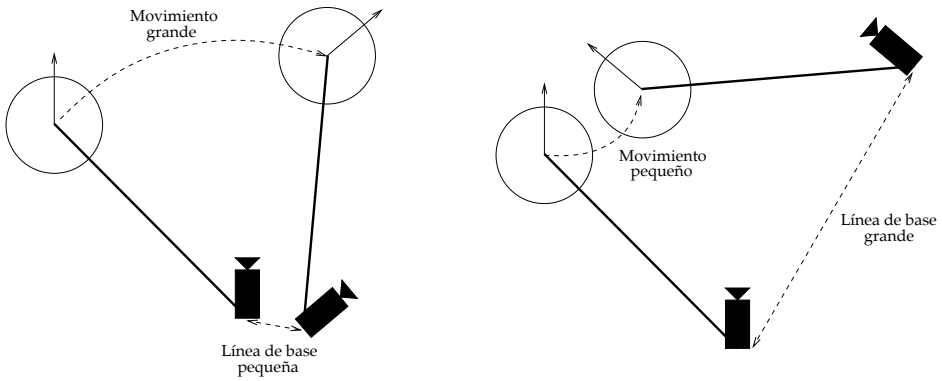


Figura 3.3: *Traslación parásita asociada a una cámara montada sobre un robot con brazo no nulo. a) Un movimiento largo correspondiente a una línea de base corta. b) Caso contrario.*

$(t_x^2 + t_y^2)^{1/2}$. Esto es debido a que no existe ninguna traslación "parásita" como resultado del giro θ, al encontrarse el centro óptico sobre la vertical del origen del sistema S_{rob}, a una determinada altura $z_c \neq 0$. Esta restricción tiene una gran importancia, puesto que en este caso tener acceso a la odometría del robot supone también conocer la línea de base (*baseline*) exacta entre las cámaras, lo que facilitará enormemente algunos de los métodos de calibración. La figura 3.3 ilustra cómo esto último no tiene por qué ser cierto en el caso general $x_c \neq 0$ ó $y_c \neq 0$.

En segundo lugar, puede también considerarse que la cámara se monta sin ángulo de *roll*, o sea, con $\gamma = 0$. En estas condiciones, la línea del infinito del plano del suelo aparecerá siempre en la imagen como una línea horizontal, de coordenadas homogéneas[4] $\boldsymbol{l}_h = (0, 1, -y_h)^\top$, siendo y_h la altura del horizonte en la imagen, en píxeles.

Finalmente, un tercer caso que merece la pena estudiar es el de ángulo de *pan* β también nulo. Ello equivaldría a colocar la cámara mirando en la dirección de avance del robot, sin desfase apreciable entre la dirección de avance de éste y el eje óptico de aquélla.

Obsérvese que en lo que respecta al giro odométrico θ, incluso en la situación de posición más general, con brazo no nulo y ángulo de *pan* no despreciable, siempre coincide con el experimentado por la cámara, por la condición de rigidez del sistema completo plataforma-cámara.

[4]Dichas coordenadas cumplen la ecuación de línea $\boldsymbol{l}_h^\top \boldsymbol{x} = 0$ para todo píxel $\boldsymbol{x}$ de coordenadas homogéneas $(x, y_h, 1)^\top$.

3.3. Técnicas de calibración

En este apartado haremos una breve revisión de los diversos métodos de calibración de cámaras propuestos en la literatura. Tal y como hicimos en el capítulo anterior, haremos primero mención a las técnicas de aplicación más generales, tanto de calibración con plantillas como de autocalibración, para pasar después a tratar el ámbito más específico de los sistemas autónomos dotados de cámaras. De nuevo, la revisión no persigue tanto la exhaustividad como el dar una visión general de las principales herramientas disponibles, a costa de una forzosa incompletitud.

3.3.1. Métodos clásicos

En los primeros trabajos sobre el tema la calibración de cámaras se considera un problema previo a la adquisición de las imágenes de interés. En una fase anterior a la puesta en funcionamiento del sistema de visión, pues, se practica la calibración de la cámara utilizando una plantilla u objeto real de medidas perfectamente conocidas. Este objeto, puesto delante de la cámara en una o varias posiciones relativas distintas, sirve para determinar las características intrínsecas de la cámara (y, subsidiariamente, también las extrínsecas respecto a la plantilla de calibración). A menudo, las técnicas incluyen también la estimación de los coeficientes de distorsión radial o incluso tangencial. Una vez determinados todos estos parámetros, la cámara estaría lista para ser utilizada en tareas de reconstrucción.

Quizá uno de los más famosos sea el método de Tsai (1987). Este método comienza solucionando primero la ecuación $\boldsymbol{x}_i = \mathtt{P}\boldsymbol{X}_i$, con los vectores $\boldsymbol{X}_i$ (coordenadas reales de la plantilla) y $\boldsymbol{x}_i$ (coordenadas de imagen correspondientes) conocidos, y con los elementos de la matriz P desconocidos. Se obtiene así una primera aproximación a los parámetros intrínsecos usando simplemente un método lineal (Faugeras y Toscani, 1986). Después se aplica una técnica de optimización no lineal iterativa para todos los parámetros, utilizando como inicio la solución anterior, y como criterio de optimización el error en el plano de imagen. Esto último tiene ciertas ventajas sobre la minimización de otro tipo de errores algebraicos, más sencillos de cara a la optimización pero menos justificados teóricamente. El método no considera el posible *skew* de la cámara, pero sí la distorsión radial. En la misma línea se halla el trabajo de Weng *et al.* (1992), donde se amplía el trabajo anterior para tratar algunos tipos de distorsión adicionales, como la tangencial y la de prisma grueso.

Hay que recordar que el objetivo final de la calibración no deja de ser la realización de reconstrucciones (más allá de las meramente proyectivas). Es por ello que, a veces, puede que estemos más interesados en el uso final de las transformaciones (para reproyectar puntos, por ejemplo) que en la estimación individual de cada uno de los parámetros. En este caso hablaremos de *calibración implícita* (Wei y Ma, 1994), como opuesto a la *explícita*, que busca determinar los intrínsecos uno a uno. En lo que se refiere al modelo lineal propuesto en la ecuación 3.4,

no obstante, es muy sencillo obtener una matriz K de calibración consistente con una matriz P obtenida, mediante la simple descomposición RQ de las tres primeras columnas[5].

Como ya se ha dicho, el tratamiento de las distorsiones radial y tangencial fuerza el uso de métodos no lineales. Algunos trabajos se ocupan específicamente de determinar estos parámetros, separándolos del resto del problema de estimar la matriz K. Devernay y Faugeras (1995), por ejemplo, proponen un método para calcular la distorsión radial a partir de imágenes que contengan segmentos en la escena, en distintas orientaciones. El tipo de primitiva empleada lo hace útil para entornos estructurados como los que nos ocupan, por ejemplo. Se elimina de esta forma la necesidad del patrón de calibración. La mayoría de estas técnicas parten de una solución inicial aproximada, algunas veces usando cierto tipo de información externa, otras escogiendo una arbitraria, para iniciar entonces un proceso de minimización iterativo sobre una determinada función de coste. En el caso descrito la función objetivo es una suma de distancias punto a punto entre cada curva detectada, candidata a ser "enderezada", y su correspondiente segmento ideal. A este tipo de procedimientos se les conoce con el nombre de *métodos de la plomada* (del inglés *plumb-line*). En el artículo mencionado, en concreto, se desprecia la distorsión tangencial, y se estiman únicamente el *aspect ratio* (f_x/f_y), el primer término de la distorsión radial y el centro de la misma. Otro tipo de solución es la presentada por Muñoz y Baumela (2002), que utilizan una propiedad de la geometría proyectiva que restringe la posición del centro de distorsión a una elipse, determinada en función de tuplas de puntos colineales cuyo cross-ratio es conocido *a priori*. A través de varias de estas tuplas, el método estima el centro de distorsión, por intersección de varias de estas elipses. Posteriormente, los coeficientes de distorsión radial se estiman también a través de un método de tipo *plumb-line*. Este artículo contiene también varias referencias adicionales interesantes relacionadas con el tema.

3.3.2. Autocalibración

En un intento de ganar en versatilidad, desde hace unos años gran parte del énfasis en la investigación se ha puesto en la posibilidad de trabajar con cámaras no calibradas, tratando de eliminar por otros medios la ambigüedad proyectiva de las reconstrucciones. En esta otra aproximación se intentan deducir las características de las cámaras a partir de la propia secuencia de imágenes, sin necesidad de plantillas o cualquier otro tipo de información externa. Ésta es la base del llamado enfoque autocalibrado. Los primeros trabajos sobre el tema se remontan a hace una década, aproximadamente (Faugeras *et al.*, 1992).

En general, hace falta introducir algún tipo de restricciones sobre las matrices de calibración $\mathtt{K}^i$ en las distintas imágenes, para poder determinar una solución única. Una de las más empleadas es suponer la K constante a lo largo de la secuencia. Esto último se ajusta perfec-

[5] La descomposición RQ de una matriz cuadrada cualquiera A obtiene una expresión de ésta en la forma $\mathtt{A} = \mathtt{RQ}$, siendo R una matriz triangular superior y Q una de rotación. Esta descomposición es única para cada matriz A no singular (Golub y Van Loan, 1989).

tamente a secuencias tomadas con una única cámara que no varía sus intrínsecos a lo largo del tiempo. Pero también se pueden resolver casos más complejos, donde sólo se imponen restricciones tales como píxel cuadrado, ausencia de *skew* o punto principal conocido, entre otras. En general, pues, se pueden resolver casos bastante interesantes, como secuencias en las que la focal puede cambiar, algo perfectamente habitual en determinadas aplicaciones.

Básicamente, el problema de la autocalibración consiste en crear una reconstrucción similar partiendo de una proyectiva, dado que ésta última puede ser obtenida a partir de la secuencia no calibrada. A partir de un conjunto de coordenadas de imagen $\boldsymbol{x}_j^i$ (donde el superíndice i denota el orden en la secuencia de imágenes, y el subíndice j a los distintos puntos dentro de cada una), con ambigüedad proyectiva sólo se recuperan una serie de matrices de cámara $\mathtt{A}^i$ y de puntos $\boldsymbol{X}_j^A$ tales que $\boldsymbol{x}_j^i = \mathtt{A}^i \boldsymbol{X}_j^A$, ambos a falta de una homografía de rectificación $\mathtt{H}$ desconocida, de tamaño 4×4. Las verdaderas matrices de proyección buscadas son, por tanto, $\mathtt{P}^i = \mathtt{A}^i\mathtt{H}$, correspondientes ya a la reconstrucción similar. Es obvio que a partir de ésta $\mathtt{H}$ pueden también obtenerse las coordenadas similares $\boldsymbol{X}_j$ de los puntos de la reconstrucción, puesto que $\boldsymbol{x}_j^i = \mathtt{P}^i \boldsymbol{X}_j = \mathtt{A}^i\mathtt{H}\mathtt{H}^{-1}\boldsymbol{X}_j^A$, de forma que premultiplicando los puntos $\boldsymbol{X}_j^A$ por $\mathtt{H}^{-1}$ se obtiene la reconstrucción similar buscada.

A continuación se ofrece una visión general de las soluciones para calcular $\mathtt{H}$ propuestas más recientemente en la literatura. Clasificaremos las técnicas en tres grandes grupos, según la herramienta geométrica empleada. Un buen estudio en profundidad del problema, en el que se exploran también las conexiones entre las distintas alternativas, puede encontrarse, una vez más, en el imprescindible trabajo de Hartley y Zisserman (2000).

Uso de la cuádrica dual absoluta

La cuádrica dual absoluta, $\mathtt{Q}_\infty^*$, es un objeto geométrico invariante a transformaciones similares. Se trata de una cuádrica tridimensional degenerada, llamada dual por estar definida en términos del conjunto de planos tangentes a ella en el espacio (Triggs, 1997). Las coordenadas homogéneas $\boldsymbol{p}$ de estos planos quedan determinadas por la ecuación $\boldsymbol{p}^\top \mathtt{Q}_\infty^* \boldsymbol{p} = 0$. Algebraicamente, pues, se trata de una matriz simétrica de 16 elementos (tamaño 4×4), homogénea, que tiene sólo rango 3, por su condición de degenerada. La característica más interesante de esta matriz, que la hará útil en la eliminación de la ambigüedad proyectiva, es que de alguna manera codifica el plano 3D del infinito, $\boldsymbol{\pi}_\infty$, dado que las coordenadas de éste se corresponden con el espacio nulo de $\mathtt{Q}_\infty^*$. Esto último se puede ver fácilmente si se examina la forma canónica de esta cuádrica:

$$\mathtt{Q}_\infty^* = \tilde{\mathtt{I}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.9}$$

El espacio nulo de esta matriz está dado por el vector $\boldsymbol{p} = (0,0,0,1)^\top$, que son precisamente las coordenadas homogéneas del plano en el infinito en un marco euclídeo, puesto que todos los puntos 3D del infinito, $\boldsymbol{X} = (X,Y,Z,0)^\top$ cumplen que $\boldsymbol{p}^\top \boldsymbol{X} = 0$.

Una segunda propiedad de esta cuádrica es que su silueta se proyecta a la imagen de la dual de la cónica absoluta, $\boldsymbol{\omega}^* = \mathtt{K}\dot{\mathtt{K}}^\top$, otra entidad bien conocida en la geometría de múltiples vistas. La utilidad principal de esta cónica es que, como se observa, depende sólo de la matriz de calibración $\mathtt{K}$ (es invariante, por tanto, a cualquier tipo de rotación y traslación), y ésta última puede obtenerse de aquélla mediante la descomposición de Cholesky[6]. De la forma de $\tilde{\mathtt{I}}$ se puede comprobar fácilmente que los planos pertenecientes a la cuádrica tienen coordenadas complejas, puesto que $\boldsymbol{p}^\top \tilde{\mathtt{I}} \boldsymbol{p} = 0$ implica que $p_1^2 + p_2^2 + p_3^2 = 0$. Así pues, la cuádrica dual absoluta tiene una difícil interpretación desde el punto de vista geométrico. Su utilidad en la calibración viene más bien de la simplificación del problema que introduce su adecuada manipulación algebraica. Usando la forma de $\mathtt{P}$ expresada en 3.4, se tiene que, la expresión $\mathtt{P}\tilde{\mathtt{I}}\mathtt{P}^\top$ (proyección de $\tilde{\mathtt{I}}$) queda reducida a $\mathtt{K}\mathtt{K}^\top = \boldsymbol{\omega}^*$, por la anulación del vector $-\mathtt{R}\boldsymbol{C}$ y la cancelación de la rotación en la matriz de cámara:

$$\mathtt{P}\tilde{\mathtt{I}}\mathtt{P}^\top = \mathtt{K}[\mathtt{R}|-\mathtt{R}\boldsymbol{C}]\tilde{\mathtt{I}}\begin{pmatrix}\mathtt{R}^\top \\ -(\mathtt{R}\boldsymbol{C})^\top\end{pmatrix}\mathtt{K}^\top = \mathtt{K}\mathtt{R}\mathtt{R}^\top\mathtt{K}^\top = \mathtt{K}\mathtt{K}^\top = \boldsymbol{\omega}^* \tag{3.10}$$

Puesto que con una ambigüedad proyectiva sólo disponemos de las matrices de cámara $\mathtt{A}$ a falta de la rectificación desconocida $\mathtt{H}$, la igualdad homogénea anterior queda:

$$\mathtt{A}\mathtt{H}\tilde{\mathtt{I}}\mathtt{H}^\top\mathtt{A}^\top = \boldsymbol{\omega}^* \tag{3.11}$$

Haciendo $\mathtt{Q}_\infty^* = \mathtt{H}\tilde{\mathtt{I}}\mathtt{H}^\top$, la igualdad 3.11 liga los elementos de $\mathtt{Q}_\infty^*$ con los de $\boldsymbol{\omega}^*$ a través de las matrices conocidas $\mathtt{A}^i$.

Para obtener entonces $\mathtt{Q}_\infty^*$ de forma lineal, se hacen suposiciones sobre la forma de $\mathtt{K}$, como píxel cuadrado, o punto principal nulo o conocido, por ejemplo. Estas suposiciones provocan la anulación de ciertos elementos de $\boldsymbol{\omega}^*$, o que algunos queden iguales entre ellos, etc. Esto se traduce en restricciones sobre $\mathtt{Q}_\infty^*$, de modo que dadas suficientes vistas (dependiendo del número y tipo de las restricciones comentadas), pueden determinarse los 16 elementos de la matriz, como siempre salvo escala. Haciendo otro tipo de suposiciones, basadas en la invariabilidad de $\mathtt{K}$ a lo largo de la secuencia, pueden obtenerse también ecuaciones sobre los elementos de $\mathtt{Q}_\infty^*$. A veces es interesante disminuir los grados de libertad de las ecuaciones, para poder resolverlas utilizando menos vistas, por ejemplo. En ese caso, siempre puede añadirse también una restricción no lineal adicional haciendo uso del hecho de que $\det(\mathtt{Q}_\infty^*) = 0$, condición directamente impuesta por el déficit de rango de la matriz.

Una vez calculada $\mathtt{Q}_\infty^*$, puede obtenerse su descomposición en valores propios[7], e, intro-

[6] La descomposición de Cholesky de una matriz $\mathtt{A}$ simétrica y semidefinida positiva expresa ésta de modo único en la forma $\mathtt{A} = \mathtt{B}\mathtt{B}^\top$, siendo $\mathtt{B}$ una matriz triangular superior (Golub y Van Loan, 1989).

[7] Para una matriz $\mathtt{A}$, simétrica y semidefinida positiva, su descomposición en valores propios tiene la forma

duciendo estos últimos (que serán tres, por el rango de $\mathtt{Q}_\infty^*$) en la $\mathtt{H}$, conseguir la forma deseada $\mathtt{Q}_\infty^* = \mathtt{H}\tilde{\mathtt{I}}\mathtt{H}^\top$. Obtenemos así finalmente la homografía tridimensional $\mathtt{H}$ que corrige las cámaras proyectivas, y la correspondiente $\mathtt{H}^{-1}$ con la que realizar la reconstrucción similar de los puntos.

Ecuaciones de Kruppa

Estas ecuaciones enlazan la restricción epipolar entre los puntos de un par de imágenes[8] con la matriz $\boldsymbol{\omega}^*$, dual de la imagen de la cónica absoluta. Para deducirlas puede partirse de la expresión para la matriz fundamental en la forma $\mathtt{F} = [\boldsymbol{e}']_\times \mathtt{K}'\mathtt{R}\mathtt{K}^{-1}$, donde $\mathtt{K}$ y $\mathtt{K}'$ son las matrices de calibración de las cámaras, $\mathtt{R}$ la rotación relativa entre ambas y $\boldsymbol{e}'$ el epipolo en la segunda imagen del par. La expresión $[\boldsymbol{x}]_\times$ denota a la matriz que, multiplicada por un vector tridimensional cualquiera $\boldsymbol{y}$, devuelve el vector $\boldsymbol{x} \times \boldsymbol{y}$. Si $\boldsymbol{x} = (x, y, z)^\top$, esta matriz tiene la siguiente forma antisimétrica:

$$[\boldsymbol{x}]_\times = \begin{pmatrix} 0 & -x & y \\ x & 0 & -z \\ -y & z & 0 \end{pmatrix} \tag{3.12}$$

Razonando de modo similar al caso anterior, por la cancelación de las matrices $\mathtt{K}$ y $\mathtt{K}^\top$ con sus inversas y la $\mathtt{R}$ con su traspuesta al operar sobre la expresión $\mathtt{F}\boldsymbol{\omega}^*\mathtt{F}^\top$, se obtiene la siguiente ecuación:

$$\mathtt{F}\boldsymbol{\omega}^*\mathtt{F}^\top = [\boldsymbol{e}']_\times \mathtt{K}'\mathtt{R}\mathtt{K}^{-1}\mathtt{K}\mathtt{K}^\top\mathtt{K}^{-\top}\mathtt{R}^\top\mathtt{K}'^\top[\boldsymbol{e}']_\times^\top = [\boldsymbol{e}']_\times \boldsymbol{\omega}^{*\prime}[\boldsymbol{e}']_\times \tag{3.13}$$

La última igualdad se deduce de la antisimetría de $[\boldsymbol{e}']_\times$, por la que $[\boldsymbol{e}']_\times = -[\boldsymbol{e}']_\times^\top$, y multiplicando por -1, lo que se puede hacer sin problemas por la condición de homogeneidad. Conocida la matriz fundamental, por tanto, esta ecuación introduce restricciones sobre $\boldsymbol{\omega}^*$ y $\boldsymbol{\omega}^{*\prime}$. Haciendo un análisis más detallado, debido al déficit de rango de la $\mathtt{F}$ en realidad la expresión contiene sólo dos restricciones independientes para dichas matrices. Estas restricciones, además, son cuadráticas por la homogeneidad de las igualdades. Haciendo la descomposición de $\mathtt{F}$ en valores singulares[9] $\mathtt{F} = \mathtt{U} \cdot \mathrm{diag}(\sigma_1, \sigma_2, 0) \cdot \mathtt{V}^\top$, y haciendo uso del hecho de que el espacio nulo izquierdo de la $\mathtt{F}$ (es decir, la tercera fila de $\mathtt{U}$, $\boldsymbol{u}_3$) es el epipolo

$\mathtt{A} = \mathtt{V} \cdot \mathrm{diag}(\sigma_1, \ldots, \sigma_m, 0, \ldots, 0) \cdot \mathtt{V}^\top$, siendo $\mathtt{V}$ una matriz ortonormal, σ_i los valores propios de la matriz original y m su rango (Golub y Van Loan, 1989). Habitualmente, esta descomposición suele realizarse de manera que los valores $\sigma_1 \ldots \sigma_m$ queden ordenados de mayor a menor (Golub y Van Loan, 1989).

[8] Esta restricción liga la posición de los puntos $\boldsymbol{x}$ de una imagen con sus homólogos en la otra $\boldsymbol{x}'$ a través de la expresión $\boldsymbol{x}^\top\mathtt{F}\boldsymbol{x}' = 0$, siendo $\mathtt{F}$ la llamada matriz fundamental, de dimensión 3×3, pero rango 2. Los vectores nulos izquierdo y derecho de esta matriz se corresponden con los epipolos $\boldsymbol{e}$ y $\boldsymbol{e}'$, imágenes de los centros ópticos de las respectivas cámaras en la imagen recíproca (Luong *et al.*, 1993; Torr y Murray, 1997).

[9] Esta descomposición se aplica sobre una matriz genérica $\mathtt{A}$, no necesariamente cuadrada, expresándola en la forma $\mathtt{A} = \mathtt{U} \cdot \mathrm{diag}(\sigma_1, \ldots, \sigma_n) \cdot \mathtt{V}^\top$, con $\mathtt{U}$ y $\mathtt{V}$ matrices ortonormales. También aquí suelen ordenarse los valores $\sigma_1 \ldots \sigma_n$ de mayor a menor (Golub y Van Loan, 1989).

$\boldsymbol{e}'$, se puede manipular algebraicamente la igualdad anterior para llegar a la expresión más cómoda y conocida de las ecuaciones de Kruppa (Hartley, 1997):

$$\frac{\boldsymbol{u}_2^\top \boldsymbol{\omega}^* \boldsymbol{u}_2}{\sigma_1^2 \boldsymbol{v}_1^\top \boldsymbol{\omega}^{*\prime} \boldsymbol{v}_1} = \frac{\boldsymbol{u}_1^\top \boldsymbol{\omega}^* \boldsymbol{u}_2}{\sigma_1 \sigma_2 \boldsymbol{v}_1^\top \boldsymbol{\omega}^{*\prime} \boldsymbol{v}_2} = \frac{\boldsymbol{u}_1^\top \boldsymbol{\omega}^* \boldsymbol{u}_1}{\sigma_2^2 \boldsymbol{v}_2^\top \boldsymbol{\omega}^{*\prime} \boldsymbol{v}_2} \tag{3.14}$$

Donde los vectores $\boldsymbol{u}_i$ y $\boldsymbol{v}_i$ son las filas de $\mathtt{U}$ y $\mathtt{V}$ respectivamente. Naturalmente, estas ecuaciones son mucho más útiles si se considera la cámara constante a lo largo del movimiento (es decir, $\boldsymbol{\omega}^* = \boldsymbol{\omega}^{*\prime}$), puesto que restringen más la solución. Si las cámaras se consideran con punto principal en el origen, y de píxel cuadrado, entonces aún permitiendo variar la focal de la primera a la segunda vista, las dos restricciones cuadráticas son suficientes para determinar las focales f y f'. Para casos más complejos, evidentemente, un solo par de imágenes no determinan la solución y hay que extender el análisis a múltiples vistas, lo que en general complica el problema.

Aproximación estratificada

Examinamos en esta sección una tercera aproximación al problema. Si el sistema de coordenadas del mundo se considera coincidente con el relativo a la primera cámara de la secuencia (es decir, $\mathtt{P}^1 = \mathtt{K}^1[\mathtt{I}\,|\,\mathbf{0}]$), entonces la matriz $\mathtt{H}$ correctora de la cámara (y de la reconstrucción, a través de su inversa $\mathtt{H}^{-1}$) tiene la siguiente forma:

$$\mathtt{H} = \begin{pmatrix} \mathtt{K}^1 & \mathbf{0} \\ -\boldsymbol{p}^\top \mathtt{K}^1 & 1 \end{pmatrix} \tag{3.15}$$

Donde $\mathtt{K}^1$ es la matriz de calibración de la primera cámara y $\boldsymbol{p}$ la localización del plano en el infinito. El cálculo de $\mathtt{H}$ puede entonces factorizarse en el cómputo separado de $\mathtt{K}$ del plano en el infinito $\boldsymbol{p}$. Este tipo de solución se denomina estratificada. El problema de la misma radica en que, aunque la determinación de $\mathtt{K}$ una vez conocido $\boldsymbol{p}$ es realmente sencilla (puede realizarse de modo lineal), no puede decirse lo mismo del cálculo previo de $\boldsymbol{p}$. Si no se tiene ningún otro tipo de información externa, pueden usarse las llamadas *restricciones de módulo*, que trabajan bajo la suposición de matriz de calibración constante. Estas restricciones establecen que los tres valores propios de la expresión conjugada resultante de calcular la homografía interimagen transferida a través del plano en el infinito, $\mathtt{H}_\infty = \mathtt{K}\mathtt{R}\mathtt{K}^{-1}$, deben tener igual módulo. Si la primera cámara es $\mathtt{K}[\mathtt{I}\,|\,\mathbf{0}]$ y la segunda tiene la forma general $[\mathtt{A}\,|\,\boldsymbol{a}]$, la homografía de transferencia del plano en el infinito resultante es $\mathtt{H}_\infty = \mathtt{A} - \boldsymbol{a}\boldsymbol{p}^\top$ (Hartley y Zisserman, 2000). Dado que $\mathtt{A}$ y $\boldsymbol{a}$ son conocidos por la reconstrucción proyectiva, la restricción de módulo anterior puede traducirse a una restricción de grado cuatro en los componentes de $\boldsymbol{p}$. Hacen falta, pues, un mínimo de tres vistas para acotar los tres grados de libertad del plano en el infinito.

La dificultad de resolver el sistema de ecuaciones de cuarto grado hacen a esta solución

quizá menos atractiva que las anteriores. A menudo, pues, la aproximación estratificada se usa sólo si hay otros medios más sencillos de determinar la posición $\boldsymbol{p}$ del plano del infinito, utilizando algún tipo de información adicional. Una posible forma de trabajar es determinar tres puntos de fuga correspondientes en un par de imágenes (a través de intersección de líneas paralelas proyectadas, por ejemplo). El plano del infinito en la reconstrucción proyectiva podría determinarse entonces fácilmente como el único plano que pasa por estos tres puntos. En cualquier caso, como ya se ha comentado, la posterior determinación de $\mathtt{K}$ queda lineal, usando la siguiente igualdad:

$$\boldsymbol{\omega}^* = \mathtt{H}_\infty \boldsymbol{\omega}^* \mathtt{H}_\infty^\top \tag{3.16}$$

Donde $\mathtt{H}_\infty$ es ya conocido a partir de $\mathtt{A}$, $\boldsymbol{a}$ y $\boldsymbol{p}$ (se puede forzar $\mathtt{H}_\infty = 1$ para eliminar la homogeneidad), y la $\mathtt{K}$ se vuelve a determinar por la descomposición de Cholesky de $\boldsymbol{\omega}^*$.

Antes de concluir la discusión sobre autocalibración, conviene hacer una consideración sobre su grado de aplicabilidad. A pesar de que, como hemos visto, se han encontrado varias soluciones para el problema teórico, en ocasiones los métodos propuestos son muy sensibles al ruido. Ello hace que algunas condiciones que deberían darse, como la definición positiva de $\boldsymbol{\omega}^*$ o el rango deficitario de $\mathtt{Q}_\infty^*$, no se produzcan numéricamente. Estos problemas de estabilidad en la estimación pueden producirse incluso en situaciones de operación perfectamente plausibles, lo que ha motivado a algunos autores a profundizar en el estudio del tema (véase, por ejemplo, (De Agapito *et al.*, 1998)).

Aún con algunos de estos problemas abiertos, la independencia absoluta de cualquier tipo de conocimiento previo sobre la escena o la cámara hace de la autocalibración una herramienta extremadamente útil para ciertas aplicaciones, impensables hasta hace pocos años. Quizá una de las más espectaculares sea la generación de modelos tridimensionales a partir de secuencias de vídeo genéricas. Se trata del viejo sueño que algunos autores han denominado "*VHS to VRML*", es decir, la conversión directa de vídeo a modelos de realidad virtual (Zisserman *et al.*, 1999). Una muy buena introducción sobre este tipo específico de aplicaciones es el tutorial ofrecido por Pollefeys (2000).

3.3.3. Calibración en sistemas autónomos

Como ya hemos dicho, el objetivo final de la calibración de las cámaras es la reconstrucción tridimensional sin ambigüedad proyectiva. Esta ambigüedad supone una limitación importante para los robots que toman sus decisiones de movimiento dependiendo de la información estructural del entorno obtenida durante dichas reconstrucciones. Así, la calibración adecuada de los sensores ópticos se convierte en una tarea crucial dentro de los sistemas autónomos. En esta sección estudiaremos algunas de las principales propuestas descritas en la literatura para enfrentarse a este problema.

Precalibración y métodos clásicos

Muchos autores optan por una solución basada en la precalibración, donde las características de la cámara o juego de cámaras son estimadas de modo previo a la actuación del robot. La ventaja fundamental es la precisión alcanzada, puesto que la calibración se hace en condiciones controladas y utilizando plantillas especialmente diseñadas para la tarea. Lebegue y Aggarwal (1993), por ejemplo, utilizan un robot con una cámara calibrada y de la que se conocen los ángulos de *roll* y *pitch* a través de un inclinómetro. La precalibración se utiliza también muy frecuentemente en sistemas dotados de cabezas estéreo, cuya operación está basada en el cálculo de correspondencias entre esquinas del par de imágenes, para obtener por triangulación las posiciones 3D de las mismas (véanse, a título de ejemplo, los robots descritos por Davison y Murray (1998) o Vicente (2002)). Si se conoce la distancia entre las cámaras (esto es, la longitud de la línea base), se elimina también el factor de escala, y se puede obtener la deseada reconstrucción euclídea. Otros sistemas utilizan un esquema similar, pero utilizando más de dos cámaras. Un ejemplo clásico, algo más antiguo, es el presentado por Zhang y Faugeras (1992). La plataforma correspondiente está dotada con tres cámaras, de las cuales se conoce su posición relativa y sus características internas. El principal inconveniente de la aproximación precalibrada, sin embargo, es la falta de flexibilidad, puesto que una vez calibrados los sensores, ninguno de los parámetros puede experimentar variaciones durante el periodo de funcionamiento.

Este problema puede solucionarse si el sistema posee algún conocimiento del entorno. Entonces aún puede adoptarse un esquema de calibración al estilo tradicional, pero que reestime constantemente los parámetros de calibración sobre la base del conocimiento de la escena. Ésta última jugaría un papel similar al de la plantilla de los métodos clásicos. Éste es el paradigma seguido por Drummond y Cipolla (2002), por ejemplo. Se trata de un brazo robótico que manipula objetos (piezas de barco, automóvil, etc.) de los que se tiene un preciso modelo tridimensional conocido con anterioridad. El sistema es capaz de localizar y seguir en 3D la posición de la pieza, al tiempo que determina los parámetros intrínsecos de la cámara, que también pueden cambiar en el tiempo. El modelo se reproyecta constantemente sobre la imagen para realizar su seguimiento usando una técnica de contornos activos o *snakes*, convenientemente adaptados para operar directamente sobre tres dimensiones. Los autores llaman a la técnica *wireframe snake*, algo que podría traducirse como "contorno activo de modelo alámbrico". La eficiencia del método permite cerrar un bucle robusto de predicción y reestimación de la posición al ritmo de la captura de imágenes (*frame rate*), unos 25 *fps*.

Enfoques no calibrados

Otros autores, conscientes de las limitaciones impuestas por la precalibración, prefieren adoptar un enfoque que elimine la necesidad de conocer las características de las cámaras. Se trataría de explotar el conocimiento del contexto concreto para minimizar el procesamiento

visual involucrado, reduciéndolo al estrictamente necesario para la tarea de interés. A veces, incluso, se evita la reconstrucción tridimensional explícita, y se trabaja sobre la imagen 2D para cerrar directamente bucles de control explícito sobre el móvil. Herman *et al.* (1997) y Robert *et al.* (1997) proponen varios ejemplos, como el seguimiento de una carretera tomando como única entrada el flujo horizontal de los contornos de ambos lados de la misma, o el seguimiento de objetivos (por ejemplo, un vehículo delante) que sólo utiliza el modelo para virar, independientemente de la distancia a la que se encuentra. La misma idea subyace en los trabajos donde se controla reactivamente al robot usando simplemente la divergencia de flujo óptico para evitar colisiones (Camus *et al.*, 1999). A pesar de la clara inferioridad expresiva de las estructuras manejadas, es cierto que el enfoque tiene también algunas ventajas significativas. Principalmente, la eficiencia computacional y la simplicidad, de las que presumiblemente se derivaría una mayor robustez en la función a realizar.

Existe también la posibilidad de trabajar directamente sobre ciertos invariantes proyectivos, sin realizar ningún tipo de calibración. Lee *et al.* (2000), por ejemplo, describen un robot que consigue autolocalizarse utilizando uno de estos invariantes. La idea es construir una base de datos de determinadas localizaciones del entorno, usando un conjunto de tuplas de cinco puntos en el suelo cuyas posiciones son conocidas. Después, durante la operación del robot, los puntos de interés detectados en la imagen se agrupan de cinco en cinco, en tuplas tentativas para intentar estimar si existe una homografía que los haga corresponder con alguna de las almacenadas. Dado que la correspondencia de cuatro puntos ya designa de modo único a una homografía, la quinta introduce la sobredeterminación necesaria para el cálculo del invariante, bidimensional en este caso. Una vez determinada una coincidencia con algún elemento de la base de datos, y realizando alguna suposición razonable sobre la situación de la cámara respecto a la plataforma, el robot queda perfectamente localizado.

Ya trabajando sobre reconstrucciones, pero relajando las restricciones sobre la condiciones que se imponen a las mismas, se encuentran sistemas como los descritos por Horaud *et al.* (1998) o Ruf y Horaud (2000). Se trata de sendos brazos robóticos que se controlan trabajando directamente sobre el espacio proyectivo. Esta relajación sobre la estructura necesaria para la generación del control obvia también, por supuesto, la necesidad de la calibración.

Uso de restricciones sobre el movimiento

En el caso de vehículos que se mueven sobre el suelo, las restricciones sobre los movimientos realizados por la cámara introducen simplificaciones adicionales. Un tipo de restricción muy utilizado es el llamado *movimiento planar*, efectuado por cámaras que se trasladan sobre un plano realizando giros únicamente alrededor de ejes perpendiculares a éste. A menudo este conocimiento se utiliza para elevar el tipo de las reconstrucciones practicadas con cámaras sin calibrar hasta un grado de afinidad (Beardsley y Zisserman, 1995). Pueden así realizarse algunas tareas elementales de navegación, como la planificación de caminos buscando el ma-

yor espacio libre, el centro de un pasillo, etc. Liang y Pears (2002) describen otras aplicaciones derivadas de las restricciones sobre el movimiento, como la estimación de la homografía del suelo a partir del seguimiento de sólo dos puntos en un movimiento de traslación pura (en lugar de los cuatro habituales), o la determinación del ángulo de giro usando únicamente la homografía interimagen inducida tras un giro. Por su mayor relación con nuestro trabajo, volveremos sobre estos puntos en apartados posteriores.

A veces, a la restricción del movimiento planar se le une también el aprovechamiento de las características poliédricas de los entornos estructurados. En estas condiciones pueden obtenerse numerosas pistas adicionales como, por ejemplo, la detección de puntos de fuga (Montiel y Zisserman, 2001). Estas pistas pueden ayudar en la realización de ciertas labores de modo relativamente preciso. En el trabajo citado, en concreto, se trabaja sobre estas suposiciones para construir modelos virtuales del entorno y estimar la posición y el movimiento relativo del robot dentro del mismo.

Si la cámara efectúa un movimiento estrictamente rotatorio (es decir, la posición del centro óptico no varía), entonces es conocido que todos los puntos de la primera imagen se transfieren a la segunda a través de una homografía $\mathtt{H} = \mathtt{K}'\mathtt{R}\mathtt{K}^{-1}$. En el caso de tener una calibración constante, puede utilizarse de nuevo la sencilla técnica de las cancelaciones introducidas al pre- y postmultiplicar $\boldsymbol{\omega}^*$ con $\mathtt{H}$ y $\mathtt{H}^\top$, respectivamente. Quedan entonces restricciones lineales sobre los elementos de $\boldsymbol{\omega}^*$ de la forma $\boldsymbol{\omega}^* = \mathtt{H}\boldsymbol{\omega}^*\mathtt{H}^\top$. Obviamente, trabajando con la inversa de la homografía puede obtenerse una derivación análoga para trabajar sobre la imagen de la cónica absoluta, $\boldsymbol{\omega} = \mathtt{K}^{-\top}\mathtt{K}^{-1}$, inversa de la $\boldsymbol{\omega}^*$ presentada en la sección 3.3.2, y que en ocasiones puede resultar más útil para imponer restricciones. La extensión para trabajar con cámaras variables es también relativamente sencilla, sólo que en este caso las ecuaciones a utilizar son de la forma $\boldsymbol{\omega}^{*\prime} = \mathtt{H}\boldsymbol{\omega}^*\mathtt{H}^\top$, y obviamente hay que introducir restricciones en la forma de las distintas matrices de calibración, del tipo píxel cuadrado, punto principal nulo, o similares. Un trabajo seminal en la calibración a partir de cámaras que efectúan movimientos rotatorios es el artículo de Hartley (1994). En el más moderno de De Agapito *et al.* (2001) se trata más explícitamente el problema de las cámaras con intrínsecos variables. Finalmente, en el ámbito más concreto de aplicación de la técnica a sistemas autónomos podríamos citar el trabajo de Beardsley *et al.* (1995), donde se adopta una actitud de visión activa para hacer rotar a la plataforma y obtener así las condiciones necesarias para la calibración.

3.4. Autocalibración odométrica

3.4.1. Generalidades

Ya en la sección anterior comentamos algunos casos del empleo de la visión activa como ayuda para llevar a cabo la calibración de las cámaras. Se trata de usar el propio movimiento para captar los cambios producidos en el entorno e inferir así propiedades tanto de la estruc-

tura de la escena como de los sensores ópticos utilizados. Naturalmente, es en el ámbito de los sistemas dotados de capacidad de actuación autónoma donde la aplicación de este tipo de técnicas resulta más atractiva. Un buen resumen del concepto de visión activa es la definición propuesta por Pahlavan, según la cual un sistema visual activo sería aquel '*capaz de manipular sus parámetros visuales de una forma controlada para extraer información útil sobre la escena en el tiempo y en el espacio*' (Bustos, 1998). Algunos autores, haciendo aún más énfasis en el carácter voluntario y oportunista del movimiento, hablan de "visión intencional" (del inglés *purposive vision*) al referirse a esta filosofía (Aloimonos *et al.*, 1988).

Una posible ampliación del enfoque activo consiste en la utilización del conocimiento del movimiento realizado. En las plataformas robóticas actuales es factible disponer de información cuantitativa bastante precisa de dicho movimiento, a través de diversos tipos de sensores más o menos adecuados dependiendo del grado de precisión deseado. Así, suelen emplearse desde sencillos mecanismos basados en la lectura del giro de las ruedas hasta el uso de sistemas de posicionamiento globales, basados en satélites, o locales, basados en estaciones repetidoras situadas en el entorno de operación, pasando por otro tipo de dispositivos como sensores inerciales, sistemas mecánicos, etc. En general, en la literatura sobre el tema se les conoce con el nombre genérico de sensores odométricos o propioceptivos, independientemente de la tecnología utilizada.

Nuestro Pioneer 2 DX, por ejemplo, utiliza el primero de los sistemas mencionados. A pesar de su sencillez, el sistema de medición del giro de las ruedas a través de la lectura de *encoders* puede llegar a ser bastante preciso si se calibran correctamente ciertas características mecánicas de la plataforma, como el radio y posición relativa entre las ruedas, el número de *encoders* por rueda, etc. Esto es sobre todo cierto para desplazamientos relativamente pequeños, puesto que dado el carácter inherentemente incremental de este tipo de medición, los errores son también acumulativos. Afortunadamente, las técnicas que propondremos en las secciones posteriores no necesitan desplazamientos largos, por lo que la precisión lograda es más que suficiente si se tiene cuidado en evitar movimientos especialmente mal condicionados para la calibración. En la última sección de este capítulo mostraremos algunos experimentos que corroboran esta última afirmación, al menos para el ámbito de aplicación que nos ocupa.

En lo que resta del capítulo, pues, estudiaremos las ventajas que la odometría puede aportar al procesamiento de la información visual. Nos centraremos en el estudio particular del movimiento planar, donde la cámara se traslada sobre un plano paralelo al suelo y gira sobre un eje perpendicular a éste. Esto se adapta perfectamente al modelo de cámara acoplada a un móvil que se desplaza sobre el suelo descrito en la sección 3.2. En estas condiciones, conocer la odometría significa tener acceso al radio de giro θ experimentado por la plataforma, y a su vector de traslación $(t_x, t_y)^\top$, ambos medidos respecto al origen de coordenadas del sistema S_{rob} introducido en el apartado 3.2.2. Naturalmente, la cámara que toma las imágenes antes y después del movimiento es la misma y, dado que no trabajaremos variando los intrínsecos, a

partir de este momento consideraremos constante la matriz de calibración $\mathtt{K}$[10].

3.4.2. El problema de la conmutación odométrica

En condiciones generales, como vimos en la sección 3.2.3, el conocimiento de la odometría de la plataforma no determina completamente el movimiento de la cámara, contrariamente a lo que podría parecer intuitivo en un primer momento. Sólo en condiciones de brazo nulo y ángulo $\beta = 0$ ambos movimientos serían coincidentes. En este apartado trataremos de aislar la influencia de este hecho en cualquier intento de utilización de la información de movimiento en la calibración. Puesto que el problema es completamente independiente de cualesquiera que sean los parámetros intrínsecos de la cámara, y por tanto común a muchos de los distintos procedimientos que propondremos en lo que resta del capítulo, lo describiremos en este momento para hacer después referencia a él en aquellos métodos en los que tenga alguna repercusión.

Dado un movimiento odométrico de traslación $(t_x, t_y)^\top$ y posterior giro θ sobre el suelo, los puntos del entorno $\boldsymbol{X} = (X, Y, Z, 1)^\top$, dados en coordenadas relativas a S_{rob} se trasladan a $\boldsymbol{X}' = \mathtt{O} \cdot \boldsymbol{X}$, donde $\mathtt{O}$ es la siguiente matriz de transformación euclídea:

$$\mathtt{O} = \begin{pmatrix} \cos(-\theta) & -\sin(-\theta) & 0 & 0 \\ \sin(-\theta) & \cos(-\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & -t_x \\ 0 & 1 & 0 & -t_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.17}$$

Para cada movimiento de este tipo existe un eje perpendicular al suelo cuyos puntos son invariantes a la transformación anterior[11]. Este eje corta al suelo en el siguiente punto:

$$\boldsymbol{X}_{fij} = (\frac{t_x + t_y \cot(\frac{\theta}{2})}{2}, \frac{t_y - t_x \cot(\frac{\theta}{2})}{2}, 0, 1)^\top \tag{3.18}$$

Estas coordenadas se corresponden a uno de los dos vectores propios con valor propio real de la matriz 3.17 (el otro es el punto en el infinito en la dirección del eje Z, $(0, 0, 1, 0)$). Podría pensarse en una cuerda anclada en dicho punto fijo desde la que se hace girar el vehículo, que se hallaría atado al otro extremo de la cuerda. La figura 3.4(a) ilustra esta interpretación física para un movimiento odométrico de ejemplo. Como veremos, esta intersección del *screw axis* con el plano del suelo adquirirá un cierto protagonismo en el razonamiento que haremos a continuación.

Dados unos valores concretos de θ, t_x y t_y, existe una familia de matrices de transformación similares, con dos grados de libertad parametrizables mediante las variables ϕ y s, que

[10] Nos referimos a cambios de focal durante el mismo proceso de autocalibrado. Si durante la operación del robot se necesitase cambiar la distancia focal de la cámara, por ejemplo, habría que volver a aplicar alguno de los procedimientos de calibración propuestos para reestimar su valor.

[11] En la literatura en inglés se le suele llamar *screw axis*, que podría traducirse como "eje de giro".

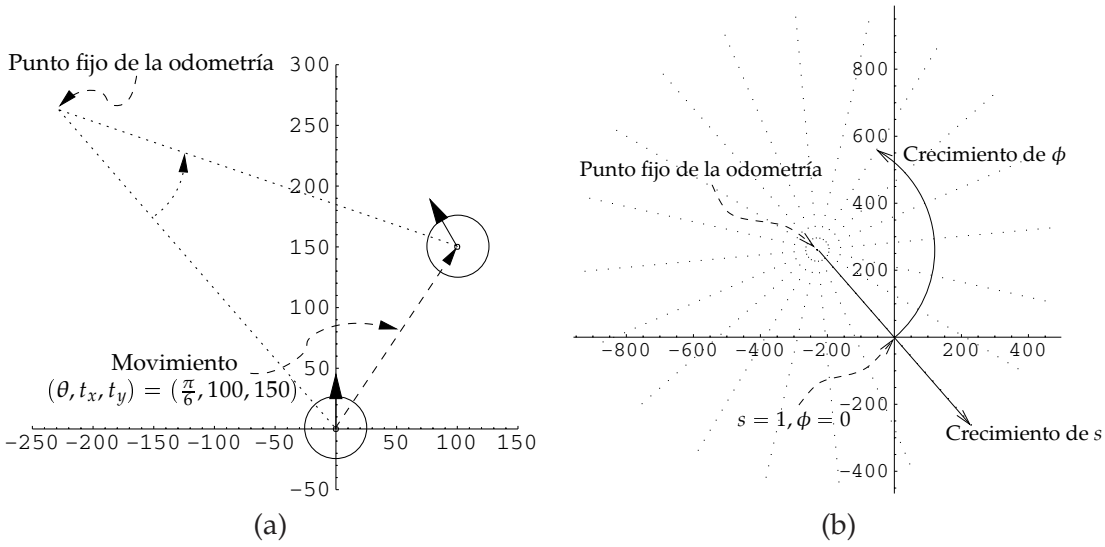


Figura 3.4: *(a) Punto fijo del movimiento odométrico* $(t_x, t_y, \theta) = (100, 150, \frac{\pi}{6})$*, de coordenadas* $(-229.9, 261.6)$ *en este caso. (b) Familia de posiciones de* (c_x, c_y) *para dicho movimiento. Se observa que estas posiciones se agrupan en círculos concéntricos, con centro en el punto fijo, a una distancia a éste igual a* s*, y con un ángulo igual a* ϕ*, si se parte de un ángulo* $\phi = 0$ *para el que pasa por el origen. De esta forma, a* $(c_x, c_y) = (0, 0)$ *le corresponden los parámetros* $s = 1$ *y* $\phi = 0$*.*

conmutan con la matriz original $\mathtt{O}$. Esto es, cada una de estas matrices, a las que denominaremos $\mathtt{O}_{c(\phi,s)}$, cumplen que $\mathtt{O}\mathtt{O}_{c(\phi,s)} = \mathtt{O}_{c(\phi,s)}\mathtt{O}$, y por tanto que $\mathtt{O} = \mathtt{O}_{c(\phi,s)}\mathtt{O}\mathtt{O}^{-1}_{c(\phi,s)}$. En definitiva, el significado de esta igualdad es que, como veremos, a todos los puntos del plano les da lo mismo moverse según una odometría dada, y entonces cambiar al nuevo sistema de coordenadas, que cambiar primero al nuevo sistema de coordenadas y entonces moverse en éste según la misma odometría. Ésta es la razón por la cual, a partir de ahora, hablaremos del *problema de la conmutación odométrica* al hacer referencia a la existencia de esta familia de matrices. Se puede comprobar que dichas matrices tienen la forma (eliminando ya la dependencia explícita de (ϕ, s) del subíndice, por abreviar)

$$\mathtt{O}_c = \begin{pmatrix} s\cos(\phi) & -s\sin(\phi) & 0 & c_x \\ s\sin(\phi) & s\cos(\phi) & 0 & c_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.19}$$

donde c_x y c_y son, respectivamente

$$c_x = \frac{(1 - s\cos(\phi))(t_x - t_y\cot(\frac{\theta}{2})) + s\sin(\phi)(t_x\cot(\frac{\theta}{2}) + t_y)}{2} \tag{3.20}$$

$$c_y = \frac{-st_x \sin(\phi) + t_y - s\cos(\phi)(t_x \cot(\frac{\theta}{2}) + t_y) + \cot(\frac{\theta}{2})(t_x + st_y \sin(\phi))}{2} \tag{3.21}$$

Recordemos que θ, t_x y t_y vienen dados por la odometría, así que las únicas variables son ϕ y s, los dos parámetros que caracterizan a cada una de las matrices de la familia. Es importante recordar que los conmutadores resultantes son siempre transformaciones similares. En la figura 3.4(b) se muestra la familia de posiciones (c_x, c_y) sobre el plano del suelo, según los valores dados a los parámetros ϕ y s para un movimiento odométrico concreto de ejemplo.

Así pues, dada una odometría cualquiera y un punto arbitrario (c_x, c_y) del plano del suelo no coincidente con el punto fijo del movimiento, siempre podremos resolver los dos parámetros libres ϕ y s utilizando las ecuaciones 3.20 y 3.21. Esto significa que cualquier punto (c_x, c_y) del plano puede ser tomado como origen de un nuevo robot (con un giro ϕ y una escala s respecto al original, determinados unívocamente por la odometría y el punto del suelo elegidos) tal que, al experimentarse sobre él el mismo movimiento odométrico relativo, todos los puntos del mundo que se muevan solidariamente respecto al primer robot lo hacen también de modo solidario con respecto al segundo.

En particular, si fijamos una cámara cualquiera en la primera plataforma con un desfase (x_c, y_c, β), ésta se moverá entre dos posiciones congruentes con un desfase alternativo (x_c', y_c', β') respecto a un segundo robot que experimentase el mismo movimiento odométrico. La figura 3.5(a) muestra la situación descrita para el mismo movimiento de ejemplo anterior, eligiendo los valores $s = 1.5$ y $\phi = \frac{\pi}{7}$. Y ello, como se muestra en la figura 3.5(b), para cualquiera que sea la posición relativa (x_c, y_c, β) de la cámara respecto al primer robot, puesto que la restricción de movimiento solidario se cumple para todos los puntos del plano.

Sin embargo, si se realizan dos movimientos odométricos con puntos fijos distintos, entonces es claro que la única forma válida para el conmutador común a ambos movimientos sería la dada por la solución trivial $(\phi, s) = (0, 1)$, que implica que $(c_x, c_y) = (0, 0)$. Esto se comprueba por simple inspección directa de la figura 3.4(b). Al tener puntos fijos desiguales, el único lugar en el que coincidirán los valores de ϕ y s para ambos movimientos será en el origen, para los valores ya comentados de 0 y 1, respectivamente.

El problema de la conmutación explica ciertas ambigüedades que pueden surgir en la calibración odométrica de una cámara fija sobre un móvil. Ello nos obligará a veces a realizar más de un movimiento y tomar imágenes desde varios lugares para determinar unívocamente una solución. No obstante, veremos también cómo ciertos problemas aún pueden resolverse con un sólo movimiento, siempre que se apliquen ciertas simplificaciones de las comentadas en la sección 3.2.3.

En los apartados que siguen iremos desgranando las distintas posibilidades de solución a la calibración, comentando en cada caso la influencia de la existencia de los conmutadores vistos en esta sección.

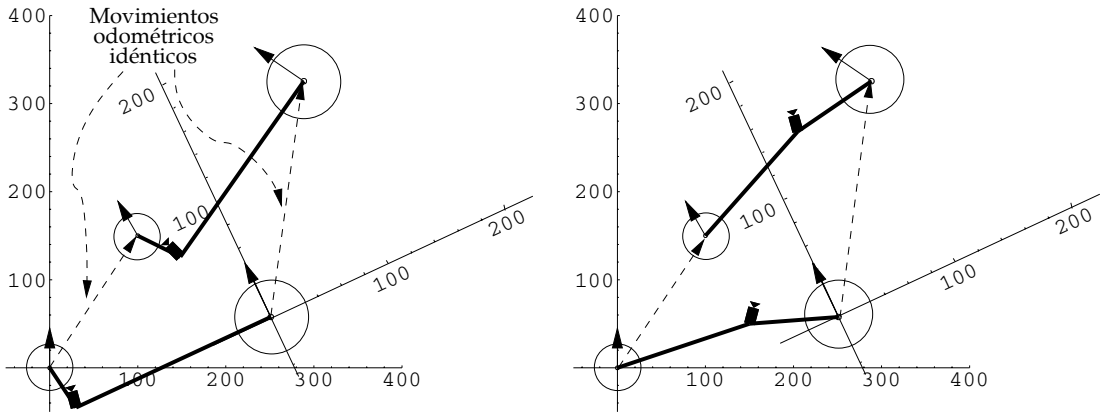


Figura 3.5: *Ilustración del problema de la conmutación odométrica: (a) Dado el movimiento odométrico de ejemplo* $(t_x, t_y, \theta) = (100, 150, \frac{\pi}{6})$*, se observa cómo la cámara fijada con un desfase* $(x_c, y_c, \beta) = (30, -45, \frac{\pi}{12})$ *respecto al primer robot queda en los mismos lugares respectivos antes y después del movimiento que otra cámara que estuviese fijada con otros desfases* $(x_c', y_c', \beta') = (-162.48, 2.26, -\frac{5\pi}{84})$ *en un segundo robot cuyo punto de partida estuviese en el punto* $(c_x, c_y) = (251.06, 57.69)$ *dado por las ecuaciones 3.20 y 3.21, una vez elegidos arbitrariamente los parámetros* $s = 1.5$ *y* $\phi = \frac{\pi}{7}$*. Obsérvese cómo las cámaras antes y después del movimiento quedan exactamente en los mismos sitios respectivos para ambos sistemas. (b) En realidad, esto se cumplirá para cualquier punto que se mueva de modo solidario con la primera plataforma, puesto que también lo hará de ese modo sobre la segunda. Esto se ilustra con el mismo movimiento, para otras cámaras con sendos desfases relativos* $(x_c, y_c, \beta) = (150, 50, -\frac{\pi}{12})$ *y* $(x_c', y_c', \beta') = (-62.92, 24.61, -\frac{19\pi}{84})$*.*

3.5. Solución general tridimensional

En los apartados que siguen supondremos que disponemos de una plataforma móvil con una cámara acoplada como la descrita en la sección 3.2, sobre la que se tiene acceso a la información de movimiento en el sistema S_{rob}, en la forma ángulo de rotación θ y traslación $(t_x, t_y)^\top$, ambos sobre el suelo. Tal y como se comentó en la sección 3.3.2, es sencillo lograr una reconstrucción de la escena a partir de un mínimo de dos imágenes, con matrices de cámara $\mathtt{A}^i$ y de puntos $\mathtt{X}_j^A$ a falta de la rectificación desconocida $\mathtt{H}$, correctora de la ambigüedad proyectiva. Por completitud en la descripción, mostramos en el algoritmo 3.1 un sencillo procedimiento para lograr esta reconstrucción a partir de un par de imágenes usando sólo métodos lineales. En caso de disponerse de más vistas, pueden igualmente calcularse el conjunto de matrices de proyección $\mathtt{A}^i$ y los puntos $\mathtt{X}_j^A$ aprovechando la sobredeterminación para obtener una solución mediante algunas de las técnicas de reconstrucción a partir de secuencias bien estudiadas en la literatura (Hartley y Zisserman, 2000).

Ahora bien, la solución anterior no utiliza para nada la información proporcionada por la odometría. Resulta interesante estudiar qué puede ganarse al conocer los valores (t_x, t_y, θ) experimentados por la plataforma portadora de la cámara. En una primera aproximación, podría pensarse en utilizar la expresión $\mathtt{F} = \mathtt{K}^{-\top}[\boldsymbol{t}]_\times \mathtt{R}\mathtt{K}^{-1}$, donde $\boldsymbol{t}$ y $\mathtt{R}$ son precisamente el

ENTRADA:
- Conjunto de puntos de imagen correspondientes entre dos imágenes, $\boldsymbol{x}_j$ y $\boldsymbol{x}'_j$, $\forall j = 1 \dots m$, en coordenadas homogéneas.

/* *Para poder atacar el cálculo de la fundamental de forma lineal, supondremos que* $m \geq 8$. *Igualmente, supondremos que las coordenadas de los puntos de entrada se encuentran normalizadas (media cero y desviación típica unidad en los ejes X e Y), con el fin de minimizar la propagación de los errores en las soluciones. Los puntos 3D originales deben estar en situación general (no coplanares).* */

SALIDA:
- Cámaras con ambigüedad proyectiva $\mathtt{A}$ y $\mathtt{A}'$.
- Conjunto de puntos 3D con ambigüedad proyectiva $\mathtt{X}_j^A$, $\forall j = 1 \dots m$, también en coordenadas homogéneas.

ALGORITMO:

Cálculo de la matriz fundamental:

- Resolver el sistema de m ecuaciones $\boldsymbol{x}_j \mathtt{F} \boldsymbol{x}'_j = 0$, con 9 incógnitas dadas por los elementos de la matriz $\mathtt{F}$, pero sólo 8 grados de libertad por el factor de homogeneidad. La solución viene dada por el espacio nulo de la matriz $\mathtt{B}_{m\times 9}$ de coeficientes resultante. Dicho espacio nulo viene dado por el vector $\boldsymbol{v}_9$ menor valor singular (cero en el caso ideal) correspondiente a la descomposición SVD de $\mathtt{B}$ (ver nota a pie de página 9 en este mismo capítulo).

Cálculo de las cámaras con ambigüedad proyectiva:

- Determinar el epipolo $\boldsymbol{e}'$ de la segunda imagen, como el vector correspondiente al menor valor singular de la SVD de $\mathtt{F}$, por la izquierda (sería el vector $\boldsymbol{u}_3$, correspondiente a la tercera fila de la matriz $\mathtt{U}$ de la nota al pie 9). Idealmente, este valor singular también debería ser cero, por la condición de rango 2 de la $\mathtt{F}$.
- Hacer $\mathtt{A} = [\mathtt{I}|\mathbf{0}]$ y $\mathtt{A}' = [[\boldsymbol{e}']_\times \mathtt{F}|\boldsymbol{e}']$. Se puede comprobar que estas dos matrices de cámara son compatibles con la $\mathtt{F}$ estimada.

Reconstrucción proyectiva del conjunto de puntos:

for $j := 1$ ***to*** m ***do***

- Para el par de puntos $(\boldsymbol{x}_j, \boldsymbol{x}'_j)$, plantear las ecuaciones homogéneas $\boldsymbol{x}_j = \mathtt{A}\boldsymbol{X}_j^A$ y $\boldsymbol{x}'_j = \mathtt{A}'\boldsymbol{X}_j^A$, donde sólo los cuatro elementos de cada $\boldsymbol{X}_j^A$ son desconocidos. Esto equivale a $\boldsymbol{x}_j \times \mathtt{A}\boldsymbol{X}_j^A = \mathbf{0}$ y $\boldsymbol{x}'_j \times \mathtt{A}'\boldsymbol{X}_j^A = \mathbf{0}$, un total de 6 ecuaciones homogéneas, pero sólo 4 independientes, para las 4 incógnitas de $\boldsymbol{X}_j^A$, con sólo 3 grados de libertad por la homogeneidad de la solución. De nuevo, se puede resolver mediante el vector derecho de menor valor singular de la SVD de la matriz $\mathtt{C}_{6\times 4}$ de coeficientes resultantes.

endfor

Algoritmo 3.1: *Obtención de una reconstrucción proyectiva a partir de correspondencias de puntos en un par de imágenes.*

vector de traslación y la rotación entre el par de cámaras. Desgraciadamente, dicha expresión es sólo válida para un sistema de coordenadas del mundo alineado con la primera cámara, cuya situación con respecto al móvil del que tomamos la odometría viene dada por el vector $(x_c, y_c, z_c)^\top$ y los ángulos α, β y γ (ver figura 3.2), que en principio son desconocidos.

De modo que hay que atacar el problema de otra manera. La reconstrucción que nos interesa es la euclídea respecto al sistema S_{rob}, anulando la ambigüedad proyectiva de las matrices de cámara $\mathtt{A}^i$ y los puntos $\mathtt{X}_j^A$, obtenidos con el algoritmo 3.1. Necesitamos, pues, la homografía $H_{4\times 4}$ correctora de esta ambigüedad. Si utilizamos la transformación odométrica

$\mathtt{O}$ introducida en 3.17, podemos establecer una restricción sobre la matriz de proyección real $\mathtt{P}$, con coordenadas de posición relativa al sistema S_{rob}, asociando cada movimiento odométrico a la correspondiente matriz $\mathtt{A}^i$. Dichas cámaras, convenientemente corregidas con la homografía $\mathtt{H}$, deben ser compatibles con las correspondientes matrices $\mathtt{O}^i$ de odometría. Así, para cada imagen deberá cumplirse la siguiente igualdad:

$$\mathtt{A}^i\mathtt{H} = \mathtt{P}\mathtt{O}^i \tag{3.22}$$

Nótese que la igualdad anterior posee un total de 28 incógnitas, 16 de la homografía $\mathtt{H}$ y 12 de la matriz de proyección $\mathtt{P}$, para un total de sólo 12 ecuaciones, las correspondientes a los 3×4 elementos de cada lado. Sin embargo, la $\mathtt{H}$ y la $\mathtt{P}$ son afortunadamente comunes a la secuencia completa de imágenes, puesto que el movimiento entre las cámaras queda completamente recogido por las matrices de odometría $\mathtt{O}^i$, y por tanto la matriz de proyección $\mathtt{P}$ es siempre la relativa a S_{rob}, justo lo que nos interesa. De esta forma, un simple argumento de cuenta de grados de libertad demuestra que harían falta al menos tres ecuaciones del tipo 3.22 para determinar una única solución. Esto significaría tomar tres imágenes desde tres puntos de odometrías conocidas[12].

Pero existe una pequeña dificultad. Si llamamos $\mathtt{B}$ a la matriz de $12n \times 28$ coeficientes del sistema homogéneo planteado por el conjunto de ecuaciones 3.22 con n vistas, donde $n \geq 3$, puede comprobarse que, en presencia de datos sin ruido, la solución a dicho sistema no es única. Dicho de otra forma, el espacio nulo de $\mathtt{B}$ no resulta ser de dimensión unidad (lo que determinaría la solución unívocamente), sino de dimensión tres[13]. Esto significa que las matrices $\mathtt{H}$ y $\mathtt{P}$ obtenidas son en realidad sendas familias de soluciones con tres grados de libertad. Esta multiplicidad en la solución es debida a la falta de generalidad del movimiento estrictamente planar: dado que el eje de giro es siempre perpendicular al suelo, y que no existe desplazamiento vertical, el sistema de ecuaciones anterior es incapaz de desambiguar la escala y el origen de coordenadas de la dirección Z.

Sin embargo, se comprueba también que todas las matrices de cámara $\mathtt{P}^*$ de la familia de soluciones tienen en común lo más importante, el desplazamiento (x_c, y_c) sobre el plano horizontal y el ángulo β de *pan*[14]. Esto significa que las correspondientes homografías correctoras $\mathtt{H}^*$ son capaces de reconstruir perfectamente las coordenadas X e Y de los puntos del mundo, y que sólo existe un problema de indeterminación de escala y origen en la coordenada Z. Se trataría, pues, de una reconstrucción únicamente afín, pero euclídea en los ejes más interesantes, sobre los que se realiza la navegación. Desafortunadamente, esta ambigüedad en el eje

[12] Un modo habitual de trabajar será considerar $\mathtt{O}^1 = \mathtt{I}$ y realizar dos movimientos posteriores $({t_x}^2, {t_y}^2, \theta^2)$ y $({t_x}^3, {t_y}^3, \theta^3)$ para obtener $\mathtt{O}^2$ y $\mathtt{O}^3$ según 3.17.

[13] Esto es sólo cierto para movimientos odométricos generales, con puntos fijos distintos. En caso contrario, existiría también el problema de la conmutación odométrica comentado en la sección 3.4.2, con los consiguientes grados de libertad adicionales derivados de la degeneración del movimiento.

[14] Esta comprobación se puede hacer factorizando dichas cámaras usando la descomposición $\mathtt{RQ}$ (ver nota al pie 5) sobre las tres primeras columnas de las matrices $\mathtt{P}$, para dejarlas en la forma introducida en 3.4.

Z se acopla con la estimación de $\mathtt{K}$, de forma que tampoco es posible hacer una calibración intrínseca explícita.

Ahora bien, cabe preguntarse en qué condiciones podríamos conseguir una reconstrucción euclídea completa. En primer lugar, podría pensarse en un movimiento odométrico capaz de desplazar la cámara verticalmente y de realizar un giro con eje no vertical. Dicho movimiento desambiguaría también la escala en Z, determinando de modo único la matriz de intrínsecos $\mathtt{K}$, así como los extrínsecos de rotación α, β, γ y la posición de la cámara $(x_c, y_c)^\top$ sobre el plano. Con respecto al único parámetro restante, la altura z_c, lo único que se puede hacer, obviamente, es fijar la altura cero en la correspondiente posición del centro óptico de la cámara, puesto que en principio se desconocen aquellos puntos de la reconstrucción que pertenecen al suelo.

De todos modos, aún queda una segunda oportunidad para practicar la calibración completa, aunque el movimiento sea estrictamente planar. Consiste en realizar algunas de las simplificaciones discutidas en el apartado 3.2.3. En particular, es útil realizar suposiciones tales como punto principal conocido o *skew* nulo, sobre cuya viabilidad ya discutimos al hablar de los métodos genéricos de autocalibración. Entonces siempre podríamos aplicar un método similar al de la cuádrica dual absoluta: según lo discutido en el apartado 3.3.2, tendríamos que la matriz de proyección $\mathtt{P} = \mathtt{AH}$ debería cumplir 3.11. Aplicando esta igualdad al caso $s = o_x = o_y = 0$, por ejemplo, la $\boldsymbol{\omega}^*$ queda diagonal en f_x^2 y f_y^2, simplificando bastante el problema.

Para resolver 3.11 utilizaremos alguna de las soluciones de $\mathtt{P}^*$ y $\mathtt{H}^*$ obtenidas del espacio nulo de las ecuaciones 3.22. Buscamos entonces una segunda corrección $\mathtt{H}^{**}$ que termine de corregir la ambigüedad restante, con $\mathtt{H} = \mathtt{H}^*\mathtt{H}^{**}$. La expresión para $\mathtt{Q}_\infty^* = \mathtt{H}^{**}\tilde{\mathtt{I}}\mathtt{H}^{**\top}$ queda ahora muy sencilla, puesto que, como comentamos anteriormente, la reconstrucción $\mathtt{H}^*$ quedaba únicamente a salvo de escala y origen en Z. Esto es, $\mathtt{H}^{**}$ es de la forma:

$$\mathtt{H}^{**} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.23}$$

Donde sólo a y b son desconocidos. $\mathtt{Q}_\infty^*$, por tanto, queda así:

$$\mathtt{Q}_\infty^* = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & a^2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.24}$$

Obsérvese que el valor de b, que permitiría calcular el origen en Z, se pierde. Esto es coherente con el hecho, argumentado anteriormente, de que el desplazamiento vertical no

puede calcularse sin conocer algún dato externo sobre la posición del suelo. La ecuación 3.11 queda entonces muy sencilla:

$$\boldsymbol{\omega}^* = \begin{pmatrix} kf_x^2 & 0 & 0 \\ 0 & kf_y^2 & 0 \\ 0 & 0 & k \end{pmatrix} = \mathtt{P}^* \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & a^2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \mathtt{P}^{*\top} \tag{3.25}$$

La constante de proporcionalidad k se añade explícitamente para poder manejar la homogeneidad en la igualdad. El sistema anterior, lineal en las incógnitas kf_x^2, kf_y^2, k y a^2 queda sobredeterminado, con lo que pueden calcularse sin problemas los correspondientes valores de las focales f_x y f_y en ambos ejes y del factor de escala a del eje Z. Haciendo, por ejemplo, $b = 0$, y usando 3.23, podemos obtener las matrices de cámara buscadas, $\mathtt{P} = \mathtt{AH}^*\mathtt{H}^{**}$. El algoritmo 3.2 muestra un resumen completo del procedimiento propuesto.

Obviamente, podría también estudiarse el efecto de otras simplificaciones sobre la solución al problema. Por ejemplo, considerar sólo píxel cuadrado, con punto principal desconocido, sólo *skew* cero (en cuanto a los parámetros intrínsecos), o brazo nulo (en lo que se refiere a los extrínsecos). En lugar de ello, pasaremos directamente al estudio de una situación más interesante, en la que se simplifican tanto los algoritmos de calibración como los requerimientos de la entrada. Se trata del caso en el que se tiene acceso de alguna forma a la situación del plano del suelo. Al estudio detallado de esta posibilidad, más práctica en lo que a navegación visual se refiere, dedicaremos lo que resta del capítulo.

3.6. Solución bidimensional lineal

Efectivamente, a veces es posible determinar de alguna forma que ciertos puntos o segmentos de la imagen pertenecen al suelo. Esta determinación se puede realizar bien a través de consideraciones estrictamente geométricas (Pears y Liang, 2001), o bien, como será nuestro caso, mediante algún tipo de interpretación basada en el color o la textura del suelo, por ejemplo. La disminución en el número de entidades geométricas a seguir vendrá del hecho de que podremos trabajar directamente con homografías planares, con menos exigencias en la entrada que el cómputo de relaciones interimagen más generales. Por ejemplo, la estimación lineal de una homografía inducida por un plano entre dos imágenes requiere correspondencias entre sólo cuatro puntos, en lugar de los ocho necesarios para la determinación de la matriz fundamental. No hay que olvidar tampoco que sólo trabajando con algún conocimiento sobre la situación del suelo podremos calcular el parámetro que nos falta, la altura z_c de la cámara. Como ya se argumentó, sería imposible determinar este valor sin algún tipo de pista externa. Pero quizá lo más importante sea que la odometría de movimiento de la que se dispone está dada precisamente sobre el plano del suelo, lo que justifica también la conveniencia de la detección y utilización de éste.

ENTRADA:
- Secuencia de cámaras con ambigüedad proyectiva $\mathtt{A}^i, \forall i = 1 \ldots n$, con $n \geq 3$.
- Secuencia de odometrías planares correspondientes $(t_x^i, t_y^i, \theta^i), \forall i = 1 \ldots n$. Por simplicidad, y sin pérdida de generalidad, supondremos que la primera imagen se toma desde la posición inicial, esto es, $(t_x^1, t_y^1, \theta^1) = (0, 0, 0)$.
- Conjunto de puntos 3D con ambigüedad proyectiva $\mathtt{X}_j^A, \forall j = 1 \ldots m$.

SALIDA:
- Secuencia de cámaras euclídeas $\mathtt{P}^i, \forall i = 1 \ldots n$.
- Reconstrucción euclídea de la escena $\mathtt{X}_j, \forall j = 1 \ldots m$.

ALGORITMO:

Cálculo de la reconstrucción a falta de escala en Z:

- Resolver el sistema de $12m$ ecuaciones con 28 incógnitas 3.22, donde las matrices $\mathtt{O}^i$ vienen dadas por 3.17. Dado que el espacio nulo de la correspondiente matriz $\mathtt{B}$ de coeficientes es de dimensión tres, existe una familia de soluciones con ese número de grados de libertad. Sean $\mathtt{P}^*$ y $\mathtt{H}^*$ dos soluciones cualesquiera de la familia (por ejemplo, uno de los tres vectores de valor singular nulo de la SVD de $\mathtt{B}$), o cualquier combinación lineal de éstos. Ambas proporcionan la reconstrucción a falta de la escala en Z.

  / *Obsérvese que, por la forma de las ecuaciones, no hace falta preocuparse por la homogeneidad de* *
  * *las igualdades, puesto que la escala final de las soluciones* $\mathtt{P}^*$ *y* $\mathtt{H}^*$ *automáticamente se determi-* *
  * *nará para compensar los valores de los determinantes de las matrices* $\mathtt{O}^i$ *y* $\mathtt{A}^i$*, respectivamente.* */

Corrección de escala en Z:

- Determinar el valor de a resolviendo el sistema 3.25 (lineal en las incógnitas kf_x^2, kf_y^2, k y a^2).
- Con el valor de a obtenido, y haciendo $b = 0$, determinar $\mathtt{H}^{**}$ utilizando la expresión 3.23.

Determinación de la cámara euclídea:

- Hacer $\mathtt{P} = \mathtt{P}^*\mathtt{H}^{**} = \mathtt{A}^1\mathtt{H}^*\mathtt{H}^{**}$.

Reconstrucción euclídea de la escena:

for $i := 1$ ***to*** m ***do***

- Hacer $\mathtt{X}_j = \mathtt{H}^{-1}\mathtt{X}_j^A$, con $\mathtt{H} = \mathtt{H}^*\mathtt{H}^{**}$.

endfor

Algoritmo 3.2: *Autocalibración a partir de tres o más imágenes, conociendo la odometría de los movimientos planares. La primera parte del procedimiento realiza la calibración a falta de escala en Z. En la segunda se amplía el procedimiento con la suposición de* skew *nulo y punto principal conocido para desambiguar dicha escala. Se parte de una reconstrucción proyectiva de la escena, que podría calcularse usando el algoritmo 3.1.*

La herramienta geométrica básica sobre la que trabajaremos a continuación será, pues, la homografía bidimensional que relaciona directamente las coordenadas de la imagen con las coordenadas del suelo en el sistema S_{rob}, sobre el que se realiza el control del robot. En primer lugar, describiremos una solución lineal para la determinación de dicha homografía a partir de la odometría, partiendo del caso de cámara en posición general respecto a la plataforma. Veremos después cómo se reduce el número de movimientos necesarios si se realiza la simplificación de brazo nulo. En una sección posterior, realizando algunas simplificaciones adicionales, comprobaremos cómo el problema puede abordarse incluso analíticamente, con las consiguientes ventajas de eficiencia computacional y disminución del número de primitivas necesarias, además de las derivadas del hecho de tener acceso a expresiones simbólicas de los parámetros intrínsecos y extrínsecos a estimar en función de las posiciones de puntos o

segmentos en la imagen y los valores de la odometría. Estas expresiones descubren de modo más claro la relación entre las entradas y las salidas del algoritmo, al tiempo que permiten tareas tales como el estudio analítico sencillo de la propagación de errores, por ejemplo. En presencia de algunas de estas simplificaciones, incluso, el conocimiento de la homografía del suelo equivaldrá a obtener la cámara completa, lo que evidentemente constituye una gran ventaja adicional. Finalmente, realizaremos una serie de experimentos con el fin de ilustrar la aplicabilidad de los métodos en condiciones realistas, para terminar con una breve discusión a modo de resumen.

3.6.1. Homografía del suelo

Si conociésemos las coordenadas reales en S_{rob} de al menos cuatro puntos del suelo visibles en la imagen, podríamos estimar directamente la homografía $\tilde{\mathtt{P}}$ de transferencia suelo-imagen de modo lineal, simplemente resolviendo el siguiente sistema:

$$\tilde{\mathtt{P}}\tilde{\boldsymbol{X}}^i = \boldsymbol{x}^i, \forall i = 1 \dots m \tag{3.26}$$

Donde $\tilde{\mathtt{P}}_{3\times 3}$ es la homografía de interés, desconocida, y $\tilde{\boldsymbol{X}}^i$ y $\boldsymbol{x}^i$ son los vectores homogéneos de coordenadas 2D de puntos de suelo e imagen, respectivamente. Puesto que se tienen nueve incógnitas (elementos de $\tilde{\mathtt{P}}$) con ocho grados de libertad por la homogeneidad, y cada punto introduce dos restricciones independientes, serían necesarios al menos cuatro puntos ($m \geq 4$) para determinar la homografía, aunque siempre pueden utilizarse más para obtener una solución de mínimos cuadrados. Obsérvese que la homografía $\tilde{\mathtt{P}}$ es simplemente la matriz de cámara $\mathtt{P}$ de la expresión 3.4, a la que se ha eliminado la tercera columna, como corresponde a la transformación de los puntos del mundo sobre los que trabaja, con $Z = 0$. La misma notación se emplea sobre los vectores $\tilde{\boldsymbol{X}}^i$ para indicar la eliminación de esta coordenada.

Sin embargo, en general no será fácil tener acceso directo a las coordenadas de los puntos del suelo en S_{rob}, a no ser que nos encontremos en condiciones controladas, lo que se saldría del enfoque autocalibrado. Supongamos, no obstante, que sí somos capaces de determinar al menos cuatro puntos $\boldsymbol{x}^i$ de la imagen que están contenidos en el plano del suelo. Nuestro objetivo es estimar directamente la homografía $\tilde{\mathtt{P}}$ a través del seguimiento interimagen de estos puntos, conociendo el movimiento planar experimentado por el vehículo entre la toma de las imágenes, y obviando la necesidad del conocimiento previo de los valores $\tilde{\boldsymbol{X}}^i$.

Llamaremos $\mathtt{H}$ a la homografía que relaciona las posiciones de los puntos de imagen $\boldsymbol{x}^i$ y $\boldsymbol{x'}^i$ antes y después de un movimiento, y $\tilde{\mathtt{O}}$ a la homografía euclídea bidimensional que recoge la odometría planar descrita por la plataforma durante el mismo. Esta matriz, siguiendo de nuevo el convenio notacional anterior, no es más que la $\mathtt{O}$ de la ecuación 3.17 a la que se le han eliminado la fila y columna terceras, correspondientes a la coordenada Z. Cabe comentar aquí que el giro θ sobre el suelo es extraíble directamente de la homografía interimagen $\mathtt{H}$, por lo que su conocimiento odométrico no añade en realidad ninguna información extra (Liang

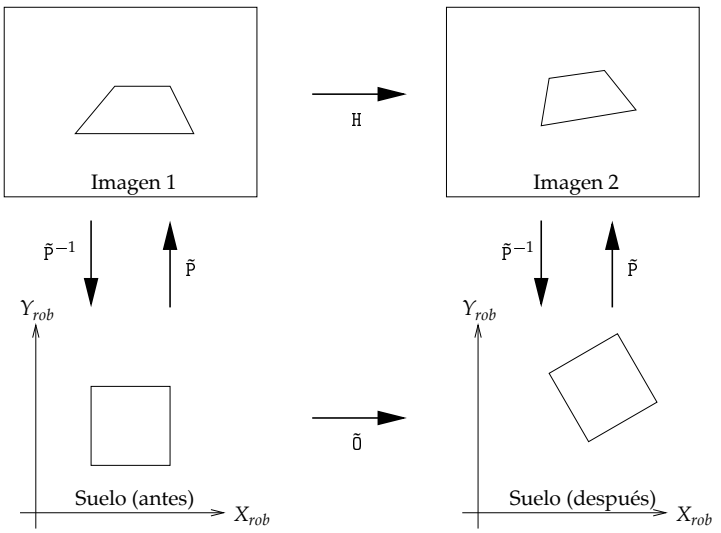


Figura 3.6: *Relación entre las homografías de transferencia interimagen* H *antes y después del movimiento, la de odometría* $\tilde{O}$*, la de proyección* $\tilde{P}$ *y su inversa, la de reconstrucción* $\tilde{P}^{-1}$*.*

y Pears, 2002). Este ángulo de giro coincide precisamente con la fase de los valores propios complejos de la H, como se justifica en la referencia mencionada. Su única utilidad en los procedimientos descritos en este apartado, por tanto, estaría en comprobar la validez de la homografía interimagen estimada.

La relación entre todas las transformaciones de interés se describe en la figura 3.6. De la observación de dicha figura se deduce que los puntos de la imagen izquierda, transferidos a la de la derecha a través de H, deben transformarse de modo idéntico mediante la homografía inversa (de reconstrucción) $\tilde{P}^{-1}$, sobre la que se aplica posteriormente el movimiento $\tilde{O}$, para proyectarlos finalmente en la imagen derecha a través de $\tilde{P}$. Esto se traduce en la igualdad entre homografías $H = \tilde{P}\tilde{O}\tilde{P}^{-1}$, equivalente a la siguiente ecuación:

$$H\tilde{P} = \tilde{P}\tilde{O} \tag{3.27}$$

$\tilde{O}$ es conocida a través de la odometría (t_x, t_y, θ), y H se puede estimar linealmente siguiendo un mínimo de cuatro pares de puntos entre las imágenes $(\boldsymbol{x}^i, \boldsymbol{x}'^i)$. Así pues, las únicas incógnitas de la ecuación 3.27 son los nueve elementos de la homografía de proyección buscada, $\tilde{P}$, con sólo ocho grados de libertad por la homogeneidad. Es importante resaltar que, en realidad, la igualdad anterior es cierta salvo factor de escala. Para resolverla, por tanto, conviene normalizar con anterioridad las matrices conocidas $\tilde{O}$ y H para que tengan el mismo determinante, de forma que nos quede una igualdad ordinaria.

3.6.2. Posición relativa general

Una sola pareja de imágenes proporciona nueve restricciones (una por cada uno de los 3×3 elementos de cada lado de la igualdad 3.27). En principio, pues, podría pensarse que las nueve incógnitas de $\tilde{\mathtt{P}}$ se resuelven de modo único con sólo dos imágenes, tomadas antes y después de un movimiento de odometría conocida. Sin embargo, de nuevo el espacio nulo de la matriz de coeficientes del sistema resultante es de rango tres, en lugar del rango uno que correspondería a una solución única. Los dos grados de libertad adicionales provienen del conmutador odométrico explicado en la sección 3.4.2. En efecto, dada una odometría planar $\tilde{\mathtt{O}}$, siempre existe una familia de conmutadores odométricos $\tilde{\mathtt{O}}_c$ (análogos a los conmutadores $\mathtt{O}_c$ de la ecuación 3.19, pero de dimensión 3×3 por la eliminación de la fila y columna terceras) tales que $\tilde{\mathtt{O}} = \tilde{\mathtt{O}}_c^{-1}\tilde{\mathtt{O}}\tilde{\mathtt{O}}_c$. Así, dada una homografía de proyección $\tilde{\mathtt{P}}^*$ solución de 3.27, es claro que la matriz $\tilde{\mathtt{P}}'^* = \tilde{\mathtt{P}}^*\tilde{\mathtt{O}}_c$ es también una solución válida, puesto que $\tilde{\mathtt{O}}_c\tilde{\mathtt{O}} = \tilde{\mathtt{O}}\tilde{\mathtt{O}}_c$, y entonces:

$$\mathtt{H}\tilde{\mathtt{P}}^* = \tilde{\mathtt{P}}^*\tilde{\mathtt{O}} \quad \Leftrightarrow \quad \mathtt{H}\tilde{\mathtt{P}}^*\tilde{\mathtt{O}}_c = \tilde{\mathtt{P}}^*\tilde{\mathtt{O}}\tilde{\mathtt{O}}_c \quad \Leftrightarrow \quad \mathtt{H}\tilde{\mathtt{P}}^*\tilde{\mathtt{O}}_c = \tilde{\mathtt{P}}^*\tilde{\mathtt{O}}_c\tilde{\mathtt{O}} \quad \Leftrightarrow \quad \mathtt{H}\tilde{\mathtt{P}}'^* = \tilde{\mathtt{P}}'^*\tilde{\mathtt{O}}$$

De modo que con un solo par de imágenes la solución para la homografía de proyección (y por tanto la posible reconstrucción de puntos del suelo a través de su inversa) posee aún una ambigüedad similar, introducida por la forma del conmutador $\mathtt{O}_c$. Es decir, a pesar del conocimiento de la odometría, si se permite una posición general de la cámara respecto al móvil, se hará necesario utilizar una tercera imagen para eliminar completamente la ambigüedad. Ésta debe ser tomada desde un lugar cuya odometría no tenga el mismo punto fijo que el primer movimiento, para eliminar la posibilidad de un conmutador común, tal y como se discutió en la sección 3.4.2. Con esa tercera imagen, $\tilde{\mathtt{P}}^*$ queda determinada de modo único. Como siempre, si tenemos más imágenes siempre podremos obtener una solución de mínimo error cuadrático, aprovechando la sobredeterminación.

Una vez conocida $\tilde{\mathtt{P}}^*$, si no se imponen restricciones adicionales sobre la $\mathtt{K}$, naturalmente, no se pueden calcular todos los parámetros intrínsecos y extrínsecos de la calibración, puesto que éstos se acoplan entre sí de manera en principio indeterminada para dar lugar a una misma homografía suelo-imagen. No obstante, aún quedaría la posibilidad de hacer alguna de las simplificaciones habituales, como píxel cuadrado, o punto principal conocido, por ejemplo, para obtener la matriz de proyección $\mathtt{P}$ completa. En estos casos se puede calibrar utilizando técnicas similares a las descritas en el apartado 3.3.2. Por ejemplo, examinando la ecuación 3.4, y sabiendo que $\tilde{\mathtt{P}}$ es la resultante de eliminar la tercera columna de $\mathtt{P}$, se tiene:

$$\tilde{\mathtt{P}} = \mathtt{KR}\begin{pmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & -z_c \end{pmatrix} \tag{3.28}$$

Utilizando la imagen de la cónica absoluta $\boldsymbol{\omega} = \mathtt{K}^{-\top}\mathtt{K}^{-1}$, se puede comprobar lo siguiente:

$$\tilde{\mathtt{P}}^\top \boldsymbol{\omega} \tilde{\mathtt{P}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_c & -y_c & -z_c \end{pmatrix} \mathtt{R}^\top \mathtt{K}^\top \boldsymbol{\omega} \mathtt{K} \mathtt{R} \begin{pmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & -z_c \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_c & -y_c & -z_c \end{pmatrix} \begin{pmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & -z_c \end{pmatrix} = \begin{pmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ -x_c & -y_c & x_c^2 + y_c^2 + z_c^2 \end{pmatrix} = \mathtt{B} \tag{3.29}$$

Aún sin tener conocimiento alguno del vector (x_c, y_c, z_c), la ecuación anterior introduce dos restricciones independientes sobre $\boldsymbol{\omega}$ a través de la matriz simétrica $\mathtt{B} = \tilde{\mathtt{P}}^\top \boldsymbol{\omega} \tilde{\mathtt{P}}$, a saber:

$$\begin{cases} b_{1,1} = b_{2,2} \\ b_{1,2} = 0 \end{cases} \tag{3.30}$$

Nótese que no podemos hacer directamente $b_{1,1} = 1$ por el factor desconocido de homogeneidad. La calibración, por tanto, quedará perfectamente determinada si permitimos sólo dos grados de libertad sobre $\boldsymbol{\omega}$. Esto se puede conseguir, por ejemplo, si se supone sesgo $s = 0$ y punto principal conocido, aunque haya distintas distancias focales f_x y f_y. En esas condiciones, $\boldsymbol{\omega} = \text{diag}(\frac{1}{f_x^2}, \frac{1}{f_y^2}, 1)$, y puede resolverse linealmente en $\frac{1}{f_x^2}$ y $\frac{1}{f_y^2}$, para después obtener las correspondientes focales. Son también posibles otras combinaciones con sólo dos incógnitas, como píxel cuadrado y una de las coordenadas o_x u o_y conocidas, por ejemplo.

Una vez conocida la $\mathtt{K}$, es sencillo calcular el resto de la calibración explícitamente, de modo lineal. Simplemente premultiplicamos $\tilde{\mathtt{P}}$ por $\mathtt{K}^{-1}$, para obtener la matriz $\mathtt{C}$ siguiente:

$$\mathtt{C} = \mathtt{K}^{-1} \tilde{\mathtt{P}} = k \mathtt{R} \begin{pmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & -z_c \end{pmatrix} \tag{3.31}$$

Donde de nuevo se extrae explícitamente la constante k de homogeneidad, puesto que nos interesarán los valores exactos de x_c, y_c y z_c. Normalizamos entonces $\mathtt{C}$ dividiendo por la norma de cualquiera de las dos primeras columnas, de modo que los vectores columna de la rotación $\mathtt{R}$ se pueden obtener directamente partiendo de las dos primeras columnas de $\mathtt{C}$ ($\boldsymbol{c}^1$ y $\boldsymbol{c}^2$) de la siguiente forma:

$$\mathtt{R} = (\boldsymbol{c}^1 \mid \boldsymbol{c}^2 \mid \boldsymbol{c}^1 \times \boldsymbol{c}^2) \tag{3.32}$$

Finalmente, el desfase $(x_c, y_c, z_c)^\top$ entre el robot y el centro óptico de la cámara se puede obtener simplemente como la tercera columna (d^3) de la siguiente matriz $\mathtt{D}$ (haciendo uso de que $\mathtt{R}^{-1} = \mathtt{R}^\top$):

$$\mathtt{D} = \mathtt{R}^\top \mathtt{K}^{-1} \tilde{\mathtt{P}} = \mathtt{R}^\top \mathtt{C} \tag{3.33}$$

ENTRADA:
- Homografías interimagen inducidas por el suelo entre las imágenes primera y segunda ($\mathtt{H}_2$) y primera y tercera $\mathtt{H}_3$, respectivamente.
- Odometrías planares correspondientes (t_x^2, t_y^2, θ^2) y (t_x^3, t_y^3, θ^3). Por simplicidad, y sin pérdida de generalidad, supondremos que la primera imagen se toma desde la posición inicial, es decir, $(t_x^1, t_y^1, \theta^1) = (0, 0, 0)$.

SALIDA:
- Homografía de proyección suelo-imagen $\tilde{\mathtt{P}}$.
- Bajo la suposición de sesgo y punto principal nulos, matriz de cámara $\mathtt{P}$.

ALGORITMO:

Cálculo de la homografía suelo-imagen:

- Resolver el sistema de ecuaciones resultante de utilizar la ecuación 3.27 simultáneamente sobre las homografías interimagen $\mathtt{H}_i$ y sus respectivas odometrías $\tilde{\mathtt{O}}_i$, con $i = 2, 3$. La solución $\tilde{\mathtt{P}}$ es la homografía suelo-imagen buscada. No olvidar normalizar previamente las matrices $\tilde{\mathtt{O}}_i$ y $\mathtt{H}_i$ para que tengan igual determinante.

Cálculo de la matriz de proyección completa:

Matriz de intrínsecos $\mathtt{K}$:

$/ Suponemos que $(\mathtt{s}, o_x, o_y) = (0, 0, 0)$, es decir, $\mathtt{K} = diag(f_x, f_y, 1)$. $*/$*

- Construir la matriz $\mathtt{B}$ dada en 3.29, utilizando la solución $\tilde{\mathtt{P}}$ calculada en el paso anterior, con $\boldsymbol{\omega} = \text{diag}(\frac{1}{f_x^2}, \frac{1}{f_y^2}, 1)$.
- Resolver el sistema 3.30, lineal sobre las incógnitas $\frac{1}{f_x^2}$ y $\frac{1}{f_y^2}$, y determinar las focales en ambas direcciones como las correspondientes inversas de las raíces cuadradas.
- Hacer $\mathtt{K} = \text{diag}(f_x, f_y, 1)$.

Matriz de rotación $\mathtt{R}$:

- Construir la matriz $\mathtt{C}$ dada en 3.31, a partir de $\mathtt{K}^{-1}$ y $\tilde{\mathtt{P}}$.
- Normalizar la matriz $\mathtt{C}$ para que las dos primeras columnas tengan norma unidad.
- Calcular $\mathtt{R}$ a partir de $\mathtt{C}$ utilizando 3.32.

Traslación $(x_c, y_c, z_c)^\top$:

- Construir la matriz $\mathtt{D}$ a partir de $\mathtt{R}$, y $\mathtt{C}$ utilizando 3.33.
- El vector $(x_c, y_c, z_c)^\top$ se corresponde con la tercera columna de la matriz $\mathtt{D}$ (3.34).

Matriz de proyección $\mathtt{P}$:

- Usar los valores de $\mathtt{K}$, $\mathtt{R}$ y $(x_c, y_c, z_c)^\top$ calculados para determinar $\mathtt{P}$, utilizando 3.4 y 3.7.

Algoritmo 3.3: *Autocalibración a partir de tres imágenes, sobre las que se estiman las correspondientes homografías inducidas por el suelo, conociendo la odometría de los movimientos planares. La primera parte del procedimiento estima de modo único la homografía de proyección $\tilde{\mathtt{P}}$, sin restricciones de calibración de ningún tipo. La segunda, haciendo uso de la restricción $(\mathtt{s}, o_x, o_y) = (0, 0, 0)$, calcula completamente la matriz de proyección P, y su descomposición explícita en la forma dada en 3.4.*

$$(x_c, y_c, z_c)^\top = d^3 \tag{3.34}$$

El algoritmo 3.3 resume el procedimiento completo de autocalibración a partir de tres imágenes con odometría conocida, entre las cuales se conocen la homografías inducidas por el plano del suelo. Dichas homografías se pueden estimar, por ejemplo, detectando y siguiendo al menos cuatro puntos del suelo, o bien cuatro líneas (en este caso estimando la homografía traspuesta inversa), siempre de modo lineal. La primera parte del procedimiento describe la forma de calcular la homografía del suelo, sin restricciones de ningún tipo sobre $\mathtt{K}$. En la

segunda se sigue la simplificación de sesgo cero y punto principal conocido para realizar la calibración completa. El procedimiento es fácilmente extensible para secuencias de mayor número de imágenes resolviendo los sistemas lineales sobredeterminados resultantes.

3.6.3. Brazo nulo

La necesidad de tres imágenes del procedimiento 3.3 viene impuesta, como hemos visto, por el problema de la conmutación odométrica. Cabe preguntarse si realizando algún tipo de simplificación en la posición de la cámara respecto al móvil podríamos reducir el número de imágenes requeridas a la entrada. La respuesta a esta pregunta es afirmativa, si se realiza la suposición que en el apartado 3.2.3 denominamos de brazo nulo, pero con alguna consideración adicional. En las condiciones $x_c = y_c = 0$, la odometría (t_x, t_y, θ) conocida se corresponde también con el movimiento realizado por la proyección vertical del centro óptico de la cámara sobre el suelo. No existe, por tanto, ningún tipo de traslación "parásita" asociada a la existencia de brazo efectivo en la posición de la cámara, y por tanto la línea base entre las cámaras es equivalente al módulo de la traslación. Aún así, como veremos, si no se realiza alguna otra simplificación adicional sobre la $\mathtt{K}$, el problema seguiría sin poder resolverse. En este apartado describimos un posible procedimiento para realizar la calibración con sólo dos imágenes, junto con las condiciones necesarias para poder aplicarlo.

Como ya hemos comentado, la matriz de coeficientes del sistema de ecuaciones planteado en 3.27 tiene un espacio nulo de rango tres. Las dos dimensiones extra son debidas a la familia de matrices de conmutación $\mathtt{O}_c$, inherentes a cada movimiento odométrico. En principio, pues, cualquier matriz de la forma $\tilde{\mathtt{P}}^*\mathtt{O}_c$ es una solución válida para la homografía del suelo. Dado que sobre toda matriz cuadrada, de tamaño 3×3, siempre puede realizarse una descomposición RQ como la comentada en la nota al pie 5, está claro que si no imponemos ninguna restricción sobre la matriz de calibración $\mathtt{K}$, siempre habrá una calibración compatible con cualquiera de las soluciones de la familia $\tilde{\mathtt{P}}^*\mathtt{O}_c$, incluso estando en la suposición de brazo nulo. Esto se puede comprobar de modo muy sencillo tomando una altura $z_c = -1$, por ejemplo, y la parte triangular de la descomposición RQ como matriz $\mathtt{K}$, para obtener una forma de $\tilde{\mathtt{P}}$ como la introducida en 3.28, para $x_c = y_c = 0$.

Parece por tanto claro que si queremos resolver el problema con un solo movimiento (dos imágenes), tenemos que partir de algún tipo de suposición previa sobre la $\mathtt{K}$. Por consistencia y aplicabilidad de la simplificación, volveremos a suponer que $\mathtt{K} = \mathrm{diag}(f_x, f_y, 1)$. En este caso, siempre podremos calcular los valores de los intrínsecos f_x y f_y utilizando la misma técnica empleada anteriormente, aplicando las ecuaciones 3.29 y 3.30 a cualquier homografía de proyección $\tilde{\mathtt{P}}_{en}$ perteneciente al espacio nulo de las soluciones de 3.27. Se puede comprobar que, afortunadamente, todas las matrices pertenecientes a dicho subespacio comparten la misma solución para las focales f_x y f_y. Una opción sencilla consiste en tomar cualquiera de los tres vectores que definen dicho espacio nulo, o incluso una combinación lineal de éstos.

Una vez realizada la calibración intrínseca procedemos al cálculo de la rotación $\mathtt{R}$ compatible, siguiendo un procedimiento de nuevo idéntico al descrito en la sección anterior, usando 3.31 y 3.32. Se genera así una cámara virtual, consistente con los parámetros de nuestra cámara, pero con un brazo $(x_c, y_c)^\top$ arbitrario. En este momento, en lugar de calcular la traslación correspondiente procederemos a construir una nueva matriz de proyección $\tilde{\mathtt{P}}'_{en}$, partiendo de 3.28, pero forzando los valores $x_c = y_c = 0$, tal y como corresponde a la suposición de brazo nulo. Tanto la matriz construida como la homografía buscada deberán proyectar el origen de S_{rob} (es decir, el punto de coordenadas homogéneas $(0, 0, 1)^\top$) al mismo punto de la imagen. El valor escogido para z_c es en realidad irrelevante, puesto que el único uso que daremos a esta homografía intermedia es el de averiguar la imagen del origen de coordenadas, y dicha proyección es independiente de la altura z_c. Por comodidad, podemos utilizar $z_c = 1$, con lo que la matriz resulta ser:

$$\tilde{\mathtt{P}}'_{en} = \mathtt{K}(\boldsymbol{c}^1 \mid \boldsymbol{c}^2 \mid -\boldsymbol{c}^1 \times \boldsymbol{c}^2) \tag{3.35}$$

Y, por tanto, la proyección del vector $(0, 0, 1)^\top$, simplemente $\mathtt{K} \cdot (\boldsymbol{c}^1 \times \boldsymbol{c}^2)$ (donde el signo se elimina por la homogeneidad del vector). El único paso restante consiste en volver sobre el sistema de ecuaciones original 3.27 para añadir la nueva restricción de la proyección del origen de coordenadas, es decir:

$$\tilde{\mathtt{P}} \cdot (0, 0, 1)^\top = \mathtt{K} \cdot (\boldsymbol{c}^1 \times \boldsymbol{c}^2) \tag{3.36}$$

Siendo c^1 y c^2 las columnas primera y segunda de la matriz definida en 3.31 (sobre $\tilde{\mathtt{P}}_{en}$). Por la homogeneidad de la ecuación anterior, conviene añadirla al sistema de ecuaciones con la forma acostumbrada de anulación del producto vectorial:

$$\tilde{\mathtt{P}} \cdot (0, 0, 1)^\top \times \mathtt{K} \cdot (\boldsymbol{c}^1 \times \boldsymbol{c}^2) = \mathbf{0} \tag{3.37}$$

Las dos igualdades independientes de la ecuación anterior eliminan la ambigüedad de 3.27, determinando de modo único la solución. El algoritmo 3.4 resume el procedimiento propuesto.

3.7. Solución bidimensional analítica

Hasta ahora, todos los métodos que hemos estudiado se basan en la determinación directa de las matrices de interés, es decir, la homografía suelo-imagen $\tilde{\mathtt{P}}$ y la correspondiente matriz de proyección $\mathtt{P}$, trabajando simultáneamente sobre todos sus elementos, y siempre utilizando técnicas de naturaleza lineal para hallar las soluciones. En lo que resta de capítulo, no obstante, abordaremos el problema de la autocalibración odométrica desde una aproximación completamente distinta, basada en un estudio analítico de dichas matrices. En este segundo

ENTRADA:
- Homografía interimagen H inducida por el suelo entre las imágenes del par.
- Odometría planar correspondiente al movimiento, (t_x, t_y, θ) (sin pérdida de generalidad, supondremos que la primera imagen se toma desde la posición inicial $(t_x^1, t_y^1, \theta^1) = (0, 0, 0)$).

SALIDA:

/ Bajo la suposición de brazo, sesgo y punto principal nulos: */*
- Homografía de proyección suelo-imagen $\tilde{\mathtt{P}}$.
- Matriz de cámara P.

ALGORITMO:

Cálculo de la homografía suelo-imagen:
- Plantear el sistema de ecuaciones 3.27 sobre H y $\tilde{\mathtt{0}}$, estando ésta última definida como en 3.17, con la fila y columna terceras eliminadas. Sea $\tilde{\mathtt{P}}_{en}$ cualquiera de las soluciones pertenecientes al espacio nulo (de dimensión tres) de la matriz de coeficientes resultante.
- Calcular f_x y f_y como en el apartado "*Matriz de intrínsecos* K" del algoritmo 3.3, utilizando la solución $\tilde{\mathtt{P}}_{en}$ obtenida en el paso anterior.
- Calcular C a partir de K y $\tilde{\mathtt{P}}_{en}$ utilizando 3.31.
- Añadir la ecuación 3.37 al sistema 3.27 para volver a resolver $\tilde{\mathtt{P}}$, esta vez de modo único.

Cálculo de la matriz de proyección completa:

Cálculo de la matriz de rotación R:
- Proceder como en el paso correspondiente del algoritmo 3.3.

Cálculo de la traslación (x_c, y_c, z_c)**:** */* Sólo interesa la altura z_c; la x_c e y_c deben salir 0. */*
- Ídem.

Cálculo de P:
- Ídem.

Algoritmo 3.4: *Autocalibración a partir de dos imágenes partiendo de la correspondiente homografía inducida por el suelo, conociendo la odometría planar del movimiento realizado y suponiendo brazo nulo ($x_c = y_c = 0$) y matriz de calibración diagonal ($s = o_x = o_y = 0$), para poder determinar de modo unívoco la homografía de proyección* $\tilde{\mathtt{P}}$*. Para calcular la matriz de proyección completa* P *a partir de* $\tilde{\mathtt{P}}$*, se procede de modo idéntico a la segunda parte del algoritmo 3.3.*

enfoque utilizaremos directamente las expresiones simbólicas resultantes, en función de los parámetros de calibración de todas las entidades intervinientes. Partiendo de dichas expresiones intentaremos una estimación directa, variable a variable, de los valores de la matriz de calibración, los ángulos de la rotación y el vector de traslación entre la cámara y el robot. Una vez resueltas las ecuaciones para estos valores se podrá reconstruir la matriz P completa a partir de sus parámetros, usando las ecuaciones 3.4, 3.5, 3.6 y 3.7.

Como se verá a lo largo del desarrollo, todos los procedimientos propuestos obtendrán los parámetros de la calibración a partir de expresiones funcionales con los siguientes tipos de variables de entrada:

1. Coordenadas de imagen de puntos o rectas en distintos instantes de tiempo, para los que la posición del robot es conocida a través de la odometría.

2. Valores de rotación y traslación del vehículo medidos sobre el plano del suelo. En este caso, al contrario de lo que sucedía en los métodos lineales generales, la información de rotación sí que resultará imprescindible. La razón es que ahora no habrá forma de

determinar θ a partir de los valores propios de la homografía interimagen, ya que ésta última no podría computarse directamente a partir de las mínimas configuraciones de entrada de los procedimientos propuestos (como mucho una recta y un punto seguidos entre las imágenes).

3. Parámetros de la calibración (intrínsecos y extrínsecos) obtenidos en pasos previos del procedimiento.

Naturalmente, la disminución en los requerimientos de entrada será sólo posible bajo ciertas suposiciones, que permiten hacer viable y al mismo tiempo eficiente el enfoque analítico. A continuación resumimos esta serie de simplificaciones, algo más restrictivas que las realizadas hasta ahora, tanto en lo relativo a la posición de la cámara en el robot como sobre la forma de la matriz `K`.

3.7.1. Simplificaciones necesarias

Utilizando la terminología introducida en el apartado 3.2.3, éstas serán las simplificaciones empleadas:

- Respecto a la cámara, consideraremos los píxeles perfectamente cuadrados, esto es, $s = 0$ y $f_x = f_y$. También supondremos que el punto principal de la imagen de cámara está situado en el mismo centro. Esto equivale a hacer $o_x = o_y = 0$, eligiendo un sistema de coordenadas centrado en el píxel $(c/2, r/2)$, siendo c y r el número de columnas y filas de la imagen, respectivamente. De esta forma, la `K` empleada es la más simple posible, diagonal y con un solo parámetro libre, la focal f (ecuación 3.8).

- Respecto a la posición relativa de la cámara en la plataforma, se supone también que la cámara está montada con el eje X paralelo al plano del suelo. Esto es, que la proyección de la línea en el infinito de dicho plano (el horizonte) aparece en la imagen completamente horizontal. Obsérvese que esto no significa en absoluto que el eje óptico sea paralelo al suelo, puesto que sí se permite que la cámara tenga un ángulo de picado no despreciable, ni que esté alineado con la dirección de avance del robot. Es decir, el ángulo de *roll* $\gamma = 0$, pero α y β, los respectivos ángulos de *tilt* y *pan*, no quedan sujetos a ninguna restricción (ver figura 3.2). Tampoco impondremos, *a priori*, ninguna restricción adicional sobre el vector (x_c, y_c, z_c). La cámara podrá estar, por tanto, colocada a cualquier altura y con cualquier valor de brazo efectivo sobre el eje de giro de la plataforma.

Todas estas suposiciones simplifican enormemente los cálculos y permiten resolver el problema de modo analítico, a través de expresiones funcionales sencillas para cada uno de los parámetros a estimar. A pesar de que el conjunto de restricciones parece limitar mucho el

modelo, en realidad aún se conservan seis de los once grados de libertad de P. Más concretamente, obsérvese que el análisis tendrá que realizarse sobre sólo uno de los cinco parámetros intrínsecos posibles (f), y sobre cinco de los seis extrínsecos (α, β, x_c, y_c y z_c). Como veremos posteriormente, en realidad la segunda de las simplificaciones ($\gamma = 0$) también podrá ser relajada si se puede determinar de alguna manera la inclinación en imagen de la proyección de la línea del horizonte correspondiente al suelo. Usando su posible desalineamiento con el eje X de la imagen, podrían corregirse previamente todas las características de entrada a los algoritmos, que serían perfectamente válidos aún para ángulos de *roll* no despreciables. Volveremos a comentar esta posibilidad en un apartado posterior.

En algunos casos aplicaremos simplificaciones aún más restrictivas, como brazo nulo ($x_c = y_c = 0$), o cámara apuntando hacia adelante ($\beta = 0$). La finalidad es obtener versiones de calibración que, a pesar de ser muy simples, pueden utilizarse en situaciones realistas cometiendo errores despreciables en las reconstrucciones practicadas. El uso de este tipo de simplificaciones adicionales sobre la posición de la cámara en el móvil es además habitual en muchos sistemas monoculares. Como ejemplo, Zhang *et al.* (1993) montan la cámara en la plataforma con el eje óptico paralelo al suelo y el centro óptico en la vertical del centro del robot, un modelo aún más restringido que el nuestro y, a pesar de ello, muy útil en cierto tipo de tareas de navegación.

3.7.2. Ventajas del modelado analítico

El modelado analítico tiene una serie de potenciales ventajas, entre las que podríamos destacar las siguientes:

- La primera, y quizá más importante, es la reducción del número de entidades de imagen (puntos o líneas) a seguir para realizar la calibración. Veremos como, bajo las suposiciones comentadas, podremos resolver el caso general siguiendo simplemente una línea con un punto destacado (semirrecta) entre tres imágenes, frente a los cuatro puntos en tres imágenes de la solución lineal general. Del mismo modo, a medida que vayamos aumentando el número de simplificaciones, iremos obteniendo sucesivos algoritmos con menos requerimientos. Por ejemplo, bajo ciertas suposiciones adicionales perfectamente asumibles para multitud de aplicaciones de robótica móvil, veremos cómo es posible calibrar aceptablemente el sistema siguiendo un solo punto (o, alternativamente, un solo segmento) entre dos imágenes tomadas desde posiciones distintas, para las cuales la posición del robot portador de la cámara es conocida a través de la odometría. Esta propiedad permite la calibración incluso a partir de imágenes con información mínima, relativamente habituales en entornos estructurados con pocos objetos y, por tanto, poca disponibilidad de características fiables para realizar el *tracking*. Consúltese, por ejemplo, la figura 5.5 (página 203) del capítulo 5 para ver ejemplos de imágenes típicas con las que se enfrenta nuestro sistema, y en las que se aprecia perfectamente este problema.

- Al obtenerse expresiones cerradas para cada uno de los parámetros, los algoritmos de calibración quedarán extraordinariamente sencillos y eficientes, consistentes únicamente en una pequeña secuencia de cálculos directos a aplicar sobre las variables de entrada. Los requisitos computacionales, pues, son mínimos, y por tanto ideales para ser utilizados bajo condiciones de tiempo real como las exigidas por la navegación visual.

- También como consecuencia del modelado analítico explícito, se pueden realizar sencillos análisis de sensibilidad de las funciones obtenidas frente a errores en las variables de entrada.

- Otra propiedad interesante, en los casos en los que se obtengan varias expresiones alternativas para el cálculo de algún valor, será el hecho de poder seleccionar aquella que esté mejor condicionada numéricamente, a la vista de los valores concretos de las entradas.

- Por último, el tipo de procedimiento utilizado puede tener también alguna ventaja añadida desde el punto de vista de la robustez en el cálculo. Esta robustez puede conseguirse aprovechando el hecho de que los procedimientos obtenidos necesitan seguir muy pocas características (un punto, una recta, o una semirrecta, por ejemplo). Así, en el caso de que haya varias entidades de entrada candidatas, se podrá aplicar el procedimiento de cálculo a todas y cada una de ellas, para hacer posteriormente una eliminación de aquellos valores de salida no compatibles con la solución obtenida por la mayoría, al estilo de la técnica RANSAC (Fischler y Bolles, 1981). Se evita así el tener que recurrir a minimizaciones globales mediante técnicas de mínimos cuadrados (muy sensibles a *outliers* en el tracking de puntos o segmentos), y en donde las técnicas estadísticamente robustas, como el propio RANSAC, podrían en algunos casos ser computacionalmente demasiado costosas, por el aumento en el número de entidades mínimo para el cálculo de un modelo candidato a conseguir el consenso. Por ejemplo, si siguiendo una sola línea del suelo tras una rotación podemos obtener la focal de la cámara, y somos capaces de seguir, digamos, seis o siete segmentos, podemos obtener a partir de éstos otras tantas focales distintas, para después quedarnos con la mediana de todos los valores obtenidos.

3.7.3. Homografía suelo-imagen parametrizada

Como base de las técnicas propuestas en este apartado, partiremos de sendas expresiones analíticas para la matriz de cámara $\mathtt{P}$ y su correspondiente homografía suelo-imagen $\tilde{\mathtt{P}}$. Si partimos de las ecuaciones 3.4, 3.6, 3.7 y 3.8 y utilizamos las simplificaciones ya comentadas

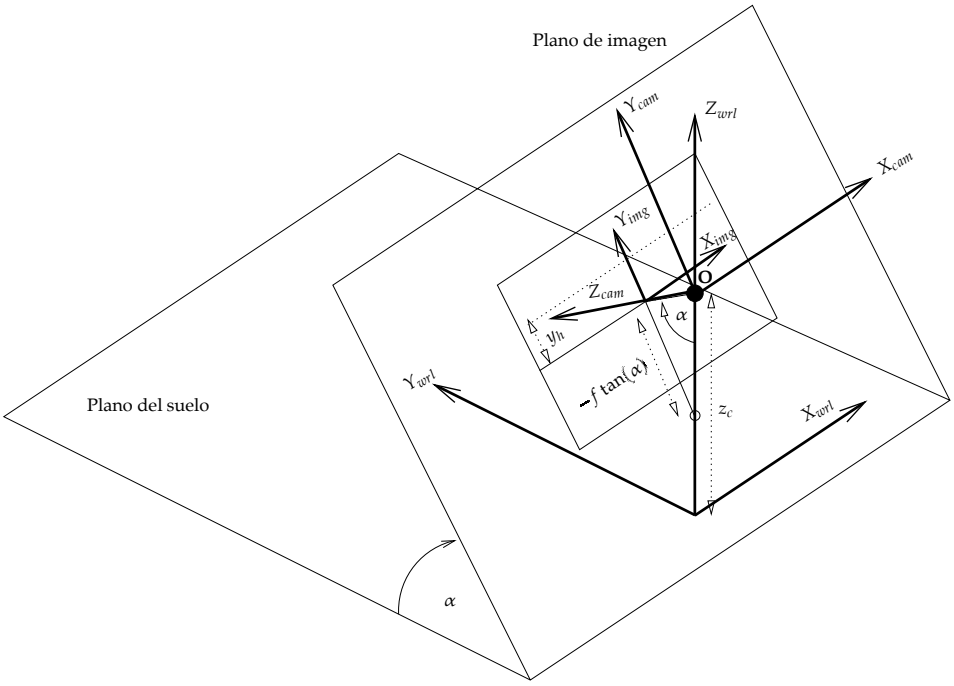


Figura 3.7: *Definición del sistema de coordenadas del mundo respecto a la posición de la cámara. El punto sólido indica la posición del centro óptico de la cámara. El punto hueco es la proyección del origen del mundo, situado exactamente en el punto* $(0, -f\tan(\alpha))$*, y el horizonte está a una altura* $y_h = f\cot(\alpha)$ *(ambos en coordenadas de imagen).*

de $f_x = f_y = f$ y $s = o_x = o_y = \gamma = 0$, llegamos a la siguiente expresión para $\mathtt{P}$:

$$\mathtt{P} = \begin{pmatrix} -f\cos(\beta) & -f\sin(\beta) & 0 & f\,x_c\cos(\beta) + f\,y_c\sin(\beta) \\ f\cos(\alpha)\sin(\beta) & -f\cos(\alpha)\cos(\beta) & -f\sin(\alpha) & f\,y_c\cos(\alpha)\cos(\beta) + f\,z_c\sin(\alpha) - f\,x_c\cos(\alpha)\sin(\beta) \\ \sin(\alpha)\sin(\beta) & -\cos(\beta)\sin(\alpha) & \cos(\alpha) & -z_c\cos(\alpha) + y_c\cos(\beta)\sin(\alpha) - x_c\sin(\alpha)\sin(\beta) \end{pmatrix} \quad (3.38)$$

Por conveniencia, sin embargo, factorizaremos en dicha expresión la influencia de los desfases extrínsecos β y $(x_c, y_c)^\top$ entre los sistemas de la cámara y el robot, aislándolos del resto de parámetros. Para ello, creamos un sistema de coordenadas alternativo para el mundo, S_{wrl}, con origen justo en la vertical sobre el suelo del centro óptico de la cámara $\boldsymbol{O}$, eje X_{wrl} paralelo al eje X_{cam} de la cámara, eje Y_{wrl} simplemente en vertical a X_{wrl} y contenido también en el plano del suelo, y eje Z_{wrl}, por tanto, perpendicular al suelo y conteniendo a $\boldsymbol{O}$. Nótese que dicho sistema no tiene por qué coincidir con el sistema de coordenadas sobre el que se dan las órdenes de movimiento y se toma la odometría, S_{rob}, definido en la sección 3.2. Finalmente, S_{img} será el sistema de coordenadas bidimensional correspondiente al plano de imagen, sobre el que se tomarán los puntos y rectas medidos en unidades de píxel. La figura 3.7 resume esta situación, donde, por claridad, se ha eliminado el sistema S_{rob}, relacionado con S_{wrl} través la siguiente matriz de transformación:

$$\mathtt{P}_{rw} = \begin{pmatrix} \cos(-\beta) & -\sin(-\beta) & 0 & 0 \\ \sin(-\beta) & \cos(-\beta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -x_c \\ 0 & 1 & 0 & -y_c \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.39}$$

El subíndice *rw* nos recuerda que el papel de $\mathtt{P}_{rw}$ es transformar puntos del mundo entre coordenadas relativas a los sistemas S_{rob} y S_{wrl}. Se puede comprobar que esta matriz es equivalente a esta otra expresión:

$$\mathtt{P}_{rw} = \begin{pmatrix} \cos(\beta) & -\sin(\beta) & 0 & x_c \\ \sin(\beta) & \cos(\beta) & 0 & y_c \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1} \tag{3.40}$$

En estas condiciones, podemos factorizar la matriz de proyección así:

$$\mathtt{P} = \mathtt{P}_{wi}\mathtt{P}_{rw} \tag{3.41}$$

Donde la matriz de proyección simplificada $\mathtt{P}_{wi}$ tiene la siguiente forma:

$$\mathtt{P}_{wi} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f\,\cos(\alpha) & f\,\sin(\alpha) & -f\,z_c\,\sin(\alpha) \\ 0 & \sin(\alpha) & -\cos(\alpha) & z_c\,\cos(\alpha) \end{pmatrix} \tag{3.42}$$

Por conveniencia, toda la matriz ha sido multiplicada por -1, aprovechando la condición de homogeneidad. La homografía $\tilde{\mathtt{P}}_{wi}$ inducida desde el suelo (plano $Z_{wrl} = 0$) en el plano de imagen será, pues, la matriz resultante de eliminar la tercera columna de $\mathtt{P}$:

$$\tilde{\mathtt{P}}_{wi} = \begin{pmatrix} f & 0 & 0 \\ 0 & f\,\cos(\alpha) & -f\,z_c\,\sin(\alpha) \\ 0 & \sin(\alpha) & z_c\,\cos(\alpha) \end{pmatrix} \tag{3.43}$$

Puesto que las homografías pueden multiplicarse por un factor arbitrario no nulo sin alterar la transformación proyectiva resultante, una forma alternativa de esta homografía se puede obtener dividiendo todos los elementos de la matriz por $\sin(\alpha)$, suponiendo que $\alpha \neq 0$ (esto es, que la cámara no está orientada en perpendicular hacia el suelo):

$$\tilde{\mathtt{P}}_{wi} = \begin{pmatrix} \frac{f}{\sin(\alpha)} & 0 & 0 \\ 0 & \frac{f}{\tan(\alpha)} & -f\,z_c \\ 0 & 1 & \frac{z_c}{\tan(\alpha)} \end{pmatrix} \tag{3.44}$$

Llamando y_h a la altura del horizonte en la imagen, y haciendo uso de las siguientes

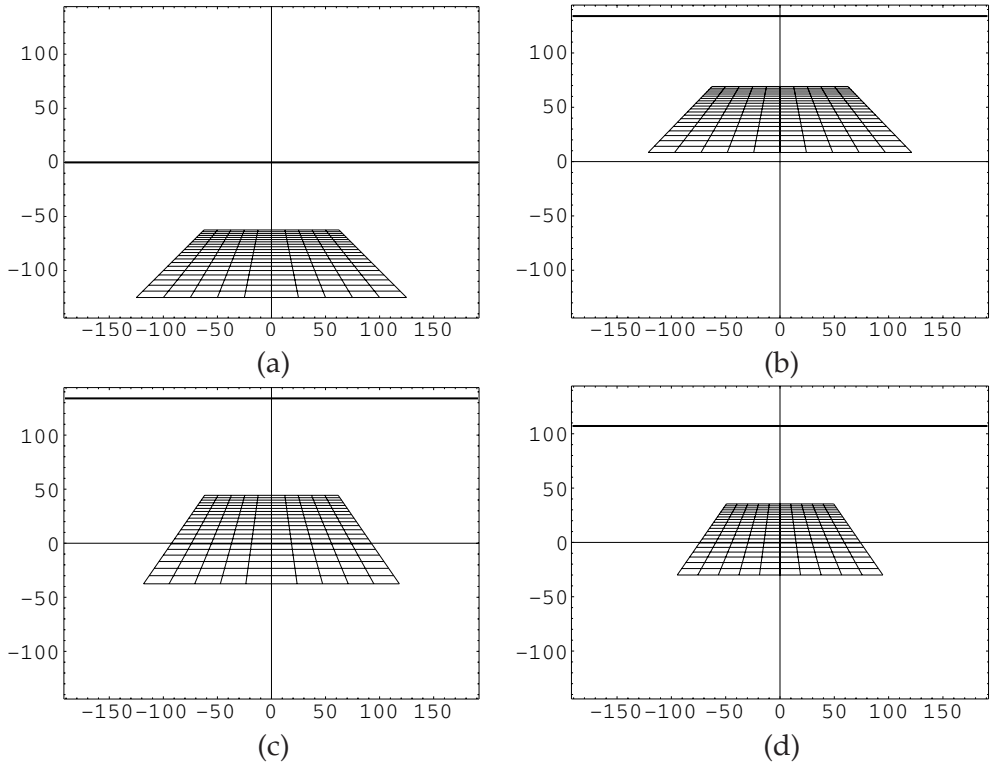


Figura 3.8: *Imágenes de un pasillo de* 10×20 *baldosas de* 20×20 *cm, situado* 420 *cm por delante del eje* X_{wrl} *(justo debajo del centro óptico de la cámara). En todas las imágenes la línea del horizonte aparece con grosor doble. De izquierda a derecha y de arriba a abajo: a)* $f = 500$ *píxeles,* $\alpha = \pi/2$ *y* $z_c = 100$ *cm (*$y_h = f/\tan(\alpha) = 0$*). b) Cambio del ángulo de* tilt *a* $\alpha = 5\,\pi/12$ *(*$y_h = f/\tan(\alpha) = 133.97$*). c) Cambio de la altura a* $z_c = 140$ *cm (*$y_h = 133.97$*, no cambia). d) Cambio de la focal a* $f = 400$ *píxeles* $y_h = f/\tan(\alpha) = 107.18$*.*

equivalencias, que pueden deducirse directamente de la figura 3.7,

$$y_h = f\,\cot(\alpha) = \frac{f}{\tan(\alpha)} \tag{3.45}$$

$$\frac{f}{\sin(\alpha)} = \sqrt{f^2 + y_h^2} \tag{3.46}$$

puede llegarse también a esta otra forma de la homografía:

$$\tilde{\mathrm{P}}_{wi} = \begin{pmatrix} \sqrt{f^2 + y_h^2} & 0 & 0 \\ 0 & y_h & -f\,z_c \\ 0 & 1 & \frac{z_c\,y_h}{f} \end{pmatrix} \tag{3.47}$$

En la figura 3.8 se muestran distintas imágenes de un pasillo virtual generadas con esta parametrización, para distintos valores de f, α y z_c, en tamaños de 288×384 píxeles (PAL a media resolución). Obsérvese que el origen del sistema de coordenadas de la imagen se coloca justo en el centro de la imagen, en donde se supone *a priori* que está el centro óptico.

La homografía $\tilde{\mathtt{P}}_{wi}$ transfiere puntos del plano del suelo, en coordenadas del sistema S_{wrl}, al de la imagen, en coordenadas de S_{img}. Pero lo que realmente nos interesa es obtener la homografía de transferencia directa entre el plano del suelo del sistema S_{rob} y el plano de imagen, puesto que el control y la odometría del robot se realizan sobre dicho sistema de coordenadas. Como es posible que haya un desfase relativo (x_c, y_c, β) entre los sistemas S_{wrl} y S_{rob}, tendremos que utilizar una transformación euclídea previa, de modo análogo a como hicimos con la cámara en 3.41. Usaremos la homografía adicional $\tilde{\mathtt{P}}_{rw}$, que viene simplemente de eliminar la fila y columna terceras de 3.40, como de costumbre. La expresión completa para la homografía de interés, $\tilde{\mathtt{P}}_{ri}$, que pasa de coordenadas del robot directamente a coordenadas de imagen queda entonces así:

$$\tilde{\mathtt{P}}_{ri} = \tilde{\mathtt{P}}_{wi}\tilde{\mathtt{P}}_{rw} \tag{3.48}$$

La homografía $\tilde{\mathtt{P}}_{ri}$ tiene seis grados de libertad, correspondientes a los parámetros α, f, z_c, x_c, y_c y β (todos independientes entre sí). De ellos, los tres primeros corresponden a $\tilde{\mathtt{P}}_{wi}$, mientras que los otros tres proceden de $\tilde{\mathtt{P}}_{rw}$. Obsérvese que si se utiliza la expresión alternativa 3.47 para $\tilde{\mathtt{P}}_{wi}$, entonces sigue habiendo tres grados de libertad, pero en este caso eliminando α e introduciendo y_h (que son mutuamente dependientes a través de f). Podemos afirmar, por tanto, que las simplificaciones empleadas sirven para eliminar dos de los ocho grados de libertad que tiene una homografía general (nueve elementos menos el factor de escala).

Si, además, asumimos también la hipótesis de brazo nulo, equivalente a tomar $x_c = y_c = 0$, entonces quedarían sólo cuatro grados de libertad para $\tilde{\mathtt{P}}_{ri}$. Otras simplificaciones adicionales, como $\beta = 0$ (para una cámara alineada con la dirección de avance del robot) o $\alpha = \pi/2$ (si suponemos el eje óptico paralelo al suelo), o incluso el conocimiento previo de alguno de los parámetros (como la focal f o la altura del horizonte y_h), podría restringirse aún más a la homografía.

3.7.4. Determinación del horizonte

A las simplificaciones comentadas anteriormente añadiremos en lo que sigue la de que podemos calcular previamente la altura del horizonte y_h por medio de algún procedimiento externo. Este valor no cambia en ninguna de las imágenes tomadas por un robot que experimenta cualquier tipo de movimiento estrictamente planar. Con horizonte conocido, el número de grados de libertad de $\tilde{\mathtt{P}}_{ri}$ queda reducido a sólo cinco, lo que facilitará aún más la autocalibración.

Existen múltiples posibilidades para el cómputo de este horizonte, cada una con sus necesidades de tipo y número de características a seguir entre las imágenes, movimiento de la plataforma o tipo de conocimiento externo empleado. Describimos a continuación un par de ellas, ambas bastante sencillas de aplicar.

- La primera técnica se basa en hallar la intersección de dos líneas paralelas en el plano del suelo. Estas líneas se cortarán siempre en un punto de la imagen con coordenadas $(x, y_h)^\top$, independientemente de su posición y orientación respecto a la cámara. Este tipo de líneas son habituales en los entornos estructurados de interiores que constituyen nuestro dominio de aplicación. Un ejemplo serían las líneas de intersección del suelo y las paredes de un pasillo, los bordes paralelos entre baldosas, etc. Estos segmentos son fácilmente detectables por el procedimiento descrito en el capítulo anterior, por ejemplo utilizando la información de color, y usando sencillas heurísticas de sentido común se puede determinar su condición de paralelismo. En nuestro caso, el ajuste de y_h se hace al ir navegando por un pasillo, utilizando las líneas de intersección de las paredes con el suelo como segmentos paralelos de entrada.

- Una segunda posibilidad consistiría en "fabricar" dos líneas paralelas realizando el seguimiento de un único segmento de la imagen tras realizar una traslación pura, sin giro de la plataforma. Si evitamos movernos en perpendicular o en paralelo a esta recta, entonces el corte del segmento antes y después del movimiento también nos daría y_h. En este caso, el movimiento óptimo para que la estimación no esté mal condicionada es movernos en oblicuo hacia la línea, en un ángulo de unos 45 grados aproximadamente. Obviamente este ángulo de ataque tendrá que decidirse heurísticamente, puesto que se supone que aún no sabemos nada de la calibración, ni por tanto de la orientación exacta del segmento de suelo respecto a nosotros.

Es interesante volver en este momento sobre la simplificación de ángulo de *roll* γ nulo. Si somos capaces de determinar completamente la línea del horizonte $\boldsymbol{l}_h$, aunque ésta presente una cierta inclinación, entonces aún podremos tratar el caso $\gamma \neq 0$. El tratamiento se basará en corregir las características de imagen que forman la entrada de los algoritmos utilizando la inclinación de dicha línea respecto al eje X_{img}, de modo similar a como se rectifica previamente la distorsión radial, por ejemplo (recordemos que $\boldsymbol{l}_h$ es invariante a cualquier movimiento planar). Pero hallar esta línea es relativamente sencillo, simplemente determinando dos puntos distintos de la misma utilizando cualquiera de los procedimientos discutidos en los párrafos anteriores, por ejemplo. Vemos, pues, que en estas condiciones la única restricción restante sería la diagonalidad de la `K` y, al menos en principio, se podrían estimar todos los parámetros extrínsecos correspondientes a la posición general de la cámara respecto al vehículo.

3.7.5. Configuraciones de entrada mínimas

Simplemente razonando en términos de grados de libertad es posible especificar, al menos en teoría, las configuraciones de entrada mínimas que serán necesarias para determinar la homografía suelo-imagen de modo único en presencia de las distintas simplificaciones; esto es, cuántas correspondencias entre características (puntos, líneas o semirrectas) son necesarias

Modelo		**Configuración de entrada**					**Movimientos**	
Simplificaciones adicionales	gdl	#puntos	#rectas	#semirrectas	#vistas	gdl	1^o	2^o
Ninguna	5	-	-	1	3	6	Rot.	Trasl.
$x_c = y_c = 0$	3	-	-	1	2	3	Gen.	-
$x_c = y_c = \beta = 0$	2	1	-	-	2	2	Trasl.	-
$x_c = y_c = \beta = 0$	2	-	1	-	2	2	Gen.	-

Tabla 3.1: *Algunas configuraciones de entrada mínimas en distintas condiciones de modelado y movimientos del robot. En la columna del modelo se expresan las simplificaciones aplicadas (a añadir a las ya mencionadas de píxel cuadrado, punto principal en el centro de la imagen y* roll *nulo) y los grados de libertad que tiene la homografía resultante en cada caso (considerando y_h conocido en todos los casos). En la columna de configuración de la entrada se indican las entidades de imagen a seguir entre vistas, el número de vistas necesarias y los grados de libertad de cada configuración. En la columna de movimientos se expresan las condiciones impuestas sobre cada movimiento para aplicar el algoritmo correspondiente. Dichas restricciones podrán ser de movimiento traslacional puro, rotacional puro o ninguna (movimiento general). El número de movimientos, obviamente, será siempre el número de vistas menos uno.*

en cada caso para desambiguar completamente $\tilde{\mathrm{P}}_{ri}$. Así, considerando que los puntos y las rectas tienen dos grados de libertad, y las semirrectas tres (dos del punto destacado y una de ángulo), hay varias posibilidades de entrada mínima (teórica) para cada situación. En la tabla 3.1 se enumeran algunas de dichas configuraciones, para distintas simplificaciones del modelo, entidades de imagen, número de vistas y tipos de movimiento del robot.

Por supuesto, las configuraciones mostradas en la tabla no son las únicas posibles. Sí que son, sin embargo, bastante representativas, y en torno a ellas trabajaremos en este apartado. El motivo para escoger precisamente éstas es, básicamente, la simplicidad de los respectivos algoritmos obtenidos, así como la aplicabilidad en casos prácticos. En algunos casos se exigen ciertas condiciones a los movimientos del robot (por ejemplo, que sea una rotación o una traslación pura), con el fin de hacer viable y sencilla la aproximación analítica. Sin embargo, esto no debería ser un problema, puesto que, como se argumentó en la introducción, los movimientos del robot sobre el suelo no sólo son medibles, sino también controlables desde el sistema de navegación. Algunos autores ya han aprovechado el hecho de que determinados movimientos restringidos simplifican los requerimientos en los algoritmos de calibración. Como ejemplo típico, Liang y Pears (2002) describen un procedimiento para estimar la homografía del suelo siguiendo solamente dos puntos, en lugar de los cuatro habituales, si se asegura que el movimiento realizado entre las cámaras es una traslación pura.

En otros casos más sencillos, que necesitan sólo dos vistas, se permiten movimientos generales, con una componente rotacional y otra traslacional. En algunos de estos casos es incluso necesario que el movimiento sea general, puesto que una traslación o una rotación aisladas no eliminarían toda la ambigüedad de la configuración, o bien, aunque en teoría la eliminen, complican la obtención de una solución analítica simple. En los estudios que realizaremos a continuación entraremos más en detalle en las particularidades de cada caso.

Para el estudio de todas estas posibilidades utilizaremos el siguiente esquema. Partimos

del caso general de $\tilde{\mathtt{P}}_{ri}$, en el cual hay que determinar los cinco parámetros libres f, x_c, y_c, z_c y β (α quedaría entonces determinado por el conocimiento de y_h y f), resolviéndolo a partir del seguimiento de una semirrecta en tres imágenes, tomadas desde tres posiciones conocidas tras dos movimientos medidos por la odometría. Después iremos aplicando las sucesivas simplificaciones que reducen paulatinamente los requerimientos de la entrada, e incluso daremos, en algunos casos, soluciones alternativas para un mismo modelo de homografía. Para cada algoritmo discutiremos igualmente las posibles secuencias críticas, esto es, aquellas configuraciones de entrada que quedan prohibidas en determinadas condiciones de movimiento, ya que dan lugar a ambigüedad en la solución. En general, el problema de las secuencias críticas en los algoritmos de geometría proyectiva tiene gran importancia y ha sido también tratado con frecuencia en la literatura (Kahl y Triggs, 1999).

3.7.6. Posición relativa general

En este apartado resolvemos el caso de la primera fila de la tabla 3.1. Es decir, realizaremos dos movimientos de robot, el primero de rotación pura y el segundo de traslación pura. En estas condiciones, siguiendo una semirrecta (una recta con un punto destacado) sobre el suelo a lo largo de tres imágenes se podrán calcular todos los parámetros del modelo general.

Distancia focal y ángulo de picado

Tratamos primero de extraer la focal f. El hecho de seguir una semirrecta implica poder seguir también el punto en el infinito resultante de su intersección con el horizonte. A partir de la medición de dicho punto antes y después de una rotación θ cuyo valor es conocido por la odometría, veremos cómo podemos determinar la focal f. Obsérvese que al tratarse de la proyección de un punto en el infinito, es invariante a la posible traslación parásita implicada por no tener la cámara centrada en el eje de rotación del robot. Es decir, todo lo expuesto a continuación será válido tanto para una cámara con brazo nulo como para una con una traslación relativa $(x_c, y_c)^\top$ no despreciable.

Efectivamente, si llamamos $(t_x^{par}, t_y^{par})^\top$ a la traslación parásita del giro (ver figura 3.9), las imágenes $(x_1, y_h)^\top$ y $(x_2, y_h)^\top$ del punto en el infinito antes y después de la rotación no dependerán de t_x^{par} y t_y^{par}. Equivalentemente, por tanto, los puntos (en el infinito) del mundo correspondientes, que pueden ser obtenidos a través de $\tilde{\mathtt{P}}_{wi}^{-1}$, tampoco dependen de estos valores. La condición de correspondencia entre ambos puntos se puede expresar entonces como:

$$\tilde{\mathtt{0}}_{(0,0,\theta)}\tilde{\mathtt{P}}_{wi}^{-1} \cdot (x_1, y_h, 1)^\top = \tilde{\mathtt{P}}_{wi}^{-1} \cdot (x_2, y_h, 1)^\top \tag{3.49}$$

La igualdad es homogénea, y $\tilde{\mathtt{0}}_{(t_x,t_y,\theta)}$ expresa la traslación y rotación medidas por la odometría, definida una vez más como en 3.17, con la tercera fila y columna eliminadas. Obsérve-

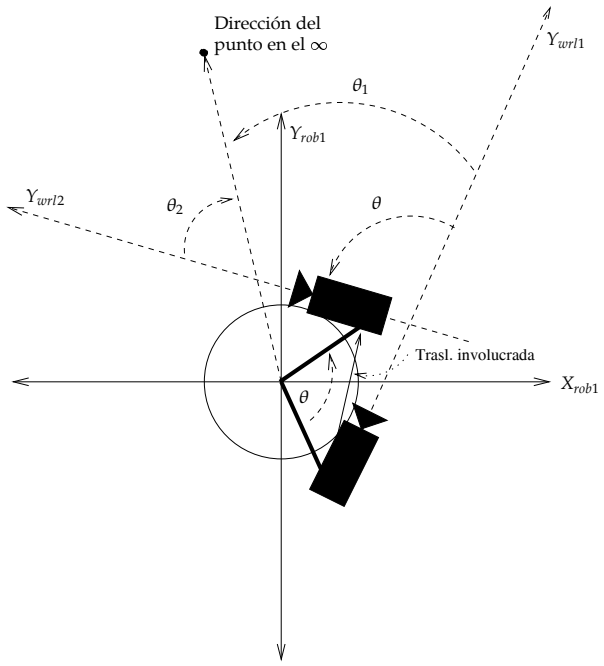


Figura 3.9: *Rotación pura del sistema robot-cámara.*

se que en la ecuación hacemos cero las traslaciones t_x y t_y, ya que no afectan al movimiento de los puntos del infinito. La matriz $\tilde{\mathtt{P}}_{wi}^{-1}$, usando 3.47, resulta ser:

$$\tilde{\mathtt{P}}_{wi}^{-1} = \begin{pmatrix} \sqrt{f^2 + y_h^2} & 0 & 0 \\ 0 & y_h & f^2 \\ 0 & -\frac{f}{z_c} & \frac{f\, y_h}{z_c} \end{pmatrix} \tag{3.50}$$

La igualdad homogénea resulta más cómoda de manejar igualando el producto vectorial a cero. La condición 3.49 queda pues así, después de desarrollar y simplificar:

$$\begin{aligned} &(\tilde{\mathtt{0}}_{(0,0,\theta)}\tilde{\mathtt{P}}_{wi}^{-1} \cdot (x_1, y_h, 1)^\top) \times (\tilde{\mathtt{P}}_{wi}^{-1} \cdot (x_2, y_h, 1)^\top) = (0,0,0)^\top \Rightarrow \\ &(x_1 - x_2)\cos(\theta)\sqrt{f^2 + y_h^2} + (f^2 + x_1\, x_2 + y_h^2)\sin(\theta) = 0 \end{aligned} \tag{3.51}$$

Vemos que, efectivamente, dicha condición tiene un solo valor desconocido, la focal f, puesto que desaparece la altura de la cámara z_c. Podemos, pues, despejar dicho valor, que quedará como sigue:

$$f = \pm\sqrt{\frac{(x_1-x_2)^2 cot(\theta)^2 \pm (x_1-x_2)\cot(\theta)\sqrt{(x_1-x_2)^2\cot(\theta)^2 - 4x_1\, x_2}}{2} - x_1\, x_2 - y_h^2} \tag{3.52}$$

De las cuatro posibles soluciones (correspondientes a las cuatro combinaciones de signos $+$ y $-$), las dos negativas no tienen significado físico en nuestro problema. De las otras dos, si

el movimiento es suficientemente amplio, una de ellas será imaginaria, y quedará descartada. Como el ángulo θ es positivo cuando se gira en sentido antihorario, y cero justo cuando se mira al frente (dirección del eje Y_{wrl}), entonces el valor $(x_1 - x_2)\cot(\theta)$ será siempre negativo. La solución válida será, pues, la que hace más grande el radicando, esto es, la correspondiente al segundo signo negativo. Existen, de todas formas, algunos casos en los que puede haber dos soluciones alternativas, para ángulos de giro pequeños. En cualquier caso, si aseguramos que x_1 y x_2 tienen signos distintos (es decir, que el punto en el horizonte pasa de un lado a otro de la mitad de la imagen), la ambigüedad queda siempre resuelta en favor de la solución propuesta, con segundo signo negativo.

Una vez conocido el valor de f, es trivial computar α, puesto que estos dos parámetros quedan ligados por la relación 3.45. Despejando α, queda:

$$\alpha = \arctan(\frac{f}{y_h}) \tag{3.53}$$

Ángulo de desfase entre sistemas de coordenadas de cámara y robot

El siguiente paso en la calibración consiste en realizar una traslación $(t_x, t_y)^\top$ en el plano del suelo siguiendo la proyección de un punto del suelo en las imágenes anterior y posterior al movimiento. Sean $(x_1, y_1)^\top$ y $(x_2, y_2)^\top$ las coordenadas de imagen de dichos puntos. Las únicas restricciones son, primero, que el punto seguido no podrá estar en el horizonte (donde, como sabemos, las proyecciones sólo dependen de la orientación, y no de la traslación), y segundo, que la traslación debe ser pura, esto es, que los ángulos de odometría del robot antes y después del movimiento deben ser el mismo.

En estas condiciones, deduciremos una expresión para el ángulo β entre el eje Y_{rob} del sistema de movimiento del robot sobre el suelo, en el cual se toman las medidas odométricas, y el eje Y_{wrl}, del sistema de coordenadas del mundo sobre el cual trabaja la homografía $\tilde{\mathtt{P}}_{wi}$.

Obsérvese que, a pesar del desfase (x_c, y_c, β) entre S_{rob} y S_{wrl}, la condición de no rotación entre la toma de ambas imágenes nos asegura que el módulo de la medida $(t_x, t_y)^\top$ del robot trasladándose sobre el suelo es también válido para la traslación experimentada sobre el sistema de coordenadas del mundo. Aunque, eso sí, la traslación expresada en S_{wrl}, $(t_{xw}, t_{yw})^\top$ obviamente cambiará con respecto a $(t_x, t_y)^\top$ en una rotación igual a β, el parámetro que ahora tratamos de determinar. Esta situación se ilustra en la figura 3.10.

Utilizaremos en este caso la matriz fundamental, de la cual tenemos la expresión analítica completa, puesto que conocemos el movimiento experimentado por la cámara y también su calibración intrínseca (Trucco y Verri, 1998):

$$\mathtt{F} = \mathtt{K}^{-1}[\boldsymbol{t}]_\times \mathtt{R}\mathtt{K}^{-1} \tag{3.54}$$

Donde $\mathtt{K}$ es la matriz de calibración, $\mathtt{R}$ la rotación y $\boldsymbol{t}$ la traslación entre las posiciones de

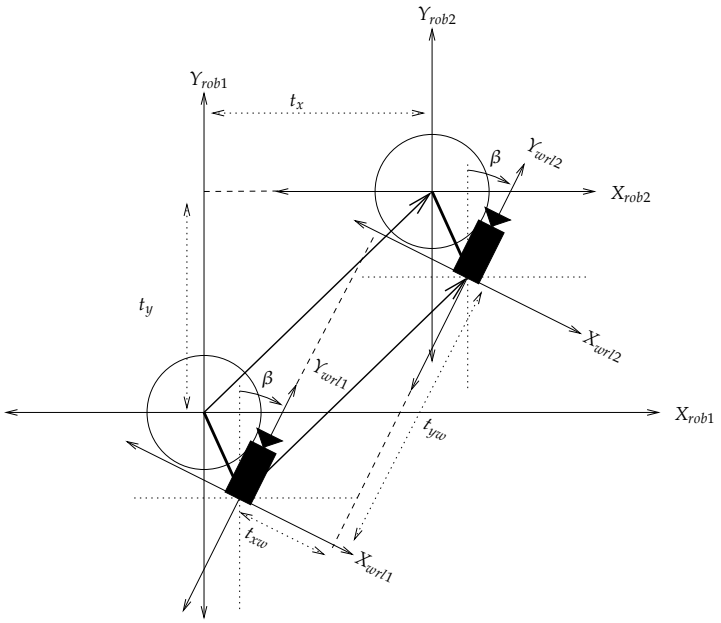


Figura 3.10: *Traslación pura del sistema robot-cámara.*

la cámara antes y después del movimiento, expresadas ambas en coordenadas relativas a la cámara.

Sabemos que ésta última se mueve paralela al suelo, experimentando un desplazamiento $(t_{xw}, t_{yw})^\top$, desconocido inicialmente (puesto que no sabemos la alineación respecto al sistema del robot, único del que podemos tomar la odometría). El vector de traslación, expresado en coordenadas relativas a S_{cam}, sería $\boldsymbol{t} = (t_{xw}, t_{yw}\ \cos(\alpha), t_{yw}\ \sin(\alpha))^\top$, siendo α el ángulo de picado anteriormente determinado (consultar de nuevo la figura 3.7). Usando 3.54, y teniendo en cuenta que no hay rotación relativa entre las dos posiciones de la cámara, se tiene:

$$\mathtt{F} = \mathtt{K}^{-1}[\boldsymbol{t}]_\times \mathtt{R}\mathtt{K}^{-1} = \mathtt{K}^{-1}[\boldsymbol{t}]_\times \mathtt{K}^{-1} =$$

$$\begin{pmatrix} \frac{1}{f} & 0 & 0 \\ 0 & \frac{1}{f} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -t_{yw}\sin(\alpha) & t_{yw}\cos(\alpha) \\ t_{yw}\sin(\alpha) & 0 & -t_{xw} \\ -t_{yw}\cos(\alpha) & t_{xw} & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{f} & 0 & 0 \\ 0 & \frac{1}{f} & 0 \\ 0 & 0 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} 0 & -\frac{t_{yw}\ \sin(\alpha)}{f^2} & \frac{t_{yw}\ \cos(\alpha)}{f} \\ \frac{t_{yw}\ \sin(\alpha)}{f^2} & 0 & -\frac{t_{xw}}{f} \\ -\frac{t_{yw}\ \cos(\alpha)}{f} & \frac{t_{xw}}{f} & 0 \end{pmatrix} \tag{3.55}$$

Ahora, cualquier punto del espacio (y, en particular, del suelo) que se observe en ambas imágenes debe cumplir la condición $\boldsymbol{x}_1^\top \mathtt{F}\boldsymbol{x}_2 = 0$, siendo $\boldsymbol{x}_1$ y $\boldsymbol{x}_2$ las coordenadas homogéneas

de imagen del punto antes y después del movimiento. Haciendo explícitas estas coordenadas $\boldsymbol{x}_1 = (x_1, y_1, 1)^\top$ y $\boldsymbol{x}_2 = (x_2, y_2, 1)^\top$, esta condición se traducirá en lo siguiente:

$$\boldsymbol{x}_1^\top \mathtt{F} \boldsymbol{x}_2 = 0 \Rightarrow (x_1, y_1, 1)^\top \mathtt{F} \, (x_2, y_2, 1) = 0 \Rightarrow$$
$$\frac{f\, t_{xw}(y_2 - y_1) + f\, t_{yw}(x_1 - x_2)\cos(\alpha) + t_{yw}(x_2\, y_1 - x_1\, y_2)\sin(\alpha)}{f^2} = 0 \tag{3.56}$$

Eliminamos f^2 del denominador, y definimos los escalares a y b como sigue:

$$a = f(y_2 - y_1) \tag{3.57}$$

$$b = f(x_1 - x_2)\cos(\alpha) + (x_2\, y_1 - x_1\, y_2)\sin(\alpha) \tag{3.58}$$

El vector $(a, b)^\top$ en el plano del suelo debe ser perpendicular al vector de traslación de la cámara $(t_{xw}, t_{yw})^\top$, puesto que de 3.56 se deduce que $(a, b) \cdot (t_{xw}, t_{yw})^\top = 0$. Pero como conocemos la dirección del movimiento llevado a cabo por el robot a través de la odometría, de la condición de perpendicularidad entre $(t_{xw}, t_{yw})^\top$ y $(a, b)^\top$ se puede calcular directamente el desfase β entre $(t_x, t_y)^\top$ y $(t_{xw}, t_{yw})^\top$ a través de $(a, b)^\top$, de la siguiente manera:

$$\beta = \arcsin\left(\frac{(a, b) \cdot (t_x, t_y)^\top}{\|(a, b)^\top\| \, \|(t_x, t_y)^\top\|}\right) \tag{3.59}$$

La posible ambigüedad en el signo de a y b (que daría lugar a un β de signo cambiado) se elimina en este caso simplemente observando la definición de a en 3.57. Si suponemos que, de modo natural, la cámara apunta hacia adelante en la plataforma (es decir, $\beta < |\pi/2|$), entonces un movimiento en el que $t_y > 0$ implica que $t_{yw} > 0$. En ese caso, es claro que el punto seguido en la imagen debe acercarse al robot, y por tanto $y_2 < y_1 \Rightarrow a < 0$. Un razonamiento análogo llevaría a que $t_y < 0 \Rightarrow a > 0$. Es decir, $(a, b)^\top$ apunta a la izquierda de $(t_{xw}, t_{yw})^\top$, y siguiendo nuestro convenio de ángulos positivos en el sentido antihorario, las expresiones 3.57 y 3.58 para a y b tienen el signo adecuado para que la expresión 3.59 para β sea la correcta.

Una ventaja de esta forma cerrada es que es más estable numéricamente cuanto mayores sean los módulos de los vectores $(a, b)^\top$ y $(t_x, t_y)^\top$. Lo primero se consigue cuando $(x_1, y_1)^\top$ está bastante separado de $(x_2, y_2)^\top$, y lo segundo, realizando un movimiento lo mayor posible (lo que aumenta la línea base del par de cámaras). Así, si disponemos de varias parejas candidatas de puntos correspondientes, para distintos movimientos, lo mejor es quedarnos con aquel punto para el que se maximice el denominador de la expresión 3.59.

Una aproximación alternativa sería aprovechar el hecho de que el movimiento es puramente traslacional para razonar en términos del epipolo. El corte de la recta que une los puntos $(x_1, y_1)^\top$ y $(x_2, y_2)^\top$ con el horizonte debe ser el epipolo del par de cámaras en ambas imágenes. Por otro lado, este epipolo debe coincidir con la imagen de la dirección dada por

el punto en el infinito $(t_{xw}, t_{yw}, 0)^\top$. Usando esta restricción también podemos llegar a obtener el valor de β. El corte de la recta que une los dos puntos con la línea del horizonte, de coordenadas homogéneas $(0, -1, y_h)^\top$, será:

$$((x_1, y_1, 1)^\top \times (x_2, y_2, 1)^\top) \times (0, -1, y_h)^\top = \\ (\frac{x_2\, y_1 - x_1\, y_2 + x_1\, y_h - x_2\, y_h}{y_1 - y_2}, y_h, 1)^\top \tag{3.60}$$

De nuevo, la igualdad involucrada es homogénea. Por otro lado, usando de nuevo la expresión 3.50 para $\tilde{\mathtt{P}}_{wi}^{-1}$, podemos obtener la imagen de $(t_{xw}, t_{yw}, 0)^\top$, que será (empleando también igualdades homogéneas):

$$\tilde{\mathtt{P}}_{wi}^{-1} \cdot (t_{xw}, t_{yw}, 0)^\top = (\tfrac{t_{xw}}{t_{yw}} \sqrt{f^2 + y_h^2}, y_h, 1)^\top \tag{3.61}$$

Igualando 3.60 y 3.61 se obtiene una sencilla condición para el ratio t_{yw}/t_{xw}. Como lo que nos interesa es el desfase β entre $(t_{xw}, t_{yw})^\top$ y $(t_x, t_y)^\top$, podemos calcularlo simplemente así:

$$\beta = \arctan(\frac{t_y}{t_x}) - \arctan(\frac{t_{yw}}{t_{xw}}) = \\ \arctan(\frac{t_y}{t_x}) - \arctan(\frac{(y_1 - y_2)\sqrt{f^2 + y_h^2}}{x_2\, y_1 - x_1\, y_2 + x_1\, y_h - x_2\, y_h}) \tag{3.62}$$

Llegamos de esta forma a la expresión alternativa para el cálculo de β. Quizá este segundo desarrollo sea más sencillo que el realizado a través de la matriz fundamental. Sin embargo, en alguno de los algoritmos que siguen no siempre podremos realizar la simplificación de tener un movimiento traslacional puro. En esos casos aún podremos utilizar un razonamiento análogo al primero (ecuaciones 3.54-3.59) para calcular β en movimientos generales, con componentes traslacional y rotacional simultáneos.

Altura de la cámara

Para averiguar la altura z_c a la que se encuentra la cámara respecto al suelo utilizaremos también el punto de imagen anterior, seguido tras la traslación pura. Continuando con la misma notación, sean $(x_1, y_1)^\top$ y $(x_2, y_2)^\top$ las coordenadas de imagen del punto antes y después del movimiento $(t_x, t_y)^\top$. Debido al ángulo de desfase β, este movimiento en S_{rob} induce una traslación $(t_{xw}, t_{yw})^\top$ en el sistema S_{wrl}, de modo que

$$(t_{xw}, t_{yw})^\top = \begin{pmatrix} \cos(-\beta) & -\sin(-\beta) \\ \sin(-\beta) & \cos(-\beta) \end{pmatrix} \cdot (t_x, t_y)^\top \tag{3.63}$$

De nuevo necesitamos la inversa de la homografía $\tilde{\mathtt{P}}_{wi}^{-1}$, para pasar de coordenadas de imagen a coordenadas del mundo. Podríamos volver a usar 3.50, pero en este caso, alter-

nativamente, trabajaremos sobre la inversa de 3.43. Puede comprobarse fácilmente que otra expresión para $\tilde{\mathtt{P}}_{wi}^{-1}$ es:

$$\tilde{\mathtt{P}}_{wi}^{-1} = \begin{pmatrix} \frac{1}{f} & 0 & 0 \\ 0 & \frac{\cos(\alpha)}{f} & \sin(\alpha) \\ 0 & -\frac{\sin(\alpha)}{f\,z_c} & \frac{\cos(\alpha)}{z_c} \end{pmatrix} \tag{3.64}$$

Ahora, aplicando $\tilde{\mathtt{P}}_{wi}^{-1}$ sobre $(x_1, y_1, 1)^\top$ y $(x_2, y_2, 1)^\top$, y transformando el vector homogéneo resultado para que la tercera componente sea la unidad, los vectores no homogéneos de posición respecto a coordenadas en S_{wrl} mundo antes y después del movimiento serán, respectivamente:

$$(x_{w1}, y_{w1})^\top = \left(\frac{x_1\, z_c}{f\, \cos(\alpha) - y_1\, \sin(\alpha)}, \frac{z_c\, (y_1\, \cos(\alpha) + f\, \sin(\alpha))}{f\, \cos(\alpha) - y_1\, \sin(\alpha)} \right)^\top \tag{3.65}$$

$$(x_{w2}, y_{w2})^\top = \left(\frac{x_2\, z_c}{f\, \cos(\alpha) - y_2\, \sin(\alpha)}, \frac{z_c\, (y_2\, \cos(\alpha) + f\, \sin(\alpha))}{f\, \cos(\alpha) - y_2\, \sin(\alpha)} \right)^\top \tag{3.66}$$

Estos dos vectores están medidos en un sistema de coordenadas que ha sufrido una traslación $(t_{xw}, t_{yw})^\top$. Puede plantearse, pues, la igualdad siguiente:

$$(x_{w1}, y_{w1})^\top = (x_{w2}, y_{w2})^\top + (t_{xw}, t_{yw})^\top \tag{3.67}$$

En esta doble igualdad, usando 3.63, 3.65 y 3.66 todos los valores son ya conocidos, excepto la altura z_c. Podríamos resolverla por mínimos cuadrados, pero en la práctica será mejor despejar z_c de ambas ecuaciones, y utilizar aquella de las dos expresiones que resulte más estable, dependiendo de los valores de entrada. Las dos expresiones resultantes son:

$$z_c = \frac{t_x\, \cos(\beta) + t_y\, \sin(\beta)}{\frac{x_1}{f\, \cos(\alpha) - y_1\, \sin(\alpha)} + \frac{x_2}{-f\, \cos(\alpha) + y_2\, \sin(\alpha)}} \tag{3.68}$$

$$z_c = \frac{(f\, \cos(\alpha) - y_1\, \sin(\alpha))\, (f\, \cos(\alpha) - y_2\, \sin(\alpha))\, \left(t_y\, \cos(\beta) - t_x\, \sin(\beta)\right)}{f\, (y_1 - y_2)} \tag{3.69}$$

Traslación entre sistemas de coordenadas de cámara y robot

Después de obtener la focal y los parámetros extrínsecos respecto a S_{wrl} (aunque aún no todos los relativos a S_{rob}), tenemos la homografía $\tilde{\mathtt{P}}_{wi}$ completamente determinada. Como también sabemos el ángulo de desfase β, ya sólo queda saber la traslación relativa $(x_c, y_c)^\top$ entre ambos sistemas. Para ello, hay que volver a seguir un punto $(x_1, y_1)^\top$ de la imagen (que no esté en el horizonte) hasta otro punto $(x_2, y_2)^\top$, pero en este caso después de la rotación pura de robot, medida con la odometría como θ. La figura 3.11 muestra la geometría básica de la situación.

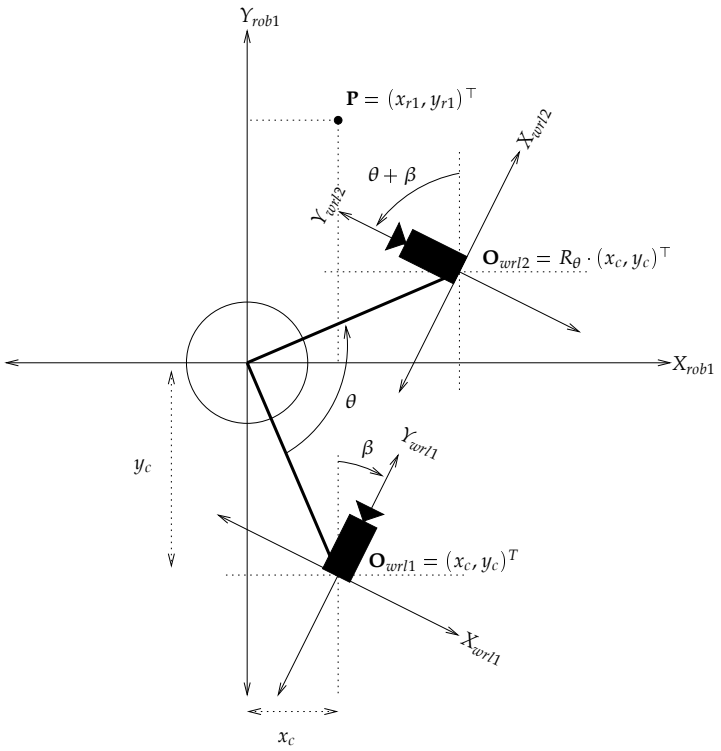


Figura 3.11: *Vista superior del sistema robot-cámara antes y después de una rotación θ.*

Para obtener x_c y y_c, primero usaremos $\tilde{\mathrm{P}}_{wi}^{-1}$ para determinar $(x_{w1}, y_{w1}, 1)^\top$ y $(x_{w2}, y_{w2}, 1)^\top$, coordenadas respecto a los sistemas del mundo antes y después del giro, respectivamente, a partir de $(x_1, y_1, 1)^\top$ y $(x_2, y_2, 1)^\top$:

$$(x_{w1}, y_{w1}, 1)^\top = \tilde{\mathrm{P}}_{wi}^{-1} \cdot (x_1, y_1, 1)^\top \tag{3.70}$$

$$(x_{w2}, y_{w2}, 1)^\top = \tilde{\mathrm{P}}_{wi}^{-1} \cdot (x_2, y_2, 1)^\top \tag{3.71}$$

Estas expresiones quedan exactamente igual que en 3.65 y 3.66, después de eliminar la homogeneidad. Realizamos ahora un cambio de sistemas de coordenadas sobre ambos puntos, para poder imponer una igualdad. De la figura 3.11 se observa que:

$$(x_{r1}, y_{r1})^\top = \begin{pmatrix} \cos(\beta) & -\sin(\beta) \\ \sin(\beta) & \cos(\beta) \end{pmatrix} \cdot (x_{w1}, y_{w1})^\top + (x_c, y_c)^\top =$$

$$\begin{pmatrix} \cos(\theta+\beta) & -\sin(\theta+\beta) \\ \sin(\theta+\beta) & \cos(\theta+\beta) \end{pmatrix} \cdot (x_{w2}, y_{w2})^\top + \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \cdot (x_c, y_c)^\top \tag{3.72}$$

Ambas partes de la ecuación pasan el punto $\boldsymbol{P}$ de coordenadas de los dos sistemas del mundo S_{wrl} al del robot S_{rob} antes del movimiento. Operando sobre ellas puede obtenerse la

solución para los valores de x_c e y_c:

$$(x_c, y_c)^\top = \left(\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} - \mathtt{I} \right)^{-1} \cdot$$
$$\left(\begin{pmatrix} \cos(\beta) & -\sin(\beta) \\ \sin(\beta) & \cos(\beta) \end{pmatrix} \cdot (x_{w1}, y_{w1})^\top - \begin{pmatrix} \cos(\theta+\beta) & -\sin(\theta+\beta) \\ \sin(\theta+\beta) & \cos(\theta+\beta) \end{pmatrix} \cdot (x_{w2}, y_{w2})^\top \right) \tag{3.73}$$

Simplificando la igualdad se obtienen las siguientes expresiones más compactas para x_c e y_c, en función de valores ya todos conocidos:

$$x_c = \frac{x_{w1}\ \sin(\beta - \frac{\theta}{2}) - x_{w2}\ \sin(\beta + \frac{\theta}{2}) + y_{w1}\ \cos(\beta - \frac{\theta}{2}) - y_{w2}\ \cos(\beta + \frac{\theta}{2})}{2\ \sin(\frac{\theta}{2})} \tag{3.74}$$

$$y_c = \frac{-x_{w1}\ \cos(\beta - \frac{\theta}{2}) + x_{w2}\ \cos(\beta + \frac{\theta}{2}) + y_{w1}\ \sin(\beta - \frac{\theta}{2}) - y_{w2}\ \sin(\beta + \frac{\theta}{2})}{2\ \sin(\frac{\theta}{2})} \tag{3.75}$$

El algoritmo 3.5 resume el procedimiento completo para la calibración en el caso general, con la estimación de todos los parámetros f, α, β, x_c, y_c y z_c.

Secuencias críticas

En la resolución del caso general con el procedimiento expuesto en esta sección, los únicos requerimientos a la entrada son los ya comentados de giro amplio para la determinación única de la focal (con que el punto seguido pase de una mitad a otra de la imagen sería suficiente) y que ninguna de las características estén en el infinito (tanto el punto como la recta deben estar en el suelo y a una distancia finita del robot). En principio, pues, no hay combinaciones de entradas adicionales que puedan dar lugar a una solución ambigua. Como también dijimos, no hay ningún problema en que el punto seguido esté sobre la recta. Ésta, de hecho, sería la configuración mínima, con sólo tres grados de libertad. Pero tampoco se trata una condición necesaria, ya que el procedimiento funcionará perfectamente aunque el punto no esté incluido en la línea.

Merece la pena, sin embargo, realizar un comentario sobre la determinación de la altura de la cámara, z_c. Puede haber secuencias en las que alguna de las dos fórmulas para calcular z_c queden indeterminadas. Por ejemplo, si en el seguimiento del punto tras la traslación éste no cambia de altura en la imagen ($y_1 = y_2$), entonces 3.69 puede quedar indeterminada. Este caso correspondería a traslaciones laterales puras de la cámara. Pero en ese caso bastaría con usar la expresión alternativa, 3.68. Algo similar podría ocurrir con 3.68 si $x_1 = x_2 = 0$ (por ejemplo, para puntos justo en el centro de la imagen y traslaciones puras hacia adelante). De nuevo, lo único que habría que hacer sería optar por la expresión alternativa, 3.69.

ENTRADA:
- Ángulo θ de la rotación pura, y vector $(t_x, t_y)^\top$ correspondiente a la traslación pura, ambos medidos por la odometría.
- Altura del horizonte, y_h.
- Coordenadas de imagen homogéneas $\boldsymbol{l}_0$ y $\boldsymbol{l}_1$ de una línea seguida antes y después de la rotación pura.
- Coordenadas de imagen homogéneas $\boldsymbol{x}_1 = (x_1, y_1, 1)^\top$ y $\boldsymbol{x}_2 = (x_2, y_2, 1)^\top$ de un punto seguido antes y después de la traslación pura.
- Coordenadas de imagen homogéneas $\boldsymbol{x}'_1 = (x'_1, y'_1, 1)^\top$ y $\boldsymbol{x}'_2 = (x'_2, y'_2, 1)^\top$ de un punto seguido antes y después de la rotación pura.

SALIDA:
- Parámetros de calibración explícitos f, α, β, x_c, y_c y z_c.
- Matriz de cámara `P` correspondiente.

ALGORITMO:

Cálculo de los intrínsecos:

Cálculo de la focal f:

- Sea $\boldsymbol{l}_h = (0, -1, y_h)^\top$ la recta del horizonte en la imagen.
- Obtener las coordenadas X de los cortes de las rectas $\boldsymbol{l}_0$ y $\boldsymbol{l}_1$ con $\boldsymbol{l}_h$, quedándonos con el segundo componente de los vectores $\boldsymbol{l}_h \times \boldsymbol{l}_0$ y $\boldsymbol{l}_h \times \boldsymbol{l}_1$, después de normalizar para hacer la tercera coordenada igual a uno.
- Obtener f utilizando la expresión 3.52 (con el primer signo $\pm$ tomado como positivo, y el segundo como negativo), siendo x_1 y x_2 los respectivos valores obtenidos en el paso anterior, y θ el ángulo odométrico de giro.

Cálculo de los extrínsecos:

Cálculo del ángulo de picado α:

- Calcular α usando 3.53.

Cálculo del ángulo de desfase β:

- Calcular a y b a través de las ecuaciones 3.57 y 3.58, respectivamente, utilizando los valores $(x_1, y_1)^\top$ y $(x_2, y_2)^\top$ del punto seguido durante la traslación.
- Usando la traslación odométrica $(t_x, t_y)^\top$, y los valores a y b calculados en el punto anterior, calcular β mediante 3.59.
- Alternativamente, puede también calcularse β directamente a partir de $(x_1, y_1)^\top$, $(x_2, y_2)^\top$ y $(t_x, t_y)^\top$, usando 3.62.

Cálculo de la altura z_c:

- Utilizando de nuevo las coordenadas del punto antes y después de la traslación, calcular z_c mediante 3.68, o bien 3.69.

Cálculo del brazo $(x_c, y_c)^\top$:

- Determinar $(x_{w1}, y_{w1})^\top$ y $(x_{w2}, y_{w2})^\top$ a partir de las expresiones 3.70 y 3.71, respectivamente, pero usando en esta ocasión las coordenadas $(x'_1, y'_1)^\top$ y $(x'_2, y'_2)^\top$ del punto seguido antes y después de la rotación. Obsérvese que todos los valores de $\tilde{\mathtt{P}}_{wi}^{-1}$, dada en cualquiera de las formas 3.50 o 3.64, han sido ya calculados previamente.
- Calcular x_c e y_c utilizando 3.74 y 3.75.

Cálculo de la matriz de cámara `P`:

- Calcular `P` a través de la expresión 3.4, estando `K` definida como en 3.8, `R` como en 3.6 (tomando $\gamma = 0$), y $\boldsymbol{C}$ como en 3.7.

Algoritmo 3.5: *Procedimiento analítico de calibración a partir de un movimiento de rotación seguido de uno de traslación, ambos puros, y siguiendo un punto y una recta en la imagen (el punto puede o no estar sobre la recta). Se aplican las suposiciones de píxel cuadrado, punto principal centrado en la imagen y ángulo de* roll γ *despreciable.*

3.7.7. Brazo nulo

Si realizamos la suposición adicional de que la traslación relativa entre los sistemas del robot y la cámara es despreciable (esto es, $x_c = y_c = 0$, equivalente a suponer que la cámara está situada aproximadamente sobre el eje de rotación del robot), entonces el problema de la autocalibración queda significativamente simplificado. Para empezar, los grados de libertad de $\tilde{\mathtt{P}}_{ri}$ quedan reducidos de cinco a tres. En estas condiciones, basta seguir una semirrecta entre sólo dos imágenes (en lugar de tres) tras un movimiento general, con una componente traslacional y otra rotacional. A continuación describimos el procedimiento a seguir en la autocalibración, siguiendo una metodología de cálculo similar a la anteriormente expuesta. Denotaremos de nuevo como (t_x, t_y, θ) el movimiento realizado entre las dos imágenes, sólo que en esta ocasión los movimientos traslacional y rotacional están acoplados en uno solo.

Distancia focal y ángulo de picado

Para determinar f y α, procederemos exactamente igual que en el apartado correspondiente del punto 3.7.6, dado que la traslación relativa $(t_x, t_y)^\top$ no influye en la proyección de los puntos en el infinito. Así, usaremos otra vez las expresiones 3.52 y 3.53, siendo x_1 y x_2 los cortes de la semirrecta seguida con el horizonte en ambas imágenes.

Ángulo de desfase entre sistemas de coordenadas de cámara y robot

Para calcular este desfase, volveremos a trabajar sobre la matriz fundamental, aplicando un razonamiento similar al realizado en este mismo apartado en la sección 3.7.6. Pero ahora el robot también realiza una rotación sobre su eje. En coordenadas del sistema centrado en la cámara (ver de nuevo figura 3.7), un vector unitario en la dirección de dicho eje es $(0, -\sin(\alpha), \cos(\alpha))^\top$ (vertical al suelo). Usando entonces la fórmula de Euler para la matriz de rotación de un ángulo θ alrededor de un eje $\boldsymbol{n} = (n_1, n_2, n_3)^\top$ (Trucco y Verri, 1998), se tiene:

$$\mathtt{R}_{(\boldsymbol{n},\theta)} = \mathtt{I}\cos(\theta) + (1 - \cos(\theta))\,\boldsymbol{n}\boldsymbol{n}^\top + \sin(\theta)\,[\boldsymbol{n}]_\times \tag{3.76}$$

Podemos sustituir $\mathtt{R}_{((0,-\sin(\alpha),\cos(\alpha))^\top,\theta)}$ en 3.54, para obtener una nueva condición sobre t_{xw} y t_{yw}. Desarrollando la expresión, y trabajando como antes, obtenemos un nuevo par de valores para a y b:

$$\begin{gathered} a = -f\,\cos(\alpha)^2\,(y_1 - y_2\,\cos(\theta)) + \\ \cos(\alpha)(-2(f^2 - y_1\,y_2)\sin(\alpha)\sin(\tfrac{\theta}{2})^2 + f\,x_2\,\sin(\theta)) + \\ \sin(\alpha)\,(f(y_2 - y_1\,\cos(\theta))\,\sin(\alpha) - x_2\,y_1\,\sin(\theta)) \end{gathered} \tag{3.77}$$

$$\begin{gathered} b = x_1\ (f\ \cos(\alpha) - \sin(\alpha)\, y_2) + \\ (f\ \cos(\alpha) - \sin(\alpha)\, y_1)\ (-(\cos(\theta)\, x_2) + \sin(\theta)\ (f\ \sin(\alpha) + \cos(\alpha)\, y_2)) \end{gathered} \tag{3.78}$$

Estos valores son análogos a los obtenidos en 3.57 y 3.58, para el caso más complejo que incluye también una rotación. De hecho, puede comprobarse que 3.77 y 3.78 se reducen a 3.57 y 3.58 si $\theta = 0$. El vector $(a, b)^\top$ tiene que ser de nuevo perpendicular a $(t_{xw}, t_{yw})^\top$. Usando, pues, otra vez la expresión 3.59, pero con a y b redefinidos por 3.77 y 3.78, podremos determinar el ángulo β a partir del punto seguido tras el movimiento combinado.

Altura de la cámara

Para obtener, por último, el parámetro z_c, volvemos a proceder como en el caso general. Es decir, primero aplicamos $\tilde{\mathrm{P}}_{wi}^{-1}$ sobre el punto de imagen seguido antes y después del movimiento, para obtener $(x_{w1}, y_{w1})^\top$ y $(x_{w2}, y_{w2})^\top$ como en 3.65 y 3.66. Dichas expresiones dependen linealmente de z_c. Después, planteamos una condición similar a 3.67. Pero en este caso hay también una rotación añadida, con lo que dicha condición queda:

$$(x_{w1}, y_{w1})^\top = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \cdot (x_{w2}, y_{w2})^\top + \begin{pmatrix} \cos(-\beta) & -\sin(-\beta) \\ \sin(-\beta) & \cos(-\beta) \end{pmatrix} \cdot (t_x, t_y)^\top \tag{3.79}$$

Trabajando sobre esta condición podemos volver a despejar z_c obteniendo dos expresiones alternativas, más complejas que las correspondientes 3.68 y 3.69, pero que se también reducen a aquéllas si hacemos $\theta = 0$:

$$z_c = \frac{t_x\ \cos(\beta) + t_y\ \sin(\beta)}{\frac{x_1}{f\ \cos(\alpha) - y_1\ \sin(\alpha)} + \frac{x_2\ \cos(\theta)}{-f\ \cos(\alpha) + y_2\ \sin(\alpha)} + \frac{(y_2\ \cos(\alpha) + f\ \sin(\alpha))\ \sin(\theta)}{f\ \cos(\alpha) - y_2\ \sin(\alpha)}} \tag{3.80}$$

$$z_c = \frac{t_y\ \cos(\beta) - t_x\ \sin(\beta)}{\frac{y_1\ \cos(\alpha) + f\ \sin(\alpha)}{f\ \cos(\alpha) - y_1\ \sin(\alpha)} + \frac{(y_2\ \cos(\alpha) + f\ \sin(\alpha))\ \cos(\theta)}{-f\ \cos(\alpha) + y_2\ \sin(\alpha)} + \frac{x_2\ \sin(\theta)}{f\ \cos(\alpha) + y_2\ \sin(\alpha)}} \tag{3.81}$$

En el algoritmo 3.6 se resume el procedimiento de calibración para el caso $x_c = y_c = 0$.

Secuencias críticas

La resolución del caso simplificado expuesto en este apartado tampoco tiene mayores restricciones en cuanto a las entidades de imagen a seguir. Sólo hay que asegurarse de que el movimiento entre la toma de las dos imágenes tenga componentes rotacional y traslacional no nulas (esto es, $\theta \neq 0$, y $t_x \neq 0$ o $t_y \neq 0$), y que ni el punto ni la recta seguidos estén

ENTRADA:
- Valores del movimiento (t_x, t_y, θ), medidos por la odometría.
- Altura del horizonte, y_h.
- Coordenadas de imagen homogéneas $\boldsymbol{l}_0$ y $\boldsymbol{l}_1$ de una línea seguida durante el movimiento.
- Coordenadas de imagen homogéneas $\boldsymbol{x}_1 = (x_1, y_1, 1)^\top$ y $\boldsymbol{x}_2 = (x_2, y_2, 1)^\top$ de un punto seguido durante el movimiento.

SALIDA:
- Parámetros de calibración explícitos f, α, β y z_c.
- Matriz de cámara `P` correspondiente.

ALGORITMO:

Cálculo de los intrínsecos:

Cálculo de la focal:

- Proceder igual que en el apartado correspondiente del algoritmo 3.5.

Cálculo de los extrínsecos:

Cálculo del ángulo de picado α:

- Ídem.

Cálculo del ángulo de desfase β:

- Calcular a y b a través de las ecuaciones 3.77 y 3.78, respectivamente, utilizando los valores $(x_1, y_1)^\top$ y $(x_2, y_2)^\top$ del punto seguido durante el movimiento.
- Usando la traslación odométrica $(t_x, t_y)^\top$, y los valores a y b calculados en el punto anterior, calcular β mediante 3.59.

Cálculo de la altura z_c:

- Usando de nuevo las coordenadas del punto antes y después del movimiento, calcular z_c mediante 3.80, o bien 3.81.

Cálculo de la matriz de cámara `P`:

- Calcular `P` a través de la expresión 3.4, estando `K` definida como en 3.8, `R` como en 3.6 (tomando $\gamma = 0$), y $\boldsymbol{C}$ como en 3.7 (haciendo $x_c = y_c = 0$).

Algoritmo 3.6: *Procedimiento analítico de calibración a partir de un movimiento combinado de rotación y traslación, siguiendo un punto y una recta entre el par de imágenes (el punto puede o no estar sobre la recta). De nuevo se aplican las suposiciones de píxel cuadrado, punto principal centrado en la imagen y ángulo de* roll γ *despreciable, a la que se añade la condición de brazo nulo.*

en el infinito. Análogamente a lo que ocurría en el caso general, hay que escoger con cuidado la fórmula para determinar z_c, puesto que en algunos casos una de ellas puede quedar indeterminada. De nuevo, la solución es simplemente emplear la expresión alternativa.

3.7.8. Brazo nulo y sin desfase de rumbo

Si además de despreciar la traslación relativa $(x_c, y_c)^\top$ obviamos también el posible ángulo de desfase β, entonces quedan sólo dos grados de libertad: la focal f (que determina también α) y la altura de la cámara z_c. La simplificación $x_c = y_c = \beta = 0$ equivale a suponer que la cámara está montada sobre el eje de rotación del robot y apuntando en su dirección de avance, o lo que es lo mismo, que los sistemas S_{rob} y S_{wrl} son coincidentes.

Para ser capaces de realizar la autocalibración, pues, necesitaremos una configuración de entrada con al menos dos grados de libertad. Consideraremos dos posibilidades: el seguimiento de una recta y el seguimiento de un punto entre dos imágenes. En el primer caso

realizaremos un movimiento de odometría general entre la toma de las dos imágenes, y en el segundo únicamente una traslación pura. Resolveremos ambos casos por separado.

Siguiendo una recta

Focal y ángulo de picado: Si seguimos una recta tras un movimiento de odometría general (t_x, t_y, θ) podemos determinar sin problemas f como en las secciones anteriores, puesto que el punto de corte con el horizonte es inmune a la traslación. Como siempre, α se calcula trivialmente una vez conocido f, usando 3.53.

Altura de la cámara: Para determinar z_c sin ambigüedad es necesaria la traslación. En este caso $\tilde{\mathtt{P}}_{wi} = \tilde{\mathtt{P}}_{ri}$ (por ser $x_c = y_c = \beta = 0$). Entonces, dado que $\tilde{\mathtt{P}}_{wi}$ transfiere puntos del suelo a la imagen, $\tilde{\mathtt{P}}_{wi}^{-\top}$ transferirá líneas, y por tanto $\tilde{\mathtt{P}}_{wi}^{\top}$ devolverá líneas de la imagen al mundo. Del mismo modo, dado un movimiento (t_x, t_y, θ) que transforma puntos en el suelo de acuerdo con la matriz $\tilde{\mathtt{0}}_{(t_x,t_y,\theta)}$, las transformaciones experimentadas por las líneas vendrán dadas por $\tilde{\mathtt{0}}_{(t_x,t_y,\theta)}^{-\top}$. Podemos entonces igualar las coordenadas de las rectas observadas en ambas imágenes de modo similar al empleado hasta ahora. Siendo $(l_1, m_1, n_1)^\top$ y $(l_2, m_2, n_2)^\top$ las coordenadas homogéneas de imagen de la recta antes y después del movimiento, la condición puede expresarse así:

$$\tilde{\mathtt{0}}_{(t_x,t_y,\theta)}^{-\top} \tilde{\mathtt{P}}_{wi}^{\top} \cdot (l_1, m_1, n_1)^\top = \tilde{\mathtt{P}}_{wi}^{\top} \cdot (l_2, m_2, n_2)^\top \tag{3.82}$$

La condición de igualdad homogénea puede transformarse en inhomogénea dividiendo ambos lados de la igualdad por la tercera componente. Quedan entonces dos igualdades (una para la componente x y otra para la y), que son lineales en z_c, con el resto de variables todas conocidas. Podemos entonces despejar la altura, obteniendo las expresiones alternativas:

$$z_c = -\frac{\begin{array}{c} f\, l_1\, \sin(\theta)\, \left(f^2\, t_y\, l_2 + t_y\, l_2\, {y_h}^2 - t_x\, (n_2 + m_2\, y_h)\, \sqrt{f^2 + {y_h}^2}\right) + \\ f\, l_1\, \cos(\theta)\, \left(f^2\, t_x\, l_2 + t_x\, l_2\, {y_h}^2 + t_y\, (n_2 + m_2\, y_h)\, \sqrt{f^2 + {y_h}^2}\right) \end{array}}{\begin{array}{c} \sin(\theta)\, (n_2 + m_2\, y_h)\, \left(f^2\, m_1 - n_1\, y_h\right) - \\ \cos(\theta)\, l_2\, \left(f^2\, m_1 - n_1\, y_h\right)\, \sqrt{f^2 + {y_h}^2} + l_1\, \left(f^2\, m_2 - n_2\, y_h\right)\, \sqrt{f^2 + {y_h}^2} \end{array}} \tag{3.83}$$

$$z_c = -\frac{\begin{array}{c} f\, (n_1 + m_1\, y_h)\, \cos(\theta)\, \left(t_y\, n_2 + t_y\, m_2\, y_h + t_x\, l_2\, \sqrt{f^2 + {y_h}^2}\right) - \\ f\, (n_1 + m_1\, y_h) \sin(\theta)\, \left(t_x\, n_2 + t_x\, m_2\, y_h - t_y\, l_2\, \sqrt{f^2 + {y_h}^2}\right) \end{array}}{\begin{array}{c} -\cos(\theta)\, (n_2 + m_2\, y_h)\, \left(f^2\, m_1 - n_1\, y_h\right) + \\ (n_1 + m_1\, y_h)\, \left(f^2\, m_2 - n_2\, y_h\right) - \sin(\theta)\, l_2\, \left(f^2\, m_1 - n_1\, y_h\right)\, \sqrt{f^2 + y_h^2} \end{array}} \tag{3.84}$$

ENTRADA:
- Valores del movimiento (t_x, t_y, θ), medidos por la odometría.
- Altura del horizonte, y_h.
- Coordenadas de imagen homogéneas $\boldsymbol{l}_1 = (l_1, m_1, n_1)^\top$ y $\boldsymbol{l}_2 = (l_2, m_2, n_2)^\top$ de una línea seguida durante el movimiento.

SALIDA:
- Parámetros de calibración explícitos f, α, y z_c.
- Matriz de cámara `P` correspondiente.

ALGORITMO:

Cálculo de los intrínsecos:

Cálculo de la focal:

- Proceder como en el apartado correspondiente del algoritmo general 3.5.

Cálculo de los extrínsecos:

Cálculo del ángulo de picado α:

- Ídem.

Cálculo de la altura z_c:

- Usando de las coordenadas homogéneas de la recta proyectada antes y después del movimiento, calcular z_c mediante 3.83, o bien 3.84.

Cálculo de la matriz de cámara `P`:

- Calcular `P` a través de la expresión 3.4, estando `K` definida como en 3.8, `R` como en 3.6 (tomando $\beta = 0$ y $\gamma = 0$), y $\boldsymbol{C}$ como en 3.7 (haciendo $x_c = y_c = 0$).

Algoritmo 3.7: *Procedimiento analítico de calibración a partir de un movimiento combinado de rotación y traslación, siguiendo una recta entre el par de imágenes. Se aplican las suposiciones de píxel cuadrado, punto principal centrado en la imagen, brazo nulo y ángulos de* roll *(γ) y* pan *(β) despreciables.*

Secuencias críticas: En este caso sí que hay una restricción sobre la traslación realizada, y es que ésta no puede ser paralela a la recta observada. Esta condición se impone porque, de no haber diferencia de distancia del robot a la recta en los dos instantes, sería imposible desambiguar la altura de la cámara. Las expresiones 3.83 y 3.84 quedarían indeterminadas en esos casos. De todos modos, una vez conocido el corte de la recta con el horizonte en la primera imagen (antes de realizar el movimiento), es sencillo decidir qué movimiento traslacional no incurre en esta ambigüedad. Por ejemplo, si dicho corte cae a la derecha de la imagen ($x_{h1} > 0$), podemos decidir hacer un movimiento hacia adelante con $t_x < 0$, y viceversa. Si x_{h1} es exactamente cero, entonces bastaría con hacer un movimiento con $t_x \neq 0$.

Una vez más esquematizamos en el algoritmo 3.7 el funcionamiento del procedimiento de autocalibración para este caso simplificado, donde $x_c = y_c = \beta = 0$ y se sigue una sola recta entre dos imágenes.

Siguiendo un punto

Focal y ángulo de picado: Para el caso de seguimiento de un punto optaremos por un solo movimiento traslacional puro, lo que simplifica el cálculo de la focal. Sea $(t_x, t_y)^\top$, de nuevo, la traslación medida por la odometría entre la toma de las dos imágenes. Para una traslación pura los epipolos caen en la misma posición en ambas imágenes, exactamente en la proyec-

ción del punto en el infinito que marca la dirección del movimiento:

$$\tilde{\mathrm{P}}_{wi} \cdot (t_x, t_y, 0)^\top = (\frac{t_x\sqrt{f^2 + y_h^2}}{t_y}, y_h, 1)^\top \tag{3.85}$$

Pero este epipolo debe coincidir exactamente con la intersección de la recta que une $(x_1, y_1)^\top$ y $(x_2, y_2)^\top$ con el horizonte, siendo éstas las coordenadas de imagen del punto seguido antes y después de la traslación. Igualando la expresión 3.60 para el punto de corte con 3.85 podemos despejar de modo muy sencillo la focal f:

$$f = \sqrt{\left(\frac{t_y(x_1\, y_h - x_2\, y_h + x_2\, y_1 - x_1\, y_2)}{t_x(y_1 - y_2)}\right)^2 - y_h^2} \tag{3.86}$$

Una vez más, α se calcula entonces usando 3.53.

Altura de la cámara: La altura de la cámara es también fácil de determinar, puesto que no es más que un caso particular del discutido en la sección 3.7.6, donde también se realizaba un movimiento traslacional puro. Si usamos las ecuaciones 3.68 y 3.69, pero haciendo $\beta = 0$, obtenemos las dos expresiones alternativas para z_c:

$$z_c = \frac{t_x}{\frac{x_1}{f\cos(\alpha) - y_1\sin(\alpha)} + \frac{x_2}{-f\cos(\alpha) + y_2\sin(\alpha)}} \tag{3.87}$$

$$z_c = \frac{t_y(f\cos(\alpha) - y_1\sin(\alpha))(f\cos(\alpha) - y_2\sin(\alpha))}{f(y_1 - y_2)} \tag{3.88}$$

Secuencias críticas: Tenemos también aquí un tipo de movimiento prohibido en la traslación. Se trata de los avances puros, en los que $t_x = 0$, o los desplazamientos laterales puros, en los que $t_y = 0$. En estos casos el epipolo tiene coordenadas de imagen $(0, y_h)^\top$ y $(\infty, y_h)^\top$, respectivamente, con lo que es imposible calcular f usando 3.86. Existe entonces una ambigüedad en la solución, en la que puntos distintos del mundo, observados con focales y alturas de cámara distintas, generan tras un mismo movimiento odométrico exactamente las mismas imágenes del suelo, con el horizonte a la misma altura. Esta ambigüedad queda eliminada si $t_x \neq 0$ y $t_y \neq 0$, en cuyo caso el procedimiento funcionará sin problemas obteniendo una solución única.

En el algoritmo 3.8, último de los discutidos en este capítulo, se resume el funcionamiento de esta segunda versión de la autocalibración bajo las suposiciones $x_c = y_c = \beta = 0$, aplicable al seguimiento de un solo punto tras un movimiento traslacional.

ENTRADA:
- Valores del movimiento (t_x, t_y), medidos por la odometría.
- Altura del horizonte, y_h.
- Coordenadas de imagen homogéneas $\boldsymbol{x}_1 = (x_1, y_1, 1)^\top$ y $\boldsymbol{x}_2 = (x_2, y_2, 1)^\top$ de un punto seguido durante el movimiento.

SALIDA:
- Parámetros de calibración explícitos f, α, y z_c.
- Matriz de cámara P correspondiente.

ALGORITMO:

Cálculo de los intrínsecos:

Cálculo de la focal:

- Calcular f usando la expresión 3.86.

Cálculo de los extrínsecos:

Cálculo del ángulo de picado α:

- Calcular α usando 3.53.

Cálculo de la altura z_c:

- Utilizar cualquiera de las expresiones 3.87 o 3.88.

Cálculo de la matriz de cámara P:

- Proceder de modo idéntico al apartado correspondiente del algoritmo 3.7.

Algoritmo 3.8: *Procedimiento analítico de calibración a partir de un movimiento de traslación siguiendo un punto entre el par de imágenes. Se aplican las mismas suposiciones que en el procedimiento 3.7.*

3.8. Ejemplos de operación

3.8.1. Ejemplo sintético

Con el fin de aclarar el funcionamiento de los algoritmos, así como de comprobar el efecto de perturbación sobre la solución obtenida que introducen las distintas simplificaciones, exponemos en este apartado un ejemplo sintético de aplicación. Volvemos a utilizar el pasillo artificial de la figura 3.8, observado en esta ocasión desde una cámara situada en una plataforma virtual cuyos parámetros fijamos arbitrariamente como $f = 400$ píxeles, $\alpha = 5\pi/12 = 1.31$ radianes, $z_c = 140$ cm, $x_c = 20$ cm, $y_c = -30$ cm y $\beta = -0.1$ radianes, para observar posteriormente cómo son recuperados por los distintos procedimientos.

Trabajaremos sobre las cuatro imágenes de la figura 3.12, tomadas desde las respectivas posiciones de plataforma mostradas en la figura 3.13. El pasillo, aunque completamente visible, está en una situación en principio desconocida en relación a la plataforma. La línea del horizonte aparece de nuevo con grosor doble, y su altura se supone determinada con anterioridad, por ejemplo intersectando los dos lados del pasillo. Su valor es $y_h = 107.18$ píxeles. En todas ellas localizamos el punto inferior izquierdo del pasillo, así como la recta lateral en la que está contenido. De izquierda a derecha y de arriba a abajo, las coordenadas de imagen del punto son, respectivamente, $(x_a, y_a)^\top = (-95.06, -59.83)^\top$, $(x_b, y_b)^\top = (85.83, -64.41)^\top$, $(x_c, y_c)^\top = (-35.59, -119.42)^\top$ y $(x_d, y_d)^\top = (-165.82, -88.63)^\top$, y la recta del lado izquierdo tiene coordenadas homogéneas $(l_a, m_a, n_a)^\top = (-87.13, 5.8, -7935.6)^\top$,

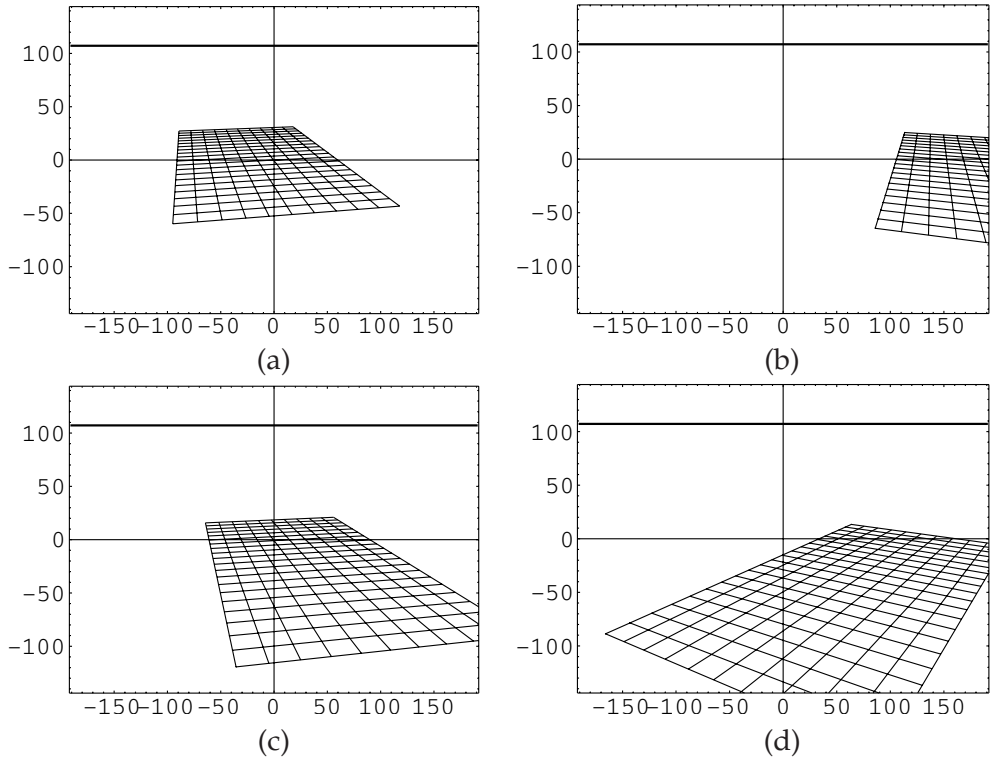


Figura 3.12: *Imágenes de un pasillo de* 10×20 *baldosas de* 20×20 *cm situado delante de una plataforma con una cámara montada como en la figura 3.2 (pero con* $\gamma = 0$*), con los parámetros de calibración prefijados como se comenta en el texto. En la figura 3.13 se muestran las posiciones del robot desde las que se toman estas imágenes.*

$(l_b, m_b, n_b)^\top = (-89.26, 27.6, 9438.7)^\top$, $(l_c, m_c, n_c)^\top = (-135.23, -28.86, -8258.6)^\top$ y finalmente $(l_d, m_d, n_d)^\top = (-102.01, 229.41, 3416.29)^\top$, para las respectivas imágenes 3.12(a), 3.12(b), 3.12(c) y 3.12(d). Las posiciones de la plataforma medidas por la odometría son $t_{xa} = t_{ya} = \theta_a = 0$ (posición inicial), $t_{xb} = t_{yb} = 0$ y $\theta_b = \pi/6$ (movimiento rotacional puro), $t_{xc} = -50$, $t_{yc} = 100$ y $\theta_c = 0$ (movimiento traslacional puro) y $t_{xd} = 200$, $t_{yd} = 175$ y $\theta_d = \pi/4$ (movimiento combinado), tal y como se muestra en las correspondientes figuras 3.13(a-d).

Como entrada para los distintos algoritmos utilizaremos un conjunto de características mínimo para cada caso, además de las respectivas odometrías. Así, para probar el algoritmo 3.5 correspondiente al caso general, emplearemos únicamente $(x_a, y_a)^\top$, $(x_b, y_b)^\top$, $(x_c, y_c)^\top$, $(l_a, m_a, n_a)^\top$, $(l_b, m_b, n_b)^\top$ y $(l_c, m_c, n_c)^\top$, puesto que hay que seguir una semirrecta en tres imágenes: una inicial, otra tomada tras un movimiento traslacional y otra tras otro rotacional[15]. Para el algoritmo 3.6, que utiliza la simplificación $x_c = y_c = 0$, seguiremos sólo una semirrecta antes y después de un movimiento combinado, y por tanto usaremos $(x_a, y_a)^\top$, $(x_d, y_d)^\top$, $(l_a, m_a, n_a)^\top$ y $(l_d, m_d, n_d)^\top$. Si se añade la simplificación $\beta = 0$ tenemos dos po-

[15]En realidad, el seguimiento de la recta durante la traslación ni siquiera es necesario, por lo que tampoco se utilizará $(l_c, m_c, n_c)^\top$

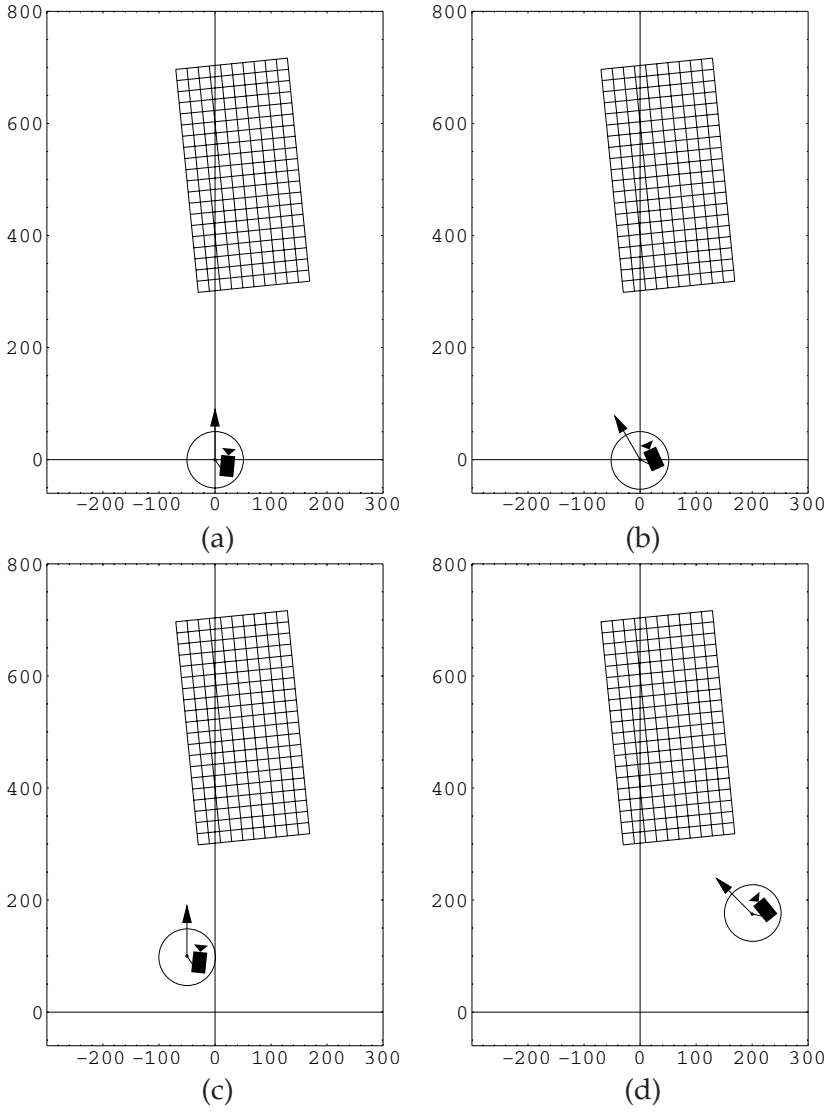


Figura 3.13: *Posiciones de la plataforma desde las que se toman las imágenes del ejemplo sintético mostradas en la figura 3.12. En todos los casos se observan los parámetros de desfase entre la cámara y el robot,* $x_c = 20$ *cm,* $y_c = -30$ *cm y* $\beta = -0.1$.

sibilidades: o bien seguimos una sola recta antes y después de un movimiento combinado (algoritmo 3.7), y entonces usaremos $(l_a, m_a, n_a)^\top$, y $(l_d, m_d, n_d)^\top$, o bien seguimos un punto tras una traslación pura (algoritmo 3.8), en cuyo caso utilizaremos $(x_a, y_a)^\top$ y $(x_c, y_c)^\top$.

La tabla 3.2 resume los resultados obtenidos. Como es de esperar, el algoritmo general recupera perfectamente todos los parámetros extrínsecos e intrínsecos, en presencia de datos sin ruido. El resto de algoritmos pueden perder algo de precisión en dicha estimación, como consecuencia de despreciar alguno de (o todos) los parámetros de desfase entre los sistemas de coordenadas del robot y del suelo, S_{rob} y S_{wrl}. Las distintas simplificaciones aplicadas pueden llegar a distorsionar en algún caso ciertos parámetros de modo importante. Por ejemplo,

Algoritmo		f	α	β	z_c	x_c	y_c
General	(3.5)	400	1.31	-0.1	140	20	-30
$x_c = y_c = 0$	(3.6)	400	1.31	-0.12	127.2	-	-
$x_c = y_c = \beta = 0 (recta)$	(3.7)	400	1.31	-	138.05	-	-
$x_c = y_c = \beta = 0 (punto)$	(3.8)	512	1.36	-	118.75	-	-

Tabla 3.2: *Parámetros de calibración obtenidos por los cuatro algoritmos descritos en el ejemplo sintético.*

suponer $\beta = 0$ cuando en realidad $\beta = -0.1$, y el giro empleado en la calibración de la focal es $\alpha = \pi/6 \approx 0.5$ resulta en un error en la estimación de la focal del 20 %, aproximadamente, si nos fiamos del seguimiento de un punto. No obstante, siguiendo una recta se aprecia que la focal se estima sin error alguno, puesto que en este caso el desfase relativo β no influye en el cálculo de f (ver expresión 3.52). Al final del capítulo incluímos una breve discusión en la que hablaremos de este tipo de problemas, y de la serie de consideraciones a tener en cuenta para decidirnos por alguno de los procedimientos.

3.8.2. Ejemplo real

Finalmente, y para ilustrar la aplicabilidad práctica de los algoritmos analíticos descritos, mostramos un ejemplo de operación real en el cual se calibra el agente autónomo utilizado como plataforma experimental en esta tesis. Aunque detallaremos más en profundidad las características del sistema en el capítulo 5, para los propósitos de este apartado basta aclarar que se trata de un robot con acceso a la odometría planar, dotado de una sola cámara montada a una altura de entre 60 y 120 cm sobre el suelo, sin ángulo de *roll* y aproximadamente centrada y alineada con la dirección de avance (ver figura 5.1 en el capítulo mencionado). La cámara se coloca también con un cierto ángulo de picado α, a fin de ganar campo de visión sobre el suelo en las inmediaciones del robot. En la instalación de la misma no se utiliza ningún tipo de instrumento de precisión, por lo que puede haber pequeños desfases debidos al montaje manual. En resumen, podemos decir que el vehículo se ajusta a las suposiciones del modelo más simplificado descrito en la sección 3.7.8, si bien la imprecisión en el montaje puede hacer que los valores de x_c, y_c y β tomen en algunos casos valores no despreciables. Esto último hace que a veces pueda resultar interesante aplicar alguno de los algoritmos más generales.

En la figura 3.14 se ilustra el proceso de autocalibración practicado. El robot realizó un movimiento controlado que incluía una rotación a ambos lados de aproximadamente 50° en total y una traslación posterior de unos 200 cm a lo largo de un pasillo. Las imágenes 3.14(a-c) muestran tres *frames* tomados por la cámara en el instante inicial, durante el giro a la derecha y al final de la traslación, respectivamente. Los correspondientes segmentos extraídos con el algoritmo propuesto en el capítulo anterior se muestran en las figuras 3.14(d-f). Los procedimientos de calibración utilizaron como entrada tanto directamente las líneas en las que

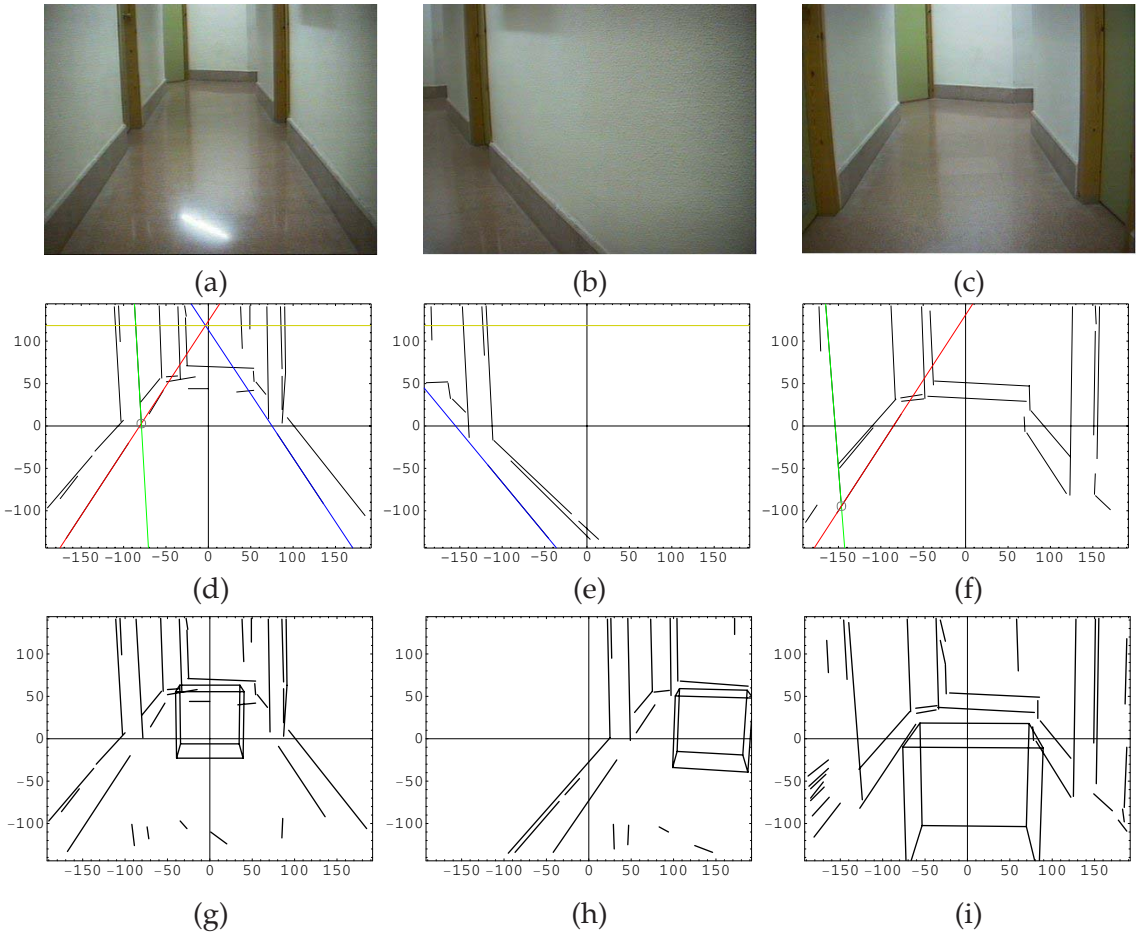


Figura 3.14: *Ejemplo de calibración en condiciones reales. (a-c) Imágenes originales de la secuencia. (d-f) Segmentos extraídos, con las líneas del horizonte, puerta, y paredes izquierda y derecha del pasillo resaltadas en amarillo, verde, rojo y azul respectivamente. Se muestra también resaltado con un círculo el punto de intersección entre las líneas verde y roja. Todas estas entidades son las utilizadas como entrada en el procedimiento de calibración. (g-i) Imágenes de la secuencia aumentadas con un cubo virtual según la matriz de proyección* `P` *computada.*

estaban contenidos algunos segmentos como ciertos puntos determinados a partir de sus intersecciones. Más concretamente, la información de imagen empleada se muestra en colores en dichas figuras. Las líneas en rojo y azul corresponden a los lados izquierdo y derecho de pasillo, respectivamente, y la línea en amarillo a la del horizonte, hallada por intersección de las dos anteriores. La altura computada para éste último es de 118.4 píxeles. Finalmente, se usó también la línea verde vertical correspondiente al marco de la puerta izquierda para seguir el punto marcado con un círculo, intersección con la línea de base de la pared correspondiente. Las odometrías medidas (en cm y grados) para los *frames* mencionados fueron $(t_{xa}, t_{ya}, \theta_a) = (0, 0, 0)$, $(t_{xb}, t_{yb}, \theta_b) = (0.5, 59.5, -23.28^\circ)$ y $(t_{xc}, t_{yc}, \theta_c) = (1.9, 205.8, 0.09^\circ)$, respectivamente.

Sobre el punto resaltado, seguido entre la primera y tercera imágenes, y la recta de la

derecha, entre la primera y la segunda, se aplicó el algoritmo 3.6, bajo la hipótesis de brazo nulo pero estimando el posible ángulo β de desfase (a pesar de que éste último debería estar cercano a cero, según se intentó al montar manualmente la cámara sobre el robot). Aunque se utilizaron tres imágenes, sólo se necesitó seguir un punto y una recta en cada par, y en realidad se podría haber trabajado sólo sobre dos (en el ejemplo comentado se utilizaron tres simplemente porque el punto seguido no aparecía en la segunda). La focal estimada por el procedimiento fue de $f = 566.3$ píxeles, para el tamaño de imagen 288×384. En cuanto a los extrínsecos, los ángulos de *tilt* y *pan* obtenidos fueron $\alpha = 78.19^\circ$ y $\beta = -0.43^\circ$ respectivamente, y la altura $z_c = 88.1$ cm. Estos valores coincidían aproximadamente con lo comprobado visualmente mediante la inspección directa del sistema robot-cámara.

Aún así, también se construyó la matriz de proyección `P` correspondiente a los parámetros estimados, con el fin de realizar una prueba adicional sobre su validez[16]. Dicha `P` se utilizó entonces para reproyectar en varias imágenes de la secuencia tomada durante el movimiento un cubo virtual de 50 cm de lado, situado 350 cm por delante del robot al comenzar, y utilizando la odometría correspondiente a cada *frame*. Las figuras 3.14(g-i) muestran finalmente estas reproyecciones, que corroboran la aplicabilidad de los resultados obtenidos.

3.9. Discusión

Ante la multitud de técnicas de autocalibración a partir de odometría propuestas a lo largo del capítulo, cabe preguntarse bajo qué condiciones deberíamos optar por una u otra. Con el fin de ayudar a tomar la decisión correcta, recopilamos las características principales de todos los procedimientos en la tabla 3.3, que cierra el capítulo. En general la elección dependerá de varios factores, entre los que destacaremos los siguientes:

- En primer lugar, habrá que tener en cuenta la validez de cada una de las simplificaciones, y optar por algoritmos más generales en caso de que las suposiciones no se ajusten a la situación de interés.

- Un segundo aspecto muy importante es el número de primitivas que estén disponibles habitualmente en los entornos de aplicación. Si observamos el tipo de escenas de nuestros experimentos, veremos que a menudo no es sencillo encontrar suficientes puntos para realizar una estimación fiable de la matriz fundamental o de la homografía del suelo (véanse de nuevo las imágenes de la figura 3.14, por ejemplo). En esas condiciones pueden resultar muy útiles los procedimientos analíticos, con escasos requerimientos de entrada.

- Por supuesto, no hay que olvidar los movimientos que puedan resultar en secuencias

[16] Obsérvese que la utilización del modelo de cámara simplificado tiene la ventaja de que se puede obtener una matriz `P` completa, a pesar de que el procedimiento de calibración utiliza sólo elementos contenidos en un plano.

Alg.	Simplificaciones	Vistas	Movimientos	Entidades	Comentarios
3.2	$s = o_x = o_y = 0$	3	Generales.	8 puntos en posición general.	Se parte de una reconstrucción proyectiva, a través de la estimación de F (alg. 3.1). Si no interesa la escala en Z, no necesita $s = o_x = o_y = 0$.
3.3	$s = o_x = o_y = 0$	3	Generales.	4 puntos situados sobre el plano del suelo.	Si sólo interesa la homografía del suelo, no necesita $s = o_x = o_y = 0$.
3.4	$s = o_x = o_y = 0$ $x_c = y_c = 0$	2	General.	4 puntos situados sobre el plano del suelo.	Necesita $s = o_x = o_y = 0$, aunque sólo interese la homografía del suelo.
3.5	$s = o_x = o_y = 0$ $f_x = f_y$ $\gamma = 0$	3	Traslación pura, seguida de una rotación pura.	1 punto y 1 recta situados sobre el plano del suelo.	El punto puede estar o no sobre la recta. La recta sólo tiene que seguirse durante la rotación.
3.6	$s = o_x = o_y = 0$ $f_x = f_y$ $\gamma = 0$ $x_c = y_c = 0$	2	General.	1 punto y 1 recta situados sobre el plano del suelo.	El punto puede estar o no sobre la recta.
3.7	$s = o_x = o_y = 0$ $f_x = f_y$ $\gamma = \beta = 0$ $x_c = y_c = 0$	2	General.	1 recta situada sobre el plano del suelo.	El movimiento no puede ser paralelo a la recta.
3.8	$s = o_x = o_y = 0$ $f_x = f_y$ $\gamma = \beta = 0$ $x_c = y_c = 0$	2	Traslación pura.	1 punto situado sobre el plano del suelo.	Se exige componente traslacional no nulo en ambos ejes X e Y.

Tabla 3.3: *Resumen de características de los procedimientos de autocalibración a partir de odometría propuestos en el capítulo. En cada fila se muestran, en este orden, el algoritmo comentado, las simplificaciones sobre el modelo cámara-robot empleado, el número de vistas necesarias, los tipos de movimientos a realizar, las entidades de imagen a seguir, y una columna con comentarios adicionales sobre cada método.*

críticas para los distintos algoritmos. Siempre habrá que evitarlos e, incluso, intentar mantenerse lejos de ellos en la medida de lo posible.

- Hay que tener asimismo en cuenta el grado de precisión de la odometría y de los extractores de características utilizados. En los ámbitos en los que éste sea bajo, la sobredeterminación de los procedimientos de estimación (utilizando más características o imágenes de entrada) puede resultar de utilidad.

- Por último, debe considerarse también la estabilidad numérica de los distintos procedimientos empleados. Como discutimos en las secciones correspondientes, a veces es posible elegir entre varias alternativas en algún paso de la calibración. Si se dispone de la información adecuada, puede realizarse dicha elección de modo que disminuya el

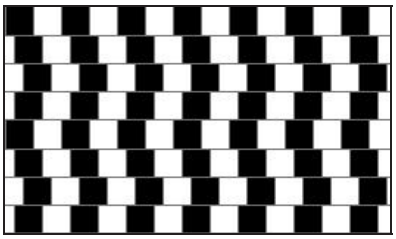
error cometido. Otra vía de solución puede ser el uso de métodos de estimación robustos, basándonos de nuevo en la redundancia en las entradas de los procedimientos.

3.10. Resumen

Resumimos en este último apartado las principales propuestas del capítulo:

- Frente al enfoque precalibrado clásico se opta por un esquema autocalibrado, en el que los elementos de la propia escena se pueden usar en la estimación de las características del sensor. Se elimina así la necesidad de introducir plantillas o marcas artificiales.
- Se plantea la utilización de la odometría como apoyo natural a la calibración, con el fin de obtener medidas reales de la escena sin ningún tipo de ambigüedad proyectiva, afín o similar en las reconstrucciones practicadas. La información euclídea conseguida resultará de gran utilidad en la interpretación estructural posterior y, en consecuencia, también en la navegación.
- Se propone un variado repertorio de técnicas con distintos grados de simplificación del problema, al objeto de cubrir distintas necesidades en el número y tipo de características de imagen y de movimientos realizados.
- Los métodos presentados se dividen en dos grandes grupos: en primer lugar, los basados en técnicas lineales, que obtienen directamente la homografía de transformación entre el suelo y la imagen y, en su caso, la matriz de proyección correspondiente; en segundo lugar, las técnicas analíticas, que calculan explícita e individualmente cada uno de los parámetros, y cuya principal ventaja es la disminución del número de características requeridas con respecto a los anteriores.
- Los experimentos realizados demuestran la viabilidad del uso de la odometría en la autocalibración, dada la aceptable precisión lograda por las técnicas descritas en tareas relacionadas con la navegación visual. La escasa carga computacional de todas ellas, junto con su adecuación a los entornos estructurados, hacen a la aproximación idónea para ser utilizada *on-line* en la arquitectura de percepción propuesta.

CAPÍTULO IV



"Cafe wall illusion", R. Gregory, 1979

Percepción de alto nivel

La pura verdad es que en el mundo pasa en todo instante, y, por tanto, ahora, infinidad de cosas. La pretensión de decir qué es lo que ahora pasa en el mundo ha de entenderse, pues, como ironizándose a sí misma. Mas por lo mismo que es imposible conocer directamente la plenitud de lo real, no tenemos más remedio que construir arbitrariamente una realidad, suponer que las cosas son de una cierta manera. Esto nos proporciona un esquema, es decir, un concepto o enrejado de conceptos. Con él, como al través de una cuadrícula, miramos luego la efectiva realidad, y entonces, sólo entonces, conseguimos una visión aproximada de ella. En esto consiste el método científico. Más aún: en esto consiste todo uso del intelecto. Cuando al ver llegar a nuestro amigo por la vereda del jardín decimos "Éste es Pedro", cometemos deliberadamente, irónicamente, un error. Porque Pedro significa para nosotros un esquemático repertorio de modos de comportarse física y moralmente –lo que llamamos "carácter"–, y la pura verdad es que nuestro amigo Pedro no se parece, a ratos, en casi nada a la idea "nuestro amigo Pedro". (...)
Esta teoría del conocimiento de la razón hubiera irritado a un griego. Porque el griego creyó haber descubierto en la razón, en el concepto, la realidad misma. Nosotros, en cambio, creemos que la razón, el concepto, es un instrumento doméstico del hombre que éste necesita y usa para aclarar su propia situación en medio de la infinita y archiproblemática realidad que es su vida. Vida es lucha con las cosas para sostenerse entre ellas. Los conceptos son el plan estratégico que nos formamos para responder a su ataque. Por eso, si se escruta bien la entraña última de cualquier concepto, se halla que no nos dice nada de la cosa misma, sino que resume lo que un hombre puede hacer con esa cosa o padecer de ella.

J. ORTEGA Y GASSET, *La rebelión de las masas.*

4.1. Introducción

Apoyándonos en la potencia expresiva de los segmentos con información de color, que respetan el grueso de la información *espaciotópica* de la imagen, y una vez calibradas las características de nuestro sensor óptico, estamos en condiciones de atacar el problema de la percepción a más alto nivel. En la aplicación que nos ocupa, el objetivo básico de ésta es la elaboración y constante actualización de una estructura modelizadora del entorno, que actúe

como motor causal en la toma de las decisiones de navegación por parte del agente móvil. Este mismo movimiento debe a su vez ayudar en el mantenimiento de la estructura interpretativa, cerrando un ciclo de percepción y acción caracterizado por la deseada capacidad de anticipación sobre lo percibido. Hablaremos de interpretación de alto nivel de la escena, porque va más allá del intento de asociación directa entre los estímulos sensoriales y las acciones de movimiento. Este último enfoque, desde nuestro punto de vista, conduce a claras limitaciones en condiciones de operación mínimamente realistas. En su lugar, nuestro concepto de interpretación busca una abstracción cualitativa del entorno que, obviando los detalles, y teniendo siempre en cuenta el contexto espacio-temporal de operación, permita la generación de comportamientos coherentes a medio y largo plazo.

En el sentido que se atribuye a la palabra en esta tesis, pues, *interpretar* significa instanciar la información sensorial recibida en términos de un determinado modelo de la realidad. En nuestro caso, éste se basará en la estructuración planar del entorno, es decir, en la correcta localización y categorización de los principales planos de la escena. Esta representación se adapta bien al dominio de aplicación donde evaluaremos la arquitectura, las escenas parcialmente estructuradas de interiores de edificios. A pesar de sus aparentes limitaciones, como veremos, el esquema es relativamente resistente a la existencia de elementos u obstáculos ajenos a esta representación. Dicho de otra forma, el modelo propuesto es fundamentalmente escalable, puesto que no entra en conflicto con la presencia de objetos que no se ajustan al mismo, sino que, antes al contrario, actúa como una estructura de fondo básica de la escena, ampliable con otro tipo de descripciones más detalladas sobre el resto de elementos percibidos. Adicionalmente, el manejo del plano como primitiva tridimensional básica en la categorización del entorno introduce también una importante ventaja desde el punto de vista de la geometría proyectiva: la simplicidad de manejo de las homografías como herramientas de transferencia entre las entidades involucradas.

Sobre la base de la detección y seguimiento de estos planos trataremos de reconocer situaciones tipo elementales, construidas a partir de los estímulos sensoriales cuando se les dota de interpretación. De nuevo, a pesar de la sencillez del conjunto de prototipos utilizados, las estructuras interpretativas creadas servirán para adaptarse a una gran variabilidad de condiciones sobre distintas situaciones del dominio, gracias a la flexibilidad introducida por su condición de parametrizables. De acuerdo con estas interpretaciones se moverá el agente, coordinando efectivamente los subsistemas de percepción y control para lograr determinados objetivos de navegación y seguimiento de trayectorias adaptadas a la estructura del entorno. Dentro de este diseño, que intenta evitar el comportamiento estrictamente reactivo, es también clave dotar a la arquitectura de la capacidad de tener en cuenta no sólo los datos obtenidos dentro del rango efectivo de los sensores en cada instante, sino también la información de la estructura cercana percibida con anterioridad y que ya no está al alcance de los mismos. En este sentido, el papel de la memoria a corto plazo del agente, convenientemente actualizada con continuidad por el uso de los sensores propioceptivos del movimiento, jugará también

un papel fundamental.

Con el fin de tratar las soluciones propuestas para todas estas cuestiones, este capítulo se estructurará como sigue. Comenzaremos, como viene siendo habitual, revisando algunos de los trabajos más recientes sobre navegación visual descritos en la literatura. En un apartado posterior, definiremos las bases de la navegación interpretativa, sobre las que se asienta el mecanismo de percepción de alto nivel propuesto. Se describe después en detalle el modelo planar que utilizaremos como herramienta de regularización para entornos estructurados, haciendo énfasis en las consideraciones prácticas de uso del mismo, desde sus características geométricas hasta la elaboración de un algoritmo que, con la información adecuada, sea capaz de recuperar información tridimensional de la escena incluso a partir de una sola imagen, adecuadamente interpretada. Finalmente, en el último apartado se especifican punto por punto todos los elementos que conforman la arquitectura de acción y percepción propuesta, tanto en lo que se refiere a las representaciones y elementos de datos como en lo relativo a los procesos de información empleados.

4.2. Aproximaciones a la navegación visual

En la figura 4.1 se recoge un posible esquema genérico para atacar el problema de la navegación visual en agentes autónomos. Como se observa, el agente se considera ubicado dentro de un entorno desde el cual recibe una secuencia de imágenes **I** para, tras una adecuada secuencia de procesamiento, generar una serie de órdenes de control **C** sobre sus actuadores, que son los que provocan el movimiento. Este movimiento induce un cambio de percepción sobre el entorno, cerrándose así un ciclo de modo continuo.

Dentro de la cadena de tratamiento de la información, tradicionalmente, se distinguen distintos tipos de procesos y representaciones. En primer lugar, tenemos los procesos perceptivos, encargados de ir acumulando una conveniente representación interna del entorno **R** a partir de la secuencia de imágenes. Sobre la base de esta representación, un subsistema de planificación se encarga de la toma de decisiones sobre el camino **P** a seguir por el agente, de acuerdo con unos objetivos de navegación posiblemente determinados *a priori*. Finalmente, el plan **P** es leído por los módulos encargados de generar el control **C**, con el fin de producir en última instancia el movimiento.

La figura anterior puede considerarse una visión clásica más o menos consensuada de la definición del problema. Sin embargo, en el momento de su aplicación, dependiendo de los tipos de representaciones o procesos empleados y de la importancia que se atribuye a cada uno de los componentes, encontramos muy diversos tipos de propuestas concretas de implementación. Por ejemplo, en algunos casos se concede gran importancia a las representaciones internas, mientras que otras soluciones son abiertamente no representacionales y persiguen comportamientos donde el cierre del lazo entre la percepción y la acción se realice del modo más inmediato posible. De modo coherente con cada uno de los respectivos enfoques, el con-

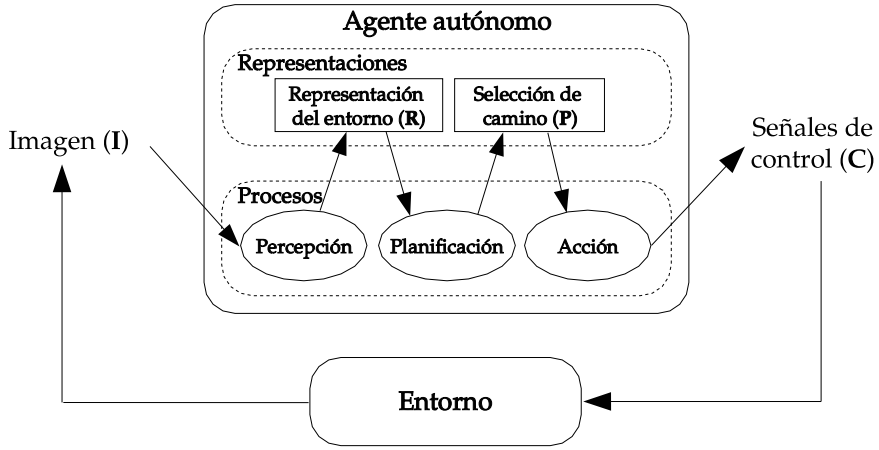


Figura 4.1: *Esquema genérico de los sistemas de navegación visual para agentes autónomos.*

cepto de percepción, que es el que fundamentalmente nos ocupa en esta tesis, es entendido de muy diversas maneras en los distintos trabajos. Así, mientras que para algunos incluye necesariamente la abstracción de los estímulos sensoriales en algún tipo de representación de complejidad variable, para otros consiste simplemente en la asociación más o menos directa entre imágenes y órdenes de control. En las secciones que siguen enumeramos y clasificamos algunos de los trabajos sobre el tema, tratando simplemente de bosquejar una taxonomía sobre la gran variabilidad de acercamientos posibles, aun siendo conscientes de la inevitable incompletitud de cualquier intento al respecto.

4.2.1. Con conocimiento del entorno

Un primer grupo englobaría aquellos sistemas en los que se dispone de algún tipo de modelo previo del entorno, que se intenta relacionar con la imagen o secuencia de imágenes recibidas en cada instante. Entra dentro de esta categoría, por ejemplo, ya mencionado trabajo de Lee *et al.* (2000), donde se disponía de una base de datos de localizaciones caracterizadas por una serie de invariantes proyectivos. A partir de ellos se puede determinar la situación actual del móvil en el escenario, y tomar la decisión adecuada para realizar una navegación guiada por objetivos, como ir de un lugar conocido a otro usando un mapa topológico de las localizaciones. Otro ejemplo similar es el propuesto por Lundquist (1997), donde los distintos lugares se caracterizan por la posición de una serie de líneas verticales de la escena (por ejemplo, esquinas, marcos de puerta, etc.), guardados en forma de lista antes de la puesta en marcha del sistema. Durante la operación se localizan las líneas verticales en la imagen, para realizar un *matching* con el modelo previo del entorno y estimar la posición relativa de la cámara respecto al mismo. En este trabajo concreto se utiliza una sola cámara colocada en el robot horizontalmente y con eje óptico paralelo al suelo. De este modo se reduce el problema de la estimación de posición a dos dimensiones, aunque conservando la perspectiva. Este

sencillo sistema de autolocalización permite realizar de modo relativamente preciso algunas tareas de navegación como el "aparcamiento" (*docking*) del robot en una zona determinada.

Utilizando ya modelos completamente tridimensionales, muchos trabajos con el enfoque llamado *visual servoing* caben también dentro de este apartado. El *visual servoing* se caracteriza por relacionar directamente las coordenadas de imagen de puntos o líneas cuya posición tridimensional es conocida con las órdenes de navegación, a través de un estudio geométrico que relaciona el movimiento de cámara con el flujo óptico de dichas características. Así, si se tiene un modelo 3D del entorno donde queremos movernos, se conoce la posición inicial del robot, y se quiere realizar una determinada trayectoria dada como una secuencia odométrica proporcionada en lazo abierto, se puede cerrar un bucle de control estable aplicando el estudio anteriormente realizado. Algunos ejemplos son las propuestas de Swain y Devy (1997), y Taylor *et al.* (1999), en el campo de los robots navegantes, o de Drummond y Cipolla (2002), en el manejo de brazos robóticos.

Un modelo más básico lo constituye la aproximación *intrusiva*, basada en la introducción de *landmarks* artificiales en la escena (líneas, marcas reflectantes, etc.) para ser utilizadas en las tareas de localización y posterior navegación. Un ejemplo típico es el descrito por Beccari *et al.* (1997). Se trata de un robot que navega siguiendo líneas rectas negras e interpretando un repertorio de señales de tamaño fijo, colocadas todas en el suelo. Tras una sencilla etapa previa de corrección de la perspectiva, que emplea una tabla de transformación precalculada por estar el sistema precalibrado y la cámara fija respecto al móvil, se practica un sencillo umbralizado de color para detectar y seguir las marcas. Las señales rectificadas son también clasificadas para guiar la navegación, usando una red neuronal preentrenada a tal efecto.

En algunos sistemas, finalmente, aún no disponiéndose de modelos previos del entorno, se explota el conocimiento sobre ciertas peculiaridades del mismo para realizar la navegación. Son las denominadas aproximaciones *ad-hoc* comentadas en la sección 3.3.3 del capítulo anterior, y en donde no existe ningún tipo de representación interna **R**, puesto que lo único que se pretende es lograr comportamientos esencialmente reactivos al entorno directamente perceptible mediante procedimientos sencillos adaptados a la tarea, como por ejemplo el seguimiento de contornos en la imagen (Herman *et al.*, 1997; Robert *et al.*, 1997).

4.2.2. Métodos basados en apariencia

Este segundo grupo de técnicas tiene también una fase previa de adquisición, en la que se almacena una base de datos de situaciones conocidas. Pero en este caso cada localización se caracteriza por su imagen cruda, sin ninguna extracción de características, aunque típicamente submuestreada buscando la eficiencia. La base de datos sirve para construir una memoria asociativa, más o menos optimizada con algún método de aprendizaje supervisado, que liga directamente imágenes de entrada con órdenes de control. No se busca, pues, ningún tipo de estructuración o modelización del entorno. Evidentemente, la aproximación tiene posibilida-

des de éxito sólo en dominios restringidos y en condiciones muy acotadas, dado que en otro caso se estrella contra la explosión combinatoria de las virtualmente infinitas posibilidades de la entrada. Weng *et al.* (1997), por ejemplo, describen un sistema de tales características, donde las imágenes prealmacenadas se organizan de forma jerárquica. Otras veces la organización de la base de datos se hace usando un modelos de tipo *eigenspace*. Es el caso de Jogan *et al.* (2003), que trabajan con imágenes panorámicas tomadas con una cámara omnidireccional. Los sistemas de este tipo tratan de recoger toda la variabilidad del entorno en un espacio de dimensión mucho menor que la de las imágenes originales, utilizando sólo las direcciones más expresivas de la muestra. El modelo del entorno está entonces representado por un subespacio optimizado donde se encuentran las imágenes de entrenamiento, asociadas cada una a una descripción mucho más corta basada en un sistema de coordenadas local a dicho subespacio.

En ocasiones se añaden sofisticaciones mayores sobre la idea original, como la posibilidad de incrementar dinámicamente el modelo durante la operación, eliminando las teóricas limitaciones de la clásica división entre las fases de aprendizaje y funcionamiento. Pero la defensa que sus autores suelen hacer del amplio espectro de aplicabilidad de estas técnicas, por su pretendida generalidad, no queda experimentalmente demostrada casi nunca. A menudo, incluso critican los intentos de extracción de información de más alto nivel, es decir, de explicar la imagen de entrada en términos de algún tipo de primitiva, argumentando la fuerte dependencia del dominio de estos intentos interpretativos. Si bien no les falta razón en muchas de las críticas, cerrar directamente un ciclo de control $\mathbf{C}$=f($\mathbf{I}$,$\mathbf{P}$) (imagen cruda + selección de camino = orden de control) no deja de ser absolutamente inviable en el momento en que se exige cierta capacidad de inducción o extrapolación al sistema: evidentemente, sólo cabe esperar de él que repita lo que ha sido explícitamente enseñado a hacer en un momento dado.

En un intento de ganar en versatilidad, los métodos basados en apariencia se combinan también a veces con la geometría proyectiva. Es el caso de Basri *et al.* (1998), que para generar el movimiento hacia un lugar desde donde se tomó la imagen modelo buscan correspondencias con esquinas de la imagen actual, como es habitual en las técnicas proyectivas del tipo *shape from motion* o *shape from stereo*. Entonces se determina la matriz fundamental para navegar utilizando la posición del epipolo en la imagen de entrada, puesto que ésta debe marcar la dirección adecuada de movimiento hacia el lugar de interés si la cámara está calibrada. Frente a los trabajos mencionados anteriormente, en éste no se renuncia a la extracción de características, como vemos. También Frizera *et al.* (2000) combinan el uso de la geometría de las líneas laterales de un pasillo para navegar centrados en él con un método basado en apariencia puro, en un esquema híbrido. Cada vez que se toma una imagen nueva, se calcula la suma de diferencias al cuadrado, píxel a píxel, con una base de datos de imágenes a cada una de las cuales le corresponde con una acción especial, del tipo girar en una esquina, atravesar una puerta, etc. Si la diferencia con alguna de ellas está por debajo de un umbral determinado,

se dispara la acción de control adecuada. En este caso, además, la base de datos de imágenes puede ordenarse con un criterio de causalidad marcado por el orden en el que el robot se las puede ir encontrando.

Un método algo más interesante, también basado en la utilización directa de la imagen sin extracción previa de características, es la técnica de *mosaicing*. Consiste en fijar una cámara en una posición determinada del móvil, para tomar imágenes del suelo y localizarse posteriormente en un mosaico global de toda la escena utilizando alguna técnica de *template-matching*. El mosaico puede construirse previamente, de modo supervisado (al estilo de los métodos del apartado anterior) o bien durante el mismo proceso de navegación, en un proceso simultáneo de construcción del mapa y posterior autolocalización (lo que se conoce comúnmente en la literatura como SLAM, del inglés *Simultaneous Localization and Map Building*). La técnica ha sido empleada con éxito en entornos relativamente grandes, donde se requiere una gran capacidad de almacenamiento para el mosaico, así como de procesamiento para el *matching* (Kelly, 2000).

4.2.3. Búsqueda de espacio libre

Se persigue aquí una estructuración local muy elemental, que sólo busca espacio libre en el entorno inmediato, pero al menos se trabaja ya sobre escenarios en principio completamente desconocidos. El ya mencionado sistema autónomo de Zhang y Faugeras (1992), por ejemplo, va construyendo desde cero un modelo tridimensional con los segmentos obtenidos de las imágenes de tres cámaras a bordo la plataforma. A partir de esta reconstrucción, y utilizando una triangulación incremental de Delaunay, se estiman posibles huecos de espacio libre y se planifica en función de ellos la trayectoria de navegación. Las particularidades del procedimiento se pueden consultar con más detalle en (Buffa *et al.*, 1992). Básicamente, el espacio libre se calcula trazando rayos desde la posición del robot hacia la triangulación practicada sobre el suelo delante de éste. Todos aquellos triángulos que puedan atravesarse sin topar con un segmento 2D se marcan como libres. Se tiene entonces un "contorno frontera", entre los triángulos accesibles y los no accesibles, en función del cual se pueden tomar decisiones de navegación hacia el correspondiente espacio libre. La potencia expresiva de los segmentos es también aprovechada para la realización de diversas tareas, como la calibración (Zhang *et al.*, 1993), la determinación del movimiento (Zhang y Faugeras, 1991) y la reconstrucción euclídea del entorno (Zhang, 1995).

En una línea similar se encuentra el robot con cabeza estéreo descrito por Beardsley *et al.* (1995). El robot realiza reconstrucciones afines de una nube de puntos 3D, lo que elimina la necesidad de calibrar. Puesto que el punto medio de un conjunto de puntos es un invariante afín, el movimiento se puede planificar buscando huecos, por ejemplo.

También buscando espacio libre, pero en un acercamiento más sencillo, que no exige ninguna etapa de extracción de características, se encuentra la utilización directa del flujo óptico.

Por ejemplo, Camus *et al.* (1999) utilizan esa información para calcular el tiempo de impacto con el objeto más cercano y cerrar un lazo de control capaz de navegar periodos de tiempo más o menos largos evitando los obstáculos circundantes.

En general, sin embargo, cualquier navegación hacia espacio libre comparte los inconvenientes básicos de la navegación reactiva. Algunos autores, en efecto, critican este tipo de aproximaciones como muy limitadas, puesto que imposibilitan la navegación orientada hacia objetivos en entornos a media o gran escala (Riseman *et al.*, 1997).

4.2.4. Mapas de ocupación

Concediendo una importancia mayor a la representación **R**, tenemos las técnicas apoyadas en los denominados mapas de ocupación. Se trata de representaciones probabilísticas discretas del conocimiento espacial del entorno del robot, en forma de *grids* o rejillas de elementos finitos. En cada celda se acumula de alguna forma la evidencia de que el robot se encuentre o no en ella, y también quizá de la presencia de determinados tipos de obstáculos. Como observamos, pues, a pesar de que hay ya una preocupación por la creación de una representación interna a partir de la entrada sensorial, no se tiene aún una clara vocación de estructuración del entorno, al menos en el sentido de buscar algún tipo de regularización.

Un modo de actualización del *grid* muy empleado es el basado en el esquema bayesiano recursivo clásico, con una fase de predicción y otra de ajuste. En la primera se emplea un modelo de movimiento para, partiendo del estado actual y la orden de control proporcionada, obtener posteriormente el posible estado futuro. En la segunda se usa un modelo de medida para corregir la predicción anterior usando alguna medida visual tomada del entorno a partir de la imagen **I** de entrada. Se actualiza así la densidad de probabilidad en cada posición de la rejilla mediante un bucle en el que la información visual nos resitúa en todo momento en el mapa corrigiendo a la información odométrica. El principal problema es el cómputo intensivo. A veces se aplican técnicas para no actualizar siempre todo el *grid*, sino sólo una parte relevante, con la idea de adaptar el método para su funcionamiento en tiempo real. Un buen trabajo que trata todas estas cuestiones es el realizado por Fox *et al.* (1999). Aunque no está basado en la percepción visual, sino que se aplica a medidas de sónar, describe de modo bastante serio el tratamiento eficaz de la incertidumbre, incluyendo tanto la proveniente de las medidas de ultrasonidos como la de los sensores propioceptivos del movimiento.

Ya con sensores visuales, Howard y Kitchen (1997) aprovechan los bordes entre el suelo y las paredes para, tras una corrección de la perspectiva, crear un *grid* que se va actualizando con un esquema bayesiano similar al anterior, usando también la odometría. Alternativamente, Moravec (1996) emplea rejillas de evidencia tridimensionales que se actualizan usando un par de cámaras precalibradas. Esta representación se discretiza con grano más o menos grueso dependiendo de la capacidad de cómputo disponible, y se va llenando con la posición de puntos 3D localizados con el par estéreo. Sobre este modo denso de cubrir el espacio se pro-

pone desde la sencilla navegación a través de las celdas vacías, hasta una más deliberativa si se reconocen determinados patrones de ocupación de las celdas. Aún así, los resultados expuestos están más encaminados a ilustrar la calidad de las representaciones obtenidas que a justificar el uso de éstas en la navegación.

Otras veces se intenta crear algún tipo de estructuración elemental, por ejemplo creando un mapa de carácter más topológico, en lugar de emplear un *grid* regular de grano fino. Estos mapas suelen ser incrementales, e incorporan mecanismos de autoorganización para ir cubriendo de modo más o menos denso el espacio de movimiento. Por ejemplo, Mallot *et al.* (1997) construyen un mapa donde cada localización se caracteriza por una signatura unidimensional de la imagen, y los nodos correspondientes se van conectando mediante arcos orientados, utilizando la información odométrica. El problema de la navegación puede después plantearse en términos de sistemas dinámicos con espacios temporales y de estados discretos, y resolverse usando modelos de Markov, por ejemplo (Dean y Marion, 1997).

Finalmente, en lugar de mantener información sobre la incertidumbre de la posición en absolutamente todas las celdas del mapa discretizado, es posible utilizar también métodos basados en el muestreado. En este caso se representa la distribución probabilística de la posición del robot en el entorno como un conjunto de ejemplos aleatoriamente escogidos. El conocido método *CONDENSATION* (de *Conditional Density Propagation*) entra dentro de este grupo (Isard y Blake, 1998). En el ámbito concreto que nos ocupa, Dellaert *et al.* (1999) describen una aplicación muy sencilla de este algoritmo para localizar un robot en un museo. El robot está equipado con una cámara que mira al techo, y obtiene un solo valor escalar de intensidad de gris. Entonces se aplica un razonamiento probabilístico bayesiano similar al descrito para los *grids*, con un modelo de movimiento y un modelo de medida para determinar en cada momento el lugar del museo en el que se encuentra el robot. Pero en lugar de mantener un mapa completo del entorno, se retiene sólo un conjunto de muestras de posición, que constituye una representación no paramétrica de la densidad de probabilidad de la situación del robot, al estilo de los métodos de Montecarlo. Se disminuyen los requerimientos computacionales con respecto a los *grids* de grano fino, pero aún así hay que mantener un conjunto grande de muestras para lograr un mínimo de robustez. El ejemplo descrito, además, utiliza conocimiento previo del entorno, aunque también se apunta la posibilidad de que éste pueda ir creándose y actualizándose durante la misma operación.

4.2.5. Estructuración del entorno

Finalmente, trataremos en este punto aquellos trabajos en los que hay una mayor intención de estructuración del entorno, y que por tanto están más relacionados con nuestra propuesta. En ellos se trata de adquirir dinámicamente una representación interna **R** lo suficientemente potente como para hacer viable una planificación elemental de la navegación **P**, permitiendo así la generación de comportamientos más complejos.

Como primer acercamiento, tendríamos la simple construcción y posterior mantenimiento de estructuras de anclaje del sistema autónomo con respecto al mundo real, creadas a partir del propio entorno natural. Constituirían lo que podríamos llamar *landmarks* naturales, frente a los artificiales de los esquemas *intrusivos*. Con ellas se puede cerrar un lazo de control robusto y establecer una relación básica entre percepción y acción. Sin embargo, aún no queda claro como tomar decisiones de navegación relativamente complejas, más allá de la mera navegación hacia el espacio libre o el seguimiento de objetos. Quizá lo más interesante en estos sistemas es la generación de modelos parciales dirigidos por los datos (*bottom-up*), en un proceso en el que, a partir de los puntos de anclaje cuyas posiciones parecen fiables tras movimientos amplios del robot, se posibilita la puesta en marcha del proceso de generación de modelos iniciales (*bootstrap*).

Por ejemplo, Davison y Murray (1998) ponen el énfasis en construir mapas de características manejando de modo adecuado la incertidumbre. Las características, de nuevo, son puntos 3D obtenidos a partir de una cabeza estéreo precalibrada. También al igual que en algunos trabajos anteriores, se tiene un modelo de medida y otro de movimiento, ambos con su correspondiente información de ruido. En este caso se trata la incertidumbre con un filtro de Kalman extendido. Como peculiaridad frente a modelos anteriores, se aplica aquí un criterio de utilidad de las características, restringiendo el número de ellas que se siguen para minimizar la incertidumbre en la autolocalización. Se va creando así un mapa elemental en el que dinámicamente se añaden unas características y se eliminan otras de modo automatizado. Este mantenimiento del modelo se basa en la corroboración obtenida tras el propio movimiento de la plataforma y en la disminución de incertidumbre que se gana al añadir cada nueva información.

Algunos autores adoptan esquemas similares de detección, predicción y búsqueda con otro tipo de características, como líneas 3D (Deriche y Faugeras, 1990). Riseman *et al.* (1997) utilizan también los segmentos como herramienta básica, pero en este caso para localizar modelos de agrupación más completos, respecto a los cuales se calcula en cada momento la posición de la plataforma. Es una filosofía parecida a las anteriores, pero aumentando la complejidad de las estructuras de anclaje. El enfoque está basado en un trabajo previo donde se trata de resolver el problema conjunto del *matching* y la estimación de la transformación óptima que superpone un modelo de segmentos sobre líneas de datos extraídas de la imagen (Beveridge y Riseman, 1997). En principio, se trataría de modelos de estructura conocida, por lo que entraría en el enfoque de los *landmarks*. Pero también se apuntan algunas técnicas para realizar una puesta en marcha desde cero, en la que las estructuras de anclaje se generen durante el mismo ciclo operativo.

En general, en todos estos estudios se concede más importancia a la creación de la estructura como apoyo a la autolocalización que al hecho en sí de planificar el camino y tomar decisiones de navegación. En este sentido, un paso más allá sería imponer algún tipo de regularización a la estructura construida. A veces, se puede tratar simplemente de una regulariza-

ción por criterios de consistencia espacial. Vicente (2002), por ejemplo, construye una esfera centrada en el robot que cubre de modo denso la estructura 3D del espacio que le rodea. Se trata de una esfera de profundidad que se inicia a partir de la triangulación estéreo de puntos que aparecen en la imagen con suficiente textura como para realizar con fiabilidad el *matching*. Después se llena el resto del espacio mediante un proceso de difusión similar al empleado en el capítulo 3 para la recuperación de las imágenes a color a partir de los segmentos, pero en este caso trabajando con las posiciones espaciales de las esquinas 3D conocidas. La esfera circundante así construida se utiliza para condicionar el movimiento de la cabeza estéreo, que realiza *sacádicos* dirigidos a completar y desambiguar paulatinamente la representación bajo un paradigma de visión activa. Esta estructuración elemental por continuidad espacial sirve para dotar al sistema de cierta capacidad de anticipación sobre el entorno.

Muchos autores utilizan también los segmentos como herramienta para imponer restricciones sobre la estructura del espacio. Lebegue y Aggarwal (1993), por ejemplo, *fuerzan* la percepción de líneas en tres direcciones únicas del espacio, correspondientes a la anchura, altura y profundidad de una habitación. La cámara está calibrada intrínseca y extrínsecamente respecto al robot y, a través de la odometría, también se conoce su posición con respecto al sistema de coordenadas del suelo. En estas condiciones pueden computarse los puntos de fuga de las tres direcciones principales de dicho sistema (las dos horizontales, X e Y, y la vertical Z) y, consecuentemente, la orientación local que debe tener un borde en un determinado punto de la imagen para corresponder a un segmento en cada una de las direcciones predeterminadas. Tomando esta información como base, se desarrolla un rápido algoritmo de detección de ese tipo de segmentos, buscando puntos salientes de borde en los cuales la dirección del gradiente está alineada con alguna de esas tres direcciones. El sistema es así capaz de trabajar en tiempo real, reestimando en cada momento la posición de la plataforma para no perder precisión por la odometría acumulada (*dead-reckoning*). La aplicabilidad es sin embargo limitada, puesto que sólo se detectan segmentos en cada una de las tres direcciones elegidas, y se necesita información de la posición del robot respecto al mundo para operar.

En escenas de interiores, por último, y utilizando ideas más cercanas a nuestra propuesta, se puede estructurar también el entorno a través de la localización y posterior ajuste dinámico de los planos principales de la escena, como el suelo, las paredes, etc. Es, por ejemplo, el caso de Iocchi *et al.* (2000). Se trata, una vez más, de un robot dotado de odometría y un par estéreo que va reconstruyendo puntos 3D en zonas con suficiente detalle. A las proyecciones sobre el suelo de estos puntos se les aplica la transformada de Hough para intentar encontrar planos verticales. El robot va incorporando al moverse nuevos puntos a los planos ya existentes o, si se detecta evidencia suficiente, crea un plano nuevo y lo añade también al modelo. El sistema es incluso capaz de practicar reconstrucciones densamente texturizadas del entorno, utilizando un modelo de perspectiva inversa y el correspondiente alineado de las imágenes, aunque esta parte del trabajo tiene que realizarse ya *off-line* por exigir una capacidad de cómputo superior a la permitida durante la navegación en tiempo real.

Sobre la misma idea de estructuración planar trabajan Liu *et al.* (2001), pero con más resolución por la ayuda de un láser montado sobre el robot. Se usa aquí el algoritmo de *expectación-maximización* (EM) como herramienta de regularización, de modo análogo al empleado en (López de Teruel y Ruiz, 1997) y (López de Teruel y Ruiz, 1998) para detectar segmentos en imágenes 2D. A diferencia de éste último, sin embargo, los puntos de entrada utilizados son tridimensionales, y obtenidos con un láser de barrido vertical que recorre un plano perpendicular a la dirección de avance del robot. El algoritmo EM se utiliza como procedimiento de maximización de la verosimilitud de los datos sobre un modelo de mezcla de componentes, cada uno de los cuales captura un plano 3D, más un componente adicional que recoge el ruido del sensor (para captar *outliers*). Cada cierto tiempo se actualiza el número de componentes haciendo uso de ciertas heurísticas, tales como borrar planos que no tengan suficiente soporte o considerar periódicamente ternas de puntos 3D aleatoriamente muestreados para añadir planos tentativos a la mezcla.

Un trabajo más moderno sobre modelado poliédrico de entornos estructurados es el descrito por Montiel y Zisserman (2001). Aquí las líneas y sus puntos de fuga se utilizan para elaborar sobre la marcha modelos arquitecturales muy simples de la escena, aprovechando también la peculiaridad del movimiento en trayectoria paralela al suelo de la cámara. Como vemos, todas estas características lo relacionan doblemente con nuestra aproximación.

Cabe decir aquí, por último, que existen asimismo multitud de trabajos que regularizan el entorno a través de planos en otros campos de la visión artificial, no necesariamente orientados a la navegación visual. Citaremos algunos de ellos en la sección 4.4, cuando expliquemos nuestra propuesta concreta de modelización.

4.3. Navegación interpretativa

El paradigma de navegación visual que nosotros proponemos se enmarca dentro del último tipo de los descritos en el apartado anterior, puesto que busca categorizar cada situación en función de los estímulos sensoriales elementales con el fin de posibilitar una planificación deliberativa del movimiento. En algún lugar por encima de la extracción de características, por tanto, debe haber un procedimiento de más alto nivel que juegue con la agrupación de aquéllas para elaborar una hipótesis congruente con la entrada sensorial actual. El objetivo último es la generación de una interpretación consistente en términos de *situaciones tipo* reconocibles que, debidamente categorizadas, sean adecuadas para decidir el comportamiento del agente.

En este mecanismo de percepción de alto nivel cabría distinguir dos tipos de flujos de información, no excluyentes, sino complementarios. En primer lugar, estarían los procedimientos que podríamos llamar de tipo ascendente (*bottom-up*), o dirigidos por los datos. Éstos, en principio, tratan de generar estructuras tentativas progresivamente más complejas a partir de los estímulos sensoriales crudos, involucrando procesos heurísticos de prueba y error. Este

proceso de extrapolación que pone en marcha las hipótesis interpretativas forma el núcleo de la percepción inteligente. En segundo lugar, tendríamos los procesos duales a los anteriores, descendentes (*top-down*), cuya misión es tratar de verificar los modelos obtenidos sobre la secuencia de datos proporcionada de modo continuo por los sensores, con el fin de confirmar la validez de las hipótesis anteriores, e incluso de guiar a los procesos ascendentes para evitar la explosión combinatoria de interpretaciones alternativas.

Si, como es nuestro caso, se pretende trabajar en un entorno desconocido, está claro que son los procedimientos del primer tipo los que deben actuar en primer lugar. Es, pues, primordial acceder de alguna manera inicial a la estructura aproximada de la escena, en base a la cual pueda entrar la posterior presión descendente. En el ámbito concreto de las reconstrucciones tridimensionales, existen diversas técnicas para extraer la información 3D a partir de la 2D, según las pistas existentes en las imágenes que se utilicen como entrada[1]. Algunos autores llaman a estas técnicas "*forma a partir de X*" (del inglés *shape from X*), siendo X los distintos tipos de propiedades en las que se apoya la extrapolación (Pajares y de la Cruz, 2000). Entre otras, las siguientes posibilidades son las utilizadas más frecuentemente:

Forma a partir de la visión estereoscópica: Se trata del enfoque clásico de la geometría proyectiva, en el que se recupera la información 3D con un procedimiento de triangulación a partir de imágenes de una escena tomadas desde dos (o en general más) puntos de vista distintos.

Forma a partir del movimiento: La idea es la misma que la anterior, pero en este caso se trabaja con una sola cámara que se mueve variando su posición en el tiempo. Habitualmente, por tanto, las técnicas no difieren mucho de las empleadas en el caso precedente.

Forma a partir de las sombras: Aquí la estructura tridimensional se obtiene aprovechando el hecho de que las superficies con distinta orientación aparecen con distintos grados de luminosidad en la imagen. Normalmente es una técnica muy sensible a las condiciones ambientales de iluminación, y por tanto bastante propensa al ruido.

Forma a partir de la textura: Sacan la forma a partir de la distorsión experimentada por un determinado patrón de textura. Si el patrón utilizado tiene suficiente detalle, pueden llegar a tener mayor estabilidad que las anteriores.

Forma a partir del enfoque: En este último caso la profundidad de la escena se mide buscando la posición de enfoque óptimo para cada punto, usando técnicas de visión activa.

En esta tesis se propone un paradigma alternativo, no incluido en la clasificación anterior. Se trata de la extracción de la información 3D a partir de una sola imagen realizando una interpretación de la misma, en una aproximación a la que podríamos llamar "*forma a partir de la*

[1]Otra opción sería el apoyo en sensores de profundidad de otros tipos, como los ultrasonidos o el láser, por ejemplo. No entraremos en ellos, sin embargo, dado que se salen del ámbito de la percepción exclusivamente visual.

interpretación". Teniendo en cuenta las características del dominio de trabajo, se asigna un modelo de datos predeterminado a la realidad, para intentar un *matching* con la imagen actual. El objetivo es la generación de estructuras tridimensionales, que, debidamente categorizadas, hacen el doble papel de modelos para las posteriores presiones *top-down* y de elementos motivadores para el comportamiento dinámico del agente. Este conjunto de situaciones tipo ha de ser convenientemente *configurable* y *parametrizable*, con el fin de lograr un grado de flexibilidad suficiente que permita su aplicabilidad en condiciones realistas. Para reconocer una determinada situación, pues, hay que identificarla con el correspondiente modelo ideal, pero éste debe ser lo suficientemente fluido como para permitir su instanciación práctica en las circunstancias cambiantes de los diversos escenarios del dominio.

Nuestra arquitectura de percepción se basa en un mecanismo bidireccional, mezcla de construcción, destrucción, reagrupamiento y reordenación de estructuras tentativas. Las hipótesis interpretativas se generan utilizando diversos tipos de heurísticas a partir de las pistas sensoriales iniciales (*bottom-up*), y entran después en una fase de validación a partir de la consistencia con el propio movimiento (*top-down*). Las estructuras con una cierta persistencia quedan entonces consolidadas, para pasar a ser seguidas durante el movimiento y utilizadas en la navegación. El ataque trata de romper la explosión combinatoria inducida por el constante intento de interpretación desde cero, que usualmente conduce a sistemas ineficaces y poco robustos. Continuamente se reproyecta el modelo interpretado sobre la información sensorial cruda, de modo que lo corroborado necesita de poca evidencia para seguir siendo percibido y seguido. Por el contrario, las hipótesis que no muestran una cierta consistencia en el tiempo son refutadas y no se incluyen en la interpretación. El enfoque es viable porque la velocidad de los cambios en el entorno, determinada por el propio movimiento del robot, se produce en una escala de tiempo inferior a la de la constante actualización del modelo. Esto se puede lograr sólo bajo un paradigma anticipativo, nunca sobre un permanente mecanismo de interpretación *bottom-up*.

En este sentido, los sensores propioceptivos juegan un papel igualmente primordial. A través de ellos, el ciclo perceptivo con flujos de información bidireccionales dota al sistema de la deseada capacidad de predicción sobre el entorno. Como veremos, será posible incluso perder de vista temporalmente ciertas partes de la estructura para recuperarlas con posterioridad, mediante su conservación en una memoria a corto plazo que es incesantemente actualizada, aún a ciegas, a través de la odometría. De esta manera se gana en robustez y continuidad en el comportamiento global, al tiempo que se atenúa la sensibilidad a posibles obstáculos móviles, por ejemplo.

En resumen, la visión de alto nivel debe integrar el proceso perceptual con un modelo de entendimiento del mundo. Evidentemente, el sistema no puede centrarse solamente en la búsqueda reiterada del modelo a partir de los datos, lo que suele llevar a serias limitaciones. Junto con el mecanismo eficiente de arranque de hipótesis, pues, es básico confirmar y corregir las mismas durante el movimiento, en busca de la convergencia a un modelo estable del

entorno cercano. También lo es, por supuesto, la correcta conmutación a situaciones distintas en el momento adecuado. El mantenimiento de la hipótesis actual, además, debe ser capaz de ubicar en todo momento al agente autónomo dentro de cada situación, a fin de que se puedan tomar las decisiones de control pertinentes. Hay que destacar aquí que el subsistema de percepción no sólo encuentra huecos de espacio libre, sino que toda la información estructural se pone a disposición del subsistema de actuación para permitir una planificación premeditada de la trayectoria. Sobre esta idea, por tanto, se cierra el lazo entre la percepción y la acción en el que se apoya la navegación interpretativa.

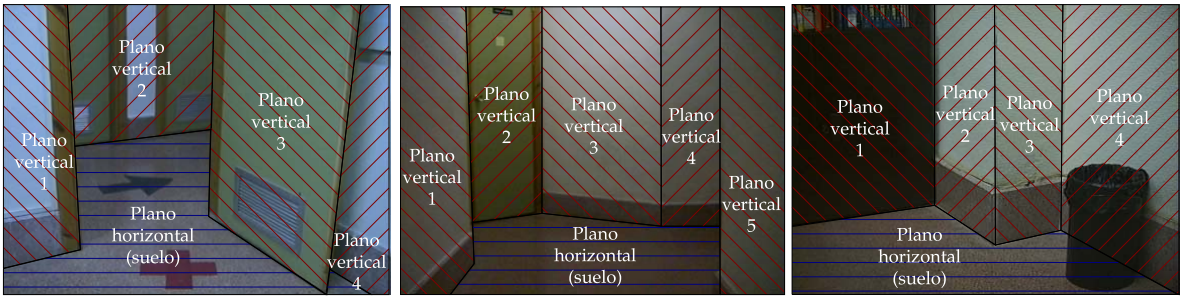
En los apartados que siguen profundizaremos en los detalles técnicos de la propuesta. En primer lugar, describiremos el modelo planar utilizado, justificando su adecuación al dominio que nos interesa, los entornos parcialmente estructurados de interiores. La descripción se centrará especialmente en la geometría necesaria para el manejo eficiente del modelo. En el apartado posterior se describe la arquitectura procedimental asociada, construida sobre el ciclo de *interpretación*, *predicción* y *corroboración* cuya filosofía hemos introducido en esta sección.

4.4. Representación planar de entornos estructurados

4.4.1. Descripción

En este apartado describiremos el modelo representacional elegido. Como todo modelo, el que nosotros utilizaremos introduce una serie de restricciones y simplificaciones sobre la realidad representada. En nuestro caso, si cabe, las restricciones están mucho más justificadas, dada la extrema complejidad del problema de la percepción en su sentido más amplio. En principio, la elección debería realizarse en función de las características del dominio, pero, incluso con esta idea en mente, las posibilidades son múltiples. De hecho, ni siquiera hay un claro acuerdo entre los distintos autores a la hora de definir las características que cabría exigir a esta clase de especificaciones. No en vano, como ya se ha comentado, existe incluso una corriente de opinión abiertamente *no representacional* que niega su necesidad, intentando cerrar un lazo directo entre la percepción y la acción sin utilizar ninguna representación explícita intermedia.

Como se ha venido subrayando a lo largo de la exposición, nuestro sistema autónomo ha sido diseñado para moverse en un tipo de entorno determinado, los espacios interiores de edificios con un grado medio-alto de estructuración. A ellos trata de adaptarse ventajosamente el modelo elaborado, explotando las restricciones que usualmente encontramos en este contexto. En concreto, la característica más definitoria de la representación empleada es la reducción del entorno a un conjunto de planos en diferentes orientaciones del espacio. Más aún, las orientaciones permitidas están también restringidas a planos horizontales y verticales, en respectiva correspondencia con el suelo y las paredes, puertas, elementos del mobiliario, etc.,



(a) (b) (c)

Figura 4.2: *Modelos planares de distintas escenas de interior. Los planos verticales se muestran con entramado diagonal rojo, y el de suelo con entramado horizontal azul.*

de las construcciones realizadas por el hombre.

El modelo, que de entrada podría parecer demasiado restrictivo, se ajusta sin embargo bien al microdominio que trata de representarse. Si se observan, por ejemplo, las distintas imágenes de la figura 4.2, tomadas por el robot durante periodos reales de navegación, se comprueba que todas ellas pueden explicarse en términos de un plano horizontal (el suelo) más una serie de planos en orientaciones arbitrarias respecto a la cámara, pero siempre verticales (paredes, puertas, muebles). A su vez, algunos objetos pueden estar contenidos dentro de los planos, como las señales que aparecen en el suelo o las rejillas de las puertas que aparecen en la figura 4.2(a), por ejemplo. Como veremos, gracias a la interpretación de la escena en términos planares estos objetos podrán ser situados en medidas reales respecto a la plataforma, así como adecuadamente reconocidos y clasificados, a pesar de sufrir en ocasiones notables deformaciones de perspectiva.

Ya adelantamos en la sección 4.2.5 que son muchos los robots autónomos que utilizan el modelo planar para navegar, sobre todo si se trata de entornos estructurados. Pero estos modelos de agrupamiento son también utilizados frecuentemente en problemas de reconstrucción y calibración a partir de múltiples vistas no necesariamente relacionados con la navegación, aprovechando la simplicidad en el manejo de las homografías inducidas por los planos entre las imágenes. Como muestras de esto último, cabría citar algunas aplicaciones alternativas en escenarios de interiores (Xu *et al.*, 2000; Baillard y Zisserman, 1999), o incluso sobre imágenes aéreas de ciudades (Schaffalitzky y Zisserman, 1999), por ejemplo.

4.4.2. Escalabilidad

Ocasionalmente pueden aparecer en la escena ciertos objetos no contenidos en los planos principales. Es el caso, por ejemplo, de la papelera de la figura 4.2(c). Hay que destacar que estos objetos, aún no entrando dentro del modelo, en principio no impedirían el ajuste del resto de componentes. En el proceso de interpretación se tratará de aprovechar el movimiento de la plataforma para facilitar la identificación de los planos aún en presencia de dicho

tipo de oclusiones. Esta posibilidad es muy interesante, puesto que aquellos elementos que no cumplen la restricción planar tampoco tendrían por qué degradar el comportamiento del procedimiento de modelado. A veces, incluso, estos obstáculos podrán ser, si no interpretados, sí al menos localizados a efectos de ser evitados en la navegación. El modelo es por tanto *escalable*, en el sentido de que, al menos *a priori*, admitiría ampliaciones para el tratamiento de casos no recogidos sin necesidad de entrar en conflicto con ellos.

Se trata, en el fondo, de otra versión del clásico problema de la distinción entre el fondo y la forma en la percepción. La resolución de este problema hace inevitable la existencia de una etapa de interpretación. En nuestro caso, la correcta categorización de la situación exige la discriminación entre el entorno en sí mismo, que definimos arbitrariamente como todo aquello que se ajusta al modelo planar (suelo, paredes, etc.), y el resto de objetos (obstáculos, señales), ajenos a dicha restricción.

4.4.3. Geometría del modelo

La ventaja del modelo representacional elegido es que, si se consiguen caracterizar bien los planos de interés a través de su descripción en términos de segmentos, la reconstrucción tridimensional del modelo planar a partir de una sola imagen es, como se verá, inmediata. Del mismo modo, se podrá también reconstruir y situar un contorno si se determina que está contenido en uno de los planos involucrados. Para todo ello únicamente necesitaremos conocer dos cosas:

- En primer lugar, la calibración intrínseca y extrínseca de la cámara respecto a la plataforma móvil. Como se vio en el capítulo anterior, el sistema es capaz de determinar esta información de modo autónomo, localizando la línea del horizonte y siguiendo la posición de algunas características (básicamente puntos o líneas) durante una serie de movimientos controlados de odometría conocida. Recordemos que, dependiendo de las simplificaciones empleadas en el modelo cámara-plataforma, disponíamos de varios procedimientos con distintos requerimientos en la cantidad y tipo de características a seguir, así como en el número de imágenes necesarias.

- En segundo lugar, la asignación de los segmentos de la imagen a los distintos componentes del modelo. Esto es, al plano del suelo, a los distintos planos verticales, a los objetos dentro de esos planos, o al resto de objetos no contenidos en los mismos. Posteriormente describiremos cómo se construyen estas interpretaciones de la escena. Por el momento simplemente supondremos que se dispone de dicha interpretación.

La reconstrucción más sencilla dada la imagen actual y la información de calibración será, obviamente, la de los elementos contenidos en el plano del suelo. La homografía $\tilde{\mathtt{P}}$ (resultante de la eliminación de la tercera columna de la matriz de proyección $\mathtt{P}$, determinada a su vez con cualquiera de los algoritmos 3.2 a 3.8) servía para transformar coordenadas del mundo de

puntos situados en dicho plano a coordenadas de imagen. La matriz inversa $\tilde{\mathtt{P}}^{-1}$, por tanto, nos permite realizar la tarea de reconstrucción deseada para todos los elementos del suelo. Ejemplos de tales reconstrucciones ya se mostraron en las figuras 3.12 y 3.13 del capítulo anterior. Utilizando una notación más consistente con el desarrollo que se seguirá en este apartado, construiremos una matriz $\mathtt{R}_{sue}$ de tamaño 4×3 que, a partir de la homografía de transferencia mencionada, realice la rectificación adecuada y añada la coordenada $Z_{rob} = 0$. De esta forma, simplemente premultiplicando las coordenadas homogéneas de un punto de la imagen perteneciente al suelo por esta matriz se obtendrán directamente las coordenadas tridimensionales (en el sistema S_{rob}) del punto del mundo correspondiente[2].

$$\mathtt{R}_{sue} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \tilde{\mathtt{P}}^{-1} \tag{4.1}$$

Para realizar las reconstrucciones correspondientes a los planos verticales, sin embargo, es necesario dar un pequeño rodeo, cuya interpretación geométrica se explica en la figura 4.3. Los puntos 3D pertenecientes a cada plano vertical se transferirán al de la imagen a través de una determinada homografía. La forma de esta homografía, obviamente, dependerá del sistema de coordenadas bidimensional que definamos sobre aquél (ya que el de la imagen se encuentra perfectamente definido). Una forma sencilla de establecer este sistema uniformemente para todos los posibles planos verticales de la escena es relacionarlo de alguna manera con el sistema tridimensional general del robot, S_{rob}. Para ello, podemos pensar simplemente en "tumbar" cada plano sobre el suelo, utilizando su línea de intersección con éste como eje de giro, tal y como se muestra en la figura en la figura 4.3(a). A partir de ese momento podremos utilizar el plano del suelo de S_{rob} (ejes X_{rob} e Y_{rob}) como sistema de referencia del plano vertical, y construir en base a él la homografía de transferencia de puntos al sistema de coordenadas de la imagen. Una vez fijada dicha homografía, como veremos, será muy sencillo construir una matriz de tamaño 4×3, similar a la matriz 4.1, capaz de realizar la rectificación de 2D a 3D para puntos en dicho plano. Al contrario que en aquélla, no obstante, esta otra matriz dependerá obviamente de las coordenadas de imagen de la proyección de la línea base del plano vertical involucrado.

En lo que sigue realizamos el desarrollo algebraico asociado al procedimiento descrito. Supóngase un determinado plano vertical $\boldsymbol{\pi}$ cuya recta de intersección con el suelo es $\boldsymbol{l} = (a, b, c)^\top$, en coordenadas homogéneas definidas sobre el plano XY del sistema de coordenadas del robot. Denotaremos esta parametrización del plano utilizando la expresión $\boldsymbol{\pi}(\boldsymbol{l})$.

[2]En este capítulo utilizaremos la letra $\mathtt{R}$ para denotar distintas matrices de reconstrucción 2D a 3D, que toman coordenadas de imagen como entrada y producen coordenadas del mundo como salida. Estas matrices, por tanto, tendrán todas el tamaño 4×3, y no deben ser confundidas con las matrices de rotación genéricas utilizadas en el capítulo anterior.

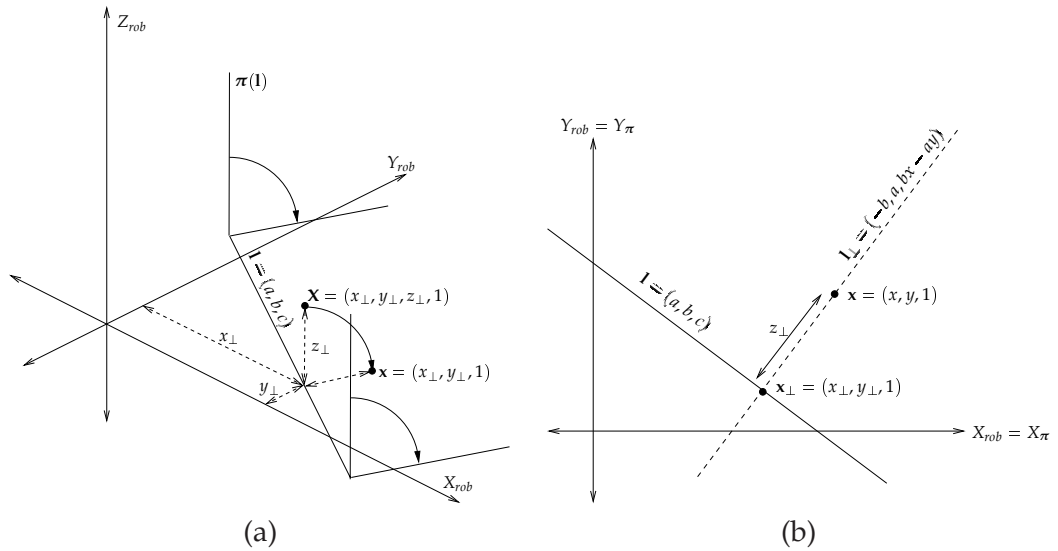


Figura 4.3: *(a) Rotación del plano vertical hasta situarlo sobre el suelo. (b) Proyección perpendicular de un punto sobre la recta de intersección entre el suelo y el plano vertical.*

A la hora de establecer una homografía que relacione directamente coordenadas de imagen con coordenadas del plano del espacio euclídeo $\boldsymbol{\pi}(\boldsymbol{l})$, hay que fijar primero el origen de coordenadas y la orientación de los ejes de dicho plano. El modo de hacerlo, como comentamos, consiste en utilizar el mismo sistema S_{rob}, girando previamente 90° el plano vertical sobre la recta de intersección $\boldsymbol{l}$, hasta hacerlo coincidir con el suelo. Esta situación se muestra en la figura 4.3(a). A partir de ese momento, cualquier punto 2D de coordenadas homogéneas $\boldsymbol{x} = (x, y, 1)^\top$ en el nuevo sistema $S_{\boldsymbol{\pi}}$ se corresponderá con un punto 3D original (en S_{rob}) $\boldsymbol{X} = (x_\perp, y_\perp, z_\perp, 1)^\top$, siendo $\boldsymbol{x}_\perp = (x_\perp, y_\perp, 1)^\top$ la proyección vertical de $\boldsymbol{x}$ sobre $\boldsymbol{l}$, y $z_\perp$ la distancia euclídea entre $\boldsymbol{x}$ y $\boldsymbol{x}_\perp$. El significado de estas variables se ilustra en la figura 4.3(b).

Una sencilla manipulación algebraica nos permite hallar las coordenadas de $\boldsymbol{X}$ en función de las de $\boldsymbol{x}$ y $\boldsymbol{l}$. En primer lugar, calculamos las coordenadas homogéneas de la recta $\boldsymbol{l}_\perp$ perpendicular a $\boldsymbol{l}$, y que pasa por $\boldsymbol{x}$. Por la condición de perpendicularidad respecto a $\boldsymbol{l} = (a, b, c)^\top$, $\boldsymbol{l}_\perp$ debe tener la forma $(-b, a, \mu)^\top$. Simultáneamente, puesto que la recta pasa por $\boldsymbol{x}$, debe cumplirse que $\boldsymbol{l}_\perp^\top \boldsymbol{x} = 0$. Esto es:

$$\boldsymbol{l}_\perp^\top \boldsymbol{x} = 0 \Leftrightarrow (-b, a, \mu) \cdot (x, y, 1)^\top \Leftrightarrow -bx + ay + \mu = 0 \Leftrightarrow \mu = bx - ay$$

Se deduce, por tanto, que $\boldsymbol{l}_\perp = (-b, a, bx - ay)^\top$. El punto $\boldsymbol{x}_\perp$ debe estar en la intersección de $\boldsymbol{l}$ y $\boldsymbol{l}_\perp$, así que podemos calcularlo utilizando el siguiente producto vectorial:

$$\boldsymbol{x}_\perp = \boldsymbol{l}_\perp \times \boldsymbol{l} = (-b, a, bx - ay)^\top \times (a, b, c)^\top = (ac - b^2x + aby, bc + abx - a^2y, -a^2 - b^2)^\top \approx$$

$$(\frac{b^2x - aby - ac}{a^2 + b^2}, \frac{-abx + a^2y - bc}{a^2 + b^2}, 1)^\top = (x_\perp, y_\perp, 1)^\top$$

Con lo que quedan determinados los valores $x_\perp$ e $y_\perp$. Por último, el valor de $z_\perp$ se obtiene sencillamente como la distancia euclídea entre $\boldsymbol{x}$ y $\boldsymbol{x}_\perp$, que resulta ser:

$$z_\perp = \sqrt{(\frac{b^2x - aby - ac}{a^2+b^2} - x)^2 + (\frac{-abx + a^2y - bc}{a^2+b^2} - y)^2} = \frac{ax + by + c}{\sqrt{a^2+b^2}}$$

En resumen, las coordenadas homogéneas del punto del espacio $\boldsymbol{X}$ serán:

$$\boldsymbol{X} = (x_\perp, y_\perp, z_\perp, 1)^\top = (\frac{b^2x - aby - ac}{a^2+b^2}, \frac{-abx + a^2y - bc}{a^2+b^2}, \frac{ax + by + c}{\sqrt{a^2+b^2}}, 1)^\top \approx$$

$$(b^2x - aby - ac, -abx + a^2y - bc, (ax + by + c)\sqrt{a^2+b^2}, a^2+b^2)^\top \tag{4.2}$$

Si observamos detenidamente la forma de $\boldsymbol{X}$, podemos separar su dependencia de las coordenadas de la línea $\boldsymbol{l} = (a, b, c)^\top$ y el punto $\boldsymbol{x} = (x, y, 1)^\top$. Un modo de hacerlo es mediante el producto matriz-vector $\boldsymbol{X} = \mathtt{C}(\boldsymbol{l}) \cdot \boldsymbol{x}$, donde la matriz $\mathtt{C}(\boldsymbol{l})$ depende sólo de las coordenadas a, b y c de la línea, y tiene la siguiente forma:

$$\mathtt{C}(\boldsymbol{l}) = \mathtt{C}(a, b, c) = \begin{pmatrix} b^2 & -ab & -ac \\ -ab & a^2 & -bc \\ a\sqrt{a^2+b^2} & b\sqrt{a^2+b^2} & c\sqrt{a^2+b^2} \\ 0 & 0 & a^2+b^2 \end{pmatrix}$$

Esta matriz de dimensión 3×4 es la que nos va a ayudar en la eliminación de la ambigüedad inherente a toda reconstrucción tridimensional a partir de una sola imagen. El modo de hacerlo es utilizar un criterio de interpretación, por el cual se establecerá que un punto 2D de la imagen pertenece a un plano vertical 3D determinado, del cual conocemos la proyección de su recta de intersección con el suelo. Puesto que $\mathtt{C}(\boldsymbol{l})$ basa su acción en la *elevación* de un punto a través de la línea $\boldsymbol{l}$, la denominaremos *matriz de elevación* sobre $\boldsymbol{l}$. Utilizando esta matriz y la matriz de proyección $\mathtt{P}$ (de la cual, recordemos, puede obtenerse la homografía suelo-imagen $\tilde{\mathtt{P}}$ simplemente eliminando la tercera columna), podemos obtener una forma cerrada para la homografía $\mathtt{H}_{\pi i}$ que transfiere puntos del sistema S_π a la imagen, dada la proyección $\boldsymbol{l}'$ de la intersección $\boldsymbol{l}$ del plano vertical con el suelo:

$$\mathtt{H}_{\pi i} = \mathtt{P} \cdot \mathtt{C}(\tilde{\mathtt{P}}^\top \boldsymbol{l}') \tag{4.3}$$

La línea dada como argumento a $\mathtt{C}(\boldsymbol{l})$ se obtiene con el siguiente razonamiento. Si la homografía $\tilde{\mathtt{P}}$ transfiere puntos del suelo a la imagen, entonces $\tilde{\mathtt{P}}^{-\top}$ transferirá líneas, y, por tanto, dada una línea en la imagen $\boldsymbol{l}'$ la correspondiente línea en el suelo tendrá la forma $\boldsymbol{l} = \tilde{\mathtt{P}}^\top \boldsymbol{l}'$.

Invirtiendo $\mathtt{H}_{\pi i}$ y premultiplicando de nuevo por la matriz de elevación obtenemos la forma definitiva de la matriz $\mathtt{R}_{vert}(\boldsymbol{l}')$, de nuevo de dimensión 4×3, y esta vez en función

de $\boldsymbol{l}'$. Esta matriz es capaz de rectificar a coordenadas 3D del sistema del robot un punto cualquiera de la imagen que se supone incluido en un plano vertical dado, al cual se conoce a su vez por la proyección $\boldsymbol{l}'$ de su recta de intersección con el suelo:

$$\mathtt{R}_{vert}(\boldsymbol{l}') = \mathtt{C}(\tilde{\mathtt{P}}^{\top}\boldsymbol{l}') \cdot \mathtt{H}_{\boldsymbol{\pi}i}^{-1} = \mathtt{C}(\tilde{\mathtt{P}}^{\top}\boldsymbol{l}') \cdot (\mathtt{P} \cdot \mathtt{C}(\tilde{\mathtt{P}}^{\top}\boldsymbol{l}'))^{-1} \tag{4.4}$$

Como puede se observar, para obtener la matriz $\mathtt{R}_{vert}(\boldsymbol{l}')$ sólo es necesario conocer la matriz de proyección $\mathtt{P}$ (lo que equivale a tener la cámara calibrada intrínseca y extrínsecamente respecto al robot), y las coordenadas homogéneas de la línea $\boldsymbol{l}'$ (imagen de la intersección del plano vertical con el suelo). Disponiendo de esta información, la matriz generada será capaz de realizar la reconstrucción tridimensional $\boldsymbol{X}$ de aquellos puntos $\boldsymbol{x}$ de la imagen que a través de la interpretación se asignen como pertenecientes a dicho plano, mediante la simple operación $\boldsymbol{X} = \mathtt{R}_{vert}(\boldsymbol{l}') \cdot \boldsymbol{x}$.

La estructura del lado derecho de la ecuación 4.4 descubre en cierto sentido la utilidad de la matriz obtenida. En realidad se trata de una especie de *inversión alternativa* de la matriz de proyección, sólo que realizada a través de una matriz de apoyo $\mathtt{C}(\tilde{\mathtt{P}}^{\top}\boldsymbol{l}')$, que se usa para introducir la información necesaria de la posición de la línea de base del plano vertical. Esta interpretación de $\mathtt{R}_{vert}(\boldsymbol{l}')$ como un tipo especial de *inversa* de la $\mathtt{P}$ revela su verdadera naturaleza: si ésta última transfiere puntos del mundo a la imagen, aquélla lo hace justamente en el sentido contrario, calculando posiciones en el espacio a partir de coordenadas estrictamente bidimensionales.

4.4.4. Rectificación planar de una imagen

La importancia de lo anteriormente expuesto reside en que, suponiendo que se disponga de un método adecuado para detectar los segmentos que delimitan las zonas de imagen que pertenecen a cada plano de la escena, ésta puede reconstruirse sin ningún tipo de ambigüedad proyectiva, afín, o ni siquiera de escala, al menos en lo que se refiere a todos aquellos elementos contenidos en el modelo de planos descrito.

Para ilustrar esto último hemos generado un ejemplo artificial, el cual se expone en la figura 4.4. Supongamos que tenemos una cámara montada en una plataforma y perfectamente calibrada intrínseca y extrínsecamente con respecto a ésta. Emplearemos una cámara virtual semejante a la utilizada en el ejemplo del capítulo anterior, donde, por simplicidad, y sin pérdida de generalidad, se ha supuesto que $d_x = d_y = \beta = 0$. El resto de parámetros son, de nuevo, $f = 400$ píxeles, $\alpha = 5\pi/12$ y $z_c = 140$ cm. Supongamos también que, al igual que en aquel ejemplo, tomamos de nuevo una imagen de un pasillo de 10×20 baldosas de 20×20 cm cada una, situado delante de la cámara en una posición desconocida. Pero esta vez vamos a colocar cuatro puertas adicionales en las tres paredes visibles del pasillo (una en la pared izquierda, otra en la frontal, y dos en la derecha). La imagen tomada se muestra en la figura 4.4(a). Admitiremos, por último, que somos capaces de determinar los segmentos que

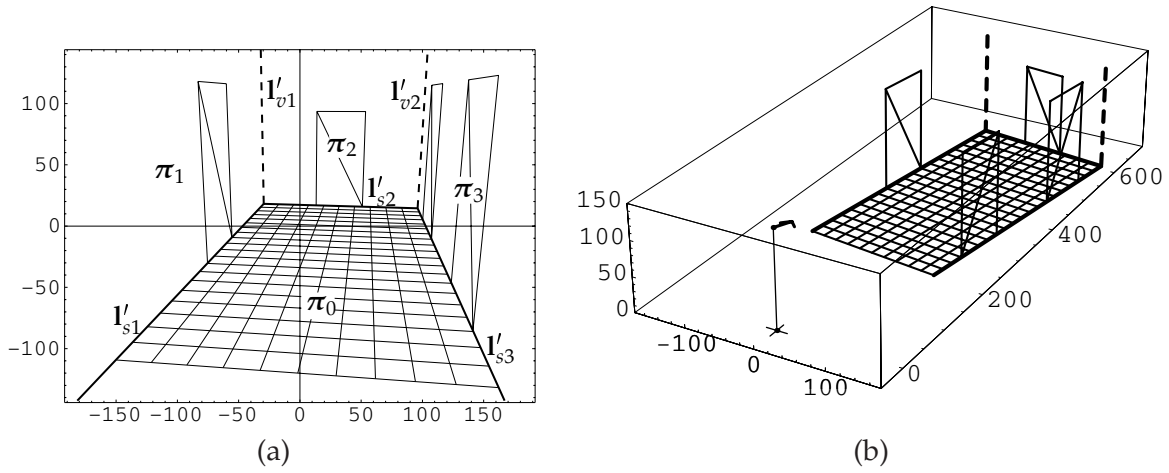


Figura 4.4: *Reconstrucción planar monocular de una escena virtual: (a) Vista de la escena, donde se han detectado los segmentos base de las paredes (en línea continua resaltada), y los segmentos verticales delimitadores de las mismas (en línea discontinua resaltada). Dichos segmentos dividen a la imagen en cuatro zonas; $\boldsymbol{\pi}_0$, para el plano del suelo, y $\boldsymbol{\pi}_1$, $\boldsymbol{\pi}_2$ y $\boldsymbol{\pi}_3$ para los planos verticales. (b) Reconstrucción euclídea de la escena a partir de la imagen, aplicando las matrices $\mathtt{R}_{vert}(\boldsymbol{l}'_{s1})$, $\mathtt{R}_{vert}(\boldsymbol{l}'_{s2})$, $\mathtt{R}_{vert}(\boldsymbol{l}'_{s3})$ y $\mathtt{R}_{sue}$ a las respectivas zonas de la imagen. Las coordenadas de la escena están en cm sobre el sistema S_{rob}, y la cámara se muestra como una flecha situada a la altura z_c e inclinación α dadas por la calibración.*

delimitan las zonas de imagen pertenecientes a cada plano mediante algún mecanismo de interpretación. Estos segmentos se muestran con trazos más gruesos en la imagen, continuos los pertenecientes a las bases de los planos verticales, y discontinuos los que corresponden a las intersecciones entre dichos planos.

En estas condiciones, el sistema es capaz de recuperar correctamente la posición 3D respecto a la plataforma de todos los elementos de la escena, simplemente conociendo el plano al que pertenece cada uno y, en caso de que éste sea vertical, la posición de la línea de intersección de dicho plano con el suelo. Para ello tenemos que utilizar, en cada caso, la matriz de reconstrucción 2D a 3D adecuada. Esto es, $\mathtt{R}_{vert}(\boldsymbol{l}'_{s1})$, $\mathtt{R}_{vert}(\boldsymbol{l}'_{s2})$ y $\mathtt{R}_{vert}(\boldsymbol{l}'_{s3})$, para los elementos situados en las respectivas zonas $\boldsymbol{\pi}_1$, $\boldsymbol{\pi}_2$ y $\boldsymbol{\pi}_3$, o $\mathtt{R}_{sue}$ para $\boldsymbol{\pi}_0$ sobre el suelo, siendo $\boldsymbol{l}'_{s1}$, $\boldsymbol{l}'_{s2}$ y $\boldsymbol{l}'_{s3}$ las coordenadas homogéneas de imagen de las líneas que marcan la base de los respectivos planos verticales. El resultado de la reconstrucción se puede observar en la figura 4.4(b). Hay que resaltar que la reconstrucción es euclídea y *robocéntrica*, es decir, con el sistema de coordenadas centrado en el robot y orientado hacia su dirección de avance. Ésta es, indudablemente, la forma más útil posible para el posterior control de la navegación.

4.4.5. Máquina de clasificación por zonas

Para aplicar el procedimiento descrito hay que dividir la imagen en zonas, con el fin de saber qué matriz de reconstrucción aplicar en cada caso. Necesitamos, pues, una máquina de clasificación que, dados los segmentos de imagen que delimitan estas zonas, divida la

imagen en áreas disjuntas, cada una perteneciente a un plano distinto de la escena. Para la construcción de esta máquina se pueden usar directamente las coordenadas homogéneas de las líneas que contienen a estos segmentos delimitadores. Utilizaremos tanto los segmentos base de los planos ($\boldsymbol{l}'_{s1}$ y $\boldsymbol{l}'_{s2}$ y $\boldsymbol{l}'_{s3}$ en el ejemplo de la figura 4.4(a)) como los verticales ($\boldsymbol{l}'_{v1}$ y $\boldsymbol{l}'_{v2}$), que separan las zonas de las distintas paredes entre sí. La correcta elección de estos segmentos dependerá del mecanismo de interpretación de la imagen, que será descrito más adelante.

Una vez realizada esta interpretación, podemos construir una función que compruebe si un punto $\boldsymbol{x}$ está incluido en el área interior a los segmentos delimitadores de un plano π proyectado sobre la imagen. Para ello, llamaremos $\overline{\mathbf{M}}_{izq}$, $\overline{\mathbf{M}}_{der}$ y $\overline{\mathbf{M}}_{sue}$ a los tres segmentos tridimensionales que, idealmente, constituyen los límites de un tramo de plano vertical del mundo real que fue detectado por algún mecanismo interpretativo. Utilizando P podemos proyectarlos a coordenadas de imagen, para obtener los segmentos 2D correspondientes, $\overline{\mathbf{m}}_{izq}$, $\overline{\mathbf{m}}_{der}$ y $\overline{\mathbf{m}}_{sue}$. A partir de los extremos de estos segmentos se pueden obtener las respectivas coordenadas homogéneas de las líneas de imagen sobre las que se encuentran. En la figura anterior, por ejemplo, al plano $\boldsymbol{\pi}_2$ le corresponderían las líneas $\boldsymbol{l}_{v1}$ y $\boldsymbol{l}_{v2}$ para los lados izquierdo y derecho, y $\boldsymbol{l}_{s2}$ como segmento de suelo base. Si llamamos $\boldsymbol{x}_i^{\overline{\mathbf{m}}}$ a las coordenadas de imagen homogéneas del extremo i del segmento proyectado $\overline{\mathbf{m}}$, con $i = 1, 2$, y suponemos que los extremos de los segmentos izquierdo y derecho vienen ordenados de arriba a abajo en la imagen, y los del suelo de izquierda a derecha, se tiene que la condición a cumplir por cualquier x para que esté en el lado correcto de cada una de las líneas delimitadoras será:

$$((\boldsymbol{x}_1^{\overline{\mathbf{m}}_{izq}} \times \boldsymbol{x}_2^{\overline{\mathbf{m}}_{izq}}) \cdot \boldsymbol{x} \geq 0) \wedge ((\boldsymbol{x}_1^{\overline{\mathbf{m}}_{der}} \times \boldsymbol{x}_2^{\overline{\mathbf{m}}_{der}}) \cdot \boldsymbol{x} \leq 0) \wedge ((\boldsymbol{x}_1^{\overline{\mathbf{m}}_{sue}} \times \boldsymbol{x}_2^{\overline{\mathbf{m}}_{sue}}) \cdot \boldsymbol{x} \geq 0) \qquad (4.5)$$

Naturalmente, si alguno de los segmentos $\overline{\mathbf{m}}$ no aparece en la imagen porque queda fuera al ser proyectado (sería el caso de los planos $\boldsymbol{\pi}_1$ y $\boldsymbol{\pi}_3$ en el ejemplo, a los que faltan los delimitadores izquierdo y derecho, respectivamente), entonces simplemente se elimina la parte de la condición correspondiente, con lo que ésta quedaría expresada de modo más simple.

Si se asume el modelo planar para todos los puntos de la escena, utilizando esta técnica de clasificación se puede rectificar, en principio, cualquier lugar de la imagen. La figura 4.5 muestra un ejemplo de reconstrucción similar al anterior, pero practicado en este caso sobre una imagen real tomada por el robot. Obsérvese que la generación de un modelo virtual realista es también muy sencilla, puesto que todos los puntos de un plano se reconstruyen con la misma matriz de rectificación. Por esta razón, el modelo texturizado puede incluso generarse en tiempo real, aunque está claro que, al menos en principio, esto no tendría ninguna utilidad añadida para la navegación. Además, el problema posterior de ir alineando las imágenes conforme el robot se va moviendo, para ir construyendo incrementalmente un modelo realista de estas características es bastante más complejo, por lo que tiene habitualmente que realizarse *off-line*. Iocchi *et al.* (2000), por ejemplo, describen un trabajo en este sentido.

La importancia de una correcta interpretación de la imagen para la reconstrucción tri-

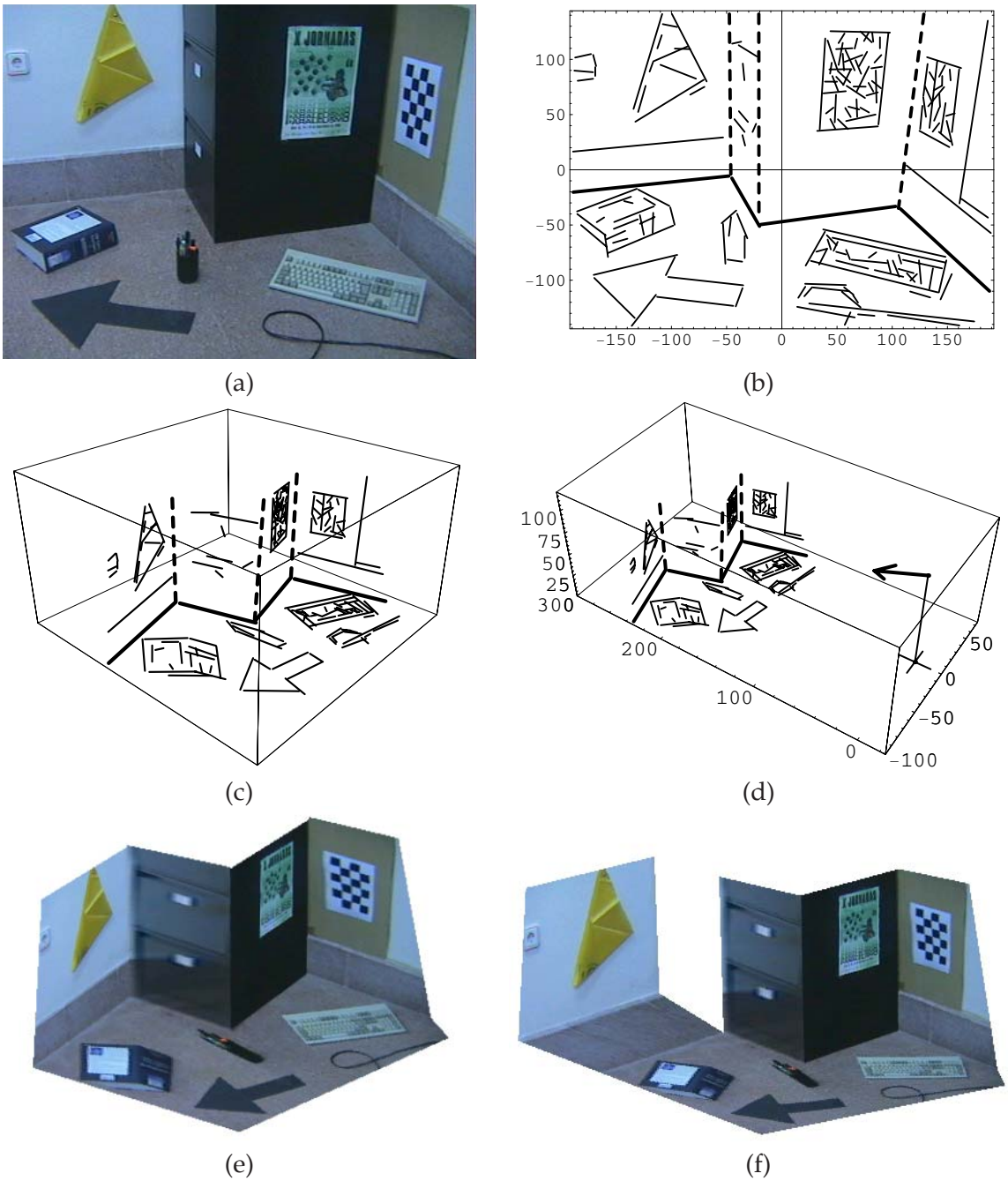


Figura 4.5: *Reconstrucción planar monocular de una escena real: (a) Imagen inicial. (b) Segmentos extraídos, en coordenadas de píxeles. (c) Reconstrucción tridimensional practicada en tiempo real, vista desde un ángulo distinto. (d) Vista de la reconstrucción euclídea, donde se puede observar que los elementos están situados en medidas reales (en cm) respecto a la posición de la cámara, la cual aparece como una flecha gruesa en la figura. El sistema de coordenadas empleado es el del robot, S_{rob}. (e) Escena tridimensional reconstruida, con la imagen original texturizada sobre los respectivos planos. (f) Escena tridimensional reconstruida a partir de la misma imagen, en el caso de que el segmento correspondiente al rodapié del suelo fuese erróneamente interpretado como la base de la pared vertical.*

dimensional queda patente en la figura 4.5(f). En ella se muestra el modelo erróneo que se hubiese obtenido si la línea del rodapié hubiese sido clasificada como línea de intersección vertical con el suelo. Afortunadamente, como veremos, este tipo de interpretaciones tentativas equivocadas serán eliminadas mediante el mecanismo de refutación que se pone en marcha durante la navegación del robot. La razón es que en cuanto éste empiece a moverse la reproyección del modelo incorrecto utilizando la información odométrica no encontrará soporte en los segmentos extraídos de la imagen. En apartados posteriores describiremos con mayor detalle este mecanismo de validación por consistencia temporal, que resultará clave para lograr una mínima robustez en el sistema.

4.4.6. Procedimiento de reconstrucción

Terminamos esta sección con un resumen del procedimiento de rectificación propuesto (ver algoritmo 4.1). Éste toma como entrada las coordenadas de imagen del punto a rectificar y un conjunto de planos verticales de la escena, cuya información de posición con respecto al robot es conocida. Llamaremos $Model_k$ al modelo individual de cada uno de estos tramos de plano, y $\overline{\mathbf{M}}_{k,j}$ a los respectivos segmentos tridimensionales que los delimitan. Se supone que los correspondientes segmentos del suelo y de los lados izquierdo y derecho de cada plano se hallan convenientemente destacados en el modelo. Es necesario también conocer la matriz de proyección `P` y la altura del horizonte asociada.

El procedimiento utiliza el criterio de clasificación dado en la expresión 4.5 para localizar el plano de soporte del punto. Si éste pertenece a algún plano vertical, entonces se usa el segmento proyectado base de éste para realizar la rectificación adecuada. En caso contrario, si el punto está por debajo del horizonte se rectifica con $\mathtt{R}_{sue}$. Si tampoco se da esta condición, finalmente, no se conoce su plano de soporte, por lo que no puede ser rectificado. Alternativamente, por ejemplo, podría considerarse situado en el plano del infinito, con lo que su posición quedaría determinada directamente por el punto ideal asociado al rayo de cámara que pasa por $\boldsymbol{x}$, convenientemente corregido con la calibración. En cualquier caso, hay que decir que ello tampoco resultaría demasiado útil para la navegación.

4.5. Arquitectura de percepción y acción

4.5.1. Introducción

La determinación de la profundidad de cada punto de la escena partiendo de una única imagen no es gratuita. Exige un paso previo de interpretación, que en el caso del procedimiento anterior consiste básicamente en la utilización de un modelo plausible de escena de interior, con el posterior ajuste de la imagen al mismo. Pero aún no ha quedado claro cómo obtener y mantener esta clase de interpretaciones. En esta sección describiremos la arquitectura de acción y percepción propuesta para atacar estos problemas.

ENTRADA:
- Coordenadas de imagen homogéneas $\boldsymbol{x} = (x, y, 1)^\top$ del punto a rectificar, en píxeles.
- Conjunto de M planos verticales $\{Model_k = \{\overline{\mathbf{M}}_{k,j} \mid j = 1, \ldots, N_k\} \mid k = 1, \ldots, M\}$ presentes en la interpretación actual. Cada $Model_k$ consta de N_k segmentos 3D $\overline{\mathbf{M}}_{k,j}$, para los que se guardan sus extremos en coordenadas de S_{rob} en mm, y de los que se destacan tres, correspondientes a la intersección con el suelo y los límites izquierdo y derecho del plano. El primero es obligatorio, mientras que los otros dos son opcionales, si el tramo de plano vertical no aparece completo en la imagen, por cualquiera de los lados. En principio puede haber otros segmentos adicionales, que no se usarán por ahora, pero que se incluyen por consistencia con posteriores algoritmos.
- Matriz de proyección P, obtenida mediante cualquiera de los procedimientos 3.2-3.8.
- Altura del horizonte, y_h.

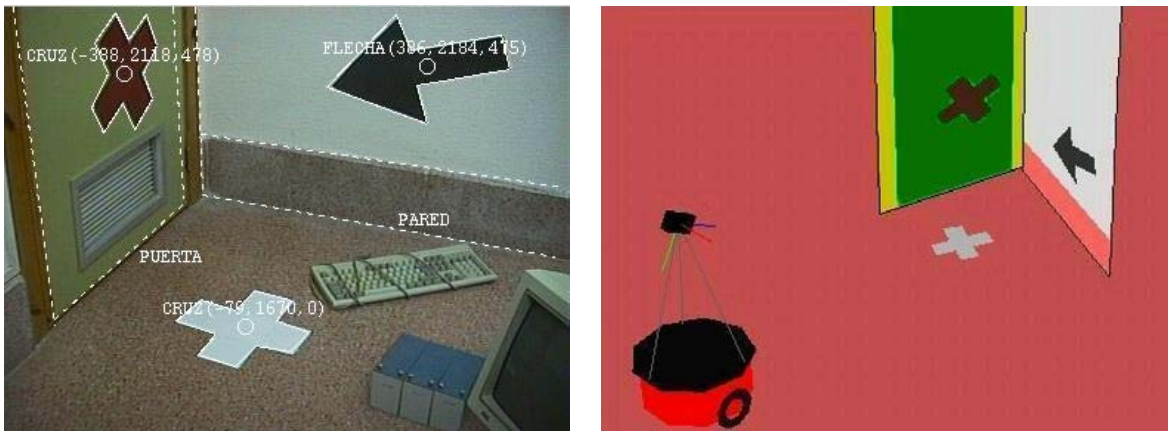
SALIDA:
- Coordenadas espaciales homogéneas $\boldsymbol{X}$ en S_{rob} del punto rectificado, en mm, con indicación del plano al que pertenece.

ALGORITMO:

for $k = 1$ ***to*** M ***do*** /* *Bucle de recorrido de planos verticales:* */
  for $j = 1$ ***to*** N_k ***do*** /* *Bucle de recorrido de segmentos del plano:* */
    if el segmento $\overline{\mathbf{M}}_{k,j}$ aparece destacado como límite izquierdo, derecho o de base ***then***
      - Proyectar el segmento 3D $\overline{\mathbf{M}}_{k,j}$ al correspondiente 2D $\overline{\mathbf{m}}_{k,j}$, usando la matriz de proyección P, y recortándolo al tamaño de la imagen.
    endif
  endfor
  - Determinar si $\boldsymbol{x}$ pertenece al plano vertical k usando la condición 4.5, siendo $\overline{\mathbf{m}}_{izq}, \overline{\mathbf{m}}_{der}, \overline{\mathbf{m}}_{sue}$ los correspondientes segmentos del modelo $Model_k$ proyectados anteriormente.
  if la condición anterior es cierta ***then***
    - Rectificar $\boldsymbol{x}$ para obtener $\boldsymbol{X}$ usando la matriz de reconstrucción $\mathtt{R}_{vert}$ dada en 4.4, utilizando como parámetro la línea que contiene al segmento $\overline{\mathbf{m}}_{k,sue}$, es decir, $\boldsymbol{l}' = x_1^{\overline{\mathbf{m}}_{k,sue}} \times x_2^{\overline{\mathbf{m}}_{k,sue}}$.
    - El punto pertenece al plano vertical k.
    - Terminar el procedimiento.
  endif
endfor
/* *Si no pertenece a ningún plano vertical, aún puede pertenecer al suelo, si su coordenada y está por debajo del horizonte. En caso contrario, no puede recuperarse el punto 3D con el modelo planar:* */
if $y < y_h$ ***then***
  - Rectificar $\boldsymbol{x}$ para obtener $\boldsymbol{X}$ usando la matriz de reconstrucción $\mathtt{R}_{sue}$ dada en 4.1.
  - El punto pertenece al plano del suelo.
endif else
  - El punto $\boldsymbol{x}$ no está en ninguno de los planos modelados, por lo que no puede recuperarse $\boldsymbol{X}$.
endelse

Algoritmo 4.1: *Procedimiento de reconstrucción tridimensional de un punto cualquiera de la imagen a partir de los planos verticales detectados y la información de calibración.*

Un ejemplo de lo que se quiere hacer se muestra en la figura 4.6. Como se observa en la imagen izquierda, una cierta parte de la escena se adapta perfectamente al modelo comentado en la sección anterior. En principio, querríamos detectar los tres planos principales, correspondientes al suelo, la puerta de la izquierda y la pared frontal. Para su detección podríamos apoyarnos en una serie de heurísticas tales como la comprobación del color de sus segmen-



(a) (b)

Figura 4.6: *Ejemplo de interpretación planar de escenas: (a) Imagen original con la interpretación sobreimpresionada. Las señales aparecen etiquetadas con sus posiciones relativas al sistema S_{rob}, en mm. (b) Representación tridimensional interna asociada a la situación, con los elementos reconocidos debidamente posicionados y categorizados.*

tos de borde y su situación geométrica sobre la imagen. Ésta última, por ejemplo, debe ser compatible con un criterio de perpendicularidad entre ellos una vez rectificados a 3D. Del mismo modo, si se dispone *a priori* de cierta información como el color aproximado del suelo, las puertas, una parte del mobiliario, etc., puede incluso aumentarse la potencia del mecanismo interpretativo, haciendo más robusta la generación de hipótesis de planos verticales en la escena y su posible clasificación como uno u otro tipo de elemento.

Una vez determinada la posición de estos planos, podrían asimismo detectarse algunas estructuras completamente contenidas en las zonas de imagen correspondientes. Para estos objetos es viable recuperar la secuencia de segmentos que forman sus contornos, utilizando como ayuda la información de color tal y como se describió en la sección 2.5.2. Algunos ejemplos son las señales de colores, la rejilla de la puerta, los objetos del suelo, etc. Ciertamente, lo único que cabe hacer con ellos a partir de una sola imagen es, en principio, rectificarlos con la matriz correspondiente, empleando el algoritmo 4.1. Está claro que algunas reconstrucciones serán erróneas, puesto que ciertas regiones del espacio pueden no ajustarse al modelo planar. Es el caso del monitor de la figura 4.6, o del libro y el bote con lápices de la anterior 4.5. Otros objetos como la rejilla, el teclado, o las señales, sin embargo, sí que pueden considerarse, con mayor o menor error, como incluidos en los planos correspondientes, por lo que su reconstrucción será aproximadamente correcta.

Es sencillo clasificar estos contornos dado que, tras su rectificación euclídea, cualquier procedimiento ordinario de reconocimiento de formas basado en invariantes métricos podría determinar si se trata de alguna figura de una base de datos de señales preestablecidas. Éste es el caso de la flecha situada en la pared de enfrente, o las cruces del suelo y de la puerta,

respectivamente. Obsérvese cómo éstas dos últimas son correctamente categorizadas gracias a su oportuna ubicación en la escena planar, a pesar de sus diferentes aspectos en la imagen. El resto de contornos, correspondientes a otros objetos o marcas, simplemente son no reconocidos y, o bien no se añaden al modelo interpretado, o bien simplemente se etiquetan como obstáculos y se localizan usando la homografía del suelo, a fin de poder evitarlos. La figura 4.6(b) muestra la representación tridimensional reducida que maneja el robot tras la interpretación. En esta representación de ejemplo se incluyeron sólo aquellos elementos de la escena que fueron correctamente identificados, es decir, el tramo de la pared, la puerta y las señales. No se muestran el resto de obstáculos de los que, aunque sin categorizar, podría también conocerse una posición aproximada rectificándolos como si estuvieran en el suelo.

La estructura del espacio de navegación se definirá entonces en términos de *situaciones tipo* creadas en función de los planos que se encuentran en el entorno inmediato del robot. A las distintas situaciones categorizadas las denominaremos *schemaps* (de *schematic maps*, o mapas esquemáticos locales elementales). Se trata de algo comparable a lo que en el diseño de Saphira (Konolige *et al.*, 1997), la popular arquitectura de percepción y control para robots propuesta por los diseñadores del robot Pioneer, se llaman "*artefactos*" (puertas, pasillos, esquinas, paredes, etc.). Pero, mientras que en aquel sistema dichos artefactos se detectaban a partir de nubes de ecos devueltos por los sensores de ultrasonido, nosotros los localizaremos a partir de elementos a un nivel de interpretación más alto, los planos verticales del entorno detectados visualmente tal y como se comentó anteriormente.

A continuación describiremos la arquitectura de procesamiento propuesta para la realización de este tipo de interpretaciones. En primer lugar daremos una visión global de la arquitectura, organizada en torno al ciclo de interpretación, predicción y corroboración de hipótesis y el modelo planar del espacio descritos en las secciones 4.3 y 4.4, respectivamente. En los apartados siguientes, que cierran el capítulo, describiremos en detalle cada uno de los componentes y procedimientos que, debidamente ensamblados, ponen en marcha el mecanismo de percepción y actuación del robot.

4.5.2. Visión global de la arquitectura

Estructura general

La figura 4.7 resume la arquitectura de acción y percepción que proponemos para la navegación visual de nuestro agente autónomo. Dado su carácter central en este capítulo, conviene comentarla detenidamente. Como puede observarse, el esquema se compone principalmente de una serie de cajas que representan las distintas estructuras de información manejadas, tales como los segmentos coloreados, la calibración, las estructuras tridimensionales, etc. Éstas se ubican en la figura en un determinado nivel que indica su lugar en la jerarquía interpretativa. Siguiendo una organización perceptual más o menos clásica, similar a la descrita por Sarkar y Boyer (1993), los niveles incluidos se denominan *sensorial*, de *primitiva*, de *agrega-*

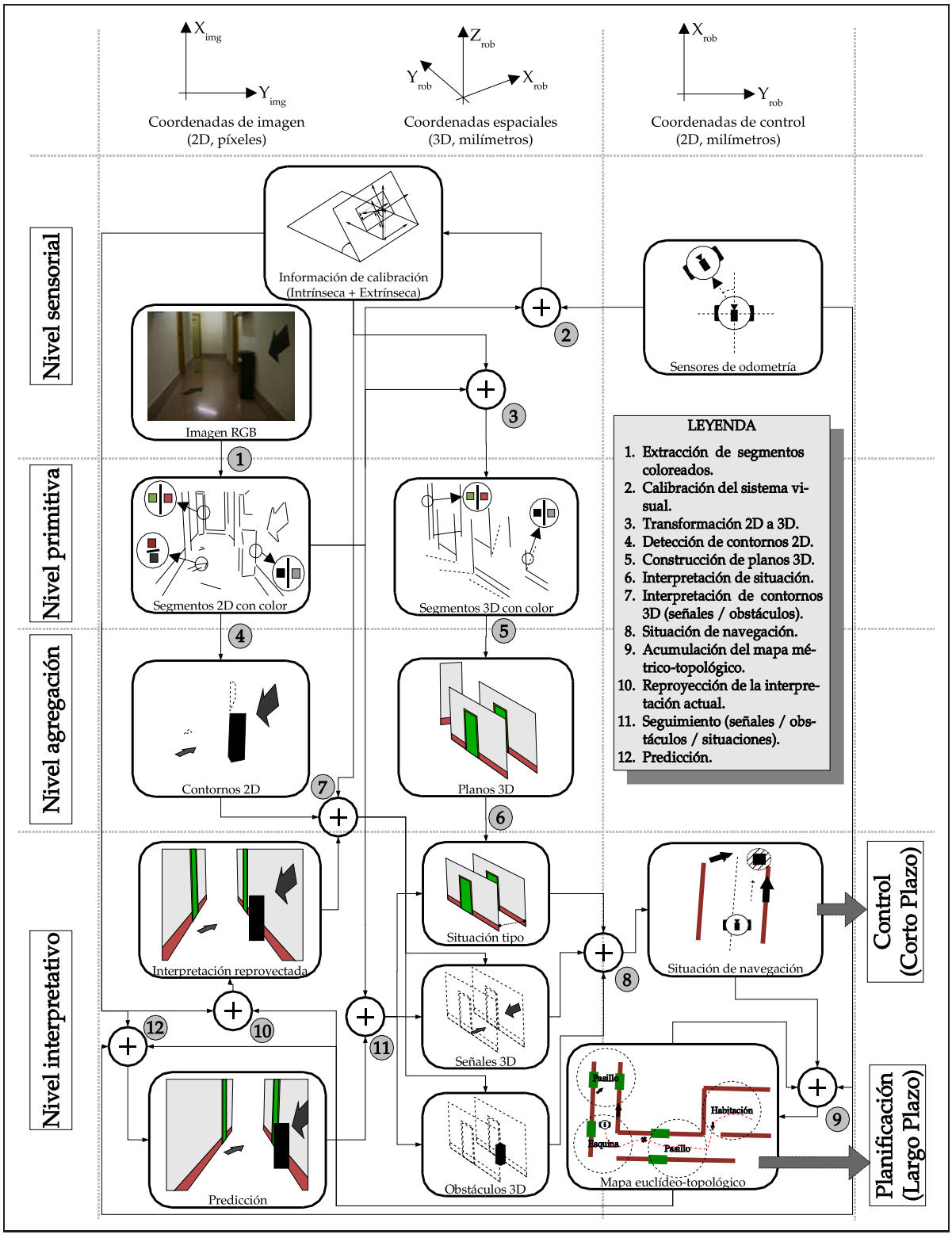


Figura 4.7: *Esquema del sistema de percepción de alto nivel propuesto (ver texto).*

ción e *interpretativo*, en orden creciente de complejidad. Así, la imagen cruda o la información odométrica, por ejemplo, se encuentran en el nivel sensorial, mientras que las señales o el mapa del entorno se hallan en el nivel superior, o interpretativo. Del mismo modo, los datos manejados pueden venir expresados en términos de distintos sistemas de coordenadas. Esto se indica en la parte superior, donde se divide el esquema en tres zonas que, de izquierda a derecha, hacen referencia a los respectivos espacios de *imagen*, *3D del mundo* y *2D de control del robot*. Siguiendo este convenio, la información de la cámara, por ejemplo, se encuentra entre el espacio de imagen y el del mundo, en concordancia con los respectivos datos intrínsecos y extrínsecos de la calibración.

Los círculos grises numerados denotan a los procedimientos con los que se tratan estos bloques de información. A menudo funcionan sobre un solo tipo de dato de entrada para producir otro de salida. Es el caso, por ejemplo, del proceso de extracción de segmentos coloreados a partir de la imagen RGB, marcado con el número 1. Otras veces la información a procesar viene de varias fuentes, como en el proceso 2, que realiza la calibración a partir de los segmentos y de la información de odometría. Estos casos se indican mediante círculos con un signo + en el centro.

Componentes

Una vez explicada la estructura general del esquema, procedemos a describir uno a uno todos sus componentes. Siguiendo la estructuración por niveles comentada anteriormente, éstos son los tipos de información con los que trabajará la arquitectura:

Nivel sensorial: Se incluyen en este nivel tanto la imagen RGB cruda como la odometría proporcionada por los sensores propioceptivos. Arbitrariamente, incluimos también aquí la información de calibración actual, puesto que está relacionada con las características del sensor principal del sistema, la cámara.

Nivel de primitiva: En este segundo nivel se incluyen las estructuras perceptuales de nivel intermedio utilizadas, los segmentos con información de color. En principio, éstos se extraen directamente de la imagen bidimensional, pero a continuación pueden ser también rectificados a 3D utilizando el procedimiento 4.1.

Nivel de agregación: Tras una adecuada categorización de los segmentos como intersecciones entre planos de suelo y pared, pared y puerta, rodapié, etc., agrupando éstos será posible elaborar hipótesis tentativas de planos 3D. De igual forma, la agregación de segmentos por criterios de continuidad geométrica y compatibilidad cromática podría dar lugar a contornos cerrados bidimensionales, que en niveles superiores podrán ser clasificados como distintos tipos de objetos.

Nivel interpretativo: A partir de las estructuras anteriores pueden crearse modelos ya utilizables en la planificación de la navegación. Así, podremos tener contornos 3D interpre-

tados como señales u obstáculos según el contexto en el que se encuentran. Pero lo más importante en este nivel es la creación de estructuras tridimensionales más complejas, correspondientes a situaciones tipo esquemáticas dentro de los entornos parcialmente estructurados (pasillos, esquinas, habitaciones, etc.). Ésta es la categorización del espacio perceptual local que servirá para tomar las decisiones inmediatas de movimiento.

En paralelo a esta interpretación local se puede ir construyendo un mapa incremental de representación del espacio perceptual global. Éste no es denso, al estilo de los *grid* de ocupación, sino más compacto y explicativo, conteniendo información resumida de la localización (euclídea) y del tipo de situaciones y señales percibidas (topológica). El mapa es corregido constantemente utilizando la odometría para aquella parte de la representación que no está ya dentro del alcance efectivo de la cámara, y usando lo percibido actualmente para lo que sí lo está. Su construcción se realiza de modo simultáneo a la autolocalización dentro del mismo, en un esquema SLAM como el comentado en la sección 4.2. Puesto que básicamente se construye en el espacio bidimensional de control del robot, pero también contiene cierta información tridimensional, lo incluimos a caballo entre las dos zonas correspondientes en el esquema global de la figura 4.7. Finalmente, al menos en teoría, cabría también la posibilidad de utilizar esta representación como base de una política de planificación a más largo plazo, si bien esta última posibilidad no ha sido contemplada aún en la versión actual de la implementación.

Como último punto en este nivel, toda la información interpretada es reproyectada sobre la imagen, con el fin de realizar el seguimiento de las estructuras para su corroboración y predecir su situación en el *frame* siguiente. Esta última parte, pues, cierra el ciclo de percepción en el que se basa la arquitectura.

Todas las estructuras están caracterizadas por un conjunto de parámetros. Por ejemplo, cada segmento bidimensional $\overline{\mathbf{m}}$ tiene dos extremos $\boldsymbol{x}_1 = (x_1, y_1, 1)$ y $\boldsymbol{x}_2 = (x_2, y_2, 1)$, y dos vectores de color $\boldsymbol{rgb}_{der} = (r_{der}, g_{der}, b_{der})$ y $\boldsymbol{rgb}_{izq} = (r_{izq}, g_{izq}, b_{izq})$ a derecha y a izquierda. Algunos de estos parámetros son heredados directamente en la construcción de agregaciones de orden superior, mientras que otros son generados a partir de los originales utilizando algún tipo de procesamiento. Por ejemplo, al crearse un segmento 3D a partir de uno 2D los colores son directamente heredados, pero los extremos $\boldsymbol{X}_1 = (X_1, Y_1, Z_1, 1)$ y $\boldsymbol{X}_2 = (X_2, Y_2, Z_2, 1)$ son generados mediante el procedimiento de perspectiva inversa descrito en el algoritmo 4.1, utilizando la información de autocalibración y los extremos del segmento 2D a partir del cual se generó. Algunas estructuras pueden tener una configuración adicional, en el sentido de que es incluso variable su número de parámetros. Por ejemplo, una pared podría tener un número indeterminado de puertas, cada una de las cuales tendrá una anchura y posición determinada dentro de la pared.

El ciclo interpretativo-predictivo-corroborativo

El esquema de procesamiento que gobierna el funcionamiento de la arquitectura se basa en el ciclo de interpretación, predicción y corroboración de estructuras que se introdujo en la sección 4.3. En el libro de Hofstadter citado en la introducción, ciertamente inspirador de gran parte de la filosofía de trabajo subyacente en esta tesis, el autor proponía un *pool* de procesos de escasa carga computacional individual, que se ejecutaban bajo un esquema paralelo, distribuido y probabilístico. Pero los dominios de aplicación propuestos eran de carácter más simbólico, sin el volumen de información ni las restricciones de tiempo real que exige la percepción visual. En su lugar, nosotros optaremos por un modelo de cómputo más determinista, voraz, en el que en primera instancia se ejecutan los procesos guiados por los datos, más costosos por su carácter de búsqueda (aunque dirigidos por heurísticas eficientes para evitar la explosión combinatoria), para lanzar posteriormente las presiones *top-down* conducidas por los modelos que, de modo más sencillo, buscan la corroboración y el seguimiento de hipótesis ya elaboradas. Éstas últimas tienen también el efecto favorable de simplificar los subsecuentes procesos *bottom-up*, ya que pueden descartar gran parte de la información a tratar en la creación de nuevas estructuras.

En los procesos del primer tipo, encargados de la generación de hipótesis, el sistema tratará de relacionar primitivas entre sí, siempre que cumplan una serie de condiciones sobre sus parámetros. Los segmentos que forman la silueta de una señal o un obstáculo son extraídos en primer lugar, de modo que no afecten a la interpretación del resto del entorno. Se podría hablar de una *saliencia interpretativa* de este tipo de siluetas. El resto de segmentos del entorno seguirán posteriormente un proceso de agrupamiento similar, aprovechando también la información geométrica y de color, así como la de calibración. Por ejemplo, una vez conocida la homografía suelo-imagen, podemos lanzar una hipótesis de interpretación de un segmento como si correspondiese a una intersección entre la pared y el suelo. En ese momento, el sistema intentaría buscar pistas adicionales que confirmasen esa interpretación, tales como segmentos verticales incidentes en los extremos de la pared, un segmento de rodapié paralelo, etc. Esta operación se correspondería con la acción combinada de los *procesos de búsqueda* y los *procesos constructivos* del modelo computacional de Hofstadter. La construcción de situaciones tipo a partir de la posición relativa de los planos cercanos al robot podría también enmarcarse dentro de este grupo. Naturalmente, también podrían buscarse en ese momento segmentos que refutasen la hipótesis, por ejemplo, por incidir inadecuadamente en el segmento inicial. Estos otros intentos, por su parte, serían análogos a los *procesos destructivos* de aquel modelo.

En general, se puede aprovechar toda la información útil de cada segmento, plano o estructura en esta etapa de construcción de agrupaciones tentativas. El color de los lados de un segmento, por ejemplo, puede descartar directamente su interpretación como rodapié, suelo o segmento vertical que limita un tramo de pared, entre otras. Su ubicación geométrica relati-

va a otros segmentos, por otro lado, puede llevar a una reconstrucción 3D coherente o no con el modelo planar, validando así la estructura correspondiente. Por ejemplo, un segmento de intersección entre el suelo y la pared debe tener asociados segmentos verticales en el mundo real que limiten el tramo de pared asociado. La ubicación en la imagen de cada segmento, junto con la calibración completa llevada a cabo previamente, nos permiten comprobar si el segmento cumple o no la restricción interpretativa correspondiente.

Los procesos del segundo tipo, guiados por los modelos, tienen la labor de predecir, corregir y corroborar las hipótesis, complementando a los anteriores. Las interpretaciones provenientes de estados anteriores son actualizadas con la odometría, reproyectadas sobre la imagen y contrastadas con el *frame* actual para comprobar su "credibilidad". No se trata, por tanto, de un clásico seguimiento interimagen (*tracking*) de segmentos, sino de un continuo ajuste del modelo a los datos presentes en la entrada. Hay también aquí, por tanto, un paralelismo claro con los procesos que ejercen las *presiones descendentes* del modelo de Hofstadter.

Es igualmente clave la correcta conmutación de hipótesis en el momento en el que el seguimiento de las actuales deja de ser consistente. Ciertas partes de la interpretación van entonces perdiendo fuerza por dejar de ser corroboradas, hasta que finalmente quedan descartadas. Volverían a adquirir importancia entonces los procesos *bottom-up*, encargados de elaborar nuevas construcciones tentativas, para poder adaptarse a los cambios a más largo plazo del entorno inducidos por el movimiento. A veces, aunque la hipótesis actual no deje de ser corroborada, ésta puede ser ampliada hasta transformarse en otra más complicada. Una situación de pasillo, por ejemplo, podría pasar a ser considerada una esquina en la que habrá que girar al encontrarnos con una pared enfrente. Esta clase de actualizaciones se pueden implementar tras la etapa de corroboración. Con los segmentos restantes, aún no asignados a ningún elemento de la interpretación, se trata de aumentar la hipótesis actual. Naturalmente, la carga de procesamiento añadida es también menor en estos casos que la generación de una interpretación completa para todos los segmentos de una imagen, partiendo de cero.

En los dos siguientes apartados describiremos los procesos de generación y posterior corroboración de hipótesis por separado, tal y como se instancian en nuestra implementación.

4.5.3. Generación de hipótesis

Detección de planos verticales

En la generación de una hipótesis de plano vertical a partir de una única imagen, para ser candidato a formar parte de la misma cada segmento extraído debe de cumplir una serie de condiciones. Naturalmente, estas condiciones serán distintas dependiendo de si ese segmento va a ser interpretado como base, delimitador izquierdo o delimitador derecho del plano, por ejemplo. Una situación general típica se muestra en la figura 4.8. En ella se puede observar cómo el tramo de pared del centro es completamente visible, ya que en él se pueden caracterizar el segmento de intersección con el suelo, los segmentos laterales izquierdo y derecho y,

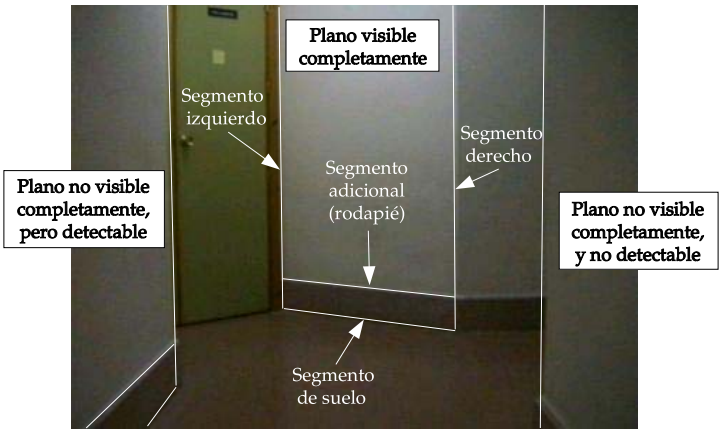


Figura 4.8: *Ejemplos de planos verticales detectables y no detectables por el mecanismo de generación de hipótesis.*

en este caso, un segmento adicional de rodapié. La correcta localización del primero de estos segmentos es clave para poder obtener la posición 3D del plano vertical correspondiente. Así, un posible método de detección intentaría probar con cada uno de los segmentos de entrada rectificándolo con $\mathtt{R}_{sue}$ como si estuviese en el suelo, para comprobar después si existen segmentos adicionales que, rectificados con la matriz $\mathtt{R}_{vert}$ obtenida a partir de dicho segmento, pudiesen ejercer el papel de correspondientes bordes verticales.

Pero no todos los delimitadores del plano tienen por qué ser completamente visibles. Para el plano de la izquierda de la misma figura, por ejemplo, su límite izquierdo no aparece en la imagen. No obstante, es aún detectable debido a que aparece un tramo de su segmento de suelo y éste llega hasta uno de los bordes de la imagen, lo que explica la ausencia de dicho delimitador. No habría problema, pues, en disponer de la línea $\boldsymbol{l}'$ con la que construir la matriz $\mathtt{R}_{vert}(\boldsymbol{l}')$ usando la ecuación 4.4. El plano de la derecha, por el contrario, no sería detectable con el método propuesto a pesar de ser parcialmente visible, puesto que su segmento base queda fuera de la imagen en el ejemplo.

El esqueleto de un procedimiento basado en las consideraciones anteriores se muestra en el algoritmo 4.2. Acepta como entrada el conjunto de segmentos bidimensionales con información de color extraídos de la imagen original, a los que denominaremos simplemente $\overline{\mathbf{d}}_i$ para abreviar. Se consideran también conocidos el color aproximado del suelo y la información de calibración. Finalmente, se proporcionan tres parámetros adicionales utilizados como umbrales en la comprobación de distintas condiciones a cumplir por los segmentos, y cuyo significado se aclara en la propia especificación del procedimiento. Por consistencia con el algoritmo 4.1, volvemos a utilizar la misma notación para los modelos de planos 3D obtenidos como salida.

El procedimiento funciona recorriendo el conjunto de segmentos de entrada a la búsqueda de algún $\overline{\mathbf{d}}_i$ cuya información cromática sea compatible con el color conocido del suelo,

ENTRADA:
- Conjunto de segmentos $Data = \{\overline{\mathbf{d}}_i \mid i = 1, \ldots, D\}$ extraídos de la imagen usando el algoritmo 2.2. Cada $\overline{\mathbf{d}}_i$ incluye los extremos en coordenadas de píxel y la información de color.
- Matriz de proyección P, obtenida mediante cualquiera de los algoritmos 3.2-3.8.
- Distancia máxima entre segmentos 2D para ser considerados colindantes, τ_{geom}.
- Distancia máxima entre colores en el espacio RGB para ser considerados compatibles, τ_{crom}.
- Ángulo máximo entre dos segmentos 3D para ser considerados paralelos, τ_{ang}.
- Color aproximado del suelo, $\mathbf{rgb}_{sue} = (r_{sue}, g_{sue}, b_{sue})$.

SALIDA:
- Conjunto de M modelos de plano vertical $\{Model_k = \{\overline{\mathbf{M}}_{k,j} \mid j = 1, \ldots, N_k\} \mid k = 1, \ldots, M\}$ añadidos a la interpretación actual. Cada $Model_k$ consta de N_k segmentos 3D $\overline{\mathbf{M}}_{k,j}$, para los que se guardan sus extremos en coordenadas de S_{rob} en mm y la información de color, y de los se destacan tres, correspondientes a la intersección con el suelo y los límites izquierdo y derecho del plano. El primero es obligatorio, mientras que los otros dos son opcionales, si el tramo de plano vertical no aparece completo en la imagen, por cualquiera de los dos lados. El resto de segmentos pueden ser detalles adicionales, como un posible rodapié, p.e. (ver figura 4.8).

ALGORITMO:
- Inicializar el contador de planos $k = 0$.

for $i = 1$ ***to*** D ***do*** /* *Búsqueda de candidatos de suelo:* */
- Rectificar $\overline{\mathbf{d}}_i$ para obtener el segmento 3D $\overline{\mathbf{D}}_i$, usando la matriz $\mathtt{R}_{sue}$ dada en 4.1 a partir de P.

if (la distancia entre el color inferior de $\overline{\mathbf{d}}_i$ y $\mathbf{rgb}_{sue}$ es inferior a τ_{crom}) ***and***
($\overline{\mathbf{D}}_i$ mide al menos 0.5 m y está en un área delante del robot a menos de 6 m) ***then***
- Guardar $\overline{\mathbf{D}}_i$ como candidato a *segmento de suelo* de un plano vertical.

for $i' = 1$ ***to*** D ***do*** /* *Búsqueda de candidatos correspondientes verticales y adicionales:* */
- Rectificar $\overline{\mathbf{d}}_{i'}$ para obtener el segmento 3D $\overline{\mathbf{D}}_{i'}$, usando la matriz $\mathtt{R}_{vert}(\overline{\mathbf{d}}_i)$, como en 4.1.

if (la distancia entre el extremo izquierdo de $\overline{\mathbf{d}}_i$ y el segmento $\overline{\mathbf{d}}_{i'}$ es menor que τ_{geom}) ***and***
(el ángulo formado por $\overline{\mathbf{D}}_{i'}$ con la vertical al suelo es menor que τ_{ang}) ***then***
- Guardar $\overline{\mathbf{D}}_{i'}$ como candidato a *segmento izquierdo* del plano vertical.

endif else if (... ídem para el segmento derecho ...) ***then***
- Guardar $\overline{\mathbf{D}}_{i'}$ como candidato a *segmento derecho* del plano vertical.

endif else if (... comprobaciones similares para posible rodapié, etc. ...) ***then***
- Guardar $\overline{\mathbf{D}}_{i'}$ como candidato a *segmento adicional* del plano vertical.

endif

endfor

endif

if (hay candidato de suelo) ***and***
(hay candidato izquierdo ***or*** el de suelo llega a un borde de la imagen por la izquierda) ***and***
(hay candidato derecho ***or*** el de suelo llega a un borde de la imagen por la derecha) ***then***
- Incrementar el contador de planos, $k = k + 1$.
- Añadir el nuevo modelo $Model_k$ a la interpretación, formado por los N_k segmentos 3D $\overline{\mathbf{M}}_{k,j}$ candidatos confirmados, previamente regularizados para cumplir posibles condiciones ideales de coincidencia entre extremos, verticalidad exacta, etc., guardando la información de color, y cada uno etiquetado con su condición de suelo, lado izquierdo o derecho, o similar.

endif

endfor
- Hacer $M = k$, y devolver el conjunto de planos $\{Model_k\}$ encontrados, con $k = 1, \ldots, M$.

Algoritmo 4.2: *Generación de hipótesis de planos verticales a partir de un conjunto de segmentos de imagen con información de color, conociendo la matriz de calibración* P *y el color aproximado del suelo.*

y cuya rectificación a 3D $\overline{\mathbf{D}}_i$ utilizando la homografía del suelo quede en un lugar razonable delante del robot. Para cada segmento que cumpla estas restricciones preliminares se buscan segmentos $\overline{\mathbf{d}}_{i'}$ adicionales que incidan en los extremos de $\overline{\mathbf{d}}_i$ y que, debidamente rectificados con la correspondiente $\mathtt{R}_{vert}(\overline{\mathbf{d}}_i)$, queden en posición aproximadamente vertical respecto a $\overline{\mathbf{D}}_i$. Si se encuentran los tres segmentos necesarios (base, izquierdo y derecho) para completar el plano, éste se añade a la interpretación actual. Alternativamente, se permite también que falte alguno de los delimitadores laterales, si el correspondiente extremo del segmento base toca algún borde de la imagen por el lado correspondiente. Por último, a cada plano localizado se le aplica un tratamiento de regularización final en el que se aseguran ciertas condiciones ideales sobre los segmentos 3D extraídos, como la verticalidad de los delimitadores o la coincidencia entre extremos en el modelo, por ejemplo.

Dependiendo de la aplicación concreta se pueden incorporar otras heurísticas de sentido común, como la comprobación de los colores de los segmentos verticales, o la inclusión de segmentos adicionales que tengan en cuenta posibles detalles contenidos dentro de los planos (un posible rodapié, detalles de las puertas o el mobiliario, etc.). Pueden asimismo hacerse otras comprobaciones de consistencia complementarias, como verificar que los planos ocupen zonas disjuntas de imagen, o impedir que haya inclusiones parciales de unos respecto a otros. Aunque, por brevedad, no se muestren en el algoritmo 4.2, algunas de estas ideas se implementaron también en el prototipo construido, con el fin de adaptarlo adecuadamente a su entorno de operación.

Hay que decir por último que en éste, como en general en todos los mecanismos de generación de las distintas hipótesis que iremos describiendo, es relativamente frecuente que se produzcan fracasos interpretativos debidos a pequeños problemas en las etapas inferiores de procesamiento. Estos problemas son casi siempre motivados por el carácter esencialmente local de la extracción de características. Por ejemplo, segmentos particionados o ausentes en la descripción podrían dificultar la detección de algunos planos que, al menos en teoría, deberían haber sido capturados. Sin embargo, debido a la alta velocidad de procesamiento, es extremadamente probable que en alguno de los repetidos intentos de interpretación de cada *frame* se genere antes o después la hipótesis correcta. A partir de ese momento, el mecanismo de corroboración introducido en las secciones anteriores (y cuyos detalles de implementación cubriremos en la sección 4.5.4) hará posible que dicha estructura sea mantenida y consolidada durante el movimiento. Sólo a través de este proceso, al que podríamos llamar de *histéresis interpretativa*, se garantizará la continuidad perceptiva necesaria para la navegación.

Detección de *schemaps*

Una vez que se tienen localizados los planos, es relativamente sencillo examinar las posiciones de los que se hallan en las inmediaciones del robot para categorizar de alguna manera el entorno local en el que éste se encuentra. Las posibles *situaciones tipo* que actualmente con-

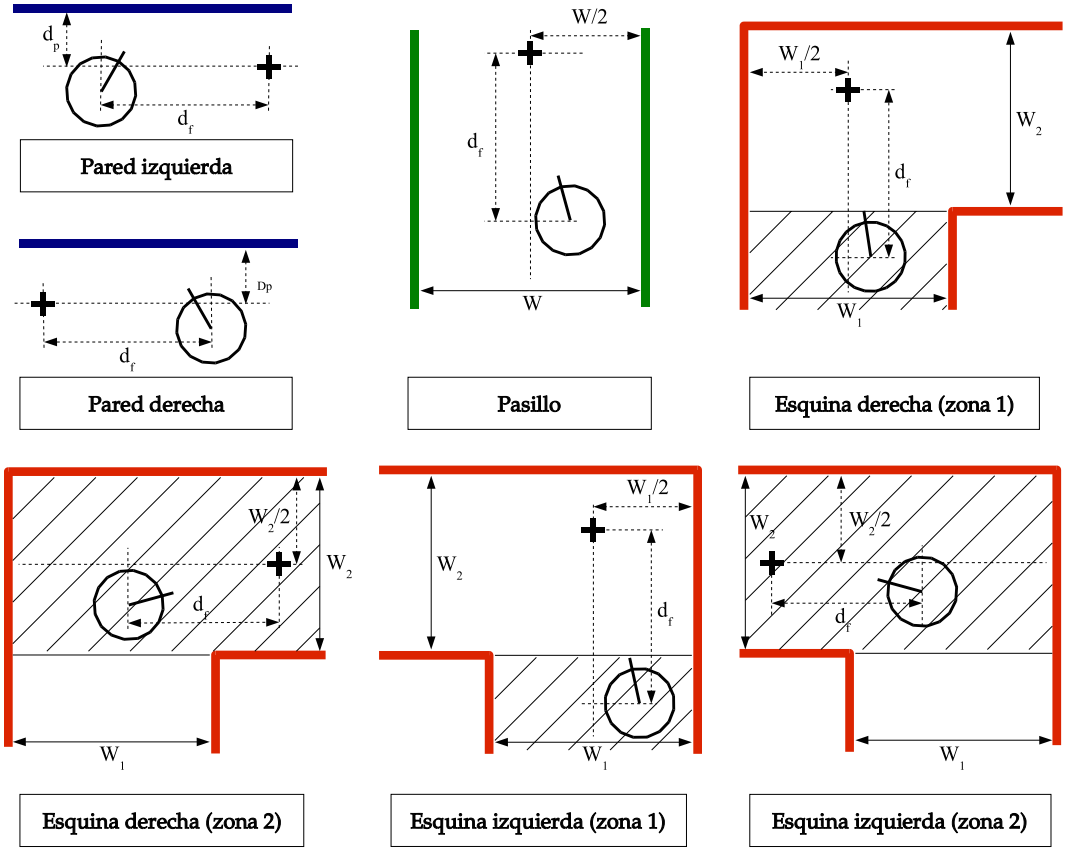


Figura 4.9: *Tipos de* schemaps *manejados por el sistema autónomo (ver texto).*

sidera el prototipo construido, o *schemaps*, tal y como los denominamos ya en la sección 4.5.1, se muestran en la figura 4.9. Una definición importante asociada a cada *schemap* es el llamado *punto de control*. Se trata simplemente de una posición del suelo, dada en el sistema de coordenadas robocéntrico S_{rob}, y que es una suerte de "zanahoria virtual" que especifica la intención de movimiento del sistema, siempre en función de la interpretación de la situación actual y de la posición relativa del robot respecto a ella. Por ejemplo, dentro de un pasillo, este punto se colocará siempre a una distancia predefinida d_f por delante de la plataforma y centrado entre ambas paredes, con el fin de lograr la navegación continua a lo largo del mismo. El punto de control correspondiente a cada *schemap* se muestra como una cruz negra de trazo grueso en las respectivas figuras.

En la posterior sección 4.5.5 concretaremos la generación de órdenes de movimiento en función de la situación de dicho punto con respecto a la plataforma. Veremos cómo un sencillo procedimiento de control basado en un par de reglas simples servirá para generar un comportamiento robusto frente a los detalles irrelevantes del espacio en todas las situaciones con las que se puede encontrar el agente. Esto incluye también el seguimiento de señales (que veremos en la sección siguiente) y la posible navegación de emergencia guiada por el sónar

cuando se detecten obstáculos cercanos peligrosos no percibidos visualmente por el robot[3].

A continuación comentamos brevemente las situaciones tipo reconocidas, justificando la posición concreta del punto de control en función del comportamiento deseado en cada una de ellas:

- Si algún plano corta el camino de avance delante del robot, es considerado un *schemap* de pared frontal. El punto de control se coloca al lado de la misma que minimiza el giro necesario para evitarla y navegar posteriormente en paralelo a ella. Las correspondientes situaciones, que genéricamente llamaremos "*pared izquierda*" y "*pared derecha*", se muestran en la esquina superior izquierda de la figura 4.9, en color azul[4]. Obsérvese que la posición exacta de la cruz se fija en función de dos parámetros predefinidos d_p y d_f, que denotan la distancia perpendicular a la pared y en paralelo a la misma con respecto a la posición actual del robot, respectivamente. Estos parámetros se pueden ajustar para modular la curvatura de la trayectoria deseada de alineamiento con la pared.

- Dos planos verticales paralelos por delante de la dirección de avance se interpretan como un pasillo (en verde en la figura 4.9). El *schemap* de "*pasillo*" tiene un parámetro de configuración variable, para ajustarse a distintas anchuras W entre las paredes correspondientes. La definición del punto de control, en este caso, es en función del valor d_f, que marca un margen de distancia frontal respecto a la plataforma, y de $W/2$, para la navegación centrada entre las paredes.

- Si estando en un pasillo se detecta una pared frontal, se trata de un *schemap* de "*esquina*". En este caso se calcula el mayor espacio libre de salida para ver si es con giro a la izquierda o a la derecha. Las situaciones correspondientes se marcan en rojo en la figura 4.9. En este caso los parámetros configurables son dos, las anchuras W_1 y W_2 de los pasillos de entrada y salida de la esquina, respectivamente. Durante la primera parte del giro, antes de llegar al lugar donde la esquina se dobla, se navega como en el pasillo del que se procede, es decir, con el punto de control centrado en $W_1/2$ y a una distancia frontal d_f. Esto se indica en las figuras con los nombres "*esquina derecha (zona 1)*" y "*esquina izquierda (zona 1)*". Cuando se llega a la altura adecuada, se coloca el punto de control en la dirección de giro pertinente, a una distancia d_f de la plataforma y centrado en el pasillo de salida, otra vez equidistando $W_2/2$ de las respectivas paredes, con el fin de tomar la curva siguiendo una trayectoria suave. Esta segunda etapa se indica en los diagramas "*esquina derecha (zona 2)*" y "*esquina izquierda (zona 2)*". Al salir se recuperaría nuevamente el *schemap* de pasillo, reanudándose la navegación centrada.

[3]Esta última funcionalidad se incluyó sólo por seguridad, y básicamente no afecta al resto de la arquitectura. Hablaremos más detenidamente sobre su inclusión y su mínima influencia en los ejemplos de navegación reales en el capítulo final de la tesis, dedicado a la arquitectura hardware-software del agente autónomo.

[4]El código de colores empleado en cada *schemap* será útil para el seguimiento de los ejemplos mostrados a partir de ahora, tanto en este capítulo como en el posterior.

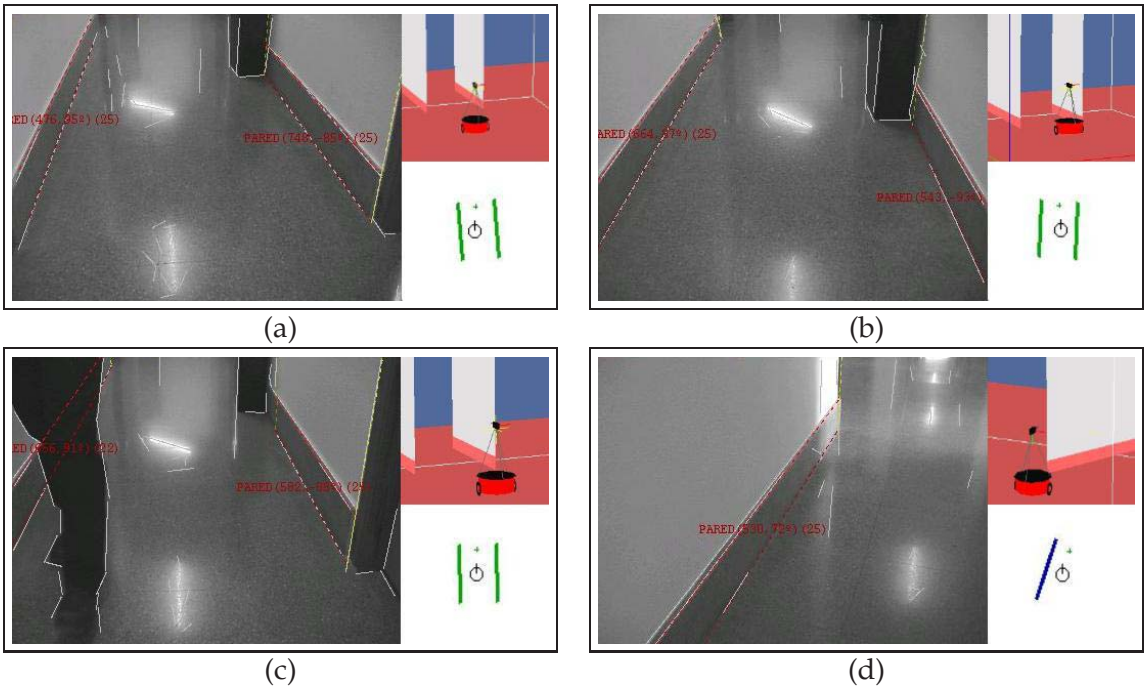


Figura 4.10: Schemaps *de pasillo y de pared: a) Pasillo inicial, en el momento de la detección. b) Instantes después, tras centrarse el robot en el mismo por la navegación hacia el punto de control. c) Robustez frente a obstáculos móviles, por el mecanismo de mantenimiento de la consistencia. d)* Schemap *de pared frontal. En presencia de éste, el robot coloca el punto de control al lado en el que tiene que realizar un menor giro para evitar la colisión (en este caso a la derecha). En caso de encontrarse una pared paralela a la anterior, el* schemap *de pared puede conmutar adecuadamente a una situación de pasillo.*

- Por último, y si no hay ningún plano en las inmediaciones, entonces se considera que hay una situación de "*espacio libre*" y se navega simplemente hacia adelante, colocando el punto de control por delante de la dirección de avance del robot.

En la figuras 4.10 y 4.11 se muestran ejemplos de interpretaciones de *schemaps* de pasillo, pared y esquina desde distintos lugares, en algún caso incluso con mantenimiento temporal de la consistencia a pesar de la presencia de obstáculos móviles (en la posterior sección 4.5.4 comentaremos cómo se logra esta propiedad). La información mostrada en las figuras incluye la reproyección de la interpretación sobre la imagen original y los segmentos extraídos (a la izquierda), la correspondiente reconstrucción tridimensional de la escena practicada en tiempo real durante la navegación (en la esquina superior derecha), y el esquema de la situación actual con el punto de control utilizado en la navegación (en la esquina inferior derecha). En la imagen de la izquierda se sobreimpresiona también la posición de los elementos reproyectados, siempre en mm respecto a S_{rob}. En lo que resta de capítulo, todas las figuras utilizadas para ilustrar las distintas características de la arquitectura se mostrarán siguiendo esta misma organización.

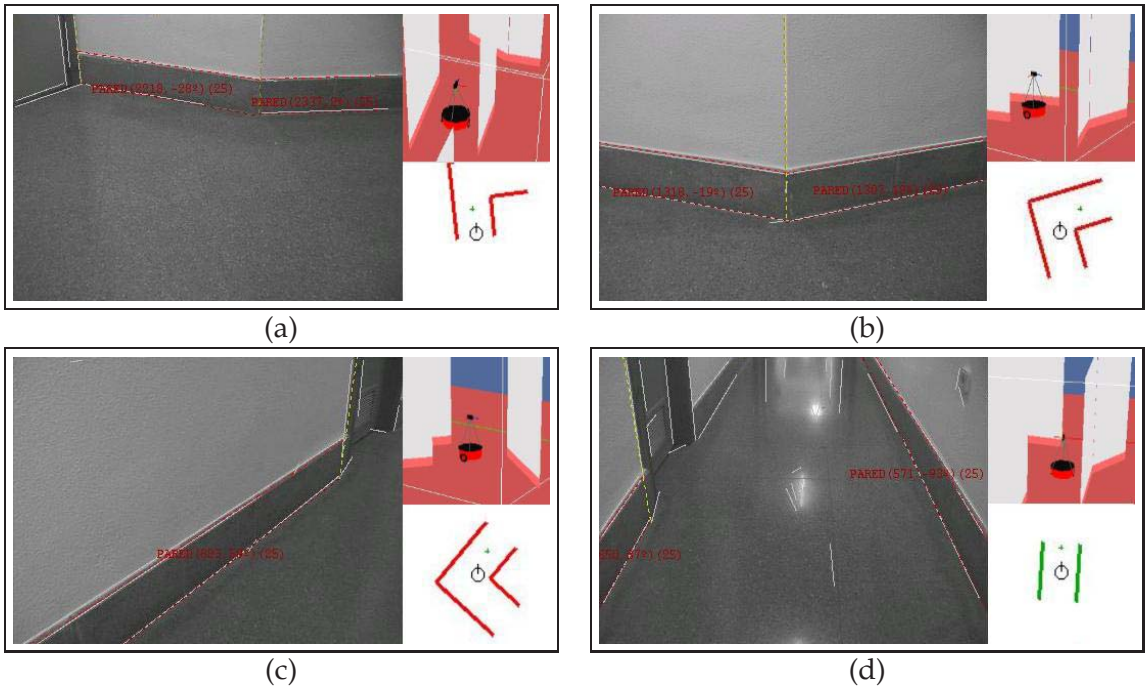


Figura 4.11: *Mantenimiento del* schemap *de esquina con giro a la derecha en cuatro instantes consecutivos: a) Momento de la conmutación de hipótesis de pasillo a la de giro, al detectar los tramos de pared frontales. Obsérvese el uso en la interpretación de elementos que, aunque no son directamente perceptibles, se encuentran en la memoria a corto plazo de lo ya visto (en este caso se trata de los planos correspondientes a las paredes del pasillo). b) Comienzo del giro, al sobrepasar la esquina. c) En mitad del giro. d) Final del giro, con la conmutación de nuevo a la situación de pasillo.*

Detección de señales y obstáculos

Una vez conocida la estructura de fondo de la escena, no supone ninguna dificultad reconocer cierto tipo de contornos contenidos en los planos que forman la misma. Algunos de estos contornos se podrían entonces interpretar como señales a las que se asocian comportamientos especiales, dependiendo de su posición respecto al robot y de propiedades tales como su forma, tamaño, color, etc. En nuestro caso, como ejemplos, emplearemos flechas para indicar un cambio de dirección en la trayectoria de navegación, y cruces para señalar lugares de parada obligatoria. Estas señales se podrán encontrar en el suelo o en cualquiera de los planos verticales, indistintamente. En la estimación de la forma y el tamaño de cada figura, como ya adelantamos en la introducción a esta sección, resultará básica la determinación del contexto en el que se encuentran, puesto que aparecerán deformadas por la perspectiva. Así, en lugar de intentar un reconocimiento directo a través de invariantes proyectivos, al estilo de los descritos por Pizlo y Rosenfeld (1992), Carlsson *et al.* (1996) y trabajos relacionados, nosotros rectificaremos primero la señal según el plano en el que esté ubicada, para poder clasificar después su forma euclídea de modo computacionalmente más sencillo y eficiente.

La localización inicial de contornos candidatos puede realizarse buscando secuencias de segmentos con extremos cercanos y colores compatibles que se cierren sobre sí mismas. Co-

mo ya hicimos en el algoritmo 4.2 para comparar con el color del suelo, la compatibilidad cromática la determinaremos mediante un simple umbralizado de la distancia euclídea en el espacio RGB entre los respectivos colores laterales de los segmentos de la secuencia. Esta sencilla técnica demostró ser suficiente para la detección de señales en nuestro dominio de aplicación. Aún así, en circunstancias en las que esta aproximación pudiera resultar poco robusta siempre se podría emplear algún procedimiento más sofisticado, que usase otros espacios de representación de color más apropiados, o medidas de correlación entre colores mejor justificadas estadísticamente, por ejemplo.

Un posible procedimiento de extracción basado en estas ideas se muestra en el algoritmo 4.3. En cada paso del bucle externo se considera un segmento $\overline{\mathbf{d}}_i$ para iniciar un nuevo contorno. Las iteraciones del bucle interno intentan buscar otro $\overline{\mathbf{d}}_{i'}$ con el que continuar el contorno actual. Para ello, primero se coloca el segmento candidato en la orientación correcta, con su primer extremo como el más cercano al final del contorno que se está formando. Después se comprueba si dicho punto está lo suficientemente próximo al extremo final actual, a una distancia menor que un determinado umbral τ_{geom}. También se comprueba la compatibilidad cromática con los segmentos del contorno en construcción, verificando que la diferencia con los colores izquierdo y derecho del candidato no sean mayores que otro valor τ_{crom}. Así se continúan captando segmentos hasta que no pueden añadirse más, o bien hasta que logra cerrarse el contorno llegando al punto inicial. En el primer caso se descarta el intento actual para continuar buscando nuevas secuencias partiendo de otro segmento, mientras que en el segundo se considera hallada una solución y se añade a la salida. En ambas situaciones se marcan los segmentos correspondientes como visitados para no recorrerlos de nuevo en siguientes intentos.

Cada contorno extraído debe ser entonces rectificado y ubicado en el espacio tridimensional, donde podrá ser interpretado y, en su caso, utilizado en la navegación. El correspondiente procedimiento de generación de hipótesis de señales se recoge en el algoritmo 4.4. En caso de que el contorno completo esté íntegramente contenido en uno de los planos de la escena, ya sea el del suelo o cualquiera de los verticales, se obtienen sus coordenadas 3D con la correspondiente matriz rectificadora. Entonces se intenta clasificar como alguna de las señales reconocidas, una flecha o una cruz en nuestra implementación. Al trabajar ya en coordenadas euclídeas sobre una figura plana, el reconocimiento se puede practicar simplemente por la relación de distancias y ángulos relativos entre los segmentos de la secuencia, de modo incluso independiente del tamaño de la señal. Por simplicidad, no incluimos el algoritmo de reconocimiento correspondiente, que resulta en todo caso trivial.

Si se logra clasificar alguno de los contornos, automáticamente pasa a formar parte de la interpretación actual. Por consistencia con los modelos de planos de los algoritmos anteriores, las señales se representan como conjuntos $Model_{k_s}^{signal}$ de segmentos tridimensionales. En caso contrario, y si el contorno está contenido completa o parcialmente dentro del suelo, aún podría rectificarse usando $\mathtt{R}_{sue}$ para ser sencillamente considerado como un obstáculo y

ENTRADA:
- Conjunto de segmentos $Data = \{\overline{\mathbf{d}}_i \mid i = 1, \ldots, D\}$ extraídos de la imagen usando el algoritmo 2.2. Cada $\overline{\mathbf{d}}_i$ incluye los extremos en coordenadas de píxel y la información de color.
- Distancia máxima entre extremos de segmentos para ser considerados colindantes, τ_{geom}.
- Distancia máxima entre colores en el espacio RGB para ser considerados compatibles, τ_{crom}.

SALIDA:
- Conjunto de C contornos cerrados $\{Contour_k = \{\overline{\mathbf{c}}_{k,j} \mid j = 1, \ldots, N_k\} \mid k = 1, \ldots, C\}$. Cada $Contour_k$ consta de N_k segmentos de imagen $\overline{\mathbf{c}}_{k,j}$ contiguos, ordenados y cerrados.

ALGORITMO:

- Inicializar el contador de contornos $k = 0$.
for $i = 1$ ***to*** D ***do*** /* *Bucle externo de recorrido de segmentos de datos:* */
  if $\overline{\mathbf{d}}_i$ no está marcado como *"Visitado"* ***then***
    - Inicializar un nuevo contorno candidato $Contour_k = \{\overline{\mathbf{d}}_i\}$, con $N_k = 1$ segmentos.
    - Guardar los extremos $\boldsymbol{x}_1^{\overline{\mathbf{d}}_i}$ y $\boldsymbol{x}_2^{\overline{\mathbf{d}}_i}$ de $\overline{\mathbf{d}}_i$ como puntos inicial y final actuales, $\boldsymbol{x}_{ini}$ y $\boldsymbol{x}_{fin}$.
    while *Cierto* ***do*** /* *Bucle hasta cerrar* $Contour_k$*, o bien saber que ya no puede completarse:* */
      for $i' = 1$ ***to*** D ***do*** /* *Bucle interno de recorrido de segmentos de datos:* */
        if $\text{dist}(\boldsymbol{x}_{fin}, \boldsymbol{x}_2^{\overline{\mathbf{d}}_{i'}}) < \text{dist}(\boldsymbol{x}_{fin}, \boldsymbol{x}_1^{\overline{\mathbf{d}}_{i'}})$ ***then*** /* *Damos a* $\overline{\mathbf{d}}_{i'}$ *la orientación correcta:* */
          - Invertir los extremos $\boldsymbol{x}_1^{\overline{\mathbf{d}}_{i'}}$ y $\boldsymbol{x}_2^{\overline{\mathbf{d}}_{i'}}$ y colores $\mathbf{rgb}_{izq}^{\overline{\mathbf{d}}_{i'}}$ y $\mathbf{rgb}_{der}^{\overline{\mathbf{d}}_{i'}}$ del segmento $\overline{\mathbf{d}}_{i'}$.
        endif
        /* *Calculamos las distancias geométrica y cromática del segmento candidato al contorno que se está formando, para ver si es compatible y añadirlo al final del mismo:* */
        $d_{geom} = \text{dist}(\boldsymbol{x}_{fin}, \boldsymbol{x}_1^{\overline{\mathbf{d}}_{i'}})$. $\quad d_{crom} = \text{máx}\{\text{dist}(\mathbf{rgb}_{izq}^{\overline{\mathbf{d}}_i}, \mathbf{rgb}_{izq}^{\overline{\mathbf{d}}_{i'}}), \text{dist}(\mathbf{rgb}_{der}^{\overline{\mathbf{d}}_i}, \mathbf{rgb}_{der}^{\overline{\mathbf{d}}_{i'}})\}$.
        if $d_{geom} < \tau_{geom}$ ***and*** $d_{crom} < \tau_{crom}$ ***then*** /* *Son compatibles:* */
          if $\text{dist}(\boldsymbol{x}_2^{\overline{\mathbf{d}}_{i'}}, \boldsymbol{x}_{ini}) < \tau_{geom}$ ***then*** /* *El contorno se ha completado:* */
            - Marcar todos los segmentos de $Contour_k$ como *"Visitados"*.
            - Cerrar los segmentos del contorno intersectando los extremos consecutivos, y guardar $Contour_k$ como parte de la salida.
            - Hacer $k = k + 1$.
            break(***while***). /* *Salimos para buscar nuevos contornos.* */
          endif else /* *Añadimos nuevo segmento al contorno:* */
            - Hacer $Contour_k = Contour_k \cup \{\overline{\mathbf{d}}_{i'}\}$.
            - Hacer $N_k = N_k + 1$.
            - Hacer $\boldsymbol{x}_{fin} = \boldsymbol{x}_2^{\overline{\mathbf{d}}_{i'}}$.
            continue(***while***). /* *Continuamos buscando un nuevo segmento para el contorno.* */
          endelse
        endif
      endfor
      /* *Si se llega hasta aquí significa que no se pudo cerrar la secuencia. Aún así, marcamos los segmentos intentados como visitados (puesto que sabemos que no llevan a cerrar un contorno):* */
      - Marcar todos los segmentos de $Contour_k$ como *"Visitados"*.
      break(***while***). /* *Pasamos a intentar otro.* */
    endwhile
  endif
endfor
- Hacer $C = k$, y devolver el conjunto de contornos encontrados $\{Contour_k\}$, con $k = 1, \ldots, C$.

Algoritmo 4.3: *Localización de contornos cerrados a partir de segmentos con información color.*

ENTRADA:
- Conjunto de modelos $\{Model_{k_p}^{plane} |\ k_p = 1, \ldots, M^{plane}\}$ de plano vertical, definidos como en el algoritmo 4.2.
- Conjunto de contornos $\{Contour_{k_c} \mid k_c = 1, \ldots, C\}$ de objetos en la imagen, obtenidos con el algoritmo 4.3.

SALIDA:
- Conjunto de modelos $\{(Model_{k_s}^{signal}, etiq_{k_s}) \mid k_s = 1, \ldots, M^{signal}\}$ de señales reconocidas. Cada modelo contiene los segmentos 3D que definen el contorno de la señal, en coordenadas de S_{rob} en mm, y una etiqueta que indica el tipo de señal reconocida (flecha, cruz, etc.).

ALGORITMO:

Inicializar el contador de señales encontradas, $k_s = 0$.
for $k_c = 1$ ***to*** C ***do*** $/* \textit{Recorrido de contornos bidimensionales:} */$
- Comprobar si todos los extremos de los segmentos contenidos en $Contour_{k_c}$ están en un mismo plano (de suelo o vertical), usando el algoritmo 4.1.

if se cumple la condición anterior ***then***
- Rectificar todos los segmentos del contorno usando el mismo procedimiento, para obtener el modelo candidato $Model_{k_s}^{signal}$.
- Emplear cualquier técnica de reconocimiento de contornos euclídeos para determinar si el modelo candidato es algún tipo de señal conocida (cruz, flecha, etc.).

if se reconoce como algún tipo de señal ***then***
- Etiquetar $Model_{k_s}^{signal}$ con el tipo de señal reconocida, y guardarla como parte de la salida.
- Regularizar el contorno reajustando los extremos de los segmentos, para hacerlos coincidentes entre elementos colindantes de la secuencia.
- Hacer $k_s = k_s + 1$.

endif

endif

endfor

- Hacer $M^{signal} = k_s$, y devolver el conjunto de señales $\{(Model_{k_s}^{signal}, etiq_{k_s})\}$, con $k_s = 1, \ldots, M^{signal}$.

Algoritmo 4.4: *Generación de hipótesis de señales y su posición 3D a partir de los contornos extraídos de la imagen y el modelo planar de la escena.*

navegar rodeándolo, aunque esta posibilidad no se incluye en el algoritmo esquematizado.

El conjunto de señales reconocidas y las correspondientes ubicaciones del punto de control que guiará la navegación se muestran en la figura 4.12. Por ejemplo, ante una cruz en el suelo este punto se coloca directamente sobre el centro de la señal, para provocar la parada del robot justamente encima. En una flecha, alternativamente, se distingue una fase previa en la que el agente está aún lejos de ella, y el punto se sitúa sobre la misma para provocar un acercamiento inicial. Esto se indica en los diagramas con la leyenda "*zona 1*". Una vez que el robot se ha acercado lo suficiente, en concreto a menos de una distancia predeterminada d_s, se cambia la situación del punto de control a una longitud d_f en la dirección indicada por la flecha para inducir el giro. Nótese que de nuevo estos valores se pueden proporcionar directamente en milímetros para planificar la trayectoria de navegación deseada, gracias al carácter euclídeo de la reconstrucción practicada. Esta nueva situación se corresponde con los diagramas etiquetados como "*zona 2*". Naturalmente, para los efectos de la navegación

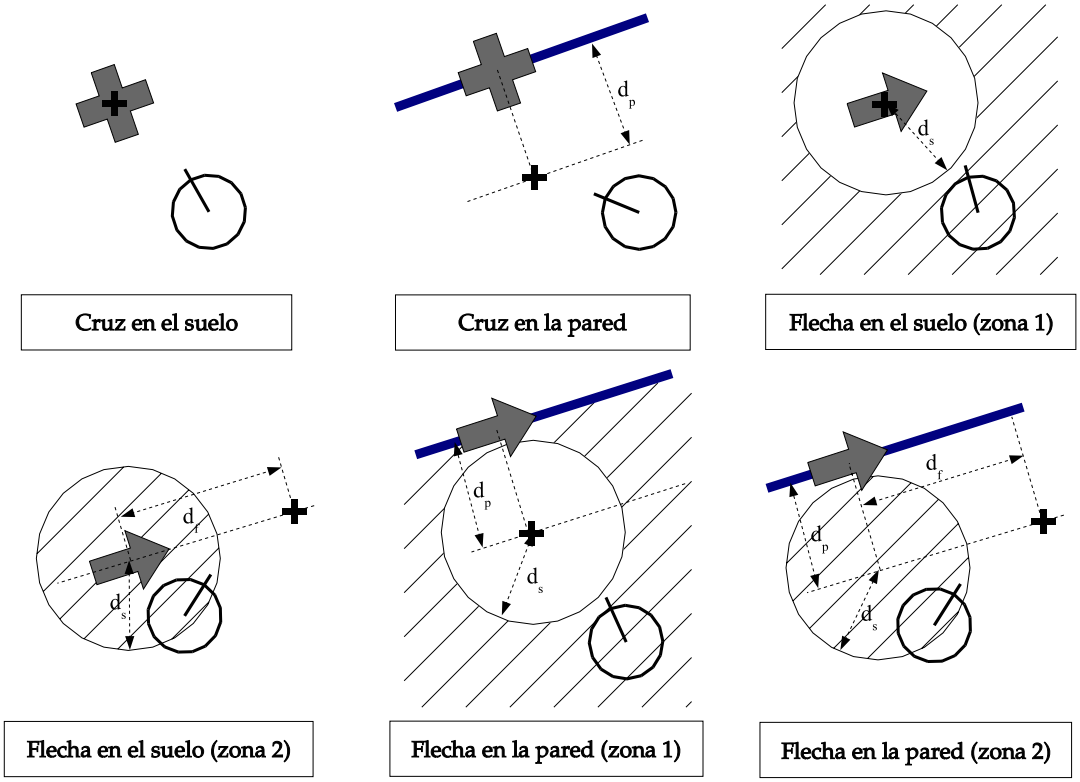


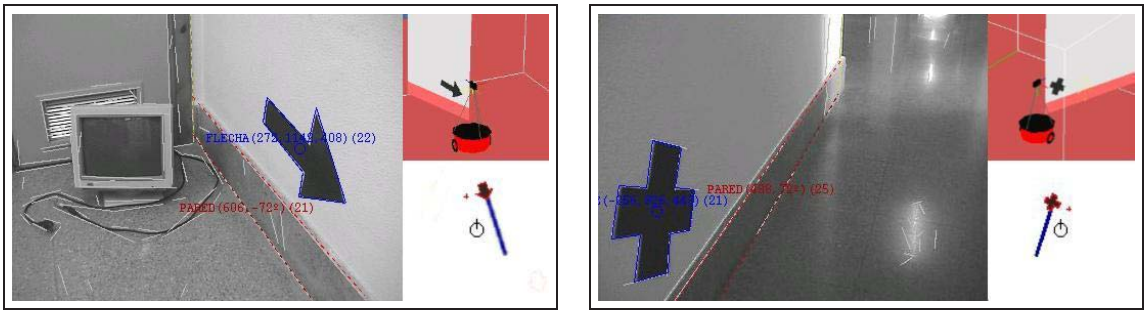
Figura 4.12: *Tipos de señales reconocidas por la arquitectura propuesta, y definición de la posición del punto de control asociado. Obsérvese la diferencia en la ubicación de éste último en función de si la señal se encuentra en el suelo o en una pared.*

se tienen que distinguir las señales que están en el suelo de las que están en las paredes. Así, mientras que en las primeras el movimiento se rige por las reglas ya comentadas, en las segundas se define un comportamiento análogo, pero situando los respectivos puntos de control a una distancia determinada d_p por delante de la pared implicada.

Las imágenes de las figuras 4.13(a) y 4.13(b) muestran un par de situaciones de navegación reales en las que, a pesar de la deformación proyectiva, se reconocen adecuadamente una señal de cruz y otra de flecha, respectivamente. Al igual que en ejemplos anteriores, es perfectamente apreciable la reconstrucción euclídea y la posición de la cruz de control en las correspondientes interpretaciones esquemáticas.

4.5.4. Corroboración y seguimiento de hipótesis

Los procedimientos anteriores son eficientes, en el sentido de que evitan la exploración de un número intratable de combinaciones de segmentos en la ineludible etapa de generación de las hipótesis de partida, gracias a la utilización de propiedades naturales del dominio. De todos modos, no sería viable tener que disparar continuamente este tipo de procesos desde cero. El tiempo de respuesta aumentaría significativamente y se producirían numerosos fa-



(a) (b)

Figura 4.13: *Reconocimiento de señales, aún en presencia de deformaciones proyectivas por la situación del plano en que están contenidas con respecto a la cámara: a) Flecha. b) Cruz.*

llos en las sucesivas interpretaciones, factores ambos que acabarían afectando seriamente a la operatividad del sistema.

En lugar de ello, las hipótesis de alto nivel generadas de modo más o menos exigente en instantes anteriores serán reproyectadas sobre los niveles inferiores usando la información de movimiento, hasta llegar a los estímulos sensoriales donde podrán o no encontrar corroboración. Así, en lugar de construir modelos nuevos a cada momento, los procesos *bottom-up* se ejecutarán sólo sobre aquellos segmentos no recogidos ya por la interpretación actual, con el fin de reflejar los cambios estructurales importantes. Mientras tanto, la mayor parte del peso de la percepción recaerá sobre un mecanismo de predicción anticipativa, en el que se siguen durante largos periodos de tiempo las hipótesis elaboradas con anterioridad, a través de presiones descendentes (*top-down*) que focalizan su atención sobre la mayoría de los segmentos extraídos del *frame* en curso.

El mecanismo de seguimiento y confirmación de hipótesis debe mantener actualizada la estructura de la interpretación al moverse el agente. Las características que cabe exigir a este proceso son básicamente las siguientes:

- Capacidad de anticipación sobre lo que se espera ver en cada instante, derivada directamente de la estructura percibida previamente y de la información propioceptiva de movimiento. La predicción, al menos en teoría, podría incluso llegar al nivel del estímulo elemental (colores de los píxeles), aunque en nuestro caso nos bastará con pronosticar la situación y los colores de los segmentos extraídos en el *frame* siguiente.

- Ayuda en el rechazo de hipótesis de interpretación erróneas, por clara incompatibilidad con la predicción.

- Robustez en el seguimiento, puesto que la eliminación de una hipótesis debería ocurrir sólo en el caso de que realmente dejara de percibirse la estructura asociada.

- Mantenimiento de la consistencia frente a obstáculos. Las estructuras deberían poder seguirse aunque aparezcan eventualmente objetos (móviles o no) que las oculten par-

cialmente. Esta propiedad añade un componente importante de flexibilidad, puesto que en presencia de estos obstáculos quizá sería muy difícil interpretar la situación desde cero, mientras que no hay tanto inconveniente en mantener el modelo adecuado si éste viene impuesto desde instantes anteriores.

- Rapidez de procesamiento entre *frames*. Esta característica reduce la disparidad entre imágenes consecutivas, lo que contribuye también a la facilidad en el *tracking*.

La instanciación práctica en la arquitectura de la idea de presión descendente (de los modelos hacia los datos) es también muy sencilla. Primero hay que reproyectar sobre la imagen la estructura interpretativa actual, convenientemente corregida con la odometría entre la toma del *frame* anterior y el actual. Obviamente, en la reproyección hay que recortar los segmentos sobre la imagen, para tener en cuenta en el proceso de corroboración sólo aquellos fragmentos de la interpretación que deberían aparecer en ella. Volvemos a denotar con $\overline{\mathbf{M}}_j$ a los segmentos 3D provenientes de los modelos (ya sean planos verticales o señales), y con $\overline{\mathbf{m}}_j$ a los correspondientes 2D, una vez proyectados y recortados.

Hay que medir entonces el error de reproyección sobre los datos, para ver si decidimos corroborar la estructura o, por el contrario, penalizarla para posiblemente eliminarla al cabo de un cierto tiempo. Sean de nuevo $\overline{\mathbf{d}}_i$ los segmentos extraídos de la imagen. Para decidir los segmentos de datos $\overline{\mathbf{d}}_i$ que encajan con un segmento $\overline{\mathbf{m}}_j$ del modelo, establecemos una zona de distancia máxima a éste último, de la cual no pueden salirse los extremos de los $\overline{\mathbf{d}}_i$ candidatos para el *matching*. Al conjunto de segmentos incluidos en esta zona y que tengan colores compatibles con los del modelo lo llamaremos $Match(\overline{\mathbf{m}}_j)$. Podemos tomar entonces como función de error de reproyección a la suma de las distancias de los tramos de segmentos del modelo ($\overline{\mathbf{m}}_j$) no cubiertos por segmentos de datos correspondientes ($\overline{\mathbf{d}}_i \in Match(\overline{\mathbf{m}}_j)$), dividida por la suma total de distancias de los $\overline{\mathbf{m}}_j$ para normalizar (por supuesto siempre dentro de la imagen, ya que alguna parte del modelo podría caer fuera de la misma). La figura 4.14 esquematiza la idea propuesta, con un modelo de plano de ejemplo formado por cuatro segmentos. La forma concreta de la función de *error por omisión* de los datos sobre el modelo que utilizaremos es:

$$E_{om}(Model, Data) = 1 - \frac{\sum_{\overline{\mathbf{m}}_j \in Model} \sum_{\overline{\mathbf{d}}_i \in Match(\overline{\mathbf{m}}_j) \subseteq Data} \mathcal{P}(\overline{\mathbf{d}}_i)}{\sum_{\overline{\mathbf{m}}_j \in Model} \mathcal{L}(\overline{\mathbf{m}}_j)} \tag{4.6}$$

Donde $\mathcal{L}(\overline{\mathbf{m}}_j)$ es la longitud de $\overline{\mathbf{m}}_j$ y $\mathcal{P}(\overline{\mathbf{d}}_i)$ la longitud de $\overline{\mathbf{d}}_i$ proyectado sobre el correspondiente $\overline{\mathbf{m}}_j$, tal y como se ilustra en la figura.

Más exactamente, para evitar que la suma del numerador pueda superar a la del denominador en la expresión 4.6 cuando se solapen los segmentos de datos, y la función de error quede inconsistente, definiremos $\mathcal{P}(\overline{\mathbf{d}}_i)$ como la longitud de la proyección de $\overline{\mathbf{d}}_i$ sobre $\overline{\mathbf{m}}_j$ que no haya sido ya cubierta por un $\overline{\mathbf{d}}_k$ anterior, con $k < i$. Esta definición de $\mathcal{P}$ se ilustra en la

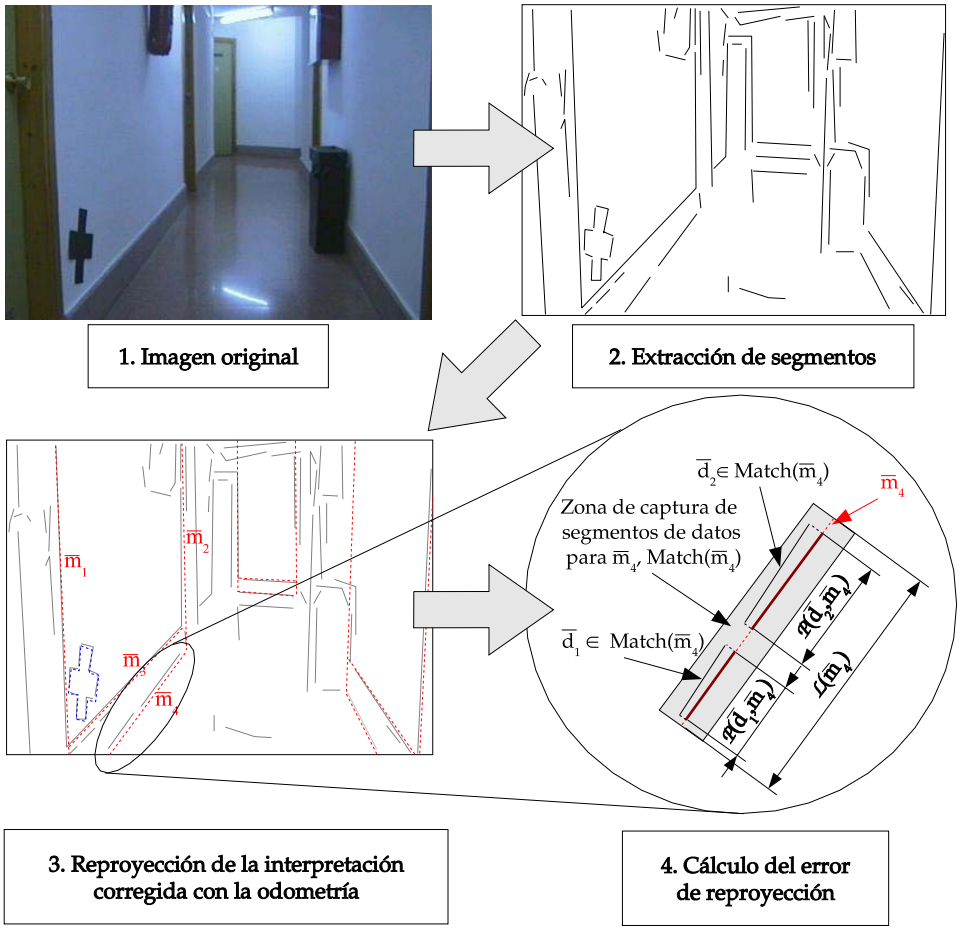


Figura 4.14: *Ilustración del cálculo de error de reproyección (ver texto).*

figura 4.15. La función de error $E_{om}(Model, Data)$ tiene entonces un valor mínimo de cero para modelos completamente cubiertos por los datos, y máximo de uno para aquellos que no tienen ningún soporte. Obsérvese también que no se penaliza la fragmentación de segmentos en la imagen. De acuerdo con la filosofía propuesta, pues, resulta mucho más sencillo corroborar un modelo que generarlo como hipótesis desde cero. La función de error descrita es una simplificación de la propuesta por Beveridge y Riseman (1997) (ver sección 4.2.5)[5].

El paso siguiente es realinear el modelo de acuerdo con la información visual. En nuestro caso, cada segmento reproyectado del modelo $\overline{\mathbf{m}}_j$ es corregido utilizando la dirección de alineamiento dada por el segmento $\overline{\mathbf{d}}_i$ más largo de los incluidos en $Match(\overline{\mathbf{m}}_j)$. En caso de que este conjunto sea vacío, simplemente se deja el valor de $\overline{\mathbf{m}}_j$ modificado previamente por la odometría. Se rectifica así continuamente el error cometido en el seguimiento del modelo, que de otra forma crecería indefinidamente hasta desajustar a éste por completo de la secuen-

[5]En la referencia comentada, además de este error por omisión se considera otro sumando que tiene en cuenta el desalineamiento entre los datos y el modelo, integrando la distancia media al cuadrado de los $\overline{\mathbf{d}}_i$ a los correspondientes $\overline{\mathbf{m}}_j$ infinitamente prolongados, aunque nosotros no lo hemos incluido en nuestra implementación.

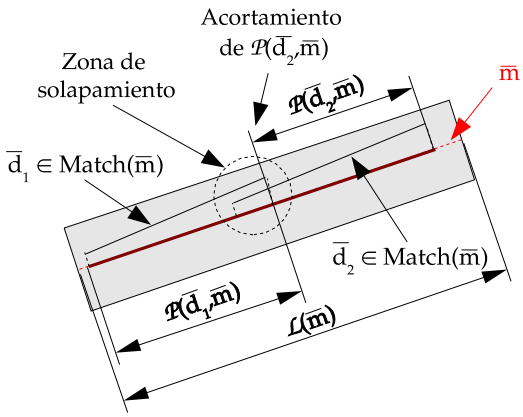


Figura 4.15: *Definición de los valores $\mathcal{P}(d_i)$ en caso de solapamientos entre segmentos (ver texto).*

cia de imágenes. Por supuesto, los extremos de los segmentos $\overline{\mathbf{m}}_j$ corregidos de este modo se calcularán siempre por intersección con los segmentos contiguos del modelo (es decir, los que forman el contorno para una señal, o los correspondientes a intersecciones entre planos en el caso de las paredes, puertas, etc.). Dicho de otra forma, la única información del segmento $\overline{\mathbf{d}}_i$ que se utiliza en la actualización de $\overline{\mathbf{m}}_j$ es la dirección del mismo, nunca sus extremos concretos. Esta regularización es similar a la practicada en los procedimientos de generación de hipótesis 4.2 y 4.4, y es obligatoria para asegurar que los modelos se mantienen consistentes en su evolución.

Otros métodos de seguimiento y actualización, como los basados en 3D a partir de estéreo, por ejemplo, poseen una incertidumbre no despreciable. Ello puede llevar a la necesidad de aplicar esquemas de procesamiento más complejos. Uno de los más habituales en el seguimiento de características es el filtro de Kalman (Deriche y Faugeras, 1990) y sus ampliaciones (Davison y Murray, 1998). En nuestro caso, no obstante, podremos permitirnos la utilización del enfoque descrito, más sencillo, puesto que la información 3D procede de la interpretación de una sola imagen en la que los segmentos bidimensionales son localizados con una precisión relativamente alta. Puede así corregirse el modelo usando directamente estos segmentos, en lugar de tener que combinarlos de alguna forma con la información anterior, al estilo del filtro de Kalman. En otras palabras, asignaremos mucha mayor credibilidad a los estímulos visuales percibidos en el último instante (debidamente corroborados) que al modelo previo actualizado con la información de movimiento (que posiblemente presentará un mayor error, acumulado por la odometría incremental).

El algoritmo 4.5 resume el procedimiento de seguimiento, corroboración y corrección de estructuras propuesto. Cada uno de los modelos (planos y señales) presentes en la interpretación se etiqueta con una variable entera c, a la que llamaremos *índice de corroboración*. Este valor se fija inicialmente a un valor máximo arbitrario c_{max} en el momento de la generación de la estructura. Posteriormente, en cada *frame* en el que el elemento de la interpretación re-

ENTRADA:
- Valores del movimiento (t_x, t_y, θ) medidos por la odometría entre los instantes (t) y $(t+1)$.
- Conjunto de segmentos de datos $Data = \{\overline{\mathbf{d}}_i^{(t+1)} \mid i = 1, \ldots, D\}$ extraídos de la imagen en el instante $(t+1)$ usando el algoritmo 2.2. Cada $\overline{\mathbf{d}}_i^{(t+1)}$, por tanto, viene dado por sus extremos en coordenadas de píxel e información de color a cada lado.
- Conjunto $\{Model_k^{(t)} = (\{\overline{\mathbf{M}}_{k,j}^{(t)} \mid j = 1, \ldots, N_k^{(t)}\}, c_k) \mid k = 1, \ldots, M^{(t)}\}$ de modelos presentes en la interpretación en el instante (t), obtenidos con los algoritmos de generación de hipótesis para planos (4.2) o señales (4.4), y posiblemente corregidos por este algoritmo en pasos anteriores. Cada $Model_k^{(t)}$ consta de $N_k^{(t)}$ segmentos 3D $\overline{\mathbf{M}}_{k,j}^{(t)}$, dados por sus extremos en coordenadas de S_{rob} en mm y con información de color, y de un valor entero de corroboración c_k, con $1 \leq c_k \leq c_{max}$.
- Matriz de proyección `P`, obtenida mediante cualquiera de los algoritmos 3.2-3.8.
- Máximo valor de error de reproyección permitido a cada modelo, τ_{proy}, con $0 < \tau_{proy} < 1$.

SALIDA:
- Conjunto de modelos $\{Model_k^{(t+1)} \mid k = 1, \ldots, M^{(t+1)}\}$, actualizados al instante $(t+1)$.

ALGORITMO:

Predicción:

- Calcular la matriz de transformación odométrica `O` a partir de (t_x, t_y, θ), usando 3.17.

for $k = 1$ ***to*** M ***do***

for $j = 1$ ***to*** N_k ***do***

- Transformar los extremos del segmento $\overline{\mathbf{M}}_{k,j}^{(t)}$ utilizando la matriz `O`, para obtener $\overline{\mathbf{M}}_{k,j}^{(t+1)}$.
- Obtener la proyección $\overline{\mathbf{m}}_{k,j}^{(t+1)}$ de $\overline{\mathbf{M}}_{k,j}^{(t+1)}$ usando `P`, y recortando a los límites de la imagen.

endfor

endfor

Corroboración y corrección:

for $k = 1$ ***to*** M ***do***

- Calcular el error de reproyección $E_{om}(Model_k^{(t+1)}, Data^{(t+1)})$ usando 4.6.

if $E_{om}(Model_k^{(t+1)}, Data^{(t+1)}) < \tau_{proy}$ ***then*** */* Si hay corroboración: */*

- Hacer $c_k = \text{máx}\{c_k + 1, c_{max}\}$. */* Incrementamos la corroboración del modelo. */*

for $j = 1$ ***to*** N_k ***do***

if $Match(\overline{\mathbf{m}}_{k,j}^{(t+1)}) \neq \varnothing$ ***then*** */* Si hay algún* $\overline{\mathbf{d}}_i$ *de soporte para* $\overline{\mathbf{m}}_{k,j}^{(t+1)}$*, corregimos con él: */*

- Corregir $\overline{\mathbf{m}}_{k,j}^{(t+1)}$ utilizando el segmento $\overline{\mathbf{d}}_i$ más largo del conjunto $Match(\overline{\mathbf{m}}_{k,j}^{(t+1)})$.

endif

- Recuperar el segmento 3D $\overline{\mathbf{M}}_{k,j}^{(t+1)}$ a partir del 2D $\overline{\mathbf{m}}_{k,j}^{(t+1)}$ usando el algoritmo 4.1.

endfor

- Regularizar el modelo $Model_k^{(t+1)}$ haciendo coincidir extremos colindantes, forzando características ideales (como la verticalidad en los lados de los planos, p.e.), etc.

endif else */* Si no hay corroboración: */*

- Hacer $c_k = c_k - 1$.

if $c_k = 0$ ***then***

- Borrar el modelo $Model_k^{(t+1)}$ de la interpretación en $(t+1)$.

endif

endelse

endfor

Algoritmo 4.5: *Procedimiento de seguimiento, corroboración y corrección de estructuras (planos y señales) que forman parte de la interpretación, entre el instante actual* (t) *y el siguiente* $(t+1)$.

proyectado sobre la imagen no encuentre suficiente estímulo sensorial que le dé soporte (es decir, E_{om} sea mayor que un determinado umbral τ_{proy}), c se decrementará en una unidad. Si este contador llega a cero en algún momento, la correspondiente estructura es eliminada de la interpretación. Inversamente, c se incrementará cada vez que exista dicha confirmación, aunque saturándose en el valor máximo c_{max}. Este último valor, por tanto, se relacionaría con el máximo tiempo que se desea mantener una interpretación no corroborada. El resto del procedimiento simplemente resume el ciclo de actualización odométrica, reproyección de la estructura, comprobación del error y posterior corrección de la misma expuesto en los párrafos anteriores.

La figura 4.16 ilustra un caso real de operación del algoritmo, en el que incluso se priva momentáneamente de visión al agente autónomo, poniendo a prueba la capacidad de anticipación sobre la imagen a más largo plazo. Los planos de la escena aparecen en este caso etiquetados con su posición respecto al robot en la forma (d, α), donde d es la distancia en perpendicular desde la base del plano hasta el centro de la plataforma (origen de coordenadas de S_{rob}), en mm, y α el ángulo en grados de la normal del plano respecto a la dirección de avance. Las señales detectadas, en azul, se etiquetan con la posición (x, y, z) de su centro de masas, marcado con un círculo en cada una. Tanto los planos como las señales van asimismo etiquetados con el valor c de corroboración, también entre paréntesis. En todos los ejemplos comentados en este capítulo se utilizó un valor de $c_{max} = 25$, correspondiente a aproximadamente 4 segundos de tiempo real[6].

En el momento inicial (figura 4.16(a)), las hipótesis de planos y señales están completamente corroboradas y presentan valores de c cercanos a la saturación, por el respaldo sensorial de la interpretación. Durante la obstrucción de la cámara con una hoja de papel (figura 4.16(b)) la interpretación se mantiene, incluso actualizándose a ciegas al girar la plataforma con la lectura de la odometría. Sin embargo, los valores de corroboración de los planos y las señales comienzan a disminuir, al no existir segmentos de imagen que los sustenten. Al retirar la hoja (figura 4.16(c)), la interpretación se reproyecta de nuevo de forma aproximadamente correcta, y se vuelven a incrementar los correspondientes valores de corroboración. La figura 4.16(d), finalmente, muestra la reconstrucción tridimensional de la escena mantenida durante el periodo de ceguera.

4.5.5. Generación de comportamiento

La trayectoria de navegación, como se ha argumentado a lo largo de todo el capítulo, debe estar determinada por la estructura local del entorno percibida. Para la instanciación práctica de esta idea la arquitectura propuesta emplea un esquema centralizado, en el que el movimiento está siempre guiado por el *punto de control* introducido en la sección 4.5.3, inde-

[6]La velocidad de procesamiento media en estos ejemplos fue de $6 - 7$ *fps*, por la configuración hardware concreta utilizada para las pruebas (ver capítulo siguiente).

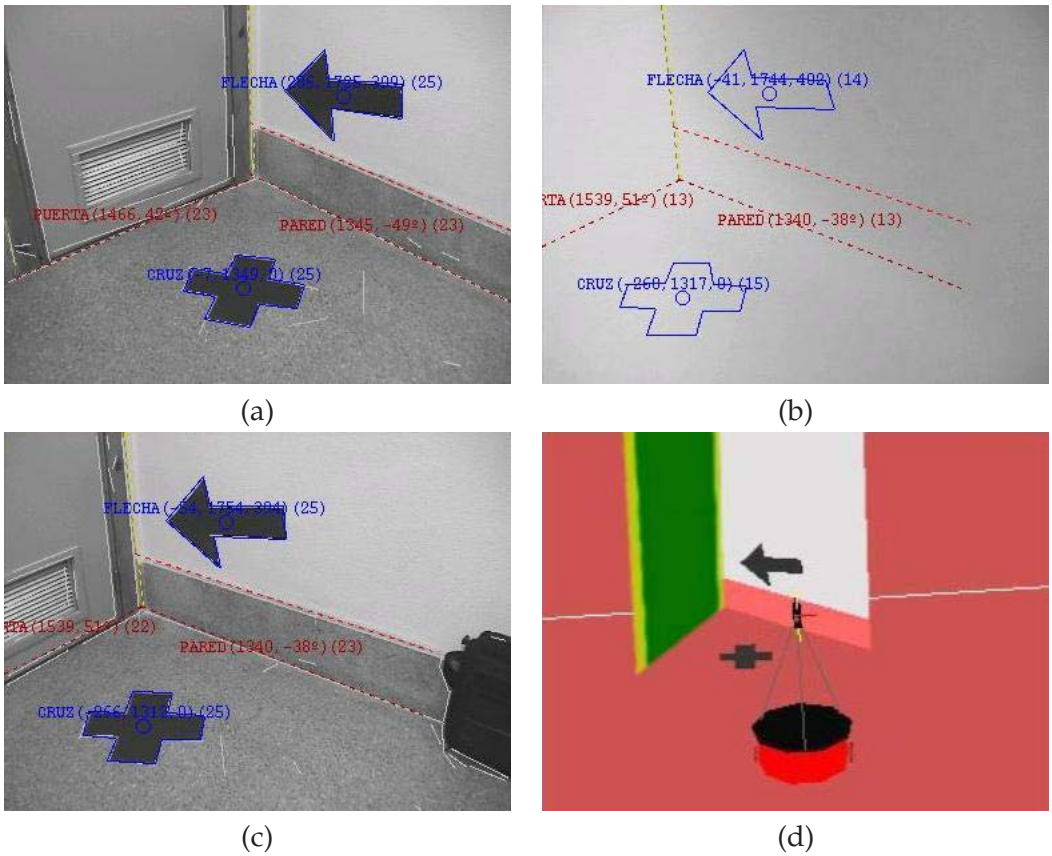


Figura 4.16: *Ejemplo de mantenimiento temporal de la consistencia en la interpretación: a) Hipótesis inicial, convenientemente corroborada. b) Hipótesis mantenida temporalmente con la cámara tapada, aún después de un movimiento odométrico (giro a la derecha de unos 10°). c) Recuperación de la interpretación. d) Representación tridimensional de la escena después del giro.*

pendientemente de si viene impuesto por un *schemap* o por una señal reconocida. Dado que a cada interpretación se le asocia una única acción deseada, no se hace necesario ningún tipo de arbitraje entre acciones de control contradictorias, al estilo de otras arquitecturas distribuidas basadas en el comportamiento. Hay que resaltar también que, al menos en principio, cualquier movimiento del robot puede resultar válido para mantener el mecanismo de corroboración y refutación de hipótesis. De esta forma, no tiene por qué existir contradicción alguna entre la exploración del entorno y el enfoque de visión activa, en el que el movimiento se efectúa fundamentalmente con la finalidad de ayudar a la percepción.

Para asegurar un movimiento suave en función de la posición del punto de control se propone un sencillo procedimiento de amortiguación de las órdenes transmitidas. El esquema es similar al empleado en trabajos anteriores en el ámbito del seguimiento de objetos con una cámara situada en una torreta controlable (López de Teruel y Ruiz, 1998). En aquel sistema se aplicaba una regla de actualización bayesiana, mediante la cual en cada instante se repartía la credibilidad sobre la posición del objeto a seguir entre la estimación actual y el valor medido. La técnica que aplicaremos aquí utiliza un *parámetro de ganancia* ω análogo a esta credibilidad,

ENTRADA:
- Coordenadas del punto de control en el instante actual, $(x_{ctrl}^{(t)}, y_{ctrl}^{(t)})$, en coordenadas de S_{rob} sobre el suelo (mm), y proporcionadas durante la operación del robot por la interpretación de la situación actual en términos del *schemap* y las señales percibidas.
- Velocidades rotacional y traslacional reales de la plataforma, $v_{rot,med}^{(t)}$ y $v_{trasl,med}^{(t)}$, medidas en el instante actual.
- Velocidades rotacional y traslacional máximas permitidas ($v_{rot,max}$ y $v_{trasl,max}$), dadas en grados/s y mm/s, respectivamente.
- Parámetro de ganancia ω, con $0 \leq \omega \leq 1$, para amortiguar las órdenes de control y suavizar el movimiento del robot.

SALIDA:
- Órdenes de control sobre el robot, en forma de velocidades rotacional y traslacional deseadas para el instante siguiente, $v_{rot,des}^{(t+1)}$ y $v_{trasl,des}^{(t+1)}$.

ALGORITMO:

Velocidad rotacional:

-Calcular del ángulo relativo entre la dirección de avance del robot y el punto de control, $\beta_{rel} = \arctan(\frac{y_{ctrl}^{(t)}}{x_{ctrl}^{(t)}}) - 90°$, expresándolo dentro del rango $[-180°; 180°)$.

/ La velocidad rotacional debe ser proporcional al ángulo β_{rel}, hasta un máximo de $\pm 90°$. Pero * * en lugar de solicitar directamente la velocidad deseada, se amortigua con la actual, usando el * * parámetro de ganancia ω: */*

- Hacer $v_{rot,des}^{(t+1)} = (1-\omega) v_{rot,med}^{(t)} + (\omega) \frac{\text{sign}(\beta_{rel}) \, \text{máx}\{|\beta_{rel}|, 90°\}}{90°} v_{rot,max}$.

Velocidad traslacional:

/ La velocidad traslacional, dado que se da sobre la dirección de avance Y_{rob}, también debe ser pro- * * porcional a la distancia al punto de control proyectada sobre este eje (es decir, y_{ctrl}), pero sólo si * * ésta es positiva, y saturada a $v_{trasl,max}$. También se amortigua usando ω. El parámetro d_f utili- * * zado en la siguiente expresión está definido como en las figuras 4.9 Y 4.12: */*

- Hacer $v_{trasl,des}^{(t+1)} = (1-\omega) v_{trasl,med}^{(t)} + (\omega) \frac{\text{máx}\{0, y_{ctrl}\}}{d_f} v_{trasl,max}$.

Algoritmo 4.6: *Procedimiento de navegación por seguimiento del punto de control.*

cuyo objetivo es modular un cambio progresivo de las velocidades rotacional y traslacional de la plataforma en su orientación hacia el objetivo. Estas velocidades, por otra parte, deberán ser proporcionales en cada instante a los respectivos valores de ángulo de desfase y distancia del robot en relación a dicho punto.

El algoritmo 4.6 resume el procedimiento para la generación de movimiento en el instante $(t+1)$ a partir de la posición actual del punto control $(x_{ctrl}^{(t)}, y_{ctrl}^{(t)})$, expresado en coordenadas relativas a S_{rob}. Son necesarios también los valores de las velocidades rotacional y traslacional reales medidas en el instante (t), así como los correspondientes valores máximos permitidos. La ganancia concreta ω conviene ajustarla por prueba y error. La idea es que durante la ejecución indefinida del procedimiento sobre la secuencia continua de valores $(x_{ctrl}^{(t)}, y_{ctrl}^{(t)})$ proporcionada por los módulos de percepción, el robot describa unas trayectorias suaves adaptadas localmente a la estructura de la escena. No nos preocuparemos de planificaciones a más largo plazo, por quedar fuera del alcance de esta tesis. Las simples reglas resumidas en las figuras

4.9 y 4.12, de todos modos, producen comportamientos de cierta variedad, aparentemente bien integrados en el entorno y que, en definitiva, muestran cierta "*comprensión predictiva*" en el sentido planteado en la introducción.

Esta última afirmación, sin embargo, sólo se puede contrastar estudiando periodos de operación largos, donde se ponga a prueba la adecuada integración de los componentes de la arquitectura. Pero para ello hay que describir primero cómo se organizará el hardware y el software utilizado en la implementación, un aspecto práctico de vital importancia para el correcto funcionamiento del sistema. En el capítulo siguiente se describirá esta organización y, dado que hasta ahora sólo se han mostrado una serie de ejemplos que ilustraban aisladamente algunas características de la arquitectura, podremos realizar también entonces un estudio más dinámico y global del comportamiento del prototipo implementado.

4.6. Resumen

Las principales aportaciones de este capítulo se pueden resumir en los siguientes puntos:

- Frente a enfoques no representacionales, se propone una navegación interpretativa, basada en la estructuración de la escena. Para ello se categoriza el entorno local del robot en términos de modelos adaptados al dominio, a los que denominamos *schemaps*, y que son configurables y parametrizables en función de las condiciones particulares de cada situación.

- Se propone un modelo planar para la representación interna, en el que la estructura de fondo de la escena queda capturada por un conjunto de planos en posiciones restringidas (horizontales y verticales). El modelo elegido es esencialmente escalable, puesto que, al menos en principio, no entra en conflicto con la presencia de otros objetos ajenos al mismo.

- Partiendo de esta representación, se presenta un procedimiento de reconstrucción tridimensional capaz de obtener un modelo virtual de la escena que, incluso texturizado con los píxeles de la imagen original, puede ser elaborado y mantenido en tiempo real.

- El mecanismo interpretativo se basa en un procedimiento de agrupación tentativa de primitivas en estructuras de orden superior, que son validadas o rechazadas en un ciclo de procesamiento posterior que evalúa la persistencia temporal de las construcciones realizadas. La propia acción y la información sensorial de movimiento toman parte activa en este ciclo, contribuyendo a la oportuna corroboración o refutación de las hipótesis consideradas.

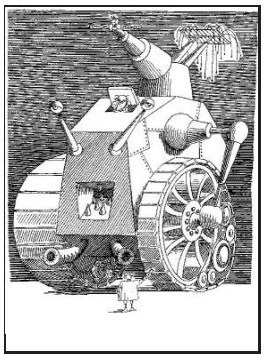
- Los procedimientos de generación de hipótesis se basan en heurísticas que emplean conocimiento del dominio para la localización inicial de los principales elementos de la

escena. Por razones de eficiencia y robustez, sin embargo, el peso de la percepción recae fundamentalmente sobre el seguimiento de las estructuras localizadas con anterioridad. Este seguimiento facilita al mismo tiempo la actualización de la representación, al mantenerse controlado el número de combinaciones de segmentos a considerar en la creación de nuevas estructuras.

- A pesar del sencillo procedimiento de control centralizado empleado, y de que no se utiliza ningún tipo de planificación a largo plazo, el comportamiento mostrado por el agente se adapta bien a las condiciones del entorno, gracias a la interpretación de alto nivel que, en última instancia, guía la navegación.

- En el capítulo anterior mostramos cómo la información de movimiento permitía la calibración euclídea y robocéntrica del sistema de visión. En éste hemos explotado aún más este tipo de información para desarrollar las habilidades anticipativas del agente en su actuación sobre el entorno. La odometría ocupa, por tanto, un lugar central en nuestra arquitectura de percepción.

CAPÍTULO V



"Cyberiad illustration", D. Mroz, 1967

Arquitectura hardware-software del sistema

An ingenious man who had built a flying-machine invited a great concourse of people to see it go up. At the appointed moment, everything being ready, he boarded the car and turned on the power. The machine immediately broke through the massive substructure upon which it was builded, and sank out of sight into the earth, the aeronaut springing out barely in time to save himself.
"Well," said he, "I have done enough to demonstrate the correctness of my details. The defects," he added, with a look at the ruined brick-work, "are merely basic and fundamental."
Upon this assurance the people came forward with subscriptions to build a second machine.

AMBROSE BIERCE, *Fantastic Fables - The flying machine.*

5.1. Introducción

Presentamos en este último capítulo el diseño de GeoBot, la plataforma experimental sobre la que se implementaron y probaron el conjunto de ideas y procedimientos propuestos en esta tesis. En primer lugar, enumeraremos los principales componentes hardware y software empleados en su desarrollo, para comentar posteriormente el modo en el que se organizan e interaccionan dentro de una arquitectura de percepción y control suficientemente modular y flexible. Finalmente, se realiza una descripción completa de un ejemplo de navegación visual del sistema en condiciones de trabajo realistas. A través de la observación directa tanto del comportamiento del agente como de su estado interno a través del interfaz de usuario, trataremos de evaluar las diversas funcionalidades basadas en las cualidades interpretativas

descritas en el capítulo anterior. Como se verá, el sistema desarrollado consigue cerrar un lazo de percepción y actuación estable durante periodos de operación relativamente largos, sin ningún tipo de intervención externa. El reconocimiento de las distintas situaciones de interés, aún en condiciones concretas no experimentadas con anterioridad, y la consiguiente generación de órdenes de control acordes con las pautas de comportamiento deseado, contribuyen a mostrar la viabilidad del enfoque interpretativo.

5.2. Componentes

Desarrollar un sistema autónomo de navegación visual exige la integración de multitud de componentes hardware (plataforma robótica, cámaras, dispositivos de cómputo, tarjetas de adquisición de vídeo, comunicaciones, etc.) y software (sistemas operativos, software de control del robot, bibliotecas de adquisición y procesamiento de imagen, interfaz de usuario, etc.). A continuación realizaremos una especificación detallada de los elementos que se utilizaron en el desarrollo de GeoBot. Es conveniente conocer de modo previo sus características individuales, ya que éstas influyen decisivamente en la organización global de la arquitectura (que cubriremos en la posterior sección 5.3).

5.2.1. Componentes hardware

Plataforma robótica

GeoBot es en realidad una adaptación del modelo Pioneer 2-DX, fabricado por ActivMedia Robotics (ver figura 5.1). Se trata de una versátil plataforma móvil autónoma, con dos ruedas motoras laterales y una tercera multidireccional (*rueda loca*), colocada en la parte trasera para lograr la estabilidad del conjunto. Las ruedas motoras están dotadas de *encoders* de precisión para la medición de los desplazamientos y giros efectuados por el robot. La alta concentración de estos sensores, con 9850 marcas por revolución en cada rueda (correspondientes a unas 19 marcas por mm, aproximadamente), proporciona un grado de precisión en la odometría más que suficiente para las tareas de autocalibración y anticipación en la percepción descritas en los capítulos 3 y 4, respectivamente. La plataforma dispone también de un conjunto de 8 sensores de ultrasonidos (cubriendo un rango frontal de 180°) para la detección de obstáculos cercanos (a distancias entre 10 y 500 cm), y que funcionan a una frecuencia de unos 25 Hz cada uno. El conjunto está gobernado por un microcontrolador Siemens 88C166 a 20 MHz, con 64 KB de memoria, organizados en 32 KB de *flash-ROM* para almacenar el microkernel del sistema operativo que controla todos los sensores y actuadores, y otros 32 KB de RAM dinámica para los datos de operación.

La carrocería tiene 33 cm de ancho por 44 de largo y 22 de altura, sin incluir la torreta con la cámara. Esta última va incorporada sobre un trípode de altura regulable, que permite situarla a una altura de entre 70 y 105 cm sobre el suelo, para ganar campo de visión. A pesar de lo

(a) (b)

Figura 5.1: *Imágenes de GeoBot, la plataforma móvil utilizada en los experimentos. Aunque hay dos cámaras sobre el trípode, nunca se utilizan simultáneamente (todo el procesamiento es monocular). Se muestra el sistema en dos modos de funcionamiento distintos (ver sección 5.4): (a) En modo* host *remoto: en la parte superior derecha se muestra el equipo de comunicación con el* host *(receptor de imagen UHF, en grande, y puente de Radio Ethernet, en pequeño; el primero se conecta directamente a la tarjeta de adquisición de la estación base, y el segundo a un punto de acceso a la red local). Sobre el propio robot puede observarse el pequeño emisor de UHF, con la antena desplegada. El dispositivo de Radio Ethernet queda oculto en el interior de la carcasa. (b) En modo procesamiento compartido autónomo: se monta un ordenador portátil a bordo, conectado con el PC del robot mediante un cable de par trenzado cruzado, y el conjunto es completamente independiente (no necesita conectarse a ningún dispositivo externo). El propio portátil se utiliza para monitorizar el funcionamiento durante la navegación.*

reducido de las dimensiones comentadas, que dotan al Pioneer de una gran maniobrabilidad incluso en espacios pequeños, la situación ligeramente descentrada de las ruedas permite un diámetro de giro mínimo sobre sí mismo de 52 cm. En cuanto a las características energéticas, estos modelos operan con tres baterías recargables de 12V que le dan una autonomía de funcionamiento de aproximadamente una hora, dependiendo de la cantidad de dispositivos conectados (cámara, PC, equipos de comunicaciones, etc.). El peso total del conjunto es de 9 Kg, aunque puede soportar hasta 20 Kg de carga útil adicional.

El robot alberga también un PC completo en el interior de su carrocería, provisto de un procesador Intel Pentium MMX a 266 MHz (modelo Tillamook, integrado en la placa madre para ahorrar espacio), 32 MB de memoria RAM, disco duro de 20 GB, y un bus de expansión con capacidad para hasta cinco tarjetas PC104+[1]. Dos de los cinco espacios disponibles vienen ocupados de fábrica con una tarjeta de adquisición de vídeo y otra de red Ethernet, que a su vez puede conectarse con otros equipos bien a través de un cable, o bien a través de un enlace de radio, contenido también en el interior de la carcasa del robot. La comunicación local entre el PC y el microcontrolador se realiza a través de un puerto serie RS-232. Una descripción más detallada de las características del Pioneer 2 y sus distintos componentes puede encontrarse

[1] El PC104+ es un estándar de bus compatible con el PCI, también diseñado para que las tarjetas de expansión ocupen el mínimo espacio posible

en el manual de descripción del hardware del robot, cuya referencia se proporciona en la bibliografía (ActivMedia Robotics, 1999).

Sensores de imagen

Las cámaras empleadas son los modelos Mitsubishi 300 y 300E, ambas de focal variable. Los dos sensores ópticos van montados sobre el trípode pero, puesto que estamos interesados en el procesamiento monocular, nunca se usan simultáneamente (se dispone de dos cámaras sólo para hacer diferentes pruebas). Normalmente, se opera con focal mínima fija para ganar campo de visión. Aún así, el enfoque autocalibrado permitiría la extensión natural a focal variable incluso durante el mismo periodo de operación, siempre que se volviese a aplicar cualquiera de los procedimientos de estimación propuestos en el capítulo 3 cada vez que se realizase un cambio en los intrínsecos[2]. Dicho valor mínimo es de aproximadamente 460 píxeles en la M-300E (para imágenes de tamaño PAL/2, de 288×384 píxeles), y ligeramente superior para la M-300. Una ventaja ofrecida por estas cámaras es que incluso para estos valores más pequeños de la focal (correspondientes a unos ángulos de visión de algo más de 40°), la distorsión radial es poco apreciable y, en la aplicación que nos ocupa, puede ser despreciada sin problemas.

Para la conexión a los equipos de cómputo, se dispone de dos tarjetas de adquisición distintas. La primera es una capturadora basada en el *chip* BT848, situada en el PC a bordo del robot para un modo de operación completamente autónomo. La segunda es una Matrox Meteor de mayor calidad de imagen, pero situada en un computador externo, para la recepción de la imagen analógica enviada desde el robot a través de un emisor UHF. Ambas tarjetas soportan el volcado directo a memoria a través del bus PCI de imágenes PAL a color de tamaño completo (576×768 píxeles), a una velocidad de hasta 30 *fps*. De todos modos, como se comentó anteriormente, normalmente se trabaja a media resolución como buen compromiso entre la calidad necesaria para las tareas de percepción y la velocidad de procesamiento para mantener una operación robusta en tiempo real.

Comunicaciones

La tarjeta de red del PC de GeoBot está conectada a un pequeño adaptador de Radio Ethernet situado en el interior de la carcasa, que a su vez permite enlazar sin cables con la red local con un ancho de banda máximo de 1 Mbit/s. La conexión se realiza a través de un puente que se conecta a cualquier punto de acceso de la red. Este puente se muestra en la esquina superior derecha de la figura 5.1(a) (el pequeño dispositivo con dos antenas de color gris).

[2]Como veremos en la sección 5.3.4, este proceso de autocalibración podría implementarse en un hilo de ejecución independiente (*thread*), que tendría prioridad máxima en el momento en que se produjesen dichos cambios.

El robot está también equipado con un emisor de vídeo portátil UHF, de bajo consumo, capaz de enviar las imágenes tomadas por la cámara (aún en formato analógico) a un receptor conectado a la tarjeta de adquisición de un computador externo, donde se puede realizar la digitalización y el posterior procesamiento. De este modo, si se desea, se pueden emplear dispositivos de computación adicionales para aumentar la eficiencia del sistema, descargando al PC del robot, que puede limitarse entonces a ejecutar las órdenes de control que recibe desde el exterior a través del enlace de Radio Ethernet. Tanto el emisor como el receptor se muestran también en la figura 5.1(a). El primero es la pequeña caja negra sobre el robot, y el segundo la caja blanca grande de la esquina superior derecha. Ambos son fácilmente distinguibles por estar dotados de una antena vertical metálica y extensible.

5.2.2. Componentes software

Sistemas operativos

En la implementación se utilizaron dos sistemas operativos diferentes. En primer lugar, el controlador del robot está gobernado por el P2OS (*Pioneer 2 Operating System*), un *microkernel* de tamaño reducido (menos de 32 KB), que está contenido en la flash-ROM programable situada junto al mismo. Este pequeño sistema operativo ofrece al programador el nivel de abstracción necesario para dar las órdenes de movimiento y leer los sensores de ultrasonidos y de posición de modo intuitivo, ocultando las complejidades del control de los motores de las ruedas, los *encoders* o las lecturas crudas de los sensores de ultrasonidos. El resto del software, ejecutado tanto en el PC de a bordo como en computadores externos, funciona sobre distintas versiones del sistema operativo Linux, en distribuciones RedHat para arquitecturas IA-32 de Intel.

Lectura de sensores y control del robot

La comunicación entre el microcontrolador y el PC se realiza mediante el puerto serie, a través del cual el programador envía órdenes y recibe información de estado del P2OS usando un protocolo diseñado por los fabricantes a tal efecto. El software ejecutándose sobre Linux puede realizar la lectura de los sensores y dar las órdenes a los actuadores a través de una biblioteca denominada PAI (*Pioneer Application Interface*), que libera al programador de la necesidad de tener que conocer las interioridades de dicho protocolo.

Monitorización

El Pioneer 2 viene equipado con un útil software de monitorización, denominado Saphira. Este software proporciona una potente interfaz gráfica sobre X-Windows, y permite la visualización de toda la información relevante del sistema en cada momento, desde los vectores de posición y velocidad actuales del robot hasta los datos proporcionados por los sensores

de ultrasonidos, pasando por los valores actuales de carga de las baterías, entre otros. Desde esta interfaz también pueden darse órdenes de movimiento directas al robot, manejándolo con los cursores y abortando cualquier tipo de acción de control solicitada por otro software de menor prioridad.

En realidad, Saphira es más que un mero entorno de monitorización y control manual del Pioneer. Se trata de una completa arquitectura de control para robots, que añade, incluso, un conjunto de rutinas de más alto nivel, tales como la detección de estructuras complejas en las nubes de puntos proporcionadas por los ultrasonidos, o la generación de comportamientos más o menos reactivos al entorno (evitación de obstáculos, navegación en paralelo a una pared, etc.) (Konolige *et al.*, 1997). De todos modos, nosotros no utilizaremos estas facilidades, por estar basadas en las lecturas de sensores distintos a los puramente visuales en los que se centra esta tesis. Nuestro interés en Saphira queda, pues, limitado a sus funcionalidades como interfaz informativa del estado interno del robot y control supervisado del mismo en situaciones de emergencia.

Bibliotecas de procesamiento de imagen

El grueso de las operaciones sobre las imágenes es llevado a cabo a través de la biblioteca OpenCV, un código de fuente abierta patrocinado por Intel que implementa una gran variedad de funciones para diversas aplicaciones de la visión por computador (Intel Corporation, 2000b). La biblioteca no sólo proporciona operaciones para el procesamiento elemental de las imágenes (filtros, transformadas, detección de bordes, esquinas, tratamiento del color, etc.), sino también funciones de más alto nivel (como la calibración de las cámaras, el seguimiento de objetos, el reconocimiento de contornos, etc.), y todo tipo de utilidades relacionadas (entrada/salida, almacenamiento, transformaciones geométricas, etc.).

La biblioteca es constantemente actualizada mediante contribuciones de la comunidad científica, y se apoya sobre un conjunto de rutinas de bajo nivel escritas por los propios desarrolladores de Intel, denominadas IPL (*Image Processing Library*) (Intel Corporation, 2000a). Estas rutinas están optimizadas para su ejecución en los distintos modelos de microprocesadores de este fabricante, mediante el uso eficiente de sus extensiones multimedia (MMX, *Multimedia Extensions*) y de procesamiento vectorizado (SSE, *Streaming SIMD Extensions*).

En la actualidad, las IPL se encuentran integradas dentro de las IPP (*Integrated Performance Primitives*), una biblioteca más general diseñada también para aprovechar al máximo las capacidades de las microarquitecturas de estos procesadores (Intel Corporation, 2002). Las IPP incluyen, además de las anteriormente comentadas rutinas de tratamiento de imagen, todo tipo de utilidades para el procesamiento multimedia, como el sonido, los vídeos y los gráficos, o el procesamiento matemático de grandes matrices de datos, entre otras.

Bibliotecas auxiliares

Además de los elementos básicos ya comentados, el sistema utiliza una serie de componentes software adicionales relacionados con aspectos secundarios, tales como el interfaz de usuario, el cálculo matricial, las comunicaciones, o la planificación de procesos. Así, el interfaz de usuario utiliza la biblioteca GTK (*Gimp Tool Kit*) para el manejo de ventanas sobre X-Windows y la visualización de las imágenes (GTK es la biblioteca utilizada por el conocido escritorio de Linux GNOME), y la biblioteca OpenGL para todo lo relacionado con la visualización de estructuras tridimensionales. Esta última tiene también la ventaja de estar optimizada para el uso de tarjetas gráficas aceleradoras, de modo que se reserva el tiempo de CPU para las tareas más críticas de percepción y control. Adicionalmente, dado que el sistema realiza una interpretación del entorno, el robot está dotado de la capacidad de verbalizar el estado interno en el que se encuentra, justificando su régimen de movimiento actual (por ejemplo, si está navegando a través de un pasillo, está doblando una esquina o está siendo guiado por una señal externa). Para esta parte sonora del interfaz de usuario se utilizó el software de generación de voz MBROLA, desarrollado también bajo licencia de uso libre por la Universidad de Mons-Hainaut, en Bélgica.

Las tareas de cálculo matricial asociadas a los distintos procedimientos de geometría visual se llevan a cabo utilizando las bibliotecas BLAS (*Basic Linear Algebra*) y LAPACK (*Linear Algebra Package*), en sus versiones para Linux sobre máquinas monoprocesadoras basadas en IA-32 (Dongarra *et al.*, 1988).

También resultan críticos algunos servicios básicos del sistema operativo, como los *threads*, *sockets* y temporizadores, entre otros. Como se verá a continuación, la robustez del sistema exige un diseño software con las distintas tareas de percepción y control ejecutándose como hilos de procesamiento independientes, punto donde entra la necesidad del uso de los *threads* y las distintas facilidades de comunicación entre hilos asociadas. Los *sockets* sirven para solucionar las tareas de comunicación con el computador *host*, o incluso, como también discutiremos, otros dispositivos de procesamiento externo adicionales. Finalmente, los temporizadores son también imprescindibles, dados los requerimientos de tiempo real de las tareas de percepción e interacción con el entorno.

Software de desarrollo propio

Por último, y como es lógico, tanto los algoritmos propuestos en esta tesis como el software global del sistema tuvieron que ser codificados expresamente. En total, los distintos procedimientos de extracción de segmentos, autocalibración, interpretación (generación, seguimiento y corroboración de hipótesis), interfaz de usuario, control del movimiento, etc., fueron programados en C, utilizando todas las bibliotecas comentadas y ocupando un total de unas 12000 líneas de código.

5.3. Organización

5.3.1. Arquitectura de procesos distribuida

Una vez conocidos en detalle los componentes utilizados, describiremos en este apartado la organización del software desarrollado, así como su relación con el hardware del agente autónomo. Veremos que el diseño de la arquitectura de procesamiento de GeoBot está principalmente guiado por un criterio de modularidad. Básicamente, esto significa que las distintas funcionalidades del sistema, tales como el interfaz, las lecturas de sensores, el control de los actuadores o el procesamiento de la imagen, por ejemplo, están implementados por procesos (o, en su caso, hilos de procesamiento) independientes. Éstos se comunican entre sí bien a través de zonas de memoria comunes, y sincronizados por un sencillo protocolo de exclusión mutua basado en semáforos, o bien a través de comunicaciones entre procesos a través de canales dedicados (en concreto, *sockets*). Esta arquitectura modular intenta mejorar la calidad del resultado final en dos direcciones:

1. Se consigue una mayor *robustez* en la navegación, a través de la asignación de distintas prioridades de ejecución a los módulos de procesamiento, dependiendo de su importancia para el correcto comportamiento del agente autónomo. Como ejemplos de procesos a los que se les debe asignar mayor prioridad tendríamos el encargado del control del movimiento, o el que lee los datos de los sensores de proximidad por ultrasonidos. Estas tareas son críticas para evitar el choque contra obstáculos inminentes no detectados por el sistema de percepción visual. Por esta razón se les debe asegurar un intervalo de CPU mínimo cada cierto tiempo, independientemente de que se estén ejecutando simultáneamente tareas mucho más intensivas en cómputo, como pueden ser todas las relacionadas con el procesamiento de la imagen. Un buen diseño modular facilita enormemente la satisfacción de este tipo de tareas con restricciones de tiempo real.

2. Una arquitectura modular también redunda en beneficio de la *flexibilidad* y *extensibilidad* del sistema. El software de GeoBot fue diseñado con la idea de encajar de modo natural ampliaciones en la funcionalidad, objeto de posibles líneas de investigación futuras. Por supuesto, la asignación de prioridades comentada anteriormente sigue siendo importante en este punto. Cada proceso añadido tendrá que ubicarse en el esquema de acuerdo con su importancia, de modo que los recursos computacionales queden correctamente repartidos y se asegure la coherencia y robustez del comportamiento global.

5.3.2. Arquitectura de datos centralizada

La figura 5.2 resume la organización del software de percepción y control de GeoBot. En ella mostramos cómo los distintos procesos en ejecución se sitúan alrededor de una estructura de datos central, en una zona de memoria compartida donde se almacena el estado global del

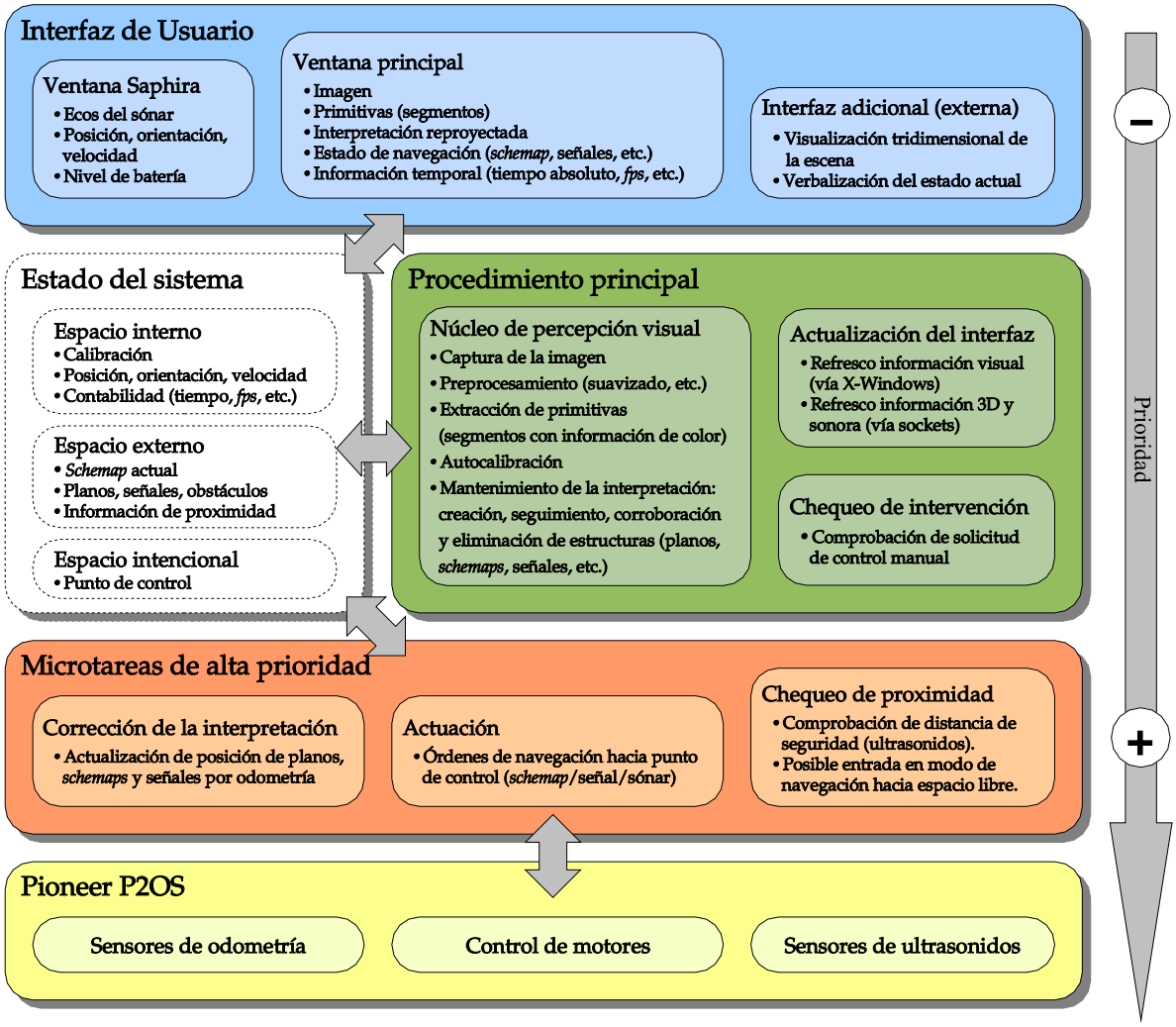


Figura 5.2: *Arquitectura software del sistema. En cajas coloreadas se muestran los distintos módulos de procesamiento de GeoBot, ordenados según su prioridad en la ejecución. En el nivel inferior (mayor prioridad) se encuentran las microtareas de percepción y control, en comunicación directa con los servicios del P2OS. El sistema Saphira asegura que dichas tareas se ejecutan al menos diez veces por segundo. El bucle de procesamiento de imagen principal se ejecuta también continuamente, pero con una prioridad más baja, a una velocidad directamente dependiente de la potencia de cómputo de la CPU principal. Finalmente, la actualización del interfaz es invocada con la frecuencia deseada desde el anterior, con el fin de modular la carga adicional de la CPU. El estado del sistema, almacenado en una zona de memoria compartida (caja blanca, de borde discontinuo) constituye el canal de comunicación entre los distintos procesos.*

sistema, y que actúa como canal de comunicación. Como se comentó anteriormente, el acceso concurrente a dicha zona se protege a través de un sencillo protocolo de exclusión mutua que garantiza la integridad de los datos, a pesar de que constantemente son consultados y actualizados por parte de distintos procesos que se ejecutan asíncronamente. Dentro de esta estructura cabe distinguir tres tipos de información almacenada:

- En primer lugar, toda la información relacionada con el estado interno del robot, donde

se almacenan los datos relacionados con su propia estructura y movimiento, independientes del exterior. Podría corresponderse con lo que algunos autores han llamado *espacio interno* del agente (Bustos, 1998). Caben aquí los datos relativos a la calibración de la cámara (altura, focal, ángulo de ataque, etc.), los resultantes de la lectura de los sensores propioceptivos (velocidad, posición y orientación calculados por odometría), el tiempo real de operación, y otros datos útiles para la monitorización del sistema, como la velocidad de procesamiento en *frames* por segundo, por ejemplo.

- En segundo lugar encontramos lo que, siguiendo con la terminología anterior, podríamos llamar *espacio externo* del agente, que contiene los datos relativos a la percepción del mundo exterior. Caben aquí desde los elementales valores de los ecos devueltos por los ultrasonidos hasta las más complejas interpretaciones de *schemap* actual, señales, planos, etc. En este sentido, podemos diferenciar entre lo que está dentro del rango efectivo de los sensores y lo que, a pesar de no ser ya directamente alcanzable, permanece en la memoria a largo plazo de lo ya percibido. Esto último podría encajar con lo que Bustos denomina *espacio extendido*. Nosotros incluiremos dentro de este bloque cualquier información, más o menos elaborada, que provenga de cualquiera de los sensores exteroceptivos.

- Finalmente encontramos el punto de control, dentro de un tercer componente del estado global al que denominaremos *espacio intencional*. Recordemos que, como se comentó en la sección 4.5.5, este punto denota la intención de movimiento que en ese momento muestra el agente, dependiendo de su actual percepción del mundo. En general, como vimos, en cada tipo de situación reconocida (esquinas, pasillos, espacio libre, distintos tipos de señales, etc.), la ubicación del punto de control seguirá una política determinada, encaminada a generar el comportamiento deseado en dicha situación.

5.3.3. Núcleo sensorimotor de bajo nivel

Situados alrededor de la estructura anterior, y utilizando ésta como vía de comunicación principal, los procesos del sistema se organizan en una jerarquía de prioridades. En la base de dicha jerarquía se encuentran las rutinas del sistema operativo del microcontrolador del robot, el anteriormente mencionado P2OS (en amarillo en la figura 5.2). Estas rutinas son las únicas que interactúan directamente con el hardware del robot, tanto para controlar los motores que determinan su movimiento como para leer los sensores de ultrasonidos y de odometría, que sirven para determinar la presencia cercana de obstáculos y el movimiento realizado por el robot. Estas rutinas se encuentran en la memoria del propio microcontrolador, y son ejecutadas por éste de modo independiente del resto del software.

En el siguiente nivel de la jerarquía, ejecutándose ya en el *host* principal (normalmente, el PC de a bordo, aunque puede haber configuraciones alternativas, sobre las que volveremos

en la sección 5.4), se encuentran las denominadas microtareas de alta prioridad, que se ejecutan como hilos de procesamiento a los cuales se asegura una frecuencia de ejecución mínima de 10 Hz (en color naranja en la figura). Estas microtareas están en contacto constante con el P2OS a través de llamadas a la biblioteca PAI, utilizando el puerto serie como vía de comunicación con el microcontrolador. En una de ellas, por ejemplo, el planificador de movimiento da las órdenes de actuación deseadas basándose en la interpretación actual del exterior. Otra microtarea prioritaria es la encargada de corregir toda la interpretación actual (planos, *schemap*, señales, etc.) en función de la información de movimiento proporcionada por los *encoders* de las ruedas. Este mecanismo de continua corrección a partir de la odometría, tal y como vimos en el capítulo anterior, resulta básico para el mantenimiento de la consistencia, y dota a GeoBot de la deseada capacidad de anticipación en el seguimiento de dichas estructuras.

Finalmente, se incluye también una microtarea de seguridad, que comprueba la presencia de obstáculos demasiado cercanos usando los sensores de proximidad basados en ultrasonidos. Esta microtarea protege al sistema de choques inminentes debidos a un posible fallo en la interpretación, por ejemplo. En caso de detectarse alguno de estos obstáculos, el sistema entra en modo de navegación hacia espacio libre, pasando a ser guiado única y exclusivamente por los sónares hasta que éstos vuelven a detectar suficiente espacio libre para la navegación. En realidad, éste es el único uso que se hace de los ultrasonidos en toda la arquitectura. Como se observa, su función se ha limitado a una tarea de seguridad añadida. Cabe decir en este punto que también podría haberse intentado una mayor integración entre estos sensores y la percepción visual. Un posible método, por ejemplo, sería utilizar los valores de los ultrasonidos como ayuda en la interpretación y el seguimiento de estructuras, teniendo en cuenta que debe haber una consistencia entre los planos detectados visualmente y la información de distancia devuelta por los sónares. Algunos autores han explorado estas posibles vías de integración sensorial (Coianiz y Aste, 1993). No obstante, nosotros hemos optado por no hacerlo, puesto que en esta tesis estábamos más interesados en el estudio de la percepción visual como única guía para la navegación.

En general, añadiremos en este segundo nivel todas las tareas para las cuales es importante asegurar que se ejecutan con una determinada frecuencia mínima. En nuestro caso, como hemos visto, son aquellas relacionadas con el control inmediato de movimiento, el chequeo de proximidad, la lectura de la odometría y la consecuente actualización de la interpretación con el objeto de predecir lo que se percibirá en el siguiente *frame*. Por razones tanto de seguridad como de robustez en el control, es imprescindible que estas tareas se ejecuten muy frecuentemente. De hecho, en ellas radica la responsabilidad de cerrar el lazo continuo que mantiene al agente en sintonía con el entorno en el que se mueve. Obviamente, debemos limitar en la medida de lo posible el procesamiento correspondiente, puesto que de otro modo se colapsaría el sistema por falta de recursos computacionales. Afortunadamente, las rutinas asociadas consisten en unas pocas operaciones en punto flotante o unas cuantas llamadas a la biblioteca PAI, lo que en la práctica consume muy pocos ciclos de cómputo útil (de ahí el

nombre de microtareas). De esta forma, a pesar de la alta frecuencia con la que son llamadas, se deja suficiente tiempo de CPU disponible para los procesos de percepción visual, significativamente más costosos, pero menos exigentes en cuanto a restricciones de tiempo real en su ejecución.

5.3.4. Núcleo de percepción visual

En el nivel central de la figura se sitúa el proceso principal, encargado de ejecutar el núcleo de percepción visual (cuadro verde de la figura 5.2). Dicho núcleo consiste en la captura de la imagen y el procesamiento de bajo y medio nivel, así como todas las tareas posteriores relacionadas con la calibración y la interpretación de alto nivel. En concreto, aquí se controla la creación de nuevas estructuras (generación de hipótesis), y su posterior seguimiento o eliminación (corroboración de hipótesis). Para ello, se comparan las primitivas extraídas de cada nuevo *frame* con el estado del sistema, donde se almacena la interpretación actual. De este cruce de datos surge el ciclo de creación, seguimiento y corroboración de estructuras propuesto en el capítulo 4, en cuya mezcla de flujos ascendentes y descendentes de información subyace la idea esencial de la arquitectura.

Desde el punto de vista perceptivo, éste es obviamente el proceso más importante de todo el esquema. Puesto que es el que trabaja con la imagen, es también el más costoso computacionalmente y el que, por tanto, más tiempo de CPU acaba consumiendo. Sin embargo, dadas las características de continuidad y anticipación en la percepción del mundo de GeoBot, no tiene tantas restricciones de tiempo real como las microtareas anteriormente mencionadas. Dependiendo de la capacidad de cómputo disponible , el sistema se comporta de modo robusto tanto en configuraciones lentas, donde la velocidad media de procesamiento de imágenes viene a ser de unos 4-5 *fps* , como en configuraciones más rápidas, donde los *frames* son procesados prácticamente a la misma velocidad con la que son capturados, a unos 25-30 *fps*. En el siguiente apartado comentaremos las distintas configuraciones hardware que han sido estudiadas.

Dentro del procedimiento principal, un sencillo módulo se encarga también de monitorizar el posible requerimiento de intervención manual por parte del operador. En caso de que esta solicitud se produzca, el agente puede dejar de comportarse de modo autónomo para pasar a ser controlado directamente por el usuario. En cualquier momento, el operador puede de nuevo reanudar la navegación autónoma.

Otro componente del programa principal se encarga de comunicar los cambios en el estado del sistema al siguiente nivel, de modo que éstos se reflejen adecuadamente en el interfaz de usuario. Al estar perfectamente monitorizada la velocidad de procesamiento actual, el sistema puede ajustar la frecuencia con la que se refresca dicho interfaz con el fin de no aumentar en exceso la carga computacional.

5.3.5. Interfaz de usuario

Llegamos, finalmente, a los módulos encargados de mostrar la información de estado de GeoBot, que forman el interfaz de usuario (cuadro azul de la figura 5.2). Dentro de éste cabe distinguir tres componentes:

- En primer lugar, la ventana proporcionada por Saphira, donde se visualiza la información elemental de los sensores de bajo nivel: ecos del sónar, posición, orientación, velocidad, nivel de batería, etc.

- En segundo, la ventana principal, donde se muestra toda la información relativa a la percepción visual. Es decir, la imagen original, los segmentos extraídos, la reproyección de la interpretación sobre la imagen, las señales, el *schemap* y el punto de control asociado a la situación actual. También se muestra en esta ventana cierta información temporal y de contabilidad, como el tiempo de navegación absoluto, la velocidad de refresco de los distintos módulos de procesamiento y el porcentaje del tiempo total que GeoBot estuvo navegando en modo íntegramente visual (sin tener que recurrir a los sensores de ultrasonidos).

- En tercer lugar, un interfaz adicional empleado para la visualización en tres dimensiones del entorno percibido por el robot (que utiliza la biblioteca OpenGL), y la emisión periódica de mensajes sonoros con los que GeoBot indica su estado de navegación actual (que hace uso del sintetizador de voz MBrola).

El interfaz se ejecuta normalmente en un PC externo desde donde se monitoriza la operación. Se aprovecha así la estructura cliente-servidor de la especificación X-Windows, que permite la ejecución separada del interfaz de usuario (ventanas, teclado, ratón, etc.) y el núcleo de la aplicación. La comunicación entre ambos se realiza a través de la red local Ethernet, mediante el protocolo TCP/IP. Si el interfaz gráfico consume mucho ancho de banda, esta comunicación entre cliente y servidor puede suponer un cuello de botella. Por esta razón, es a menudo interesante limitar los datos mostrados por la interfaz a la información ya procesada (segmentos, interpretación reproyectada, datos numéricos, etc.), con un formato más compacto que la imagen original. Esta misma razón nos llevó a separar del proceso principal aquellas partes del interfaz que no quedan bien resueltas utilizando el protocolo X-Windows. Estos otros módulos se encuentran en procesos aparte, comunicados con el proceso principal a través de protocolos específicamente diseñados para reducir el ancho de banda de red utilizado (y que también usan sockets TCP/IP). Como se observa en la figura 5.2, éste es el medio utilizado para la visualización tridimensional de la escena y la parte sonora del interfaz. Una vez más, todas estas consideraciones van encaminadas a maximizar el rendimiento de los núcleos de procesamiento sensorimotor y de percepción, descargándolos en la medida de lo posible de todas las tareas secundarias.

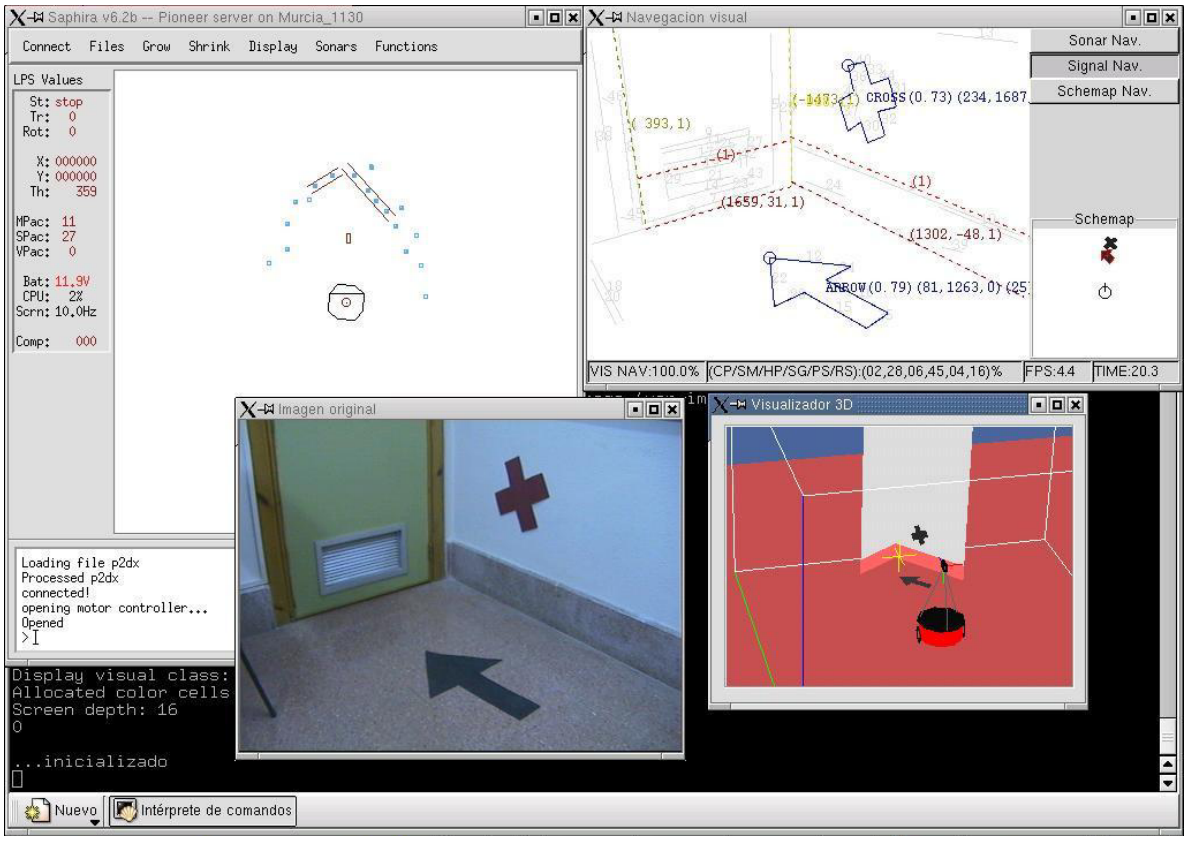


Figura 5.3: *Interfaz de usuario de GeoBot. De arriba a abajo, y de izquierda a derecha: ventana Saphira, ventana principal del módulo de percepción visual, imagen original y visualizador tridimensional de la escena.*

La figura 5.3 muestra un ejemplo completo de interfaz, tal y como se observa en el computador remoto utilizado para el seguimiento de la navegación. La ventana superior izquierda es la ventana Saphira, donde se muestra toda la información sensorial de bajo nivel, excluida la visual. A través de ella pueden monitorizarse las informaciones de posición medidas por la odometría, los niveles de batería, la frecuencia de ejecución de las microtareas de bajo nivel, la ventana de mensajes enviados por el P2OS, etc. También aparece un mapa esquemático del entorno del robot, tal y como es percibido a partir de las señales devueltas por los ultrasonidos. Interactuando con esta ventana el usuario también puede, si así lo desea, pasar a controlar manualmente la plataforma, inhibiendo temporalmente el comportamiento autónomo.

Las ventana inferior izquierda muestra la imagen RGB cruda actualmente captada por la cámara. Esta imagen puede ser enviada bien digitalmente a través de la red Ethernet, o bien por el canal UHF y digitalizada en la estación remota. Este otro modo tiene la ventaja de que se evita el cuello de botella que supondría saturar el ancho de banda de la red con el continuo flujo de imágenes. Alternativamente, puede enviarse la imagen digitalizada cada cierto periodo de tiempo, configurable por el usuario, o simplemente no mostrar esta imagen.

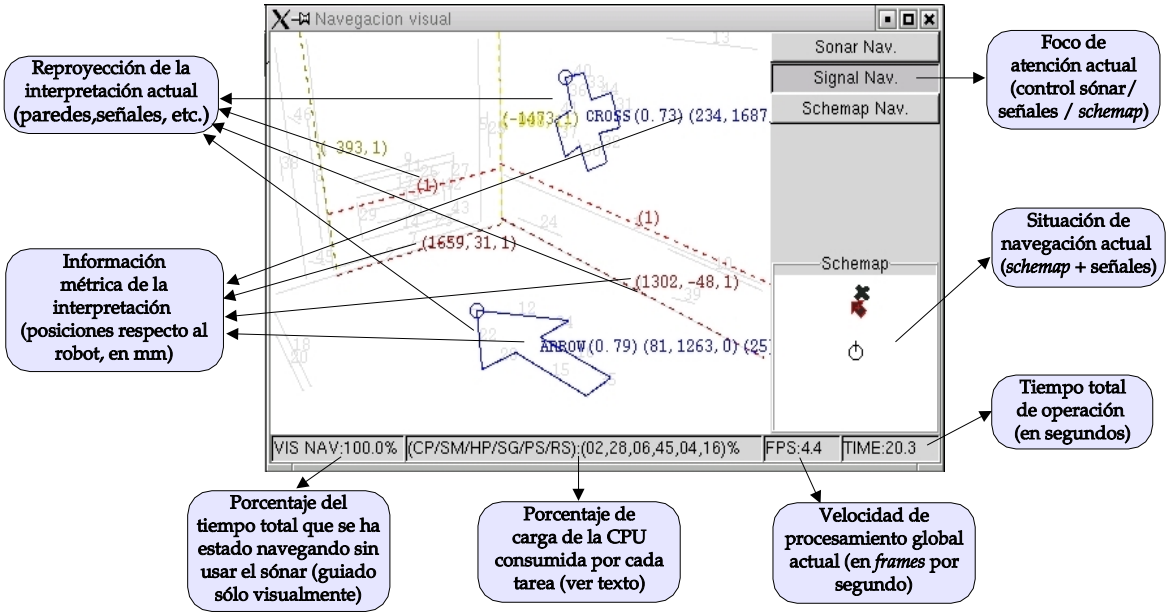


Figura 5.4: *Información mostrada en la ventana principal del módulo de percepción visual.*

Abajo a la derecha se muestra la ventana de visualización 3D, donde se resume esquemáticamente la percepción actual que GeoBot tiene del mundo exterior, de acuerdo con el modelo planar utilizado. Como ya se ha comentado, este visualizador se ejecuta en la estación remota, y se comunica con el módulo central a través de un protocolo TCP/IP dedicado con un bajo tráfico de datos para evitar el cuello de botella que supondría enviar la escena tridimensional ya generada.

Finalmente, por la gran cantidad de información que ofrece, y para una mejor interpretación de los ejemplos que mostraremos en la sección experimental, la ventana principal del interfaz merece comentario aparte. En la figura 5.4 se muestra una ampliación de esta ventana, con la explicación de las distintas informaciones que en ella aparecen. En el centro, ocupando la mayor parte, se encuentra la reproyección de la interpretación actual, sobreimpresionada encima de los segmentos extraídos del *frame* actual. En la imagen del ejemplo se aprecian los planos verticales correspondientes a la pared y a la puerta, así como las señales de flecha y cruz situadas en el suelo y la pared derecha, respectivamente, una vez que han sido correctamente interpretados y corroborados. A través de la hipótesis planar, y de la autocalibración practicada, la situación relativa de todos los elementos con respecto al robot es conocida euclídeamente, sin ningún tipo de ambigüedad proyectiva, afín o de escala.

El segundo bloque de interés aparece en la parte derecha, donde se muestra esquemáticamente el estado de navegación de GeoBot, que depende de su interpretación actual del entorno. Puesto que en este momento la prioridad en la navegación la tiene la flecha situada en el suelo, el punto de control se sitúa directamente sobre ella. Esta situación se corresponde con la marca roja sobre la flecha en el esquema inferior derecho, y el foco de atención actual

resaltado en la esquina superior. Esto provocará que GeoBot navegue hacia la flecha en primer lugar para, una vez situado a una distancia predeterminada de la misma, comenzar a girar en la dirección indicada, con la consiguiente actualización del punto de control. Como se verá en el completo ejemplo posterior, al detectarse situaciones previamente categorizadas, tales como pasillos, esquinas, paredes frontales, etc., y en ausencia de señales de mayor prioridad, el *schemap* detectado se muestra en esta parte del interfaz, así como el punto de control correspondiente que generará el movimiento deseado. Si por alguna razón (por ejemplo, un error en la interpretación visual), los sónares detectan un obstáculo demasiado cercano, GeoBot entra en modo de navegación hacia espacio libre guiado únicamente por estos sensores, comienza a emitir un pitido periódico y advierte de dicha situación, una vez más, en la esquina superior derecha de esta ventana. La navegación visual normal, por supuesto, se recupera en el momento en el que se evita el obstáculo.

En la barra de estado inferior, por último, se muestra alguna información adicional. Abajo a la izquierda se puede observar el porcentaje del tiempo total de operación que el robot ha estado navegando utilizando sólo la información visual (esto es, sin usar el sónar). El tiempo total de operación, en segundos, y la velocidad de procesamiento del bucle principal de procesamiento de imágenes, por su parte, se muestran abajo a la derecha. Finalmente, una distribución porcentual de la carga computacional consumida por las distintas tareas involucradas se muestra en el centro de la barra de estado. Esta información es de utilidad para detectar los posibles cuellos de botella del sistema. La explicación de las seis subtareas monitorizadas es la siguiente:

1. Captura (CP): Tiempo empleado por el manejador de dispositivo de la tarjeta de adquisición en obtener una nueva imagen. Suele ser muy bajo (del orden de un 1-2 % del total), gracias al uso del modo de funcionamiento asíncrono de dicho driver, en el cual la captura de la siguiente imagen se solapa con el procesamiento de la actual.

2. Suavizado de la imagen (SM): Este otro valor recoge el porcentaje de tiempo empleado en el suavizado de la imagen (habitualmente, un filtro de mediana). Dependiendo del nivel de ruido y de las características de las distintas texturas en la imagen de entrada, este filtrado previo puede ser necesario para el óptimo funcionamiento de las etapas de procesamiento posteriores. Los valores habituales de este porcentaje se hallan en un rango de entre el 15 % y el 25 % del tiempo total, según el tamaño de la máscara empleada.

3. Filtrado pasa-alta (HP): Es el tiempo consumido por la cómputo de la imagen de bordes, a partir de la cual trabaja el extractor de segmentos. Se trata de una etapa poco costosa en cómputo, que consume alrededor de un 5 % del tiempo total.

4. Extracción de segmentos (SG): Esta es la tarea más costosa del sistema, a pesar del cuidado que, como vimos en el segundo capítulo, se puso en su optimización. No en vano,

es aquí donde se concentra el grueso del procesamiento de bajo y medio nivel, reduciendo la imagen a primitivas perceptuales, en nuestro caso, segmentos con información de color. Esto justifica el consumo medio de CPU más alto, alrededor de un 50 % del total.

5. Interpretación, predicción y seguimiento (PS): La última etapa de tratamiento de la imagen consiste en la interpretación de la escena, correspondiente al ciclo de creación y mantenimiento de las hipótesis de estructuración del entorno descrito en el capítulo 4. A pesar de conformar el núcleo de la interpretación de alto nivel, el hecho de que trabaje con datos de tamaño significativamente reducido con respecto a la imagen original (los segmentos extraídos) hace que esta etapa sea bastante eficiente, con usos de CPU de un 5 % aproximadamente.

6. Resto de carga (RS): Se recoge aquí el resto de tareas ejecutadas en el computador principal. Incluye todo aquello que no se ha mencionado en los puntos anteriores, es decir, todas las microtareas de alta prioridad, el software de monitorización de Saphira, la comunicación con los módulos de interfaz de usuario, la sobrecarga del sistema operativo multitarea, etc. En general, todas estas tareas consumen el resto de tiempo restante, aproximadamente entre un 15 % y un 20 %.

La última parte del interfaz que comentaremos en este apartado es la correspondiente a la línea de comandos, a través de la cual el usuario puede configurar una gran cantidad de opciones en el momento de la puesta en marcha. Estas opciones van desde el tipo y la cantidad de información mostrada por el interfaz (visualización 3D, sonido, ventanas) hasta los parámetros de movimiento de la plataforma (velocidades máximas de traslación y giro, dentro de los rangos permitidos), etc. Todo ello, de nuevo, tiene el objeto de maximizar la flexibilidad del sistema y su adaptación a las condiciones del entorno y a las distintas configuraciones hardware sobre las que se ejecutará. Precisamente sobre este último aspecto profundizaremos en el siguiente apartado.

5.4. Modos de procesamiento

El diseño modular del sistema permite una cierta flexibilidad en la ubicación final de los componentes software. En este apartado describiremos brevemente cuatro posibles configuraciones de procesamiento, cada una con sus propias peculiaridades, y, por tanto, ventajas e inconvenientes.

Procesamiento autónomo

En esta configuración, quizá la más interesante desde el punto de vista de la autonomía, todo el software se ejecuta sobre el PC a bordo del robot. En este modo el sistema aún puede ser monitorizado desde un PC externo, vía X-windows, siempre que el ancho de banda

exigido por el interfaz gráfico no sobrepase la capacidad del canal de comunicación, de tan sólo 1 Mbit/s (impuesto por la Radio Ethernet). En general, si no es necesario mostrar la imagen original en el cliente, el resto de la información (segmentos extraídos, interpretación de la escena, estructuras tridimensionales, interfaz Saphira) puede monitorizarse con agilidad, a pesar del reducido ancho de banda.

Por supuesto, el sistema también puede lanzarse de modo local, sin monitorización externa. Esta configuración tiene una autonomía aún mayor, puesto que no existe la figura del *host*, y por tanto no hay que respetar ninguna distancia máxima de operación más allá de la cual se pueda perder la cobertura de la Radio Ethernet.

Como inconveniente principal, sin embargo, tenemos la reducida capacidad de cómputo del PC de GeoBot, que limita la velocidad de procesamiento a unos 3-5 *fps*, dado que sobre él recaen todas las tareas involucradas.

Procesamiento compartido

En esta otra configuración el procesamiento se reparte entre el computador de a bordo, que ejecuta las tareas de percepción de nivel inferior (reducción de las imágenes de entrada a segmentos con información de color), y un dispositivo de computación externo, donde se ejecutan el resto de componentes software (percepción de alto nivel, generación de órdenes de control, interfaz de usuario, etc.). El conjunto de segmentos extraídos es enviado a través de la Radio Ethernet, aprovechando la reducción en el ancho de banda necesario con respecto a la imagen original, como se comentó en el capítulo 2. Una vez en el computador que ejerce de *host*, todas las tareas de interpretación y generación de órdenes de control se ejecutan allí, descargando de este modo al limitado PC de la plataforma. Finalmente, se utiliza un sencillo protocolo de envío de órdenes de movimiento de vuelta al robot, de nuevo a través del enlace de radio.

La mayor ventaja de este otro esquema es el aumento global de prestaciones, ya que la carga computacional queda repartida entre ambos sistemas, y el cuello de botella del procesamiento (constituido por la extracción de segmentos) supone un retardo menor, puesto que dispone de una CPU casi íntegramente dedicada. En esta configuración la velocidad de procesamiento alcanza unos 5-7 *fps* en media. Una ventaja adicional es que la información que fluye entre el robot y el *host* (segmentos coloreados en un sentido, y órdenes de control en el otro) es digital, y viaja utilizando el protocolo TCP/IP. Esto significa que va protegida contra posibles errores en el enlace y, como consecuencia, los probables fallos de transmisión en el mismo no afectan a la información enviada (frente a las posibles interferencias derivadas de enviar la imagen vía UHF, opción que examinaremos en la siguiente configuración).

Como principal contrapartida, hay una cierta pérdida de autonomía con respecto a la configuración anterior. La razón es que se tiene que respetar una distancia máxima de operación con respecto al computador central. Esta distancia es de unos 25-30 metros como máximo en

presencia de obstáculos tales como paredes, muebles, etc., habituales en el entorno de operación del robot, y viene de nuevo impuesta por la pérdida de potencia de la señal inalámbrica del enlace de red.

Procesamiento externo

En un tercer modelo de procesamiento, que busca maximizar la velocidad de operación, el PC de a bordo ejecuta únicamente un "demonio" servidor de control y sensores. Este proceso se comunica con un dispositivo de cómputo externo al cual llega la imagen en formato analógico, vía un emisor de imagen portátil UHF de bajo consumo instalado en el robot (ver figura 5.1(a)). La digitalización de la imagen y todo el procesamiento posterior (extracción de características, interpretación, etc.) se realizan íntegramente en este otro computador. Las órdenes de movimiento son entonces enviadas de vuelta al robot, que las recibe a través del mismo demonio comentado anteriormente. En cada paso de operación, pues, este microproceso envía un paquete con la información de los ultrasonidos y la odometría, y recibe en otro posterior las órdenes de control generadas.

La velocidad de procesamiento queda aquí sólo limitada por las capacidades del dispositivo de cómputo externo. Como vimos en la comparativa de rendimientos de la figura 2.12, se pueden alcanzar sin dificultad los 30 *fps* del estándar PAL utilizando un simple PC de sobremesa equipado con un procesador Pentium IV, por ejemplo. El limitado ancho de banda del enlace radio no supone ningún cuello de botella, puesto que en realidad la información enviada y recibida en cada paso por el protocolo se reduce a unos pocos bytes para codificar las distintas medidas de los sensores odométricos y de ultrasonidos en los paquetes de ida y las órdenes de movimiento del robot en los de vuelta.

Sin embargo, a la pérdida de autonomía por el alcance del enlace radio hay que unir también en este caso el inconveniente de las posibles interferencias en el envío analógico de la imagen, derivadas de la escasa potencia del emisor UHF y el posible ruido ambiental. Así, este modelo será útil únicamente en situaciones en que las posibles pérdidas de calidad en la imagen recibida sean lo suficientemente despreciables como para no afectar al posterior procesamiento.

Modelo híbrido

Una variante interesante de los modelos anteriores consiste en la utilización de un portátil como computador *host*, situado físicamente en la misma plataforma de GeoBot y comunicado localmente a través de un enlace cableado, de ancho de banda diez veces mayor que el de radio (ver figura 5.1(b)). De este modo se tienen las ventajas del aumento de prestaciones y sencillez de monitorización del procesamiento compartido, sin los inconvenientes de la disminución de autonomía y la pérdida de calidad de la imagen en el enlace vía UHF. Dependiendo de si el portátil dispone o no de tarjeta de adquisición de vídeo, puede utilizarse el

segundo o el tercer modelo de cómputo comentados. En ambos casos el conjunto dispone de completa autonomía de movimiento al no depender de ningún computador central.

5.5. Resultados experimentales

Cerramos el capítulo mostrando un completo experimento de navegación visual realizado con GeoBot en un amplio entorno del interior de un edificio. El ejemplo escogido es bastante ilustrativo, puesto que recoge una gran variedad de circunstancias y cubre con amplitud las distintas situaciones que el sistema es capaz de interpretar. Como se verá, el alto grado de autonomía logrado por el agente en la navegación interpretativa valida en gran medida la arquitectura de percepción propuesta, mostrando su viabilidad en este tipo de entornos. Puesto que el estudio experimental detallado de los distintos módulos de procesamiento (calibración, extracción de primitivas, ciclo interpretativo de alto nivel, etc.) ha quedado cubierto en capítulos anteriores, en este punto nos centraremos en el estudio dinámico del sistema global en funcionamiento. Este estudio incluye, por tanto, la correcta interpretación de las distintas estructuras, el mantenimiento de la consistencia en la percepción del entorno y las consecuencias en el comportamiento del agente.

Comentaremos en profundidad un recorrido largo, de unos tres minutos de duración, en el cual, por la variedad de las situaciones que atraviesa el robot, se ilustran buena parte las habilidades del sistema desarrollado. Las figuras 5.5 y 5.6 muestran, respectivamente, la ventana principal del interfaz y diversas vistas externas del robot durante el periodo de navegación estudiado. Como se observa en las fotografías de la figura 5.6, la configuración escogida para el experimento es la última de las comentadas en el apartado anterior: el núcleo central de procesamiento se ejecuta íntegramente en el PC de GeoBot, pero se utiliza un ordenador portátil instalado a bordo y conectado mediante un cable de par trenzado cruzado local para realizar todas las tareas de interfaz. Este modo de trabajo asegura un grado máximo de autonomía, al tiempo que nos permite una monitorización detallada de la operación a través de la pantalla del portátil.

El recorrido realizado durante el experimento cubre un área aproximada de 80 m^2, sobre los cuales GeoBot navega describiendo una trayectoria de algo más de 30 m de longitud, realizando distintos giros, movimientos y paradas según un comportamiento acorde con las distintas situaciones presentadas. La velocidad media del recorrido, por tanto, ronda en torno a los 20 cm/s, si bien hay que tener en cuenta que el robot se detiene en algunos puntos obedeciendo a las órdenes de ciertas señales percibidas.

En la trayectoria descrita en este ejemplo concreto, el robot parte de una habitación de la que sale guiado por una flecha en la pared, para localizar después un pasillo y comenzar a navegar centrado en el mismo. En un momento determinado, encuentra una cruz en la pared y se para unos instantes a su altura. Posteriormente, detecta que el pasillo termina y gira a la derecha, tomando otro pasillo y navegando a través de él hasta que una flecha en el suelo da la

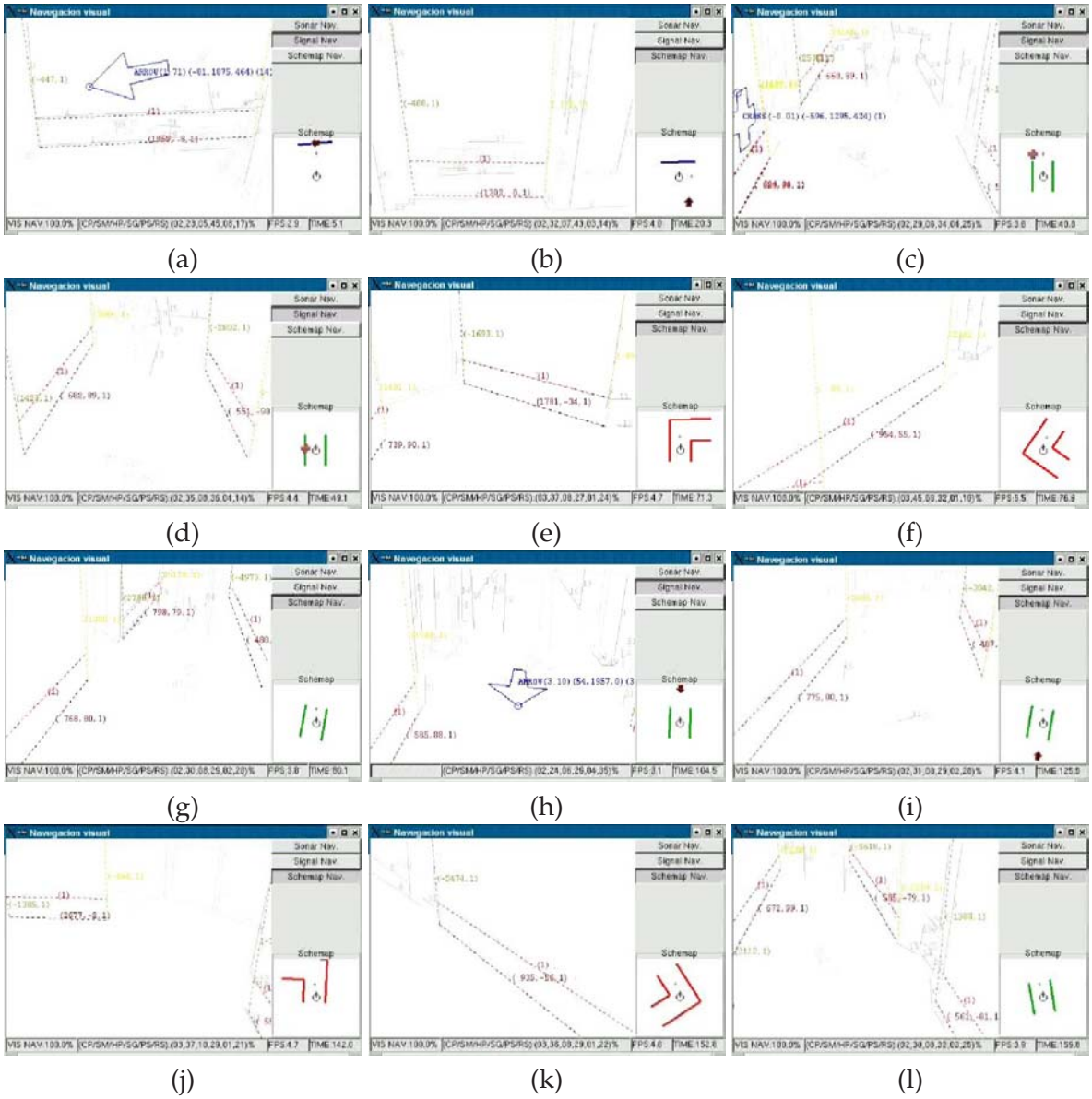


Figura 5.5: *Ventana principal del interfaz durante un recorrido autónomo de tres minutos de duración, guiado únicamente a partir de la información visual. La explicación de las distintas informaciones mostradas se encuentra en la figura 5.4. De especial interés resulta la descripción esquemática situada en la esquina inferior derecha de cada ventana, y la reproyección de la interpretación sobre los segmentos extraídos.*

orden de girar 180°. En ese momento, GeoBot da media vuelta y deshace el camino andado, tomando la esquina entre pasillos esta vez hacia la izquierda, y terminando el recorrido en el pasillo de partida, a petición del operador. Como se observa en algunas imágenes de las figuras 5.5 y 5.6, el mecanismo validación de hipótesis por consistencia temporal descrito en el capítulo 4 dota al sistema de una gran robustez ante obstáculos móviles, como personas caminando, por ejemplo.

(a) (b) (c)

Figura 5.6: *Vistas externas del robot durante la secuencia de navegación anterior: (a) Atravesando una puerta guiado por la flecha de la pared, antes de girar al detectar la puerta de enfrente. (b) En la esquina con giro a la derecha. (c) Navegando a lo largo de un pasillo. Obsérvese cómo la presencia momentánea de obstáculos móviles (en este caso una persona) no afecta a la navegación, por el mecanismo de mantenimiento temporal de la consistencia en la interpretación.*

El cuadro 5.1 recoge una descripción detallada de la secuencia de navegación. Cada fila se refiere a un momento concreto de la misma, y a su vez se corresponde con una imagen de la figura 5.5. La información mostrada en estas imágenes se puede interpretar en los términos descritos en la figura 5.4 y la sección 5.3.5. Las dos primeras columnas indican el número de la figura y el instante de navegación, respectivamente, mientras que la tercera recoge una breve descripción de la situación correspondiente. Es interesante observar en cada caso la interpretación esquemática que GeoBot realiza del contexto en el que se encuentra. En cada ventana, abajo a la derecha, se pueden apreciar los *schemaps* detectados en distintos colores (según se trate de un pasillo, una esquina o una pared frontal), las señales que implican órdenes de navegación (flechas y cruces en el ejemplo, mostradas en negro) y la posición actual del punto de control, que guía la navegación de acuerdo con la interpretación actual.

Es importante notar que en todo momento la trayectoria seguida por el robot se adapta suavemente a la estructura de la situación observada. La utilización de un esquema de control reactivo difícilmente hubiese podido realizar giros suaves de este estilo en las esquinas, por ejemplo. Obsérvese también que el grado de autonomía alcanzado durante el experimento es muy elevado. En la esquina inferior izquierda de cada ventana se puede apreciar que el porcentaje de tiempo empleado en la navegación visual es del 100 %, es decir, que la práctica totalidad del recorrido se realiza sin apoyarse en ningún otro sensor exteroceptivo distinto a la cámara. En ocasiones, no obstante, algunos fallos en el mecanismo interpretativo pueden provocar breves periodos de navegación guiada por el sónar, disminuyendo ligeramente este valor.

Resulta asimismo interesante estudiar la eficiencia de la arquitectura. A pesar de la limitada potencia de cómputo del Pentium MMX a 266 MHz del PC de a bordo empleado en la ejecución del procedimiento principal, el sistema es capaz de trabajar a una velocidad global en torno a los 4-5 *fps*. La ligera variabilidad en el tiempo de ejecución observada en los distintos *frames* es debida a la desigual complejidad ofrecida por las imágenes de entrada, en

Figura	Tiempo	Comentario
5.5(a)	5.1 s	GeoBot comienza su operación en una habitación de la que saldrá guiado por una flecha en la pared. En este momento, dicha señal gobierna la navegación, tal y como marca el indicador de foco de atención. Puesto que la flecha está en la pared, el punto de control se sitúa delante de ésta, a una distancia prefijada de 50 cm.
5.5(b)	20.3 s	Una vez realizado el giro a la izquierda, el módulo de percepción detecta un plano vertical obstaculizando el avance. En ese momento el *schemap* de pared frontal, mostrado en azul en la situación de navegación actual abajo a la derecha, pasa a gobernar la navegación. Consecuentemente, el punto de control se coloca a un lado del robot para evitar el obstáculo.
5.5(c)	40.8 s	Al realizar el giro, GeoBot reconoce una situación de pasillo, que se muestra en verde en el recuadro correspondiente. Sin embargo, también ha localizado una cruz en su pared izquierda, por lo que coloca el punto de control delante de la misma, a 50 cm, y comienza a dirigirse hacia ella.
5.5(d)	49.1 s	Se alcanza la altura de la cruz en el pasillo, y la navegación se detiene allí por unos instantes.
5.5(e)	71.3 s	Tras reanudar la trayectoria, la presencia de un plano frontal saliendo del pasillo indica la existencia de un giro en el mismo, en este caso a la derecha. El control pasa a este nuevo *schemap*, mostrado en rojo. A pesar de haber detectado una esquina, hasta que el robot no la sobrepasa el punto de control sigue apuntando hacia adelante.
5.5(f)	76.9 s	Esta figura marca el momento en que GeoBot supera la esquina y el punto de control se sitúa a la derecha, provocando el giro.
5.5(g)	80.1 s	Una vez tomada la curva, se entra de nuevo en un pasillo, otra vez indicado por el color verde del *schemap* correspondiente.
5.5(h)	104.5 s	Tras avanzar unos metros, una flecha situada en el suelo indica a GeoBot que deberá volver sobre sus pasos una vez alcanzada su altura.
5.5(i)	125.0 s	Segundos más tarde ya ha girado sobre sí mismo y recupera la línea central del pasillo, disponiéndose a recorrerlo ahora en sentido contrario.
5.5(j)	142.0 s	Al volver sobre sus pasos entra de nuevo en la esquina, pero en este caso deberá realizar el giro a la izquierda. La situación se indica con un nuevo *schemap* en rojo.
5.5(k)	152.8 s	Momento del giro, saliendo de la esquina anterior por el cambio de situación del punto de control.
5.5(l)	159.8 s	Por último, a la salida del giro vuelve a recuperarse el primer pasillo. En este momento el operario marca el final de la navegación.

Tabla 5.1: *Explicación de la secuencia de navegación mostrada en la figura 5.5. Cada fila se refiere a una figura, indicada en la primera columna. La segunda columna muestra el momento de la navegación en el que se produjo la situación comentada, en tiempo real medido en segundos desde el comienzo de la operación. La tercera columna explica con un comentario el comportamiento observado en el agente.*

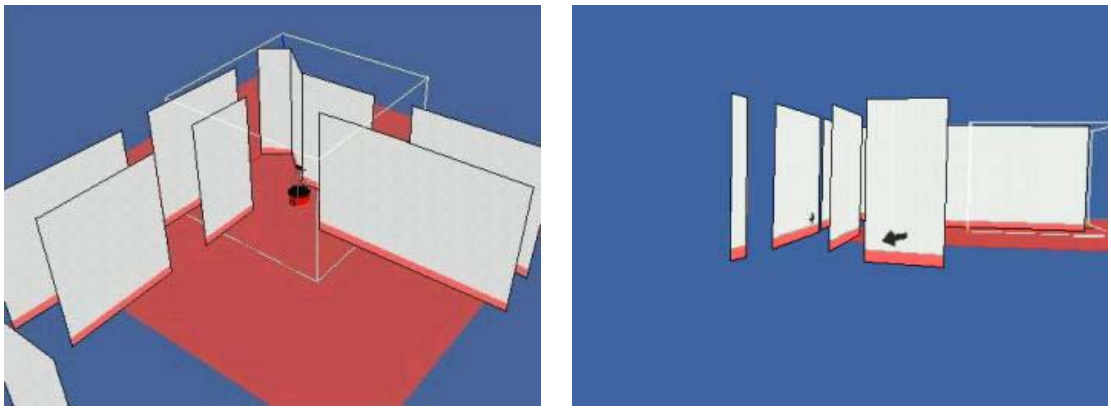


Figura 5.7: *Dos vistas de la reconstrucción tridimensional del entorno practicada en tiempo real por GeoBot durante el periodo de navegación indicado.*

términos de la longitud y cantidad de los segmentos extraídos. En la barra de estado inferior de cada ventana puede también apreciarse cómo la distribución del tiempo procesamiento en cada *frame* se mueve en torno a los porcentajes comentados en el apartado anterior.

En la figura 5.7, por último, se muestran dos vistas distintas de la reconstrucción tridimensional del entorno realizada por GeoBot durante la navegación. En dichas vistas se aprecian los distintos planos verticales de los pasillos y habitaciones del área recorrida, correspondientes a tramos de pared, y los espacios entre dichos planos, correspondientes a las puertas. En la vista de la derecha se observa también el detalle de las señales en las paredes, una flecha en la habitación inicial y una cruz en el pasillo del fondo. Hay que insistir en que esta reconstrucción se va realizando de modo incremental en tiempo real durante el mismo periodo de navegación, y que, como se ha comentado anteriormente, carece de todo tipo de ambigüedad proyectiva, afín o ni siquiera de escala. Todos los elementos de la reconstrucción, por tanto, están medidos en coordenadas reales, relativas a la posición en cada momento del robot. Esta reconstrucción, pues, puede hacer el papel de mapa euclídeo-topológico con las características comentadas en el capítulo anterior, dado que, además de la reconstrucción euclídea, podría recuperarse con sencillez un grafo topológico con las distintas interpretaciones esquemáticas correspondientes a cada posición, si así se desease para algún tipo de representación a más largo plazo.

En relación a esto último, hay que aclarar que la implementación actual carece de mecanismos de recuperación sobre este mapa al estilo del espacio extendido mencionado en la sección 5.3.2. GeoBot utiliza el seguimiento temporal como mecanismo de apoyo a la consistencia en cada momento, mientras las estructuras seguidas se encuentran al alcance de la vista. El resto del entorno que, habiendo sido percibido anteriormente, ha quedado ya fuera del rango de los sensores, se mantiene aún a través de la actualización odométrica, puesto que sigue siendo importante en la navegación. Piénsese, por ejemplo, en la parada encima de una señal, que obviamente deja de percibirse instantes antes por caer fuera del alcance de la

cámara. Algo análogo ocurre en las esquinas, donde las paredes del pasillo de entrada dejan de verse durante el mismo giro. Pero si, como en la trayectoria de ejemplo, un mismo lugar es recorrido en distintos instantes de tiempo (por ejemplo, al volver sobre sus propios pasos), el sistema "borra" la zona del mapa asociada elaborando una interpretación completamente nueva. La consistencia a largo plazo constituye un problema de gran interés en sí mismo, estudiado en numerosas ocasiones por la comunidad científica (ver, por ejemplo, los trabajos de Gutmann y Konolige (2000) o Unnikrishnan y Kelly (2002)). Su estudio, de todos modos, quedaba fuera del alcance de nuestros objetivos iniciales, aunque se propondrá en el apartado correspondiente como futura línea de investigación. Lo mismo cabe decir de la implementación de técnicas de planificación del control basadas en comportamientos más complejos, generados posiblemente a partir de interpretaciones del entorno a mayor escala.

Naturalmente, la mejor manera de comprobar el funcionamiento autónomo del agente es a través de la observación directa de su comportamiento. Desde la página *web* del proyecto de investigación *'Percepción Estructural en Tiempo Real'* en el que se ha apoyado esta investigación[3], se puede acceder a una serie de vídeos de operación donde se ilustra el comportamiento de GeoBot en diversas condiciones de funcionamiento (Ruiz y López de Teruel, 2003).

5.6. Resumen

Finalizamos el capítulo, una vez más, con un resumen de las principales cuestiones tratadas en el mismo:

- Se explica en detalle la organización global del prototipo desarrollado. En concreto, se describen todos los componentes hardware y software utilizados, así como su integración en un esquema de procesamiento distribuido y priorizado que asegura el correcto reparto de recursos entre los módulos perceptivos y los mecanismos de actuación.
- Se propone un variado repertorio de modos de procesamiento para adaptar la arquitectura a distintas configuraciones del hardware empleado, especificando la distribución oportuna de la carga computacional en cada caso.
- El sistema diseñado dispone de un completo interfaz de usuario, a través del cual se puede monitorizar y evaluar convenientemente el comportamiento del robot durante su funcionamiento.
- Finalmente, se describe un ejemplo real de navegación del agente en un periodo largo de operación, que ilustra su comportamiento dinámico y que, en definitiva, demuestra la viabilidad de la arquitectura de acción y percepción propuesta.

[3]Financiado por la CICYT con el código TIC98-0559.

Conclusiones y perspectivas

In discussing the sense of sight, we have to realize that (outside of a gallery of modern art!) one does not see random spots of color or spots of light. When we look at an object we see a man *or a* thing*; in other words, the brain interprets what we see. How it does that, no one knows, and it does it, of course, at a very high level.*

R. FEYNMAN. *The Feynman Lectures on Physics.*

Concluimos recapitulando las principales aportaciones de esta tesis y proponiendo una serie de posibles vías futuras de investigación. El objetivo último de este trabajo era construir un prototipo operativo de agente autónomo capaz de interactuar de modo inteligente con su entorno, utilizando el *criterio predictivo* como guía principal en el diseño. Creemos que este objetivo ha quedado cubierto en gran medida gracias a la integración de una serie de procedimientos y técnicas en todos los niveles de la percepción. Más concretamente, como propuestas fundamentales destacaremos las siguientes:

- Se ha desarrollado una técnica eficiente de extracción de características, basada en una potente primitiva de medio nivel, los **segmentos con información de color**. El método es capaz de reducir en dos órdenes de magnitud el tamaño de la imagen original, capturando su estructura geométrico-cromática esencial en tiempo real. La información extraída resulta de gran utilidad en la calibración del sensor óptico (a través del seguimiento interimagen de los segmentos y sus puntos de intersección) y la extrapolación de estructura a partir de la interpretación (utilizando el color y la disposición espacial de los segmentos, así como un cierto conocimiento *innato* del dominio).

- Hemos presentado un repertorio de métodos de **autocalibración a partir de odometría**, adecuados para trabajar sobre el propio entorno (sin necesidad de patrón) y con requisitos mínimos de datos y de procesamiento.

- Se ha propuesto un paradigma de percepción del espacio capaz de realizar **reconstrucciones 3D a partir de la interpretación**. El paradigma está basado en un **modelo planar**

adaptado al dominio de aplicación, los entornos estructurados del interior de edificios. A través de la categorización de "situaciones tipo", y el reconocimiento de formas y objetos gracias a su adecuada contextualización, se logra generar un **comportamiento no reactivo** en el agente autónomo. La interpretación de alto nivel de la escena, unida al uso de los sensores propioceptivos de movimiento, permiten al robot mostrar la deseada **comprensión predictiva** sobre el entorno. Esta capacidad de anticipación se hace posible gracias al uso de estructuras conceptuales adaptadas al dominio, que permiten el procesamiento en tiempo real de la gran cantidad de información sensorial tratada.

- Finalmente, los componentes hardware y software empleados en el desarrollo del prototipo se integraron en una **arquitectura global modular**, diseñada para ser fácilmente extensible con funcionalidades añadidas. El alto grado de autonomía mostrado por el robot móvil en experimentos realizados en condiciones realistas nos permite ser optimistas sobre la aplicabilidad a mayor escala de la propuesta.

En definitiva, creemos que se han alcanzado una buena parte de los objetivos planteados en la introducción. No obstante, la relativa inmadurez del campo de investigación en percepción artificial hace que, hoy por hoy, todo estudio relacionado deje forzosamente numerosas lagunas. El nuestro, lógicamente, muestra también muchas limitaciones. Resumimos brevemente algunas de las más notables:

- En primer lugar, la dificultad de aplicación de la propuesta a otro tipo de dominio. Todas las soluciones adoptadas están fuertemente influenciadas por éste, lo que puede complicar su adaptación a otros escenarios, como los exteriores, por ejemplo.

- En segundo, la degradación del comportamiento en entornos con muchos objetos ajenos al modelo planar. El sistema se defiende relativamente bien frente a cantidades moderadas de obstáculos, pero en presencia de gran cantidad de ellos la detección de planos se vuelve más complicada. La generación de hipótesis tendría que adoptar en estos casos heurísticas más elaboradas y complejas.

- En tercer lugar, la interpretación estática del entorno. La propuesta inicial era que la navegación debía estar guiada por la estructura de fondo de la escena. El sistema, por tanto, se diseñó para ser robusto ante obstáculos móviles, pero no para evitarlos o interactuar con ellos explícitamente.

A pesar del esfuerzo empleado por los investigadores en percepción artificial en los últimos años, lo cierto es que se está aún muy lejos de llegar a algún tipo de consenso, no ya en cuanto a las soluciones ofrecidas, sino incluso en la clara definición del problema planteado. Hoy por hoy, muchos de los trabajos relacionados se centran sólo en determinados aspectos, dejando a un lado otras cuestiones igualmente importantes. De ahí que, desde nuestro

punto de vista, toda aproximación deba tener una vocación fundamentalmente escalable, que no sólo permita sino que incluso estimule la experimentación con posibles alternativas. He aquí algunas de las posibles líneas de trabajo que consideramos dignas de exploración en futuras investigaciones:

- Enlazando con las limitaciones anteriormente expuestas, sería interesante ampliar las estructuras de interpretación a varios niveles. En las etapas inferiores, por ejemplo, podría pensarse en extender la primitiva de segmentación a contornos curvos, conservando la idea de compresión de color. A más alto nivel, podría dotarse al sistema de un mayor conocimiento innato, aumentando el repertorio de *schemaps* con situaciones más complejas, como habitaciones de varios tipos, puertas, etc.

- Un segundo objetivo sería la interacción con objetos en movimiento (personas, etc.), separándolos de forma adecuada del fondo de la escena, incorporando mecanismos de atención y tratando con mayor flexibilidad las oclusiones. En este sentido, puede ser interesante también experimentar con la inclusión en la arquitectura de paradigmas de interpretación distintos para ciertas tareas concretas (por ejemplo, métodos basados en apariencia para la identificación de objetos pequeños, una vez aislados).

- Podría flexibilizarse el enfoque de visión activa adaptando la calibración odométrica, y en general de toda la arquitectura, a robots con cámaras en torretas controladas independientemente del movimiento del vehículo. También quizá a cabezas estéreo o trinoculares, utilizando esta redundancia como fuente de robustez en el modelado 3D. Una línea de investigación ya en marcha, gracias a la reciente concesión de financiación para un proyecto relacionado, es la adaptación de la arquitectura a robots con patas, capaces de realizar movimientos más complejos y en entornos más agresivos.

- Otra vía de exploración atrayente es la extensión del modelo interpretativo para mantener la consistencia estructural a medio y largo plazo, utilizando el concepto de espacio perceptivo extendido comentado en la sección 5.3.2. Esto contribuiría a resolver el problema del reconocimiento de lugares ya visitados, entre otros. Una posibilidad sería la construcción *on-line* de mapas topológicos jerárquicos sobre las reconstrucciones euclídeas, incorporando mecanismos eficientes para el mantenimiento de estas representaciones.

- En los aspectos más relacionados con la implementación se abre también un amplio abanico de posibilidades. Desde la implementación directa en hardware de las etapas inferiores (preprocesamiento, extracción de primitivas, etc.), por ejemplo utilizando FPGAs, hasta la adaptación de la arquitectura a dispositivos computacionales más potentes (multiprocesadores, procesadores con tecnología *hyperthreading*, etc.).

- Finalmente, y quizá como objetivo más ambicioso y a largo plazo, extender el paradigma interpretativo a entornos progresivamente menos estructurados, como los exteriores, por ejemplo. Estos entornos exigirían modelos conceptuales mucho más complejos y flexibles, multiplicando consecuentemente todas y cada una de las dificultades inherentes al problema de la percepción.

Bibliografía

ActivMedia Robotics (1999). "Pioneer 2 DX Mobile Robot Operations Manual." Peterborough (USA), URL: `http://www.activmedia.com`

Aloimonos, Y. (ed.) (1997). *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*. Lawrence Erlbaum Associates, New Jersey (USA).

Aloimonos, Y., Weiss, I., y Bandyopadhyay, A. (1988). "Active vision." *International Journal of Computer Vision*, 1, 333–356.

Alter, T. y Basri, R. (1998). "Extracting salient contours from images: An analysis of the saliency network." *International Journal of Computer Vision*, 27, 51–69.

Amir, A. y Lindenbaum, M. (1998). "A generic grouping algorithm and its quantitative analysis." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2), 168–185.

Arbib, M. (1995). *The Handbook of Brain Theory and Neural Networks*. MIT Press, Cambridge (USA).

Aste, M. y Boninsegna, M. (1993). "A fast straight line extractor for vision-guided robot navigation." *Informe Técnico No. I-38050*, Istituto per la Ricerca Scientifica e Tecnologica, Trento (Italy).

Aström, K. (2000). "Multiple view vision." *Proceedings of the 15th International Conference on Pattern Recognition*, Vol. 1, Barcelona (Spain). IEEE Computer Society Press, 59–66.

Baillard, C. y Zisserman, A. (1999). "Automatic reconstruction of piecewise planar models from multiple views." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Fort Collins (USA). 559–565.

Basri, R., Rivlin, E., y Shimshoni, I. (1998). "Visual homing: Surfing on the epipoles." *Proceedings of the 6th IEEE International Conference on Computer Vision*, Bombay (India). 863–869.

Beardsley, P., Reid, I., Zisserman, A., y Murray, D. (1995). "Active visual navigation using non-metric structure." *Proceedings of the 5th IEEE Conference on Computer Vision*, Cambridge (USA). 58–64.

Beardsley, P. y Zisserman, A. (1995). "Affine calibration of mobile vehicles." *Proceedings of the Europe-China Workshop on Geometrical Modelling and Invariants for Computer Vision*, Xian (China). 214–221.

Beccari, G., Caselli, S., Zanichelli, F., y Calafiore, A. (1997). "Vision-based line tracking and navigation in structured environments." *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Monterey (USA). 406–411.

Beveridge, J. y Riseman, E. (1997). "How easy is matching 2D line models using local search?." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6), 564–579.

Blanz, W. y Gish, S. (1990). "A connectionist classifier architecture applied to image segmentation." *Proceedings of the 10th International Congress on Pattern Recognition*, Atlantic City (USA). 272–277.

Bougnoux, S. (1998). "From projective to euclidean space under any practical situation." *Proceedings of the 6th IEEE International Conference on Computer Vision*, Bombay (India). 790–796.

Bouguet, J. y Perona, P. (1995). "Visual navigation using a single camera." *Proceedings of the 5th IEEE International Conference on Computer Vision*, Los Alamitos (USA). 645–652.

Brooks, R. (1991). "Intelligence without representation." *Artificial Intelligence*, 47, 139–159.

Brooks, R. (2001). "The relationship between matter and life." *Nature*, 409, 409–411.

Bruce, J., Balch, T., y Veloso, M. (2000). "Fast and inexpensive color image segmentation for interactive robots." *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 3, Takamatsu (Japan). 2061–2066.

Buffa, M., Faugeras, O., y Zhang, Z. (1992). "Obstacle avoidance and trajectory planning for an indoor mobile robot using stereo vision and delaunay triangulation." *Vision-based Vehicle Guidance*, 268–283. En (Masaki, 1992).

Buluswar, S. y Draper, B. (1998). "Color machine vision for autonomous vehicles." *International Journal for Engineering Applications of Artificial Intelligence*, 11(2), 245–256.

Bustos, P. (1998). "Generación de comportamiento complejo en robots autónomos." Tesis Doctoral, Facultad de Informática, Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid (España).

Camus, T., Coombs, D., Herman, M., y Hong, T. (1999). "Real-time single-workstation obstacle avoidance using only wide-field of view divergence." *Videre: Journal of Computer Vision Research*, 1(3), 30–57.

Canny, J. (1986). "A computational approach to edge detection." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), 679–698.

Carlsson, S., Mohr, R., Moons, T., Morin, L., Rothwell, C., Diest, M. V., Gool, L. V., Veillon, F., y Zissermann, A. (1996). "Semi-local projective invariants for the recognition of smooth plane curve." *International Journal of Computer Vision*, 19(3), 211–236.

Churchland, P. y Sejnowski, T. (1992). *The Computational Brain*. MIT Press, Cambridge (USA).

Coianiz, T. y Aste, M. (1993). "Improving robot's indoor navigation capabilities by integrating visual, sonar and odometric measurements." *Informe Técnico No. 9302-11*, Istituto per la Ricerca Scientifica e Tecnologica, Trento (Italy).

Coltheart, V. (ed.) (1999). *Fleeting Memories*. MIT Press, Cambridge (USA).

Comaniciu, D. y Meer, P. (1997). "Robust analysis of feature spaces: Color image segmentation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, San Juan de Puerto Rico (USA). 750–755.

Comaniciu, D., Ramesh, V., y Meer, P. (2000). "Real-time tracking of non-rigid objects using mean shift." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, Hilton Head Island (USA). 2142–2151.

Daniilidis, K. y Spetsakis, M. (1997). "Understanding noise sensitivity in structure from motion." *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*. En (Aloimonos, 1997).

Davies, E. (1997). *Machine Vision: Theory, Algorithms, Practicalities*. Academic Press, San Diego (USA), 2^{nd} edition.

Davison, A. y Kita, N. (2001). "Sequential localisation and map-building for real-time computer vision and robotics." *Robotics and Autonomous Systems*, 36(4), 171–183.

Davison, A. y Murray, D. (1998). "Mobile robot localisation using active vision." *Proceedings of the 5^{th} European Conference on Computer Vision*, Freiburg (Germany). 809–825.

De Agapito, L., Hayman, E., y Reid, I. (2001). "Self-calibration of rotating and zooming cameras." *International Journal of Computer Vision*, 45(2).

De Agapito, L., Huynh, D., y Brooks, M. (1998). "Self-calibrating a stereo head: An error analysis in the neighbourhood of degenerate configurations." *Proceedings of the 6^{th} IEEE International Conference on Computer Vision*, Bombay (India). 747–753.

Dean, T. y Marion, J. (1997). "Planning and navigation in stochastic environments." *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*. En (Aloimonos, 1997).

Dellaert, F., Burgard, W., Fox, D., y Thrun, S. (1999). "Using the CONDENSATION algorithm for robust, vision-based mobile robot localization." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, Fort Collins (USA). 588–594.

Dennet, D. (1991). *Consciousness Explained*. Little, Brown and Company, Boston (USA).

Dennet, D. (1995). *Darwin's Dangerous Idea*. Simon and Schuster, New York (USA).

Deriche, R. y Faugeras, O. (1990). "Tracking line segments." *Image and Vision Computing*, 8(4), 261–270.

Devernay, F. y Faugeras, O. (1995). "Automatic calibration and removal of distortion from scenes of structured environments." *Proceedings of the SPIE (Society of Photo-Optical Instrumentation Engineers) Conference on Investigate and Trial Image Processing*, Vol. 2567, San Diego (USA).

Devroye, L., Györfi, L., y Lugosi, G. (1996). *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York (USA).

Dongarra, J., Du Croz, J., Hammarling, S., y Duff, I. (1988). "An extended set of level 3 Basic Linear Algebra Subprograms." *ACM Transactions on Mathematical Software*, 14(1), 1–32.

Douglas, D. y Peuker, T. (1973). "Algorithms for the reduction of the number of points required to represent a digitised line or its caricature." *The Canadian Cartographer*, 10, 112–122.

Drummond, T. y Cipolla, R. (1999). "Visual tracking and control using lie algebras." *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, Fort Collins (USA). 652–657.

Drummond, T. y Cipolla, R. (2002). "Real-time tracking of complex structures with on-line camera calibration." *Image and Vision Computing*, 20(5-6), 427–433.

Faugeras, O. (1992). "What can be seen in three dimensions with an uncalibrated stereo rig?." *Proceedings of the 2nd European Conference on Computer Vision*, Vol. 588 de *Lecture Notes in Computer Science*, Santa Margherita (Italy). Springer-Verlag, 563–578.

Faugeras, O. (2001). *The geometry of multiple images*. MIT Press, Cambridge (USA).

Faugeras, O., Luong, Q., y Maybank, S. (1992). "Camera self-calibration: Theory and experiments." *Proceedings of the 2nd European Conference on Computer Vision*, Vol. 588 de *Lecture Notes in Computer Science*, Santa Margherita (Italy). 321–334.

Faugeras, O. y Toscani, G. (1986). "The calibration problem for stereo." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Miami (USA). 15–20.

Fermüller, C. y Aloimonos, Y. (1997). "Direct motion perception." *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*. En (Aloimonos, 1997).

Fischler, M. y Bolles, R. (1981). "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography." *Communications of the ACM*, 24(6), 381–395.

Foley, J., Dam, A. V., Feiner, S., y Hughes, J. (1990). *Computer Graphics: Principles and Practice*. Addison-Wesley.

Fox, D., Burgard, W., y Thrun, S. (1999). "Markov localization for mobile robots in dynamic environments." *Journal of Artificial Intelligence Research*, 11, 391–427.

Frizera, R., Schneebeli, H., y Santos-Victor, J. (2000). "Visual navigation using visual servoing and appearance based methods." *Robotics and Autonomous Systems*, 31, 87–97.

Garey, M. y Johnson, D. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman and Company, New York (USA).

Geman, S. y Geman, D. (1984). "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6, 721–741.

Golub, G. y Van Loan, C. (1989). *Matrix Computations*. Johns Hopkins University Press, Baltimore (USA), 2^{nd} edition.

Gonzalez, R. y Woods, R. (1993). *Digital Image Processing*. Addison-Wesley.

Granlund, G. y Knutsson, H. (1995). *Signal Processing for Computer Vision*. Kluwer Academic Publishers, Dordrecht (The Netherlands).

Gros, P., Hartley, R., Mohr, R., y Quang, L. (1997). "How useful is projective geometry?." *Computer Vision and Image Understanding*, 65(3), 442–446.

Gutmann, J. y Konolige, K. (2000). "Incremental mapping of large cyclic environments." *Proceedings of the IEEE International Conference on Robotics and Automation*, Monterey (USA). 318–325.

Haralick, R. y Shapiro, L. (1985). "Survey: Image segmentation techniques." *Computer Vision Graphics and Image Processing*, 29, 100–132.

Harris, C. y Stephens, M. (1988). "A combined corner and edge detector." *Proceedings of the 4^{th} Alvey Vision Conference*, Manchester (UK). 147–151.

Hartley, R. (1994). "Self-calibration from multiple views with a rotating camera." *Proceedings of the 3rd European Conference on Computer Vision*, Vol. 800-801 de *Lecture Notes in Computer Science*, Stockholm (Sweden). Springer-Verlag, 471–478.

Hartley, R. (1997). "Kruppa's equations derived from the fundamental matrix." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2), 133–135.

Hartley, R., Gupta, R., y Chang., T. (1992). "Stereo from uncalibrated cameras." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Champaign (USA). 761–764.

Hartley, R. y Zisserman, A. (2000). *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge (UK).

Hebb, D. (1949). *The Organization of Behavior*. John Wiley and Sons, New York (USA).

Herman, M., Nashman, M., Hong, T., Schneiderman, H., Coombs, D., Young, G., Raviv, D., y Wavering., A. (1997). "Minimalist vision for navigation." *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*. En (Aloimonos, 1997).

Hofstadter, D. (1995). *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought*. Basic Books, New York (USA).

Horaud, R., Dornaika, F., y Espiau, B. (1998). "Visually guided object grasping." *IEEE Transactions on Robotics and Automation*, 4(14), 525–532.

Hough, P. (1962). "Methods and means for recognising complex patterns." U.S. Patent 3-069-654.

Howard, A. y Kitchen, L. (1997). "Fast visual mapping for mobile robot navigation." *Proceedings of the IEEE International Conference on Intelligent Processing Systems*, Beijing (China). 1251–1255.

Intel Corporation (2000a). "The Intel Image Processing Library Homepage." Intel Software Development Products, URL: `http://www.intel.com/software/products/perflib/ipl/`

Intel Corporation (2000b). "The Open Source Computer Vision (OpenCV) Library Homepage." Intel Research, URL: `http://www.intel.com/research/mrl/research/opencv/`

Intel Corporation (2002). "The Intel Integrated Performance Primitives Homepage." Intel Software Development Products, URL: `http://www.intel.com/software/products/ipp/`

Iocchi, L., Konolige, K., y Bajracharya, M. (2000). "Visually realistic mapping of a planar environment with stereo." *Proceedings of the 7th International Symposium on Experimental Robotics*, Hawaii (USA). 521–532.

Isard, M. y Blake, A. (1998). "Condensation - conditional density propagation for visual tracking." *International Journal of Computer Vision*, 29(1), 5–28.

Jain, A., Zhong, Y., y Dubuisson, J. (1998). "Deformable template models: A review." *Signal Processing*, 71(2), 109–129.

Jain, A., Zhong, Y., y Lakshmanan, S. (1996). "Object matching using deformable templates." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3), 267–278.

Jermyn, I. y Ishikawa, H. (1999). "Globally optimal regions and boundaries." *Proceedings of the IEEE International Conference on Computer Vision*, Kerkyra (Greece). 904–910.

Jogan, M., Artac, M., Skocaj, D., y Leonardis, A. (2003). "A framework for robust and incremental self-localization of a mobile robot." *Proceedings of the 3rd International Conference on Computer Vision Systems*, Vol. 2626 de *Lecture Notes in Computer Science*, Graz (Austria). Springer-Verlag, 460–469.

Kahl, F. y Triggs, B. (1999). "Critical motions in euclidean structure from motion." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Fort Collins (USA). 366–372.

Kalviainen, H. y Hirvonen, P. (1995). "Connective randomized Hough transform." *Proceedings of the 9th Scandinavian Conference on Image Analysis*, Uppsala (Sweden). 1029–1035.

Kass, M., Witkin, A., y Terzopoulos, D. (1988). "Snakes: Active contour models." *International Journal of Computer Vision*, 1(4), 312–331.

Kelly, A. (2000). "Pose determination and tracking in image mosaic-based vehicle position estimation." *International Conference on Intelligent Robots and Systems*, Takamatsu (Japan).

Kiryati, N., Eldar, Y., y Bruckstein, A. (1991). "A probabilistic Hough transform." *Pattern Recognition*, 24(4), 303–316.

Konolige, K., Myers, K., Ruspini, E., y Saffiotti, A. (1997). "The Saphira architecture: A design for autonomy." *Journal of Experimental and Theoretical Artificial Intelligence*, 9(1), 215–235.

Leavers, V. (1993). "Survey: Which Hough transform?." *Computer Vision, Graphics, and Image Processing: Image Understanding*, 58, 250–264.

Lebegue, X. y Aggarwal, J. (1993). "Significant line segments for an indoor mobile robot." *IEEE Transactions on Robotics and Automation*, 9(6), 801–815.

LeCun, Y., Bottou, L., Bengio, Y., y Haffner, P. (1998). "Gradient-based learning applied to document recognition." *Proceedings of the IEEE*, 86(11), 2278–2324.

Lee, W., Roh, K., y Kweon, I. (2000). "Self-localization of a mobile robot without camera calibration using projective invariants." *Pattern Recognition Letters*, 21, 45–60.

Li, H., Lavin, M. A., y LeMaster, R. J. (1986). "Fast Hough transform: A hierarchical approach." *Computer Vision, Graphics and Image Processing*, 36, 139–161.

Liang, B. y Pears, N. (2002). "Visual navigation using planar homographies." *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 1, Washington (USA). 205–210.

Liu, Y., Emery, R., Chakrabarti, D., Burgard, W., y Thrun, S. (2001). "Using EM to learn 3D models of indoor environments with mobile robots." *Proceedings of the* 18^{th} *IEEE International Conference on Machine Learning*, Williamstown (USA). 329–336.

Longuet-Higgins, H. (1981). "A computer algorithm for reconstructing a scene from two projections." *Nature*, 293, 133–135.

López de Teruel, P. y Ruiz, A. (1997). "A probabilistic learning algorithm for real time line detection." *Informe Técnico No. DIS-9-97*, Departamento de Informática y Sistemas, Universidad de Murcia (España).

López de Teruel, P. y Ruiz, A. (1998). "On-line probabilistic learning techniques for real-time computer vision." *Proceedings of the Learning'98*, Madrid (Spain).

López de Teruel, P. y Ruiz, A. (1999a). "On efficient line detection and tracking in noisy images: An approach based on saliency and probabilistic techniques." *Actas del VIII Symposium Nacional de Reconocimiento de Formas y Análisis de Imágenes*, Bilbao (España). 77–78.

López de Teruel, P. y Ruiz, A. (1999b). "On efficient line detection and tracking in noisy images: An approach based on saliency and probabilistic techniques." *Informe Técnico No. LSI-5-99*, Departamento de Informática y Sistemas, Universidad de Murcia (España).

López de Teruel, P., Ruiz, A., y García, J. (2000). "A parallel algorithm for tracking of segments in noisy edge images." *Proceedings of the* 15^{th} *International Conference on Pattern Recognition*, Vol. 4, Barcelona (Spain). IEEE Computer Society Press, 807–811.

López de Teruel, P., Ruiz, A., García-Mateos, G., y García, J. (2003). "Real-time extraction of colored segments for robot visual navigation." *Proceedings of the* 3^{rd} *International Conference on Computer Vision Systems*, Vol. 2626 de *Lecture Notes in Computer Science*, Graz (Austria). Springer-Verlag, 428–437.

Lucas, B. y Kanade, T. (1981). "An iterative image registration technique with an application to stereo vision." *Proceedings of the International Joint Conference on Artificial Intelligence*, Vancouver (Canada). 674–679.

Lundquist, A. (1997). "Line based visual navigation using pose clustering." Tesis Doctoral, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm (Sweden).

Luong, Q., Deriche, R., Faugeras, O., y Papadopoulo, T. (1993). "On determining the fundamental matrix: Analysis of different methods and experimental results." *Informe Técnico No. 1894*, Institut National de Recherche en Informatique et en Automatique, France.

Mallot, H., Franz, M., Schölkopf, B., y Bülthoff, H. (1997). "The viewgraph approach to visual navigation and spatial memory." *Proceedings of the 7th International Conference on Artificial Neural Networks*, Lausanne (Switzerland). Springer-Verlag, 751–756.

Marr, D. y Hildreth, E. (1980). "Theory of edge detection." *Proceedings of the Royal Society of London*, 207, 187–217.

Masaki, I. (ed.) (1992). *Vision-based Vehicle Guidance*. Springer-Verlag, New York (USA).

Matas, J., Galambos, C., y Kittler, J. (2000). "Robust detection of lines using the progressive probabilistic Hough transform." *Computer Vision and Image Understanding*, 78, 119–137.

McCulloch, W. y Pitts, W. (1943). "A logical calculus of the ideas immanent in nervous activity." *Bulletin of Mathematical Biophysics*, 5, 115–133.

McKenna, S., Raja, Y., y Gong, S. (1999). "Tracking colour objects using adaptive mixture models." *Image and Vision Computing*, 17, 223–229.

McLachlan, G. y Krishnan, T. (1997). *The EM Algorithm and Extensions*. John Wiley and Sons, New York (USA).

McLaughlin, R. y Alder, M. (1998). "The Hough transform versus the UpWrite." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20, 396–400.

Medioni, G., Lee, M., y Tang, C. (2000). *A Computational Framework for Segmentation and Grouping*. Elsevier Science, Amsterdam (The Netherlands).

Mirmehdi, M. y Petrou, M. (2000). "Segmentation of colour textures." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(2), 140–159.

Modestino, J. y Zhang, J. (1992). "A Markov random field model-based approach to image interpretation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(6), 606–615.

Montiel, J. y Zisserman, A. (2001). "Automated architectural acquisition from a camera undergoing planar motion." *Proceedings of the International Symposium on Virtual and Augmented Architecture*, Dublin (Ireland).

Moravec, H. (1996). "Robot spatial perception by stereoscopic vision and 3D evidence grids." *Informe Técnico No. CMU-RI-TR-96-34*, Robotics Institute, Carnegie Mellon University, Pittsburgh (USA).

Muñoz, E. y Baumela, L. (2002). "Conic-based lens distortion estimation." *Actas de la VII Conferencia Iberoamericana de Inteligencia Artificial*, Sevilla (España).

Newell, A. y Simon, H. (1976). "Computer science as empirical inquiry: Symbols and search." *Communications of the ACM*, 19(3), 113–126.

Nilsson, N. (1965). *Learning Machines*. McGraw-Hill, New York (USA).

O'Gorman, F. y Clowes, M. (1976). "Finding picture edges through collinearity of feature points." *IEEE Transactions on Computers*, 25(4), 449–456.

O'Regan, J. y Noe, A. (2001). "A sensorimotor account of vision and visual consciousness." *Behavioral and Brain Sciences*, 24(5).

Pajares, G. y de la Cruz, J. (2000). "Formas a partir de X." *Revista Electrónica de Visión por Computador*, 3.

Pal, N. y Pal, S. (1993). "A review on image segmentation techniques." *Pattern Recognition*, 26, 1277–1294.

Paler, K., Foglein, J., Illingworth, J., y Kittler, J. (1984). "Local ordered grey levels as an aid to corner detection." *Pattern Recognition*, 17, 535–543.

Pears, N. y Liang, B. (2001). "Ground plane segmentation for mobile robot visual navigation." *Proccedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Hawaii (USA). 1513–1518.

Penrose, R. (1994). *Shadows of the Mind. A Search for the Missing Science of Consciousness*. Oxford University Press, Oxford (UK).

Perlovsky, L. (1998). "Conundrum of combinatorial complexity." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(6), 666–670.

Pikaz, A. y Dinstein, I. (1995). "An algorithm for polygonal approximation of digital curves." *Pattern Recognition Letters*, 16, 557–563.

Pizlo, Z. y Rosenfeld, A. (1992). "Recognition of planar shapes from perspective images using contour-based invariants." *Computer Vision, Graphics, and Image Processing: Image Understanding*, 56(3), 330–350.

Pollefeys, M. (2000). "Tutorial on 3D modeling from images." Catholic University of Leuven (Belgium).

Riseman, E., Hanson, A., Beveridge, J., Kumar, R., y Sawhney, H. (1997). "Landmark-based navigation and the acquisition of environmental models." *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*. En (Aloimonos, 1997).

Robert, L., Zeller, C., Faugeras, O., y Hebert, M. (1997). "Applications of non-metric vision to some visually guided robotics tasks." *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*. En (Aloimonos, 1997).

Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan, Washington (USA).

Rosin, P. (1997a). "Edges: Saliency measures and automatic thresholding." *Machine Vision and Applications*, 9, 139–159.

Rosin, P. (1997b). "Techniques for assesing polygonal approximation of curves." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19, 659–666.

Ruf, A. y Horaud, R. (2000). "Vision-based guidance and control of robots in projective space." *Proceedings of the 6th European Conference on Computer Vision*, Vol. 2, Dublin (Ireland). 50–66.

Ruiz, A. y López de Teruel, P. (2003). "Página *web* del Proyecto GeoBot." Universidad de Murcia (España), URL: `http://dis.um.es/~alberto/geobot/geobot.html`

Ruiz, A., López de Teruel, P., y García-Mateos, G. (2002). "A note on principal point estimability." *Proceedings of the 16th International Conference on Pattern Recognition*, Vol. 2, Quebec (Canada). 304–307.

Ruiz, A., López de Teruel, P., y Garrido, M. (1998). "Probabilistic inference from arbitrary uncertainty using mixtures of factorized generalized gaussians." *Journal of Artificial Intelligence Research*, 9, 167–217.

Sarkar, S. y Boyer, K. (1993). "Perceptual organization in computer vision: A review and a proposal for a classification structure." *IEEE Transactions on Systems, Man and Cybernetics*, 23(2), 382–399.

Schaffalitzky, F. y Zisserman, A. (1999). "Planar grouping for automatic detection of vanishing lines and points." *Image and Vision Computing*, 18(9), 647–658.

Schölkopf, B., Burges, C., y Smola, A. (eds.) (1999). *Advances in Kernel Methods. Support Vector Learning*. MIT Press, Cambridge (USA).

Schmid, C. y Zisserman, A. (1997). "Automatic line matching across views." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, San Juan de Puerto Rico (USA). 666–671.

Shapiro, L. y Stockman, G. (2000). *Computer Vision*. Prentice Hall, New Jersey (USA).

Shashua, A. y Ullman, S. (1991). "Grouping contours by iterated pairing networks." *Advances in Neural Information Processing Systems*, 3, 335–341.

Shi, J. y Malik, J. (2000). "Normalized cuts and image segmentation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 888–905.

Shi, J. y Tomasi, C. (1994). "Good features to track." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Seattle (USA). 593–600.

Simons, D. y Chabris, C. (1999). "Gorillas in our midst: Sustained inattentional blindness for dynamical events." *Perception*, 28, 1059–1074.

Smith, S. y Brady, J. (1997). "Susan: A new approach to low level image processing." *International Journal of Computer Vision*, 23, 45–78.

Swain, R. y Devy, M. (1997). "Visually-guided navigation of a mobile robot in a structured environment." *Proceedings of the* 5^{th} *International Symposium on Intelligent Robotic Systems*, Stockholm (Sweden). 143–152.

Taylor, C., Ostrowski, J., y Jung, S. (1999). "Robust visual servoing based on relative orientation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Fort Collins (USA). 574–580.

Tommasini, T., Fusiello, A., Trucco, E., y Roberto, V. (1998). "Making good features to track better." *Proceedings of the IEEE Conference on Computer Vision Pattern Recognition*, Santa Barbara (USA). 145–149.

Torr, P. y Murray, D. (1997). "The development and comparison of robust methods for estimating the fundamental matrix." *International Journal of Computer Vision*, 24(3), 271–300.

Triggs, B. (1997). "Autocalibration and the absolute quadric." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, San Juan de Puerto Rico (USA). 609–614.

Trucco, E. y Verri, A. (1998). *Introductory Techniques for 3D Computer Vision*. Prentice Hall, New Jersey (USA).

Tsai, R. (1987). "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses." *IEEE Journal of Robotics and Automation*, 3(4), 323–344.

Unnikrishnan, R. y Kelly, A. (2002). "Mosaicing large cyclic environments for visual navigation in autonomous vehicles." *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 4, Washington (USA). 4299–4306.

Van Gelder, T. (1998). "The dynamical hypothesis in cognitive science." *Behavioural and Brain Sciences*, 21, 615–665.

Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley and Sons, New York (USA).

Verri, A. y Poggio, T. (1989). "Motion field and optical flow: Qualitative properties." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5), 490–498.

Vicente, J. (2002). "Métodos visuales de reconstrucción 3D en robots autónomos." Tesis Doctoral, E.T.S. de Ingenieros de Telecomunicación, Departamento de Tecnologías Especiales Aplicadas a la Telecomunicación, Universidad Politécnica de Madrid (España).

Waldherr, S., Thrun, S., y Romero, R. (1998). "A neural-network based approach for recognition of pose and motion gestures on a mobile robot." *Proceedings of the 5th Brazilian Symposium on Neural Networks*, Belo Horizonte (Brazil). 79–84.

Wei, G. y Ma, S. (1994). "Implicit and explicit camera calibration: Theory and experiments." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5), 469–480.

Welch, G. y Bishop, G. (1995). "An introduction to the Kalman filter." *Informe Técnico No. TR95-041*, Department of Computer Science, University of North Carolina (USA).

Weng, J., Chen, S., y Huang, T. (1997). "Visual navigation using fast content-based retrieval." *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*. En (Aloimonos, 1997).

Weng, J., Cohen, P., y Herniou, M. (1992). "Camera calibration with distortion models and accuracy evaluation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(102), 965–980.

Williams, L. y Jacobs, D. (1997a). "Local parallel computation of stochastic completion fields." *Neural Computation*, 9, 859–881.

Williams, L. y Jacobs, D. (1997b). "Stochastic completion fields: A neural model of contour shape and salience." *Neural Computation*, 9, 837–858.

Wolfe, J. (1999). "Inattentional amnesia." *Fleeting Memories*, 71–94. En (Coltheart, 1999).

Worgotter, F. y Cozzi, A. (1999). "A parallel noise-robust algorithm to recover depth information from radial flow fields." *Neural Computation*, 1, 381–416.

Wu, Z. y Leahy, R. (1993). "An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15, 1101–1113.

Xu, G., Terai, J., y Shum, H. (2000). "A linear algorithm for camera self-calibration, motion and structure recovery for multi-planar scenes from two perspective images." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, Hilton Head Island (USA). 2474–2479.

Xu, L. y Oja, E. (1993). "Randomized Hough transform: Basic mechanisms, algorithms, and computational complexities." *Computer Vision, Graphics, and Image Processing: Image Understanding*, 57(2), 131–154.

Yang, M., Lee, J., Lien, C., y Huang, C. (1997). "Hough transform modified by line connectivity and line thickness." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(8), 905–909.

Yu, D. y Yan, H. (1997). "An efficient algorithm for smoothing, linearization and detection of structural feature points of binary image contours." *Pattern Recognition*, 30(1), 57–69.

Zhang, Z. (1994). "Token tracking in a cluttered scene." *Image and Vision Computing*, 12(2), 110–120.

Zhang, Z. (1995). "Estimating motion and structure from correspondences of line segments between two perspective images." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12), 1129–1139.

Zhang, Z. y Faugeras, O. (1991). "Determining motion from 3D line segment matches: A comparative study." *Image and Vision Computing*, 9(1), 10–19.

Zhang, Z. y Faugeras, O. (1992). "A 3D world model builder with a mobile robot." *International Journal of Robotics Research*, 11(4), 269–285.

Zhang, Z., Weiss, R., y Hanson, A. (1993). "Automatic calibration and visual servoing for a robot navigation system." *Informe Técnico No. CMPSCI TR93-14*, Department of Computer Science, University of Massachusetts, Amherst (USA).

Zisserman, A., Fitzgibbon, A., y Cross, G. (1999). "VHS to VRML: 3D graphical models from video sequences." *International Conference on Multimedia Systems*, Firenze (Italy). 51–57.