Universidad Zaragoza

TRABAJO FIN DE GRADO

DE INGENIERÍA INFORMÁTICA

Seguimiento y relocalización de una cámara monocular en mapas densos tridimensionales.

Directores

Javier Civera Sancho (dir)

Luis Montesano del Campo (codir)

Autor

José María Fácil Ledesma

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

Escuela de Ingeniería y Arquitectura

—

Zaragoza, Diciembre de 2014

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. José María Fácil Ledesma,

con nº de DNI 18049340-K en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Seguimiento y relocalización de una cámara monocular en mapas densos tridimensionales,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 12 de Febrero de 2015

Fdo: José María Fácil Ledesma

Agradecimientos

Me gustaría dedicar esta pagina para agradecer la colaboración de aquellos que han ayudado directa o indirectamente en la realización de este trabajo. En primer lugar a Alejo Concha por toda la ayuda que me ha dado y la paciencia que ha tenido conmigo.

Quiero agradecer profundamente a mis directores, Javier Civera y Luis Montesano, la gran oportunidad que me han dado, permitiendo trabajar en un entorno de investigación. Y quiero agradecerles también todo lo que me han ayudado, gracias.

A mis compañeros de laboratorio, Jason Omedes, Eduardo Lopez y Carlos Escolano por sus consejos y apoyo.

A mis amigos, que siempre me han apoyado y ayudado, ya sea directa o indirectamente. Y por ultimo, pero no por ello menos importante, a mi familia, que nunca se han dejado de animarme.

A todos, muchas gracias.

Resumen

La localización de una cámara es un problema clásicos de la Visión por Computador con aplicaciones en diferentes campos como la robotica o la fotogrametría. El objetivo de la localización es recuperar la posición y orientación de la cámara respecto a una referencia global en el mundo. Para ello normalmente se utilizan mapas tri-dimensionales basados en características. Estas características son puntos salientes e invariantes a escala. Emparejando los puntos salientes del mapa con los de la imagen es posible recuperar la posición de la cámara. Aunque estas técnicas son ampliamente utilizadas, no son robustas a situaciones donde las características no sean estables. Por ejemplo, en entornos con brillos y reflejos no es posible extraer este tipo de características.

En los últimos años se han propuesto soluciones de mapas densas. Este tipo de mapas, al contrario de los basados en características, obtienen una representación continua del espacio. Este proyecto pretende desarrollar algoritmos de seguimiento y localización para una cámara monocular basados en mapas densos. La idea principal es que utilizando la información densa, la localización va a ser más robusta en situaciones como las descritas anteriormente.

El proyecto ha propuesto dos algoritmos. El primero es un algoritmo de seguimiento basado en la minimización del error fotométrico entre la imagen y el mapa denso. El algoritmo utiliza la posición anterior de la cámara como semilla inicial y optimización jerárquica para evitar mínimos locales en los primeros pasos. El segundo algoritmo extiende el anterior a casos donde la semilla inicial no es conocida, usualmente conocido como el problema de la relocalización. En este caso, se ha propuesto un método mixto en dos pasos. Comienza con una aproximación grosera basada en características y después minimiza el error fotométrico con el mapa denso.

Ambos algoritmos han sido evaluados en dos *datasets*, uno de laboratorio y otro en un entorno de compra real. Se ha comparado la precisión del algoritmo con técnicas de *Bundle Ajustment*, que son el estado del arte en el problema de *structure from motion*. Los resultados muestran que la precisión obtenida con mapas densos en seguimiento es mejor que a la obtenida con un proceso de minimización sobre todas las imágenes. En cuanto a la relocalización, la estrategia híbrida permite encontrar una semilla suficientemente buena para correr la localización densa.

Índice general

Índice de figuras

Índice de cuadros

1. Introducción

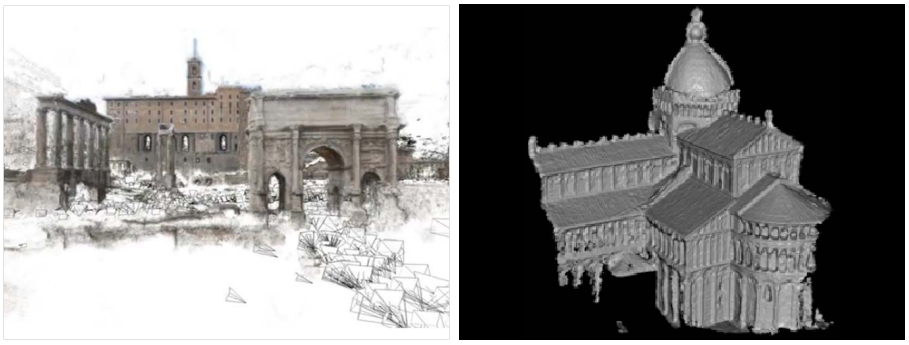
La Visión por Computador es el campo de la computación cuyo objetivo es conseguir que los ordenadores vean. Dicho de otra manera, trata de extraer información del mundo a través de las imágenes obtenidas por una cámara. La visión por computador cubre un amplio abanico de problemas y técnicas tales como el reconocimiento de patrones, la construcción de modelos del entorno, la identificación y seguimiento de objetos o la localización. Tiene aplicaciones muy diversas tanto en robótica, en imágen médica, en fotogrametría, reconocimiento facial, tareas de vigilancia entre otras muchas.

En particular, la localización de la cámara es un problema clásico de la Visión por Computador, que consiste en la estimación de su posición y orientación en el mundo. Su problema dual es la construcción de mapas de una escena ya que para la la construcción de mapas es necesario localizar la cámara y el problema de localización es relativamente más sencillo si se dispone de un mapa de la escena. La localización y los mapas son una parte fundamental en algunas de las aplicaciones mencionadas anteriormente. Por ejemplo, son ampliamente utilizadas en robótica para localizar un robot o un vehículo, aviones no tripulados tipo drones o en tareas de manipulación con brazos robóticos. También son la base de los sistemas de mapas ampliamente utilizados actualmente tipo Google Maps, o de sistemas de realidad aumentada como Oculus que han cobrado relevancia en los últimos tiempos. Otros ejemplos pueden ser la ayuda a personas con discapacidad o aplicaciones de publicidad personalizada basada en tu localización actual.

Tanto la localización de una cámara como la construcción de mapas se llevan a cabo utilizando sensores, como pueden ser láser, ultrasonidos, cámaras estéreo o cámaras monoculares. En la practica, se pueden utilizar combinaciones de varios sensores. En este trabajo pretendemos llevar a cabo la localización utilizando una cámara monocular como único elemento sensorial. Esto hace más difícil la localización ya que no se dispone de información de la profundidad. Sin embargo, el uso de cámaras monoculares está altamente extendido debido a su bajo tamaño, consumo y precio. Se pueden encontrar en muchos dispositivos de la vida cotidiana como cámaras web, cámaras de seguridad, teléfonos móviles, coches, etc.

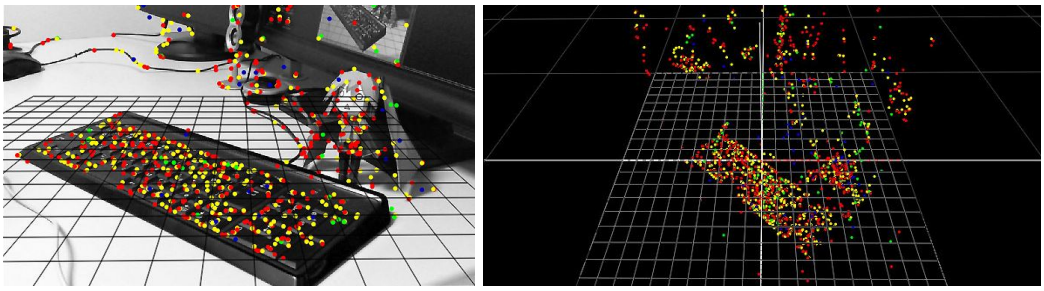
La combinación de la localización y los mapas es un problema conocido como *Structure from motion* (SfM), definido en [1]. SfM es un problema similar a *Simultaneous Localization And Mapping* (SLAM). SLAM es un problema que trata de construir incre-

mentalmente un mapa de la escena por la que se esta moviendo el robot, y localizar el robot en dicho mapa. SfM tiene los mismos objetivos, pero en SLAM se realiza la localización de cada imagen conforme se toma, actualizando con ella la información del mapa y SfM utiliza conjuntos de imágenes para reconstruir la escena. Cuando el problema de SLAM se resuelve utilizando una cámara monocular se conoce como *Monocular SLAM*. Algunos ejemplos de SfM se pueden ver en la figura 1.1.
Las técnicas empleadas para resolver SfM tratan de extraer la estructura de la escena a partir del movimiento realizado por una cámara monocular. Para reconstruir una escena y conocer la profundidad de sus puntos, se requieren diversas imágenes tomadas desde diferentes puntos de vista de los que se conoce su localización.



(a) (b)

Figura 1.1: Structure from motion. Imágenes utilizadas del libro [2]



(a) Puntos en la imagen (b) Puntos en el mapa

Figura 1.2: Métodos basados en puntos salientes

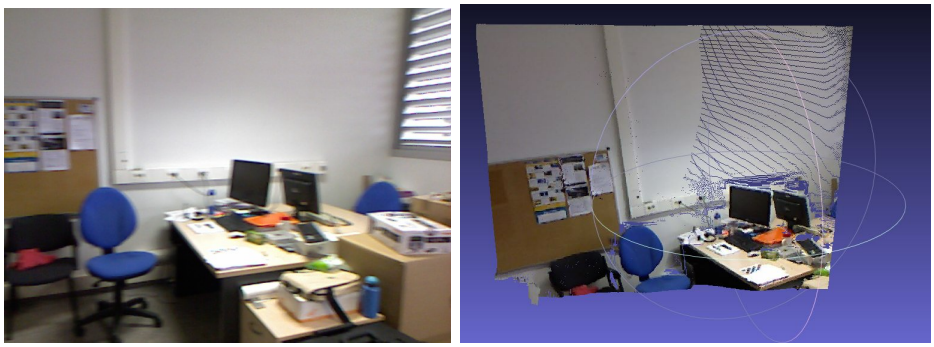
Los métodos de localización más utilizados son los que se basan en la extracción de puntos salientes en la imagen, como los presentados en [3]. Estos métodos, denominados *sparse*, localizan la cámara basándose en puntos característicos, ver figura 1.2. Los métodos *sparse* funcionan muy bien en la mayoría de los casos, aunque existen entornos en los que la extracción de puntos caracteristicos no es robusta. Esto provoca que la estimación de la localización no sea precisa e incluso se pueda perder. Un ejemplo de características que pueden hacer que un entorno dificulte la localización son los brillos o reflejos, ya que

(a) (b)

Figura 1.3: Imágenes con brillos y reflejos.

varían dependiendo del punto de vista y hacen que el emparejamiento de puntos no sea correcto o preciso, como ejemplo de un entorno real con estas características ver figura 1.3.

Una alternativa a los métodos *sparse* es la construcción de mapas densos. Los mapas densos no se basan en la extracción de puntos salientes sino que representan la escena de manera continua en el espacio. Por lo tanto son más robustos a entornos con características problemáticas como las descritas anteriormente ya que no son dependientes de la extracción de puntos en cada momento sino que utilizan toda la información de la escena. En los últimos años se han desarrollado algoritmos que permiten construir mapas densos de una escena de manera robusta como los presentados en [4] y [5]. Se puede ver dos ejemplos de mapas densos de una escena en la figura 1.1 y otro ejemplo de un mapa denso más pequeño que utilizaremos en nuestros experimentos en la figura 1.4.



(a) Imagen (b) Mapa denso

Figura 1.4: Métodos densos

Este trabajo pretende explorar la posibilidad de aprovechar estos mapas para mejorar la localización de una cámara monocular utilizando métodos densos en la localización.

Los métodos densos utilizan toda la información de la imagen y del mapa denso tridimensional. El objetivo del trabajo es poder llevar a cabo la localización de una cámara en entornos reales como los que se muestran en la figura 1.3. Para lograrlo, durante el trabajo llevado cuatro etapas entre las que se incluyen en desarrollo de dos algoritmos de localización que proponemos utilizando la información del mapas densos tridimensionales.

- La primera parte del trabajo ha consistido en el estudio del estado del arte de las tecnicas *sparse* de localización así como la construcción de mapas densos tridimensionales.
- El primer algoritmo que presentamos, es el seguimiento, consiste en realizar la localización de la cámara teniendo una semilla inicial de la posición y orientación cercana. Esto se produce por ejemplo cuando se trata de localizar una cámara en movimiento mientras graba un vídeo. Se puede utilizar la posición de la cámara en la toma de la imagen anterior.
- El segundo algoritmo que presentamos, es la relocalización, consiste en realizar la localización sin tener una semilla inicial de la posición y orientación cercana. Este algoritmo nos permite corregir los errores que se puedan producir durante el seguimiento
- La ultima etapa de nuestro trabajo ha consistido en la evaluación de los algoritmo propuestos en el trabajo. Para ello hemos realizado experimentos en entornos reales comparado nuestros algoritmos con el algoritmo *Bundle adjustment* que es el mejor algoritmo de localización utilizando tecnicas *sparse*.

En el siguiente capítulo, el 2, vamos a explicar más en detalle en qué consiste la localización de una cámara con mapa tridimensional de la escena. También se introducirán algunos conceptos necesarios para la comprensión de este trabajo, como son el modelo proyectivo y el álgebra de Lie. Más adelante en los capítulos 3 y 4, se explicarán en profundidad el seguimiento y la relocalización respectivamente, que son los dos problemas que buscamos resolver en este trabajo. También se explicarán los algoritmos que hemos desarrollado. Por ultimo en el capítulo 5 se presentan una serie de experimentos que hemos realizado para validar los algoritmos que presentamos, y para ver la mejora con respecto a los métodos *sparse*. Los experimentos se han llevado a cabo en entornos conocidos y controlados, así como en entornos reales.

2. Localización de una cámara con mapa conocido

En este capítulo se va a introducir en qué consiste la localización con mapa conocido en general. Más adelante en los capítulos 3 y 4 se explicará en detalle los dos tipos de localizaciones que se han estudiado en este trabajo.

La localización de una cámara consiste en estimar su posición y orientación (r_x, r_y, r_z, t_x, t_y, t_z) utilizando información sensorial. En en este trabajo utilizamos la imagen capturada por la cámara y el mapa tridimensional de la escena. Un esquema general de localización con respecto al mundo se puede ver en la figura 2.1. Nótese que aunque no se incluye en el esquema, la imagen capturada por la cámara puede contener ruido, producido por la representación en matriz de pixeles, y oclusiones, producidas por objetos en movimiento.

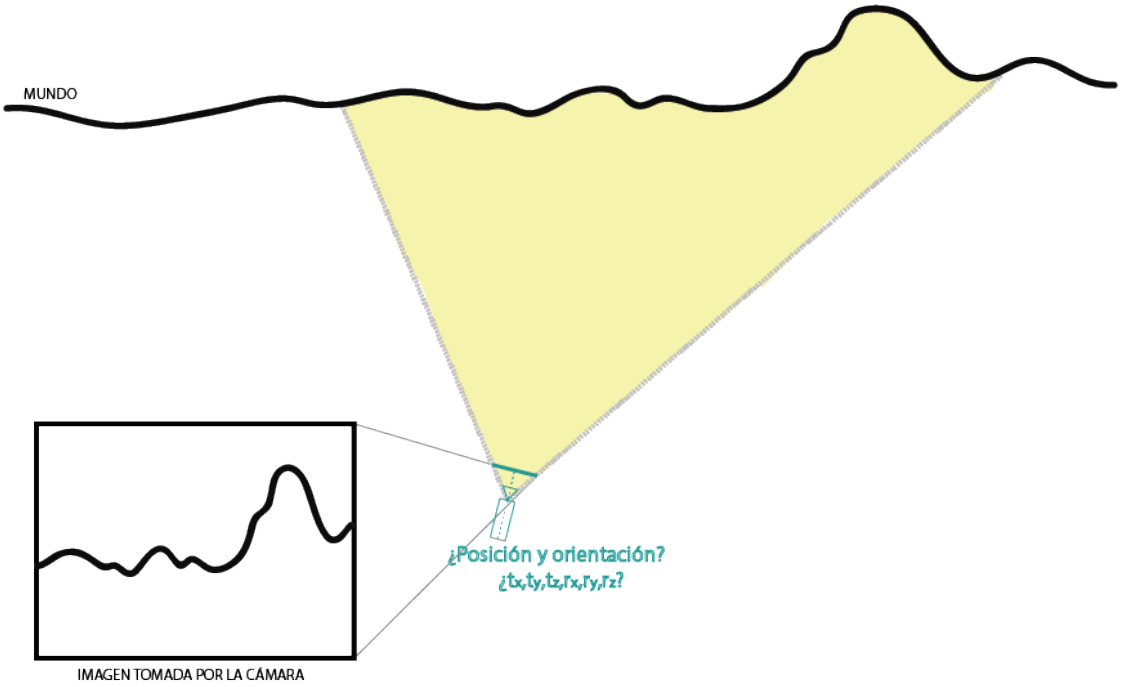


Figura 2.1: Localizacion de la cámara con respecto al mundo, se basa en lo que ha capturado en la imagen.

Existen dos tipos de técnicas para llevar a cabo la localización con mapa conocido, y en ambas varía el modo de representar el mapa tridimensional. La primera son los métodos basados en puntos salientes o métodos *sparse*. En los métodos *sparse* el mapa se compone de un conjunto de puntos característicos de la escena. Estos puntos característicos se componen de una posición en el espacio 3D y un descriptor (un vector de dimensión N). El descriptor del punto se utiliza para compararlo con otros puntos y determinar si son el mismo o no. La localización con métodos *sparse*, consiste en extraer los puntos característicos de la imagen y emparejarlos con los puntos del mapa. Los métodos *sparse* se basan en estos emparejamientos para estimar la posición. Se puede ver un esquema de localización utilizando métodos *sparse* en la figura 2.2 y un ejemplo de un mapa tridimensional de puntos característicos en la figura 1.2b.

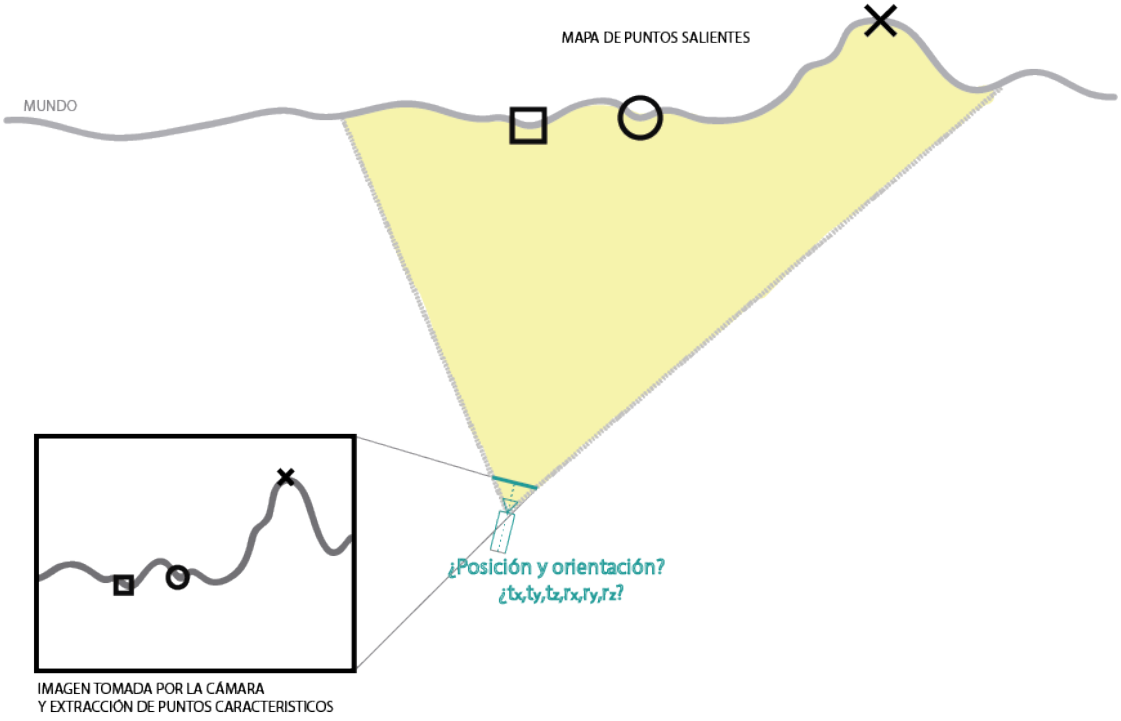


Figura 2.2: Localización de la cámara a un mapa de puntos salientes. Los descriptores de los puntos se representan con diferentes formas, en este caso un cuadrado, un circulo y una cruz.

El otro tipo de técnicas, los métodos densos, son en lo que nos hemos centrado en este trabajo. El mapa tridimensional denso no se centra en puntos característicos, sino que trata de representar toda la escena. Es un conjunto de puntos mayor, los cuales se componen de una posición en el espació 3D y un valor cromático (siendo la escala de grises una gama de color monocromática). Se puede ver un ejemplo de mapa denso tridimensional en la figura 1.4b. Los métodos densos utilizan toda la información de la imagen y del mapa denso tridimensional para llevar a cabo la localización. Son métodos

más robustos que los *sparse*, son menos sensibles a ruidos, cambios de iluminación y superficies sin textura. Se puede ver un esquema de localización con mapa denso conocido en la figura 2.3.

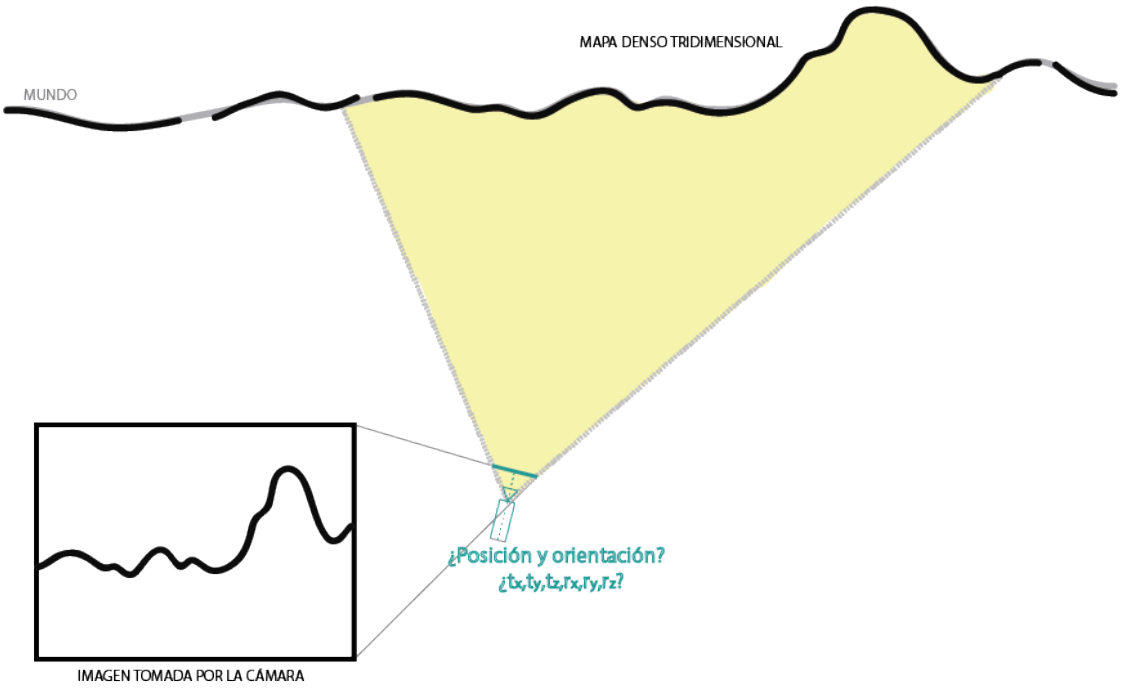


Figura 2.3: Localización de la cámara a un mapa de denso tridimensional. El mapa denso se dibujo encima del esquema del *mundo* para facilitar la representación.

Dada una representación, la localización con mapa conocido se resuelve mediante minimizando el error que supone proyectar la información del mapa en la imagen.

$$\{\hat{r}_x, \hat{r}_y, \hat{r}_z, \hat{t}_x, \hat{t}_y, \hat{t}_z\} = \underset{\{r_x, r_y, r_z, t_x, t_y, t_z\}}{\arg\min} \; e(M, I) \tag{2.1}$$

En la ecuación anterior, M representa la información del mapa tridimensional conocido e I representa la información de la imagen. $e(M, I)$ representa el error que supone la comparación entre el la información del mapa y la de la imagen dados unos parámetros de posición y rotación (r_x, r_y, r_z, t_x, t_y, t_z).

Para calcular el error entre M e I utilizamos un modelo proyectivo. Este modelo y la notación utilizada se explican en la siguiente sección 2.1. Además como veremos más adelante la parametrizamos de la rotación se realiza mediante una matriz 3×3 de rotación. Esta matriz presenta dificultades a la hora de realizar una minimización. Es por ello que nos vemos en la necesidad de utilizar otra forma de representar las rotaciones que son los coeficientes de Lie, los cuales se presentaran en la sección 2.2.

2.1. Modelo proyectivo y notación

La base principal del modelo de cámara que utilizamos es la proyección de 2D a 3D descrito en el modelo de cámara de Tsai [6]. Una representación esquemática del modelo se muestra en la figura 2.4.

El modelo de Tsai se compone de 6 parámetros extrínsecos y 5 parámetros intrínsecos. Los parámetros extrínsecos describen la localización en x, y y z de la cámara respecto a el sistema de coordenadas absoluto, que es el que usaremos como punto de referencia. Son 3 de rotación (r_x, r_y, r_z) y 3 de traslación (t_x, t_y, t_z). Los parámetros intrínsecos (f, C_x, C_y, s, K) dependen de la construcción interna de la cámara. Como suele ser habitual, para asegurar que los pixeles sean cuadrados, el parámetro s (*skew coefficient*), que determina el angulo entre los ejes x e y de los pixeles, ha sido fijado a 0, y el *aspect ratio* ha sido fijado a 1. El parámetro de distorsión $K = (k_1, ..., k_n)$ corrige la distorsión radial de la cámara. Aunque para facilitar la comprensión no se ha incluido en las formulas, todas las imágenes que se han utilizado en todos los experimentos han sido des-distorsionadas con anterioridad. El parámetro f es la distancia focal, que determina la posición del plano de la imagen con respecto al centro óptico de la cámara. Los parámetros C_x y C_y representan las coordenadas en el sensor de la cámara de la intersección con eje óptico (el centro óptico del sensor).

Los 5 parámetros intrínsecos de la cámara son constantes, necesitan ser estimados mediante tecnicas de calibración. En los algoritmos que presentamos se suponen conocidos. Por otro lado los 6 parámetros extrínsecos representan el localización de la cámara y por lo tanto varían con su movimiento.

Como se puede ver en la figura 2.4, en el modelo de Tsai la posición de la cámara en el espacio viene dada por la posición del centro optico, también se utilizará para nombrarlo O_c, como origen (un punto) del sistema de coordenadas de la cámara respecto al mundo:

$$O_c = (x_w, y_w, z_w). \tag{2.2}$$

La posición del centro óptico define la traslación que tiene la cámara respecto al sistema de referencia:

$$O_c = (t_x, t_y, t_z), \tag{2.3}$$

que coincide con la posición frontal de la cámara y el eje z_c de la cámara coincide con la dirección y el sentido del eje óptico de la cámara. El plano de la imagen es paralelo al plano generado por los vectores x_c e y_c del sistema de coordenadas de la cámara, esta a una distancia f de O_c en la dirección y sentido del eje z_c, donde f es la distancia focal.

La relación entre un punto definido en el sistema de coordenadas del mundo ($p_w = (x_w, y_w, z_w)$) y el mismo punto en el plano de la imagen de la cámara ($p_i = (x_i, y_i)$)

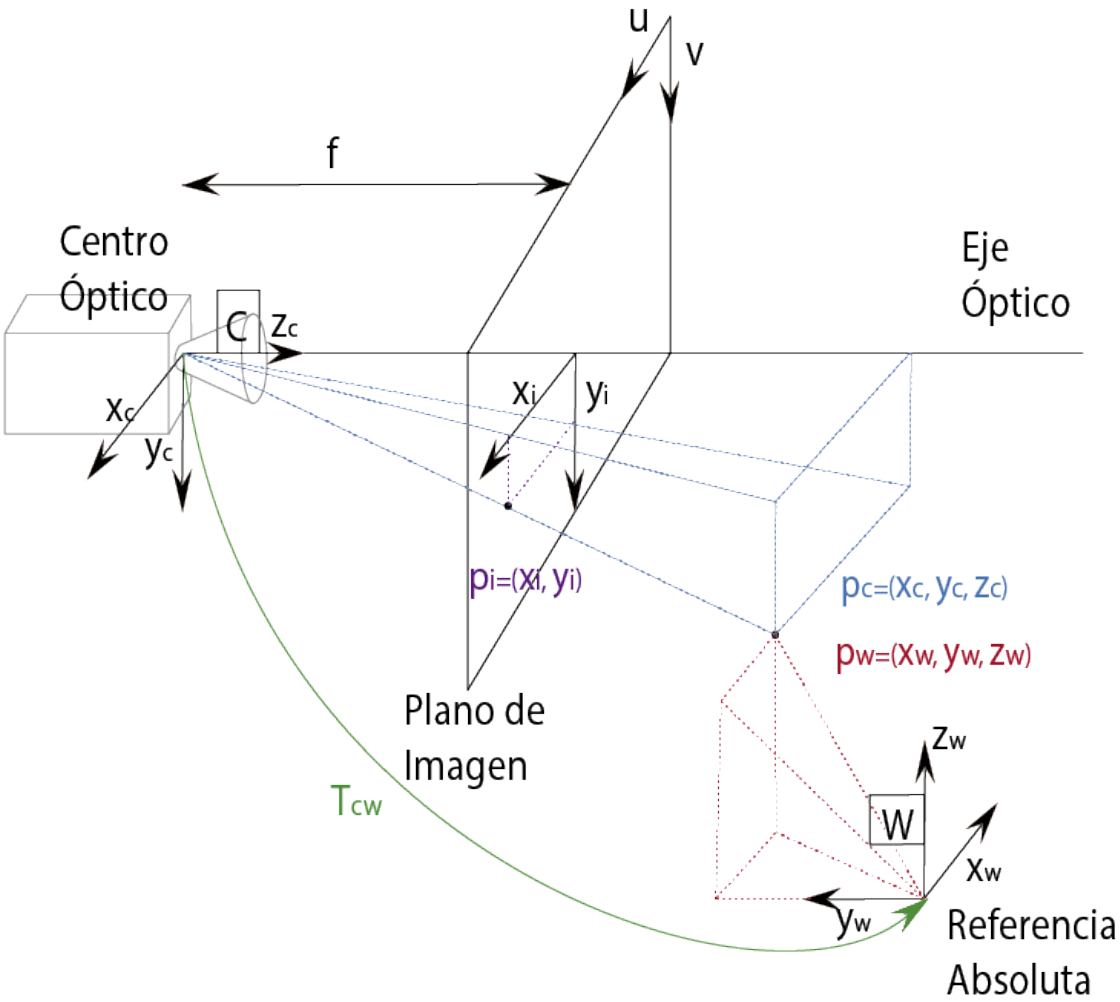


Figura 2.4: Modelo de cámara de Tsai

esta compuesta por una secuencia de transformaciones. Una transformación del punto de coordenadas del mundo a coordenadas de la cámara y una proyección del punto al plano de la cámara.

La primera consiste en transformar el punto p_w al sistema de coordenadas de la cámara ($p_c = (x_c, y_c, z_c)$):

$$p_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R_{cw} \cdot p_w + t_{cw} = R_{cw} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{2.4}$$

donde

$$R_{cw} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} = Rot(r_x)Rot(r_y)Rot(r_z), \tag{2.5}$$

siendo R_{cw} una matriz de rotación 3×3. Los vectores columna de una matriz de rotación representan las direcciones normalizadas de los ejes x, y y z del sistema de referencia al que ejerce la rotación sobre el sistema base a partir del cual la define. En el ejemplo de la ecuación 2.5 el vector $[r_1, r_4, r_5]'$ define la dirección del eje x del sistema de la cámara $_c$ con respecto al sistema de ejes absoluto $_w$.
Una matriz de rotación representa una rotación en el espacio euclídeo, no es una representación mínima ya que se representan 3 grados de libertad (r_x, r_y, r_z) mediante una matriz 3×3. Esta sobreparametrización se traduce en una serie de restricciones que una matriz de rotación debe cumplir:

- Es una matriz ortogonal, lo que significa que su matriz inversa es igual a su matriz transpuesta.
- Su determinante es igual a 1.

En adelante las transformaciones como la mostrada en la ecuación 2.4 se representarán como transformaciones homogéneas utilizando la siguiente notación:

$$\dot{p_c} = \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \omega x_c \\ \omega y_c \\ \omega z_c \\ \omega \end{bmatrix} = T_{cw} \cdot \dot{p_w}, \tag{2.6}$$

donde $\dot{p}$ es la representación homogénea de un punto en el espacio 3D. Las coordenadas homogéneas se utilizan para describir un punto en el espacio proyectivo. En el espacio 3D proyectivo la representación de un punto viene dada por un vector de cuatro elementos, siendo ω un factor de escala que generalmente fijaremos a 1. T_{cw} representa la matriz de transformación de la cámara respecto al mundo, para trabajar con ella es necesario utilizar coordenada homogéneas (ver figura 2.4):

$$T_{cw} = \begin{bmatrix} R_{cw} & t_{cw} \\ 0_{1\times 3} & 1 \end{bmatrix} \tag{2.7}$$

La segunda transformación es una proyección de perspectiva donde:

$$x_i = f\frac{x_c}{z_c} \qquad y_i = f\frac{y_c}{z_c}, \tag{2.8}$$

siendo x_i e y_i las coordenadas del punto en el plano de la imagen. Para representar esta transformación proyectiva utilizamos la notación: $p_i = \pi(\dot{p_c})$. Como resultado obtenemos que la transformación de un punto en el mundo al plano de la cámara es:

$$p_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \pi(\dot{p_c}) = \pi(T_{cw} \cdot \dot{p_w}) \tag{2.9}$$

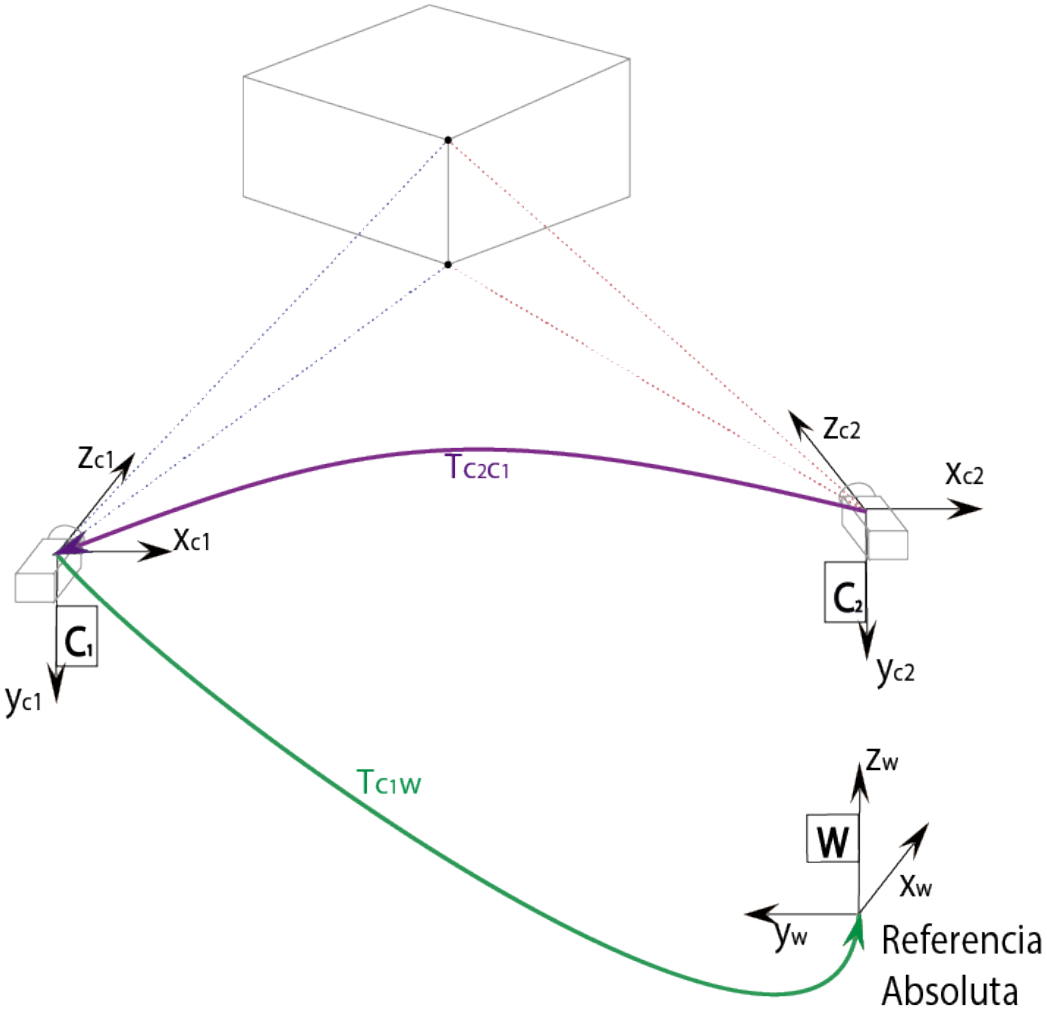


Figura 2.5: Composición de transformaciones

La composición de transformaciones nos permite pasar de unos sistemas de referencia a otros:

$$\dot{p_{c_2}} = T_{c_2c_1} T_{c_1w} \cdot \dot{p_w}. \tag{2.10}$$

Conocidas dos transformaciones como las que se muestran en la figura 2.5, podemos calcular la matriz que define la transformación entre la cámara C_2 y el mundo mediante una composición de transformaciones:

$$T_{c_2w} = T_{c_2c_1} T_{c_1w}. \tag{2.11}$$

La inversa de una matriz de transformación define una transformación inversa de manera que :

$$T_{c_1c_2} = T_{c_2c_1}^{-1} = \begin{bmatrix} R_{c_2c_1} & t_{c_2c_1} \\ 0_{1\times3} & 1 \end{bmatrix}^{-1} = \begin{bmatrix} R_{c_2c_1}^{-1} & -R_{c_2c_1}^{-1} \cdot t_{c_2c_1} \\ 0_{1\times3} & 1 \end{bmatrix} \tag{2.12}$$

donde

$$R_{c_2c_1}^{-1} = R_{c_2c_1}^{\top} \tag{2.13}$$

Utilizando estas técnicas de geometría podemos, dado un mapa de puntos tridimensional y la posición de una cámara determinar que puntos se proyectarían en el plano de imagen de la cámara y por lo tanto qué ve.

2.2. Álgebra de Lie

En la sección anterior hemos visto dos formas de representar las rotaciones: los ángulos de Euler y las matrices de rotación. Ambas dos presentan problemas a la hora de realizar una minimización sobre ellas, como la presentada en la ecuación 2.1. Los ángulos de Euler (r_x, r_y, r_z) no es una representación continua y las matrices de rotación, a pesar de ser una representación continua, ya hemos visto que están sujetas a una serie de restricciones que dificultan realizar una optimización. Por ello, en este proyecto se utiliza álgebra de Lie como modelo de representación de los parámetros extrínsecos de la cámara (r_x, r_y, r_z, t_x, t_y, t_z). Un grupo de Lie es continuo, es decir tanto sus elementos como la operación varían continuamente. Además esta representación vuelve a ser mínima.Como contra la representación mediante los coeficientes de Lie, sólo trabaja bien al rededor de la identidad, es decir pierden precisión conforme se alejan.

Nos hemos basado en el trabajo realizado en [7] que utiliza álgebra de Lie como método de representación de estos parámetros dado que las matrices de rotación 3×3 pertenecen al *Lie Special Orthogonal group* $SO(3)$. Este grupo es parametrizado alrededor de la identidad por un vector de tres coeficientes de Lie $\mathfrak{so}(3)$. Esta parametrización es localmente euclídea alrededor del 0.
En el álgebra de Lie se trabaja en el espacio tangente de O(3) y SO(3) mediante la definición de unos vectores base, llamados *generadores*. La definición tanto del espacio tangente como los generadores y las operaciones que explicamos en este apartado están mucho más detalladas en [8]. Dados $\omega = (\omega_1, \omega_2, \omega_3)$ como el vector de coeficientes de Lie y G_1, G_2, G_3 como los tres generadores de Lie definimos la siguiente ecuación.

$$[\omega]_\times = \sum_{i=1}^{3} (\omega_i G_i) \tag{2.14}$$

2.2.1. Mapeo exponencial sobre SO(3)

Para transformar los coeficientes de Lie en una matriz de rotación se usa el mapeo exponencial de manera que para un vector de coeficientes ω y un punto x la transformación $R \cdot x$ con $R = exp([\omega]_\times)$ rota el punto x sobre los ejes u_ω con un ángulo θ siendo $\theta = \|\omega\|_2$.

El mapeo exponencial se estima de una manera conocida como *Rodriguez Formula*:

$$exp([\omega]_\times) = L \begin{cases} I + [\omega]_\times + \frac{1}{2}[\omega]_\times^2 = I, & \text{para } (\theta \to 0) \\ I + \frac{\sin(\theta)}{\theta}[\omega]_\times + \frac{1-\cos(\theta)}{\theta^2}[\omega]_\times^2, & \text{sino} \end{cases} \quad \text{con } \theta = \|\omega\|_2 \tag{2.15}$$

2.2.2. Mapeo logarítmico sobre SO(3)

Se define la relación inversa del mapeo exponencial como el logaritmo de una matriz de manera que:

$$exp(log(\Omega)) = \Omega \quad \text{y} \quad log(exp(A)) = A \tag{2.16}$$

El logaritmo de matrices se define como:

$$log(R) = \begin{cases} \frac{1}{2}(R - R^{\mathrm{t}}) = 0, & \text{para } (d \to 1) \\ \frac{\arccos(d)}{2\sqrt{1-d^2}}(R - R^{\mathrm{t}}), & \text{para } (d \in (-1, 1)) \end{cases} \quad \text{con } d = \frac{1}{2}(\text{trace}(R) - 1) \tag{2.17}$$

2.2.3. Espacio Euclídeo Especial

SE(3) es un grupo que incorpora traslación y rotación en el espacio de tres dimensiones. La posición en este espacio viene definida por una rotación $R \in \mathbf{SO}(3)$ junto con una traslación $t \in \mathbb{R}^3$ y se expresa de la siguiente manera:

$$R_{ba}\mathrm{x}_a + t_{ba} = \mathrm{x}_b \tag{2.18}$$

Esta transformación consiste en trasladar un punto x_a en coordenadas de la cámara a a x_b en coordenadas de la cámara b. Los elementos de **SE**(3) a menudo son expresados como matrices de transformación 4×4.

$$\dot{x_b} = T_{ba}\dot{x_a} = \begin{pmatrix} x_b \\ 1 \end{pmatrix} = T_{ba} \begin{pmatrix} x_a \\ 1 \end{pmatrix} \quad \text{con } T_{ba} = \begin{pmatrix} R_{ba} & t_{ba} \\ 0_{1\times 3} & 1 \end{pmatrix} \tag{2.19}$$

El espacio tangente $\mathfrak{se}(3)$ es abarcado por los generadores $G_i = (\hat{e_i})_{\mathfrak{se}(3)}$ con e_i siendo en i-esimo vector Cartesiano unitario de $\mathbb{R}^6$ y $\hat{\cdot}_{\mathfrak{se}(3)}$,

$$\hat{\cdot}_{\mathfrak{se}(3)} : \mathbb{R}^6 \to \mathbb{R}^6, \quad \widehat{\begin{pmatrix} \nu \\ \omega \end{pmatrix}}_{\mathfrak{se}(3)} = \begin{pmatrix} [\omega]_\times & \nu \\ 0_{1\times 3} & 0 \end{pmatrix} \tag{2.20}$$

del mismo modo definimos un mapeo exponencial para pasar de coeficientes del espacio tangente a una matriz de transformación en el espacio euclídeo,

$$exp(\nu, \omega)_{\mathfrak{se}(3)} = exp\left(\widehat{\begin{pmatrix} \nu \\ \omega \end{pmatrix}}\right) = \begin{pmatrix} exp([\omega]_\times) & V\nu \\ 0 & 1 \end{pmatrix} \in \mathbf{SE}(3), \tag{2.21}$$

con

$$V = \begin{cases} I + [\omega]_\times + \frac{1}{6}[\omega]_\times^2 = I, & \text{para } (\theta \to 0) \\ I + \frac{1-\cos(\theta)}{\theta^2}[\omega]_\times + \frac{\theta-\sin(\theta)}{\theta^3}[\omega]_\times^2, & \text{sino} \end{cases} \quad \text{con } \theta = \|\omega\|_2 \tag{2.22}$$

3. Seguimiento

Este capitulo presenta el algoritmo diseñado para el problema de seguimiento de una cámara sobre un mapa denso. Para ello primero se va a explicar cual es el problema, después se explicarán los algoritmos basados en puntos salientes que se suelen utilizar para resolverlo y por ultimo el algoritmo que proponemos.

El seguimiento consiste en estimar la posición y orientación de una cámara en movimiento a partir de una secuencia de imágenes tomadas secuencialmente. Es un caso particular del problema de localización, ver capítulo 2. El seguimiento se resuelve estimando la localización de cada imagen en el orden en que se tomaron, de esta manera se puede aprovechar el conocimiento de las posiciones anteriores de la cámara. Se suele estimar el modelo de movimiento seguido por la cámara mediante filtros de Kalman y usarlo como semilla inicial en la minimización.

En nuestro caso la secuencia de imágenes se han tomado en vídeo a tiempo real, por lo que nuestro *frame rate* es muy pequeño (30 ms). Por lo tanto al localizar una secuencia de imágenes no es necesario procesar el modelo de movimiento para calcular una semilla inicial para la minimización. Utilizamos directamente la posición de la cámara en el momento de tomar la imagen anterior a la actual, y a partir de ahí realizamos la minimización.

Hemos desarrollado un algoritmo de minimización para llevar a cabo la localización en seguimiento de una cámara monocular basado en mapa denso, el esquema genera del minimización se ha explicado en la ecuación 2.1. En las siguientes secciones se detallará esta ecuación para las dos diferentes técnicas mencionadas en el capítulo 2.

3.1. Método de seguimiento

El método de seguimiento que hemos seguido se basa en estimar en orden la posición de la cámara en la toma de cada uno de los *frames* del vídeo. Para cada *frame* se realiza una minimización del error de proyección y se estima su posición basandose en la estimación del *frame* anterior.

A continuación se van a presentar dos tipos de minimizaciones, la minimización ba-

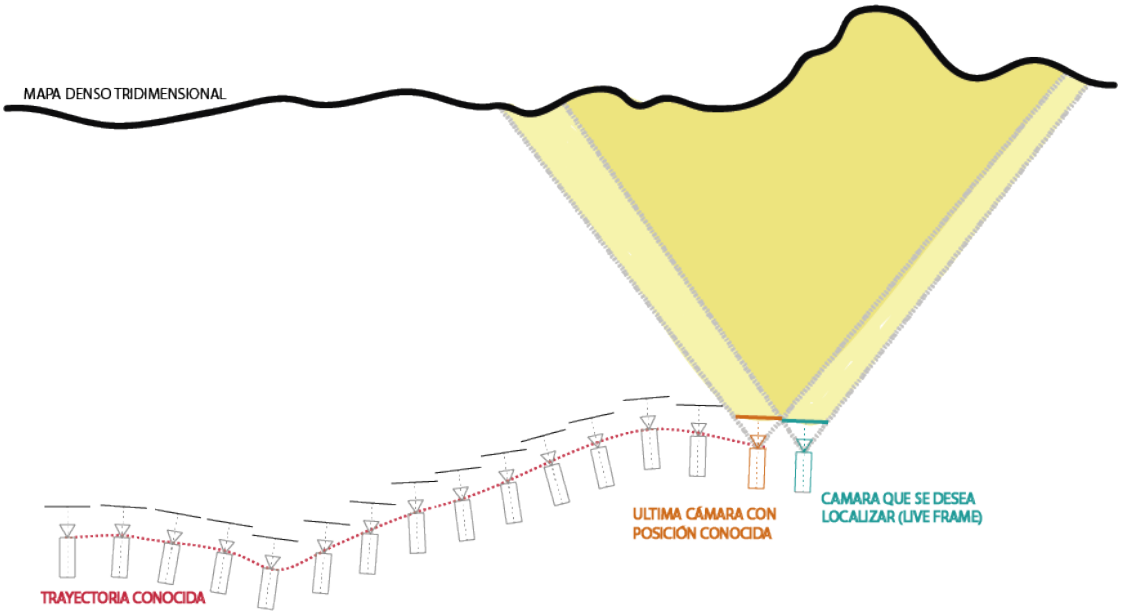


Figura 3.1: Seguimiento de una cámara, utilizando un mapa denso

sada en puntos salientes y la minimización densa. La minimización basada en puntos salientes es la utilizada en los métodos *sparse*. La minimización densa es la que se utiliza en los métodos densos y es la que hemos utilizado en este trabajo. Para desarrollar la minimización densa nos hemos basado en el trabajo realizado en [5].

3.1.1. Minimización basada en puntos salientes

Los métodos basados en puntos salientes definen un mapa tridimensional como un conjunto de características salientes extraídas a partir de las imágenes con las que se construyo el mapa, ver figura 1.2b. De cada punto característico conocemos la posición en el espacio y un descriptor. El descriptor lo utilizamos para emparejarlos con los puntos característicos extraídos de la imagen que queremos localizar, definida como I_l o *live-frame*. Ver el esquema de la figura 2.2 en el capítulo 2 donde esto se explica.
Para realizar la minimización lo primero que hemos hecho es definir una medida de error, a partir de la cual basarnos al elegir una solución correcta. Cada punto característico viene definido por unas coordenadas, ya sean en la imagen o en el espacio 3D del mapa denso, y un descriptor, que se utiliza para comparar diferentes puntos característicos.

Sea P_w el conjunto de puntos característicos del mapa y sea Z_i el conjunto de puntos característicos extraídos de la imagen. Hallar los subconjuntos de puntos emparejados entre si, $P_w^* \subseteq P_w$ y $Z_i^* \subseteq Z_i$, que son el subconjunto de puntos del mapa y el subconjunto de puntos en la imagen respectivamente. Para emparejarlos entre si se utilizan los

descriptores de cada punto. Por lo que para cada punto en P_w^* hay un correspondiente único en Z_i^* lo que implica que $|P_w^*| = |Z_i^*|$. Una vez encontrados los emparejamientos la función de coste, que cuantifica el error de la posición y rotación estimada de la cámara sería la siguiente:

$$\epsilon = \sum_{k \in \{1:|P_w^*|\}} d(Z_i^*[k], \pi(T_{cw} \cdot \dot{P}_w^*[k])) \tag{3.1}$$

definiendo $d(p, q)$ como la distancia entre los puntos p y q:

$$d(p, q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2} \tag{3.2}$$

En la ecuación 3.1 $Z_i^*[k]$ es la representación del k-esimo punto emparejado de I_l en coordenadas de la imagen representado como vector 2×1 y $P_w^*[k]$ un vector 3×1 representando las coordenadas en el mundo del punto en el mapa, $P_w^*[k]$ es la representación homogénea de $P_w^*[k]$ por lo que es un vector 4×1. Es decir la ecuación 3.1 suma la distancia de cada punto característico en la imagen con la proyección de su emparejado en el mundo.
Estos métodos pretenden minimizar el error de re-proyección (ϵ) de los puntos del mapa en la imagen, por lo que buscan los parámetros de rotación y traslación de la cámara que lo minimizan, esta es la función de localización especifica para métodos *sparse*, es una especialización de 2.1:

$$\{\hat{r}_{lw}, \hat{t}_{lw}\} = \underset{\{r_{lw}, t_{lw}\}}{\arg\min} \ \epsilon = \underset{\{r_{lw}, t_{lw}\}}{\arg\min} \sum_{j \in \{1:|P_w^*|\}} d(Z_i^*[j], \pi(T_{lw} \cdot \dot{P}_w^*[j])) \tag{3.3}$$

con

$$T_{lw} = \begin{bmatrix} R(r_{lw}) & t_{lw} \\ 0_{1\times 3} & 1 \end{bmatrix} \tag{3.4}$$

La evolución de un sistema de optimización que trata de realizar esta minimización se muestra en la figura 3.2.
En la practica se aprovecha la rotación y traslación de la cámara T_{kw} al tomar la imagen anterior a la que se desea localizar que denominaremos *key-frame* o I_k y trataremos de estimar la traslación y posición de la segunda cámara con respecto a la primera T_{lk}:

$$\{\hat{r}_{lk}, \hat{t}_{lk}\} = \underset{\{r_{lk}, t_{lk}\}}{\arg\min} \ \epsilon = \underset{\{r_{lk}, t_{lk}\}}{\arg\min} \sum_{j \in \{1:|P_w^*|\}} d(Z_i^*[j], \pi(T_{lk} \cdot T_{kw} \cdot \dot{P}_w^*[j])) \tag{3.5}$$

esto nos permite poder utilizar el sistema de movimiento que esta siguiendo la cámara.

3.1.2. Minimización densa

En esta sección se va a presentar la especialización de la minimización general de localización, ver ecuación 2.1, para el caso de los métodos densos. Los metodos densos se han explicado en el capítulo 2. Utilizan un mapa tridimensional denso de la escena y

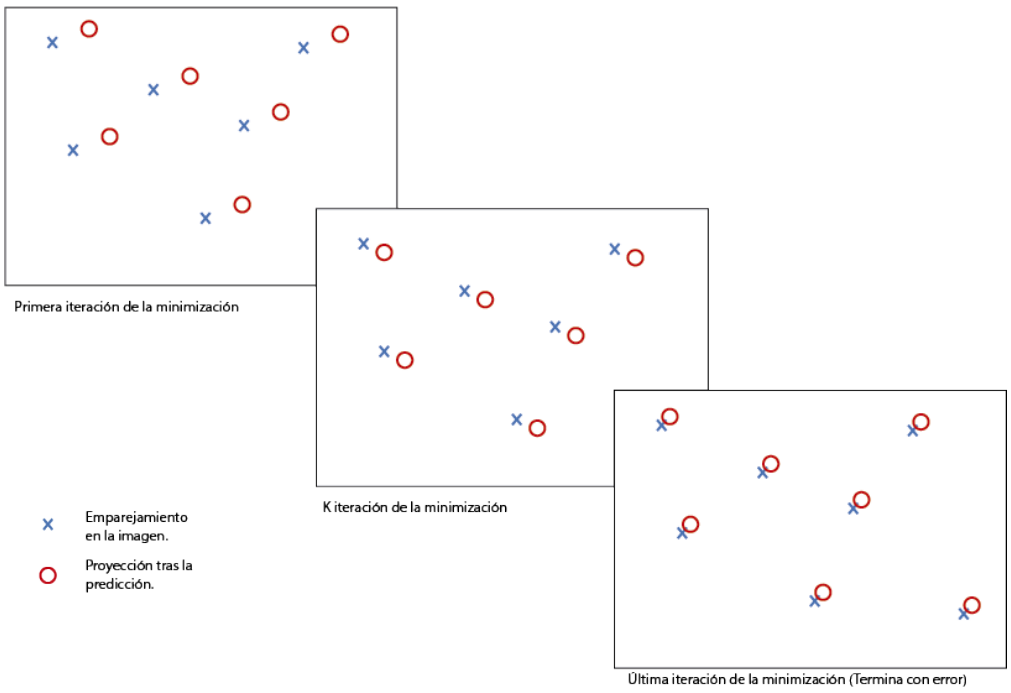


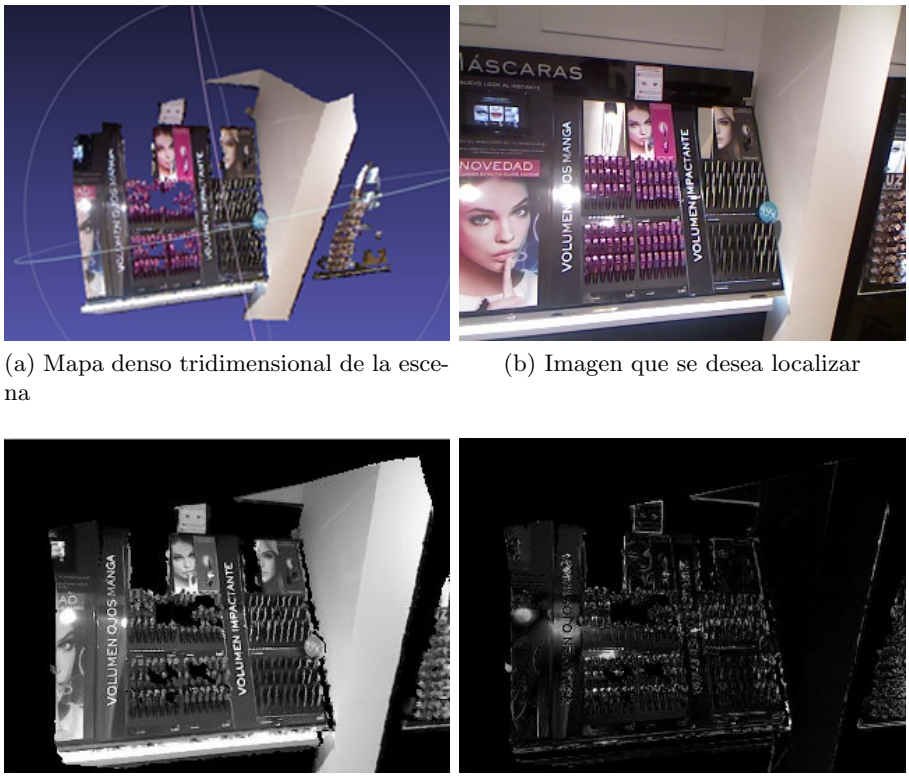
Figura 3.2: Localización de una cámara, minimización *sparse*

aprovechan toda la información de la imagen. Un ejemplo de mapa denso, se puede ver en la figura 1.4b.

En los métodos densos en lugar de utilizar el error de re-proyección de puntos característicos se realiza una proyección de los puntos del mapa en la posición de la cámara creando una imagen virtual que podemos comparar con la imagen real y restando las dos imágenes obtenemos un error fotométrico de la estimación (ver figura 3.3).

Recordamos que I_l (*live frame*) es la imagen de la cual queremos conocer la rotación y traslación. Definimos $\Omega_l \in \mathbb{R}^2$ como el conjunto de coordenadas de los pixeles de la imagen I_l. Si $p \in \Omega_l$, $I_l(p)$ corresponde al pixel del *live frame* con coordenadas p. En una imagen 200×200, el conjunto de coordenadas Ω esta compuesto por todos los puntos $p = (u, v)^t$ con $u, v \in [1, \dots, 200]$.

Ahora definimos M como el mapa de puntos en 3D y $\Psi \in \mathbb{R}^3$ el conjunto de coordenadas del mapa de puntos. Si $x_c \in \Psi$ $(x_c = (x,\ y,\ z)^t)$, $M(x_c)$ representa un punto de la nube de puntos. Esta notación nos permite acceder a los puntos del mapa indexandolos $(M(x)\ )$.



(a) Mapa denso tridimensional de la escena

(b) Imagen que se desea localizar

(c) Proyección del mapa en la posición y rotación estimada

(d) Error fotometrico de la estimación (ϵ)

Figura 3.3: Calculo del error fotométrico de una estimación densa

Para poder realizar una optimización definimos primero la función de coste a minimizar, que en métodos densos se suele llamar diferencia de *energía* (ϵ):

$$\epsilon = \frac{1}{2} \sum_{x_i=\Psi} [M(x_i) - I_l(f \cdot \pi(T_{lk} \cdot T_{kw} \cdot \dot{x_i}))]^2 \tag{3.6}$$

siendo f la distancia focal de la cámara y T_{kw} la **semilla inicial** que en nuestro caso es la transformación que define la posición y orientación de la cámara en el instante anterior. Para asegurar la convergencia de la optimización es necesario contar con una semilla inicial cercana al mínimo global ya que la función de coste no es lineal y tiene muchos mínimos locales.

En este trabajo parametrizamos las actualizaciones de T_{lk} por $\psi \in \mathbb{R}^6$ que pertenece a los coeficientes del álgebra de Lie, mencionada en la sección 2.1, tal que $\psi = \{\omega, \nu\}$ y

por lo cual nuestro sistema de optimización es:

$$\{\hat{\psi}\} = \arg\min_{\{\psi\}} \frac{1}{2} \sum_{x_i=\Psi} \left[M(x_i) - I_l(f \cdot \pi(T_{lk} \cdot T_{kw} \cdot \dot{x_i}))\right]^2 \tag{3.7}$$

$$T_{lk} = exp\left(\sum_{i=1}^{6} \psi_i \underset{\mathbb{SE}(3)}{\mathrm{G}_i}\right) = \begin{pmatrix} exp([\omega]_\times) & V\nu \\ 0 & 1 \end{pmatrix} \in \mathbb{SE}(3) \tag{3.8}$$

Para calcular el valor de $I_l(f \cdot \pi(T_{lk} \cdot T_{kw} \cdot \dot{x_i}))$ dado que las coordenadas no tienen por que ser enteras (coordenadas en pixeles) se ha realizado una interpolación lineal de de los pixeles alrededor.

Para realizar la minimización 3.7 se puede pueden utilizar diferentes optimizadores. En nuestro caso dado que la función de error no es lineal, hemos utilizado un optimizador de errores mínimos cuadrados para sistemas no lineales. Concretamente hemos utilizado la función *lsqnonlin* de Matlab, con el algoritmo Levenberg-Marquardt.

Optimización en dos fases: En la implementación se han añadido algunas optimizaciones para hacer que la ejecución sea más eficiente, una de ellas es realizar con antelación una primera optimización de únicamente los parámetros de rotación.
Hacer una primera estimación de solamente la rotación consigue mejorar la semilla para la siguiente dado que la rotación de la cámara es un factor que genera cambios en el plano de imagen de manera más rápida que la traslación, hacer esta primera optimización reduce el número de iteraciones en la segunda.
Esta estimación no es definitiva, después de hacerla, se vuelve a estimar junto con la traslación en la minimización que se muestra en la ecuación 3.7, dado que es una estimación de sólo la rotación aun cuando ha podido haber traslación. Por esta razón no requerimos de la nube de puntos y reducimos la resolución de la imagen. Se realiza la estimación con respecto al *frame* de la semilla inicial reduciendo en ambas imágenes la resolución en dos órdenes de magnitud. Esto nos permite, además de realizar una aproximación mucho más eficiente, eliminar detalles pequeños de la imagen que podrían empeorar el resultado.

Dada una matriz de rotación R_{ba} definimos la homografía H_{ba} que traslada los puntos de la imagen a a la imagen b como

$$H_{ba} = \left(K R_{ba} K^{-1}\right) \tag{3.9}$$

donde K es la matriz 3×3 de calibración intrínseca de la cámara

$$K = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{3.10}$$

por lo que un punto en la imagen a es trasladado al sistema de coordenadas de la imagen b como

$$\dot{p_b} = H_{ba} \cdot \dot{p_a} \tag{3.11}$$

El parámetro que queremos estimar es $r \in \mathbb{R}^3$ que define la rotación en tres dimensiones, y $R(r)$ es la matriz de rotación para los parámetros de rotación r. La minimización a realizar por lo tanto es

$$\hat{r} = \arg\min_{r} \frac{1}{2} \sum_{p_i=\Omega_k} [I_l(H_{lk} \cdot \dot{p_i}) - I_k(p_i)]^2 \quad \text{con } H_{lk} = KR(r)K^{-1} \tag{3.12}$$

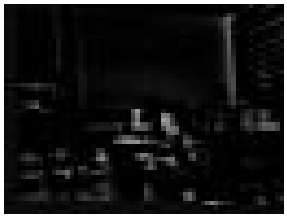
dado que es la minimización de una función no lineal, es necesario contar con una semilla cercana al mínimo global. Es por eso que supone que las imágenes han sido tomadas consecuentemente a modo vídeo y que por lo tanto la rotación entre una imagen y la siguiente es mínima.
Para hacer esta estimación finalmente se usaron como parámetros para la optimización los coeficientes del álgebra de Lie ω, explicados en la sección 2.1. Transformando la minimización en

$$\hat{\omega} = \arg\min_{\omega} \frac{1}{2} \sum_{p_i=\Omega_k} [I_l(H_{lk} \cdot p_i) - I_k(p_i)]^2 \quad \text{con } H_{lk} = K \; exp([\omega]_\times) \; K^{-1} \tag{3.13}$$

Para que el error sea una función continua es necesario interpolar, dado que las coordenadas del punto $H_{lk} \cdot \dot{p_i}$ no tienen por que ser valores enteros y por lo tanto $I_l(H_{lk} \cdot \dot{p_i})$ no es el valor de un único pixel sino la interpolación de varios. En este caso hemos utilizado una interpolación lineal.
Se muestra en la figura 3.4 la evolución del error en cada iteración de la minimización.

Optimización jerárquica: Otra de las mejoras que se han realizado es la optimización del error fotométrico en diferentes niveles de la pirámide (ver figura 3.5), empezando por una imagen reducida 8 veces la original. La mejora aportada es que en los niveles más altos de la pirámide los detalles de la escena no pesan en el error lo que hace la estimación menos sensible a la semilla inicial, además es mucho más rápida ya que el número de pixeles es mucho menor. Por ultimo se aprovecha la estimación de un nivel de la pirámide como semilla para la estimación en el siguiente lo que agiliza la optimización y reduce la probabilidad de terminar en un mínimo local, más adelante en la sección 5.3.4 se muestran las ventajas de esta mejora. Esta técnica de optimización es la utilizada en [9].

(a) Imagen original, de la cual se quiere estimar la rotación



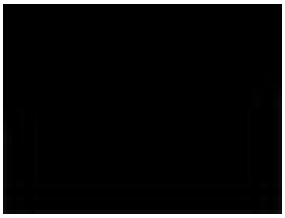
(b) Error inicial

(c) Primera iteración

(d) Segunda iteración

(e) Tercera iteración

(f) Cuarta iteración



(g) Quinta iteración

Figura 3.4: Evolución del error al estimar la rotación.

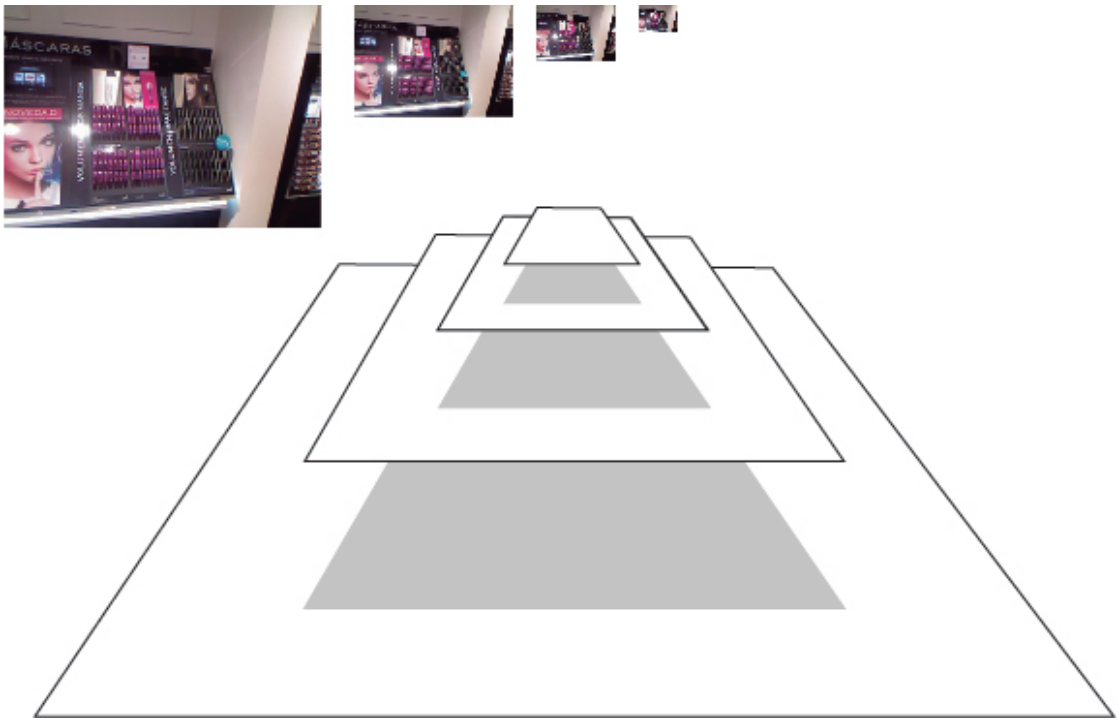


Figura 3.5: Pirámide

4. Relocalización

Como ya hemos explicado el seguimiento es un caso particular de Localización que aprovecha los conocimientos precisos sobre la posición de la cámara en los instantes anteriores. No obstante existen casos en los que se puede perder precisión en el conocimiento sobre estos datos o no disponer de ellos. Por ejemplo si se produce un movimiento brusco de la cámara o el objetivo deja de capturar la escena durante un intervalo de tiempo. La interrupción en la visión de la escena se puede dar si un elemento móvil que no forma parte de la escena interrumpe la visión momentáneamente. Se puede ver en la figura 4.2 un ejemplo de movimiento brusco que ocasiona la perdida de la posición de la cámara.

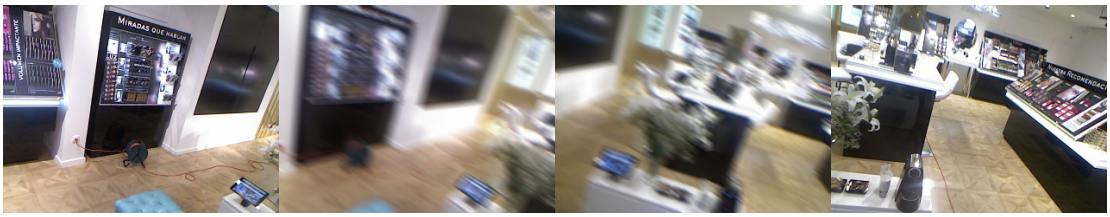


Figura 4.1: Las imagenes de izquierda a derecha en el orden en que se tomaron, en la primera la localización es conocida pero durante la segunda y la tercera el movimiento es muy brusco y la localización continua se pierde, en la cuarta imagen el algoritmo se ha perdido y requiere re-localizarse.

La relocalización es otro caso de localización más general que el seguimiento. En este caso no podemos suponer el conocimiento de una trayectoria anterior o un modelo de movimiento seguido por la cámara, es decir no disponemos de una semilla inicial cercana. En este problema nuestros datos son la imagen que queremos localizar (relocalizar), el mapa de puntos tridimensional denso y un conjunto de imágenes de las cuales estamos seguros de conocer su posición, este ultimo es en general el conjunto de imágenes con las que se llevo a cabo la construcción del mapa tridimensional, ver figura 4.2. La definición del mapa denso tridimensional se ha explicado el capítulo 2.

En este caso no se han utilizado, ya que no se conocen, métodos densos que resuelvan el problema directamente. En su lugar hemos utilizado una combinación de los métodos basados en puntos salientes y los métodos densos. El algoritmo que presentamos en

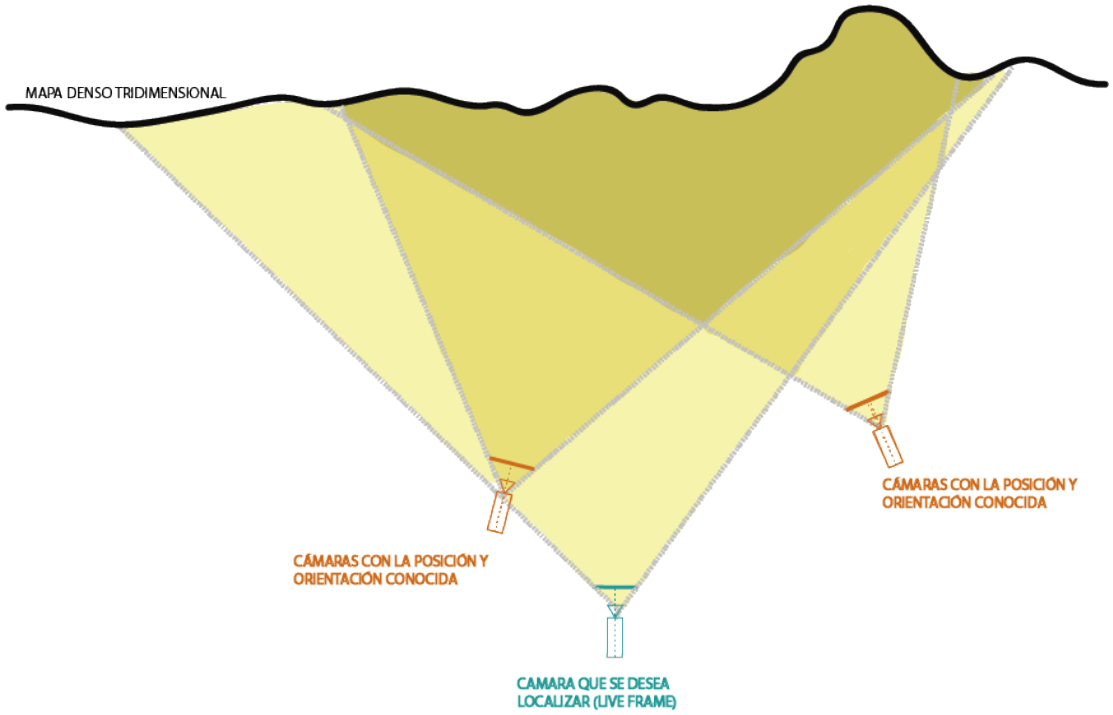


Figura 4.2: Esquema de relocalización

este capitulo calcula una semilla inicial utilizando métodos basados en puntos salientes, para a partir de ella realizar una minimización densa. Para el calculo de la semilla inicial se han utilizado la ideas presentadas en [10]. Realizamos emparejamientos de puntos característicos entre las imágenes para sacar información sobre la posición entre ambas cámaras utilizando el algoritmo de los cinco puntos (*Five-Point Relative Pose Algorithm*). En ese mismo estudio se muestra que el algoritmo de los cinco puntos ofrece resultados más consistentes que el de siete u ocho puntos, y esa es la razón por la que finalmente hemos utilizado este algoritmo.
Una vez hallada la posición mediante el algoritmo de los cinco puntos, utilizamos el resultado como semilla para llevar a cabo una minimización densa, presentada en la ecuación 3.7.

Antes de continuar con la explicación de la relocalización llevada a cabo en este trabajo se van a presentar algunos conocimientos geométricos necesarios para llevarla a cabo, como son la matriz esencial y cómo funciona el algoritmo de los cinco puntos.

4.1. Matriz Esencial

La matriz esencial es una especialización de la matriz fundamental en el caso de que las coordenadas de la imagen hayan sido normalizadas, toda la teoría relacionada con

la matriz esencial puede ser encontrada en el libro [11]. La matriz fundamental F es una forma resumida de describir la geometría en dos vistas, la transformación entre una cámara y otra.

Coordenadas normalizadas. Definimos x como las coordenadas de un punto X en la imagen y $\hat{x}$ como las coordenadas normalizadas del punto en la imagen. Si conocemos la matriz de calibración de la cámara K las coordenadas normalizadas se calculan aplicando la inversa a las coordenadas del punto $\hat{x} = K^{-1}x$.

Dadas dos imágenes de la misma escena desde dos puntos de vista diferentes I_1 e I_2, y un punto Q de la escena cuyas coordenadas en la imagen son q y q' respectivamente, si la matriz esencial que representa el movimiento de la cámara de una imagen a otra es E tenemos que:

$$q'^{\top} F q = \hat{q}'^{\top} K_2^{-\top} F K_1^{-1} \hat{q} = \hat{q}'^{\top} E \hat{q} = 0 \tag{4.1}$$

donde

$$q \equiv \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^{\top} \text{ y } \quad q' \equiv \begin{bmatrix} q'_1 & q'_2 & q'_3 \end{bmatrix}^{\top}. \tag{4.2}$$

La matriz esencial tiene solamente cinco grados de libertad: tres de rotación y tres de traslación que se quedan en cinco debido a una ambigüedad global en la escala. Un número tan reducido de grados de libertad se traduce en una serie de restricciones que la matriz esencial ha de cumplir:

- Una matriz 3×3 real no nula F, es una matriz fundamental si y sólo si satisface la ecuación
$$det(F) = 0. \tag{4.3}$$
- Una matriz 3×3 es una matriz esencial si y sólo si dos de sus valores singulares son iguales y el tercero es cero.
- Una matriz 3×3 real no nula E, es una matriz esencial si y sólo si satisface la ecuación
$$EE^{\top}E - \frac{1}{2} trace(EE^{\top})E = 0. \tag{4.4}$$

La matriz esencial puede ser calculada directamente a partir de 4.1 usando coordenadas normalizadas y el algoritmo de los cinco puntos. Se requieren cinco puntos para derivar la matriz esencial ya que tiene seis grados de libertad (tres de rotación y tres de traslación), que se quedan en cinco por el invariante en la escala. Una vez sea conocida podemos extraer las matrices de la cámara a partir de E, para más detalles consultar la sección 9.6.2 del libro de Hartley y Zisserman [11].

4.2. Algoritmo de los cinco puntos

En grandes rasgos el algoritmo presentado en [12] consiste en emparejar cinco puntos de las imágenes y sacar la matriz esencial a partir de estos emparejamientos utilizando la restricción 4.1, además las ecuaciones 4.3 y 4.4 nos ayudaran a lograrlo y , una vez sea conocida, tanto R como t pueden ser recuperadas de ella. La restricción 4.1 también puede ser escrita como:

$$\tilde{q}^\top \tilde{E} = 0, \tag{4.5}$$

donde

$$\tilde{q} \equiv \begin{bmatrix} q_1 q'_1 & q_2 q'_1 & q_3 q'_1 & q_1 q'_2 & q_2 q'_2 & q_3 q'_2 & q_1 q'_3 & q_2 q'_3 & q_3 q'_3 \end{bmatrix}^\top \tag{4.6}$$

y

$$\tilde{E} \equiv \begin{bmatrix} E_{11} & E_{12} & E_{13} & E_{21} & E_{22} & E_{23} & E_{31} & E_{32} & E_{33} \end{bmatrix}^\top . \tag{4.7}$$

La definición de los puntos q y q' esta en la ecuación 4.2.

Apilando los vectores $\tilde{q}^\top$ de los cinco puntos obtenemos una matriz 5×9 a la que llamamos Q. Si hallamos los vectores del espacio nulo (*nullspace*) de Q obtenemos los vectores $\tilde{X}$, $\tilde{Y}$, $\tilde{Z}$ y $\tilde{W}$. Un vector del espacio nulo V de una matriz A cumple:

$$A \cdot V = 0 \tag{4.8}$$

Los cuatro vectores corresponden cada uno a una matriz 3×3 X, Y, Z y W, donde se cumple que la matriz esencial E es una combinación lineal de estas 4 matrices como

$$E = xX + yY + zW + wW, \tag{4.9}$$

donde los escalares x, y, z y w están definidos bajo un mismo factor de escala por lo que podemos asumir que $w = 1$. Notar que el algoritmo puede ser extendido a más de 5 puntos.

Sabiendo que la matriz esencial debe ser construida a partir de la ecuación 4.9 y dadas las restricciones 4.3 y 4.4 obtenemos un sistema de 10 ecuaciones, enumeradas de

la a a la j en la siguiente ecuación, tras haber aplicado la eliminación Gauss-Jordan:

$$\begin{pmatrix} a: & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & [2] & [2] & [3] \\ b: & & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & [2] & [2] & [3] \\ c: & & & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & [2] & [2] & [3] \\ d: & & & & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & [2] & [2] & [3] \\ e: & & & & & 1 & & & & & & [2] & [2] & [3] \\ f: & & & & & & 1 & & & & & [2] & [2] & [3] \\ g: & & & & & & & 1 & & & & [2] & [2] & [3] \\ h: & & & & & & & & 1 & & & [2] & [2] & [3] \\ i: & & & & & & & & & 1 & & [2] & [2] & [3] \\ j: & & & & & & & & & & 1 & [2] & [2] & [3] \end{pmatrix} \begin{pmatrix} x^3 \\ y^3 \\ x^2y \\ xy^2 \\ x^2z \\ x^2 \\ y^2z \\ y^2 \\ xyz \\ xy \\ x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{4.10}$$

donde $\cdot$ denota un valor escalar y $[N]$ denota un polinomio de grado N en la variable z. Notar que en las ecuaciones de la e a la j son casi nulas, solo tienen cuatro elementos no nulos.

Definimos ahora tres ecuaciones adicionales como combinación lineal de las ecuaciones anteriores:

$$k = (e) - z(f) \tag{4.11}$$

$$l = (g) - z(h) \tag{4.12}$$

$$m = (i) - z(j) \tag{4.13}$$

Las ecuaciones que resultan se pueden expresar en una matriz 3×3 que denominamos B.

$$B \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} k: & [3] & [3] & [4] \\ l: & [3] & [3] & [4] \\ m: & [3] & [3] & [4] \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{4.14}$$

El vector $[x\ y\ 1]^\top$ es un vector nulo de la matriz B. El determinante de B debe ser nulo, ya que las ecuaciones son lineal-mente dependientes (Dado que es un sistema de 3 ecuaciones con 2 incógnitas). Por lo que podemos extraer de esta restricción otra ecuación n. La ecuación n es un polinomio de grado 10 en z por lo que obtenemos 10 raíces para z.

Para cada raíz de z podemos obtener los valores de x e y, y podemos calcular 10 matrices esenciales utilizando la ecuación 4.9. Se pueden descartar las soluciones imaginarias.

4.3. Método de relocalización

En esta sección vamos a hablar del método de relocalización que usamos para localizar la cámara en las condiciones explicadas al principio de este capítulo. Los datos de entrada en el problema son: una imagen de la cual no conocemos los parámetros de posición y rotación de la cámara cuando la tomo, un mapa de puntos tridimensional y un conjunto de imágenes (que no tienen por que seguir ninguna trayectoria) de las cuales conocemos la posición de manera precisa.
El método consiste en escoger cinco puntos característicos y emparejarlos entre la imagen que queremos localizar y una de las que tenemos localizadas, para extraer mediante el algoritmo de los cinco puntos la posición de una con respecto a la otra y de esta manera tener localizada la imagen que no conocíamos.

Empezamos escogiendo una de las imágenes del conjunto de imágenes conocidas, por ejemplo la que más emparejamientos de puntos característicos tenga con la imagen a localizar (ver la figura 4.3), y después utilizamos el algoritmo de RANSAC [13] para entre los emparejamientos realizados escoger los cinco que mejor localización generan.
En cada iteración del RANSAC escogemos cinco puntos aleatorios de entre los emparejamientos, aplicamos el algoritmo de los cinco puntos y estimamos la matriz esencial que define la geometría entre dos vistas. Con la matriz esencial calculamos las rotaciones y traslaciones (existen cuatro combinaciones posibles de rotación y traslación que pueden generar una misma matriz esencial). Para cada posible combinación de rotación y traslación realizamos una optimización densa y con la que obtiene un error residual más pequeño, si es el menor hasta ahora nos guardamos la localización y seguimos iterando. Al final hemos sacado la solución que menos error obtiene de la función de coste densa. Se muestra la aplicación de este método en el algoritmo 1.

El método de optimización densa utilizado en este apartado es el mismo que se utiliza en el caso del seguimiento, ver ecuación 3.7, la única diferencia es la manera de generar la semilla inicial.

Algoritmo 1 RANSAC en la relocalización

Entrada: M mapa de puntos denso tridimensional.
I_l imagen que queremos localizar.
I_k imagen escogida, de la cual conocemos su rotación y traslación.
T_{kw} la posición de la imagen I_k respecto al mundo.
$P = \{P_k, P_l\}$ conjunto de puntos emparejados entre las imagenes I_k e I_l.

Salida: T_{lk} la transformación de la imagen que se pretendía re-localizar con respecto a la la imagen conocida.

$MIN\_ERR \leftarrow \infty$
T_{lk}
para $i = 1 : Iteraciones$ **hacer**
  $P \leftarrow permutacion\_aleatoria(P)$
  $P_{five} \leftarrow P[1:5]$
  $E \leftarrow FivePointsAlgorithm(P_{five})$
  $\{T_1, T_2, T_3, T_4\} = Esencial2Tranformacion(E)$
  para $j = 1 : 4$ **hacer**
    $\{T_f, Residual\} = OptimizacionDensa(M, I_k, I_l, T_{kw}, T_j)$
    si $Residual < MIN\_ERR$ **entonces**
      $MIN\_ERR \leftarrow Residual$
      $T_{lk} \leftarrow T_f$
    fin si
  fin para
fin para
devolver $T_{kw} \cdot T_{lk}$

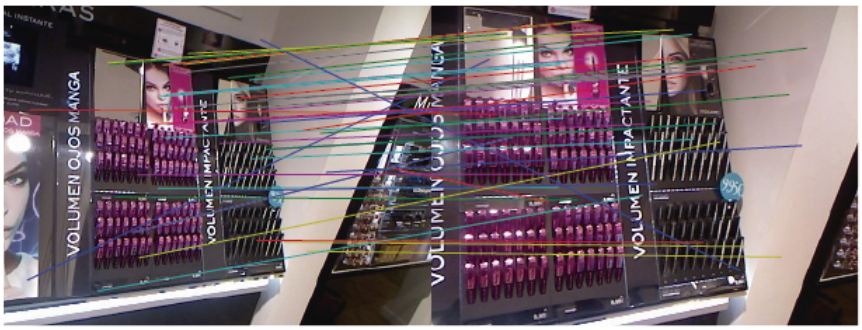


Figura 4.3: Emparejamiento de puntos SIFT entre dos imágenes (Incluye tanto *inliers* como *outliers*)

5. Resultados Experimentales

En este capítulo vamos a presentar los experimentos que hemos realizado en este trabajo para validar los dos algoritmos que hemos implementado, presentados en los capítulos 3 y 4. Hemos llevado a cabo experimentos de diferentes tipos, algunos de los cuales se han llevado a cabo en entornos reales. El objetivo es validar el correcto funcionamiento de los algoritmos y mostrar mejoras con respecto a la utilización de métodos *sparse*.

También se ha validado la optimización jerárquica explicada en la sección 3.1.2. Las pruebas que se han realizado se detallaran más adelante. En la siguiente sección se van a presentar los diferentes conjuntos de imágenes de prueba que se han utilizado durante los experimentos.

5.1. Conjuntos de datos utilizados

Durante los experimentos, para las pruebas hemos utilizado dos conjuntos diferentes de datos. El objetivo no es centrarnos en un sólo tipo de escena por lo que se han utilizado las dos que se presentan a continuación. Uno de ellos es un entorno controlado y el otro es un entorno real.

Experimiento en laboratorio: El primer conjunto de datos que hemos utilizado, es el de un entorno controlado en el que podemos trabajar con seguridad y no presenta excesiva dificultad. Este conjunto de datos son imágenes tomadas en el laboratorio junto con un mapa denso tridimensional del mismo. Ver figura 5.1.

Experimento en entorno real: También hemos realizado experimentos con imágenes de un entorno real. En este caso con una tienda de la marca LOREAL en Madrid. Estas imágenes presentan características que dificultan la localización utilizando métodos *sparse*, como son los brillos y reflejos. Ver la figura 5.2.

Figura 5.1: Conjunto de datos de prueba, entorno conocido.

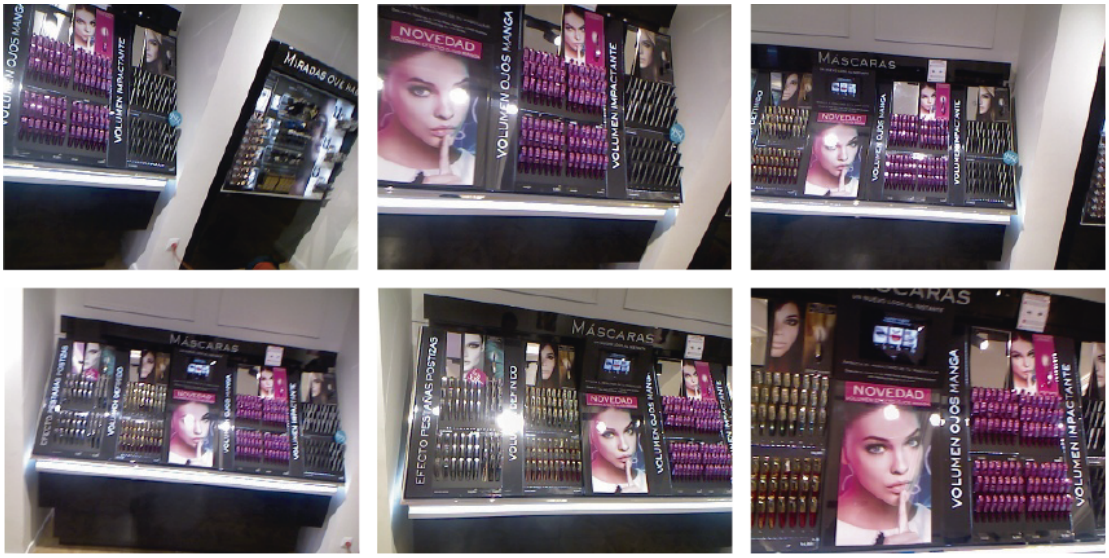


Figura 5.2: Conjunto de datos de prueba, entorno real.

5.2. Metodología

En esta sección explicamos la metodología que hemos seguido para validar los algoritmos presentados en este trabajo. En primer lugar hemos seleccionado un algoritmo de localización con el que podamos comparar los resultados obtenidos por nuestros algoritmos. El algoritmo que hemos utilizado es el *Bundle adjustment*. *Bundle adjustment* es un algoritmo que realiza simultáneamente la reconstrucción del mapa y la localización

de las cámaras. Es un método basado en puntos característicos que consigue resultados consistentes. No es lo mismo que aplicar los algoritmos de seguimiento o relocalización con una minimización basada en puntos salientes, como la explicada en la sección 3.1.1. *Bundle adjustment* resuelve un sistema de ecuaciones utilizando todas las imágenes al mismo tiempo, lo que lo hace más robusto. Además lo hemos aprovechado en el caso del primer conjunto de datos, ver figura 5.1, para la construcción del mapa denso. En el segundo caso, ver figura 5.2, se ha utilizado una cámara Kinect para la construcción del mapa.

Por último queda definir en qué nos hemos basado para comparar diferentes métodos, nuestro *Ground truth*. Dado que el objetivo de los métodos presentados son localizar la cámara con respecto a un mapa, hemos utilizado la información del mapa como punto de referencia. Haciendo la asunción de que el mapa se ha construido correctamente. Una proyección de los puntos del mapa denso en la imagen generaría la imagen que la cámara esta viendo, lo que denominamos imagen virtual. Por lo tanto tras estimar una posición y generar la proyección de los puntos, una resta de las imágenes, la virtual proyectada y la original, es una medida de error confiable.
Se puede ver en la figura 5.3 un ejemplo de esta medida de error. En la practica usamos la suma del error de cada *pixel*.

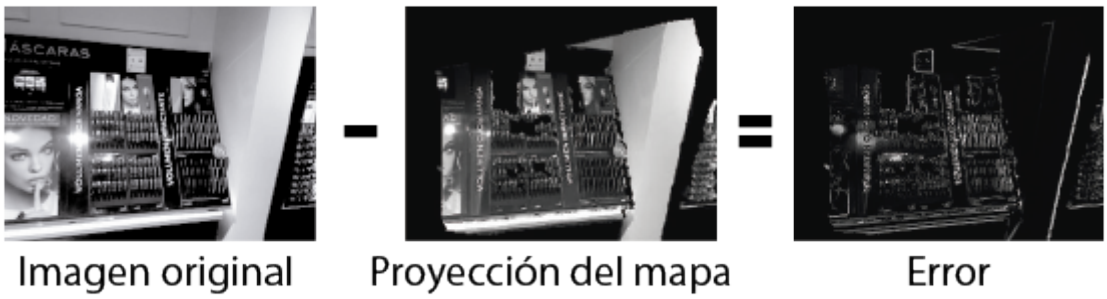


Figura 5.3: Medida de error, para la comparación de soluciones

Esta medida de error la calculamos de una manera diferente al coste definido para la minimización densa, ver ecuación 3.7, ya que en lugar de proyectar la imagen al mapa, proyectamos el mapa en la imagen lo que da una interpolación menos eficiente pero más fiable. De esta manera el valor de cada *pixel* es una interpolación del valor de los puntos del mapa que se proyectan cerca.

5.3. Pruebas realizadas

En esta sección se detalla cada una de la pruebas realizadas. Se han realizado dos pruebas de seguimiento, la primera de las cuales realiza una comparación con el método de *Bundle adjustment* y la segunda sirve para mostrar un ejemplo de seguimiento en

entorno real partiendo de una relocalización. Por ultimo explicaremos otros dos experimentos que hemos llevado a cabo para validar el algoritmo de relocalización. Ambos experimentos de relocalización se realizaron en el entorno real. El primero de ellos realiza una comparación con los resultados obtenidos por el *Bundle adjustment* y el segundo pretende mostrar la mejora que supone realizar la minimización densa de manera jerárquica, explicada en la sección 3.1.2. Las optimizaciones presentadas para la minimización densa se han utilizado en todos los experimentos, de no ser así se especifica en la descripción de la prueba.

5.3.1. Seguimiento en entorno conocido

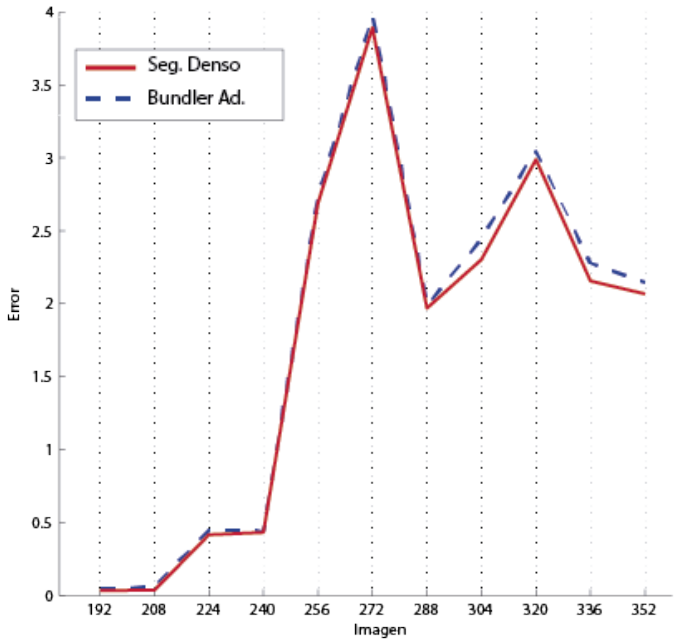
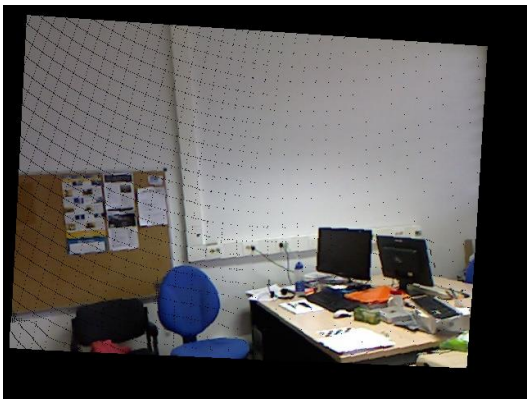


Figura 5.4: Seguimiento a lo largo del tiempo, alrededor de 11 segundos. Comparación con el algoritmo *Bundle adjustment*

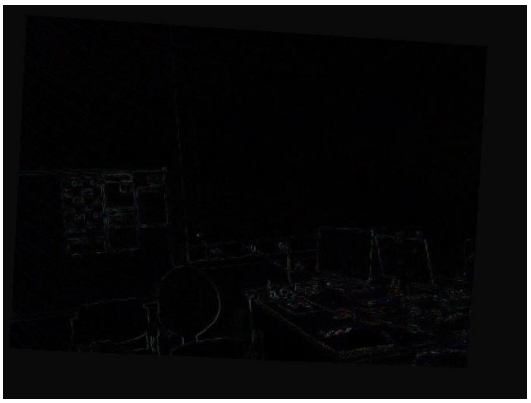
Esta prueba se ha realizado utilizando el primer conjunto de datos, mencionado en la sección 5.1. El experimento consiste en llevar a cabo el seguimiento partiendo de una semilla inicial conocida, la posición de la cámara en el instante anterior a tomar la primera imagen. El objetivo de este experimento es probar que durante el seguimiento utilizando técnicas densas no se pierde precisión conforme se aleja de la primera estimación. El experimento consiste en realizar el seguimiento de alrededor de 330 imágenes (11 segundos de vídeo) y comparar algunas imágenes de control (aproximadamente cada 20 imágenes) con el algoritmo *Bundle adjustment*. De esta manera podemos saber que tan buena es la estimación en diferentes momentos de tiempo.
El seguimiento se realiza estimando la localización de la cámara en la toma de cada

imagen y utilizando como semilla la localización estimada con la imagen anterior.

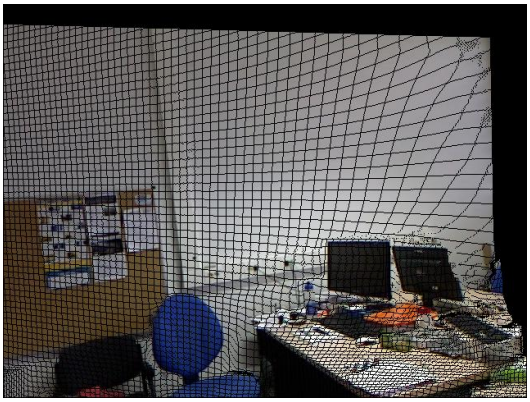
Se puede ver en la figura 5.4 que el error de el seguimiento con técnicas densas se mantiene estable con respecto al error generado por las estimaciones del *Bundle adjustment*, lo que demuestra que mantiene precisión en la localización. Además el error que generan nuestras estimaciones nunca esta por encima de las generadas por el *Bundle adjustment*. Hay que tener en cuenta que este experimento se realizo sin objetos móviles que pudieran obstaculizar la visión de la cámara.



(a) Reproyección del mapa al comienzo del seguimiento



(b) Imagen de error al comienzo del seguimiento



(c) Reproyección del mapa en la última imagen

(d) Imagen de error al final del seguimiento

Figura 5.5: Comparación de error al comienzo y al final del vídeo.

Se muestra en la figura 5.5 la comparación entre la primera estimación y la ultima de este seguimiento. Como se puede ver el error de la proyección se mantiene. La causa de que el error aumente en la gráfica presentada en la figura 5.4 es que durante el vídeo se produce un cambio de iluminación. Este cambio de iluminación nos ha permitido probar la robustez del método a este tipo de acontecimientos.

5.3.2. Seguimiento en entorno real

El experimento que se va a explicar en esta sección consiste en realizar seguimiento de una cámara monocular en un entorno real. Partiendo de una semilla estimada mediante el algoritmo de relocalización explicado en el capitulo 4. Se va a mostrar un gráfico con la trayectoria seguida por la cámara. Dado que el objetivo es realizar el seguimiento en un entorno real se ha usado el segundo conjunto de datos presentado en la sección 5.1.

Figura 5.6: En esta figura se muestra el mapa denso del entorno real utilizado para este experimento

(a) Proyección de la relocalización (b) Error de la relocalización

Figura 5.7: Error de la relocalización utilizada de semilla

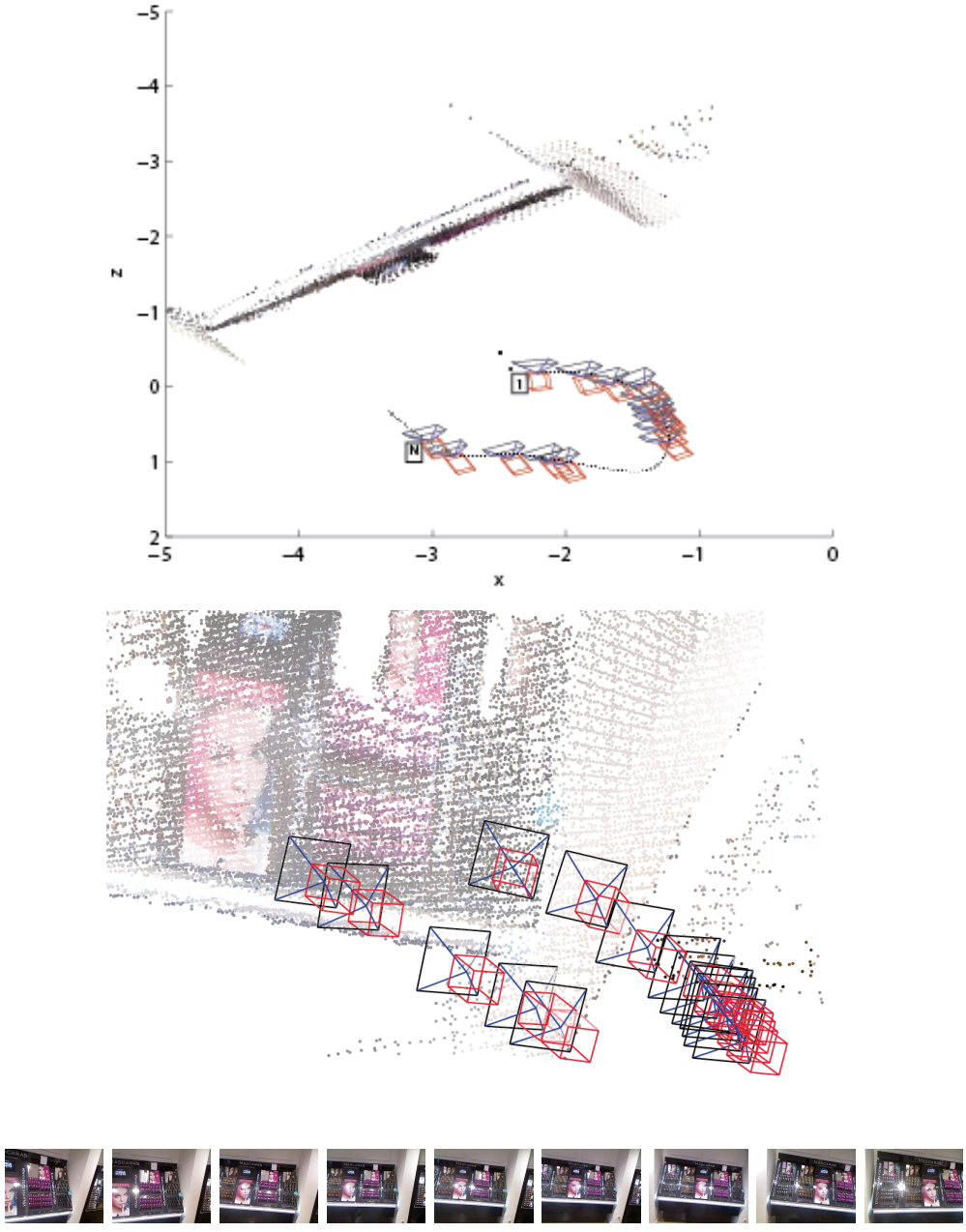


Figura 5.8: En esta figura se muestra el mapa denso y las diferentes posiciones que toma la cámara durante la toma del vídeo al cual se le ha realizado seguimiento. Además se muestran algunas de las imágeness de la secuencia de vídeo (en orden).

El seguimiento se ha realizado de la misma manera que en el experimento anterior pero partiendo de una semilla inicial estimada a partir del algoritmo de relocalización.

Se presenta el mapa denso tridimensional utilizado en la figura 5.6

La estimación de la localización de la primera cámara se ha logrado con la precisión mostrada en la figura 5.7. El resultado del experimento, tras realizar el seguimiento de un vídeo tomado en la escena, se muestra en la figura 5.8.

5.3.3. Comparación de la relocalización con el Bundle

Para este experimento se ha utilizado el segundo conjunto de imágenes. Las tomadas en un entorno real. El objetivo es llevar a cabo la relocalización utilizando nuestro algoritmo y comparar el error de reproyección con las estimaciones obtenidas por el *Bundle adjustment*. Nuestro algoritmo consiste en la generación de una semilla mediante el algoritmo de los cinco puntos y finaliza la estimación realizando una minimización densa. La comparación entre nuestros resultados con los del *Bundle* ha consistido en localizar un conjunto de imágenes utilizando el *Bundle adjustment* y localizarlas con nuestro método de relocalización, una vez hecho, hemos calculado el error para cada caso de prueba y los hemos comparado obteniendo los resultados mostrados en la figura 5.9.

Como se puede ver en la imagen 5.9 en la mayoría de los casos el error conseguido por nuestro método es menor que el del *Bundle adjustment* no obstante hay tres imágenes en las que no es así y por lo tanto vamos a analizar estos casos en particular. Estos casos son las imágenes 1, 3 y 4 en la gráfica.

Imagen 1: En la primera imagen el *Bundle adjustment* presenta un error al cuadrado de 4081 mientras que nuestro método logra un 4171, que es una diferencia muy pequeña y que no representa demasiada perdida en la localización de la cámara. A continuación el la figura 5.10 se muestran las diferencias de las imágenes de error entre los dos casos.

Imagen 3: Este es un caso similar al caso 1 en el que la diferencia de error es mínima y que a pesar de no tener la mejor solución es una buena semilla para retomar el seguimiento y por lo tanto resulta factible.

Imagen 4: Este caso por otro lado es un caso en el que nuestro algoritmo no converge en un mínimo cercano al mínimo global, es decir no logra una solución aceptable, lo cual se ve en la diferencia de error. Se puede ver en la figura 5.11 la re-proyección de la posición estimada junto con la imagen que se deseaba re-localizar donde se puede observar el error.

Los motivos de que este caso funcione peor es que la imagen que se escogió para realizar la re-localización a partir de ella era una imagen con la que no comparte muchos

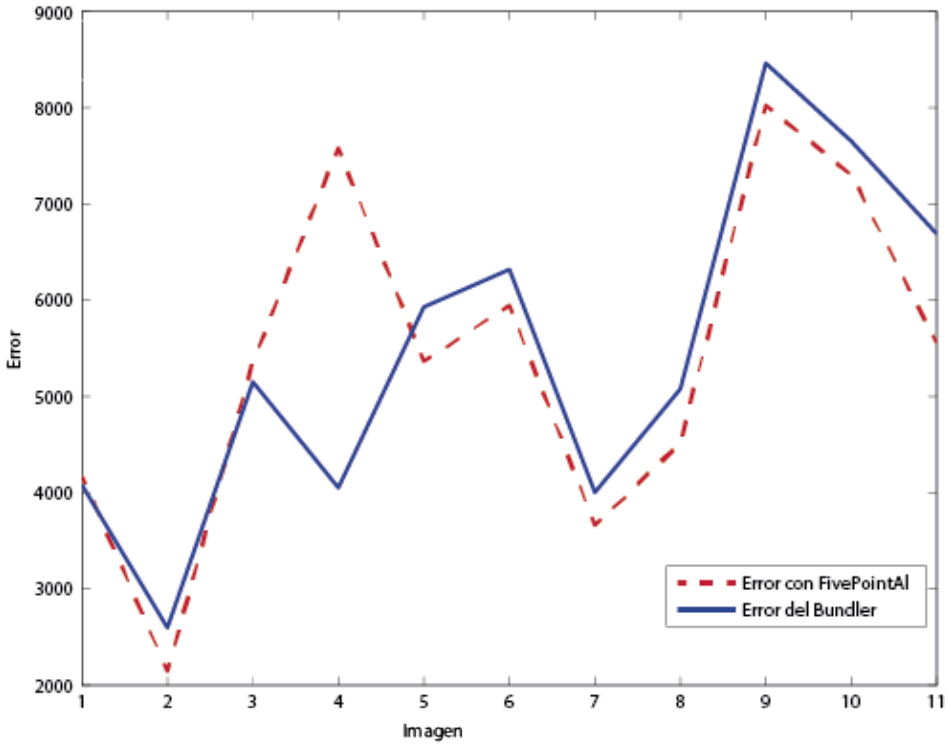
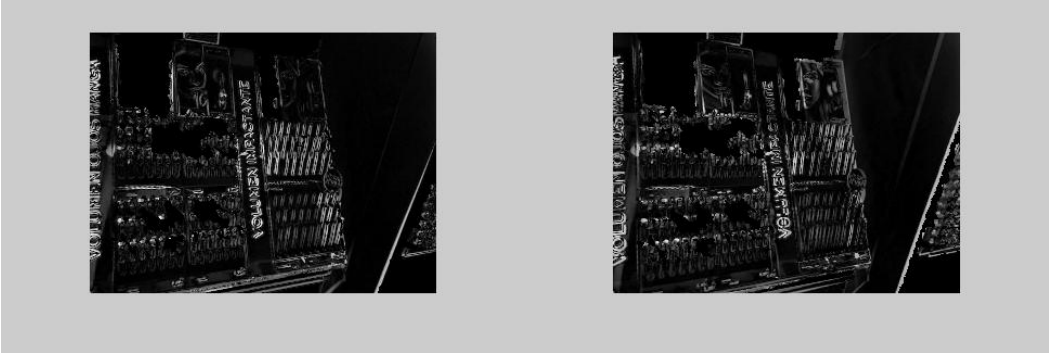


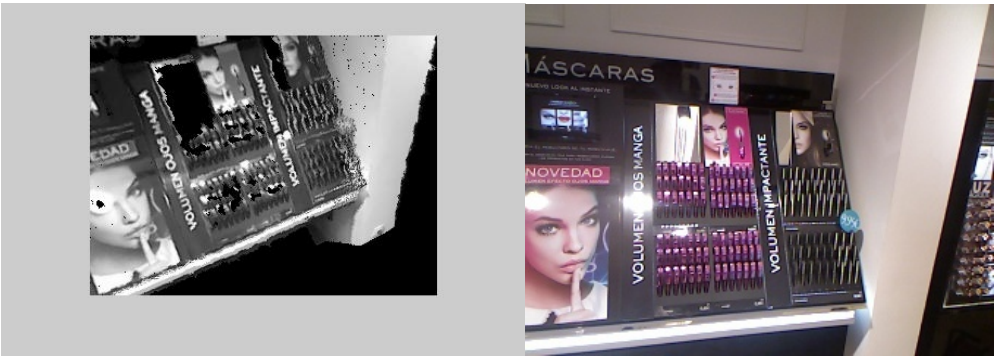
Figura 5.9: En el eje de las x numerados del 1 al 11 se muestran todos los casos de prueba y en el eje de las y se muestra la suma de los errores de los *pixeles* de la imágenes al cuadrado. La linea roja (discontinua) representa el error en cada caso del método que presentamos en el trabajo, mientras que la linea azul representa el error que consigue el *Bundle adjustment*.

emparejamientos de putos, al rededor de unos 30 incluyendo emparejamientos espurios, los cuales no son suficientes para que el algoritmo converja de manera adecuada. Se puede ver en la figura 5.12 los emparejamientos de puntos de este caso.



(a) Error cometido por el *Bundle adjustment* (b) Error cometido por nuestro metodo

Figura 5.10: Comparación del error entre nuestro método y el *Bundle adjustment* de la imagen 1.



(a) Proyección tras la estimación (b) Imagen a re-localizar

Figura 5.11: Mapa proyectado en un caso de estimación errónea

5.3.4. Relocalización utilizando la optimización jerárquica

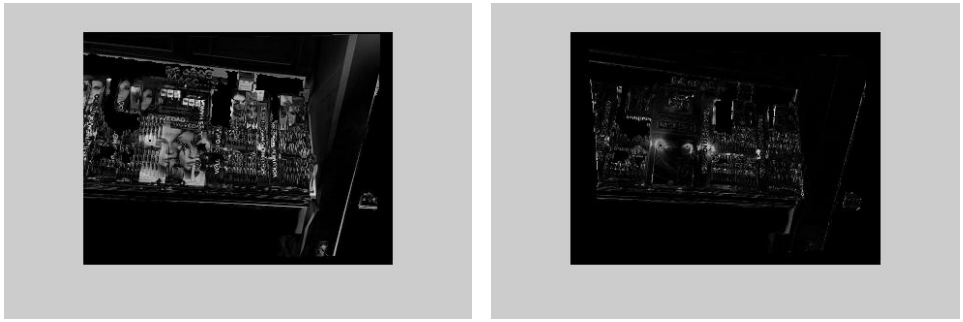
En el ultimo experimento que presentamos, pretendemos valorar cuanta mejora supone utilizar la optimización jerárquica presentada para el algoritmo de minimización densa, explicado en la sección 3.1.2.
Para realizar este experimento utilizamos el mismo grupo de imágenes utilizadas en el punto anterior. En este caso para cada imagen se ha partido de la misma semilla inicial y se ha llevado a cabo la estimación de la posición usando la optimización jerárquica y sin usarla. En ambas estimaciones se ha mantenido la optimización en dos fases, en el caso del uso de la optimización jerárquica, la optimización en dos fases se realiza antes que la optimización jerárquica.

Figura 5.12: Emparejamientos de puntos en el la imagen 4.

Imagen	Con pirámide	Sin pirámide
1	**4122**	4930
2	**4616**	6458
3	**5040**	6925
4	**7867**	8398
5	**5368**	**5368**
6	**5947**	6569
7	**6276**	9015
8	**6382**	8508
9	**6244**	8187
10	**7177**	7858
11	5557	**5551**

Cuadro 5.1: Tabla de errores para comparar el uso de pirámide o no. (En **negrita** el error mínimo en la estimación para cada imagen).

La mejora ha sido considerable, teniendo en cuenta que salvo en la ultima imagen, en el resto, utilizar la optimización ha mejorado siempre las estimaciones. Se muestra un ejemplo en la figura 5.13 de error final tras realizar una minimización usando y sin usar la optimización jerárquica.



(a) Error trar relocalización sin uso de pirámide

(b) Error tras relocalización haciendo uso de la pirámide

Figura 5.13: Comparación de los errores del uso o no de la pirámide de optimización jerárquica durante la relocalización.

6. Conclusiones

Hemos presentado dos algoritmos de localización basados en la utilización de un mapa de puntos denso. El primero de ellos utiliza técnicas completamente densas sin la utilización de puntos salientes, trata de resolver un caso particular de la localización, el seguimiento, presentado en el capítulo 3. El segundo trata de resolver el problema de la relocalización, un caso más general. En este segundo caso, hemos utilizado técnicas mixtas, ya que partimos de generar una semilla inicial mediante métodos basados en puntos salientes y utilizamos la minimazición densa para ajustar y valorar cada solución, este algoritmo viene presentado en el capítulo 4.

Aunque no hemos podido prescindir totalmente del uso de puntos caracteristicos para la localización, hemos cumplido nuestros objetivos. Hemos desarrollado dos algoritmos de localización que aprovechan la información de un mapa denso tridimensional y hemos llevado a cabo la localización de una cámara monocular en entornos reales.

En la evaluación del algoritmo de seguimiento hemos realizado una comparación con el *Bundle adjustment*, que es un algoritmo que marca el estado del arte en metodos *sparse*. En esta comparación hemos probado que nuestro algoritmo es igual de robusto, e incluso en algunos casos mejora la precisión obtenida por el *Bundle adjustment*. Nótese que el algoritmo *Bundle adjustment* es un algoritmo de SfM, el cual utiliza los emparejamientos de puntos de un grupo de imágenes, en lugar de utilizar únicamente una, y por lo tanto nuestro algoritmo parte con desventaja en la comparación. Esta comparación se puede ver en la sección 5.3.1. Además gracias a los resultados obtenidos en el experimento de la sección 5.3.2, podemos demostrar que nuestro algoritmo de relocalización se puede utilizar para calcular la semilla inicial a partir de la cual arrancar el algoritmo de seguimiento.

En el segundo caso, la relocalización de una cámara monocular en un mapa denso tridimensional, los resultados muestran que, a excepción de algunos casos, el algoritmo de relocalización es tan preciso como el *Bundle adjustment*, ver el experimento en la sección 5.3.3. Es necesario la destacar la importancia en el calculo de la semilla inicial para arrancar una minimización densa, ver ecuación 3.7. Nótese que estos experimentos se han utilizado conjuntos de imágenes de naturaleza complicada para este tipo de tareas, ya que tienen espejos, muchos brillos y patrones que dificultan la localización, ver figura 5.2.

6.1. Trabajo futuro

Una de las continuaciones del trabajo es desarrollar una técnica de relocalización que prescinda del uso de puntos salientes para el calculo de la semilla inicial. Es decir desarrollar técnicas densas para este tipo de localización.

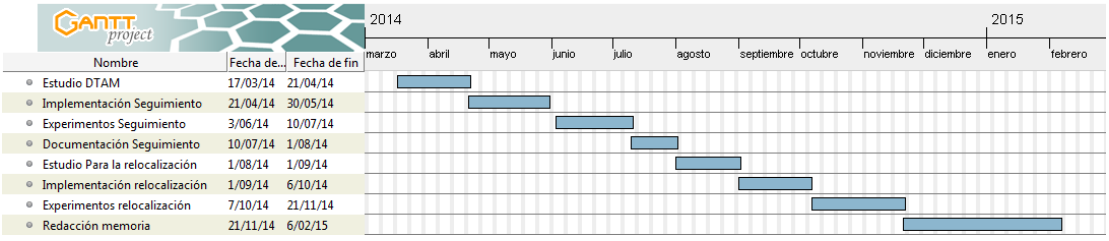
Otra de las posibles continuaciones es el desarrollo de estos algoritmos en tiempo real para orientarlos a aplicaciones reales combinándolos con técnicas de construcción de mapas tridimensionales para poder realizar una ejecución continua.

Por ultimo, quedaría contemplar la posibilidad de que la cámara utilizada para construir el mapa y la cámara que se desea localizar tengan características muy diferentes: mayor resolución y aumento o disminución de la distancia focal. Es un problema conocido como *Wide baseline problem*, con el que nos hemos encontrado durante la realización de este trabajo. Otra posible continuación consistiría en tratar de resolver este problema utilizando técnicas densas.

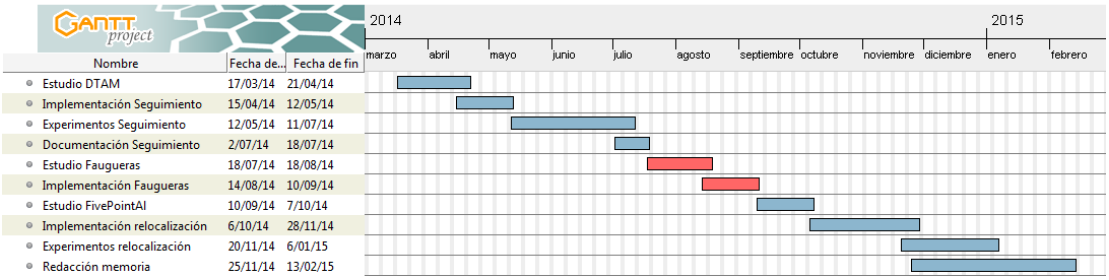
7. Planificación y herramientas utilizadas

La planificación de este proyecto se llevó a cabo teniendo en cuenta la necesidad de cursar 8 asignaturas durante su realización. El objetivo inicial de la planificación era finalizar el proyecto para la convocatoria de Febrero de 2015 como se puede ver en la figura 7.1a.

Finalmente a pesar de ciertos imprevistos que están incluidos en la figura 7.1b de color



(a) Diagrama de Gantt de la planificación inicial



(b) Diagrama de Gantt del desarrollo real del proyecto

Figura 7.1: Diagramas de Gantt

rojo, hemos conseguido cumplir con los plazos que nos marcamos en la planificación. Los imprevistos mencionados consistieron en el estudio e implementación de otra técnica para la relocalización diferente a la mencionada en el capítulo 4. Esta técnica no dio los resultados esperados, por lo que finalmente se desecho y se utilizo la que ya hemos presentado en este trabajo.

7.1. Herramientas y tecnología utilizada.

El lenguaje en el que se han realizado todos los algoritmos y los experimentos es Matlab, ya que es el lenguaje utilizado por el grupo de investigación con el que he estado trabajando y por la cantidad de módulos implementados que hemos podido utilizar.

Módulos reutilizados: En este trabajo se han utilizado algunos módulos que trae incorporado el *framework* de Matlab como son la función de optimización utilizada (*lsqnonlin*) y las herramientas necesarias para generar vídeos y gráficas. También se ha hecho uso de la *toolbox* VLFeat [14] para la extracción de puntos característicos en la relocalización. Hemos incorporado al algoritmo de relocalización, la modulo para Matlab del algoritmo de los cinco puntos, implementado por [12]. Hemos utilizado descriptores SIFT en los puntos. Hemos utilizado el algoritmo del *Bundle adjustment* para tener una referencia objetivo. Y por ultimo añadir que hemos utilizado la implementación para Matlab del algoritmo de los cinco puntos aportada por [15].

Módulos implementados: Para poder realizar este trabajo hemos implementado los módulos para la transformaciones de los parámetros extrínsecos a los coeficientes de Lie, ver sección 2.2. Hemos implementado un modulo de carga y manipulación de las imágenes y los parámetros de las cámaras. También hemos implementado los módulos necesarios para llevar a cabo la minimización densa, como son las funciones de error y evaluación. También se han diseñado e implementado todos los experimentos realizados para validar la calidad de los algoritmos presentados en este trabajo.

Bibliografía

[1] Jan J Koenderink, Andrea J Van Doorn, et al. Affine structure from motion. *JOSA A*, 8(2):377–385, 1991.

[2] R. Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. Springer, 2010.

[3] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[4] Alejo Concha and Javier Civera. Using superpixels in monocular slam. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 365–372. IEEE, 2014.

[5] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. DTAM: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE, 2011.

[6] Roger Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using o the shelf tv cameras and lenses. . *IEEE Journal of Robotics and Automatione*, pages 323–344, 1987.

[7] Steven Lovegrove and Andrew J Davison. Real-time spherical mosaicing using whole image alignment. In *Computer Vision–ECCV 2010*, pages 73–86. Springer Berlin Heidelberg, 2010.

[8] H. Strasdat, A. Davison, and E. Edwards. *Local Accuracy and Global Consistency for Efficient SLAM*. Imperial College London, 2012.

[9] Jakob Engel, Thomas Schöps, and Daniel Cremers. LSD-SLAM: Large-scale direct monocular slam. In *Computer Vision–ECCV 2014*, pages 834–849. Springer, 2014.

[10] Henrik Stewenius, Christopher Engels, and David Nistér. Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60(4):284–294, 2006.

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

[12] David Nistér. An efficient solution to the five-point relative pose problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):756–770, 2004.

[13] 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.

[14] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008.

[15] H. Stewénius, C. Engels, and D. Nistér. Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60:284–294, June 2006.