Resumen de la Tesis Doctoral

Introducción

La *visión* es el sentido más importante para muchos animales, incluidos los seres humanos, en cuanto a la cantidad de información que proporcionan al cerebro. En concreto, se estima que hasta el 80% de la percepción, el aprendizaje, la cognición y las actividades humanas dependen de alguna manera de la visión [158]. Mediante la captura de los rayos de luz que provienen de los objetos que nos rodean, nuestros ojos transmiten señales eléctricas a nuestro cerebro para formar una imagen completa de lo que tenemos delante. Estas imágenes permiten a nuestro cerebro interpretar adecuadamente nuestro entorno, lo cual ha sido de capital importancia para la evolución humana. Durante décadas, los investigadores han reconocido las enormes capacidades que la visión proporciona y han intentado replicar este proceso, desde la captura de imágenes hasta su interpretación semántica a alto nivel, para diseñar aplicaciones basadas en la visión. La combinación de cámaras y ordenadores supone el marco de trabajo natural para desarrollar dichas aplicaciones.

Entre los existentes enfoques basados en la visión, los sistemas de *visión estéreo* presentan características particularmente interesantes que surgen a partir de la obtención de dos imágenes simultáneas desde puntos de vista ligeramente distintos. En concreto, esta disposición permite inferir de manera sencilla la profundidad mediante la correlación de la información visual perteneciente al área solapada en ambas imágenes.

Por otro lado, la *robótica* se ha convertido en un importante campo de investigación que persigue el diseño de dispositivos (denominados *robots*) que puedan sustituir, e incluso mejorar, la acción humana en ciertas situaciones tediosas, repetitivas, duras o incluso peligrosas. El funcionamiento de los robots se basa en un conjunto de sensores utilizados para capturar información tanto del propio robot como del entorno que lo rodea, con el objetivo de maniobrar adecuadamente dentro de él. Este objetivo lleva de forma natural al uso de sistemas de visión artificial en los robots, especialmente en los denominados *robots móviles*, buscando una navegación autónoma y fiable. De esta forma, podría decirse que, gracias a la robótica móvil basada en visión artificial, el proceso de *interpretar* las imágenes capturadas ha pasado de ser un fin en sí mismo a un medio para desarrollar aplicaciones robóticas de alto nivel.

Como consecuencia de esto, el número de aplicaciones robóticas basadas en visión por computador está creciendo constantemente en estos últimos años. Esta tesis contribuye a este propósito, estando enfocada a la investigación en aras del desarrollo de robots verdaderamente autónomos, que sean capaces de explotar las capacidades que proporciona la visión estéreo para localizarse dentro de sus entornos de operación.

Ámbito de la tesis

Esta tesis se centra en la combinación de dos campos tecnológicos muy populares: la *robótica móvil* y la *visión por computador* o visión artificial, los cuales están experimentando un gran auge investigador en estos años.

De forma breve, la robótica móvil aborda el desarrollo de robots con capacidad de moverse de manera autónoma (o semi-autónoma) a través de su entorno. Entre otros, la estimación de la *pose* (posición y orientación) de un robot mientras se mueve es uno de problemas más importantes a los que la robótica móvil debe enfrentarse. Habiéndose basado tradicionalmente en sensores como los sónares o los escáneres láser, las aplicaciones de robótica móvil han abarcado desde el seguimiento de la pose del robot, hasta su localización de manera simultánea a la construcción de un mapa del entorno (denominado SLAM, por sus siglas en inglés), pasando por, entre otros, la detección de obstáculos, la estimación del propio movimiento del robot entre instantes de tiempo consecutivos o la localización global. La visión por computador, a su vez, engloba al conjunto de técnicas utilizadas para capturar y procesar imágenes con el objetivo de extraer información significativa de ellas. El filtrado de la imagen, la detección de puntos de interés o el emparejamiento de imágenes basado en la apariencia son meros ejemplos de estas técnicas, pero el conjunto de métodos existente en la visión artificial está en constante crecimiento. Las características especiales de las cámaras, como la gran cantidad de información que proporcionan o la mejora de la calidad de sus imágenes manteniéndose con un coste relativamente bajo, han contribuido a la estandarización de las cámaras como sensores robóticos. En las últimas décadas, la convergencia de la robótica móvil y la visión por computador ha establecido un marco de trabajo muy prometedor, con potencial para desarrollar robots verdaderamente autónomos. Las metodologías más populares pertenecientes a ambas disciplinas se describen en el capítulo 2.

En particular, esta tesis se centra en el uso de cámaras estéreo como único sensor para estimar la pose del robot mientras se construye, de forma simultánea, un mapa del entorno. El denominado problema del SLAM es considerado uno de los problemas más arduos en la robótica móvil y representa un pilar básico en el desarrollo de sistemas de navegación autónomos y fiables. El uso de imágenes estéreo reduce las inherentes dificultades de los enfoques basados en cámaras monoculares en relación al factor de escala, proporcionando métodos directos para obtener información 3D del entorno. Como contrapartida, el procesamiento de dos imágenes en cada instante de tiempo implica un coste computacional mayor. Por esto, el uso de métodos robustos

y eficientes para la extracción de información de las imágenes se antoja obligatorio para el desarrollo de aplicaciones que funcionen en tiempo real. En este sentido, los métodos basados en comparaciones de píxeles se han convertido en un estándar a la hora de detectar puntos de interés de forma rápida, mientras que los descriptores binarios están posicionándose como los métodos más utilizados para el emparejamiento de puntos.

En otro orden de cosas, existen muchos enfoques y metodologías que pueden ser adoptadas para realizar SLAM visual, siendo el *filtrado* y el *suavizado* (también conocido como *bundle adjustment*) las dos tendencias más populares hoy en día. La primera marginaliza la información conocida hasta el instante actual acerca de la pose del robot y representa el estado del sistema (que incluye tanto la pose del robot como el mapa) con una distribución de probabilidad que se actualiza a lo largo del tiempo mediante una secuencia de acciones y observaciones. La segunda, por su parte, descarta parte de la información capturada anteriormente y estima, de forma iterativa y desde cero, una solución de mínimos cuadrados para el mapa y el conjunto de todas las poses del robot hasta el instante actual. Esta tesis explora ambas metodologías y propone una solución dentro de cada una de ellas, presentadas en el capítulo 4.

En esta tesis también se presta especial atención a la estimación del propio movimiento del robot entre instantes de tiempo consecutivos, sin tener en cuenta la construcción del mapa. La denominada *odometría visual* puede emplearse como modelo de movimiento en sustitución de la odometría tradicional basada en codificadores en las técnicas de filtrado para SLAM, mientras que también puede ser usada como una estimación inicial para los métodos de suavizado. Ambos enfoques se exploran en esta tesis, presentándose dos métodos nuevos en el capítulo 3.

Tanto en las aplicaciones de odometría visual como en las de SLAM visual, la presencia de *outliers* (o datos espurios) dentro de la información extraída de las imágenes supone un problema considerable, ya que las metodologías de filtrado y suavizado son significativamente sensibles a ellos, derivando en resultados erróneos. En general, este problema ha sido paliado mediante técnicas de hipótesis y verificación entre las que destaca RANSAC como la más representativa, con el coste de un incremento sensible de la carga computacional. Sin embargo, se pueden emplear alternativas más rápidas que arrojan resultados similares, como el que proponemos en el capítulo 3, formando parte de uno de nuestros sistemas de odometría visual.

Finalmente, para comparar el funcionamiento de diferentes técnicas de odometría y SLAM visual, es importante disponer de conjuntos de datos que incluyan información sobre el movimiento real del robot (típicamente denominado *ground truth*). La disponibilidad pública de este tipo de datos está aumentando en estos años y esta tesis contribuye a ello mediante dos nuevas colecciones grabadas en exteriores y descritas en el capítulo 5.

Contribuciones de la tesis

Las contribuciones más relevantes de esta tesis son:

- El desarrollo de un sistema de odometría visual estéreo basado en una formulación cerrada que es preciso, eficiente y no presenta los problemas de convergencia que sufren los métodos iterativos, bajo la asunción de ausencia de *outliers* en los datos de entrada. En este trabajo también probamos la adecuación de adoptar un modelo linearizado para estimar la incertidumbre de un punto 3D a partir de los errores de sus proyecciones en las imágenes. Los datos experimentales demuestran resultados similares a los obtenidos usando técnicas más complejas como la *Unscented Transform* (UT) o la UT escalada, manteniendo un coste significativamente menor. Este trabajo ha sido presentado en [132].

- El diseño de un nuevo método para detectar *outliers* basado en *kernels* robustos que pueden sustituir a las funciones cuadráticas de error estándar dentro de las optimizaciones iterativas de mínimos cuadrados, como alternativa rápida a los enfoques basados en el algoritmo RANSAC. Esta técnica ha sido publicada en [134].

- El desarrollo de un sistema completo de SLAM visual estéreo basado en filtros de partículas Rao-Blackwellised. Este sistema define un modelo de observación específico basado en descriptores SIFT y realiza la asociación de datos mediante la marginalización de la probabilidad de la observación entre todas las correspondencias posibles, incluida la nula. Este trabajo fue publicado en [133].

- El desarrollo de otro sistema de SLAM visual estéreo basado en un novedoso enfoque de *bundle adjustment* relativo, siguiendo la tendencia actual de emplear técnicas de suavizado para abordar el SLAM visual. La contribución en este caso se presenta en el diseño y la implementación de un sistema completo construido a partir de la integración de técnicas actuales ya existentes, así como en la validación del sistema global. Los resultados se publicarán en un artículo de revista que está, actualmente, en preparación.

- La publicación de dos colecciones de *datasets* grabadas en entornos urbanos sin modificar. Estas colecciones presentan una combinación heterogénea de sensores, incluyendo cámaras monoculares y estéreo, unidades de medidas inerciales, dispositivos GPS de última generación y escáneres láser. También se proporciona un *ground truth* del movimiento de los vehículos para ambas colecciones, habiéndose estimado a partir de las medidas GPS de alta precisión tomadas en la primera colección y a partir de las de un GPS estándar en la segunda. Finalmente, se realiza un estudio completo de la bondad de dicho *ground truth* para los primeros conjuntos de datos. Ambas colecciones se han publicado en [16] y [17].

Aquí está la lista de publicaciones derivadas de esta tesis, indicando el número de citas obtenidas hasta Febrero de 2015 (fuente: Google Scholar):

Revistas

- *Francisco-Angel Moreno, José-Luis Blanco and Javier González.* **A complete visual SLAM system based on Sparser Relative Bundle Adjustment**, (En preparación).
- *José-Luis Blanco-Claraco, Francisco-Ángel Moreno-Dueñas, Javier González-Jiménez.* **The Málaga urban dataset: High-rate stereo and LiDAR in a realistic urban scenario**, International Journal of Robotics Research (IJRR), n.2, vol. 33, pp. 207-214, (2014). 4 citations. [17]
- *Jose-Luis Blanco, Francisco-Angel Moreno, Javier Gonzalez.* **A collection of outdoor robotic datasets with centimeter-accuracy ground truth**, Autonomous Robots (AR), n.4, vol. 27, pp. 327-351, (2009). 54 citations. [16]
- *Francisco-Ángel Moreno, José-Luis Blanco, and Javier González.* **Stereo vision-specific models for Particle Filter-based SLAM**, Robotics and Autonomous Systems (RAS), n.9, vol. 57, pp. 955-970, (2009). 23 citations. [133]

Conferencias

- *Francisco-Angel Moreno, Jose-Luis Blanco, Javier González-Jiménez.* **ERODE: An Efficient and Robust Outlier Detector and its Application to Stereovisual Odometry**, in IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe (Germany), pp. 4676-4682, (2013). 1 citation. [134]
- *Francisco-Angel Moreno, José-Luis Blanco and Javier González.* **An Efficient Closed-form Solution to Probabilistic 6D Visual Odometry for a Stereo Camera**, in Advanced Concepts for Intelligent Vision Systems, Delft (The Netherlands), pp. 932-942, (2007). 10 citations. [132]

Marco de la tesis

Esta tesis es la culminación de 6 años de trabajo investigador como miembro del grupo MAPIR (MAchine Perception and Intelligent Robotics[1]) dentro del Departamento de Ingeniería de Sistemas y Automática de la Universidad de Málaga. Este trabajo ha sido parcialmente financiado por el programa FPU (Formación de Profesorado Universitario) promovido por el Ministerio de Educación, Cultura y Deporte de España.

Como estudiante de doctorado, el autor completó con éxito el programa doctoral en Ingeniería Mecatrónica coordinado por el propio Departamento de Ingeniería de

[1] `http://mapir.isa.uma.es`

Sistemas y Automática de la Universidad de Málaga, obteniendo una base sólida en una combinación de disciplinas (mecánica, electrónica, de control e informática) en las que se sustenta enormemente la robótica móvil.

Además, el autor completó su educación académica con la participación en la escuela de verano BMVA sobre visión por computador en 2010, albergada en Kingston (UK). En ella, se realizó una semana intensiva de clases y sesiones de laboratorio que abarcaron una gran variedad de temas en visión artificial e imágenes digitales, impartidas por investigadores pertenecientes a algunos de los grupos de visión por computador más activos del Reino Unido. Por otro lado, el autor también ha asistido a otros cursos de doctorado sobre visión artificial en el Departamento de Informática e Ingeniería de Sistemas de la Universidad de Zaragoza.

Por otro lado, el autor también ha llevado a cabo una estancia investigadora de tres meses de duración en el Department of Computer Science de la University of Bristol (UK), en 2011, bajo la supervisión del Dr. Andrew Calway y el Dr. Walterio Mayol-Cuevas [2] . Durante esta estancia, se investigó principalmente en el uso de diferentes descriptores de puntos de interés para realizar asociación de datos. Además de esto, el autor ha realizado otra estancia de casi tres meses de duración a principios de 2013 en la University of Lincoln (UK), llevando a cabo un estudio de investigación sobre fusión sensorial enfocado a la planificación de tareas para fumigadoras agrícolas. Durante esta estancia, se realizaron cooperaciones con los investigadores Dr. Tom Duckett y Dr. Grzegorz Cielniak.

Finalmente, es importante destacar que, durante estos años, se han realizado bastantes colaboraciones en diferentes proyectos de investigación desarrollados dentro del grupo MAPIR y relacionados con la aplicación de sensores distintos a los de visión para la localización de robots. Estas colaboraciones, aunque no estuvieron estricamente relacionadas con el tema de la visión por computador tratado aquí, merecen la pena ser mencionadas como otros trabajos de investigación del autor.

En ellas se incluyen los siguientes proyectos:

- **AGAVE**. Este proyecto estuvo enfocado al estudio de la Ultra-Wide Band (UWB), basada en radio, como tecnología con potencial para realizar localización robusta de vehículos autónomos que se mueven en almacenes de trabajo. También se estudió la combinación de la UWB con sensores GPS para extender las posibilidades de aplicación de la propuesta. Este proyecto supuso para el autor una primera toma de contacto con las metodologías de localización.

- **RoadBot**. El núcleo de este proyecto consistió en la fusión sensorial y la localización de vehículos. En él se utilizaba un vehículo eléctrico para realizar inspección de carreteras de forma muy precisa a partir de la combinación de medidas láser y lecturas de GPS milimétrico.

[2] `http://www.cs.bris.ac.uk/Research/Vision/Realtime/index.html`

- **Agrimap**. De manera muy similar a RoadBot, Agrimap fusionaba información láser con medidas GPS para inspeccionar campos agrícolas alrededor de fumigadoras industriales, y así planificar los movimientos de sus brazos.

Los resultados obtenidos en estos trabajos, en forma de artículos científicos, se enumeran a continuación:

Revistas

- *Francisco-Angel Moreno, Javier Gonzalez-Jimenez, Jose-Luis Blanco, Antonio Esteban*. **An instrumented vehicle for efficient and accurate 3D mapping of roads**, Computer-Aided Civil and Infrastructure Engineering (CACAIE), n.6, vol. 28, pp. 403-419, (2013). 4 citations.

- *Javier González, José-Luis Blanco, Cipriano Galindo, Antonio Ortiz-de-Galisteo, Juan-Antonio Fernández Madrigal, Francisco-Angel Moreno and Jorge Martínez*. **Mobile Robot Localization based on Ultra-Wide-Band Ranging: A Particle Filter Approach**, Robotics and Autonomous Systems (RAS), n.5, vol. 57, pp. 496-507, (2009). 47 citations.

Conferencias

- *Francisco-Angel Moreno-Duenas, Grzegorz Cielniak, Tom Duckett*. **Evaluation of laser range-finder mapping for agricultural spraying vehicles**, in Towards Autonomous Robotic Systems (TAROS), Oxford (UK), pp. 210-221, (2013). 1 citation.

- *Javier González, Cipriano Galindo, Jose-Luis Blanco, Juan-Antonio Fernandez-Madrigal, Vicente Arevalo and Francisco-Angel Moreno*. **SANCHO, a fair host robot. A description**, in IEEE International Conference on Mechatronics (ICM), Seville (Spain), pp. 1-6, (2009). 8 citations.

- *Javier González, José-Luis Blanco, Cipriano Galindo, Antonio Ortiz-de-Galisteo, Juan-Antonio Fernández-Madrigal, Francisco-Angel Moreno and Jorge Martínez*. **Combination of UWB and GPS for indoor-outdoor vehicle localization**, in IEEE International Symposium on Intelligent Signal Processing, Alcala de Henares (Madrid), Alcala de Henares (Spain), (2007). 18 citations.

Estructura de la tesis

Los capítulos que componen esta tesis se organizan de esta manera:

El **capítulo 2** proporciona una recopilación de los métodos y algoritmos principales de la visión por computador que se utilizan en este trabajo. Esto incluye, por ejemplo, detectores y descriptores de puntos de interés, técnicas de emparejamiento

o métodos para realizar asociación de datos. También se incluye en este capítulo un análisis de los métodos de localización más populares, incluyendo filtros probabilísticos y técnicas actuales de suavizado o *bundle adjustment*. Aún siendo no exhaustiva, esta recopilación constituye la base necesaria para abordar los temas que se tratan aquí y sirve de respaldo para los siguientes capítulos de esta tesis.

El **capítulo 3** presenta dos métodos para estimar el movimiento propio de un robot o cámara entre dos instantes de tiempo consecutivos, es decir, la odometría visual. Uno de estos métodos se basa en una formulación cerrada que trabaja con puntos 3D observados desde dos puntos de vista diferentes, mientras que el otro se construye a partir de una optimización iterativa no lineal de mínimos cuadrados que utiliza una función de coste robusta basada en la función pseudo-Huber. Este segundo método también presenta un enfoque alternativo a RANSAC, denominado ERODE, para la eliminación rápida de outliers. En este capítulo también se presentan experimentos con datos reales para validar el uso de ambas propuestas.

En el **capítulo 4** vamos un paso más allá y proponemos dos nuevos enfoques para abordar el problema del SLAM visual estéreo. Nuestra primera propuesta sigue un esquema de filtrado probabilístico particularizado en el denominado filtro de partículas Rao-Blackwellized y define un modelo de movimiento basado en el método de odometría visual de formulación cerrada mencionado en el párrafo anterior. El modelo de observación está basado en descriptores SIFT y la marginalización de la probabilidad de la observación sobre todas las posibles asociaciones, incluida la nula. Por otra parte, nuestra segunda propuesta pertenece a la categoría de soluciones de suavizado para el SLAM visual y presenta un sistema completo que incluye un *front-end* basado en puntos de interés y descriptores de tipo ORB, así como en una *bolsa de palabras* para ayudar a realizar la asociación de datos. El *back-end*, a su vez, utiliza *keyframes* bajo una metodología denominada *Sparser Relative Bundle Adjustment*, desarrollada en el seno del grupo MAPIR [15], situada a caballo entre las técnicas globales y puramente relativas de suavizado. En este método, implementamos nuestra técnica ERODE dentro del procedimiento de optimización iterativo no lineal de mínimos cuadrados. Ambas propuestas han sido validadas mediante una serie de experimentos que se presentan también en este capítulo.

El **capítulo 5** describe dos colecciones de *datasets* de exteriores grabados en la ciudad de Málaga que contienen imágenes estéreo junto con otros datos sensoriales como escáneres láser, medidas inerciales o lecturas de GPS. Nuestra primera colección está compuesta por seis recorridos en exteriores y proporciona un *ground truth* para la pose en 6D del vehículo, derivada a partir de tres receptores GPS de última generación. Nuestros segundos datasets han sido grabados en un entorno urbano a lo largo de una trayectoria de 36.8km, e incluyen imágenes estéreo de alta resolución junto con datos de otros sensores. Ambas colecciones se añaden a los bancos de datos ya existentes en repositorios públicos que son utilizados para probar y validar nuevos métodos de localización.

En el **capítulo 6** se muestran algunas de las conclusiones de esta tesis, resumiendo el trabajo de investigación presentado aquí e introduciendo nuevos objetivos a alcanzar.

Conclusiones

A lo largo de esta tesis se ha abordado principalmente el problema de localizar un robot móvil (o una cámara) dentro de un entorno mientras se construye simultáneamente un mapa del mismo, utilizando para ello imágenes estéreo. Este escenario, denominado SLAM visual, representa uno de los grandes problemas que conciernen tanto a la robótica móvil como a la visión artificial. A pesar de la gran cantidad de trabajos de investigación relacionados con este asunto que han sido presentados en las últimas décadas, es un tema que aún sigue abierto.

En esta tesis hemos trabajado tanto en odometría visual como en SLAM visual, aportando una nueva solución para cada uno de estos problemas. Así, hemos comenzado desarrollando técnicas para realizar odometría visual estéreo, intentando estimar el cambio de pose del robot entre instantes de tiempo consecutivos, proporcionando de esta manera un medio para realizar el seguimiento del robot mientras se mueve. Los sistemas típicos de odometría basados en codificadores aún se utilizan de manera bastante extendida pero su uso está restringido a robots con ruedas que se mueven en superficies planas. Es ahí donde la odometría visual juega un papel crucial, ya que puede ser aplicada a cualquier tipo de robot sin restricción alguna en su movimiento.

Podemos distinguir entre soluciones de formulación cerrada y enfoques basados en optimizaciones iterativas para realizar odometría visual, aunque ambos tipos comparten muchas de las etapas involucradas (por ejemplo, el preprocesamiento de la imagen, la detección y el emparejamiento de puntos de interés, etc.). En esta tesis, hemos presentado dos métodos nuevos, uno de cada tipo. Nuestra solución de formulación cerrada evita los problemas inherentes a los métodos iterativos como la necesidad de una estimación inicial y los problemas de convergencia a la vez que arroja buenos resultados en entornos de interiores. Sin embargo, necesita de un conjunto de datos de entrada libre de *outliers*, ya que los datos corruptos llevan al algoritmo a soluciones erróneas. Para solucionar este problema, se podría utilizar el conocido método RANSAC y hacer frente a los *outliers*, pero con el coste de un aumento considerable en la carga computacional, especialmente cuando el número esperado de *outliers* es alto.

En respuesta a esto, hemos desarrollado una nueva metodología, denominada ERODE, basada en funciones de coste robustas, que realiza una detección rápida de los *outliers* mientras se ejecuta un proceso iterativo de optimización no lineal para calcular la odometría visual. Esta técnica reduce significativamente el efecto de los

outliers en el resultado obtenido, a la vez que incurre en un coste computacional en torno a un orden de magnitud menor que el de RANSAC.

La odometría en general, y la odometría visual en particular, adolece de la acumulación de errores a lo largo del tiempo, aumentando la imprecisión en la estimación de la pose del robot. Por este motivo, las estimaciones odométricas se han considerado, típicamente, como parte de sistemas más complejos que realizan no sólo localización sino también construcción de mapas. De esta forma, en esta tesis hemos abordado también el problema del SLAM para sistemas estéreo dentro de dos marcos de trabajo diferentes que coexisten actualmente: *filtrado* y *suavizado*. De forma simplificada, el primero funciona mediante la ejecución iterativa de fases de *predicción* y *observación* dentro de un marco probabilístico, mientras que resumen toda la información hasta el instante de tiempo actual mediante una distribución de probabilidad. El segundo, a su vez, incluye el conjunto de técnicas que, bajo algunas asunciones, equivalen a un estimador de máxima probabilidad para el problema de estimar conjuntamente el grupo de puntos 3D del entorno y las poses del robot para toda la navegación.

En nuestra primera propuesta, basada en un filtro de partículas Rao-Blackwellised, el movimiento del robot se estima a partir de la formulación cerrada mencionada anteriormente para realizar odometría visual. Para el modelo de observación definido en este algoritmo, se considera a las observaciones como conjuntos de puntos formados por sus posiciones 3D (determinadas a partir de las coordenadas de sus proyecciones en las imágenes) y sus descriptores SIFT (extraídos del área local de dichas proyecciones), así como su incertidumbre asociada. También evitamos una asociación explícita de datos mediante la marginalización de la probabilidad de la observación sobre todas las asociaciones posibles, mitigando de esta forma los problemas derivados de los emparejamientos erróneos.

Nuestra segunda propuesta implementa un sistema completo de SLAM visual con un *front-end* basado en puntos de interés de tipo ORB y un *back-end* cuyo núcleo es una novedosa versión de un de las técnicas de suavizado mencionadas anteriormente: *bundle adjustment* relativo. Esta propuesta también implementa ERODE para realizar la odometría visual en presencia de *outliers*. En este trabajo, hemos utilizado un método basado en bolsas de palabras de descriptores ORB para ayudar a realizar la asociación de datos, reduciendo el área de búsqueda a la hora de emparejar las observaciones y el mapa. El *back-end* está diseñado siguiendo el método de *Sparser Relative Bundle Adjustment* (SRBA) [15], el cual representa el estado del sistema como un grafo cuyos nodos son las poses del robot en los denominados *keyframes* (*frames* especiales seleccionados cuidadosamente de entre todas las imágenes grabadas) y los puntos 3D del entorno. Los arcos del grafo representan las restricciones existentes entre nodos y son las incógnitas a resolver. Nuestro sistema basado en SRBA define sub-mapas en el grafo con el objetivo de reducir la distancia topográfica entre los *keyframes*. Esto nos lleva a un aumento del nivel de dispersión de las matrices involucradas en el proceso de optimización con respecto a otras implementaciones relativas de *bundle adjustment*. Junto con una limitación en el número de incógnitas a optimizar, este método mantiene acotado el tiempo de optimización en cada *keyframe*. A pesar de que el mapa construido es sólo localmente consistente (en contraposición del

método de *bundle adjustment* global), es posible su utilización en multitud de aplicaciones relacionadas con la localización. Aún así, si fuera necesario, se puede estimar un mapa global a partir de la representación relativa de los *keyframes* y los puntos 3D.

Todos los métodos presentados en esta tesis han sido validados con datos tanto reales como sintéticos a partir de simulaciones o imágenes reales, demostrando sus capacidades tanto para odometría visual como para SLAM visual.

Finalmente, nos hemos centrado en proporcionar a las comunidades de la visión por computador y la robótica móvil dos colecciones de datos grabados en exteriores, que son de acceso público y que contienen información de una gran variedad de sensores diferentes, incluyendo escáneres láser, unidades inerciales y cámaras.

Nuestra mayor contribución en el primer conjunto de datos es la derivación de un *ground truth* preciso basado en medidas GPS para la trayectoria del vehículo, junto con el estudio de sus bandas de incertidumbre asociadas (en el rango del 1 cm y 0.5°). Estas bandas se mantienen constantes a lo largo del tiempo en todos los conjuntos de datos, convirtiéndolos en un banco de pruebas ideal para la evaluación de métodos de odometría y SLAM visual. También se presentan nubes de puntos 3D de referencia para validar otras técnicas como la reconstrucción de superficies 3D o la planificación de caminos. Nuestra segunda colección, a su vez, contiene imágenes estéreo de alta resolución tomadas en entornos urbanos sin modificar, incluyendo situaciones exigentes con objetos móviles y tráfico real. La presencia de bucles en muchos de los datasets presentados también contribuye a la validación de la escalabilidad de las técnicas de SLAM.

Líneas futuras

Realmente creemos que la combinación de robótica móvil y visión artificial supone un marco de trabajo muy interesante y con mucho potencial a la hora de desarrollar robots verdaderamente autónomos. Sin embargo, alcanzar esto conlleva seguir un camino complicado y aún queda mucho por recorrer, más allá de las propuestas de esta tesis.

La robustez es uno de los conceptos básicos que se deben cumplir a la hora de realizar aplicaciones que trabajen en dicha dirección, ya que el mundo real presenta dificultades que, a veces, se tornan insalvables y desencadenan resultados no deseados. Entre ellos, la presencia de objetos dinámicos afecta especialmente a los algoritmos basados en visión ya que muchos de ellos asumen que trabajan en entornos estáticos. En este sentido, el desarrollo y la integración de técnicas como el *clustering*, capaces de detectar objetos móviles en la escena y aislar su movimiento de la estimación del movimiento del robot, se antojan necesarios para alcanzar la robustez deseada.

Por otro lado, los datos tomados en exteriores presentan ciertas problemáticas a la hora de calcular la posición 3D de puntos lejanos a la cámara, ya que la disparidad entre puntos correspondientes es muy pequeña en estos casos. Por tanto, pequeños errores en la localización de puntos de interés se convierten en grandes errores en la estimación de la posición 3D, especialmente con cámaras estéreo que presentan líneas base pequeñas. Este efecto puede mitigarse mediante el uso de detectores que proporcionan puntos de interés con precisión sub-píxel. Por tanto, el utilizar un detector rápido, con localización sub-píxel y que, además, añada detectores binarios a los puntos de interés supondría una gran mejora para el desarrollo de aplicaciones basadas en visión en exteriores.

Respecto a problemas más prácticos, la optimización del código y el uso de técnicas multi-hilo suponen las mejoras futuras que deben ser aplicadas a nuestros algoritmos para poder utilizarlos en aplicaciones de tiempo real. Estas mejoras pueden ser realizadas en combinación con el uso de imágenes de menor resolución (por ejemplo, 320x240 píxeles), aunque hay que ajustar nuestros métodos para poder trabajar correctamente con ellas. En este sentido, sería interesante realizar un pre-procesado de las primeras imágenes capturadas para un experimento concreto y así ajustar automáticamente muchos de los parámetros utilizados para definir el funcionamiento de nuestros métodos.

Finalmente, existen alternativas más rápidas a la hora de hacer emparejamiento de descriptores ORB que los algoritmos de fuerza bruta. Las técnicas de guiado para emparejar puntos deben ser consideradas para reducir el coste computacional, tanto en emparejamientos estéreo como entre frames.

Chapter 1
Introduction

Sleep is good... and books are better. – Tyrion Lannister

Vision is the dominant sense for many animals, including humans, in terms of the amount of sensed information provided to the brain. Specifically, it is estimated that up to eighty percent of human perception, learning, cognition and activities are mediated at least to some extent through vision [158]. By capturing the rays of light that come from the objects that populate our surroundings, our eyes transmit electrical signals to our brain in order to form a complete image of what we have in front of us. These images allow our brain to properly interpret our environment, which has been of capital importance for human evolution. For decades, researchers have recognized the powerful capabilities that vision provides and have been trying to replicate this process (from image capture to high-level semantic image interpretation) in order to design vision-based applications. The combined use of cameras and computers establishes the natural framework for developing such applications.

Among the existing vision-based approaches, *stereo vision* or *stereovisual* systems exhibit particularly interesting features that emerge from the fact of simultaneously getting two images from slightly different viewpoints. In particular, this setup allows easily inferring depth by correlating information from the overlapping area in both images.

On the other hand, *robotics* has become a prominent research field that pursues the design of devices (called *robots*) that can substitute, and even improve, human actions in certain tedious, repetitive, hard or even dangerous situations. Robots performance relies on a set of sensors employed to capture information from both the robot itself and the environment, in order to properly operate within it. This naturally leads to the use of vision systems on robots, specially on the so-called *mobile robots*, which pursue reliable, autonomous navigation. Thus, it can be said that vision-equipped mobile robotics has transformed the process of understanding captured images from an objective to a means to perform higher-level robotics applications.

As a consequence, the number of computer vision-based applications in robotics is constantly growing nowadays. This thesis contributes to this purpose, focusing on research towards truly autonomous robots that exploit stereovisual capabilities to localize themselves within their working environment.

1.1 Thesis Scope

This thesis addresses the combination of two popular technological fields, namely *mobile robotics* and *computer vision*, both of them experiencing a significant research effort from the scientific community nowadays.

In short, the former addresses the developing of robots that can autonomously move through their environment. Among other issues, estimating the position and orientation of a robot while it navigates is one of the most fundamental problems that mobile robotics must deal with. Traditionally relying on sensors such as sonars or laser scanners, mobile robotics solutions range from pure robot pose tracking to simultaneous localization and mapping (SLAM), passing through, for example, obsta-

cle avoidance, ego-motion estimation or global localization. Computer vision, in turn, encompasses a set of techniques employed to capture and process images in order to extract meaningful information from them. Image smoothing, keypoint detection or appearance-based image matching are simple examples of these techniques, but the collection of computer vision methodologies is constantly growing. The favorable characteristics of cameras, such as the high amount of information they provide or their increasing image quality while keeping a reduced cost, have contributed to the establishment of cameras as primary robotic sensors. In the last decades, the convergence of mobile robotics and computer vision has created a suitable framework with potential to lead us to the deployment of fully autonomous robots. The required background encompassing the most popular algorithms belonging to these disciplines is reviewed in chapter 2.

In particular, this thesis concerns about the use of stereo cameras as the only sensor to estimate a robot's position and orientation while simultaneously building a map of its environment. The so-called SLAM can be considered one of the most difficult problems for a mobile robot and represents a cornerstone when moving towards reliable autonomous navigation. The use of stereo images mitigates the inherent difficulties of monocular camera-based approaches regarding the estimation of scale factors, providing straightforward methods to get 3D information from the environment. As a downside, processing two images at every time-step incurs in a more expensive computational burden than monocular image-based approaches. Thus, the use of efficient and robust methods for extracting information from images is mandatory in favor of developing real-time SLAM solutions. In this sense, methods based on pixel comparisons have become a standard for fast image feature extraction and binary descriptors are becoming the state-of-the-art for keypoint matching.

On the other hand, several approaches and methodologies can be adopted to deal with visual SLAM, being *filtering* and *smoothing* the most popular tendencies nowadays. The former marginalizes previous information about the robot pose and represents the current state (which includes the robot pose and the map) by a probability distribution that is updated through time in an action-and-observation fashion. The latter, in turn, discards some of the previously captured information and computes, from scratch, the non-linear least-squares iterative solution for the map and all robot poses up to the current time-step. This thesis explores both frameworks and proposes a solution within each of them, as shown in chapter 4.

Special attention is devoted here to the estimation of the robot ego-motion between consecutive time-steps disregarding the map building. The so-called *visual odometry* can be employed as motion model in substitution of traditional encoder-based robotic odometry in filtering techniques, while it can be used as an initial estimation for smoothing methodologies. Both approaches are explored in this thesis and two new methods are presented in chapter 3.

In both visual odometry and SLAM applications, the presence of outliers (i.e. spurious data) within the information extracted from the stereo images represents a prominent problem since both filtering and smoothing approaches are highly sensitive to it, typically leading to unreliable results. This issue has been generally palliated

by hypothesis-and-verify techniques (among which the RANSAC approach has been the most representative) at the cost of sensibly increasing the computational burden. Nevertheless, alternative faster techniques can be applied with similar results, as the one presented as a part of one of our visual odometry proposals in chapter 3.

Finally, benchmarking of visual odometry and SLAM techniques requires computer vision datasets that provide a proper ground truth for the robot or camera movement. The collection of such publicly available datasets is becoming increasingly richer these years, and this thesis contributes with two new outdoor datasets described in chapter 5.

1.2 Thesis Contributions

The most relevant contributions of this thesis are:

- The development of a visual odometry system based on a closed-form solution for stereo cameras which is accurate, efficient, and does not exhibit convergence issues common to iterative approaches, under the assumption of outlier-free input data. In this work, we also test the suitability of adopting a linearized model to estimate the uncertainty of a 3D point given the errors in its corresponding image features. The experimental data prove similar performance to using more complex techniques such as the Unscented Transform (UT) or the scaled UT at a fraction of their computational cost. This work has been presented in [132].

- The introduction of a new method for detecting outliers based on robust kernels that can substitute standard quadratic error functions employed within iterative least-squares optimizations, as a fast alternative to RANSAC-based approaches. This method has been published in [134].

- The development of a complete stereovisual SLAM system relying on non-parametric Rao-Blackwellised particle filters that introduces a specific observation model for stereo observations based on SIFT descriptors that performs data association by marginalizing the observation likelihood among all the possible correspondences. This work was published in [133].

- The development of another stereovisual SLAM system based on a novel approach to Relative Bundle Adjustment, following the recently re-considered tendency of employing smoothing techniques to address visual SLAM. The contribution here consists of the design and implementation of a complete SLAM system built upon the integration of existing state-of-the-art techniques, as well as the experimental validation of the whole system. A journal paper is in preparation regarding this contribution.

- The publication of two collections of outdoor datasets within unmodified urban environments, comprising a heterogeneous combination of sensors including

monocular and stereo cameras, inertial measurement units, state-of-the-art GPS devices and laser scanners. Highly accurate GPS-based ground truth is also provided for the vehicle trajectory in our first collection, as well as a complete study of its uncertainty bounds, while standard GPS data is presented as ground truth for the second collection. Both datasets were published in [16] and [17].

Here is the list of all the publications derived from this thesis, with number of cites up to February 2015 (source: Google Scholar):

Journals

- *Francisco-Angel Moreno, José-Luis Blanco and Javier González*. **A complete visual SLAM system based on Sparser Relative Bundle Adjustment**, (In preparation).
- *José-Luis Blanco-Claraco, Francisco-Ángel Moreno-Dueñas, Javier González-Jiménez*. **The Málaga urban dataset: High-rate stereo and LiDAR in a realistic urban scenario**, International Journal of Robotics Research (IJRR), n.2, vol. 33, pp. 207-214, (2014). 4 citations. [17]
- *Jose-Luis Blanco, Francisco-Angel Moreno, Javier Gonzalez*. **A collection of outdoor robotic datasets with centimeter-accuracy ground truth**, Autonomous Robots (AR), n.4, vol. 27, pp. 327-351, (2009). 54 citations. [16]
- *Francisco-Ángel Moreno, José-Luis Blanco, and Javier González*. **Stereo vision-specific models for Particle Filter-based SLAM**, Robotics and Autonomous Systems (RAS), n.9, vol. 57, pp. 955-970, (2009). 23 citations. [133]

Conference proceedings

- *Francisco-Angel Moreno, Jose-Luis Blanco, Javier González-Jiménez*. **ERODE: An Efficient and Robust Outlier Detector and its Application to Stereovisual Odometry**, in IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe (Germany), pp. 4676-4682, (2013). 1 citation. [134]
- *Francisco-Angel Moreno, José-Luis Blanco and Javier González*. **An Efficient Closed-form Solution to Probabilistic 6D Visual Odometry for a Stereo Camera**, in Advanced Concepts for Intelligent Vision Systems, Delft (The Netherlands), pp. 932-942, (2007). 10 citations. [132]

1.3 Thesis Framework

This thesis is the culmination of 6 years of research work as a member of the MAPIR (MAchine Perception and Intelligent Robotics[1]) group of the System Engineering

[1] `http://mapir.isa.uma.es`

and Automation Department of the University of Málaga. This work has been partially funded by the FPU (Formación de Profesorado Universitario) program supported by the Spanish Education, Culture and Sport Ministry.

As a PhD student, the author successfully completed the doctoral program in Mechatronics Engineering coordinated by the System Engineering and Automation Department of the University of Málaga that provided the author with a solid basis on a combination of different disciplines (mechanical, electrical, control and computer engineering) which mobile robotics is heavily based on.

In addition, the author completed his academic education by participating in the 2010 BMVA Summer School of Computer Vision held in Kingston consisting of an intensive week of lectures and lab sessions covering a wide range of topics in Computer Vision and Digital Image Computing and lectured by researchers from the most active Computer Vision research groups in the UK. Moreover, some others PhD courses regarding Computer Vision topic have also been attended by the author at the Computer Science and System Engineering Department of the University of Zaragoza.

Besides, the author has carried out a three months research visit at the Department of Computer Science of the University of Bristol (UK) in 2011 under the supervision of Dr. Andrew Calway and Dr. Walterio Mayol-Cuevas [2] . During this stay, research was mainly focused on the use of different keypoint descriptors to perform data association. Furthermore, the author has performed an almost three months long stay in early 2013 at the University of Lincoln (UK) carrying out a research study about sensor fusion aimed to aid operation planning for agricultural sprayers. In this stay, cooperations with researchers Dr. Tom Duckett and Dr. Grzegorz Cielniak have been established.

Finally, it is important to remark that, during these years, several other collaborations have been carried out in different research projects developed within the MAPIR group related to the application of non-vision based sensors for mobile robots localization. These collaborations, although not being strictly anchored to the computer vision topic addressed in this thesis, are worth being mentioned as parallel research work of the author.

They include the following projects:

- **AGAVE**. This project aimed at the study of radio-based Ultra-Wide Band (UWB) as a potential technology to perform robot robust localization for unmanned mobile vehicles within warehouses. The combination of UWB and GPS sensors was also studied to enlarge the approach application possibilities. This project meant to the author a first contact with localization methodologies.

- **RoadBot**. Data fusion and vehicle localization were the core of this project where a electrical vehicle was employed to perform highly precise road inspection through the combination of scanner laser measurements and mm-GPS readings.

[2] `http://www.cs.bris.ac.uk/Research/Vision/Realtime/index.html`

- **Agrimap**. Similarly to RoadBot, Agrimap merged laser data and GPS measurements to inspect agricultural fields in front of industrial sprayers to forecast required boom movements.

The outcome in form of scientific papers related to these works are enumerated next:

Journals

- *Francisco-Angel Moreno, Javier Gonzalez-Jimenez, Jose-Luis Blanco, Antonio Esteban*. **An instrumented vehicle for efficient and accurate 3D mapping of roads**, Computer-Aided Civil and Infrastructure Engineering (CACAIE), n.6, vol. 28, pp. 403-419, (2013). 4 citations.

- *Javier González, José-Luis Blanco, Cipriano Galindo, Antonio Ortiz-de-Galisteo, Juan-Antonio Fernández Madrigal, Francisco-Angel Moreno and Jorge Martínez*. **Mobile Robot Localization based on Ultra-Wide-Band Ranging: A Particle Filter Approach**, Robotics and Autonomous Systems (RAS), n.5, vol. 57, pp. 496-507, (2009). 47 citations.

Conference proceedings

- *Francisco-Angel Moreno-Duenas, Grzegorz Cielniak, Tom Duckett*. **Evaluation of laser range-finder mapping for agricultural spraying vehicles**, in Towards Autonomous Robotic Systems (TAROS), Oxford (UK), pp. 210-221, (2013). 1 citation.

- *Javier González, Cipriano Galindo, Jose-Luis Blanco, Juan-Antonio Fernandez-Madrigal, Vicente Arevalo and Francisco-Angel Moreno*. **SANCHO, a fair host robot. A description**, in IEEE International Conference on Mechatronics (ICM), Seville (Spain), pp. 1-6, (2009). 8 citations.

- *Javier González, José-Luis Blanco, Cipriano Galindo, Antonio Ortiz-de-Galisteo, Juan-Antonio Fernández-Madrigal, Francisco-Angel Moreno and Jorge Martínez*. **Combination of UWB and GPS for indoor-outdoor vehicle localization**, in IEEE International Symposium on Intelligent Signal Processing, Alcala de Henares (Madrid), Alcala de Henares (Spain), (2007). 18 citations.

1.4 Thesis Structure

The remaining chapters of this thesis are organized as follows:

Chapter 2 provides a compilation of the main computer vision algorithms and methods applied throughout this work. This considers, for example, image feature detectors and descriptors, matching techniques or data association methodologies.

Also, a review on the most popular localization methods is presented, including probabilistic filters and state-of-the-art smoothing or bundle adjustment approaches. These compendiums, although non-exhaustive, form the required background on the topics addressed here and endorse the subsequent chapters in this thesis.

Chapter 3 presents two methods for estimating the camera ego-motion between consecutive time-steps, i.e. visual odometry. One of them is based on a closed-form solution that operates with 3D points observed from two different viewpoints, whilst the other is built on an iterative non-linear least-squares optimization procedure that relies on a robust kernel based on a pseudo-Huber cost function. The latter method also represents an alternative approach to RANSAC, coined ERODE, for fast outlier rejection. Experiments with real data to validate both methods are provided in this chapter.

Chapter 4 goes one step further and proposes two new approaches to deal with stereovisual SLAM. Our first proposal follows a probabilistic *filtering* scheme particularized in the so-called Rao-Blackwellized particle filter that defines a camera motion model based on the above-mentioned visual odometry closed-form solution and an observation model that relies on SIFT descriptors and the marginalization of the observations likelihood over all possible associations. In turn, our second presented method belongs to the *smoothing* category of solutions for visual SLAM and presents a complete SLAM system that encompasses a front-end based on ORB binary descriptors and a bag-of-words method for aiding data association, while the back-end employs keyframes in a blended solution, named Sparser Relative Bundle Adjustment, previously developed within the MAPIR group [15], in between global and pure relative bundle adjustment formulations. In this method, the non-linear least-squares optimization procedure implements our ERODE technique. Both proposals are supported by experimental results presented here.

Chapter 5 describes two outdoor datasets recorded in the city of Málaga which contain stereo images along with several other sensory data such as laser scanners, inertial measurement units or GPS devices. Our first collection includes six smaller outdoor datasets and provide a proper ground truth, i.e. full 6D vehicle pose, derived from three state-of-the-art RTK GPS receivers. Our second dataset has been gathered in an urban environment along a 36.8km trajectory featuring high-resolution stereo images among with other sensor data. Both datasets contribute to the existing collection of public repositories employed to benchmark mobile vehicle localization solutions.

Chapter 6 gives some conclusions of this thesis, summarizing the research work presented here and introducing new objectives for further exploration.

Chapter 2
Background

Overview

Building a map of the environment while localizing a camera within it by exclusively employing information extracted from the captured images as the camera moves is coined Visual SLAM. Several techniques must be combined to achieve this in a robust fashion, merging computer vision algorithms with localization methodologies generally applied in mobile robotics. This chapter provides the required background on the methods employed in this thesis.

Life's barely long enough to get good at one thing.
So be careful what you get good at. – Rust Cohle

2.1 Introduction

Robotics is experiencing nowadays a huge growth due to its strong potential to help human beings in several routine, difficult, or unsafe tasks. For years, robotic arms have been present in factories improving speed, accuracy and safety in assembly lines and they have even jumped to space where they performed hazardous tasks in the Space Shuttle. Assistance robots performing medical works are now a reality[1] , tele-operated robots for aiding elderly people[2] or deactivating explosives[3] are clear examples of robotics integrated into human's life. Even autonomous cars[4] are starting to become feasible and will be present in ours streets in the near future.

In particular, mobile robotics exhibits a variety of characteristics that make them especially interesting for developing practical applications. Providing a robot with the capability of freely traversing an environment opens a wide range of possibilities related with the development of more dynamic and complex tasks than those that a robotic arm, for instance, can perform.

Nevertheless, mobility inherently entails a larger complexity in robot operation since mobile robots must perform the work they were designed for, in parallel to other tasks exclusively related with their movement such as, among others, localization, obstacle avoiding or path planning. These difficulties become harder when the robots must move with complete or almost complete autonomy (i.e. with minimal human intervention) within a complex, often dynamic and unpredictable, environment.

Autonomous self-localization represents the first major problem that a robot must solve when operating, since having an accurate estimation of its position is mandatory in order to travel towards their target. In this sense, robust mobile robot localization has become a cornerstone in modern robotics and a huge research effort is being made in the technical community regarding this topic.

Mobile robot localization implies the presence of sensors that can provide information from both the environment and the robot itself. For years, the use of encoders in wheeled robots for estimating the robot ego-motion (denominated *odometry*) has been typically combined with other sensors, such as sonars [68, 102, 140] or laser scanners [46, 71] to gather information from the environment hence improving the robot pose estimation.

However, in recent years, cameras have become the most popular sensors in robotics due to a number of reasons, including: their irrefutable capability of providing an enormous amount of information about the environment, their continuous improvement in terms of image quality while simultaneously decreasing their price, and the growing computational capabilities of modern computers. A proof of this is the large part of the mobile-robotics literature that employs cameras as primary sensors in a variety of applications. Thus, cameras have been extensively used for

[1] http://www.davincisurgery.com/

[2] http://www.giraffplus.eu/

[3] http://www.frontline-robotics.com/RoboticsTechnology/TUGV.html

[4] https://plus.google.com/+GoogleSelfDrivingCars

localization in a variety of configurations: monocular [41, 164], stereo [2, 36, 88], trinocular [5], and even omni-directional cameras [39, 124, 188] have been applied for this task. Furthermore, depth-based sensors such as time-of-flight cameras [84, 123], or Kinect devices [33, 63] are becoming popular nowadays and represent an emerging trend within the world of robotic sensors. This reveals a close relationship between robotics and computer vision that has been built up for years, developing a significant set of solutions to the problem of robot localization.

In order to make this document more self-contained, first we formally introduce some relevant concepts that are indispensable to understand the work presented here. Therefore, and regarding the computer vision part, two main concepts emerge:

- The first step in most visual-based localization methods is to extract relevant information from the input images. In this sense, we can define an *image feature* (or just a *feature*) as any *distinctive* element within an image that provides some kind of useful information. Typically, corners and/or blobs are the most popular image features. In the case of corners, since they are defined by a single pixel, can be also named *keypoints*, *interest points* or even *corners* (if allowed by the context). The concept *image feature* may also encompass more complex structures such as lines, segments or circles, although these will not be considered in this thesis. On the other hand, there exist methods that address the so-called *dense stereo* problem and employ the input images to build a disparity map of the environment (typically by image correlation). Furthermore, some cameras even implement this feature at hardware level. However this topic goes beyond the scope of this thesis, which only focuses on *sparse* methods.

- Image features are usually augmented by a *descriptor* of their local area. They provide a manner to differentiate one image feature among the rest of them. Usually, descriptors are vectors, either of floats or binary elements, that summarize the appearance or the relevant characteristics of the keypoint neighborhood. Being computed through very different methods, a wide variety of descriptors can be employed ranging from the simplicity of an image patch surrounding the keypoint to more complex scale-and-orientation invariant descriptors such as SIFT [119] or SURF [6].

On the other hand, in terms of mobile robotics-related concepts, these terms need to be defined:

- The *pose* of a robot can be represented by a vector with six elements: $\mathbf{p} = (x, y, z, \alpha, \beta, \gamma)$. The first three elements represent the spatial coordinates of the robot whilst the rest defines its orientation through the Euler angles *yaw*, *pitch* and *roll*, respectively (shown in figure 2.1a). In general this thesis estimates the full 6D pose of the robot although in some applications only a planar representation of the space where the robot moves may be necessary. In those cases the robot pose reduces to $\mathbf{p} = (x, y, \alpha)$ with the *yaw* angle representing the robot orientation on the plane π (see figure 2.1b). However, it is important to note that

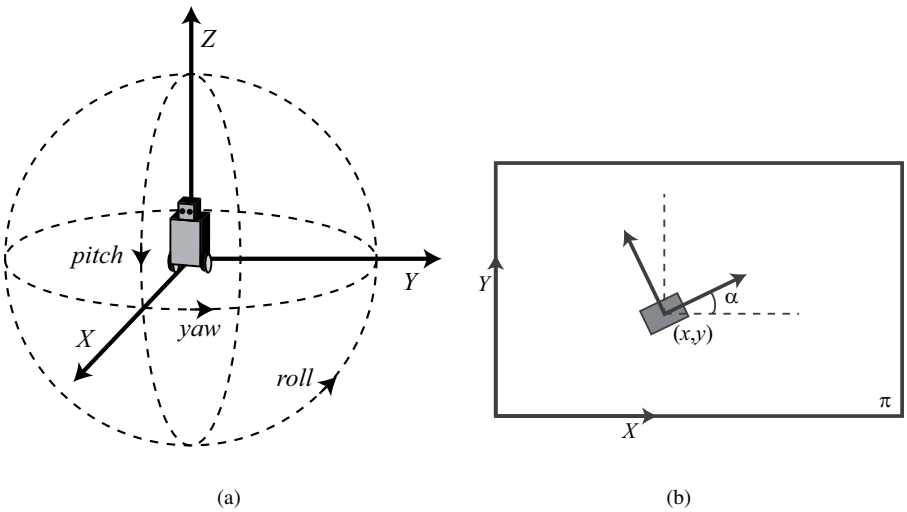


Figure 2.1: Representation of robot orientation with Euler angles in (a) 3D and (b) 2D.

the Euler angles mean just one way of representing the orientation of a rigid body (probably the most intuitive) but they suffer from some limitations. Unit quaternions, rotation vectors [45] or manifold-based transformations [14] can represent orientations in a more robust fashion although they require a more complex interpretation.

- A *landmark* can be defined as a stationary, characteristic element in the environment which can be easily and repeatedly detected by a sensor. In this thesis the landmarks are salient 3D elements with coordinates $\mathbf{X} = (X,Y,Z)^{\mathsf{T}}$ (either global or relative coordinates) that projects to stereo images. Furthermore, for map building applications, we can augment the definition of the landmark with a vector descriptor which makes each one distinguishable among the complete set of detected landmarks.

- The representation that a robot has from its surrounding space is denoted by the term *map*. In this work, a map is composed by a set of landmarks with estimated positions and, sometimes, also other properties such as a descriptor.

- *Localization* is the process of estimating both the position and orientation of a robot (which jointly are denoted as the *pose*) within a reference coordinate system. At this point it has to be highlighted that in this work we will not distinguish between a robot or a camera in terms of the object to be localized. Although the proposed methods here are aimed to localize a robot, it does not exist any difference with the problem of localizing a camera undergoing the same movements as the robot. Thus, both *camera* and *robot* terms will be interchangeably-used to refer to the localization subject.

- The *state* of the problem is defined as the set of all the variables that must be estimated to solve the localization problem. Depending on the application, the state may comprise only the robot pose, or also contain the estimation of a map (visual localization or visual SLAM, respectively). If the pose is only incremental, the problem is referred as visual odometry. These problems will be further explained later.

- A *sensor* is a device capable of detecting or measuring certain qualities or physical phenomena from the environment such as, for example, light, speed, color, range to objects, position, etc. Sensors can be categorized according to the kind of information they provide:

  - *Proprioceptive* sensors directly measure qualities related to the robot egomotion such as speed, acceleration or displacement. Under this category we find encoders, tachometers, accelerometers and gyroscopes. The main drawback of these sensors is that their errors are cumulative since no other references apart from the robot itself are taken into account.
  - *Exteroceptive* sensors include all those that gather information from the environment and not from the robot. Examples of this category are laser scanners, sonars and, of course, cameras. Information from these sensors refines the initial estimation of the robot pose computed according to the proprioceptive sensory data. Thereby, a combination of these kinds of sensors are typically present in most mobile robots.

  As a kind of analogy, think about a man walking within a room. Localizing himself with only proprioceptive sensors would be equal to walk with his eyes closed, trying to estimate his position by only counting his steps and using his balance system (placed in his inner ear) to determine his orientation. Exteroceptive sensors would allow him to open his eyes, hear noises and use his hands to sense the environment, hence highly enriching his sensory input and significantly easing his localization process.

- *Uncertainty* is present in every mobile robotics system in forms of unreliable estimations of the robot pose and the landmarks in the map. Multiple factors contribute to the system uncertainty such as the physical limitations that affect the sensors (e.g. laser scanners have limited range, cameras provide only finite-resolution images, etc.), inaccuracies in the models employed to simulate the processes that the robot undergoes, hardware malfunctions and inaccuracies related to robot motors, or limitations of the mathematical algorithms implemented in robotics software.

  All these sources of error contribute to increase the uncertainty about the estimated robot pose and map (if needed). Therefore the system state is usually modeled by probability distributions so that the localization problem can be solved in the most reliable manner by means of probabilistic methods.

Now, as a brief survey on robot localization and mapping, we can define three variants of the problem with increasing difficulty levels:

- *Position tracking*. Under the assumption that an initial, approximate estimation of the robot pose is available, we want to track the robot pose using sensory data as it moves along time. Some works that deal with this issue can be found in [168, 203, 208].

- *Global localization*. No initial estimation of the robot pose is available in this case, but the robot keeps a complete pre-built map of its environment. Thus, it will gather sensory data and will employ the map to estimate its pose within it. Typically, probabilistic solutions have been applied to solve this localization problem such as multi-hypothesis Kalman filters [34,163], Markov localization processes [22, 59], and particle filters [58, 190].

- *SLAM*. The *Simultaneous Localization And Mapping* problem does not assume any knowledge about the robot environment, but instead deals with building a new map from sensory data while simultaneously estimating its pose within the partially-built map itself. Finding a robust, reliable solution to this complex problem means the cornerstone for the deployment of fully autonomous mobile robots.

  Another concept related to SLAM of paramount importance for this thesis is the so-called *loop closure*, defined as the re-observation of previously detected landmarks that where unseen during traversing a path (the loop), meaning that the robot is re-visiting an already explored area.

It is the capital importance of this last problem what motivated the work presented here. Thus, this thesis focuses on the development of methods and algorithms to contribute to the use of stereo cameras both to perform visual odometry and to solve the full visual SLAM problem.

Please, refer to figure 2.2 where it is shown our proposed scheme for a complete system that performs visual odometry and visual SLAM with a stereo camera as the only sensor. This main scheme sketches all the steps needed to convert a pair of input images from two slightly different viewpoints to a robust, accurate estimation of the robot pose and (if desired) a featured-based representation of the robot environment. Most of the concepts within the blocks displayed in the figure will be described in detail throughout this thesis.

Two main aspects can be highlighted in this scheme:

- The scheme has two clearly separated parts, named *front-end* and *back-end*. The first part includes all the procedures that are strictly related to computer vision: i.e. image pre-processing, feature detection and description, keypoint matching and data association. The back-end, in turn, comprises the estimation of both the robot pose and the map.

  The link between those parts falls on the block that contains the *visual features* and the *observations*. The front-end generates a set of features that are already

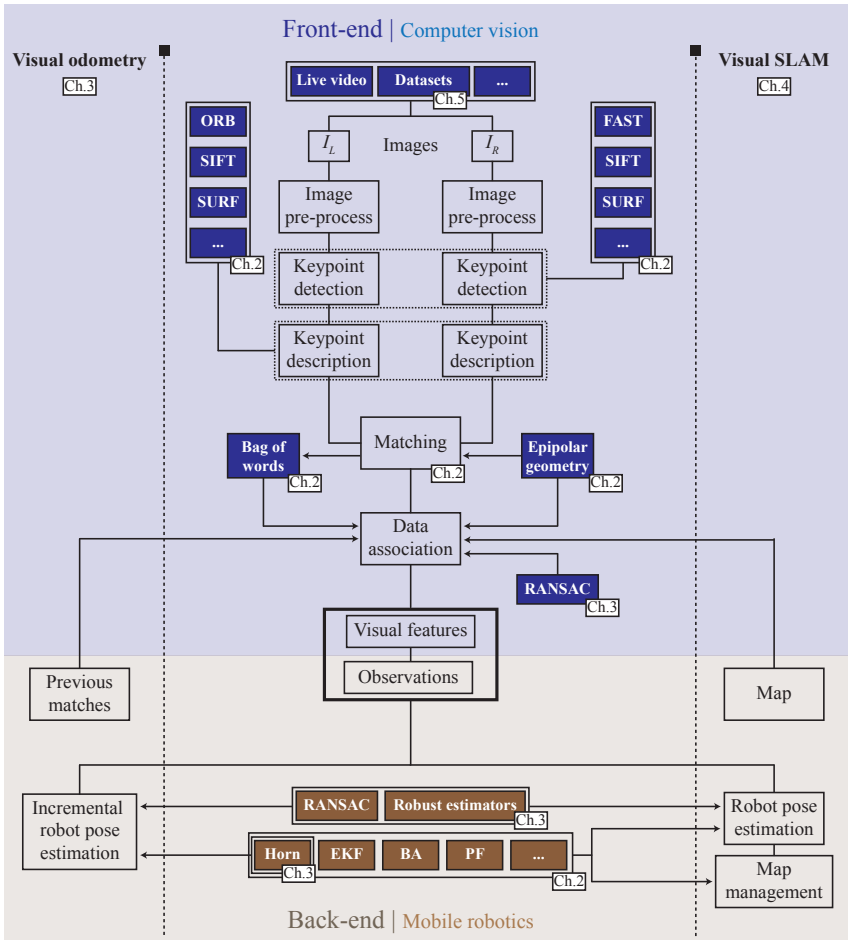


Figure 2.2: Our proposed main scheme for visual odometry and visual SLAM. The small white labels indicate the chapters within this work that address the corresponding concept.

associated to other previously detected ones, including null-associations, i.e. new, not already observed features. These features correspond to the current observation that the camera has captured from its environment, and defines the input of the estimation procedure performed by the back-end. In particular, when using stereo cameras, as in this work, the front-end output is formed by *visual* features. Nevertheless, different front-ends (built upon sensors distinct from cameras, for instance) might be employed to feed the back-end part. The only requirement needed is the existence of a relationship between the features sensed by the robot and the system state so that it converts such front-end output into a set of observations. Then, the same methods can be used in this part to achieve the localization goal although, of course, the methods must be adapted to the kind of map that the sensory data produces.

Note that this scheme slightly moves away from the pose-graph SLAM definition of front-ends and back-ends [107, 187]. In such approaches, the SLAM problem is represented by connected graphs that are built by the front-end. On the other hand, the back-end is in charge of optimizing it, typically by means of nonlinear least-squares optimization methods. Here, in turn, we propose a more general division of the visual odometry and SLAM systems where the optimization process of the back-end can be indistinguishably implemented by bundle adjustment methods or probabilistic filters.

- The complete visual SLAM approach and the much simpler visual odometry problem share several common tasks. In fact, their difference is twofold: (i) visual odometry only cares about incremental changes in pose between time steps while visual SLAM keeps a complete estimation of the robot path and (ii) map management is disregarded by visual odometry while is a keystone in visual SLAM. These differences mean a huge step between both approaches, enormously increasing the complexity of the solution.

Following this scheme, we first deal in the next section with the most relevant aspects of the computer vision-related front-end, whilst we subsequently address the algorithms and techniques employed to perform robot localization that constitute the core of the back-end.

2.2 Front-end (Computer vision)

Here we address our proposed front-end for the visual odometry and visual SLAM applications which, in essence, deals with the methods and algorithms strongly related to computer vision. As stated before, the front-end extracts relevant information from the captured stereo images, associating it with the current knowledge of the system state, hence providing the input that our proposed back-end demands.

Thus, this section deals with keypoint detectors and descriptors, matching methods to look for correspondences between keypoints detected in different images, a brief introduction to epipolar geometry that may assist during keypoint matching (being specially useful in stereo vision applications) and a method for creating a bag of binary words to aid data association procedures.

Regarding the notation, in this thesis we will indistinctly call *keypoints*, *image features*, *visual features* or even *corners* (when applicable) to those salient elements in the image that can be extracted by feature detectors. In general, desirable characteristics of features are repeatability and robustness against changes in the image such as rotations, scale, illumination, etc. since it is of the maximum importance that they can be robustly detected in different images of the same scene.

Usually, before detecting keypoints, images pass through a pre-processing stage where they are typically smoothed by applying, for example, a Gaussian kernel in order to reduce the effect of noise. In addition, other techniques might be also employed to further prepare the image to suit the chosen feature detector such as, for example, converting the input into integral images [6] or building image pyramids [119]. In any case, such pre-processing stage is not addressed in this thesis.

2.2.1 Feature detectors and descriptors

This section presents some of the most employed image feature detectors in computer vision. It does not intent to be an exhaustive list with a thorough description (for that, please refer to [196]), but an informal reference guide which depicts the most important aspects of each method. Most of these methods are employed later in the works presented in this thesis while others have been included here for its relevance.

The methods will be presented here in chronological order and classified into one of three categories: detectors (Harris [82], Shi&Tomasi [173] and FAST [162]), descriptors (BRIEF [23]) and detector-and-descriptors (SIFT [119], SURF [6] and ORB [165]).

Harris' Corner Detector (1988)

One of the most widely employed methods for extracting image features was developed by Harris y Stephens [82] in 1988, and is based on the self-correlation function of the image. Formally, let $I(x,y)$ be a grayscale image with $\mathbf{x} = (x,y)$ being a cer-

tain point on the image, and let $(\Delta x, \Delta y)$ be a displacement on the image. Then, the self-correlation function is defined as:

$$c(x,y) = \sum_W \left[I(x_i,y_i) - I(x_i+\Delta x, y_i+\Delta y) \right]^2 \tag{2.1}$$

where $\mathbf{x}_i = (x_i, y_i)$ are pixels within a Gaussian window W centered in $\mathbf{x}$. If we approximate the displaced image by its first order Taylor expansion, we get:

$$\begin{aligned} I(x_i+\Delta x, y_i+\Delta y) &\approx I_T(x_i+\Delta x, y_i+\Delta y) \\ &= I(x_i,y_i) + \begin{pmatrix} I_x(x_i,y_i) & I_y(x_i,y_i) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \end{aligned} \tag{2.2}$$

with $I_x(x_i,y_i)$ and $I_y(x_i,y_i)$ representing the partial derivatives of the image centered on $\mathbf{x}_i$ for the x and y variables, respectively. By replacing equation (2.2) in equation (2.1) we come to the following expression of the self-correlation function:

$$\begin{aligned} c(x,y) &\approx \sum_W \left[I(x_i,y_i) - I_T(x_i+\Delta x, y_i+\Delta y) \right]^2 \\ &= \sum_W \left(I(x_i,y_i) - I(x_i,y_i) - \begin{pmatrix} I_x(x_i,y_i) & I_y(x_i,y_i) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right)^2 \\ &= \sum_W \left(- \begin{pmatrix} I_x(x_i,y_i) & I_y(x_i,y_i) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right)^2 \\ &= \sum_W \left(\begin{pmatrix} I_x(x_i,y_i) & I_y(x_i,y_i) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right)^2 \\ &= \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \begin{pmatrix} \sum_W (I_x(x_i,y_i))^2 & \sum_W I_x(x_i,y_i) I_y(x_i,y_i) \\ \sum_W I_x(x_i,y_i) I_y(x_i,y_i) & \sum_W \left(I_y(x_i,y_i) \right)^2 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \\ &= \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \mathbf{M}(x,y) \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \end{aligned} \tag{2.3}$$

Matrix $\mathbf{M}(x,y)$ comprises the intensity information of the local neighborhood of the (x,y) point and it can be expressed by:

$$\mathbf{M}(x,y) = \begin{pmatrix} \left\langle I_x(x_i,y_i)^2 \right\rangle & \left\langle I_x(x_i,y_i) I_y(x_i,y_i) \right\rangle \\ \left\langle I_x(x_i,y_i) I_y(x_i,y_i) \right\rangle & \left\langle I_y(x_i,y_i)^2 \right\rangle \end{pmatrix} \tag{2.4}$$

with angle brackets denoting summation over the window W.

In practice, in order to compute the horizontal and vertical derivatives of the image, we can simply convolve it with the following $\mathbf{m_x}$ and $\mathbf{m_y}$ kernels:

$$\mathbf{m_x} = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \quad \mathbf{m_y} = \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} \tag{2.5}$$

The analysis of the two eigenvalues of $\mathbf{M}$ in each image point (λ_1 and λ_2) classifies the pixel in one of these three categories.

- If both eigenvalues are high, a small displacement in any direction on the image will yield a large change in the image gray level, revealing this pixel as a *corner*.
- If one of the eigenvalues is high while the other remain low, a displacement in one direction will produce a small change in the self-correlation function whilst a big one in the orthogonal direction, indicating that the pixel is on a *border*.
- Finally, if both eigenvalues are small, any displacement in any direction will produce a small change in the self-correlation function, which is an indicative that the pixel belongs to a nearly constant gray level area in the image.

In order to avoid the explicit computation of the eigenvalues, Harris proposed an alternative function to detect corners:

$$\begin{aligned} R &= \det(\mathbf{M}) - k\,\mathrm{trace}(\mathbf{M})^2 \\ &= \left\langle I_x^2 \right\rangle \left\langle I_y^2 \right\rangle - \left(\left\langle I_x I_y \right\rangle \right)^2 - k \left(\left\langle I_x^2 \right\rangle + \left\langle I_y^2 \right\rangle \right)^2 \end{aligned} \tag{2.6}$$

where k is variable parameter with a suggested value of $k = 0.04$ in the original paper. Note that the notation for the specific point (x_i, y_i) has been dropped for convenience.

Local maxima of the R function will determine the position of the corners found by the Harris detector while sub-pixel precision can be achieved by a quadratic approximation of the function on the corner surrounding.

The main advantage of the Harris detector can be found on its small computational cost while maintaining a reasonable rate of repeatability, turning it, for years, into one of the first options for real-time applications. More recently, some methods based on binary comparisons have been developed that find a significantly larger amount of corners and operate at a fraction of Harris detector's computational time. The FAST method is probably one of the more representative and will be explained in section 2.2.1.

Shi&Tomasi (1994)

Shi&Tomasi detector (also known in the technical literature as the Kanade-Lucas-Tomasi or KLT detector) was developed in 1994 [173] and is highly related to the concept of *optical flow* [7] and the Harris method since it also works on the image self-correlation function.

In [173] the authors show that the minimum eigenvalue of the $\mathbf{M}$ matrix, ($\min(\lambda_1, \lambda_2)$) is a better measurement of a corner quality than the R value proposed by Harris when the images suffer from affine transformations.

These corners, in addition, are also reported to have characteristics that make them specially suitable for being tracked in subsequent images and a method for performing such tracking was also presented in [173].

This method assumes that images taken at similar time steps are highly related between them since they are images of the same scene taken from very similar points of view. This property is reflected in equation:

$$I(x,y,t+\tau) = I\left(x-\Delta_x, y-\Delta_y, t\right) \tag{2.7}$$

where I is the grayscale image defined by two parameters: the corner position $\mathbf{x} = (x,y)$ and the time where the image was recorded t. The displacement of the corner position between the times t and $t+\tau$ is determined by $d_{\mathbf{x}} = \left(\Delta_x, \Delta_y\right)$ and is a function of $\mathbf{x}$, t and τ.

Due to image noise, it is hard to determine $d_{\mathbf{x}}$ for an unique pixel (unless its gray level is very different to their nearby pixels). Therefore, the KLT method tracks a window of pixels (or patch) which has been selected so that it has enough texture to be distinguishable.

However, tracking a patch implies that the pixels within may behave differently and move in a different fashion between consecutive images. To cope with this, the similarity of the tracked windows is checked between images, discarding the patch if the difference is significant. However, there is still the problem of assigning an unique displacement $d_{\mathbf{x}}$ for the whole patch even though not all the pixels within it move equally. This could be mitigated by defining affine transformations between patches instead of mere translations. The KLT method, though, tries to select patches as small as possible and models the differences due to the affine transformations as an error that has to be minimized by selecting the best possible displacement vector. Thus, we have:

$$J(\mathbf{x}) = I(\mathbf{x} - d_{\mathbf{x}}) + n(\mathbf{x}) \tag{2.8}$$

where, for convenience, $J(\mathbf{x})$ stands for $I(x,y,t+\tau)$ and $I(\mathbf{x}-d_{\mathbf{x}})$ for $I(x-\Delta_x, y-\Delta_y, t\Big)$, being $\mathbf{x}$ the coordinates of a certain pixel and $n(\mathbf{x})$ the noise that affects such pixel.

The displacement vector $d_{\mathbf{x}}$ is chosen so that it minimizes the error defined by the summation over the window:

$$\epsilon = \sum_W w\,[I(\mathbf{x} - d_{\mathbf{x}}) - J(\mathbf{x})]^2 \tag{2.9}$$

where w represents a weighting function. For convenience, and without loss of generality, we assume a constant weighting function, hence dropping the term in the following formulation although, typically, Gaussian kernels are employed.

If we assume that $d_{\mathbf{x}}$ is small, we can approximate the image $I(\mathbf{x}-d_{\mathbf{x}})$ by its first order Taylor expansion:

$$I(\mathbf{x} - d_{\mathbf{x}}) \approx I(\mathbf{x}) - G d_{\mathbf{x}} \tag{2.10}$$

with G comprising the spatial derivative of the image in the considered window. So, expression 2.9 can be rewritten as:

$$\epsilon = \sum_W [I(\mathbf{x}) - G d_{\mathbf{x}} - J(\mathbf{x})]^2 = \sum_W (H - G d_{\mathbf{x}})^2 \tag{2.11}$$

where $H = I(\mathbf{x}) - J(\mathbf{x})$.

To minimize ϵ, we derive the expression 2.11 with respect to $d_{\mathbf{x}}$ and set it to equal zero:

$$\sum_W (H - Gd_{\mathbf{x}})(-G) = \sum_W -HG + \sum_W Gd_{\mathbf{x}}G = 0 \tag{2.12}$$

Since, in this case, $Gd_{\mathbf{x}}G = GG^{\mathrm{T}}d_{\mathbf{x}}^{\mathrm{T}}$ and $d_{\mathbf{x}}$ is assumed constant for the whole patch, we come to:

$$\left(\sum_W GG^{\mathrm{T}}\right) d_{\mathbf{x}}^{\mathrm{T}} = \sum_W HG \tag{2.13}$$

which builds a system of two equations with two unknowns (the components of $d_{\mathbf{x}}$):

$$\mathbf{G}d_{\mathbf{x}}^{\mathrm{T}} = \mathbf{e} \tag{2.14}$$

Please note that **G** contains the information of the image gradient and coincides with the matrix **M** in equation (2.4) demonstrating the strong relation between this method and the Harris corner detector. On the other hand **e** contains the difference between the considered patches.

Finally, we must explain the KLT method's criteria to select a certain window of pixels in the image, i.e. which points we should select to track them. This way a patch will be a good candidate if its associated system of equations 2.14 can be solved in a robust and reliable way. This condition is fulfilled if the matrix **G** is over the noise level (which happens when its eigenvalues λ_1, λ_2 are large enough) and if its well-conditioned (which is achieved if the difference between its eigenvalues is relatively small). Both conditions are satisfied when the center of the patch is a corner in the image since both eigenvalues of **G** are large.

In practice, if the smallest eigenvalue is greater than a certain threshold, matrix **G** is well-conditioned and, therefore, the Shi&Tomasi (or KLT) method considers a pixel in the image as a good feature to track if:

$$\min(\lambda_1, \lambda_2) > \lambda_{th} \tag{2.15}$$

where, usually, the threshold λ_{th} is empirically determined.

SIFT: Scale-Invariant Feature Transform (2004)

David Lowe presented in a seminal paper [119] an algorithm to detect distinctive image features which are invariant against scale and rotation changes. Unlike the Harris and Shi&Tomasi methods, SIFT is a blob detector, i.e. it doesn't not have high response at corners in the image but in blob-like structures. In addition to the detection, the SIFT method also proposes a local descriptor based on the histogram of orientations of the feature neighborhood, providing detected features with a highly distinctive signature that allows matching them with other views of the same feature. The descriptor, besides the scale and orientation invariance, presents partial invariance to changes in illumination and points of view.

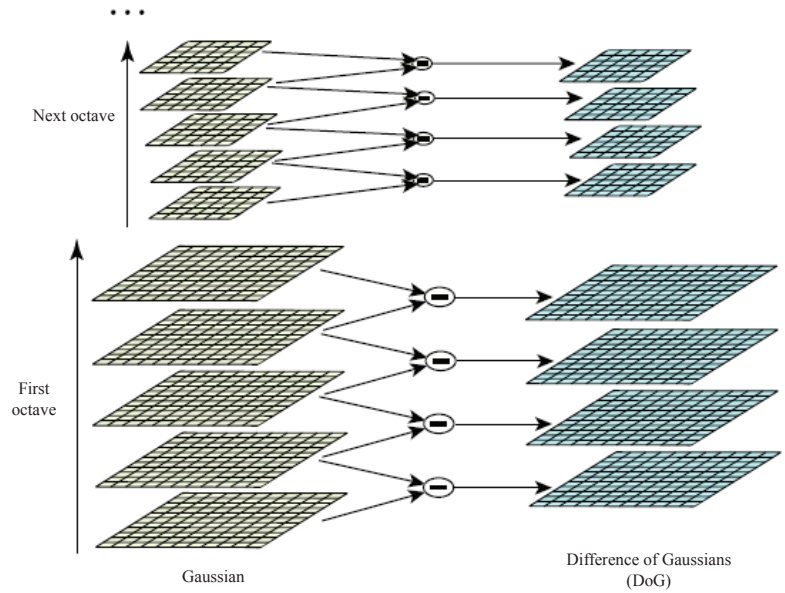


Figure 2.3: Scale space and Difference of Gaussians operation (DoG) (figure adapted from [119]).

For years, SIFT has been the most powerful of the feature detectors and descriptors due to its robustness and performance when matching features that undergo different transformations. However, its computational burden is significant, sometimes rendering it useless when dealing with real-time applications. Although some SIFT-based methods have been reported dealing with this issue [100], their performance is usually affected.

The original Lowe's method starts by searching for stable points through the different scales of an image, aiming to provide detected features with a scale invariance. For this purpose, a continuous function of the scale (known as scale-space) $L(x,y,\sigma)$ is built from the convolution of a Gaussian variable $G(x,y,\sigma)$ with the input image $I(x,y)$:

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y) \tag{2.16}$$

where * is the convolution operator in x and y and:

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{2.17}$$

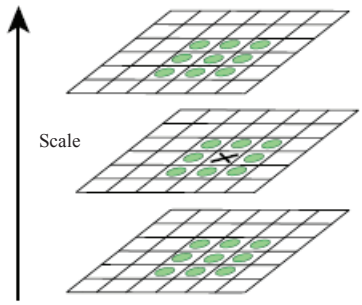


Figure 2.4: Search for extrema in the scale-space and neighbor points to be compared for each considered point (figure taken from [119]).

In order to find stable interest points in the scale-space, SIFT searches for local extrema (both minima and maxima) of the Difference of Gaussians (DoG) function convolved with the image $D(x,y,\sigma)$:

$$\begin{aligned} D(x,y,\sigma) &= (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) \\ &= L(x,y,k\sigma) - L(x,y,\sigma) \end{aligned} \tag{2.18}$$

where k is a constant factor that sets the distance between the involved scales and which in Lowe's paper takes a value of $\sqrt{2}$. The construction of the scale-space and the DoG operation are shown in figure 2.3. The input image is repeatedly convolved with Gaussians to create a set of images with a distance of k in the scale-space. Once an octave is complete, the image is half-scaled and the process repeated. Each octave is composed by a set of s images so that $k = 2^{1/s}$.

In order to detect local extrema of the $D(x,y,\sigma)$ function, each point in the image is compared against its twenty six neighbor: eight in the same scale, nine in the upper one and nine in the lower one. Thus, a point will be considered as a potential interest point if it is higher (or lower) than all its neighbor (refer to figure 2.4). At this step, the detected scale is assigned to the found interest point. Next, the interest point orientation is computed from the gradient image. Thus, for each point $\mathbf{x} = (x,y)$ in the smoothed image (L) at the same scale of the interest point, the magnitude ($m_{\mathbf{x}}$) and orientation ($\theta_{\mathbf{x}}$) are computed through:

$$\begin{aligned} m_{\mathbf{x}} &= \sqrt{\left(L_{x+1,y} - L_{x-1,y}\right)^2 + \left(L_{x,y+1} - L_{x,y-1}\right)^2} \\ \theta_{\mathbf{x}} &= \arctan\left(\left(L_{x,y+1} - L_{x,y-1}\right) / \left(L_{x+1,y} - L_{x-1,y}\right)\right) \end{aligned} \tag{2.19}$$

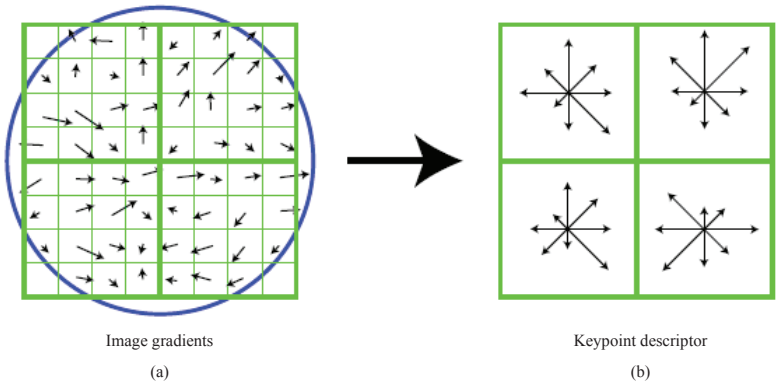


Figure 2.5: Descriptors and gradients. (a) Gradient of the interest point's neighborhood (size 8×8). The length and angle of the arrow are set according to the gradient magnitude and orientation, respectively. The displayed circle represents the Gaussian weighting window. (b) Orientation accumulation in larger areas (2×2 areas of size 4×4 pixels each) in histograms determining the values of the descriptor (figure taken from [119]).

Figure 2.5(a) shows the gradient at each point of the image represented by an arrow whose length is proportional to its magnitude while the angle indicates its orientation.

The orientations of the feature's neighbor pixels (within a circular window) are accumulated in an histogram of 36 bins covering the whole 360° range of possible orientations. Each value added to the histogram is weighted according to both the gradient magnitude and a Gaussian function with a value of σ equal to three times the one associated to the interest point scale. The interest point's main orientation is determined by the histogram maximum, whose value is refined by fitting a parabola passing through the maximum value and those in the contiguous bins. If there is any other bin with more than 80 % value of the maximum, another interest point with the same position and scale, but different main orientation, is created.

Finally, interest points are provided with a distinctive descriptor, computed taking into account the main orientation, gaining in this way invariance to orientation changes. The descriptor is computed as follows.

First, the magnitude of the gradient in nearby pixels is weighted with a Gaussian function with a σ equal to half the descriptor width (the Gaussian window is shown in figure 2.5a as a circle). Next, the orientations in nearby pixels are accumulated into discrete orientation histograms so that for every pixel in the neighborhood, a certain value is added to the affected bins in the histogram, taking into account the gradient angle and magnitude of the involved pixels. Figure 2.6 shows a plain example of the orientation histogram accumulation.

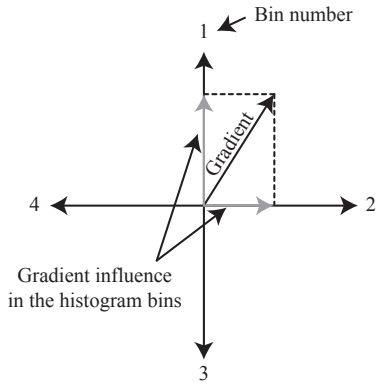


Figure 2.6: Example of orientation histogram accumulation with 4 bins. The gradient of the pixel affects the histogram bins 1 and 2 with different values, according to both its magnitude and proximity to the bin center.

The descriptor dimensionality is given by the expression $N \times N \times B$, where B is the number of orientations considered in the histograms (i.e. the number of bins). Figure 2.5 shows, as an example, an interest point's neighborhood of 8×8 pixels. From that we get 2×2 histograms with 8 bins each resulting in descriptors with size $2 \times 2 \times 8 = 32$. However, in [119] it is stated that the best results are achieved with descriptors of size $4 \times 4 \times 8 = 128$.

Finally, a normalization is performed in order to reduce the influence of illumination changes in the final descriptor. Changes in the image contrast are equivalent to the multiplication of a constant value with all the pixels in the image, getting their gradients also multiplied by the same factor. By normalizing the descriptor so that it has norm 1 we get some invariance to changes in contrast. Similarly, changes in the image brightness implies the addition of a constant value to all the image pixels. However, the gradient is invariant to this issue since it is computed from the subtraction of adjacent pixels. Finally, non-lineal changes in the illumination that affects the images (e.g. camera saturation, different illumination changes for different surfaces in the scene, etc.) affect more significantly to the gradient magnitude but are not so severe to the gradient orientation. Then, the descriptor values are saturated according to a certain threshold (Lowe proposed a value of 0.2) and, subsequently, re-normalized to norm 1 in order to reduce the influence of the gradient magnitude into the orientation histogram.

FAST (2006)

The FAST (Features from Accelerated Segment Test) detector, presented in [161], is a machine learning-based method for high-speed corner detection which has become

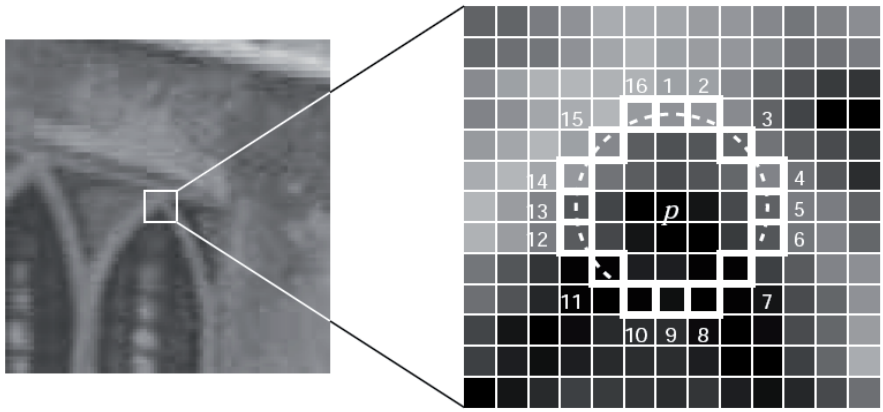


Figure 2.7: FAST segment test example for a certain pixel p, which is the center of the Bresenham circle highlighted with white squares. The dashed line passes through 12-pixels length arc ($n = 12$) which are all brighter than p by more than the tolerance t (figure taken from [161]).

one of the most widely employed detectors since its appearance in 2006. Even though the detected keypoints are not as stable as the ones extracted by the SIFT or Harris algorithms, its computationally efficient performance has made this method a suitable candidate to be employed in many real-time computer vision applications as, for example, in [103].

It is important to note that the FAST method is only a detector, so it does not provide any descriptor for the extracted features. Hence, if one is needed, it must be employed along with another method that computes any of the different keypoints descriptors existing in the literature such as, for example, SIFT (already explained in section 2.2.1, SURF [6] or BRIEF descriptors (refer to section 2.2.1).

The basis of the FAST detector is the so called segment test criterion which consists on computing the gray level difference between a certain pixel and a Bresenham circle of 16 pixels around it. Thus, a certain pixel p in the image is considered to be a corner if there exists a set of n (with $n \geq 9$, typically) contiguous pixels in the circle with a gray level all higher or lower than the one in p with a defined tolerance t (refer to figure 2.7). These contiguous pixels are called the *arc*.

The full segment test implies the comparison of all the 16 pixels in the circle with the corner candidate but by performing an *accelerated* segment test we can quickly exclude a very large number of non-corners. This is carried out by simply comparing p with the four pixels at positions 1, 5, 9 and 13 we can determine if the candidate is not a corner. Note that if p is a corner, at least three of those pixels must fulfill the gray level difference criterion. Otherwise, p is not a corner. Once this fast rejection step is performed, the full segment test is performed with the remaining candidates.

This algorithm, which is in fact the original FAST corner detector, was also further developed in [161] by using machine learning techniques in order to overcome the main weaknesses of the original method: its dependence on the comparison order for the circle pixels and the absence of non-maximal suppression. Thus, they employ the full segment test in a training set of images to build a decision tree that correctly classifies all the corners present in those images and minimizes the number of comparisons needed to determine the presence of a corner.

In addition to that, a small modification of the Sum of Absolute Differences (SSD) between the arc pixels and the corner is employed to measure its strength:

$$V = max\left(\sum_{\mathbf{x}\in P_b}\left|I_{\mathbf{x}} - I_p\right| - t, \sum_{\mathbf{x}\in P_d}\left|I_p - I_{\mathbf{x}}\right| - t\right) \tag{2.20}$$

where $I_{\mathbf{x}}$ stands for the intensity level of the pixel $\mathbf{x}$, P_b is the set of the arc pixels that are brighter than p plus the tolerance while P_d is the set of darker (also with the tolerance) arc pixels. The V score is subsequently used to perform non-maximal suppression.

BRIEF (2010)

The BRIEF (Binary Robust Independent Elementary Features) descriptor [23] is a binary string built from a set of pairwise comparisons between pixel intensities. Thus, given a smoothed image patch $\mathbf{p}$ of size $S \times S$, an intensity test τ is defined as:

$$\tau(\mathbf{p};\mathbf{x},\mathbf{y}) := \begin{cases} 1 & \texttt{if } I_{\mathbf{x}} < I_{\mathbf{y}} \\ 0 & \texttt{otherwise,} \end{cases} \tag{2.21}$$

where $I_{\mathbf{x}}$ is the gray level of a certain pixel $\mathbf{x}$ in the patch. From a set of n_d wisely chosen locations pairs, the BRIEF descriptor is built through

$$f_{n_d}(\mathbf{p}) := \sum_{1\le i\le n_d} 2^{i-1}\tau(\mathbf{p};\mathbf{x},\mathbf{y})\,. \tag{2.22}$$

The binary nature of the BRIEF descriptor allows a high-speed matching process by employing the Hamming distance (refer to section 2.2.2), which can be performed very fast on modern computers. In addition to this, since it is based on pixel comparisons, its computational burden is very low when being built, in a way that somehow resembles the process of finding FAST features.

There are only two aspects that must be taken into account about its configuration: the size of the smoothing Gaussian kernel and the distribution of the pixel location pairs. For this purpose, a set of tests with the publicly available computer vision dataset [126] is carried out in [23]. The performed experiments yield the best results by selecting a Gaussian kernel with a variance of value 2 and a Gaussian distribution $N\left(0, \frac{1}{25}S^2\right)$ for the locations of the pixel pairs, with S the size of the patch.

The BRIEF descriptor has also been reported to yield high recognition rates despite its simplicity and speed, even outperforming SURF based descriptors in many situations.

ORB (2011)

The ORB (Oriented FAST and Rotated Brief) detector [165] takes advantage of the combination of two previous techniques: the FAST detector and the BRIEF descriptor, both of them explained above. Furthermore, it deals with the principal shortcomings that these approaches suffer from.

The FAST method has become the state-of-the art corner detector because of its low computational burden but there are some issues that FAST does not address in comparison with other approaches, namely the keypoint scale and orientation. For the scale issue, the ORB method builds a scale pyramid of images and searches for FAST features in all the levels. In order to keep the strongest keypoints, the Harris score (refer to section 2.2.1) is employed to filter out the weakest ones in all the scales since they are prone to be lost under small changes in the image. In addition to that, the ORB approach computes an efficient estimation of the keypoint main orientation called the *intensity centroid* [160]. For this purpose, the moments m_{10} and m_{01} of the patch around the keypoint placed at pixel $\mathbf{x}$ are computed

$$m_{pq} = \sum_{x,y} x^p y^q I_{\mathbf{x}}, \tag{2.23}$$

so that the main orientation is given by:

$$\theta = atan2\,(m_{01}, m_{10}) \tag{2.24}$$

This angle coincides with the orientation of the vector that joins the patch center (i.e. the keypoint) with the intensity centroid of the patch:

$$C = \frac{1}{m_{00}}\,(m_{10}, m_{01}) \tag{2.25}$$

Next, the ORB method employs the so-computed angle to enforce the descriptor invariance to in-plane rotations since BRIEF has shown to decrease its performance with slightly large rotations. For this purpose, the pattern of binary tests that BRIEF accomplishes (see section 2.2.1) are rotated according to the computed orientation. However, this reduces the distinction capabilities of the descriptors so a machine learning method is carried out to determine the binary tests that best de-correlates the descriptors, recovering this way their initial performance while keeping the rotational invariance.

These ORB features have shown to outperform other popular approaches in terms of computational efficiency with a reasonable rate of good matchings under different changes in the images. For these reasons, the ORB method has been used in the application shown in section 4.3.

2.2.2 Matching features

Once some of the most popular methods for detecting and describing image features have been presented, we focus now on some algorithms that can be employed to find

matches between sets of keypoints extracted in different images. These techniques are the core of the *Matching* and *Data association* blocks in figure 2.2.

Here we present a brief survey of some of the existing methods to perform keypoint matching. The first two approaches are applied to features described by its local area, i.e. by an image patch that surrounds the keypoint position. These kind of descriptors are typically employed when the chosen feature detector does not provide a proper descriptor by itself as, for instance, the above explained FAST or Harris methods. In these cases, the matching methods are only adequate for pairing visual features that undergo certain changes since they are not robust, for example, against changes in rotation or scale. However, these approaches are usually good enough for stereo matching processes. The two first methods presented in this section belong to this category.

On the other hand, more elaborated descriptors provided by complex approaches such as SIFT or SURF are usually harder to both compute and match but, on the other hand, they can cope with stronger changes in the keypoints appearance and, therefore, are more useful for data association processes. It is important to note that the emergence of binary-strings as keypoint descriptors (e.g. BRIEF, ORB) means a noticeable improvement in terms of efficiency for these kind of descriptors. Two of these approaches are presented at the end of this section.

Normalized Cross Correlation (NCC)

Although not robust to changes in neither rotation, nor scale, nor change of view, Normalized Cross Correlation (NCC) has been employed as a standard method to feature matching [51] and to determine the position of a given pattern within a certain image. The NCC approach is a version of the cross correlation method that features invariance against some changes in illuminations such as contrast or bright.

Formally, let $I_{\mathbf{x}}$ be the intensity value of an image I at pixel position $\mathbf{x} = (x,y)$ and, similarly, $T_{\mathbf{x}} = T(x,y)$ the equivalent for a certain image pattern or patch. Thus, the NCC value of the pattern placed at position $(x-u, y-v)$ is given by expression:

$$\gamma(u,v) = \frac{\sum_{x,y}\left(I_{\mathbf{x}} - \overline{I}_{u,v}\right)\left(T(x-u,y-v) - \overline{T}\right)}{\sqrt{\sum_{x,y}\left(I_{\mathbf{x}} - \overline{I}_{u,v}\right)^2 \sum_{x,y}\left(T(x-u,y-v) - \overline{T}\right)^2}} \tag{2.26}$$

where $\overline{I}_{u,v}$ stands for the mean intensity value of the image within the area of the shifted patch and $\overline{T}$ is the mean intensity value of the patch.

In general, if we are determining the position of the pattern within the image, NCC must be computed for all values of (u,v) and keep the position that maximizes equation (2.26). However, in terms of comparing the local area of two keypoints in an image, NCC gives us a measure of the similarity of the patches hence, with appropriate thresholds, determining the positive or negative keypoints matching.

The main drawback that a naive implementation of equation (2.26) suffers is its computational burden, since its denominator implies the re-calculation of the image mean $\overline{I}_{u,v}$ and the energy of the zero-mean image

$$\sum_{x,y} \left(I_{\mathbf{x}} - \overline{I}_{u,v} \right)^2 \tag{2.27}$$

for every point u,v, becoming unacceptable in terms of complexity.

Some approaches have been presented that overcome this issue although their explanation goes beyond the scope of this thesis and can be found elsewhere [21, 113, 195].

Sum of absolutes differences (SAD)

An interesting and simple alternative to NCC when coping with keypoints suffering from small-translational changes is the Sum of Absolutes Differences (SAD) or L1-distance, which follows the expression:

$$\gamma = \sum_{x,y} |I_{\mathbf{x}} - T_{\mathbf{x}}|, \tag{2.28}$$

being again $I_{\mathbf{x}}$ and $T_{\mathbf{x}}$ the intensity levels at pixel $\mathbf{x}$ for the image and the pattern, respectively, (or simply the two image patches to compare, depending on the application) and operating over the pixel window on the patch.

This approach is significantly faster than NCC and achieves acceptable performance when performing stereo matching although it is not robust against changes in illumination. On the other hand, it can also be applied to compute distance between descriptors. Some applications that employs SAD can be found in [81, 199].

Euclidean and Mahalanobis distance

Float or integer vectors used as descriptors (e.g. SIFT, SURF) are usually compared by means of the Euclidean or the Mahalanobis distance, the latter being a generalization of the former.

The Mahalanobis distance d_M between two descriptor vectors $\mathbf{x} = (x_1,...,x_N)^\mathrm{T}$ and $\mathbf{y} = (y_1,...,y_N)^\mathrm{T}$ that are observations of the same probability distribution is defined as:

$$d_M = \sqrt{(\mathbf{x}-\mathbf{y})^\mathrm{T}\mathbf{C}^{-1}(\mathbf{x}-\mathbf{y})}, \tag{2.29}$$

where $\mathbf{C}$ is the covariance matrix which models the uncertainty in each of the dimensions of the descriptor. If matrix $\mathbf{C}$ is diagonal, the Mahalanobis distance becomes:

$$d_M = \sqrt{\sum_i \frac{(x_i - y_i)^2}{\sigma_i^2}}. \tag{2.30}$$

with σ_i being the standard deviation of the descriptors in the i-th dimension. Finally, if $\mathbf{C}$ is the identity matrix d_M becomes the Euclidean distance or L2-distance between descriptors.

In general, if uncertainty information is available, Mahalanobis distance is a more robust method for keypoint matching since it takes into account the real shape of the descriptor space. The definition of a set of rules regarding the distance between descriptors that must be fulfilled allows the establishment of correspondences between keypoints. An example of this will be shown in further chapters.

Hamming distance

In recent years, due to the rise of mobile devices and smartphones with huge possibilities but fewer computation capabilities than computers, the search for new fast image detectors and descriptors has become a cornerstone in the computer vision community. Hence the development of algorithms that compute keypoint descriptors based on binary strings [3, 23, 165].

The matching process for these kind of descriptors is based on the so called Hamming distance [80] which can be defined as the number of bits that changes between two descriptors $\mathbf{x} = (x_1,...,x_N)^\mathrm{T}$ and $\mathbf{y} = (y_1,...,y_N)^\mathrm{T}$:

$$d_H = |\{i|1 \leq i \leq N, x_i \neq y_i\}| \tag{2.31}$$

In modern computers equipped with SSE2 instructions the Hamming distance can be computed extremely efficiently. This method is employed with ORB descriptors in chapter 3.3.4.

2.2.3 Epipolar geometry

In order to improve the process of matching keypoints through the above explained methods, the so called *epipolar geometry* provides us with constraints that eases and improves the process.

Epipolar geometry is defined as the projective intrinsic geometry between two views [83] and only depends on the cameras' intrinsic parameters (i.e. focal length, principal point, etc.) and their relative poses. This implies the powerful fact that epipolar geometry is independent of the scene structure.

In this thesis, the use of the epipolar geometry is motivated by the need of restricting the search area for finding correspondences between keypoints. Furthermore, it can be useful when detecting bad tracked keypoints.

Two-view epipolar geometry is essentially the geometry of the intersection of image planes with the sets of planes which have as a common axis the line (called *baseline*) that joins the optical centers of the cameras.

Figure 2.8 shows an example of this two-view geometry with the projection of a certain 3D point P. In the figure, the following items must be highlighted:

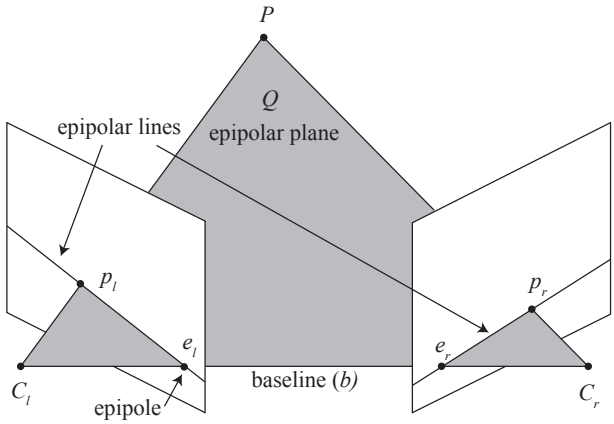


Figure 2.8: Illustration of epipolar geometry for two views.

- **Epipole**. The epipoles (e_l and e_r in the figure) are the intersection points of the baseline b with the image planes. Moreover, the epipole in one image is the projection of the other camera optical center.
- **Epipolar plane**. The epipolar plane (Q) contains the baseline and the considered 3D point P.
- **Epipolar line**. The epipolar lines arise from the intersection of the epipolar plane with the image plane in each camera. For each epipolar plane, the associated epipolar lines are said to be correspondent between them.

In the figure, the 3D point P is projected to both the left and right images at points p_l and p_r, respectively. These three points (P, p_l y p_r) and the cameras' optical centers (C_l y C_r) lay on the same plane: the epipolar plane Q.

Thus, the position p_l of the projected 3D point in the left image (for instance) determine a unique line on the right image which contains the projection of the 3D point on the image plane. This line is the epipolar line. So, in order to find a match for a certain projected point it is enough to look for the correspondent along its associated epipolar line, restricting the search area from an image 2D area to only a line (1D). Analogously, the left projection of the 3D point lies on the epipolar line associated to the right projection.

Note that the epipolar plane defined by any point in the 3D space always contains the baseline, hence all the epipolar lines intersect at the same point: the epipole, which is also the intersection point of the baseline with the image planes.

For an ideal configuration of the stereo cameras, i.e. with parallel optical axes, the epipolar lines are horizontal and parallel on the images. Therefore, when performing stereo matching, in practice, it is enough to search for correspondences on the same

row of pixels, considerably simplifying the computations that involve the projection equations. This is the case assumed throughout this thesis. However, if the employed cameras are not ideally arranged, a *rectification* procedure can be applied to convert the captured images into so-called *rectified* images, which can be thought of as acquired by an ideal configuration of cameras [61].

Finally, it is important to remark that all the information regarding the two-view epipolar geometry is contained in the so called fundamental matrix **F**. Given a pair of images, for each pixel p_l in the left image there exists an associated epipolar line l_r on the right image so that its correspondent pixel p_r lies on it. Therefore, there is a transformation between a point in one image and a line on the other:

$$p_l \mapsto l_r \tag{2.32}$$

This transformation is represented by a matrix **F** denoted *fundamental matrix* with the following properties:

- The fundamental matrix fulfills that, for each pair of correspondent points p_l and p_r on the images represented by their homogeneous coordinates, then

$$p_r^T \mathbf{F} p_l = 0 \tag{2.33}$$

- **F** has rank 2 with 7 degrees of freedom.
- The epipolar line associated to the point p_l can be found through $l_r = \mathbf{F}p_l$ whilst, similarly, the associated to the point p_r comes from $l_l = \mathbf{F}^T p_r$
- The relation between the epipoles and **F** is defined by $\mathbf{F}e_l = 0$ y $\mathbf{F}^T e_r = 0$.

In summary, the fundamental matrix **F** contains all the information we need to employ the epipolar geometry to the problem of finding correspondences between keypoints in stereo images. The use of **F** allows for the computation of the epipolar lines that restrict the search area when finding keypoint correspondences.

However, for an ideal configuration of the stereo pair, as the one employed in this thesis, the epipolar geometry restriction reduces the problem to search for matched keypoint that lay on the same row of pixels, highly simplifying the process.

2.2.4 Data association – Bag-of-words

In the context of this thesis, data association can be defined as the process of establishing correspondences between a set of current observations and the map of already observed landmarks. In general, it can be understood as a complex case of keypoint matching where the image features may undergo severe changes as a result of arbitrary, unrestricted camera movement. Data association is at the core of most vision-based localization techniques and robust methodologies must be applied to perform it.

The purpose of data association is twofold: (i) creating relationships between camera poses along time and (ii) detecting loop closures, i.e. re-observations of already traversed zones. The former can be understood as the simple process of matching visual features between consecutive time steps, whilst the latter possesses larger implications in terms of localization.

In general, as the camera moves, the set of currently detected keypoints contains both new and recently seen image features. Occasionally, features that where seen long time ago are re-observed again, meaning that the camera is traversing an already explored area. As mentioned in previous sections, this is called loop closure, and it is of capital importance in localization since re-visiting an area significantly reduce the uncertainty in the estimation of both the robot pose and the landmark positions.

In any case, different computer vision methods have been developed to efficiently handle data association with a large amount of data. Among others, we can mention k-d trees [8, 119, 135], hash techniques [110, 172, 193, 204] or pyramid matching kernels [72].

In section 4.3 we present a complete visual SLAM application based on ORB keypoints and descriptors. Data association and loop closure detection are aided by a bag-of-words-based approach [62] that performs robust place recognition. The basic technique here consists of building a database from the images recorded as the robot navigates and finding the most similar one within the database when a new image is captured. Although initially developed for BRIEF descriptors, the mentioned method has proven to perform well with ORB descriptors due to their close relationship with BRIEF.

The bag-of-words technique uses a previously built visual vocabulary to transform an image to a numerical vector that is subsequently employed to look up among the images stored in the database. Since the bag-of-words presented in [62] is hierarchical, the vocabulary structure becomes a tree, and its operation, in the context of our developed visual SLAM application explained in section 4.3 can be summarized as follows:

- First, a vocabulary is created off-line from all the binary descriptors extracted from a collection of training images.
- For each new captured image, a set of binary descriptors is computed. Then, the vocabulary is employed to convert those descriptors to a single vector that represents the image.
- Similar images in the database are retrieved according to the distance between the currently computed vector and those already stored.
- Finally, the new vector is added to the database for later use.

This approach allows us to detect when the camera is re-visiting an already explored area, which is of paramount importance for visual SLAM.

2.3 Back-end (Robot localization)

This section discusses our proposed back-end for the visual odometry and visual SLAM applications. In this work, the back-end is in charge of optimizing the estimation of the system state from the observations provided by the front-end. Thereby, here we present some of the most popular methods to handle the general full SLAM problem.

Nowadays, the core of the back-end part of the SLAM problem is mainly approached by two coexisting methodologies namely *probabilistic filters* (e.g. Kalman filter, Extended Kalman filter, particle filters, etc.) and *graph-SLAM* [73] (also known as *bundle adjustment* in the computer vision community). The former (refer to section 2.3.1) focuses on iterative solutions relying on the Bayes' rule that try to estimate the evolution of the system state through a sequence of actions and observations. The latter (see section 2.3.2) makes use of modern sparse algebra methods to efficiently compute the least-squares minimization of the mismatch between all the observations and the predictions, yielding an estimation of both the camera pose and landmarks positions.

Although, in general, bundle adjustment achieves better results in terms of accuracy per unit of computing time than filtering approaches [185], the latter has been traditionally applied to the SLAM problem until recent years, and still has its niche of applicability.

Chapter 4 presents two applications of the solutions explained here, one for each category.

2.3.1 Probabilistic filters for SLAM

As mentioned in the introductory section, uncertainty in mobile robot localization suggests the use of probability distributions to model the system state and the application of tools capable of dealing with them. These tools handle the inherent inaccuracies of sensors and other sources of error, and have proven to strengthen robustness in localization, achieving better results than deterministic approaches.

Definitions

Before addressing the procedure of some of the probabilistic filters used in SLAM, here we define some basic concepts related to them.

- **System state**. The system state is a vector encompassing the robot path (i.e. the sequence of robot poses) and the parametrization of the map. As mentioned before, in this thesis, the robot pose includes three spatial coordinates and three Euler angles defining its orientation. However, if the robot undergoes a planar movement, the robot pose can be represented by just two spatial coordinates and one angle. Finally, the map comprises all the 3D coordinates of the landmarks that the robot has sensed until the current time-step.

Thus, we can formally denote the system state at time t by $\mathbf{x}_t$:

$$\mathbf{x}_t \equiv (\mathbf{s}_t, \mathbf{m}_t), \tag{2.34}$$

with $\mathbf{s}_t$ standing for the robot pose and $\mathbf{m}_t$ representing the map.

- **Action**. An action changes the system state between time-steps. For robot navigation, actions correspond to robot movements. In practice, it is assumed that the robot moves between consecutive time-steps even if it actually remains still, i.e. *not moving* is also considered an action. Although actions are usually measured by means of proprioceptive sensors (encoders mostly), here we will employ cameras to estimate the robot ego-motion (the so called visual odometry).

  Formally, let $\mathbf{u}_t$ be the action measured at time t and $\mathbf{u}_{t_1:t_2}$ the set of actions between two time-steps t_1 and t_2 (with $t_1 \leq t_2$):

$$\mathbf{u}_{t_1:t_2} = (\mathbf{u}_{t_1}, \mathbf{u}_{t_1+1}, \ldots, \mathbf{u}_{t_2}) . \tag{2.35}$$

- **Transition model**. The transition model (or motion model for robot navigation) specifies how the system state evolves based on the actions $\mathbf{u}_t$:

$$p(\mathbf{x}_t | \mathbf{u}_t, \mathbf{x}_{t-1}) . \tag{2.36}$$

  Due to uncertainty, the transition model becomes a probability distribution and not a deterministic function. Note that, due to the Markov assumption, the current system state does not depend on the whole sequence of states $\mathbf{x}_{1:t-1}$, but only on the previous one.

- **Observation**. Observations provide the system with information about the robot environment. Laser scanners, sonar data or information extracted from images are examples of typical observations.

  Here we denote by $\mathbf{z}_t$ the observation at time t and, similarly to $\mathbf{u}_{t_1:t_2}$, $\mathbf{z}_{t_1:t_2}$ is the set of observations between time-steps t_1 and t_2:

- **Observation model**. The function that relates the obtained observations with system state is called the observation model. This model specifies the way the observations $\mathbf{z}_t$ are generated from the state $\mathbf{x}_t$:

$$p(\mathbf{z}_t | \mathbf{x}_t) . \tag{2.37}$$

  In general, it assumed that, during the navigation, actions and observations alternate in time in an action-observation fashion:

$$(\mathbf{u}_1, \mathbf{z}_1, \mathbf{u}_2, \mathbf{z}_2, \ldots, \mathbf{u}_t, \mathbf{z}_t) , \tag{2.38}$$

  with t standing for the current time-step.

- **Belief**. The belief represents the inner knowledge of the robot regarding the system state. Since the state is not directly measurable, the robot can only keep an estimation of it, therefore the *real* state and the *estimated* state of the system refers to different concepts.

  In probabilistic robotics, estimations are modeled as conditional probability distributions that assign a certain probability to each hypothesis about the state with respect to the real one. These distributions are conditional with respect to the available data up to the current time. So, the estimation of the system state $\mathbf{x}_t$ conditioned to the set of actions and observations from $t = 1$ until the current time-step reads:

$$bel(\mathbf{x}_t) = p(\mathbf{x}_t|\mathbf{z}_{1:t}, \mathbf{u}_{1:t}). \tag{2.39}$$

- ***Prior* and *posterior* estimations**. Please, note in equation (2.39) that in order to estimate the state at time t all the actions and observations up to that time have been taken into account. However, sometimes it is useful to refer only to the state estimation just before introducing the last observation, i.e.:

$$\overline{bel}(\mathbf{x}_t) = p(\mathbf{x}_t|\mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}). \tag{2.40}$$

  This probability distribution function is usually named *prior* estimation (or *prediction*, since it predicts the state at time t from only the previous state estimation and the current action). In the same way, equation (2.39) is also denominated *posterior* estimation, and the step between the prior and the posterior estimations is named *update*.

The common conceptual and mathematical base for any system that rigourosly handles the SLAM problem from a probabilistic perspective is the *Bayes' rule* or *Bayes' theorem*. The formulation of this theorem for solving the SLAM problem is commonly known as *Bayesian filtering*, meaning a temporal generalization of the Bayes' rule which has been extensively discussed elsewhere [127, 189].

Bayes' rule sets how to update the prior estimation of a variable x given a new observation z and an observation model:

$$\underbrace{p(x|z)}_{\text{posterior}} \propto \underbrace{p(x)}_{\text{prior}} \underbrace{p(z|x)}_{\text{obs. model}}. \tag{2.41}$$

This filter can be implemented through the iterative application of prediction and update steps. During prediction, the system state distribution is propagated through time according to some given transition model, leading to a prior estimation of the state. The update step refines this prior estimation based on the data provided by the robot exteroceptive sensors and a certain observation model, yielding the *posterior* estimation of the state, which comprehend all the available information up to current time-step.

Figure 2.9 shows a scheme representing one iteration of the Bayes' filter. The interactions of the above explained concepts can be seen in the figure as well as the

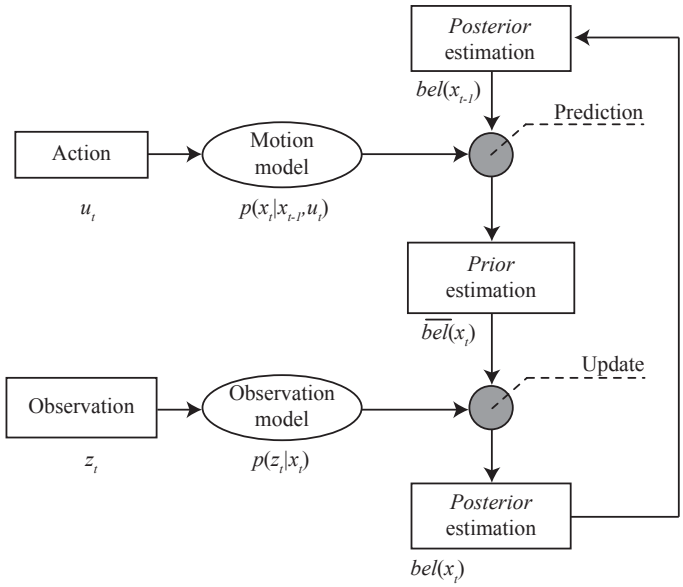


Figure 2.9: One iteration of a Bayesian filter showing the interactions between estimations, actions and observations.

procedure for obtaining the posterior estimation at time t from the estimation in the previous time-step and the sensory data gathered between them. Actions are typically measured by means of odometric systems while observations are gathered by the exteroceptive sensors on the robot.

Two of the most widely applied implementations of the Bayesian filter are the Kalman filter (KF) [99], among with its most popular version (the Extended Kalman filter (EKF) [96]), and the sequential Monte-Carlo methods (SMC) or particle filters (PFs) [4]. For years, standard EKFs have been thoroughly employed for tackling SLAM [40, 47] although they are limited by the assumption of Gaussian nature of both the state and the observations and by its quadratic cost with map size [25]. This issue prevents them to be properly applied to global localization problems where multi-hypothesis distributions models better the state uncertainty. On the contrary, PFs can cope with these kinds of distributions, not being limited by the Gaussianty assumption. Furthermore, with the introduction of *Rao-Blackwellised* particle filters (RBPFs) [136], SMC methods have become an unified framework for SLAM and global localization. Chapter 4 presents an application that utilizes this type of filter, further developing this approach.

Kalman filter

Developed in the 60's by Rudolph Emil Kalman [99], the Kalman filter is a recursive technique for filtering and predicting continuous linear systems.

It models beliefs at time t with parametric Gaussian distributions defined by their means $\boldsymbol{\mu}_t$ and their covariances $\boldsymbol{\Sigma}_t$ and ensures Gaussianity of posterior distributions under the following circumstances:

- The transition model is a *linear* function in the actions and in the previous state which is corrupted by zero-mean Gaussian noise, i.e.:

$$\mathbf{x}_t = \mathbf{A}_t \mathbf{x}_{t-1} + \mathbf{B}_t \mathbf{u}_t + \boldsymbol{\epsilon}_t, \tag{2.42}$$

  where matrix $\mathbf{A}_t$ relates the state at time-step $t-1$ to the state at the next time-step in the absence of either an action or noise in the transition. Matrix $\mathbf{B}_t$ relates the action u_t to the current state, while $\boldsymbol{\epsilon}_t$ is a zero-mean Gaussian random variable representing the transition model noise:

$$\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{Q}). \tag{2.43}$$

- The observation model is also a *linear* function in its arguments:

$$\mathbf{z}_t = \mathbf{C}_t \mathbf{x}_{t-1} + \boldsymbol{\delta}_t, \tag{2.44}$$

  with matrix $\mathbf{C}_t$ relating the current state to the observation and $\boldsymbol{\delta}_t$ standing for a zero-mean Gaussian random variable representing the observation model noise:

$$\boldsymbol{\delta}_t \sim \mathcal{N}(0, \mathbf{R}). \tag{2.45}$$

- The initial belief $bel(\mathbf{x}_0)$ must be Gaussian distributed:

$$bel(\mathbf{x}_0) \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0). \tag{2.46}$$

The Kalman filter operates in a sequence of prediction-update stages. Thus, to obtain the estimation of system state at time-step t, the filter evolves as follows:

- The posterior estimation at previous time-step $t-1$, represented by its mean $\boldsymbol{\mu}_{t-1}$ and covariance $\boldsymbol{\Sigma}_{t-1}$ is propagated according to the transition model and the current action u_t:

$$\overline{\boldsymbol{\mu}}_t = \mathbf{A}_t \boldsymbol{\mu}_{t-1} + \mathbf{B}_t \mathbf{u}_t \tag{2.47}$$

$$\overline{\boldsymbol{\Sigma}}_t = \mathbf{A}_t \boldsymbol{\Sigma}_{t-1} \mathbf{A}_t^{\mathsf{T}} + \mathbf{R}_t, \tag{2.48}$$

  becoming the prior estimation of the system state at time-step t.

- The prior is refined by introducing the current observation z_t:

$$\boldsymbol{\mu}_t = \overline{\boldsymbol{\mu}}_t + \mathbf{K}_t \mathbf{y}_t \tag{2.49}$$
$$\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{C}_t)\,\overline{\boldsymbol{\Sigma}}_t, \tag{2.50}$$

  with $\mathbf{y}_t$ being the *innovation* and whose value is given by:

$$\mathbf{y}_t = \mathbf{z}_t - \mathbf{C}_t \overline{\boldsymbol{\mu}}_t, \tag{2.51}$$

  and where $\mathbf{K}_t$ is the *Kalman gain*:

$$\mathbf{K}_t = \overline{\boldsymbol{\Sigma}}_t \mathbf{C}_t^{\mathrm{T}} \left(\mathbf{C}_t \overline{\boldsymbol{\Sigma}}_t \mathbf{C}_t^{\mathrm{T}} + \mathbf{Q} \right)^{-1}. \tag{2.52}$$

Extended Kalman filter

The main limitation of the Kalman filter is the assumption of linear functions for both the transition and the observation model, rendering the filter inapplicable in many real life applications. This is where the Extended Kalman filter plays a crucial role in robotics.

In the EKF, equations (2.42) and (2.44) become:

$$\mathbf{x}_t = g(\mathbf{u}_t, \mathbf{x}_{t-1}) + \boldsymbol{\epsilon}_t \tag{2.53}$$
$$\mathbf{z}_t = h(\mathbf{x}_{t-1}) + \boldsymbol{\delta}_t, \tag{2.54}$$

being $g(\cdot)$ and $h(\cdot)$ nonlinear functions that define the transition and the observation model, respectively. Since Gaussianity is not preserved under nonlinear transformations, KF equations can not be directly applied to these models. However, the EKF evolution does not change significantly and the matrices that related the current state to the actions, observations and the previous system state in the KF are now substituted by the linearized versions of the functions governing the motion and observation model. This linearization is performed by means of the first order *Taylor expansion*.

Thus, similarly to the Kalman filter performance, the EKF evolves as follows:

- The posterior estimation at time-step $t-1$ is now propagated according to the nonlinear transition function and the current action u_t:

$$\overline{\boldsymbol{\mu}}_t = g\left(\mathbf{u}_t, \boldsymbol{\mu}_{t-1}\right) \tag{2.55}$$
$$\overline{\boldsymbol{\Sigma}}_t = \mathbf{J}_t \boldsymbol{\Sigma}_{t-1} \mathbf{J}_t^{\mathrm{T}} + \mathbf{R}_t, \tag{2.56}$$

  where $\mathbf{J}_t$ stands for the *Jacobian* matrix of the motion model with respect to the N elements of the system state evaluated at the current time-step:

$$\mathbf{J}_t = \begin{pmatrix} \frac{\partial g_1}{\partial x_t^1} & \cdots & \frac{\partial g_1}{\partial x_t^N} \\ \vdots & & \vdots \\ \frac{\partial g_M}{\partial x_t^1} & \cdots & \frac{\partial g_M}{\partial x_t^N} \end{pmatrix}. \tag{2.57}$$

- The posterior estimation is again computed by refining the prior through:

$$\boldsymbol{\mu}_t = \overline{\boldsymbol{\mu}}_t + \mathbf{K}_t \mathbf{y}_t \tag{2.58}$$

$$\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t)\,\overline{\boldsymbol{\Sigma}}_t, \tag{2.59}$$

  with the innovation taking this expression:

$$\mathbf{y}_t = \mathbf{z}_t - h\left(\overline{\boldsymbol{\mu}}_t\right), \tag{2.60}$$

  and with Kalman gain:

$$\mathbf{K}_t = \overline{\boldsymbol{\Sigma}}_t \mathbf{H}_t^{\mathrm{T}} \left(\mathbf{H}_t \overline{\boldsymbol{\Sigma}}_t \mathbf{H}_t^{\mathrm{T}} + \mathbf{Q}\right)^{-1}. \tag{2.61}$$

  Similarly to $\mathbf{J}_t$, $\mathbf{H}_t$ is the Jacobian matrix of the observation model evaluated at the current time-step.

In short, EKF is a linearized version of the KF when the involved models are nonlinear, approximating the posterior estimation (which are no longer normal distributed) by Gaussians, following this way a similar formulation than the original linear filter. The EKF accuracy depends mainly in the degree of nonlinearities in the models and the level of uncertainty present in the system. Although being confined to the assumption of Gaussianity, the main advantages of these two approaches are their simplicity that turns into a low computational complexity.

Complete derivations of the Kalman and the Extended Kalman filter mathematics can be found elsewhere [139, 159, 189, 205].

Particle filters

Particle filters belong to an alternative category of solutions to the implementation of the Bayesian filter, namely *non-parametric filters*. Unlike the KF and EKF approaches, non-parametric filters do not represent posteriors as Gaussian distributions but they approximate them by a finite set of values. Therefore, they can deal with any kind of distribution, even multi-modal, becoming highly valuable when coping with certain types of robot localization issues, such as global localization, or when handling multi-hypotheses in SLAM.

Specifically, particle filters represent posterior estimations by a set of N samples drawn from the real distribution, each of them representing a hypothesis about the system state at time t:

$$\boldsymbol{\chi}_t := \left\{\mathbf{x}_t^i\right\}_{i=1\ldots N}. \tag{2.62}$$

Thus, the set of particles approximates the posterior distribution of the belief at time t, so that each particle $\mathbf{x}_t^i$ has a likelihood of being within the particle set proportional to its posterior belief:

$$\mathbf{x}_t^i \sim p\left(\mathbf{x}_t | \mathbf{z}_{1:t}, \mathbf{u}_{1:t}\right). \tag{2.63}$$

This way, particles will populate more densely those areas of the system space where the real state is more likely to appear. For a relatively large number of particles, the particle set will become a good approximation to the posterior belief.

Similarly to the Kalman filters in the previous section, the particle filter estimates the system state at time-step t (represented by the set of particles) from the previous estimation, the action $\mathbf{u}_t$ and the observation $\mathbf{z}_t$ as follows:

- A set of N particles is drawn from the prior estimation $p(\mathbf{x}_t|\mathbf{u}_t, \chi_{t-1})$. This can be accomplished by propagating each sample in the previous set according to the system motion model, yielding a new set of particles $\overline{\chi}_t$ meaning a representation of $\overline{bel(\mathbf{x}_t)}$.

- Each particle $\mathbf{x}_t^i$ is given an *importance factor* w_t^i that weights it according to the observation likelihood conditional to the particle hypothesis:

$$w_t^i = p\left(\mathbf{z}_t|\mathbf{x}_t^i\right). \tag{2.64}$$

- Finally, a re-sampling stage is performed in order to avoid particle depletion. This stage draws N replacement particles from the particle set built in the previous step. Each particle is drawn with a probability proportional to its weight so that particles with low weights (i.e. which are less probable) will tend to disappear while those more probable will keep alive or even duplicate in the particle set. The resulting set χ_t approximates the posterior belief of the system.

The main drawback of the particle filter is threefold. First, the number of particles in the set needs to be high enough since a small number of samples can bias the estimation of the posterior and can lead the filter to lose track of the real state, specially when its dimensionality is high. Second, the re-sampling process should be carried out with care since highly frequent re-samplings may lead the filter to increase its variance to unacceptable levels. In this sense, several works have focus on such effects trying to mitigate them [13, 49]. Finally, in any case, particle filters are usually more computational demanding than KF filters.

A significant improvement to the above explained particle filter that deals with high dimensional system states is presented in section 4.2, namely Rao-Blackwellised particle filters (RBPFs), and applied to the SLAM problem. The built map is subsequently employed to perform global localization in an indoor environment.

2.3.2 Bundle adjustment

Although early developed in the 50's and widely employed in the 80's and 90's in photogrametry [180], bundle adjustment (BA) has recently experienced a new youth in terms of its application to mobile robotics [116, 174, 185, 210].

Bundle adjustment can be defined as the problem of using 2D image features extracted from a set of images taken from different points of view to jointly estimate the

optimal 3D scene structure along with the camera parameters (including their poses and/or calibration). Under the assumption of Gaussian noise in the observations, BA becomes the *maximum likelihood estimator* (MLE) [169] of the 3D structure from motion problem [55]. Although initially considered as an off-line batch process that refines an initial estimation of camera poses and landmarks positions, modern sparse algebra techniques have allowed BA to be applied in real-time to the visual SLAM problem.

A popular representation of the SLAM problem when tackled within the computer vision framework are graph models where nodes represent the unknowns (camera poses and landmarks positions) and edges define constraints between nodes (odometry, observations, etc. [44, 109]). In this representation, camera poses are also called *keyframes* (KFs) and form a set of relevant frames chosen through some kind of heuristics among all the image frames.

In order to estimate both the structure and the camera parameters, it is carried out a nonlinear least-squares optimization process that minimizes a certain error function which measures the mismatch between all the observed image features and their predicted positions in the image. Generally, such an error function is assumed to be quadratic in the image feature projection errors. Non-quadratic error functions can be chosen to implicitly deal with the presence of outliers within the input set of observations, being sometimes referred as *robust kernels*. An example of a visual odometry application that makes use of such robust kernels is presented in section 3.3. Nevertheless, this section addresses the standard quadratic error function-based BA method.

Formally, and in order to establish a consistent, unified notation for different versions of the BA problem, let us define the system state by:

$$\mathbf{s} = (\mathbf{p}, \mathbf{x}) \tag{2.65}$$

which encompasses the problem unknowns, i.e. the collection $\mathbf{p}$ of P keyframes and the set $\mathbf{x}$ of L 3D landmarks positions:

$$\mathbf{p} = \left\{\mathbf{p}_{j,b}\right\}_{j=1\ldots P,\ b\in\{1\ldots P\}|b\neq j} \qquad \mathbf{x} = \left\{\mathbf{x}_{j,b}\right\}_{j=1\ldots L,\ b\in\{1\ldots P\}} \tag{2.66}$$

Regarding the keyframes, this notation defines j as an identifier for the keyframe while b indicates the reference system which it is referred to. Note that b cannot be the same as j since a keyframe referred to its own reference system is senseless. On the other hand, in the case of the landmarks, j stands again for an identifier whilst b specifies the so-called *base* KF, which means the reference system where the j-th landmark coordinates are referred to. It is important to remark that the choice of the b keyframes represents the main difference of the diverse bundle adjustment approaches.

Now, let $\mathbf{z}$ be a set of N observations each one of them corrupted by zero-mean Gaussian noise with inverse covariance matrix Λ_j:

$$\mathbf{z} = \left\{\mathbf{z}_j^o\right\}_{j=1\ldots N,\ o\in\{1\ldots P\}} \tag{2.67}$$

The notation here defines the observation of the j-th landmark from the o KF, which we call the *observer* KF. Despite this formulation, the subscript and superscript concerning the base and/or the observer KFs for the keyframes, the landmarks and/or the observations will be dropped when their reference is unnecessary.

In the context of this thesis, we define the observations as 4D vectors composed of the image coordinates of the 3D landmarks projections in both the left and right images:

$$\mathbf{z}_j = (u_L, v_L, u_R, v_R)_j^{\mathsf{T}}. \tag{2.68}$$

On the other hand, the standard quadratic error function F for both the KFs and the landmarks positions takes into account all the individual observations and follows this expression:

$$F(\mathbf{s}) := F(\mathbf{p}, \mathbf{x}) = \frac{1}{2} \sum_j \Delta\mathbf{z}_j^{\mathsf{T}} \Lambda_j \Delta\mathbf{z}_j, \tag{2.69}$$

where

$$\Delta\mathbf{z}_j = \mathbf{h}_j(\mathbf{p}, \mathbf{x}) - \mathbf{z}_j, \tag{2.70}$$

is the error between the prediction of the j-th feature and its observation. In this context, the function $\mathbf{h}_j(\cdot)$ computes the predicted image coordinates of the j-th 3D landmark in both stereo images according to the current system state.

The most popular approaches to minimize F in equation (2.69) over the system parameters are the well-known Gauss-Newton (GN) algorithm and, its variant, the Levenberg-Marquardt (LM) method, both relying on the linearization of the cost function by means of its truncated Taylor series expansion. Minimizing this linearized version can be done by iteratively estimating increments of the unknowns $\Delta\mathbf{s} = [\Delta\mathbf{p}, \Delta\mathbf{x}]^{\mathsf{T}}$ so that:

$$\mathbf{H}\Delta\mathbf{s} = -\mathbf{g}, \tag{2.71}$$

where $\mathbf{H}$ and $\mathbf{g}$ stand for the Hessian matrix and the gradient of F, respectively. In order to simplify the equation construction and under some mild assumptions, $\mathbf{H}$ and $\mathbf{g}$ can be substituted by:

$$\mathbf{H} \approx \mathbf{J}^{\mathsf{T}} \Lambda \mathbf{J}, \tag{2.72}$$

$$\mathbf{g} = \mathbf{J}^{\mathsf{T}} \Lambda \Delta\mathbf{z}, \tag{2.73}$$

with Λ being a block diagonal matrix containing the individual information matrices Λ_j of the observations, so that equation (2.71) becomes:

$$\left(\mathbf{J}^{\mathsf{T}} \Lambda \mathbf{J}\right) \Delta\mathbf{s} = -\mathbf{J}^{\mathsf{T}} \Lambda \Delta\mathbf{z}, \tag{2.74}$$

with $\mathbf{J} = \partial\mathbf{h}/\partial\mathbf{s}$ being the Jacobian matrix of F.

Equation (2.74) establishes the core formulation of the GN iterative method. However, setting up an initial estimation that is not close enough to the real solution may increase the convergence time of the GN algorithm. In this case, the usual solution

is to introduce a variable dumping parameter in such equation so that the minimization process operation ranges from standard GN to a method more close to gradient descent, defining this way the so-called Levenberg-Marquardt method:

$$\left(\mathbf{J}^{\mathrm{T}}\Lambda\mathbf{J}+\lambda\mathbf{I}\right)\Delta\mathbf{s}=-\mathbf{J}^{\mathrm{T}}\Lambda\Delta\mathbf{z}, \tag{2.75}$$

with $\mathbf{I}$ standing for the identity matrix and λ being the dumping factor, which is updated in every iteration.

Finally, the so computed increment $\Delta\mathbf{s}^{[i]}$ at iteration $[i]$ is added to the previous system state estimation:

$$\mathbf{p}^{[i]} = \mathbf{p}^{[i-1]}+\Delta\mathbf{p}^{[i]}, \tag{2.76}$$

$$\mathbf{x}^{[i]} = \mathbf{x}^{[i-1]}+\Delta\mathbf{x}^{[i]}, \tag{2.77}$$

and the process is repeated until convergence is achieved, hence obtaining the final least-squares solution.

The sparse structure of the Jacobian matrix $\mathbf{J}$ plays an important role in BA since it can lead to an efficient solution to equation (2.74) (or equation (2.75), depending on the approach). Then, consider that each observation $\mathbf{z}_j$ contributes to the Jacobian matrix in form of a row $\mathbf{J}_j$ of block matrices extracted from the derivation of the prediction function with respect to the unknowns, i.e.:

$$\begin{aligned}\mathbf{J} &= \begin{pmatrix}\mathbf{J}_1\\ \vdots\\ \mathbf{J}_N\end{pmatrix}\\ \mathbf{J}_j &= \left(\frac{\partial\mathbf{h}_j}{\partial\mathbf{p}_1},\ldots,\frac{\partial\mathbf{h}_j}{\partial\mathbf{p}_P},\frac{\partial\mathbf{h}_j}{\partial\mathbf{x}_1},\ldots,\frac{\partial\mathbf{h}_j}{\partial\mathbf{x}_L}\right)\end{aligned} \tag{2.78}$$

At this point, let us recall that the landmarks are stored in the system state with coordinates relative to its base, which may be different from the current observer. Therefore, in order to compute the prediction $\mathbf{h}_j(\cdot)$ of a certain landmark $\mathbf{x}_j$ with base KF b, it is necessary to transform such coordinates to the observer o KF local reference system and, subsequently, to project the locally referred landmark into both stereo images, yielding this way a vector with the same dimension as the observation. Formally, here we will employ $\mathbf{x}^o_{j,b}$ to denote such local coordinates.

Then, the projected *homogeneous* coordinates of the landmark into one of the images (say, the left one) can be computed by:

$$(ku_L,kv_L,k)^{\mathrm{T}}_j=(\mathbf{K}|\mathbf{0})\,\mathbf{T}_c^{-1}\mathbf{x}^o_{j,b}, \tag{2.79}$$

where $\mathbf{K}$ represents the camera projection matrix, which encompasses the standard pin-hole camera intrinsic parameters, namely the focal length and the principal point image coordinates [83]:

$$\mathbf{K}=\begin{pmatrix}f_x & 0 & c_u\\ 0 & f_y & c_v\\ 0 & 0 & 1\end{pmatrix}, \tag{2.80}$$

and with $\mathbf{T}_c \in \mathbf{SE}(3)$ being a 4×4 transformation matrix that determines the camera pose within the robot local system. Such transformation matrix allows the use of multi-cameras systems and handles habitual situations where the camera is not placed at the origin of the robot reference system. Note that, for stereo vision systems, two slightly different matrices have to be defined (one for each camera), although they typically differ in just a translation along the x axis of the camera reference system, as assumed in this thesis.

Now we find $\mathbf{x}^o_{j,b}$ by changing the reference system of the stored landmark coordinates:

$$\mathbf{x}^o_{j,b} = (\mathbf{T}_{o,b})^{-1}\,\mathbf{x}_{j,b} \tag{2.81}$$

where the matrix $\mathbf{T}_{o,b} \in \mathbf{SE}(3)$ stands for the 4×4 transformation matrix between the reference systems of the observer o and the base b keyframes. It has to be highlighted that depending on the considered version of the BA problem, it may be necessary to compute this by composing a chain of relative poses, as we will address later.

Finally, since matrices $\mathbf{K}$ and $\mathbf{T}_c$ do not change over time, they can be combined into a single, constant matrix $\mathbf{P}$ yielding the final expression that relates the j-th landmark position stored in the system state $\mathbf{x}_{j,b}$ with its predicted coordinates in the left image:

$$(ku_L, kv_L, k)^{\intercal}_j = \mathbf{P}\,(\mathbf{T}_{o,b})^{-1}\,\mathbf{x}_{j,b} \tag{2.82}$$

The non-homogeneous version of equation (2.82) can be determined through the division of its first two coordinates by its third one. However, this expression yields the predicted coordinates of a landmark in just one image, i.e. half of the prediction. For stereo vision, though, this has to be computed twice (one with each of the above mentioned $\mathbf{T}_c$ camera transformations) in order to get the four elements that comprises the prediction $\mathbf{h}_j$.

According to this formulation, each observation $\mathbf{z}^o_j$ depends on the landmark 3D position $\mathbf{x}_{j,b}$ and the observer KF pose. If both KFs are not the same (which usually are not), a set of non-zero blocks appears in the Jacobian matrix corresponding to the chain of pose compositions between them.

To illustrate this, please refer to figure 2.10, which shows an example of a bundle adjustment problem. There, a robot follows an arbitrary path through an environment while gathering a total of 22 observations (shown as red-dashed arrows) of the 6 landmarks present in the scenario (displayed as numbered stars). In the figure, the observations are labeled following the notation introduced in equation (2.67). During navigation, 15 KFs have been defined among the whole set of captured frames.

In this situation, GBA defines all the KF poses and landmark positions referred to the global reference system centered at KF#1, as shown in figure 2.11. Consequently, all the observations obtained from KFs different from KF#1 imply the transformation of the landmark coordinates to its local reference system, hence inserting a non-zero block into the Jacobian matrix corresponding to the current observer KF pose. As a result of the global representation of KFs, this implies that $\mathbf{T}_{o,b}$ in equation (2.82) is composed by only one transformation with $b = 1$ for all the observations gathered from KFs different to the first one.

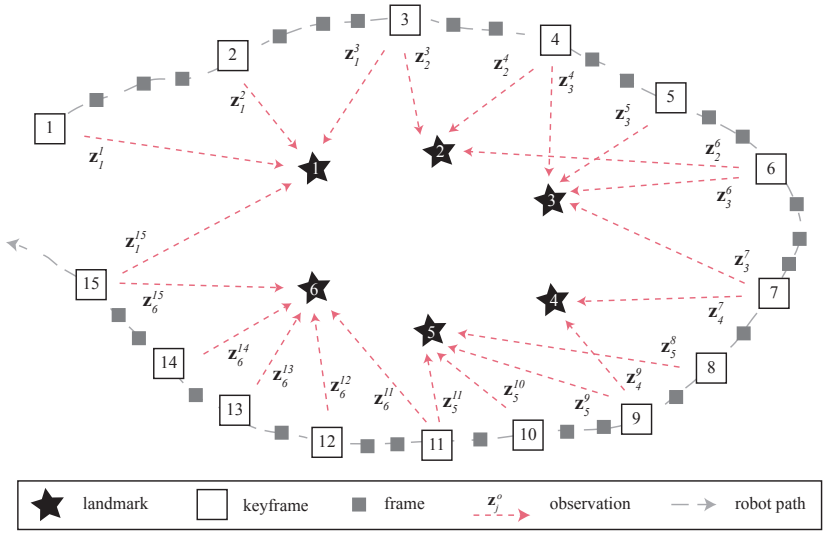


Figure 2.10: Bundle adjustment example.

Therefore, the scenario shown in figure 2.10 when addressed under a GBA formulation results in the highly sparse Jacobian and Hessian matrices shown in figure 2.12. In that figure, the labels $\mathbf{p}_{j,b}$ represent the keyframes poses as defined in equation (2.66), with $b = 1$ for all the keyframes, as corresponds to a GBA parametrization.

That sparsity pattern can be exploited by some sparse algebra techniques such as Cholesky decomposition [37] or Schür complement [152] to reduce the computational burden incurred when working with those large matrices.

However, the main drawback of the global bundle adjustment approach still appears related to scalability. The nonlinear iterative methods employed to compute the solution imply the inversion of the Hessian matrix (derived from equation (2.71)), which can grow arbitrarily as new observations and camera poses are introduced into the system. Moreover, there exists no upper bound for the computational complexity needed to process a loop closure since the larger the loop is, the higher is the number of unknowns that needs to be re-estimated and, in addition to that, sparsity is reduced at loop closure. In this situation, the computational complexity may reach $O(n^3)$ with n the number of KFs in the problem, becoming intractable even taking advantage of the Jacobian matrix sparse nature to reduce the computational cost. This renders GBA inapplicable to solve large localization problems, specially at loop-closure. Some new methodologies have appeared recently trying to overcome this performance issue [98] and, in this sense, *relative bundle adjustment* (RBA) means a promising framework, which we further develop next.

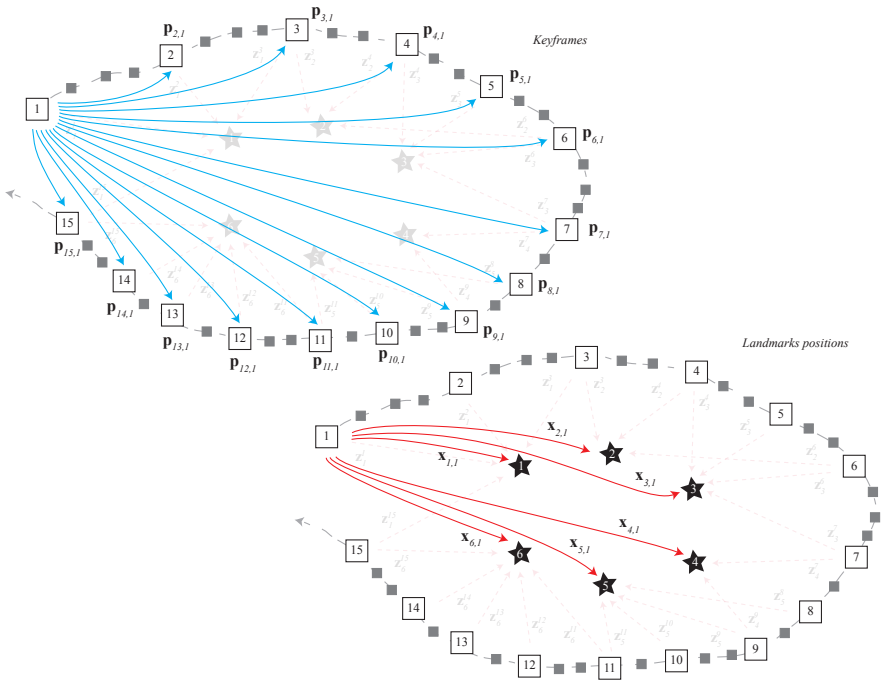


Figure 2.11: Global bundle adjustment example. All KFs (blue arrows) and landmark positions (red arrows) are referred to the first KF, which acts as the global origin of coordinates.

Relative bundle adjustment

Modifying the way how keyframes are defined can lead us to a huge difference in terms of efficiency when tackling the visual SLAM problem under a BA perspective. This statement is the core of the so-called *relative bundle adjustment* (RBA) [176], which essentially represents a certain keyframe as a node with a pose relative to the previous one. This transforms the graph shown in figure 2.11 into the *linear graph* displayed in figure 2.13, while representing the same scenario.

Unlike GBA, in RBA the global poses of the keyframes are not the unknowns of the problem, but the relative transformations between them. Thereby, it seems natural here to think of the transformations $\mathbf{T}_{j,b}$ as being part of the system state instead of the more general $\mathbf{p}_{j,b}$ poses defined for GBA. On the contrary, regarding the landmarks, we can keep the notation introduced in the previous section (equation (2.66)) as it properly encompasses all the useful information about the landmark, that is, its index and both its base and observer KFs.

As a subtle consequence of the RBA formulation, unknowns are now represented by edges in the graph instead of nodes as happened with GBA. In addition to that,

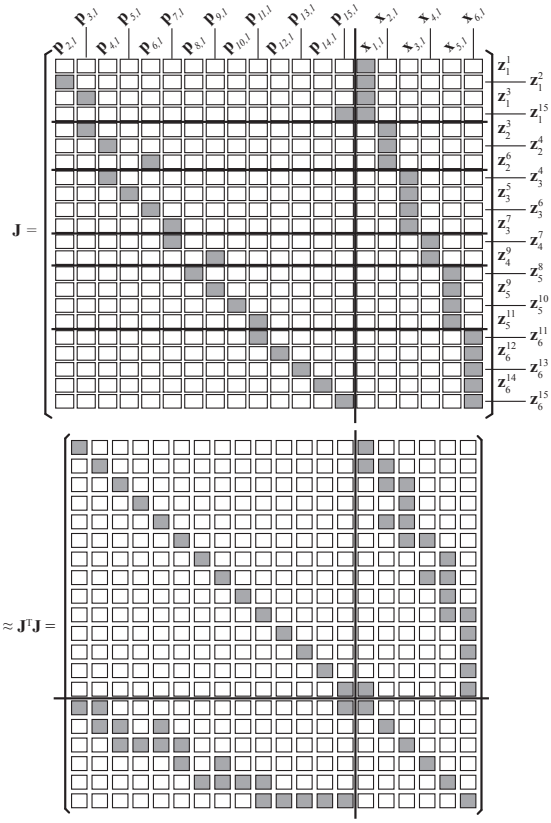


Figure 2.12: Sparsity patterns present in the Jacobian and Hessian matrices associated to a global bundle adjustment problem for the scenario shown in figure 2.10. Gray squares represent non-zero blocks.

another difference with GBA arises when the camera closes the loop where the problem structure must be modified due to the addition of a new unknown: the edge that joins the current KF and the *old*, re-visited one (please, refer to the green dotted edge in figure 2.13 joining KF#15 and KF#1).

On the other hand, as a negative side effect of the relative formulation, the sparsity level of both the Jacobian and the Hessian matrices results in fact reduced, degrading the efficacy of sparse algebra methodologies. Figure 2.14 shows the sparsity patterns of such matrices for the scenario previously presented in figure 2.10 which, compared to the GBA approach (figure 2.12), clearly exhibit more dense areas.

However, the big strength of the RBA approach lies in its flexibility degree in comparison to its global counterpart since it allows to optimize only a subset of the problem unknowns, chosen according to the area affected by the current observation.

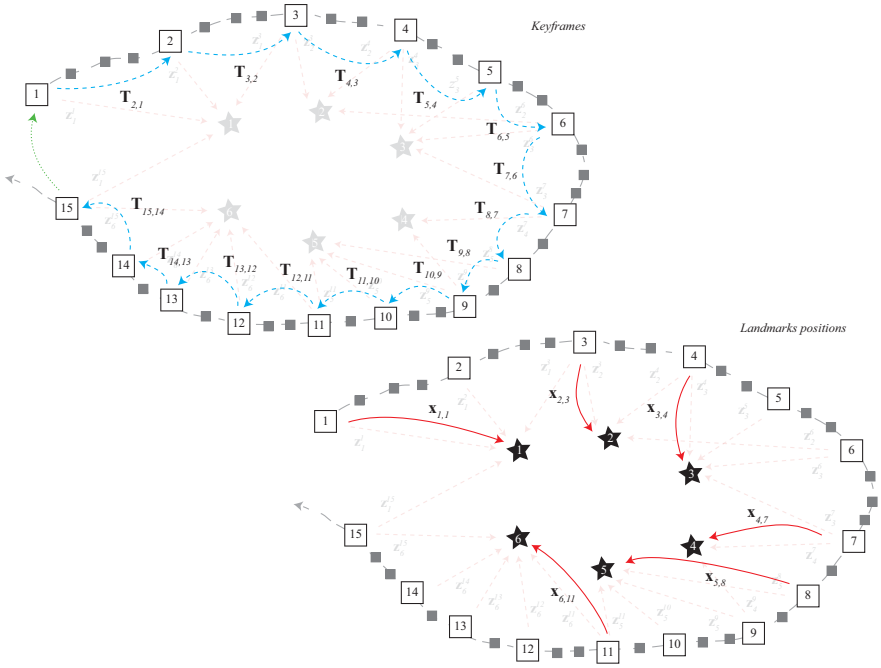


Figure 2.13: Relative bundle adjustment graph example. Note that every KF is referred with respect to the previous one (blue dashed arrows) while the landmarks positions are referred to their base KF (red solid arrows). Both types of connections are represented by edges in the figure and constitute the unknowns of the problem.

This area is called the *active region* and it is built through a breadth-first-search starting at current keyframe where the re-projection errors are computed in each keyframe, being added to the active zone those showing errors over a defined threshold. Moreover, the observed landmarks at current keyframe are also considered to be part of the active region.

In addition to that, the active region is augmented by another set of keyframes called the *static region*, which is composed by any non-active keyframes that have measurements of the active landmarks. The measurements at the static keyframes are included in the least-squares solution but their relative positions are not optimized in the process, hence the term *static*.

Since only the unknowns involved in the active region are optimized in RBA, the Jacobian and Hessian matrices become significantly smaller and, more importantly, the complexity burden of the solution remains bounded even in the presence of large loop closures. Fixed and adaptive versions of the active region can be found in [176] and [175], respectively.

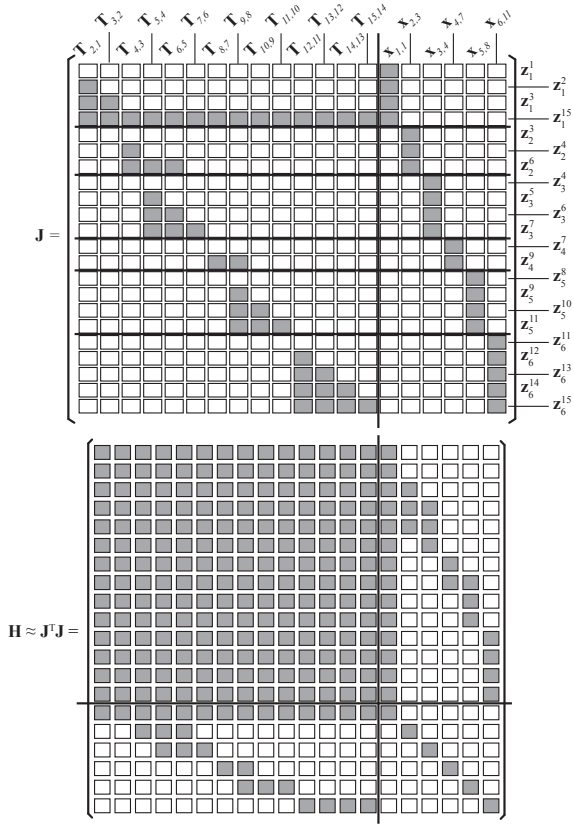


Figure 2.14: Sparsity patterns present in the Jacobian and Hessian matrices associated to a relative bundle adjustment problem for the scenario shown in figure 2.10. Gray squares represent non-zero blocks.

Finally, to mitigate the main drawback of the RBA methodology, i.e. the presence of denser matrices, a new approach is presented in [15] that exploits the flexibility of RBA solutions to build a sparser version that can be efficiently solved. Section 4.3 further develops this new technique and builds a complete visual SLAM system based on it.

Chapter 3
Visual odometry

Overview

Measuring the ego-motion of a mobile robot has been traditionally achieved by means of encoder-based odometry. However, this method presents several drawbacks, such as the existence of accumulative drifts, its sensibility to slippage, and its limitation to planar environments. The set of techniques that employ cameras as the only sensors to perform ego-motion estimation has been coined visual odometry. This chapter specifically focuses on a subset of these algorithms which uses stereo cameras, hence being named stereovisual odometry.

First we propose an optimal closed-form formulation that replaces the traditional iterative methods applied for stereovisual odometry and which does not exhibit convergence problems and is more accurate and efficient, under the assumption of complete absence of outliers. We also derive a formulation for the covariance associated to this estimation, which enables the integration of our approach into vision-based SLAM frameworks.

Despite the advantages of this method, it is significantly sensitive to the presence of outliers in the input data, so image features must be filtered to avoid wrong data associations. Typically, RANSAC is employed for this issue, but it is interesting to consider other alternatives when performing visual odometry. The use of iterative optimizations (based on least-squares methods) allows the introduction of robust techniques for detecting outliers at a fraction of the cost that RANSAC incurs.

In this chapter, we present two methods for deriving the camera ego-motion: one of them is based on a closed form solution while the other relies on an iterative optimization process that employs a robust kernel based on the pseudo-Huber function.

It's a dangerous business, Frodo, going out your door. You step onto the road, and if you don't keep your feet, there's no knowing where you might be swept off to. – Bilbo Baggins

3.1 Introduction

Odometry is one of the most widely-used means for estimating the motion of a mobile robot. Traditionally, odometry is derived from encoders measuring the revolutions of the robot's wheels, thus providing information for estimating the change in the robot pose. Unfortunately, the usage of encoder-based odometry is limited to wheeled robots operating on plane surfaces. Also, systematic errors such as drift, wheel slippage, and uncontrolled differences in the robot's wheels induce incremental errors in the displacement estimation, which cannot be properly modeled by a zero-mean Gaussian distribution. This erroneous assumption about the encoder-based odometry errors is accepted in most probabilistic filters for robot localization and SLAM [189], and may eventually contribute to the divergence of the filter estimation.

In order to overcome the limitations of encoder-based odometry, other non proprioceptive sensors such as laser sensors [69, 79, 183], vision-based systems [24, 202] and, more recently, RGB-D cameras [93, 101] have been used in the last years. The performance of laser sensors is restricted to purely planar motions, whereas vision-based odometry exploits the advantages of the wider field-of-view of cameras. Nowadays, cameras are cheap and ubiquitous sensors capable of collecting a huge amount of information from the environment. The existence of powerful methods for extracting and tracking features from images (refer to section 2.2.1), along with the above-mentioned advantages of cameras, establish an appropriate framework for applying vision to ego-motion estimation, along with several other applications.

A challenging issue found in computer vision approaches is the presence of noise and outliers that corrupt data and features gathered from images, compromising the accuracy and reliability of the intended applications. In general, though, the effects of random noise tend to cancel out when using least-squares estimation or other filtering techniques. In contrast, the presence of outliers (defined as *observations that appear to deviate markedly from other members of the sample in which it occurs* [76]) represents a more serious problem and, in fact, may lead to completely inaccurate results. Hence, the development of robust techniques for minimizing their impact has become essential.

In this sense, a wide range of robust estimators have been proposed in the technical literature, being the RANSAC method, originally published by Fischler and Bolles [56], probably the most widely employed in Computer Vision. RANSAC is a hypothesis-and-verify algorithm that iteratively generates a tentative solution from a randomly selected minimal subset of data and searches for consensus among the rest of the data, generating consensus sets (CS). The solution with highest support from all the hypotheses is taken as the final estimation of the model. Although being highly robust, RANSAC becomes computationally unfeasible when the percentage of outliers in the data increases significantly, since the number of hypotheses to be tested also grows, often preventing its usage in real time.

Hypothesis-and-verify methods are not the only solutions to properly cope with outliers. Methods that define robust distributions [149, 194] to model the presence of

errors in the data make also possible to estimate the camera motion from all the data, regardless the presence of outliers. Furthermore, although they can achieve a good estimation of the camera motion without explicitly distinguishing between inliers and outliers, it is straightforward to use them to perform fast outlier detection and, after their removal, to refine the final estimation.

3.1.1 General scheme of stereovisual odometry

The term *stereovisual odometry* stands for the process of estimating the change in pose that a stereo camera (or a robot equipped with a stereo camera) undergoes between two consecutive time-steps by exclusively employing the information from stereo images.

The general solution to the problem of estimating the change in pose with a stereo camera is depicted in figure 3.1 and follows this procedure:

1. **Feature detection**. An interest point detector (e.g. Harris corner detector [82], SIFT [119], etc. – please, refer to section 2.2.1) is employed to extract image features in a pair of images I_t grabbed with the stereo pair at time-step t.

2. **Stereo matching**. Match the features in the left image with their correspondent in the right one, normally aided by epipolar restrictions and using some measurement of the similarity between the features (e.g. normalized cross-correlation, sum of absolutes differences, SIFT descriptors, etc.).

3. Repeat steps (1) and (2) for the pair of images I_{t+1}.

4. **Data association**. Search for correspondences between the sets of matched features in images I_t and I_{t+1}. This is typically performed by using again some measurement of the similarity between the features and by searching within a window (of a certain size $w \times w$) around the position of the features in I_t. As a general rule, the association will be considered correct only if found in all the four involved images. Special care must also be taken to avoid outliers in the set of associated features.

5. **Estimation of the change in pose**. The set of so associated features represents the input data to an estimation process of the change in pose of the robot between the two consecutive time-steps. Formally, the change in pose can be modeled by a Gaussian random variable $\mathbf{q}_{t,t+1}$:

$$\mathbf{q}_{t,t+1} \sim \mathcal{N}\left(\boldsymbol{\mu}_q^{t,t+1}, \boldsymbol{\Sigma}_q^{t,t+1}\right), \tag{3.1}$$

where

$$\boldsymbol{\mu}_q = (\Delta x, \Delta y, \Delta z, \Delta\alpha, \Delta\beta, \Delta\gamma) \tag{3.2}$$

stands for the mean value and $\boldsymbol{\Sigma}_q$ is the covariance matrix associated to the random variable. Although some applications demand the computation of the covariance matrix, this might be unnecessary in many occasions. Please, note that

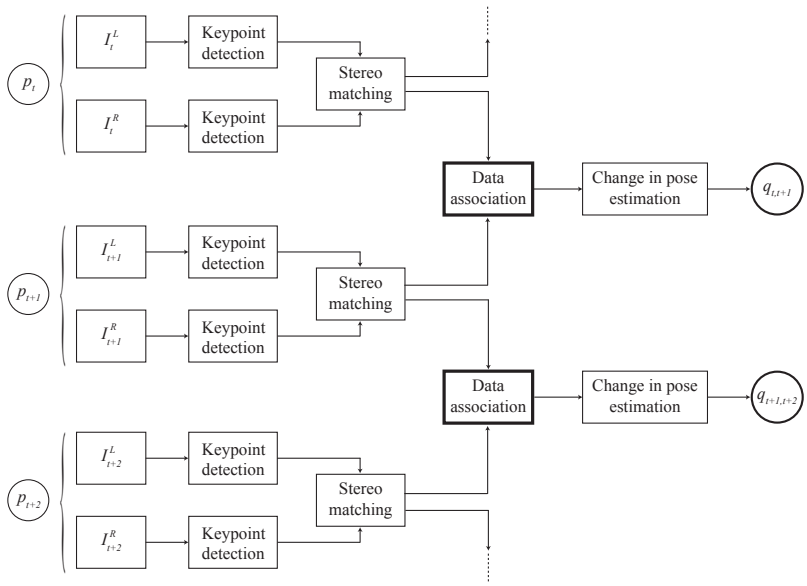


Figure 3.1: General procedure of a typical stereovisual odometry system.

the superscripts $t,t+1$ indicating the involved time-steps have been dropped for the sake of clarity.

The variables $\Delta x,\Delta y$ and Δz are the increments in the X, Y, and Z coordinates respectively, while $\Delta\alpha,\Delta\beta$ and $\Delta\gamma$ stand for the increments in the *yaw*, *pitch*, and *roll* angles, respectively.

At this point, we can employ closed-form methods and iterative optimization approaches to achieve this. Each of them has their pros and cons which will be considered next.

Closed-form solutions to the problem of finding the camera pose estimation avoids the need of a reliable initial guess, eluding this way divergence problems that may affect iterative methods. Besides, their closed-form nature makes them more computationally efficient. On the other hand, they are typically very sensitive to the presence of outliers in the input data, so that special care must be taken in this sense (e.g. using RANSAC).

On the contrary, iterative methods that perform optimizations to carry out visual odometry allow the use of robust error functions (sometimes referred as *robust kernels*) that cope well with outliers in the input set, making the result much more tolerant to their presence. These kernels can replace more time-consuming methods (generally RANSAC [56]) to avoid the influence of bad data associations at a frac-

tion of the typical cost. Under the assumption of a reasonably good initial estimation, iterative methods are a suitable option for visual odometry.

3.1.2 Related work

Several approaches have been proposed in the technical literature for estimating the displacement of a vision-equipped mobile robot from a sequence of images. Both works in [144, 145] reports a monocular and a stereo visual odometry system based on iterative methods for estimating the 3D change in robot pose, while [24] performs monocular visual odometry with uncalibrated consumer-grade cameras under the assumption of purely planar motion. In [178] it is presented a probabilistic method for performing SLAM which uses visual odometry as the robot motion model. This approach looks for sets of features in the stereo images and computes their SIFT descriptors in order to establish correspondences. The camera motion is subsequently estimated using an iterative optimization algorithm which minimizes the re-projection error of the 3D points.

Other solutions do not rely on feature extraction but they estimate motion directly from intensity values in the images. These so-called *direct methods* perform robustly in low textured scenes or even with blurred images at the cost of intensive computation [31, 92, 117, 138]. Blended solutions divided between feature-based and direct methods have also been proposed [57].

Finally, while some visual odometry methods have been improved by the combination with other sensors that reduce errors [151] and long-drive drift [148], other types of cameras have been also employed to perform visual odometry such as omnidirectional [32] or RGB-D [101, 197, 206] cameras.

On the other hand, and independently of their underlying method, most of these approaches rely on RANSAC for certain tasks, specially when dealing with outliers. Regarding this issue, several works in computer vision have tried to outperform the accuracy or overcome the main drawbacks of RANSAC. Thus, et al. presented in [192] a pair of approaches called MSAC and MLESAC which introduce some modifications in the RANSAC method to reinforce its robustness, and applied them to estimate some multiple view relations between images related by rigid motions. Briefly, MSAC evaluates the quality of a certain consensus set (CS) through a measurement of how well the inliers fit the model, instead of simply using the CS cardinality (i.e. the number of inliers). On the other hand, MLSAC goes one step further and uses the negative log-likelihood of the mixed distribution of the errors (including the outliers) as a score of the CS and simultaneously estimates the percentage of outliers present in the observation. Nevertheless, these approaches do not reduce the computational time of the RANSAC algorithm.

Focusing on the computational burden, a set of methods has been developed which adds a preemptive approach to RANSAC. Chum and Matas presented R-RANSAC [27], probably the first preemptive attempt to reduce RANSAC computational complexity, that decreases the amount of matches to test when evaluating a

hypothesis by selecting a random subset of them. The same authors also presented PROSAC [28], that orders the putative matches according to their scores and selects the minimal consensus set from a subset of them to instantiate the hypotheses. With this modification, they report a decrease of up to two orders of magnitude in processing time.

Based on MLESAC, Tordoff and Murray presented Guided-MLESAC [191], an improved version that reports a reduction of its computational time by an order of magnitude when applied to the estimation of camera motions. They employ the score of a match (in the paper, the normalized cross correlation of their grey-level patches) to derive its prior probability of being an inlier and to guide the selection of the features which form the initial minimal CS for a hypothesis. Hence, those matches with higher score are more probable to be selected to build an initial CS.

Finally, David Nister presented another preemptive RANSAC-based method [143] that estimates both structure and motion in multiple views in real time. This approach follows a *breadth-first* preemption scheme consisting of the generation of a set of hypotheses which are iteratively scored as new points are introduced into the CS. In each iteration, only a decreasing number of the highest scored hypotheses are evaluated until a winner is found.

Apart from the series of works by Nister [144, 145] which relies on preemption, little attention has been paid to the rest of the above-mentioned techniques in vision-based motion estimation, even though they have shown to be sound improvements of RANSAC. Instead, standard RANSAC has been extensively employed [2, 105, 106, 141].

3.1.3 Contribution

This chapter proposes the usage of two methods (one in closed-form and another following an iterative approach) for computing the complete 6D ego-motion (x, y, z, *yaw*, *pitch*, *roll*) of a camera/robot from a set of stereo images. These methods elude any constraint about the potential movements of the camera unlike other approaches as, for instance, [41].

Our first proposal, introduced in section 3.2, combines the speed of the Kanade-Lucas-Tomasi (KLT) detector and tracker [173] with the selectivity of SIFT descriptors [119] to match features in the stereo images. Since SIFT-based stereo matching is only carried out when the number of distinctive points in the tracker falls below a given threshold, we avoid the high computational cost involved in computing and comparing the Euclidean distance between SIFT descriptors for all the features in each pair of stereo images. It is important to note that, at the time this work was developed, SIFT descriptors were the state-of-the-art for keypoint description whilst Harris and KLT keypoints were commonly employed in computer vision. Hence, they were chosen to be the core of our proposal. However, more efficient detectors and descriptors, such as FAST or ORB techniques, were developed and published afterwards, being applied in our more recent approaches (refer to sections 3.3 and 4.3).

Finally, we also model the uncertainty of the pose estimate by propagating the uncertainty in the 3D positions of the observed points.

Our second proposal (see section 3.3) falls into the category of visual odometers that implement iterative optimization methods for estimating camera/robot egomotion. We propose an efficient and robust outlier detector (named ERODE) aimed at rejecting outliers in the feature associations employed in stereovisual odometry. Unlike most previous proposals, it neither follows a hypothesis-and-verify approach, nor relies on prior information about the points and achieves results comparable to those by RANSAC. Its fundamental advantage, in comparison to RANSAC, is a significant reduction in the processing time (about one order of magnitude), while still performing almost as good in rejecting outliers for a wide range of outliers ratios. Its main weakness is the need for a decent initial estimation of the unknown model, thus it cannot replace RANSAC in all applications but definitively is a potential alternative wherever such a gross estimate is available. In visual odometry, using the previous camera location as starting point for the new pose is good enough for our method, as will be demonstrated with experiments with both real and simulated datasets.

Finally, we designed DaCOR (Divide and Conquer Outlier Rejector), another alternative method to RANSAC, trying to overcome its computational issues. However, although it theoretically overcomes RANSAC timing performance when the outlier ratio is high, it has limitations that restricts the ratio it can cope with, rendering DaCOR unusable in practice. Nevertheless, it is presented in appendix A as a small research work.

3.2 Closed-form Solution to Stereovisual Odometry

This section, which presents our approach to stereovisual odometry relying on a closed-form solution, is organized as follows. First we depict a brief outline of our proposed method for performing visual odometry, being further described with more detail in section 3.2.2. Finally, we provide experimental results with real input images to validate our approach.

3.2.1 Method overview

Our proposed method, depicted in figure 3.2 and being a particularization of the general scheme shown in figure 3.1, can be summarized by the following stages which will be further expanded in next sections:

1. Search for a set of interest features in a first pair of stereo images, and computation of their corresponding SIFT descriptors.

2. Stereo matching based on the Euclidean distance between descriptors and epipolar geometry restrictions.

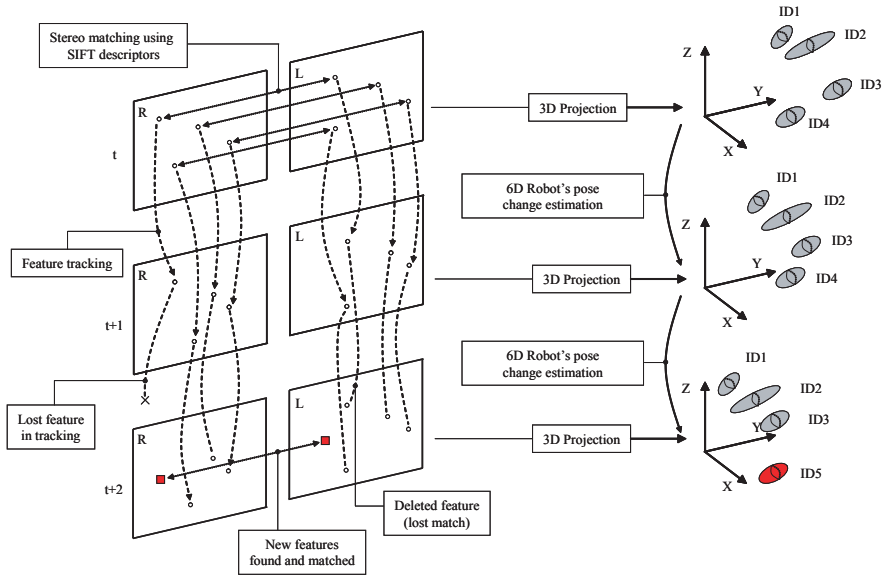


Figure 3.2: A schematic representation of the proposed closed-form method.

3. Projection of the matched features into 3D space, obtaining this way a set of three dimensional points with coordinates relative to the current robot pose.

4. Track the features in the next pair of stereo images. Notice that this tracking allows us to avoid a new SIFT-based matching step.

5. These tracked features are projected into 3D space, yielding a new set of three dimensional points with known correspondences to the previous set of 3D points.

6. Robot/camera 6D change in pose estimation through a closed-form solution of the absolute orientation problem [86], given the correspondences between the two sets of 3D points.

7. If the number of tracked features falls below a certain threshold, new features are searched in the stereo images and their SIFT descriptors computed. Subsequently, they are matched according to their descriptors and added to the current set of points.

8. Repeat from step 4.

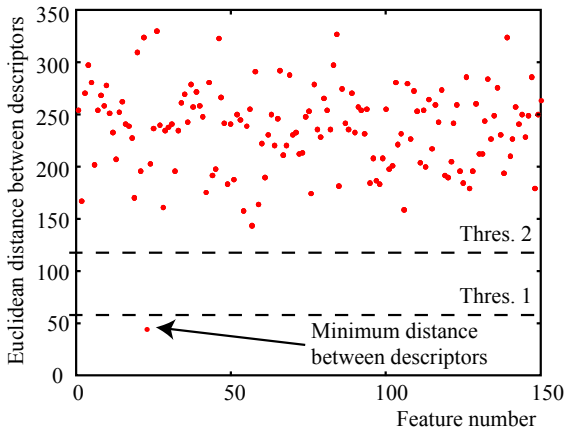


Figure 3.3: Euclidean distance between the SIFT descriptors of an example feature in the left image and all the features in the right one. The two defined thresholds are plotted with dashed lines

3.2.2 Detailed description of the method

Extraction and matching of reliable features from stereo images

This section addresses the detection of interest points in the images through the method proposed by Shi&Tomasi [173] (extensively described in section 2.2.1), since the extracted keypoints are easy to be tracked along the sequence of captured images. In addition, their corresponding SIFT descriptors (introduced in section 2.2.1) are also computed to make them sufficiently distinguishable and to improve the robustness of the stereo matching process. After detecting the set of keypoints in each image, they are matched according to both the similarity of their descriptors and the restriction imposed by the epipolar geometry (refer to sections 2.2.2 and 2.2.3, respectively).

More precisely, in the former restriction, for each keypoint in the left image, the Euclidean distance between its descriptor and those of the keypoints in the right image is computed. For a pair of conjugate keypoints to be considered a candidate match, both the minimum distance must be below a fixed threshold and the second lowest distance must be sufficiently apart from the minimum (see figure 3.3). This method slightly differs from the one proposed by Lowe in [119], which imposes that the ratio of the two minimum distances between descriptors is below a certain threshold. Although this criterion is effective for stereo matching, it may establish correspondences between descriptors that have different levels of distinctiveness, and, therefore, are not easily identifiable. In our proposal, the lower threshold ensures that matched points have very distinctive descriptors, leading to low absolute Euclidean distances, whilst the second threshold is set to avoid ambiguity in correspondences.

In addition, matched points must fulfill the epipolar constraint, i.e. they have to lay on their conjugate epipolar lines. In a stereo vision system with parallel optical

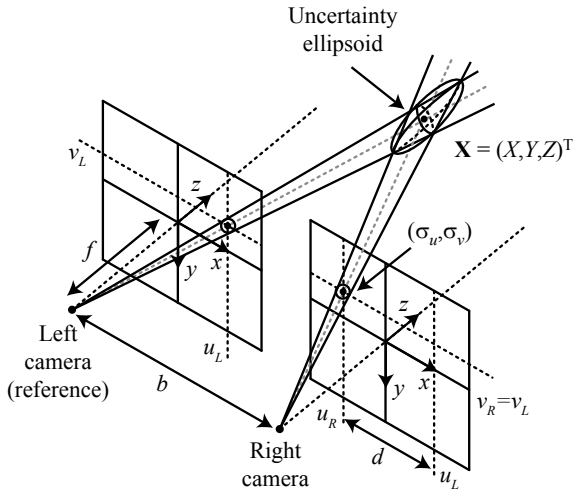


Figure 3.4: Configuration of a stereo vision system and schematic representation of the uncertainty in the localization of a 3D landmark

axis as the one shown in figure 3.4, or with rectified images, epipolar lines are parallel and horizontal, thus, the epipolar constraint reduces to checking that both features are in the same row of the image.

Finally, each pair of matched features is assigned a unique ID which will be used to identify the point projected from their image coordinates in subsequent time-steps.

Feature Projection into 3D Space

Once the features have been robustly matched, the most likely coordinates of their corresponding 3D points are estimated from their positions on the images and the intrinsic parameters of the stereo system by projecting them back to 3D space [83, 170]. We also consider here the uncertainty in the 3D landmark position due to errors in the image quantization and in the feature detection process. Formally, let (u_L^i, v_L^i) be the image coordinates of a feature $\mathbf{x}^i$ in the left image (which will be taken as the reference one) and d^i the disparity of its conjugate feature in the right one. Then, assuming a stereo system with parallel optical axes and a pinhole camera model, as the one shown in figure 3.4, the 3D coordinates $\mathbf{X}^i$ of the projected i-th point are computed as [170] (for the sake of clarity, the i superscript will be dropped in the following formulation):

$$\mathbf{X} = (X,Y,Z)^{\mathsf{T}} = \left(\frac{b\,(u_L - c_{u_L})}{d}, \frac{b\,(v_L - c_{v_L})}{d}, \frac{bf}{d} \right)^{\mathsf{T}}, \tag{3.3}$$

where (c_{u_L}, c_{v_L}) are the image coordinates of the principal point in the reference image, b is the baseline of the stereo system, and f stands for the identical focal length of the cameras.

The errors in the variables u_L, v_L, and d are usually modeled as uncorrelated zero-mean Gaussian noise [122]. We can compute a first-order approximation of the distribution of variables in equation (3.3) as multivariate Gaussians, obtaining the following covariance matrix for the 3D coordinates:

$$\Sigma = \begin{pmatrix} \sigma_X^2 & \sigma_{XY} & \sigma_{XZ} \\ \sigma_{YX} & \sigma_Y^2 & \sigma_{YZ} \\ \sigma_{ZX} & \sigma_{ZY} & \sigma_Z^2 \end{pmatrix} = \mathbf{J} diag\left(\sigma_u^2, \sigma_v^2, \sigma_d^2\right) \mathbf{J}^{\mathrm{T}}, \tag{3.4}$$

where $\mathbf{J}$ stands for the Jacobian matrix of the functions in equation (3.3), and σ_X,σ_Y, σ_Z,σ_u,σ_v and σ_d are the standard deviations of the corresponding variables. Expanding equation (3.4) we come up with the following expression for Σ:

$$\Sigma = \left(\frac{b}{d}\right)^2 \begin{pmatrix} \sigma_u^2 + \frac{(u_L - c_{u_L})^2 \sigma_d^2}{d^2} & \frac{(u_L - c_{u_L}) \sigma_d^2 (v_L - c_{v_L})}{d^2} & \frac{(u_L - c_{u_L}) \sigma_d^2 f}{d^2} \\ \frac{(u_L - c_{u_L}) \sigma_d^2 (v_L - c_{v_L})}{d^2} & \sigma_v^2 + \frac{(v_L - c_{v_L})^2 \sigma_d^2}{d^2} & \frac{(v_L - c_{v_L}) \sigma_d^2 f}{d^2} \\ \frac{(u_L - c_{u_L}) \sigma_d^2 f}{d^2} & \frac{(v_L - c_{v_L}) \sigma_d^2 f}{d^2} & \frac{f^2 \sigma_d^2}{d^2} \end{pmatrix}, \tag{3.5}$$

which models the uncertainty in the 3D coordinates of points computed from the noisy measurements of a stereo system.

Please, note that the uncertainties of the camera intrinsic parameters, i.e. the baseline, the focal length and the principal point coordinates, are not taken into account, since the camera employed in our experiments is supposed to be accurately calibrated by the manufacturer and, therefore, the errors in these parameters may be considered negligible, as is common in the literature. However, they could be easily introduced by linear uncertainty propagation using equations (3.3) and (3.4).

In order to validate this error model, we have performed an experiment where the real density of the landmark location (derived from a Monte-Carlo simulation) has been compared with the approximated density from the linearized model. For that purpose, we have chosen by hand a set of matches in a pair of stereo images and computed their disparity, having in this way the values of u_L, v_L and d in equations (3.3) and (3.5). For each stereo match, the Monte-Carlo simulation has been performed by drawing a set of 10,000 samples from the Gaussian distributions of u_L, v_L and d (assuming a variance of $\sigma_u^2 = \sigma_v^2 = 1$ and $\sigma_d^2 = 2$ pixels, respectively), and by projecting them through equation (3.3), to yield a set of 10,000 samples of the landmark 3D position. These samples accurately model the real density and allow the estimation of the real means and covariances. Figure 3.5 shows the planar projections of the 3D uncertainty ellipses associated to both estimations for one of the landmarks, where the black-thick one corresponds to the Monte-Carlo method and the blue-dotted one corresponds to the linearized model.

To measure the similarity between these two density distributions, we have employed the Kullback-Leibler divergence (D_{KL}) [111], widely employed in statistics

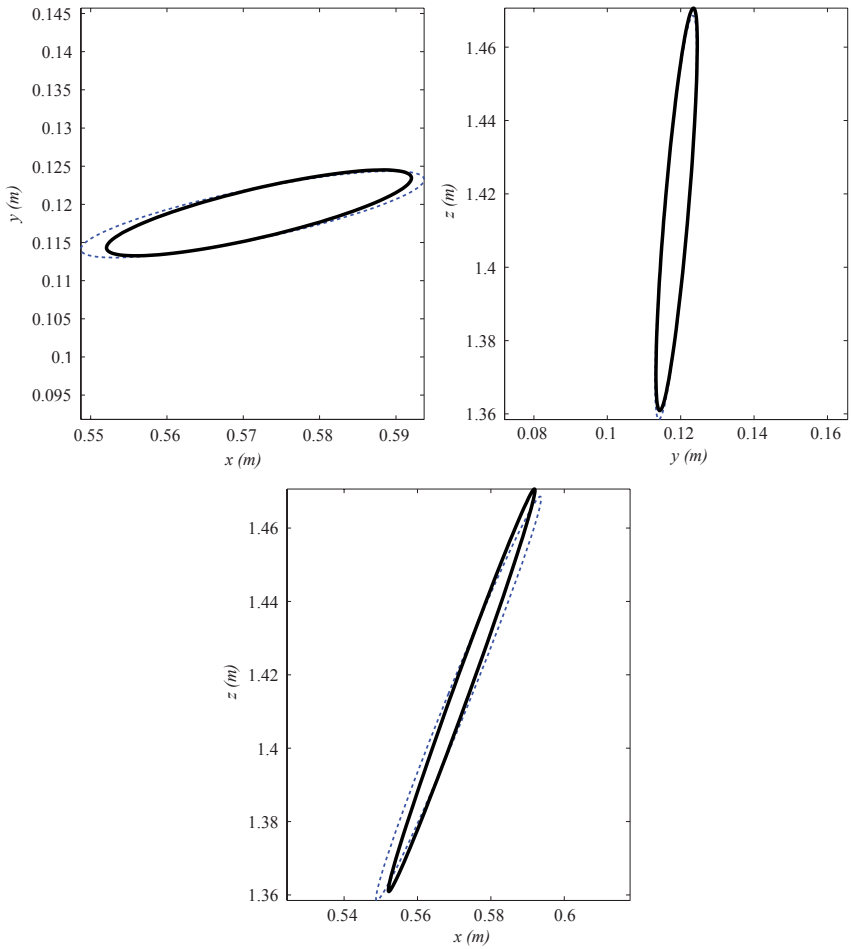


Figure 3.5: Planar projections of the uncertainty ellipses associated to both Monte-Carlo simulation (black-thick line) and linearized error model estimations (blue-dotted line).

for that purpose. We must remark that, since the involved distributions are Gaussians, D_{KL} can be computed through a closed-form solution [85]; this measure yields an average value of 0.36, which can be considered a reasonably good result as it is similar to that obtained from two normalized one-dimensional Gaussians with an offset of about 0.8σ in their means.

We have checked other ways of obtaining the covariance matrix Σ like the Unscented Transformation (UT) [201] and the scaled UT [95]. We have contrasted them to the real density and, since the comparison of the results achieved by the three aforementioned methods (linear, UT and scaled UT) shows similar performance, we have opted for the linear approach (summarized by equations (3.4) and (3.5)) because of its efficiency.

Finally, to distinguish the projected points among them, each 3D point is assigned the unique ID of the matched pair of image features from which it was generated.

Tracking Features

In successive stereo frames, the features are tracked using the Kanade-Lucas-Tomasi (KLT) method [173], explained in section 2.2.1, in order to determine their coordinates in the new pair of stereo images.

The correct tracking of a pair of matched features in the left and right images at time t yields another matched pair of features in the stereo images at time $t+1$. At this point the epipolar constraint is considered to detect improperly tracked features and, hence, to avoid the presence of unreliable matched pairs. By using this tracking process, we avoid both the search for new SIFT features and the descriptor-based stereo matching at the new camera pose. Thus this method speeds up the process of extracting and matching features, reducing, consequently, the computational burden of the whole visual odometry procedure. The resulting set of tracked features are also projected to space following the method described in the previous section yielding a new set of 3D points which keep their IDs from the image features in order to maintain an implicit matching relationship with the points in the previous set.

Finally, if the number of tracked features falls below a certain threshold, the algorithm searches for new features in the images to maintain a proper amount of elements in the 3D point sets.

Probabilistic Estimation of the Pose Change

In this section we present a method for estimating the probability distribution of the change in the robot pose between two time-steps from the sets of 3D points determined as described above.

Formally, let $\mathbf{X}_t$ be a set of N 3D points obtained at time t:

$$\mathbf{X}_t = \left\{\mathbf{X}_t^i\right\}_{i=1\ldots N} \tag{3.6}$$

where the position of each 3D point is assumed to follow a Gaussian distribution with mean $\boldsymbol{\mu}_{\mathbf{X}_t^i} = \left\langle X_t^i, Y_t^i, Z_t^i \right\rangle$ and covariance $\boldsymbol{\Sigma}_{\mathbf{X}_t^i}$ determined by equations 3.3 and 3.5, respectively:

$$\mathbf{X}_t^i \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{X}_t^i}, \boldsymbol{\Sigma}_{\mathbf{X}_t^i}\right) \tag{3.7}$$

Recalling the notation introduced in 3.1.1, now we define the Gaussian random variable $\mathbf{q}_{t,t+1}$ that models the pose change between time-steps t and $t+1$ as a function of the sets of projected 3D points $\mathbf{X}_t$ and $\mathbf{X}_{t+1}$:

$$\mathbf{q}_{t,t+1} \sim \mathcal{N}\left(\boldsymbol{\mu}_q^{t,t+1}, \boldsymbol{\Sigma}_q^{t,t+1}\right) \qquad \mathbf{q}_{t,t+1} = \mathbf{f}(\mathbf{X}_t, \mathbf{X}_{t+1}). \tag{3.8}$$

We propose to compute the mean value $\boldsymbol{\mu}_q$ in equation 3.2 through the method reported by Horn in [86], where it is derived a closed-form solution to the least-squares problem of finding the rigid transformation between two coordinate systems for the coordinates of a number of points in both systems. We use the mean values, $\boldsymbol{\mu}_{\mathbf{X}_t^i}$ and $\boldsymbol{\mu}_{\mathbf{X}_{t+1}^i}$, of the 3D point positions as the input to this algorithm. The advantage of this closed-form solution is that no initial estimation of the change in pose is required, in contrast to visual odometry proposals based on iterative methods [147, 178].

The Horn's method performs as follows:

1. Compute the centroids ($\mathbf{c}_t$ and $\mathbf{c}_{t+1}$) of the two sets of points and subtract them from their coordinates in order to deal only with coordinates relative to their centroids:

$$\mathbf{c}_t = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\mu}_{\mathbf{X}_t^i} \tag{3.9}$$

$$\mathbf{c}_{t+1} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\mu}_{\mathbf{X}_{t+1}^i} \tag{3.10}$$

$$\overline{\boldsymbol{\mu}}_{\mathbf{X}_t^i} = \left(\overline{X}_t^i, \overline{Y}_t^i, \overline{Z}_t^i\right)^{\mathrm{T}} = \boldsymbol{\mu}_{\mathbf{X}_t^i} - \mathbf{c}_t \tag{3.11}$$

$$\overline{\boldsymbol{\mu}}_{\mathbf{X}_{t+1}^i} = \left(\overline{X}_{t+1}^i, \overline{Y}_{t+1}^i, \overline{Z}_{t+1}^i\right)^{\mathrm{T}} = \boldsymbol{\mu}_{\mathbf{X}_{t+1}^i} - \mathbf{c}_{t+1} \tag{3.12}$$

2. For the i-th 3D point, compute the following nine products of its coordinates at time t and $t+1$:

$$\begin{array}{ll} P_{XX}^i = \overline{X}_t^i \overline{X}_{t+1}^i & P_{YX}^i = \overline{Y}_t^i \overline{X}_{t+1}^i \\ P_{XY}^i = \overline{X}_t^i \overline{Y}_{t+1}^i & P_{YY}^i = \overline{Y}_t^i \overline{Y}_{t+1}^i \\ P_{XZ}^i = \overline{X}_t^i \overline{Z}_{t+1}^i & P_{YZ}^i = \overline{Y}_t^i \overline{Z}_{t+1}^i \\ P_{ZX}^i = \overline{Z}_t^i \overline{X}_{t+1}^i & P_{ZY}^i = \overline{Z}_t^i \overline{Y}_{t+1}^i \\ P_{ZZ}^i = \overline{Z}_t^i \overline{Z}_{t+1}^i & \end{array} \tag{3.13}$$

3. Accumulate the products in equation (3.13) for all the 3D points to end up with the following nine values:

$$\begin{aligned} S_{XX} &= \sum_i P^i_{XX} \quad S_{YX} = \sum_i P^i_{YX} \\ S_{XY} &= \sum_i P^i_{XY} \quad S_{YY} = \sum_i P^i_{YY} \\ S_{XZ} &= \sum_i P^i_{XZ} \quad S_{YZ} = \sum_i P^i_{YZ} \\ S_{ZX} &= \sum_i P^i_{ZX} \quad S_{ZY} = \sum_i P^i_{ZY} \\ S_{ZZ} &= \sum_i P^i_{ZZ} \end{aligned} \tag{3.14}$$

4. Form a 4x4 symmetric matrix with the elements in equation (3.14):

$$\mathbf{N} = \begin{pmatrix} N_{11} & N_{12} & N_{13} & N_{14} \\ N_{21} & N_{22} & N_{23} & N_{24} \\ N_{31} & N_{32} & N_{33} & N_{34} \\ N_{41} & N_{42} & N_{43} & N_{44} \end{pmatrix} \tag{3.15}$$

where

$$\begin{aligned} N_{11} &= S_{XX} + S_{YY} + S_{ZZ} \\ N_{12} &= N_{21} = S_{YZ} - S_{ZY} \\ N_{13} &= N_{31} = S_{ZX} - S_{XZ} \\ N_{14} &= N_{41} = S_{XY} - S_{YX} \\ N_{22} &= S_{XX} - S_{YY} - S_{ZZ} \\ N_{23} &= N_{32} = S_{XY} + S_{YX} \\ N_{24} &= N_{42} = S_{ZX} + S_{XZ} \\ N_{33} &= -S_{XX} + S_{YY} - S_{ZZ} \\ N_{34} &= N_{43} = S_{YZ} + S_{ZY} \\ N_{44} &= -S_{XX} - S_{YY} + S_{ZZ} \end{aligned} \tag{3.16}$$

5. Find the eigenvector corresponding to the largest eigenvalue of $\mathbf{N}$, which will be the quaternion that determines the optimal rotation between the two sets of points.

6. Compute the rotation matrix ($\mathbf{R} \in SO(3)$) associated to the so obtained quaternion, and compute the translation $\mathbf{t} = (\Delta x, \Delta y, \Delta z)^T$ as the difference between the centroid at time t and the rotated centroid at time $t+1$, thus completing the sought SE(3) transformation:

$$\mathbf{t} = \mathbf{c}_t - \mathbf{R}\mathbf{c}_{t+1} \tag{3.17}$$

7. Finally, we extract the values of the increments in *yaw*, *pitch*, and *roll* angles $\langle \Delta\alpha, \Delta\beta, \Delta\gamma \rangle$ between poses from the rotation matrix $\mathbf{R}$, having in this way all the components of $\boldsymbol{\mu}_q$.

Covariance matrices are usually obtained through a linear approximation of the functions involved in a given transformation between variables (see, for example, section 3.2.2). However, linearizing the closed-form formulation described above implies to determine Jacobians of complex functions including eigenvectors computation [155] and transformations of quaternions into rotation matrices [45]. Instead, we propose here to use the linearized version of the problem for estimating the covariance matrix of the pose increment $\mathbf{\Sigma}_q$, iterative in nature, but which does not need to iterate just for evaluating the uncertainty at the optimal solution found by Horn's method.

The rigid transformation that relates two sets of N corresponding points in two time-steps t and $t+1$ can be defined in terms of the random variable $\mathbf{q}_{t,t+1}$ that represents the change in pose:

$$\mathbf{X}_{t+1}^i = \mathbf{h}_i\left(\mathbf{q}_{t,t+1}, \mathbf{X}_t^i\right) = \mathbf{q}_{t,t+1} \oplus \mathbf{X}_t^i \qquad \{i = 1,\ldots,N\}, \tag{3.18}$$

with $\oplus$ representing the point-pose composition operator. This system becomes linear when using homogeneous coordinates, such that:

$$\begin{pmatrix} X_{t+1}^i \\ Y_{t+1}^i \\ Z_{t+1}^i \\ 1 \end{pmatrix} = \left(\begin{array}{ccc|c} & \mathbf{R} & & \mathbf{t} \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} X_t^i \\ Y_t^i \\ Z_t^i \\ 1 \end{pmatrix}, \tag{3.19}$$

The blocks $\mathbf{R}$ and $\mathbf{t}$ stand for the rotation matrix and the translation vector associated to the change in pose represented by $\boldsymbol{\mu}_q$, respectively.

Now, we define a cost function that models the fitting error of the two sets of N 3D associated points according to the given 6D change in pose:

$$F\left(\boldsymbol{\mu}_q, \mathbf{X}_t\right) = \sum_i \Delta\mathbf{z}_i^{\mathrm{T}} \mathbf{\Lambda}_i \Delta\mathbf{z}_i. \tag{3.20}$$

Note that, for the sake of clarity in subsequent formulations that include subscripts and superscripts, we have changed the notation from $\boldsymbol{\mu}_{\mathbf{X}_t}$ to $\mathbf{X}_t$ (and $\boldsymbol{\mu}_{\mathbf{X}_{t+1}}$ to $\mathbf{X}_{t+1}$), the latter now becoming the means of the random variables that represent the positions of the N 3D points at time-step t (and $t+1$). Following this notation, $\Delta\mathbf{z}_i$ in equation (3.20) represents the individual error between the coordinates of the i-th associated points $\mathbf{X}_{t+1}^i$ and $\mathbf{X}_t^i$:

$$\Delta\mathbf{z}_i = \mathbf{X}_{t+1}^i - \mathbf{X}_t^i = \mathbf{h}_i\left(\boldsymbol{\mu}_q, \mathbf{X}_t^i\right) - \mathbf{X}_t^i, \tag{3.21}$$

and $\mathbf{\Lambda}^i$ in equation (3.20) stands for the individual information matrices of the errors. These, in turn, can be derived from the individual covariance matrices of the random variables that models the 3D point positions as follows:

$$\mathbf{\Lambda}^i = \left(\mathbf{\Sigma}^i\right)^{-1} = \left(\mathbf{\Sigma}_{\mathbf{X}_{t+1}^i} + \mathbf{\Sigma}_{\mathbf{X}_t^i}\right)^{-1}, \tag{3.22}$$

since the error has been defined as the subtraction of the corresponding random variables, as shown in equation (3.21).

On the other hand, the $[6+3N \times 6+3N]$ Hessian matrix $\mathbf{H}$ can be approximated from the Jacobian $\mathbf{J}$ through the expression:

$$\mathbf{H} \approx \mathbf{J}^{\mathrm{T}} \mathbf{\Lambda} \mathbf{J}, \tag{3.23}$$

where $\mathbf{\Lambda}$ is a $[3N \times 3N]$ block diagonal matrix containing the individual information matrices for each pair of associated points:

$$\mathbf{\Lambda} = \begin{pmatrix} \mathbf{\Lambda}^1 & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{\Lambda}^N \end{pmatrix}. \tag{3.24}$$

The Jacobian matrix, in turn, is formed by N row blocks, each one corresponding to a pair of associated points:

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}^1 \\ \vdots \\ \mathbf{J}^N \end{pmatrix}. \tag{3.25}$$

Each individual Jacobian $\mathbf{J}^i$ comprises the partial derivatives of $\mathbf{h}_i(\cdot)$ with respect to the elements of both $\boldsymbol{\mu}_q$ and $\mathbf{X}_t$:

$$\mathbf{J}^i = \left(\frac{\partial \mathbf{h}_i}{\partial \boldsymbol{\mu}_q} \;\middle|\; \frac{\partial \mathbf{h}_i}{\partial \mathbf{X}^1} \quad \cdots \quad \frac{\partial \mathbf{h}_i}{\partial \mathbf{X}^N} \right) \tag{3.26}$$

$$= \left(\mathbf{J}_q^i \;\middle|\; \mathbf{J}_{\mathbf{X}^1}^i \quad \cdots \quad \mathbf{J}_{\mathbf{X}^N}^i \right). \tag{3.27}$$

In this case, the $\mathbf{J}_q^i$ and $\mathbf{J}_{\mathbf{X}^i}^i$ sub-matrices have a dimensionality of $[3 \times 6]$ and $[3 \times 3]$, respectively, yielding a total size for the individual Jacobian of $[3 \times 6 + 3N]$ and generating a complete Jacobian $\mathbf{J}$ of dimension $[3N \times 6 + 3N]$. However, note that $\mathbf{J}_{\mathbf{X}^j}^i$ sub-matrices are only non-zero when $i = j$, so the Jacobian in equation (3.25) becomes highly sparse:

$$\mathbf{J} = \left(\begin{array}{c|ccc} \mathbf{J}_q^i & \mathbf{J}_{\mathbf{X}^1}^1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{J}_q^N & \mathbf{0} & \cdots & \mathbf{J}_{\mathbf{X}^N}^N \end{array} \right). \tag{3.28}$$

For clarity in the following formulation, we will drop one of the identical indexes that indicates the associated pair of points, i.e. $\mathbf{J}_{\mathbf{X}}^i := \mathbf{J}_{\mathbf{X}^i}^i$ from now on. Then, by

substituting equations (3.24) and (3.28) into equation (3.23), the Hessian matrix can be now derived through:

$$
\begin{aligned}
\mathbf{H} &\approx \mathbf{J}^{\mathsf{T}}\boldsymbol{\Lambda}\mathbf{J} \\
&= \left(\begin{array}{c|ccc} \mathbf{J}_q^{1\mathsf{T}} & \cdots & \mathbf{J}_q^{N\mathsf{T}} \\ \hline \mathbf{J}_{\mathbf{X}}^{1\mathsf{T}} & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{J}_{\mathbf{X}}^{N\mathsf{T}} \end{array}\right)\left(\begin{array}{ccc} \boldsymbol{\Lambda}^1 & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \boldsymbol{\Lambda}^N \end{array}\right)\left(\begin{array}{c|ccc} \mathbf{J}_q^i & \mathbf{J}_{\mathbf{X}}^1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{J}_q^N & \mathbf{0} & \cdots & \mathbf{J}_{\mathbf{X}}^N \end{array}\right) \\
&= \left(\begin{array}{c|ccc} \mathbf{J}_q^{1\mathsf{T}}\boldsymbol{\Lambda}^1 & \cdots & \mathbf{J}_q^{N\mathsf{T}}\boldsymbol{\Lambda}^N \\ \hline \mathbf{J}_{\mathbf{X}}^{1\mathsf{T}}\boldsymbol{\Lambda}^1 & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{J}_{\mathbf{X}}^{N\mathsf{T}}\boldsymbol{\Lambda}^N \end{array}\right)\left(\begin{array}{c|ccc} \mathbf{J}_q^i & \mathbf{J}_{\mathbf{X}}^1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{J}_q^N & \mathbf{0} & \cdots & \mathbf{J}_{\mathbf{X}}^N \end{array}\right) \\
&= \left(\begin{array}{c|ccc} \mathbf{J}_q^{1\mathsf{T}}\boldsymbol{\Lambda}^1\mathbf{J}_q^1 + \cdots + \mathbf{J}_q^{N\mathsf{T}}\boldsymbol{\Lambda}^N\mathbf{J}_q^N & \mathbf{J}_q^{1\mathsf{T}}\boldsymbol{\Lambda}^1\mathbf{J}_{\mathbf{X}}^1 & \cdots & \mathbf{J}_q^{N\mathsf{T}}\boldsymbol{\Lambda}^N\mathbf{J}_{\mathbf{X}}^N \\ \hline \mathbf{J}_{\mathbf{X}}^{1\mathsf{T}}\boldsymbol{\Lambda}^1\mathbf{J}_q^1 & \mathbf{J}_{\mathbf{X}}^{1\mathsf{T}}\boldsymbol{\Lambda}^1\mathbf{J}_{\mathbf{X}}^1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{J}_{\mathbf{X}}^{N\mathsf{T}}\boldsymbol{\Lambda}^N\mathbf{J}_q^N & \mathbf{0} & \cdots & \mathbf{J}_{\mathbf{X}}^{N\mathsf{T}}\boldsymbol{\Lambda}^N\mathbf{J}_{\mathbf{X}}^N \end{array}\right) \\
&= \left(\begin{array}{c|ccc} \sum_i \mathbf{J}_q^{i\mathsf{T}}\boldsymbol{\Lambda}^i\mathbf{J}_q^i & \mathbf{J}_q^{1\mathsf{T}}\boldsymbol{\Lambda}^1\mathbf{J}_{\mathbf{X}}^1 & \cdots & \mathbf{J}_q^{N\mathsf{T}}\boldsymbol{\Lambda}^N\mathbf{J}_{\mathbf{X}}^N \\ \hline \mathbf{J}_{\mathbf{X}}^{1\mathsf{T}}\boldsymbol{\Lambda}^1\mathbf{J}_q^1 & \mathbf{J}_{\mathbf{X}}^{1\mathsf{T}}\boldsymbol{\Lambda}^1\mathbf{J}_{\mathbf{X}}^1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{J}_{\mathbf{X}}^{N\mathsf{T}}\boldsymbol{\Lambda}^N\mathbf{J}_q^N & \mathbf{0} & \cdots & \mathbf{J}_{\mathbf{X}}^{N\mathsf{T}}\boldsymbol{\Lambda}^N\mathbf{J}_{\mathbf{X}}^N \end{array}\right).
\end{aligned}
\tag{3.29}
$$

Since the individual information matrices $\boldsymbol{\Lambda}^i$ are symmetrical, we can assume that $\left(\mathbf{J}_q^{1\mathsf{T}}\boldsymbol{\Lambda}^i\mathbf{J}_{\mathbf{X}}^1\right)^{\mathsf{T}} = \mathbf{J}_{\mathbf{X}}^{i\mathsf{T}}\boldsymbol{\Lambda}^i\mathbf{J}_q^i$, and, thereby, the symmetrical Hessian matrix structure can be represented as follows:

$$
\mathbf{H} = \left(\begin{array}{c|c} \mathbf{A} & \mathbf{B} \\ \hline \mathbf{B}^{\mathsf{T}} & \mathbf{C} \end{array}\right). \tag{3.30}
$$

As we are only interested in estimating the covariance matrix of the random variable $\mathbf{q}$ (which represents the change in pose between time-steps t and $t+1$), we proceed to compute the Schür complement [152] of the $\mathbf{A}$ block in the Hessian matrix, which corresponds to the elements in $\boldsymbol{\mu}_q$, yielding the information matrix $\boldsymbol{\Lambda}_q$:

$$
\begin{aligned}
\boldsymbol{\Lambda}_q &= \mathbf{A} - \mathbf{B}\mathbf{C}^{-1}\mathbf{B}^{\mathsf{T}} \\
&= \sum_i \mathbf{J}_q^{i\mathsf{T}}\boldsymbol{\Lambda}^i\mathbf{J}_q^i - \sum_i \left(\mathbf{J}_q^{i\mathsf{T}}\boldsymbol{\Lambda}^i\mathbf{J}_{\mathbf{X}}^i\right)\left(\mathbf{J}_{\mathbf{X}}^{i\mathsf{T}}\boldsymbol{\Lambda}^i\mathbf{J}_{\mathbf{X}}^i\right)^{-1}\left(\mathbf{J}_{\mathbf{X}}^{i\mathsf{T}}\boldsymbol{\Lambda}^i\mathbf{J}_q^i\right).
\end{aligned}
\tag{3.31}
$$

Note that, due to the fact that $\mathbf{C}$ is block diagonal, its inverse is also a block diagonal matrix whose elements are the inverse of the elements in $\mathbf{C}$. This highly simplifies the computation of $\boldsymbol{\Lambda}_q$ since it only entails the addition, inversion and multiplication of small matrices of dimension $[6\times3]$, $[3\times3]$ and $[3\times6]$, being performed in $O(N)$, with N being the number of 3D involved points.

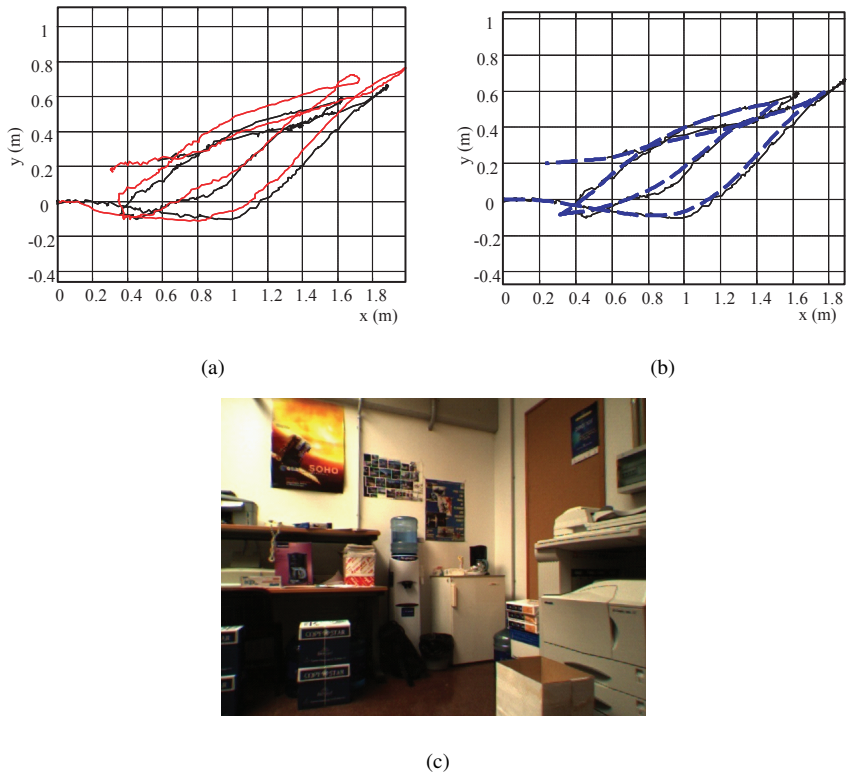


Figure 3.6: (a) Path of the robot estimated from the laser scanner built map (black line) and our proposed visual odometry method (red line). (b) Estimated paths from the laser scanner measurements and the encoder-based odometry readings (blue-dashed line). (c) Example of the images employed in the experiments.

Finally, the desired covariance matrix $\mathbf{\Sigma}_q$ is determined by the inverse of the so-computed information matrix:

$$\mathbf{\Sigma}_q = \left(\mathbf{\Lambda}_q\right)^{-1}. \tag{3.32}$$

3.2.3 Experimental results

We have performed a variety of experiments to compare classical encoder-based odometry with our proposed method for visual odometry in an indoor environment. In this section, we present one of them where Sancho [68], one of the mobile robots developed by MAPIR, along with SENA [70] and Rhodon [94], is equipped with a PointGrey Bumblebee stereo camera and driven through a room while gathering stereo images and odometry readings. We also use laser scans to build a map of the environment and estimate the real path of the robot, which will be taken as the ground

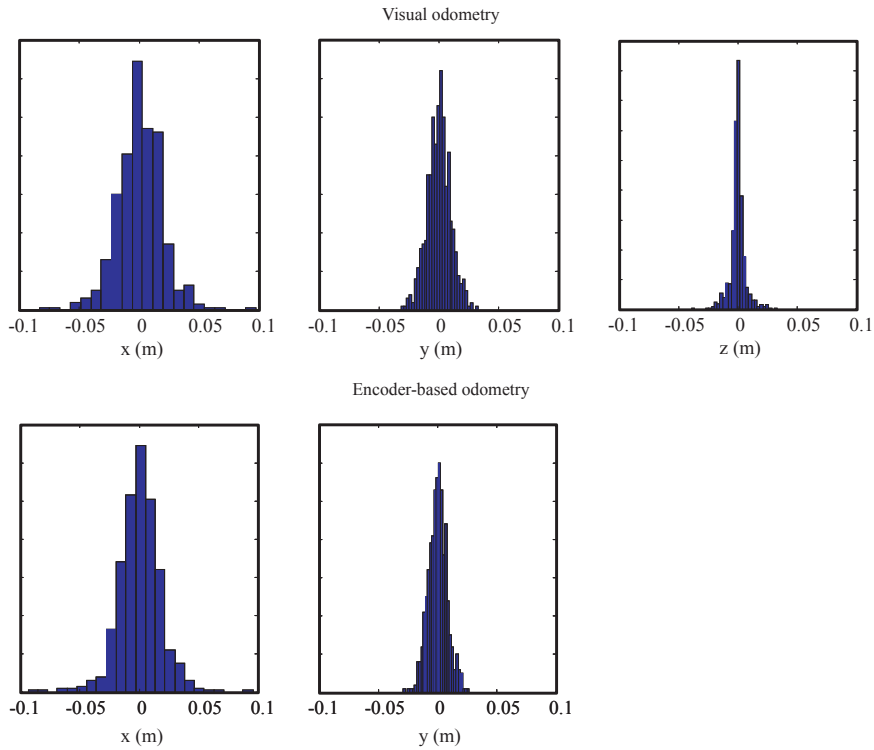


Figure 3.7: Histograms of the errors committed in the estimation of the changes in the robot position for the visual odometry (top plots) and classical odometry (bottom plots) approaches.

truth in this experiment (black lines in figures 3.6a-b). An example of the images managed in this experiment is shown in figure 3.6c.

In order to compare the performance of the odometry techniques, we compute the errors incurred by both methods at each time-step as the difference between their estimates and the ground truth.

The histograms of the 3D position errors of both approaches are shown in figure 3.7. We have found that both methods perform similarly, with most of the errors in Δx and Δy below 5 cm. Notice that since the robot moves in a planar environment, Δz should be zero for the whole experiment. Consequently, our algorithm provides a coherent estimation which is always close to $\Delta z = 0$ with a small error (typically 1 cm), as can be seen in figure 3.7. The distribution of the error in the 3D position is illustrated in the last plot in figure 3.8.

Regarding the estimation of the orientation, visual odometry achieves an error in *yaw* (the only rotational degree of freedom of a planar robot) similar to conventional odometry. However, we should highlight the accuracy of our algorithm in the other

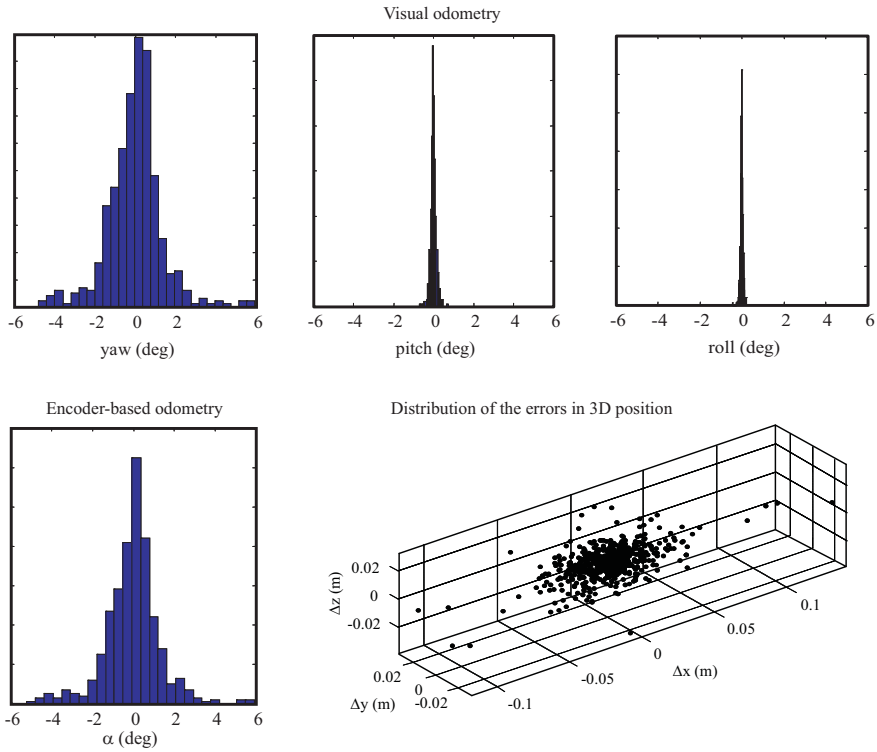


Figure 3.8: Histograms of the errors committed in the estimation of the changes in the robot orientation for the visual odometry (top plots) and conventional encoder-based odometry (bottom-left plot) approaches. (bottom-right) Distribution of the errors in the estimation of the change in the robot 3D position for the visual odometry approach.

components of the orientation, where the largest error is below 1 deg (please, refer to the histograms for *pitch* and *roll* in figure 3.8).

Recalling the estimated paths of the robot in figure 3.6 according to both odometry methods, we can now remark their similar accuracy in spite of the higher dimensionality of visual odometry, which, *a priori*, is prone to accumulate larger errors. We can conclude that the reason for this performance is the small estimation errors of visual odometry in the dimensions not involved in planar odometry, i.e. Δz, $\Delta\beta$, $\Delta\gamma$.

3.3 Iterative Robust Solution to Stereovisual Odometry

Here we outline our iterative formulation-based solution for stereovisual odometry and introduce the notation employed throughout this section.

3.3.1 Method statement

The proposed method particularizes the general scheme introduced in section 3.1.1 with an iterative optimization process that estimates the robot/camera ego-motion. In this case, we disregard the computation of the covariance matrix of the change in pose, although it could be easily computed from the Hessian matrix following the same procedure described for the previous method in section 3.2.2. Thus, after detecting features in the stereo images in both time-steps and performing data association (steps 1-4 in the general scheme depicted in 3.1.1), the procedure evolves as explained next.

Formally, let $\mathbf{p}$ be the mean of the Gaussian random variable that models the change in pose for certain t, $t+1$ time-steps, i.e. $\mathbf{p} := \mu_q^{t,t+1}$ in equation (3.1). Since it will be estimated iteratively, a $[i]$ superscript will be added to indicate the iteration number of the optimization process, while the time-step superscripts will be dropped as the process will remain the same for any pair of consecutive time-steps. In this section, $\mathbf{p}$ will also be denoted as the *model*.

First, we choose an initial estimation $\mathbf{p}^{[0]}$ for the full 6D motion of the stereo camera which, among with the observation, will constitute the input data to the process of computing the camera ego-motion. Under the mild assumption of a small change in pose between frames, an all-zeros initial estimation

$$\mathbf{p}^{[0]} = (0,0,0,0,0,0)^{\mathsf{T}} \tag{3.33}$$

will be sufficient.

Then, we triangulate the N matches in I_t into the 3D space (yielding a set $\mathbf{x}$ of 3D points) and back-project them on I_{t+1} according to the current estimation of the motion. The function $\mathbf{h}$ that relates the current model $\mathbf{p}$ to the coordinates of the features in I_{t+1} is known as the *prediction function* and such coordinates form the *prediction*:

$$\mathbf{h}(\mathbf{p},\mathbf{x}) = \overline{\mathbf{z}} = \{\overline{\mathbf{z}}_i\}_{i=1,\ldots,N}. \tag{3.34}$$

Finally, we iteratively improve the motion estimation $\mathbf{p}$ by triangulating the matches again until the mismatch between the predictions and observations converges to a minimum. Such mismatch is usually measured as the Mahalanobis distance (a weighted squared error, after all) between the predicted and the observed positions of the detected features in the images.

The iterative minimization of the mismatch can be achieved with a Maximum Likelihood Estimator (MLE) which, essentially, selects the model $\mathbf{p}^*$ for which the total probability of the observed data becomes the highest. If the data are only corrupted by zero-mean Gaussian noise, the MLE coincides with a nonlinear least-squares estimator which minimizes a cost function like:

$$F(\mathbf{p}) = \sum_i \frac{1}{2} \Delta \mathbf{z}_i^T \mathbf{W}_i \Delta \mathbf{z}_i, \tag{3.35}$$

where $\mathbf{p}$ is the vector of model parameters, $\Delta \mathbf{z}_i = \bar{\mathbf{z}}_i - \mathbf{z}_i$ stands for the error between the prediction of the i-th feature and its observation, and $\mathbf{W}_i$ is an appropriate weighting matrix, proportional to the inverse covariance of the normally distributed noise. Assuming identical error distributions for all detected features in both x and y directions leads to the simplification $\mathbf{W}_i = \frac{1}{\sigma_p^2}\mathbf{I}, \forall i$.

3.3.2 Gauss-Newton nonlinear least-squares minimization

The aim of the minimization process is to find a suitable model $\mathbf{p}^*$ that produces the minimum combined error for all the observed features. In general, cost functions present non-linearities that prevent obtaining a closed-form solution, thus a truncated Taylor series expansion of the function is iteratively minimized instead, re-linearizing the approximation around the current model estimation until convergence. At each iteration $[i+1]$ we look for a small change $\Delta \mathbf{p}^{[i+1]}$ in the model for updating its estimation like $\mathbf{p}^{[i+1]} = \mathbf{p}^{[i]} + \Delta \mathbf{p}^{[i+1]}$. It can be demonstrated [55] that the optimal step arises by solving:

$$\mathbf{H} \Delta \mathbf{p} = -\mathbf{g}. \tag{3.36}$$

Here, $\mathbf{H}$ stands for the Hessian matrix, $\mathbf{g}$ is the gradient vector of the cost function $F(\mathbf{p})$, and the iteration superscripts have been dropped for clarity. However if we consider equation (3.35) to be the cost function and under the assumption that the prediction errors $\Delta \mathbf{z}_i$ are small, we find:

$$\mathbf{H} \approx \mathbf{J}^T \mathbf{W} \mathbf{J} \tag{3.37}$$

$$\mathbf{g} = \mathbf{J}^T \mathbf{W} \Delta \mathbf{z}, \tag{3.38}$$

where $\mathbf{J} = \partial \mathbf{h} / \partial \mathbf{p}$ stands for the Jacobian matrix of the prediction model, and $\Delta \mathbf{z}$ is a block-column vector containing the errors $\Delta \mathbf{z}_i$ of the individual predictions, hence converting equation (3.36) into

$$\left(\mathbf{J}^T \mathbf{W} \mathbf{J} \right) \Delta \mathbf{p} = -\mathbf{J}^T \mathbf{W} \Delta \mathbf{z}, \tag{3.39}$$

which expresses the relation between the update $\Delta \mathbf{p}$ of the estimated model in terms of the error $\Delta \mathbf{z}$ (often referred as the *residual*) and the prediction function $\mathbf{h}$.

Note that this formulation is exactly the same as the core of the bundle adjustment framework presented in 2.3.2, except for the fact that, here, the state only includes the robot pose $\mathbf{p}$ instead of both the robot pose and the landmarks positions that comprises the full system state. Since the purpose of visual odometry is to estimate the robot ego-motion, map management is not needed and hence the landmarks positions are disregarded in the solution. Furthermore, their 3D coordinates are triangulated from their image projections and are considered *fixed*, hence not being optimized.

3.3.3 Outlier rejection: the RANSAC estimator

Due to inaccuracies in interest point detectors and mismatches in the data association process, the presence of outliers is inevitable and they should be removed from the input data in order to accurately estimate the camera motion. The most widely employed method to overcome this issue is RANSAC.

RANSAC performs by randomly selecting a minimal set of individual observations from the input data (three observations for stereovisual odometry) which leads the Gauss-Newton least-squares minimization process until an estimation of the model, called the hypothesis, is computed. Subsequently, the residuals of the rest of the data are computed subject to the hypothesized model and, according to a defined threshold, the data are classified in *inliers* or *outliers*, being the set of inliers referred as the consensus set (CS) and its cardinality as the support for the hypothesis.

This process is repeated a certain number of times and the hypothesis with the highest support is selected as the best estimation $\mathbf{p}^*$ for the model.

The theoretical number of hypotheses (n_h) that RANSAC needs to explore in order to find an outlier-free consensus set with a certain level of confidence follows this expression [83]:

$$n_h \geq \left\lceil \frac{log\,(1-q)}{log\,(1-\varepsilon^m)} \right\rceil, \tag{3.40}$$

with ε being the estimated percentage of inliers in the data and q the probability of obtaining an outlier-free CS. For instance, with $m = 3$ as in stereovisual odometry, to achieve a 95% of probability of getting an outlier-free consensus set, the value of n_h spans from 2 to 2995 for an inlier percentage of 5% to 90%, respectively. However, these number of hypotheses have often been considered to be overoptimistic and, in practice, they must be increased to cope with degenerate configurations in the data. This issue renders the capability of RANSAC of dealing with data contaminated with a large ratio of outliers when used in real-time applications.

3.3.4 Our proposal: ERODE

In this work we propose a novel, efficient method for stereovisual odometry based on robust kernels to perform fast and reliable outlier rejection even under conditions of large ratio of outliers. The core of our approach is the usage of a robust radial distribution to model the errors in the data, including both the noise and the outliers.

Outliers produce errors in the data which do not follow a Gaussian distribution and, therefore, their presence unavoidably leads to unreliable results of the MLE due to the fact that the probability of finding gross errors under a Gaussian assumption is extremely low (the tails of a Gaussian rapidly diminish). Here is where a realistic model of the error distribution plays a crucial role for the MLE to be useful, since unmodeled outliers are sufficient to render standard least-squares estimators useless. Mixture of both Gaussians and uniform distributions, or more sophisticated ones as Cauchy or Huber distributions [91], are examples of models for both inliers and outliers (hence defined as *total distributions*). All of these distributions, among with the Gaussian itself, are also known as *radial distributions* and have negative log likelihood of the form:

$$F(\mathbf{p}) = \sum_i \frac{1}{2}\rho_i\left(\Delta\mathbf{z}_i^T\mathbf{W}_i\Delta\mathbf{z}_i\right), \tag{3.41}$$

where $\rho_i(s)$ can be any increasing function that fulfills $\rho_i(0) = 0$ and $\rho_i'(0) = 1$. Note that equation (3.35) is a particular case of this expression with $\rho_i(s) = s$ whilst more robust cost functions are sub-linear in s, often tending to a constant value at ∞. It is important to highlight that the use of robust error distributions would make unnecessary to follow hypothesis-and-verify approaches as they are practically *immune* to the presence of outliers. Nevertheless, although the influence of outliers in the cost function is almost negligible, in practice, better results are achieved if they are detected and removed.

In this work we consider to employ a pseudo-Huber distribution [83] to model the errors and to lead a robustified Gauss-Newton least-squares minimization process which will split the input data set in inliers and outliers. The negative log-likelihood of the probability density function of the pseudo-Huber distribution forms the cost function to be minimized:

$$F_R(\mathbf{p}) = \sum_i \frac{1}{2}\left[2b^2\left(\sqrt{1+\left(\frac{s_i}{b^2}\right)}-1\right)\right], \tag{3.42}$$

with $s_i = \Delta\mathbf{z}_i^T\mathbf{W}_i\Delta\mathbf{z}_i$ and b being a parameter which tunes the shape of the function.

Figure 3.9a shows the comparison between the cost functions of the standard least-squares approach and the pseudo-Huber version with $b = 2$. Note that, for the robustified version, as the error increases (abscissa axis) the contribution of the observation to the cost function (ordinate axis) decreases compared to the standard least-squares case.

It is important to note that using a different cost function than that in equation (3.35) produces a modification in the above mentioned Gauss-Newton expressions. In particular, we need to introduce the first derivative of ρ_i by weighting the gradient vector of the cost function with the vector $\bar{\rho}' = \{\rho_i'\}$ so that outliers contribute more slightly to it:

$$\mathbf{g} = \bar{\rho}'\mathbf{J}^T\mathbf{W}\Delta\mathbf{z}, \tag{3.43}$$

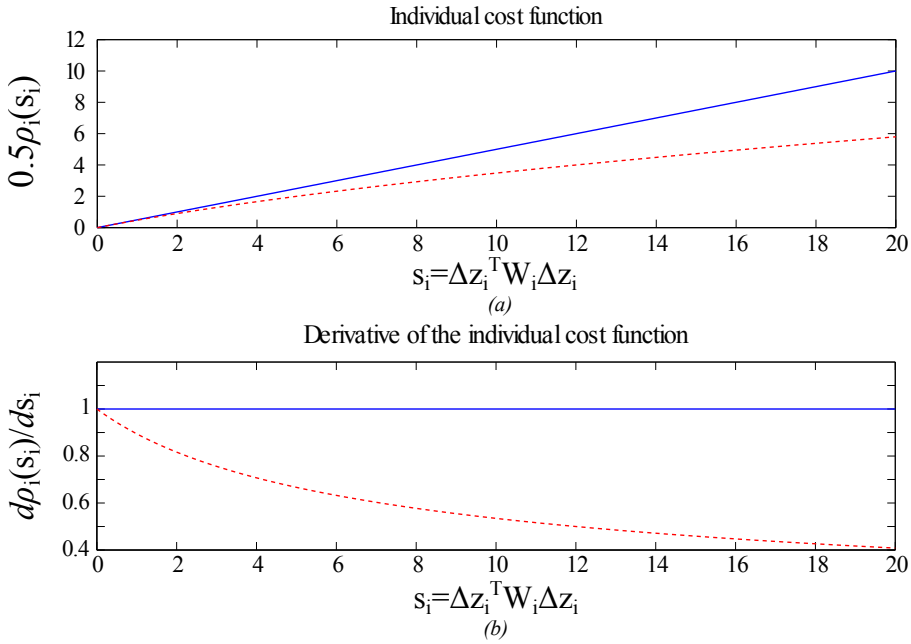


Figure 3.9: (a) Cost functions and (b) the first derivative of ρ_i for a Gaussian (blue-solid) and a pseudo-Huber (red-dashed) distribution.

thus leading to a modified version of equation (3.39):

$$\left(\mathbf{J}^T\mathbf{W}\mathbf{J}\right)\Delta\mathbf{p} = -\bar{\rho}'\mathbf{J}^T\mathbf{W}\Delta\mathbf{z}, \tag{3.44}$$

where, for the pseudo-Huber distribution, we have

$$\rho_i' = \frac{d\rho_i}{ds} = \frac{1}{\sqrt{1+\frac{s}{b^2}}}. \tag{3.45}$$

The effects of this modification are shown in figure 3.9b where it can be noted that the value of ρ_i' decreases as the residual grows, so that, when multiplied by the residual itself in equation (3.44), the contribution of large errors to the gradient vector is attenuated.

This approach implies that all input data are considered in the minimization process but, on the other hand, there is no need to try different hypotheses of the model. With this method, the estimation process naturally converges towards the true solution and, after a few iterations, the outliers appear clearly visible in the vector of residuals so that we can remove them and, subsequently, refine the estimated solution to achieve higher accuracy.

3.3.5 Computational performance

Here we address the computational burden of both RANSAC and our proposed method when estimating the motion of the stereo camera. In this section, the superscripts ^{a} and ^{b} will be used to refer to the RANSAC method and ERODE, respectively.

Let M be the number of elements that form the input data for a certain time-step, being m_{in} of them inliers. RANSAC explores n_h hypotheses and, for each one of them, it picks up m elements from the data and performs a complete minimization process which iteratively computes both the prediction and the Jacobian of the prediction function, spending p_p and p_J seconds per element, respectively. The number of iterations it performs until convergence will be denoted by n_i^a. Then, RANSAC evaluates the rest of the data ($M - m$ elements) against the estimated motion, spending just p_p seconds per element as the Jacobian is not computed here. After the n_h iterations, the largest CS is assumed to be the set of inliers and the computed solution considered the best possible. Finally, it is a common practice to subsequently start a new minimization process with only the set of inliers, and take the best solution so far as the initial estimation, in order to refine the final result. Let n_f^a be the number of iterations this refinement minimization would take. Thus, the computation time that RANSAC would spend to yield the final estimate of a particular motion of the camera would be

$$c^a = n_h \left[n_i^a m \left(p_p + p_J \right) + (M - m) p_p \right] + n_f^a m_{in} \left(p_p + p_J \right)$$

On the other hand, ERODE performs n_i^b iterations of the minimization process with the whole set of input data and, subsequently, the elements whose residual fall over a certain threshold are considered to be outliers. Finally, a refinement minimization process is started with only the inliers, reaching convergence in n_f iterations. Thus, the computational burden for this approach can be expressed as

$$c^b = n_i^b M \left(p_p + p_J \right) + n_f^b m_{in} \left(p_p + p_J \right)$$

In order to quantitatively evaluate the performance, let $p_p = p_J = 1$ be the computational time of each operation, $m = 3$, as stated for a stereovisual odometry application; let also the ratio of outliers (pct_o) to vary between 0.2 and 0.8 so that $m_{in} = (1 - pct_o) M$, and the rest of the parameters to take the realistic values $n_i^a = n_f^a = n_i^b = n_f^b = 4$ and $M = 300$. Figure 3.10 shows the theoretical computational time for both methods as the percentage of outliers grows.

3.3.6 Experimental results

This section presents two experiments which test the performance of our approach for computing stereovisual odometry. The first experiment simulates a typical scenario with a robot traversing an office-like environment while detecting interest points in stereo images while the second one uses the stereo video sequences published in the Karlsruhe outdoor dataset [64], gathered with a moving vehicle.

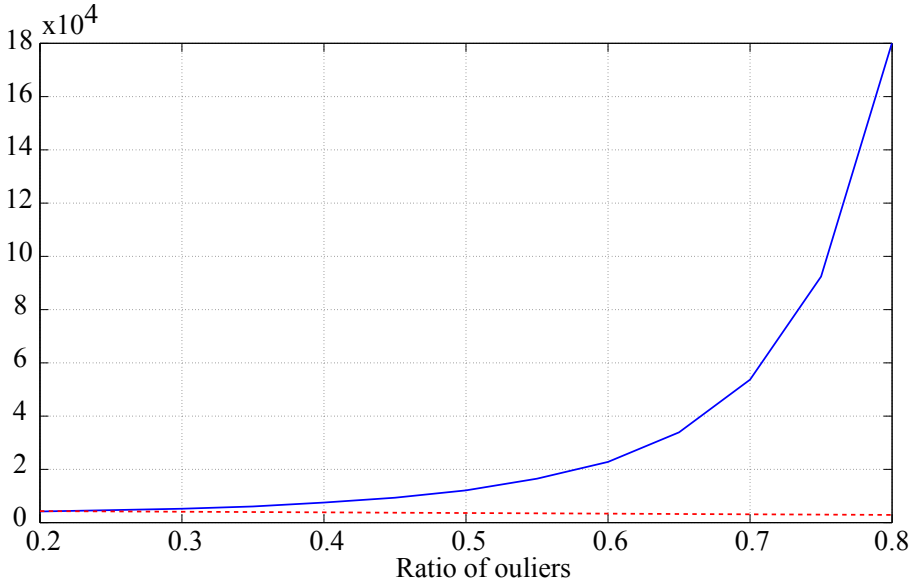


Figure 3.10: Computational time for RANSAC (blue-solid) and ERODE (red-dashed) as the ratio of ouliers grows from 0.2 to 0.8.

In these experiments we have performed a Gauss-Newton minimization process following both ERODE and RANSAC approaches, without taking into account the existence of other techniques (such as Horn's method [86]) that can be employed to estimate the change in pose of a stereo camera between two time-steps. The usage of a maximum likelihood estimator here relies on the purpose of addressing the problem from a more generic approach that can be employed, for instance, for monocular cameras.

Simulated dataset

The use of a simulated dataset to test our approach is motivated by both the availability of a ground truth and the capability of properly control the amount of outliers within the data. This dataset has been created with the freely available Recursive World Toolkit[1] which provides a recursive language to define a 3D virtual world as a set of landmarks and to simulate a projective camera moving within it.

First, we set up a simple experiment where the camera just moved forward for about 15 cm, with no rotation. In order to cover different random sets of outliers, this experiment was repeated 100 times for each ratio of outliers. The error with respect to the ground truth was measured separately for the translation and the rotation where,

[1] `http://code.google.com/p/recursive-world-toolkit/`

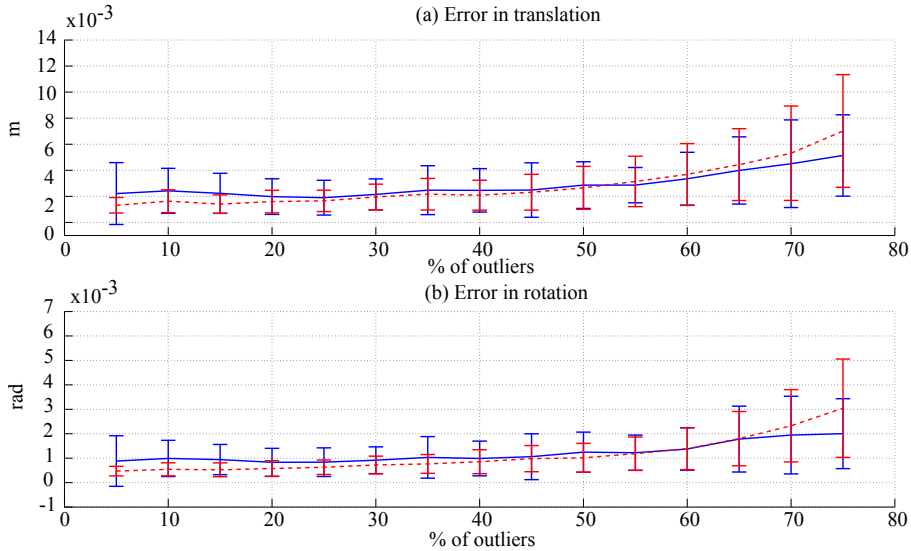


Figure 3.11: Comparison of the errors in (a) translation and (b) rotation for standard RANSAC (blue-solid) and ERODE (red-dashed) for a forward movement with an outlier ratio ranging from 0.05 to 0.75. 2σ confidence bars are also displayed.

in the latter, the Euler angles corresponding to the motion were converted to a 3D *rotation vector* [45]:

$$[\alpha,\beta,\gamma] \rightarrow [w_1,w_2,w_3]\,. \tag{3.46}$$

Thus, the two components of the error may be computed by the Euclidean distance between the real value and estimated one. The average error (with 2σ confidence bars) for the translation and the rotation with respect to the outlier ratio for this experiment are shown in figure 3.11. Note the similarity in the accuracy between RANSAC and our proposal.

Finally, a more complete experiment involving the full motion of the camera was also carried out. In this case, the stereo camera traversed a 50×90 meters virtual environment following a path of about 300-meters long. As the camera moved, the 3D landmarks were projected to the images at each time-step, and their image coordinates (corrupted by zero-mean Gaussian noise with $\sigma = 0.5$ pixels) were stored in a text file. The presence of outliers was simulated by adding uniformly distributed noise in the interval $[-w/2, w/2]$ (with w being the size of the search window mentioned in section 3.1.1) to a randomly selected subset of points. The amount of outliers at each time-step was set according to a pre-defined ratio, which ranged from 0.05 to 0.75.

Figure 3.12 shows the estimated paths for RANSAC (blue-solid), ERODE (red-dashed) and the ground truth (black-dotted) for the case of 50% of outliers, respectively, which has been taken as representative. As can be seen, ERODE yields similar results to RANSAC even when the ratio of outliers is high.

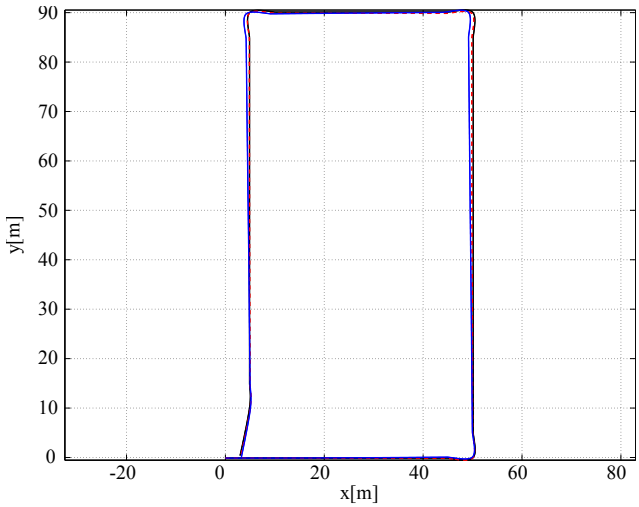


Figure 3.12: Estimated paths with 50% of outliers for RANSAC (blue-solid) and ERODE (red-dashed). Ground truth is plotted in black.

In order to evaluate the computational performance for this simulated dataset, we have measured the time spent by both RANSAC and ERODE methods as the ratio of outliers grows from 0.2 to 0.8, yielding figure 3.13. Note as it fits the theoretical plot shown in figure 3.10.

Finally, another measurement of the goodness of an outlier rejection method is the Receiver Operating Characteristic (ROC) curve (see figure 3.14 for ERODE's ROC curve) and, in particular, the area under the curve (AUC). The ROC curve plots the sensitivity (ratio between the true positives and the total positives) against one minus the specificity (the fraction of false positives out of the total negatives). In general, a method whose AUC value is over 0.96 is considered to have high discriminatory ability and, in our experiments, ERODE achieved an AUC value of 0.9957, getting considerably close to the ideal RANSAC's AUC value of 1.

Outdoor dataset

For a more realistic experiment, we implemented our algorithm inside the LibViso2[2] vision library by Andreas Geiger. This library offers solutions to the problem of estimating visual odometry for both monocular and stereo cameras, and also provides a collection of datasets to test its performance. The selected dataset for this experiment was taken with a pair of cameras mounted on a car following an about 270 meters trajectory within a city. The images had a size of 1344×391 pixels (see figure 3.15 for an example) while the stereo pair presented a baseline of 0.572 meters.

[2] `http://www.cvlibs.net/software/libviso2.html`

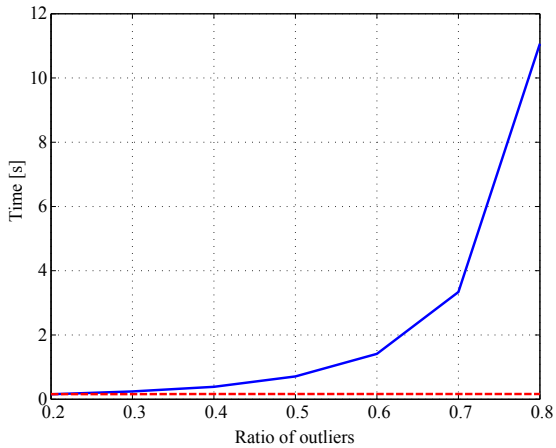


Figure 3.13: Computational time for RANSAC (blue-solid) and ERODE (red-dashed) as the ratio of ouliers grows from 0.2 to 0.8 with the simulated dataset.

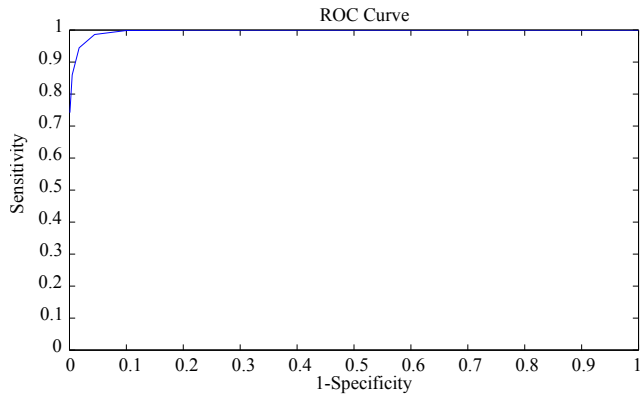


Figure 3.14: ROC curve for ERODE in an experiment with 50% of outliers, taken as a representative example.

Figure 3.15: Example of the images in the outdoor dataset.

Since, in this experiment, the ratio of outliers within the data is not known beforehand, a conservative number of hypotheses for the RANSAC approach must be set in order to minimize the probability of not finding a outlier-free consensus set. In the original code the value for this parameter was set to ensure with 99% of probability that a proper CS is selected even in the presence of about 65% of outliers.

In order to compare the results and the time burden of our approach, we have measured the time that the original RANSAC-based code spent in the estimation of the camera motion between two consecutive time-steps and, subsequently, after replacing such code by our approach as described in section 3.3, we measured it again, yielding the results presented in figure 3.16a.

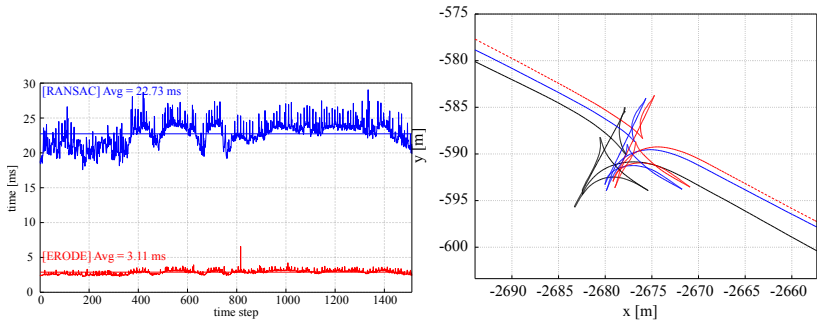
The estimated paths for both approaches are compared in figure 3.16b, showing their similar performance in terms of accuracy, while a comparative video illustrating the development of the experiment can be accessed on-line[3] . Please, note that ERODE considerably reduces the effect of outliers to the final result but such effect cannot be completely canceled since they are still present in the minimization process, hence the small deviation that ERODE incurs with respect to RANSAC.

3.4 Conclusions

This chapter has presented a method to perform visual odometry through a closed-form, optimal solution to the problem of finding the 6D transformation between two sets of corresponding points. The results show that the performance of our approach for visual odometry is quite similar to that of conventional odometry for planar environments, whereas the former additionally allows unrestricted movements in 6D even for non-wheeled robots.

On the other hand, we have dealt with the issue of detecting outliers that affects many computer vision tasks. RANSAC-based robust estimators and other hypothesis-and-verify approaches have been extensively employed in computer vision applications, and, in particular, in visual odometry solutions. Although proven to be highly robust, they suffer from a very high computational burden when the outlier ratio is

[3] `http://youtu.be/pcUhuM3pPOU`



(a) (b)

Figure 3.16: (a) Computation time for each time-step during the experiment when using RANSAC (blue-solid) and ERODE (red-dashed). The overall average time per time-step are also shown as solid horizontal lines. (b) Camera paths as estimated with RANSAC (blue) and ERODE (red). Ground truth is plotted in black.

significant. Here we propose a new approach based on robust functions which avoids the sampling nature of the above-mentioned methods an instead, employs all the input data to detect the outliers, and efficiently perform stereovisual odometry.

The results show similar accuracy than the RANSAC method but reducing its computational cost in about one order of magnitude. Experiments in both simulated and real datasets support this contribution.

Chapter 4
Visual SLAM

Overview

This chapter addresses the SLAM problem for stereo vision systems within two different families of methodologies: filtering and smoothing. First, we will address stereovisual SLAM under the unified formulation of particle filtering. In contrast to most existing approaches to visual SLAM, the presented method does not rely on restrictive camera motion models, but on computing incremental 6D pose differences from the image flow through a probabilistic visual odometry method. Moreover, our observation model, which considers both the 3D positions and the SIFT descriptors of the landmarks, avoids explicit data association between the observations and the map by marginalizing the observation likelihood over all the possible associations. We have experimentally validated our research with a real robot in indoor scenarios.

Our second proposal presents a complete solution for stereovisual SLAM that implements a back-end relying on a modified relative bundle adjustment methodology, and includes a front-end based on ORB keypoints with associated binary descriptors. Data association and loop closure is tackled with a bag-of-words technique that reduces the search area to look for correspondences between observations and the map.

Beliefs don't change facts.
Facts, if you're reasonable, should change your beliefs – Ricky Gervais

4.1 Introduction

Vision systems have acquired growing importance in mobile robotics during the last years due to their low cost and the rich information that cameras provide in comparison with traditional robotic sensors, like laser scanners or sonars. Vision-based systems are employed in a wide range of robotic applications such as object recognition [9, 10, 118], obstacle avoidance [87, 125, 167], navigation [137, 146], topological global localization [108, 198] and, more recently, in simultaneous localization and mapping (SLAM) [41, 176, 178], which has become a prominent research area in mobile robotics since the early nineties.

SLAM is one of the most challenging open problems for developing truly autonomous robots. As a kind of reminder, it can be stated as the problem of a robot building a map of an unknown environment while simultaneously tracking its position using the partially built map. If we also want to estimate the full path of the robot (not only the last pose) jointly with the map, the problem takes the name of *full SLAM* [189] in robotics, although it can be also referred as *Bundle Adjustment* (BA) or *Structure from Motion* (SfM) in computer vision. Most of the approaches to SLAM have been proposed for two kind of sensory data: raw range scans [74, 120], and features either extracted from scans [47, 128] or from images, leading in this case the so-called visual SLAM (e.g. [38, 41, 114, 171, 175, 178].

Two different tendencies coexist in visual SLAM literature, namely *probabilistic filtering* and *bundle adjustment* or *smoothing* [185]. In short, *filtering* approaches represent beliefs about the system state with probability distributions, which are estimated over time through iterative implementations of the Bayes' rule that employs sequences of actions and observations. Filtering methods marginalize out past camera poses while keeping landmarks positions, and summarize the information obtained up to the current time-step with a *posterior* probability distribution. This distribution represents the inherent uncertainty present in visual SLAM solutions due to measurements errors. Probabilistic Bayesian filtering techniques (e.g. EKF, particle filters) have been extensively employed to cope with uncertainty in visual SLAM, and have been widely discussed elsewhere [127, 189]. In this thesis, section 2.3.1 summarized the most popular approaches belonging to this framework. Among them, particle filters (PFs) have proven to properly cope with complex, non-parametric and even multi-modal probability distributions over the system space, suitable for SLAM thanks to the introduction of Rao-Blackwellised Particle Filters (RBPFs) [136].

On the other hand, classical bundle adjustment techniques have been recently reconsidered to be applied to visual SLAM due to advances in sparse algebra software packages. Under the assumption of Gaussian distributed errors, BA represents the maximum likelihood estimator (MLE) for the problem of jointly estimating the set of camera poses and a collection of landmark positions. Although this methodology implies to optimize the full problem taking into account all the detected camera poses and landmarks up to the current time-step, the number of considered poses is generally reduced by only optimizing over a subset of the last poses or by selecting

some of them (the so-called *keyframes* (KF)) according to some heuristic rules. BA suffers from scalability issues due to its increasing computational cost that becomes prohibitive for large problems. Nevertheless, in recent years, improvements in sparse algebra methods and relative formulations for camera poses and landmarks positions have been successfully employed to overcome this limitation. A seminar paper that provides a thorough survey on the BA framework can be found in [194], while, in this work, section 2.3.2 briefly describes the most popular BA techniques.

4.1.1 Related work

A number of works in the technical literature have addressed robot localization and SLAM using vision sensors, including omni-directional, monocular, stereo, and trinocular cameras, under a wide range of different approaches included in the two families of solutions mentioned above.

Filtering methods have been widely employed in vision-based SLAM during the last decades. For example, in [124] an omni-directional camera is used to estimate the distance of the closest color transition in the environment, mimicking laser rangefinders performance. These measurements are introduced into a particle filter to determine the position of the robot within a previously constructed map. Tamini *et al.* [188] also present an omni-directional camera-based global localization approach for mobile robots using modified SIFT features that decrease the number of detected points and, therefore, the computation time of the localization process. The work in [43] presents a vision-based robot localization approach with just one camera which obtains a visual map of the ceiling and localizes the robot using a simple scalar brightness measurements as input. The robot localization within the map is carried out by a particle filter-based algorithm. In [207], an image retrieval system based on invariant features is combined with particle filter-based localization. These approaches only address global localization and do not deal with SLAM.

The SLAM problem is tackled in the series of papers [38, 40, 41] using a single camera (called MonoSLAM). The proposed method, which performs in real time, extracts a reduced but enough number of salient image features through the operator of Shi&Tomasi [173], which are identified by their associated image patches. The scale factor, which represents one of the main limitations of monocular SLAM, is resolved by initializing the system pointing at a pattern of known size. Other monocular SLAM approaches [29, 129] introduce the inverse depth parametrization for the undelayed initialization of features. Similarly, the work in [157] adds an Inertial Measurement Unit (IMU) to an implementation of the inverse depth-based monocular SLAM, reporting an improved accuracy in the estimation of the scale factor of the map. An RBPF-based method for performing monocular SLAM is reported in [114], which extracts SIFT features from the images and applies an Unscented Kalman Filter (UKF) within the robot localization algorithm both to sample new particles poses and to update the observed landmarks. Typically, monocular approaches to SLAM employ motion models which assume smooth paths for the camera by restricting its

velocities and accelerations. In addition, they suffer from ambiguity when estimating small displacements and rotations of the camera.

Stereo and trinocular systems elude the above-mentioned problems by exploiting the special characteristics of the epipolar geometry to directly extract 3D information from the detected features in the images. Hence, these camera configurations are widely extended in vision-based systems for robot localization and SLAM [36, 67, 171, 178, 179]. Thus, a trinocular camera is employed in [171] to address SLAM by tracking SIFT visual features [119] in unmodified environments. The egomotion estimation is computed from the robot odometry (as an initial estimation) and a least-squares procedure that finds the camera movement with the best alignment between the observed and the predicted image coordinates of the 3D landmarks in the map. The spatial uncertainty of the landmarks in the map is modeled by a Kalman filter. In [36] it is also proposed a trinocular SLAM system which uses 3D line segments as the elements of the map (instead of point features). They approximate the distribution of the robot pose with a particle filter and model the uncertainty in the 3D segments of the map with a Gaussian distribution which is updated over time with an EKF. An experiment in a simulated environment is presented to validate the results of this approach. Although it is an interesting variation of the traditional approaches, its application is limited to environments where straight lines can be easily found. Moreover, since these approaches employ the information provided by three images at each time-step, there exists an improvement in the robustness of the matching process, but, on the other hand, the computational burden of the method increases significantly, which becomes a significant problem in visual SLAM.

The works in [178] and [179] extract SIFT features from stereo images and compute their corresponding 3D points in space, which are taken as landmarks for a map built through a RBPF. In their approach, the weights of the particles are computed from the distance between the positions of the observed landmarks and the predicted positions (based on the particles pose) of their corresponding landmarks in the map. The matches are determined by computing the Euclidean distance between a 36-dimensional reduced version of the SIFT descriptors of the 3D landmarks. In that work, the motion model is based on an iterative Levenberg-Marquardt non-linear optimization algorithm which minimizes the re-projection error of the 3D coordinates of the landmarks on the images. Similarly, another RBPF approach which also employs SIFT features is presented in [67] as a vision-based solution to SLAM. In this case, the motion model relies on the robot odometry, while the observation model is derived from the Mahalanobis distance between the positions of the observed and mapped landmarks. Data association is determined from the Mahalanobis distance between the SIFT descriptors of the 3D landmarks assuming independence between the elements of the 128D SIFT vector.

The majority of these methods deal with the visual SLAM problem within a probabilistic point of view through *filtering* methods (mainly, EKF and PFs). Alternatively, a large collection of approaches can be found in technical literature that address visual SLAM within a *smoothing* framework. Different techniques have been proposed to reduce the computational burden of solving the full SLAM problem

[189, 194], which quickly increases as the size of the map grows. Most of them propose to discard some of the previous measurements so that full optimization from scratch is viable.

The general approach is to take a collection of heuristically chosen frames (coined *keyframes*) over which optimization is performed, discarding the information gathered in the rest of frames (e.g. [15, 115, 176]). Nevertheless, several other techniques have been applied to further alleviate the computational burden of bundle adjustment methodologies, most of them in combination with the presence of keyframes.

One of them is to perform some kind of marginalization to reduce the size of the involved matrices in the optimization procedure. Thus, the work in [177] presents a sliding window filter based on a delayed state marginalization. Its operation ranges from EKF to full SLAM according to the window size, while performing both landmarks and camera pose marginalization to achieve constant time operation, demonstrating similar convergence properties to the full batch solution and outperforming those from standard visual odometry. Similarly, [104] builds a *skeleton* from the camera poses that represents a reduced system that approximates the full problem, leading to feasible solutions when dealing with large loop closures. This skeleton is formed by marginalizing features over camera poses and subsequently further reducing the latter.

Submapping is another popular methodology to deal with bundle adjustment. The basic idea is to divide the full map into smaller local sub-maps that can be more efficiently optimized. The work in [115] proposes a hybrid metrical-topological representation of the map that contributes to scalability for the bundle adjustment problem. The proposal benefits from the topological map properties to allow for instant loop closures while metric locally consistent maps are maintained by embedding neighbor keyframes and landmarks into a single Euclidean space and optimizing over the submap.

Another interesting approach to the BA-based SLAM problem is the so-called incremental Smoothing And Mapping (iSAM), developed in [98] where a factored representation of the Hessian matrix is exploited to provide easy access to the marginal covariances needed for data association. This matrix is incrementally updated and maintained sparse by reordering the involved variables.

Relative formulations have become a feasible alternative to perform bundle adjustment since they provide a bounded-cost performance that can lead to on-line, real-time operation [53, 54, 150, 175, 176]. These techniques represent camera poses and landmark positions relative to some local reference systems, hence allowing for local optimizations instead of solving the full BA problem, although at the cost of denser matrices. Typically, each camera pose is referred to a coordinate system centered at the previous one, whilst landmarks are referred to the coordinate system of the camera pose from where it was observed for the first time. A blended approach between global and relative formulations was recently proposed in [15], which will be further explained and exploited in section 4.3.

Regarding the method employed to carry out optimization, the non-linear so called Levenberg-Marquardt technique is the most widely employed although other alternatives such as stochastic gradient descent have been successfully proposed in [75,150].

Finally, a thorough comparison between filtering and smoothing families of solutions is presented in [184, 185], ultimately advocating for the application of BA techniques, as they have proven to outperform filtering alternatives in terms of accuracy per unit of time.

4.1.2 Contribution

This chapter presents two solutions for the stereovisual SLAM problem, one of them based on probabilistic filtering and another one relying on bundle adjustment.

Our first proposal, described in section 4.2 addresses the problem of SLAM for stereo vision systems within a filtering framework. Our contribution here consists of providing appropriate probabilistic models for the motion and observations of a stereo camera, both of them suitable for particle filters methods. In addition, the same models can also be applied to global localization. Our method takes a sequence of stereo images as the only input and does neither rely on any other sensory data (odometers, IMU, etc) nor assume a priori knowledge about the camera movement. More concretely, the novelties regarding to the motion and observation models are:

- The robot ego-motion estimation is addressed by performing 6D visual odometry through a reliable 3D landmark registration method which models the uncertainty in the pose increment estimation as a Gaussian. This motion model was presented in section 3.2 and, in summary, provides three main advantages in relation to other proposals: i) it is applicable to either, wheeled and not-wheeled robots that navigate on any type of surface (even flying robots), ii) it is efficient, and iii) it overcomes the divergence and local-minima issues that iterative approaches to visual odometry may entail, as well as the need of an initial estimation.

- The observation model avoids explicit data association by applying marginalization over all the possible associations, thus discarding the possibility of potential incorrect correspondences between the observed landmarks and the map.

This proposal has been validated by experiments with a real robot that has been driven within an office-like scenario where it builds a map of the environment and tracks its position simultaneously. To test the quality of the constructed map, that map is provided to the robot in another experiment where it performs global localization.

On the other hand, our second approach applies a bundle adjustment technique to solve stereovisual SLAM. The proposed system implements the Sparser Relative Bundle Adjustment (SRBA) approach presented in [15] as the core of a visual SLAM system back-end. SRBA is a variant of the relative bundle adjustment method which,

unlike global BA methodologies (which represent camera poses and landmark positions referred to a single, global coordinate system) and standard relative formulations (which set camera poses referred to their preceding one), proposes a hybrid methodology that incrementally builds local submaps where each keyframe is referred to its submap *origin* keyframe. In turn, all the submap origins are *linearly* connected between them, i.e., each one is referred to its preceding one. This leads to sparser Hessian and Jacobian matrices than standard relative approaches while keeping their advantages over global formulations.

Our contribution in this second proposal is the development of a complete front-end that complements the above-mentioned SRBA-based optimization stage. The presented front-end combines the extraction of ORB keypoints and binary descriptors with a robust visual odometry approach based on a pseudo-Huber cost function. Keyframe creation is decided by a set of heuristic rules based on keypoints tracking, change in pose between keyframes and, ultimately, image similarities. On the other hand, data association is addressed through a bag-of-words approach relying on the ORB binary descriptors that incrementally builds an image database for later retrieval of the most similar one to the current stereo frame.

This system has been validated with both synthetic and real data in indoor and outdoor environments, demonstrating its capabilities to efficiently deal with the stereovisual SLAM problem.

4.2 A Particle Filter-based Approach to Visual SLAM

A visual SLAM solution is presented in this section based on a Rao-Blackwellized particle filter (refer to section 2.3.1 for a brief introduction on particle filters). It is important to note that this approach shares many of the techniques developed to perform visual odometry through the closed-form formulation explained in section 3.2. In fact, that method is employed as a part of the visual SLAM system proposed here. Therefore, in order to keep this chapter more self-contained, the most important steps regarding those methods will be recalled but not exhaustively explained.

Formally, let $\mathbf{x}_t$, $\mathbf{u}_t$ and $\mathbf{z}_t$ be the robot pose, the action, and the observation at time-step t, respectively, and let $\mathbf{m}$ be the map of the environment. As we mentioned in the background chapter, the aim of the full SLAM problem addressed under a probabilistic filtering framework is to estimate the joint distribution of both the robot path and the map [189], i.e. to compute $p(\mathbf{x}_{1:t},\mathbf{m}|\mathbf{z}_{1:t},\mathbf{u}_{1:t})$ with $\mathbf{z}_{1:t} = \{\mathbf{z}_1,\ldots,\mathbf{z}_t\}$, $\mathbf{u}_{1:t} = \{\mathbf{u}_1,\ldots,\mathbf{u}_t\}$ and $\mathbf{x}_{1:t} = \{\mathbf{x}_1,\ldots,\mathbf{x}_t\}$. In this work, the action $\mathbf{u}_t$ represents the robot pose change between time-steps $t-1$ and t, which, in our case, is unknown and will be estimated by means of visual odometry.

Due to the high dimensionality of the system state in SLAM, we employ a Rao-Blackwellized particle filter which reduces the complexity of the estimation problem by sampling only over a subset of the state variables. By factoring $p(\mathbf{x}_{1:t},\mathbf{m}|\mathbf{z}_{1:t},\mathbf{u}_{1:t})$ we can sample the distribution of the possible robot paths and analytically compute the map distribution from those samples [48, 189]:

$$p(\mathbf{x}_{1:t},\mathbf{m}|\mathbf{z}_{1:t},\mathbf{u}_{1:t}) = \underbrace{p(\mathbf{x}_{1:t}|\mathbf{z}_{1:t},\mathbf{u}_{1:t})}_{\text{robot path}}\underbrace{p(\mathbf{m}|\mathbf{x}_{1:t},\mathbf{z}_{1:t},\mathbf{u}_{1:t})}_{\text{map}} \tag{4.1}$$

According to this approach, there is a map distribution associated to each sample of the robot path.

Notice that $\mathbf{u}_{1:t}$ can be eliminated from the second term in equation (4.1) since the robot path $\mathbf{x}_{1:t}$ d-separates the map $\mathbf{m}$ and the actions, hence they become conditionally independent (please, refer to [166] for further technical details). Moreover, the term regarding the map in (4.1) can be further factored due to the conditional independence between the landmarks in the map, given a robot path hypothesis:

$$p(\mathbf{m}|\mathbf{x}_{1:t},\mathbf{z}_{1:t}) = \prod_{j=1}^{M} p\left(\mathbf{m}^j|\mathbf{x}_{1:t},\mathbf{z}_{1:t}\right) \tag{4.2}$$

The above expressions state that the joint probability density of the robot path and the map, given the set of measurements, can be computed using one estimator for the robot path and M estimators for the landmarks in the map for each of the P particles. In this work, we use a particle filter to estimate $p(\mathbf{x}_{1:t}|\mathbf{z}_{1:t})$, and a Kalman filter (KF) to update the positions of the landmarks at each time-step.

Regarding the robot path estimation, it is updated at each time-step by appending the latest robot pose, which is computed from:

$$\underbrace{p(\mathbf{x}_t|\mathbf{z}_{1:t},\mathbf{u}_{1:t})}_{\substack{\text{pose estimation}\\ \text{at time } t}} \propto \underbrace{p(\mathbf{z}_t|\mathbf{x}_t)}_{\text{observation model}} \cdot \overbrace{\int \underbrace{p(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{u}_t)}_{\text{transition model}} \underbrace{p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1},\mathbf{u}_{1:t-1})}_{\substack{\text{pose estimation}\\ \text{at time } t-1}} d\mathbf{x}_{t-1}}^{\text{pose } \textit{prior} \text{ estimation}} \quad (4.3)$$

where $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ is approximated by a set of particles, each of them representing a possible robot pose.

In short, the Rao-Blackwellized Particle Filter that estimates $p(\mathbf{x}_t,\mathbf{m}|\mathbf{z}_{1:t},\mathbf{u}_{1:t})$ evolves as follows:

1. The robot path particles are propagated according to the transition model which, in our case, is the mobile robot *motion model*. In this work we estimate the motion between consecutive time-steps through the closed-form visual odometry algorithm explained in detail in section 3.2 and summarized again in section 4.2.2.

2. These particles are subsequently weighted according to the observation model which estimates the likelihood of obtaining the current observation from the pose hypothesis hold by each particle. The observation model, based on 3D landmarks augmented with SIFT descriptors, will be exposed in section 4.2.3.

3. Next, a resampling stage is performed (if necessary) over the particles. The probability of surviving for each particle is proportional to its importance weight, as explained in section 2.3.1.

4. Finally, the map associated to each particle is updated through the Kalman Filter equations.

4.2.1 Map and observations

This section presents the process for obtaining 3D visual landmarks from the environment, which will be the elements of both the observations and the map. The notation followed throughout this section is also introduced.

Notation and definitions

In this work, an observation $\mathbf{z}_t$ and the map $\mathbf{m}$ are defined as collections of 3D landmarks:

$$\begin{array}{rclcrcl} \mathbf{z}_t & = & \left\{\mathbf{z}_t^i\right\}_{i=\{1,\ldots,N\}} & \text{where} & \mathbf{z}_t^i & = & \left\langle \mathbf{X}_t^i, \mathbf{F}_t^i \right\rangle \\ \mathbf{m} & = & \left\{\mathbf{m}^j\right\}_{j=\{1,\ldots,M\}} & \text{where} & \mathbf{m}^j & = & \left\langle \mathbf{X}_m^j, \mathbf{F}_m^j \right\rangle \end{array} . \quad (4.4)$$

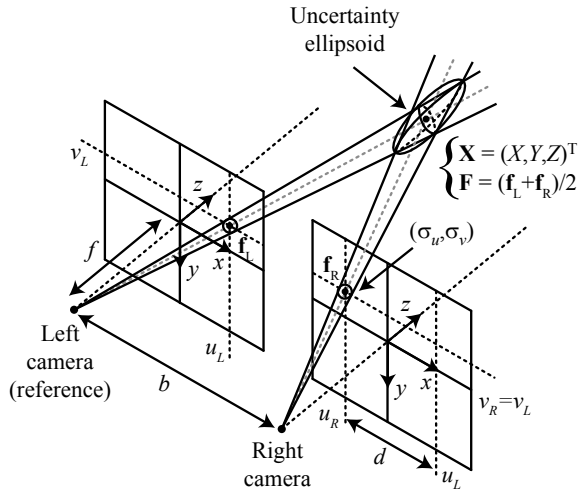


Figure 4.1: Stereo system configuration for the visual SLAM method. The left and right SIFT descriptors are labeled $\mathbf{f}_L$ and $\mathbf{f}_R$ while the landmark associated descriptor is denoted by $\mathbf{F}$.

Each landmark, either of an observation or in the map, comprises a 3D location $\mathbf{X}$, and an associated SIFT descriptor $\mathbf{F}$ [119]. This definition extends the one presented in section 3.2.2 with the presence of a SIFT descriptor. For clarity, we reproduce in figure 4.1 the same configuration shown in figure 3.4 but augmented with the addition of the SIFT descriptor notation.

The uncertainty in the 3D positions of the landmarks is modeled by normal distributions with mean $\boldsymbol{\mu}$ and a 3×3 covariance matrix $\boldsymbol{\Sigma}$:

$$\mathbf{X}_t^i \sim \mathcal{N}\left(\boldsymbol{\mu}_t^i, \boldsymbol{\Sigma}_t^i\right) \qquad \mathbf{X}_m^j \sim \mathcal{N}\left(\boldsymbol{\mu}_m^j, \boldsymbol{\Sigma}_m^j\right). \tag{4.5}$$

The SIFT descriptor $\mathbf{F}$ of each landmark is also assumed to be normally distributed with a diagonal covariance matrix containing a constant value for each dimension, say $\left(\sigma^2_{\mathrm{S1}},\ldots,\sigma^2_{\mathrm{S128}}\right)$.

$$\mathbf{F}_t^i \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{F}_t^i}, \boldsymbol{\Sigma}_{\mathbf{F}_t^i}\right) \qquad \mathbf{F}_m^j \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{F}_m^j}, \boldsymbol{\Sigma}_{\mathbf{F}_m^j}\right). \tag{4.6}$$

Summarizing, we define a generic 3D landmark l^i as a normally distributed random variable with the following statistics:

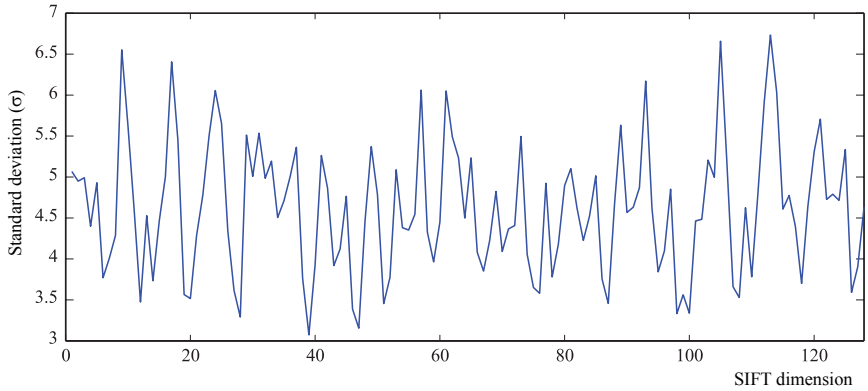


Figure 4.2: Estimation of the standard deviation for each dimension of the SIFT descriptor.

$$
\begin{aligned}
l^i &\sim \mathcal{N}\left(\left\langle \boldsymbol{\mu}^i_{\mathbf{X}}, \boldsymbol{\mu}^i_{\mathbf{F}} \right\rangle, \left(\begin{array}{c|c} \boldsymbol{\Sigma}^i_{\mathbf{X}} & \mathbf{0} \\ \hline \mathbf{0}^{\mathsf{T}} & \boldsymbol{\Sigma}^i_{\mathbf{F}} \end{array}\right)\right) \qquad (4.7)\\
&= \mathcal{N}\left(\left\langle \begin{pmatrix} X^i \\ Y^i \\ Z^i \end{pmatrix}, \begin{pmatrix} d^i_1 \\ \vdots \\ d^i_{128} \end{pmatrix} \right\rangle, \left(\begin{array}{ccc|ccc} \sigma^2_X & \sigma_{XY} & \sigma_{XZ} & & & \\ \sigma_{YX} & \sigma^2_Y & \sigma_{YZ} & & \mathbf{0} & \\ \sigma_{ZX} & \sigma_{ZY} & \sigma^2_Z & & & \\ \hline & & & \sigma^2_{\mathrm{S1}} & \cdots & 0 \\ & \mathbf{0}^{\mathsf{T}} & & \vdots & \ddots & \vdots \\ & & & 0 & \cdots & \sigma^2_{\mathrm{S128}} \end{array}\right)\right)
\end{aligned}
$$

where $\mathbf{0}$ is a 3×128 null matrix, $\boldsymbol{\mu}^i_{\mathbf{X}}$ stands for the mean of the 3D landmark position, $\boldsymbol{\mu}^i_{\mathbf{F}}$ denotes the mean of the 128D SIFT descriptor, and $\boldsymbol{\Sigma}^i_{\mathbf{X}}$ and $\boldsymbol{\Sigma}^i_{\mathbf{F}}$ are their associated covariance matrices, respectively. The parameters $\left(\sigma^2_{\mathrm{S1}},\ldots,\sigma^2_{\mathrm{S128}}\right)$ stand for the variance of the components of the SIFT descriptor, and model the uncertainty when computing the descriptor of the same feature from different points of view. Their values have been determined empirically from an independent experiment that tracks a set of 576 features in a sequence of 50 images, recorded in an office-like environment, while computing their SIFT descriptors at each time-step. In this experiment, a feature is visible and tracked over 26 images on average. The standard deviations of the descriptor dimensions $(\sigma_{\mathrm{S1}},\ldots,\sigma_{\mathrm{S128}})$ are shown in figure 4.2. Notice that different environments and/or camera movements would require a new characterization of the descriptor uncertainty as the SIFT descriptor may vary differently among points of view.

Once the landmark notation has been introduced, next we address the process of obtaining those landmarks from the stereo images.

Extraction of reliable observation landmarks

To obtain a set of 3D landmarks from a pair of stereo images we need to find feature points in both images, to match them, and to estimate their corresponding 3D locations.

In this work we follow the procedure described in section 3.2.2 to detect and match keypoints in the stereo images. As a reminder, the Shi & Tomasi feature detector is applied to extract keypoints which are subsequently augmented with a 128D SIFT descriptor. Since the Shi and Tomasi detector does not provide any information about the best scale for the detected points, the SIFT descriptor computation is accomplished in one scale only (i.e. the original image) losing, therefore, the scale invariance. However, the resulting SIFT descriptor has been proved to be distinguishable enough for performing stereo matching and for measuring the similarity between the projected landmarks when detected from different points of view during indoor navigation. Although a shorter key vector may be employed for stereo matching with proper results, in [119] it is suggested the use of a 128D one to achieve the best matching performance, which is specially interesting to perform data association that leads to robust detection of loop closures in visual SLAM.

Subsequently, the extracted keypoints are matched according to the Euclidean distance between their SIFT descriptors while fulfilling the restrictions imposed by the epipolar geometry. Finally, each 3D landmark is assigned a 128D SIFT descriptor which is simply computed as the average value of the descriptors ($\mathbf{f}_L$ and $\mathbf{f}_R$) from each image: $\mathbf{F} = (\mathbf{f}_L + \mathbf{f}_R)/2$.

The so-computed 3D landmarks are the elements of both the observations and the map, and constitute the basis of the probabilistic SLAM method proposed in this work.

Map initialization and update

The management of the map built during the SLAM process entails the insertion, update and deletion of landmarks. In short, this management can be summarized by this sequence of steps, which will be further explained next:

1. All the observed landmarks in the first observation are introduced into the map.
2. Then, at each time-step, data association is performed to obtain a set of correspondences between the observed landmarks and those in the map.
   (a) The positions of the landmarks in the map with a correspondence within the observation are updated.
   (b) The landmarks in the map without a correspondence are not modified.
   (c) The observed landmarks with no correspondences are considered new landmarks and hence are introduced into the map.

3. Landmarks that are not observed a significant number of times while still being within the camera field of view are deleted from the map, as they are considered *non-stable*.

Every landmark in the map has two associated attributes indicating the number of times and the last time-step it has been observed, respectively.

At the beginning of the SLAM process, all the landmarks which are detected in the first observation are introduced into an initially empty map, and their associated counters are initialized accordingly. Then, as new observations are gathered, the detected landmarks at each time-step are compared with those in the map in order to obtain a set of matches. To that purpose, their probability of being in correspondence is evaluated from the distance between both their 3D positions and their SIFT descriptors, whereas the matching decision is taken according to a certain threshold. This correspondence measure is computed from the same expression which is proposed for our observation model (to be described in section 4.2.3).

Once the correspondences have been established, the 3D positions of the landmarks with positive matches are updated through the Kalman Filter equations [99], which, according to our linear observation model, become:

$$\begin{aligned} \boldsymbol{\mu}_m &= \boldsymbol{\Sigma}_m \left(\boldsymbol{\Sigma}_{\tilde{m}}^{-1} \boldsymbol{\mu}_{\tilde{m}} + \boldsymbol{\Sigma}_t^{-1} \boldsymbol{\mu}_t \right) \\ \boldsymbol{\Sigma}_m &= \left(\boldsymbol{\Sigma}_{\tilde{m}}^{-1} + \boldsymbol{\Sigma}_t^{-1} \right)^{-1}, \end{aligned} \tag{4.8}$$

where $(\boldsymbol{\mu}_{\tilde{m}}, \boldsymbol{\Sigma}_{\tilde{m}})$ and $(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$ represent the distributions of the landmark position before and after the update process, respectively, while $(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ stands for the observed position of the landmark at time-step t. Please, refer to appendix B for a complete derivation of these expressions from the Kalman Filter equations.

Finally, we update the counters of all the landmarks in the map, and delete non-stable ones.

4.2.2 Motion model: Visual odometry

Typically, the motion model in localization and SLAM for mobile robots is given by a probabilistic characterization of the robot displacement obtained from encoder-based odometry. However, this thesis focuses on the solutions to SLAM through the use of cameras as the only sensor. Therefore, in this first approach to visual SLAM, we define a motion model which does not rely on the robot odometry but is based on matching 3D landmarks between two consecutive robot poses, hence denoted visual odometry. This motion model is not restricted to planar robot motion since it estimates the incremental change in 6D: x, y, z, *yaw*, *pitch*, and *roll*. The employed algorithm, extensively described in section 3.2, takes as inputs two sets of 3D points computed at different time-steps, with known correspondences and coordinates relative to each robot pose, and estimates their relative change through a closed-form solution derived in [86]. Since this visual odometry technique has been already explained, we just briefly depict its main stages.

In the first pair of stereo images, two sets of features are extracted and matched according to both their SIFT descriptors and the epipolar constraint (as explained in the previous section). The matched pairs are subsequently projected into space and their 3D spatial uncertainty is also computed.

Then, the features are tracked in the next pair of stereo images (using the KLT or Shi & Tomasi method), which produces a new set of matched image features at this time-step. Notice that, since the correspondences between the tracked features in the left and right images are already known, it is not necessary to match each other again. This speeds up the process significantly and reduces the computational burden of the whole visual odometry procedure. Finally, the new set of matched points is also projected into 3D space.

We must remark that, due to the tracking process, the associations between the two sets of 3D landmarks are also known at each time-step. This set of 3D landmark pairs is taken as input for the closed-form solution which computes the 6D robot pose increment and its associated uncertainty, as fully explained in section 3.2.

4.2.3 An observation model for stereo vision

In the following we introduce our proposal for a probabilistic observation model $p(\mathbf{z}_t|\mathbf{x}_t)$, which stands for the likelihood of an observation at time t, given the robot pose $\mathbf{x}_t$. Notice that, as each particle $\mathbf{x}_t^{[k]}$ represents a hypothesis of the robot pose (k represents the index of the particle), this likelihood will be evaluated at each particle in the filter. In the following formulation, however, for clarity, we will omit the particle indexes.

First, assuming conditional independence between the errors in the detection of the individual landmarks $\mathbf{z}_t^i$, the observation likelihood function can be factorized as follows:

$$p(\mathbf{z}_t|\mathbf{x}_t) \overset{\text{cond.ind}}{=} \prod_i p\left(\mathbf{z}_t^i|\mathbf{x}_t\right). \tag{4.9}$$

To avoid explicit data association between landmarks in the observation and in the map, we apply next the law of total probability to marginalize out the observation likelihood of individual landmarks by considering all the possible associations:

$$p\left(\mathbf{z}_t^i|\mathbf{x}_t\right) = \sum_{\psi=\{1,\ldots,M,\varnothing\}} p\left(\mathbf{z}_t^i|\mathbf{x}_t, c_i=\psi\right) \underbrace{P(c_i=\psi|\mathbf{x}_t)}_{\eta}, \tag{4.10}$$

where c_i is an unknown discrete variable that represents the correspondence of the i-th observed landmark. Its possible values are $1,\ldots,M$ for map landmarks, or $\varnothing$ for no correspondence with the map. Notice that the *a priori* probability of any given correspondence $P(c_i=\psi|\mathbf{x}_t)$ is a constant since it does not depend on the actual observation $\mathbf{z}_t^i$. If we do not have any other information, we can assume the same probability for all the possible correspondences, including the null one:

$$p\left(\mathbf{z}_t^i|\mathbf{x}_t\right) = \eta \sum_{\psi=\{1,\ldots,M,\varnothing\}} p\left(\mathbf{z}_t^i|\mathbf{x}_t, c_i=\psi\right). \tag{4.11}$$

The term $p\left(\mathbf{z}_t^i|\mathbf{x}_t, c_i = \psi\right)$ can be seen as the probability of the observed landmark $\mathbf{z}_t^i$ and its corresponding landmark $\mathbf{m}_\psi$ to coincide in both the 3D space of the position and the 128-dimensional space of the SIFT descriptors. This can be computed by simply evaluating at the origin a Gaussian distribution whose mean $\boldsymbol{\mu}$ is the difference between the means of $\mathbf{z}_t^i$ and $\mathbf{m}_\psi$ and the covariance $\boldsymbol{\Sigma}$ is the sum of their covariance matrices:

$$\begin{aligned} p\left(\mathbf{z}_t^i|\mathbf{x}_t, c_i = \psi\right) &= \mathcal{N}\left(0; \underbrace{\bar{\mathbf{z}}_t^i - \bar{\mathbf{m}}_\psi}_{\boldsymbol{\mu}}, \underbrace{\boldsymbol{\Sigma}_{\mathbf{z}_t^i} + \boldsymbol{\Sigma}_{\mathbf{m}_\psi}}_{\boldsymbol{\Sigma}}\right) \\ &= \eta' \exp\left\{-\frac{1}{2}\left(\bar{z}_t^i - \bar{m}_\psi\right)^{\mathrm{T}}\left(\Sigma_{z_t^i} + \Sigma_{m_\psi}\right)^{-1}\left(\bar{z}_t^i - \bar{m}_\psi\right)\right\} \\ &= \eta' \exp\left\{-\frac{1}{2}\|\bar{\mathbf{z}}_t^i - \bar{\mathbf{m}}_\psi\|^2_{\boldsymbol{\Sigma}_{\mathbf{z}_t^i} + \boldsymbol{\Sigma}_{\mathbf{m}_\psi}}\right\}, \end{aligned} \tag{4.12}$$

where

$$\eta' = \left(2\pi\left|\boldsymbol{\Sigma}_{\mathbf{z}_t^i} + \boldsymbol{\Sigma}_{\mathbf{m}_\psi}\right|\right)^{-\frac{1}{2}}. \tag{4.13}$$

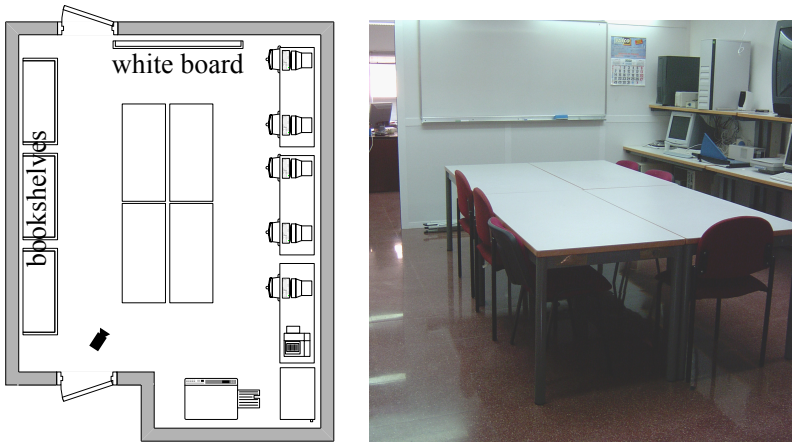
Due to the particular structure of the mean $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$ (similar to those shown in equation (4.7)), the exponential term in equation (4.12) can be split in two factors related to the position and descriptor dimensions of the random variable, respectively:

$$\begin{aligned} \mathcal{N}(0; \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \eta' \exp\left\{-\frac{1}{2}\left(\begin{array}{c|c} \boldsymbol{\mu}_{\mathbf{X}}^{\mathrm{T}} & \boldsymbol{\mu}_{\mathbf{F}}^{\mathrm{T}} \end{array}\right)\left(\begin{array}{c|c} \boldsymbol{\Sigma}_{\mathbf{X}} & \mathbf{0} \\ \hline \mathbf{0}^{\mathrm{T}} & \boldsymbol{\Sigma}_{\mathbf{F}} \end{array}\right)^{-1}\left(\begin{array}{c} \boldsymbol{\mu}_{\mathbf{X}} \\ \hline \boldsymbol{\mu}_{\mathbf{F}} \end{array}\right)\right\} \\ &= \eta' \exp\left\{-\frac{1}{2}\left(\boldsymbol{\mu}_{\mathbf{X}}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{X}}^{-1}\boldsymbol{\mu}_{\mathbf{X}} + \boldsymbol{\mu}_{\mathbf{F}}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{F}}^{-1}\boldsymbol{\mu}_{\mathbf{F}}\right)\right\} \\ &= \eta' \exp\left\{-\frac{1}{2}\boldsymbol{\mu}_{\mathbf{X}}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{X}}^{-1}\boldsymbol{\mu}_{\mathbf{X}}\right\} \exp\left\{-\frac{1}{2}\boldsymbol{\mu}_{\mathbf{F}}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{F}}^{-1}\boldsymbol{\mu}_{\mathbf{F}}\right\} \qquad (4.14) \\ &= \underbrace{\eta' \exp\left\{-\frac{1}{2}\|\boldsymbol{\mu}_{\mathbf{X}}\|^2_{\boldsymbol{\Sigma}_{\mathbf{X}}}\right\}}_{\text{position}} \underbrace{\exp\left\{-\frac{1}{2}\|\boldsymbol{\mu}_{\mathbf{F}}\|^2_{\boldsymbol{\Sigma}_{\mathbf{F}}}\right\}}_{\text{descriptor}} \qquad (4.15) \end{aligned}$$

Furthermore, since $\boldsymbol{\Sigma}_{\mathbf{F}}$ is a diagonal matrix containing constant values (σ_{S}^2), the exponent of the descriptor term in equation (4.14) is greatly simplified.

4.2.4 Experimental results

The proposed method for performing visual SLAM within a particle filter framework has been tested in two experiments involving different kinds of camera movements:



(a) (b)

Figure 4.3: Plan and snapshot of the office-like experiment scenario.

1. In the first experiment, Sancho, one of our mobile robots equipped with a BumbleBee[1] stereo vision system, was manually driven following an almost circular trajectory in an office environment while gathering stereo images.

2. The second experiment shows the performance of our approach when coping with data from a hand-held camera describing a totally unconstrained trajectory.

In both experiments, the camera ego-motion was computed using the visual odometry process described in section 4.2.2 and the estimations were stored in a log file in order to be subsequently used as the motion model of the SLAM process, which does not operate in real time.

Office-like environment experiment

Figure 4.3 shows a plan and a snapshot of the environment where this first experiment has been carried out. This environment offers some interesting features for testing the robustness of the proposed visual SLAM approach. Thus, for example, the white board shown at the top of the image 4.3a is a place where the detection of visual landmarks is very unlikely. On the other hand, there are some bookshelves at the left side of the room which produces many reliable landmarks. Finally, the bottom-right zone of the room is not well illuminated, which hampers the extraction of landmarks. The results achieved by our SLAM method in all these different situations are shown

[1] http://www.ptgrey.com

Table 4.1: Camera configuration and experimental setup

Camera Configuration *PointGrey BumbleBee*	
Baseline (b)	11.9 cm
Focal length (f)	507.808 px
Principal Point coordinates (c_u,c_v)	(252.922, 356.237) px
Image size	640×480 px
Experimental Setup	
Path length (approx.)	40 m
Total number of stereo images	1000
Image capture rate	3 Hz
$\left(\sigma_u^2,\sigma_v^2,\sigma_d^2\right)$	(1,1,2) px^2
Number of particles	80
Number of landmarks in the final map	927

in the next section, while a video showing the complete evolution of this experiment can be watched online[2].

In the feature extraction process, it has been assumed that errors in the variables u and v (i.e. the column and the row in the image, respectively, of the detected interest points) have a variance of 1 squared pixel (px^2), while the errors in d (i.e. the disparity) is considered to be 2 px^2. This value arises from the assumption of independence between the errors in the estimation of the column variables for both images; thereby the variance of the disparity ($d = u_L - u_R$) becomes the sum of the variances of both u variables.

Others parameters of the camera configuration and for the experiment setup are summarized in Table 4.1.

Map Building

In this experiment, the recorded sequence of stereo images is used to build a map of the environment through the Rao-Blackwellized Particle Filter (RBPF) described in section 4.2.

The evolution of the constructed map as the robot moves is shown in figure 4.4, where a top view of the 3D landmarks of the map being built (shown as ellipses representing 95% confidence intervals) is displayed at six different time-steps. In addition, the hypothesis of the robot path estimated by each particle is also shown in the figure. The uncertainty in both the robot pose and the map depends on the number of observed landmarks at each time-step and the number of times they are detected.

[2]http://www.youtube.com/watch?v=m3L8OfbTXH0

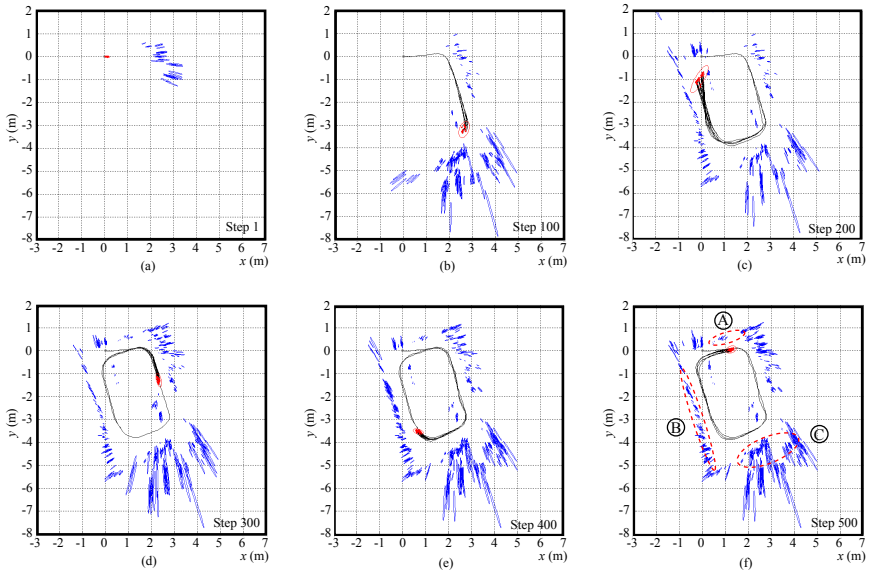


Figure 4.4: Map building with a Rao-Blackwellized Particle Filter. Map representation at different time-steps where a 95% Gaussian confidence interval of the landmark positions is represented by blue ellipses. The red ellipse surrounding the particle set represents the uncertainty in the robot pose corresponding with a 95% Gaussian confidence interval.

Thereby, the uncertainty in the landmark positions decreases as they are detected in successive observations.

The zones A, B and C indicated in figure 4.4f correspond to the interest zones which where mentioned in the previous section: the white board, the bookshelves and the poorly illuminated zone, respectively. As can be seen, the group of observed landmarks in each situation are sensibly different. Thus, in zone A, the number of landmarks is quite low, as it was expected from the visual characteristics of the environment. In zone B there is a high number of observed landmarks with low uncertainty, which manifests the adequate visual characteristics of that zone. Finally, in zone C, because of the poor illumination, the landmarks are observed only a few number of times and, therefore, the initial uncertainty of their position does not reduce much.

Regarding the uncertainty of the robot pose estimation, directly related to the size of the red covariance ellipse in figure 4.4, it grows during the first part of the experiment since the vast majority of the observed landmarks are new. This situation can be appreciated in the first part of the plot in figure 4.5a, which shows the evolution of the determinant of the particles' covariance matrix. The substantial increase around iteration 225 is due to the combination of two adverse factors: (i) the robot is performing a turn (which entails significant errors in the visual odometry estimations)

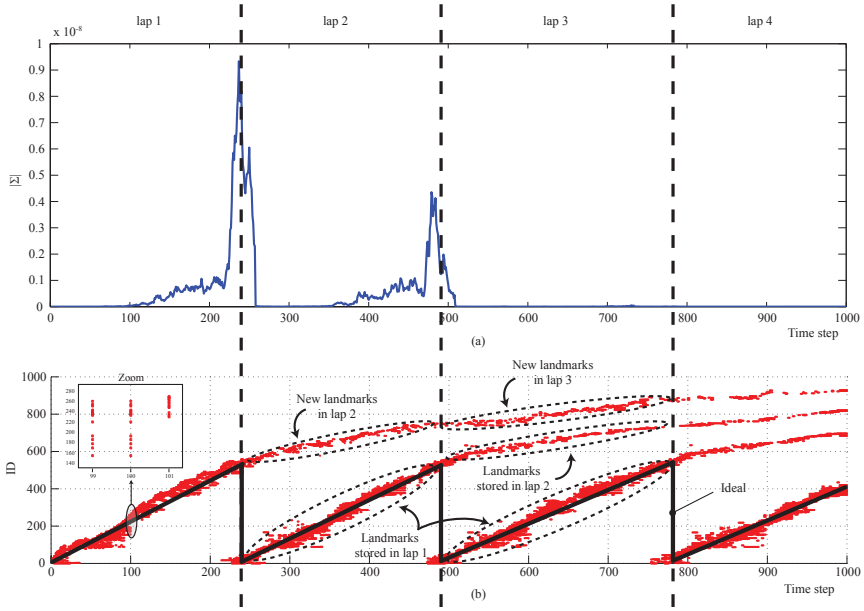


Figure 4.5: (a) Determinant of the particles' covariance matrix through time and (b) the landmarks IDs at each time-step

and (ii) the scarce visual information available in that zone since the camera sees an almost totally texture-less surface (the above-mentioned white board).

However, notice that this increasing trend changes drastically around iteration 240 (at some point between figures 4.4c and 4.4d) since the robot *closes the loop*, reaching an already navigated position. Now, most of the observed landmarks correspond to those previously stored in the map, hence the estimation of the robot position improves (the particles converge towards the real robot location) and, therefore, the determinant of the covariance matrix reduces significantly. This *loop closure* can be also appreciated in figure 4.5b where it is represented the IDs of the observed landmarks with correspondences in the map for each time-step. In this work, the management of the landmark ID is accomplished as follows:

- As the robot explores the environment, a unique ID is assigned (in an increasing order) to each new observed landmark.
- A landmark which was observed previously keeps its original ID while its position and covariance are updated with the new observation.

Thus, the increasing zone at the beginning of the figure 4.5b (up to time-step 225 approx.) illustrates the initial situation where the robot navigates an area for the first

time while observing both new landmarks (the majority) and landmarks which were recently observed and stored in the map.

At loop closure (approximately at time-step 240), the robot observes a large number of already-stored landmarks while the number of new landmarks reduces significantly. The already observed landmarks correspond to those observed at the beginning of the experiment (with the lowest IDs). This manifests in figure 4.5b as two different groups of IDs: one for the new landmarks and other for those already stored in the map. This situation repeats every time that the robot reaches the initial position of the path.

Please note that, in an ideal situation, with the robot observing all the possible landmarks in the environment and establishing correct correspondences with the map at each time-step, the representation of the IDs in this experiment should be sawtooth-shaped (the solid black line in the figure). In real situations, although the robot moves through an already explored zone, it observes new landmarks which are assigned new IDs, giving the particular shape of figure 4.5b.

Finally, it must be remarked that there exists another interval of iterations (between time-steps 375 and 500) where the determinant of the particles' covariance matrix grows (see figure 4.5a). The initial increasing zone is caused by the high uncertainty of the landmarks in that zone of the map (zone B in figure 4.4f) while the peak at time-step 475 comes again from the combination of a turn and the presence of the texture-less white board (zone A in figure 4.4f). The magnitude of this increase is sensibly lower than that in the first lap of the experiment since some correspondences between the observed landmarks and those of the map have been established, hence reducing the landmark uncertainty and, therefore, improving the robot pose estimation. Following this trend, notice that the influence of this situation is practically negligible in the third and the fourth laps.

Visual SLAM Performance

In order to evaluate the performance of our visual SLAM approach, we have compared the robot path estimates from both a RBPF algorithm based on laser data, gathered with a SICK LMS-200 laser scanner, and our visual SLAM approach (see figure 4.6a). In this work, we have considered the former as the *ground truth* of the robot path since laser data is highly accurate. We must remark that both laser data and stereo images have been gathered simultaneously during the navigation in order to ensure that the path followed by the robot is exactly the same for both types of data.

Figure 4.6b shows the root-square-error committed by the proposed method when estimating the robot path, yielding a Root-Mean-Square-Error (RMSE) of 19.2 cm with a standard deviation of 7.4 cm.

Regarding the computational time, this experiment was carried out on a Desktop PC Intel Pentium 4 at 2.60 GHz running under Windows XP SP2 with 2 GB of RAM memory. The processing time of each iteration of the particle filter is represented in

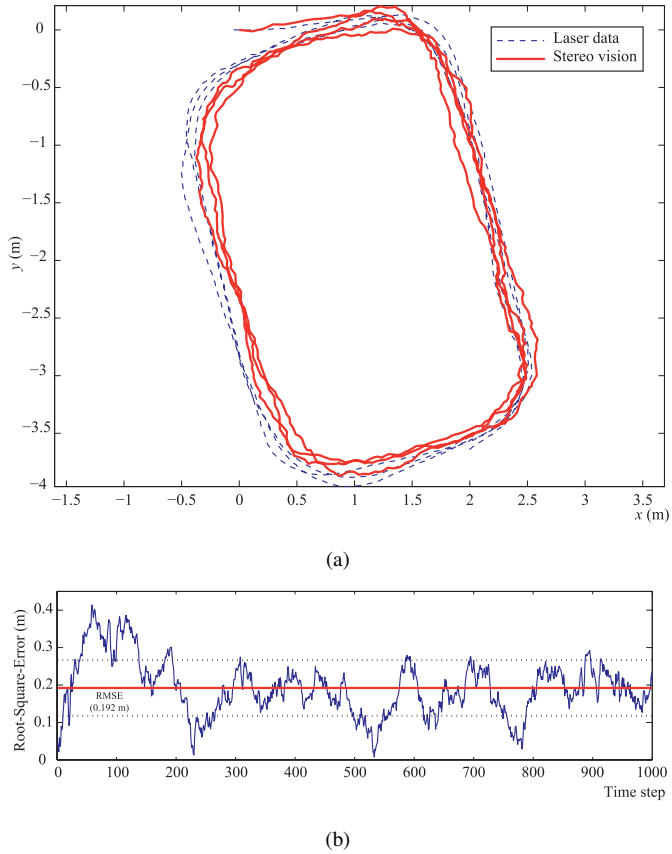


Figure 4.6: (a) Comparison between the robot path estimated through a RBPF based on laser data (blue, dashed line) and our visual SLAM approach (red, thick line). (b) Root-Square-Error committed by the visual SLAM method (the horizontal thick line indicates the RMSE ≈ 19.2 cm.)

figure 4.7, where it can be seen the increasing tendency due to the growing amount of stored landmarks in the map, which involves an increment in the processing time of both computing the likelihood and inserting the observations into the map.

Our implementation of the particle filter does not accomplish the insertion stage at every iteration, but only when the robot has moved a certain distance. This leads to the noisy appearance of the figure 4.7, since iterations which perform and do not perform the insertion stage alternate through time spending a mean time of 22.4 and 11.65 seconds, respectively (shown in the figure as dotted red lines), while the overall mean time for each iteration is 15.3 seconds (solid red line in the figure).

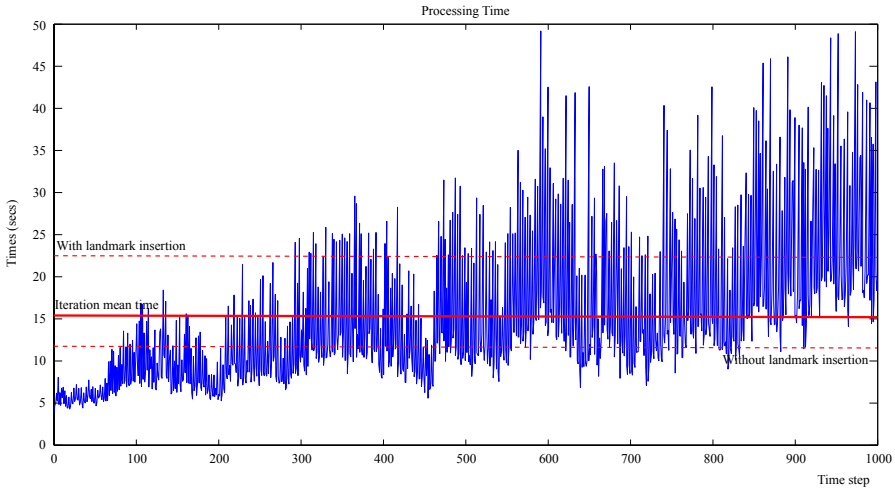
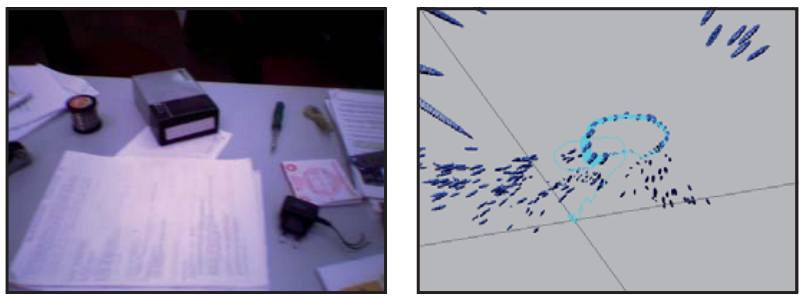


Figure 4.7: Processing time evolution for each iteration of the PF. The dotted red lines represent the mean time of an iteration in the cases of inserting and not inserting the observed landmarks into the map while the solid red line indicates the overall mean time.



(a) (b)

Figure 4.8: Our 6 DoF camera experiment. (a) A representative image employed in this experiment, and (b) the 3D estimated path of the camera.

6 DoF camera experiment

This second experiment has been carried out in order to test the suitability of our SLAM approach when handling totally unconstrained movements. Thus, a stereo camera has been moved by hand following an arbitrary 6 DoF trajectory in one of our labs, while gathering images and computing visual odometry from them. The estimated ego-motion information is subsequently employed as the motion model of our SLAM proposal.

Figures 4.8a and 4.8b show a representative image employed in this experiment and a 3D representation of the estimated path, respectively. Please note that it is difficult to obtain any kind of *ground truth* in this type of experiments, since the camera real trajectory cannot be estimated from typical exteroceptive sensors such as laser scans or sonars. Therefore, the performance of our method may only be estimated by visual inspection from a video[3] showing the sequence of the stereo images captured by the camera, as well as the evolution of both the constructed map and the camera estimated path.

As can be seen in the video, the camera is initially located on a table, and later on it is lifted up while describing a pair of turns in the air. The estimated 6 DoF path of the camera closely resembles this trajectory, thus validating our method for arbitrary 6D movements.

[3] `http://www.youtube.com/watch?v=b73W53Kwgjw`

4.3 A Complete Stereovisual SLAM System Based on Sparser Relative Bundle Adjustment

In recent years, classic bundle adjustment (BA) techniques have experienced a rebirth in form of computationally efficient methodologies to perform real-time visual SLAM, also known as *structure from motion* within the computer vision community.

BA was initially devised as a batch, off-line procedure that takes as input a collection of images taken from different viewpoints and, then, jointly minimizes the errors in the position of a set of 3D landmarks (LMs) and the poses of the cameras that captured the images of the scene, usually taking the name of *keyframes* (KFs). In this framework, the problem is typically represented as a graph (as the one shown in figure 2.11) where LMs and KFs are the nodes and edges represent constraints between them. The use of sparse algebra methods such as sparse Cholesky decomposition [37], Schür complement [152] or, more recently, inexact Newton type algorithms to deal with large, unstructured, datasets [1] have all lead to efficient BA solutions. Moreover, as stated in section 2.3.2, different parameterizations have been proposed in the technical literature. Thus, global and relative coordinates for the LMs positions and KFs have been applied leading to different solutions, coined global bundle adjustment (GBA) and relative bundle adjustment (RBA), each one of them presenting their own benefits and drawbacks. GBA and RBA methods were extensively discussed in section 2.3.2.

In between these two methodologies, other blended approaches have been also presented in recent years. This section follows the lately developed method proposed in [15] to alleviate the main drawbacks of RBA while keeping some of its advantages over GBA. In short, sparser relative bundle adjustment (SRBA) defines some KFs to act as local reference frames for other sets of KFs creating *submaps*. The key idea that lies beneath this formulation is to create *shortcuts* for long chains of relative poses between KFs. This approach, together with an efficient algorithm to create and update shortest-path spanning trees, are the key aspects of SRBA, which will be fully explained in section 4.3.2.

However, there exist key vision problems that were not addressed in [15] such as feature detection and description, data association or outliers management. The experiments presented there mainly focused on proving its performance in terms of computational complexity, thereby assuming an already solved, robust front-end that supplied properly associated, outlier-free data to the SRBA engine. In this sense, SRBA performance was tested by generating a large synthetic dataset of 3D landmarks and driving a virtual monocular camera through the environment while gathering noise-free measurements. A nearly constant time complexity during standard navigation was demonstrated and, most importantly, with bounded, loop-size-independent cost

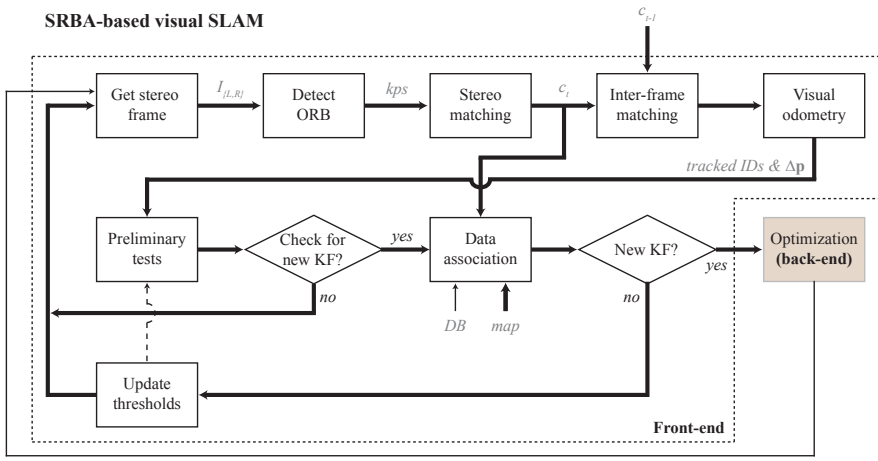


Figure 4.9: Scheme of the proposed SRBA-based visual SLAM system where the front-end is highlighted.

at loop closure, revealing itself as a promising candidate to fill the gap between GBA and RBA.

The contribution of this thesis in this context is the development of a proper front-end based on a stereo camera that completes a SRBA-based back-end in order to build a complete visual SLAM solution. Then we evaluate the performance of such system with real data extracted from both virtual and real scenes, demonstrating in this way its potential when dealing with visual SLAM. In addition, keyframe creation is also addressed in this chapter, defining a set of heuristics to properly decide when to insert new keyframes into the graph.

Figure 4.9 shows the general scheme of an iteration performed by the complete system proposed here, sketching all the stages from the capture of the stereo images to the joint estimation of the landmarks positions and the robot poses up to the current time-step, carried out by the back-end. Note that the procedures included in our front-end have been emphasized in this figure, since they will be explained next, whilst the back-end has been represented by a single brown block. Later figures will extend such representation of the back-end, though.

As the core of our proposed front-end we employ the ORB feature detector [165] (described in section 2.2.1) to extract keypoints in the stereo images. Relying on the FAST and BRIEF methods, ORB is not only a fast and reliable feature detector but also provides binary descriptors for the keypoints, which are efficiently matched through measuring Hamming distances. Data association is aided by the management of an image database based on binary bag of words [62]. At each iteration, the set of all the ORB descriptors extracted in an image are employed to query the database

looking for the most similar image stored within, hence leading the process of matching features between the current observation and the stored map.

Further details regarding the steps shown in figure 4.9 will be presented in following sections. However, most of the approaches employed here have already been stated in previous sections, hence no in-depth explanations will be provided but only a brief description of them.

4.3.1 Front-end

This section provides a description of the procedures implemented within the system front-end, taking as input a pair of stereo images and yielding a set of associated visual features between the current observation and the stored map.

Keypoints – ORB Features

Due to real-time computational requirements, valid feature-based visual SLAM frameworks must rely on efficient methods to detect keypoints and extract descriptors that are reliable and fast to both compute and match.

Among the existing detectors, probably the most popular one nowadays is the FAST detector [161], based on grey level comparisons in a pixel circle around keypoint candidates. However, being simply a detector, it does not provide any descriptor for the detected features. Relying on this approach, the ORB method [165] combines the FAST detector with a variant of the BRIEF binary descriptor [23], which is also based on comparisons of pixels grey level. Furthermore, this method deals with the principal lacks that FAST and BRIEF approaches suffer from. Thus, ORB detects FAST features in a pyramid of images, subsequently performing non-maximal suppression based on the Harris score of the detected keypoints. Then it computes the keypoint principal orientation through the *intensity centroid* [160] and builds a binary rotation-invariant BRIEF descriptor according to it. An example of ORB features detected in an image is shown in figure 4.10. In this work we have employed the ORB implementation within the publicly available OpenCV [4] software package.

The Hammming distance (please, refer to section 2.2.2) is applied to match the descriptors of the so-extracted keypoints, consisting basically of counting the number of different bits between them. The lower this distance is, the more similar the keypoints are. Further details regarding the FAST, BRIEF and ORB methods can be found in section 2.2.1.

Finally, in order to provide tracking information to later stages, an unique ID is assigned to every stereo match found following this approach.

[4] `http://opencv.org`

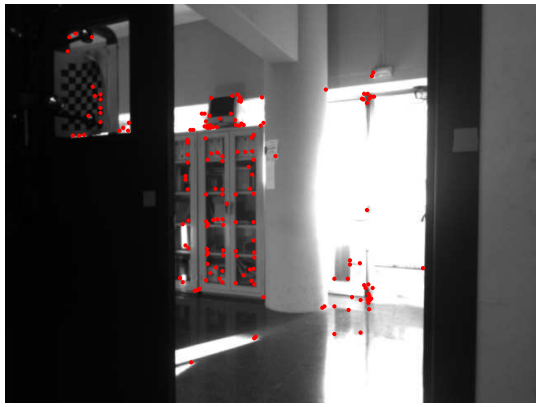


Figure 4.10: An example of ORB features detected in a real image.

Visual Odometry

After stereo correspondences are found, we perform an *inter-frame* matching process that can be understood as a simplified version of the data association block that will be explained later. It operates similarly to stereo matching but by employing, in this case, the current matched features and those extracted at the previous time-step (c_t and c_{t-1} in figure 4.9, respectively). Like stereo matching, this process relies on descriptor distances and epipolar geometry restrictions, performing as follows:

1. Keypoints from current and previous left images are matched according to the Hamming distance between their descriptor yielding a set of candidate correspondences between current and previous time-steps. Analogously, this is repeated with the right images.

2. To refine the found correspondences, the two-view epipolar geometry restriction between the current frame and the previous one is exploited by means of the computation of the left and right fundamental matrices. Therefore, keypoints in the previous left (or right) image will only match keypoints lying on their correspondent epipolar line on the current left (or right) image.

These inter-frame matches are subsequently employed to estimate the camera ego-motion between consecutive time-steps by means of the visual odometry approach proposed in section 3.3. As a reminder, such an approach is based on an iterative optimization method that minimizes the re-projection error of the current observation with respect to their associated landmarks in the map (refer to equation (3.41) in section 3.3). However, instead of using a standard quadratic cost function, our visual odometry approach implements the robust kernel-based outlier rejector, coined ERODE and already described in section 3.3. Although most of the outliers were detected and deleted through the fundamental matrix-based filter described above, some

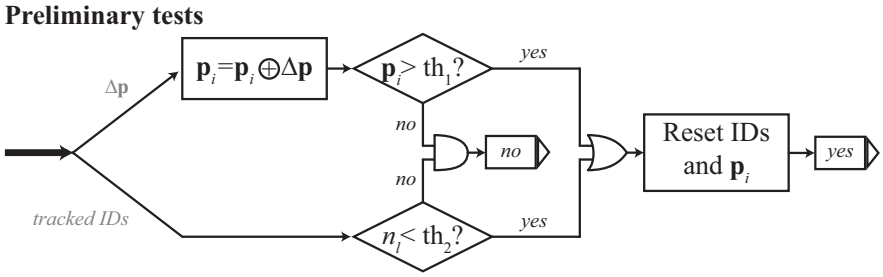


Figure 4.11: Decision scheme for keyframe creation. Incremental change in pose $\mathbf{p}_i$ is monitored as camera moves until it rises above a certain threshold (th_1). On the other hand, the number n_t of features tracked by the visual odometry system is also observed until it falls below another threshold (th_2). Once any of these conditions fulfills, both the incremental pose and the IDs are reset.

of them may still become unperceived, thus corrupting the set of matches and leading to unreliable results achieved by the back-end.

It is important to remark that, although visual odometry is computed at every time-step, we follow the usual practice in literature of keyframes not being created that often, but only when a set of conditions are fulfilled, as explained next.

Keyframe creation decision

Creating a new keyframe is a computationally expensive process because it implies inserting new variables and constraints in the graph and optimizing the whole system. This is specially noticeable when working with real data, since the number of features detected in the images may become considerably large. However, in [15] little attention was payed to this issue as working with synthetic sparse landmarks allowed the system to insert keyframes at every time-step. In this work, on the contrary, a set of heuristics has been defined to decide when to create new keyframes, following the basic idea of inserting a new one only when the explored area is becoming different enough from the last stored keyframe.

Figure 4.11 displays our proposed scheme for deciding when to create keyframes. The process starts from the visual odometry output, which consists of: (i) a list comprising the IDs of the tracked keypoints (assigned during stereo matching) and (ii) the estimation of the pose change between consecutive time-steps ($\Delta\mathbf{p}$). The former is employed to keep track of the IDs assigned to the keypoints that have been correctly associated between consecutive time-steps. The drop of the number n_t of *tracked* keypoints below a certain threshold indicates that the observed scene is becoming significantly different. The latter, in turn, is employed to incrementally build a vector $\mathbf{p}_i$ that stands for the transformation between a certain camera pose and the one at the current time-step. Thus, as the camera moves, $\mathbf{p}_i$ incrementally grows in terms of translation and/or rotation. When such translation or rotation rise above some pre-

defined thresholds, the camera is considered to have substantially moved from the previous keyframe and, therefore, it might be observing a significantly different area.

Any of these two indicators will trigger a data association process that will definitely confirm whether or not, in fact, the current observation has little in common with our knowledge of the environment. In any case, both the monitored incremental pose $\mathbf{p}_i$ and the IDs of the tracked keypoints are reset, as the decision scheme will be started again from the beginning. Our proposed data association process is explained next.

In this work, the number of associated visual features yielded by data association represents a measure of the similarity between the compared keyframe and the current images. Thus, if such number falls below a certain threshold (refer to figure 4.9), a new keyframe is created with the information extracted from the current pair of images. Otherwise, the present observation is considered to still share sufficient information with the most similar keyframe observed up to the current time-step, hence discarding the addition of a new keyframe. Finally, the thresholds employed in the decision of keyframe insertion are updated according to the number of visual features in order to adapt their corresponding tests to our knowledge of the *measured* similarity.

Data Association

One of the key stages in the design of SLAM systems is the so-called *data association*. This term stands for the process of looking for correspondences between the current observation and the knowledge that the system has about the environment, i.e. the map. As mentioned in the background chapter (section 2.1), a special situation of paramount importance arises when the current observation matches to landmarks stored some time ago, hence implying a re-observation of an already explored area. This is called a loop closure.

In this work, data association follows the approach of the above-explained inter-frame matching process and extends it by introducing a previous stage that selects the most similar KF to the current one among all the already observed by the camera. Again, this previous stage relies on the ORB descriptors computed during keypoint detection to identify the corresponding observed landmarks.

However, the search for correspondences turns harder as the map grows since the observations have to be matched against an increasing number of stored landmarks, eventually becoming computationally intractable. Therefore, smart ways of storing the map have been proposed to deal with this issue by reducing the search space where to look for correspondences, similarly to the aim of using two-view epipolar geometry for stereo matching. Among the existing techniques that alleviates data association computational burden, we employ here a bag of words [62] based on ORB binary descriptors, as was already described in section 2.2.4.

Figure 4.12 shows a schematic view of the different stages carried out in this work to perform robust and reliable data association. In our approach we make use of a pre-built word vocabulary to summarize all the ORB descriptors extracted from an image (we only employ the left image) into a single word w_i that is stored within

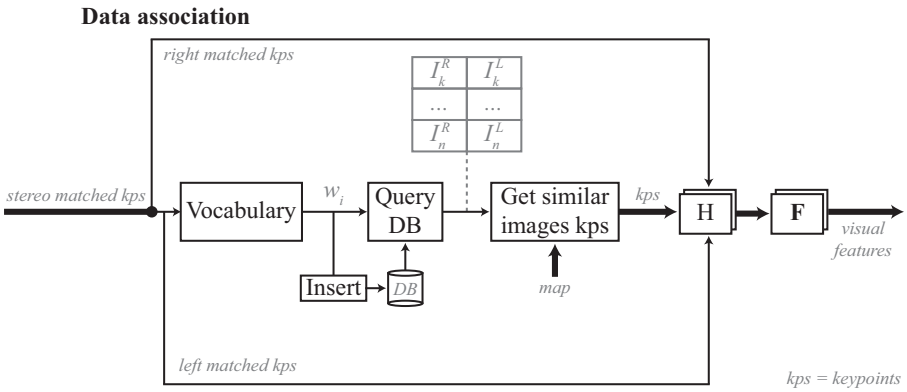


Figure 4.12: Scheme of the stages that compose our proposed data association procedure. Matched ORB keypoints (kps) from the left image are converted into a vocabulary word w_i employed to query (and which is subsequently inserted into) the image database (DB), retrieving a list of similar images. Then, current left and right keypoints are matched against those from the retrieved similar images by means of the Hamming distance between their descriptors (H blocks). Fundamental matrices **F** are finally employed to discard outliers.

the database. As new images are captured, the so-obtained words are employed to query the database, retrieving this way a small list containing the most similar images within it. Therefore, the query result restricts the search for correspondences between individual landmarks to only those belonging to the most similar images instead of the whole map.

Once the most similar stereo images are retrieved, their keypoints and those from the current image pair are matched following the inter-frame matching process explained in section 4.3.1, yielding a set of associated visual features which will represent the input of the SRBA-based back-end. Finally, in order to consider an association as valid, all the matched projections of the landmark in the four images must be consistent with the epipolar geometry and the distance between their descriptors must fall below a threshold. This methodology reduces the presence of outliers in the data association output.

It is important to note that, during normal operation, i.e. as the camera explores unknown areas, the most similar image will be the last one inserted into the database, but, on the contrary, older images will be also retrieved when visiting already explored zones, hence detecting loop closures.

4.3.2 Back-end

After deciding the insertion of a new keyframe, the set of visual features obtained through data association feeds the SLAM system back-end, which is in charge of optimizing the estimation of the system state up to the current time-step, as described in the following sections. The scheme shown in figure 4.13 sketches the stages carried

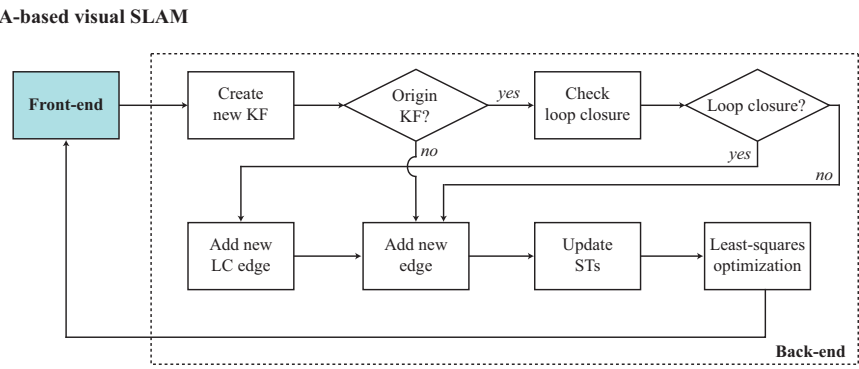


Figure 4.13: Back-end description.

out within our proposed back-end, which will be further explained. As already said, this back-end is based on a relative bundle adjustment technique that represents the problem by a pose graph whose nodes correspond to keyframes and landmarks while the edges (constraints) between them are the unknowns of the problem, which is explained next.

Sparser Relative Bundle Adjustment

Sparser relative bundle adjustment (SRBA) [15] was proposed as a blended solution in between global bundle adjustment (GBA) and an *intuitive* implementation of relative bundle adjustment (RBA) (explored in [175, 176]). Although being explained in detail in section 2.3.2, we summarize here the basics of these approaches in order to provide context for SRBA.

In short, GBA methods impose a single KF to be the *base* for all the created edges, either *keyframe-to-keyframe* (KF-to-KF) or *keyframe-to-landmark* (KF-to-LM). Thus, GBA solutions lead to globally consistent maps since all the KFs and LMs positions are jointly optimized in a single, common reference frame at each time-step. The main drawback of GBA that motivates the use of *relative* approaches to BA lies on the computational burden that global formulations suffer when dealing with large graphs, specially during loop closures that affects many nodes. In those situations, GBA cost becomes prohibitive since the involved matrices, although sparse, turn dense enough when factored as to render the method useless in practice.

On the other hand, standard RBA methods connect every KF to its immediately preceding one [175, 176], with the exception of loop closure situations. Besides, KF-to-LM edges are set to connect landmarks with the KFs from where they were observed for the first time (their *base* KF). In addition, relative methods optimize only a small part of the graph, leading to smaller matrices that can be handled efficiently. However, such matrices are significantly denser than their global counterparts, as can be noted when comparing figures 2.12 and 2.14 shown in section 2.3.2. This is due

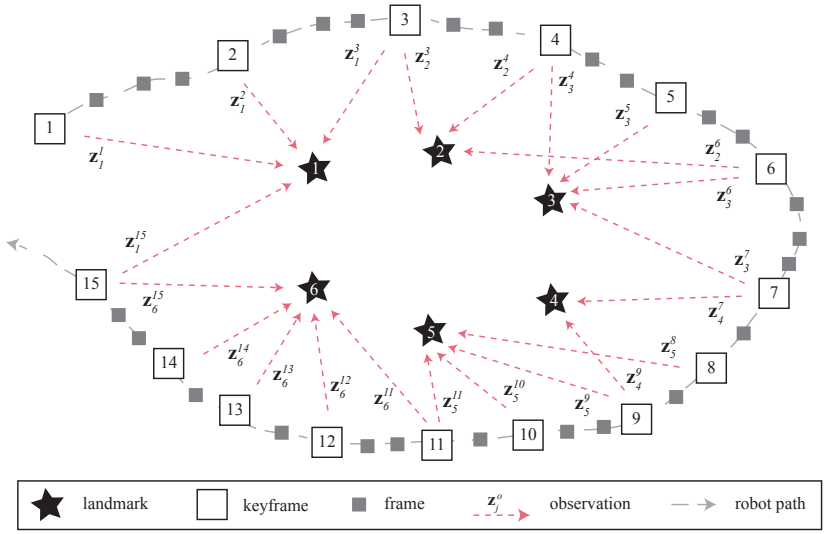


Figure 4.14: Example of a bundle adjustment problem.

to the need of performing chains of pose compositions at each observation, since the base-KF referred coordinates of the stored landmarks have to be transformed to the current observer KF. This, in turn, is a consequence of the relative definition of KF poses in RBA, which builds *linearly* connected graphs.

RBA approaches also introduce two new concepts that are involved in the optimization process:

- Maximum topological distance (D_{max}), which sets the limit up to where optimize the graph at each time-step, i.e. the maximum topological distance from the current observer KF that will be included in the optimization. In addition to that, it also determines the minimum distance between the base KF of a certain observed landmarks and its current observer KF to reveal a loop closure, as explained later.

- Shortest-path spanning trees (STs), that indicate the shortest paths between the KFs within the graph, determining the chains of pose compositions employed to transform landmark coordinates. In this approach, STs are maintained only up to distance D_{max} from each KF. An algorithm for incrementally update STs over time is proposed in [15] and will be summarized later.

As a consequence of only optimizing a part of the graph, RBA approaches only create locally consistent maps near the current pose of the robot. Nevertheless, this limitation does not render RBA useless to solve several problems of autonomous navigation, such as path planning or obstacle avoidance. Furthermore, RBA represents an

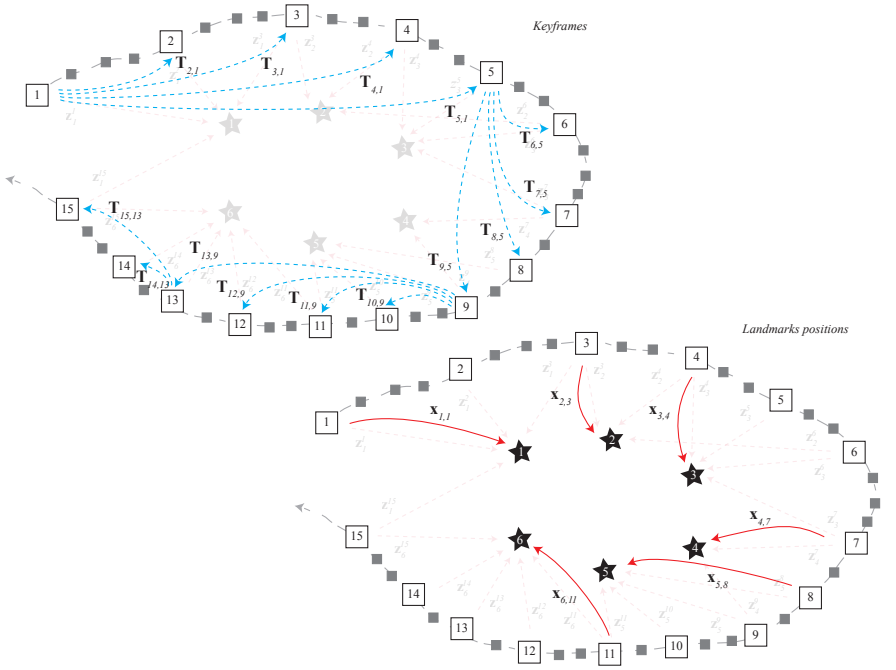


Figure 4.15: Graph representation under a SRBA approach. Blue-dashed arrows represent edges between origin KFs and submap members, while red-solid arrows indicate landmark positions with respect to their base KFs. In this example, the number of KFs within each submap has been set to 5.

efficient approach to perform bounded-time visual SLAM, although the computation of a global map, if needed, must be performed as a separate optimization process.

The sparser relative bundle adjustment approach, by carefully changing the policy about how to create new edges, introduces a new formulation where some KFs (called *origin* KFs from now on) are dynamically selected as origins of local reference systems, creating *submaps*. All the KFs within a submap are referred to its origin KF. The underlying idea of this representation is to generate edges between origin KFs so that paths along KFs become shorter, thus reducing the loss of sparsity that the standard RBA approaches incur in, while keeping their advantages over the global alternative.

In order to illustrate this, we refer again to the example scenario proposed in section 2.3.2 whose representation is reproduced in figure 4.14, for the convenience of the reader. A possible graph created under the SRBA framework for such scenario is shown in figure 4.15. Note how the RBA *linearly* connected graph shown in figure 2.13 has been divided in four submaps of size 5 whose origin KFs are those numbered 1, 5, 9 and 13. All the KFs within the submaps are referred to their origin KF and, similarly to the RBA case, their pose transformations with respect to it have

been labeled by $\mathbf{T}_{j,b}$, with j standing for the KF index and b representing its reference frame (or origin KF). On the other hand, landmarks positions $\mathbf{x}_{j,b}$ are represented following the same convention as for the standard RBA approach, with their coordinates referred to their base b KF (red-solid arrows in figure 4.15). Both the KFs poses and the landmarks positions define the system state (i.e. the problem unknowns), to be determined from the observations, as in any bundle adjustment problem.

It can be seen in the figure that edges between origin KFs have been created (e.g. from KF#1 to KF#5) so that when the coordinates of a landmark have to be transformed from its base KF b (member of submap s_i) to the observer KF o (member of submap s_j), the corresponding chain of poses only involves the existing edges between the origin KFs of the traversed submaps (from s_i to s_j) in addition to those between KFs b and o and their respective submap origins. Generally, such path across the graph will be shorter than the chain of poses stated by standard *linearly* connected RBA, being this effect more beneficial as the defined submap size grows. Thus, this approach may be intuitively understood as a way of defining *shortcuts* in the pose graph to shorten paths along KFs chains, therefore leading to a sparser matrices, as shown later.

Finally, it is important to note that, by setting up submap sizes from 1 to infinity, SRBA seamlessly integrate all the possible KFs configurations ranging from pure relative BA to global BA, proving the flexibility of this blended approach.

New KF management and loop closure

The first block in figure 4.13 is defined in a purely implementation level and simply represents the process of creating the necessary data structures to store a new KF n. Subsequently we determine if n is simply a member belonging to an already existing submap in the graph or, on the contrary, it defines a new one. The edge creation policy for the proposed SRBA formulation establishes a fixed number of KFs within the submaps so that when this limit is reached, a new submap is started with the new keyframe acting as its origin KF.

In any case, a new edge e is introduced between the last origin KF and n, although, if the latter is defined to be origin of a new submap, the system additionally searches for loop closures. This is achieved by checking if the visual features produced by the front-end contains a significant number of associations with landmarks whose base KFs are more distant to the current KF than D_{max}. In this case, the system determines that the camera is observing a far, already explored area, thereby detecting a loop closure and adding a new edge e_{lc} to the graph. However, there is no difference in the way that edges e and e_{lc} are inserted, no matter if they represent loop closure or simple constraints between origin KFs and submap members.

Shortest-path spanning trees

Regarding shortest-path spanning trees, the SRBA approach presented in [15] creates and maintains STs for the graph up to a maximum distance of D_{max}, coinciding with the maximum map area that is also maintained locally consistent. There, the set of

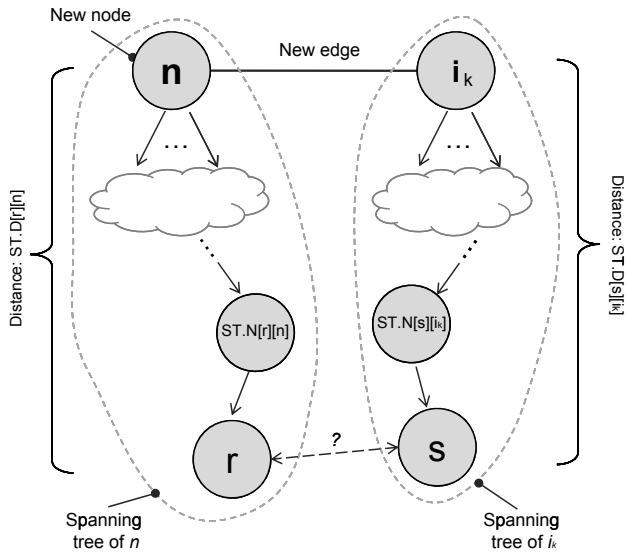


Figure 4.16: Example of ST update (figure taken from [15]).

STs is implemented by means of two symbolic tables containing, for any two KFs i and j, both the topological distance in the graph between them, and the next node in the path from one to the other. Therefore, the shortest chain of poses between both KFs can be build by traversing these STs.

Although a STs for all the nodes in a graph could be easily created through classic breadth-first-search algorithms [130], SRBA pursues the incremental build and update of both the graph and the STs as the camera explores the environment. Hence, the original paper proposed an incremental update algorithm for STs which we summarize next. When a new node is added to the graph, a set of N edges connecting to other existing nodes must be defined (although usually it will be only one). Let n be the new node in the graph and i_k the destiny node for one of the created edges, as shown in figure 4.16. The STs for all the nodes belonging to the STs of the involved n and i_k nodes must be check for update. This is performed by measuring the topological distance between any node r within the ST of n and any node s within the ST of i_k, which is determined by summing the distance between r and n (already stored in the ST of n), the edge $n-i_k$ and the distance between the s and i_k (stored again in the ST of i_k). If n and i_k were not already in each other's ST and the resulting distance is D_{max} or less, both STs are updated to account for this new relation. Otherwise, if there was already a path between such nodes, their STs are only updated if the new path is shorter than the existing one. This has to be repeated for all the created new edges. Further implementation details can be found in the original paper [15].

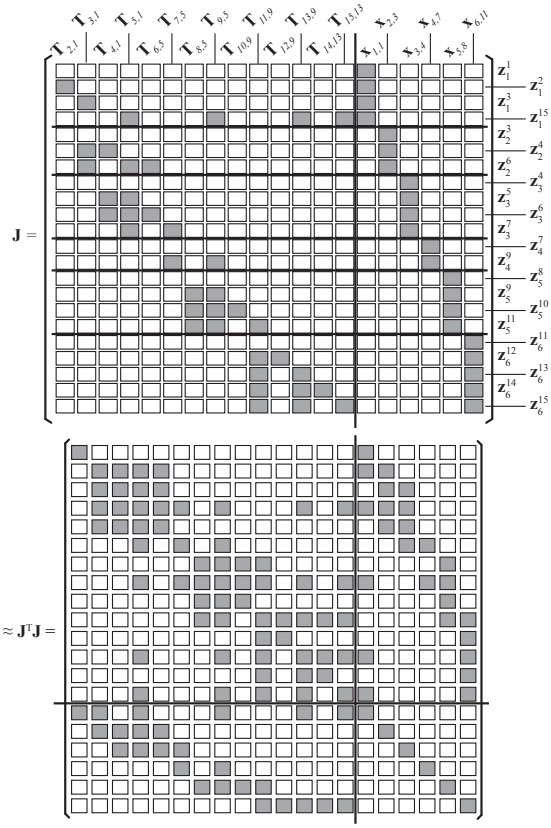


Figure 4.17: Sparsity pattern of Jacobian and Hessian matrices under a SRBA approach for the scenario shown in figure 4.14.

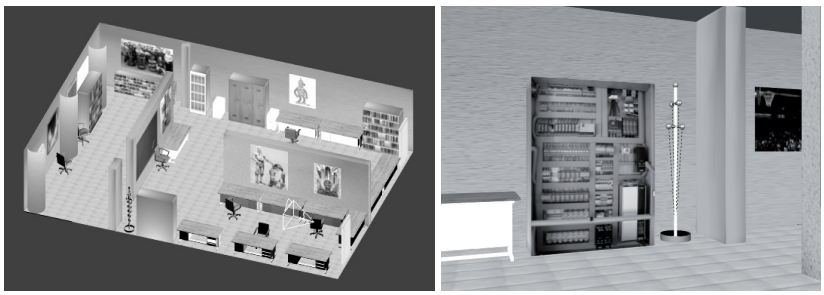
Least-squares optimization

The graph optimization is performed by an iterative Levenberg-Marquardt algorithm that minimizes the re-projection error ($\Delta \mathbf{z}_j$) of the observed landmarks in the current pair of stereo images, defined by the cost function shown in equation (2.69), reproduced here for completeness:

$$F(\mathbf{s}) := F(\mathbf{p},\mathbf{x}) = \frac{1}{2}\sum_j \Delta \mathbf{z}_j^{\mathsf{T}} \Lambda_j \Delta \mathbf{z}_j . \tag{4.16}$$

Minimizing this function achieves the joint estimation of both the poses of the cameras ($\mathbf{p}$) that captured the images (KFs) and the landmarks positions ($\mathbf{x}$), as explained in section 2.3.2, leading to the equation:

$$\mathbf{H}\Delta\mathbf{s} = -\mathbf{g}, \tag{4.17}$$



(a) (b)

Figure 4.18: (a) Plan and (b) example image of the synthetic dataset.

with **H** and **g** being the Hessian matrix and the gradient of the cost function, respectively and $\Delta\mathbf{s}$ standing for the incremental change in the system unknowns $\mathbf{s} = (\mathbf{p},\mathbf{x})$.

The sparsity of the Jacobian and Hessian matrices play here an important role in terms of computational burden, demonstrating the SRBA approach improved performance with respect to the standard RBA method. Figure 4.17 represents the sparsity patterns of such matrices for the example proposed in figure 4.14 under a SRBA formulation. In comparison to those for the GBA and standard RBA approaches described in section 2.3.2 (refer to figures 2.12 and 2.13), it can be noted that SRBA falls in between both formulations, yielding sparser matrices than RBA but denser than GBA.

4.3.3 Experimental Results

In order to validate the SRBA-based method proposed here, some experiments have been carried out with both synthetic and real images in indoor and outdoor conditions.

Synthetic Images Dataset

In our first experiment, we have moved a virtual stereo camera through a synthetic office-like environment created with the 3D design software Blender[5] while capturing stereo images to form a dataset of synthetic images. A representation of such environment and an example of the captured images are shown in figure 4.18. The use of a virtual environment to test our approach is motivated here by the fact that the camera movement is known beforehand, hence making a *ground truth* available for the camera path.

As presented before, the main advantage of our relative representation is that the time spent in creating and inserting new keyframes (including the creation of the

[5] `http://www.blender.org/`

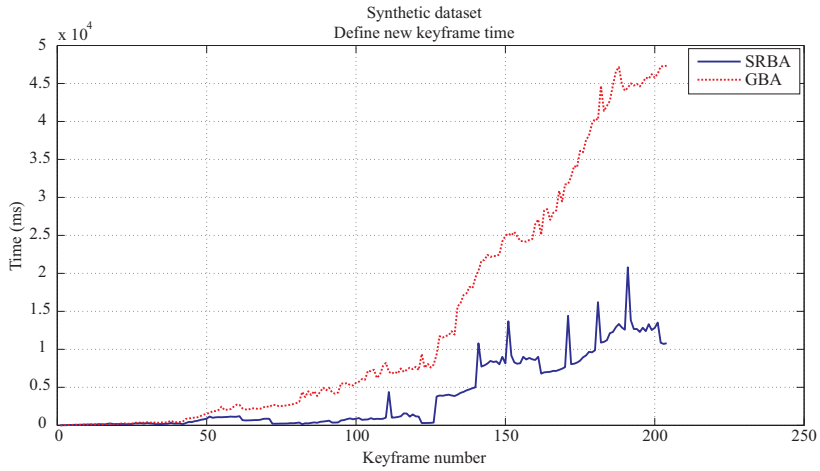


Figure 4.19: Comparison of inserting new keyframe time (including graph optimization) for SRBA (blue-solid line) and GBA (red-dashed line) for the synthetic dataset.

Hessian matrix and the subsequent optimization process) remains bounded, unlike the global approach. This is a consequence of the smaller matrices employed in our method due to the existence of submaps and a limit depth for graph optimization. To test this, we have employed the same dataset as input for both our SRBA-based approach and a pure global one (GBA).

In this sense, a performance comparison can be seen in figure 4.19, where the keyframe creation time (including the graph optimization step) is plotted for every keyframe defined during the experiment under both approaches. Please, note that we have not performed any optimization in our code to pursue efficient performance, hence the mentioned plot should be understood as a mere comparison between both techniques, disregarding the absolute time values. Figure 4.20 also presents the number of landmark positions that are optimized at each keyframe insertion for the SRBA method, showing that it remains bounded over time, unlike global approaches.

Finally, we present in figure 4.21 a comparison between the estimated camera trajectory and the ground truth. The path of the camera has been obtained through a final full optimization of the graph built through our method in order to determine both the keyframes poses and landmarks positions within an unique global reference system whose origin is set at the first camera position. We also present in figure 4.22 the errors at the estimated camera positions associated to all the keyframes.

It has to be noted that, although the creation of an unique global map can not be considered part of our SRBA formulation, the presented path comparison and accuracy results represent indicators about the suitability of our method to store and manage enough information to create a global map in case that the application demands it. At the same time, it allows a more efficient way of estimating a camera

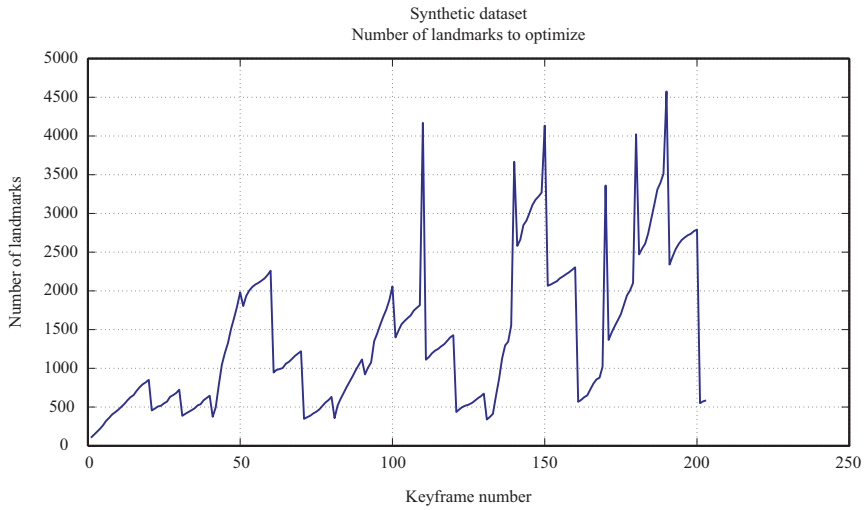


Figure 4.20: Number of landmarks to optimize with SRBA for the synthetic dataset.

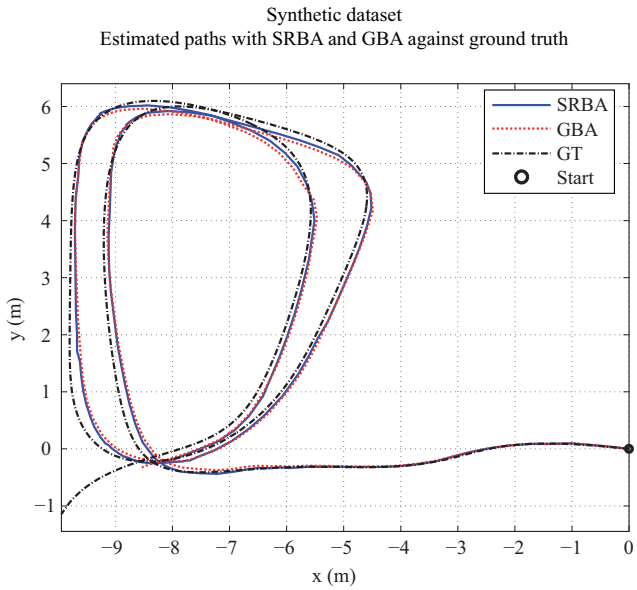


Figure 4.21: Estimated paths for the synthetic dataset with SRBA (blue-solid line) and GBA (red-dashed line). Ground truth is also represented (black dashed-dotted line).

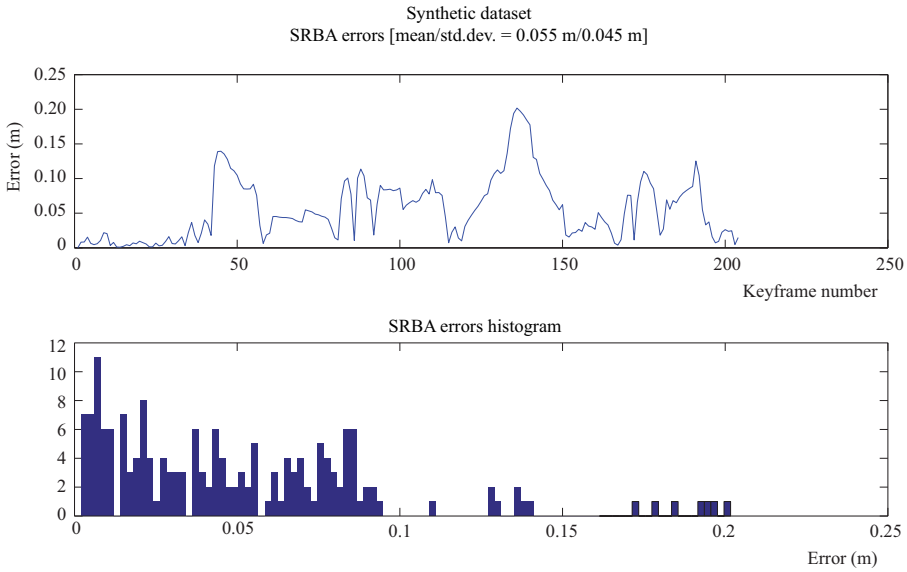


Figure 4.22: Absolute error and histogram of errors for our SRBA-based method with respect to the ground truth for the synthetic indoor dataset.

trajectory while creating a map of the environment which is locally consistent and useful enough for many applications.

Real Indoor Dataset

In this second experiment we have tested our method against a real image dataset gathered with our mobile robot Sancho [68], while being manually driven through our laboratory following a trajectory that includes a loop.

Unfortunately no ground truth could be provided for this experiment since no other sensors where available on the robot. Therefore both the obtained map and the robot path can only be validated by means of visual inspection.

Figure 4.23 presents the timing comparison when creating new keyframes under both our SRBA-based approach and a global one. Note that, regarding the time absolute values, similar considerations to those mentioned in the previous experiment must be taken into account here.

Two examples of the images employed in this experiment can be found in figure 4.24a, while figure 4.24b shows the estimated trajectory of the camera for both the GBA method and our SRBA proposal.

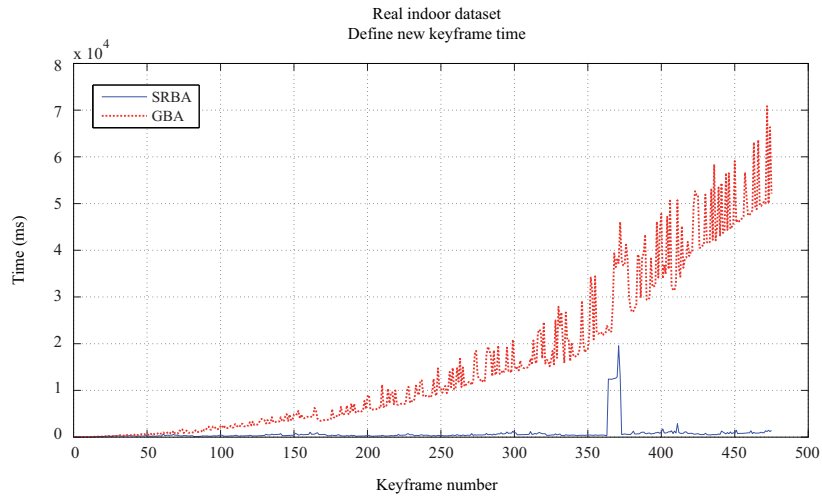


Figure 4.23: Inserting new keyframe time comparison for SRBA (blue-solid line) and GBA (red-dashed line) for the real indoor dataset.

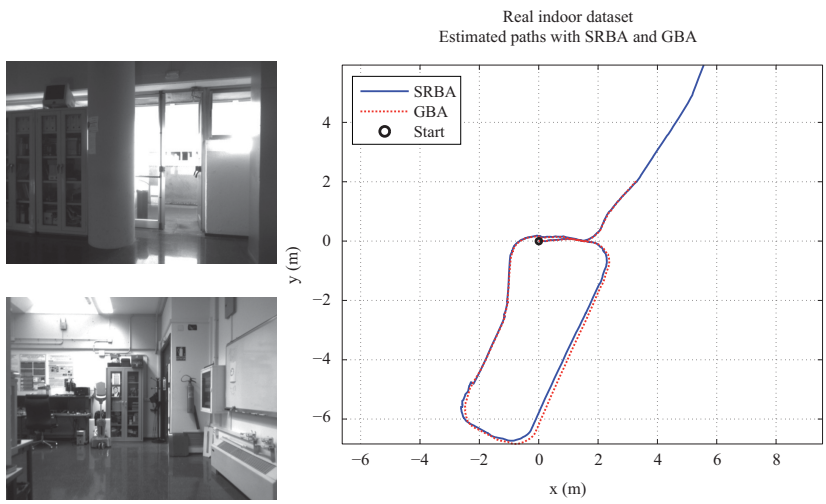


Figure 4.24: (a) Example images and (b) estimated path for the real indoor dataset.

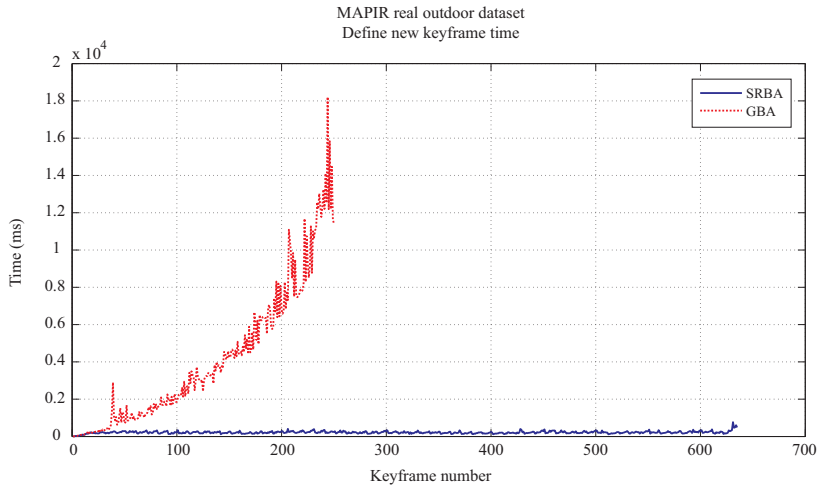


Figure 4.25: Inserting new keyframe time comparison for SRBA (blue-solid line) and GBA (red-dashed line) for the MAPIR real outdoor dataset.

Real Outdoor Dataset

Finally, we have employed two outdoor datasets to test our method under more challenging conditions. In particular, we use one of our published stereo datasets (which will be extensively described in chapter 5), and one of the *road* datasets from the KITTI Vision Benchmark Suite [60, 64].

First, we have chosen the fragment number 7 of the datasets presented in section 5.3, which will be referred here as MAPIR outdoor dataset. This fragment shows a ∼0.7 km long loop within an urban scenario gathered by an on-board stereo camera placed on a standard car. Here we have considered the positioning information provided by a standard differential GPS device as the experiment ground truth.

The timing comparison when creating new keyframes for this experiment is shown in figure 4.25. Note that due to the size of the resulting map, with a total amount of more than 110k landmarks, the time spent when inserting keyframes by the GBA approach rapidly becomes excessive, rendering the method useless. Therefore, beyond the insertion of the keyframe #250, GBA time values are no longer shown in the plot. The small peak that appears around the keyframe #630 for the SBRA-method is due to the loop closure detection.

One of the main drawbacks of this dataset stands on the relatively small baseline of the employed stereo camera (∼12 cm), which makes difficult the estimation of distant points' 3D position. Still, the achieved results can be considered to be promising. A comparison between the estimated camera paths and the GPS-based ground truth is shown in figure 4.26 while the errors between the trajectory estimated by our proposal and the ground truth are presented in figure 4.27. The camera path estimated by the GBA method is only depicted up to keyframe #250.

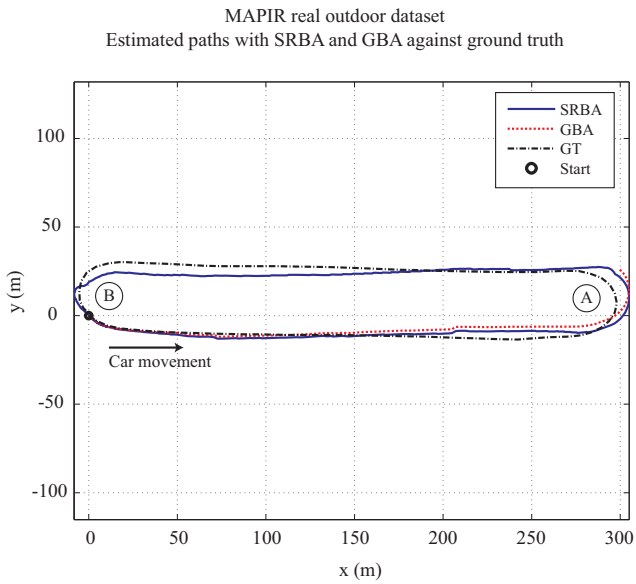


Figure 4.26: Estimated paths for the MAPIR real outdoor dataset with SRBA (blue-solid line) and GBA (red-dashed line). Ground truth is also represented (black dashed-dotted line).

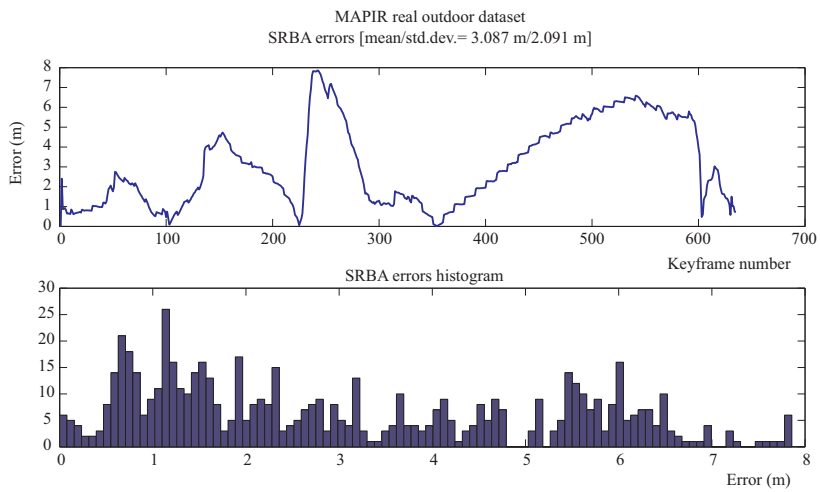
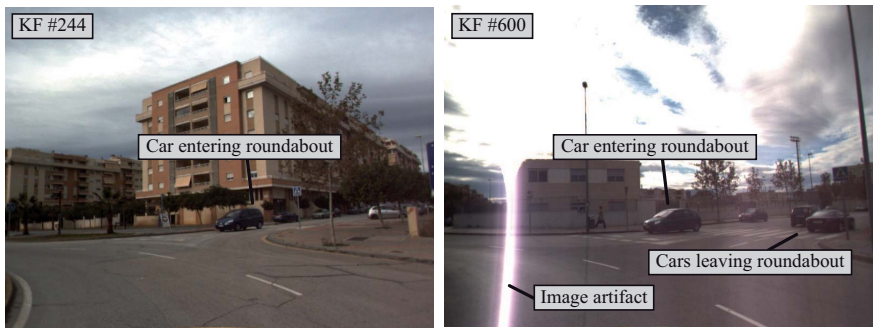


Figure 4.27: Absolute error and histogram of errors for the SRBA method with respect to the ground truth for the MAPIR real outdoor dataset.



(a) (b)

Figure 4.28: Example of problematic images in the MAPIR outdoor dataset. (a) A car entering a roundabout and (b) image artifact and moving cars at a roundabout.

Moreover, it has to be noted that, in some areas of this dataset, the environment is not static due to the presence of moving cars and/or pedestrians. Since our system works under the assumption of a non-dynamic environment, this leads to large inaccuracies in the camera path estimation. Hence, manual assistance has been required in such areas. In particular, a car approaching the roundabout marked as 'A' in figure 4.26 and the presence of some cars entering and leaving the roundabout marked as 'B' in the same figure (nearly the end of the trajectory) are the specific areas where capturing keypoints on moving objects has been avoided. Example images of these situations are shown in figure 4.28. Finally, the presence of image artifacts due to direct sunlight (refer to figure 4.28b) has also been managed in a similar way at the very end of the dataset.

Finally, we have also taken segment 7 from the KITTI's road datasets to test our approach. This segment contains a set of 1100 stereo images of size 1226×370 px. (refer to figure 4.29 for an example image) captured by a car while moving along a ∼0.7 km long trajectory and performing a loop. In this case, the stereo camera baseline is significantly wider than the dataset employed in the previous experiment and, similarly to it, ground truth is available for this dataset.

Again, we have measured the insertion time for new keyframes (including graph optimization) for both the GBA and the SRBA approaches, presenting the results in figure 4.30. Note that, similarly to the previous experiment, the high number of landmarks in the map prevent the process of inserting new keyframes for GBA to perform in reasonable time. Loop closure detection is achieved near to KF#400 for the SRBA approach, hence the small peak shown in the figure at that point.

The paths estimated by both methods are compared against the provided ground truth and shown in figure 4.31. The graph built contained ∼400 keyframes for the whole trajectory and the resulting map was composed by more than 36k landmarks. Finally, the evolution and the histogram of the errors committed by the SRBA method with respect to the ground truth are presented in figure 4.32.

Figure 4.29: Example image for the KITTI outdoor dataset.

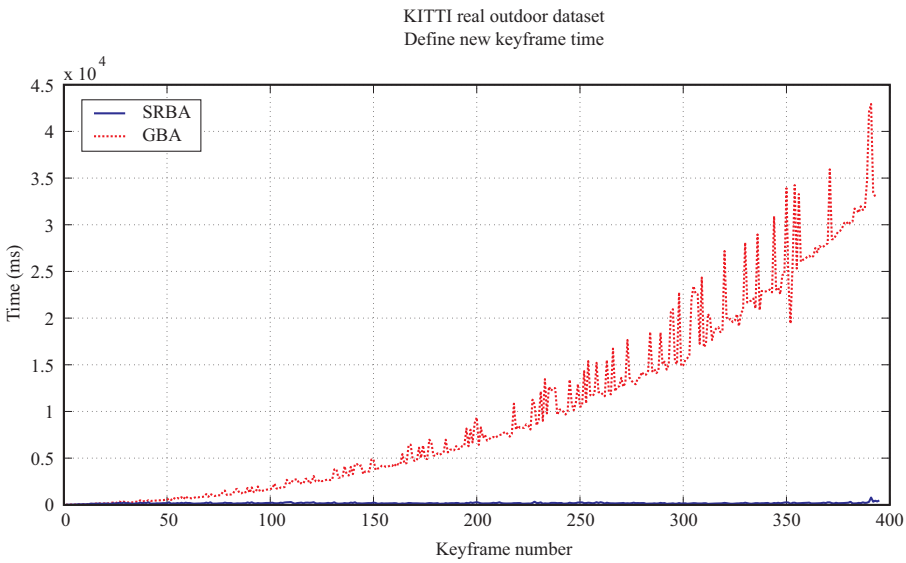


Figure 4.30: Inserting new keyframe time comparison for SRBA (blu-solid line) and GBA (red-dashed line) for the KITTI real outdoor dataset.

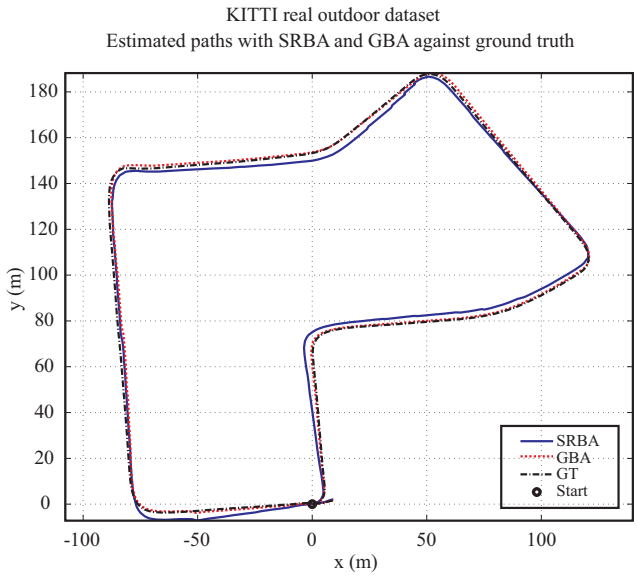


Figure 4.31: Estimated paths for the KITTI real outdoor dataset with SRBA (blue-solid line) and GBA (red-dashed line). Ground truth is also represented (black dashed-dotted line).

4.4 Conclusions

In this chapter we have addressed the SLAM problem for stereo vision systems within two different frameworks: the probabilistic approach of particle filters methods and a sparser relative bundle adjustment methodology.

In our first proposal, based on Rao-Blackwellised PFs, the robot ego-motion is estimated through a closed-form formulation to perform 6D visual odometry which models the uncertainty of the pose increment estimation as a Gaussian distribution. This ego-motion estimation relies only on the visual information provided by the stereo camera and avoids the divergence and local-minima problems of the iterative approaches to visual odometry. On the other hand, the observation model for the RBPF algorithm considers observations as sets of landmarks determined by their 3D positions and their SIFT descriptors, as well as their associated uncertainty. As an important contribution, we avoid explicit data association by marginalizing out the observation likelihood over all the possible associations, thus overcoming the problems derived from establishing incorrect correspondences between the observed landmarks and those in the map.

An experiment with a real robot has been performed in order to validate our first proposal in the context of map building. The experimental results illustrate its adequate performance when coping with the SLAM problem and reveal the proposed models as promising approaches for stereo vision in robotics. The MSE committed by

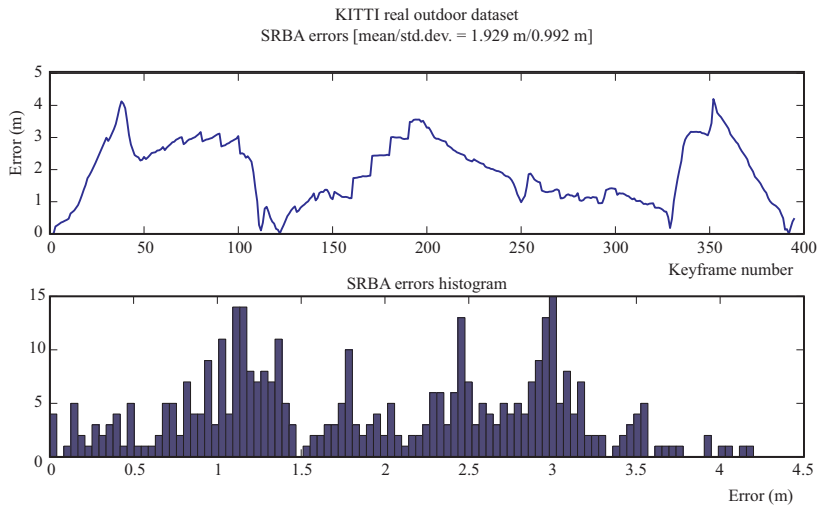


Figure 4.32: Absolute error and histogram of errors for the SRBA method with respect to the ground truth for the KITTI real outdoor dataset.

our method in comparison to a RBPF approach employing laser data is approximately 16.3 cm.

Our second proposal implements a complete visual SLAM system with a front-end based on ORB image features, and a back-end whose core is a novel, sparser version of relative bundle adjustment techniques. In this case, our proposal for visual odometry relies on ERODE, an outlier detector built up on robust kernels to improve least-squares optimization methods endurance against outliers. Data association is aided by a bag-of-words approach that reduces the search area when looking for correspondences between the current observation and the built map. The back-end is implemented following a SRBA framework, which represents the problem as a graph whose nodes are keyframes and landmarks while the constraints (edges) between them are the unknowns to be estimated. SRBA defines a methodology for adding edges that leads to the creation of submaps in the graph, in order to reduce the topological distance between keyframes. The motivation behind this formulation is the increase of the sparsity level in both the Jacobian and Hessian matrices involved in the optimization process, which is a capital drawback of relative bundle adjustment approaches. Moreover, SRBA keeps the optimization time bounded at every keyframe by applying a limitation on the number of unknowns to be optimized at the cost of maintaining a map which is only locally consistent unlike GBA approaches.

Experiments with both synthetic and real data (including two in outdoor conditions) have been provided to demonstrate the potential of our proposal to cope with the visual SLAM problem.

Chapter 5
Datasets

Overview

This chapter describes two outdoor datasets recorded in the city of Málaga which contain stereo images along with several other sensor data such as laser scans, inertial measurements and GPS information.
First, we present a collection of datasets that tries to alleviate the lack of publicly accessible datasets with a reliable ground truth. The goal is allowing a fair and coherent comparison of different methods proposed in the mobile robot SLAM literature. Providing such a ground truth becomes specially challenging in the case of visual SLAM, where the world model is 3D and the robot path is 6D. Thus, this first work addresses both the practical and theoretical issues found while building a collection of six outdoor datasets. It is discussed how to estimate the 6D vehicle path from readings of a set of three Real Time Kinematics (RTK) GPS receivers, as well as the associated uncertainty bounds that can be employed to evaluate the performance of SLAM methods. The vehicle was also equipped with several laser scanners, from which reference point clouds are built as a testbed for other algorithms such as segmentation or surface fitting.
Our second work introduces a dataset gathered entirely in urban scenarios with a car equipped with one stereo camera and five laser scanners, among other sensors. One distinctive feature of the present dataset is the existence of high-resolution stereo images grabbed at high rate (20 fps) during a 36.8 km trajectory, which makes this work a suitable benchmark for a variety of computer vision techniques. We describe the employed sensors and highlight some applications which could be benchmarked with the presented work. Both plain text and binary files are provided, as well as open source tools for working with the binary versions.

Input, more input! – Johnny 5

5.1 Introduction

The field of Simultaneous Localization and Mapping (SLAM) has witnessed a huge activity in the last decade due to the central role of this problem in any practical application of robotics to the real life. While the theoretical bases of SLAM are well understood and are quite independent of the kind of sensors employed by the robot, in practice many of the reported works focus on either 2D SLAM (i.e. they assume a planar world) or 6D SLAM (i.e. features have full 3D positions and the robot can also move freely in 3D). Typically, works in the first group rely on laser scanners while the latter employ monocular or stereo cameras.

A critical issue which must be considered for any SLAM techniques, either 2D or 6D, is the *evaluation of its performance*. Applying the scientific method to computer vision SLAM implies being able to perform rigorous benchmarking of the different algorithms in order to determine their suitability and relative performance. Usually, more recent techniques claim to be *better* in any sense with respect to previous works. Here, *better* can mean more accurate (in the case of building metric maps), less prone to divergence or more scalable, among other possibilities.

In principle, the advantages of some techniques in comparison to others should be *quantified*, but that is not a goal easy to achieve in practice. In some cases, the differences between two methods are more qualitative than quantitative, but most often measuring the accuracy of the results becomes necessary. Instead of contrasting the maps produced by the different methods (which usually relies on visual inspection), it is more convenient to consider the robot *reconstructed paths* due to its reduced dimensionality in comparison to the maps. Additionally, the evaluation of robot paths would even enable comparing a 2D method (based, for instance, in grid mapping) to a 6D technique such as vision-based SLAM. This would not be possible if the maps, of different nature, were instead employed in the comparison.

The traditional problem found by the SLAM community in this sense is the lack of a reliable *ground truth* for the robot paths. Some works have to rely on simulations to overcome this difficulty, but this approach ignores the problems that arise with real-world sensors. The existence of public reference datasets with an accurate ground truth would provide an ideal testbed for many SLAM techniques.

This chapter first mentions other works presenting datasets for robotics and computer vision applications and enumerate the contributions of the two presented here. Then they are thoroughly described in sections 5.2 and 5.3, while section 5.4 summarizes the contributions achieved here.

5.1.1 Related work

A good introduction to the SLAM problem by Durrant-Whyte and Bailey can be found in [52], which also includes a review of the few known publicly available datasets at that moment. Among them, the Victoria's park [77] has become the most widely employed testbed by the community of 2D SLAM, having been used in dozens

Dataset	GPS	GT	IMU	LS	Images	Path length
New College (2008) and Oxford city center [35]	✓	×	×	×	Mono.: color 640×480@ ~ 1 fps	College: 2 km City: 28 km
New College (2009) [181]	✓	×	✓	2	Stereo: b/w 512×384@20 fps Ladybug: 5×(color 384×512@5 fps)	2.2 km
Rawseeds datasets (2009) [26]	✓	✓	✓	4	Front: color 320×240@29.95 fps Omni.: b/w 640×640@15 fps Trino.: b/w 640×480@15 fps	Indoors: 0.89 km Outdoors: 1.9 km
MIT DARPA [90]	✓	✓	✓	13	Mono.: 4×(color 376×240@10 fps) Mono.: color 752×480@22.8 fps	90 km
The Marulan datasets [156]	✓	✓	✓	4	Mono.: b/w 1360×1024@10 fps	~ 1 km
Karlsruhe sequences [65] [66]	✓	✓	✓	×	Stereo: b/w 1344×391@10 fps	6.9 km
Ford campus [153]	✓	✓	✓	1	Omni.: color 1600×600@8 fps	~ 6 km
KITTI [64]	✓	✓	✓	1	Stereo: b/w 1392×512@10 fps Stereo: color 1392×512@10 fps	~ 50 km
Málaga 2009 dataset (dataset 1 here – [16])	✓	✓	✓	5	Stereo: color 1024×768@7.5 fps	6 km
Málaga Urban dataset (dataset 2 here – [17])	✓	×	✓	5	Stereo: color 1024×768@20 fps	36.8 km

Table 5.1: A comparison of some datasets (including the two presented here) regarding the presence (✓) or not (×) of GPS sensors, ground truth (GT), inertial units (IMU), the usage of laser scanners (LS), the kind of cameras on the vehicle and the dataset path lengths.

of works, e.g. [78, 97, 127, 142, 154, 200]. The interest of the community in this sense is clear, given the number of projects and workshops devoted to the topic [18, 186]. Many other datasets can be found in the *Radish repository* [89], most of them consisting of laser scanner and odometry data logs. Nevertheless, no previous dataset contains an accurate ground truth[1].

Regarding datasets oriented to vision-based SLAM, at the time our first dataset was published, there was no previous work where a ground truth was associated to the path of the camera or the robot. Indeed, the on-going project RawSeeds [19], aimed at SLAM benchmarking, was barely starting and no released datasets could be found yet. In addition to that, in recent years, more releases that include images of urban areas [64, 156] or both images and laser data of park-like zones [181] have also received the attention of the community, clearly reflecting the demand for this kind of publications.

Table 5.1 shows a summary of some of the above mentioned datasets along with others also publicly available for the robotics community.

5.1.2 Contribution

The presented collections of datasets makes two major contributions to the field of mobile robotics.

[1] Some datasets may include GPS positioning, without an accurate orientation.

Firstly, the release of two collections of outdoor datasets with data from a large and heterogeneous set of sensors comprising color cameras, several laser scanners, precise GPS devices and inertial measurement units. These collections provide a unified and extensive testbed for comparing and validating different solutions to common robotics applications, such as SLAM, video tracking or the processing of large 3D point clouds. The datasets comprise accurate sensor calibration information and both raw and post-processed data (such as the 3D point clouds for each dataset). The format of all the data files is fully documented and open-source applications are also published to facilitate their visualization, management and processing.

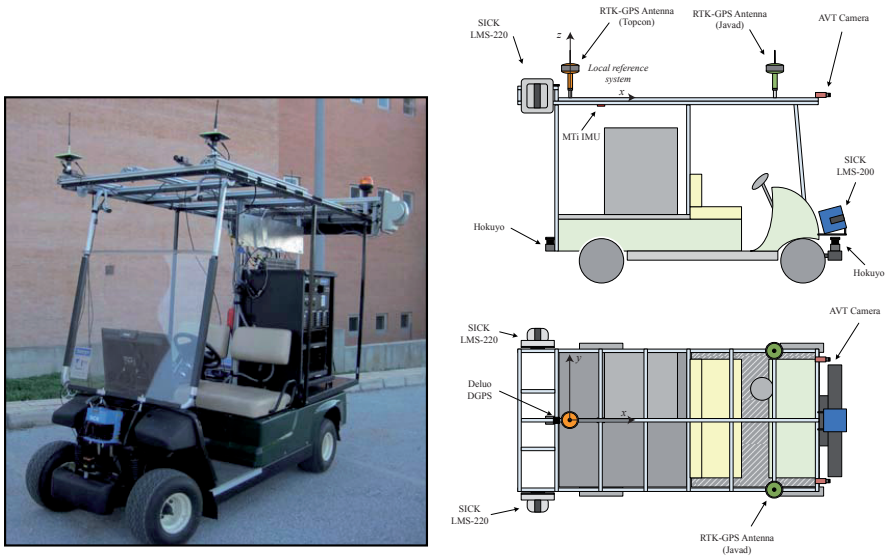
Secondly, along with our first dataset collection, we also present a methodology for obtaining a complete 6D centimeter-accuracy ground truth estimation, as well as its associated uncertainty bound. We also discuss innovative auto-calibration methods for some of the sensors, which virtually discard human errors in the manual measurement of the sensor positioning on the vehicle. Additionally, we also introduce a method to measure the consistence of the ground truth, which confirms the accuracy of our datasets.

To the best of our knowledge, there is no previous work with such an accurate ground truth in full 6D. Visual SLAM techniques are clearly the best candidates to be tested against the presented datasets, although our work may be also applicable to 2D SLAM techniques due to the existence of two horizontal laser scanners and the planar trajectories followed in some of the datasets.

On the other hand, our second dataset provides a unique combination of (i) multiple laser scanners pointing in various orientations and (ii) high-rate (20 fps) and high-resolution (1280×768) *stereo images* of good quality (e.g. minimal motion blur). Additionally, a significant part of our dataset reflects dynamic environments with real-life traffic, thus becoming a challenging testbed for SLAM, visual odometry and object detection methods.

5.2 Dataset #1: Dataset Collection with cm-accuracy Ground Truth

This section presents our first collection of outdoor datasets including the description of the employed vehicle, the specific sensors that are used for grabbing data and the dataset locations. The derivation of a proper ground truth from the GPS readings is also addressed here along with a calibration procedure for the involved sensors. Finally, a study of the uncertainty associated to the vehicle estimated pose and the validation of the constructed maps are presented.



(a) (b)

Figure 5.1: (a) The buggy-typed electric vehicle employed for grabbing the dataset and (b) a scheme of the sensors positions on the vehicle.

5.2.1 Vehicle description

In order to carry all the measurement devices for this first collection, we employed an electric buggy-typed vehicle (see figure 5.1a) that can navigate through different environments with the appropriate load and autonomy capabilities. The major benefit of using an electric vehicle, besides of the reduced environmental impact, is the avoidance of the inherent vibrations of common vehicles equipped with internal combustion engines.

The vehicle was modified and adapted for a more suitable arrangement of the sensors. Specifically, we designed and built a rigid top structure which allowed a robust and easy-to-modify assembly of most of the employed devices. In total, we placed twelve sensors on the vehicle of such heterogeneous types as laser scanners, inertial measurement units, GPS receivers (both consumer-grade DGPS and GPS-RTK) and color cameras.

Figure 5.1b shows a descriptive illustration with vehicle sensor positions, and table 5.2 provides a listing of sensors and their respective 6D pose in the vehicle local reference frame. These poses were initially measured by hand and, most of them, were subsequently optimized through different calibration methods, as will be explained in section 5.2.4. In the following, we describe each group of sensors.

Sensor	***x*** (m)	***y*** (m)	***z*** (m)	***yaw*** (deg)	***pitch*** (deg)	***roll*** (deg)
Rear GPS-RTK	0.000	0.000	0.132	×	×	×
Front Left GPS-RTK	1.729	0.5725	0.115	×	×	×
Front Right GPS-RTK	1.733	−0.5725	0.1280	×	×	×
DGPS*	−0.250	0.000	0.100	×	×	×
Left Camera	2.216	0.430	0.022	−88.43	−2.99	−87.23
Right Camera	2.200	−0.427	0.025	−90.31	−3.53	−86.19
Front Hokuyo*	2.420	0.000	−1.740	0.00	0.00	0.00
Rear Hokuyo*	−0.299	0.084	−1.725	178.81	0.00	0.00
Front SICK LMS-200*	2.278	0.000	−1.565	0.00	−6.84	0.00
Left SICK LMS-221	−0.3642	0.7899	0.0441	90.58	6.82	−89.66
Right SICK LMS-221	−0.3225	−0.8045	−0.0201	−90.33	−2.87	89.85
IMU*	×	×	×	0.00	0.00	0.00

**Sensor pose is not optimized. × (irrelevant or not applicable).*

Table 5.2: Summary of the sensor 6D poses.

Laser Scanners

The vehicle was equipped with three different types of laser scanners: two Hokuyo UTW-30LX, two SICK LMS-221 and, finally, a SICK LMS-200 (see figure 5.2).

The Hokuyo UTW-30LX[2] are small, lightweight outdoor laser scanners with a detection range from 0.1 to 30 m and a 270 deg. field of view. Per manufacturer's specifications the sensors provide an angular resolution of 0.25 deg. and achieve an accuracy of 30 mm for measurements up to 10 m and 50 mm for higher ranges. Data transmission between the Hokuyo devices and the PC is conveniently accomplished through a USB link, which makes the sensor, along with its lightness and reduced dimensions, quite suitable for mobile robotics applications.

The two Hokuyo laser scanners were located in the front and the rear of the vehicle, respectively, and were scanning the environment in a plane parallel to the ground.

On the other hand, SICK[3] laser scanners are heavy and robust devices widely employed in robotics and industrial applications. Their performance is sensibly higher than Hokuyo's and, furthermore, they are highly configurable. In our dataset collection, the LMS-200 was configured to measure up to a maximum range of 80 meters with an accuracy of 40 mm (as reported by the manufacturer), whereas the accuracy of the LMS-221 device is 5 mm in a measurement range from 0.8 to 32 meters. Both models provide an angular resolution of 0.5 deg. and a field of view of 180 deg., narrower than that of Hokuyo scanners. Finally, although both the Hokuyo and the LMS-221 devices are designed to operate outdoors, we found in our experiments that the former are more sensitive to sunlight and, therefore, prone to higher errors when operating outdoors.

The LMS-200 sensor was located at the front of the vehicle, scanning a plane slightly tilted upwards, whereas the LMS-221 laser scanners were placed at the sides

[2] `http://www.hokuyo-aut.jp/02sensor/07scanner/utm_30lx.html`

[3] `http://www.sick.com`

Hokuyo UTW-30LX | SICK LMS 200 | SICK LMS 221

Figure 5.2: The three laser scanner models employed for the datasets.

of the vehicle such as their scans become perpendicular to the ground plane. This layout allows the vehicle to scan objects located at its sides when navigating.

Cameras

Image grabbing was performed by means of two CCD color cameras (AVT Marlin F-131C model[4]), which can capture up to 1280x1024-size images at a maximum frame speed of 25 fps and transfer them to the PC via a standard FireWire link.

In order to properly adjust the devices according to light conditions, some of their features, such as *shutter*, *bright*, or *color mode* can be easily configured by software, while *focus* and *aperture* are manually controlled. The images within the presented datasets were captured at 7.5 fps with a dimension of 1024x768 pixels, being offered sets of both raw and rectified images for each dataset.

It is important to note that, to properly rectify the grabbed images, the cameras need to be calibrated, an issue which is addressed in section 5.2.4.

The two AVT cameras were mounted on the top structure of the vehicle with their optical axes pointing forward, as depicted in figure 5.1b.

Inertial measurement unit (IMU)

In this collection of datasets we employed a MTi device from xSens[5] (see figure 5.3) to provide a source of inertial raw measurements.

[4] `http://www.alliedvisiontec.com/avt-products/cameras/marlin.html`
[5] `http://www.xsens.com/en/products/machine_motion/mti.php?`

(a) (b)

Figure 5.3: (a) The MTi IMU model by xSens and (b) its mounting point on the vehicle.

The convention followed in this work for 3D orientations is to consider them as the sequence of rotations *yaw*, *pitch* and *roll* around the z, y and x axes, respectively, being positive angles performed counter-clockwise.

This device is a miniature, gyro-enhanced, 3-axis IMU, which is both powered and communicated through a USB link. It contains gyroscopes, accelerometers and magnetometers in 3D which are combined through an Extended Kalman Filter (EKF) to provide 3D orientation data at a maximum rate of 100 Hz.

According to the manufacturer specifications, the MTi device achieves a static accuracy of 0.5 deg. in attitude (both *pitch* and *roll*) and 1.0 deg. in heading (*yaw*) measurements, whereas in dynamic situations this performance degrades up to 2.0 deg. RMS. The angular resolution of this device is 0.05 deg., while the full scale of the on-board accelerometers and gyroscopes is 5g and 300 deg/s, respectively.

This sensor was also fixed to the top structure of the vehicle in order to move as a rigid solid with the GPS receivers (which define the local coordinates frame, as explained later). Note that the exact position of this sensor on the vehicle is not relevant when required to provide only orientation measurements.

GPS devices

Global Positioning System (GPS) has become the most reliable system for land surveying and vehicle positioning due to the high accuracy it can provide and the worldwide spreading of GPS receivers.

In short, the principle of operation of normal GPS systems consists of the trilateration of the distances between a mobile receiver (usually called *rover*) and a constellation of satellites, as illustrated in figure 5.4a. These satellites transmit microwaves signals containing a pseudo-random code (PRC) that is compared by the rover with a stored local copy. The existing delay between the received and the local code de-

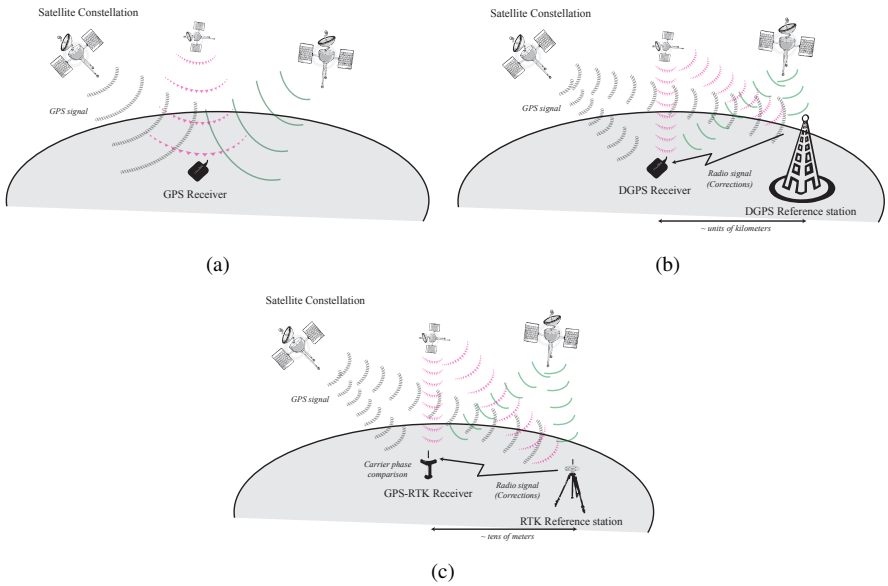


Figure 5.4: GPS scenarios. (a) Normal GPS operation, (b) DGPS operation and (c) Real-Time-Kinematics GPS operation.

termines the actual distances between the receiver and the satellites and, therefore, allows the mobile unit to pinpoint its position. However, normal GPS systems are prone to some error sources such as receiver clock inaccuracies or signal alteration due to atmospheric conditions, and, therefore, provide a relatively low accuracy for mobile robots localization (about 3 meters under ideal conditions).

Differential GPS (DGPS) technique overcomes some of the major limitations and error sources of the normal GPS system by setting a static reference station with a known fixed position which re-broadcast via radio the differential corrections according to the area conditions (refer to figure 5.4b). These corrections allow the rovers in the area to localize themselves with a higher accuracy (typically tens of centimeters). Normally, DGPS reference stations have large coverage areas (in the order of kilometers) to provide service to as many receivers as possible, but, in return, it involves a loss of accuracy proportional to the distance to the reference station.

Real-Time-Kinematics (RTK) represents an improvement to the DGPS technique which employs both the reference station corrections and the carrier phase of the GPS satellite signals to achieve up to a centimeter level of accuracy. Moreover, GPS-RTK systems usually arrange their own reference stations which can be placed much closer to the rovers than those employed for standard DGPS systems, as shown in figure 5.4c. In this state-of-the-art technique, the GPS receiver gages the distance to the satellites by adjusting the received and the local copy of the signal not only by comparing the PRC but also the phase of the signals, thus leading to a much more

Figure 5.5: The GPS devices on the vehicle. (a) Low cost Deluo DGPS, (b) Javad Maxor GPS-RTK and (c) Topcon GR-3 GPS-RTK.

fine delay estimation and, therefore, to an improved localization (ideally up to 1 cm of accuracy).

GPS-RTK devices can operate in two different modes: RTK-float and RTK-fixed. In the former, the RTK reference station has not enough information from the satellite constellation to precisely determinate its static position, leading to significant errors in rovers localization, while the latter is only attainable when the number of visible satellites and the quality of their signals allow a total disambiguation in its positioning.

We placed four different GPS receivers in our vehicle: one consumer-grade DGPS device and three RTK-GPS (two Javad Maxor and one Topcon GR-3) which constitute a solid frame for establishing a reliable 6D ground truth estimation (see figure 5.5). The two Maxor-GGDT devices from Javad[6] are dual frequency GPS receivers which provide RTK-accuracy-level measurements at a rate of 4 Hz through a RS-232 link. These receivers are associated to a pair of MarAnt+ antennas which were mounted, as separated as possible, at the front of the vehicle top structure, as can be seen in figure 5.1b. On the other hand, the Topcon GR-3[7] device is a robust, state-of-the-art positioning receiver which can combine signals from the three existing satellite constellations: American GPS, Russian GLONASS and European Galileo, thus ensuring an optimal coverage 24 h a day achieving a high performance. Positioning data is measured and transmitted to the PC at a rate of 1 Hz. This device was placed at the rear of the top structure.

It is important to remark that Javad and Topcon devices are connected to different reference stations as they are configured to receive RTK corrections from their own stations through different radio channels. This caused the positioning readings to be

[6] `http://www.javad.com/jns/index.html`
[7] `http://www.topconeurope.com`

biased with constant offsets depending on the specific measuring device. This offset can be estimated through a least squares-error minimization algorithm, which will be explained in section 5.2.3.

Finally, the Deluo[8] DGPS is a low-cost, consumer-grade, compact and portable receiver extensively employed for extending PDAs and laptops with self-localization capabilities. Its performance and accuracy are sensibly poorer than in the RTK-GPS sensors but its measurements may be suitable for testing and developing rough localization applications which are intended to be accessible to wider markets. For example, it could be taken into account together with the cameras to test large-scale SLAM approaches. Section 5.2.4 presents a brief evaluation of the Deluo DGPS performance.

5.2.2 Data collection

This section describes the placement of the six presented datasets. Three of them were collected at the parking of the Computer Science School building while the other three were located at the Campus boulevard of the University of Málaga, as depicted in figure 5.6.

From now on, and for clarity purposes, the datasets will be denoted by the combination of a prefix, identifying its corresponding emplacement (PARKING or CAMPUS) and two-letters mnemonic indicating a distinctive feature that makes it easily identifiable. In five of the six datasets, this mnemonic denotes the number of complete loops performed by the vehicle (e.g. 0L for zero loops) whereas in the remaining one (CAMPUS-RT), it indicates that the trajectory is a round-trip. Finally, please note that we have considered the starting point of the dataset PARKING-0L as the origin of the Cartesian reference system for all the datasets in this collection, as can also be seen in figure 5.6.

Next, we describe more extensively the most relevant characteristics of the two groups of datasets.

Parking datasets

The three *parking* datasets are characterized for almost planar paths through a scenario which mostly contains trees and cars. These datasets present more loops than the *campus* ones, which makes them specially interesting for testing SLAM approaches.

The lack of planar surfaces, such as buildings, and the presence of dynamic objects such as moving cars and crossing people, make this emplacement particularly interesting for computer vision applications (e.g. segmentation or object visual tracking). Finally, the presence in this group of datasets of nearby objects in the images, facilitates the estimation of the camera displacement in common vision-based solutions while the detection of far objects leads to a better estimation of the rotation [24, 30].

[8] `http://www.deluogps.com`

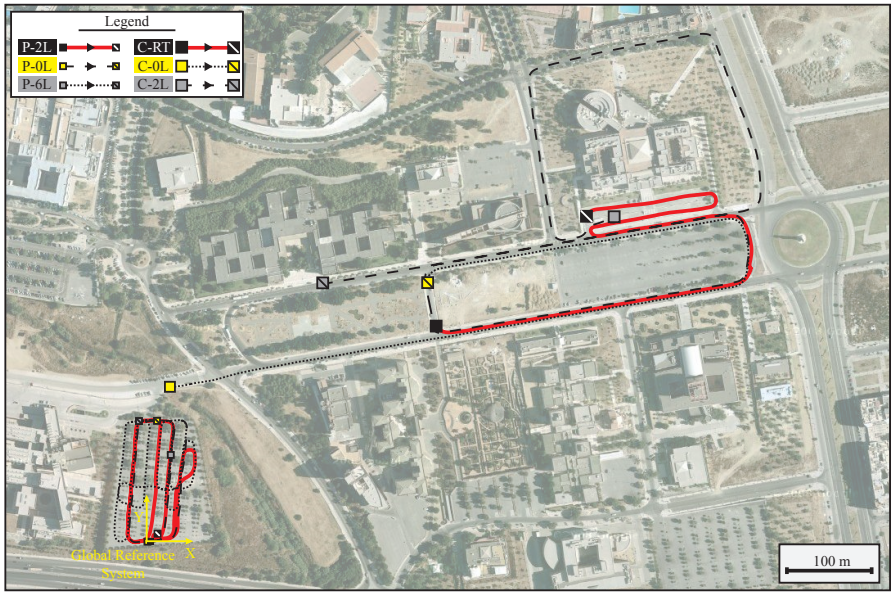


Figure 5.6: Vehicle trajectories for the six grabbed datasets, where P and C stand for PARKING and CAMPUS, respectively. The global reference system represents the origin in our transformed Cartesian coordinate system for all the datasets.

Figures from 5.7 to 5.9 contain visual descriptions of the *parking* datasets, showing the 3D point clouds (subfigures (a), (c) and (e)) generated through the projection of the vertical laser scans from their estimated pose at each time step. The methodology for the estimation of the sensors trajectories will be described later on in section 5.2.5. On the other hand, subfigures (b) illustrate aerial views of the whole vehicle trajectory where it has been highlighted the start and end points of the paths. Finally, we depict in subfigures (d) a summary of the operation modes for the three GPS-RTK receivers during the whole data collection process (please, refer to section 5.2.1 for a descriptive review of GPS operation modes). Notice that we can estimate a reliable ground truth only when the three devices are simultaneously operating in RTK-fixed mode, thereby achieving the highest accuracy of the 6D pose vehicle estimation.

Campus datasets

In general, the *campus* datasets include fairly large loops, specially suitable for validating large-scale SLAM algorithms. Besides, they also contain long straight trajectories which may be a proper testbed for visual odometry methods. Approaches for other vision-based applications such as automatic detection of overtaking cars or dynamic objects tracking are also good candidates for being evaluated by employing these datasets.

	Length	LC	vSLAM	2D SLAM	vOdometry	D.Obj.Seg.
PARKING-0L	524 m	×	✓	✓	✓	×
PARKING-2L	543 m	✓	✓	✓	✓	×
PARKING-6L	1222 m	✓	✓	✓	✓	✓
CAMPUS-0L	1143 m	×	✓	×	✓	✓
CAMPUS-2L	2150 m	✓	✓	×	✓	✓
CAMPUS-RT	776 m	×	✓	×	✓	✓

Table 5.3: Datasets lengths and recommended applications. **Legend**: LC (*Loop Closure*), vSLAM (*Visual SLAM*), vOdometry (*Visual Odometry*), D.Obj.Seg. (*Dynamic Objects Segmentation*)

Their visual descriptions are shown in figures from 5.10 to 5.12 following an identical structure as the one explained above.

Summary of dataset applications

Table 5.3 presents a summary of some datasets properties and an evaluation of their suitability for being employed in a set of common robotics and computer vision applications. This should be understood as a recommendation, related to the own special characteristics of each dataset as, for example, the number of relevant sensors, the planar nature of the movement, or the presence of loop closures.

Software

The vehicle is equipped with sensors of quite distinct types, each generating data at different rates. Thus, the software intended to grab the data logs must be capable of dealing with asynchronous sensor data. For this purpose, our data log recording application, named *rawlog-grabber*, creates one thread for each individual sensor on the robot. Each thread collects the data flow from its corresponding sensor, and converts it into discrete entities, or *observations*. Distinct kinds of sensors produce different observation objects.

At a predetermined rate (1 Hz in our case), the main thread of the logger application collects the incoming observations from all sensors, which are then sorted by their timestamps and dumped into a compressed binary file. Documentation about the format of these files, as well as the source-code of the grabber and some data viewer applications are published as part of the Mobile Robot Programming Toolkit (MRPT) [12].

5.2.3 Derivation of the *ground truth*

In this section we address one of the major contributions of this work: a detailed description of how to compute a ground truth for the 6D path of the vehicle from the GPS readings. Section 5.2.5 will derive a bound for the uncertainty of this reconstructed path.

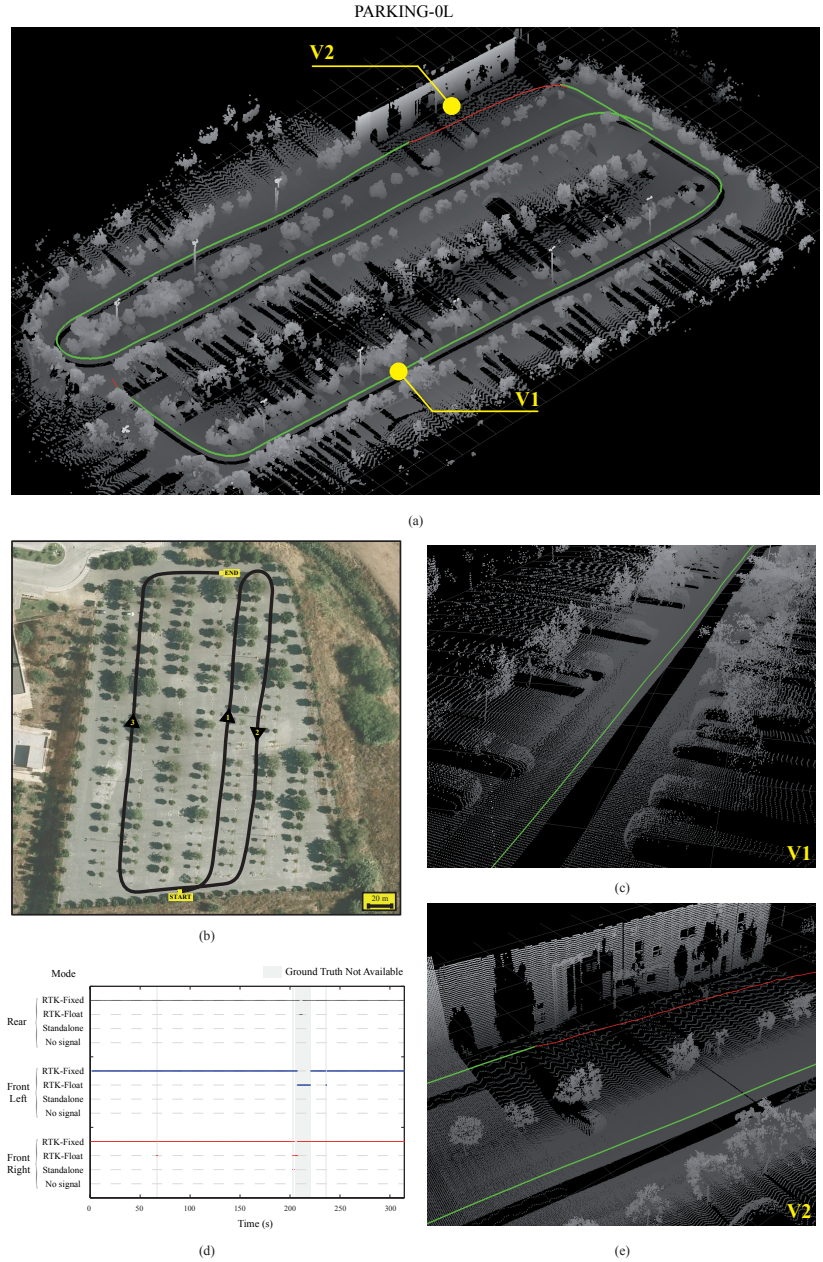


Figure 5.7: PARKING-0L dataset. (a) 3D projected points for the lateral laser scanners and vehicle path. (c,e) Zoom of zones V1 and V2. (b) Top-view of the vehicle path and (d) status (modes) of the 3 GPS devices during the experiment.

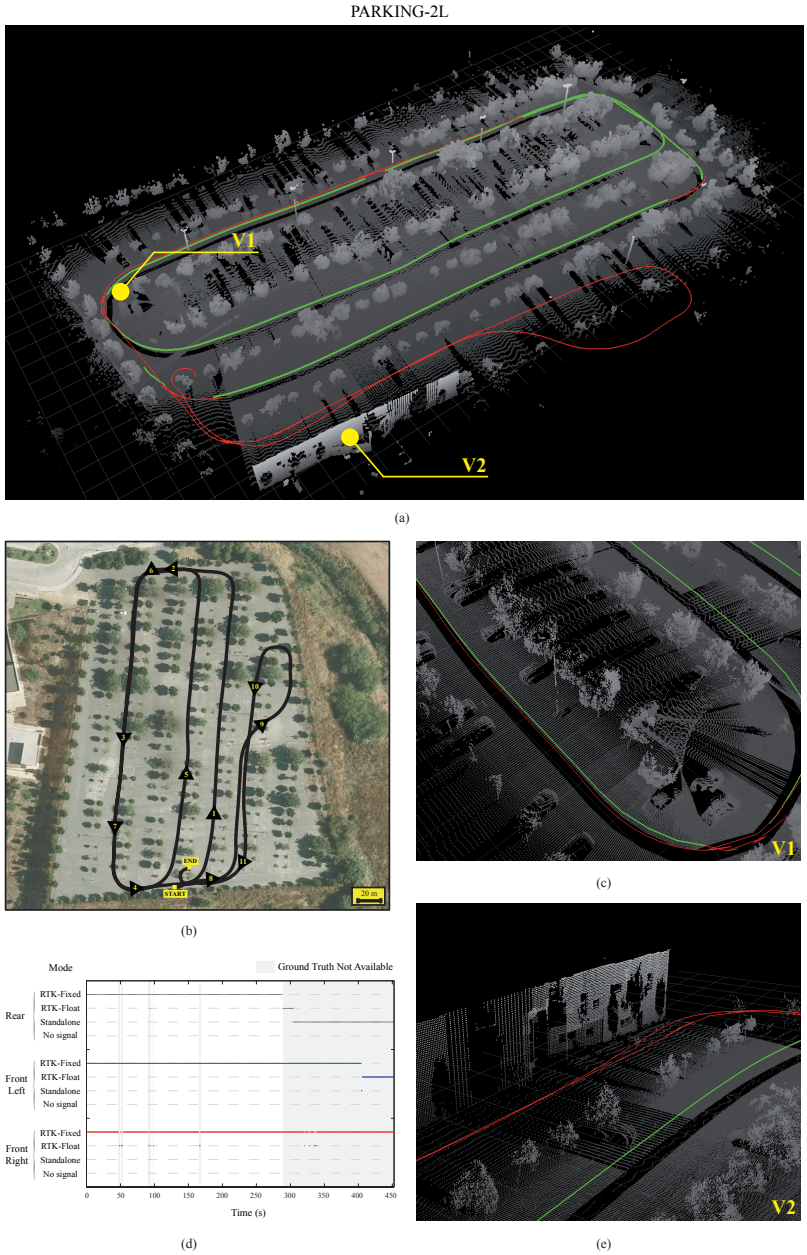


Figure 5.8: PARKING-2L dataset. (a) 3D projected points for the lateral laser scanners and vehicle path. (c,e) Zoom of zones V1 and V2. (b) Top-view of the vehicle path and (d) status (modes) of the 3 GPS devices during the experiment.

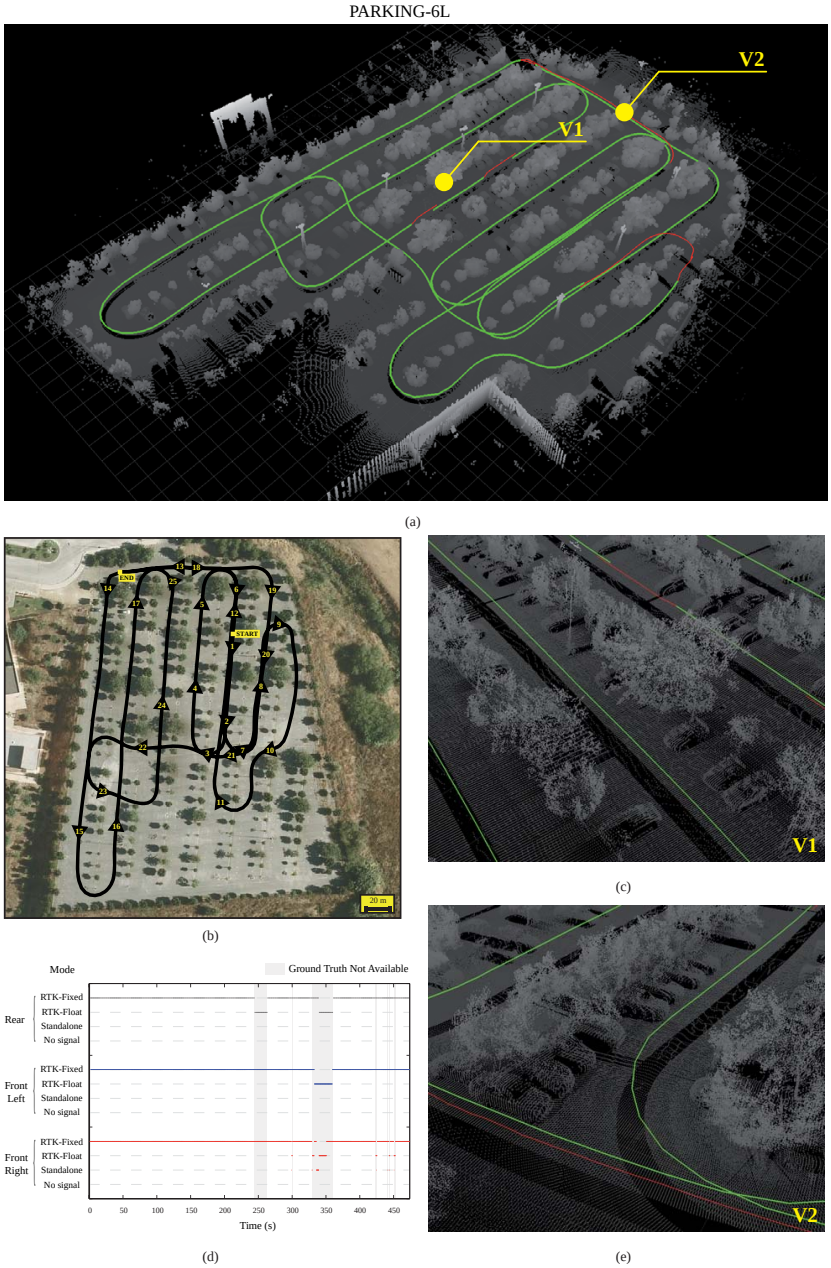


Figure 5.9: PARKING-6L dataset. (a) 3D projected points for the lateral laser scanners and vehicle path. (c,e) Zoom of zones V1 and V2. (b) Top-view of the vehicle path and (d) status (modes) of the 3 GPS devices during the experiment.

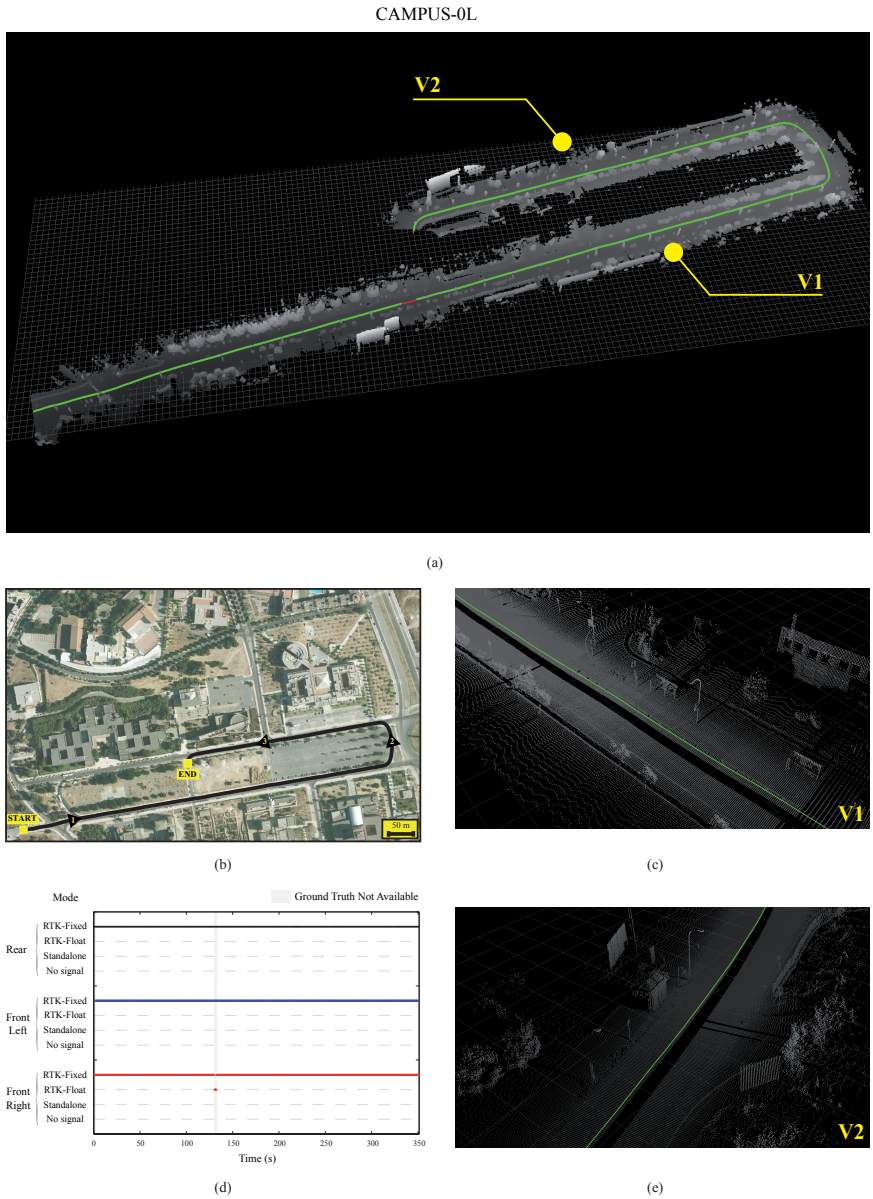


Figure 5.10: CAMPUS-0L dataset. (a) 3D projected points for the lateral laser scanners and vehicle path. (c,e) Zoom of zones V1 and V2. (b) Top-view of the vehicle path and (d) status (modes) of the 3 GPS devices during the experiment.

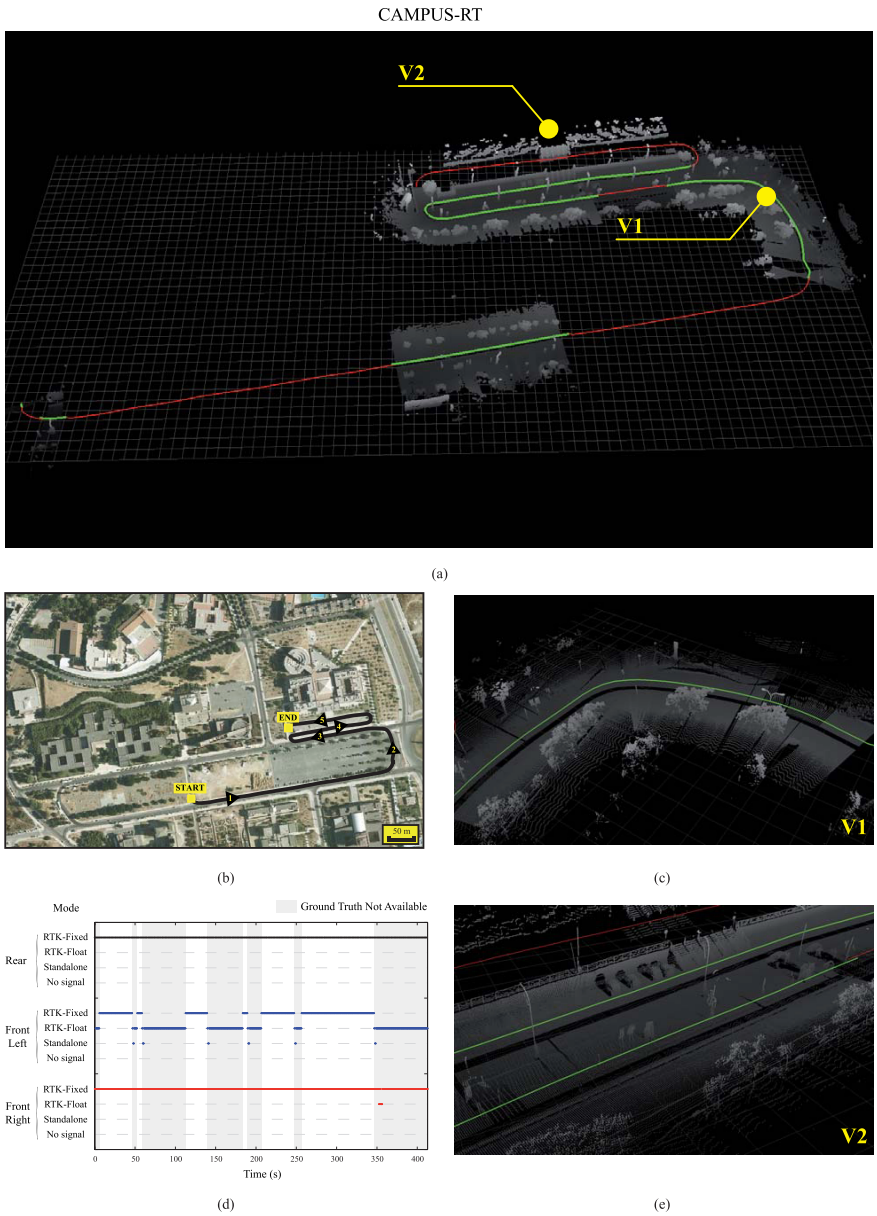


Figure 5.11: CAMPUS-RT dataset. (a) 3D projected points for the lateral laser scanners and vehicle path. (c,e) Zoom of zones V1 and V2. (b) Top-view of the vehicle path and (d) status (modes) of the 3 GPS devices during the experiment.

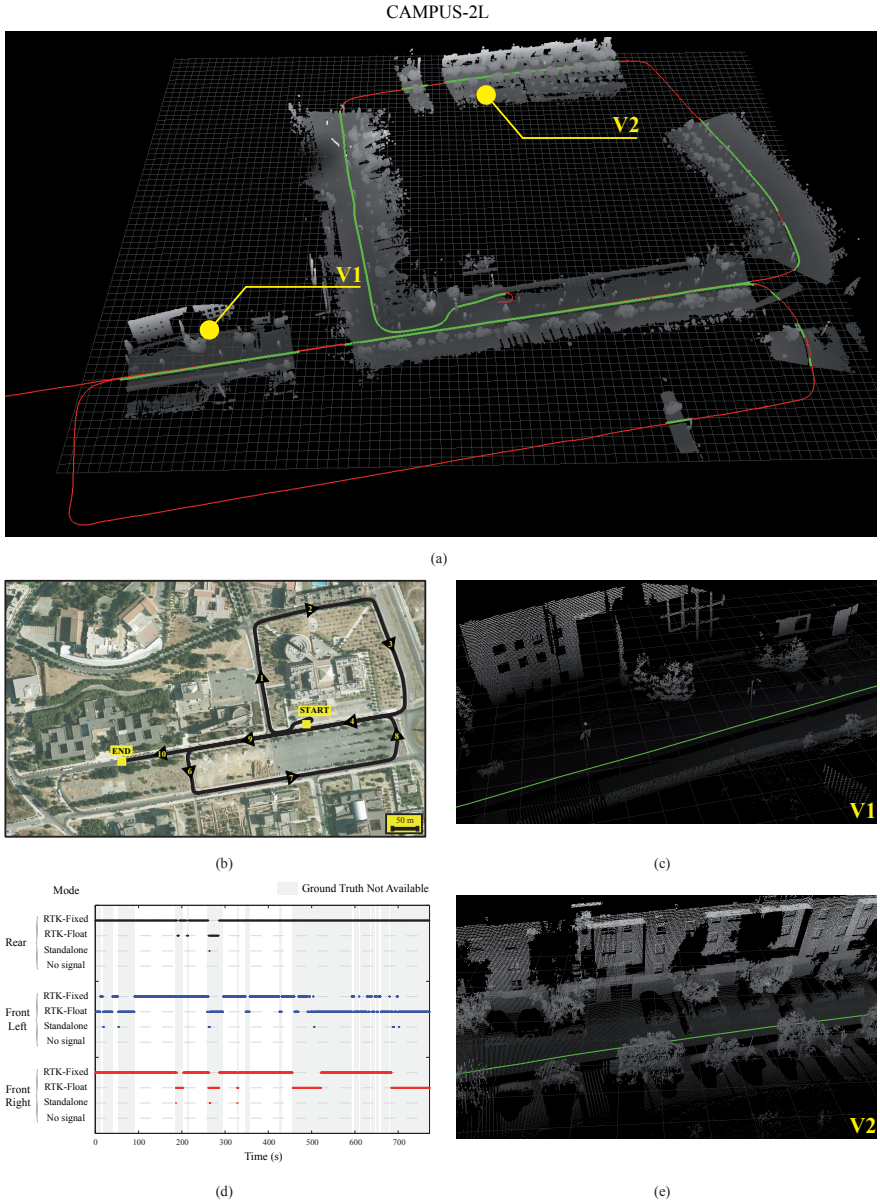


Figure 5.12: CAMPUS-2L dataset. (a) 3D projected points for the lateral laser scanners and vehicle path. (c,e) Zoom of zones V1 and V2. (b) Top-view of the vehicle path and (d) status (modes) of the 3 GPS devices during the experiment.

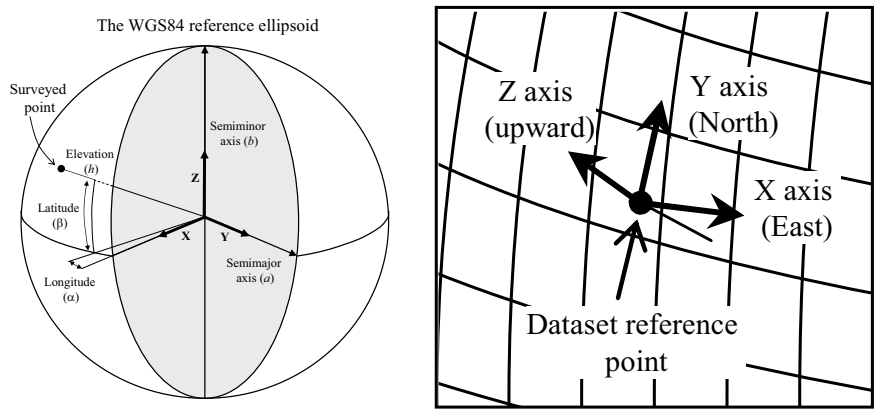


Figure 5.13: (a) A schematic representation of the variables involved in the WGS84 reference ellipsoid used for GPS localization. We define a more convenient local Cartesian coordinate system with the origin at an arbitrary location, as shown in (b).

Coordinates transformation

We will focus first on the problem of tracking the position of a *single* GPS receiver, that is, we are interested in the 3D coordinates (x,y,z) of a particular point. In our situation, these points are the phase-centers of the GPS antennas (see figure 5.1).

GPS receivers provide datums through three parameters: *longitude*, *latitude* and *elevation*. To fully exploit the precision of RTK receivers, these coordinates must be interpreted exactly using the same system the GPS network uses, the World Geodetic System (WGS)-84 reference ellipsoid. This coordinate framework has been optimized such as its center matches the Earth center of mass as accurately as possible. We briefly explain next the meaning of the three coordinates in WGS-84.

The *longitude* datum states the angular distance from an arbitrarily defined meridian, the International Reference Meridian, just a few arc-seconds off the prior Greenwich's observatory reference. The *geodetic latitude* is the other angular coordinate, which measures the angle between the equatorial plane and the normal of the ellipsoid at the measured point. Note that this is not exactly equal to the angle for the line passing through the Earth center and the point of interest (the *geocentric* latitude [20]). Finally, the *elevation* represents the height over the reference geoid.

We now review the equations required to reconstruct local Cartesian coordinates, the natural reference system employed in SLAM research, from a sequence of GPS-collected data. Let D_i be the datums of the i-th GPS reading, comprised of longitude α_i, latitude β_i and elevation h_i:

$$D_i = \begin{bmatrix} \alpha_i \\ \beta_i \\ h_i \end{bmatrix} \tag{5.1}$$

The geocentric Cartesian coordinates of this point, G_i, can be obtained by [112]:

$$G_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} (N_i + h_i)\cos\beta_i \cos\alpha_i \\ (N_i + h_i)\cos\beta_i \sin\alpha_i \\ (N_i \cos^2 æ + h_i)\sin\beta_i \end{bmatrix} \tag{5.2}$$

with the radius of curvature N_i computed from the semi-major axis a and the angular eccentricity æ as:

$$N_i = \frac{a}{\sqrt{1 - \sin^2 æ \sin^2 \beta_i}} \tag{5.3}$$

At this point we have assigned each GPS readings a 3D location in rectangular coordinates. However, these coordinates are of little practical utility due to two problems: the large scale of all the distances (the origin is the center of the Earth), and the orientation of the XYZ axes – refer to figure 5.13a. We would rather desire a coordinate transformation where coordinates are local to a known point in the environment and the axes have a more convenient orientation, with the XY plane being horizontal and Z pointing upward, as illustrated in figure 5.13b. This kind of reference system is called ENU (East-North-Up) and is commonly used in tracking applications, as opposed to the so far described Earth-Centered Earth-Fixed (ECEF) system [112].

Unlike other previous methods such as [50], our change of coordinates is not an approximation but an exact simple rigid transformation, i.e. computed accurately without approximations.

This change of coordinates can be described as the mapping of a 3D point G_i into local ENU coordinates L_i relative to another point R, with rotation represented by the three new orthogonal base vectors u (East), v (North) and w (up)[9]. Mathematically, the operation can be written down using homogeneous matrices as:

$$\begin{aligned} L_i &= \left[\begin{array}{ccc|c} u & v & w & R \\ \hline 0 & 0 & 0 & 1 \end{array}\right]^{-1} G_i \\ &= \left[\begin{array}{c|c} u^\top & -u^\top R \\ v^\top & -v^\top R \\ w^\top & -w^\top R \\ \hline 0 & 1 \end{array}\right] G_i \end{aligned} \tag{5.4}$$

However, we have found that a direct implementation of the equation above suffers of unacceptable inaccuracies due to numerical rounding errors with standard 64-bit floating point numbers. These errors arise in the multiplications of numbers in a

[9] In the published datasets, coordinates use an *up* vector that follows the same direction than the line passing through the Earth center and the reference point. In the literature, this vector is often defined as perpendicular to the ellipsoid (very close but not exactly equal to our *up* vector). Thus, strictly speaking, ours are not ENU coordinates but a slightly rotated version. Nevertheless, this does not affect at all the accuracy of the obtained coordinates.

	Campus Datasets			Parking Datasets		
	Optimal	d_{ij} **RMSE**	d_{ij}^2 **RMSE**	**Optimal**	d_{ij} **RMSE**	d_{ij}^2 **RMSE**
$d_{1,2}$	1.8226 m	1.16cm	4.23510^{-2}	1.8208 m	1.02cm	3.71110^{-2}
$d_{1,3}$	1.8255 m	0.89cm	3.32610^{-2}	1.8247 m	0.71cm	2.61110^{-2}
$d_{2,3}$	1.1444 m	0.81cm	1.84810^{-2}	1.1457 m	0.96cm	2.18910^{-2}
	Optimal RTK offsets $(\Delta_x^i, \Delta_y^i, \Delta_z^i)$ **(meters)**					
Δ^1	0,0,0			0,0,0		
Δ^2	-145.2875, 28.6745, 2.3344			-260.168, 23.8158, -1.64305		
Δ^3	-145.2914, 28.6712, 1.2915			-260.169, 23.8075, -2.74191		

Table 5.4: Results of the least squares optimizations of GPS parameters.

wide dynamic range, i.e. the orthogonal vectors (in the range [0,1]) and the geocentric coordinates (with a order of 10^6).

As a workaround, we propose the following rearrangement of the transformation:

$$L_i = \left[\begin{array}{c|c} u^\top & 0 \\ v^\top & 0 \\ w^\top & 0 \\ \hline 0 & 1 \end{array}\right] (G_i - R) \tag{5.5}$$

which can be easily derived from equation (5.4) but avoids the numerical inaccuracies. Finally, the geodetic coordinates of the reference point used in all the datasets presented in this work can be found next (see also its representation in the map of figure 5.6):

Longitude: -4.4789588283 deg.
Latitude: 36.7144590750 deg.
Elevation: 38.8887 m.

Compensation of RTK offsets

At this point we can compute the trajectories of the three RTK GPS receivers. However, as mentioned in section 5.2.1, the fact that RTK corrections are taken from different base stations introduces constant offsets in the locations of each device. A part of these offsets is also due to inaccuracies in the initial positioning of the RTK base stations during the system setup.

Let $\hat{P}_t^i$ denote the local Cartesian coordinates of the i-th GPS receiver, computed as described in the previous section. Our goal is to obtain the corrected coordinates P_t^i:

$$P_t^i\left(\Delta^i\right) = \left[\begin{array}{c} x_t^i \\ y_t^i \\ z_t^i \end{array}\right] = \hat{P}_t^i + \Delta^i = \left[\begin{array}{c} \hat{x}_t^i \\ \hat{y}_t^i \\ \hat{z}_t^i \end{array}\right] + \left[\begin{array}{c} \Delta_x^i \\ \Delta_y^i \\ \Delta_z^i \end{array}\right] \tag{5.6}$$

by means of the offset vectors which are different for each receiver i but constant with time. Let $\mathbf{\Delta}$ denote the concatenation of all these vectors, such as

$$\mathbf{\Delta} = \begin{bmatrix} \Delta^1 \\ \Delta^2 \\ \dots \\ \Delta^i \end{bmatrix} \tag{5.7}$$

We show next how these parameters can be determined automatically *without any further a-priori known data, measurements or approximations*. This is a crucial point supporting the quality of our subsequently derived ground truth, since our method is insensitive to errors in any manually acquired (i.e. noisy) measurement.

The basis of this automatic calibration procedure is that the three GPS receivers move as a single rigid body, that is, they are robustly attached to the vehicle (refer to figure 5.1). From this follows that the distances between GPS receivers (or *inter-GPS distances*) must be constant with time, which allows us to set up the determination of $\mathbf{\Delta}$ as the following least square optimization problem:

$$\begin{aligned} \mathbf{\Delta}^\star = & \quad \arg\min_{\mathbf{\Delta}} E(\mathbf{\Delta}, \mathbf{D}) \\ E(\mathbf{\Delta}, \mathbf{D}) = & \quad \sum_{(i,j)} \sum_{t} \left(\left| P_t^i(\Delta^i) - P_t^j(\Delta^j) \right| - d_{ij} \right)^2 \end{aligned} \tag{5.8}$$

where (i, j) represents all the possible unique pairs of GPS devices[10], and $\mathbf{D} = \{d_{ij}\}$ are the real distances between those pairs. Notice that those distances d_{ij} can be determined by measuring them manually, what for our vehicle gives us:

$$\begin{aligned} d_{rear,front-left} = & \quad 1.79 \text{ meters} \\ d_{rear,front-right} = & \quad 1.79 \text{ meters} \\ d_{front-left,front-right} = & \quad 1.15 \text{ meters} \end{aligned}$$

Obviously, small errors must certainly be present in these values due to the limited accuracy of any practical method for manual measuring. In order to make our calibration independent of those errors, the set of optimal inter-GPS distances $\mathbf{D}^\star$ is determined as the result of another optimization problem, taking the values above as the initial guess for the optimization:

$$\mathbf{D}^\star = \arg\min_{\mathbf{D}} \left(\min_{\mathbf{\Delta}} E(\mathbf{\Delta}, \mathbf{D}) \right) \tag{5.9}$$

We must remark that this is a nested optimization problem, where the inner optimization of $\mathbf{\Delta}$ was stated by equation (5.8). Put in words, our approach to the automatic determination of the offsets $\mathbf{\Delta}$ and inter-GPS distances $\mathbf{D}$ consists of an iterative optimization of the inter-GPS distances, where the error function being optimized is

[10] In our specific case of three receivers, the possible values are (1, 2), (1, 3) and (2, 3), where the indices 1, 2, and 3 correspond to the rear, front-left and front-right receivers, respectively.

the residual error of another optimization of the RTK offsets, maintaining the distances fixed.

The optimization has been implemented as two nested instances of the Levenberg-Marquardt algorithm [121, 131]. The method exhibits an excellent robustness, in the sense that it converges to the optimal solution starting from virtually any set of initial values for Δ. Regarding the execution time, our custom C++ implementation takes about 2 minutes to optimize the data from a representative sample of 600 time-steps.

Since the six datasets, described in section 5.2.2, were collected during two days with the RTK base stations installed at different locations each day, the RTK offset parameter $\mathbf{\Delta}$ has different values in the *campus* datasets than in the *parking* datasets. On the other hand, the inter-GPS distances $\mathbf{D}$ should remain constant throughout the different datasets.

The optimization results are summarized in table 5.4: the average difference between the values of $\mathbf{D}$ in the two independent optimizations is 1.3 mm, i.e. in practice both estimates converge to the same point. This is a clear indication of the accuracy of all the 3D locations given by the sensors, an issue which is quantified later on in this work. We must highlight again that our method has precisely estimated these distances and the RTK offsets (in the order of hundreds of meters) by simply relying on the rigid body assumption of the GPS receivers.

Computation of the vehicle trajectory

As a result of the previous section, we now have an accurate 3D reconstruction of the path followed by each GPS receiver on the vehicle, namely the sequences P_t^i for the three GPS sensor $i = 1,2,3$ and for the sequence of time-steps t.

At this point it must be pointed out that the timing reference used by each GPS receiver is very accurately synchronized to the others since they are all synchronized with the satellite signal. Therefore, when the timestamp of datum readings from different receivers coincide we can assume that all the GPS locations were measured exactly at the same time, with timing errors negligible for the purposes of this work.

The synchronization of the data is a prerequisite for our goal of computing the 6D ground truth of the vehicle path, since three simultaneous 3D measurements unequivocally determine the complete 6D pose of a rigid body, in our case the vehicle. Therefore, ground truth is available when the data from all the three receivers coincide in time. As illustrated in figure 5.14, this happens with a rate of 1 Hz in our system.

Regarding the computation of the sequence of 6D vehicle poses v_t from the GPS locations P_t^i, we can set up the following optimization problem which gives the optimal solution at each timestamp t:

$$\hat{v}_t = \arg\min_v \sum_i \left(P_t^i - [v \oplus p_i] \right)^2 \tag{5.10}$$

with p_i stating the location of each receiver on the vehicle local coordinate framework (refer to figure 5.1), and $\oplus$ being the pose composition operator [182]. This

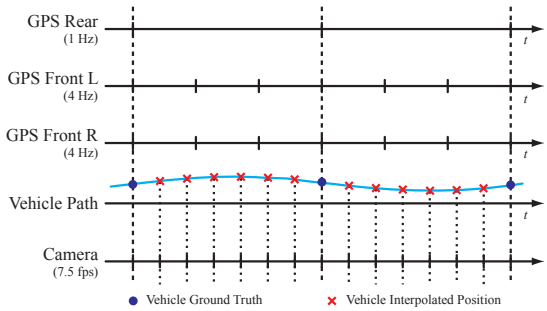


Figure 5.14: Sensor rates and vehicle path interpolation. Vertical ticks represent the timestamps at which the different sensors give their readings. In the case of the GPS receivers, the rear device works at 1 Hz while both the front left and right devices have a frequency of 4 Hz. Thus the ground truth for the vehicle pose is obtained at 1 Hz, the rate at which all three sensors coincide. Since other sensors (in this example, a camera) operate at different rates we need to interpolate the 6D ground truth to assign a valid pose to each sensor reading.

can be regarded as a problem of matching three pairs of corresponding 3D points. A widely employed solution to this generic problem is the Iterative Closest Point (ICP) algorithm [11]. In turn, we propose to apply the closed-form solution derived by Horn in [86] since in our case there are no uncertain data associations (the main issue solved by ICP). Other interesting characteristics of [86] are its closed-form nature and that providing an initial guess of the solution is unnecessary.

One advantage of stating the problem of recovering the vehicle path as in equation (5.10) is that GPS sensors can be freely positioned at arbitrary locations on the vehicle, e.g. there is no assumption about they being disposed in orthogonal directions. The process to automatically refine the local coordinates p_i is discussed later on in section 5.2.4. We must remark that a minimum of three GPS devices are required to obtain instantaneous ground truth measurements, but our method can trivially incorporate any larger number of devices with the purpose of reducing the overall reconstruction error.

Interpolation of the 6D vehicle trajectory

The process described in section 5.2.3 gives us the desired ground truth for the vehicle trajectory, one of the main goals of this work.

However, there are two tasks that force us to interpolate the 1 Hz ground truth. Firstly, vision SLAM methods usually reconstruct the 6D path of the *cameras*, not the *vehicle*. Therefore, in order to measure the quality of a given SLAM method under evaluation, a ground truth for the path of the cameras (or in general any other sensor, e.g. laser scanners) must be available. This is why the vehicle path must be interpolated, since in general the timestamps of other sensors will not coincide with those of the GPS readings, as shown in figure 5.14. Secondly, our goal of building

accurate point clouds from the laser scanner data also requires the precise locations of the scanners with time, again requiring the vehicle pose at instants of time that require interpolation.

A priori, a potentially promising technique for carrying on this interpolation would be spline fit [42], which results in smooth interpolated curves with continuous derivatives that exactly pass through all the input points to interpolate.

The consequences of this last property of splines in our specific case require a closer examination. Each of the six dimensions of the vehicle pose, 3D for Cartesian coordinates and 3D for the angles, follows characteristic and distinctive variations with time. For example, the coordinates of the planar locations (x and y) are specially suited for being approximated by splines since the vehicle movement over the road tends to describe smooth arcs. On the other hand, the pitch and roll angles are typically around zero and disregarding the small part produced by real rocking of the vehicle, variations are caused by the noise (errors) in the 3D coordinates of the RTK receivers. In those cases, splines typically magnify this error making it an unacceptable choice.

In our datasets, we finally chose to interpolate the vehicle x and y coordinates using splines, and the rest of dimensions using a least square linear fit of the four closer ground truth points. Once the pose of the vehicle can be estimated at any arbitrary instant t, the sensor pose is obtained by 6D pose composition with the local sensor location (see table 5.2), for example, using homogeneous coordinates. This method has led to accurate reconstructed paths, as can be visually verified with the maps discussed in section 5.2.6. In section 5.2.5 we go back to the issue of sensor paths and their uncertainty.

5.2.4 Calibration of sensors

The 6D positioning of each sensor in vehicle local coordinates is as important as the ground truth of the vehicle trajectory itself. By means of carefully manual measuring we can obtain a first estimate of these parameters associated to every sensor on-board. The purpose of this section is to discuss automatic calibration methods for some of the sensors used in the datasets.

It must be noted the hierarchical nature of our calibration methodology: first of all, the inter-GPS distances were automatically calibrated through the least square problem stated in equation 5.9. As explained in section 5.2.3, that solution is free of human measuring errors. In the subsequent sections these optimal inter-GPS distances $\mathbf{D}^\star$ will be used to first refine the Cartesian coordinates of the GPS receivers. Next, those coordinates are the basis for an accurate determination of the vehicle path ground truth, which in turn is indirectly used later to optimize the positions of the vertical laser scanners and the cameras. The final result of all this process can be seen in table 5.2.

Positioning of RTK-GPS devices

Let p_i be the local coordinates of each GPS receiver for $i = 1,2,3$, as already employed in equation (5.10). Since we already know the optimal inter-GPS distances $\mathbf{D}$ (see table 5.4), the optimal set of locations is that one fulfilling

$$\{p_i^\star\} = \arg\min_{\{p_i\}} \sum_{(i,j)} \left(\left| p_i - p_j \right| - d_{ij} \right)^2 \tag{5.11}$$

Before solving the above least square problem, additional constraints must be set since the system is under-determined. We arbitrarily set the position of the rear device (index $i = 1$) to be at the origin of the XY plane, that is:

$$p_1 = \begin{bmatrix} 0 \\ 0 \\ 0.132 \end{bmatrix} \tag{5.12}$$

with the z coordinate conveniently chosen such as the XY plane coincides with the top structure of our vehicle (see figure 5.1). There is still an additional degree-of-freedom in the form of a radial symmetry around the Z axis. It can be avoided by arbitrarily imposing that the x axis must exactly bisect the two front GPS devices (with indexes $i = 2$ and $i = 3$, respectively), which agrees with our coordinate system with the x axis pointing forward, as seen in figure 5.1b.

Under these constrains the optimization can be successfully solved, obtaining the sensor coordinates shown in table 5.2.

DGPS characterization

In this section we provide an empirically estimated bound for the Deluo DGPS error.

This is accomplished by batch processing the positioning measurements collected in all the six presented datasets and measuring the root mean square error (RMSE) between the DGPS readings and our ground truth estimation in the time steps where both are available and also synchronized. In the comparison, we considered data from both standalone and DGPS operation modes for the Deluo sensor.

The results yield quite similar RMSE values for the vertical (z coordinate) and horizontal (a combination of x and y coordinates) errors: 13.729 m and 13.762 m, respectively. These figures can be put in contrast with our bounded uncertainty of the ground truth estimate, in the order of $\sim$1 cm, thus clearly illustrating the different performance between DGPS and RTK devices.

Positioning of vertical laser scanners

We have devised an algorithm to automatically calibrate the location of the two vertical SICK LMS-221 laser scanners. To illustrate the underlying idea, please consider the example in figure 5.15a, where the ground truth of the vehicle path (the continuous line) and a first rough estimate of the left vertical laser location on the vehicle are

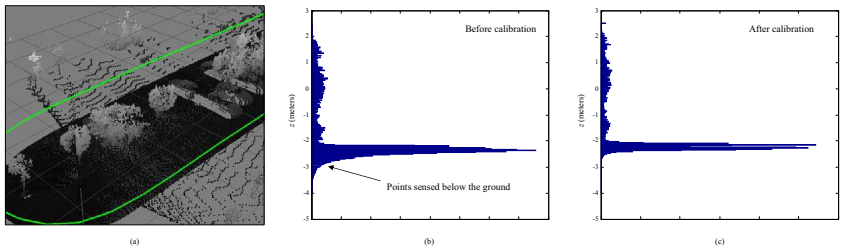


Figure 5.15: Calibration results. (a) An example of the 3D point clouds used to calibrate the on-board location of vertical laser scanners. (b) For the first manually measured location, clear slope errors can be observed in the reconstructed ground, as indicated by the histogram with several values up to 1 meter below the actual ground plane (around -2.2 m). (c) After calibrating the sensor location, the reconstructed 3D points are more consistent with the real scenario.

enough to build the 3D point cloud also shown in the figure. Our algorithm relies on the realization that the *optimal* sensor localization is that one maximizing the matching between the point clouds generated each time the vehicle passes by the same area. For instance, it can be seen how in figure 5.15a the vehicle first scans the central area while moving in one direction, then turns and scans it again from the opposite direction.

By identifying a dozen of segments from all the datasets fulfilling this requisite, a Levenberg-Marquardt optimization has been applied to each laser scanner giving us the poses shown in table 5.2. The optimized values only differ from those measured manually by a few centimeters and less than 7 degrees in the angles.

To quantify the improvement due to the optimized sensor poses, we have computed the histograms of the z coordinate in two point clouds, both with manually measured and optimized coordinates, in figure 5.15b–c, respectively. Given that the coordinate system reference is on the top structure of our vehicle (see figure 5.1) and its approximate height is 2.2 m to the ground, a distinctive peak can be expected at -2.2 m in the histograms due to large portions of the scans being points on the ground. This peak is clearly visible in the graphs, and the interesting observation is that the peak is narrower after calibrating the sensor, which is mainly due to a correction in the angular offsets in the sensor location.

Cameras

In general, camera calibration involves estimating a set of parameters regarding its projective properties and its pose within a global reference system.

The projective properties of the camera are defined by both its intrinsic and distortion parameters, which determine the position where a certain 3D point in the scene is projected onto the image plane. Without considering any lens distortion phenomena, a 3D point $p = (X,Y,Z)^\top$ (related to the camera reference system) projects to an im-

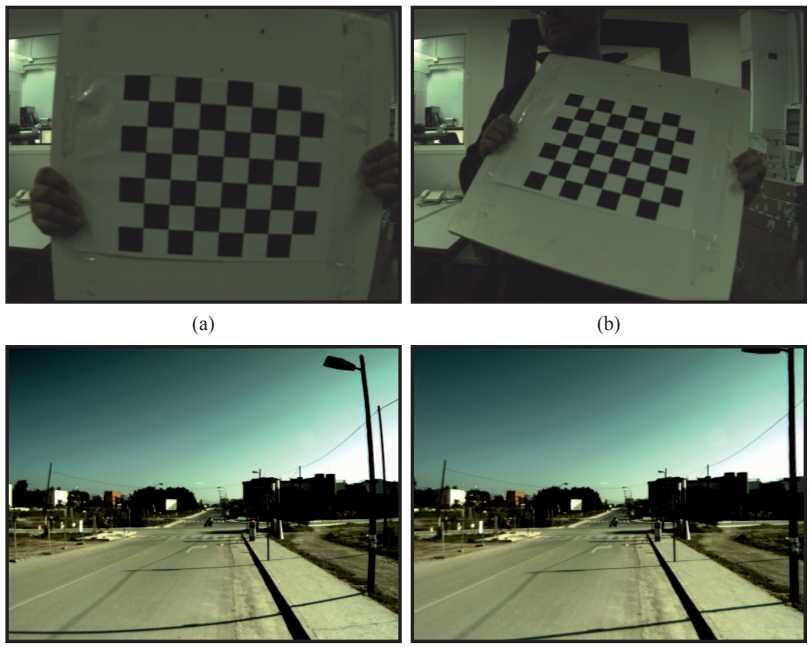


Figure 5.16: Camera calibration. (a-b) Two examples of the images employed to estimate both the intrinsic and distortion parameters. (c-d) Raw and rectified versions of one of the images in the CAMPUS-0L dataset.

age point $p' = (uw, vw, w)^\top$, with w being a scale factor, through the pin-hole model equation:

$$\begin{pmatrix} uw \\ vw \\ w \end{pmatrix} = \mathbf{R} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{5.13}$$

where f_x and f_y stand for the focal length in units of pixel width and height, respectively, $\left(c_x, c_y\right)$ are the coordinates of the principal point of the camera (also known as the image center), and (u, v) represent the projected point coordinates within the image reference system, in pixel units.

Besides, cameras are typically affected by distortion, which produces a displacement of the projected image points and whose major components are known as radial and tangential distortion. Thus, the pin-hole projection model can be extended as:

$$u' = u + u^r + u^t \tag{5.14}$$
$$v' = v + v^r + v^t \tag{5.15}$$

	Left Camera	Right Camera
f_x (px)	923.5295	911.3657
f_y (px)	922.2418	909.3910
c_x (px)	507.2222	519.3951
c_y (px)	383.5822	409.0285
k_1	−0.353754	−0.339539
k_2	0.162014	0.140431
p_1	$1.643379 \cdot 10^{-03}$	$2.969822 \cdot 10^{-04}$
p_2	$3.655471 \cdot 10^{-04}$	$-1.405876 \cdot 10^{-04}$

Table 5.5: Calibrated intrinsic parameters of the cameras.

being the radial terms

$$u^r = u\left(k_1 r^2 + k_2 r^4\right) \tag{5.16}$$

$$v^r = v\left(k_1 r^2 + k_2 r^4\right) \tag{5.17}$$

and the tangential ones

$$u^t = 2p_1 uv + p_2\left(r^2 + 2u^2\right) \tag{5.18}$$

$$v^t = p_1\left(r^2 + 2v^2\right) + 2p_2 uv \tag{5.19}$$

with $r = \sqrt{u^2 + v^2}$.

In this work, the intrinsic and distortion parameters of the cameras are estimated through the Zhang's well-known procedure presented in [209]. As the input for the calibration method we employed a sequence of images showing a calibration pattern (in particular, a checkerboard), from several different points of view – see figure 5.16a–b for an example of the calibration images. Figures 5.16c–d show the raw and rectified versions of a representative image from the CAMPUS-0L dataset, respectively. The so-obtained intrinsic parameters are shown in table 5.5.

On the other hand, an initial rough estimation of the camera pose on the vehicle was obtained by means of a measuring tape. Subsequently, it was refined by applying a Levenberg-Marquardt optimization for a set of points with known 3D coordinates (from point clouds generated by the vertical scanners) and manually selected pixel coordinates. The algorithm minimizes the overall square error of the point projections in the images, resulting in the final camera poses that can be seen in table 5.2.

IMU

As mentioned above, the MTi sensor position on the vehicle is not relevant when employed to measure only the attitude and heading of the vehicle. However, it may be interesting to roughly determine its performance during the vehicle navigation, which can be accomplished by comparing the MTi readings with the estimated angles

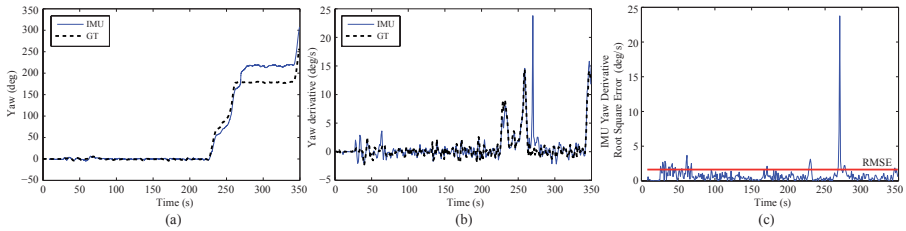


Figure 5.17: IMU performance with respect to the ground truth. (a) *Yaw* estimation provided by the MTi device (*IMU*) and the ground truth (*GT*). (b) *Yaw* velocity computed from both data. (c) IMU error in *yaw* derivative estimation with respect to the ground truth and corresponding root mean-square-error (RMSE) (Data corresponding to the CAMPUS-0L dataset).

provided by the ground truth. Since the vehicle movement through the environment may be considered locally almost-planar, in the comparison we focus on the *yaw* angle (i.e. the vehicle heading), as it undergoes larger variations than the attitude components. For this comparison experiment, we use the data from the CAMPUS-0L dataset because the ground truth estimation is available in almost the whole trajectory.

Figure 5.17a shows a comparison between the *yaw* angle provided by the IMU and that computed from the ground truth. The vehicle trajectory is shown in figure 5.10b, where it can be seen a complete 180 deg turn performed at the end of the campus boulevard. This is illustrated in figure 5.17a around $t = 250s$, where it can be noticed that the IMU readings indicate a turn of about 200 deg, which represents a severe drift with respect to the ground truth.

This drift becomes more evident in the peaks present in Figures 5.17b–c, where they are shown both the *yaw* velocity (i.e. its derivative) and the root square error of the IMU *yaw* velocity with respect to the ground truth at each time step. This device performance seems to be acceptable in most of the navigation but its data can not be considered reliable when experiencing noticeable turns. The RMSE value of this error is about 1.631 deg/s, being the mean and the maximum velocities 1.199 deg/s and 14.354 deg/s, respectively.

For the measured *pitch* velocity, we obtain an RMSE value of 0.812 deg/s, while 0.092 deg/s and 0.487 deg/s are the values of the mean and the maximum velocities, respectively. Finally, values for the RMSE, mean and maximum *roll* velocities are 0.609 deg/s, 0.149 deg/s, and 0.882 deg/s, respectively.

5.2.5 Uncertainty analysis

Comparing the estimate from a given SLAM method against our ground truth can only be significant if uncertainties (both in the SLAM output and the ground truth) are carefully taken into account. In this section we derive an estimate of the probability distribution of our reconstructed vehicle poses v_t. For that goal it is first needed the distribution of the errors in the 3D locations measured by the three GPS receivers.

Estimation of RTK-GPS noise

From the sensor characterization, the working principle of RTK-GPS and the manufacturer specifications, it comes out that noise in latitude and longitude coordinates is smaller than in the height. There is no reason for the statistical errors in the latitude to be different to those in the longitude, and experimental data confirm this point. This characteristic and the fact that the XY plane in our transformation to local coordinates is tangent to the GPS ellipsoid imply that the measuring errors in x and y should be equal. Let denote the standard deviation of these errors as σ_x and σ_y, respectively, being the error in the height (z axis) modeled by σ_z.

A characteristic of RTK-GPS technology is that errors in z are always larger than those in (x,y). We model this fact by introducing a scale factor η such as

$$\begin{aligned} \sigma_x &= \sigma_y = \sigma_0 \\ \sigma_z &= \eta\sigma_0 \end{aligned} \tag{5.20}$$

By grabbing thousands of GPS readings with the vehicle stopped, we have experimentally determined this factor to be approximately $\eta = 1.75$.

At this point we only need to estimate one parameter (σ_0) to have a complete model of the GPS errors. Our approach to estimate σ_0 relies on the information given by the residuals of the optimization of $\mathbf{D}$ (the inter-GPS distances), derived in section 5.2.3. Since we know that those distances must be constant, the probability distribution of the deviations with respect to the optimal value, shown in figure 5.18 as histograms, can be used to characterize the GPS localization errors.

In particular, we will focus on the probability density distribution (pdf) of d_{ij}^2, being d_{ij} the distance between two 3D points measured by the GPS receivers i and j. It can be proven that this density, which cannot be approximated by first order uncertainty propagation, is a function of the error in x and y (σ_0) and the relative locations of the GPS receivers on the vehicle l_{ij}, as estimated in section 5.2.4. Thus we can express its variance as:

$$\sigma_{d^2}^2 = f\left(\sigma_0, \{l_{ij}\}\right) \tag{5.21}$$

Since the values of l_{ij} are known, we can accurately estimate σ_0 from equation (5.21) through a Monte-Carlo (MC) simulation, arriving to the numerical results

$$\begin{aligned} \sigma_x &= \sigma_y = 0.64cm \\ \sigma_z &= 1.02cm \end{aligned} \tag{5.22}$$

which characterize the errors of the GPS 3D measurements. These values predict a standard deviation in the inter-GPS distances of 9.1 mm, which is consistent with the residuals of the LM optimization in table 5.4.

Next sections rely on this error model to finally provide the uncertainty of the vehicle path.

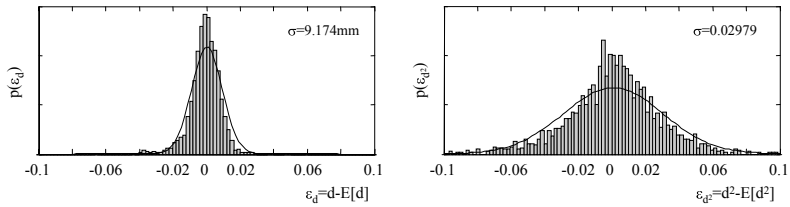


Figure 5.18: Histograms of (left) the residuals from the optimization of d_{ij} in section 5.2.3, and (right) the same for the square distances. Units are meters and square meters, respectively. The variance of the square distance residuals is a convenient value for modeling the 3D positioning errors of the GPS receivers.

Uncertainty in the vehicle 6D pose

Our model for the pose uncertainty is a multivariate Gaussian distribution, such as its mean $\hat{v}_t$ coincides with the optimal value derived in equation (5.10) and its 6×6 covariance matrix W_t, is determined next. That is,

$$v_t \sim \mathcal{N}(\hat{v}_t, W_t)$$

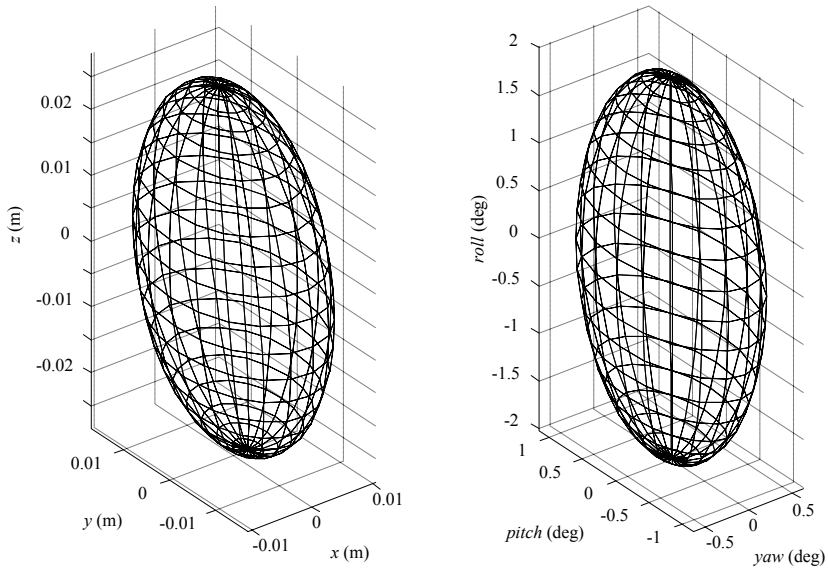
The covariance matrix W_t depends on the vehicle orientation, thereby varying with time. Furthermore, it should be estimated in each case from a Monte-Carlo (MC) simulation which provides a better approximation than linearization approaches. To avoid executing one MC estimation for every possible v_t we propose to only estimate the covariance for the case of the vehicle at the origin, then rotate this reference covariance according to the real attitude angles of the vehicle. That is, if $W^\star$ stands for the reference covariance matrix,

$$W_t = J(v_t) W^\star J(v_t)^\top \tag{5.24}$$

where the Jacobian of the transformation $J(v_t)$ incorporates the 3×3 rotation matrix $R(v_t)$ associated to the vehicle pose v_t:

$$J(v_t) = \left[\begin{array}{c|c} R(v_t) & 0 \\ \hline 0 & I_3 \end{array}\right] \tag{5.25}$$

$$W^\star = \begin{bmatrix} 1.4472\cdot10^{-5} & -7.6603\cdot10^{-9} & -7.5266\cdot10^{-6} & 1.2761\cdot10^{-8} & -6.5127\cdot10^{-6} & 3.4478\cdot10^{-8} \\ -7.6603\cdot10^{-9} & 3.6903\cdot10^{-5} & 1.3863\cdot10^{-7} & -1.7907\cdot10^{-5} & 1.1747\cdot10^{-7} & 2.0664\cdot10^{-5} \\ -7.5266\cdot10^{-6} & 1.3863\cdot10^{-7} & 1.0407\cdot10^{-4} & -1.5037\cdot10^{-7} & 6.0132\cdot10^{-5} & -1.9469\cdot10^{-7} \\ 1.2761\cdot10^{-8} & -1.7907\cdot10^{-5} & -1.5037\cdot10^{-7} & 1.5434\cdot10^{-5} & -1.2943\cdot10^{-7} & -7.3799\cdot10^{-7} \\ -6.5127\cdot10^{-6} & 1.1747\cdot10^{-7} & 6.0132\cdot10^{-5} & -1.2943\cdot10^{-7} & 5.2093\cdot10^{-5} & -1.2652\cdot10^{-7} \\ 3.4478\cdot10^{-8} & 2.0664\cdot10^{-5} & -1.9469\cdot10^{-7} & -7.3799\cdot10^{-7} & -1.2652\cdot10^{-7} & 1.5840\cdot10^{-4} \end{bmatrix} \tag{5.23}$$



(a) (b)

Figure 5.19: The 95% confidence ellipsoids for the uncertainty of the vehicle 6D pose, represented separately for (x, y, z) (a) and the three attitude angles (b).

By means of 10^7 simulations of noisy GPS measurements we arrive to the reference matrix shown in equation (5.23) and pictured in figure 5.19 as 95% confidence ellipsoids. This covariance matrix is specific for the noise levels of our GPS receivers and their relative positions on the vehicle, thus it would not be applicable to other configurations.

Our public datasets available for download incorporate the precomputed values of the covariance matrix rotated accordingly to equation (5.24) for each time-step.

Trajectory of the sensors

Many existing SLAM techniques assume that the origin of the *robocentric* coordinate system coincides with that of the sensor – MonoSLAM, or SLAM based on monocular cameras, is a prominent example [41]. This approach is clearly convenient when there is only one sensor (e.g. one camera in MonoSLAM or one laser scanner in 2D SLAM).

A ground truth for those methods can be easily obtained from the vehicle trajectory v_t derived in section 5.2.3 and the local coordinates of the sensor of interest, denoted by u in the following. A summary of these local coordinates for our vehicle is presented in table 5.2.

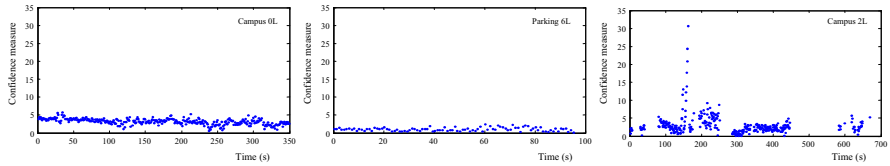


Figure 5.20: Our measure of the confidence of the ground truth for three different datasets. Lower values mean that the measured 3D positions for the GPS devices are closer to the condition of rigid rotations, thus they must be more accurate.

For each time-step t, the pose of the given sensor in global coordinates s_t is obtained as the result of the 6D pose composition

$$s_t = v_t \oplus u \tag{5.26}$$

Taking into account that both v_t and u have associated uncertainties, modeled by the covariance matrices W_t and U, we also represent the global coordinates as multivariate Gaussians with mean $\hat{s}_t$ and covariance S_t. The mean is simply given by equation (5.26) applied to the means of the vehicle path and the local coordinates, while the derivation of the covariance can be found in [14].

For the purpose of estimating the ground truth for a sensor path s_t, we could assume a perfect knowledge of the local coordinates, which implies a null U matrix. However, for the sake of accuracy this matrix should roughly model the potential errors in the process of manually measuring these quantities (or in the optimization methods used for some sensors, e.g. the cameras in section 5.2.4). The sensor paths in the published datasets assume a standard deviation of 0.02 mm and 0.1 degrees for all the translations and rotations, respectively.

A measure of the ground truth consistency

Each pose in the ground truth of the vehicle trajectory can be seen as our estimate of the corresponding *random variable*. As such, we can only expect to obtain an optimal estimation and its corresponding uncertainty bound, as derived in previous sections.

However, in our problem there is an additional constraint which can be used to provide a measure of the consistency for each ground truth point v_t. The constraint is the rigid body assumption, and as discussed next will determine how much confident shall we be about the estimate of each vehicle pose. It must be highlighted that this measure cannot fix in any way the optimal estimations already derived in previous sections.

Our derivation starts with the vector $\mathbf{D}$ of square distances between each different pair of GPS sensors, that is:

$$\mathbf{D} = [d_{12}^2\ d_{13}^2\ d_{23}^2] = f(\{x_i, y_i, z_i\}) \quad , i = 1,2,3$$

Since each distance is a function of the measured 3D locations (x_i, y_i, z_i),

$$d_{ij}^2 = (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 \tag{5.27}$$

the covariance matrix of D can be estimated by linearization as follows:

$$\begin{aligned} \Sigma_{\mathbf{D}} &= \mathbf{F} diag\left(\sigma_x^2, \sigma_x^2, \sigma_x^2, \sigma_y^2, \sigma_y^2, \sigma_y^2, \sigma_z^2, \sigma_z^2, \sigma_z^2\right) \mathbf{F}^\top \\ &= \sigma_0^2 \mathbf{F} diag\left(1,1,1,1,1,1,\eta^2,\eta^2,\eta^2\right) \mathbf{F}^\top \end{aligned}$$

where in the last step we employed the definitions in equation (5.20).

Let $\Delta_{d_{ij}}$ be the distance between the points i and j in the XY plane, and $\Delta_{z_{ij}} = z_i - z_j$ their distance in z. Since the GPS units are roughly at the same height in our vehicle, it turns out that $\Delta_{z_{ij}}^2 \ll \Delta_{d_{ij}}^2$. By using this approximation, it can be shown after some operations that the covariance matrix $\Sigma_{\mathbf{D}}$ becomes:

$$4\sigma_0^2 \begin{bmatrix} 2\Delta d_{12}^2 & \Delta_{12}\Delta_{13}\cos\phi_{13}^{12} & -\Delta_{12}\Delta_{23}\cos\phi_{23}^{12} \\ \Delta_{12}\Delta_{13}\cos\phi_{13}^{12} & 2\Delta d_{13}^2 & \Delta_{13}\Delta_{23}\cos\phi_{23}^{13} \\ -\Delta_{12}\Delta_{23}\cos\phi_{23}^{12} & \Delta_{13}\Delta_{23}\cos\phi_{23}^{13} & 2\Delta d_{23}^2 \end{bmatrix}$$

Here it has been employed ϕ_{ik}^{ij} as the angle between the pair of 2D vectors passing by the GPS sensors i and j, and i and k, respectively. Disregarding small variations due to terrain slopes, these angles and all the distances in equation (5.28) remain constant with time, thus it provides a good approximation of the covariance of $\mathbf{D}$ during all the datasets.

We can now define our measure of the quality of the GPS positioning at each time-step t as the Mahalanobis distance between the vector of measured square distances $\mathbf{D}_t$ and the optimized values $\hat{\mathbf{D}}$, using the above derived covariance matrix $\Sigma_{\mathbf{D}}$, that is:

$$q_t = \sqrt{\left(\mathbf{D}_t - \hat{\mathbf{D}}\right)^\top \Sigma_{\mathbf{D}}^{-1} \left(\mathbf{D}_t - \hat{\mathbf{D}}\right)} \tag{5.28}$$

Lower values of this measure reflect a better quality in the ground truth. We illustrate the behavior of this confidence measure in figure 5.20 for three different datasets. It can be remarked how the measure remains relatively constant with time, thus guaranteeing the accuracy of our estimated ground truth. Regarding the high values observed about $t = 170$ for the dataset CAMPUS-2L, they coincide with a moderate slope in the terrain, which may render our approximation of $\Sigma_{\mathbf{D}}$ less accurate for those specific segments of the dataset. Therefore, our quality measure should be taken more into account when the vehicle moves in relatively more horizontal areas, e.g. all three *parking* datasets.

5.2.6 Validation

Quantitative validations have been already presented along the chapter. For example, the narrow distribution of the residuals in figure 5.18a implies that our reconstructed

paths have errors in the order of 1 cm. The uncertainties derived in section 5.2.5 confirm our expectation of centimeter-level errors.

In addition, this section provides qualitatively illustrations of the accuracy of the reconstructed trajectories for the vehicle and the sensors by means of building different kinds of maps directly from the ground truth.

Construction of 3D point clouds

To validate the accuracy of the path ground truth and the calibration of vertical laser scanners, we can compare, by visual inspection, the actual shape of a representative real object in the scene with its 3D cloud point structure generated from laser data and the vehicle path.

In that sense, consider figure 5.9e, corresponding to PARKING-6L, which shows a group of cars parked under some trees. Please note how some of the cars had at least two of their sides scanned by the vertical laser scanners, as the vehicle navigated around them in different directions. It can be appreciated in the figure that the generated 3D point cloud describing the cars visually fits their real shape quite accurately, revealing a considerable consistency between the real world and its corresponding point cloud representation[11]. Therefore, both ground truth estimation and laser scanners calibration can be considered as notably reliable. Nevertheless, it must be remarked that expected errors in the point clouds are larger than those in the vehicle path, since the projected points magnify the effects of small orientation errors in the vehicle.

Construction of colored 3D maps

The camera calibration, which involved the estimation of both distortion and projection parameters and its location on the vehicle, can be also validated by determining the real color of the points in the above-mentioned 3D point clouds.

For that purpose, we re-projected those 3D points into the captured images, at each time step, and determined their real colors from the re-projected pixel coordinates in the images. As shown in figure 5.21a–b for the CAMPUS-0L dataset, the color information and the 3D structure of the scene seems to be sensibly coherent. A pair of representative examples can be found in the buildings surrounding the road, whose colors are that of the red bricks which compose their fronts, and in the crosswalk shown in the middle of figure 5.21a, which by visual inspection seems to be properly projected on the floor.

5.3 Dataset #2: The Málaga Urban Dataset

An outdoor dataset within urban scenarios is presented in this section. Similarly to the section above, we first describe the employed vehicle along with the characteristics

[11] The complete 3D point clouds are published along the dataset files.

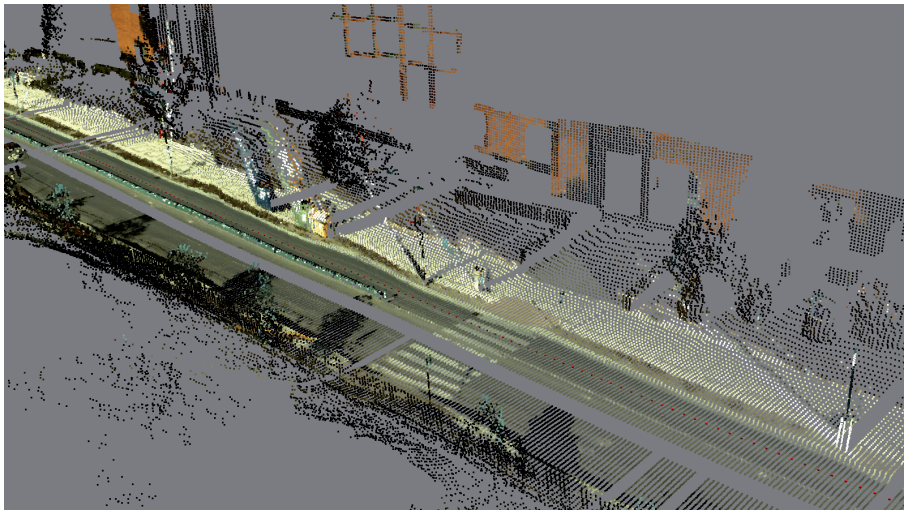


Figure 5.21: A snapshot of the colored 3D point map employed for the validation of cameras calibration, corresponding to the CAMPUS-0L dataset.

of the available sensors. Note that, although some of them coincide with those used in the previously presented datasets, we depict here their features for the sake of completeness. Finally, we describe the characteristics of our dataset, splitting it in segments which somehow may be considered to be a consistent unity.

5.3.1 Vehicle setup

In order to be able to navigate outdoors in a safe way throughout typical urban scenarios we decided to employ a common *Citroën C4* car, shown in figure 5.22. All the sensors were installed in a modified roof-rack, designed for a flexible placement of heterogeneous devices. This configuration allows us to drive among the city traffic without restrictions.

Two computers were also installed inside the vehicle to cope with the computational and storage bandwidth requirements. All the electrical power for computers and sensors was obtained from the vehicle's own power system.

We recorded data from an overall of eight sensors: one stereo camera, five laser scanners, one inertial measurement unit (IMU) and one GPS receiver. Figure 5.23 schematically illustrates the placement of each sensor on the vehicle, with approximate (hand-measured) coordinates shown in table 5.6. The local frame of reference is set such that the positive x axis always points forwards while z points upwards, as customary in mobile robotics.

(a) (b)

Figure 5.22: The instrumented vehicle employed for collecting the dataset: (a) general view and (b) close-up of the sensors.

Sensor	***x*** (m)	***y*** (m)	***z*** (m)	***yaw*** (deg)	***pitch*** (deg)	***roll*** (deg)
`CAMERA1`	0.785	0	0.273	0	−8.2	0
`XSensMTi`	0.400	0.040	0.000	0	0	0
`GPS_DELUO`	0.155	0.069	0.004	n/a	n/a	n/a
`LASER1` (Rear SICK)	−0.023	0	0.097	−180	0	0
`LASER2` (Front SICK)	0.536	0	0.093	0	0	0
`HOKUYO1` (Front)	0.536	0	0.273	0	21.4	0
`HOKUYO2` (Right)	0.075	−0.489	0.055	−90	0	−90
`HOKUYO3` (Left)	0.075	0.489	0.055	90	0	90

(n/a: not applicable)

Table 5.6: Summary of approximate sensor positioning on the vehicle. Refer to figure 5.23.

Next we briefly describe the relevant characteristics of each sensor and the reasons for their inclusion in the dataset.

Stereo camera

Color imaging was provided by a Point Grey Research's *Bumblebee 2* stereo camera, configured to capture images at its maximum resolution of 1024×768 at 20 fps. As opposed to our previous dataset [16], the usage of a stereo camera instead of two independent ones assures a precise synchronization in both image streams. The camera gain and white-balance control were left in automatic mode.

Once one determines the camera intrinsic parameters, the rigid mounting of the two CCD sensors inside the camera and the use of a fixed focal distance lead to a reliable calibration that is not affected by shocks and vibrations. Although the dataset includes all camera calibration parameters, we also publish a collection of raw stereo images of a checkerboard to allow the reader applying different calibration methods.

This camera was placed pointing forwards and slightly tilted up, to avoid capturing part of the vehicle chassis. Despite the small parallax obtained during navigation

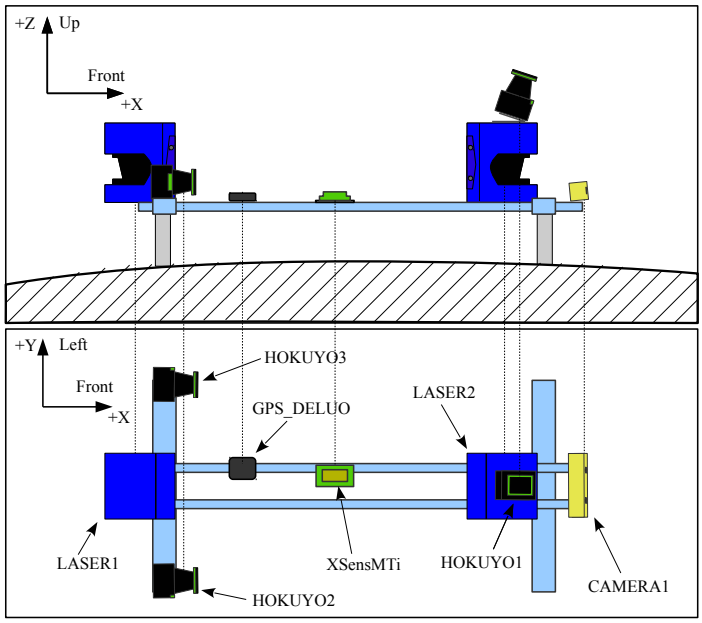


Figure 5.23: Side and top views, respectively, of the relative positions of sensors on the vehicle's roof-rack structure. Compare to figure 5.22b. Not to scale.

from forward-looking cameras, we decided to use this configuration for its interesting applications in detecting other vehicles, pedestrians, traffic lights, etc.

Laser scanners

The vehicle was equipped with five laser scanners: three Hokuyo UTM-30LX and two SICK LMS-200. The former are small, energy-efficient scanners with a range of 30 meters and a field of view of 270°. With an angular resolution of 0.25°, they provide nominal accuracies of 30 mm and 50 mm for distances below and above 10 meters, respectively. The latter models, manufactured by SICK for industrial use, are considerably heavier, more robust and more energy demanding. In turn, their working range extends up to 80 meters and are less prone to detecting *phantom points* near sharp edges, a problem occasionally found in range data from the Hokuyo sensor.

Regarding the placement of the scanners on-board, they can be divided in three groups:

- The two SICK scanners sense in the horizontal plane. These data may be useful for 2D SLAM for parts of the trajectory that are flat enough.
- Two lateral Hokuyo sensors provide a vertical scanning of the vehicle surroundings.

- Finally, one Hokuyo scanner is placed pointing forwards and tilted down, in order to sequentially scan the road ahead the vehicle, detect possible obstacles, etc.

The two scanners (SICK and Hokuyo) pointing forwards may find applications to detection algorithms that fuse visual and range information.

IMU

Inertial sensors based on inexpensive MEMS technology are present nowadays in many portable devices such as tablets or smartphones. Therefore, it seems reasonable to try to explore the possibilities that these sensors create for improving visual odometry or visual SLAM methods.

To endow our dataset with this kind of information we installed an *xSens MTi* inertial unit on the vehicle. It was firmly attached to the roof structure like all the other sensors, thus angular velocities sensed by the device can be accurately assigned to the rest of sensors as well, disregarding the negligible effects of the structure elastic deformations during the drive.

With a rate of 100 Hz, the measurements provided by this device include:

- 3-axis acceleration. We have experimentally measured its static error, which has a standard deviation of $\sigma_{acc} \approx 0.05\ m/s^2$.
- 3-axis instantaneous angular velocity. Its experimental angular velocity error has been found to be $\sigma_{gyro} \approx 0.4\ ^\circ/s$, while systematic errors were noticed for *yaw* (rotations around the Z axis) in the order of $\sim 0.6\ ^\circ/s$.
- Attitude dead-reckoning in 3D, as provided by the internal filter implemented by the manufacturer.

GPS receiver

We also installed a consumer-grade, low-cost GPS receiver on the car, with a two-fold purpose: (i) providing approximate positioning for a better understanding of the whole trajectory traversed in this dataset (see figure 5.24), and (ii) offering realistic GPS data for usage in visual SLAM applications aimed at the automotive industry. This sensor provides positioning data at 2 Hz during the whole dataset, with the exception of a few unavoidable segments (urban canyons and dense groves) were the signal was too weak to provide good localization. Two additional industry-grade GPS receivers were also installed in the vehicle (*mmGPS* devices from Topcon, the two cylindrical yellow devices in figure 5.22b), but, unfortunately, positioning information was not available from these receivers during the recording of the dataset.

However, frames with GPS timing information were collected from both receivers in order to accurately synchronize the local clocks of the two computers. By grabbing satellite timestamps from two identical receivers in both computers we have been able

Figure 5.24: An overview of the complete trajectory, as reconstructed from GPS data. A zoomable version is available on-line.

to establish a least-square fit of the mapping between the reference GPS time and the local clocks. More importantly, this mapping provides an accurate way of merging the partial datasets grabbed in each machine during an off-line post-processing stage. Interestingly, we found out that not only the local clocks had an offset (as could be expected) but that they exhibit a small drift (6.06 ppm and 83.33 ppm, respectively), which has been corrected in the published dataset.

5.3.2 Software

The vehicle is equipped with sensors of quite different types, each generating data at different rates. Thus, the software intended to record the data logs must be capable of dealing with asynchronous streams from the sensors. For this purpose, we employed the data logger application *rawlog-grabber*, as we also did for the dataset collection presented in the previous section.

This program launches one thread for each individual sensor. Then, each thread splits the sensory data into their corresponding natural discrete pieces (called *observations*), e.g. a complete 2D scan for laser scanners, and marks them with timestamps. Since our system does not run on a real-time OS, we have to assure that no observation is lost by creating a FIFO queue for each thread, then merging all of their outputs into a thread-safe timestamp-sorted queue, which is periodically pushed to a binary *rawlog* file. We chose binary log files for their bandwidth efficiency in contrast to other pure-text formats. Afterwards, we have post-processed the binary logs to generate plain text logs for the convenience of readers.

Collecting large images (1024×768) at real-time without dropping frames presented an additional challenge, because hard-disk bandwidth is not enough for saving raw images, while lossy compression solves the issue but introduces a high computational burden. Our approach consisted in parallelizing the latter task by creating

Sensor label	Count	Duration (s)	Actual rate (Hz)	Nominal rate (Hz)
CAMERA1	113082	5654.6	19.998	20
GPS_DELUO	11244	5653.0	1.989	2
HOKUYO1	225416	5654.62	39.864	40
HOKUYO2	225631	5654.62	39.902	40
HOKUYO3	225510	5654.62	39.880	40
LASER1	398531	5315.58	74.974	75
LASER2	404487	5498.11	73.568	75
XSensMTi	549816	5498.15	100.000	100

Table 5.7: Summary of grabbed data from each sensor. The *actual rates* shown here are the average values obtained as the ratio *count*/*duration*.

additional threads with the sole purpose of compressing images in a high-quality format (JPEG format, quality=95).

5.3.3 Dataset summary

The following paragraphs describe the most relevant characteristics and statistics about the presented dataset. However, accessing to the supplementary material online[12] is recommended for having a better insight about its content.

Description

The dataset was recorded as a single sequence during a car trip throughout different urban areas of Málaga, with a total duration of ~93 minutes. An overlaid impression of the GPS-reconstructed path over a map of the city is provided in figure 5.24.

Observations from all sensors were recorded at their maximum nominal rates. These values, along with the actual average rates obtained from the logged stream of data, are shown in table 5.7. The similarity of actual and nominal rates means that only a tiny fraction of sensory data was dropped for most sensors (mostly due to corrupt frames for communication errors), with the worst case being the sensor LASER2 (front SICK laser) for which a 1.9% of all frames were lost. An overall of 2.2 millions of individual *observations* were collected.

Regarding the trajectory followed during the recording, we can split the dataset into the following *segments* or *epochs* (within parentheses, the starting and end points measured in minutes since start):

- Epoch 1 (0 – 6 min): Four loops within the parking lot of the Computer Science School of the University of Málaga. This area was also recorded 13 months earlier for a previous dataset with a different camera [16], making this segment ideal for testing place recognition algorithms.

[12] See http://www.mrpt.org/MalagaUrbanDataset.

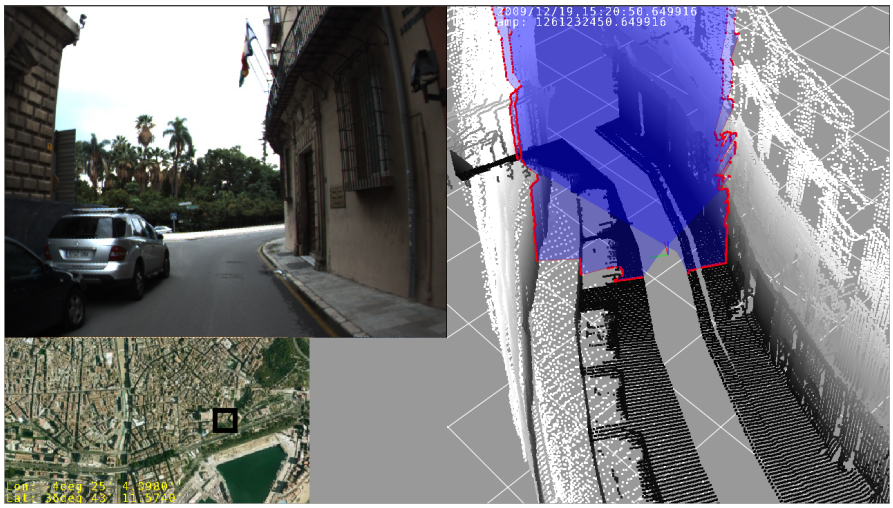


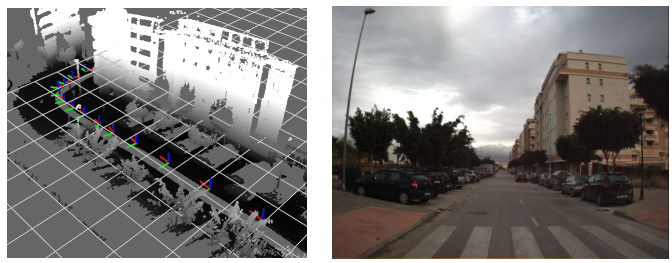
Figure 5.25: A view of the dataset *video index*, which simultaneously displays: (top-left) raw video frames, (bottom-left) the current location of the vehicle over the city map and (right) local 3D point cloud from laser scanners.

- Epoch 2 (6 – 10 min): Driving towards a nearby suburb, crossing one under-construction road.
- Epoch 3 (10 – 52 min): One of the main parts of the dataset, in which North-West Málaga suburbs (*El Cónsul* and *El Romeral*) are transversed several times including nested loop closures. The car underwent a parking maneuvering during minutes 17 – 19. Traffic lights and take overs also appear in this segment.
- Epoch 4 (52 – 60 min): A trip towards downtown, traversing a highway-like road. In contrast to the velocity range of 20 – 40 km/h (12.4 – 24.9 mph) in the other epochs, in this segment the vehicle moves faster than 50 km/h (31 mph).
- Epoch 5 (60 – 93 min): Another of the most interesting segments, since it includes several loop closures in downtown. Here we find the highest traffic density for the entire dataset.

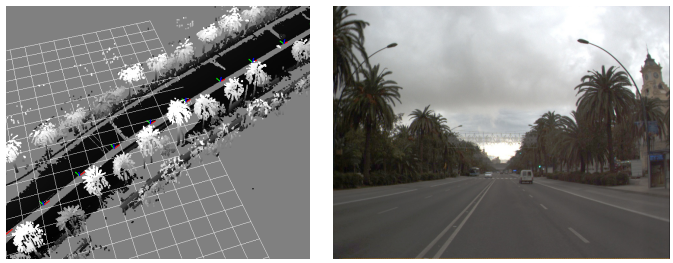
As an additional tool to help the interested reader to pick relevant segments from the dataset we created a *video index* (see figure 5.25), available online[13] . Apart from camera images, the video shows a 3D point cloud reconstruction of the environment from the vertical laser scanners and GPS data as a gross estimate of the ground truth path. Some snapshots of the obtained scenarios can be seen in figure 5.26.

In order to make working with the dataset easier, it has been further divided into 15 smaller sequences or *extracts*, illustrated in figure 5.27. A video is also available

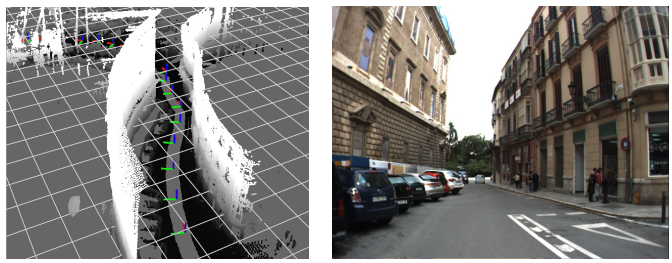
[13] `http://www.youtube.com/watch?v=tM5BSLKUSxU`



(a) t=1278s



(b) t=3732s



(c) t=3983s

Figure 5.26: Three sample screenshots from the dataset: (left) 3D reconstructions from vertical laser scanners and GPS-only information, (right) images from the stereo camera in the same places.

online for each individual sequence, such that they can be easily inspected. Next, we enumerate the length in seconds of each extract and provide a brief description of its contents:

1. Straight path in the faculty parking (39 s).
2. Through an under-construction road (92 s).
3. Three-quarters of a turn in a roundabout (41 s).
4. Crossing a roundabout, some traffic (32 s).
5. Loop closure (~ 1.7 km) in a straight avenue (240 s).

6. Loop closure (~ 1.2 km) around building blocks (230 s).
7. Loop closure (~ 0.7 km) around a small avenue (106 s).
8. Long loop closure (~ 4.5 km) (501 s).
9. Through the campus boulevard, with some traffic (50 s).
10. Multiple loop closures in a suburb area (865 s).
11. High-way incorporation, some traffic (144 s).
12. Long avenue (~ 3.7 km), dense traffic (443 s).
13. At downtown. Dense traffic and pedestrians (1572 s).
14. Direct sun conditions at a parking area (112 s).
15. Direct sun conditions at a suburb area (69 s).

Although all sensory data are provided in plain-text format, it is worth mentioning that two ready-to-use applications (named `RawLogViewer` and `rawlog-edit`) are provided to inspect, filter or split the generated binary log files. These programs are already shipped within Debian and Ubuntu GNU/Linux distributions as part of the package `mrpt-apps`. Example C++ source code is also available on-line for readers interested in parsing binary logs.

Challenges

We found that a particularly challenging problem during the recording of outdoor images was the appearance of vertical smears caused by direct sun exposure. After several attempts at different dates we obtained, in a cloudy day, the present dataset which exhibits a minor occurrence of such smears. Another challenging aspect of the images, from the point of view of computer vision, is the dynamic gain control of the camera which may introduce hurdles to feature tracking algorithms. Anyway, we believe that these challenges are intrinsic and unavoidable for any real-world problem where cameras are to be placed on vehicles for navigation in uncontrolled, outdoor scenarios.

5.4 Conclusions

The presented work aimed to contribute to the existence of a freely accessible compilation of datasets with an associated ground truth suitable for evaluating the different SLAM techniques. Thus, the first one intends to provide such an accurate ground truth for a series of outdoor datasets, including its estimated uncertainty bounds, in the range of 1 cm and 0.5 deg. These bounds remain constant with time along all the

Figure 5.27: Summary of the 15 dataset extracts available for download. For each segment, the vehicle path is shown together with aerial urban images for reference. Refer to the on-line material for color images and interactive maps.

datasets, turning the datasets into an ideal testbed for the evaluation of methods such as visual SLAM or visual odometry. The scalability of SLAM methods can be also assessed with regard to their capability of closing loops, since the more than 6 km of recorded datasets comprise several loops of different lengths.

As an additional result of our ground truth estimation, we also provide reference 3D point clouds for each of the datasets, which may be proper to test a variety of other techniques such as 3D surface reconstruction or path planning.

On the other hand, our second dataset exhibits, as its most relevant component, the presence of high-rate and high-resolution stereo video in unmodified urban scenarios. We do believe that the mobile robotics community will find it specially suited for benchmarking of visual odometry, visual SLAM and appearance-based recognition methods. Moreover, the presence of several laser scanners enables Lidar-vision object detection and recognition within realistic traffic situations.

Finally, all the datasets are freely available on-line among with extensive documentation and open-source software which allows the easy and configurable post-processing of the raw data:

- Dataset #1: `http://www.mrpt.org/malaga_dataset_2009`.
- Dataset #2: `http://www.mrpt.org/MalagaUrbanDataset`.

Chapter 6
Conclusions

Let's shake some dust. – Samson

This thesis has been mainly focused on the problem of localizing a mobile robot (or camera) within an environment while simultaneously building a map of it from stereo images. This topic has been coined visual Simultaneous Localization And Mapping (SLAM) and represents one important problem that concerns both mobile robotics and computer vision. Although a large amount of research works on this issue has been presented during the last decades for different sensors, it still remains open.

In this thesis we have worked on both visual odometry and visual SLAM, contributing to each of them with new approaches. Thus, this thesis started by developing techniques to perform stereo visual odometry, which, in short, tries to estimate the robot change in pose between consecutive time-steps, hence tracking its pose as it navigates. Typical encoder-based odometry systems are still being extensively employed but they are restricted to wheeled robots operating on planar surfaces. Visual odometry plays here a crucial role by developing new methods that are applicable to any kind of robot without any restriction about its movement.

We can distinguish between *closed-form* solutions and *iterative*, optimization-based approaches to perform visual odometry, although both types share many of the involved stages (e.g. image pre-processing, keypoint detection and matching, etc.). In this thesis, chapter 3 presented two new methods, one of each type. Our proposed closed-form solution avoids the inherent problems of iterative methods such as convergence issues and the need for an initial estimation while performing well for indoor environments. However, it demands an outlier-free input, since corrupted data lead the algorithm to erroneous results. In this sense, the well known RANSAC method could be applied to deal with outliers, but at the cost of a severe decrease of the performance, specially when the expected ratio of outliers in the input set is large.

In response to this, we have developed a new methodology (coined ERODE) based on robust kernels for fast outlier detection that operates during an iterative, non-linear optimization process to compute stereo visual odometry. This technique significantly reduces the effect of outliers in the robot ego-motion estimation, while keeping a computational cost about one order of magnitude lower than RANSAC's.

Odometry in general, and visual odometry in particular, suffers from cumulative errors along time that increase the inaccuracy of the robot pose estimation as it navigates. For this reason, odometry estimations have been typically considered as part of more complex systems that perform not only localization but also mapping. Thus, in this work we have addressed the SLAM problem for stereo vision systems within two different frameworks that coexist nowadays: *filtering* and *smoothing*. Briefly, the former operates by iteratively performing *prediction* and *observation* steps within a probabilistic framework, and by summarizing all the information up to the current time step with a probability distribution. The latter, on the other hand, includes the set of techniques that, under some assumptions, represents the maximum likelihood estimator for the problem of jointly estimating the set of camera poses and a collection of landmark 3D positions during the whole navigation.

In our first proposal, based on Rao-Blackwellised Particle Filters (RBPFs), the robot ego-motion is estimated through our above-mentioned closed-form formulation

to perform 6D visual odometry. The observation model for the RBPF algorithm considers observations as sets of landmarks determined by the combination of their 3D positions (computed from their projection in the images) and their SIFT descriptors (extracted from the local area of their image projections), as well as their associated uncertainty. We also avoid explicit data association by marginalizing out the observation likelihood over all the possible associations, mitigating this way the issues derived from incorrect matches between the observed landmarks and the map.

Our second proposal implements a complete visual SLAM system with a front-end based on ORB image features, and a back-end whose core is a novel, sparser version of relative bundle adjustment techniques (a smoothing technique). This solution also implements ERODE to improve visual odometry endurance against outliers. In this work, a bag-of-words built upon ORB descriptors assists data association to reduce the search area when matching observations and the map. The back-end is implemented following a Sparser Relative Bundle Adjustment (SRBA) methodology, which represents the problem as a graph whose nodes are the robot position at the so-called keyframes (special frames carefully selected among all the grabbed images) and the 3D landmarks. The graph constraints (edges) between nodes are the unknowns to be estimated. Our SRBA-based method defines *submaps* in the graph, in order to reduce the topological distance between keyframes. This leads to an increase of the sparsity level in the matrices involved in the optimization process with respect to other relative bundle adjustment approaches. Along with a limitation on the number of unknowns to be optimized, this keeps the optimization time bounded at every keyframe. Despite the fact that the built map is only locally consistent (unlike Global Bundle Adjustment approaches), it is possible to employ SRBA in several applications related to localization. Still, a global map can be built from the relative representation of both keyframes and landmarks positions.

All the methods presented in this thesis have been tested with synthetic and real data from simulations or real image datasets, validating this way their capabilities to deal with both visual odometry and visual SLAM.

Finally, we have focused on providing the computer vision and mobile robotics communities with two public-access collections of outdoor datasets that contain data from a relatively large and heterogeneous set of sensors, including, but not limited to, laser scanners, inertial measurement units and cameras.

Our major contribution for the first collection is the derivation of an accurate GPS-based ground truth for the vehicle trajectory along with an exhaustive study of its associated uncertainty bounds, in the range of 1 cm and 0.5 deg. These bounds remain constant with time along all the datasets, turning the datasets into an ideal benchmark for the evaluation of visual SLAM and visual odometry methods. Reference 3D point clouds are also provided to test a variety of other techniques such as 3D surface reconstruction or path planning. Our second dataset, in turn, provides high-rate and high-resolution stereo video in unmodified urban environments, including challenging scenarios with moving objects and realistic traffic situations. The presence of loops within many of the presented datasets also contribute to the validation of scalability for SLAM techniques.

We do believe that the combination of mobile robotics and computer vision techniques represents a powerful framework towards the development of truly autonomous robots. However, achieving this implies following a difficult path and there is still a long way to go beyond the proposals presented in this work.

Robustness is one of the cornerstones that must be fulfill, since real situations present several challenges that often become intractable and lead to undesired results. Among them, the presence of dynamic environments specially affects to vision-based algorithms since most of them assume to operate in static scenarios. In this sense, developing and integrating techniques such as *clustering*, capable to detect moving objects within the scene and to isolate their movements so that ego-motion estimations remain unaltered, seem to be a promising approach. Furthermore, permanent changes in the environment are even more challenging situations. In this sense it would be interesting that the system could detect them and update its already stored map to adapt to these new changes. For bundle adjustment techniques, this would mainly imply to develop a system of landmarks deletion that would modify the built graph.

On the other hand, outdoor datasets present some difficulties when computing the 3D position of far landmarks, since the disparities between their correspondent image projections are typically small. Therefore, small errors in keypoint detection turn into large errors in 3D positions, specially for small-baseline stereo cameras. This effect can be mitigated by using detectors that provide *subpixel*-level keypoints. Therefore, employing a fast, subpixel-level keypoint detector that provides binary descriptors is of major interest for vision-based outdoor applications.

Regarding more practical issues, code optimization and multi-threading are future improvements that our algorithms must undergo in order to be applicable in real-time situations. This could be also combined with the use of lower resolution images (e.g. 320x240 pixels), situations against our methods must be tuned and validated. In this sense, it would be interesting to pre-process the first images captured for a certain experiment so that the system can auto-adjust many of the parameters that define its operation. Finally, faster alternatives than brute-force matching for ORB descriptors must be explored to reduce the time consumption of stereo and inter-frame search for correspondences.

Appendices

Appendix A
A (tentative) alternative to RANSAC

This section describes DaCOR (Divide and Conquer Outlier Rejector), a tentative method designed as a potential alternative to RANSAC that overcomes its high computational cost when dealing with large outlier ratios. However, as we will show next, there is a limitation on the outlier ratio that DaCOR can assume to guarantee a final correct solution. Unfortunately, this limit is low enough to render DaCOR inconvenient in practice. Nevertheless, we include DaCOR in this thesis as a small research work towards outliers detectors that overcome RANSAC method performance issues. In this appendix, we will follow the nomenclature shown in table A.1.

Table A.1: RANSAC and DaCOR parameters nomenclature

Parameter	Meaning
general	–
N	Number of points in the input set
p	Percentage of outliers in the input set
m	Minimum size of a set to be used to find a model
n_h	Number of hypotheses to be explored [RANSAC]
q	Probability of getting a outlier-free final set [RANSAC]
timing	–
t_m	Time to compute a model
t_e	Time to evaluate a model against a certain data value
t_c^d	Time to compute the candidate solution [DaCOR]
t^r	Time complexity [RANSAC]
t^d	Time complexity [DaCOR]
t_{mf}^r	Time to compute the final model (refinement) [RANSAC]
t_{mf}^d	Time to compute the final model (refinement) [DaCOR]

A.1 RANSAC Summary

In order to keep this appendix self-contained, we briefly describe RANSAC technique operation. RANSAC is a popular hypothesis-and-verify method capable of computing a model that fits a set of input data corrupted by outliers. Its general procedure is:

1. Randomly selects m elements from the input set of N elements, being m the minimum amount of data elements needed to find a model.
2. Finds the model x that fits the selected elements (candidate model).
3. Evaluates the rest of the elements against the candidate model in order to find those that support it.
4. Repeats the process a pre-defined (n_h) number of times so that n_h hypotheses are tested.
5. Selects the model with the maximum support.
6. Performs a final refinement of the most supported solution with the full set of inliers.

The value of n_h depends on the estimated ratio of outliers p and the desired probability of finding an outlier-free final set q as follows:

$$n_h = \frac{log(1-q)}{log\left[1-(1-p)^m\right]}. \tag{A.1}$$

A high ratio of outliers p produces an unacceptable increase in the RANSAC method computational time, often preventing its use in real-time applications. In this scenario, we look for alternative methods that overcome this limitation and hence the design of our tentative proposal.

A.2 Method Description

Our proposed method DaCOR proceeds as follows:

1. Randomly divide the input set of N elements into $G = \frac{N}{m}$ minimal groups of elements.
2. Find the model $\mathbf{x}_i$ that best fits each of the input groups.
3. Compute a candidate solution $\mathbf{x}_c$ from the groups of solutions $\{\mathbf{x}_i\}_{i=1,\ldots,G}$, following one of this two approaches:
   (a) Compute the *median* of all the computed solutions for each of the model $\mathbf{x}$ dimensions.

(b) Compute one histogram for each of the dimensions of the model $\mathbf{x}$ and compute their *modes*.

4. Classify as *inliers* those elements corresponding to the groups that generated a model $\mathbf{x}_i$ close enough to $\mathbf{x}_c$.

5. Evaluate the rest of the elements against the candidate solution $\mathbf{x}_c$. Those generating a residual below a defined threshold are added to the inlier set, otherwise the element is considered an *outlier* and therefore discarded.

6. Perform a final refinement of the candidate solution $\mathbf{x}_c$ with the full inlier set.

A.3 Method Limitations

DaCOR suffers from a limitation on the assumed ratio of outliers that corrupt the data depending on the option chosen in step 3 above.

- In option 3(a), the number of groups leading to nearly correct solutions must be at least $G/2$ (half of the total number of groups) in order to ensure that the median leads to a proper candidate solution (i.e. close to the true solution).

- In option 3(b), the number of groups leading to nearly correct solutions can be inferior to $G/2$ but it has to be large enough to create a mode in the histogram close to the true solution.

Thus, let p be the ratio of outliers in the group of N input points (so there are pN outliers within the data). In the *worst case*, just one outlier will be inserted in each group, leading the group to an incorrect solution. Therefore, with G initial groups, there will be $G_o = pN$ groups containing outliers and the number of groups containing all inliers is

$$G_i = G - G_o = \frac{N}{m} - pN = N\left(\frac{1}{m} - p\right) \tag{A.2}$$

If we choose option 3(a), G_i must be at least $G/2$ so

$$\begin{aligned} G_i &> \frac{G}{2} \\ N\left(\frac{1}{m} - p\right) &> \frac{N}{2m} \\ p &< \frac{1}{2m}. \end{aligned} \tag{A.3}$$

With $m = 3$ (as for visual odometry), we find that the theoretical limit in the ratio of outliers for this option is $\mathbf{p < 1/6}$. In practice this upper limit is already optimistic as there may exist degenerate configuration of points in groups composed by all inliers.

On the other hand, in option 3(b), the limitation is not so restrictive but, obviously the minimum number of groups that contain only inliers must be over zero, so

$$\begin{aligned} G_i &> 0 \\ N\left(\frac{1}{m} - p\right) &> 0 \\ p &< \frac{1}{m}. \end{aligned} \tag{A.4}$$

Again, with $m = 3$, the theoretical limit in the ratio of outliers for this option is $\mathbf{p < 1/3}$ with the same considerations about degenerate cases as in option 3(a).

A.4 Computational Time Considerations

This section addresses the theoretical time performance of DaCOR with respect to the computational burden of RANSAC.

First, we define a model for the time spent by RANSAC to find a solution at each time-step, which is given by

$$t^r = n_h \left(t_m + (N-m)t_e\right) + t^r_{mf}, \tag{A.5}$$

This includes the random selection of minimal groups, the computation of their solutions, the evaluation of the rest of the points against the found solutions and a final refinement of the solution.

On the other hand, the expression that describes the time spent by DaCOR is

$$t^d = Gt_m + mG_o t_e + t^d_c + t^d_{mf}. \tag{A.6}$$

where t^d_c represents the time to compute the candidate solution $\mathbf{x}_c$ either using the median or the histogram approaches described above.

In this expression, the number of groups containing at least one outlier G_o spans from

$$G_o \in \left[\frac{pN}{m}, pN\right] \tag{A.7}$$

which corresponds to the best case, with all the outliers selected in groups of m elements, and the worst case, where each outlier is inserted in a different group of m elements, respectively.

From a C++ implementation of the methods, we have computed real values for the time parameters in those expressions:

$$\begin{aligned} t_e &= 0.26\ \mu\text{s} \\ t^d_c &= 21.7\ \mu\text{s} \\ t^r_{mf} &= 400.8\ \mu\text{s} \\ t^d_{mf} &= 825.4\ \mu\text{s} \\ t_m &= 16\ \mu\text{s}, \end{aligned}$$

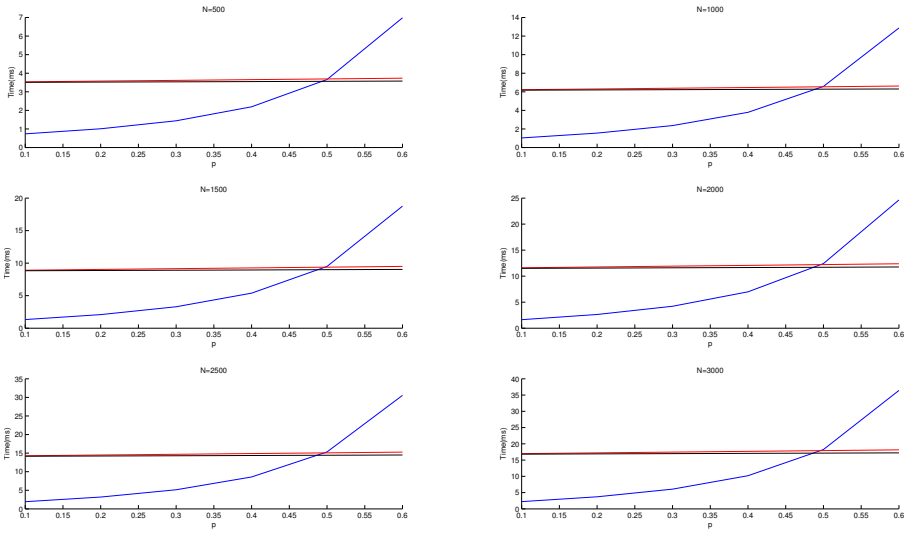


Figure A.1: Time comparison for RANSAC (blue) and DaCOR (red - worst case, and black - best case) as the ratio of outliers p grows and for $N = \{500, 1000, 1500, 2000, 2500, 3000\}$ (from top-left to bottom-right).

Then, we plotted equations A.5 and A.6 for different values of the number of input points N and the ratio of outliers p. The results are shown in figures A.1 and A.2.

As it can be seen in figure A.1, the shape of the plots are very similar independently of the number of points and only when $p > 0.5$ RANSAC computational time is higher than that for DaCOR. On the other hand, figure A.2 shows that DaCOR time is almost independent of the ratio of outliers and only when the ratio of outliers is over 0.5 the number of hypothesis that RANSAC has to test significantly increases and DaCOR is more efficient than RANSAC.

A.5 Conclusions

The theoretical study shows that DaCOR outperforms RANSAC with regards to computational complexity for high ratio of outliers ($p > 0.5$). However, the theoretical limit of the outlier ratio for DaCOR to be usable has been found to be $p < 1/3$, rendering DaCOR useless in practice.

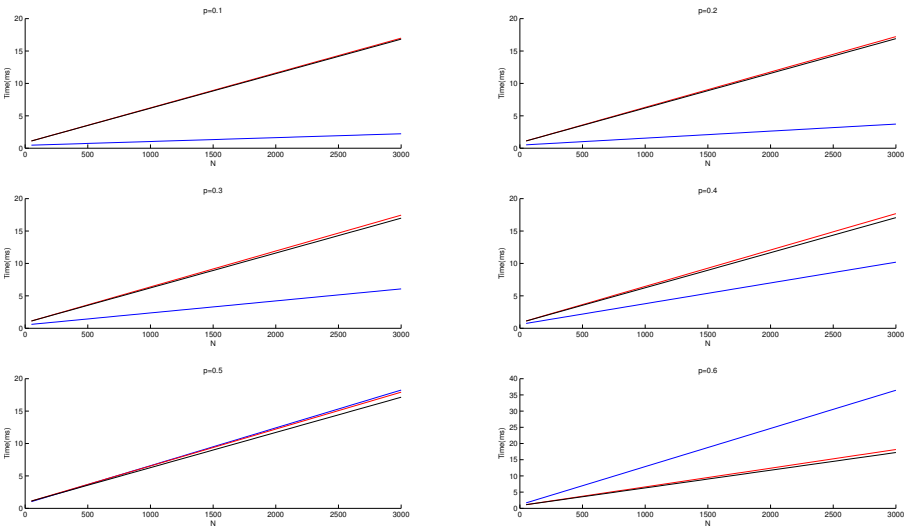


Figure A.2: Time comparison for RANSAC (blue) and DaCOR (red - worst case, and black - best case) as the number of points N grows and for $p = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$ (from top-left to bottom-right).

Appendix B
Derivation of the landmark update equations

In section 4.2 we have employed a linear Kalman filter to estimate the 3D position of the landmarks in the map constructed during the visual SLAM process. This appendix addresses the derivation of the position update equations, from those of the Kalman Filter so, in order to build here a complete derivation, let us start by recalling them although they were already presented in section 2.3.1.

As stated in [99], the Kalman filter method comprises two different steps: *prediction* and *update*. In the prediction steps it is estimated the state of the system at the current time-step $\mathbf{x}_{k|k-1}$ from the previous estimate $\mathbf{x}_{k-1|k-1}$ and a control action $\mathbf{u}_k$, whereas the update step refines the state estimation by introducing the information provided by the observation $\mathbf{y}_k$, yielding a more accurate estimation $\mathbf{x}_{k|k}$. In addition, the KF provides a covariance matrix associated to the estimation uncertainty $\mathbf{P}_{k|k}$.

The equations for the discrete-time KF algorithm are as follows:

Prediction

$$\begin{aligned} \mathbf{x}_{k|k-1} &= \mathbf{F}_k \mathbf{x}_{k-1|k-1} + \mathbf{B}_k \mathbf{u}_k \\ \mathbf{P}_{k|k-1} &= \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^{\mathsf{T}} + \mathbf{Q}_k \end{aligned} \tag{B.1}$$

where $\mathbf{F}_k$ relates the state at the previous time step to that at the current step (also known as the *transition model*), $\mathbf{B}_k$ relates the control action with the state, and $\mathbf{Q}_k$ is the covariance of a zero-mean multi-variate Gaussian distributed noise which affects the *a priori* estimation.

Update

$$\begin{aligned} \mathbf{x}_{k|k} &= \mathbf{x}_{k|k-1} + \mathbf{K}_k \mathbf{y}_k \\ \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\, \mathbf{P}_{k|k-1} \end{aligned} \tag{B.2}$$

where $\mathbf{H}_k$ is the observation model, which relates the state of the system to the observation at the current time step, $\mathbf{y}_k$ stands for the innovation, and $\mathbf{K}_k$ represents the Kalman filter gain:

$$\begin{aligned} \mathbf{y}_k &= \mathbf{z}_k - \mathbf{H}_k \mathbf{x}_{k|k-1} \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}} \left(\mathbf{H}_k \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_k\right)^{-1} \end{aligned} \tag{B.3}$$

being $\mathbf{R}_k$ the covariance of the zero-mean multi-variate Gaussian distribution affecting the measurement process.

Note that, in this work, $\mathbf{x}_{k|k-1}$ and $\mathbf{x}_{k|k}$ represents the mean of the estimated position of a landmark in the map before and after the update process, respectively, while $\mathbf{z}_k$ stand for the observed position of the landmark at time step k. Besides, $\mathbf{P}_{k|k-1}$, $\mathbf{P}_{k|k}$ and $\mathbf{R}_k$ are the covariance matrices of their associated uncertainties.

For clarity, let us take up again the notation introduced in section 4.2.1, renaming these variables as follows:

$$\begin{aligned} \mathbf{x}_{k|k} &\equiv \boldsymbol{\mu}_m & \mathbf{P}_{k|k} &\equiv \boldsymbol{\Sigma}_m \\ \mathbf{x}_{k|k-1} &\equiv \boldsymbol{\mu}_{\tilde{m}} & \mathbf{P}_{k|k-1} &\equiv \boldsymbol{\Sigma}_{\tilde{m}} \\ \mathbf{z}_k &\equiv \boldsymbol{\mu}_k & \mathbf{R}_k &\equiv \boldsymbol{\Sigma}_k \end{aligned} \tag{B.4}$$

Thus, with this change in the nomenclature and merging expressions (B.2) and (B.3), the KF prediction and update equations become:

$$\begin{aligned} \boldsymbol{\mu}_{\tilde{m}} &= \mathbf{F}\boldsymbol{\mu}_{\tilde{m}'} + \mathbf{B}\mathbf{u}_k \\ \boldsymbol{\Sigma}_{\tilde{m}} &= \mathbf{F}\boldsymbol{\Sigma}_{\tilde{m}'}\mathbf{F}^{\mathrm{T}} + \mathbf{Q}_k \end{aligned} \tag{B.5}$$

being $(\boldsymbol{\mu}_{\tilde{m}'}, \boldsymbol{\Sigma}_{\tilde{m}'})$ the estimation of the landmark position at the previous time step, and

$$\begin{aligned} \boldsymbol{\mu}_m &= \boldsymbol{\mu}_{\tilde{m}} + \boldsymbol{\Sigma}_{\tilde{m}}\mathbf{H}^{\mathrm{T}} \left(\mathbf{H}\boldsymbol{\Sigma}_{\tilde{m}}\mathbf{H}^{\mathrm{T}} + \boldsymbol{\Sigma}_k\right)^{-1} \left(\boldsymbol{\mu}_k - \mathbf{H}\boldsymbol{\mu}_{\tilde{m}}\right) \\ \boldsymbol{\Sigma}_m &= \left(\mathbf{I} - \boldsymbol{\Sigma}_{\tilde{m}}\mathbf{H}^{\mathrm{T}} \left(\mathbf{H}\boldsymbol{\Sigma}_{\tilde{m}}\mathbf{H}^{\mathrm{T}} + \boldsymbol{\Sigma}_k\right)^{-1} \mathbf{H}\right) \boldsymbol{\Sigma}_{\tilde{m}} \end{aligned} \tag{B.6}$$

Note that the subscript k in matrices $\mathbf{F}$, $\mathbf{B}$ and $\mathbf{H}$ has been dropped, since they are considered to be constant through time. Furthermore, in our work, the transition model and the control action for the landmarks in the map are very simple, since they remain static with time. Therefore, we do not consider $\mathbf{u}_k$ and take $\mathbf{F} = \mathbf{I}$, which entails that the predicted landmark positions are identical to their estimations at the previous time step:

$$\begin{aligned} \boldsymbol{\mu}_{\tilde{m}} &= \boldsymbol{\mu}_{\tilde{m}'} \\ \boldsymbol{\Sigma}_{\tilde{m}} &= \boldsymbol{\Sigma}_{\tilde{m}'} \end{aligned} \tag{B.7}$$

On the other hand, as the observations in our work directly determine the 3D spatial coordinates of the landmarks model, we have $\mathbf{H} = \mathbf{I}$, thereby simplifying equations in (B.6) considerably:

$$\boldsymbol{\mu}_m = \boldsymbol{\mu}_{\tilde{m}} + \boldsymbol{\Sigma}_{\tilde{m}} \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1} \left(\boldsymbol{\mu}_k - \boldsymbol{\mu}_{\tilde{m}}\right) \tag{B.8}$$

$$\boldsymbol{\Sigma}_m = \left(\mathbf{I} - \boldsymbol{\Sigma}_{\tilde{m}} \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1}\right) \boldsymbol{\Sigma}_{\tilde{m}} \tag{B.9}$$

These simplifications can be further transformed in order to get the more compact expressions shown in equations (4.8) in section 4.2.1, as derived next.

Regarding the covariance equation (B.9), we can substitute the identity matrix by $\left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right) \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1}$ and factor out to obtain:

$$\begin{aligned} \boldsymbol{\Sigma}_m &= \left(\left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right) \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1} - \boldsymbol{\Sigma}_{\tilde{m}} \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1}\right) \boldsymbol{\Sigma}_{\tilde{m}} \\ &= \boldsymbol{\Sigma}_k \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1} \boldsymbol{\Sigma}_{\tilde{m}} \end{aligned} \tag{B.10}$$

which is equivalent to:

$$\boldsymbol{\Sigma}_m = \left(\boldsymbol{\Sigma}_k^{-1} + \boldsymbol{\Sigma}_{\tilde{m}}^{-1}\right)^{-1} \tag{B.11}$$

On the other hand, the equation of the mean (B.8) can be expanded as follows:

$$\boldsymbol{\mu}_m = \boldsymbol{\mu}_{\tilde{m}} + \boldsymbol{\Sigma}_{\tilde{m}} \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1} \boldsymbol{\mu}_k - \boldsymbol{\Sigma}_{\tilde{m}} \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1} \boldsymbol{\mu}_{\tilde{m}} \tag{B.12}$$

and by factoring out the $\boldsymbol{\mu}_{\tilde{m}}$ variable, it becomes:

$$\boldsymbol{\mu}_m = \left(\mathbf{I} - \boldsymbol{\Sigma}_{\tilde{m}} \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1}\right) \boldsymbol{\mu}_{\tilde{m}} + \boldsymbol{\Sigma}_{\tilde{m}} \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1} \boldsymbol{\mu}_k \tag{B.13}$$

Now, the identity matrix can be substituted in the same way as with the covariance expression, and by factorizing out again, we obtain:

$$\boldsymbol{\mu}_m = \boldsymbol{\Sigma}_k \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1} \boldsymbol{\mu}_{\tilde{m}} + \boldsymbol{\Sigma}_{\tilde{m}} \left(\boldsymbol{\Sigma}_{\tilde{m}} + \boldsymbol{\Sigma}_k\right)^{-1} \boldsymbol{\mu}_k \tag{B.14}$$

Finally, by using equation (B.10), we can conclude that:

$$\begin{aligned} \boldsymbol{\mu}_m &= \boldsymbol{\Sigma}_m \boldsymbol{\Sigma}_{\tilde{m}}^{-1} \boldsymbol{\mu}_{\tilde{m}} + \boldsymbol{\Sigma}_m \boldsymbol{\Sigma}_k^{-1} \boldsymbol{\mu}_k \\ &= \boldsymbol{\Sigma}_m \left(\boldsymbol{\Sigma}_{\tilde{m}}^{-1} \boldsymbol{\mu}_{\tilde{m}} + \boldsymbol{\Sigma}_k^{-1} \boldsymbol{\mu}_k\right) \end{aligned} \tag{B.15}$$

which, together with equation (B.11), constitute the simplified expressions 4.8 presented in section 4.2.1.

Bibliography

[1] S. Agarwal, N. Snavely, S. M. Seitz, and R. Szeliski. Bundle adjustment in the large. In *European Conference on Computer Vision (ECCV)*, pages 29–42. Springer, 2010.

[2] M. Agrawal and K. Konolige. Real-time localization in outdoor environments using stereo vision and inexpensive gps. In *International Conference on Pattern Recognition (ICPR)*, volume 3, pages 1063–1068. IEEE, 2006.

[3] A. Alahi, R. Ortiz, and P. Vandergheynst. Freak: Fast retina keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–517. IEEE, 2012.

[4] M. Arulampalam, S. Maskell, N. Gordon, T. Clapp, D. Sci, T. Organ, and S. Adelaide. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[5] N. Ayache and F. Lustman. Trinocular stereo vision for robotics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(1):73–85, 1991.

[6] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. In *Proceedings of the 9th European Conference on Computer Vision*, volume 13, pages 404–417, 2006.

[7] S. S. Beauchemin and J. L. Barron. The computation of optical flow. *ACM Computing Surveys (CSUR)*, 27(3):433–466, 1995.

[8] J. S. Beis and D. G. Lowe. Indexing without invariants in 3d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1000–1015, 1999.

[9] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[10] A. Berg, T. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 1, 2005.

[11] P. Besl and N. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[12] J. Blanco. The Mobile Robot Programming Toolkit (MRPT) website, 2008.

[13] J. Blanco, J. González, and J.-A. Fernández-Madrigal. Optimal filtering for non-parametric observation models: applications to localization and slam. *The International Journal of Robotics Research*, 2010.

[14] J.-L. Blanco. A tutorial on se (3) transformation parameterizations and on-manifold optimization. *University of Malaga, Tech. Rep*, 2010.

[15] J.-L. Blanco, J. González-Jiménez, and J.-A. Fernández-Madrigal. Sparser relative bundle adjustment (srba): constant-time maintenance and local optimization of arbitrarily large maps. In *IEEE International Conference on Robotics and Automation*, 2013.

[16] J.-L. Blanco, F.-A. Moreno, and J. Gonzalez. A collection of outdoor robotic datasets with centimeter-accuracy ground truth. *Autonomous Robots*, 27(4):327–351, 2009.

[17] J.-L. Blanco-Claraco, F.-A. Moreno-Dueñas, and J. González-Jiménez. The málaga urban dataset: High-rate stereo and lidar in a realistic urban scenario. *The International Journal of Robotics Research*, 33(2):207–214, 2014.

[18] A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti, and J. D. Tardos. Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 6. IEEE, 2006.

[19] A. Bonarini, D. Migliore, G. Fontana, and M. Matteucci. The Raw Seeds project website, 2009.

[20] K. Borkowski. Transformation of geocentric to geodetic coordinates without approximations. *Astrophysics and Space Science*, 139(1):1–4, 1987.

[21] K. Briechle and U. D. Hanebeck. Template matching using fast normalized cross correlation. In *Aerospace/Defense Sensing, Simulation, and Controls*, pages 95–102. International Society for Optics and Photonics, 2001.

[22] W. Burgard et al. *Estimating the Absolute Position of a Mobile Robot Using Position Probability Grids*. Sekretariat für Forschungsberichte, Inst. für Informatik III, 1996.

[23] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In *IEEE European Conference on Computer Vision (ECCV)*, pages 778–792. Springer, 2010.

[24] J. Campbell, R. Sukthankar, I. Nourbakhsh, and A. Pahwa. A Robust Visual Odometry and Precipice Detection System Using Consumer-grade Monocular Vision. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3421–3427, 2005.

[25] J. A. Castellanos and J. D. Tardos. *Mobile robot localization and map building: A multisensor fusion approach*. Kluwer academic publishers, 2000.

[26] S. Ceriani, G. Fontana, A. Giusti, D. Marzorati, M. Matteucci, D. Migliore, D. Rizzi, D. G. Sorrenti, and P. Taddei. Rawseeds ground truth collection systems for indoor self-localization and mapping. *Autonomous Robots*, 27(4):353–371, 2009.

[27] O. Chum and J. Matas. Randomized ransac with td, d test. In *British Machine Vision Conference (BMVC)*, volume 2, pages 448–457, 2002.

[28] O. Chum and J. Matas. Matching with prosac-progressive sample consensus. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 220–226. Ieee, 2005.

[29] J. Civera, A. Davison, and J. Montiel. Inverse Depth to Depth Conversion for Monocular SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2778–2783, 2007.

[30] J. Civera, A. Davison, and J. Montiel. Inverse depth parametrization for monocular slam. *IEEE Transactions on Robotics*, 24(5):932–945, Oct. 2008.

[31] A. I. Comport, E. Malis, and P. Rives. Accurate quadrifocal tracking for robust 3d visual odometry. In *IEEE International Conference on Robotics and Automation*, pages 40–45. IEEE, 2007.

[32] A. I. Comport, E. Malis, and P. Rives. Real-time quadrifocal visual odometry. *The International Journal of Robotics Research*, 29(2-3):245–266, 2010.

[33] D. S. O. Correa, D. F. Sciotti, M. G. Prado, D. O. Sales, D. F. Wolf, and F. S. Osorio. Mobile robots navigation in indoor environments using kinect sensor. In *Brazilian Conference on Critical Embedded Systems (CBSEC)*, pages 36–41. IEEE, 2012.

[34] I. Cox and J. Leonard. Modeling a dynamic environment using a Bayesian multiple hypothesis approach. *Artificial Intelligence*, 66(2):311–344, 1994.

[35] M. Cummins and P. Newman. Fab-map: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008.

[36] M. Dailey and M. Parnichkun. Landmark-based simultaneous localization and mapping with stereo vision. In *Proceedings of the Asian Conference on Industrial Automation and Robotics*, 2005.

[37] T. A. Davis. *Direct methods for sparse linear systems*, volume 2. Siam, 2006.

[38] A. Davison. Real-time simultaneous localisation and mapping with a single camera. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1403–1410, 2003.

[39] A. Davison. Dyson360, https://www.dyson360eye.com, 2014.

[40] A. Davison, Y. Cid, and N. Kita. Real-time 3D SLAM with wide-angle vision. *Proc. IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon*, 2004.

[41] A. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6), June 2007.

[42] C. De Boor. *A Practical Guide to Splines*. Springer, 2001.

[43] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the CONDENSATION algorithm for robust, vision-based mobile robot localization. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:588–594, 1999.

[44] F. Dellaert and M. Kaess. Square root sam: Simultaneous localization and mapping via square root information smoothing. *The International Journal of Robotics Research*, 25(12):1181–1203, 2006.

[45] J. Diebel. Representing attitude: Euler angles, unit quaternions, and rotation vectors. *Matrix*, 2006.

[46] A. Diosi and L. Kleeman. Laser scan matching in polar coordinates with application to slam. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3317–3322. IEEE, 2005.

[47] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

[48] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 176–183, 2000.

[49] A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

[50] S. Drake. Converting GPS coordinates λ φ h to navigation coordinates (ENU). Technical report, Defence Science and Technology Organisation. Australian Government Department of Defence., 2002.

[51] P. E. Duda and O. Richard. *Hart, Pattern Classification and Scene Analysis*. John Wiley and Sons, New York, 1973.

[52] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part I. *IEEE Robotics & Automation Magazine*, 13(2):99–110, 2006.

[53] E. Eade and T. Drummond. Unified loop closing and recovery for real time monocular slam. In *British Machine Vision Conference (BMVC)*, volume 13, page 136. Citeseer, 2008.

[54] R. Eustice, H. Singh, J. J. Leonard, M. Walter, and R. Ballard. Visually navigating the rms titanic with slam information filters. In *Robotics: Science and Systems*, pages 57–64, 2005.

[55] J.-A. Fernández-Madrigal and J.-L. Blanco. *Simultaneous Localization and Mapping for Mobile Robots: Introduction and Methods*. IGI Global, September 2012.

[56] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[57] C. Forster, M. Pizzoli, and D. Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[58] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte Carlo localization: Efficient position estimation for mobile robots. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, volume 113, page 114, 1999.

[59] D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11(3):391–427, 1999.

[60] J. Fritsch, T. Kuehnl, and A. Geiger. A new performance measure and evaluation benchmark for road detection algorithms. In *International Conference on Intelligent Transportation Systems (ITSC)*, 2013.

[61] A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12(1):16–22, 2000.

[62] D. Galvez-Lopez and J. D. Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, 2012.

[63] N. Ganganath and H. Leung. Mobile robot localization using odometry and kinect sensor. In *IEEE International Conference on Emerging Signal Processing Applications (ESPA)*, pages 91–94. IEEE, 2012.

[64] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361. IEEE, 2012.

[65] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Asian Conference in Computer Vision (ACCV)*, pages 25–38. Springer, 2010.

[66] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 963–968. IEEE, 2011.

[67] A. Gil, Ó. Reinoso, O. Martinez Mozos, C. Stachniss, and W. Burgard. Improving Data Association in Vision-based SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2076–2081, 2006.

[68] J. Gonzalez, C. Galindo, J. Blanco, J. Fernandez-Madrigal, V. Arevalo, and F. Moreno. Sancho, a fair host robot. a description. In *IEEE International Conference on Mechatronics (ICM)*, pages 1–6, April 2009.

[69] J. Gonzalez and R. Gutierrez. Direct motion estimation from a range scan sequence. *Journal of Robotic Systems*, 16(2):73–80, 1999.

[70] J. Gonzalez, A. Muñoz, C. Galindo, J. Fernández-Madrigal, and J. Blanco. A Description of the SENA Robotic Wheelchair. In *Proceedings of the IEEE Mediterranean Electrotechnical Conference*, pages 437–440, 2006.

[71] J. Gonzalez, A. Stentz, and A. Ollero. A mobile robot iconic position estimator using a radial laser scanner. *Journal of Intelligent and Robotic Systems*, 13(2):161–179, 1995.

[72] K. Grauman and T. Darrell. Efficient image matching with distributions of local invariant features. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 627–634. IEEE, 2005.

[73] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard. A tutorial on graph-based slam. *Intelligent Transportation Systems Magazine, IEEE*, 2(4):31–43, 2010.

[74] G. Grisetti, C. Stachniss, and W. Burgard. Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters. *IEEE Transactions on Robotics*, 23:34–46, Feb 2007.

[75] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Robotics: Science and Systems*, 2007.

[76] F. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, pages 1–21, 1969.

[77] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map-buildingalgorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, 17(3):242–257, 2001.

[78] J. Guivant, E. Nebot, J. Nieto, and F. Masson. Navigation and mapping in large unstructured environments. *International Journal of Robotics Research*, 23(4):449–472, 2004.

[79] D. Hahnel, W. Burgard, D. Fox, and S. Thrun. An efficient fastslam algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, 2003.

[80] R. W. Hamming. Error detecting and error correcting codes. *Bell System technical journal*, 29(2):147–160, 1950.

[81] R. A. Hamzah, R. A. Rahim, and Z. M. Noh. Sum of absolute differences algorithm in stereo correspondence problem for stereo matching in computer vision application. In *IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, volume 1, pages 652–657. IEEE, 2010.

[82] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of Alvey Vision Conference*, volume 15, 1988.

[83] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[84] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *International Symposium on Experimental Robotics (ISER*.

[85] J. Hershey and P. Olsen. Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages 317–320, 2007.

[86] B. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4(4):629–642, 1987.

[87] K. Hosoda, K. Sakamoto, and M. Asada. Trajectory generation for obstacle avoidance of uncalibrated stereovisual servoing without 3D reconstruction. *Proceedings of IEEE/RSJ International Conference on the Intelligent Robots and Systems (IROS): 'Human Robot Interaction and Cooperative Robots'*, 1, 1995.

[88] A. Howard. Real-time stereo visual odometry for autonomous ground vehicles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3946–3952. Ieee, 2008.

[89] A. Howard and N. Roy. The robotics data set repository (radish), 2003.

[90] A. S. Huang, M. Antone, E. Olson, L. Fletcher, D. Moore, S. Teller, and J. Leonard. A high-rate, heterogeneous data set from the darpa urban challenge. *The International Journal of Robotics Research*, 29(13):1595–1601, 2010.

[91] P. Huber, E. Ronchetti, and MyiLibrary. *Robust statistics*, volume 1. Wiley Online Library, 1981.

[92] M. Irani and P. Anandan. About direct methods. In *Vision Algorithms: Theory and Practice*, pages 267–277. Springer, 2000.

[93] M. Jaimez-Tarifa and J. Gonzalez-Jimenez. Fast visual odometry for 3d range sensors. *IEEE Transactions on Robotics*, Under review.

[94] M. Jaimez-Tarifa, J. Gonzalez-Jimenez, and J.-L. Blanco. Efficient reactive navigation with exact collision determination for 3d robot shapes. *International Journal of Advanced Robotic Systems*, to appear 2015.

[95] S. Julier. The scaled unscented transformation. In *Proceedings of the American Control Conference*, volume 6, pages 4555–4559, 2002.

[96] S. Julier and J. Uhlmann. A new extension of the Kalman filter to nonlinear systems. *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, 3, 1997.

[97] M. Kaess, A. Ranganathan, and F. Dellaert. Fast incremental square root information smoothing. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, pages 2129–2134, 2007.

[98] M. Kaess, A. Ranganathan, and F. Dellaert. isam: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008.

[99] R. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[100] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–506. IEEE, 2004.

[101] C. Kerl, J. Sturm, and D. Cremers. Robust odometry estimation for rgb-d cameras. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3748–3754. IEEE, 2013.

[102] L. Kleeman. Advanced sonar and odometry error modeling for simultaneous localisation and map building. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 699–704. IEEE, 2003.

[103] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 225–234. IEEE, 2007.

[104] K. Konolige and M. Agrawal. Frameslam: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, 2008.

[105] K. Konolige, M. Agrawal, R. Bolles, C. Cowan, M. Fischler, and B. Gerkey. Outdoor mapping and navigation using stereo vision. In *Experimental Robotics*, pages 179–190. Springer, 2008.

[106] K. Konolige, M. Agrawal, and J. Sola. Large-scale visual odometry for rough terrain. *Robotics Research*, pages 201–212, 2011.

[107] K. Konolige, J. Bowman, J. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *The International Journal of Robotics Research*, 2010.

[108] J. Kosecka and F. Li. Vision based topological Markov localization. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2, 2004.

[109] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[110] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2130–2137. IEEE, 2009.

[111] S. Kullback. *Information Theory and Statistics*. Dover Publications, 1997.

[112] A. Leick. *GPS Satellite Surveying*. Wiley, 2004.

[113] J. Lewis. Fast normalized cross-correlation. In *Vision interface*, volume 10, pages 120–123, 1995.

[114] M. Li, B. Hong, Z. Cai, and R. Luo. Novel Rao-Blackwellized Particle Filter for Mobile Robot SLAM Using Monocular Vision. *International Journal of Intelligent Technology*, 1(1):63–69, 2006.

[115] J. Lim, J.-M. Frahm, and M. Pollefeys. Online environment mapping. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3489–3496. IEEE, 2011.

[116] M. I. Lourakis and A. A. Argyros. Efficient, causal camera tracking in unprepared environments. *Computer Vision and Image Understanding*, 99(2):259–290, 2005.

[117] S. Lovegrove, A. J. Davison, and J. Ibanez-Guzmán. Accurate visual odometry from a rear parking camera. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 788–793. IEEE, 2011.

[118] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, 1999.

[119] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[120] F. Lu and E. Milios. Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans. *Journal of Intelligent and Robotic Systems*, 18(3):249–275, 1997.

[121] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial & Applied Mathematics*, 11(2):431–441, 1963.

[122] L. Matthies and S. Shafer. Error modeling in stereo navigation. *IEEE Journal of Robotics and Automation*, 3(3):239–248, 1987.

[123] S. May, B. Werner, H. Surmann, and K. Pervolz. 3d time-of-flight cameras for mobile robotics. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 790–795. IEEE, 2006.

[124] E. Menegatti, A. Pretto, A. Scarpa, and E. Pagello. Omnidirectional vision scan matching for robot localization in dynamic environments. *IEEE Transactions on Robotics*, 22(3):523–535, 2006.

[125] J. Michels, A. Saxena, and A. Ng. High speed obstacle avoidance using monocular vision and reinforcement learning. In *Proceedings of the 22nd international conference on Machine learning*, volume 2, pages 593–600. ACM Press New York, NY, USA, 2005.

[126] K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1615–1630, 2005.

[127] M. Montemerlo. *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem With Unknown Data Association*. PhD thesis, University of Washington, 2003.

[128] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 593–598, 2002.

[129] J. Montiel, J. Civera, and A. Davison. Unified Inverse Depth Parametrization for Monocular SLAM. *Robotics Science and Systems*, 2006.

[130] E. F. Moore. *The shortest path through a maze*. Bell Telephone System., 1959.

[131] J. Mor. The Levenberg-Marquardt algorithm: implementation and theory. *Lecture Notes in Mathematics*, 630:105–116, 1977.

[132] F. Moreno, J. Blanco, and J. Gonzalez. An efficient closed-form solution to probabilistic 6d visual odometry for a stereo camera. In *Advanced Concepts for Intelligent Vision Systems*, volume 4678 of *LNCS*, pages 932–942. Springer, 2007.

[133] F. A. Moreno, J.-L. Blanco, and J. Gonzalez. Stereo vision specific models for particle filter-based slam. *Robotics and Autonomous Systems*, 57(9):955–970, 2009.

[134] F.-A. Moreno, J.-L. Blanco, and J. González-Jiménez. Erode: An efficient and robust outlier detector and its application to stereovisual odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4691–4697. IEEE, 2013.

[135] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, pages 331–340, 2009.

[136] K. Murphy. Bayesian map learning in dynamic environments. *Advances in Neural Information Processing Systems (NIPS)*, 12:1015–1021, 1999.

[137] D. Murray and J. Little. Using Real-Time Stereo Vision for Mobile Robot Navigation. *Autonomous Robots*, 8(2):161–171, 2000.

[138] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2320–2327. IEEE, 2011.

[139] P. Newman. On the structure and solution of the simultaneous localisation and map building problem. *Doctoral diss., University of Sydney*, 1999.

[140] P. M. Newman, J. J. Leonard, and R. J. Rikoski. Towards constant-time slam on an autonomous underwater vehicle using synthetic aperture sonar. In *Robotics Research*, pages 409–420. Springer, 2005.

[141] K. Ni and F. Dellaert. Stereo tracking and three-point/one-point algorithms-a robust approach in visual odometry. In *IEEE International Conference on Image Processing*, pages 2777–2780. IEEE, 2006.

[142] J. Nieto, J. Guivant, E. Nebot, and S. Thrun. Real time data association for FastSLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, 2003.

[143] D. Nistér. Preemptive ransac for live structure and motion estimation. *Machine Vision and Applications*, 16(5):321–329, 2005.

[144] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages I–652. IEEE, 2004.

[145] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1):3–20, 2006.

[146] I. Ohya, A. Kosaka, and A. Kak. Vision-based navigation by a mobile robot with obstacle avoidance using single-camera vision and ultrasonic sensing. *IEEE Transactions on Robotics and Automation*, 14(6):969–978, 1998.

[147] C. Olson, L. Matthies, M. Schoppers, and M. Maimone. Rover navigation using stereo ego-motion. *Robotics and Autonomous Systems*, 43(4):215–229, 2003.

[148] C. F. Olson, L. H. Matthies, M. Schoppers, and M. W. Maimone. Robust stereo ego-motion for long distance navigation. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 453–458. IEEE, 2000.

[149] E. Olson and P. Agarwal. Inference on networks of mixtures for robust robot mapping. *The International Journal of Robotics Research*, 32(7):826–840, 2013.

[150] E. Olson, J. Leonard, and S. Teller. Fast iterative alignment of pose graphs with poor initial estimates. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2262–2269. IEEE, 2006.

[151] T. Oskiper, Z. Zhu, S. Samarasekera, and R. Kumar. Visual odometry system using multiple stereo cameras and inertial measurement unit. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007.

[152] D. V. Ouellette. Schur complements and statistics. *Linear Algebra and its Applications*, 36:187–295, 1981.

[153] G. Pandey, J. R. McBride, and R. M. Eustice. Ford campus vision and lidar data set. *The International Journal of Robotics Research*, 30(13):1543–1552, 2011.

[154] L. M. Paz, J.Guivant, J. D. Tardós, and J. Neira. Data association in *O(n)* for divide and conquer SLAM. In *Robotics: Science and Systems,RSS*, Atlanta, GA, USA, June 2007.

[155] K. Petersen and M. Pedersen. The matrix cookbook, nov 2012. *URL http://www2. imm. dtu. dk/pubdb/p. php*, 3274.

[156] T. Peynot, S. Scheding, and S. Terho. The marulan data sets: Multi-sensor perception in a natural environment with challenging conditions. *The International Journal of Robotics Research*, 29(13):1602–1607, 2010.

[157] P. Pinies, T. Lupton, S. Sukkarieh, and J. Tardós. Inertial Aiding of Inverse Depth SLAM using a Monocular Camera. *IEEE International Conference on Robotics and Automation*, pages 2797–2802, 2007.

[158] D. L. Ripley and T. Politzer. Vision disturbance after tbi. *NeuroRehabilitation*, 27(3):215–216, 2010.

[159] B. Ristic, S. Arulampalam, and N. Gordon. *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House, 2004.

[160] P. L. Rosin. Measuring corner properties. *Computer Vision and Image Understanding*, 73(2):291–307, 1999.

[161] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision (ECCV)*, pages 430–443. Springer Berlin Heidelberg, 2006.

[162] E. Rosten, R. Porter, and T. Drummond. Faster and better: A machine learning approach to corner detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):105–119, 2010.

[163] S. Roumeliotis and G. Bekey. Bayesian estimation and Kalman filtering: a unified framework formobile robot localization. *IEEE International Conference on Robotics and Automation*, 3, 2000.

[164] E. Royer, M. Lhuillier, M. Dhome, and J.-M. Lavest. Monocular vision for mobile robot localization and autonomous navigation. *International Journal of Computer Vision*, 74(3):237–260, 2007.

[165] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to sift or surf. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2564–2571. IEEE, 2011.

[166] S. Russell and P. Norvig. *Artificial intelligence: a modern approach, chapter 14*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1995.

[167] K. Sabe, M. Fukuchi, J. Gutmann, T. Ohashi, K. Kawamoto, and T. Yoshigahara. Obstacle avoidance and path planning for humanoid robots using stereo vision. In *Proccedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, 2004.

[168] B. Schiele, J. Crowley, and G. LIFIA-IMAG. A comparison of position estimation techniques using occupancygrids. In *Proceedings of the IEEE International Conference on Robotics and Automation.*, pages 1628–1634, 1994.

[169] F. Scholz. Maximum likelihood estimation. *Encyclopedia of statistical sciences*, 1985.

[170] S. Se, D. Lowe, and J. Little. Local and global localization for mobile robots using visuallandmarks. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, 2001.

[171] S. Se, D. Lowe, and J. Little. Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks. *The International Journal of Robotics Research*, 21(8):735, 2002.

[172] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *IEEE International Conference on Computer Vision*, pages 750–757. IEEE, 2003.

[173] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[174] H.-Y. Shum, Q. Ke, and Z. Zhang. Efficient bundle adjustment with virtual key frames: A hierarchical approach to multi-frame structure from motion. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2. IEEE, 1999.

[175] D. Sibley, C. Mei, I. Reid, and P. Newman. Adaptive relative bundle adjustment. In *Robotics: science and systems*, 2009.

[176] G. Sibley. Relative bundle adjustment. *Department of Engineering Science, Oxford University, Tech. Rep*, 2307(09), 2009.

[177] G. Sibley, L. Matthies, and G. Sukhatme. Sliding window filter with application to planetary landing. *Journal of Field Robotics*, 27(5):587–608, 2010.

[178] R. Sim, P. Elinas, M. Griffin, and J. Little. Vision-based SLAM using the Rao-Blackwellised particle filter. *IJCAI Workshop on Reasoning with Uncertainty in Robotics (RUR)*, 2005.

[179] R. Sim, P. Elinas, M. Griffin, A. Shyr, and J. Little. Design and analysis of a framework for real-time vision-based SLAM using Rao-Blackwellised particle filters. In *Proceedings of the 3rd Canadian Conference on Computer and Robot Vision*, pages 21–29, 2006.

[180] C. C. Slama, C. Theurer, S. W. Henriksen, et al. *Manual of photogrammetry.* Number Ed. 4. American Society of photogrammetry, 1980.

[181] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman. The new college vision and laser data set. *The International Journal of Robotics Research*, 28(5):595–599, 2009.

[182] R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. *The fourth international symposium on Robotics Research*, pages 467–474, 1988.

[183] C. Stachniss, G. Grisetti, and W. Burgard. Recovering Particle Diversity in a Rao-Blackwellized Particle Filter for SLAM After Actively Closing Loops. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2005.

[184] H. Strasdat, J. Montiel, and A. J. Davison. Real-time monocular slam: Why filter? In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2657–2664. IEEE, 2010.

[185] H. Strasdat, J. M. Montiel, and A. J. Davison. Visual slam: why filter? *Image and Vision Computing*, 30(2):65–77, 2012.

[186] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 573–580. IEEE, 2012.

[187] N. Sunderhauf and P. Protzel. Towards a robust back-end for pose graph slam. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1254–1261. IEEE, 2012.

[188] H. Tamimi, H. Andreasson, A. Treptow, T. Duckett, and A. Zell. Localization of mobile robots with omnidirectional vision using Particle Filter and iterative SIFT. *Robotics and Autonomous Systems*, 54:758–765, 2006.

[189] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, September 2005.

[190] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2):99–141, 2001.

[191] B. Tordoff and D. Murray. Guided-mlesac: Faster image transform estimation by using matching priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1523–1535, 2005.

[192] P. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138–156, 2000.

[193] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

[194] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment—a modern synthesis. In *Vision algorithms: theory and practice*, pages 298–372. Springer, 2000.

[195] D.-M. Tsai and C.-T. Lin. Fast normalized cross correlation for defect detection. *Pattern Recognition Letters*, 24(15):2625–2631, 2003.

[196] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: a survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3):177–280, 2008.

[197] T. Tykkala, C. Audras, and A. I. Comport. Direct iterative closest point for real-time visual odometry. In *IEEE International Conference on Computer Vision Workshops*, pages 2050–2056. IEEE, 2011.

[198] I. Ulrich and I. Nourbakhsh. Appearance-Based Place Recognition for Topological Localization. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1023–1029, 2000.

[199] S. Vassiliadis, E. A. Hakkennes, J. Wong, and G. G. Pechanek. The sum-absolute-difference motion estimation accelerator. In *Euromicro Conference, 1998. Proceedings. 24th*, volume 2, pages 559–566. IEEE, 1998.

[200] M. Walter, R. Eustice, and J. Leonard. A provably consistent method for imposing sparsity in feature-based SLAM information filters. In *International Symposium of Robotics Research (ISRR)*. Springer, 2004.

[201] E. Wan and R. Van Der Merwe. The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pages 153–158, 2000.

[202] H. Wang, K. Yuan, W. Zou, and Q. Zhou. Visual odometry based on locally planar ground assumption. In *Proceedings of the IEEE International Conference on Information Acquisition*, pages 59–64, 2005.

[203] G. Weiss, C. Wetzler, and E. von Puttkamer. Keeping track of position and orientation of moving indoor systemsby correlation of range-finder scans. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, 1994.

[204] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in neural information processing systems*, pages 1753–1760, 2009.

[205] G. Welch and G. Bishop. An introduction to the kalman filter, 1995.

[206] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard, and J. McDonald. Robust real-time visual odometry for dense rgb-d mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5724–5731. IEEE, 2013.

[207] J. Wolf, W. Burgard, and H. Burkhardt. Robust vision-based localization for mobile robots using an image retrieval system based on invariant features. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, 2002.

[208] B. Yamauchi and R. Beer. Spatial learning for navigation in dynamic environments. *IEEE Transactions on Systems, Man and Cybernetics, Part B,*, 26(3):496–505, 1996.

[209] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000.

[210] Z. Zhang and Y. Shan. Incremental motion estimation through modified bundle adjustment. In *International Conference on Image Processing (ICIP)*, volume 2, pages II–343. IEEE, 2003.