Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Detección y cierre de ciclos en sistemas SLAM basados en visión estéreo

Tesis presentada para optar al título de
Licenciado en Ciencias de la Computación

Gastón Ignacio Castro

Director: Dr. Taihú Pire
Codirector: Dr. Pablo De Cristóforis
Buenos Aires, 2016

Detección y cierre de ciclos en sistemas SLAM basados en visión estéreo

Las aplicaciones relativas a robots autónomos móviles requieren la construcción de una representación o mapa del entorno y la localización fiable del robot en el mismo. En SLAM (*Simultaneous Localization And Mapping*) se plantea abordar ambos problemas de manera simultánea. Los métodos que resuelven SLAM deben tratar con la acumulación del error en la estimación de la posición y orientación del robot, que va creciendo de forma no acotada a medida que aumenta la longitud de la trayectoria recorrida y la dimensión del mapa construido. Para abordar este problema los métodos de SLAM deben contar con la capacidad de detectar una región previamente visitada por el robot y ajustar tanto, la localización del robot como el mapa construido minimizando el error acumulado hasta el momento. Este problema se conoce como *Loop Closure*, que incluye la detección y cierre de ciclos en la trayectoria realizada por el robot.

El presente trabajo propone una solución al problema de la detección y cierre de ciclos en sistemas de SLAM que utilizan cámaras estéreo como sensor principal. Para lograr este objetivo, se divide el problema en tres etapas: la detección de ciclos en la trayectoria, el cálculo del desvío cometido en la localización y la corrección tanto de la localización actual del robot como del mapa construido hasta el momento. Por cada etapa se lleva a cabo una profunda revisión del estado del arte, exponiendo aquellos métodos y técnicas que motivaron la solución propuesta en este trabajo.

Para realizar la detección de ciclos se entrena un vocabulario visual en una etapa previa, de manera de obtener una discretización del espacio de descriptores de las imágenes capturadas por la cámara. Este vocabulario visual se utiliza luego para obtener una representación eficiente de las imágenes (*bag-of-words*) que permite evaluar la similitud entre las imágenes obtenidas en un determinado momento y las que componen el mapa construido. A través de la comparación de vectores *bag-of-words* asociados a cada imagen se hallan candidatos a ciclos que se validan utilizando técnicas geométricas para establecer la consistencia espacial entre los distintos momentos de la trayectoria. Por último, se efectúan los cierres de ciclo y corrección del mapa por medio de técnicas de optimización de grafos y algoritmos de minimización no lineal.

La solución propuesta se implementó como un módulo para ser incorporado al sistema S-PTAM (*Stereo Parallel Tracking And Mapping*) de código abierto. Los experimentos realizados con *datasets* de dominio público bajo el framework ROS (*Robot Operating System*) muestran que la detección y cierre de ciclos mejoran drásticamente la estimación de la localización del robot en el entorno y la calidad del mapa construido, permitiendo al mismo tiempo la ejecución del sistema en tiempo real.

Loop detection and closure on stereo-vision based SLAM systems

Applications concerning mobile autonomous robots requires building a representation or map of the environment, and the accurate localization of the robot in it. SLAM (*Simultaneous Localization and Mapping*) proposes to address both problems simultaneously. The methods that solve SLAM must deal with error accumulation on the estimation of position and orientation, which grows unbounded as it increases the length of the path traveled and the size of the map constructed. To address this issue, SLAM methods must be able to detect a region previously visited by the robot and adjust both, the robot's localization and the map built minimizing the accumulated error up to that moment. This problem is known as *Loop Closure*, including the detection and closure of loops in the trajectory carried out by the robot.

This work proposes a solution to the problem of detecting and closing loops in SLAM systems that use stereo cameras as the main sensor. To achieve this objective, the problem is divided into three stages: detection of loops in the trajectory, calculation of the deviation occurred in the localization and correction of both, current robot's localization and the map built up to that moment. For each stage a thorough review of the state of the art is carried out, exposing those methods and techniques that led to the solution proposed in this work.

To perform the loop detection a visual vocabulary is trained in a previous stage in order to obtain a discretization of the descriptor space of images captured by the camera. This visual vocabulary is then used to obtain an efficient representation of images (*bag-of-words*) that allows the similarity evaluation of images obtained at a given time and those composing the map built. Loop candidates are formulated through the comparison of the *bag-of-words* vectors associated to each image, these candidates are then validated using geometric vision techniques for the establishment of the spatial consistency between the different moments of the trajectory. Finally, loop closing and map correction is performed using graph optimization techniques and nonlinear minimization algorithms.

The proposed solution is implemented as a module to be incorporated into the S-PTAM (*Stereo Parallel Tracking And Mapping*) open source system. Experiments performed with public domain *datasets* under the ROS (*Robot Operating System*) framework, show that detection and closure of loops dramatically improve the estimation of the robot localization on the environment and quality of the map built, while allowing system execution in real time.

A mi familia

mis padres, mi hermana y mi perro

Agradecimientos

Mi familia me acompaño en cada paso, su apoyo incondicional es lo que me permite descubrir mi camino en la vida. Mis padres y mi hermana siempre conmigo para aconsejarme e impulsarme. Sasha, gran compañera y confidente en mi vida, la extraño todos los días. Los momentos en familia con mis primos, tías, tíos y mi abuela Delia formaron mi infancia, grandes momentos de felicidad. Un buen día llego Adri, trayendo alegres experiencias. Les agradezco a todos por estar siempre.

Lulii, esa personita especial, siempre conmigo para darme ánimos y cuidarme. Me acompañó durante toda la realización y escritura de esta tesis. Amo todo camino juntos.

Quiero agradecer a aquellas personas con las que cruce camino y compartí momentos, ideas, risas y delirios. Uno aprende y se forma de experiencias, todas estas personas tomaron parte en algún momento de mi vida y siempre seré lo que soy gracias a todos ellos.

Mi primer mejor amigo, Nico, a quien admiro muchísimo. Mis amigos de la primaria con los que crecí, aprendí y me divertí tanto. Facu, con quien no paro de bizarrearla, charlas que pueden llegar hasta el campo de la metafísica y más allá. Baruffa, gran compañero de aventuras digitales.

Germán, ese amigo que dejé ir y perdí para siempre, me enseño a valorar lo misterioso y enigmático. En esas noches de vicio jugando al Final Fantasy comencé a asombrarme por la computación y nunca me detuve.

Los pibes del secundario, grandes compañeros de vida junto con los que crecí y muté. Joako, Fede, Aye, Agus, Pinus, Conny, Carito, el Chineeh, Caro, el Etzee, Sòfia, Glen, Molle, Sophie, Male y varios limados más. Ese curso fue una fiesta del que saqué amigos para toda la vida. Male, mi mejor amiga, con la que caminé las mismas cuadras cientos de veces. Esas primeras giras con Marianito, Pinus y Juanma; increíbles veranos. Banderas, rock y recitales con Aye y Juli. Desgastando noches sin final con La Banda del Tío Zenden; Alan, Leo, Rafa, Santi, el Tío, Pinus, Diega, el Valen. Verdulerías, caminatas bajo la lluvia y tantas giras con ese indetenible rincón; Valen, Diega, Aye, Pinus, Etzee, Fabi, el Duende, Facu, Luquita, Rochi, Jons y tantas otras figuras del rock. Pinus, una clara constante, testigo y artífice de grandes momentos. Una gran época donde todos fuimos verdaderos dueños del tiempo.

Grandes giras de rock y metal entre las Sierras con Psicolerica; Ferdinand y Lito, titanes de la guitarra. Raque siempre deleitándonos con su sonrisa y comidas exquisitas. Ya sea tocando blues o haciendo el ridículo en un casamiento, todos buenos momentos con Euge, Fede, Sol, Miguel el repositor, Dani, Ale y el Patman.

Lysu, una maravillosa persona que se sumó un día y expandió nuestros horizontes. Aventuras recorriendo montañas, lagos y morros junto a Uli, Mati, Jor y el Gordis. Descarrilados cumpleaños y emotivos años nuevos con Agus. Tefis siempre haciendo el aguante y cayendo con alguna bizarreada. Increíbles tardes descubriendo cosas nuevas con Juanis y la Chompis.

Aprendí lo que era trabajar en equipo con ese legendario BlackBerry Team; Pablito, el titán Ale y Gus Furini. Dando todo en cada cursada junto al Dream Team; Martín, Pablanga y Feli. Incontables tardes estudiando para llegar hasta el final de esta carrera. Gera y Martín Celave, grandes compañeros en esta larga travesía.

Los chicos del laboratorio de Robótica me abrieron las puertas y cada día aprendo con ellos sobre este gran mundo de la investigación; Mati, Facu, Thomas, Pablo, Taihú, Pato y Carlos. Marta, Pachi, Julio, Dani y los chicos de Imágenes me dieron su apoyo en cada paso. Taihú me acompaño a lo largo de toda esta tesis; su apoyo, dirección, esfuerzo y amistad fue lo que la hizo posible.

Índice general

Capítulo 1

Introducción

La robótica móvil tiene un fuerte impacto en ámbitos tales como la industria, la agricultura, y en un espectro de aplicaciones que va desde misiones de exploración planetaria, tareas de búsqueda y rescate de personas en territorios peligrosos para el ser humano, hasta la realización de actividades domésticas o de servicio. Uno de los desafíos actuales de la robótica móvil es lograr completa autonomía, es decir, dotar a un robot de la capacidad de llevar a cabo sus tareas sin la necesidad de un operador humano. Para lograr esto el problema de la navegación autónoma se puede dividir en tres: el de la localización, la construcción de mapas del ambiente y la planificación de las trayectorias. El problema de la localización consiste en dotar a un robot móvil de la habilidad de inferir su propia locación dentro de un entorno conocido. La construcción de una representación detallada o mapa del entorno puede ser abordada a partir de una correcta localización del robot en el ambiente. Y finalmente, conociendo su localización y contando con una representación del ambiente que lo rodea, el robot debe ser capaz de planificar una trayectoria que le permita moverse de manera segura mientras realiza la tarea de forma autónoma. Desde el punto de vista de la autonomía, un robot capaz de construir mapas del entorno mientras realiza su tarea, y localizarse por sus propios medios, tiene una gran ventaja comparativa sobre aquellos que dependen de mapas estáticos predefinidos y/o un sistema de posicionamiento externo.

Dependiendo del sistema de sensado a bordo del robot, el entorno por el cual se desplaza es observado periódicamente de manera de proveer suficiente información a los métodos de localización. Los métodos de localización basados en el reconocimiento de referencias externas, conocidos como de localización absoluta o global, utilizan marcas artificiales en el ambiente (por ejemplo patrones visuales, señales de radio, etc.), localización satelital (GPS), o referencias naturales como el campo gravitatorio o magnético de la tierra. Si bien este tipo de sistemas son estables a largo plazo, presentan ruido y/o imprecisiones en las estimaciones a corto plazo y se encuentran limitados en un área de funcionamiento determinada (por ejemplo, la localización satelital es viable solo en ambientes exteriores donde sean visibles al menos tres señales de satélites).

Se conoce como navegación por estima (*dead reckoning*) al proceso por el cual se estiman los cambios en la pose (posición y orientación) a partir de la última pose conocida y la integración de la información proveniente de diferentes sensores montados en el robot. Este tipo de localización relativa tiene la desventaja de que errores pequeños, cometidos al estimar el desplazamiento, se acumulan ilimitadamente produciendo estimaciones de decreciente calidad. Algunos de estos métodos utilizan, por ejemplo, mediciones en la rotación de las ruedas por

medio de *encoders* (robots terrestres) lo que se conoce como odometría y/o información proveniente de unidades de medición inercial (*inertial measurement unit*, IMU) para estimar el desplazamiento bajo un modelo de movimiento establecido. Existen métodos de localización relativa que utilizan una o más cámaras como sensor (lo que se conoce como odometría visual) los cuales son capaces de estimar el movimiento relativo con 6 grados de libertad (posición y orientación en un espacio tridimensional). Esto los hacen especialmente atractivos para robots que se desplazan en un espacio tridimencional, como por ejemplo, robots aéreos no tripulados, robots artrópodos o submarinos automatizados.

Resolver la localización a partir de un mapa del ambiente dado *a priori* puede ser resuelto de diversas maneras y utilizando distintos sensores. El problema resulta más atractivo cuando es preciso localizar el robot mientras se construye un mapa del ambiente. En este caso, la localización y la construcción del mapa deben ser consideradas simultáneamente, lo que se conoce como SLAM (*Simultaneous Localization And Mapping*) [1, 2].

1.1. Motivación y definición del problema

Los sistemas que resuelven SLAM deben tratar con la acumulación del error en la estimación de la posición y orientación del robot que va creciendo de forma no acotada a medida que aumenta la longitud de la trayectoria recorrida y la dimensión del mapa construido. Para abordar este problema los métodos de SLAM deben contar con la capacidad de detectar una región previamente visitada por el robot. Esto introduce nueva información valiosa que permite ajustar tanto la localización del robot como el mapa construido, reduciendo el error acumulado hasta el momento. Este problema se conoce como *Loop Closure*, que incluye la detección y cierre de ciclos en la trayectoria realizada por el robot.

El presente trabajo propone una solución al problema de la detección y cierre de ciclos en sistemas SLAM que utilizan cámaras estéreo como sensor principal. Para lograr este objetivo se divide el problema en tres etapas: la detección de ciclos en la trayectoria, el cálculo del desvío cometido en la localización y la corrección tanto de la posición y orientación actual del robot como del mapa construido hasta el momento. Por cada etapa se lleva a cabo una profunda revisión del estado del arte, exponiendo aquellos métodos y técnicas que motivaron la solución propuesta en este trabajo.

Es menester destacar que el problema de la detección y cierre de ciclos esta fuertemente relacionado con el problema de la relocalización que consiste en relocalizar al robot en el mapa luego de un fallo irrecuperable en la localización debido a la oclusión inesperada de los sensores, cambios repentinos en el entorno o la acción de un agente externo que manipulan el robot de manera brusca (*kidnapping problem*). La diferencia principal radica en que la detección y cierre de ciclos no debe interferir con el funcionamiento del sistema SLAM si este se encuentra correctamente localizado.

1.2. Objetivos

El objetivo de este trabajo es hacer un análisis exhaustivo del estado del arte sobre los diferentes métodos disponibles para resolver el problema de *Loop Closure*, de manera de obtener una solución eficiente que permita su aplicación como parte de un sistema SLAM. Además, se propone implementar la solución alcanzada como un módulo capaz de detectar y cerrar ciclos de manera eficiente en la trayectoria percibida por el sistema Visual SLAM desarrollado en el

Laboratorio de Robótica y Sistemas Embebidos (LRSE) denominado *Stereo Parallel Tracking And Mapping* (S-PTAM) [3].

1.3. Organización del trabajo

El en Capítulo 2 se define el modelo geométrico de la cámara como un sensor exteroceptivo y la geometría epipolar que involucra a las cámaras estéreo. En el Capítulo 3 se define el marco teórico del problema a resolver junto con una reseña de distintos métodos de localización y sistemas Visual SLAM utilizados en la actualidad. En el Capítulo 4 se aborda el problema de la detección de ciclos. El Capítulo 5 está dedicado al cálculo de la transformación relativa entre cámaras estéreo. En el Capítulo 6 se analizan diferentes métodos para el cierre de ciclos y la corrección de la trayectoria sobre los distintos tipos de mapas. El Capítulo 7 detalla la solución propuesta en función del estudio realizado en los capítulos anteriores. En el Capítulo 8 se muetran los experimentos realizados con *datasets* de dominio público bajo el *framework* ROS (*Robot Operating System*). Finalmente el Capítulo 9 concluye el trabajo realizado y propone alguos lineamientos como trabajo futuro.

Capítulo 2

Cámaras como sensores

En el último tiempo, el desarrollo de la robótica móvil y de la visión por computadora han dado lugar a una nueva disciplina conocida como visión robótica (*robot vision*) que propone utilizar cámaras como los sensores primarios para capturar la información del ambiente.

Las cámaras poseen varias ventajas por sobre otros sensores. Para empezar son mucho más económicas, de bajo consumo y fáciles de montar en robots móviles que otros sensores típicos como los lásers. Las nuevas generaciones de cámaras digitales pueden proporcionar datos de alta resolución en tiempo real con rangos de medición virtualmente ilimitados. Además, las cámaras son sensores pasivos dado que no emiten señales al entorno, y por tanto, no interfieren con otros sensores. En la actualidad, las unidades de procesamiento disponibles en un robot móvil son capaces de satisfacer los requerimientos de cómputo de los diferentes algoritmos de procesamiento de imágenes. Por lo tanto, se puede afirmar que las cámaras a bordo de los robots móviles constituyen un gran beneficio y resultan una parte importante en el sistema de sensado y percepción.

La utilización de cámaras en métodos de localización relativa (*visual odometry*) [4] permiten percibir el movimiento del robot en los 6 grados de libertad posibles (3 de traslación y 3 de rotación) utilizando como referencia el entorno que lo rodea de manera independiente al accionar de sus actuadores. Esto permite superar el escenario que se presenta en la odometría basada en *encoders* donde las ruedas pueden patinar invalidando la estimación de la pose. El modelo de sensado en el caso de las cámaras consiste básicamente en un mapeo entre el entorno tridimencional y el plano de la imagen bidimensional. Dependiendo del tipo de cámara (monocular, estéreo, omnidireccional, etc.) es posible utilizar diferentes modelos matemáticos que permitan relacionar puntos del mundo con su respectiva representación en la imagen.

A continuación se exponen los detalles del modelo de cámara monocular y estéreo expuestos en [5] junto con resultados geométricos de interés.

2.0.1. Modelo y geometría de una cámara monocular

En computación visual y campos relacionados, la cámara es definida como una relación entre el mundo 3D y una imagen 2D. Existen diversos modelos que representen una cámara, el más simple y más usado es el modelo de cámara *Pinhole*.

El modelo de cámara *Pinhole* permite describir la relación matemática existente entre las coordenadas de un punto 3D del espacio y su proyección en el plano de la imagen de una cámara ideal. En dicho modelo la apertura de la cámara es descripta como un punto y no se considera la lente de la misma. De esta manera el modelo no toma en cuenta la distorsión

geométrica que pudiera existir en la imagen ni objetos difuminados por encontrarse fuera de foco. Sin embargo, existen métodos que permiten corregir la distorsión existente y compensar muchos de los efectos negativos antes mencionados. En la mayoría de las aplicación de visión el modelo resulta una descripción razonable del accionar de una cámara.

La cámara se modela de manera que los puntos del espacio sean proyectados en el plano de la imagen o plano focal como se muestra en la Figura (a) 2.0.1. Para esto se definen los parámetros intrínsecos de la cámara como:

$$K = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.0.1}$$

donde se asume que los píxeles de la cámara son cuadrados, y por tanto, $f_x = f_y = f$ determina la distancia desde el centro de la cámara al plano de la imagen. La recta desde el centro de la cámara, perpendicular al plano de la imagen, se lo conoce como eje principal y el punto donde interseca con el plano de la imagen se lo llama punto principal $\mathbf{p} = (p_x, p_y)$.

Por similitud de triángulos y utilizando coordenadas homogéneas, es posible definir una relación entre un punto 3D $X_c = (x, y, z, 1)^\top$ y su proyección 2D $u = (x', y', 1)^\top$ perteneciente al plano de la imagen (Fig.(b) 2.0.1).

Se define entonces $(x, y, z, 1)^\top \underset{K}{\rightarrow} (f\frac{x}{z} + p_x, f\frac{y}{z} + p_y, 1)^\top$ como:

$$[K|\overrightarrow{0}](x, y, z, 1)^\top = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} fx + zp_x \\ fy + zp_y \\ z \end{pmatrix} = \begin{pmatrix} f\frac{x}{z} + p_x \\ f\frac{y}{z} + p_y \\ 1 \end{pmatrix}$$

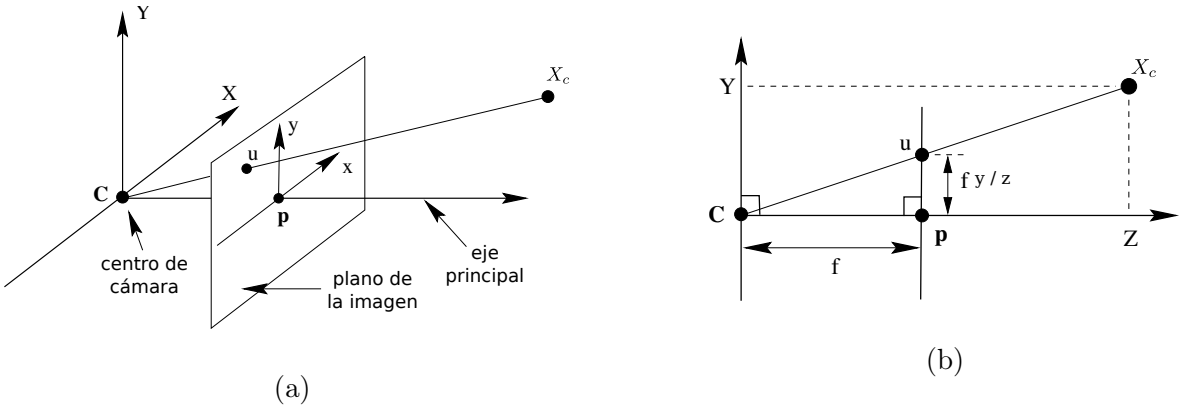


Fig. 2.0.1: Modelo geométrico *Pinhole*. Donde **C** es el centro de cámara, $\mathbf{p} = (p_x, p_y)$ punto principal, $X_c = (x, y, z, 1)$ punto 3D en coordenadas de la cámara y la cámara se considera centrada en el origen del eje de coordenadas.

Todo punto 3D puede ser expresados en relación a diferentes marcos de coordenadas, X_c se corresponde al punto X en referencia al marco de coordenadas de la cámara. En general los puntos del ambiente son expresados en referencia al sistema de coordenadas conocido como "mundo" (*world coordinate system*), el cual esta relacionado con el sistema de coordenadas de la cámara a través de una rotación y una traslación.

Sea $X_w = (X, Y, Z, 1)^\top$ un punto del espacio 3D en referencia al sistema de coordenadas del mundo y X_c el mismo punto expresado en referencia a la cámara, es posible definir una transformación T_{cw} tal que:

$$X_c = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = T_{cw} X_w \tag{2.0.2}$$

donde R es una matriz de rotación 3×3, t un vector de traslación 3×1 y T_{cw} pertenece al grupo de movimientos de cuerpo rígido 3D (*Lie Group*, **SE(3)**) [6]. De esta manera, utilizando la matriz de parámetros intrínsecos K, se define la matriz de proyección P que nos permitirá proyectar cualquier punto del espacio en coordenadas del mundo al plano de la imagen:

$$P = K[R|t] \tag{2.0.3}$$

La matriz P posee 9 grados de libertad (3 por K, 3 por R y 3 por t) y por lo tanto:

$$u = PX_w \tag{2.0.4}$$

donde u es la proyección en el plano de la imagen (3×1) del punto X_w del mundo 3D.

2.0.2. Modelo y geometría de cámaras estéreo

La geometría proyectiva entre dos cámaras es conocida como geometría epipolar (*epipolar geometry*). Esta es independiente de la escena observada, y depende únicamente de los parámetros internos y poses relativas de las cámaras involucradas. Esencialmente la geometría epipolar caracteriza la relación entre un par de cámaras como la intersección de sus planos imagen con una familia de planos que contengan la línea que une los centros de cámaras. Esta línea se la conoce como línea base o *baseline* (Fig. 2.0.2). El estudio de la geometría epipolar es de interés dado que permite corresponder puntos pertenecientes a los planos imagen de distintas cámaras.

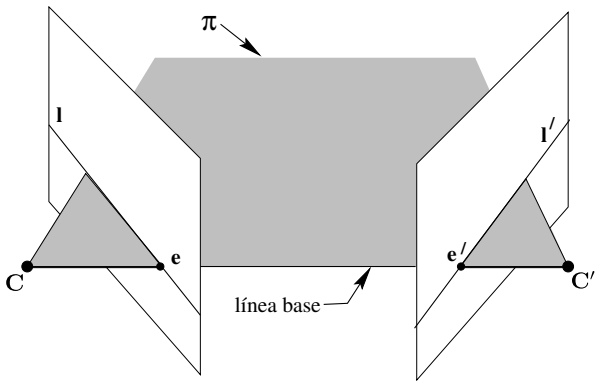


Fig. 2.0.2: La línea base interseca con cada plano imagen en los epipolos $\boldsymbol{e}$ y $\boldsymbol{e'}$. Todo plano $\boldsymbol{\pi}$ que contenga la línea base es un plano epipolar, e interseca los planos imagen formando las líneas epipolares $\boldsymbol{l}$ y $\boldsymbol{l'}$.

2.0.2.1. Correspondencia entre puntos

Sea X un punto en el espacio 3D observado simultáneamente por dos cámaras distintas $\mathbf{C}$ y $\mathbf{C}'$, de manera que x representa la proyección de X en el plano imagen de $\mathbf{C}$ y x' la proyección de X en el plano imagen de $\mathbf{C}'$. Es posible observar que los puntos imagen x y x', el punto en el espacio X y los centros de cámara son coplanares (Fig.(a) 2.0.3). Es decir, pertenecen a un mismo plano al cual denotaremos como π. Más aún, las rectas formadas por el centro de cada cámara y los puntos en la imagen x y x' intersecan en el punto del espacio 3D X.

Suponiendo el escenario donde se conozca solamente el punto x perteneciente al plano imagen de la cámara $\mathbf{C}$, se puede restringir el espacio donde buscar el correspondiente punto x' en el plano imagen de la cámara $\mathbf{C}'$. El plano π puede ser determinado por medio de la línea base entre los centros de cámara y el vector dirección que une el centro de la cámara $\mathbf{C}$ con el punto imagen x. De esta forma, sabiendo que x' debe pertenecer a π, se asegura que x' debe pertenecer a la recta l' definida por la intersección entre el plano π y el plano imagen de la cámara $\mathbf{C}'$ (Fig.(b) 2.0.3). La recta l' se la conoce como la línea epipolar (*epipolar line*) de x y es especialmente útil para acotar la búsqueda del correspondiente punto imagen x' en algoritmos de asociación de características visuales.

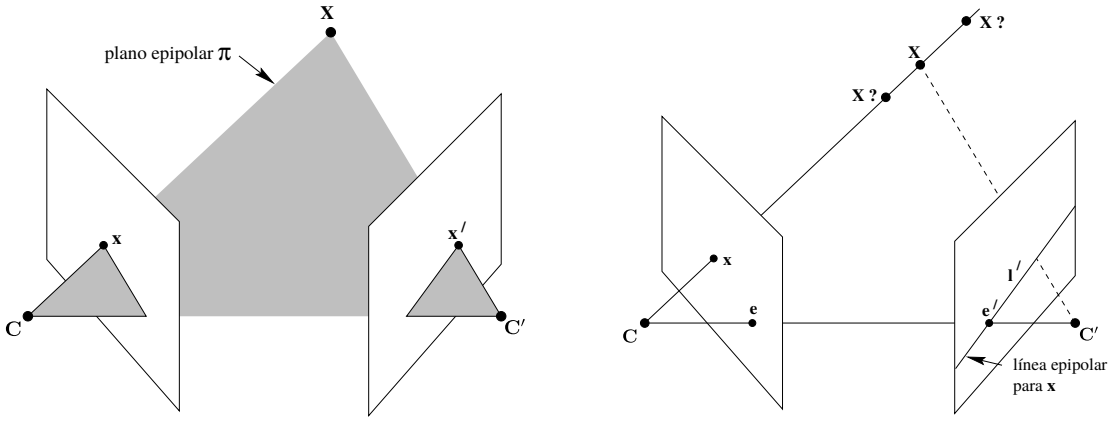


Fig. 2.0.3: (a) Los centros de cámara, el punto 3D $\boldsymbol{X}$ y los puntos imagen $\boldsymbol{x}$ y $\boldsymbol{x'}$ pertenecer a un mismo plano $\boldsymbol{\pi}$. (b) La recta definida por el centro de cámara $\mathbf{C}$ y el punto imagen $\boldsymbol{x}$ es observada en el plano imagen de $\mathbf{C}'$ como la línea epipolar $\boldsymbol{l'}$. El punto en el espacio X, el cual es proyectado en el plano imagen de $\mathbf{C}$ como $\boldsymbol{x}$, debe ser observado por la cámara $\mathbf{C}'$ sobre la línea epipolar $\boldsymbol{l'}$.

2.0.2.2. La matriz fundamental

La matriz fundamental $\boldsymbol{F}$ es una representación matricial de la geometría epipolar que permite realizar un mapeo entre puntos de un plano imagen a sus correspondientes líneas epipolares. Dadas un par de cámaras, se mencionó anteriormente que para cada punto x perteneciente a uno de los planos imagen, existe una determinada línea epipolar l' presente en el plano imagen opuesto donde debe encontrarse el punto imagen x' correspondiente (Fig.(a) 2.0.3). Por lo tanto, se requiere que la matriz fundamental $\boldsymbol{F}$ satisfaga:

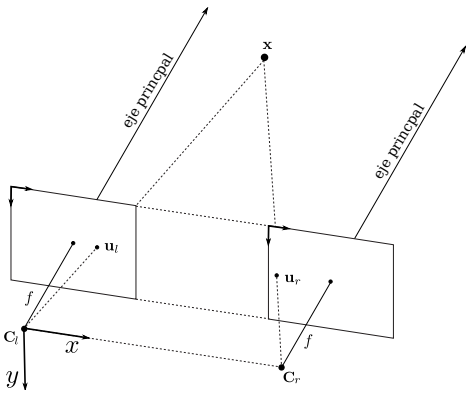


Fig. 2.0.4: Par estéreo rectificado, notar que los planos imagen de las cámaras se encuentra alineadas.

$$\boldsymbol{F}x = l' \tag{2.0.5}$$

Representando las líneas del plano imagen en coordenadas homogéneas: $l = (l_1, l_2, l_3)^\top$tal que $l_1x + l_2y + l_3 = 0$ y dado que el punto x' pertenece a la línea l', por propiedades del espacio proyectivo se cumple que:

$$(x')^\top l' = 0 \tag{2.0.6}$$

de lo cual se obtiene:

$$(x')^\top \boldsymbol{F}x = 0 \tag{2.0.7}$$

De esta forma se condiciona la relación entre los puntos imagen de ambas cámaras utilizando la matriz fundamental. Existen distintas interpretaciones geométricas para la matriz $\boldsymbol{F}$ y es posible derivarla de varias maneras. En la práctica calcularla permite la implementación de eficientes métodos de rectificación de imágenes, asociación de características visuales y triangulación de puntos del ambiente.

2.0.2.3. Rectificación estéreo

La rectificación estéreo proyecta las imágenes obtenidas por las cámaras a un plano imagen en común, de manera que puntos correspondientes (Sección 2.0.2.1) se encontrarán alineados en una misma fila (Fig. 2.0.4). Esta proyección permite considerar las cámaras involucradas como paralelas, es decir, relacionadas por únicamente por una translación en el eje horizontal.

Luego de la rectificación, la búsqueda de correspondencias entre las imágenes es simplificada a una búsqueda horizontal sobre la misma fila. Este resultado es interesante para el cálculo de la disparidad entre características visuales, la cual permite estimar la profundidad (posición en el espacio) con respecto a la cámara en que se encuentran las marcas del ambiente observadas.

2.0.2.4. Triangulación de puntos del espacio

Dada una rectificación estéreo se puede realizar una reconstrucción 3D de los puntos presentes en las imágenes. La Figura 2.0.5 muestra la geometría involucrada durante la triangulación

de un punto P del espacio. Siendo $P = (X, Y, Z)$ un punto 3D del espacio, (x^l, y^l) y (x^r, y^r) su proyección en la cámara izquierda y derecha respectivamente y B la línea base que une las cámaras, las coordenadas de P pueden ser derivadas a través de propiedades propias de triángulos semejantes. Específicamente, las siguientes relaciones pueden ser derivadas:

$$X = \frac{x^l \cdot Z}{f} \qquad Y = \frac{y^l \cdot Z}{f} \qquad Z = \frac{B \cdot f}{x^l - x^r} \tag{2.0.8}$$

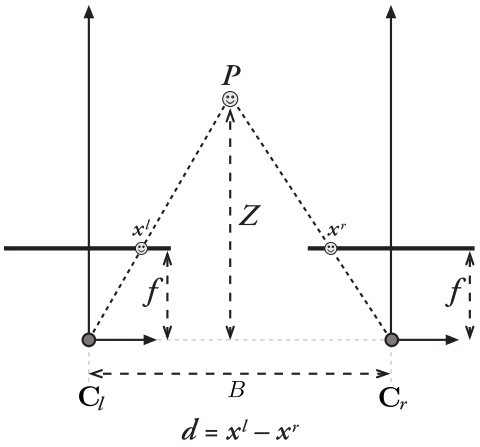


Fig. 2.0.5: Escenario de triangulación estéreo con cámaras rectificadas donde se desea calcular las coordenadas de P, en especial la profundidad Z. Los parámetros intrínsecos como la distancia focal f son conocidos, así como también los centros de cámara C_l, C_r y la línea base B.

Si bien la triangulación requiere de formulas sencillas es interesante observar que la capacidad de observar la profundidad en que se encuentra un punto del espacio esta condicionada por la resolución de las cámaras. Se define la disparidad como la distancia existente entre las proyecciones de las diferentes cámaras como $d = x^l - x^r$ y es interesante notar que la profundidad Z es inversamente proporcional a la disparidad d (Ecuación 2.0.8). Puntos del ambiente cercanos presentaran mayor disparidad en las proyecciones que puntos distantes. Cuando el valor de d es cercano a 0, pequeñas diferencias de disparidad producen un gran cambio en la profundidad percibida y cuando el valor de d es grande, pequeñas diferencias de disparidad producen pequeños cambios en la profundidad. La consecuencia de esto es que la reconstrucción 3D del ambiente utilizando cámaras estéreo es más precisa para puntos cercanos a la cámara.

2.0.3. Detección, descripción y asociación de características visuales

Los procesos de detección y descripción de características visuales (*features*) corresponden a operaciones de bajo nivel en el procesado de imágenes. Tienen como finalidad detectar marcas naturales en el entorno (*landmarks*) y describirlas de manera de poder identificarlas en otras imágenes capturadas desde otros puntos de vista o posiciones de la cámara. De esta forma, se puede abstraer las imágenes sensadas a un conjunto de descriptores. Esta técnica de discretización de imágenes es conocida como registro de imágenes (*image registration*) y es ampliamente utilizada en visión por computadoras (*computer vision*) dado que permite sustraer características visuales de interés y reducir considerablemente la magnitud de los datos con los que tratar.

Existen diversos tipos de detectores los cuales pueden dividirse en dos categorías dependiendo del tipo de características que busque en la imagen: detectores de esquinas (*corner detectors*) y detectores de regiones de interés (*region of interest detectors* o *blob detectors*). Uno de los primeros detectores de esquinas fue planteado por Moravec [7] el cual considera como puntos de interés aquellos que presentan una variación de intensidad considerable en toda dirección. Harris et al.[8] resuelve algunas de las debilidades del detector de Moravec incluyendo un análisis numérico de autovalores sobre la denominada "matriz de Harris". Posteriormente Shi y Tomasi [9] extienden este análisis de autovalores introduciendo un criterio para evaluar la calidad de las esquinas detectadas en relación a la textura observada en un área cercana. Rosten et al.[10] nombra a su método FAST, el cual utiliza técnicas de aprendizaje automático (*machine learning*) para aumentar la velocidad en la detección de esquinas. Lowe [11] introduce SIFT (*Scale Invariant Feature Transform*), donde propone utilizar un detector de regiones basado en el filtro de diferencias gaussianas (Difference of Gaussians filter), un método computacionalmente demandante pero capaz de extraer características visuales invariantes a cambios tanto de escala como de rotación e iluminación. El método propuesto por Bay et al.[12], SURF (*Speeded Up Robust Features*), inspirado en SIFT logra reducir el costo computacional a partir de utilizar imágenes integrales, manteniendo la calidad de detección y extracción de características invariantes a rotación y escala.

Asimismo, una gran variedad de métodos capaces de describir puntos y regiones fueron concebidos conforme el interés en la visión por computadoras fue creciendo. Una característica determinante de los métodos de descripción (o descriptores) es que deben estar acompañados por técnicas efectivas que permitan comparar las características visuales descriptas, de manera de poder identificar y asociar aquellas que estén presentes en distintos cuadros (*frames*). Por lo general la descripción, comparación y asociación de características es uno de los procesos más demandantes en términos de costo computacional. Métodos como SIFT y SURF permiten extraer características visuales de gran calidad, pero su extracción y comparación es muy costosa y se requieren implementaciones especificas que exploten el poder de computo de aceleradoras gráficas (Graphics Processing Unit, GPU) para satisfacer las necesidades de velocidad en sistemas de Visual SLAM. Esto se debe en medida a que las descripciones que generan son complejas y están basadas en vectores multidimensionales de valores de punto flotante.

Con el objetivo de acelerar la comparación y asociación de descriptores, investigaciones recientes se han centrado en un tipo de descriptores llamados binarios. Estos tienen la particularidad de representar características visuales como cadenas de *bits*, las cuales son comparable a través de rápidas operaciones binarias como por ejemplo la distancia Hamming. Calonder et al.[13] propone un veloz descriptor binario denominado BRIEF, el cual describe características en una única cadena de *bits*. Para esto utiliza las diferencias de intensidad entre pares de píxeles dentro de un área rectangular alrededor del punto de interés, utilizando un patrón previamente computado para la selección de los pares. Si bien la descripción de BRIEF es considerablemente veloz comparada a SIFT y SURF, este descriptor es sensible a cambios de escala y rotación, dado que genera descripciones muy disimiles de un mismo punto cuando este es rotado o escalado (por ejemplo, aumentando el zoom de la cámara). Inspirados por BRIEF, han surgido descriptores binarios invariantes a escala y rotación, como ORB [14] y BRISK [15], capaces de igualar y hasta superar en algunos casos la calidad de las características visuales extraídas por métodos más demandantes como SIFT o SURF. En las Figuras 2.0.6, 2.0.7 y 2.0.8 se pueden observar algunos ejemplos de detección y asociación de características visuales en imágenes.

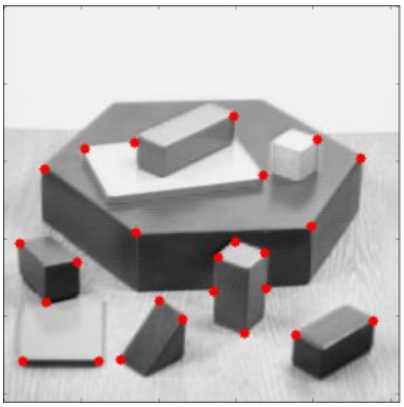


Fig. 2.0.6: Método de detección de esquinas Shi-Tomasi.

Fig. 2.0.7: Regiones de interés detectadas por el método SURF.

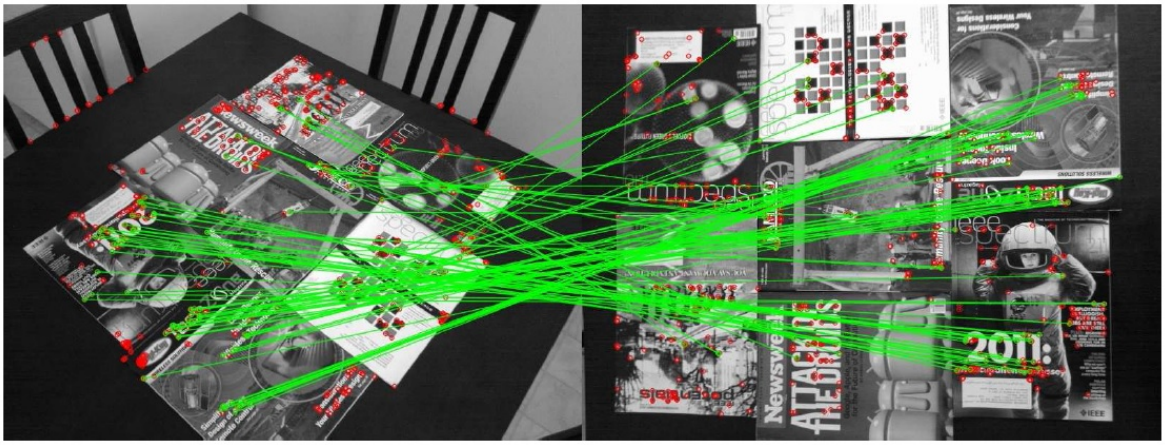


Fig. 2.0.8: Asociación de características extraídas por el método ORB. Notar la robustez del método ante la rotación de la imagen.

Capítulo 3

Visual Simultaneous Localization and Mapping

El problema de SLAM plantea el desafío de construir incrementalmente un mapa consistente de un entorno desconocido mientras se determina la posición y orientación del robot. El interés sobre este problema ha crecido en gran medida en los últimos años dado que al solucionarlo se proveen herramientas necesarias para lograr que los robots móviles puedan navegar de manera autónoma. A partir de la detección y seguimiento de marcas naturales del ambiente (*landmarks*), los sistemas SLAM estiman tanto la posición del robot como la ubicación de las marcas en el entorno. El mapa se construye con las estimaciones de las posiciones de dichas marcas, las cuales van siendo ajustadas a medida que son observadas desde distintas posiciones. Al utilizar cámaras como sensores y características visuales (*image features*) como marcas del ambiente, el problema se denomina Visual SLAM (o VSLAM). VSLAM puede abordarse utilizando diversos enfoques, los cuales son brevemente explicados a continuación.

3.1. Sistemas VSLAM probabilísticos basados en filtros

Uno de los enfoques más populares propuestos para la resolución del problema SLAM consiste en considerarlo como un caso particular del problema de filtrado (*filtering problem*). Este consiste en estimar el estado interno de un sistema dinámico cuando sólo es posible obtener observaciones parciales por medio de sensores que, a su vez, introducen perturbaciones aleatorias (ruido). Para esto se modela el problema como una probabilidad condicional de los estados posibles en un momento determinado del sistema, dadas todas las observaciones obtenidas previamente [1, 16]. Consideremos un robot móvil que se desplaza a través de un entorno desconocido realizando observaciones relativas del ambiente. Se definen entonces las siguientes variables para un instante k dado :

- x_k: El vector estado describiendo la posición y orientación (pose) del robot.
- $m = \{m_1, \ldots, m_n\}$: Conjunto de marcas del ambiente que constituyen el mapa.
- $U_{0:k} = \{u_0, \ldots, u_k\}$: El historial de entradas de control del robot, donde u_k corresponde a la entrada de control en el tiempo $k-1$ que llevo al robot al estado x_k.
- $Z_{0:k} = \{z_0, \ldots, z_k\}$: Conjunto de observaciones del ambiente hasta el momento.

De esta manera, el problema de SLAM requiere analizar la distribución de la probabilidad condicional definida como:

$$P(x_k, m|Z_{0:k}, U_{0:k}, x_0) \tag{3.1.1}$$

la cual describe la densidad conjunta de la pose del robot y de la localización de las marcas del ambiente, dadas las observaciones y entradas de control desde el momento inicial hasta el momento k. Cabe destacar que en el caso de VSLAM las observaciones del ambiente z_k responden a características visuales detectadas y descriptas en las imágenes sensadas en el tiempo k, correspondientes a proyecciones de marcas del ambiente en el plano de la imagen.

3.1.1. Filtro de Kalman Extendido

Entre los años 1950 y 1960 Rudolph Emil Kalman introdujo una técnica de filtrado y predicción de sistemas probabilísticos lineales, la cual es conocida como el Filtro de Kalman (*Kalman filter*) [16]. Si bien este filtro ha sido utilizado para la resolución de una gran variedad de problemas, no es posible aplicarlo directamente para la resolución de SLAM. Esto se debe a que SLAM tiene una naturaleza no lineal debido a los grados de libertad del robot. Esto hace que sus movimientos no puedan modelarse de manera lineal, al igual que los movimientos observables de las marcas del ambiente. Para solucionar este problema se utiliza el Filtro de Kalman Extendido (*Extended Kalman Filter*, EKF), el cual realiza una linearización previa del modelo de movimiento y sensado.

El método modela las perturbaciones introducidas por los sensores como ruido aditivo gaussiano, y utiliza una distribución normal multivariada (conocida como gaussiana multivariada) para caracterizar la probabilidad condicional definida por la ecuación 3.1.1. Siguiendo este enfoque, se utiliza la media como estimador de la pose del robot y la locación de las marcas del ambiente, y la covarianza como estimador de la incertidumbre del sistema. Estimar la correlación entre las mediciones del entorno y las estimaciones de las poses del robot resulta fundamental, y el método utiliza esta información para determinar el grado en que nuevas mediciones influirán en subsecuentes estimaciones. El método requiere, entonces, mantener la matriz de covarianza luego de cada observación del ambiente, por lo que la complejidad del método EKF crece cuadráticamente con el número de marcas sensadas del ambiente haciendo dificultosa su implementación en entornos de grandes dimensiones.

Uno de los primeros trabajos exitosos en emplear EKF para resolver el problema de VSLAM con cámaras monoculares fue Davison et al.[17], que luego se denominó MonoSLAM [18]. Si bien este trabajo logró manejar decenas de características visuales en tiempo real, sólo puede utilizarse en ambientes de pequeñas dimensiones. Paz et al.[19] utiliza cámaras estéreo para la implementación de un eficiente método de EFK-SLAM con la particularidad de que el mapa construido se divide en submapas locales de dimensión fija que permite controlar la cantidad de marcas en cada submapa, logrando operar en tiempo real en entornos de grandes dimensiones.

3.1.2. Filtro de Partículas

A diferencia del Filtro de Kalman, los métodos basados en el Filtro de Partículas son no paramétricos, es decir, no asumen ninguna distribución probabilística para caracterizar la probabilidad condicional del estado y el mapa (3.1.1). El Filtro de Partícula (*Particle Filter*), también conocido como método de Monte Carlo, aproxima dicha distribución a partir de

generar sucesivas hipótesis a cerca de las poses del robot y de las marcas en el entorno. A estas hipótesis se las representa como un conjunto de partículas a las cuales se les asigna un grado de verosimilitud. Y luego de sensar el entorno, se actualiza el grado de verosimilitud de cada partícula dependiendo de que tan consistente sea con respecto a las observaciones obtenidas. Partículas improbables son descartadas para dar lugar a nuevos muestreos y la generación de hipótesis más probables. Si bien la complejidad de este tipo de métodos es lineal con respecto al número de partículas, su precisión aumenta a mayor cantidad de particulas. Dada la dimensionalidad del espacio de posibilidades (considerando pose del robot y de las marcas), una implementación *naive* resulta inviable debido a la cantidad de partículas requeridas.

Montemerlo et al.[20] presentan FastSLAM, uno de los primeros trabajos en aplicar el Filtro de Partículas de manera exitosa en VSLAM. Basándose en el teorema de Rao-Blackwell y tomando el historial de todas las poses del robot en el ambiente (trayectoria), se particiona la distribución conjunta de manera de estimar trayectoria del robot y ubicación de las marcas por separado. La distribución sobre la trayectoria del robot es representada como un conjunto de partículas. Al mismo tiempo, considerando la trayectoria del robot fija, las poses de las marcas del ambiente son consideradas probabilitiscamente independientes respecto de la trayectoria y sus poses son estimadas utilizando el Filtro de Kalman Extendido de complejidad lineal. Este tipo de filtro se lo llama *Rao-Blackwellized Particle Filter* (RBPF) y tiene una complejidad lineal sobre el número de partículas. Existen otros métodos basados en el Filtro de Partículas [21, 22] que logran resolver el problema de VSLAM en tiempo real para entornos de pequeñas dimensiones.

3.2. Sistemas VSLAM basados en pose-graph optimization

La soluciones basadas en EKF emplean modelos linearizados los cuales pierden precisión progresivamente, dada la naturaleza no lineal del problema. Esto conduce a la inevitable divergencia en las estimaciones tanto de la pose del robot como del mapa construido. Por otro lado, métodos no paramétricos basados en el Filtro de Partículas requieren un número de partículas cada vez mayor conforme el entorno explorado crece, excediendo eventualmente la capacidad del sistema para operar en tiempo real.

Como alternativa a los enfoque basados en filtros surgieron métodos para resolver VSLAM inspirados en sistemas de visión por computadora. En *Structure from Motion* (SfM) el objetivo es lograr una reconstrucción tridimensional del entorno a través del análisis de imágenes tomadas desde distintas perspectivas y donde no se requiere operar en tiempo real. Para resolver SfM se suele emplear la técnica de *Bundle Adjustment [23]* que consiste en refinar la ubicación de las marcas visuales que describen la geometría del ambiente, los parámetros del movimiento relativo de la cámara y las matrices de proyección de las cámaras empleadas para adquirir las imágenes. Para esto se utiliza un criterio de optimalidad con respecto a las proyecciones de las marcas visuales sobre los planos imagen de las cámaras involucradas. El problema de VSLAM puede abordarse entonces como un problema de SfM donde todas las imágenes son capturadas por una misma cámara y donde se agrega el requerimiento de tiempo real.

Consideremos la situación donde un conjunto de marcas visuales representadas por puntos 3D del mundo, X_j, son observadas por un conjunto de cámaras con matrices de proyección P^i, denotando x^i_j como la coordenada 2D correspondiente a la proyección de la j-ésima marca visual al ser observada por la i-ésima cámara. En la práctica $x^i_j \approx P^i X_j$, teniendo en cuenta las perturbaciones introducidas al observar el ambiente. En este caso, se busca un estimador de

máxima verosimilitud que permita, asumiendo que el ruido introducido es Gaussiano, estimar matrices de proyección $\widehat{P}^i$ y puntos 3D $\widehat{X}_j$ tal que sus proyecciones 2D $\widehat{x}^i_j$ cumplan $\widehat{x}^i_j = \widehat{P}^i \widehat{X}_j$. Este estimador minimiza la distancia entre los puntos detectados (observados) en la imagen x^i_j y las correspondientes proyecciones de los puntos 3D $\widehat{x}^i_j$, denominado error de re-proyección:

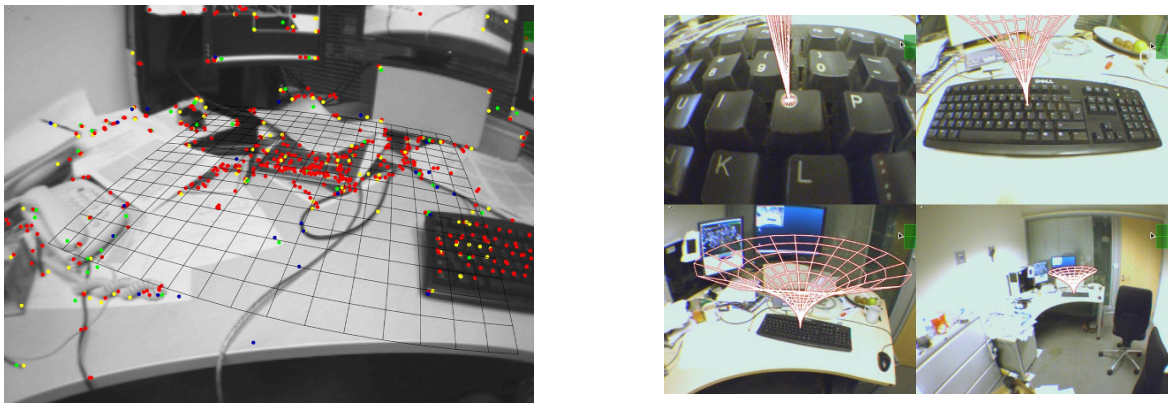
$$\underset{\widehat{P}^i, \widehat{X}_j}{\operatorname{argmin}} \sum_{ij} a_{ij} d(\widehat{P}^i \widehat{X}_j, x^i_j)^2 \tag{3.2.1}$$

donde $d(x, y)$ es la distancia geométrica en la imagen entre los puntos x e y, y a_{ij} denota una variable binaria tal que es igual a 1 si el j-ésimo punto es visible por la i-ésima cámara y 0 en cualquier otro caso. Esta minimización se resuelve utilizando algoritmos de Cuadrados Mínimos no lineales (*nonlinear Least Square*). Dentro de este tipo de técnicas una de las más exitosas es el método de minimización Levenberg–Marquardt [24, 25].

Originariamente por su alto costo computacional las técnicas de optimización y ajuste como *Bundle Adjustment* eran aplicadas como un proceso último de refinamiento en métodos fotométricos, fue Dellaert et al.[26] quien las introdujo en la comunidad de robótica móvil para abordar el problema de VSLAM. Dellaert hace un profundo análisis de la naturaleza rala del problema y explota diferentes técnicas de factorización numérica para alcanzar una solución que presenta mejores resultados en relación al tamaño del entorno, en comparación a métodos basados en filtros.

Posteriormente Klein y Murray [27] proponen PTAM (*Parallel Tracking and Mapping*), un sistema diseñado originariamente para estimar la posición de una cámara en ambientes pequeños para aplicaciones de realidad aumentada (Fig.(a) 3.2.1). Los autores proponen dividir el proceso en dos tareas distintas: el seguimiento (*tracking*) de la cámara por un lado, y el mapeo (*mapping*) de las marcas visuales del entorno por otro. De esta manera se plantea el problema de forma de aprovechar unidades de procesamiento paralelo (*multi-core*). El hilo de ejecución del *tracking* se encarga de localizar la cámara, de manera rápida y robusta, en referencia al mapa de marcas visuales (llamados puntos de mapa o *map points*) construido hasta el momento. La localización se lleva a cabo en cada cuadro (*frame*) obtenido de la cámara y requiere el agregado de nuevas marcas visuales al mapa sólo cuando se encuentra en áreas aún no explorados del ambiente (Fig.(b) 3.2.1). El hilo del *mapping* responde al requerimiento tomando sólo ciertos cuadros clave (*keyframes*) para la construcción de un mapa basado en un grafo de poses de cámara y puntos 3D. En forma paralela, este mapa es a su vez continuamente optimizado utilizando la técnica de *Bundle Adjustment*. El sistema PTAM es capaz de localizar la cámara en el espacio, conforme optimiza la ubicación de cientos de marcas visuales en tiempo real.

Konolige y Agrawal [28] enfocan su trabajo en tratar el problema de VSLAM en entornos de grandes dimensiones. Los autores introducen FrameSLAM, un sistema que utiliza odometría visual para determinar el movimiento de una cámara estéreo mientras se construye un grafo *frame-frame*. A diferencia del mapa construido por PTAM, este grafo mantiene sólo las relaciones entre *frames* reduciendo considerablemente el tiempo requerido para la optimización del mapa, logrando localizarse en trayectorias de 10km o más en tiempo real. Strasdat et al. [29] proponen utilizar una ventana doble de optimización (*double window optimization*), donde una ventana interna se encarga de utilizar las restricciones entre *keyframes* y puntos del mapa mientras que una ventana externa utiliza restricciones solo entre *keyframes*. De esta manera, mientras la ventana interna sirve para modelar el área local de la forma más precisa posible, la ventana exterior sirve para estabilizar la periferia del mapa. Strasdat et al.[30] demuestran



(a) (b)

Fig. 3.2.1: Desempeño del método PTAM (a) Seguimiento de características visuales. (b) Manteniendo el seguimiento de la cámara en una aplicación de realidad aumentada.

que enfoques basados en PTAM ofrecen mejor precisión en comparación con métodos basados en EKF. Más aún, los autores señalan que para ambas técnicas aumentar el numero de marcas visuales detectadas es la mejor manera de aumentar la precisión de los métodos, mientras que aumentar la cantidad de *keyframes* tiene un impacto principal sobre la robustez (por ejemplo, en la perdida de la localización de la cámara). Recientemente Mur-Artal et al.[31] presentan ORB-SLAM, el cual construye sobre las ideas introducidas por PTAM añadiendo técnicas del estado del arte. Como resultado presentan un sistema VSLAM monocular robusto, capaz de trabajar en largas trayectorias en tiempo real. ORB-SLAM realiza detección de ciclos [32] y utiliza un mapa de co-visibilidad, conjunto a otras técnicas propuestas en [33, 29], para superar la acumulación de error y lograr operar en ambientes de grandes dimensiones.

3.3. S-PTAM

En [3] se presenta un sistema SLAM basado en visión estéreo, capaz de estimar la localización de un robot móvil en tiempo real en trayectorias de gran longitud. S-PTAM se inspira en PTAM [27] aprovechando la capacidad de cómputo de unidades de procesamiento paralelo, dividiendo el problema al igual que PTAM en dos tareas principales: seguimiento de la cámara y la construcción del mapa.

Al inicio del sistema, S-PTAM considera la pose de la cámara en el centro de coordenadas del mundo, y un mapa que será inicializado a partir de la triangulación de las características visuales presentes en el primer par de imágenes obtenidas de la cámara estéreo. Desde ese momento, el hilo del *Tracking* estima la pose actual de la cámara por cada nuevo par de imágenes, minimizando el error de re-proyección entre las características visuales en las imágenes y puntos existentes en el mapa. Ocasionalmente, algunos *frames* son estratégicamente seleccionados de manera de aumentar el tamaño del mapa, a través de la triangulación de las características visuales extraídas. A estos *frames* seleccionados se los denomina *keyframes* y proveen restricciones espaciales que son utilizadas para el refinamiento de los puntos del mapa. El hilo del *Local Mapping* se encarga de optimizar las poses de los *keyframes* y puntos del mapa recientemente agregados. Esta optimización tiene como objetivo minimizar el error de

re-proyección de manera de incrementar la precisión del mapa.

En la Figura 3.3.1 se pueden observar los distintos componentes que constituyen S-PTAM. La Figura 3.3.2 muestra un escenario de ejemplo, donde se utiliza S-PTAM para la estimación de la trayectoria.

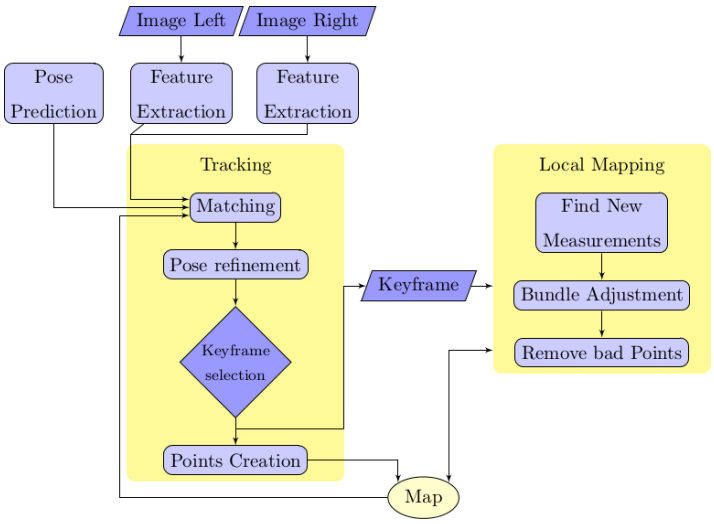


Fig. 3.3.1: Esquema general de los diferentes componentes de S-PTAM

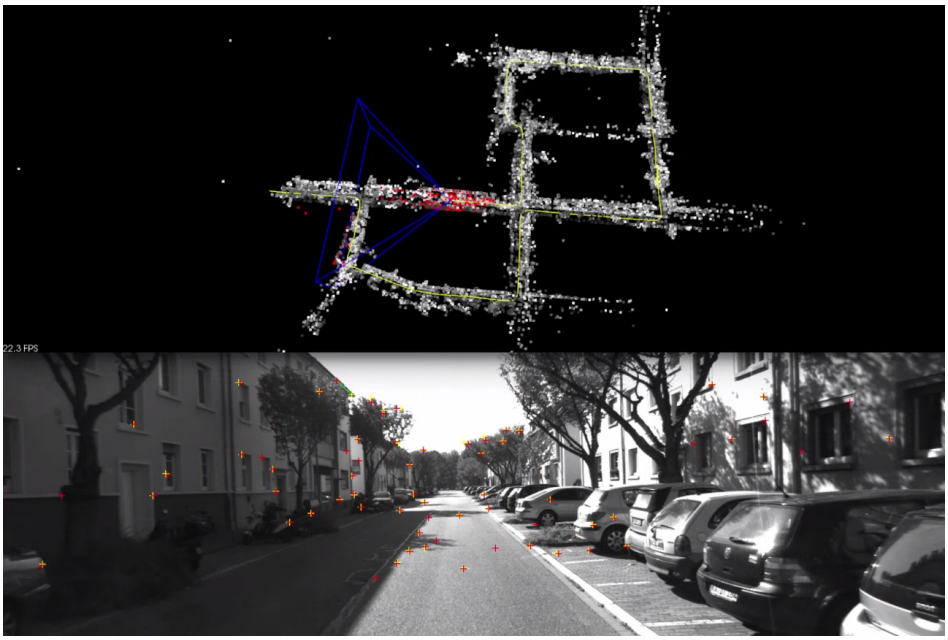


Fig. 3.3.2: Reconstrucción del ambiente (mapa) y trayectoria estimada por S-PTAM durante el procesamiento de una secuencia de imágenes obtenida del recorrido de un vehículo.

A continuación se resume el funcionamiento del sistema S-PTAM, exponiendo los principales detalles de cada componente.

3.3.1. Tracking

El módulo *Tracking* se encarga de realizar el seguimiento de la cámara en referencia al mapa hasta el momento construido. Su costo computacional es reducido de manera de operar en tiempo real.

3.3.1.1. Extracción de características visuales

Todo par de imágenes provisto por la cámara estéreo es procesado de manera de extraer características visuales. El sistema permite la utilización de cualquier detector y descriptor de *features*, con preferencia por aquellos de menor costo computacional que permitan operar en tiempo real como los descriptores binarios. La información extraída de las imágenes es utilizada para asociar características visuales presentes en las imágenes con los puntos presentes en el mapa (*Matching*).

3.3.1.2. Proyección y asociación de puntos del mapa

Por cada nuevo *frame* se busca determinar los puntos del mapa que se encuentran dentro del campo visual de la cámara de manera de facilitar la búsqueda de correspondencias entre puntos del mapa y características visuales extraídas de las imágenes. Para esto, primero se realiza una predicción de la pose de la cámara utilizando información de *dead reckoning* externa (como por ejemplo, odometría basada en *encoders* de un robot con ruedas) o un modelo de movimiento pre-establecido (como por ejemplo, considerando un modelo de velocidad lineal o decreciente). El *Pose Predictor* es el encargado de realizar esta predicción y proveerla al hilo del *Tracking*. La pose predicha es utilizada para filtrar puntos del mapa dentro del campo visual y proyectarlos sobre el plano imagen de la cámara. Los descriptores de las características visuales que se encuentren en la vecindad de una proyección, son comparados con el descriptor asociado al punto del mapa correspondiente. Se busca que los descriptores sean similares, y que la diferencia entre el ángulo en que se observa al punto del mapa desde la nueva pose y el ángulo en el que fue creado sea menor a un cierto *threshold*.

Las correspondencias establecidas entre las características visuales y los puntos del mapa, son utilizadas para refinar la pose estimada de la cámara. Este refinamiento de pose corresponde a la minimización del error de re-proyección utilizando el método de minimización Levenberg–Marquardt [24, 25] (*Pose Refinement*).

3.3.1.3. Selección y procesamiento de *keyframes*

Se considera necesario incluir un nuevo *keyframe* al mapa cuando la cantidad de asociaciones establecidas entre la cámara y el mapa se ve reducida. Esto ocurre cuando la cantidad de correspondencias establecidas entre el *frame* actual y los puntos del mapa es menor al 90 % de la cantidad de correspondencias establecidas en el *frame* anterior.

Los *keyframes* seleccionados son procesados de manera de establecer las correspondencias entre el par de imágenes del *frame* estéreo (Sección 2.0.2.1). Por medio de la geometría epipolar se triangulan aquellos puntos para los cuales no se hubieran encontrado correspondencias con el mapa (Sección 2.0.2.4) y son incluidos al mapa (*Points Creation*). De esta manera, al considerar que se podría estar perdiendo referencia del mapa, se aumenta el tamaño del mapa con un nuevo *keyframe*. Finalmente, este *keyframe* es almacenado para ser procesado por el *Local Mapping*.

3.3.2. Local Mapping

Se extiende el enfoque presentado en el método monocular PTAM [27], agregando restricciones propias del modelo y geometría de las cámaras estéreo (Sección 2.0.2). El módulo se encarga de ajustar las poses de cámara de los *keyframes* pertenecientes al mapa de manera continua. Con cada nuevo *keyframe* incluido al mapa, se establece un conjunto de *keyframes* vecinos que compartan puntos visibles entre sí. Considerando únicamente esta vecindad de *keyframes*, se ajustan las poses de cámara y puntos del mapa utilizando métodos de optimización que resuelvan *Bundle Adjustment*. Esta optimización minimiza los errores de re-proyección entre los puntos del mapa y las características visuales (*features*) presentes en los planos imagen de los *keyframes* considerados. A este tipo de optimización, sobre un subconjunto de *keyframes* y puntos, se lo denomina *Local Bundle Adjustment*. Durante el proceso de minimización podrían existir puntos que presenten errores de re-proyección considerados como *outliers*. Estos afectan negativamente a la solución y para reducir su impacto son descartados (*Remove bad Points*).

Capítulo 4

Detección de ciclos en la trayectoria

El reconocimiento de lugares previamente visitados es esencial en la robótica móvil dado que provee de información valiosa respecto a la relación entre las estimaciones llevadas a cabo por los sistemas de localización. Tomar conocimiento de que se ha efectuado un ciclo en la trayectoria permite calcular el error cometido en la estimación de la posición y da origen a una serie de procesos que permiten corregir tanto la localización actual del robot como el mapa hasta ese momento construido. A lo largo del tiempo han surgido varios enfoques para dar solución al problema de la detección de ciclos, y esto se debe en parte a que las técnicas concebidas, por lo general, trabajan con información provista por los sistemas SLAM subyacentes. De esta manera, la evolución de los métodos SLAM fue acompañado por un continuo desarrollo de técnicas de detección de ciclos.

Es posible interpretar y tratar el problema de la detección de ciclos como un problema de asociación de información (*data association*), en el que el objetivo es lograr relacionar datos provenientes de una misma fuente. Para el caso de SLAM el objetivo es entonces asociar diferentes mediciones provenientes de una misma marca del ambiente. De no tener éxito en esta asociación, se podría considerar una medición como proveniente de una marca aún no observada cuando en realidad ya había sido observada. Esto afecta la calidad de las estimaciones y hasta podría generar inconsistencias críticas en el modelo sobre el cual se esta trabajando. Por dar un ejemplo, en el caso de sistemas EKF-SLAM es de suma importancia mantener actualizada la correlación entre las observaciones de las marcas del ambiente. Para esto se deben relacionar dichas observaciones eficientemente a lo largo del tiempo ya que una matriz de covarianza inconsistente, que no refleje fehacientemente la correlación entre los datos, impide el cálculo de la incertidumbre y degrada progresivamente el desempeño del sistema.

4.1. Métodos métricos para la detección de ciclos

Estos métodos trabajan sobre información espacial estimada por los métodos SLAM. En base a estimaciones de la ubicación del robot y la distribución geométrica de las marcas del ambiente, se busca identificar mediciones provenientes de una misma marca. Inspiradas por métodos de seguimiento de objetivos (*target tracking*) aplicados en radares [34, 35], las primeras soluciones propuestas consideraban la asociación entre observaciones y marcas del ambiente de manera individual, comprobando si la observación de una determinada marca se encontraba cerca de la ubicación predicha. La estabilidad y precisión de este tipo de métodos se ve comprometida por el error acumulado en las estimaciones y la incertidumbre sobre la locación

del robot. Neira y Tardós [36] introdujeron el método JCBB (*Joint Compatibility Branch and Bound*), en donde se define un criterio de compatibilidad donde múltiples observaciones son consideradas al mismo tiempo, tomando en cuenta de manera explicita la correlación entre las mismas. El método se extiende en [37] obteniendo buenos resultados utilizando sensores láser. Luego, Clemente et al.[38] implementan el método en un sistema VSLAM basado en EKF utilizando una cámara monocular logrando cerrar ciclos en trayectorias de grandes dimensiones aunque aún no en tiempo real. Baley et al.[39] utilizan técnicas de clusterización para la generación de grafos que modelen las relaciones de distancia relativa y angular entre marcas del ambiente, de manera de poder detectar sub-grafos contenidos que representen distribuciones de marcas antes observadas. Todos estos métodos resultan efectivos a corto y mediano plazo, donde la asociación de observaciones es requerida para mantener la consistencia de los métodos SLAM, pero se vuelven demasiado complejos e inestables al tratar con entornos de grandes dimensiones donde el error acumulado en las estimaciones es considerable.

4.2. Métodos basados en apariencia para la detección de ciclos

Al utilizar cámaras como sensores es posible obtener, a través de la detección y descripción de características visuales, información valiosa sobre la apariencia de lo que se esté observando. Esta información de apariencia es independiente de las estimaciones geométricas llevadas a cabo por los sistemas SLAM, y por lo tanto no se ve afectada por el error acumulado. El problema a resolver reside, en cambio, en encontrar un método efectivo que permita interpretar el entorno observado de manera de reconocer la apariencia de lugares que hubieran sido observados anteriormente.

Williams et al.[40, 41] se basa en un método de reconocimiento de objetos introducido por Lepetit y Fua [42], el cual es capaz de detectar características visuales recurrentes en sucesivos cuadros de vídeo. Se interpreta el problema de la detección de ciclos como un problema de clasificación, donde un clasificador es entrenado de manera que cada clase se construye con todas las posibles apariencias de una característica en particular. El clasificador se implementa sobre un árbol de decisión que permite consultar a que clase pertenece una determinada característica visual. En cada nivel del árbol se realizan diferentes comprobaciones logrando una clasificación veloz y eficiente (Fig. 4.2.1). Williams utiliza una GPU para realizar un entrenamiento "online" del clasificador con 400 versiones distorsionadas de cada nueva característica visual detectada en cada cuadro obtenido de la cámara. Se comprueba la re-ocurrencia de características visuales en sucesivos cuadros y a través de un sistema de votación, se analizan las posibles hipótesis de manera de determinar si las características visuales detectadas determinan en su conjunto la apariencia de un lugar previamente observado. El método se implementa en un hilo de ejecución paralelo usando una versión mejorada del clasificador de Lepetit y Fua implementada sobre listas de decisión, logrando operar en tiempo real sobre un sistema basado en EKF-SLAM. Si bien el sistema logra detectar ciclos de manera efectiva, este requiere de unidades de procesamiento gráfico (GPU) y su escalabilidad es limitada dado que el consumo de memoria es muy elevado al construir las clases de apariencia de cada característica visual.

Por otro lado, surgieron varios métodos basados en una técnica denominada *Bag of Words*, la cual fue aplicada originalmente para el análisis de texto. Sivic y Zisserman [43] fueron quienes introdujeron dicha técnica en el análisis de video para el reconocimiento de objetos. En su trabajo, proponen discretizar el espacio de descriptores visuales entrenando un vocabulario

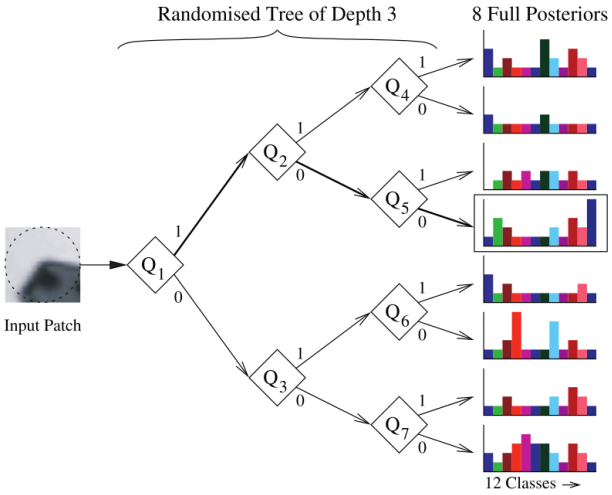
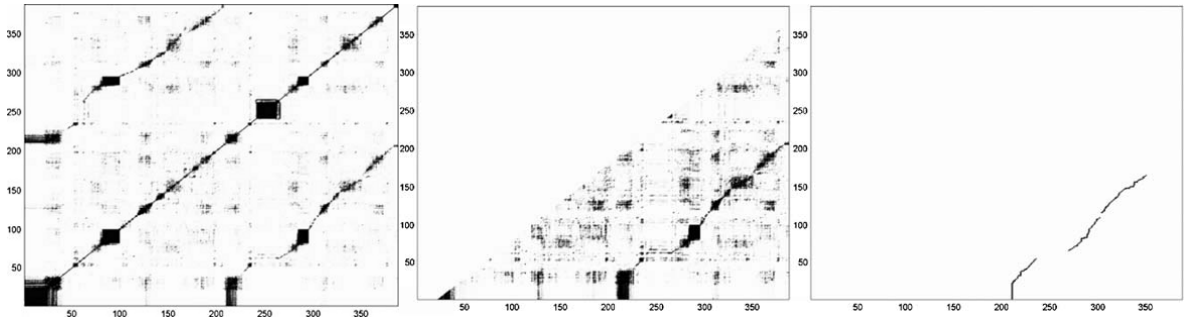


Fig. 4.2.1: Árbol clasificador entrenado donde cada Q_i representa una comprobación binaria sobre una característica visual. Cada hoja tiene asociada una distribución correspondiente a la probabilidad de que dicha característica pertenezca a alguna de las clases. Fuente: [40]

visual de palabras (*words*), el cual representa una división en *clusters* de todos los posibles descriptores. Por cada *cluster*, un único descriptor visual es considerado como su representante, y a su vez, todo descriptor representante es considerado una palabra (*word*). Para lograr esta discretización se utilizan métodos de clusterización de datos que permitan agrupar, por similitud, descriptores extraídos de una secuencia de imágenes de entrenamiento. El vocabulario visual resultante del entrenamiento es utilizado para interpretar una imagen como el conjunto de palabras que representen a los descriptores visuales presentes en la misma. Este conjunto de palabras es implementado como un vector de palabras conocido en la literatura como vector *bag-of-words*. A cada palabra del vocabulario visual se le asigna un "peso" en relación a la frecuencia con que son observadas características visuales que corresponden a dicha palabra. De manera que palabras más frecuentes son consideradas menos discriminativas, y por consiguiente los descriptores visuales a los que representan son considerados, a su vez, menos discriminativos. Utilizando esta idea, es posible caracterizar la similitud entre dos imágenes como una relación entre los pesos asignados a las palabras que las componen.

Ho y Newman [44] hacen un profundo análisis de este método aplicado para la detección de ciclos. En su trabajo se define la matriz de similaridad visual (*visual similarity matrix*) M donde cada elemento M_{ij} es el valor de similitud $s(I_i, I_j)$ entre las imágenes I_i e I_j pertenecientes a la secuencia de imágenes $I = [I_1, I_2, \ldots]$ obtenidas por la cámara. Sobre esta matriz de similaridad se analiza la relación entre los ciclos de la trayectoria y la aparición de subdiagonales de altos valores de similitud en la matriz M (Figs. 4.2.2). En el trabajo se obtienen además interesantes resultados e interpretaciones del impacto de características visuales muy recurrentes provenientes de, por ejemplo, arbustos o formas geométricas presentes en casas o edificios. Basados en este trabajo, Cummins y Newman [45, 46] desarrollan FAB-MAP, que hasta la fecha permanece como uno de los métodos más confiables para la detección de ciclos y análisis de apariencia. El mismo plantea un modelo probabilístico de complejidad lineal donde se toma en cuenta la dependencia entre las palabras encontradas en las imágenes. Se pondera el grado de distinción que aporta la aparición de una palabra en la imagen, en relación a

cuales otras palabras aparecen junto a ésta. Así, palabras que comúnmente no se las encuentre juntas se las considera más discriminativas. El método procesa una secuencia de imágenes de entrenamiento para la creación de un vocabulario visual junto con un árbol de Chow Liu [47], este último se utiliza para aproximar la relación de dependencia entre las palabras.



(a) (b) (c)

Fig. 4.2.2: (a) Matriz M de similitud entre imágenes, elementos M_{ij} representan la similitud entre las imágenes I_i e I_j. Elementos de alta similitud se representan en un tono oscuro mientras que los de baja similitud en tono claro. Notar que M es simétrica. (b) Triangular inferior de M excluyendo, además, elementos cercanos a la diagonal. (c) Resultado de aplicar un algoritmo propuesto en [44] sobre la matriz M, en este, se acentúan subdiagonales de altos valores de similitud. Fuente: [44]

FAB-MAP fue implementado utilizando SURF para la detección y descripción de características visuales, el cual es computacionalmente demandante, por lo que para trabajar en tiempo real se requiere el uso de procesadores gráficos (GPU). Esto inspiró la introducción de detectores de ciclos capaces de operar con descriptores binarios de rápida procesamiento. Galvez-Lopez y Tardós [48, 32] introducen por primera vez un detector de ciclos basado en descriptores binarios desarrollado utilizando la biblioteca DBoW, de su propia autoría, para el manejo de vectores *bag-of-words*. Este método, a diferencia de FAB-MAP, no utiliza un modelo probabilístico para tomar en cuenta la similitud de lugares previamente visitados. En vez de esto, introducen el concepto de consistencia temporal (*temporal consistency*) de manera que grupos de imágenes tomadas en momentos cercanos se consideran de manera conjunta aumentando la confiabilidad de la detección. Al utilizar descriptores binarios el computo de vectores *bag-of-words* es muy veloz, así como también la comparación entre estos. El resultado es un sistema capaz de detectar ciclos en la trayectoria realizada, en tiempo real considerando toda imagen tomada por la cámara. Mur-Artal y Tardós [49] utilizan posteriormente la librería DBoW para la implementación de un sistema capaz de detectar ciclos a nivel de solo algunos cuadros clave (*keyframes*) provistos por la cámara. En el trabajo se utiliza el descriptor binario ORB el cual provee una buena calidad de asociación ante cambios de escala y definen un método de verificación de ciclos utilizando el mapa provisto por un sistema VSLAM basado en PTAM [27]. Este sistema logra tanto el cierre de ciclos de manera consistente como la relocalización de la cámara luego de fallas o pérdida en la localización y es la base del sistema ORB-SLAM [31] desarrollado por los mismos autores posteriormente.

La biblioteca DBoW resulta de gran interés para la solución final presentada en este trabajo, por lo que es revisada en detalle en la sección siguiente.

4.3. Análisis de apariencia y detección de ciclos utilizando *bag-of-words*

Galvez-Lopez y Tardós introducen DBoW [48, 32], una biblioteca completa para el análisis de apariencia a través de vocabularios visuales y comparaciones entre vectores *bag-of-words* extraídos de las imágenes. La misma se basa en los trabajos de Sivic y Zisserman [43] y Nister y Stewenius [50]. Utilizando estas técnicas los autores proponen un eficiente método de detección de ciclos introduciendo la evaluación de distintos criterios de consistencia.

4.3.1. Independencia del descriptor

Los modelos de evaluación de apariencia son independientes del tipo de descriptor con el que se trabaje. Las técnicas involucradas requieren que los descriptores de características visuales puedan ser comparados bajo algún criterio de distancia. Por lo general, para descriptores binarios se utiliza la norma Hamming mientras que para descriptores vectoriales se utiliza la norma 2 correspondiente a la distancia Euclídea. En el caso de los descriptores binarios, se requiere además una metodología que permita establecer el elemento medio o centroide de un conjunto de descriptores. Para esto se utiliza una técnica denominada *k-majority* [51] que define un esquema de votación entre descriptores binarios para la determinación de centroides.

Cabe destacar que si bien las técnicas introducidas son independientes de los descriptores, el desempeño del detector de ciclos no lo será. Descriptores invariantes a escala y rotación serán capaces de comparar la apariencia entre locaciones previamente visitadas aún cuando estas fueran recorridas desde diferentes puntos de vista.

4.3.2. Vocabulario visual

El vocabulario visual consta de una estructura jerárquica arbolada donde sus nodos están compuestos por descriptores, los cuales representan a un conjunto de descriptores cercanos. Para construirlo, se discretiza el espacio de descriptores en un vocabulario compacto de W palabras. Se extraen características visuales de un conjunto de imágenes de entrenamiento diferente e independiente del entorno donde se utilizará luego el sistema. A través de una clusterización de k-medias (*k-means clustering*) utilizando el método k-means++ [52], los descriptores asociados a las características visuales son repartidos en k_w clusters donde un único descriptor es elegido como representante y centro de cada cluster. Estos clusters forman el primer nivel de nodos del vocabulario visual y cada uno será dividido, a su vez, en k_w nuevos clusters de manera recursiva, hasta L_w veces. k_w representa, entonces, un factor de ramificación que define el número de nuevas ramas a generar en cada subsecuente nivel del vocabulario visual. Al final del proceso se obtiene un vocabulario visual donde los nodos de cada nivel contienen un descriptor el cual es el centro de un cluster a un determinado nivel de subdivisión del espacio de descriptores. Se consideran como palabras las hojas del vocabulario visual, por lo que el método es capaz de representar hasta $k_w^{L_w}$ palabras solo determinadas por la distribución de los descriptores de entrenamiento (Fig. 4.3.1).

Al momento de crear el vocabulario visual se le asigna a cada palabra un peso (*weight*) de acuerdo a su relevancia en la secuencia de entrenamiento. Mientras más frecuente sea una palabra (menos discriminativa), más bajo el peso que se le asigna. Sea w_i una palabra del vocabulario, se asigna durante el entrenamiento un peso correspondiente a su frecuencia de documento inversa (*inverse document frequency* o *idf*):

$$idf(i) = log(\frac{N}{n_i}) \tag{4.3.1}$$

donde N es el número total de imágenes de entrenamiento y n_i el número total de ocurrencias de la palabra w_i en dichas imágenes.

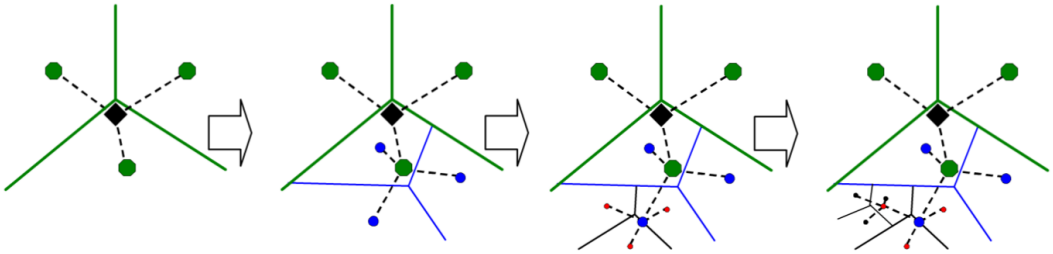


Fig. 4.3.1: Construcción del vocabulario visual, los nodos representan centros de cluster a cada nivel de subdivisión. Fuente: [50]

4.3.3. Conversión de imágenes a bag-of-words

Para convertir una imagen I_t, tomada en el tiempo t, en un vector *bag-of-words*, sus descriptores deben recorrer el vocabulario visual desde la raíz hasta las hojas. En cada nivel se elige el nodo siguiente de forma que la distancia entre el descriptor asociado al nodo y el descriptor obtenido de la imagen, sea mínima. De esta manera al llegar a una hoja del vocabulario visual, se obtiene por cada descriptor, una palabra que lo representa junto a un peso asignado en la etapa de entrenamiento. Habiendo identificado las palabras presentes en I_t, es posible medir además la relevancia descriptiva que estas tengan en la apariencia de I_t. Para eso se calcula la frecuencia de termino (*term frecuency* o *tf*) para cada palabra w_i presente en I_t:

$$tf(i, I_t) = \frac{n_{iI_t}}{n_{I_t}} \tag{4.3.2}$$

donde n_{iI_t}representa el número de ocurrencias de la palabra w_i en la imagen I_t y n_{I_t} el número total de palabras encontradas en I_t. Teniendo esta información resulta beneficioso definir vectores *bag-of-words* para evaluar la similitud entre dos imágenes, en relación al factor *tf-idf* [43]. Es decir, para una imagen I_t se construye el vector *bag-of-words* $v_t \in \mathbb{R}^W$ tal que

$$(v_t)_i = tf(i, I_t) \times idf(i) \tag{4.3.3}$$

para todo $i \in [1, \ldots, W]$ donde el factor $idf(i)$ es calculado anteriormente en la etapa de entrenamiento del vocabulario visual.

4.3.4. Base de datos de imágenes

Para lograr la detección de lugares visitados previamente es necesario que al momento de analizar una nueva imagen, sea posible obtener todas aquellas imágenes anteriores que compartan algún grado de similitud con esta. Para esto se construye una base de datos que permita ser consultada eficientemente. Esta base de datos esta compuesta por el vocabulario visual y los índices directos e indirectos que vinculan imágenes con palabras (Fig. 4.3.2),

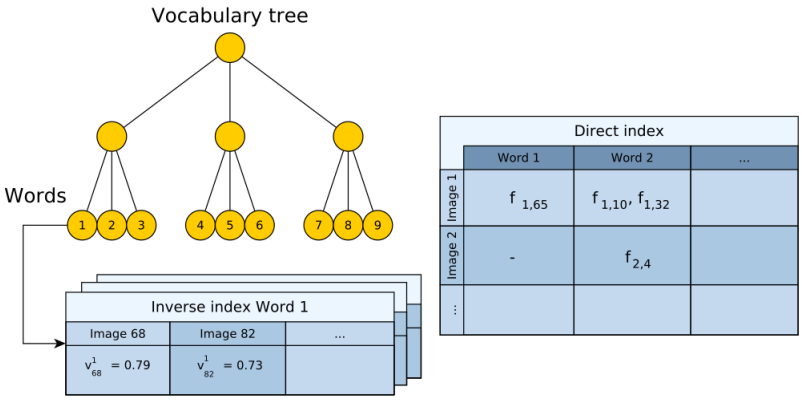


Fig. 4.3.2: Estructuras que componen la base de datos de imágenes. En el ejemplo el vocabulario visual posee $L_w = 2$ niveles y un factor de ramificación $k_w = 3$, formando 9 diferentes palabras en las hojas. El índice inverso vincula cada palabra con toda imagen en la que estuvo presente junto con el peso *tf-idf* asociado. El índice directo vincula cada característica visual presente en las imágenes con la palabra a la cual pertenece. Fuente: [48, 32]

permitiendo la búsqueda eficiente de imágenes que posean una determinada palabra y, de manera inversa, recuperar todas las palabras presentes en una imagen.

Las imágenes recibidas se agregan a la base de datos de forma progresiva utilizando su representación en vectores *bag-of-words* conformados por los pesos *tf-idf*. Los índices se actualizan de manera acorde permitiendo la comparación entre imágenes de manera eficiente. La similitud entre imágenes se mide comparando sus vectores *bag-of-words*.

Sean v_1 y v_2 vectores *bag-of-words,* se evalúa la similitud entre éstos calculando la evaluación L_1 (L_1-*score*) propuesta por [50]:

$$s(v_1, v_2) = 1 - \frac{1}{2}\left|\frac{v_1}{|v_1|} - \frac{v_2}{|v_2|}\right| \tag{4.3.4}$$

donde $s(v_1, v_2) \in [0 \dots 1]$. De esta manera se utiliza el vocabulario visual para obtener el vector *bag-of-words* asociado a cada nueva imagen recibida. Este vector es luego comparado contra el vector asociado a toda otra imagen anterior, presente en la base de datos, que comparta alguna palabra en común. Habiendo obtenido un conjunto de posibles candidatos a ciclos junto con sus respectivas valoraciones de similitud, se agrega la nueva imagen a la base de datos y se procede a un análisis de consistencia que permita determinar si el sitio en el que se encuentra el robot había sido previamente visitado.

4.3.5. Detección de ciclos y análisis de consistencia

Las estructuras y métodos anteriormente expuestos tienen como fin proveer de herramientas suficientes para poder comparar imágenes de manera eficiente. Sin embargo, discriminar en qué casos pudiera haberse cometido un ciclo en la trayectoria requiere un análisis más profundo. Por dar un ejemplo, un robot que transite un pasillo visualmente recurrente genera sucesivas imágenes muy similares entre sí pero no debería considerarse esto como un ciclo en la trayectoria. Analizar la relación temporal entre candidatos de alta similitud permite discernir entre este tipo de situaciones y verdaderos ciclos en la trayectoria.

Cuando una nueva imagen I_t es adquirida, se obtiene su vector *bag-of-words* v_t asociado y se lo utiliza para consultar la base de datos. De este proceso se obtiene una lista de candidatos a ciclos $< v_t, v_{t_1} >, < v_t, v_{t_2} >, \ldots, < v_t, v_{t_c} >$ junto con sus respectivos valores de similitud $s(v_t, v_{t_j})$, con $1 \leq j \leq C$. El valor de similitud entre los posibles candidatos y v_t varía dependiendo de la distribución de las palabras en v_t, por lo que la similitud entre imágenes se normaliza con el valor de similitud entre las imágenes I_t y $I_{t-\Delta t}$ correspondiente a la anterior:

$$\eta(v_t, v_{t_j}) = \frac{s(v_t, v_{t_j})}{s(v_t, v_{t-\Delta t})} \tag{4.3.5}$$

donde $v_{t-\Delta t}$ corresponde al vector *bag-of-words* de la imagen anterior a I_t y $s(v_t, v_{t-\Delta t})$ representa el grado de similitud entre I_t y $I_{t-\Delta t}$. Aquellos candidatos tal que $\eta(v_t, v_{t_j})$ no supere un *threshold* α definido son rechazados. Además, podría ocurrir que el valor de $s(v_t, v_{t-\Delta t})$ sea pequeño (por ejemplo, cuando el robot gira rápidamente) generando, erróneamente, que $\eta(v_t, v_{t_j})$ sea muy alto. Para solucionar esto se rechaza la existencia de ciclos si el valor de $s(v_t, v_{t-\Delta t})$ no llegará un mínimo establecido.

Por otro lado, de manera de prevenir que imágenes cercanas en tiempo compitan entre ellas, se las agrupa en "islas" y son tratadas como un único candidato. Es decir, se utiliza T_i para representar un intervalo de tiempo comprendido por $t_{n_i}, \ldots, t_{m_i}$y V_{T_i} para islas que agrupen vectores *bag-of-words* asociados $v_{t_{n_i}}, \ldots, v_{t_{m_i}}$. De esta manera, varios candidatos $< v_t, v_{t_{n_i}} >, \ldots, < v_t, v_{t_{m_i}} >$ son interpretados como un único candidato $< v_t, V_{T_i} >$, si el intervalo de tiempo entre ellos es suficientemente chico. Estas islas son entonces valoradas de manera conjunta:

$$H(v_t, V_{T_i}) = \sum_{j=n_i}^{m_i} \eta(v_t, v_{t_j}) \tag{4.3.6}$$

La isla de mayor similitud acumulada H es seleccionada como el mejor grupo candidato al cual se lo nota como $V_{T'}$. Sobre este grupo se aplica una restricción temporal: $< v_t, V_{T'} >$ debe ser consistente con k detecciones positivas anteriores $< v_{t-\Delta t}, V_{T_1} >, \ldots, < v_{t-k\Delta t}, V_{T_k} >$ donde los intervalos T_i y T_{i+1} son cercanos. Si $V_{T'}$ cumple esta restricción temporal, se considera como mejor y único candidato a ciclo al par $< v_t, v_{t'} >$ con $t' \in T'$ y $v_{t'} \in V_{T'}$ es el vector *bag-of-words* de máximo valor η dentro de $V_{T'}$.

4.3.6. Validación geométrica de la detección

Con los métodos anteriormente descriptos existe la posibilidad de falsos positivos en la detección de ciclos en ambientes visualmente recurrentes. Para lidiar con esto, es posible llevar a cabo un análisis de consistencia geométrica luego de haber detectado un ciclo entre un par de imágenes I_t e $I_{t'}$. Esta validación geométrica no esta relacionada con el mapa construido hasta el momento, sino que depende únicamente de la distribución de las características visuales presentes en las imágenes. El concepto general se basa en que si las imágenes I_t e $I_{t'}$ provienen del mismo lugar y muestran un alto grado de similitud, entonces debería ser posible obtener una matriz fundamental (vista en la Sección 2.0.2) consistente que las relacione.

Los métodos numéricos para el cálculo de la matriz fundamental requieren el establecimiento de correspondencias entre las características visuales de las imágenes, en este caso, de las presentes en la imagen I_t con aquellas presentes en la imagen $I_{t'}$. Es posible realizar una búsqueda lineal para obtener las correspondencias, donde toda característica visual en la

imagen I_t se compara contra toda característica visual en la imagen $I_{t'}$ asociando aquellas que tengan distancia mínima. Este tipo de búsqueda lineal para la asociación de características visuales es costosa, por lo que se aprovecha el índice directo presente en la base de datos de imágenes (Fig. 4.3.2) para acceder a las palabras contenidas de cada imagen y comparar características visuales solo entre aquellas representadas por una misma palabra, es decir, pertenecientes al mismo cluster del vocabulario visual. De esta forma, se acelera la búsqueda de correspondencias comparando las características visuales entre I_t e $I_{t'}$ que pertenezcan a la misma palabra. Es posible relajar el método extendiendo la comparación entre características visuales pertenecientes a palabras (clusters) cercanas.

Obtenidas las correspondencias entre las imágenes, se procede con el cálculo de la matriz fundamental entre ellas utilizando RANSAC [53]. El esquema RANSAC consiste en comprobar la consistencia de sucesivas matrices fundamentales calculadas en base a una selección aleatoria de correspondencias. Si el proceso es capaz de hallar una matriz fundamental consistente con al menos 12 correspondencias entre I_t e $I_{t'}$, entonces el ciclo detectado es validado y considerado positivo.

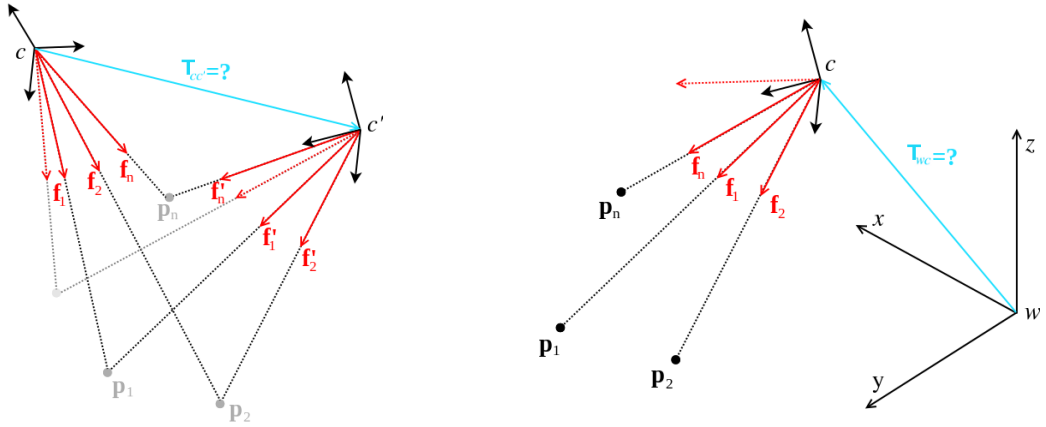
La utilización de este tipo de validación geométrica se aplica para el caso monocular. En el caso estéreo es posible realizar una validación geométrica aprovechando que se puede obtener para cada par de imágenes capturadas la profundidad a las características vistuales detectadas, como veremos en el Capítulo 5.

Capítulo 5

Cálculo de la transformación entre cámaras estéreo

En el contexto de un robot móvil localizándose por medio de un sistema VSLAM estéreo, calcular la transformación (rotación y traslación) entre dos estimaciones de localización es trivial dado que conlleva simples operaciones numéricas. Al hacerlo de esta manera, la transformación calculada acarrea el error de estimación acumulado. Es por esto que al detectar y validar un ciclo entre dos momentos de la trayectoria se requieren métodos capaces de estimar la transformación entre las cámaras utilizando información independiente a los errores de estimación cometidos. Esto es, utilizando la calibración de las cámaras involucradas en el ciclo, las imágenes capturadas en los distintos momentos de la trayectoria y las características visuales detectadas. El cálculo de esta transformación es de interés dado que permite estimar el error cometido hasta el momento y servirá como punto de partida para los métodos que permitan corregir tanto la localización actual como el mapa construído hasta el momento.

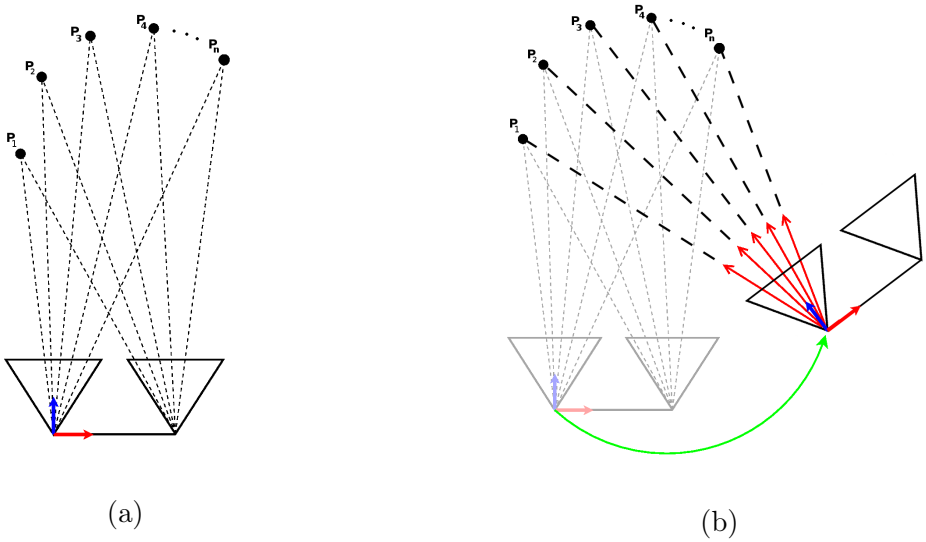
Es posible distinguir dos tipos distintos de métodos geométricos para la estimación de la pose de una cámara: los relativos (*relative pose estimation*) y los absolutos (*absolute pose estimation*). Estos se diferencian por el tipo de información que utilizan para llevar a cabo la estimación. Los métodos relativos calculan la pose de una cámara con respecto a otra utilizando "correspondencias 2D-2D" las cuales refieren a pares asociados entre características visuales extraídas de las imágenes (Fig.(a) 5.0.1). Los métodos absolutos calculan la pose de una cámara con respecto a un eje de coordenadas determinado (usualmente el eje de coordenadas del mundo) utilizando "correspondencias 2D-3D" las cuales refieren a pares asociados entre características visuales presentes en las imágenes y puntos 3D en referencia al eje de coordenadas antes mencionado (Fig.(b) 5.0.1). Estos métodos de estimación absoluta de la pose se los conoce además como métodos de Perspectiva por n Puntos (PnP, *perspective-n-points*) y resultan más precisos que los métodos relativos. Esto se debe a que los métodos relativos no poseen información de profundidad y deben trabajar solo con vectores de origen en el centro de la cámara y en dirección a las características visuales extraídas, a lo cual se le llama vectores dirección (*bearing vectors*).



(a) (b)

Fig. 5.0.1: (a) Escenario de estimación de pose de la cámara c' relativa a la cámara c utilizando correspondencias 2D-2D representadas como vectores dirección $\mathbf{f}_i$. (b) Escenario de estimación de pose absoluta de la cámara c con respecto al eje de coordenadas w utilizando correspondencias 2D-3D representadas como vectores dirección $\mathbf{f_i}$ y coordenadas 3D $\mathbf{p_i}$.

Es posible interpretar, entonces, el escenario planteado anteriormente como un problema de estimación relativa de la pose, dado que al detectar un ciclo interesa estimar la pose de la cámara actual en relación a la cámara anterior con la cual ocurrió el ciclo en la trayectoria. Sin embargo, al trabajar con cámaras estéreo siempre es posible recuperar la distancia a la que se encuentran las características visuales por medio de triangulación (Sección 2.0.2.4) y gracias a esto es posible utilizar métodos de Perspectiva por n Puntos para calcular la transformación entre las cámaras estéreo de manera precisa (Figs. 5.0.2).



(a) (b)

Fig. 5.0.2: (a) Par estéreo del cual se triangulan puntos de control en referencia a la cámara izquierda. (b) Utilizando correspondencias 2D-3D (flechas rojas) entre características visuales en la imagen y puntos de control en referencia al par estéreo anterior, es posible calcular la transformación relativa (flecha verde) existente entre las cámaras izquierdas de los dos pares estéreo.

5.1. Métodos para la resolución del problema PnP

El problema de Perspectiva por n Puntos (PnP) consiste en determinar la posición y orientación de la cámara en relación a un determinado eje de coordenadas a través de un número arbitrario n de correspondencias 2D-3D, llamados "puntos de control" (Fig.(b) 5.0.1). El problema tuvo origen como parte del problema de calibración de una cámara (*camera calibration*), en el cual el objetivo es calcular los parámetros intrínsecos de una cámara a partir de imágenes de escenas controladas con objetos de dimensiones conocidas.

The direct linear transformation fue el primer método desarrollado por fotogrametristas utilizado como solución al problema PnP [54, 5]. Actualmente se lo sigue usando por su simpleza y como base de comparación para métodos más complejos. La estabilidad del método se ve comprometida en el caso de que los puntos de control sean coplanares (pertenezcan todos al mismo plano) y su precisión decrece drásticamente en presencia de ruido en las observaciones. Los métodos iterativos presentan soluciones al problema PnP de gran precisión [55], pero su efectividad y velocidad de convergencia esta condicionada por una estimación inicial de la solución (*initial guess*). Lepetit et al.[56] es el primer método no iterativo de complejidad lineal para la resolución del problema. El método denominado *Efficient Perspective-n-Point* (*EPnP*) toma los puntos de control en referencia a un eje de coordenadas determinado y busca calcular las coordenadas de dichos puntos en referencia a la cámara, obteniendo dos conjuntos de coordenadas en referencia a distintos ejes que representen a los puntos de control. Se utilizan luego métodos numéricos conocidos para obtener la orientación y traslación que permitan alinear ambos conjuntos de coordenadas [57], y de esta manera obtener la posición de la cámara en referencia al eje de coordenadas externo. El método resulta eficiente y estable ante ruido pero requiere de una variante especial para el caso donde los puntos de control son coplanares y puede fallar en casos donde existan múltiples soluciones por la cantidad y distribución espacial de los puntos de control (por ejemplo, en el caso minimal de 3 puntos de control). Kneip et al.[58] introduce el método UPnP (*Unified PnP*) de complejidad lineal, capaz de manejar aquellos casos conflictivos de múltiples soluciones y garantizar de manera teórica un criterio de optimalidad geométrica en las soluciones encontradas. En la investigación presentada UPnP es comparado de manera exhaustiva contra varios otros métodos del estado del arte, obteniendo buenos resultados en términos de velocidad de cómputo, estabilidad y precisión de las soluciones.

5.1.1. P3P: El caso minimal

Este es un caso particular del problema PnP donde $n = 3$ y es además el subconjunto más pequeño de puntos de control requeridos para obtener una cantidad finita de soluciones al problema (Fig. 5.1.1). Para una cámara calibrada (se conocen sus parámetros intrínsecos) pueden existir hasta cuatro soluciones al problema, requiriendo de una cuarta correspondencia para restringir y desambiguar el problema a una única solución.

Gao et al.[59] presentan un completo análisis del problema en el que clasifican por primera vez las condiciones necesarias para que existan múltiples soluciones. En el trabajo se plantea un sistema de ecuaciones algebraicas en el que se expresa la posición y orientación de la cámara por medio de relaciones trigonométricas entre el centro de la cámara, los puntos 3D de control y los ángulos formados entre estos. A este sistema se integran condiciones de realidad (*reality conditions*) de manera de que las soluciones obtenidas sean soluciones físicas posibles en el mundo real. Una condición de realidad importante es que los tres puntos de control y

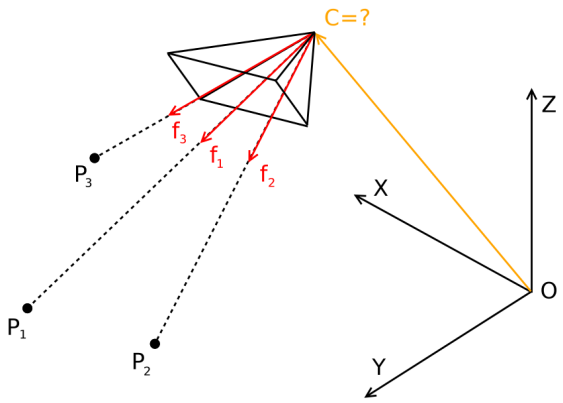


Fig. 5.1.1: Escenario de estimación de pose de una cámara a partir de 3 correspondencias 2D-3D

el centro de la cámara deben ser no coplanares, es decir, no deben pertenecer al mismo plano dado que en caso contrario los puntos estarían alineados desde la perspectiva de la cámara lo cual no provee suficiente información, degradando a infinitas posibles soluciones al sistema de ecuaciones planteado. Kneip et al.[60] propone una novedosa parametrización del problema permitiendo una resolución más directa, una alta eficiencia y estabilidad en la resolución. El método es comparado con el de Gao en escenarios simulados en presencia de ruido gaussiano obteniendo mayor robustez y velocidad de computo.

5.1.2. Importancia del esquema RANSAC

La efectividad de los métodos PnP esta condicionada a la calidad de las correspondencias entre características visuales extraídas de las imágenes. Idealmente se desea trabajar en espacios que mantengan su apariencia a lo largo del tiempo, de manera que los algoritmos de detección y descripción de características visuales sean capaces de encontrar marcas estacionarias en el ambiente. Si bien los métodos PnP son capaces de proveer algún grado de robustez ante el ruido en las observaciones, en entornos reales existe la posibilidad de que características extraídas de las imágenes sean completamente inconsistentes. Cambios en la iluminación, oclusiones u objetos dinámicos pueden generar características visuales efímeras las cuales dificultan el proceso de asociación de descriptores y conformación de correspondencias entre imágenes. La utilización de métodos PnP sobre correspondencias que contengan asociaciones inconsistentes o erróneas puede dar como resultado estimaciones de poses inconsistentes y hasta errores numéricos que imposibiliten la terminación de los métodos antes mencionados.

En el caso de cámaras estéreo la correspondencia de características visuales entre cámaras previamente calibradas permite hacer asociaciones robustas asegurando que las restricciones epipolares sean satisfechas (Sección 2.0.2). Esto aumenta la confianza en la precisión de los métodos de triangulación para calcular los puntos de control requeridos por los algoritmos PnP. Sin embargo, al momento de la detección de un ciclo, no se tiene información que relacione geométricamente las cámaras estéreo involucradas. Por lo tanto, la calidad de asociación de correspondencias entre las características visuales detectadas dependen únicamente de la expresividad de los descriptores extraídos en cada una de las cámaras.

El esquema RANSAC introducido por Fischler y Bolles [53] define una técnica de aprendizaje para estimar los parámetros de un modelo de forma iterativa a través de la selección

aleatoria de observaciones. Dado un conjunto de datos permite diferenciar aquellos que se ajusten a un modelo (*inliers*) de aquellos espurios que no lo hagan (*outliers*). En el caso de la estimación de poses permite usar el caso minimal P3P para la selección de un conjunto consistente de correspondencias. Se eligen de manera aleatoria 4 (para desambiguar a una única solución) correspondencias 2D-3D para estimar la pose a través de algún método P3P, la consistencia de las correspondencias restantes se evalúa proyectando los puntos 3D al plano imagen de la cámara utilizando la pose estimada. Aquellos puntos 3D que proyecten lo suficientemente cerca de sus respectivas correspondencias 2D serán considerados como *inliers*. El proceso se repite una cantidad predeterminada de veces de manera de encontrar una estimación consistente con un número considerable *inliers*. Luego de haber establecido un conjunto de correspondencias 2D-3D consistentes, se utilizan métodos generales PnP de manera de obtener estimaciones de la pose más precisas.

Capítulo 6

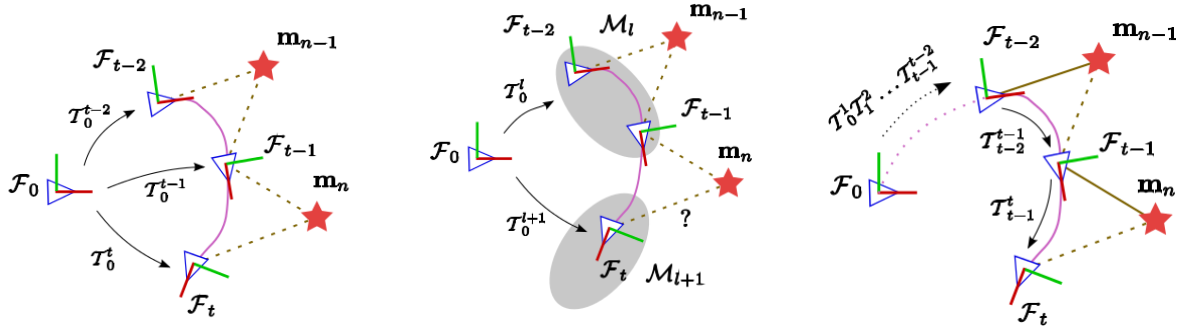
Cierre del ciclo en la trayectoria

Detectar la ocurrencia de un ciclo en la trayectoria permite evaluar el error acumulado en las estimaciones de localización. En base a esto es posible efectuar correcciones al mapa construido del entorno por el sistema SLAM. En el caso de sistemas VSLAM, al mejorar la fidelidad del mapa, aumenta la cantidad y calidad de los puntos del mapa que se emplean para realizar el seguimiento en la etapa de *Tracking* (Sección 3.3.1.2).

Dada la información de asociación entre mediciones de un sistema VSLAM, el problema de cierre de ciclos puede interpretarse como un caso particular de *Bundle Adjustment*. La diferencia radica en que al cerrar un ciclo, las propias estimaciones de localización y mapa del sistema SLAM representan una solución aproximada al problema de minimización. De esta manera, se desean utilizar las poses de cámara y ubicación de las marcas estimadas de manera eficiente. Realizar una optimización de todo el mapa usando todas las mediciones obtenidas por el sistema SLAM a lo largo de la trayectoria no es viable en aplicaciones de tiempo real.

Es interesante notar que los métodos de cierre de ciclos están directamente relacionados con el tipo de mapa que construye el sistema SLAM. Clemente et al.[38] introducen un sistema EKF-VSLAM que construye un mapa jerárquico de dos niveles (Fig.(b) 6.0.1). Un nivel global contiene información que relaciona submapas de dimensión limitada, mientras que cada submapa a su vez mantiene la relación espacial entre las estimaciones de locación y la ubicación de marcas próximas del ambiente. En este escenario, el trabajo utiliza un método denominado *Iterated Extended Kalman Filter* (IEFK) [61] de manera de re-estimar la relación espacial entre submapas al momento de detectar un ciclo. Mei et al.[62] utiliza una representación continua relativa (*continuous relative representation*, CRR), la cual resulta muy novedosa dado que no utiliza un eje de coordenadas de referencia global (Fig.(c) 6.0.1). Cada marca del ambiente esta en referencia a una única estimación de localización, y a su vez, las estimaciones de localización mantienen relaciones relativas de la transformación entre ellas. De esta forma, al detectar un ciclo entre estimaciones de localización, se requiere únicamente relacionarlas a través de la transformación relativa entre las cámaras involucradas (Sección 5). Esto simplifica muchas operaciones y procedimientos a costa de dificultar la obtención de un mapa globalmente consistente. Mur-Artal y Tardós [49, 31], inspirados por PTAM [27], utilizan un mapa conformado por *keyframes* y puntos 3D (*map points*). Los *keyframes* corresponden a una selección de las estimaciones de localización, e igual que los puntos del mapa, se encuentran en referencia a un eje de coordenadas global (Fig.(a) 6.0.1). Cada *keyframe*, además, esta relacionado con todo punto del mapa que haya observado y de esta manera se tiene un mapa globalmente consistente. Al cerrar un ciclo en la trayectoria, el sistema utiliza técnicas

de *Bundle Adjustment*. Las poses de cámara son ajustadas optimizando un grafo conformado solo por los *keyframes*, excluyendo los puntos del mapa. La optimización "une" los *keyframes* involucrados en el ciclo y ajusta las poses de cámara de los *keyframes* restantes de manera de mantener las relaciones de distancia existentes antes del cierre. Los autores denominan al grafo optimizado como grafo Esencial (*Essential graph*) y tiene la particularidad de que dos *keyframes* se relacionan entre sí, solo si estos comparten puntos de mapa observados. De esta manera, al optimizarlo, se mantienen relaciones de vecindad entre los *keyframes*. En una etapa posterior, la posición de los puntos del mapa se corrige de acuerdo al desplazamiento de los *keyframes* luego de la optimización. Esto es, cada punto del mapa se corrige aplicando la misma transformación que hubiera ajustado al *keyframe* que lo trianguló.



(a) Mapa global

(b) Mapa compuesto de submapas

(c) Mapa relativo continuo

Fig. 6.0.1: Diferentes tipos de mapas donde $\mathbf{m_i}$ representa las marcas del ambiente, $\mathcal{F}_i$ representa estimaciones de localización, $\mathcal{T}_i^j$ transformaciones relativa entre elementos del mapa y $\mathcal{M}_i$ a sub-mapas donde aplicable. Fuente: [62]

Capítulo 7

Método propuesto de detección y cierre de ciclos

A continuación se expone en detalle la solución propuesta para el problema de detección y cierre de ciclos. El método fue concebido para integrarse con el sistema S-PTAM (*Stereo Parallel Tracking and Mapping*)[3], de forma que las diferentes técnicas involucradas en el desarrollo están motivadas por las características propias del sistema de SLAM estéreo considerado. El uso eficiente de la información disponible es un requerimiento del método, minimizando el impacto en el desempeño del sistema SLAM en su conjunto.

7.1. Esquema general

Se extiende el funcionamiento de S-PTAM añadiendo un módulo de detección y cierre de ciclos denominado *Loop Closing*. Este módulo trabaja de manera paralela en un hilo de ejecución distinto a los del *Tracking* y *Local Mapping*. Los *keyframes* procesados por el *Local Mapping* son encolados, luego, para ser tratados por el hilo de *Loop Closing* (Fig. 7.1.1). *Loop Closing* procesa los *keyframes* de forma secuencial, realizando la detección, validación y cierre de ciclos. En caso de validar la ocurrencia de un ciclo, el módulo considera la sincronización entre los hilos de ejecución del sistema respetando la consistencia de las estructuras de datos concurrentes utilizadas. La actualización del mapa se realiza sin entorpecer el funcionamiento general del sistema S-PTAM. En caso de que se requiera corregir la trayectoria actual en curso, el módulo se comunica con el *Pose Predictor* de manera de notificar los cambios en la localización actual del sistema.

El módulo *Loop Closing* (al igual que S-PTAM) está diseñado para operar con cualquier tipo de descriptor de características visuales que se desee utilizar. Se requiere únicamente la construcción previa de un vocabulario visual para la etapa de detección de ciclos (Sección 4.3.2). No se realizan procesamientos adicionales sobre las imágenes de los *keyframes*, de manera que los métodos involucrados utilizan la información procesada previamente por el sistema S-PTAM (Sección 3.3.1.3).

7.2. Detección de ciclos y construcción de la base de datos

Siguiendo las técnicas introducidas en la Sección 4.3, se entrena un vocabulario visual utilizando una colección de imágenes seleccionadas. Este entrenamiento se lleva acabo en una

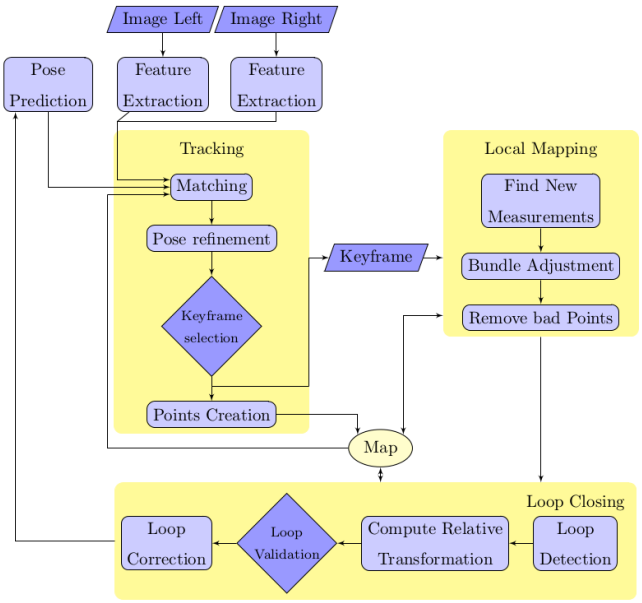


Fig. 7.1.1: Esquema general del sistema S-PTAM incluyendo detección y cierre de ciclos.

etapa previa al funcionamiento del sistema, cuidando que las imágenes utilizadas representen, en términos de apariencia, entornos diversos. Es importante utilizar tanto imágenes de ambientes interiores (*indoor*) como exteriores (*outdoor*) y evitar largas secuencias de imágenes pertenecientes a un mismo entorno. De esta forma, se logra un análisis de frecuencias de características visuales representativo de trayectorias posibles en entornos reales.

La base de datos se construye utilizando la información de apariencia proveniente de los sucesivos *keyframes* que se agreguen al mapa. A medida que los *keyframes* son procesados, estos se interpretan como vectores *bag-of-words* utilizando el vocabulario visual. Para esto se utilizan las características visuales detectadas en la etapa del *Tracking*, de manera que no es necesaria la utilización de métodos de descripción al momento de la detección de ciclos. La base de datos es consultada utilizando los vectores *bag-of-words* siguiendo los métodos de análisis de similitud expuestos en la Sección 4.3.5. Se establece un *threshold* mínimo permisivo de $\alpha = 0,3$ sobre el valor de similitud normalizado η, considerando como candidatos a ciclos *keyframes* pertenecientes a la base de datos que superen dicho *threshold*. Los candidatos a ciclos serán agrupados y evaluados a través de la valoración de similitud conjunta H, obteniendo finalmente un único candidato a ciclo. Dado que la generación de nuevos *keyframes* se realiza de manera espaciada debido a la selección que se lleva a cabo en el hilo del *Tracking* (Sección 3.3.1.3), no se impone una restricción temporal de sucesivas detecciones positivas. Esto es así dado que podría no existir una relación de continuidad temporal entre *keyframes* generados.

En caso de que la detección de ciclos en la trayectoria sea positiva, se la caracteriza como un par de *keyframes* K_c, K_ℓ pertenecientes al mapa. K_c referencia al *keyframe* recientemente procesado que originó la consulta y K_ℓ al mejor candidato a ciclo obtenido por el método de detección anteriormente descripto.

7.3. Validación del ciclo y cálculo de la transformación relativa

Habiendo establecido un par de *keyframes* K_c, K_ℓ como candidato a ciclo en la trayectoria, se desea realizar un análisis geométrico que permita validar la ocurrencia del ciclo. Concretamente, se analiza si es posible establecer una transformación relativa entre las poses de cámara de K_c, K_ℓ utilizando técnicas de Perspectiva por n Puntos (Capítulo 5).

Para llevar a cabo esta validación, primero es necesario comparar y asociar los descriptores extraídos de K_c y K_ℓ de manera de establecer correspondencias entre las características visuales detectadas en las imágenes. Sobre esta correspondencia entre descriptores se utiliza un método P3P (Sección 5.1.1) bajo un esquema RANSAC de manera de determinar una transformación entre los *keyframes* K_c, K_ℓ consistente con una cantidad considerable de *inliers* (Sección 5.1.2).

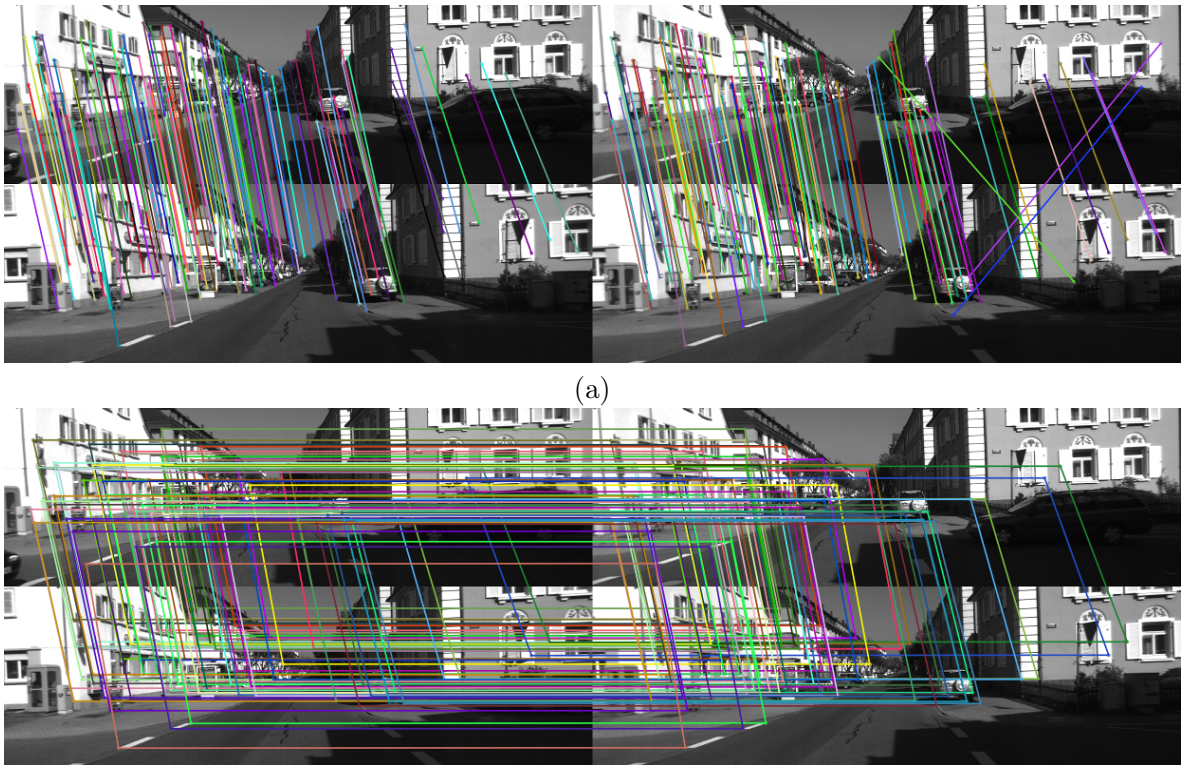
En caso de que al menos un 80 % del total de correspondencias sean encontradas *inliers*, el ciclo entre los *keyframes* K_c, K_ℓ se considera válido. Finalmente se estima la transformación relativa $T_{c\ell}$ entre las poses de cámara de los *keyframes* K_c, K_ℓ utilizando un método PnP sobre el conjunto de correspondencias *inliers*.

7.3.1. Correspondencias robustas entre cámaras estéreo

Dado el procesamiento sobre los *keyframes,* llevado a cabo durante *Tracking* (Sección 3.3.1.3), las características visuales entre cámaras de un mismo par estéreo se encuentran asociadas satisfaciendo las restricciones intrínsecas de la geometría epipolar. Sin embargo, al momento de la detección del ciclo entre K_c, K_ℓ no existe información que relacione las características visuales presentes entre los dos pares estéreo.

Para lograr una robusta correspondencia entre las 4 cámaras involucradas (2 por cada cámara estéreo), se realizan dos procesos de comparación y asociación, uno entre las cámaras "izquierdas" y otro entre las cámaras "derechas". Se lleva a cabo una comparación de "fuerza bruta" donde para cada descriptor extraído de una imagen, se evalúa la distancia a todo descriptor de la segunda imagen. Relacionar descriptores de mínima distancia entre imágenes da buenos resultados en términos de asociación, pero podrían cometerse errores al considerar descriptores de alguna manera "ambiguos". Es decir, si para un descriptor existiesen múltiples candidatos que presenten una distancia pequeña y similar, elegir el de mínima distancia podría no ser la asociación correcta. Para solventar esto, se consideran los dos descriptores de menor distancia, y se evalúa la relación del segundo vecino cercano (*second-closest neighbour ratio*) propuesta en [11]. Esto permite evaluar la ambigüedad de un descriptor al momento de realizar la comparación y asociación entre imágenes, de forma de descartar asociaciones que podrían no ser correctas.

Por ultimo, dadas las correspondencias entre cámaras "izquierdas" y "derechas" de diferentes pares estéreo, se analiza si estas asociaciones cumplen con las restricciones epipolares y las correspondencias establecidas entre cámaras de un mismo par estéreo (Fig. 7.3.1). Este proceso determina correspondencias robustas entre cámaras estéreo, y en la práctica, si no se logra establecer un mínimo de correspondencias entre los *keyframes* K_c, K_ℓ, el ciclo es descartado y declarado como inválido.



(a)

(b)

Fig. 7.3.1: Correspondencias encontradas entre cámaras estéreo pertenecientes a un ciclo detectado en la trayectoria. El par de imágenes superior corresponde al momento en la trayectoria cuando se detecto el ciclo, el par inferior corresponde al mejor candidato a ciclo evaluado. (a) Asociación entre imágenes "izquierdas" y "derechas" de ambos pares. (b) Asociación luego de aplicar las restricciones epipolares. Notar que se descartan asociaciones erróneas realizadas entre las imágenes "derechas".

7.4. Cierre del ciclo y corrección de la trayectoria

El mapa construido por el sistema S-PTAM es globalmente consistente, es decir, los puntos del mapa y las poses de cámara de los *keyframes* se encuentran en relación a un único eje de coordenadas global. Por esta razón se requiere que el proceso de cierre del ciclo y ajuste de la trayectoria, mantenga la consistencia del mapa.

Al validar un ciclo entre los *keyframes* K_c y K_ℓ, se considera a la transformación $T_{c\ell}$ como una estimación del error acumulado hasta el momento. El error en la trayectoria tiende a incrementarse con el tiempo, por lo que K_ℓ porta menor error de estimación que cualquier otro *keyframe* generado posteriormente. Con esto en mente, se lleva a cabo una corrección inicial propagando la transformación $T_{c\ell}$ por los *keyframes* comprendidos entre K_ℓ y K_c.

Sean $\{E_{wc}, \ldots, E_{w(j+1)}, E_{wj}, \ldots, E_{w\ell}\}$ las poses de cámara asociadas a los *keyframes* pertenecientes al ciclo detectado, la propagación se define como:

$$
\begin{aligned}
E_{wc}^{prop} &= E_{w\ell}(T_{c\ell})^{-1} = E_{w\ell}T_{\ell c} \\
E_{wj}^{prop} &= Interpolar_j(E_{wj}, E_{w(j+1)}^{prop}E_{(j+1)j}) \\
E_{w\ell}^{prop} &= E_{w\ell},
\end{aligned}
$$

donde el índice superior "*prop*" refiere a las poses de cámara luego de la propagación, $T_{\ell c}$ corresponde a la transformación inversa de $T_{c\ell}$ previamente estimada y $E_{(j+1)j}$ es la transformación entre las poses $E_{w(j+1)}$ y E_{wj} ($E_{(j+1)j} = (E_{w(j+1)})^{-1}E_{wj} = E_{(j+1)w}E_{wj}$). $Interpolar_j(*,*)$ realiza una interpolación entre poses de manera de "suavizar" la propagación efectuada sobre el *keyframe* K_j, de acuerdo a su distancia al ciclo detectado. De esta manera, *keyframes* cercanos a K_c (y por tanto cercanos al punto donde fue detectado el ciclo) serán fuertemente corregidos, mientras que la corrección aplicada a *keyframes* alejados del punto de cierre de ciclo es suavizada. Este suavizado se logra utilizando una representación en *quaternions* de las rotaciones y el método de interpolación linear esférica (SLERP*, Spherical Linear Interpolation*) [63]. Esta corrección inicial logra cerrar el ciclo "uniendo" a K_c y K_ℓ a través de $T_{c\ell}$, propagando una corrección suavizada a lo largo de todos los *keyframes* involucrados en el ciclo.

Utilizando esta propagación como solución inicial, se lleva a cabo la optimización del grafo de poses formado por todos los *keyframes* del mapa. Este grafo se construye relacionando cada *keyframe* con aquel que haya sido generado posteriormente a este. Entre *keyframes* relacionados se establece como "restricción" la transformación relativa entre estos. Se obtiene entonces, una cadena de *keyframes* donde dos *keyframes* subsiguientes se conectan por una única transformación. A esta cadena se le añaden, además, las transformaciones relativas estimadas correspondientes a todos los ciclos detectados. Así, se busca estimar las poses de cámara que mejor satisfagan las restricciones de transformación establecidas entre los *keyframes.*

En términos de minimización, se define el error residual entre poses de cámara con respecto a las restricciones de transformación. De esta manera, a través de algoritmos de minimización, se obtienen las poses de cámara que minimicen el error acumulado de todos los errores residuales. Sea $E_{i(i+1)}$, $\forall i \in [0, \mathcal{C}\text{-}1]$, la transformación entre la pose de cámara E_{wi} y la subsecuente pose $E_{w(i+1)}$. Se define como el error residual entre *keyframes* subsiguientes como $r_i = E_{i(i+1)}(E_{w(i+1)})^{-1}E_{wi} = E_{i(i+1)}E_{(i+1)w}E_{wi}$. Para todo par de *keyframes* K_j, K_k tal que se haya validado un ciclo entre estos, se define el error residual entre *keyframes* que cierran ciclos como $r_{j,k} = T_{jk}(E_{wk})^{-1}E_{wj} = T_{jk}E_{kw}E_{wj}$. Las transformaciones T_{jk} corresponden a las transformaciones estimadas en la etapa de validación de cada ciclo (Sección 7.3). De esta manera interesa ajustar las poses de cámara, de manera de minimizar los errores residuales definidos:

$$
\underset{E_{w0},\ldots,E_{w\mathcal{C}}}{argmin} \sum_i r_i^\top r_i + \sum_{j,k} r_{j,k}^\top r_{j,k} \tag{7.4.1}
$$

La solución al cierre del ciclo propagada inicialmente servirá como estimación inicial para los métodos de minimización utilizados, reduciendo considerablemente el tiempo requerido para optimizar las poses del mapa. Finalmente, una vez ajustadas las poses de cámara de todos los *keyframes*, cada punto del mapa es corregido aplicando la misma transformación que haya sufrido el *keyframe* que lo trianguló originalmente.

7.4.1. Actualización del mapa y sincronización de componentes

Para permitir al sistema operar en tiempo real junto con la extensión de detección y cierre de ciclos, dos propiedades deben tomar lugar en todo momento:

- El hilo del *Tracking* debe permanecer operacional en tiempo real, siendo capaz de trabajar con cualquier punto del mapa requerido y de crear nuevos *keyframes,* de ser necesario.
- El hilo de *Local Mapping* debe ser capaz de efectuar ajustes sobre *keyframes* y puntos dentro de un área del mapa definida.

Para asegurar estas dos propiedades, luego de que un ciclo es validado, se establece un área del mapa de "uso seguro" para el *Local Mapping*. Luego, la corrección del ciclo inicial y la optimización del grafo de poses se realizan en una copia interna de todos los *keyframes* presentes. Por último, la actualización del mapa se lleva a cabo en tres etapas:

- Actualización de *keyframes* y puntos del mapa corregidos que se encuentren por fuera del área de uso seguro definida.
- Actualización de *keyframes* y puntos del mapa corregidos que se encuentren dentro del área de uso seguro definida, aplicando toda optimización introducida por el *Local Mapping* desde el comienzo del proceso de detección y cierre del ciclo.
- Corregir cualquier *keyframe* creado y añadido al mapa luego de haber realizado la copia interna de *keyframes*. Esta corrección se realiza aplicando la misma transformación que fue utilizada para la corrección del *keyframe* más reciente K_c.

De esta manera, los hilos del *Tracking* y *Local Mapping* sólo necesitan ser detenidos durante las dos últimas etapas, donde solo una pequeña fracción del mapa necesita actualizarse. Luego de la actualización del mapa, resta notificar al *Pose Predictor* la transformación que debe aplicarse a la trayectoria en curso para corregir la estimación de localización próxima.

Durante este proceso el *Tracking* podría encontrarse con puntos del mapa que están siendo corregidos activamente por el *Loop Closing*, sin embargo se espera que estos sean descartados durante el proceso de proyección (Sección 3.3.1.2).

7.5. Detalles de implementación

A lo largo de todo el desarrollo de la solución propuesta se utilizó la biblioteca OpenCV para el procesamiento de imágenes, extracción de descriptores y asociación de correspondencias. A continuación se detallan diferentes decisiones de implementación y bibliotecas utilizadas para la confección del módulo *Loop Closing*.

Inclusión del módulo y manejo de la concurrencia

Originalmente, S-PTAM contaba con un único *lock* compartido para la sincronización requerida entre los hilos de *Tracking* y *Local Mapping* para mediar el acceso al mapa. Al incluir el método de detección y cierre de ciclos como un nuevo módulo concurrente, fue necesario modificar el diseño original de S-PTAM diferenciando distintos tipos de acceso. Se implementaron *locks* de lectura-escritura individuales para cada punto y *keyframe*, de manera que el *Tracking* y *Loop Closing* puedan trabajar en diferentes áreas del mapa simultáneamente.

Durante la corrección y actualización del mapa se requiere el establecimiento de un área de "uso seguro" para el *Local Mapping* de forma tal de evitar problemas de concurrencia con el accionar del *Loop Closing*. Esta área se define como una vecindad de 30 *keyframes* cercanos a la última pose estimada. Para el establecimiento de dicha vecindad, se incluyeron mensajes de sincronización entre los hilos de ejecución del *Tracking*, *Local Mapping* y *Loop Closing*. Durante la actualización del mapa, este tamaño de vecindad asegura que el *Tracking* y *Local Mapping* sean detenidos durante un breve periodo de tiempo, que se mantiene constante y no escala con el tamaño del mapa.

Detección y validación de ciclos

La detección de ciclos fue desarrollada en base al sistema DLoopDetector [48, 32] que implementa las técnicas descriptas en la Sección 4.3. A su vez, dicho sistema utiliza la librería DBoW2, introducida por los mismos autores, la cual provee soporte para el manejo de vectores *bag-of-words* y la construcción de bases de datos de similitud.

Asimismo, se modificó el comportamiento del detector DLoopDetector de manera de utilizar métodos PnP para la validación geométrica de los candidatos a ciclos, teniendo en cuenta que en este trabajo se consideran cámaras estéreo como sensor. Se utilizó la biblioteca OpenGV [64] para el cálculo de las transformaciones relativas entre las cámaras estéreo.

Propagación inicial y optimización del ciclo

Para el cálculo de la propagación, que se utiliza como solución inicial en el cierre de ciclos, se aplica una interpolación entre poses de cámara mediante el uso de una función exponencial. De esta forma, *keyframes* cercanos al punto donde se originó el ciclo se corrigen fuertemente, mientras que la transformación aplicada se ve rápidamente suavizada al propagar *keyframes* más distantes.

La optimización del grafo de restricciones y la minimización del error se lleva a cabo utilizando la biblioteca g2o [65] que implementa técnicas de *Bundle Adjustment* y eficientes variantes del algoritmo de minimización no lineal Levenberg–Marquardt [24, 25].

Capítulo 8

Experimentación y resultados

A partir de la solución propuesta para la resolución del problema planteado, se diseñaron diferentes experimentos que permiten evaluar la precisión, robustez y eficiencia de todos los métodos involucrados en la detección y cierre de ciclos. Los resultados se presentan de forma constructiva, representando el rumbo que se tomo durante desarrollo de este trabajo.

Los experimentos se llevan a cabo sobre el entorno de desarrollo ROS (*Robot Operating System*), el cual provee las herramientas necesarias para comprobar el desempeño del sistema en distintos tipos de escenarios. Se utilizan colecciones de datos recolectados durante trayectos realizado por diferentes robots o vehículos. A estas colecciones de datos de dominio público se las llama *datasets*. Los datos recolectados corresponden a las mediciones efectuadas por los diferentes sensores montados en el robot móvil o vehículo. ROS posibilita la confección y posterior reproducción de estos *datasets*, permitiendo evaluar el desempeño del sistema en diferentes condiciones.

En todos los casos, el hardware utilizado para el procesamiento de los experimentos corresponde a una computadora estándar de escritorio Intel(R) Core(TM) i7-860 de 2.80GHz de frecuencia y 16GB de memoria.

8.1. Elección del descriptor y entrenamiento del vocabulario visual

En todos los experimentos se utiliza el detector de esquinas Shi-Tomasi [9] y el descriptor binario BRIEF [13] para la detección y extracción de características visuales de las imágenes. Este descriptor es rápido de calcular y comparar, lo que permite al sistema operar en tiempo real extrayendo cientos de características visuales en cada cuadro. BRIEF no es invariante a rotación, sin embargo esto no representa un problema ya que se trabaja con *datasets* obtenidos por robots terrestres, por lo que la cámara realiza una trayectoria en un plano y no presenta rotaciones en el eje de balanceo (*roll*).

El vocabulario visual (Sección 4.3.2) fue entrenado en 6 niveles con 10 *clusters* por nivel, utilizando más de 10 mil imágenes provenientes de dos diferentes *datasets*. El MIT Stata Center Dataset [66] que consta de un robot desplazándose por las instalaciones académicas del *Massachusetts Institute of Technology* (MIT) y The Málaga Urban Dataset [67] que provee imágenes de un ambiente urbano capturadas por un vehículo transitando por las calles de la ciudad de Málaga. El vocabulario resultante consta de aproximadamente 1 millón de palabras.

Fig. 8.2.1: Ejemplos del KITTI *dataset*.

8.2. The KITTI dataset

The KITTI Vision Benchmark Suite [68] es una colección de *datasets* capturados por un vehículo transitando por las calles de la ciudad de Karlsruhe, Alemania. El *dataset* consta de imágenes estéreo con una resolución de 1344×391 *pixeles* a una frecuencia de 10 *frames* por segundo (10Hz). La colección se compone de 10 secuencias que proporcionan información de la trayectoria real (*ground truth*), por lo cual se conoce la localización precisa del vehículo en cada momento. Asimismo, ocurren ciclos en la trayectoria en 6 de estas secuencias (las secuencias: 00, 02, 05, 06, 07 y 09).

The KITTI Vision Benchmark Suite no provee información de *ground truth* sobre la ocurrencia de ciclos en la trayectoria. Arroyo et al.[69] realiza un análisis de las secuencias y propone un *ground truth* de ciclos donde define, para cada imagen, candidatos de alta similitud. Se utilizará esta asociación entre imágenes como *ground truth* de la ocurrencia de ciclos en las experimentaciones.

8.2.1. Desempeño de los métodos PnP

La validación de un ciclo erróneo podría atentar contra la estabilidad del sistema completo. Los métodos PnP determinarán, en última instancia, si un ciclo es validado por lo que resulta importante comprobar su desempeño. En la literatura, la evaluación de este tipo de métodos es generalmente llevado a cabo en ambientes simulados donde se incluye ruido en las mediciones de manera artificial [60, 58]. Se plantea, entonces, evaluar la efectividad de estos métodos al estimar las transformaciones relativas entre las cámaras involucradas en los ciclos de las distintas secuencias del KITTI *dataset*. Se seleccionan para esto secuencias que presenten una cantidad de ciclos considerable para las pruebas. En total, se utilizan 4530, 4568, 1453 y 1100 asociaciones entre imágenes del *ground truth* correspondientes a las secuencias 00, 02, 05 y 06 respectivamente.

Se evalúa la efectividad de dos métodos P3P: P3PKNEIP [60] y P3PGAO [59]; y dos métodos generales PnP: UPNP [58] y EPNP [56]. Los métodos P3P son aplicados bajo un esquema RANSAC de 50 iteraciones, con un *threshold* de re-proyección de 1 píxel. Las asociaciones de imágenes que no cumplieran con un mínimo de 80 % de *inliers* son descartadas. Los métodos PnP son utilizados sobre el conjunto de correspondencias *inliers* determinadas previamente por el método P3PKNEIP [60]. Dada la selección aleatoria de correspondencias del esquema RANSAC, existen discrepancias entre las asociaciones de imágenes consideradas por cada método. Es decir, una asociación de imágenes que cumplió con el mínimo de 80 % de *inliers* utilizando alguno de los métodos, podría no cumplirlo al utilizar un método distinto. La cantidad de *inliers* requeridos para la validación restringe considerablemente la cantidad

de asociaciones del *ground truth* que son validadas por los distintos método. Las Figuras 8.2.2, 8.2.3, 8.2.4 y 8.2.5 exponen la precisión de los diferentes métodos al estimar la transformación entre imágenes asociadas por el *ground truth*. Se puede observar un parejo desempeño entre los métodos minimales P3P y los generales PnP. Los errores promedio en la traslación se encuentran entre los 0.9 y 3 metros, siendo la secuencia 02 la que presenta mayores errores promedio. Los errores promedio en la orientación se encuentran entre los 0.8 y 2 grados. Se registra una mejora en la concentración de errores de traslación utilizando el método UPNP en la secuencia 02 (Fig. 8.2.3). P3PKNEIP y P3PGAO superan en desempeño a los métodos generales PnP en la secuencia 00 (Fig. 8.2.2) donde EPNP muestra errores de hasta 8 metros en la traslación.

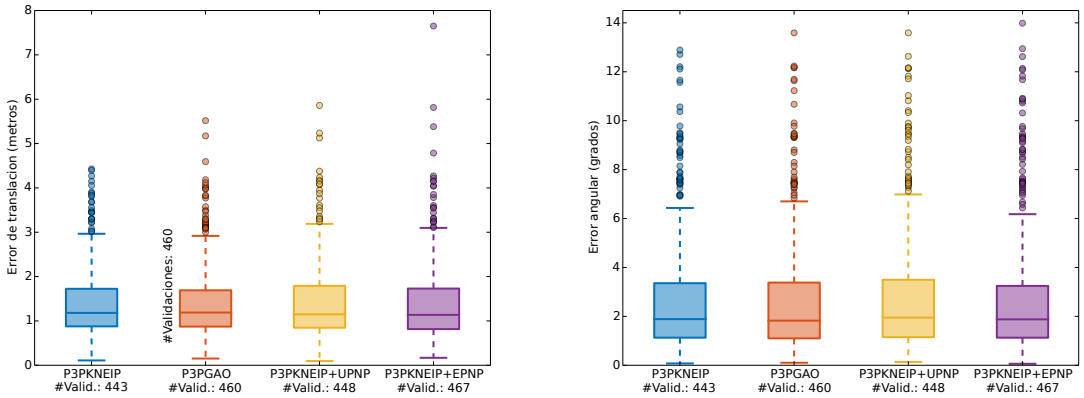


Fig. 8.2.2: Errores de estimación entre asociaciones de imágenes validadas del *ground truth* en la secuencia KITTI 00. Se incluye la cantidad de validaciones efectuadas por cada método.

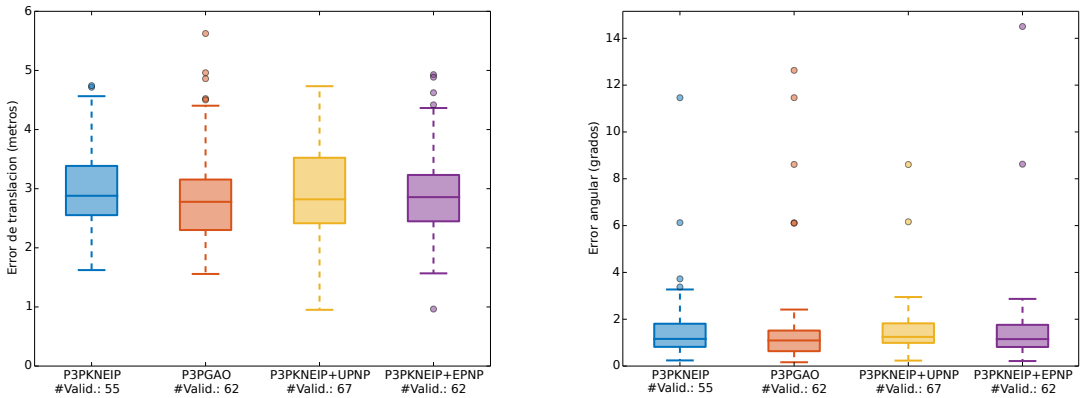


Fig. 8.2.3: Errores de estimación entre asociaciones de imágenes validadas del *ground truth* en la secuencia KITTI 02. Se incluye la cantidad de validaciones efectuadas por cada método.

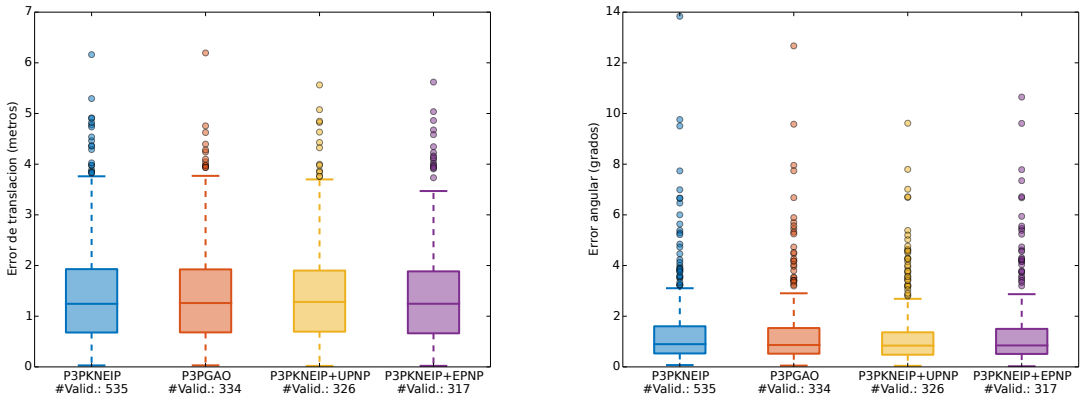


Fig. 8.2.4: Errores de estimación entre asociaciones de imágenes validadas del *ground truth* en la secuencia KITTI 05. Se incluye la cantidad de validaciones efectuadas por cada método.

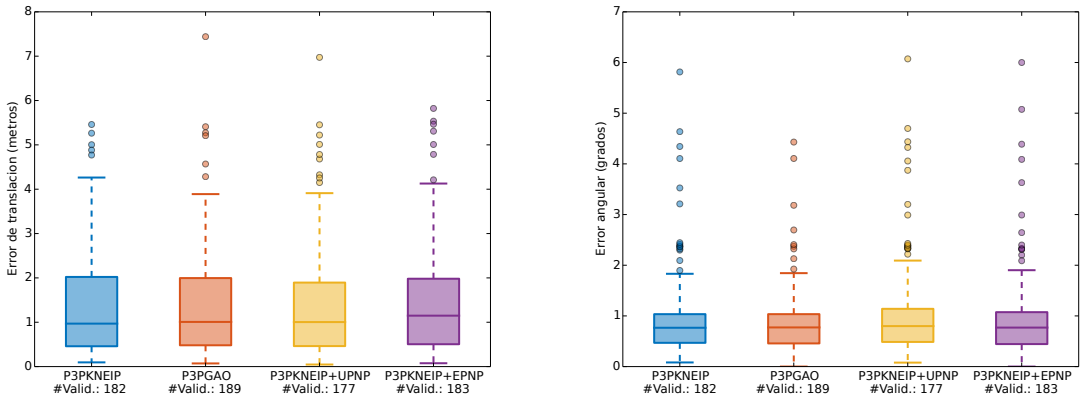


Fig. 8.2.5: Errores de estimación entre asociaciones de imágenes validadas del *ground truth* en la secuencia KITTI 06. Se incluye la cantidad de validaciones efectuadas por cada método.

La consistencia de los resultados obtenidos entre los diferentes métodos dan evidencia de la efectividad de los parámetros utilizados para la validación de ciclos. Se considera que las asociaciones de imágenes validadas del *ground truth* presentan un alto grado de similitud y los métodos fueron capaces de trabajar de manera efectiva sobre estas.

En base a los resultados obtenidos y aquellos presentados por Kneip et al.[60, 58], se utilizan los métodos P3PKNEIP y UPNP para el cálculo de la transformación relativa en experimentaciones posteriores.

8.2.2. Precisión en la detección y cantidad de ciclos reconocidos

Para evaluar la eficiencia de la detección y validación de ciclos, se realizó la ejecución del sistema S-PTAM sobre cada uno de los 6 *datasets* que presentan ciclos. La información de *ground truth* sobre la ocurrencia de ciclos presenta una asociación entre imágenes que resulta demasiado estricta en la práctica. Por esto, como se proponen los autores del *ground truth* [69], se considera una detección positiva cuando los cuadros involucrados se encuentran en

la vecindad temporal (1 segundo) de alguna asociación presente en el *ground truth*. Podrían existir varias asociaciones de ciclo para una misma imagen, por lo que el número de ciclos totales de una secuencia se calcula como la cantidad de imágenes que posean al menos una asociación de similitud en el *ground truth*.

Dado el comportamiento de selección de *keyframes* propio de S-PTAM (Sección 3.3.1.3), existen cuadros que no son agregados al mapa y no son procesados por el *Loop Closing*. Por esta razón se analizan dos medidas de reconocimiento de ciclos, una sobre el total de ciclos de la secuencia (%Recon.) y una sobre aquellos ciclos que fueron observados dada la selección de *keyframes* realizada durante la ejecución del sistema (%Recon. Obs.).

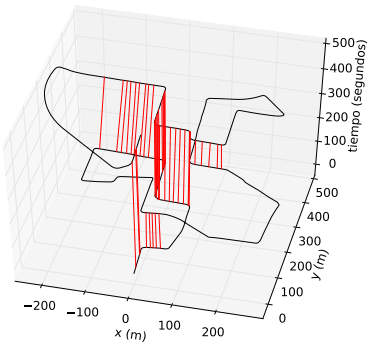
La tabla 8.1 muestra los resultados obtenidos por el método de detección de ciclos diferenciando el desempeño del análisis de apariencia y el de la validación geométrica. El análisis de apariencia resulta muy permisivo generando gran cantidad de candidatos a ciclos y un porcentaje de reconocimiento satisfactorio tomando en cuenta otros métodos del estado del arte [69]. La validación geométrica descarta falsos positivos de manera efectiva logrando una precisión del 100 % en las detecciones validadas. La cantidad de ciclos finalmente validados es proporcionalmente bajo en comparación a los ciclos definidos por el *ground truth*. Esto no resulta contraproducente dado que, al requerir un porcentaje elevado de *inliers*, las asociaciones validadas permiten la estimación de transformaciones relativas más precisas. De esta manera se prioriza la validación de una menor cantidad de ciclos, pero que resulten en estimaciones de mayor precisión, antes de arriesgarse a validar un ciclo erróneo.

Tabla 8.1: Resultados de precisión y reconocimiento en las diferentes secuencias.

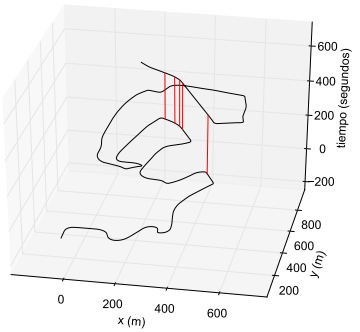
Secuencia	#Ciclos	Apariencia				Apariencia + Validación Geométrica			
		#Detec.	%Precisión	%Recon.	%Recon. Obs.	#Valid.	%Precisión	%Recon.	%Recon. Obs.
KITTI00	732	2747	13,14 %	49,32 %	57,39 %	45	100 %	6,14 %	7,15 %
KITTI02	234	3010	3,12 %	40,17 %	40,52 %	5	100 %	2,14 %	2,16 %
KITTI05	320	1596	12,4 %	61,88 %	68,51 %	28	100 %	8,75 %	9,69 %
KITTI06	269	412	24,03 %	36,8 %	38,98 %	16	100 %	5,95 %	6,30 %
KITTI07	13	434	2,07 %	69,23 %	100 %	1	100 %	7,69 %	11,11 %
KITTI09	17	836	0,60 %	29,41 %	35,71 %	0	/	0 %	0 %

Las secuencias 07 y 09 presentan un ciclo al final del trayecto, definido en el *ground truth* por 13 y 17 asociaciones de imágenes respectivamente. En ambos casos el detector de ciclos por apariencia genera candidatos validos, sin embargo solo se logran establecer suficientes *inliers* para el cálculo de la transformación relativa en el caso la secuencia 07.

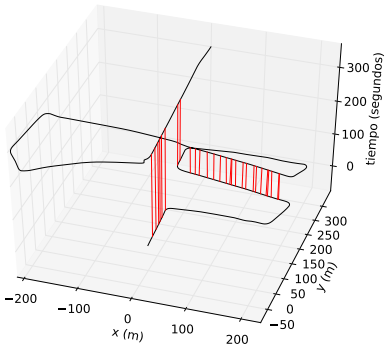
Las Figuras 8.2.6 muestran las detecciones validadas en las distintas secuencias, se expone la trayectoria *ground truth* en función del tiempo de manera de visualizar las asociaciones (líneas rojas) realizadas por el método. Es interesante destacar que líneas rectas responden a asociaciones correctas. Es posible observar que si bien la validación geométrica reduce considerablemente la cantidad de ciclos reconocidos, valida al menos uno en cada segmento previamente visitado de la trayectoria. Este comportamiento es favorable ya que reduce la cantidad de cierres de ciclos realizados por el sistema y concentra el procesamiento en ciclos con mayor cantidad de *inliers* donde las transformaciones relativas estimadas son más precisas.



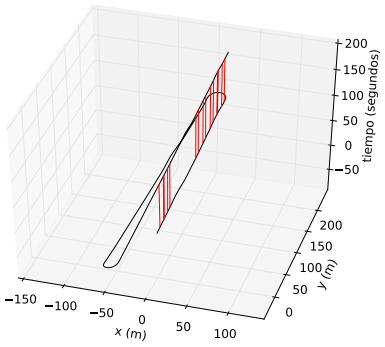
(a) Secuencia KITTI 00



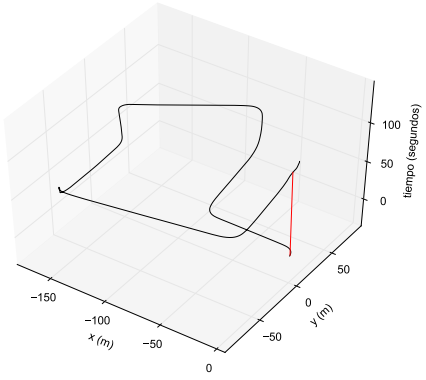
(b) Secuencia KITTI 02



(c) Secuencia KITTI 05



(d) Secuencia KITTI 06



(e) Secuencia KITTI 07

Fig. 8.2.6: Ciclos detectados y validados en las diferentes secuencias del KITTI *dataset*.

8.2.3. Trayectoria estimada y reducción del error acumulado

Se llevaron a cabo diferentes análisis sobre la información obtenida del procesamiento de las secuencias. El objetivo es corroborar la estabilidad del método en su conjunto, y determinar el impacto en la corrección de la trayectoria en cada caso. Las Figuras presentadas 8.2.7, 8.2.8, 8.2.9, 8.2.10, 8.2.11 exponen los resultados obtenidos durante la ejecución del sistema. Por cada secuencia se presenta una comparación de la trayectoria estimada por el sistema al incluir la extensión de *Loop Closing*, el resultado final del mapa luego de la corrección de los ciclos y un análisis "momento a momento" de los errores de traslación y rotación cometidos en la localización.

Dada la libertad de movimiento sobre la que se trabaja es difícil caracterizar el comportamiento del error acumulado en la estimación de localización. Es posible observar frecuentes oscilaciones en el error de traslación debido a lo que se conoce como "cancelación del error". El error acumulado en alguna de las direcciones puede verse cancelado posteriormente por la acumulación de error en el sentido contrario. Por el contrario, la calidad del mapa construido decrece progresivamente potenciando la acumulación de error en estimaciones posteriores. Al momento de realizar un cierre de ciclo, el error de traslación se ve ajustado a los valores existentes al momento en que se visitó la ubicación por primera vez. Por lo general esto se traduce en una reducción del error acumulado, pero existen situaciones donde, debido al efecto de cancelación, el error de traslación no se ve afectado. Finalmente, en todos los casos se espera que la detección y cierre de ciclos mejoren en la fidelidad del mapa hasta el momento construido.

En la secuencia 00 se puede observar una disminución del error absoluto de traslación de manera consistente desde el momento en que se produce la primera detección (Fig.(c) 8.2.7). El error angular presenta un comportamiento errático, sin embargo se puede apreciar una mejora en la estimación. Observar que la corrección de sucesivos ciclos temporalmente cercanos no se traduce en una mayor reducción del error de estimación. Dado que el sistema acumula error en intervalos de tiempo donde no ocurren ciclos, el mayor impacto en la precisión lo produce el primer ciclo corregido luego de un intervalo de tiempo prolongado donde no ocurriera ninguno.

La secuencia 02 realiza un recorrido extenso sin ciclos hasta casi el final de la trayectoria. Luego del primer cierre de ciclo no se aprecia una reducción del error en la traslación, y por el contrario el sistema S-PTAM original logra estimaciones de mejor calidad (Fig.(c) 8.2.8). Este ciclo detectado vincula la locación visitada en el segundo 170 con aquella visitada en en el segundo 440 aproximadamente. Debido a la cancelación del error, estos dos momentos de la trayectoria presentan similar error de traslación y es por esto que el cierre del ciclo no produce una notoria mejora de la estimación. Por el contrario, el cierre de ciclo reduce el efecto de cancelación del error posterior generando estimaciones menos precisas que el S-PTAM original. Se puede apreciar que la trayectoria final estimada se encuentra correctamente alineada (Fig.(b) 8.2.8), algo que no sucede con el S-PTAM original.

En la secuencia 05 el error rotacional percibido es pequeño (Fig.(d) 8.2.9). Al cerrar el primer ciclo, el cálculo de la transformación relativa introduce error en la orientación y este se traduce en error de traslación al realizar una amplia curva en la trayectoria. El sistema logra recuperarse de esta situación luego de una serie de detecciones al final de la secuencia que reducen considerablemente el error acumulado.

Las secuencias 06 y 07 presentan un único ciclo al final del trayecto. Se puede observar que al cerrar ciclos próximos al punto de partida el error de traslación se reduce de manera drástica, obteniendo una estimación con solo centímetros de error.

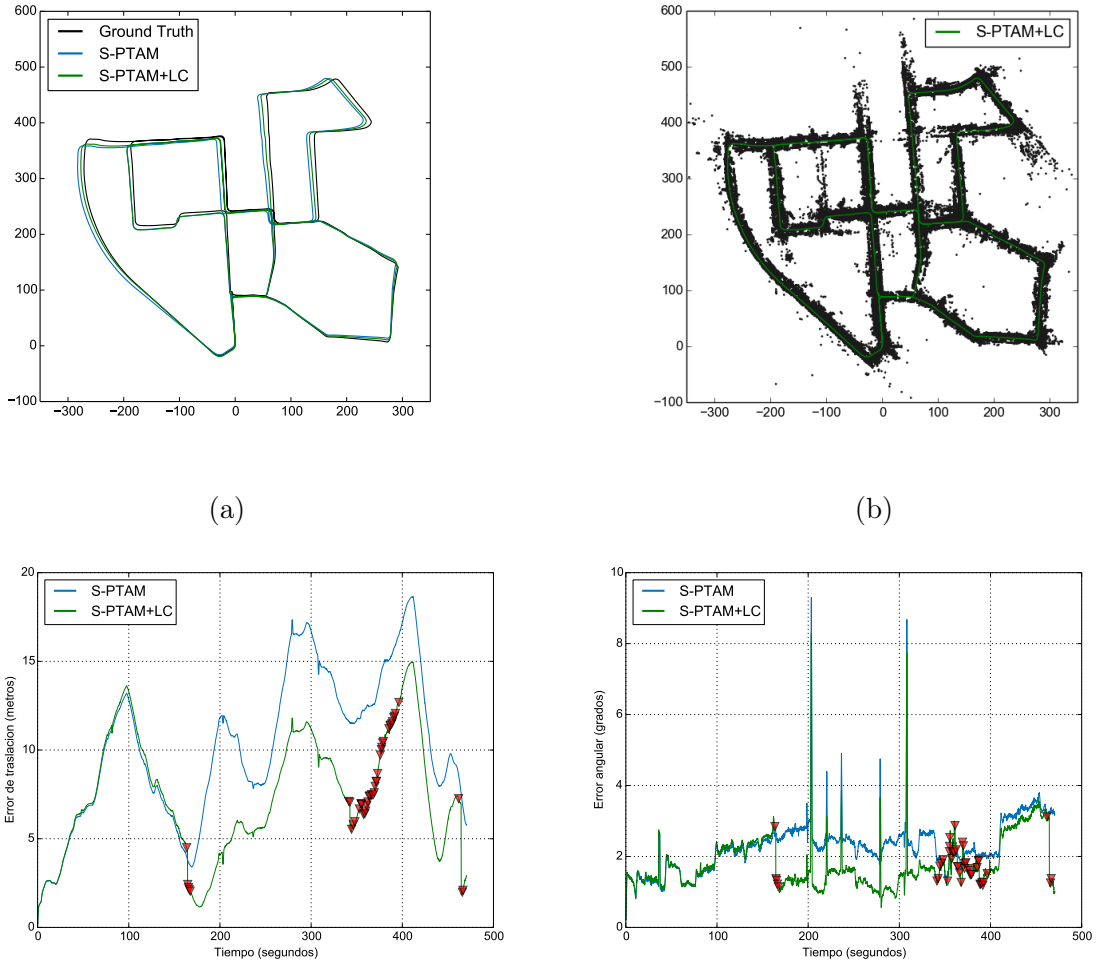
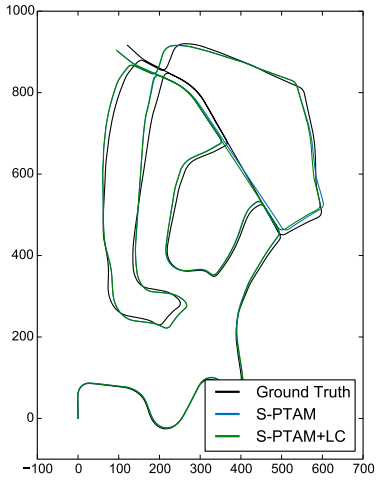
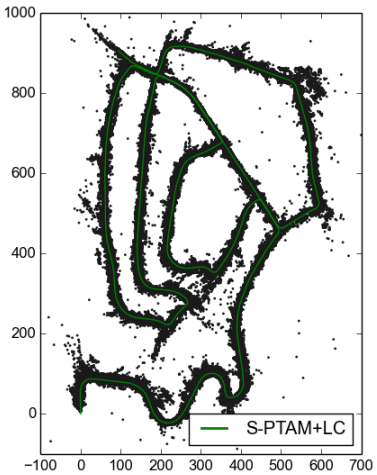


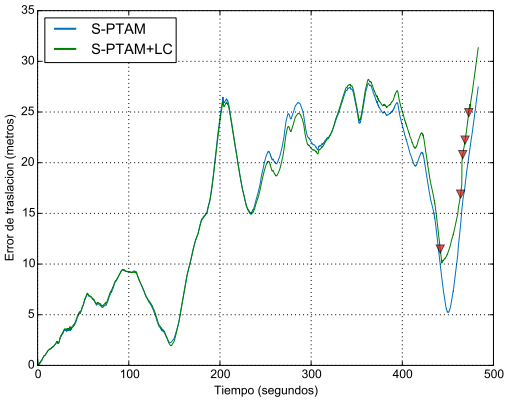
Fig. 8.2.7: Resultados del método en la secuencia KITTI 00 (a) Comparación de la trayectoria estimadas por el sistema S-PTAM original y al incluir la extensión de *Loop Closing*. (b) Visualización del mapa construido y corregido por el método luego de los cierres de ciclos. (c) y (d) Error de traslación y angular cometido al estimar la localización en cada momento de la secuencia. Los marcadores rojos indican los momentos en que se efectuaron correcciones de ciclo.



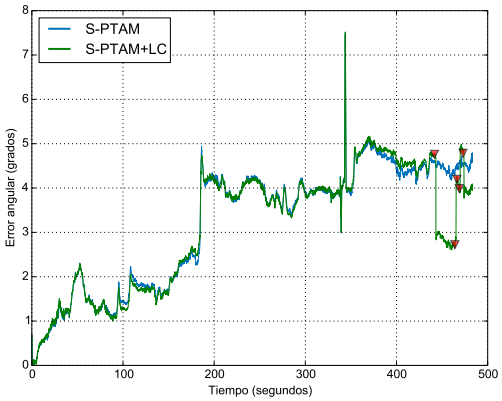
(a)



(b)

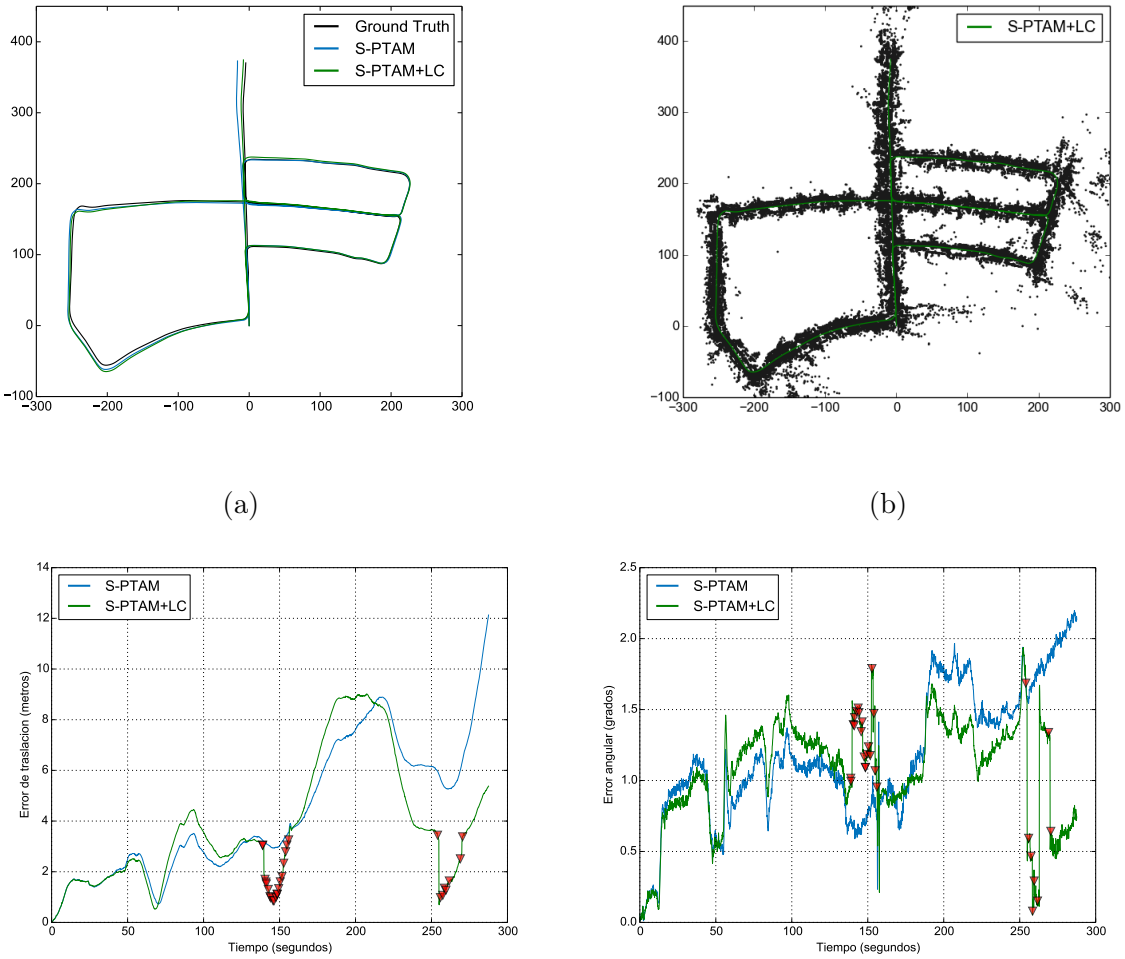


(c)



(d)

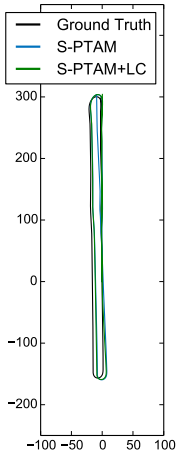
Fig. 8.2.8: Resultados del método en la secuencia KITTI 02 (a) Comparación de la trayectoria estimadas por el sistema S-PTAM original y al incluir la extensión de *Loop Closing*. (b) Visualización del mapa construido y corregido por el método luego de los cierres de ciclos. (c) y (d) Error de traslación y angular cometido al estimar la localización en cada momento de la secuencia. Los marcadores rojos indican los momentos en que se efectuaron correcciones de ciclo.



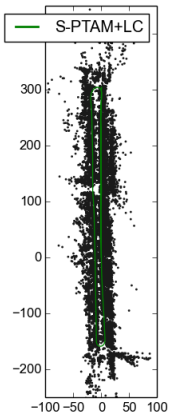
(a) (b)

(c) (d)

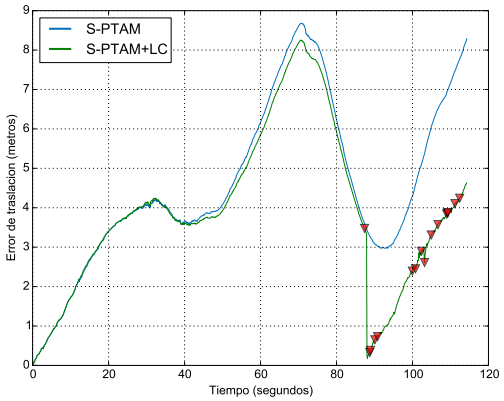
Fig. 8.2.9: Resultados del método en la secuencia KITTI 05 (a) Comparación de la trayectoria estimadas por el sistema S-PTAM original y al incluir la extensión de *Loop Closing*. (b) Visualización del mapa construido y corregido por el método luego de los cierres de ciclos. (c) y (d) Error de traslación y angular cometido al estimar la localización en cada momento de la secuencia. Los marcadores rojos indican los momentos en que se efectuaron correcciones de ciclo.



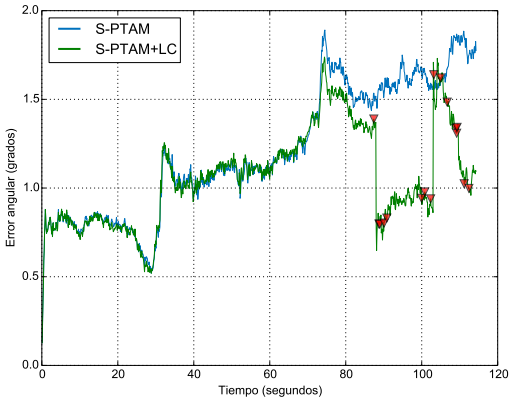
(a)



(b)



(c)



(d)

Fig. 8.2.10: Resultados del método en la secuencia KITTI 06 (a) Comparación de la trayectoria estimadas por el sistema S-PTAM original y al incluir la extensión de *Loop Closing*. (b) Visualización del mapa construido y corregido por el método luego de los cierres de ciclos. (c) y (d) Error de traslación y angular cometido al estimar la localización en cada momento de la secuencia. Los marcadores rojos indican los momentos en que se efectuaron correcciones de ciclo.

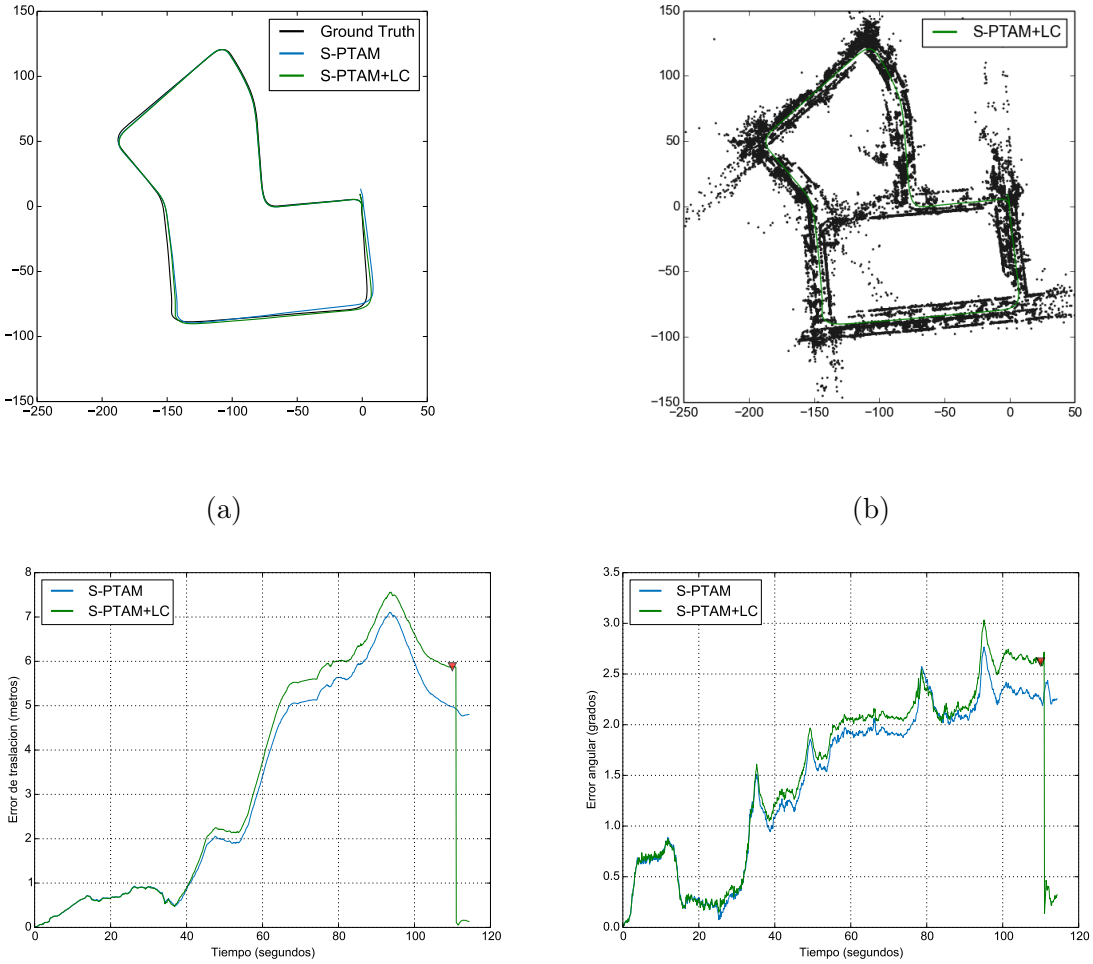


Fig. 8.2.11: Resultados del método en la secuencia KITTI 07 (a) Comparación de la trayectoria estimadas por el sistema S-PTAM original y al incluir la extensión de *Loop Closing*. (b) Visualización del mapa construido y corregido por el método luego de los cierres de ciclos. (c) y (d) Error de traslación y angular cometido al estimar la localización en cada momento de la secuencia. Los marcadores rojos indican los momentos en que se efectuaron correcciones de ciclo.

8.3. Level 7 S-Block dataset

El *Level 7 S-Block dataset* [70] provee información recolectada durante el recorrido realizado por un robot terrestre en el nivel 7 del *S-Block QUT Gardens Point campus* en Brisbane, Australia. El robot, denominado *The Adept Guiabot*, tiene montados diversos sensores que proveen gran cantidad de información como imágenes estéreo, mediciones de láser y estimaciones odométricas de posición basadas en la rotación de las ruedas. La secuencia dura aproximadamente 33 minutos proporcionando 12 imágenes por segundo (12Hz) y el recorrido realizado corresponde a una serie de vueltas en un ambiente interior (*indoor*) de oficinas. Los autores proveen información de *ground truth* de la trayectoria a través del análisis de las mediciones provistas por un escáner láser. Esto resulta en información de localización precisa pero ruidosa en comparación al *ground truth* del KITTI. No se tiene información de *ground truth* sobre los ciclos de la trayectoria, pero estos ocurren frecuentemente permitiendo realizar pruebas de estrés sobre el modulo de *Loop Closing*. La gran cantidad de ciclos presentes en la trayectoria proporcionan suficiente información para comprobar la estabilidad del cierre y medir los tiempos de procesamiento requeridos.

En todos los experimentos realizados con este *dataset* se configura al sistema S-PTAM de manera de utilizar las mediciones odométricas de las ruedas provistas por el *dataset* como predicciones iniciales de la posición.

Fig. 8.3.1: The Adept Guiabot e imágenes de ejemplo del *Level7 dataset*.

8.3.1. Trayectoria estimada y reducción del error acumulado

La secuencia presenta pasillos de escasa textura y con pocos elementos distinguibles donde se llevan a cabo pronunciados cambios de orientación en la trayectoria. Este escenario produce una acentuada acumulación de error en las estimaciones generadas por el S-PTAM original (Fig.(a) 8.3.3). El desempeño del sistema se degrada progresivamente manifestando un patrón oscilatorio que crece en amplitud en los errores de traslación y angular percibidos (Figs.(c)(d) 8.3.3). Esta degradación llega al punto de registrar errores de 25 metros en la traslación y 90 grados en la orientación (Figs.(c)(d) 8.3.3). Esto imposibilita la utilización del sistema S-PTAM original como sistema de localización en este escenario.

La inclusión del modulo de *Loop Closing* mejora notoriamente el desempeño exhibiendo errores máximos de 2,2 metros en la traslación y 14 grados en la orientación (Figs.(e)(f) 8.3.3). El sistema es capaz de realizar correctas asociaciones de apariencia a pesar de tratarse de un entorno repetitivo, en total se validan y cierran 317 ciclos (Fig. 8.3.2). Se puede apreciar que la

amplitud de los errores registrados se encuentra acotada y no crece con el tiempo a diferencia del comportamiento observado en el S-PTAM original.

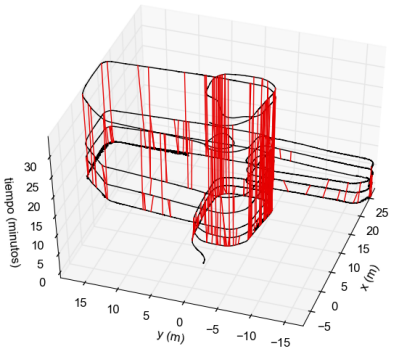
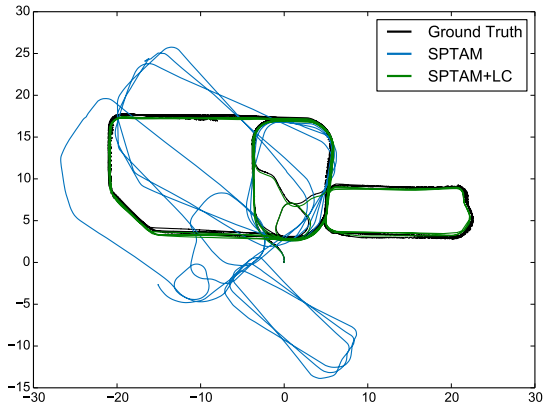
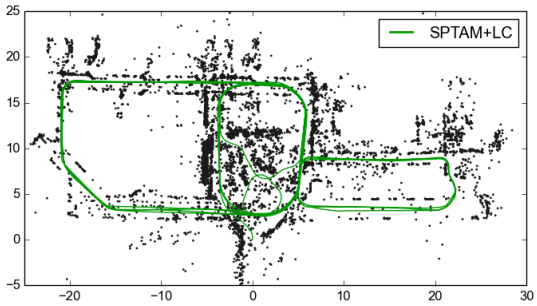


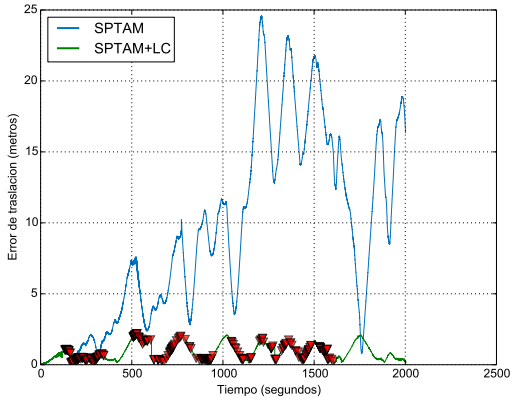
Fig. 8.3.2: Ciclos detectados y validados en la trayectoria del *Level7 dataset*.



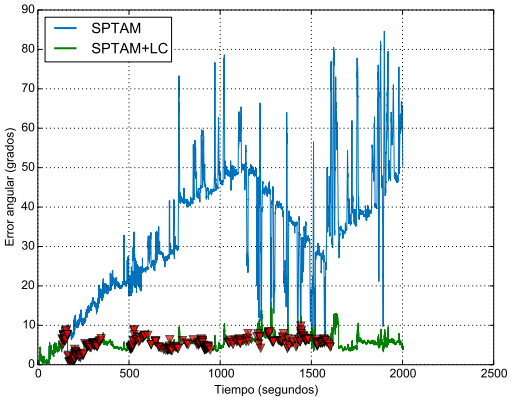
(a)



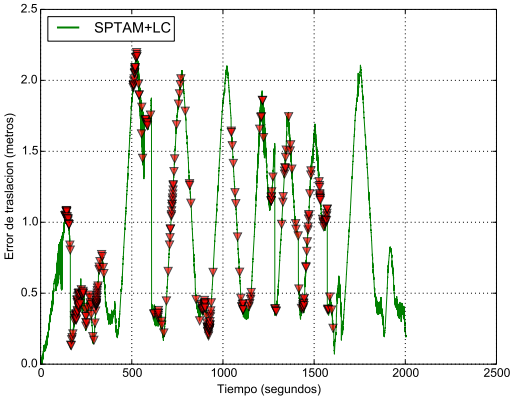
(b)



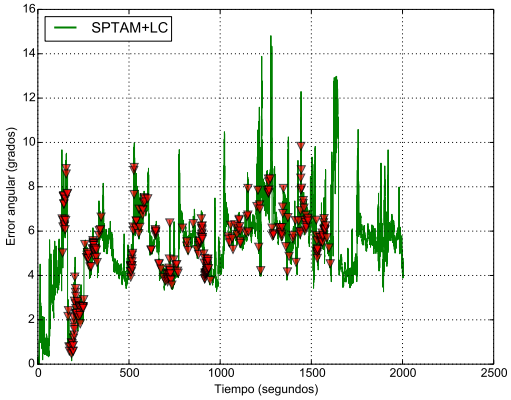
(c)



(d)



(e)



(f)

Fig. 8.3.3: Comparación de desempeño en el *Level7 dataset*. Dada la marcada diferencia de magnitud entre los errores percibidos, las Figuras (e) y (f) repiten los resultados obtenidos por el sistema en una escala más adecuada.

8.3.2. Tiempos de procesamiento

Durante la secuencia se realizan 3706 análisis de apariencia, 2218 análisis geométricos y 317 cierres de ciclos. La tabla 8.2 presenta los tiempos de procesamiento consumidos por cada una de las etapas del modulo *Loop Closing*. Dado el diseño del método (Sección 7.4.1), el cierre de los ciclos y la actualización de la mayor parte del mapa se realiza de manera completamente paralela. La actualización del área compartida del mapa resulta muy veloz requiriendo la detención del *Tracking* y el *Local Mapping* por 1,6 milisegundos en promedio. Dada la implementación realizada (Sección 7.5), el tamaño del área compartida del mapa es constante y no escala con el tamaño del mapa. De esta manera el tiempo en que permanece detenido el *Tracking,* durante un cierre de ciclo, es constante. Por el contrario, los tiempos requeridos por el cierre de ciclos y la actualización del área no compartida escalan con el tamaño del mapa. Esto no presenta un problema dado que el área no compartida se corrige de manera concurrente sin detener el *Tracking*. Las Figuras 8.3.4, 8.3.5 exhiben los tiempos de procesamiento requeridos en función a la cantidad de *keyframes* del mapa.

Tabla 8.2: Tiempos de procesamiento consumido por las diferentes etapas.

Etapas	#Mediciones	Mín.(ms)	Prom.(ms)	Máx.(ms)
Detección	3706	0	0,74	8
Val. y transf. relativa	2218	0	2,31	11
Cierre del ciclo	317	42	384,92	922
Actualización de mapa	317	17	80,53	187
Actualización de mapa (área de utilización compartida)	317	0	1,6	4

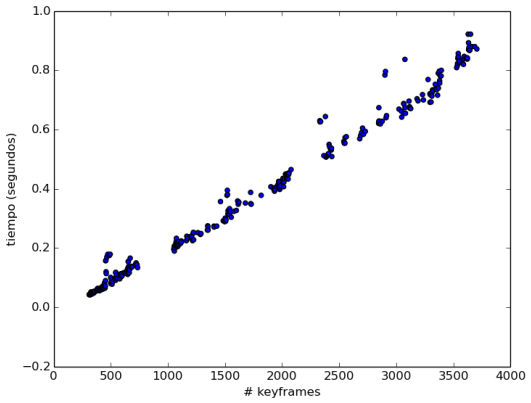


Fig. 8.3.4: Tiempos de procesamiento requeridos por el cierre de ciclos. Notar que los intervalos sin datos corresponden a momentos de la trayectoria donde no se validaron ciclos.

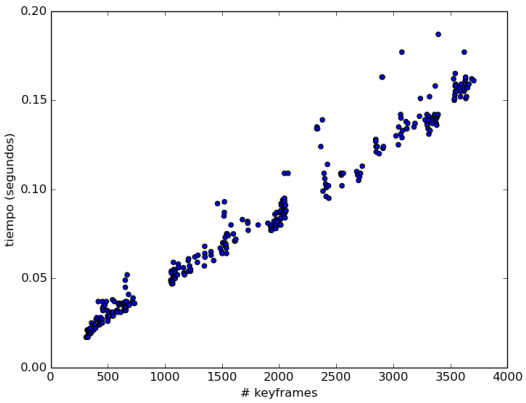


Fig. 8.3.5: Tiempos de procesamiento requeridos por la actualización del área no compartida del mapa. Notar que los intervalos sin datos corresponden a momentos de la trayectoria donde no se validaron ciclos.

Capítulo 9

Conclusiones

En este trabajo se presenta el desarrollo de un método de detección y cierre de ciclos integrado a un sistema SLAM basado en visión estéreo capaz de trabajar en tiempo real. El método se divide en tres etapas: la detección de lugares previamente visitados (ciclos en la trayectoria), el cálculo del desvío cometido en las estimaciones de localización y la corrección del ciclo detectado. Por cada etapa se presenta un profundo estudio y análisis del estado del arte, exhibiendo los diferentes métodos y técnicas que motivaron la solución propuesta.

La detección de ciclos consta de analizar la similitud, en términos de apariencia, entre las imágenes capturadas del ambiente a lo largo de la trayectoria de una cámara. Se optó por un análisis de similitud permisivo de tal manera que se consideren todos los ciclos existentes a lo largo de la trayectoria. Para realizar una correcta detección, los candidatos a ciclos son validados sólo si presentan un alto grado de similitud y consistencia geométrica. Para esto, la validación estima la transformación relativa entre las cámaras, aquellos candidatos que no presenten una gran cantidad de correspondencias *inliers* son descartados. El cálculo de esta transformación se realiza mediante el uso de métodos PnP bajo un esquema RANSAC. Cabe destacar que es posible realizar el procedimiento de detección y validación en tiempo real, gracias a la utilización de descriptores binarios y técnicas *bag-of-words*.

Una vez validado un ciclo, se efectúa la corrección de la trayectoria computando una solución inicial mediante la propagación del error acumulado hasta el momento del ciclo. Esta solución inicial es luego refinada a través de un método de optimización de grafos sobre las poses estimadas a lo largo de la trayectoria. La implementación se llevo acabo de manera de minimizar el impacto en el desempeño del sistema, la corrección y actualización del mapa se realiza de manera paralela permitiendo el seguimiento de la cámara de manera ininterrumpida.

La evaluación del sistema desarrollado se llevo acabo utilizando *datasets* de dominio público. Los diferentes experimentos realizados permitieron comprobar la precisión, robustez y estabilidad de cada etapa del método y del sistema en su conjunto. Asimismo, se caracterizó el impacto percibido en la reducción del error acumulado en las estimaciones. Los resultados obtenidos muestran que el método de detección y cierre de ciclos reduce y mantiene acotado el error global acumulado por el sistema SLAM estéreo S-PTAM, sin impactar negativamente su rendimiento.

9.1. Trabajo futuro

Como trabajo futuro, el método de detección y cierre de ciclos presentado podría extenderse para resolver el problema de relocalización, de forma de recuperar al sistema SLAM ante situaciones donde el módulo de localización falle por completo, como es en el caso del problema del robot secuestrado (*kidnapping problem*).

Otra posible línea de trabajo a futuro es realizar un completo estudio comparativo del *threshold* del número de *inliers* requeridos para la validación de ciclos. Los métodos PnP implementados para el cálculo de la transformación entre dos pares de cámaras estéreo utilizan únicamente las correspondencias establecidas entre los puntos previamente triangulados por el primer par y las características visuales presentes en la cámara izquierda del segundo par. Una posible mejora es utilizar métodos PnP que aprovechen, además, las correspondencias de la cámara derecha del segundo par [71].

Otro aspecto sobre el que se podría trabajar, es el de mejorar la etapa del cierre de ciclos. En particular, es posible implementar la técnica propuesta en [72] para la propagación realizada durante la estimación de la solución inicial. Al concluir el cierre de un ciclo, podrían existir puntos del mapa que hacen referencia a una misma marca del ambiente. Por este motivo, resulta de interés realizar una fusión de dichos puntos como se propone en [49]. Esto mejoraría la fidelidad del mapa con respecto al entorno, y por lo tanto, mejorar las estimaciones de localización del método S-PTAM.

Finalmente, es posible reutilizar el vocabulario visual para acelerar el establecimiento de correspondencias entre imágenes. Reduciendo la cantidad de comparaciones entre características visuales a solo aquellas que pertenezcan a una misma palabra, se evita realizar una búsqueda exhaustiva. Es posible aprovechar esto durante la etapa de seguimiento de la cámara donde el sistema S-PTAM realiza asociaciones entre los puntos del mapa y las características visuales presentes en la imagen más reciente.

Bibliografía

[1] H. Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *Robotics Automation Magazine, IEEE*, 13(2):99–110, June 2006.

[2] Tim Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (slam): part ii. *Robotics Automation Magazine, IEEE*, 13(3):108–117, Sept 2006.

[3] Taihú Pire, Thomas Fischer, Javier Civera, Pablo De Cristóforis, and Julio Jacobo Berlles. Stereo parallel tracking and mapping for robot localization. In *International Conference on Intelligent Robots and Systems*, Hamburg, Germany, September 2015.

[4] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–652–I–659 Vol.1, June 2004.

[5] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[6] Veeravalli Seshadri Varadarajan. *Lie groups, Lie algebras, and their representations*, volume 102. Springer Science & Business Media, 2013.

[7] H. Moravec. Towards automatic visual obstacle avoidance. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, page 584, August 1977.

[8] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[9] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 593–600, Jun 1994.

[10] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Computer Vision – ECCV 2006: 9th European Conference on Computer Vision*, pages 430–443. Springer, 2006.

[11] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[12] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *Computer Vision – ECCV 2006: 9th European Conference on Computer Vision*, pages 404–417. Springer, 2006.

[13] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. *Computer Vision – ECCV 2010: 11th European Conference on Computer Vision*, pages 778–792, 2010.

[14] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571, Nov 2011.

[15] S. Leutenegger, M. Chli, and R.Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555, Nov 2011.

[16] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT press, 2005.

[17] A.J. Davison and D.W. Murray. Simultaneous localization and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):865–880, 2002.

[18] A.J. Davison, Ian D. Reid, N.D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):1052–1067, June 2007.

[19] Lina M. Paz, P. Pinies, J.D. Tardos, and J. Neira. Large-Scale 6-DOF SLAM With Stereo-in-Hand. *IEEE Transactions on Robotics*, 24(5):946–957, Oct 2008.

[20] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem. In *Eighteenth National Conference on Artificial Intelligence*, pages 593–598, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[21] N. Karlsson, E. Di Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian, and M.E. Munich. The vSLAM Algorithm for Robust Localization and Mapping. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 24–29, April 2005.

[22] E. Eade and Tom Drummond. Scalable Monocular SLAM. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 469–476, June 2006.

[23] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. *Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings*, chapter Bundle Adjustment — A Modern Synthesis, pages 298–372. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[24] Kenneth Levenberg. A method for the solution of certain non–linear problems in least squares. 1944.

[25] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[26] Frank Dellaert and Michael Kaess. Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing. *The International Journal of Robotics Research*, 25(12):1181–1203, 2006.

[27] Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *ISMAR*, pages 1–10, Washington, DC, USA, 2007. IEEE Computer Society.

[28] K. Konolige and M. Agrawal. FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, Oct 2008.

[29] H. Strasdat, AJ. Davison, J. M M Montiel, and K. Konolige. Double window optimisation for constant time visual SLAM. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2352–2359, Nov 2011.

[30] Hauke Strasdat, J. M. M. Montiel, and Andrew J. Davison. Editors Choice Article: Visual SLAM: Why Filter? *Image Vision Comput.*, 30(2):65–77, February 2012.

[31] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *CoRR*, abs/1502.00956, 2015.

[32] Dorian Galvez-Lopez and J. D. Tardos. Bags of Binary Words for Fast Place Recognition in Image Sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, October 2012.

[33] H. Strasdat, J. M. M. Montiel, and A. Davison. Scale drift-aware large scale monocular slam. In *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010.

[34] Yaakov Bar-Shalom. *Tracking and data association*. Academic Press Professional, Inc., 1987.

[35] Yaakov Bar-Shalom, Fred Daum, and Jim Huang. The probabilistic data association filter. *Control Systems, IEEE*, 29(6):82–100, 2009.

[36] José Neira and Juan D Tardós. Data association in stochastic mapping using the joint compatibility test. *Robotics and Automation, IEEE Transactions on*, 17(6):890–897, 2001.

[37] José Neira, Juan D Tardós, and José A Castellanos. Linear time vehicle relocation in slam. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 427–433. Citeseer, 2003.

[38] Laura A Clemente, Andrew J Davison, Ian D Reid, José Neira, and Juan D Tardós. Mapping large loops with a single hand-held camera. In *Robotics: Science and Systems*, volume 2, page 2, 2007.

[39] Tim Bailey, Eduardo Mario Nebot, JK Rosenblatt, and Hugh F Durrant-Whyte. Data association for mobile robot navigation: A graph theoretic approach. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 3, pages 2512–2517. IEEE, 2000.

[40] Brian Williams, Georg Klein, and Ian Reid. Real-time slam relocalisation. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[41] Brian Williams, Georg Klein, and Ian Reid. Automatic relocalization and loop closing for real-time monocular slam. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(9):1699–1712, 2011.

[42] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, Sept 2006.

[43] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477. IEEE, 2003.

[44] Kin Leong Ho and Paul Newman. Detecting loop closure with scene sequences. *International Journal of Computer Vision*, 74(3):261–286, 2007.

[45] Mark Cummins and Paul Newman. Fab-map: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008.

[46] Mark Cummins and Paul Newman. Accelerating fab-map with concentration inequalities. *Robotics, IEEE Transactions on*, 26(6):1042–1050, 2010.

[47] CK Chow and CN Liu. Approximating discrete probability distributions with dependence trees. *Information Theory, IEEE Transactions on*, 14(3):462–467, 1968.

[48] Dorian Galvez-Lopez and Juan D Tardos. Real-time loop detection with bags of binary words. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 51–58. IEEE, 2011.

[49] Raúl Mur-Artal and Juan D Tardós. Fast relocalisation and loop closing in keyframe-based slam. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 846–853. IEEE, 2014.

[50] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, volume 2, pages 2161–2168. IEEE, 2006.

[51] Costantino Grana, Daniele Borghesani, Marco Manfredi, and Rita Cucchiara. A fast approach for integrating orb descriptors in the bag of words model. In *IS&T/SPIE Electronic Imaging*, pages 866709–866709. International Society for Optics and Photonics, 2013.

[52] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[53] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[54] YI Abdel-Aziz. Direct linear transformation from comparator coordinates in close-range photogrammetry. In *ASP Symposium on Close-Range Photogrammetry in Illinois, 1971*, 1971.

[55] Chien-Ping Lu, Gregory D Hager, and Eric Mjolsness. Fast and globally convergent pose estimation from video images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(6):610–622, 2000.

[56] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155–166, 2009.

[57] Berthold KP Horn, Hugh M Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *JOSA A*, 5(7):1127–1135, 1988.

[58] Laurent Kneip, Hongdong Li, and Yongduek Seo. Upnp: An optimal o (n) solution to the absolute pose problem with universal applicability. In *Computer Vision–ECCV 2014*, pages 127–142. Springer, 2014.

[59] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(8):930–943, 2003.

[60] Laurent Kneip, Davide Scaramuzza, and Roland Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2969–2976. IEEE, 2011.

[61] Carlos Estrada, José Neira, and Juan D Tardós. Hierarchical slam: Real-time accurate mapping of large environments. *Robotics, IEEE Transactions on*, 21(4):588–596, 2005.

[62] Christopher Mei, Gabe Sibley, Mark Cummins, Paul Newman, and Ian Reid. Rslam: A system for large-scale mapping in constant-time using stereo. *International journal of computer vision*, 94(2):198–214, 2011.

[63] Ken Shoemake. Animating rotation with quaternion curves. In *ACM SIGGRAPH computer graphics*, volume 19, pages 245–254. ACM, 1985.

[64] Laurent Kneip and Paul Furgale. Opengv: A unified and generalized approach to real-time calibrated geometric vision. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1–8. IEEE, 2014.

[65] G Grisetti, H Strasdat, K Konolige, and W Burgard. g2o: A general framework for graph optimization. In *IEEE International Conference on Robotics and Automation*, 2011.

[66] Maurice Fallon, Hordur Johannsson, Michael Kaess, and John J Leonard. The mit stata center dataset. *The International Journal of Robotics Research*, 32(14):1695–1699, 2013.

[67] José-Luis Blanco-Claraco, Francisco-Ángel Moreno-Dueñas, and Javier González-Jiménez. The málaga urban dataset: High-rate stereo and lidar in a realistic urban scenario. *The International Journal of Robotics Research*, 33(2):207–214, 2014.

[68] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, page 0278364913491297, 2013.

[69] Roberto Arroyo, Pablo F Alcantarilla, Luis M Bergasa, J Javier Yebes, and Sebastián Bronte. Fast and effective visual place recognition using binary codes and disparity information. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 3089–3094. IEEE, 2014.

[70] Liz Murphy, Timothy Morris, Ugo Fabrizi, Michael Warren, Michael Milford, Ben Upcroft, Michael Bosse, and Peter Corke. Experimental comparison of odometry approaches. In *Experimental Robotics*, pages 877–890. Springer, 2013.

[71] Laurent Kneip, Paul Furgale, and Roland Siegwart. Using multi-camera systems in robotics: Efficient solutions to the npnp problem. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3770–3776. IEEE, 2013.

[72] Gijs Dubbelman and Brett Browning. Cop-slam: Closed-form online pose-chain optimization for visual slam. *Robotics, IEEE Transactions on*, 31(5):1194–1213, 2015.