**UNIVERSIDAD DE CHILE**
**FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS**
**DEPARTAMENTO DE INGENIERÍA ELÉCTRICA**

# GENERACIÓN AUTOMÁTICA DE LANDMARKS VISUALES NATURALES TRIDIMENSIONALES PARA SLAM VISUAL

**TESIS PARA OPTAR AL GRADO DE**
**DOCTOR EN INGENIERÍA ELÉCTRICA**

**PATRICIO ALEJANDRO LONCOMILLA ZAMBRANA**

PROFESOR GUÍA:
JAVIER RUIZ DEL SOLAR SAN MARTIN

MIEMBROS DE LA COMISION:
PEDRO NUÑEZ TRUJILLO
ALVARO SOTO ARRIAZA
MARCOS ORCHARD CONCHA

SANTIAGO DE CHILE
MAYO 2011

## Resumen

En los métodos actuales de SLAM visual, los mapas son representados mediante landmarks puntuales. Como la observación de un landmark puntual entrega sólo información angular sobre la cámara, se debe estimar una matriz de covarianza que considere todos los puntos para poder converger a una escala global. Sin embargo, como la complejidad computacional asociada al trabajo con matrices de covarianza crece de una forma cuadrática respecto al número de landmarks, la cantidad máxima de puntos con los cuales se puede trabajar n tiempo real está limitada a unos cientos. En este trabajo se propone un sistema de SLAM visual basado en el uso de los denominados *landmarks cuerpo rígido*. Un landmark cuerpo rígido representa la pose completa 6D (posición y orientación) de un cuerpo rígido en el espacio, y la observación de uno de estos landmarks proporciona información completa acerca de la pose de una cámara móvil. Cada landmarks cuerpo rígido es creado a partir de un conjunto de N landmarks puntuales mediante el colapso de 3N componentes del estado en 7 nuevas componentes del estado, además de un conjunto de parámetros que describe la forma del landmark. Los landmark cuerpo rígido son representados y estimados usando *puntocuaterniones*, los cuales son introducidos en este trabajo. Mediante el uso de los landmarks cuerpo rígido, el tiempo de ejecución del sistema de SLAM puede reducirse hasta un 5.5% a medida que la cantidad de landmarks aumenta. El sistema SLAM propuesto es validado y simulado en secuencias de video reales. El método propuesto puede ser extendido a cualquier sistema de SLAM que se base en el uso de landmarks puntuales, incluyendo aquellos generados mediante sensores láser.

## Abstract

In current visual SLAM methods, point-like landmarks are used for representation on maps. As the observation of each point-like landmark gives only angular information about a bearing camera, a covariance matrix between point-like landmarks must be estimated in order to converge with a global scale estimation. However, as the computational complexity of covariance matrices scales in a quadratic way with the number of landmarks, the maximum number of landmarks that is possible to use is normally limited to a few hundred. In this paper, a visual SLAM system based on the use of what are called rigid-body 3D landmarks is proposed. A rigid-body 3D landmark represents the 6D pose of a rigid body in space (position and orientation), and its observation gives full-pose information about a bearing camera. Each rigid-body 3D landmark is created from a set of N point-like landmarks by collapsing 3N state components into 7 state components plus a set of parameters that describe the shape of the landmark. Rigid-body 3D landmarks are represented and estimated using so-called point-quaternions, which are introduced here. By using rigid-body 3D landmarks, the computational time of an EKF-SLAM system can be reduced up to 5.5%, as the number of landmarks increases. The proposed visual SLAM system is validated in simulated and real video sequences (outdoor). The proposed methodology can be extended to any SLAM system based on the use of point-like landmarks, including those generated by laser measurement.

# Agradecimientos

El desarrollo de esta tesis requirió generar conceptos y métodos nuevos asociados al problema de la representación del entorno mediante su fragmentación en regiones compactas, además de una implementación computacional orientada a validar su aplicación práctica en el problema de la localización robótica en ambientes desconocidos. La forma de abordar el problema, así como la comprensión del mismo, eran inicialmente vagos y sólo fueron aclarándose mediante un proceso que involucró instrospección, lectura y tiempo. Durante el desarrollo de este trabajo hubo momentos de éxito y logros, pero también de dudas y frustraciones. Sin embargo, gracias a la comprensión, afecto, orientación y confianza incondicional de muchas personas pude lograr la estabilidad, enfoque y claridad necesarios para completar el proyecto que me había propuesto. Tengo mucho que agradecer, y muchos a quienes agradecer, y lo que quisiera expresar se extiende mucho más allá que lo que permiten unas cuantas hojas de papel.

En primer lugar quiero agradecer a mi familia: a mi papá, mi mamá, mis hermanos Carlos, Sebas y Leny por preocuparse por mí y ayudarme en todo momento. Los quiero y son imprescindibles para mí.

Quiero agradecer también de forma especial al profesor Javier Ruiz del Solar, quien fue capaz de orientarme y apoyarme durante todo el desarrollo de la tesis, incluso en los momentos en que el avance se veía difícil.

He tenido la suerte de conocer muchas personas que me han acompañado y apoyado en diversos momentos: a los compañeros del laboratorio de visión computacional: Freddy, Marcelo, Gabriél, Rodrigo, José y Claudio, además de Jaime, Marcela y Bernardita; a los compañeros del laboratorio de robótica, entre ellos Paul, Pablo y Rodrigo; a Eliana Monardes por la enorme ayuda que me ha entregado, a Rafael Donoso Sarovic, a Angélica Muñoz; a los integrantes y exintegrantes de mi grupo Ajiaco: Meme, Waldo, Pancho, Chamu, Leo, Lucho, Momo, Diego Roda, Francisco Durán y a Diego Mora; a Pilar y Carolina; a Marcelo Troncoso por apoyarme y enseñarme a comprender mejor la música; a mis primos Mariano, Fabiola y Lily; a mi grupo de amigos de siempre: Moris, Loreto, Pablo,

Odette, Miguel, Noland, Carola, Rocío, Renato y Perla; y a mis amigos de San Antonio Rorro y Elsa. También quiero agradecer a las personas que me ayudaron a corregir errores en la tesis: a Álvaro Soto, a Pedro Nuñez y a Marcos Orchard, quien hizo aportes significativos para aclarar explicaciones conceptuales y mejorar la redacción.

Finalmente, quiero agradecer al departamento de Ingeniería Eléctrica de la Universidad de Chile, al centro AMTC, a Conicyt y al Ministerio de Educación, que son instituciones que ayudan a generar y sostener el desarrollo tecnológico y de investigación en Chile. Quiero agradecer el apoyo recibido de parte del proyecto de becas de doctorado Conicyt y de parte de las iniciativas MECESUP FSM0601, FONDECYT 1090250 y TIC-AMSUD "ARVS Hybrid model based markerless 3D tracking for augmented reality and visual servoing", los cuales permitiron financiar parte de mi trabajo, asistencias a congresos y estadías que ayudaron al desarrollo de mi tesis.

# Tabla de contenidos

# 1 Introducción

## 1.1 Breve introducción al tema

El tema principal de esta tesis es la generación y posterior reconocimiento visual de landmarks naturales basados en un modelo de cuerpo rígido usando secuencias de video monoculares y binoculares. Un landmark[1] corresponde a un modelo de una estructura existente en una escena tridimensional, el cual puede ser usado como punto de referencia para un robot móvil. Una cámara móvil, al moverse entre diferentes posiciones y orientaciones, genera varias imágenes distintas de la escena en las que aparece el landmark. Cuando las observaciones que se obtienen de un landmark son obtenidas a partir de imágenes capturadas por una cámara, se dice que el landmark es visual. Por otro lado, cuando el landmark corresponde a un objeto que no ha sido colocado artificialmente en la escena, sino que se encuentra naturalmente en ella, se dice que es un landmark natural.

Una propiedad que se desea para el sistema de generación y reconocimiento de landmarks es que pueda trabajar con escenarios de tipos muy diversos que no estuvieron disponibles durante el desarrollo del sistema. Es por esto que el sistema no puede limitarse a reconocer un número limitado de tipos de landmarks que sean resultado de un proceso de entrenamiento previo, sino que los landmarks que se van a reconocer deben ser generados y caracterizados durante su operación por el mismo sistema a partir de las imágenes adquiridas, lo que requiere un alto grado de versatilidad. Para que el sistema sea capaz de funcionar adecuadamente usando imágenes provenientes de una amplia variedad

[1] Un landmark es definido en el diccionario Oxford como un objeto, característica geográfica o ciudad que puede ser vista y reconocida fácilmente a la distancia, especialmente una que le permite a una persona establecer su localización. Puede ser traducido al español de múltiples formas tales como marca, poste, hito, etc., aunque ninguna de estas traducciones preserva fielmente el significado original de la palabra. En este trabajo se utilizará el término *landmark* debido a que es el término estándar usado en las publicaciones sobre SLAM.

de escenarios, se eligieron los descriptores locales para ser usados como características básicas, ya que aparecen en cualquier imagen que contenga información de textura visual. Los descriptores locales corresponden a vectores de características, cada uno de los cuales representa el contenido de una pequeña zona rectangular de la imagen. Dichas zonas rectangulares son elegidas mediante algún algoritmo que no es afectado por algunas familias de transformaciones que son aplicadas sobre la imagen al cambiar el punto de vista de la cámara y la iluminación ambiental, tales como rotaciones, traslaciones, cambios de escala y transformaciones lineales en el histograma. Por lo anterior, cada descriptor local es capaz de describir una pequeña zona de algún objeto tridimensional, manteniéndose solidario al objeto independientemente de la posición y orientación con la cual la cámara adquiera la imagen.

Otra propiedad que se desea para el sistema de generación y reconocimiento de landmarks es que pueda funcionar adecuadamente en escenarios amplios, es decir, escenarios de gran tamaño y complejidad, los cuales no pueden ser completamente observados desde una única posición del robot. Para lograr este objetivo se deben cumplir dos características: que el robot sea capaz de localizarse observando muy pocos landmarks, y que los landmarks sean muy distintivos de modo que, cuando se reconozca un landmark, el sistema sepa sin confusión si corresponde a alguno de los ya generados o si es uno nuevo que debe agregarse a los ya existentes, a pesar de que puedan existir una gran cantidad de landmarks almacenados. Cada descriptor local puede considerarse un landmark poco distintivo en el sentido de que es una zona del espacio que se proyecta en las distintas vistas de la escena, pero que no puede distinguirse del resto lo suficientemente bien como para servir de landmark en escenarios amplios. Es por esto que se desea generar landmarks muy distintivos, en el sentido de que cada uno de ellos pueda ser reconocido dentro de una gran cantidad de landmarks existentes. Para lograr el propósito anterior se usarán agrupaciones de descriptores, los cuales son capaces de representar en conjunto la textura visual del objeto.

La elección de los descriptores locales como características elementales se debe básicamente a dos razones. La primera es que los descriptores locales aparecen en cualquier imagen que tenga una cantidad aceptable de textura visual, lo que va a permitir obtener la versatilidad deseada para el sistema. La segunda es que, al corresponder a la proyección de una zona del espacio, entregan información espacial acerca del landmark que puede ser explotada por el sistema. La principal limitación de los sistemas basados en descriptores locales es que las imágenes que contienen áreas grandes de luminosidad uniforme no generan ningún descriptor local. Cuando la cámara captura solamente imágenes con una luminosidad uniforme, como por ejemplo las provenientes de una pared en blanco, no aparecen descriptores locales y no se puede detectar ningún landmark, razón por la cual una correcta localización de la cámara se vuelve imposible.

## 1.2 Motivación

La idea que motiva esta tesis es la búsqueda de un sistema de SLAM[2] (localización y construcción de mapa simultáneos) que use objetos tridimensionales para describir el mundo en vez de usar puntos aislados. Los seres humanos y los demás animales que disponen de sistemas de visión avanzados usan áreas extensas del mundo real para poder localizarse, desde objetos enormes como edificios hasta algunos objetos pequeños como mesas, sillas y otros muebles. El hecho anterior les permite tener un excelente nivel de seguridad al decidir cuáles objetos son aquellos que están mirando, lo cual no es posible usando puntos aislados. Las áreas extensas observadas suelen corresponder a objetos rígidos que existen en el espacio. La observación de un único objeto (una mesa por ejemplo) le permite a un ser humano localizarse en el espacio, ya que puede suponer su posición y orientación respecto al objeto; lo que no es posible en el caso en que se observen puntos aislados ya que se requiere una gran cantidad de puntos para intuir la posición y orientación de un robot. Por otro lado, cuando un ser humano está perdido y entra a un ambiente conocido

[2] SLAM: *Simultaneous Localization And Mapping*.

puede saber rápidamente en qué lugar está debido a que los objetos que observa le permiten suponer una posición y orientación relativa a todos ellos, lo cual no es posible usando puntos aislados, ya que cada punto aislado observado entrega una información muy débil como para poder suponer la localización del robot. Estos hechos motivan el desarrollo de una técnica que le permita a un robot usar cuerpos rígidos (conjuntos indeformables de puntos) como los elementos básicos usados en su localización y en la construcción del mapa, de modo que al observar un cuerpo rígido pueda suponer inmediatamente su posición y orientación en el espacio[3]. En los experimentos realizados, un robot se mantiene bien localizado en su entorno usando sólo cuatro cuerpos rígidos, lo cual no se puede lograr usando sólo cuatro puntos como landmarks. Lo anterior es posible porque la observación de un landmark cuerpo rígido entrega una mayor cantidad de información al sistema de localización cuando es observado que un landmark puntual.

## 1.3 Objetivos

Los objetivos de la tesis están relacionados con la generación de landmarks naturales tridimensionales a partir de secuencias de imágenes y con su aplicación a la localización y construcción de mapas (SLAM) de robots móviles.

### Objetivos generales

- Proponer una nueva técnica que permita generar landmarks tridimensionales naturales, usando descriptores locales a partir de distintas vistas de una escena tridimensional.
- Implementar la técnica como un sistema automatizado y evaluar su desempeño de acuerdo a diversos criterios, tales como velocidad, precisión y robustez frente a variaciones en la escena, en la iluminación y en la pose del robot; usando bases de datos disponibles públicamente y robots móviles

[3] Pose estimada de la cámara: $\eta_{CAM-MAPA} = \eta_{LAND-MAPA} \cdot h(x)^{-1}$ (ver ecuación 156)

reales, comparándolo además con otros sistemas de generación de landmarks ya existentes.

**Objetivos específicos**

1. Diseño de un sistema que permita encontrar transformaciones de una imagen a otra basado en descriptores locales visuales.
2. Diseño e implementación de una nueva técnica que permita generar y reconocer landmarks tridimensionales mediante agrupación de descriptores locales visuales.
3. Diseño de una nueva técnica SLAM que utilice landmarks tridimensionales basados en descriptores locales para representar el mapa.
4. Implementación del sistema SLAM, el que debe funcionar en tiempo real respecto a la velocidad de desplazamiento de un robot móvil.

## 1.4 Hipótesis

Las hipótesis del presenta trabajo de tesis son las siguientes:

1. Es posible agrupar automáticamente y en tiempo real los descriptores locales de una escena tridimensional en landmarks tridimensionales que pueden ser reconocidos visualmente de un modo más robusto que los descriptores individualmente.

2. El uso de descriptores locales en imágenes permite la extracción automatizada de landmarks naturales, los cuales permiten a un robot reconocer su entorno y localizarse en él.

3. Los landmarks tridimensionales permiten al robot obtener una hipótesis de su posición y orientación en el mapa mediante una única observación.

4. Un sistema de SLAM es capaz de usar adecuadamente la información de pose relativa entregada por los landmarks tridimensionales para poder mantener localizado al robot y construir el mapa.

## 1.5 Estructura del documento

En el Capítulo 2 se detallan los aportes hechos durante el desarrollo de esta tesis, los cuales abarcan áreas desde de las matemáticas hasta robótica móvil. En el Capítulo 3 se describe el trabajo previo desarrollado en áreas relacionadas con esta tesis, en particular aquellos trabajos significativos relacionados con visión computacional, con construcción de mapas y localización simultáneos (SLAM) y el estado del arte en SLAM visual. En el Capítulo 4 se explicarán una gran cantidad de conceptos básicos, de modo que una persona que posea sólo conocimientos matemáticos generales pueda comprender los planteamientos hechos en la tesis. En el Capítulo 5 se explica el sistema de SLAM visual propuesto. Finalmente, en el Capítulo 6 se mostrarán resultados experimentales, los cuales permiten concluir que un sistema SLAM es capaz de utilizar convenientemente la información de posición y orientación relativas que entrega la observación de los landmarks rígidos.

# 2 Contribuciones

Durante el desarrollo de esta tesis se realizaron contribuciones en varias áreas del conocimiento, las cuales se detallan a continuación.

## 2.1 Contribuciones en el ámbito de estructuras algebraicas

Se define un nuevo grupo algebraico denominado “grupo de los puntocuaterniones”, el cual permite representar rotaciones y traslaciones en el espacio. El grupo está formado por elementos matemáticos de 7 dimensiones, denominados puntocuaterniones y por una operación de multiplicación que permite realizar composiciones de éstos, además de contar con una operación de potencia y raíz. Al demostrar que esta estructura matemática es un grupo, se pueden realizar cálculos que involucren multiplicaciones e inversas, cuya existencia está asegurada para cualquier puntocuaternión válido. Los puntocuaterniones permiten representar adecuadamente la pose de un landmark rígido en el espacio y su incertidumbre mediante una matriz de covarianza, lo cual es imprescindible para poder utilizar los landmarks rígidos - introducidos en este trabajo - en un sistema SLAM. Los puntocuaterniones permiten representar la posición y orientación de una forma integrada, sin sufrir los problemas asociados a otras representaciones como la deformación que pueden producir las matrices homogéneas, la asimetría y la pérdida de un grado de libertad que pueden producir los ángulos de Euler. En consecuencia, los puntocuaterniones son un aporte al área del álgebra y pueden ser usados en el futuro en otras aplicaciones. Las contribuciones relacionadas a esta área se encuentran detalladas en la Sección 4.3.10 y en el Anexo.

## 2.2 Contribuciones en el ámbito de la representación de landmarks

En este trabajo se define el concepto de landmarks rígidos tridimensionales, los que están compuestos por un conjunto indeformable de puntos denominados "puntos del cuerpo del landmark". La pose del landmark rígido tridimensional se puede especificar mediante un puntocuaternión. La observación de un landmark rígido corresponde a la pose relativa entre dicho landmark y la cámara, lo cual permite generar una hipótesis acerca de la pose del robot y la covarianza respectiva usando sólo una observación, siendo esto un aporte significativo al área de robótica móvil. Las contribuciones relacionadas con esta área se encuentran detalladas en las secciones 5.6, 5.7, 5.11 y 5.12.

## 2.3 Contribuciones en SLAM visual

Se generó una metodología para integrar los landmarks rígidos tridimensionales a un sistema de SLAM visual que estima inicialmente landmarks puntuales y transforma grupos de éstos en landmarks rígidos cuando ciertas condiciones sobre la covarianza se satisfacen. Las contribuciones relacionadas con esta área se encuentran detalladas en las secciones 5.6, 5.7, 5.11 y 5.12. La metodología diseñada involucra los siguientes aportes:

- El sistema reduce el modelo dinámico del sistema SLAM cada vez que se genera un landmark rígido, ya que transforma 10 landmarks puntuales (que usan 30 componentes en el vector de estado) en un único landmark rígido (que usa 7 componentes en el vector de estado). De este modo, la complejidad del SLAM se reduce de $O(n^2)$ a $0.054\,O(n^2)$, es decir, se puede lograr un ahorro asintótico del 94,5% en el tiempo de cómputo a medida que el número de landmarks crece, lo cual permite el uso de mapas de mayor tamaño.
- El sistema de SLAM visual es capaz de realizar reducción del modelo dinámico en ambientes no estructurados; esta característica no está presente en ningún trabajo previo del área.

- El sistema de reducción del modelo dinámico es independiente del tipo de sensor, en el sentido de puede ser usado con cualquier tipo de landmark puntual, por lo cual se puede extender a landmarks obtenidos usando sensores como láser, sonar o radar.
- El sistema de SLAM visual implementado es capaz de usar convenientemente la información de pose completa y covarianza de la pose que genera cada observación de un landmark rígido. Esto permite que un robot se pueda mantener localizado en un mapa que está formado por sólo 4 landmarks rígidos, algo imposible de realizar usando solo landmarks puntuales. De este modo, se pueden generar mapas de grandes dimensiones al necesitarse una densidad baja de landmarks distribuidos en el espacio. Todo lo anterior es adecuadamente validado en esta tesis.
- Se diseñó una forma de analizar la matriz de covarianza usando imágenes integrales, una técnica computacionalmente muy eficiente que permite evaluar rápidamente la conveniencia de transformar los posibles grupos de landmarks puntuales en landmarks rígidos. La eficiencia de dicho análisis es una condición necesaria para que el uso de los landmarks rígidos no afecte negativamente la velocidad del sistema. Este enfoque de análisis rápido de la matriz de covarianza puede ser extendido a otros problemas.
- Se diseñó un procedimiento de agrupación que permite transformar la covarianza de los landmarks puntuales que se agrupan en una matriz de covarianza de la pose del landmark rígido y en matrices de covarianza para cada uno de los puntos del cuerpo del landmark rígido. El criterio diseñado permite minimizar la pérdida de información que ocurre durante la reducción del modelo y fue validado experimentalmente.
- El conjunto de descriptores asociado a cada landmark rígido permite su diferenciación, respecto de los demás landmarks existentes en el entorno, con gran certeza. Esta es una de las características necesarias para lograr que un solo landmark rígido sea capaz de generar una hipótesis de pose confiable para el sistema de SLAM.

## 2.4 Publicaciones relacionadas con la tesis

La publicación principal que incluye los aspectos principales de la tesis es la siguiente [53]:

- P. Loncomilla and J. Ruiz del Solar, “Visual SLAM based on Rigid-Body 3D landmarks”, Journal of Intelligent and Robotic Systems, Springer (aceptado)

Existen otras publicaciones anteriores relacionadas con la tesis en los cuales se define y mejora progresivamente el sistema L&R (Loncomilla & Ruiz del Solar) de tal forma que sea capaz de resolver distintos tipos de problemas de reconocimiento de objetos en imágenes [49][50][51][52][53][78][79][80][81][82][83]. El sistema L&R se usa en el sistema SLAM visual propuesto para hacer la asociación entre descriptores SURF obtenidos de la imagen y los landmarks del mapa. Los descriptores SURF se pueden calcular con mucha rapidez pero pueden aparecer sobre líneas, en consecuencia sus posiciones pueden variar mucho entre una imagen y la siguiente, lo cual produce un impacto negativo en la calidad de las observaciones. Debido a la dificultad de obtener un sistema de asociación de datos rápido y de alta calidad, la gran mayoría de los trabajos relacionados con SLAM visual usan tracking de puntos y no detección en cada imagen. Usando sólo tracking, es imposible detectar los landmarks rígidos que aparecen en la escena, por lo cual las mejoras introducidas al sistema L&R son imprescindibles.

También hay publicaciones anteriores relacionadas con la detección de la pose de un objeto en el espacio usando ángulos de Euler a partir de varias vistas del objeto [49][82][83]. Sin embargo, estos trabajos previos son incompatibles con un sistema SLAM, ya que aún no estaba definido el criterio de agrupación, la metodología de agrupación, ni la forma de generar las observaciones y sus covarianzas. Esto se debe a que para poder definir claramente estas etapas se requieren los puntocuaterniones y la base teórica relacionada, las cuales no existían en ese momento.

# 3 Trabajo relacionado

El objetivo de este capítulo es introducir al lector al desarrollo de los métodos matemáticos y de visión computacional relacionados con la Tesis. Si el lector ya conoce estos temas, se recomienda el paso directo a la Sección 3.3 estado del arte, la cual describe los últimos métodos de SLAM visual existentes al momento de la publicación de este trabajo.

## 3.1 Descriptores locales en imágenes

Dada una imagen, existen algoritmos que permiten seleccionar un conjunto de puntos que se repiten en las mismas zonas de ésta, aunque ésta sea sometida a transformaciones geométricas o a cambios en su histograma. Estos puntos son llamados “puntos de interés”. En torno a cada uno de estos puntos se puede tomar una ventana de la imagen y un vector de características, el que es denominado descriptor del punto de interés. Luego, si hay varias imágenes en que aparece un mismo objeto, pueden generarse correspondencias entre puntos equivalentes de dichas imágenes al hacer calces entre descriptores parecidos. El uso de descriptores puede ser omitido cuando se hace tracking siempre que el desplazamiento entre ambas imágenes sea muy pequeño, ya que se puede asumir que los puntos más cercanos geométricamente son los puntos correspondientes.

El trabajo con puntos de interés tiene sus orígenes en los trabajos de Crowley y Moravec. Crowley, en su Tesis de Doctorado [68], genera puntos de interés a partir de una estructura piramidal, la que obtiene usando convoluciones con Gaussianas, restando los pisos consecutivos para obtener una nueva estructura piramidal llamada SDoG (Sampled Difference of Gaussians) y, finalmente, buscando máximos locales 3D y bordes curvados en dicha estructura. Los puntos de interés encontrados son unidos entre sí y usados para crear una malla que represente el objeto y que permita reconocerlo posteriormente. Moravec, también en su Tesis de Doctorado [68], genera un detector de esquinas,

el que utiliza pares de imágenes estereográficas para resolver la estructura de la escena en la cual está inmerso el robot mientras éste se mueve con el objetivo de evadir obstáculos. El detector de Moravec pide que existan variaciones grandes en la imagen, en cuatro direcciones distintas a la vez, dentro de una vecindad de un punto. Al usar un número discreto de direcciones de búsqueda, no detecta los mismos puntos cuando la imagen es rotada. Harris y Stephens [34] mejoran el detector de Moravec, reformulándolo de tal modo que considere todas las direcciones posibles, permitiendo que entregue los mismos puntos aunque la imagen sea rotada. Shi y Tomasi [88] proponen hacer seguimiento a puntos similares a los detectados por el sistema de Harris y Stephens ya que, como en esos puntos la imagen presenta variaciones importantes en todas las direcciones, son puntos que pueden ser fácilmente seguidos en la imagen, aunque la cámara se mueva en cualquier dirección. El sistema de Harris y Stephens no entrega los mismos puntos cuando la imagen es reescalada. Lindeberg [44] muestra que, para obtener puntos de interés que sean invariantes ante cambios de escala, se debe usar una representación multiresolución de la imagen. Luego, junto a Garding [45][46] extiende el método para encontrar puntos que sean invariantes ante transformaciones afines. Finalmente Lindeberg [47] entrega una forma de transformar cualquier detector de puntos de interés homogéneo en otro invariante ante cambios de escala. Mikolajczyk y Schmid se basaron en los trabajos teóricos de Lindeberg para generar el detector Harris-Laplace [62], que es una versión del detector de Harris invariante a la escala de la imagen y enormemente robusto ante transformaciones afines que detecta esquinas en el scale space. Tras esto, generaron el detector Harris-Affine [63][64] que es igual al anterior pero con un post-procesamiento que corrige la posición de los puntos detectados de modo que su posición relativa en el scale space no se vea afectada por transformaciones afines. Existen además enfoques completamente distintos; por ejemplo, Matas, Chum, Urban y Pajdla [58] idearon un generador de puntos de interés (MSER, regiones extremas de máxima repetibilidad) que busca regiones tales que el histograma de luminosidad de su zona interior esté separado del histograma de luminosidad de su zona exterior por un umbral grande. Estas regiones son

claramente invariantes ante transformaciones afines, ya que se deforman junto con la imagen, además de ser invariantes ante transformaciones monótonas suaves del histograma. Un enfoque similar fue usado previamente por Tuytelaars y Luc Van Gool [97]. En general, es posible mezclar varios detectores de puntos, ya que reaccionan frente a características distintas en las imágenes, aunque esto provoca que el sistema de reconocimiento sea más lento.

Generar correspondencias a partir de dos nubes de puntos de interés, basándose sólo en la distribución geométrica de los mismos, es una tarea que puede tomar mucho tiempo. Zhang, Deriche, Faugeras y Luang [101], además de Torr [94], comenzaron a usar ventanas de correlación gráfica (los primeros descriptores usados) en torno a los puntos de interés con el objetivo de facilitar la generación de correspondencias, lo que se realizan pareando cada descriptor de una primera imagen con el descriptor de la segunda imagen que sea más parecido (el de menor distancia coseno), tras lo cual sigue una etapa de descarte de falsas correspondencias (basándose en restricciones geométricas entre las posiciones de los puntos en ambas imágenes). Schmid y Mohr [87] aplicaron los puntos de interés (detector de Harris simple, con posibilidad de ser escalado) junto con descriptores denominados jet (de 8 componentes, basados en derivadas de primer, segundo y tercer orden de la imagen previamente suavizada) para poder identificar varios objetos distintos a la vez en una misma imagen, siendo ésta la primera vez que se usaban las correspondencias entre puntos de interés con este propósito (anteriormente se usaban solamente para calzar pares estereográficos, para alineamiento de imágenes y para seguimiento de objetos en videos). Tras la detección de correspondencias se realizaba una verificación geométrica al pedir que los “n” puntos de interés más cercanos al que se está verificando existan tanto en la imagen del objeto como en la imagen a reconocer (no es tan estricto, se permite que falte alguno) y que tengan una disposición geométrica parecida. Lowe [55][57] generó una implementación eficiente llamada SIFT (Scale Invariant Feature Transform) que esta basada en una versión mejorada del detector SDoG mencionado anteriormente, que incluye el uso del Hessiano para descartar puntos de interés en líneas y bordes, y en un nuevo descriptor que consiste en un

histograma de los gradientes de la imagen en torno al punto de interés ponderados por una Gaussiana. Los descriptores, de 128 componentes, son usados en una etapa simultánea de descarte eficiente de falsas correspondencias y de reconocimiento de los objetos usando una transformada Hough y una tabla de Hashing. Lowe mejoró mucho el descriptor usado anteriormente en el trabajo de Schmid y Mohr (muchas más componentes, más robusto ante transformaciones afines, muy distintivo) y usó una verificación geométrica muchísimo más rápida basada en una transformada Hough sobre el espacio de las transformaciones de semejanza, con lo que mejoró el desempeño global del reconocedor y su velocidad. Además desarrolló un método adicional para poder trabajar con varias vistas de una misma imagen de entrenamiento de un modo eficiente con el objetivo de poder identificar un objeto aunque aparezca en varias poses posibles [56]. Loncomilla y Ruiz del Solar [49]-[52] generaron el método L&R, el cual agrega verificaciones extras para eliminar la gran cantidad de detecciones falsas que genera el sistema de Lowe y para refinar las correspondencias obtenidas, lo cual permite obtener un sistema genérico y robusto de reconocimiento de objetos que se ha usando en variados ámbitos de la visión computacional como visión robótica [49][50][82][83], reconocimiento de huellas dactilares [78][79], reconocimiento de firmas [80] y procesamiento de imágenes estelares [81]. Ke y Sukthankar crearon los descriptores PCA-SIFT, los cuales consisten en aplicar el método PCA a los descriptores SIFT; las mejoras obtenidas mediante este método son discutibles [64]. Bay, Tuytelaars y Luc Van Gool generaron un sistema llamado SURF [4] que consiste en un detector muy rápido basado en hessianos de Gaussiana. Este detector usa imágenes integrales [24] para aproximar los cálculos hechos con las Gaussianas, con lo que se logra una velocidad mucho mayor. En general todos los descriptores fueron diseñados para trabajar con imágenes en blanco y negro, pero pueden extenderse fácilmente a imágenes en color al procesar los canales R, G y B por separado, o bien usar pisos de pirámides correspondientes a distintos colores durante el cálculo de los puntos de interés. Este último enfoque es usado por Itti y Koch [37] para calcular la

saliencia visual de una imagen en un intento de predecir los puntos más mirados en las imágenes por los seres humanos.

Tras la aparición de los primeros descriptores invariantes a cambios de escala, comenzó el desarrollo de descriptores invariantes a transformaciones afines, lo que permite reconocer objetos planos como murallas o cajas desde distintos puntos de vista a partir de una única imagen de entrenamiento. Garding y Lindeberg [45][46] generaron una forma de calcular descriptores invariantes a transformaciones afines. El método consiste en generar una transformación de coordenadas afín sobre los puntos de interés encontrados que se puede entender como una elipse cuya área tiene que ver con la escala que está siendo procesada, y cuya relación de radios se elige de tal modo de que al transformar la elipse en un círculo, la imagen presente variaciones de igual magnitud en todas las direcciones (se usa una metodología relacionada a la del detector de Harris y Stephens) lo que se logra minimizando una medida de anisotropía [45], luego de lo cual se puede usar cualquier descriptor en las nuevas coordenadas sin que éste se vea afectado por transformaciones afines sobre al imagen. Este enfoque fue seguido por Mikolajczyk y Schmid al crear las regiones Harris-Affine[63][65], los que usan el detector de puntos de interés Harris-Laplace (detector de esquinas invariante a escala) y luego, para cada punto de interés encontrado, refinan su posición y escala característica y minimizan la anisotropía de la elipse de forma simultánea e iterativa, lo que da una posición exacta del punto de interés. Además entrega un sistema de coordenadas local sobre el cual se puede calcular el descriptor (usan como descriptor un grupo de coeficientes obtenidos convolucionando una ventana de la imagen con derivadas de Gaussianas, aunque ellos mismos reconocen que el descriptor SIFT funciona mejor [64]). Este detector corregido permite reconocer objetos en dos imágenes distintas siempre que éstos estén relacionados entre sí por una transformación afín sin una distorsión extrema. Recientemente, se desarrolló el descriptor ASIFT [69], el cual se genera simulando distintos puntos de vista de la imagen con lo cual puede manejar cambios de punto de vista de hasta 84 grados; la complejidad computacional obtenida es aproximadamente 12 veces la del método SIFT. Ferrari, Tuytelaars y Luc Van Gool [21] generaron una forma

de poder reconocer objetos “doblados”, es decir, proyectados bajo mapeos suaves por tramos arbitrarios, para lo cual usan un detector basado en luminosidad y luego usan correspondencias basadas en descriptores SIFT, las que son generados sobre un sistema local de coordenadas invariante frente a transformaciones afines en torno al punto de interés. A partir de las correspondencias, intentan generar pequeñas regiones que calzan entre sí bajo alguna transformación afín y hacen que dichas regiones crezcan, para lo que buscan nuevas transformaciones afines en los bordes de dichas regiones que sean parecidas (pero no iguales) a la transformación afín original. Este enfoque funciona relativamente bien porque cualquier mapeo suave puede ser aproximado localmente por una transformación afín (polinomio de Taylor de primer orden).

## 3.2 SLAM visual

Cuando un robot se mueve en una escena tridimensional, usualmente necesita saber su posición y orientación dentro de ese espacio. Para lograr esto, puede usar sensores para ir reconociendo objetos cuya posición es conocida, llamados landmarks, mientras se mueve. Si el robot ya conoce un mapa que contenga las posiciones de varios landmarks de la escena, el problema es llamado auto-localización del robot. En el caso de que el robot se mueva en un entorno desconocido, debe ir generando el mapa mientras se mueve pero, al mismo tiempo, debe localizarse usando el mapa incompleto que posee. Este problema, denominado SLAM (Simultaneous Localization And Mapping), requiere estimar un vector de estado del sistema, el que contiene la posición y orientación del robot y de todos los landmarks, a partir de las observaciones de la escena obtenidas usando los sensores y las órdenes dadas a los actuadores, las que están relacionadas por una dinámica no-lineal, donde hay incertidumbre en la medición y en la transición de estado. Dicha incertidumbre puede corresponder a ruidos aditivos, como el que se produce debido a que los codificadores de posición que miden la odometría son imperfectos, o el ruido de medición que se produce al estar la imagen cuantizada; o pueden corresponder a falsas detecciones, que es lo

que ocurre cuando se reconoce erróneamente un landmark en la imagen que no existe en la realidad. En consecuencia, para resolver el problema denominado SLAM es necesario diseñar un estimador de estado de alta dimensionalidad para un sistema no-lineal, que sea robusto ante diversos tipos de ruido y que pueda funcionar en tiempo real.

Este problema fue abordado inicialmente por Smith, Self y Cheeseman [89]. Ellos analizaron el caso de un robot que puede desplazarse sobre una superficie horizontal y girar, por lo que posee tres grados de libertad. Este robot se mueve en un entorno desconocido, en el cual hay varios landmarks que debe poder detectar mientras avanza. Para resolver este problema usaron un sistema basado en análisis estadístico de los datos que resulta equivalente a un filtro de Kalman extendido, el que es usado para estimar la posición de los landmarks que rodean al robot, así como la posición y orientación del mismo. Cada vez que se encuentra un nuevo landmark, la dimensionalidad del estado estimado crece. Se usan tres matrices de covarianza P, Q y R para representar la incertidumbre en la estimación del estado, en la odometría y en las observaciones respectivamente. Debido a que la matriz de covarianza P crece de forma cuadrática respecto a la dimensionalidad del estado, un esquema de este tipo no se puede hacer funcionar con una gran cantidad de landmarks. Además, el ruido de transición de estado sigue una distribución de probabilidad difícilmente modelable, ya que pueden ocurrir fenómenos tales como resbalamiento en las ruedas y raptos (el robot es transportado externamente a otra posición), por lo que su distribución sólo puede ser descrita adecuadamente por su media y covarianza cuando se encuentra bien localizado.

El desarrollo de métodos de estimación basados en partículas para resolver el problema de auto-localización robótica [16], que presenta problemas de ruido y no-linealidades similares a los del SLAM, permitió generar una nueva familia de métodos para resolver este problema. Sin embargo, los métodos basados en partículas para SLAM deben mantener una dimensionalidad reducida, ya que los métodos basados en partículas requieren una alta densidad de éstas para funcionar adecuadamente y la cantidad de dichas partículas crece

exponencialmente con la dimensionalidad del estado. Montemerlo, Thrun, Koller y Wegbreit generaron un método llamado FastSLAM [66], el cual consiste en estimar la posición y orientación del robot a través de partículas, cada una de las cuales tiene su propia estimación de cada uno de los landmarks. En este trabajo se asume que los landmarks son observados de forma independiente; así las estimaciones que tiene una misma partícula respecto de varios landmarks no están correlacionadas, por lo que cada landmark puede ser estimado usando un filtro de Kalman individual. Además, cada partícula decide si aceptar o no una observación con el objetivo de descartar falsas observaciones, por lo que los mapas que han estimado dos partículas distintas pueden diferir entre sí. Un problema en este método es que la dispersión de las partículas siempre aumenta con el tiempo, por lo cual en algún momento su densidad va a ser insuficiente para poder modelar la distribución real. Posteriormente, desarrollaron un algoritmo llamado FastSLAM 2.0 [67], el cual representa la distribución de la posición y orientación del robot usando un conjunto de partículas con covarianza, cada una de las cuales tiene su propio mapa. Cada partícula, además de corregir el mapa con cada observación, corrige su posición y covarianza. Por este motivo la convergencia del mapa y del estado del robot es más rápida aunque se usen pocas partículas, aunque se debe usar una ponderación para los pesos de las partículas que resulta difícil de evaluar. La corrección agregada en este nuevo sistema evita el problema mencionado en el método anterior, ya que la corrección de la posición de las partículas limita su dispersión.

Todos los modelos anteriores consideran que el robot tiene 3 grados de libertad. Sin embargo, existen trabajos que estiman un mapa y los 6 grados de libertad del robot. La pose completa incluye una traslación en un espacio tridimensional y una rotación con tres grados de libertad, la que puede ser representada usando un eje de rotación con 3 componentes con ángulo, un cuaternión con 4 componentes consistente en un número que incluye una componente real y tres imaginarias para representar tres ángulos, o una matriz de rotación con 9 componentes. Davison [17] generó un sistema de SLAM que funciona con una cámara movida por la mano, y que representa la posición y

velocidad de dicha cámara usando coordenadas cartesianas, su orientación usando cuaterniones, su velocidad angular usando un vector de velocidad angular y el mapa usando coordenadas rectangulares para cada uno de los landmarks. El estimador usado es un filtro de Kalman. Para evitar que la dimensionalidad del estado crezca, se mantienen siempre pocos landmarks en el vector de estado y los que salen fuera de la cámara son eliminados. Al no existir un mapa global de la escena y al no existir información sobre el movimiento de la mano que lleva la cámara (es un SLAM visual), este método funciona sólo en ambientes de pequeño tamaño y cuando los movimientos de la cámara son suaves. Al ser la distribución que representa la incertidumbre inicial de la posición de los puntos muy asimétrica, se debe usar una representación especial para dicha incertidumbre basada en una suma de Gaussianas, la cual colapsa en una Gaussiana cuando la incertidumbre disminuye. Posteriormente, se desarrollaron otros trabajos basados en este mismo enfoque, los cuales usan una cámara gran angular [18]; una inicialización especial para los puntos nuevos basada en un modelo que los representa mediante un punto origen más una dirección en coordenadas esféricas con radio inverso [12]; usan técnicas de descarte de falsos positivos al actualizar el sistema con cada medición por separado y verifican luego la compatibilidad del resultado con el resto de las mediciones [13], entre otros métodos generados en el último tiempo, los que permiten manejar mapas de mayor tamaño al podar las covarianzas cruzadas de los puntos cuando no han sido vistos al mismo tiempo.

Para que el robot pueda moverse a través de ambientes de gran tamaño manteniendo un mapa consistente, es útil que la representación del mapa tenga una estructura flexible. Bosse, Newman, Leonard, Soika, Feiten y Teller generaron un sistema de SLAM que se basa en usar un atlas (inspirado en la teoría de manifolds) compuesto de varias cartas (sistemas de coordenadas) que se superponen parcialmente para representar el entorno del robot [7]. Las referencias a las distintas cartas son almacenadas en un grafo. Las varianzas de los errores de transición de estado y observación del sistema se transforman en varianzas de las transformaciones que permiten pasar de una carta a la otra, las que se propagan por el grafo. El entorno está representado por un conjunto de cartas

relacionadas mediante transformaciones de coordenadas que contienen incertidumbre. Dado un par de cartas no adyacentes, existen varios caminos en el grafo que las unen, los que producen distintas transformaciones de coordenadas, por lo cual se elige el camino de menor varianza para elegir la transformación menos contaminada. Se, Lowe y Little [84] usan filtros de Kalman extendidos independientes para representar los landmarks, los que corresponden a descriptores SIFT detectados usando un sistema estéreo trinocular, y un método heurístico basado en transformada Hough [84] para encontrar la posición y orientación del robot cuadro a cuadro. Cuando hay muy pocos landmarks conocidos detectados en un cuadro, se genera un mapa temporal. Cada 30 cuadros se genera un nuevo mapa temporal, el que se guarda separado de los anteriores. Cuando se vuelve a encontrar una gran cantidad de landmarks del mapa principal, se calculan las transformaciones de coordenadas entre los mapas temporales usando el método de mínimos cuadrados, tras lo cual los landmarks contenidos en éstos son agregados al mapa principal. Si bien este método es altamente heurístico, muestra una forma simple de mantener un mapa principal coherente al impedir que el error de transición de estado integrado en el tiempo deforme el mapa principal cuando el robot explora zonas no vistas en la escena.

## 3.3 Estado del arte en SLAM visual

El tema del SLAM basado en visión se ha vuelto un tema de investigación importante debido a que los teléfonos móviles actuales suelen tener una cámara integrada además de una buena capacidad de cómputo, lo cual permite su uso como parte de una plataforma robótica de bajo costo. La mayor parte de los trabajos actuales relacionados con localización robótica usando una cámara como sensor suelen estar contenidos en una de dos líneas: *structure from motion recovery* (recuperación de la estructura a partir del movimiento) y SLAM monocular.

Los métodos de la familia *structure from motion* consisten en métodos que calculan la estructura de la escena y el movimiento de la cámara usando

geometría proyectiva, sin considerar modelos dinámicos. Las metodologías basadas en la odometría visual desarrollada por Nistér [72] usando RANSAC preemptivo [73] aplicado sobre conjuntos de 3 y 5 puntos (los cuales son extraídos usando un detector de Harris), permiten obtener correspondencias entre frames consecutivos, además de una estimación de movimiento relativo. Al procesar dichas correspondencias posteriormente usando métodos de la familia *bundle adjustment* [4][20] sobre secuencias de algunos cuadros (*frames*), se pueden obtener resultados impresionantemente precisos en tiempo real, aunque el error respecto a la ruta del robot y las estructuras detectadas en el mapa aumenta constantemente a medida que pasa el tiempo, lo cual se debe a que solamente se considera el movimiento relativo de la cámara.

Los métodos basados en SLAM visual, derivados de la tesis de doctorado de Davison [17], pueden lograr muy buenos resultados en mapas de tamaño pequeño a mediano, pero el manejo de mapas grandes es un tema difícil debido a que la matriz de covarianza crece de forma cuadrática respecto al número de landmarks, presenta problemas de cierre de bucles[5] debido a que se hace *tracking* de los puntos en vez de detectarlos en cada frame y aparecen fluctuaciones en la escala del mapa a medida que la cámara se mueve debido a que la escala del mapa no es observable usando una sola cámara. El mapa se puede reconstruir de forma densa en tiempo real usando una minimización del error de una grilla moldeable basada en flujo óptico [71]. Una forma de encontrar rápidamente los landmarks que se dejaron de ver basada en visión activa escalable [32] se ha desarrollado para manejar mapas mayores que involucran una gran cantidad de información de correlaciones cruzadas, la cual se simplifica usando un enfoque de

---

[4] *Bundle adjustment* es un nombre estándar que se refiere al proceso de encontrar un conjunto de poses para las cámaras y posiciones para los puntos que minimizan el error de proyección total. La palabra bundle se usa en inglés para denominar un conjunto de rayos, por lo cual *bundle adjustment* se podría traducir como “ajuste de un haz de rayos”.

[5] Cierre de bucle, o *loop closing* en inglés, se refiere al fenómeno que ocurre cuando el robot reconoce una zona del mapa ya observada tras haber generado una zona nueva del mapa, por lo cual debe corregirse el error acumulado en la trayectoria.

poda de grafos para poder reducir la información de covarianza del sistema, limitando la propagación de incertidumbre entre puntos muy lejanos. Un SLAM que maneja explícitamente las fluctuaciones en la escala se logró desarrollar mediante un modelamiento del sistema dinámico asociado al SLAM usando teoría de grupos de Lie sobre el espacio de transformaciones de rotación – traslación – cambio de escala para poder usar *bundle adjustment* para minimizar las fluctuaciones de escala cuando se detecta un cierre de bucle [92]. El uso de la teoría de grupos de Lie permite llevar la restricción del tipo de transformación que se debe considerar de un nivel global a un nivel diferencial. El tiempo que requiere RANSAC para resolver una asociación de datos aumenta exponencialmente con el número de datos necesarios para generar una hipótesis, por lo cual se desarrolló un RANSAC de 1 punto para poder realizar asociación de datos de forma eficiente, para lo cual se actualiza la posición del robot usando una sola medición y se analiza el consenso sobre el resto mediante un test chi-cuadrado [13]. Por último, tanto la apariencia gráfica de un conjunto de puntos como de su distribución tridimensional han sido usados para crear *clusters* de puntos [1]. Este enfoque puede ser usado en el futuro para poder etiquetar zonas del mapa, dándoles un valor semántico.

Algunos de los problemas actuales del SLAM visual se deben directamente al uso de los puntos para formar *landmarks*, ya que un solo punto da una información débil que es insuficiente para inferir la pose de la cámara, lo cual limita la velocidad de convergencia del sistema. Para poder superar este problema se han propuesto landmarks de más alto nivel los cuales están formados por puntos que pertenecen a las superficies de objetos reales. En [28] se usan estructuras de alto nivel, como planos y líneas, que son construidos a partir de landmarks básicos correspondientes a puntos o *edgelets* usando RANSAC para formar conjuntos coherentes para ser reducidos. En [42] se usan landmarks planos formados por grupos de 4 puntos que se inicializan directamente, codificando su posición y orientación usando *inverse-depth points*. La evolución del estado de la cámara, del estado de la normal del plano y de los errores de medición son representados usando grupos de Lie, lo cual permite mantener restricciones globales respecto al

tamaño de algunos vectores (las normales que deben ser unitarias) a un nivel diferencial.

# 4 Estimación robusta de pose en visión computacional: Definiciones, Representaciones y Algoritmos

En este capítulo se mostrarán diversas técnicas y conceptos que deben ser conocidos para poder comprender a cabalidad el trabajo desarrollado en esta tesis. Primero se introducirán diversos conceptos matemáticos que se usarán a lo largo de este texto, luego se introducirá el algoritmo de los 3 puntos, el cual es usado en el sistema diseñado para generar hipótesis acerca de la pose del landmark rígido. Finalmente se mostrarán los métodos basados en SLAM, los cuales asumen un modelo dinámico probabilístico para el mapa y la cámara.

## 4.1 Distribuciones de probabilidad y representaciones

La distribución de probabilidad $p(x)$ de una variable aleatoria $X$ en un espacio N-dimensional puede ser aproximada de diversas formas. Las siguientes son alternativas comúnmente usadas:

- Mediante una distribución Gaussiana (suave y unimodal) en un espacio N-dimensional, la que requiere especificar una media (vector de N componentes) y una covarianza (matriz de NxN componentes), la cual especifica completamente la incertidumbre asociada a la distribución.
- Mediante una suma ponderada de Gaussianas en un espacio N-dimensional, siendo la suma de las ponderaciones igual a la unidad. Cada Gaussiana tiene una media y una matriz de covarianza asociada.
- Mediante un conjunto de partículas ponderadas de N dimensiones que representan la distribución de forma aproximada. Cada partícula tiene un peso o factor de importancia asociado que representa su probabilidad relativa. El conjunto de partículas se puede generar mediante un muestreo de la distribución; sin embargo, el número de muestras necesarias crece exponencialmente con la dimensionalidad. Esta representación es muy versátil y suele ser la mejor elección para dimensionalidades bajas.

- Mediante un conjunto de sigma puntos [39], que son 2N+1 partículas cuidadosamente elegidas para representar apropiadamente la media y covarianza de la distribución.
- Mediante una representación híbrida. El estado $x$ es dividido en dos partes: $x = (x_A, x_B)$. Se usa la descomposición $p(x) = p(x_A \mid x_B)p(x_B)$. Usualmente $p(x_B)$ es representado usando partículas al tener menor dimensionalidad, mientras que $p(x_A \mid x_B)$ es representado usando un conjunto de Gaussianas, una para cada partícula.

La transformada unscented [39] permite aproximar una Gaussiana mediante un conjunto de sigma-puntos y viceversa. La Gaussiana también puede ser aproximada por un conjunto de partículas si se hace un muestreo. Finalmente, un conjunto de partículas puede aproximarse por una Gaussiana calculando su media y covarianza, aunque la aproximación es deficiente si la distribución es multimodal. De este modo, cada una de las representaciones anteriores puede ser transformada en otra, aunque se pierde información al pasar de una distribución a otra con menor versatilidad.

Al aplicar una transformación $f(\cdot)$ a una variable aleatoria $X$ con una distribución asociada $p(\cdot)$, se obtiene una nueva variable aleatoria $f(X)$, cuya distribución $p_F(\cdot)$ debe ser estimada. Hay varios métodos que logran este objetivo:

- Si la distribución inicial es representada usando un conjunto de partículas $(x_i, \omega_i)$, la nueva distribución queda representada por el conjunto de partículas $\left(f(x_i),\ \omega_i f(x_i)/S\right)$, con $S = \Sigma\omega_i * f(x_i)$ un factor de normalización.
- Si la distribución es original es Gaussiana $X \sim N(\bar{x}, C)$, y la transformación es lineal, de la forma $f(x) = Fx$, entonces la nueva distribución es Gaussiana y toma la forma $f(X) \sim N\left(F\bar{x}, FCF^T\right)$.

- En el caso en que la distribución es Gaussiana $X \sim N(\bar{x}, C)$ y la transformación es no-lineal, la distribución resultante no es Gaussiana. En algunos casos es deseable aproximar la distribución final por una Gaussiana, caso en el cual se debe estimar una media y una covarianza para la distribución final. Dos métodos comúnmente usados para lograr ese objetivo son los siguientes:
    1. La función puede ser linealizada en torno a la media de la distribución, quedando con la forma $f_{LIN}(x) = f(\bar{x}) + J(x - \bar{x})$. En este caso, se puede aproximar la distribución resultante como la Gaussiana $F(X) \sim N\left(f(\bar{x}), JCJ^T\right)$. Este método de aproximar la distribución resultante por una Gaussiana es denominado usualmente método de propagación de la incertidumbre, o método de propagación del error. El método es apropiado cuando la verdadera distribución resultante es similar a una Gaussiana (unimodal y simétrica elipsoidal). El error que se produce es igual a los términos del polinomio de Taylor de $f(\cdot)$ que fueron omitidos, los cuales son dominados por el término $(1/2)(x - \bar{x})^T H(x - \bar{x})$, el cual está asociado al Hessiano de la función. En consecuencia, la curvatura existente en la transformación genera un error en la aproximación.
    2. Se puede aplicar la transformada unscented a la distribución Gaussiana para obtener sigma puntos, aplicar la transformación a los sigma-puntos, calcular la media y covarianza de las partículas resultantes y aproximar el resultado mediante una Gaussiana, la cual aproxima a la verdadera distribución final. Este método permite manejar no-linealidades más pronunciadas que el anterior ya que el efecto causado por la curvatura de la función se ve reflejado en las partículas obtenidas, pero de todas formas requiere que la verdadera distribución resultante sea similar a una Gaussiana. Por otro lado, puede fallar cuando la distribución sufre de aliasing (distintas posiciones en el espacio de la distribución inicial pueden representar el mismo valor en la distribución final) ya que en este caso los sigma puntos resultantes no siempre son capaces de

representar la distribución final. Un ejemplo intuitivo de este problema es la aplicación de la funcion $\text{sen}(x)$ a una distribución, ya que si los sigma puntos iniciales quedan en los ceros de la función seno, la covarianza de los sigma puntos finales es cero.

## 4.2 Minimización de funciones

Dada una función $f(\cdot)$ definida sobre un espacio $U$, se define su mínimo global del siguiente modo:

$$\min_x f(x) = M^* \Leftrightarrow \forall x \in U, M^* \leq f(x) \quad (1)$$

$$\arg\min_x f(x) = X^* \Leftrightarrow \forall x \in U, f(X^*) \leq f(x). \quad \textbf{(2)}$$

Dada una función $f(\cdot)$ definida sobre una vecindad de tamaño $r_{MAX}$ en torno a un punto $x_0$, el mínimo local de la función $f(\cdot)$ asociado al punto de partida $x_0$ dentro de un radio $r_{MAX}$ se define del siguiente modo:

$$\min_x f(x \text{ desde } x_0 \mid r_{MAX}) = M^* \Leftrightarrow \forall r \in [0, r_{MAX}], \forall x, \|x - x_0\| < r \Rightarrow M^* \leq f(x) \quad (2)$$

$$\arg\min_x f(x \text{ desde } x_0 \mid r_{MAX}) = X^* \Leftrightarrow \forall r \in [0, r_{MAX}], \forall x, \|x - x_0\| < r \Rightarrow f(X^*) \leq f(x). \quad (3)$$

En este trabajo, en los casos es que $f(\cdot)$ sea una función formada por una suma de cuadrados, se usará el método iterativo Levenberg-Marquardt para encontrar los mínimos locales a menos que se especifique lo contrario. La suma de cuadrados $\Sigma e_i^2(x)$ se genera a partir de un vector de residuales $e(x)$. Este método usa el vector de residuales y su jacobiano para calcular una aproximación del mínimo en cada iteración. Un parámetro $\lambda$ determina la importancia relativa que se le da al jacobiano respecto al vector de residuales en cada iteración. A medida que $\lambda$ crece, el algoritmo converge al método del gradiente, y a medida que $\lambda$ disminuye, el algoritmo converge a un método parabólico. El valor de $\lambda$ se va modificando en cada iteración para lograr una convergencia rápida. El método

Levenberg-Marquardt que se programó funciona en cada iteración $T$ del siguiente modo:

$$f(x) = \sum_{i=1}^{n} \left(e_i(x)\right)^2 = e(x)^T e(x) \tag{4}$$

$$\sup_{T+1} = x_T - \left( \frac{de(x_T)}{dx}^T \frac{de(x_T)}{dx} + \lambda_T I \right)^{-1} \frac{de(x_T)}{dx}^T e(x_T) \tag{5}$$

$$x_{T+1} = \begin{cases} \sup_{T+1}, & E(\sup_{T+1}) < E(x_T) \\ x_T, & E(\sup_{T+1}) > E(x_T) \end{cases} \tag{6}$$

$$\lambda_{T+1} = \begin{cases} \lambda_T / 10, & E(\sup_{T+1}) < E(x_T) \\ \lambda_T * 10, & E(\sup_{T+1}) > E(x_T). \end{cases} \tag{7}$$

Al minimizar una función usando métodos iterativos se logra llegar a un mínimo local en torno a un punto inicial $x_0$ dado. Los mínimos locales que se obtienen mediante un algoritmo iterativo se definen del siguiente modo:

$$x_{T+1} = \Phi(x_T), f(x_{T+1}) \leq f(x_T) \tag{8}$$

$$\min_x \left(f(x) \text{ desde } x_0\right) = \lim_{N \to \infty} f\left(\Phi^N(x_0)\right) \tag{9}$$

$$\arg\min_x \left(f(x) \text{ desde } x_0\right) = \lim_{N \to \infty} \Phi^N(x_0). \tag{10}$$

Una elección adecuada del punto inicial $x_0$ usado en el algoritmo iterativo puede lograr que se encuentre el mínimo global *min f*(*x*). Para lograr esto, se requiere que la distancia entre el punto inicial $x_0$ y el punto mínimo global $x^*$ sea menor que un cierto radio de convergencia $\varepsilon(f,x_0)$, el cual suele ser desconocido.

$$x^* = \arg\min_x f(x) \tag{11}$$

$$\left\| x_0 - x^* \right\| \leq \varepsilon(f, x_0) \Rightarrow \min_x \left(f(x), x_0\right) = f\left(x^*\right) \tag{12}$$

Otra propiedad interesante es que se puede calcular la derivada de *argmin*, es decir, se puede calcular la derivada de una minimización. Este cálculo requiere

que la función de error tenga segunda derivada continua y es independiente del método de minimización que se use.

$$x^*(a) = \arg\min_x \{f(x, a) \text{ desde } x_0\} \tag{13}$$

$$x \in R^N, a \in R^M \tag{14}$$

$$\left(\frac{\partial}{\partial x} f(x,a)\right) | \, x^*(a) = 0_{1*N} \tag{15}$$

$$\frac{\partial f\left(x^*(a), a\right)}{\partial x} = 0_{1*N} \tag{16}$$

$$\left(\frac{\partial f\left(x^*(a), a\right)}{\partial x}\right)^T = 0_{N*1} \tag{17}$$

$$F_X\left(x^*(a), a\right) = 0_{N*1} \tag{18}$$

$$\frac{d}{da} F_X\left(x^*(a), a\right) = 0_{N*M} \tag{19}$$

$$\frac{\partial F_X}{\partial x} * \frac{dx^*}{da} + \frac{\partial F_X}{\partial a} = 0_{N*M} \tag{20}$$

$$F_{XX} * \frac{dx^*}{da} + F_{XA} = 0_{N*M} \tag{21}$$

$$\frac{dx^*}{da} = -F_{XX}{}^{-1} * F_{XA} \tag{22}$$

$$F_{XX(i,j)} = \frac{d^2 f(x,a)}{dx_i dx_j}, F_{XA(i,j)} = \frac{d^2 f(x,a)}{dx_i da_j} \tag{23}$$

$$\frac{d}{da}\left(\arg\min_x \{f(x, a) \text{ desde } x_0\}\right) = -F_{XX}{}^{-1} F_{XA} \tag{24}$$

El problema de la expresión anterior es que los términos $F_{XX}$ y $F_{XA}$ usualmente tienen una complejidad de cálculo significativa, aunque en estos casos pueden ser evaluados usando diferencias finitas con un nivel de precisión más que suficiente. Además, se requiere la condición de que el Hessiano $F_{XX}$ sea invertible. Usar esta fórmula es mejor que repetir la minimización iterativa cambiando los parámetros para obtener la variación del punto de convergencia, ya que esto

último requiere de una cantidad de tiempo inaceptable para aplicaciones en tiempo real.

## 4.3 Representación de posiciones y orientaciones en el espacio tridimensional

Las definiciones que se realizarán a continuación se refieren a objetos matemáticos definidos en el contexto de un espacio tridimensional, a menos que se especifique lo contrario.

### 4.3.1 Punto tridimensional cartesiano

Es un objeto matemático de 3 componentes que permite especificar una posición en el espacio tridimensional respecto a un sistema de referencia determinado.

$$\vec{p} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{25}$$

### 4.3.2 Matriz de covarianza de un punto tridimensional

Es un objeto matemático de 9 componentes (una matriz de 3x3) que se puede usar como una medida de dispersión de la distribución de probabilidad asociada a un punto tridimensional.

$$C(X) = E\left\{(X - E(X))(X - E(X))^T\right\},\ C(X) \in R^{3X3},\ X : \Omega \to R^3 \tag{26}$$

La matriz de covarianza de un punto cartesiano permite especificar completamente la incertidumbre de una posición en el espacio tridimensional cuando la distribución es Gaussiana, caso en el cual la distribución tiene la máxima entropía posible para una covarianza dada. Una superficie de nivel de una distribución Gaussiana tridimensional corresponde a un elipsoide que indica la incertidumbre del punto en las distintas direcciones (ver figura 1). Al aumentar los

valores propios asociados a la matriz de covarianza, aumenta el tamaño del elipsoide.

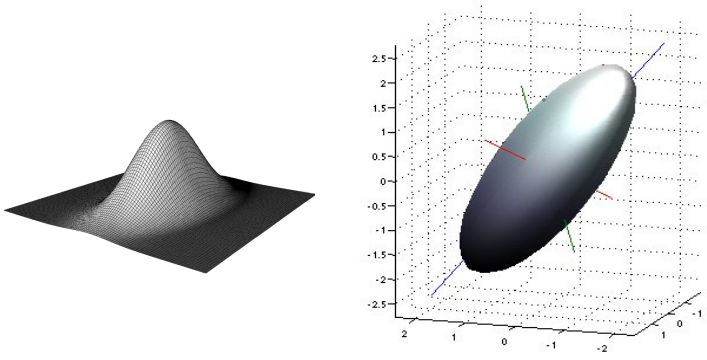


**Figura 1: Gaussiana bidimensional y un elipsoide asociado a una Gaussiana tridimensional.**

### 4.3.3 Punto tridimensional en coordenadas homogéneas

Es un objeto matemático de 4 componentes $(X,Y,Z,W)$ que permite especificar una posición en el espacio tridimensional respecto a un sistema de referencia determinado. Usualmente se suele elegir $W=1$, aunque $W$ puede tomar cualquier valor en los números reales. Puede representar puntos en el infinito cuando $W=0$; en este caso las tres componentes $(X,Y,Z)$ indican la dirección del punto en el infinito. Un punto en coordenadas homogéneas que no esté en el infinito puede ser transformado en un único punto cartesiano usando la siguiente transformación:

$$p = pun4d\left(\begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix}\right) = \begin{pmatrix} X/W \\ Y/W \\ Z/W \end{pmatrix}. \qquad (27)$$

Los puntos en coordenadas homogéneas se usan en geometría proyectiva con un $W$ arbitrario, y también se usan en robótica para representar posiciones en el espacio usando $W=1$.

### 4.3.4 Punto tridimensional con profundidad inversa

Es un objeto matemático de 6 componentes que permite especificar una posición en el espacio tridimensional respecto a un sistema de referencia determinado. Se especifica un origen $(x,y,z)$ y una dirección en coordenadas esféricas $(\rho,\theta,\phi)$, pero usando el inverso del radio en el vector. Es útil para representar puntos en el espacio observados por una cámara. Es posible transformar entre punto con profundidad inversa y punto de forma inyectiva:

$$r = \begin{pmatrix} x \\ y \\ z \\ \rho \\ \theta \\ \phi \end{pmatrix}, \quad p = pun\left(\begin{bmatrix} x \\ y \\ z \\ \rho \\ \theta \\ \phi \end{bmatrix}\right) = \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} \cos(\theta)\cos(\phi)/\rho \\ \sin(\theta)\cos(\phi)/\rho \\ \sin(\phi)/\rho \end{pmatrix} \tag{28}$$

$$\frac{dpun(r)}{dr} = \begin{pmatrix} & -\cos(\theta)\cos(\phi)/\rho^2 & -\sin(\theta)\cos(\phi)/\rho & -\cos(\theta)\sin(\phi)/\rho \\ I_{3x3} & -\sin(\theta)\cos(\phi)/\rho^2 & \cos(\theta)\cos(\phi)/\rho & -\sin(\theta)\sin(\phi)/\rho \\ & -\sin(\phi)/\rho^2 & 0 & \cos(\phi)/\rho \end{pmatrix}. \tag{29}$$

### 4.3.5 Matriz de covarianza de un punto tridimensional con profundidad inversa

Es un objeto matemático de 36 componentes (una matriz de 6x6), es una medida de la incertidumbre de un punto tridimensional con profundidad inversa.

$$C(X) = E\left\{(X - E(X))(X - E(X))^T\right\},\ C(X) \in R^{6X6},\ X : \Omega \to R^6$$

La matriz de covarianza de un punto tridimensional con profundidad inversa se puede usar para especificar completamente la incertidumbre asociada a una distribución Gaussiana en 6 dimensiones. Al ser proyectada la distribución gaussiana de 6 dimensiones en el espacio tridimensional, permite representar la incertidumbre de un punto tridimensional mediante una distribución que puede tomar diversas formas entre un elipsoide y un cono, lo cual es útil para representar

la incertidumbre en el espacio de un punto recién observado por una cámara, ya que dicha incertidumbre es altamente asimétrica (ver Figura 2).

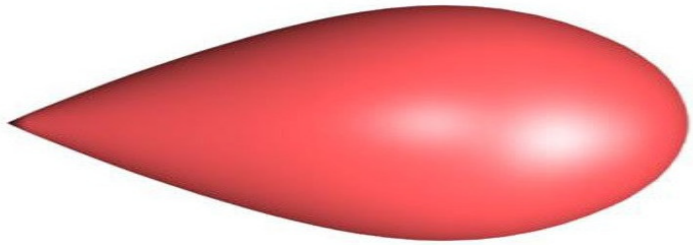


**Figura 2: Distribución asimétrica asociada a un punto con profundidad inversa en el espacio tridimensional. Puede variar entre un elipsoide y un cono.**

### 4.3.6 Matriz de 3x3 (transformación lineal)

Es un objeto matemático formado por nueve componentes reales que permite especificar una transformación lineal entre dos sistemas de referencia La transformación de rotación es un caso particular de la transformación lineal. Las transformaciones lineales en su forma más general pueden tener problemas de ortogonalidad (transformar un sistema de ejes ortogonales en un sistema de ejes no ortogonales) o de escala (relacionar dos sistemas de ejes con distinta escala), lo cual es inconveniente para el modelamiento de sistemas físicos. Las matrices de 3x3 se pueden componer mediante una multiplicación de derecha a izquierda.

Las matrices de 3x3 se pueden descomponer, usando una descomposición en valores singulares (SVD) en dos matrices ortonormales y tres cambios de escala. La descomposición en valores singulares tiene una complejidad de cálculo considerable.

$$M_{3x3} = UDV^T = R_1 \begin{pmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{pmatrix} R_2 \tag{30}$$

$$R_1, R_2 \in R^{3X3},\ R_1 R_1^{\ T} = I_{3x3},\ R_2 R_2^{\ T} = I_{3x3} \tag{31}$$

$$d_1, d_2, d_3 \in R_0^+ \tag{32}$$

Para estimar la rotación asociada a una matriz de 3x3, basta eliminar el efecto de los cambios de escala sobre la matriz [102], proceso que se denominará reparación SVD de la matriz (*svdRep*).

$$M_{3x3} = R_1 \begin{pmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{pmatrix} R_2 \rightarrow R = \text{svdRep}(M_{3x3}) = R_1 R_2 \quad (33)$$

$$RR^T = (R_1 R_2)(R_1 R_2)^T = R_1 R_2 {R_2}^T {R_1}^T = I \quad \textbf{(34)}$$

### 4.3.7 Matriz homogénea (sin perspectiva)

Es un objeto matemático de 12 dimensiones que permite especificar una transformación afín entre dos sistemas de referencia. Una transformación afín corresponde a una transformación lineal más una traslación. Si la transformación lineal corresponde a una rotación, se dirá que la transformación es de rotación-traslación.

$$H = \begin{pmatrix} m_{11} & m_{12} & m_{13} & t_X \\ m_{21} & m_{22} & m_{23} & t_Y \\ m_{31} & m_{32} & m_{33} & t_Z \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} M_{3x3} & t \\ 000 & 1 \end{pmatrix} \quad (35)$$

En el caso más general, pueden tener problemas de ortogonalidad (transformar un sistema de ejes ortogonales en un sistema de ejes no ortogonales) y de escala (relacionar sistemas de ejes con distinta escala), lo cual es inconveniente para el modelamiento de sistemas físicos.

Una matriz homogénea puede transformarse en una matriz de rotación-traslación realizando una reparación SVD sobre la submatriz correspondiente a la transformación lineal. A este proceso se le denominará reparación SVD de la matriz homogénea:

$$H_{4x4} = \begin{pmatrix} M_{3x3} & t \\ 000 & 1 \end{pmatrix} \rightarrow \text{svdRep}(H_{4x4}) = \begin{pmatrix} R & t \\ 000 & 1 \end{pmatrix} = \begin{pmatrix} svdR(M_{3x3}) & t \\ 0 \;\; 0 \;\; 0 & 1 \end{pmatrix}. \quad (36)$$

### 4.3.8 Matriz de covarianza de una matriz homogénea

Es un objeto de 144 componentes que permite representar la incertidumbre asociada a una matriz homogénea. Debido a que la matriz homogénea puede representar transformaciones no ortogonales y cambios de escala, la covarianza de una matriz homogénea puede representar la incertidumbre de la ortogonalidad de los ejes o de la escala, lo cual es algo sumamente inconveniente para el modelamiento de sistemas físicos. Además, el problema de ortogonalidad de las covarianzas de matrices homogéneas no puede ser resuelto mediante normalización SVD; es decir, no existe forma de resolver este problema manteniéndose dentro de esta representación. Esto, sumado a la alta dimensionalidad de estos objetos, hace que no sean candidatos apropiados para representar la incertidumbre de una pose en el espacio.

### 4.3.9 Cuaternión[6]

Es un objeto matemático de 4 componentes que puede usarse para representar rotaciones en el espacio. Se define como un tipo de número complejo que corresponde a la suma de una componente real y tres componentes imaginarias distintas entre si:

$$q = a + bi + cj + dk,\ k = ij = -ji,\ \ i^2 = -1,\ j^2 = -1,\ k^2 = -1. \quad (37)$$

También se puede usar una notación vectorial para el cuaternión:

[6] Número hipercomplejo introducido por W.R. Hamilton en [31]

$$q = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}. \tag{38}$$

Se usará este tipo de notación en adelante. Se cumplen las siguientes propiedades:

$$q_1 \cdot q_2 = \begin{pmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \end{pmatrix} \cdot \begin{pmatrix} a_2 \\ b_2 \\ c_2 \\ d_2 \end{pmatrix} = \begin{pmatrix} a_1a_2 - b_1b_2 - c_1c_2 - d_1d_2 \\ a_1b_2 + b_1a_2 + c_1d_2 - d_1c_2 \\ a_1c_2 - b_1d_2 + c_1a_2 + d_1b_2 \\ a_1d_2 + b_1c_2 - c_1b_2 + d_1a_2 \end{pmatrix} \tag{39}$$

$$\overline{q} = conj(q) == \begin{pmatrix} a \\ -b \\ -c \\ -d \end{pmatrix} \quad \|q\|^2 = q \cdot \overline{q} = a^2 + b^2 + c^2 + d^2 \quad q^{-1} = \frac{\overline{q}}{\|q\|^2} \tag{40}$$

$$q^{-1} \cdot q = q \cdot q^{-1} = 1 \tag{41}$$

$$q = \|q\| e^{\hat{n}\theta} = \|q\| \left( \cos\theta + i \frac{n_X}{\|n\|} \sin\theta + j \frac{n_Y}{\|n\|} \sin\theta + k \frac{n_Z}{\|n\|} \sin\theta \right). \tag{42}$$

Todas las operaciones que se pueden realizar con los números complejos se pueden realizar también con los cuaterniones. Una característica importante de los cuaterniones que se debe conocer es que su multiplicación es asociativa pero no conmutativa.

Un cuaternión se puede usar para representar una rotación de ángulo θ en torno a un eje unitario $n$ si se define como:

$$q = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ n_X \sin\left(\frac{\theta}{2}\right) \\ n_Y \sin\left(\frac{\theta}{2}\right) \\ n_Z \sin\left(\frac{\theta}{2}\right) \end{pmatrix} \quad \|n\| = 1. \tag{43}$$

Los vectores se pueden rotar de forma directa o inversa usando un cuaternión. Ambas operaciones de rotación corresponden a una forma cuadrática que se debe evaluar usando el cuaternión y tienen una complejidad numérica similar. Para poder especificar rotaciones se definen a continuación las funciones *rot* (rotación directa) y *roti* (rotación inversa), las cuales usan la expresión $(0, I_{3x3})$ para indicar que se deben tomar las filas 2, 3 y 4 del cuaternión resultante para formar un vector:

$$\begin{pmatrix} x_R \\ y_R \\ z_R \end{pmatrix} = rot\left(q, \begin{pmatrix} x \\ y \\ z \end{pmatrix}\right), \qquad \begin{pmatrix} 0 \\ x_R \\ y_R \\ z_R \end{pmatrix} = q \cdot \begin{pmatrix} 0 \\ x \\ y \\ z \end{pmatrix} \cdot q^{-1}, \quad rot(q,p) = (0, I_{3x3}) \left( q \cdot \begin{pmatrix} 0 \\ p \end{pmatrix} \cdot q^{-1} \right) \tag{44}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = roti\left(q, \begin{pmatrix} x_R \\ y_R \\ z_R \end{pmatrix}\right), \qquad \begin{pmatrix} 0 \\ x \\ y \\ z \end{pmatrix} = q^{-1} \cdot \begin{pmatrix} 0 \\ x \\ y \\ z \end{pmatrix} \cdot q, \qquad roti(q,p) = (0, I_{3x3}) \left( q^{-1} \cdot \begin{pmatrix} 0 \\ p \end{pmatrix} \cdot q \right). \tag{45}$$

Dada la identificación de la función *rot* con la rotación del vector $p$, se cumplen las siguientes propiedades de composición de la rotación (ver demostración en el anexo):

$$rot(q_1, rot(q_2, p)) = rot(q_1 \cdot q_2, p) \tag{46}$$

$$rot(q, roti(q, p)) = rot(q \cdot q^{-1}, p) = p \tag{47}$$

$$roti(q, rot(q, p)) = rot(q^{-1}, rot(q, p)) = rot(q^{-1} \cdot q, p) = p \tag{48}$$

$$rot(q, p) = R(q)p,\ R(q) = \frac{1}{a^2 + b^2 + c^2 + d^2} \begin{pmatrix} a^2 + b^2 - c^2 - d^2 & 2bc - ad & 2bd + 2ac \\ 2bc + 2ad & a^2 - b^2 + c^2 - d^2 & 2cd - 2ab \\ 2bd - 2ac & 2cd + 2ab & a^2 - b^2 - c^2 + d^2 \end{pmatrix} \tag{49}$$

$$roti(q, p) = R^T(q)p. \tag{50}$$

Existe un problema denominado aliasing del cuaternión que consiste en que existen infinitos cuaterniones que representan la misma rotación. Este problema se puede resolver parcialmente al normalizarlos, pero aún así quedan dos cuaterniones posibles, q y –q, que representan la misma rotación. Una forma de saber si $q_1$ y $q_2$ representan la misma rotación es definiendo un producto punto de cuaterniones.

$$\langle q_1, q_2 \rangle = a_1 a_2 + b_1 b_2 + c_1 c_2 + d_1 d_2 \tag{51}$$

Se cumplen las siguientes relaciones.

$$\langle q_1, q_2 \rangle = \begin{Bmatrix} \|q_1\|\|q_2\| \Rightarrow \text{igual rotación, signos iguales} \\ -\|q_1\|\|q_2\| \Rightarrow \text{igual rotación, signos opuestos} \\ \text{otro caso} \Rightarrow \text{rotaciones distintas} \end{Bmatrix} \tag{52}$$

$\langle q_1, q_2 \rangle > 0 \Rightarrow$ signos iguales (apuntan hacia el mismo semiespacio)

$\langle q_1, q_2 \rangle < 0 \Rightarrow$ signos opuestos

$\langle q, q \rangle = \|q\|^2$

Los cuaterniones siempre representan rotaciones, es decir, nunca representan transformaciones de un sistema ortogonal a otro no-ortogonal, mientras que solamente algunas matrices de 3x3 no tienen propiedades deformantes (las matrices de rotación). Debido a esto, al usar cuaterniones se

elimina la posibilidad de que la rotación pueda distorsionarse al realizar cálculos, posibilidad que está siempre presente al usar matrices.

### 4.3.10 Puntocuaternión[7]

Es un objeto matemático de 7 componentes, compuesto por un punto de 3 dimensiones y un cuaternión de 4 dimensiones, que permite especificar una pose de 6 dimensiones en el espacio respecto a un sistema de referencia determinado. También permite especificar una transformación de coordenadas rotación-traslación entre dos sistemas; en estos sentidos, cumple una función similar a la de las matrices homogéneas. Es posible transformar desde puntocuaternión a matriz homogénea de forma inyectiva. Los puntocuaterniones se pueden componer de forma cerrada mediante una multiplicación de derecha a izquierda. Los puntocuaterniones no distorsionan la forma de un conjunto de puntos al transformarlos. Se usará la letra eta ($\eta$) para denotarlos porque es la versión griega de la letra H, la cual es usada para denotar las matrices homogéneas. De este modo, se hace explícita la relación que existe entre ambas.

$$\eta = \begin{pmatrix} x & y & z & a & b & c & d \end{pmatrix}^T = \begin{pmatrix} t \\ q \end{pmatrix} \tag{53}$$

#### 4.3.10.1 Multiplicación de dos puntocuaterniones

La multiplicación entre puntocuaterniones se define del siguiente modo:

$$\eta_1 \cdot \eta_2 = \begin{pmatrix} t_1 \\ q_1 \end{pmatrix} \cdot \begin{pmatrix} t_2 \\ q_2 \end{pmatrix} = \begin{pmatrix} rot(q_1, t_2) + t_1 \\ q_1 \cdot q_2 \end{pmatrix} \tag{54}$$

[7] Introducido en la presente tesis con el objetivo de poder representar la posición y orientación de los landmarks rígidos en el espacio usando una representación única.

#### 4.3.10.2 Grupo de los puntocuaterniones

El conjunto de todos los puntocuaterniones válidos se denominará $W$.

$$W = \left\{ \begin{pmatrix} x & y & z & a & b & c & d \end{pmatrix}^T \in \Re^7 \mid a^2 + b^2 + c^2 + d^2 > 0 \right\} \tag{55}$$

Un puntocuaternión se considera válido cuando la traslación y rotación que representa están determinadas. Los puntocuaterniones no válidos tienen la forma $(x,y,z,0,0,0,0)^T$.

Los puntocuaterniones $W$ junto con la multiplicación $\cdot$ forman un grupo, lo que quiere decir que las reglas matemáticas típicas de la multiplicación no conmutativa (cierre, asociatividad, existencia de identidad e inversos) pueden ser usadas en los cálculos que los involucren, ya que siempre entregarán como resultado un puntocuaternión válido (ver demostración en el anexo).

También se pueden definir las potencias y raíces de un puntocuaternión, donde la expresión $R(q)$ representa la matriz de rotación asociada al cuaternión (ver demostración en el anexo).

$$\eta^K = \begin{cases} q \neq 1 & \begin{pmatrix} (I - R(q))^{-1} t \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ q^K \end{pmatrix} \cdot \begin{pmatrix} (I - R(q))^{-1} t \\ 1 \end{pmatrix}^{-1} \\ q = 1 & \begin{pmatrix} Kt \\ 1 \end{pmatrix} \end{cases} \tag{56}$$

$$\sqrt[k]{\eta} = \eta^{1/k} \tag{57}$$

No se definirá una suma especial para los punto-cuaterniones. Se puede usar la suma vectorial al considerarlos miembros de $\boldsymbol{R}^7$. Sin embargo, la multiplicación de puntocuaterniones no tiene propiedad distributiva sobre dicha suma. Además, la suma vectorial de puntocuaterniones puede producir como resultado un puntocuaternión no válido (que represente una rotación

indeterminada), por lo cual la suma no es una operación cerrada en el espacio de los puntocuaterniones válidos. Además ocurre un tercer problema, que consiste en que q y –q representan la misma rotación, es decir hay aliasing en los valores de q. Debido a esto, hay que reparar la suma y resta vectorial si los cuaterniones involucrados tienen signos opuestos.

$$\begin{aligned} &\langle \eta_1, \eta_2 \rangle = \langle q_1, q_2 \rangle = a_1 a_2 + b_1 b_2 + c_1 c_2 + d_1 d_2 \\ &\langle \eta_1, \eta_2 \rangle > 0 \Rightarrow \text{signos iguales} \\ &\langle \eta_1, \eta_2 \rangle < 0 \Rightarrow \text{signos opuestos} \end{aligned} \tag{58}$$

$$\eta_1 + \eta_2 = \begin{cases} \begin{pmatrix} t_1 + t_2 \\ q_1 + q_2 \end{pmatrix}, & \langle \eta_1, \eta_2 \rangle > 0 \\ \begin{pmatrix} t_1 + t_2 \\ q_1 - q_2 \end{pmatrix}, & \langle \eta_1, \eta_2 \rangle < 0 \end{cases} \tag{59}$$

$$\eta_1 - \eta_2 = \begin{cases} \begin{pmatrix} t_1 - t_2 \\ q_1 - q_2 \end{pmatrix}, & \langle \eta_1, \eta_2 \rangle > 0 \\ \begin{pmatrix} t_1 - t_2 \\ q_1 + q_2 \end{pmatrix}, & \langle \eta_1, \eta_2 \rangle < 0 \end{cases} \tag{60}$$

Esto tiene una importancia fundamental en la etapa de corrección del filtro de Kalman extendido que se explicará posteriormente (ver Capítulo 5).

#### 4.3.10.3 Transformación de coordenadas de un punto usando un puntocuaternión

Se puede mover un punto usando el puntocuaternión de forma directa (*mov*) o inversa (*movi*). Se usa una normalización durante la aplicación del puntocuaternión para evitar que el módulo del cuaternión afecte el punto resultante, lo cual resulta naturalmente al usar la operación *rot* en la definición. Se puede apreciar que las fórmulas del movimiento directo y el inverso son parecidas,

y presentan una complejidad de cálculo similar. Usando la operación *mov* se puede definir el producto de un puntocuaternión con un punto, de modo que se comporte de un modo similar a una matriz homogénea:

$$mov\left(\begin{pmatrix} t \\ q \end{pmatrix}, p\right) = rot(q, p) + t \tag{61}$$

$$movi\left(\begin{pmatrix} t \\ q \end{pmatrix}, p\right) = roti(q, p - t) \tag{62}$$

$$mov\left(\begin{pmatrix} t \\ q \end{pmatrix}, p\right) = (0, I_{3x3})\left\{q \cdot \begin{pmatrix} 0 \\ p \end{pmatrix} \cdot q^{-1}\right\} + t \tag{63}$$

$$movi\left(\begin{pmatrix} t \\ q \end{pmatrix}, p\right) = (0, I_{3x3})\left\{q^{-1} \cdot \begin{pmatrix} 0 \\ p - t \end{pmatrix} \cdot q\right\} \tag{64}$$

$$\eta \cdot p = mov(\eta, p) \tag{65}$$

Se cumplen las siguientes propiedades, las cuales indican que la propiedad distributiva del producto de los puntocuaterniones se puede extender también a la multiplicación de un puntocuaternión y un punto:

$$\eta^{-1} \cdot p = mov(\eta^{-1}, p) = movi(\eta, p) \tag{66}$$

$$(\eta_1 \cdot \eta_2) \cdot p = \eta_1 \cdot (\eta_2 \cdot p) \tag{67}$$

$$(\eta_1 \cdot \eta_2)^{-1} \cdot p = \eta_2^{-1} \cdot (\eta_1^{-1} \cdot p) \tag{68}$$

Se debe apreciar que la cantidad de operaciones necesarias para calcular $\eta \cdot p$ y $\eta^{-1} \cdot p$ son iguales, ya que en ambos casos se debe realizar una rotación de un vector usando un cuaternión más una traslación.

#### 4.3.10.4 Transformación de puntocuaternión a matriz homogénea

Se realiza la conversión del siguiente modo:

$$H(\eta) = H\left(\begin{pmatrix} t \\ q \end{pmatrix}\right) = \begin{pmatrix} rot\left(q, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}\right) & rot\left(q, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}\right) & rot\left(q, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right) & t \end{pmatrix} \tag{69}$$

$$H(\eta) = H\left(\begin{pmatrix} t \\ q \end{pmatrix}\right) = \begin{pmatrix} \frac{a^2+b^2-c^2-d^2}{a^2+b^2+c^2+d^2} & \frac{2bc-2ad}{a^2+b^2+c^2+d^2} & \frac{2bd+2ac}{a^2+b^2+c^2+d^2} & t_X \\ \frac{2bc+2ad}{a^2+b^2+c^2+d^2} & \frac{a^2-b^2+c^2-d^2}{a^2+b^2+c^2+d^2} & \frac{2cd-2ab}{a^2+b^2+c^2+d^2} & t_Y \\ \frac{2bd-2ac}{a^2+b^2+c^2+d^2} & \frac{2cd+2ab}{a^2+b^2+c^2+d^2} & \frac{a^2-b^2-c^2+d^2}{a^2+b^2+c^2+d^2} & t_Z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{70}$$

La operación $H()$ forma un homomorfismo entre los puntocuaterniones y las matrices homogéneas, lo cual les permite compartir sus propiedades matemáticas. Además, la multiplicación de un puntocuaternión con un punto (de la forma definida arriba) produce el mismo efecto que la multiplicación de la matriz homogénea con el punto en coordenadas homogéneas. En las expresiones que siguen a continuación, se usa la expresión $(I_{3x3} \quad 0)$ para descartar la última componente del punto resultante en coordenadas homogéneas:

$$H(\eta_1)H(\eta_2) = H(\eta_1 \cdot \eta_2) \tag{71}$$

$$\eta \cdot p = (I_{3x3} \quad 0)H(\eta)\begin{pmatrix} p \\ 1 \end{pmatrix} \tag{72}$$

$$\eta^{-1} \cdot p = (I_{3x3} \quad 0)H(\eta)^{-1}\begin{pmatrix} p \\ 1 \end{pmatrix} \tag{73}$$

Se puede analizar el jacobiano de la multiplicación entre un puntocuaternión y un punto:

$$\frac{df(p)}{dp} = \begin{pmatrix} \frac{df}{dx} & \frac{df}{dy} & \frac{df}{dz} \end{pmatrix} \tag{74}$$

$$\frac{dmov(\eta, p)}{dp} = H_{3x3}(\eta) \tag{75}$$

$$\frac{dmovi(\eta, p)}{dp} = H_{3x3}{}^{T}(\eta) \tag{76}$$

$$\frac{df(\eta)}{d\eta} = \left( \frac{df}{dx} \quad \frac{df}{dy} \quad \frac{df}{dz} \quad \frac{df}{da} \quad \frac{df}{db} \quad \frac{df}{dc} \quad \frac{df}{dd} \right) \tag{77}$$

$$\frac{dmov(\eta, p)}{d\eta} = \left( I_{3x3} \quad A_1(\eta)p \quad A_2(\eta)p \quad A_3(\eta)p \quad A_4(\eta)p \right) \in \mathbf{M}_{3x7} \tag{78}$$

$$\frac{dmovi(\eta, p)}{d\eta} = \left( -H_{3x3}{}^{T}(\eta) \quad B_1(\eta)(p-t) \quad B_2(\eta)(p-t) \quad B_3(\eta)(p-t) \quad B_4(\eta)(p-t) \right) \in \mathbf{M}_{3x7} \tag{79}$$

$$H\left( (x \quad y \quad z \quad a \quad b \quad c \quad d)^T \right) = H\left( (x \quad y \quad z \quad ua \quad ub \quad uc \quad ud)^T \right) \; con \; u \neq 0 \tag{80}$$

$$H\left( (0 \quad 0 \quad 0 \quad a \quad 0 \quad 0 \quad 0)^T \right) = I_{4\times4} \; con \; a \neq 0 \tag{81}$$

Las expresiones de las matrices de 3x3 $A_i$ y $B_i$ son muy complicadas y su cálculo es algo lento, pero pueden precalcularse para un $\eta$ dado. En consecuencia, las derivadas de $rot(\cdot)$ y $roti(\cdot)$ se pueden precalcular a partir de $q$ usando derivadas obtenidas a partir de la matriz de rotación.

### 4.3.11 Matriz de covarianza de un puntocuaternión

Objeto matemático de 7x7 componentes, es una medida de la dispersión de una distribución de un puntocuaternión en un espacio de 7 dimensiones. En el caso de que la distribución sea Gaussiana, representa completamente la incertidumbre de una pose en el espacio (ver Figura 3).

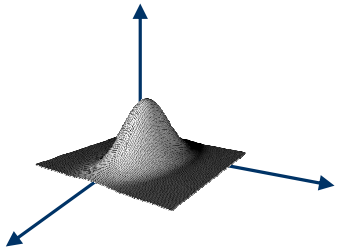


**Figura 3: Incertidumbre respecto de una pose, considera tanto incertidumbre respecto a la posición como a la orientación.**

### 4.3.12 Razones que fundamentan la elección de los puntocuaterniones para representar las poses

Los puntocuaterniones y las matrices homogéneas de rotación-traslación tienen funcionalidades similares y operaciones equivalentes. La matriz homogénea es un objeto más versátil que el puntocuaternión, ya que puede representar transformaciones más generales que la rotación y la traslación. Debido a esto, la conversión de puntocuaternión a matriz homogénea es directa y única; sin embargo, la conversión de matriz homogénea a puntocuaternión no es directa ni única.

Las principales ventajas del uso de los puntocuaterniones son las siguientes:

- Los puntocuaterniones siempre representan transformaciones de tipo rotación-traslación. En cambio las matrices homogéneas pueden contaminarse con distorsión afín y cambio de escala.
- Los puntocuaterniones tienen sólo 7 componentes, mientras que las matrices homogéneas tienen 12. Al usar puntocuaterniones se logran matrices de covarianza menores (7x7=49) que con una matriz homogénea (12x12 = 144).
- La propagación de la incertidumbre de la distribución es más simple debido a que la matriz homogénea debe ser sometida a reparaciones SVD en algunas ocasiones (para eliminar la distorsión afín) y la propagación de covarianza a través de la reparación SVD es un tema muy complicado de abordar (es muy lento de calcular y genera un jacobiano de 12x12).
- La covarianza de una matriz homogénea puede contener covarianzas relacionadas con la ortogonalidad de los ejes del sistema y con el cambio de escala del sistema; este problema no puede ocurrir con los puntocuaterniones.
- El resultado de la aplicación de la matriz homogénea sobre un punto se ve afectado por la magnitud de las componentes de rotación de la matriz homogénea, lo que puede generar efectos no deseados al calcular jacobianos (amplificación de las componentes de rotación). Por otro lado, el resultado de la aplicación del puntocuaternión sobre un punto no se ve afectado por la magnitud de las componentes de rotación, ya que las operaciones *mov* y *movi*

no se ven afectadas por variaciones en la magnitud del cuaternión. Esto se puede expresar de un modo más sencillo en términos matemáticos.

$$\frac{d\left(\begin{pmatrix} \alpha R & t \\ 000 & 1 \end{pmatrix}\begin{pmatrix} p \\ 1 \end{pmatrix}\right)}{d\alpha} \neq 0, \quad \frac{d\left(mov\left(\begin{pmatrix} t \\ \alpha q \end{pmatrix}, p\right)\right)}{d\alpha} = 0 \tag{82}$$

### 4.3.13 Píxel en coordenadas de la imagen

Objeto matemático de 2 dimensiones. Especifica la posición $(a,b)$ de un punto sobre una imagen. El significado geométrico de este objeto es dependiente de los parámetros intrínsecos (distancia focal, resolución y coeficientes de distorsión radial) de la cámara.

### 4.3.14 Matriz de covarianza de un píxel en coordenadas de la imagen

Objeto matemático de 2x2 dimensiones. Especifica la incertidumbre en la posición de un punto en la imagen. El significado geométrico de este objeto es dependiente de los parámetros intrínsecos de la cámara. Debido a que el error de muestreo es independiente en los dos ejes de la imagen, no hay correlación entre esos errores y la matriz resulta ser diagonal.

### 4.3.15 Píxel en coordenadas normalizadas

Objeto matemático de 2 dimensiones. Especifica la orientación $(u,v)$ de un rayo en el espacio que pasa por el centro de la cámara, donde $u$ representa la tangente de la proyección del rayo sobre el plano XY, y $v$ tiene un significado similar sobre el plano XZ. Esto se puede explicar de otro modo definiendo el píxel en coordenadas normalizadas como el vector pendiente del rayo. En consecuencia, este objeto tiene un significado geométrico directo. El rayo en coordenadas normalizadas intersecta a la cámara en un píxel en la imagen proyectada, el cual está especificado en coordenadas de la imagen, por lo que hay

una relación unívoca entre ambos que depende de los parámetros intrínsecos de la cámara.

### 4.3.16 Matriz de covarianza de un píxel en coordenadas normalizadas

Objeto matemático de 2x2 componentes. Representa la incertidumbre de la orientación de un rayo en el espacio que pasa por el centro de la cámara. Debido a que el error de muestreo es independiente en los dos ejes de la imagen, no hay correlación entre esos errores y la matriz resulta ser diagonal.

## 4.4 Geometría proyectiva

En este capítulo se tratarán temas relacionados con transformaciones de coordenadas que relacionan puntos en el espacio y píxeles en la pantalla.

### 4.4.1 Sistemas de referencia usados en robótica móvil y en visión computacional

La convención de ejes usados en robótica móvil y en visión computacional son distintos (ver figura 4). Este es un hecho que se debe tener en consideración al trabajar en enfoques que relacionen ambas disciplinas. A continuación se explicarán las convenciones de ejes usadas en ambas.

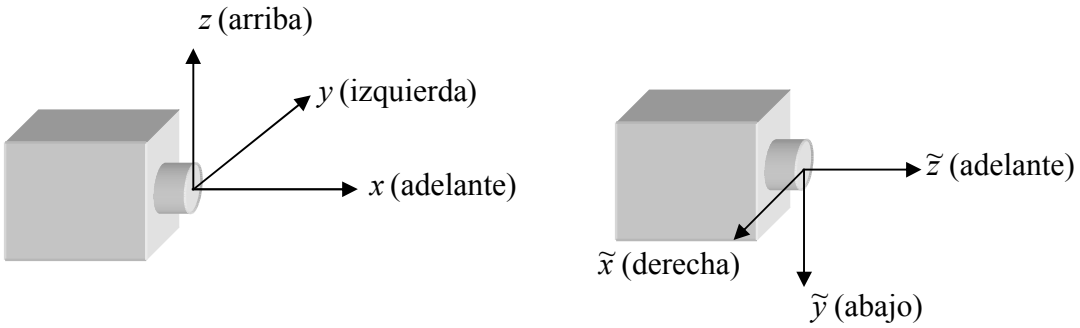



**Figura 4: Convención de ejes (x,y,z) usada en robótica móvil (figura de la izquierda) y convención de ejes (x̃,ỹ,z̃) usada en visión computacional (figura de la derecha)**

La relación entre ambas convenciones de ejes es la siguiente.

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \tilde{z} \\ -\tilde{x} \\ -\tilde{y} \end{pmatrix} \Leftrightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} \quad (83)$$

Al ser éste un trabajo que abarca ambas disciplinas, es necesario el uso de ambas convenciones de ejes en distintas partes del trabajo. El uso de una u otra convención de ejes quedará especificada por la especificación o no del carácter tilda (~) sobre las coordenadas.

### 4.4.2 Proyección

Una cámara se puede representar básicamente como un paralelepípedo con un pequeño agujero (un foco) a través del cual pueden entrar pequeños rayos de luz que proyectan los objetos que están fuera de la cámara sobre la superficie opuesta al agujero (pantalla). Este modelo de cámara es denominado *pinhole*, y corresponde al tipo de cámara que se usaba para sacar las primeras fotografías. La distancia entre el agujero y la imagen proyectada es llamada distancia focal, y es representada usualmente con la letra $f$. Se puede colocar un sistema de ejes X,Y,Z usando la convención de ejes de robótica móvil.

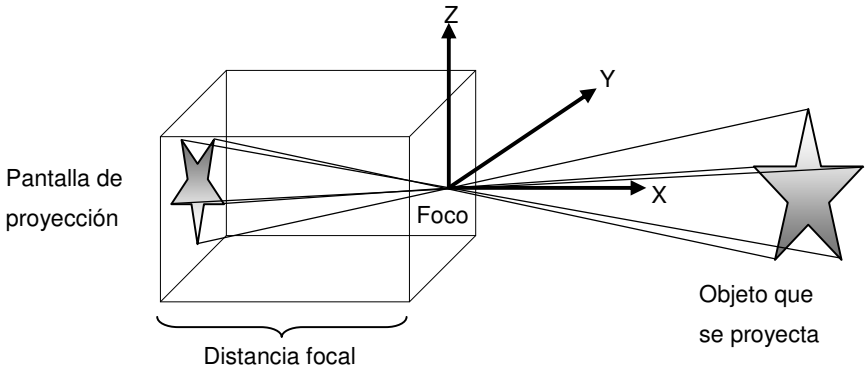



**Figura 5: Proyección real en una cámara pinhole.**

Para poder simplificar los cálculos, se puede colocar la imagen proyectada delante del foco (a la misma distancia focal $f$). Esto se hace con el objetivo de evitar la inversión de la figura. Se debe apreciar que las distancias en la imagen proyectada son las mismas en ambos casos.

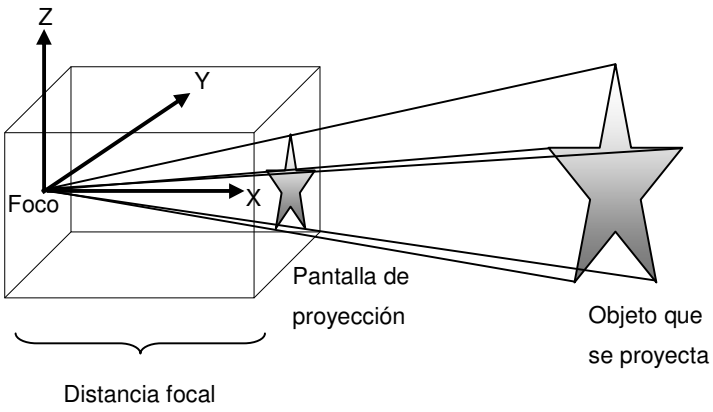



**Figura 6: Proyección virtual en una cámara pinhole.**

Se puede analizar lo que sucede en los ejes X,Y y X,Z cuando se proyecta un punto del espacio en la pantalla.

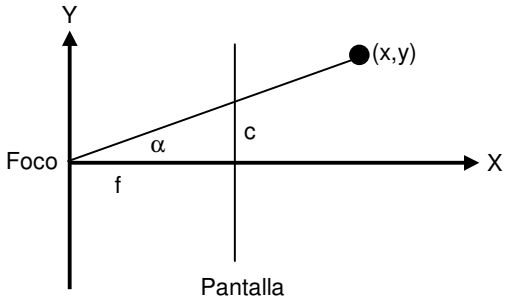



**Figura 7: Proceso de proyección en el plano XY**

En la Figura 7, la distancia focal es $f$ (distancia entre el sensor y la pantalla) y el punto $(x, y)$ se proyecta en la posición $c$ de la pantalla. Todas las variables anteriores se miden en unidades métricas, salvo $\alpha$ que se mide en radianes.

Se cumplen las siguientes relaciones:

$$\frac{y}{x} = \frac{c}{f} = \tan\alpha = u \tag{84}$$

$$u = \frac{y}{x} \tag{85}$$

La cantidad *u* corresponde a la tangente del rayo que sale de la cámara proyectado sobre el plano XY. Dicha cantidad *u* determina únicamente al rayo y, con ello, al punto proyectado sobre el plano de la pantalla.

Al hacer el mismo análisis sobre el plano XZ se llega a conclusiones similares:

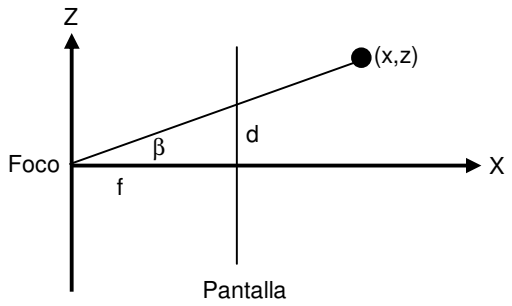



**Figura 8: Proceso de proyección en el plano XZ**

$$\frac{z}{x} = \frac{d}{f} = \tan\beta \tag{86}$$

$$v = \frac{z}{x} \tag{87}$$

Las cantidades $(u,v)$ determinan un único rayo que sale de la cámara, y también indican el punto donde dicho rayo atraviesa la pantalla; $(u,v)$ corresponde a las coordenadas normalizadas del rayo, se pueden intepretar como las tangentes de los ángulos que forma el rayo respecto a los planos XY y XZ. También puede ser interpretado como el vector tangente del rayo. Tras haber explicado las definiciones anteriores se puede dar una definición de la proyección de un punto.

La proyección de un punto se define en este trabajo como un proceso que permite obtener un píxel de 2 componentes (en coordenadas normalizadas) a partir de un punto de 3 componentes y una pose para la cámara (puntocuaternión). Si se considera que el sistema de referencia es solidario a una cámara (bajo la convención de ejes de robótica móvil), las ecuaciones que permiten obtener la proyección y sus derivadas son las siguientes:

$$\begin{pmatrix} u \\ v \end{pmatrix} = proy_1(p) = proy_1\left(\begin{pmatrix} x \\ y \\ z \end{pmatrix}\right) = \begin{pmatrix} y/x \\ z/x \end{pmatrix} \tag{88}$$

$$\frac{dproy_1(p)}{dp} = \begin{pmatrix} -y/x^2 & 1/x & 0 \\ -z/x^2 & 0 & 1/x \end{pmatrix} \tag{89}$$

La operación $proy_1(\cdot)$ representa una proyección sobre el eje x, y es usada usualmente en robótica. También se definirá $proy_2(\cdot)$ como la proyección sobre el eje y, y $proy_3(\cdot)$ como la proyección sobre el eje z, la última de las cuales es usada en visión computacional. Cuando no se especifique cual de las tres se usa (es decir, se indique sólo *proy*), se entenderá que se refiere a $proy_1$.

$$\begin{pmatrix} u \\ v \end{pmatrix} = proy_1(p) = proy_1\left(\begin{pmatrix} x \\ y \\ z \end{pmatrix}\right) = \begin{pmatrix} y/x \\ z/x \end{pmatrix} \text{ (robótica)} \tag{90}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = proy_2(p) = proy_3\left(\begin{pmatrix} x \\ y \\ z \end{pmatrix}\right) = \begin{pmatrix} x/y \\ z/y \end{pmatrix} \tag{91}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = proy_3(p) = proy_3\left(\begin{pmatrix} x \\ y \\ z \end{pmatrix}\right) = \begin{pmatrix} x/z \\ y/z \end{pmatrix} \text{ (visión computacional)} \tag{92}$$

$$proy(p) = proy_1(p) \tag{93}$$

Se pueden definir adicionalmente las funciones $prmov(\cdot)$ y $prmovi(\cdot)$, las cuales permiten obtener la proyección de un punto cuando la cámara tiene poses $\eta^{-1}$ y $\eta$ respectivamente.

$$prmov(\eta, p) = proy_1(\eta \cdot p) \tag{94}$$

$$prmovi(\eta, p) = proy_1(\eta^{-1} \cdot p) \tag{95}$$

### 4.4.3 Píxel en coordenadas homogéneas de la imagen

Objeto matemático de 3 dimensiones. Especifica la posición $(A,B,C)$ de un punto sobre una imagen. También puede representar un punto en la imagen al infinito cuando $C=0$. Este objeto es dependiente de los parámetros intrínsecos de la cámara. Puede ser transformado a un píxel en coordenadas de la imagen mediante la siguiente transformación.

$$\begin{pmatrix} a \\ b \end{pmatrix} = proy_3\left(\begin{bmatrix} A \\ B \\ C \end{bmatrix}\right) = \begin{pmatrix} A/C \\ B/C \end{pmatrix} \tag{96}$$

### 4.4.4 Píxel en coordenadas homogéneas normalizadas

Objeto matemático de 3 dimensiones. Especifica la orientación $(U,V,W)$ de un rayo en el espacio que pasa por el centro de la cámara. También puede representar una orientación paralela al plano de la imagen cuando $W=0$. Este objeto es independiente de los parámetros intrínsecos de la cámara. Puede ser transformado a un píxel en coordenadas normalizadas mediante la siguiente transformación.

$$\begin{pmatrix} u \\ v \end{pmatrix} = proy_3\left(\begin{bmatrix} U \\ V \\ W \end{bmatrix}\right) = \begin{pmatrix} U/W \\ V/W \end{pmatrix} \tag{97}$$

### 4.4.5 Recta en coordenadas homogéneas de la imagen

Objeto matemático de 3 dimensiones. Se suele usar en geometría proyectiva para representar una recta contenida en la pantalla formada por píxeles en coordenadas homogéneas de la imagen. La recta obtenida a partir de los parámetros ($A_R$, $B_R$, $C_R$) se define del siguiente modo:

$$recta\left(\begin{pmatrix} A_R \\ B_R \\ C_R \end{pmatrix}\right) = \left\{ \begin{pmatrix} A \\ B \\ C \end{pmatrix} \mid \begin{pmatrix} A_R \\ B_R \\ C_R \end{pmatrix} \bullet \begin{pmatrix} A \\ B \\ C \end{pmatrix} = 0 \right\} \tag{98}$$

### 4.4.6 Recta en coordenadas homogéneas normalizadas

Objeto matemático de 3 dimensiones. Se suele usar en geometría proyectiva para representar una recta contenida en la pantalla formada por píxeles en coordenadas homogéneas normalizadas. La recta obtenida a partir de los parámetros ($U_R$, $V_R$, $W_R$) se define del siguiente modo:

$$recta\left(\begin{pmatrix} U_R \\ V_R \\ W_R \end{pmatrix}\right) = \left\{ \begin{pmatrix} U \\ V \\ W \end{pmatrix} \mid \begin{pmatrix} U_R \\ V_R \\ W_R \end{pmatrix} \bullet \begin{pmatrix} U \\ V \\ W \end{pmatrix} = 0 \right\} \tag{99}$$

### 4.4.7 Matriz de parámetros intrínsecos de una cámara pinhole

Es una matriz que contiene de los parámetros intrínsecos de una cámara pinhole (las distancias focales en píxeles), los cuales dependen de la resolución de la cámara y el campo de visión. Permite transformar un píxel en coordenadas homogéneas normalizadas a coordenadas homogéneas en la imagen y viceversa.

$$\begin{pmatrix} A \\ B \\ C \end{pmatrix} = \begin{pmatrix} f_{U,PIX} & 0 & a_{CEN} \\ 0 & f_{V,PIX} & b_{CEN} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} U \\ V \\ W \end{pmatrix} \tag{100}$$

$$P = \begin{pmatrix} f_{U,PIX} & 0 & a_C \\ 0 & f_{V,PIX} & b_C \\ 0 & 0 & 1 \end{pmatrix} \tag{101}$$

$$f_{X,PIX} = \frac{res_U / 2}{\tan(FOV_U / 2)} \tag{102}$$

$$f_{Y,PIX} = \frac{res_V / 2}{\tan(FOV_V / 2)} \tag{103}$$

En las ecuaciones anteriores, $f_{U,PIX}$ y $f_{V,PIX}$ son las distancias focales de la cámara en píxeles, $res_U$ y $res_V$ corresponden a la resolución de la imagen en el eje U y V, y $FOV_U$ y $FOV_V$ corresponden a la apertura visual de la cámara (en radianes) en los ejes U y V. Los parámetros $a_{CEN}$ y $b_{CEN}$ corresponden al punto central de la imagen, y usualmente son iguales a la mitad de la resolución en cada eje de la imagen.

### 4.4.8 Matriz de parámetros extrínsecos de una cámara

Es una matriz que indica los parámetros extrínsecos de una cámara, que son básicamente orientación y posición de la cámara en el espacio. Permite transformar las coordenadas homogéneas de un punto en el espacio (convención de ejes usada en visión computacional) a coordenadas homogéneas normalizadas de un píxel. La matriz de parámetros extrínsecos se forma simplemente concatenando la rotación y traslación que lleva el sistema de referencia de la cámara al sistema de referencia global. Se debe notar que esta matriz se puede construir para cualquier modelo de cámara.

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} R & t \end{pmatrix} \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ 1 \end{pmatrix} \tag{104}$$

$$K = \begin{pmatrix} R & t \end{pmatrix} \tag{105}$$

### 4.4.9 Matriz de calibración de una cámara

Es una matriz que se forma al multiplicar las matrices de parámetros intrínsecos y extrínsecos de la cámara. Permite transformar un punto en el espacio (en coordenadas homogéneas bajo la convención de ejes de visión computacional) a coordenadas normalizadas de un píxel en la imagen.

$$\mathrm{C} = PK = \begin{pmatrix} f_{U,PIX} & 0 & a_C \\ 0 & f_{V,PIX} & b_C \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} R & t \end{pmatrix} \tag{106}$$

$$\begin{pmatrix} A \\ B \\ C \end{pmatrix} = \mathrm{C} \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ 1 \end{pmatrix} \tag{107}$$

## 4.5 Estimación robusta de parámetros

La estimación robusta de parámetros corresponde a un conjunto de metodologías que buscan estimar los parámetros de una relación o función matemática que representan adecuadamente un conjunto de datos, suponiendo que esos datos están contaminados con outliers (datos aleatorios que no pertenecen a la función buscada). Hay varios enfoques que se pueden adoptar para resolver este problema, siendo los principales la transformada de Hough, RANSAC y los M-estimadores.

Consideremos una relación matemática $f$ que dependa de un vector de variables $x \in R^X$ y de un vector de parámetros $\theta \in R^U$. La relación matemática $f$ se puede escribir del siguiente modo:

$$f(x, \theta) = 0_{F \times 1} \tag{108}$$

En consecuencia, para cada par de datos y restricciones $(x, \theta)$ posibles hay $F$ restricciones que relacionan ambos datos.

Consideremos el caso donde existen los siguientes datos, dados por un conjunto $D$:

$$D = \{x_i \in R^X, i = 1..N\} \tag{109}$$

El conjunto $D$ está compuesto de datos correctos o *inliers* (que pueden ser representados mediante un mismo parámetro en común $\theta$) y datos incorrectos o *outliers* (que no pueden ser representados mediante el parámetro común $\theta$). Existen varias metodologías que permiten obtener como resultado el parámetro $u$ a pesar de que los datos están contaminados con *outliers*. Dichas metodologías son denominadas técnicas de estimación robusta de parámetros. Dentro de dichas metodologías existen tres familias de métodos utilizadas en esta tesis, las cuales detallan a continuación:

### 4.5.1 Transformada Hough

La transformada Hough [2] es un método que verifica todos los parámetros posibles para encontrar el óptimo. Se define una función $C(x)$, la cual es un generador del conjunto de parámetros compatibles con un dato $x$.

$$C(x) = \{\theta \in R^U \mid f(x, \theta) = 0_{F\times 1}\} \tag{110}$$

Si todos los datos son *inliers*, el set de parámetros comunes puede obtenerse como la intersección de los conjuntos compatibles con cada dato. En otras palabras, la solución es el conjunto de parámetros que es compatible con todos los datos al mismo tiempo.

$$C(x_1) \cap C(x_2) \cap ... \cap C(x_N) = \{\theta\} \tag{111}$$

En el caso de que existan *outliers*, se puede generar un sistema basado en votaciones. Inicialmente se genera con un espacio denominado espacio de Hough, el cual tiene la misma dimensionalidad que el espacio de los parámetros y parte

inicializado con un valor de cero sobre todo el espacio. Cada dato $x$ genera un conjunto de parámetros compatibles $C(x)$ y se agrega un voto a todos los parámetros compatibles dentro del espacio de Hough. Al repetir el proceso con todos los datos, los parámetros que son compatibles con una gran cantidad de datos obtienen una votación alta en el espacio de Hough. En consecuencia, buscando parámetros en el espacio de Hough que tengan una alta votación se pueden encontrar los parámetros correspondientes a los *inliers* contenidos en los datos. Sin embargo, la cantidad de parámetros por los cuales se debe votar para cada dato crece exponencialmente con la dimensionalidad de $C(x)$, que es igual al número de parámetros $U$ menos el número de restricciones $F$. En consecuencia, este método sólo funciona eficientemente para dimensionalidades bajas del vector de parámetros. El generar restricciones extras a partir de un análisis cuidadoso del problema favorece la eficiencia de este método [2].

### 4.5.2 RANSAC

En este método se crean subconjuntos de datos denominados muestras mínimas [23]. Cada muestra mínima $M_i$ está compuesta por un conjunto de datos elegidos al azar, los cuales permiten fijar un único parámetro $u$ que se denomina la hipótesis $H_i$ obtenida a partir de $M_i$. Como los parámetros son de dimensionalidad $U$, y existen $F$ ecuaciones disponibles para cada dato, el agregar un dato a la muestra mínima permite especificar $F$ restricciones. Para definir una muestra mínima se deben fijar los $U$ parámetros, por lo cual se necesitan $L=U/F$ datos para poder generar una muestra mínima, redondeando dicha cantidad hacia el entero superior. Una muestra mínima tiene la siguiente expresión:

$$M_i = \{x_a, x_b, ..., x_L\} \tag{112}$$

Donde la dimensionalidad de la muestra mínima es:

$$L = \lceil U / F \rceil \tag{113}$$

En el caso de que una muestra mínima esté compuesta solamente por *inliers*, se dirá que esa muestra mínima es un *inlier*. Dada la probabilidad $p$ de que un elemento de los datos sea un *inlier*, la probabilidad $P$ de que la muestra sea un inlier es la siguiente:

$$P = p^L \tag{114}$$

De la expresión anterior se puede apreciar que la probabilidad $P$ de que la muestra mínima sea un *inlier* disminuye exponencialmente con la dimensionalidad $L$ de la muestra mínima. A su vez, dicha dimensionalidad $L$ es proporcional a la dimensionalidad $P$ de los parámetros. En consecuencia, para problemas donde los vectores de parámetros tienen dimensionalidad alta, es muy poco probable que una muestra sea un *inlier*, por lo cual la efectividad de RANSAC disminuye con la dimensionalidad del espacio de los parámetros.

EL método RANSAC [23] se basa en elegir muestras mínimas al azar para obtener hipótesis $H_i$. Estas hipótesis son verificadas contra todos los datos para evaluar el consenso de la hipótesis, el cual corresponde a la cantidad de datos que son compatibles con $H_i$. Si el consenso sobrepasa un umbral, significa que la hipótesis $M_i$ representa adecuadamente los datos, por lo cual se acepta como el vector de parámetros correcto que representa a los *inliers*. Existen muchas variantes de RANSAC, las cuales se diferencian por la forma de elegir los datos para formar las muestras mínimas [10], por la forma de evaluar el consenso de la muestra mínima [95] y por el orden en el cual se evalúan las hipótesis [11][73], ya que en algunos métodos se generan varias hipótesis de forma paralela y se descartan progresivamente al ir evaluando si son compatibles con los datos.

### 4.5.3 M-estimadores

Los M-estimadores corresponden a un enfoque teórico de la estimación robusta más que a una metodología debido a que su solución es un problema de optimización global multimodal [37], aunque pueden ser usados en la práctica para mejorar la estimación inicial lograda usando la transformada Hough o RANSAC.

Los M-estimadores están basados en la minimización de una función de error formada por una suma de errores elementales.

$$\hat{\theta} = \arg\min_{\theta}\left(\sum_{i=1}^{n}\rho(x_i,\theta)\right) \tag{115}$$

El error cuadrático aumenta mucho al agregar un outlier, ya que el nuevo término de error cuadrático que se genera amplifica mucho el efecto del error total. La función de error $\rho(x_i,\theta)$ puede ser elegida para atenuar el efecto que provocan los outliers sobre la solución final al dar mayor importancia a los errores pequeños que representan de mejor forma los parámetros que se desean estimar. Ejemplos de funciones de error robustas a outliers son el valor absoluto del error, el error de Huber (que se comporta como un error cuadrático para errores pequeños y como un valor absoluto para errores mayores) y el error bicuadrático de Tukey.

**El valor absoluto del error (LMS):** $\rho(x_i,\theta) = \left\|f(x_i,\theta)\right\|$ (116)

**El error de Huber:** $\rho(x_i,\theta) = \begin{cases} \left\|f(x)\right\|^2/2 & \text{para } |x| < c \\ c\left(\left\|f(x)\right\| - c/2\right) & \text{para } |x| \geq c \end{cases}$ (117)

**El error bicuadrático de Tukey:** $\rho(x_i,\theta) = \begin{cases} x\left(1-\dfrac{x^2}{c^2}\right)^2 & \text{para } |x| < c \\ 0 & \text{para } |x| \geq c \end{cases}$ (118)

La función $\rho(x_i,\theta)$ puede ser transformada a una forma cuadrática que puede ser minimizada usando el algoritmo de minimización iterativa Levenberg-Marquardt, lo cual permite una convergencia rápida.

$$\sigma(x_i,\theta) = \sqrt{\rho(x_i,\theta)} \tag{119}$$

$$\hat{\theta} = \arg\min_{\theta}\left(\sum_{i=1}^{n}\sigma^2(x_i,\theta)\right) \tag{120}$$

## 4.6 Algoritmo de los 3 puntos

El algoritmo de los tres puntos [33] tiene como objetivo encontrar la pose de un triángulo en el espacio a partir de las proyecciones del triángulo en un plano. Dados los lados a,b,c de un triángulo A,B,C y los píxeles P,Q,R que corresponden a las proyecciones del triángulo, encuentra cuatro posibles triángulos en el espacio que son compatibles con los datos entregados. A continuación se define el procedimiento alg3p_v1( ), el cual recibe tres proyecciones y los lados de un triángulo, y entrega cuatro posibles tripletas de puntos, cada una de las cuales indica la posición de los puntos del triángulo en el espacio:

$$\begin{aligned}&(A[1...4], B[1...4], C[1...4]) = \text{alg 3 p\_v 1}(a, b, c, P, Q, R)\\ &\Rightarrow \forall i = 1..4, \begin{cases} a = \|B[i] - C[i]\|, b = \|A[i] - C[i]\|, c = \|A[i] - B[i]\| \\ P = proy(A[i]),\ Q = proy(B[i]),\ R = proy(C[i]). \end{cases}\end{aligned} \quad (121)$$

El algoritmo de los 3 puntos se basa en aplicar la ley de los cosenos a los ángulos $\alpha$, $\beta$, $\gamma$ que se forman al considerar los triángulos BOC, COA, AOB (con la letra O representando el origen del sistema de coordenadas). La ley de los cosenos permite calcular la distancia a los puntos del triángulo, y las proyecciones P,Q,R permiten calcular la dirección de los puntos [33]:

$$s_1 = \|A\|, s_2 = \|B\|, s_3 = \|C\| \quad (122)$$

$$s_2{}^2 + s_3{}^2 - 2 s_2 s_3 \cos\alpha = a^2 \quad (123)$$

$$s_1{}^2 + s_3{}^2 - 2 s_1 s_3 \cos\beta = b^2 \quad (124)$$

$$s_1{}^2 + s_2{}^2 - 2 s_1 s_2 \cos\gamma = c^2 \quad (125)$$

$$\Rightarrow s_1[1..4],\ \ s_2[1..4],\ \ s_3[1..4] \quad (126)$$

$$\Rightarrow A[1...4] = s_1[1...4] * \hat{A},\ \ B[1...4] = s_2[1...4] * \hat{B},\ \ C[1...4] = s_3[1...4] * \hat{C} \quad (127)$$

El algoritmo se puede reescribir del siguiente modo: dadas las proyecciones $P,Q,R$ y puntos iniciales $A,B,C$, encontrar la transformación $\eta$ que permite que los

puntos $A$,$B$,$C$ se proyecten en $P$,$Q$,$R$. Para esto se explicará una forma que permita encontrar la transformación entre 3 correspondencias de puntos $A$-$A$', $B$-$B$', $C$-$C$'.

$$H_T(A,B,C)=\begin{pmatrix}\hat{p}_1 & \hat{p}_2 & \hat{p}_3 & A\\ 0 & 0 & 0 & 1\end{pmatrix},\ p_1=B-A,\ p_T=C-A,\ p_2=p_T-\hat{p}_1(\hat{p}_1\bullet p_T),\ \hat{p}_3=\hat{p}_1\times\hat{p}_2 \quad (128)$$

La transformación $H_T$ corresponde a un sistema de coordenadas solidario a los puntos ($A$,$B$,$C$), centrado en el punto A, apuntando hacia el eje B y a cuya izquierda se encuentra el punto C. Al ser solidario a los puntos, se cumple la siguiente propiedad:

$$MH_T(A,B,C)=\begin{pmatrix}R & t\\ 000 & 1\end{pmatrix}\begin{pmatrix}B-A & C-A & D-A & A\\ 0 & 0 & 0 & 1\end{pmatrix} \quad (129)$$

$$=\begin{pmatrix}R(B-A) & R(C-A) & R(D-A) & RA+t\\ 0 & 0 & 0 & 1\end{pmatrix} \quad (130)$$

$$=\begin{pmatrix}RB+t-(RA+t) & RC+t-(RA+t) & RD+t-(RA+t) & RA+t\\ 0 & 0 & 0 & 1\end{pmatrix} \quad (131)$$

$$=\begin{pmatrix}MB-MA & MC-MA & MD-MA & MA\\ 0 & 0 & 0 & 1\end{pmatrix} \quad (132)$$

$$=H_T(MA,MB,MC) \quad (133)$$

$$MH_T(A,B,C)=H_T(MA,MB,MC) \quad (134)$$

$$M=H_T(MA,MB,MC)H_T(A,B,C)^{-1} \quad (135)$$

Esta propiedad se puede usar para encontrar la transformación que lleva los puntos $A'$, $B'$, $C'$ a los puntos $A$, $B$, $C$:

$$(A'[1...4],B'[1...4],C'[1...4])=\mathrm{alg5p\_v1}(a,b,c,P,Q,R) \quad (136)$$

$$A=H[i]A'[i],\quad B=H[i]B'[i],\quad C=H[i]C'[i],\ i=1..4 \quad (137)$$

$$H_T(H[i]A'[i],H[i]B'[i],H[i]C'[i])=H_T(A,B,C),\ i=1..4 \quad (138)$$

$$H[i]H_T(A'[i],B'[i],C'[i])=H_T(A,B,C),\ i=1..4 \quad (139)$$

$$H[...] = H_T(A,B,C)H_T{}^{-1}(A'[...],B'[...],C'[...]) \rightarrow \eta[1...4] \quad (140)$$

En consecuencia, la segunda versión del algoritmo de los 3 puntos se define del siguiente modo:

$$\text{alg}\,3\,\text{p\_v}\,2(A,B,C,P,Q,R): \quad (141)$$

$$\bullet\ a = \|B-C\|, b = \|A-C\|, c = \|A-B\| \quad (142)$$

$$\bullet\ (A'[1...4],B'[1...4],C'[1...4]) = \text{alg}\,3\,\text{p\_v}\,1(a,b,c,P,Q,R) \quad (143)$$

$$\bullet\ H[1...4] = H_T(A,B,C)H_T(A'[1...4],B'[1...4],C'[1...4])^{-1} \rightarrow \eta[1...4] \quad (144)$$

Se puede apreciar que cada $\eta$ es una transformación que lleva los puntos $A$, $B$, $C$ a aquellos que generan las proyecciones $P$, $Q$, $R$. Hay cuatro posibles transformaciones que cumplen esta condición.

$$\Rightarrow P = proy(\eta[1...4]\cdot A),\ Q = proy(\eta[1...4]\cdot B),\ R = proy(\eta[1...4]\cdot C) \quad (145)$$

Se puede apreciar que el algoritmo de los 3 puntos versión 2 entrega 4 soluciones, de las cuales sólo una es útil. Para resolver al ambigüedad, se puede usar una cuarta correspondencia *D-D'* para encontrar el $\eta$ que genera el menor error de transformación. Se usará la notación *alg3p*( ), que es la definitiva, para representar la versión del algoritmo que resuelve la ambigüedad usando un cuarto punto, la cual es denominada algoritmo de 3+1 puntos:

$$\eta_i = \text{alg}\,3\,\text{p\_v}\,2(A,B,C,P,Q,R)[i],\ i = 1,...,4 \quad (146)$$

$$\text{alg}\,3\,\text{p}(A,B,C,D,P,Q,R,S) = \arg\min_{\eta_i}\left(\|proy(\eta_i\cdot D)-S\|^2\right) \quad (147)$$

$$\eta = \text{alg}\,3\,\text{p}(A,B,C,D,P,Q,R,S) \quad (148)$$

$$\Rightarrow P = proy(\eta\cdot A),\ Q = proy(\eta\cdot B),\ R = proy(\eta\cdot C),\ S \approx proy(\eta\cdot D). \quad (149)$$

# 5 SLAM visual basado en landmarks tridimensionales rígidos

## 5.1 Descripción del sistema

El sistema de SLAM visual basado en landmarks tridimensionales rígidos, está inspirado en el sistema MonoSLAM [17] de Andrew Davison. En el sistema MonoSLAM se usa un estimador de estado EKF para poder generar una estimación de la posición y orientación de una cámara móvil, además de un mapa formado por landmarks puntuales, los cuales pueden ser representados por puntos normales o por puntos con profundidad inversa.

## 5.2 Formulación de los pasos del algoritmo

El algoritmo SLAM desarrollado consta de las siguientes etapas: detección de descriptores SURF; generación de correspondencias entre los descriptores SURF de la imagen y los landmarks punto, landmarks con profundidad inversa y landmarks rígidos en la imagen; etapa de predicción del filtro EKF; etapa de actualización del filtro EKF; normalización de los cuaterniones; colapso de landmarks con profundidad inversa; colapso de los landmarks punto en landmarks rígidos; generación de los landmarks con distancia inversa y eliminación de landmarks con distancia inversa. A continuación se describen las etapas señaladas.

1. *Detección de descriptores SURF*: Un conjunto de descriptores SURF es detectado en la imagen actual y transformado a coordenadas normalizadas como se indica en la Sección 4.4.7

2. *Generación de correspondencias entre los descriptores SURF de la imagen y los landmarks punto, landmarks con profundidad inversa y landmarks rígidos*: Se generan correspondencias entre los descriptores de la imagen

actual y los elementos del mapa usando el sistema L&R, como se explica en la Sección 5.8.

3. *Etapa de predicción EKF*: El estado $x_k$ y la matriz de covarianza del estado $P_k$ se actualizan usando el procedimiento EKF estándar [99]. La actualización se detalla en la Sección 5.3.

4. *Etapa de actualización EKF*: El modelo de observación se usa para corregir el estado del sistema y su covarianza usando la diferencia entre la observación real y la esperada.

Como es usual, la innovación $y_k$ y su covarianza $S_k$ se calculan del siguiente modo:

$$y_k = z_k - H_k \cdot x_k^- \quad (150)$$

$$S_k = H_k \cdot P_k^- \cdot H_k^T + R_k. \quad (151)$$

Se deben considerar cuatro casos distintos para el cálculo de la innovación:

- En el caso de los landmark punto, $z_k$, $H_k$ y $R_k$ se calculan según las ecuaciones (136) y (137)
- En el caso de los landmarks con profundidad inversa, $z_k$, $H_k$ y $R_k$ se calculan según las ecuaciones (138) y (139).
- En el caso de los landmarks rígidos, $z_k$, $H_k$ y $R_k$ se calculan según las ecuaciones (144), (148) y (149).
- En el caso de que una observación virtual no se pueda generar debido a que no se observan suficientes puntos del cuerpo, las observaciones individuales de los puntos del cuerpo se pueden usar en la corrección. En este caso, $z_k$, $H_k$ y $R_k$ se calculan según las ecuaciones (153), (154) y (155).

La corrección se puede realizar de un modo rápido mediante el procedimiento detallado en la Sección 5.9.

5. *Normalización de los cuaterniones*: Se aplica una función de normalización al estado del sistema que deja los cuaterniones con valor absoluto igual a uno. Al ser una transformación aplicada a la distribución de probabilidad que representa el estado del sistema, se usa el método de propagación de la incertidumbre para obtener una matriz de covarianza consistente.

6. *Colapso de los landmarks con profundidad inversa*: Los landmarks con profundidad inversa usan 6 componentes del estado. Pueden ser transformados en un landmark punto de 3 componentes cuando la incertidumbre del punto resultante puede ser aproximada apropiadamente por una Gaussiana. Este proceso se detalla en la Sección 5.10.

7. *Colapso de los landmarks punto en landmarks rígidos*: La matriz de covarianza es analizada para evaluar si existe un conjunto de landmarks punto que puedan ser colapsados en un landmark rígido con poca pérdida de información. En este caso, se realiza un cambio de representación. Este proceso se detalla en la Sección 5.11

8. *Generación de landmarks con distancia inversa*: Los descriptores obtenidos a partir de la imagen actual que no corresponden a ninguno de los landmarks existentes en el mapa se pueden usar para generar nuevos landmarks. Este procedimiento se detalla en la Sección 5.8 y en [12].

9. *Eliminación de landmarks con distancia inversa*: Los landmarks nuevos deben ser observados durante una cierta cantidad de frames para ser confirmados. Si el número de frames que un landmark no es observado en esta etapa inicial supera un cierto umbral, dicho landmark es eliminado. Este procedimiento se detalla en la Sección 5.8.

## 5.3 Estimación del estado del sistema

El estado del sistema contiene:

1. El estado de la cámara. Está formado por un puntocuaternión $\eta_{CAM\text{-}MAPA} = (t_{CAM\text{-}MAPA}, q_{CAM\text{-}MAPA})$ que representa la pose de la cámara, y dos vectores $v_{CAM\text{-}MAPA}$ y $\omega_{CAM\text{-}MAPA}$ que representan la velocidad lineal y angular respectivamente.
2. El estado de los landmarks, representado como $x_{LANDMARKS}$.

Se usa un filtro EKF con covarianza completa para estimar el estado del sistema. Las ecuaciones que describen al sistema son las siguientes:

$$f_{CAMARA} = \begin{pmatrix} t_{CAM-MAPA} \\ q_{CAM-MAPA} \\ v_{CAM-MAPA} \\ \omega_{CAM-MAPA} \end{pmatrix} = \begin{pmatrix} t_{CAM-MAPA} + (v_{CAM-MAPA} + n_{V(k)}) \cdot \Delta t \\ quat\left((\omega_{CAM-MAPA} + n_{W(k)})\Delta t\right) \cdot q_{CAM-MAPA} \\ v_{CAM-MAPA} + n_{V(k)} \\ \omega_{CAM-MAPA} + n_{W(k)} \end{pmatrix} \tag{152}$$

con

$$quat(v) = \begin{pmatrix} \cos\|v/2\| \\ \frac{v_X}{\|v\|}\sin\|v/2\| \\ \frac{v_Y}{\|v\|}\sin\|v/2\| \\ \frac{v_Z}{\|v\|}\sin\|v/2\| \end{pmatrix} \tag{153}$$

y

$$n_{V(k)} \sim N(0, P_V),\ \ n_{W(k)} \sim N(0, P_W). \tag{154}$$

La covarianza completa es necesaria en el caso de una sola cámara debido a que la escala del mapa está indeterminada. Sin covarianza completa no es posible la propagación de información necesaria para converger a una escala global para el mapa, ya que cada observación de un punto entrega sólo información angular y no de distancia (cada par robot-landmark no tiene observabilidad completa). Debido a que la escala entre el mapa obtenido y el mundo real converge a una cantidad arbitraria, se puede llegar a una escala negativa, la cual produce un mundo reflejado con la cámara sin reflejar, lo cual es un resultado coherente pero poco deseable. Este problema se puede solucionar en la práctica eligiendo condiciones iniciales apropiadas al inicializar un landmark.

El uso de landmarks tridimensionales permite obtener observaciones que entregan información sobre distancias, lo cual podría permitir tratarlos de un modo distinto (no considerar sus covarianzas cruzadas, ya que tienen observabilidad completa). Esto permitiría acelerar enormemente el SLAM dar la posibilidad de usar una mayor cantidad de landmarks, ya que se elimina la restricción O($N^2$) que imponen las covarianzas cruzadas cambiándola por una O($M^2$+$N$), donde $M$ es el número de landmarks-punto y $N$ es el número de landmarks tridimensionales, el cual puede llegar a ser enorme sin dar problemas de velocidad o almacenamiento.

## 5.4 Propagación de la incertidumbre y obtención de las matrices de covarianza

Debido a que existen estructuras con limitaciones de rango (ángulos, cuaterniones) se usarán jacobianos para propagar la incertidumbre de la distribución y obtener la nueva matriz de covarianza. Se podría probar la propagación usando transformada unscented [39], pero queda abierta la posibilidad de que los sigmapuntos obtenidos sean, en algunos casos, objetos matemáticos no coherentes debido al aliasing que presentan los cuaterniones, y a la necesidad de mantenerlos normalizados. Se usan cálculos parciales de jacobianos exactos y regla de la cadena (multiplicación de jacobianos) para todas las propagaciones de covarianza que se necesitan, excepto aquellas que involucran cálculos iterativos forzosamente.

$$X \sim N(m, C) \Rightarrow f(X) \sim N\left(f(m), FCF^T\right) \tag{155}$$

$$F = \frac{df(x)}{dx} | x = m \tag{156}$$

En algunos casos, sólo algunas variables son afectadas por la transformación. Si el estado x se divide en variables involucradas $A$ y variables no involucradas $B$, el procedimiento de propagación de incertidumbre es el siguiente.

$$X = \begin{pmatrix} A \\ B \end{pmatrix} \sim N\left(\begin{pmatrix} m_A \\ m_B \end{pmatrix}, \begin{pmatrix} C_{AA} & C_{AB} \\ C_{BA} & C_{BB} \end{pmatrix}\right) \tag{157}$$

$$\Rightarrow f(X) = \begin{pmatrix} f_A(A) \\ B \end{pmatrix} \sim N\left(\begin{pmatrix} f_A(m_A) \\ m_B \end{pmatrix}, \begin{pmatrix} FC_{AA}F_T & FC_{AB} \\ C_{BA}F^T & C_{BB} \end{pmatrix}\right) \tag{158}$$

$$F = \frac{df_A(a)}{da} | a = m_A \tag{159}$$

En el caso en que las variables involucradas y no involucradas estén mezcladas en distintas posiciones en el vector de estado, deben reordenarse de tal modo que se pueda aplicar el proceso anterior. Una vez obtenidas las nuevas covarianzas, los elementos de las matrices de covarianza deben devolverse a su orden original.

## 5.5 Tipos de landmarks

### 5.5.1 Landmark punto

Es un landmark que corresponde a una estructura puntual en el mundo real. Se representa en el sistema mediante un punto ($x,y,z$), referida al sistema de referencia global. Se puede aproximar la observación del landmark real usando la proyección del punto.

### 5.5.2 Landmark punto con profundidad inversa

Es un landmark que corresponde a una estructura puntual en el mundo real. Se representa en el sistema mediante un punto con profundidad inversa $(x,y,z,\rho,\theta,\phi)$ [12], referido al sistema de referencia global. Se puede aproximar la observación del landmark real usando la proyección del punto equivalente.

### 5.5.3 Landmark rígido

Es un landmark que corresponde a la pose (posición y orientación) de una estructura tridimensional en el mundo real. Corresponde a un cuerpo rígido (*rigid body landmark*) que tiene una cierta posición y orientación en el espacio. El cuerpo rígido está formado por puntos, los cuales se denominarán *body points* o puntos del cuerpo del landmark. Se representa en el sistema mediante una pose codificada usando un puntocuaternión $(x,y,z,a,b,c,d)^T$ referido al sistema de referencia global, y un conjunto de puntos del cuerpo $(\Pi_1,\ldots,\Pi_N)$ referidos al sistema de referencia del landmark rígido. Los puntos del cuerpo, al estar referidos al sistema del landmark rígido, se mantienen constantes en el tiempo aunque la pose del landmark rígido se modifique. A partir de las proyecciones de los puntos del cuerpo sobre la cámara se puede estimar la pose del landmark tridimensional relativa a la cámara, la cual se usa como una observación virtual del landmark.

Un conjunto de landmarks-punto se puede colapsar en un landmark tridimensional. Este proceso produce pérdida de información (covarianzas cruzadas) debido a que se pierde la relación individual de cada landmark punto con el resto de los landmarks (covarianzas cruzadas), ya que dicho landmark punto pasa a ser un punto del cuerpo del landmark tridimensional. Como resultado del proceso, se obtiene un puntocuaternión que representa al landmark tridimensional y el conjunto de puntos del cuerpo.

Se puede observar que todos los landmarks usados tienen significados geométricos fácilmente comprensibles (posiciones o poses), lo cual es una característica necesaria para el mapa obtenido.

## 5.6 Observación de landmarks

Los landmarks puntuales corresponden a estructuras puntuales visualmente salientes en el mundo real. Para detectarlos se usan descriptores SURF, los cuales están basados en convoluciones con funciones rectangulares, lo cual permite una enorme velocidad de cálculo usando imágenes integrales. Sin embargo, al usar rectángulos para detectar los puntos de interés, genera algunos puntos que corresponden a líneas en la imagen, lo cual no sucedería si se usaran filtros Gaussianos. Los puntos de interés sobre líneas son poco deseables debido a que suelen moverse entre frames consecutivos, ya que pueden aparecer en distintas zonas de una misma línea; es decir, su posición es poco repetible. Para eliminar este tipo de puntos se aplica un filtro de Harris localmente en cada punto, lo cual es más rápido que aplicar el filtro de Harris sobre toda la imagen. Los pixels que generan valores negativos al ser filtrados usando Harris suelen corresponder a puntos ubicados sobre líneas y son eliminados. El test basado en Harris puede aplicarse después de realizar un descarte de los descriptores mediante asociación de datos, lo cual permite que se pueda aplicar de forma muy rápida.

Las observaciones, que corresponden a los puntos de interés detectados, son comparadas con las observaciones estimadas que son generadas proyectando puntos tridimensionales $l_i$ que corresponden a landmarks punto, landmarks con profundidad inversa o landmarks rígidos en las coordenadas de los píxeles[8]:

$$\begin{pmatrix} u \\ v \end{pmatrix} = h_{uv}^{(i)} = proy\left(\eta_{CAM-MAPA}{}^{-1} \cdot l_i\right) \tag{160}$$

Los píxeles en coordenadas normalizadas son llevados a píxeles en coordenadas de la imagen con el objetivo de ser procesados por el sistema de asociación de datos, el cual trabaja en el dominio de la imagen. Los píxeles en coordenadas de la imagen se obtienen usando la distancia focal en $x$ e $y$:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} u \cdot distFoc_x \\ v \cdot distFoc_y \end{pmatrix} \tag{161}$$

[8] La función de proyección $proy(\ )$ se define como $proy(x,y,z) = (y/x, z/x)$ en la Sección 4.4

En el caso de los landmarks punto, los puntos a proyectar son aquellos que definen el landmark ($p_i$). En el caso de los landmarks rígidos, los puntos a proyectar son los puntos del cuerpo del landmark, cuya posición está dada por $p_i = \eta_{LN} \cdot \Pi_i + t_{LN}$, donde $\eta_{LN}$ corresponde al puntocuaternión que indica la pose del landmark rígido.

Finalmente, en el caso de los landmarks con profundidad inversa, las coordenadas de los puntos a proyectar están dados por:

$$l_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} + \frac{1}{\rho_i} \begin{pmatrix} \cos(\phi_i)\cos(\theta_i) \\ \cos(\phi_i)\sin(\theta_i) \\ \sin(\phi_i) \end{pmatrix} \quad (162)$$

El modelo plano es detectado en el conjunto de descriptores obtenidos de la imagen mediante la búsqueda de una transformación de semejanza que relacione ambos conjuntos de descriptores. La transformación de semejanza es calculada usando el sistema de detección de imágenes L&R [49][50], el cual usa una transformada Hough y varias etapas para rechazar transformaciones incorrectas. Aquellas transformaciones con escala o traslación excesivas son rechazadas. Un test chi-cuadrado se realiza para eliminar falsas detecciones que hayan podido sobrevivir a las etapas de descarte. Para los landmarks rígidos, se usan matrices de covarianza $S$ de tamaño 7x7, mientras que para los landmarks puntuales se usan matrices de 2x2. La etapa de búsqueda de transformaciones de semejanza se puede relajar cuando la cámara se pierde.

Este sistema no usa tracking de puntos sino detección de los landmarks en cada frame. En consecuencia, el problema del *loop closure*[9] se resuelve naturalmente cuando se encuentran descriptores antiguos.

[9] *Loop closure* se refiere al evento que ocurre cuando el robot entra en una zona ya conocida.

## 5.7 Funciones de observación

### 5.7.1 Funciones de observación relacionadas con la cámara primaria

Un landmark punto tiene una función de observación[10] $h_{LAND\text{-}CAM\text{-}P}(\cdot)$, la cual depende de la pose estimada de la cámara $\eta_{CAM\text{-}MAPA}$ y del punto observado $p_{LAND\text{-}MAPA}$. Su jacobiano $H_{LAND\text{-}CAM\text{-}P}(\cdot)$ es usado para propagar la incertidumbre de la observación.

$$h_{LAND-CAM-P}(x) = proy\left(\eta_{CAM-MAPA}^{-1} \cdot p_{LAND-MAPA}\right) \tag{163}$$

$$H_{LAND-CAM-P} = \frac{dh_{LAND-CAM-P}(x)}{dx} = \left[\frac{dh_{LAND-CAM-P}(x)}{d\eta_{CAM-MAPA}} \quad \frac{dh_{LAND-CAM-P}(x)}{d\eta_{LAND-MAPA}}\right] = \left[H_C \quad H_L\right] \tag{164}$$

Un landmark punto con profundidad inversa tiene una función de observación $h_{LAND\text{-}CAM\text{-}PI}(\cdot)$, la cual depende de la pose estimada de la cámara $\eta_{CAM\text{-}MAPA}$ y del punto con profundidad inversa observado $p_{LAND\text{-}MAPA}$. Su jacobiano $H_{LAND\text{-}CAM\text{-}PI}(\cdot)$ es usado para propagar la incertidumbre de la observación.

$$h_{LAND-CAM-PI}(x) = proy\left(\eta_{CAM-MAPA}^{-1} \cdot pun(p_{LAND-MAPA})\right) \tag{165}$$

$$H_{LAND-CAM-PI} = \frac{dh_{LAND-CAM-PI}(x)}{dx} = \left[\frac{dh_{LAND-CAM-PI}(x)}{d\eta_{CAM-MAPA}} \quad \frac{dh_{LAND-CAM-PI}(x)}{dp_{LAND-MAPA}}\right] = \left[H_C \quad H_L\right] \tag{166}$$

Se puede generar una observación virtual de la pose de un landmark tridimensional mediante un proceso de tres etapas:

[10] La función de proyección $proy(\cdot)$ se define como $proy(x,y,z) = (y/x, z/x)$ en la Sección 4.4

- Cálculo de la observación virtual $z$, que corresponde a la pose observada del landmark tridimensional relativa a la cámara $\eta_{LAND\text{-}CAM}$. Se calcula a partir de los píxeles $(u,v)$ medidos y los puntos del cuerpo $\Pi_i$ observados. El punto inicial $\eta_0$ necesario para la minimización iterativa se obtiene usando el algoritmo de los tres puntos sobre tripletas elegidas al azar del conjunto de $M$ píxeles existentes y eligiendo el mejor valor obtenido en esos intentos. Se realizan 10 intentos, además de probar con el valor $\eta = I$ y con el último valor obtenido en la detección anterior.

$$E_P(\eta_{LAND-CAM}) = \sum_{i=1}^{M} \left\| proy(\eta_{LAND-CAM} \cdot \Pi_i) - \begin{pmatrix} u_i \\ v_i \end{pmatrix} \right\|^2 \tag{167}$$

$$\eta_{abcd} = \text{alg3p}\left( \Pi_a, \Pi_b, \Pi_c, \Pi_d, \begin{pmatrix} u_a \\ v_a \end{pmatrix}, \begin{pmatrix} u_b \\ v_b \end{pmatrix}, \begin{pmatrix} u_c \\ v_c \end{pmatrix}, \begin{pmatrix} u_d \\ v_d \end{pmatrix} \right) \quad a \neq b \neq c \neq d \in \{1,...,M\} \tag{168}$$

$$\eta_0 = \arg\min_{\eta_{abcd}} (E_P(\eta_{abcd})) \tag{169}$$

$$z = U(u_1, v_1, ..., \Pi_1, ...) = \eta_{LAND-CAM}^{*} = \arg\min_{\eta_{LAND-CAM}} (E_P(\eta_{LAND-CAM}) \text{ desde } \eta_0) \tag{170}$$

- Cálculo de la covarianza de la observación virtual, que corresponde a la covarianza de la pose observada del landmark tridimensional $R_{3D}$ obtenida mediante el proceso de observación virtual. Para obtenerla, se propaga la covarianza de los pixeles observados $R_{PIX}$ y la covarianza de los puntos del cuerpo $P_\Pi$ a través del procedimiento de observación virtual.

$$R_{3D} = J_{PIX} R_{PIX} J_{PIX}^{T} + J_\Pi P_\Pi J_\Pi^{T} \tag{171}$$

Los jacobianos están definidos del siguiente modo, y se calculan siguiendo el procedimiento indicado en (24):

$$J_{PIX} = \frac{\partial U(u_1, v_1, ..., \Pi_{1,...})}{\partial (u_1, v_1, ..., u_N, v_N)} \tag{172}$$

$$J_\Pi = \frac{\partial U(u_1, v_1, ..., \Pi_{1,...})}{\partial (\Pi_{1X}, \Pi_{1Y}, \Pi_{1Z}, ..., \Pi_{NX}, \Pi_{NY}, \Pi_{NZ})} \tag{173}$$

$R_{PIX}$ diagonal, $P_\Pi$ tridiagonal.

- Cálculo de la función de observación $h$, que corresponde a un puntocuaternión que representa la pose estimada del landmark tridimensional relativa a la cámara $\eta_{LAND-CAM-ESTIMADO}$. Se calcula a partir de la pose del landmark $\eta_{LAND\text{-}MAPA}$ y de la pose del robot en el mapa $\eta_{CAM\text{-}MAPA}$.

$$h(x) = \eta_{LAND-CAM-ESTIMADO}(\eta_{CAM-MAPA}, \eta_{LAND-MAPA}) \quad (174)$$

$$h(x) = \eta_{CAM-MAPA}^{-1} \cdot \eta_{LAND-MAPA} \quad (175)$$

$$H = \frac{dh(x)}{dx} = \begin{bmatrix} \frac{dh(x)}{d\eta_{CAM-MAPA}} & \frac{dh(x)}{d\eta_{LAND-MAPA}} \end{bmatrix} = [H_C \quad H_L] \quad (176)$$

También puede ocurrir que se vea una cantidad de puntos del cuerpo de un landmark tridimensional que sea insuficiente como para calcular una pose confiable. En este caso, los puntos del cuerpo del landmark tridimensional pueden generar observaciones individuales que se pueden usar para corregir el mapa.

La función de observación $h$ de un punto del cuerpo $\Pi_i$ de un landmark rígido con pose $\eta_{LAND\text{-}MAPA}$ observado por una cámara con pose $\eta_{CAM\text{-}MAPA}$ es la siguiente[11].

$$h(x) = \begin{pmatrix} u_i \\ v_i \end{pmatrix} = proy(\eta_{LAND-CAM-ESTIMADO}(\eta_{CAM-MAPA}, \eta_{LAND-MAPA}) \cdot \Pi_i) \quad (177)$$

$$h(x) = proy(\eta_{CAM-MAPA}^{-1} \cdot \eta_{LAND-MAPA} \cdot \Pi_i) \quad (178)$$

$$H = \frac{dh(x)}{dx} = \begin{bmatrix} \frac{dh(x)}{d\eta_{CAM-MAPA}} & \frac{dh(x)}{d\eta_{LAND-MAPA}} \end{bmatrix} = [H_C \quad H_L] \quad (179)$$

Como cada punto del cuerpo tiene una incertidumbre $R_{\Pi i}$ asociada, existe un error intrínseco en el proceso de observación del punto del cuerpo que es independiente de la incertidumbre $R_{pix}$ asociada a la medición del píxel. La incerteza intrínseca del proceso de observación debe sumarse a la covarianza de los pixeles observados.

[11] La función de proyección $proy(\ )$ se define como $proy(x,y,z) = (y/x, z/x)$ en la Sección 4.4

$$h(\eta_{CM}, \eta_{LM}) = proy\left(\eta_{CM}^{-1} \cdot \eta_{LM} \cdot \Pi_i\right) \quad (180)$$

$$\frac{dh}{d\Pi_i} = \frac{dproy\left(\eta_{CM}^{-1} \cdot \eta_{LM} \cdot \Pi_i\right)}{d\Pi_i} \quad (181)$$

$$R_{obs} = R_{pix} + \frac{dh}{d\Pi_i} R_{\Pi_i} \left(\frac{dh}{d\Pi_i}\right)^T \quad (182)$$

Se debe notar que es posible obtener una hipótesis acerca de la pose de la cámara a partir de la observación de un landmark tridimensional cuya precisión depende tanto de la precisión de la observación como de aquella de la pose estimada del landmark.

$$h(x) = \eta_{CAM-MAPA}^{-1} \cdot \eta_{LAND-MAPA} \quad (\mathbf{183})$$

$$\eta_{CAM-MAPA} = \eta_{LAND-MAPA} \cdot h(x)^{-.1} \quad (\mathbf{184})$$

## 5.8 Asociación de datos

En el sistema implementado se debe resolver la asociación entre los descriptores SURF de la imagen actual y los landmarks existentes en el mapa, ya que no se hace tracking visual de los puntos de interés. El usar tracking de puntos tipo esquina, que es la otra alternativa, proporciona un modo simple y muy rápido de resolver la asociación de datos, aunque obliga a tratar de un modo distinto a los landmarks que no están sujetos al tracking en cada momento, ya que deben ser buscados en la imagen en cada frame para agregarlos al sistema de tracking. En este sentido, la asociación de datos usando tracking es incompleta, a menos que se use un sistema complementario para permitir la asociación del resto de los landmarks.

En este sistema, que no usa tracking, en cada frame se calcula la proyección de los landmarks usando la pose estimada de la cámara, lo cual permite predecir los descriptores que deberían verse en la imagen capturada. El método implementado para resolver la asociación de datos funciona encontrando

una transformación de coordenadas entre los descriptores que se encontraron en la imagen actual, y aquellos que fueron predichos usando los landmarks y la pose estimada de la cámara. La transformación de coordenadas se realiza mediante el sistema L&R [50], que es una extensión del sistema de David Lowe que realiza algunas verificaciones adicionales sobre la transformación y un refinamiento de esta. Este sistema tiene la gracia de que puede manejar variaciones más bruscas en el movimiento de la cámara, ya que un desplazamiento o rotación de la cámara incompatible con el modelo de movimiento simplemente afecta la traslación o rotación de la transformación, mientras que en el caso del sistema de tracking éste podría perderse al salir los descriptores de la zona de la imagen en la cual se espera encontrarlos. En el caso de que la cámara se mueva de un modo compatible con el modelo de movimiento inercial, la transformación debería ser la identidad, y la variación de posición entre los descriptores encontrados y los predichos deberían ser producidos sólo por el ruido de la cámara y del sistema SURF.

Un problema que se observa es que hay un compromiso entre la cantidad de landmarks que se ve y el tamaño del mapa. Suponiendo que la cámara sólo rota sin trasladarse, el ángulo de visión completo corresponde a una esfera de área $4\pi r^2$. Si esta esfera se divide por el radio al cuadrado, queda la expresión $4\pi[rad^2]$. Esta medida de área angular se puede transformar a grados para poder tener mayor intución sobre las cantidades numéricas, el valor que se obtiene es:

$$4\pi[rad^2]\times\frac{180}{\pi}[gr/rad]\times\frac{180}{\pi}[gr/rad]=\frac{129600}{\pi}[gr^2]\approx 41253[gr^2] \qquad (185)$$

Si se asume que los descriptores en el mapa se distribuyen de tal modo que se visualizan $V$ landmarks en una ventana de $45[gr]\times 45[gr]$, la relación entre la cantidad de landmarks contenida en la ventana y la cantidad de landmarks total $N$ es:

$$V = N\times\frac{45[gr]\times 45[gr]}{129600[gr^2]} \qquad (186)$$

$$N = V\times\frac{129600}{45\times 45} = 64V \qquad (187)$$

La cantidad de landmarks totales en el mapa es 64 veces la cantidad de landmarks que se ven en una ventana en un tiempo dado. Para que el sistema pueda hacer una asociación de datos aceptable se requiere que existan unos 12 landmarks en el campo visual, ya que los descriptores SURF no son completamente repetibles (un mismo SURF no aparece en todos los cuadros) y, por otro lado, algunos landmarks pueden quedar mal asociados, por lo cual son eliminados antes de ser usados como observaciones. Si se mantiene una cantidad de 12 landmarks visibles con una densidad constante en el mapa, la cantidad total de landmarks necesarios es:

$$N = 64 \times 12 = 768 \tag{188}$$

Esa cantidad de landmarks es claramente inmanejable para un sistema EKF con covarianza completa, sobre todo si cada landmark es representado usando 6 componentes en el estado (debido a la representación usando profundidad inversa). En consecuencia, se propone usar una densidad variable de landmarks: la idea es mantener muchos landmarks en el espacio visible, y pocos en el espacio no visible, sólo los necesarios para permitir que el sistema pueda reconocer las zonas anteriormente exploradas.

Debido a la restricción anterior, se mantendrá una densidad de 12 descriptores dentro de un área angular de $45 \times 45[gr^2]$ para las áreas no visibles y una densidad de 40 landmarks en la misma área para los landmarks visibles. De este modo, el exceso de landmarks visibles puede ayudar en el reconocimiento que se debe realizar frame a frame; en cierto modo cumplen una función similar a los puntos que se usan para tracking, pero son reconocidos frame a frame. La metodología que se usará para lograr este propósito consiste en agregar nuevos landmarks en dos etapas en cada frame:

1. Agregar landmarks manteniendo una distancia $d_{MENOR}$ entre los descriptores. Los descriptores que se agreguen en esta etapa se marcarán como temporales.
2. De los descriptores temporales, se eligirá un subconjunto que mantenga una distancia $d_{MAYOR}$ entre sus elementos en cada frame. Estos descriptores serán considerados permanentes.

En cada frame, los descriptores temporales que salen fuera del área visible son eliminados, mientras que los permanentes se mantienen para permitir el reconocimiento de áreas anteriormente vistas.

### 5.8.1 Test chi-cuadrado

La asociación de datos basada en descriptores SURF y transformación afín tiene una alta probabilidad de rechazo de asociaciones incorrectas; sin embargo, siempre existe la posibilidad de que el sistema reciba asociaciones inadecuadas. Para ayudar a evitar que datos mal asociados saquen al estado del sistema de la zona de convergencia, se usa un test chi-cuadrado, el cual permite descartar observaciones extrañas cuya probabilidad de aparición es excesivamente baja dada la distribución normal actual. De este modo, sacrificando pocos datos correctos pueden descartarse una gran cantidad de datos incorrectos.

El test chi-cuadrado aplicado evalúa una cantidad denominada innovación, la cual corresponde al error de predicción $s_i = z_i - h_i$ que ocurre cuando se compara la observación de un landmark $z_i$ y su valor esperado $h_i$, el cual es obtenido a partir del estado $x$ mediante la función de observación $h(x)$. El error de predicción tiene una covarianza estimada $S_{ii}$, la cual depende de la covarianza del estado $P_{ii}^{-}$, del jacobiano de la función de observación $H_i$ y de la covarianza de la observación $R_i$. Se calcula un estadístico denominado $\chi^2$, el cual representa la relación entre el error de predicción obtenido $s_i$ y su covarianza estimada $S_{ii}$. Si el estadístico $\chi^2$ supera un umbral $\chi^2_{MAX}$, significa que el error de predicción $s_i$ es muy alto y escapa del patrón sperado, lo cual se puede atribuir a que la detección es incorrecta.

Para realizar el test chi-cuadrado se realizan los siguientes pasos:

- Se calcula la covarianza estimada de la innovación para cada landmark observado, siendo la innovación (o residual de la observación) la diferencia entre la observación medida y la predicha por el modelo [99].

$$S_{ii} = H_i P_{ii}^{-} H_i^{T} + R_i,\ i = 1,...,n_o \quad (189)$$

- Se evalua el estadístico chi-cuadrado correspondiente

$$\chi^2 = (z_i - h_i)^T S_{ii}^{-1} (z_i - h_i) \quad (190)$$

- Si el valor del estadístico $\chi^2$ supera un cierto umbral $\chi^2_{MAX}$, la observación se califica como extraña y se elimina.

El umbral del test $\chi^2$ depende de la dimensionalidad de la observación $z_k$ asociada a cada landmark, y el radio de descarte es equivalente a 3.3 sigmas en el caso unidimensional. Esta condición se puede trasladar a un caso de dimensionalidad arbitrario eligiendo un umbral $\chi^2_{MAX}$ correspondiente a un p-valor de 0.99, lo cual asegura que el test descarta a lo más el 0.1% de las observaciones correctas. Para el caso en que la observación es un pixel (bidimensional), el umbral es igual a 13.82. Para el caso en que la observación es una pose (7 dimensiones), el umbral es igual a 24.32.

## 5.9 Actualización del estado del sistema

Para actualizar el estado del sistema se deben efectuar los siguientes cálculos, los cuales corresponden a la actualización típica de un filtro extendido de Kalman.

$$S = H_k P_k^{-} H^T + R_K \quad (191)$$

$$K_k = P_k^{-} H_k^{T} S^{-1} \quad (192)$$

$$x_{k|k} = x_{k|k-1} + K_k \left(z - h\left(x_{k|k-1}\right)\right) \quad (193)$$

$$P_{k|} = \left(I - K_k H_k\right) P_k^{-} \quad (194)$$

Al ser la matriz de covarianza de alta dimensionalidad, los cálculos se pueden dividir en partes para acelerar los cálculos. Para esto, se separan las componentes de las matrices involucradas en componentes observadas (con subíndice o) y no observadas (con subíndice n).

$$x = \begin{pmatrix} x_o \\ x_n \end{pmatrix}, P = \begin{pmatrix} P_{oo} & P_{on} \\ P_{no} & P_{nn} \end{pmatrix}, H = \begin{pmatrix} H_o & 0 \end{pmatrix} \quad (195)$$

$$S = H_o P_{oo}^- H_o^T + R_K \tag{196}$$

$$K = \begin{pmatrix} K_o \\ K_n \end{pmatrix} = \begin{pmatrix} P_{oo}^- H_o^T S^{-1} \\ P_{no}^- H_o^T S^{-1} \end{pmatrix} \tag{197}$$

$$\begin{pmatrix} x_o \\ x_n \end{pmatrix}_{k|k} = \begin{pmatrix} x_o \\ x_n \end{pmatrix}_{k|k-1} + \begin{pmatrix} K_o \\ K_n \end{pmatrix} \left( z - h\left( x_{k|k-1} \right) \right) \tag{198}$$

$$\begin{pmatrix} P_{oo} & P_{on} \\ P_{no} & P_{nn} \end{pmatrix} = \begin{pmatrix} P_{oo}^- & P_{on}^- \\ P_{no}^- & P_{nn}^- \end{pmatrix} - \begin{pmatrix} K_o P_{oo}^- H_o^T S^{-1} \\ K_n P_{no}^- H_o^T S^{-1} \end{pmatrix} \tag{199}$$

Se puede observar que se puede evitar multiplicar por $P_{nn}$, que es la matriz más grande involucrada en el sistema. Sin embargo, se deben separar las componentes observadas de las no observadas, proceso que es rápido pero de orden $\mathrm{O}(\mathrm{n}^2)$. Por otro lado, al restar una covarianza de otra directamente pueden producirse errores numéricos que causen una corrupción de los valores propios de la matriz, los cuales pueden volverse negativos.

Frente a esto hay varias soluciones posibles. La primera es hacer los cálculos del modo indicado anteriormente y realizar una descomposición en vectores propios para poder reparar los valores propios de la matriz, pero este proceso es extremadamente lento. La otra opción es usar un procedimiento denominado *Cholesky downdating* [90], el cual permite restar una covarianza de rango 1 (de la forma $zz^{\mathrm{T}}$) de otra de rango completo.

En las ecuaciones que siguen, $L_P$ representa la descomposición Cholesky triangular inferior de $P_k^-$, $U_S$ representa la descomposición Cholesky triangular superior de $S_k^{-1}$, los términos $v_i$ representan las columnas de la matriz correspondiente y las demás variables tienen un significado similar al correspondiente en las ecuaciones de actualización del sistema EKF.

$$P_k^- = L_P L_P{}^T, S_k{}^{-1} = U_S{}^T U_S \tag{200}$$

$$\begin{aligned} K\left(H_k P_k\right) &= \left(H_k P_k\right)^T S_k{}^{-1} \left(H_k P_k\right) = \left(H_k P_k U_S\right)^T \left(H_k P_k U_S\right) \\ &= \left(v_1 \mid v_2 \mid \ldots \mid v_{n_o}\right)\left(v_1 \mid v_2 \mid \ldots \mid v_{n_o}\right)^T \end{aligned} \tag{201}$$

$$P_k = P_k^- - KH_kP_k^- \Leftrightarrow (L_P L_P{}^T)_k = (L_P L_P{}^T)_k^- - \sum_{i=1}^{n_o} v_i v_i{}^T \quad (202)$$

En este caso, deben realizarse varios downdatings en cada iteración. Cada downdating es del orden de $n^2$ ya que debe modificar la matriz L completa. En consecuencia, si hay varios downdating, sus costos temporales deben sumarse. Se debe notar que crear una descomposición Cholesky o recomponer una matriz toma un tiempo de $n^3$, que es bastante mayor; sin embargo, realizar una descomposición Cholesky es relativamente rápido; al menos es más rápido que multiplicar dos matrices, y mucho más rápido que calcular sus vectores propios.

Los cuaterniones contenidos en las matrices $h$ y $z$ se calcularon de forma independiente, por lo cual pueden tener signos distintos a pesar de representar la misma rotación. Para solucionar este inconveniente, se debe evaluar para cada par correspondiente de cuaterniones si tienen el mismo signo o no. En el caso en que tengan signos opuestos, se debe reparar el signo del cuaternión en $z$ y reparar algunos signos en $R$. El procedimiento utilizado se muestra en la expresión (169), en la cual $q_z$ representa el estado del cuaternión, el subíndice $q$ indica las componentes relacionadas con el cuaternión, y el subíndice $o$ está relacionado con el resto de las componentes.

$$z = \begin{pmatrix} q_z \\ z_o \end{pmatrix};\ R = \begin{pmatrix} R_{qq} & R_{qo} \\ R_{oq} & R_{oo} \end{pmatrix};\ h = \begin{pmatrix} q_h \\ h_o \end{pmatrix} \quad (203)$$

$$\langle q_1, q_2 \rangle = a_1a_2 + b_1b_2 + c_1c_2 + d_1d_2 \quad (204)$$

$$\langle q_z, q_h \rangle < 0 \Rightarrow \begin{cases} q_z = -q_z \\ R_{qo} := -R_{qo} \\ R_{oq} := -R_{oq} \end{cases} \quad (205)$$

## 5.10 Transformación de landmark con profundidad inversa a punto

Inicialmente el estado contiene sólo la pose del robot (puntocuaternión) y puntos con profundidad inversa que representan las posiciones estimadas de los landmarks.

La incertidumbre de los puntos con profundidad inversa se va reduciendo con el tiempo, lo cual reduce la no-linealidad involucrada, ya que cada vez la aproximación de primer orden aplicada sobre la distribución es más precisa. Cuando la distribución de la nueva representación se vuelve aproximadamente Gaussiana, se colapsa el punto con profundidad inversa de 6 dimensiones en un punto de 3 dimensiones. Este paso reduce la dimensionalidad del vector de estado de N+6 a N+3.

La incertidumbre conjunta de los puntos se va reduciendo con el tiempo. Cuando el índice de homogeneidad de un conjunto de puntos (depende de su covarianza conjunta) es menor a un umbral, se colapsa un conjunto de $M$ puntos en un puntocuaternión y un conjunto de puntos del cuerpo. Este paso reduce la dimensionalidad del vector de estado de $N$+3$M$ a $N$+7, donde $N$ representa la cantidad de componentes del estado no involucradas en la reducción de dimensionalidad.

### 5.10.1 Criterio para transformar de landmark punto con profundidad inversa a landmark punto

Dado un landmark con distancia inversa (5.5.2), se usa el siguiente criterio para evaluar la conveniencia del cambio de representación a landmark punto:

$$m = \begin{pmatrix} \cos(\theta)\cos(\phi)/\rho \\ \sin(\theta)\cos(\phi)/\rho \\ \sin(\phi)/\rho \end{pmatrix} \tag{206}$$

$$r = p + m \tag{207}$$

$$h_W = r - \vec{r}_{CAM-MAPA} \tag{208}$$

$$\sigma_d = \frac{\sigma_\rho}{\rho^2} \tag{209}$$

$$d = \|h_W\| \tag{210}$$

$$\cos\alpha = \frac{m^T h_W}{\|h_W\|} \tag{211}$$

$$L_d = 4\frac{\sigma_d}{d}\left|\cos\alpha\right| \text{ (linealidad)} \tag{212}$$

$$L_d < 0.1 \tag{213}$$

En las expresiones anteriores, $r$ representa la posición del landmark punto equivalente, $h_W$ representa la distancia entre la posición del punto equivalente y la cámara, $\sigma_p$ representa la desviación estándar asociada a la profundidad inversa, $\alpha$ representa el ángulo formado por el punto equivalente $r$, la posición de la cámara $r_{CAM\text{-}MAPA}$ y el punto origen $(x,y,z)$ del landmark con distancia inversa, y $L_d$ es un índice de linealidad, el cual es la base del criterio para evaluar la conveniencia de la transformación.

### 5.10.2 Colapso de landmark punto con profundidad inversa a landmark punto

Para colapsar el landmark punto con profundidad inversa a landmark punto se realiza el siguiente proceso (ver [12]).

- Se separa el estado en las variables involucradas $p$ y las no involucradas $o$.

$$\vec{x} = \begin{pmatrix} p \\ o \end{pmatrix},\ p = T(r) = \begin{pmatrix} x \\ y \\ z \end{pmatrix} + R_Z(\theta)R_Y(\phi)\begin{pmatrix} 1/\rho \\ 0 \\ 0 \end{pmatrix} \tag{214}$$

- Se calcula el nuevo estado $r$ y se reemplaza en el vector de estado

$$\vec{x} = \begin{pmatrix} r \\ o \end{pmatrix},\ r = \begin{pmatrix} x & y & z & \rho & \theta & \phi \end{pmatrix}^T \tag{215}$$

- Se transforma la matriz de covarianza.

$$J = \frac{\partial T}{\partial r} \mid r \tag{216}$$

$$P = \begin{pmatrix} P_{rr} & P_{ro} \\ P_{or} & P_{oo} \end{pmatrix} \tag{217}$$

$$\rightarrow P = \begin{pmatrix} JP_{rr}J^T & JP_{ro} \\ P_{or}J^T & P_{oo} \end{pmatrix} \tag{218}$$

## 5.11 Generación de landmarks rígidos tridimensionales

El mérito principal de esta tesis es la creación de una metodología para agrupar landmarks puntuales en landmarks tridimensionales. Se puede apreciar que si el estado original tiene $n_O + 3n_P$ componentes antes de la agrupación, queda con sólo $n_O + 7$ componentes tras esta. Se compararán dos casos opuestos extremos.

- Caso 1: La matriz de covarianza considera sólo el estado del robot y de N landmarks tridimensionales. El tamaño de la matriz de covarianza es de $tam_1(N) = (13 + 7N)^2$
- Caso 2: La matriz de covarianza considera sólo el estado del robot y de $N\, n_p$ landmarks puntuales, donde $n_p$ es la cantidad de puntos que forman un landmark tridimensional. El tamaño de la matriz de covarianza es de $tam_2(N, n_p) = (13 + 3N\, n_p)^2$
- A medida que el número de landmarks aumenta, la diferencia de tamaño converge a la siguiente cantidad

$$\frac{tam_1(N)}{tam_2(N, n_P)} = \frac{(13 + 7N)^2}{(13 + 3N\, n_p)^2} = \frac{49N^2 + O(N)}{9{n_p}^2 N^2 + O(N)} \approx \frac{5,\overline{4}}{{n_p}^2} \tag{219}$$

$$n_p = 10 \Rightarrow \frac{tam_1(N)}{tam_2(N, n_P)} \approx 0.05\overline{4} \tag{220}$$

Esto muestra que si se agrupan los landmarks puntuales de a 10 para formar landmarks tridimensionales, el tamaño de la matriz de covarianza que se obtiene tras agruparlos es alrededor de un 5.5% del original cuando el número de

landmarks es grande. Además, el tiempo que toma la propagación de covarianza $S = HPH^T + R$ es proporcional al tamaño de la matriz de covarianza, lo que muestra que el ahorro computacional puede llegar a ser de un 94.5% cuando el número de landmarks es grande. El tiempo que se necesita para calcular las observaciones virtuales es proporcional al número de landmarks tridimensionales vistos en cada frame, lo cual muestra que el sistema de landmarks tridimensionales se va volviendo más conveniente a medida que el tamaño del mapa crece. Como se mostrará en los resultados experimentales, se necesita una densidad muy baja de landmarks rígidos en el mapa para permitir una buena localización del robot, por lo cual estos landmarks son apropiados para representar mapas de gran tamaño.

El proceso de agrupación puede ser usado para cualquier tipo de landmark puntual, incluyendo los landmarks obtenidos mediante lectura de láser u otros procesos, ya que tanto la transformación de landmarks puntuales a un landmark rígido como el criterio para elegir cuándo realizar dicha transformación usan sólo las medias y covarianzas del sistema.

### 5.11.1 Transformación de representación

Para colapsar los landmark punto a landmark tridimensional comprende dos etapas: la transformación del estado y la transformación de la covarianza.
La transformación del estado se realiza mediante el siguiente proceso.

- Se divide el estado $x$ en los landmarks-punto a transformar $p_{CONJ}$ y el resto de los estados.

$$x = \begin{pmatrix} p_{CONJ} \\ otros \end{pmatrix} \tag{221}$$

$$p_{CONJ} = \begin{pmatrix} p_1 \\ \dots \\ p_M \end{pmatrix}, \quad p_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \tag{222}$$

- Se calcula la media de los puntos

$$m = \frac{1}{M}\sum_{i=1}^{M} p_i \tag{223}$$

- Se calculan los puntos del cuerpo

$$\Pi_i = p_i - m \tag{224}$$

- Se define la siguiente función que encuentra la transformación óptima que transforma coordenadas de un sistema A a otro B:

$$T\left(p_1^{(B)},...,p_N^{(B)},p_1^{(A)},...,p_N^{(A)}\right) = \arg\min_{\eta_{LN}}\left(\sum_{i=1}^{N}\left\|\eta_{LN}\cdot p_i^{(A)} - p_i^{(B)}\right\|\right) \tag{225}$$

- Se obtiene la transformación que lleva las coordenadas $\Pi_i$ a las coordenadas $p_i$. Esa transformación corresponde a una traslación de magnitud $m$.

$$\eta_{LAND-MAPA} = \begin{pmatrix} t_{LAND-MAPA} \\ q_{LAND-MAPA} \end{pmatrix} = T\left(p_1,...,p_N,\Pi_1,...,\Pi_N\right) = \begin{pmatrix} m \\ 1 \end{pmatrix} \tag{226}$$

- Se reemplaza el estado de los landmark punto por el del landmark tridimensional $\eta_{LAND\text{-}MAPA}$.

$$x_{NUEVO} = \begin{pmatrix} \eta_{LAND-MAPA} \\ otros \end{pmatrix} \tag{227}$$

La transformación de la covarianza se puede deducir del siguiente análisis:

- La matriz de covarianza original está formada por cuatro submatrices: $P_{pp}$ corresponde a la covarianza de los landmark punto involucrados, $P_{oo}$ corresponde a la covarianza de los landmarks no involucrados, y las otras dos $P_{po}$, $P_{op}$ son las covarianzas cruzadas restantes.

$$P = \begin{pmatrix} P_{pp} & P_{po} \\ P_{op} & P_{oo} \end{pmatrix} \tag{228}$$

$$P_{pp} = \begin{pmatrix} P_{3x3}^{(1,1)} & P_{3x3}^{(1,2)} & \cdots & P_{3x3}^{(1,N)} \\ P_{3x3}^{(2,1)} & P_{3x3}^{(2,2)} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ P_{3x3}^{(N,1)} & \cdots & \cdots & P_{3x3}^{(N,N)} \end{pmatrix} \tag{229}$$

- La matriz de covarianza final va a tener la siguiente forma:

$$P = \begin{pmatrix} P_{77} & P_{7o} \\ P_{o7} & P_{oo} \end{pmatrix} \tag{230}$$

- Ambas expresiones quedan relacionadas del siguiente modo:

$$P_{77} = J_p P_{pp} J_p^T + J_\Pi \begin{pmatrix} P_\Pi^{(1)} & & \\ & \dots & \\ & & P_\Pi^{(N)} \end{pmatrix} J_\Pi^T \tag{231}$$

$$J_p = \frac{\partial T(p_1,...,p_N,\Pi_1,...,\Pi_N)}{\partial(p_1,...,p_N)}; J_\Pi = \frac{\partial T(p_1,...,p_N,\Pi_1,...,\Pi_N)}{\partial(p_1,...,p_N)} \tag{232}$$

- Considerando que la posición de los puntos del cuerpo se eligieron de tal modo que la pose inicial del landmark tridimensional corresponde a una traslación pura, ocurre lo siguiente:

$$T(p_1,...,p_N,\Pi_1,...,\Pi_N) = \begin{pmatrix} m \\ 1 \end{pmatrix} \Rightarrow J_p = -J_\Pi \tag{233}$$

$$P_{77} = J_p \left( P_{pp} - \begin{pmatrix} P_\Pi^{(1)} & & \\ & \dots & \\ & & P_\Pi^{(N)} \end{pmatrix} \right) J_p^T \tag{234}$$

$$P_{7o} = J_p P_{po}; P_{o7} = P_{op} J_p^T \tag{235}$$

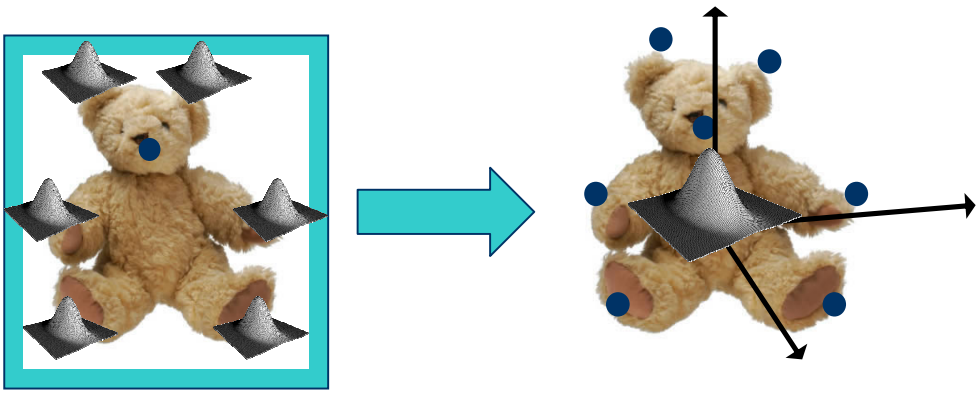


**Figura 9: La covarianza de los puntos originales ($P_{pp}$) se debe descomponer en la covarianza de la pose ($P_{77}$) más la covarianza de los puntos del cuerpo ($P_\Pi$).**

- Cualquier combinación de las matrices $P_{77}$, $P_{\Pi}^{(1)}$,…, $P_{\Pi}^{(N)}$ que sean semidefinidas positivas y que cumplan con la condición (192) conforman una descomposición correcta, ya que la covarianza $P_{pp}$ se reparte entre la covarianza de la nueva pose $P_{77}$ y las covarianzas de los puntos del cuerpo $P_{\Pi}^{(1)}$, …, $P_{\Pi}^{(N)}$. Dado que existe esa flexibilidad, se definirán dos opciones opuestas para realizar la división de la covarianza.

1. Maximizar la covarianza de la pose: En este caso, se traspasa toda incerteza $P_{pp}$ de los landmarks punto a la incerteza $P_{77}$ de la pose. Se asigna una covarianza residual a los puntos del cuerpo, la cual se obtiene restando la covarianza original $P_{pp}$ y la reconstruída $P_{REC}$. Esta última se obtiene al intentar reconstruir la covarianza original a partir de $P_{77}$.

$$P_{77} = J_p P_{pp} J_p^T \tag{236}$$

$$P_{7o} = J_p P_{po}; P_{o7} = P_{op} J_p^T \tag{237}$$

$$U(\Pi_1,...,\Pi_N,\eta_{LAND-MAPA}) = \begin{pmatrix} \eta_{LAND-MAPA}\cdot\Pi_1 \\ ... \\ \eta_{LAND-MAPA}\cdot\Pi_N \end{pmatrix} \approx \begin{pmatrix} p_1 \\ ... \\ p_N \end{pmatrix} \tag{238}$$

$$G = \frac{\partial U(\Pi_1,...,\Pi_N,\eta_{LAND-MAPA})}{\partial(\Pi_1,...,\Pi_N)} \tag{239}$$

$$P_{REC} = G P_{77} G^T \tag{240}$$

$$P_{DIFF} = P_{pp} - P_{REC} = \begin{pmatrix} D_{3x3}^{(1,1)} & ... & D_{3x3}^{(1,N)} \\ ... & ... & ... \\ D_{3x3}^{(N,1)} & ... & D_{3x3}^{(N,N)} \end{pmatrix} \tag{241}$$

$$P_{\Pi}^{(i)} = \mathrm{svdRep}\left(D_{3x3}^{(i,i)}\right) \tag{242}$$

2. Maximizar la covarianza de los puntos del cuerpo: En este caso, se intenta minimizar el traspaso de incerteza desde los landmarks puntos a la pose. Para minimizar este traspaso se intenta restar la mayor cantidad de covarianza posible de la matriz de covarianza original antes de calcular la

covarianza de la pose. La mayor cantidad de covarianza que se puede restar a una matriz de covarianza es aquella que deja su menor valor propio igual a cero, ya que si queda negativo, la matriz resultante no es una matriz de covarianza. Para lograr esto, se minimiza el cuadrado del menor valor propio de la matriz de covarianza resultante.

$$D = diag(P_{pp}) \tag{243}$$

$$\min_{\alpha_1,\alpha_2} \lambda_{MIN}{}^2 \left( P_{pp} - \alpha_1 \begin{pmatrix} P_{3x3}{}^{(1,1)} & & \\ & \dots & \\ & & P_{3x3}{}^{(N,N)} \end{pmatrix} - \alpha_2 D \right) \tag{244}$$

$$\begin{pmatrix} P_{\Pi}{}^{(1)} & & \\ & \dots & \\ & & P_{\Pi}{}^{(N)} \end{pmatrix} = \lambda_1 \begin{pmatrix} P_{3x3}{}^{(1,1)} & & \\ & \dots & \\ & & P_{3x3}{}^{(N,N)} \end{pmatrix} + \lambda_2 D \tag{245}$$

$$P_{77} = J_p \left( P_{pp} - \begin{pmatrix} P_{\Pi}^{(1)} & & \\ & \dots & \\ & & P_{\Pi}^{(N)} \end{pmatrix} \right) J_p^T \tag{246}$$

$$P_{7o} = J_p P_{po}; P_{o7} = P_{op} J_p^T \tag{247}$$

### 5.11.2 Criterio de transformación

El criterio de transformación se basa en calcular una cantidad denominada índice de variabilidad. Es un criterio rápido para buscar un conjunto de puntos que tengan covarianzas pequeñas y similares entre sí, considerando tanto las covarianzas propias como las cruzadas entre los puntos. El criterio se muestra de forma gráfica en la Figura 10. Las ecuaciones que permiten evaluar el criterio de agrupación son las siguientes:

$$x = \begin{pmatrix} x_p \\ x_o \end{pmatrix}, x_p = \begin{pmatrix} p_1 \\ \dots \\ p_n \end{pmatrix} \tag{248}$$

$$P = \begin{pmatrix} P_{pp} & P_{po} \\ P_{op} & P_{oo} \end{pmatrix}, P_{pp} = \begin{pmatrix} P_{3x3}^{(1,1)} & \dots & P_{3x3}^{(1,n)} \\ \dots & \dots & \dots \\ P_{3x3}^{(n,1)} & \dots & P_{3x3}^{(n,n)} \end{pmatrix} = \begin{pmatrix} P_{1,1} & \dots & P_{n,1} \\ \dots & \dots & \dots \\ P_{1,n} & \dots & P_{n,n} \end{pmatrix} \quad (249)$$

$$orden : a > b \Rightarrow P_{a,a\,XX} + P_{a,a\,YY} + P_{a,a\,ZZ} > P_{b,b\,XX} + P_{b,b\,YY} + P_{b,b\,ZZ} \quad (250)$$

$$C_X(u_0, v_0, u_1, v_1) = \frac{\sum_{u=u_0}^{u_1}\sum_{v=v_0}^{v_1} P_{u,v\,XX}{}^2}{(u_1 - u_0 + 1)(v_1 - v_0 + 1)} - \left( \frac{\sum_{u=u_0}^{u_1}\sum_{v=v_0}^{v_1} P_{u,v\,XX}}{(u_1 - u_0 + 1)(v_1 - v_0 + 1)} \right)^2 \quad (251)$$

$$C_Y(u_0, v_0, u_1, v_1) = \frac{\sum_{u=u_0}^{u_1}\sum_{v=v_0}^{v_1} P_{u,v\,YY}{}^2}{(u_1 - u_0 + 1)(v_1 - v_0 + 1)} - \left( \frac{\sum_{u=u_0}^{u_1}\sum_{v=v_0}^{v_1} P_{u,v\,YY}}{(u_1 - u_0 + 1)(v_1 - v_0 + 1)} \right)^2 \quad (252)$$

$$C_Z(u_0, v_0, u_1, v_1) = \frac{\sum_{u=u_0}^{u_1}\sum_{v=v_0}^{v_1} P_{u,v\,ZZ}{}^2}{(u_1 - u_0 + 1)(v_1 - v_0 + 1)} - \left( \frac{\sum_{u=u_0}^{u_1}\sum_{v=v_0}^{v_1} P_{u,v\,ZZ}}{(u_1 - u_0 + 1)(v_1 - v_0 + 1)} \right)^2 \quad (253)$$

$P = 10$ (numero de puntos a agrupar, arbitrario) (254)

$$(u,v)^* = \arg\min_{u,v} C_X(u, v, u+P, v+P) + C_Y(u, v, u+P, v+P) + C_Z(u, v, u+P, v+P) \quad (255)$$

$$\text{ivariab} = C_X(u, v, u+P, v+P) + C_Y(u, v, u+P, v+P) + C_Z(u, v, u+P, v+P) \quad (256)$$

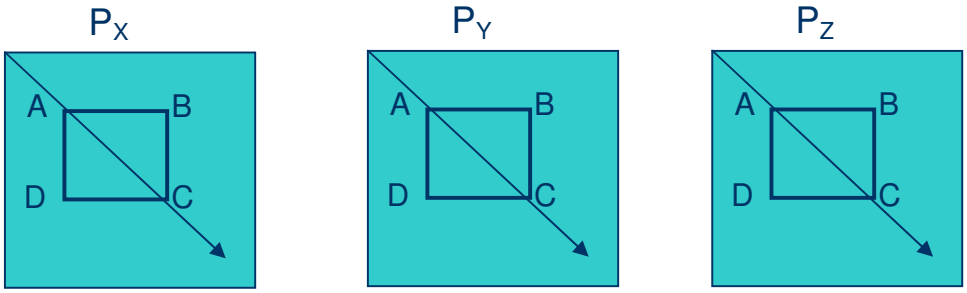



**Figura 10: Una ventana dentro de la matriz de covarianza de la componentes X, Y y Z de los landmarks es usada para evaluar conjuntos de puntos con covarianzas cruzadas similares. Los cálculos se realizan de forma eficiente usando imágenes integrales.**

Se puede notar que el índice de variabilidad de un conjunto de landmarks-punto es igual a la suma para las tres componentes $X, Y, Z$ de la variación de las sub-matrices de covarianza $P_{XX}, P_{YY}, P_{ZZ}$ de los puntos. De este modo, cuando un conjunto de puntos tiene variaciones pequeñas en las covarianzas cruzadas por cada componente $X, Y, Z$, el índice de variabilidad es bajo (es decir, las distintas covarianzas cruzadas de los puntos son indistinguibles, lo que muestra que su incertidumbre común es representable con pocos números).

Además se usan las imágenes integrales de varianza $C_X, C_Y, C_Z$, lo que hace que la evaluación de la variación de covarianza cruzada en una ventana sea rápida, lo que permite usarla para la generación de conjuntos de puntos candidatos, aunque en las pruebas realizadas el conjunto candidato que resulta casi siempre contiene los elementos de menor covarianza individual. En las pruebas hechas se ha apreciado que el índice de homogeneidad de un conjunto de puntos es decreciente con el tiempo, y disminuye a medida que se van mostrando distintas vistas del conjunto de landmarks-punto.

El orden mostrado en (250) no siempre asegura que los puntos cuya agrupación sea deseable queden cercanos entre sí en la matriz de covarianza. Para solucionar en parte este problema, se detectan los puntos que tienen covarianzas cruzadas incompatibles con los puntos cercanos. Cuando un punto tiene covarianzas cruzadas negativas con los puntos cercanos, las componentes de la covarianza asociadas a ese punto son permutadas al final de la matriz.

$$\sum_{i=1}^{n} P_{a,i\,XX} < 0 \Rightarrow \{P_{a,*}, P_{*,a}\} \text{ empujadas al final} \tag{257}$$

$$\sum_{i=1}^{n} P_{a,i\,YY} < 0 \Rightarrow \{P_{a,*}, P_{*,a}\} \text{ empujadas al final} \tag{258}$$

$$\sum_{i=1}^{n} P_{a,i\,ZZ} < 0 \Rightarrow \{P_{a,*}, P_{*,a}\} \text{ empujadas al final} \tag{259}$$

## 5.12 Detalles de implementación del sistema

El sistema fue implementado en Ansi C++ 98, y no necesita ninguna biblioteca externa para funcionar ya que todos los cálculos que se necesitan

fueron programados por el autor de la Tesis. Esta opción fue elegida para permitir su funcionamiento en cualquier plataforma, aunque forzó la escritura de una gran cantidad de código ya existente en otras bibliotecas. Sin embargo, el sistema puede usar ciertas bibliotecas si están disponibles, lo cual se debe indicar activando definiciones al momento de compilar el código.

- `#ifdef __COMPAT_EIGEN__ // usar eigen (matrices) para acelerar los cálculos`
- `#ifdef __COMPAT_ATLIMAGE__ // Usar windows (ATL) para grabar y leer imágenes`
- `#ifdef __COMPAT_SVS__ // Lectura de cámara estéreo Videre, actualmente no se está usando`
- `#ifdef __COMPAT_UBLAS__ // Usar ublas (matrices) para acelerar los cálculos`
- `#ifdef __COMPAT_FLTK__ // Usar FLTK para graficar`
- `#ifdef __COMPAT_IPLIMAGE__ // Usar OpenCV para capturar.`

Los distintos tipos de landmarks pueden ser manejados de una forma regular usando herencia: existe una clase abstracta básica `L_Landmark` de la cual se derivan las clases `L_LandmarkPunto`, `L_LandmarkPuntoInverso`, `L_LandmarkSolido` y `L_LandmarkLaser` (parcialmente implementado en este momento). Cada una de estas clases derivadas deben implementar un conjunto de funciones que son necesarias para definir su comportamiento; básicamente esas funciones permiten el calculo de las matrices $h$, $H$, $z$ y $R$ y del test chi-cuadrado a partir de los datos sensoriales disponibles. De este modo, el sistema SLAM implementado puede trabajar con distintos tipos de landmarks de una forma transparente. Debido a que la cantidad de landmarks que se ven en cada frame cambia constantemente, la secuencia de landmarks que se usa en cada iteración cambia, es decir, las matrices $h$, $H$, $z$ y $R$ tienen una estructura variable que es manejada de forma genérica por el sistema.

La matriz $H$ es rala, razón por la cual se usan funciones especializadas para calcular la multiplicación de $H$ o $H^{\mathrm{T}}$ con otra matriz que usan sub-multiplicaciones

por bloques. De este modo, los cálculos se pueden realizar de un modo eficiente manteniendo la notación bastante clara.

Para poder controlar la forma en que se realiza la asociación, se creó una clase `L_creaObservacionAsociaciones_params` que contiene todos los parámetros necesarios para realizar la asociación de datos. Actualmente contiene 42 parámetros, los cuales controlan al sistema L&R y a los tests RANSAC que se ejecutan para limpiar los datos, además de los usados para eliminar descriptores SURF que se encuentran sobre líneas. Obviamente la selección de los parámetros de la asociación es un tema que puede mejorarse debido a la gran cantidad de parámetros que es necesario especificar.

Para poder realizar las simulaciones, se creó una clase llamada `L_SLAMEKF_prueba_restricciones_params`, la cual contiene varios parámetros de ejecución de las pruebas. Cada prueba está compuesta de subpruebas, cada una de las cuales tiene una cantidad de landmarks máximos de cada tipo, un tamaño del vector de estado máximo, una distancia mínima entre los landmarks permanentes creados en la pantalla y una distancia mínima para los landmarks temporales creados en la pantalla. Especificando varias subpruebas distintas, se pueden ejecutar simulaciones con condiciones cambiantes, como las que se detallarán en la parte experimental.

El código depende fuertemente en una función de optimización numérica llamada `L_LevenbergMarquardt::minimiza_vect()` [60]. Al calcular jacobianos numéricamente se suele obtener un resultado ligeramente distinto al usar $+\delta$ o $-\delta$ en la variación del punto evaluado, lo cual puede causar que el sistema converja a un punto cercano al valor óptimo, pero no exactamente a éste. Una forma de evitar este problema es restar los valores obtenidos para $+\delta$ y $-\delta$, pero esta función duplica la cantidad de llamadas a la función de error, por lo cual se decidió llamar alternativamente a $+\delta$ y $-\delta$ cada una de las veces que se calcula una variación. Usando esta metodología, el sistema converge al valor correcto.

El código contiene funciones de Linpack y Lapack que fueron traducidas desde Fortran a C++ usando un software llamado Fable [30]. Esto fue necesario para poder implementar la actualización del EKF usando Cholesky downdating, lo

cual es necesario cuando la fórmula normal para la actualización falla debido a errores numéricos.

Lo último que se va a detallar es que las transformaciones de representación (colapso de landmarks, normalización de los cuaterniones) se realizan mediante una función llamada `transformarCovarianza()`. Esta función recibe una covarianza inicial, un jacobiano, un arreglo de índices de la covarianza inicial y un arreglo de índices de la covarianza final, y construye una covarianza final en el espacio que fue indicado. Esta función se implementó de un modo que es rápido pero no el más óptimo, por lo cual se podría mejorar su desempeño.

### 5.12.1 Medias y covarianzas iniciales

El sistema SLAM requiere condiciones iniciales adecuadas que permitan que la estimación de estados converja a una representación adecuada del mapa. Una inicialización inadecuada del mapa produce errores imposibles de neutralizar posteriormente. Esto es particularmente importante en el SLAM monocular porque, al ser la escala relativa entre el mapa y el mundo real no es observable, puede converger a un número infinito de soluciones posibles, por lo cual la inicialización tiene como objetivo evitar convergencias a escalas muy grandes, negativas o cero. La inicialización implementada es la siguiente.

$$\eta_{CAM-MAPA}(t=0) \sim N\left(0, diag\left(10^{-60} \quad 10^{-60} \quad 10^{-60} \quad 10^{-60} \quad 10^{-60} \quad 10^{-60} \quad 10^{-60}\right)\right) \tag{260}$$

$$v_{CAM-MAPA}(t=0) \sim N\left(\begin{pmatrix}0\\3\\0\end{pmatrix}, \begin{pmatrix}1.0 & & \\ & 1.0 & \\ & & 1.0\end{pmatrix}\right) \tag{261}$$

$$\omega_{CAM-MAPA}(t=0) \sim N\left(0, \begin{pmatrix}0.1^2 & & \\ & 0.1^2 & \\ & & 0.1^2\end{pmatrix}\right) \tag{262}$$

Al inicializarse el sistema, la cámara debe dejarse en una posición fija un par de segundos, y luego debe ser movida lentamente. Una vez que alcance una velocidad estable, puede moverse a voluntad de forma arbitraria, manteniendo la aceleración aplicada dentro de límites que permitan una asociación de datos

correcta. El mover la cámara de otra forma al inicio puede causar pérdida de la convergencia del sistema.

Los landmarks punto con profundidad inversa deben ser inicializados con una cierta media y covarianza que permita su convergencia a la posición real. Se inicializan del siguiente modo.

$$\begin{pmatrix} x \\ y \\ z \\ \rho \\ \theta \\ \phi \end{pmatrix} \sim N\left( \begin{pmatrix} p_{camX} \\ p_{camY} \\ p_{camZ} \\ 0.01 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.01^2 & & & & & \\ & 0.01^2 & & & 0 & \\ & & 0.01^2 & & & \\ & & & (0.01*5)^2 & & \\ & 0 & & & \sin^2(0.01) & \\ & & & & & \sin^2(0.01) \end{pmatrix} \right) \quad (263)$$

Se debe apreciar que la desviación estándar de la profundidad inversa es igual a 5 veces dicha profundidad. Al ser mayor la desviación estándar que el valor inicial de la variable, se permite que la variable pueda cruzar el valor cero. Esto permite que la distancia al punto pueda ser infinita (cuando la variable toma el valor cero), aunque también puede causar que los puntos queden detrás de la cámara (si la variable cruza el cero).

Los landmarks punto y landmarks rígidos no deben ser inicializados, ya que se forman a partir de otros landmarks usando propagación de covarianzas mediante jacobianos.

### 5.12.2 Covarianza de los ruidos

La covarianza de los ruidos se debe elegir de tal modo de que representen adecuadamente la incertidumbre inherente al movimiento de la cámara y a la limitación de los sensores.

Las covarianzas asociadas al movimiento de la cámara son $n_V$, la cual representa la incertidumbre en la velocidad lineal, y $n_W$, la cual representa la incertidumbre en la velocidad angular:

$$n_V \sim N(0, diag(4 \quad 4 \quad 4)) \tag{264}$$

$$n_W \sim N(0, diag(16 \quad 16 \quad 16)) \tag{265}$$

Las covarianzas asociadas a los sensores corresponden a un error equivalente a una desviación estándar de 6 pixeles (cov=36).

### 5.12.3 Escala negativa

Un mapa con escala negativa es válido desde un punto de vista matemático, ya que las ecuaciones de proyección que son válidas para puntos frente a la cámara pueden aplicarse también a los puntos que están detrás; luego, al reflejar todos los puntos respecto a la cámara, la evolución del sistema dinámico es coherente. Esto puede generar en teoría que los landmarks queden ubicados detrás de la cámara en el sistema SLAM. Para evitar este problema, se deben elegir valores apropiados para las medias y covarianzas de los estados de los landmarks al inicializarlos. Los valores indicados en la sección anterior permiten la convergencia del sistema a una escala positiva, aunque la determinación de esos valores adecuados de realizó en parte mediante prueba y error, y en parte mediante consideraciones matemáticas.

# 6 Experimentos

Para validar el sistema de SLAM propuesto se realizaron dos tipos de pruebas: simulaciones y pruebas con videos reales. Las simulaciones permiten obtener datos cuantitativos acerca del funcionamiento del sistema que permiten generar estadísticas de su funcionamiento, lo cual permite comparar el desempeño de los landmarks cuerpo rígido frente a los landmarks puntuales. Las pruebas reales permiten realizar análisis cualitativos solamente, ya que la ruta exacta que siguió la cámara real en el espacio es desconocida.

## 6.1 Simulaciones

El sistema fue evaluado realizando simulaciones del movimiento de la cámara y de la captura de las imágenes correspondientes usando cuatro tipos de rutas: rutas con forma de U, rutas con forma de S, rutas con pérdida continua y rutas al azar. Cada ruta está formada por un conjunto de posiciones y orientaciones de la cámara en distintos instantes y se especifica mediante la trayectoria (posiciones) de la cámara y un punto mirado por ella. Cada una de las rutas usadas consiste en una trayectoria que está definida por una función analítica y un punto fijo observado constantemente por la cámara, de modo de poder replicar los experimentos. La elección de trayectorias simples tiene como objetivo el facilitar el análisis de características relevantes del sistema. Para poder obtener los puntos intermedios de la ruta se realiza un muestreo cada 1[s] para obtener puntos de control y luego se usa una spline cúbica sobre el espacio de los puntocuaterniones para interpolar los puntos de control. De este modo, se asegura que las aceleraciones involucradas en el sistema no diverjan aunque la trayectoria presente discontinuidades ya que la spline cúbica tiene tercera derivada nula. A continuación se especifican las cuatro trayectorias utilizadas (ver Figura 11 para una representación gráfica):

- TU: Trayectoria con forma de U

$$x = -60\sin\left(\frac{\pi}{2}\sin\left(2\pi\frac{t}{8}\right)\right) \quad (266)$$

$$y = 90\cos\left(\frac{\pi}{2}\sin\left(2\pi\frac{t}{8}\right)\right) \quad (267)$$

$$z = 0 \quad (268)$$

Punto mirado: (0, 0, 0)

- TS: Trayectoria con forma de S

$$x = -40\frac{1+t}{54}\cos\left(\cos\left(\frac{1}{3t}\right)\right) \quad (269)$$

$$y = 40\frac{1+t}{54}\sin\left(\cos\left(\frac{1}{3t}\right)\right) \quad (270)$$

$$z = 0 \quad (271)$$

Punto observado: (30, 0, 0)

- TP: Trayectoria con pérdida continua:
  - Ruta con forma de S (80 segundos) seguida por cuatro posiciones (-90,-40), (-90,40), (-30,40), (-30,-40), las cuales se suceden en forma periódica. La cámara se mantiene en cada una de estas posiciones un segundo y luego se teleporta al siguiente punto de la secuencia de forma instantánea. La aceleración del sistema se mantiene acotada usando una spline.
  - Punto observado: (30, 0, 0)
- TA: Trayectoria al azar
  - Ruta con forma de S (80 segundos) seguida por una secuencia al azar cuya aceleración se mantiene acotada usando una spline. La secuencia al azar está especificada del siguiente modo:
    - $x_t = x_{t+1} + n_X, y_t = y_{t+1} + n_Y, z_t = z_{t+1} + n_Z$ (272)
    - $n_X, n_Y, n_Z \rightarrow N(0,10^2)$ (273)
    - Punto observado: (30, 0, 0)

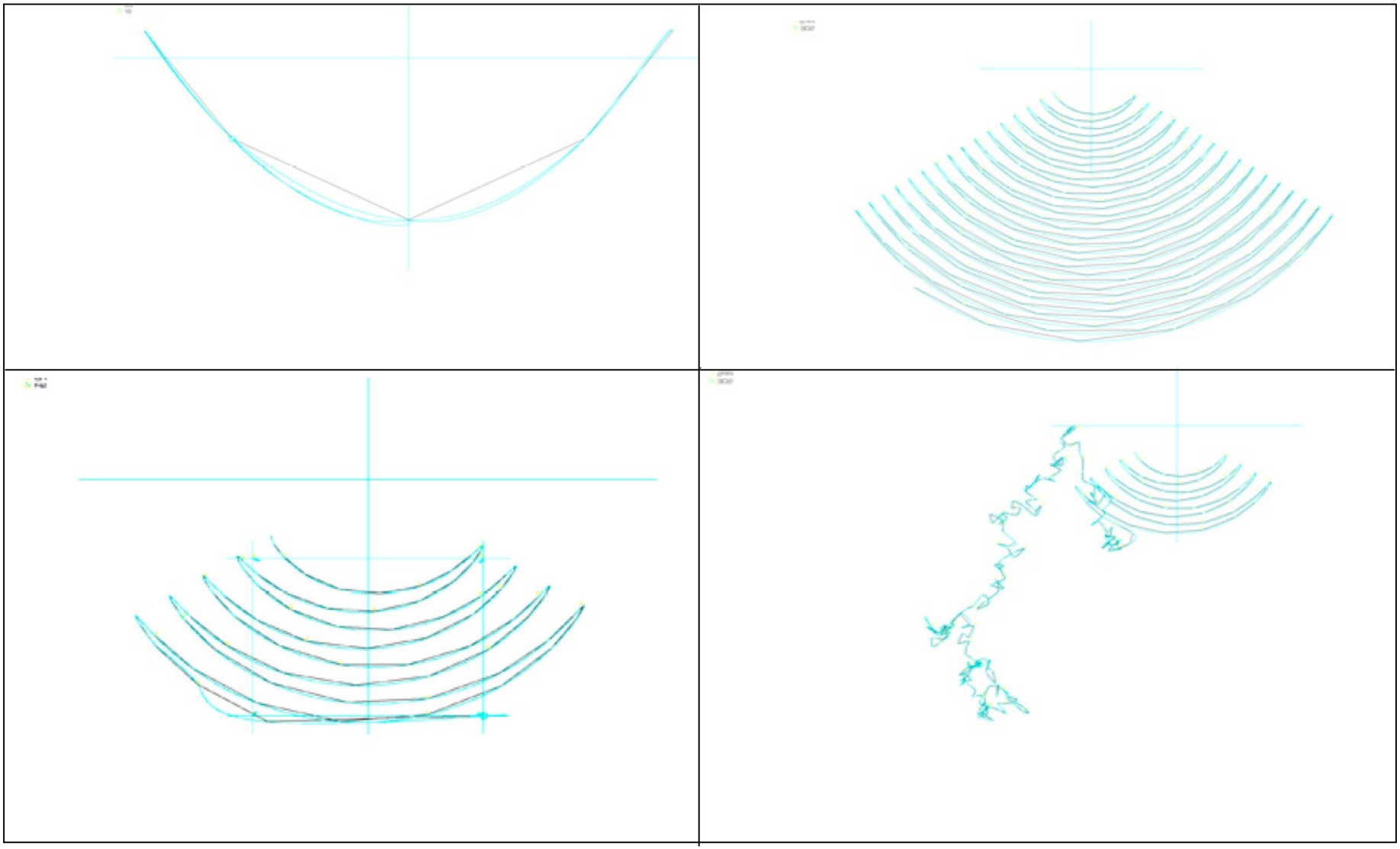


**Figura 11: Ejemplos de las cuatro rutas posibles**

Se realizaron varias pruebas, cada prueba se repitió 30 veces para poder obtener datos estadísticamente válidos. Las pruebas realizadas tienen una duración de 2600 frames, y una restricción de máximo 60 landmarks. En algunas pruebas los landmarks rígidos se crean usando el criterio de máxima covarianza de la pose y en otros usando el criterio de máxima covarianza de los puntos del cuerpo (ver Sección 5.11.1 para detalles):

- Prueba 1: Sólo se usan landmarks puntuales.
- Prueba 2: Se usan todos los tipos de landmarks, los landmarks rígidos se crean usando el criterio de máxima covarianza de la pose
- Prueba 3: Se usan todos los tipos de landmarks, los landmarks rígidos se crean usando el criterio de máxima covarianza de los puntos del cuerpo.
- Prueba 4: Sólo se usan landmarks puntuales, el máximo número de landmarks se reduce a 4 puntuales en el frame 1800.
- Prueba 5: Se usan todos los tipos de landmarks, los landmarks rígidos se crean usando el criterio de máxima covarianza de la pose, el máximo número de landmarks se reduce a 4 rígidos en el frame 1800.
- Prueba 6: Se usan todos los tipos de landmarks, los landmarks rígidos se crean usando el criterio de máxima covarianza de los puntos del cuerpo, el máximo número de landmarks se reduce a 4 rígidos en el frame 1800.

En todos los casos indicados, los landmarks son creados como landmarks con profundidad inversa, los cuales son transformados a landmarks punto y luego a landmarks rígidos cuando se cumplen los criterios necesarios.

Debido a que la reconstrucción de un mapa usando una sola cámara puede crear un mapa válido con una escala, posición y orientación distintas a la original, se aplica como postprocesamiento una transformación óptima que lleva la ruta obtenida mediante el SLAM a la ruta ground-truth. La distancia euclidiana promedio entre ambas rutas es usada como medida de error.

En la Figura 12 se muestran algunos resultados que muestran la reconstrucción de los diferentes tipos de trayectoria.

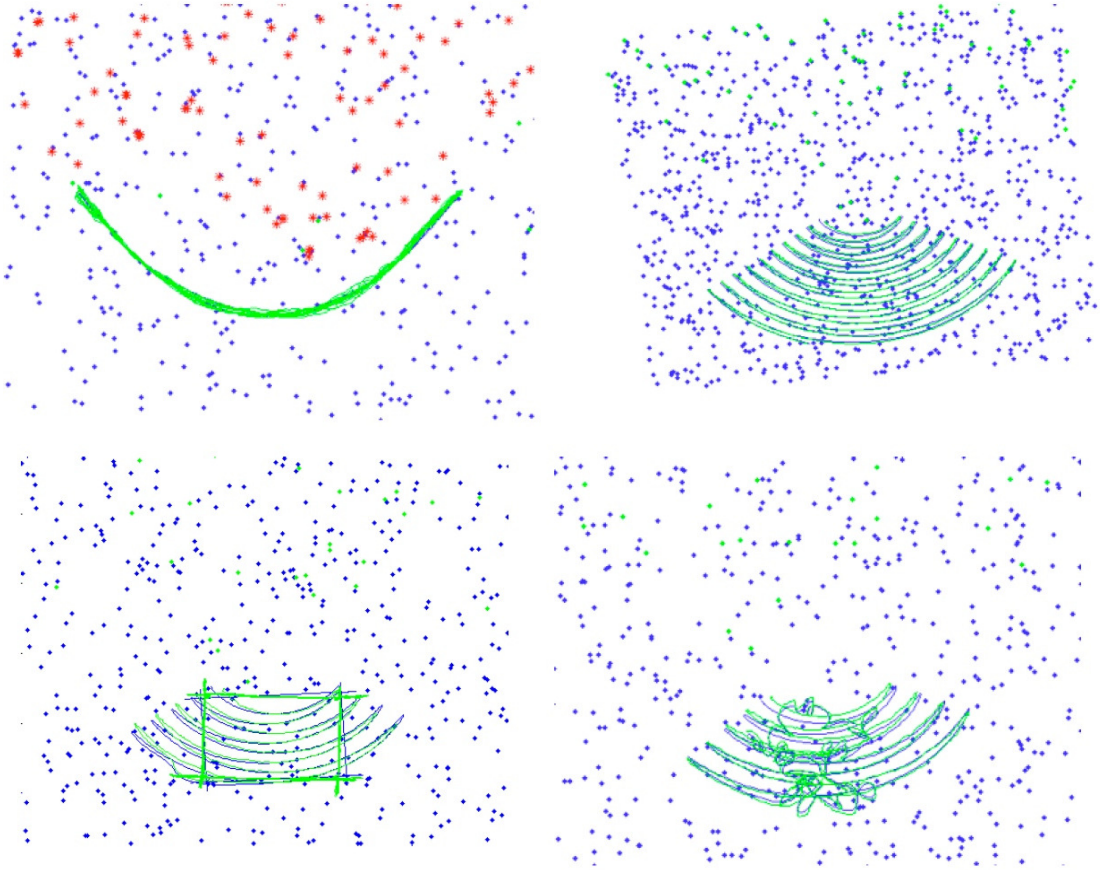


**Figura 12: Trayectorias reales (azul) y reconstruidas (verde). Los puntos azules representan los descriptores creados distribuidos en el espacio que podrían usarse como landmarks, los puntos verdes representan landmarks puntuales y los puntos rojos representan puntos pertenecientes a landmarks cuerpo rígido.**

A continuación se muestran los resultados experimentales mediante una tabla, y en las Figuras 13, 14, 15 y 16 usando histogramas de error. Los resultados con errores mayores a 60 se consideran fallos, ya que dicho error es mayor que las dimensiones del mapa. Los landmarks de tipo *rigido 1* fueron creados usando el criterio de máxima covarianza de la pose, y los landmarks de tipo *rígido 2* fueron creados usando el criterio de máxima covarianza de los puntos del cuerpo.

| Ruta | Tipo de landmarks | Max land. | Error (media) | Error (desv) | Fallos |
|---|---|---|---|---|---|
| TU | Puntuales | 60 | 1.79 | 0.69 | 0% |
| TU | Puntuales + rigido 1 | 60 | 8.02 | 5.80 | 0% |
| TU | Puntuales + rigido 2 | 60 | 3.67 | 0.84 | 0% |
| TU | Puntuales | 4 | - | - | 100% |
| TU | Rígido 1 | 4 | 41.03 | 16.18 | 76.7% |
| TU | Rígido 2 | 4 | 4.19 | 0.93 | 0% |
| TS | Puntales | 60 | 6.81 | 3.53 | 0% |
| TS | Puntuales + rígido 1 | 60 | 26.12 | 20.78 | 86.67% |
| TS | Puntuales + rígido 2 | 60 | 10.95 | 6.64 | 0% |
| TS | Puntuales | 4 | 18.34 | 5.67 | 90% |
| TS | Rígido 1 | 4 | - | - | 100% |
| TS | Rígido 2 | 4 | 13.22 | 6.52 | 0% |

| Ruta | Tipo de landmarks | Max land. | Error (media) | Error (desv) | Fallos |
|---|---|---|---|---|---|
| TC | Puntuales | 60 | 11.69 | 3.18 | 0% |
| TC | Puntuales + rígido 1 | 60 | 30.07 | 14.49 | 70% |
| TC | Puntuales + rígido 2 | 60 | 25.34 | 10.9 | 6.67% |
| TC | Puntuales | 4 | - | - | 100% |
| TC | Rígido 1 | 4 | - | - | 100% |
| TC | Rígido 2 | 4 | 25.75 | 6.25 | 0% |
| TA | Puntuales | 60 | 2.54 | 1.24 | 0% |
| TA | Puntuales + rígido 1 | 60 | 20.89 | 13.66 | 0% |
| TA | Puntuales + rígido 2 | 60 | 3.80 | 1.10 | 0% |
| TA | Puntuales | 4 | 32.97 | 7.10 | 0% |
| TA | Rígido 1 | 4 | 29.75 | 7.87 | 0% |
| TA | Rígido 2 | 4 | 3.57 | 0.86 | 0% |

**Tabla 1: Estadísticas de error de las simulaciones. Se muestra, para cada ruta y tipo de landmarks, el número de landmarks permitidos, el error promedio entre la trayectoria correcta y la estimada, la desviación estándar del error promedio entre la trayectoria correcta y la estimada y el porcentaje de fallos existentes en las simulaciones.**

De los resultados obtenidos en la Tabla 1 se puede concluir que los landmarks rígidos producen un mayor error que los landmarks puntuales (aproximadamente un 50% extra de error). Sin embargo, los landmarks rígidos tienen la fortaleza de poder mantener localizado al robot usando muy pocos landmarks (solamente 4), lo cual resulta imposible de lograr usando los landmarks puntuales. Por otro lado, se puede apreciar que el método #1 de creación de landmarks rígidos (máxima covarianza de la pose) es inferior en todas las pruebas respecto al método 2 (máxima covarianza de los puntos del cuerpo), por lo cual el método 2 es el que debe ser usado para implementar la generación de los landmarks rígidos.

De los histogramas es posible deducir que las distribuciones de probabilidad de los errores tienen en general formas irregulares que si bien pueden ser descritas mediante una media y covarianza, contienen más información que la

aportada por estos dos estadísticos. Para poder mantener una escala regular en los histogramas, los fallos se dejan en la celda número 60 de los histogramas, ya que cualquier error mayor a 60 representa un error en la ruta mayor a la escala del mapa, por lo que se considera un fallo.

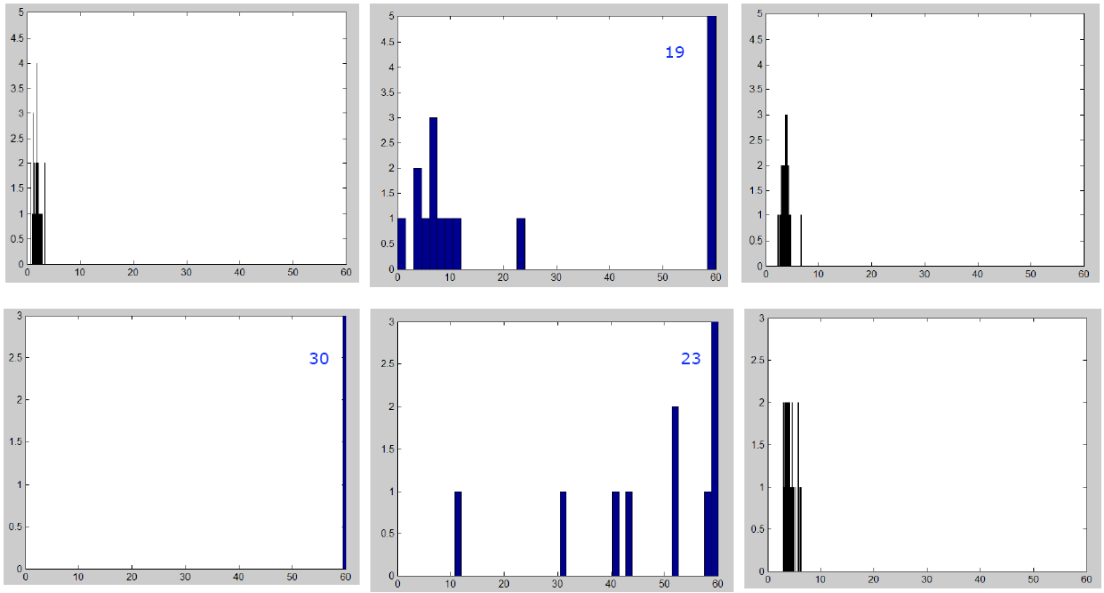



**Figura 13: Histogramas del error para las seis pruebas sobre al ruta con forma de U. El número de fallos se indica en azul en cada histograma.**

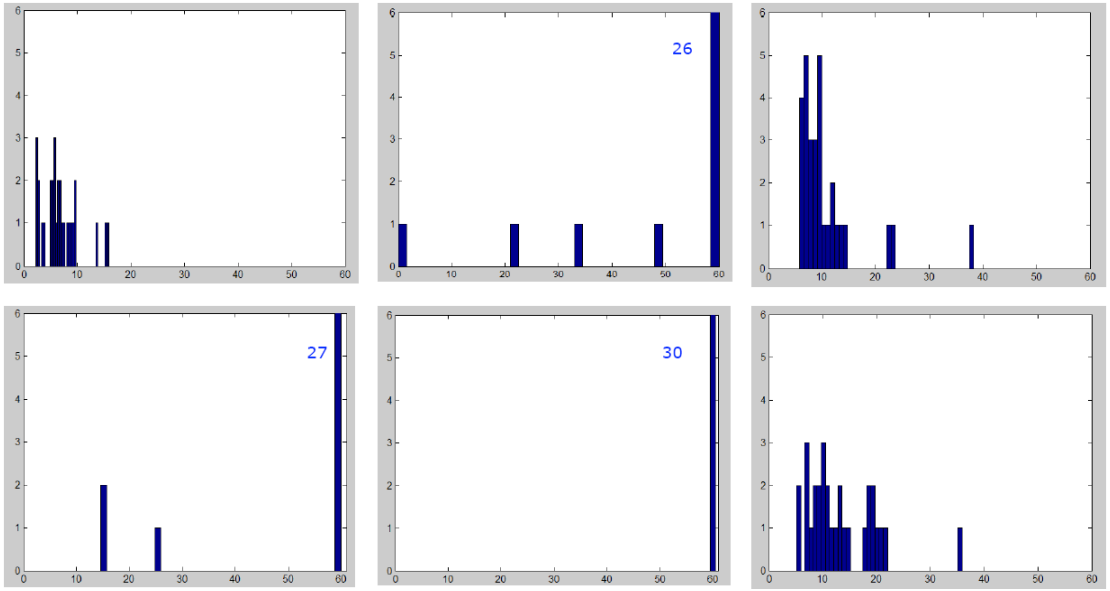



**Figura 14: Histogramas para las 6 pruebas sobre la ruta con forma de S. El número de fallos se indica en azul en cada histograma.**

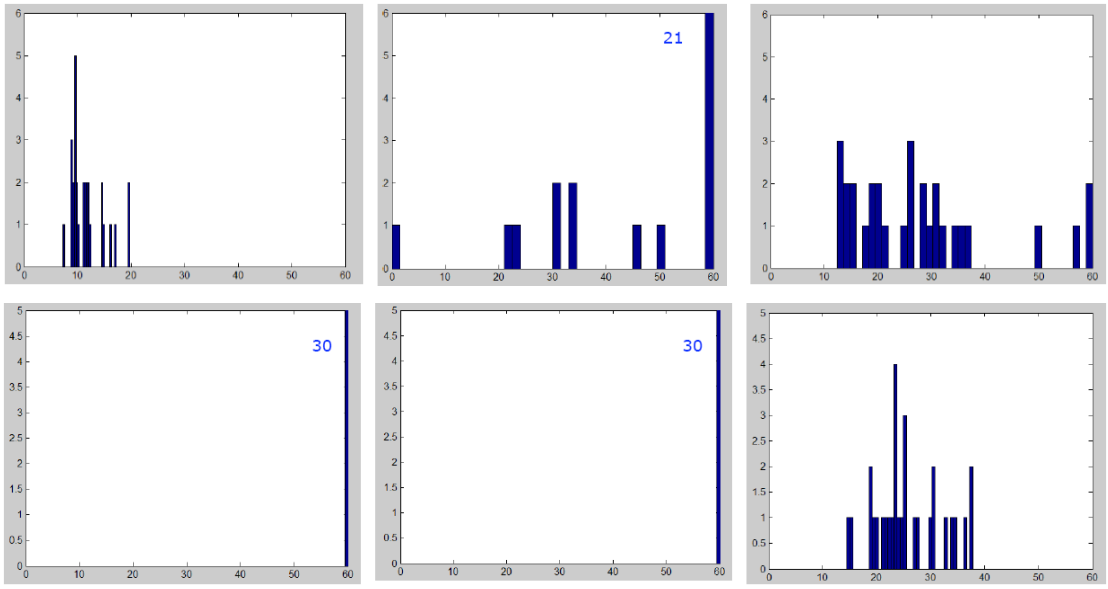



**Figura 15: Histogramas para las 6 pruebas sobre la ruta con pérdida continua. El número de fallos se indica en azul en cada histograma.**

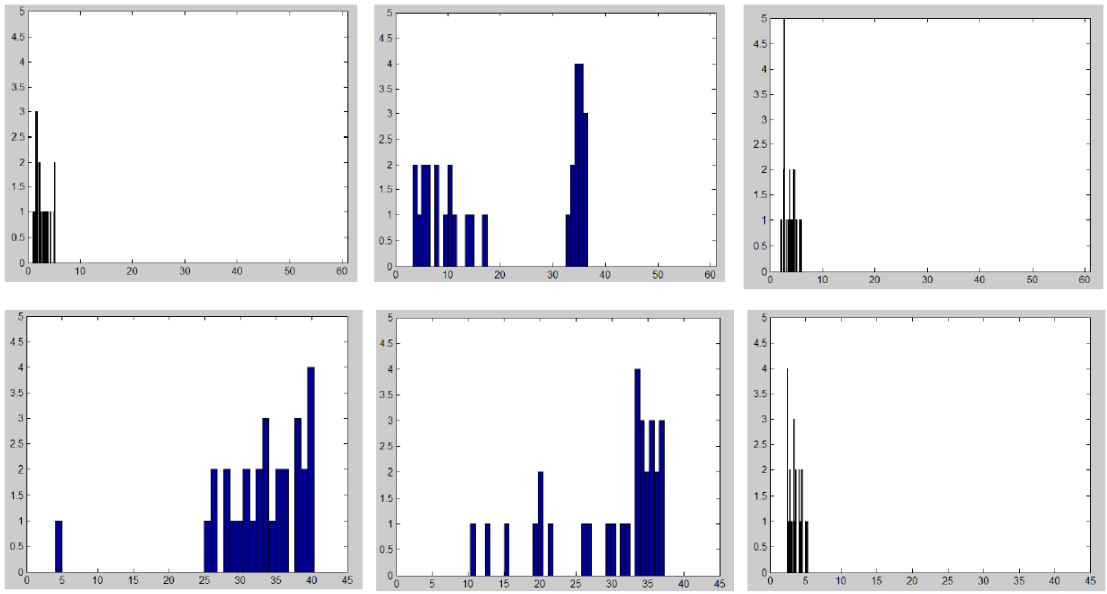


**Figura 16: Histogramas para las 6 pruebas sobre la ruta al azar. El número de fallos se indica en azul en cada histograma.**

De los histogramas se puede concluir que el sistema SLAM presenta poca estabilidad numérica cuando se usan solamente 4 landmarks puntuales, ya que el

histograma no converge a una distribución suave. Este resultado es coherente con el mostrado en la tabla.

De la tabla y de los gráficos de los histogramas se puede concluir que los landmarks rígidos deben ser creados usando la metodología de máxima covarianza de los puntos del cuerpo. La razón de esto es que la covarianza original se descompone en dos partes: la covarianza de la pose del landmark rígido, la cual se reduce con el tiempo, y la covarianza de los puntos del cuerpo, la cual se mantiene constante. Si la covarianza es traspasada por completo a la pose del landmark rígido, con el tiempo la covarianza del landmark completo va a ser cero, lo cual va a causar una subestimación grave de la covarianza de las proyecciones de los puntos del cuerpo en la pantalla. Se debe recordar que la posición de los puntos del cuerpo no se actualiza, por lo cual el error que presentaban en el momento de la agrupación se va a mantener en el tiempo. Es por eso que deben tener matrices de covarianza asociadas que representen la incertidumbre provocada por dicho error.

Un gráfico que muestra el tiempo de ejecución requerido por cada iteración de predicción-actualización del EKF en función del número de características almacenadas en el mapa se muestra a continuación.

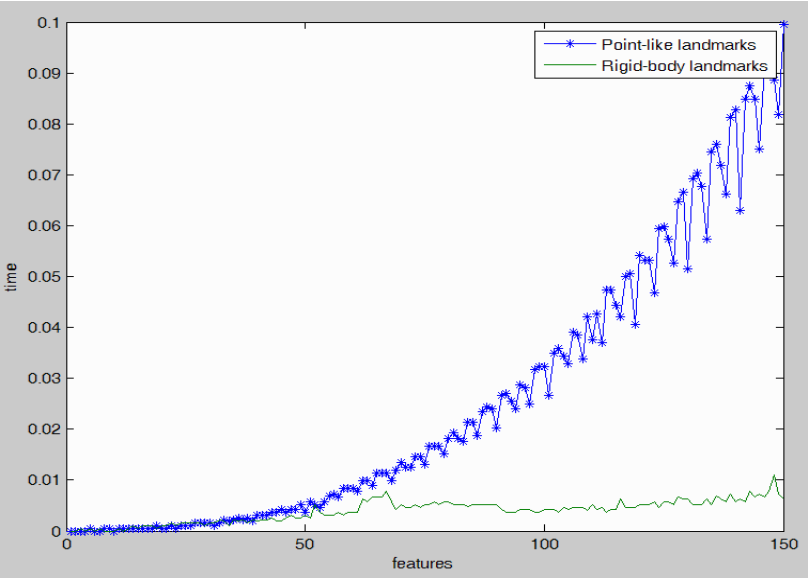



**Figura 17: Tiempo de ejecución requerido por cada iteración EKF. El sistema basado en landmarks-punto se muestra en azul, y el basado en landmarks rígidos en verde.**

De la figura 17 se puede concluir que el sistema basado en landmarks rígidos es capaz de manejar una gran cantidad de descriptores de forma muy eficiente, ya que el aumento en el tiempo computacional que se produce al agregar nuevos descriptores al mapa es mucho menor en el mapa formado por landmarks rígidos que en el mapa formado por landmarks punto. De este modo, se pueden lograr mapas densos, que consideran muchos dscriptores, en tiempo real.

## 6.2 Experimentos usando secuencias de video reales

Para poder evaluar el sistema sobre secuencias de video reales se desarrolló una interfaz de visualización gráfica simple usando FLTK [26], la cual permite visualizar el funcionamiento del sistema en tiempo real. La interfaz muestra, entre otras cosas, la imagen procesada, los descriptores y elipses de error del test chi-cuadrado, la matriz $P$, $H$ y el análisis de covarianza usado para decidir la transformación de los conjuntos de landmarks punto a landmarks rígidos (ver Figura 17). En una de las ventanas se muestra el mapa usando la extensión para OpenGL que incluye FLTK. En dicha ventana se muestra la ruta reconstruida, los landmarks puntuales existentes como puntos azules, y finalmente los landmarks rígidos existentes mostrando la pose como un sistema de referencia y los puntos del cuerpo como puntos blancos. La cámara se muestra como un cubo de color amarillo, el cual tiene otro pequeño cubo adherido a una de sus caras que representa el lente de la cámara. La interfaz de visualización se activa cuando se encuentra el flag `#ifdef __COMPAT_FLTK__` y es portable, ya que el sistema FLTK se puede ejecutar nativamente tanto en Windows como en sistemas Posix (Linux, Unix y Mac OS). El sistema de visualización implementado se muestra en la Figura 18.

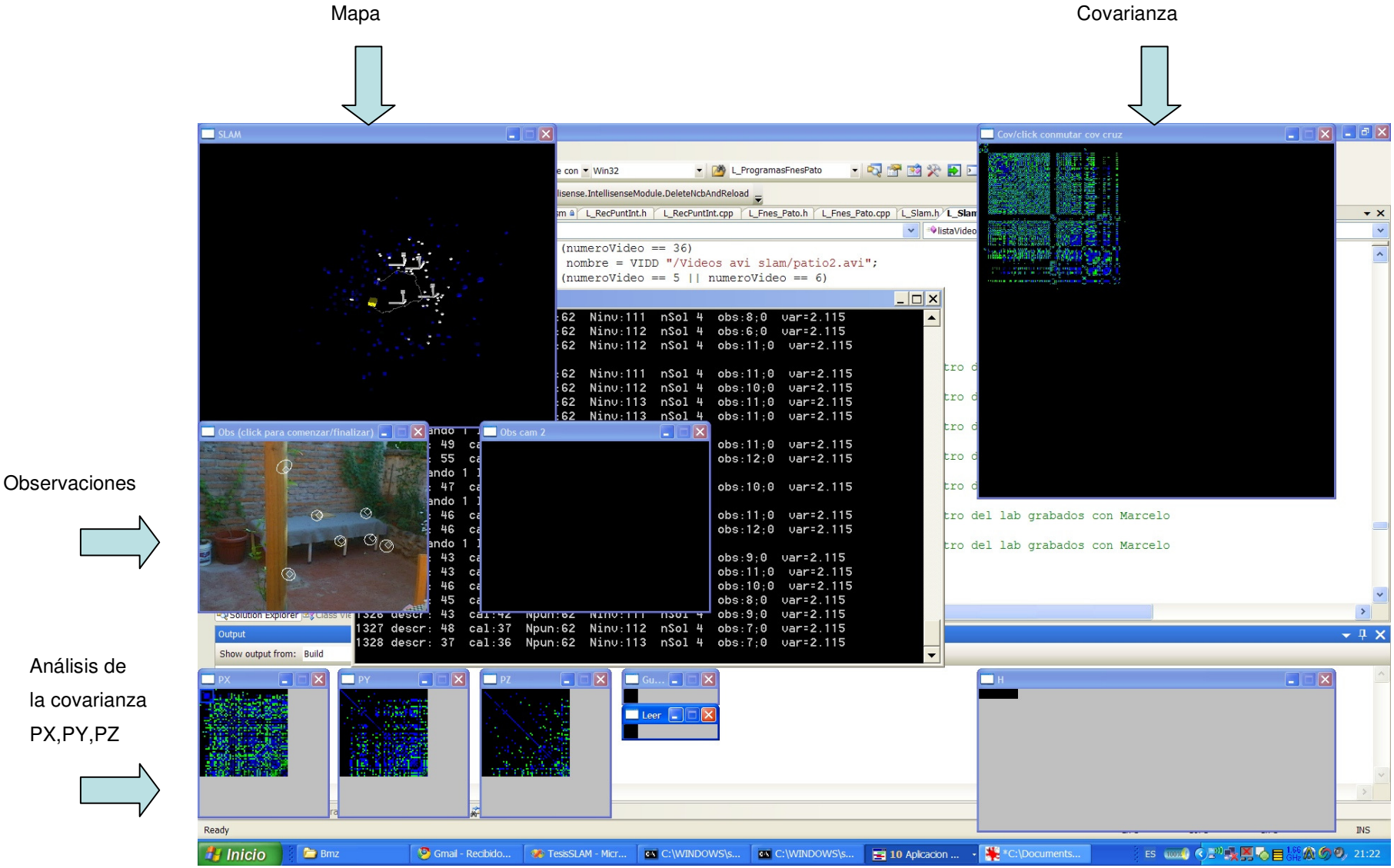
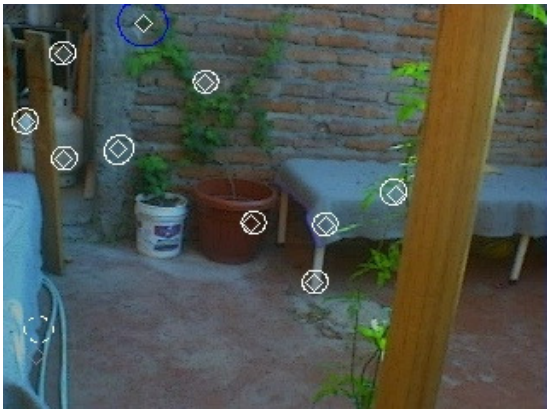
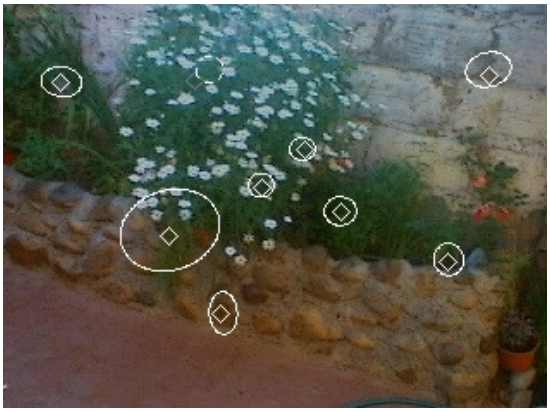



**Figura 18: Interfaz de visualización del sistema**

Se grabaron 7 secuencias de video, en las cuales una cámara web realiza una trayectoria a través de un jardín, llegando a un punto final cercano al punto inicial de la ruta. La ruta se pudo reconstruir en forma satisfactoria en las 7 secuencias de video capturadas. Algunas imágenes seleccionadas de la primera secuencia de video se muestran en la Figura 19, y la formación del mapa se muestra en la Figura 20.





ERROR: undefined
OFFENDING COMMAND: f‘~

STACK: