Universidad Zaragoza

Trabajo Fin de Máster

Visión 3D para Realidad Aumentada

3D Vision for Augmented Reality

Autor:
Alejandro Fontán Villacampa

Director:
Javier Civera Sancho

Escuela de Ingeniería y Arquitectura (EINA)

2017

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).

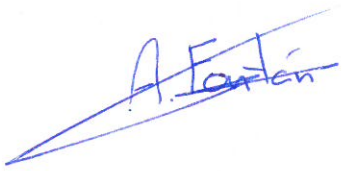
D./Dª. Alejandro Fontán Villacampa,

con nº de DNI 73131929W 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) Máster, (Título del Trabajo) Visión 3D para Realidad Aumentada,

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

Zaragoza, 23 de Noviembre de 2017



Fdo: Alejandro Fontán Villacampa

Visión 3D para Realidad Aumentada

RESUMEN

La Realidad Aumentada (en inglés "Augmented Reality", abreviado AR) es el conjunto de tecnologías que superponen una imagen generada por ordenador, sobre la visión física del mundo de una persona, combinando así elementos del mundo digital con los del mundo real. Existen ya en el mercado numerosas aplicaciones basadas en realidad aumentada para todo tipo de dispositivos (móviles, tabletas y "smart glasses"). Aunque tradicionalmente se han asociado al mercado del ocio y el entretenimiento existe, dado su gran potencial, un interés creciente en la AR por parte de la industria, empresas de marketing, arquitectura, logística... Ello hace que la carrera por "la fusión entre el mundo real y el digital" no haya hecho más que empezar.

Los problemas a los que la AR se enfrenta han sido típicamente el entendimiento de las escenas (reconocimiento de objetos, imágenes, caras...), localización instantánea de una cámara junto con el mapeo del entorno (SLAM e "Instant Tracking"), generación del contenido aumentado (imágenes u objetos 3D) y geolocalización. En la actualidad uno de los objetivos de las empresas es la creación de herramientas de software que permitan a cualquier compañía o desarrollador implementar su propia aplicación basada en realidad aumentada. Existe la expectativa de en pocos años generar aplicaciones que sincronicen los contenidos aumentados para que puedan ser apreciados por numerosos usuarios al mismo tiempo y en el mismo lugar, e incluso, la posibilidad de interactuar con ellos. Poniendo la vista en el futuro, es evidente que la búsqueda de realismo de la AR genera problemas técnicos que no solo requieren de la evolución del hardware sino de la investigación y aplicación de otras aproximaciones como la "Fusión de Sensores" o "Machine Learning".

La principal contribución de este trabajo es un algoritmo de reconstrucción semidensa del entorno 3D para una cámara monocular, basado en el cálculo probabilístico de la profundidad inversa de los puntos. Confiando en las expectativas que en la actualidad los métodos directos ofrecen, el sistema realiza el emparejamiento de puntos entre imágenes comparando las intensidades y gradientes de los píxeles a través de una búsqueda epipolar. Se incorpora también un proceso de fusión de hipótesis de profundidades fundamentado en el cálculo de sus incertidumbres. Además se añade un filtro para reducir el número de errores y un proceso de densificación para aumentar la cantidad de puntos del mapa.

El trabajo contiene un conjunto de experimentos destinados a valorar la influencia de los distintos parámetros en la precisión y la eficiencia. En ellos se discute acerca de la implementación y resultados de los procesos de interpolación bilineal, precisión subpíxel y optimización de la profundidad. Por otra parte se abordan también los procedimientos de selección de píxeles de alto gradiente y de las imágenes adecuadas para la reconstrucción.

Este tipo de algoritmo se puede conectar, por ejemplo, con la ampliación del mapa de un sistema de SLAM o con el mapeo del entorno de una aplicación de realidad aumentada. Por ello, en el trabajo se exploran las mejoras necesarias y posibilidades para su implementación en tiempo real. Además se investiga su acoplamiento en un sistema de localización basado en el seguimiento de una imagen ("Image Tracking") y en dos sistemas de SLAM Visual-Inercial (OKVIS y VINS).

Índice

Índice de imágenes y figuras

Índice de tablas

1. Introducción

Se llama “Realidad Aumentada” (AR) a una versión mejorada de la realidad, donde las vistas del mundo real se complementan con imágenes superpuestas generadas por ordenador, mejorando así la percepción de la realidad. La figura 1 contiene una muestra de cómo la realidad aumentada, en su aplicación en dispositivos móviles, mezcla el contenido digital con el real haciéndolos parecer uno solo. Modelos 3D, información y descripciones del entorno, ayuda y soporte a usuarios y trabajadores o menús interactivos son ejemplos de los contenidos que pueden ser aumentados en las imágenes.

La trayectoria de la realidad aumentada (figura 2) ha venido de la mano de la diversificación y desarrollo de los dispositivos móviles. Casi al mismo tiempo que apareció el primer dispositivo Android al final de 2008, la empresa austríaca Wikitude GmbH publicó el primer navegador de realidad aumentada para usuarios de smartphones llamado “Wikitude World Browser”. Se trataba de un buscador de realidad aumentada que exploraba el entorno utilizando la cámara y los sensores del dispositivo para junto con contenido geo-referenciado mostrar la información justo donde se encontraban los objetos reales.

El siguiente hito en el mundo de la AR fue el lanzamiento por Google en 2014 de las “Google Glass”, una pantalla óptica montada en la cabeza diseñada con la forma de un par de gafas.

En julio de 2016, Niantic presenta “Pokémon Go” un juego gratuito de realidad aumentada basado en la ubicación para dispositivos iOS y Android. El juego utiliza el dispositivo GPS para localizar y capturar criaturas virtuales, que aparecen en la pantalla como si estuvieran en la misma ubicación del mundo real que el jugador. Esta aplicación se convirtió en un fenómeno global y fue una de las más utilizadas y rentables en 2016, después de haber sido descargada más de 500 millones de veces en todo el mundo.

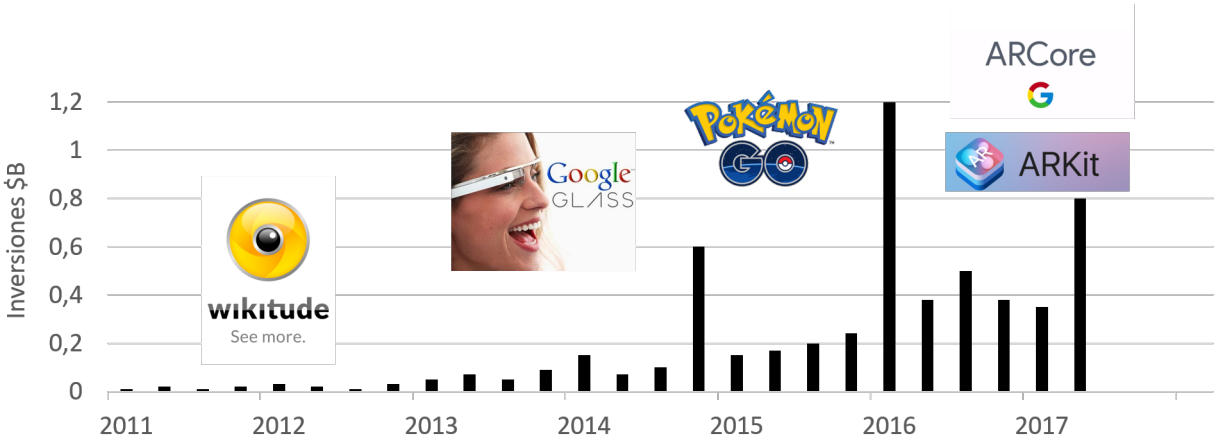
Figura 1: Aplicaciones de la AR en el mercado.

En la actualidad las plataformas promocionadas por Google y Apple (ARCore y ARKit respectivamente) permiten crear aplicaciones de realidad aumentada en dispositivos móviles. Fundamentalmente hacen dos cosas: rastrear la posición del dispositivo móvil mientras se mueve y desarrollar su propia comprensión del mundo real. Estas librerías permiten a los desarrolladores colocar objetos, anotaciones u otra información de una manera que se integre perfectamente con el mundo real.

Desde el punto de vista técnico, las tecnologías que la AR necesita son: sistemas de localización y mapeo 3D (SLAM), entendimiento de escenas, reconocimiento y rastreo de objetos e imágenes y geolocalización.

A partir de la implementación de estas tecnologías en la amplia gama de dispositivos actuales (smartphones, tabletas o "smart glasses") es inmenso el número de aplicaciones que ya existen en el mercado y las potencialmente desarrollables.

En el sector de la publicidad y el consumo aumentan las empresas que buscan en la AR formas de que el usuario interactúe con el producto. Algunas permiten aumentar los productos en 3D de un catálogo de forma que se puede tener una previsualización, por ejemplo, del mobiliario en el lugar exacto que va a ocupar. Otras empresas aumentan sus carteles y revistas con información útil para el comprador como precios, vídeos y modelos 3D. Y por supuesto dentro del mundo de los juegos y el entretenimiento como la ya mencionada "Pokemon Go" o la aplicación que permite solo con sostener el smartphone o tableta frente a la pantalla aumentar los contenidos de la televisión.



"El mercado de la realidad virtual y de la realidad aumentada estaba valorado en 5,000 millones de dolares americanos en 2016 y se prevé que alcance una valoración de mercado cercana a los 120 billones de dolares americanos al final de 2025". (Future Market Insights May 18th, 2017)

Figura 2: Inversiones en el mercado de la Realidad Aumentada.

Dentro del sector de la Industria comienzan a popularizarse las aplicaciones que dan apoyo e información de forma remota y facilitan el trabajo a los operarios. Por ejemplo el proyecto en el que un usuario con unas gafas AR puede visualizar todos los componentes (internos y externos) de una moto, así como la información completa relativa a ellos. O el proyecto que ofrece seguridad y limpieza en los controles de cualquier maquinaria al crear una versión aumentada de estos. Empresas vinculadas a los campos de la arquitectura o la construcción, la localización, la navegación y los viajes aprovechan también los recursos que la AR ofrece [8].

Asumiendo como objetivo de referencia de la AR "unir el mundo físico con el digital haciéndolos parecer uno solo" algunos de los retos a los que se enfrenta son:

- El "entendimiento del entorno" cuyo enorme espectro puede abarcar desde la búsqueda de planos en escenas, generación de mapas 3D, reconocimiento de objetos, hasta la comprensión de como los elementos se mueven en una escena. Todo ello se utiliza para dotar de la máxima realidad al contenido aumentado.

- La búsqueda de la escala métrica en la reconstrucción de una escena o una trayectoria y la construcción de un mapa en el que situar el sensor, resultan fundamentales para generar los contenidos de la imagen correctamente. Relacionado con esto ha aumentado el interés por la tecnología de "Fusión de Sensores" que permite, por ejemplo, utilizar el potencial de una cámara junto con el de los acelerómetros o giróscopos de un teléfono móvil.
- La eliminación de la deriva en la estimación de la posición de una cámara para que los elementos aumentados no varíen de posición a lo largo del tiempo.
- La resistencia de los algoritmos a movimientos agresivos que son perfectamente habituales en aplicaciones destinadas a cualquier usuario.
- Y el equilibrio entre la precisión, la demanda computacional y el tiempo real para que el aumento de la imagen funcione en cualquier dispositivo.

El algoritmo desarrollado en este trabajo aborda la generación del mapa 3D semidenso de una escena que permita a las aplicaciones situar los sensores de forma precisa (ya que un mapa elimina la deriva en el cálculo de la localización). Este programa puede desempeñar, por ejemplo, la generación del mapa 3D de un sistema de SLAM o el entendimiento de la escena de una aplicación de realidad aumentada.

Se fundamenta en la búsqueda estéreo de correspondencias mediante métodos directos (comparando la magnitud y la dirección del gradiente de intensidad de los puntos de la imagen) y el cálculo de la incertidumbre de la profundidad inversa basado en [12]. Implementa además un sistema de fusión de hipótesis de profundidades basado en la incertidumbre y en la optimización, un filtro para descartar errores y un proceso para densificar el mapa.

Se discute también acerca de otros procesos implicados como la selección de píxeles, las combinaciones de parámetros, la relación intensidad-gradiente o la elección de imágenes (paralajes y distancias). Se escogen dos de los vídeos con trayectorias reconstruidas en [15] para ejemplificar y analizar el comportamiento del algoritmo al realizar la reconstrucción.

De cara a una posible aplicación se aportan experimentos de comportamiento en tiempo real y de sincronización con algoritmos de SLAM Visual-Inercial (OKVIS [11] y VINS [13]) e "Instant-Tracking" [8].

2. Modelo de proyección

Dado que las cámaras proporcionan una visión 2D de una escena 3D se necesitan al menos dos imágenes para realizar una reconstrucción. Los procesos de "reconstrucción estéreo"tienen en común la búsqueda de correspondencias entre dos imágenes (por extracción de características o errores fotométricos), filtrado para eliminar errores de correlación y cálculo de los puntos 3D. Estos procesos requieren conocer la geometría interna de la cámara (calibración y distorsión), un modelo de proyección y las posiciones de las cámaras. En este capítulo se explican los modelos utilizados para proyectar un punto 3D en una imagen y viceversa [10].

2.1. Calibración de la cámara

La matriz de calibración ($\mathbf{k}$) contiene los parámetros intrínsecos de la cámara: la distancia focal horizontal y vertical en píxeles (f_x y f_y) y la posición del punto principal en la imagen (c_x y c_y). Adicionalmente se podría definir un coeficiente que modelase la falta de perpendicularidad de los píxeles [1].

$$\mathbf{k} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{k}^{-1} = \begin{pmatrix} 1/f_x & 0 & -c_x/f_x \\ 0 & 1/f_y & -c_y/f_y \\ 0 & 0 & 1 \end{pmatrix} \tag{1}$$

De esta forma se pueden expresar las coordenadas de un punto en píxeles de la imagen (coordenadas u y v):

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{k} \begin{pmatrix} x_d \\ y_d \\ 1 \end{pmatrix} \tag{2}$$

2.2. Distorsión de la imagen

El modelo de distorsión utilizado es el radial-tangencial [1] que utiliza cinco coeficientes para modelar la distorsión introducida por las lentes y otros componentes.

Distorsión de un píxel

Las coordenadas normalizadas del punto proyectadas en la imagen definen el radio de distorsión.

$$r_d^2 = x_n^2 + y_n^2 \tag{3a}$$

El coeficiente de distorsión radial se modela de forma polinomial.

$$c_d = (1 + k_1 r_d^2 + k_2 r_d^4 + k_5 r_d^6) \tag{3b}$$

El vector $\overline{dx}$ representa la distorsión tangencial debida al descentrado de los componentes de la lente o de otros defectos de fabricación.

$$\overline{dx} = \begin{pmatrix} 2k_3 x_n y_n + k_4(r_d^2 + 2x_n^2) \\ 2k_4 x_n y_n + k_3(r_d^2 + 2y_n^2) \end{pmatrix} \tag{3c}$$

Tras aplicar la distorsión de la lente se obtienen las coordenadas distorsionadas del punto.

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = \overline{f_d}(x_n, y_n) = c_d \begin{pmatrix} x_n \\ y_n \end{pmatrix} + \overline{dx} \tag{3d}$$

Desdistorsión de un píxel

Dado que la función de distorsión es no lineal el proceso de desdistorsión se lleva a cabo mediante la minimización del error de reproyección con una optimización no lineal de Gauss-Newton (Anexo A: Revisión del cálculo).

Se calculan las derivadas parciales del radio de distorsión con respecto a las coordenadas desdistorsionadas.

$$\frac{\partial r_d^2}{\partial x_n} = 2x_n \qquad \frac{\partial r_d^2}{\partial y_n} = 2y_n \tag{4a}$$

Se evalúan también las derivadas parciales de los parámetros de la función de distorsión con respecto a las coordenadas desdistorsionadas.

$$\frac{\partial c_d}{\partial x_n} = 2x_n(k_1 + 2k_2 r_d^2 + 3k_5 r_d^4) \qquad \frac{\partial c_d}{\partial y_n} = 2y_n(k_1 + 2k_2 r_d^2 + 3k_5 r_d^4) \tag{4b}$$

$$\frac{\partial \overline{dx}}{\partial x_n} = \begin{pmatrix} 2k_3 y_n + 6k_4 x_n \\ 2k_4 y_n + 2k_3 x_n \end{pmatrix} \qquad \frac{\partial \overline{dx}}{\partial y_n} = \begin{pmatrix} 2k_3 x_n + 2k_4 y_n \\ 2k_4 x_n + 6k_3 y_n \end{pmatrix} \tag{4c}$$

Aplicando la regla de la cadena se construye la matriz jacobiana de la función de distorsión y el error asociado al paso actual.

$$\mathbf{J}_k = \begin{pmatrix} \frac{\partial x_d}{\partial x_n} & \frac{\partial x_d}{\partial y_n} \\ \frac{\partial y_d}{\partial x_n} & \frac{\partial y_d}{\partial y_n} \end{pmatrix}_k = \begin{pmatrix} \frac{\partial c_d}{\partial x_n} x_n + c_d + \frac{\partial dx_{x_d}}{\partial x_n} & \frac{\partial c_d}{\partial y_n} x_n + \frac{\partial dx_{x_d}}{\partial y_n} \\ \frac{\partial c_d}{\partial x_n} y_n + \frac{\partial dx_{y_d}}{\partial x_n} & \frac{\partial c_d}{\partial y_n} y_n + c_d + \frac{\partial dx_{y_d}}{\partial y_n} \end{pmatrix}_k \tag{4d}$$

$$\overline{e}_k = \overline{f_d}(x_n, y_n)_k - \begin{pmatrix} x_d \\ y_d \end{pmatrix} \tag{4e}$$

Iterando con la ecuación 4f y volviendo a calcular la matriz jacobiana y el error se obtienen las componentes desdistorsionadas (como semilla se utilizan las componentes con distorsión).

$$\begin{pmatrix} x_n \\ y_n \end{pmatrix}_{k+1} = \begin{pmatrix} x_n \\ y_n \end{pmatrix}_k - (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\overline{e}_k \tag{4f}$$

2.3. Modelo de cámara Pinhole

El modelo utilizado como aproximación de la cámara es el de "cámara estenopeica (pinhole)". Los puntos 3D se proyectan en el plano imagen mediante rayos que pasan por el centro de proyección [10]. La transformación proyectiva de 3D a 2D se realiza normalizando las coordenadas con la tercera componente.

$$\begin{pmatrix} x_n \\ y_n \end{pmatrix} = \begin{pmatrix} \frac{x_c}{z_c} \\ \frac{y_c}{z_c} \end{pmatrix} \tag{5}$$

2.4. Geometría 3D

El movimiento de la cámara se describe con las matrices del grupo de transformaciones del sólido rígido ($\mathbf{P} \in \mathrm{SE}(3)$) que incluye las rotaciones ($\mathbf{R} \in SO(3)$) y las traslaciones ($\mathbf{t} \in \mathbb{R}^3$)) [14].

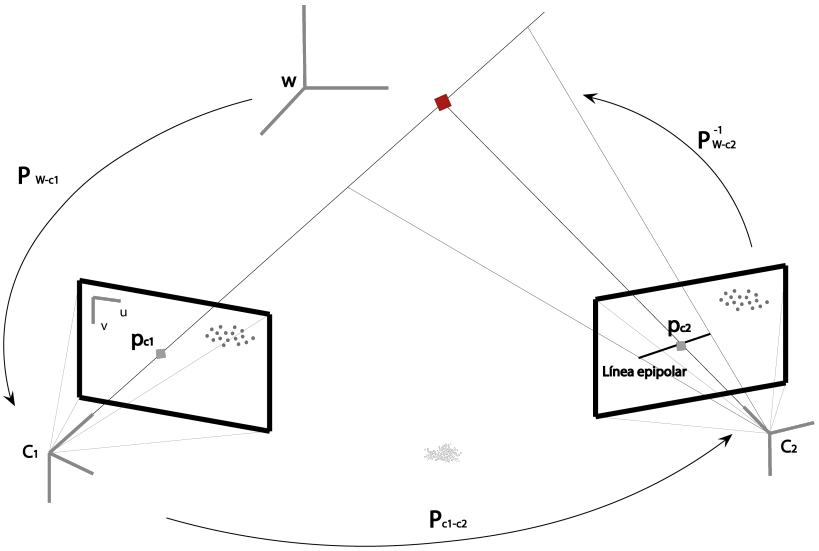


Figura 3: Modelo de proyección estéreo.

Aplicando una rotación y una traslación se realiza el cambio del sistema de referencia de la cámara al sistema de referencia global (figura 3).

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}_W = \mathbf{R} \begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix}_c + \mathbf{t} \tag{6a}$$

Dado que se van a realizar transformaciones en el espacio proyectivo resulta útil expresar el cambio de referencia en coordenadas homogéneas.

$$\mathbf{P} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \qquad \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}_W = \mathbf{P} \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix}_c \tag{6b}$$

Que las matrices de rotación pertenezcan al grupo de matrices ortonormales ($AA^T = A^T A = I$, $\forall A \in O(n)$) facilita el cálculo de la matriz para la transformación inversa. [1]

$$\mathbf{P}^{-1} = \begin{pmatrix} \mathbf{R}^T & -\mathbf{R}^T\mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \tag{6c}$$

[1] En este caso la matriz de rotación $\mathbf{R}$ es la que se construye por columnas con los vectores del sistema de referencia de la cámara expresados en el sistema de coordenadas global. De la misma forma el vector de traslación $\mathbf{t}$ expresa la posición del centro óptico de la cámara en las coordenadas del sistema global de referencia.

2.5. Proceso de proyección

Adoptando la nomenclatura y los modelos de camera pinhole, calibración de la cámara, distorsión y geometría 3D explicados en los apartados anteriores se definen dos funciones. Una función para proyectar las coordenadas 2D de un píxel en una imagen, a coordenadas 3D en el sistema de referencia global y otra que modele la transformación inversa.

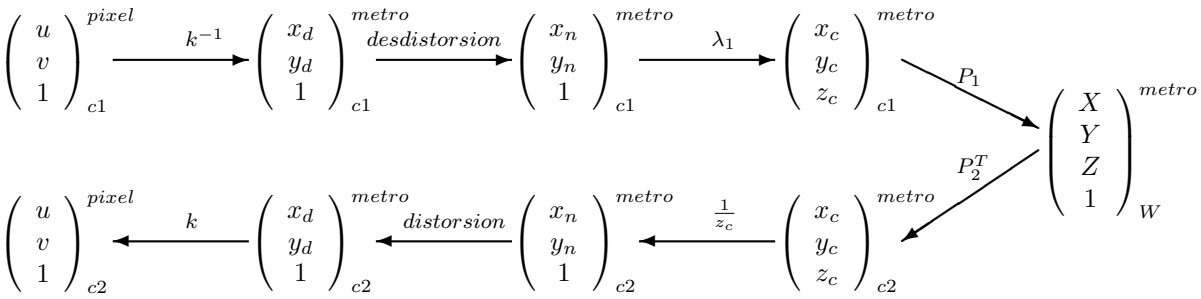


Figura 4: Proyección de un punto entre dos imágenes.

Como se observa en la figura 4, la única variable desconocida en el proceso de conversión de un punto de la imagen a un punto 3D es la profundidad a la que éste se encuentra en la dirección del rayo de proyección (en este caso λ_1 expresa el valor inverso de profundidad).

Se puede decir que: dado un punto en una imagen y conocido un punto correspondiente en otra, así como las posiciones de la cámara, se puede calcular la profundidad del punto por triangulación (figura 3). La clave de la reconstrucción de los apartados posteriores será la "búsqueda epipolar"de píxeles equivalentes en diferentes imágenes clave [2] para por triangulación calcular su profundidad y proyectar los puntos 3D [9].

[2]Para aclarar la nomenclatura, las imágenes de las que se extraigan los puntos de alto gradiente se llamarán "imágenes clave" y aquellas en las que se busquen sus emparejamientos serán "imágenes auxiliares."

3. Emparejamiento de píxeles

Métodos directos vs indirectos.

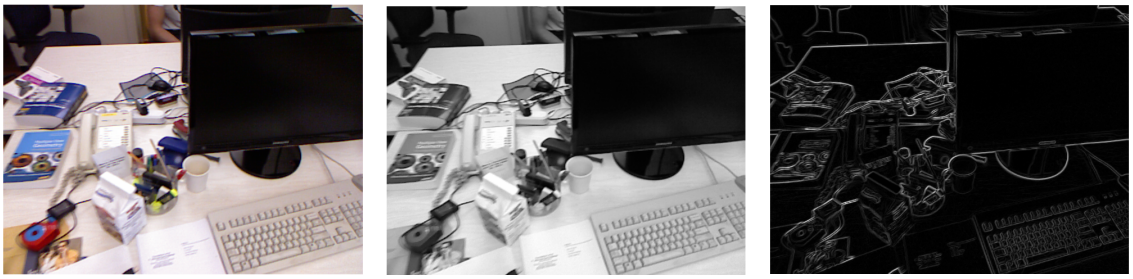
Tradicionalmente tanto los bloques de SLAM, odometría visual o mapeo han confiado en los métodos basados en características ("indirectos"), sin embargo en los últimos años un número de diferentes métodos llamados "directos" han ganado popularidad. Ambas formulaciones se basan en un modelo probabilístico que típicamente toma como entrada medidas ruidosas y calcula una estimación de los parámetros del sistema mediante la maximización de una probabilidad.

La diferencia esencial entre ambos es que mientras que los métodos indirectos preprocesan las medidas para generar una representación intermedia (como utilizar características o descriptores para calcular las coordenadas en las imágenes de puntos correspondientes) los métodos directos utilizan directamente los valores fotométricos de los sensores. De esta forma de manera general los métodos indirectos optimizan un error geométrico mientras que los directos optimizan el error fotométrico.

Existen dos razones principales por las que se espera que los métodos directos sean más robustos y precisos. Por una parte se cree en su robustez ya que usan más información al muestrear píxeles de todas las regiones de la imagen que tienen un gradiente de intensidad, incluyendo esquinas o suaves variaciones de intensidad en paredes esencialmente sin características. Por otro lado dado que no dependen del pre-procesamiento de las medidas (detectores de puntos o descriptores) existe la expectativa de que el uso de los valores reales (luz recibida en una dirección durante un cierto tiempo) proporcione mejores resultados [5].

3.1. Puntos de alto gradiente

Los puntos interesantes para reconstruir una escena son aquellos que se encuentran en las esquinas o bordes. La característica por la que se reconoce a estos puntos es un fuerte cambio en la intensidad o lo que es lo mismo un elevado valor del gradiente. Una manera de calcular una aproximación de las derivadas de una imagen es mediante la aplicación de operadores discretos de diferenciación [2].



(a) Imagen RGB. (b) Imagen en tonos de gris. (c) Representación del gradiente.

Figura 5: Resultado del cálculo del gradiente.

El operador de Sobel calcula una aproximación del gradiente mediante suavizamiento gaussiano y diferenciación por lo que el resultado es más o menos resistente al ruido. Se computan las variaciones horizontales y verticales aplicando una convolución a la imagen con dos Kernel de tamaño tres.

$$\mathbf{G}_x = \begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix} \qquad \mathbf{G}_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{pmatrix} \tag{7a}$$

En cada punto de la imagen se calcula una aproximación del gradiente y de su dirección combinando los dos resultados anteriores.

$$\mathbf{G} = \sqrt{\mathbf{G}_x^2 + \mathbf{G}_y^2} \approx \mathbf{G}_x + \mathbf{G}_y \qquad \Theta = \arctan \frac{\mathbf{G}_y}{\mathbf{G}_x} \tag{7b}$$

El operador de Sobel puede producir imprecisiones cuando se utilizan mediciones basadas en derivadas direccionales. Un ejemplo sería la utilización de la dirección del gradiente (calculado como el arcotangente de las derivadas direccionales) que se vuelve más impreciso cuanto más vertical u horizontal se encuentra.

El operador de Scharr es tan rápido y como mínimo igual de preciso que el operador Sobel por lo que se recomienda su uso cuando los Kernel utilizados sean de tamaño 3. Los Kernel que implementa son los siguientes:

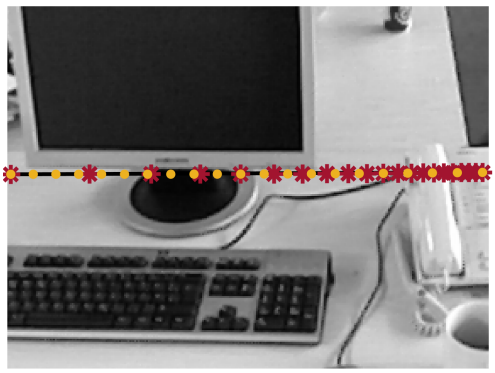
$$\mathbf{G}_x = \begin{pmatrix} -3 & 0 & +3 \\ -10 & 0 & +10 \\ -3 & 0 & +3 \end{pmatrix} \qquad \mathbf{G}_y = \begin{pmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ +3 & +10 & +3 \end{pmatrix} \tag{7c}$$

3.2. Línea epipolar

Una vez seleccionados los puntos de alto gradiente de la imagen clave se inicia el proceso de proyección en las imágenes auxiliares. Atendiendo al modelo descrito en la sección 2 los párametros requeridos son: la matriz de calibración de la cámara, los parámetros de distorsión, la profundidad de los puntos y las matrices de transformación.

El mecanismo utilizado es la proyección de los puntos de la imagen clave (por ejemplo en figura 6a) en un intervalo probable de profundidades. El resultado de esta proyección es una línea de posibles emparejamientos en la imagen auxiliar conocida como "línea epipolar".

(a) Punto en la imagen clave



(b) Epipolar en la imagen auxiliar
• Discretización en profundidad inversa
∗ Discretización en profundidad

Figura 6: Discretización de la línea epipolar

A continuación se hacen dos consideraciones importantes sobre la línea epipolar:

Línea epipolar discretizada en profundidad inversa.

La proyección de un punto del sistema de referencia de una imagen al sistema de referencia de la otra es un proceso no lineal para un intervalo de profundidades. Es decir, tal y como se aprecia en la figura 6b para un incremento constante de la profundidad los puntos proyectados aparecen cada vez más separados.

$$\rho_i = [\rho_{min} : \Delta\rho : \rho_{max}]$$

La explicación intuitiva de este resultado es que para el mismo movimiento de la cámara la proyección de los puntos que se encuentran más alejados se desplaza menos en la imagen que la de los puntos más cercanos.

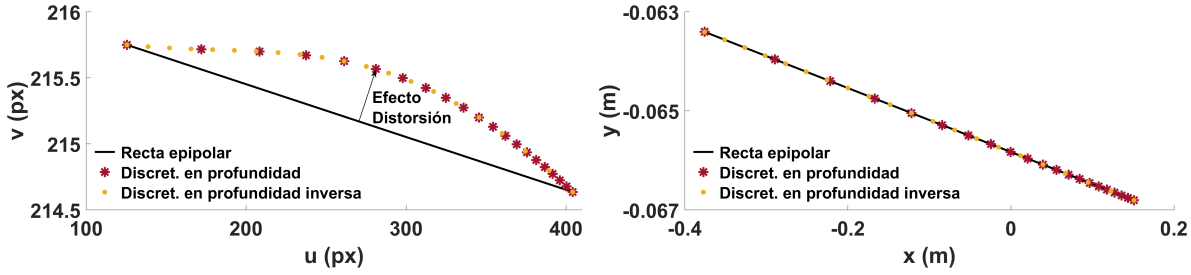
Sin embargo, para un intervalo con un incremento constante en profundidad inversa (λ) la transformación se vuelve más lineal [4] y como muestra la figura 6b los puntos aparecen equidistantes en la línea epipolar. Éste hecho facilita la búsqueda de emparejamientos para los píxeles en las imágenes auxiliares y por lo tanto mejora la precisión del cálculo de la profundidad.

$$\rho_i = \frac{1}{\lambda_i} \qquad \lambda_i = [\lambda_{min} : \Delta\lambda : \lambda_{max}]$$

Distorsión de la línea epipolar.

El efecto no lineal de la distorsión de la imagen produce que la línea epipolar no sea una recta (figura 7a). Esto dificulta recorrerla en busca de emparejamientos ya que para cada valor discreto de profundidad inversa hay que repetir el proceso de proyección.

No obstante la línea resultante de proyectar el punto en un intervalo de profundidades inversas en coordenadas normalizadas antes del proceso de distorsión sí es una línea recta (figura 7b).



(a) Epipolar en coordenadas píxel.

(b) Epipolar en coordenadas normalizadas.

Figura 7: Distorsión de la recta epipolar.

Esta recta antes del proceso de distorsión también presenta la equidistancia de los puntos al proyectar en profundidad inversa permitiendo disponer de una ecuación analítica y lineal que expresa los valores de proyección de forma continua

$$y_n = m_n x_n + b_n \tag{8}$$

3.3. Comparación de píxeles

3.3.1. Restricciones

Para que un píxel pueda ser valorado como posible emparejamiento debe cumplir tres restricciones [12]. De esta forma se reduce el número de emparejamientos erróneos y se disminuye el volumen de cálculo:

- La **magnitud del gradiente** debe de ser similar a la del píxel de la imagen clave: $G(u_j) > \lambda_G(G_1)$.
- La perpendicularidad entre el vector gradiente y la línea epipolar favorece la aparición de emparejamientos erróneos y disminuye la precisión. Por ello se define un **ángulo máximo con la epipolar**: $\theta(u_j) > \lambda_\theta$
- El **coeficiente de correlación** es una medida de similaridad entre los valores de intensidad de un recuadro de píxeles alrededor del píxel que se está comparando y del píxel de la imagen clave. Por ello se define un tamaño del recuadro (χ) y un valor mínimo del coeficiente de correlación. Al trabajar con recuadros esta magnitud permite a la vez comparar los valores de intensidades de los píxeles cercanos y considerar las direcciones de los gradientes: $C(u_j, \chi) > \lambda_C(\chi, \Theta)$.

3.3.2. Cálculo del error

Una vez seleccionados todos los píxeles de la línear epipolar que cumplen las restricciones se calcula un valor de error para cada uno de ellos. Para ello se definen los errores asociados a la intensidad y al gradiente del píxel con respecto a los de la imagen clave.

$$r_I = I_p - I(u_j) \tag{9a}$$

$$r_G = G_p - G(u_j) \tag{9b}$$

La función del error [12] se construye como la suma del cuadrado de estas diferencias normalizadas por sus respectivas varianzas:

$$e(u_j) = \frac{{r_I}^2}{{\sigma_I}^2} + \frac{{r_G}^2}{{\sigma_G}^2} \tag{9c}$$

El píxel que minimice la función error (ec: 9c) será el seleccionado como emparejamiento.

3.3.3. Precisión subpíxel

Para obtener un valor de las coordenadas del emparejamiento con precisión subpíxel se recurre a la derivada de la función error (9c):

$$\frac{\partial e(u_j)}{\partial u_j} = 2\left(\frac{r_I}{\sigma_I^2}\frac{\partial r_I}{\partial u_j} + \frac{r_G}{\sigma_G^2}\frac{\partial r_G}{\partial u_j}\right) \tag{10a}$$

Las derivadas parciales de la intensidad y el gradiente se obtienen promediando los valores del píxel siguiente y el anterior en la epipolar.

$$i(u_0) = \left.\frac{\partial r_I}{\partial u_j}\right|_{u_0} = -\frac{I(u_0+1) - I(u_0-1)}{2} \tag{10b}$$

$$g(u_0) = \left.\frac{\partial r_G}{\partial u_j}\right|_{u_0} = -\frac{G(u_0+1) - G(u_0-1)}{2} \tag{10c}$$

Se realiza una aproximación de primer orden en el desarrollo de Taylor de las funciones error.

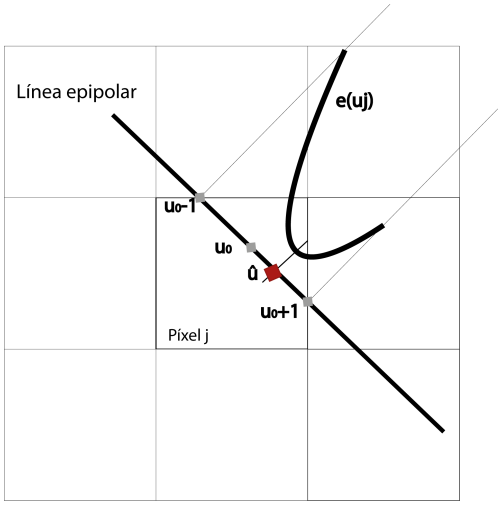
$$r_I(u^*) = r_I(u_0) + \left.\frac{\partial r_I}{\partial u_j}\right|_{u_0} (u^* - u_0) \tag{10d}$$

$$r_G(u^*) = r_G(u_0) + \left.\frac{\partial r_G}{\partial u_j}\right|_{u_0} (u^* - u_0) \tag{10e}$$

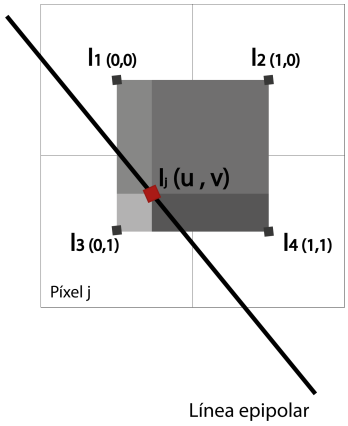
Sustituyendo e igualando a cero la derivada (ec 10a) se obtiene el valor de la coordenada con precisión subpíxel:

$$u^* = u_0 - \frac{r_G(u_0) g\sigma_I^2 + r_I(u_0) i\sigma_G^2}{g^2\sigma_I^2 + i^2\sigma_G^2} \tag{10f}$$

La figura 8a muestra el proceso de cálculo de las coordenadas del emparejamiento con precisión subpíxel.



(a) Precisión subpíxel.



(b) Interpolación bilineal.

Figura 8: Procesos de cálculo subpíxel.

Para la obtención del gradiente y la intensidad de los puntos a lo largo de la línea epipolar durante todo el proceso se ha aplicado una interpolación bilineal entre píxeles como la mostrada en la figura 8b. Si se consideran coordenadas unitarias de los puntos la expresión simplificada de la interpolación bilineal queda de la siguiente manera:

$$I = \begin{pmatrix} 1-u & u \end{pmatrix} \begin{pmatrix} I_1 & I_2 \\ I_3 & I_4 \end{pmatrix} \begin{pmatrix} 1-v \\ v \end{pmatrix} \tag{11}$$

4. Cálculo probabilístico de la profundidad inversa

El objetivo de esta sección es no solo el cálculo de la profundidad de los puntos a partir de su emparejamiento previo, sino asociar a estos valores una incertidumbre basada en el paralaje de las imágenes y a partir de ellas fusionarlos de forma congruente.

4.1. Profundidad inversa

Con el punto de alto gradiente de la imagen clave y su emparejamiento en la imagen auxiliar se procede a calcular las profundidades. Para ello se proyectan ambos puntos en el sistema de referencia global.

$$\mathbf{R}_1 \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} \frac{1}{\lambda_1} + \mathbf{t}_1 = \mathbf{R}_2 \begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} \frac{1}{\lambda_2} + \mathbf{t}_2$$

El proceso geométrico a resolver es el de corte de dos rectas que pertenecen a un mismo plano en el espacio. Cabe destacar que en la mayoría de los algoritmos de reconstrucción estas rectas no se cortan (ya que el emparejamiento se hace de forma independiente) y hace falta calcular la profundidad que minimiza la distancia entre ambas. En este caso, cómo el emparejamiento se realiza a lo largo de la línea epipolar, el cálculo de la profundidad es simplemente la resolución de un sistema compatible determinado. Para simplificarlo se calcula la rotación y la traslación de la cámara clave respecto de la cámara auxiliar:

$$\mathbf{R}_{21} = \mathbf{R}_2^T \mathbf{R}_1 \qquad \mathbf{t}_{21} = \mathbf{R}_2^T (\mathbf{t}_1 - \mathbf{t}_2)$$

Se resuelve el producto escalar entre la matriz de rotación y las componentes del punto en la imagen clave:

$$R_x = \mathbf{R}_{21}^{(1,1)} x_1 + \mathbf{R}_{21}^{(1,2)} y_1 + \mathbf{R}_{21}^{(1,3)} \qquad R_y = \mathbf{R}_{21}^{(2,1)} x_1 + \mathbf{R}_{21}^{(2,2)} y_1 + \mathbf{R}_{21}^{(2,3)}$$

$$t_x = \mathbf{t}_{21}^{(1)} \qquad t_y = \mathbf{t}_{21}^{(2)}$$

Y se plantea el sistema de dos ecuaciones con dos incógnitas (las profundidades del punto con respecto a cada cámara).

$$R_x \frac{1}{\lambda_1} + t_x = x_2 \frac{1}{\lambda_2} \qquad R_y \frac{1}{\lambda_1} + t_y = y_2 \frac{1}{\lambda_2}$$

Finalmente se obtiene la profundidad inversa del punto en función de las coordenadas del emparejamiento. Dado que las coordenadas del emparejamiento están relacionadas por la ecuación de la recta epipolar ($y_2 = mx_2 + b$), la profundidad inversa se puede expresar de la siguiente manera:

$$\lambda_1 = \frac{y_2 R_x - x_2 R_y}{x_2 t_y - y_2 t_x} = \frac{(mR_x - R_y)x_2 + bR_x}{(t_y - mt_x)x_2 - bt_x} \tag{12a}$$

4.2. Incertidumbre

La ecuación (12a) relaciona la profundidad inversa con la componente horizontal del emparejamiento. Así la derivada de la profundidad inversa de un punto resulta:

$$\frac{\partial \lambda_1}{\partial u_2} = \frac{1}{f_x} \frac{(mR_x - R_y)((t_y - mt_x)x_2 - bt_x) - ((mR_x - R_y)x_2 + bR_x)(t_y - mt_x)}{((t_y - mt_x)x_2 - bt_x)^2} \tag{13a}$$

Con la derivada, mediante el método de la propagación de errores, se estima la incertidumbre de la medición de la profundidad inversa a partir de la incertidumbre del emparejamiento del píxel.

$$\sigma_{\lambda_1}^2 = \left(\frac{\partial \lambda_1}{\partial u_2}\right)^2 \sigma_{u_2}^2 \tag{13b}$$

La estimación de la incertidumbre del emparejamiento (σ_{u_2}) se puede relacionar con el ruido de la imagen simplemente propagando las varianzas de intensidad y gradiente de la imagen en la ecuación 10f. Sin embargo considerando esta variabilidad constante para todos los píxeles fijamos que la incertidumbre de la profundidad dependa solo de la posición relativa entre las dos imágenes.

En resumen, este modelo considera la incertidumbre de la medida de profundidad derivada de la posición relativa entre dos imágenes, despreciando el ruido y la imprecisión en la medición de la posición de la cámara.

4.3. Hipótesis de fusión de profundidades inversas

Para mejorar la precisión y eliminar posibles errores se pueden calcular varios valores de la profundidad de un punto buscando emparejamientos en un número más amplio de imágenes. De esta forma se consiguen varias hipótesis para los valores de profundidad cuya compatibilidad hay que comprobar para después fusionarlos en un solo valor.

Compatibilidad de hipótesis.

El número de hipótesis obtenidas para un píxel puede ser menor que el número de imágenes comprobadas debido a cambios importantes de la vista de la escena, el rango de profundidades de la epipolar no sea suficiente, los píxeles no superen las restricciones para el emparejamiento u oclusiones.

Por otra parte no todas las hipótesis tienen por qué ser correctas dado que existen píxeles con características similares que producen emparejamientos erróneos. La compatibilidad de dos hipótesis se comprueba con una probabilidad del 95 % con el test de χ^2.

$$\frac{(\lambda_a - \lambda_b)^2}{\sigma_a^2} + \frac{(\lambda_a - \lambda_b)^2}{\sigma_b^2} < 5{,}99 \tag{14}$$

Para cada una de las hipótesis se comprueba su compatibilidad con el resto. Si el número de compatibilidades supera un cierto límite ($n > \lambda_N$) las profundidades se fusionan en una sola distribución $N(\lambda_p, \sigma_{\lambda_p}^2)$.

Fusión de profundidades.

De forma directa la fusión de las hipótesis resultantes se puede ejecutar con una media de los valores de profundidad ponderados con sus incertidumbres. La incertidumbre asociada a la profundidad resultante es la suma de todos los valores relativos a cada hipótesis.

$$\lambda_p = \frac{\sum\limits_n \frac{1}{\sigma_j^2}\lambda_j}{\sum\limits_n \frac{1}{\sigma_j^2}} \qquad \sigma_{\lambda_p}^2 = \frac{1}{\sum\limits_n \frac{1}{\sigma_j^2}} \tag{15}$$

Optimización de la profundidad.

El método de la media ponderada toma en consideración las diferentes precisiones de cada hipótesis, sin embargo, no considera que tanto el modelo de proyección (y también el de distorsión) es no lineal incluso discretizado en profundidad inversa. Por ello una manera de mejorar la precisión es plantear el problema de fusión de hipótesis de profundidades como la minimización de un error de reproyección del píxel en las diferentes imágenes.

$$\overline{e}(\lambda_p)_i = (\overline{uv}_2)_i - f_p(\overline{uv}_1, \lambda_p, \mathbf{R}_i, \bar{t}_i) \tag{16}$$

La minimización se realiza con una optimización de Gauss-Newton con la siguiente función de coste (el anexo B: Optimización de la profundidad, contiene el desarrollo completo del error y los jacobianos.):

$$C(\lambda_p) = \sum_n \overline{e}(\lambda_p)\mathbf{W}(\sigma_{\lambda_p})\overline{e}(\lambda_p) \tag{17}$$

4.4. Filtro entre hipótesis

Una vez obtenido el mapa de profundidades inversas asociado a los píxeles de una imagen, se va a aplicar un filtro que elimine los emparejamientos erróneos y suavice la reconstrucción. Para conservar la profundidad de un píxel la hipótesis asociada debe ser compatible con la de al menos dos píxeles de los ocho de alrededor según la ecuación 14. Si es compatible, se recalcula el valor de profundidad asignándole el promedio ponderado de las hipótesis válidas con la ecuación 15, pero asignándole el menor de los valores de incertidumbre. De esta manera se suaviza el mapa conservando las esquinas ya que solo los valores compatibles son promediados.

Densificación del mapa.

Con el objetivo de conseguir una reconstrucción más densa, en el mismo mapa de profundidades, se asigna a los píxeles sin hipótesis pero que estén rodeados por al menos dos hipótesis compatibles (con la incertidumbre mínima) el valor promediado de éstas. De esta forma se consigue que la reconstrucción aumente en densidad.

5. Implementación, experimentos y resultados en Matlab

Para valorar la influencia en la precisión de los procesos explicados anteriormente en el desarrollo de la reconstrucción, se ha implementado un programa en Matlab que permite variar todos los parámetros y comparar los resultados de las diferentes escenas. Este apartado también incluye la explicación de como se han resuelto algunos de los problemas que no han sido abordados exhaustivamente en este trabajo con la pretensión de clarificar los resultados obtenidos.

5.1. Selección de imágenes auxiliares

La selección de las imágenes utilizadas para buscar los emparejamientos y extraer las hipótesis de profundidad es un elemento clave en la precisión de la reconstrucción. El aumento de la distancia y el ángulo de paralaje entre dos imágenes mejora la precisión al disminuir las incertidumbres de las medidas, produciendo sin embargo, la reducción del campo visual común, crecimiento de las diferencias entre imágenes y favoreciendo la aparición de oclusiones.

En una implementación que busca la máxima precisión, las imágenes elegidas serían aquellas que disminuyesen las incertidumbres teniendo en cuenta todo el vídeo disponible. Por el contrario, una reconstrucción en tiempo real (por ejemplo la generación de un mapa para un sistema de SLAM), dispondría solo de aquellas imágenes cercanas temporalmente a la última imagen disponible.

En este caso, la solución elegida consiste en una búsqueda espacial de las imágenes, seleccionando cuatro anteriores a la imagen clave (a distancias de 15, 20, 25 y 30 centímetros) y tres posteriores (a 15, 20 y 25 centímetros). Esta alternativa permite conservar un cierto nivel de precisión al proporcionar un alto paralaje y dos puntos de vista.

5.2. RGB-D SLAM Dataset and Benchmark

Para valorar la precisión de los resultados de la reconstrucción se han utilizado los vídeos del "RGB-D SLAM Dataset and Benchmark" proporcionado por "Computer Vision Group, Faculty of Informatics, Technical University of Munich" [15].

Además de las calibraciones intrínsecas de las diferentes cámaras, los "datasets" consisten en datos RGB-D con medidas precisas de su posición en cada instante. Las imágenes de profundidad se obtienen con una cámara Kinect a una frecuencia de 30 Hz y resolución 640x480 píxel. Las posiciones provienen de un sistema de alta precisión de captura del movimiento con ocho cámaras de alta velocidad (100 Hz). Se aportan escenas que permiten testar diferentes objetivos, por ejemplo: series "xyz" o "rpy" para traslaciones o rotaciones puras, vídeos con largas traslaciones y cierres de bucle, escenas variando la textura y la estructura, reconstrucciones de objetos 3D y series con objetos dinámicos. Por otra parte, se ofrecen además los datos en formato "ROS bag" (Robotic Operating System) que permite testar fácilmente el funcionamiento del algoritmo en tiempo real.

Gracias a la calibración de la Kinect, las imágenes de profundidad son reproyectadas en las imágenes RGB, de modo que hay una correspondencia 1:1 entre los píxeles de ambas (figura 9). Sin embargo, es relevante hacer notar que los resultados de precisión obtenidos comparando los valores de profundidad de la reconstrucción con los valores proporcionados por la Kinect solo se pueden usar de forma cualitativa para la comparación de procedimientos debido a dos particularidades de los datasets:

- La cámara kinect cuenta para la medición de profundidades con un sensor infrarrojo. A pesar de que los mayores errores que puede generar este sensor se han resuelto con una buena calibración

existen dos fuentes más de error: la iluminación y la geometría de la escena. La luz intensa genera bajo contraste en las imágenes infrarrojas, y en consecuencia, valores atípicos o una brecha en la nube de puntos. Por otra parte, la geometría de la escena influye ya que se produce un aumento de la incertidumbre al crecer la distancia al sensor, o por supuesto la falta de información en zonas ocluídas o ensombrecidas.

- En el conjunto de vídeos grabados con la cámara "freiburg 3" las imágenes RGB han sido desdistorsionadas y por lo tanto ya no coinciden con las imágenes de profundidad.

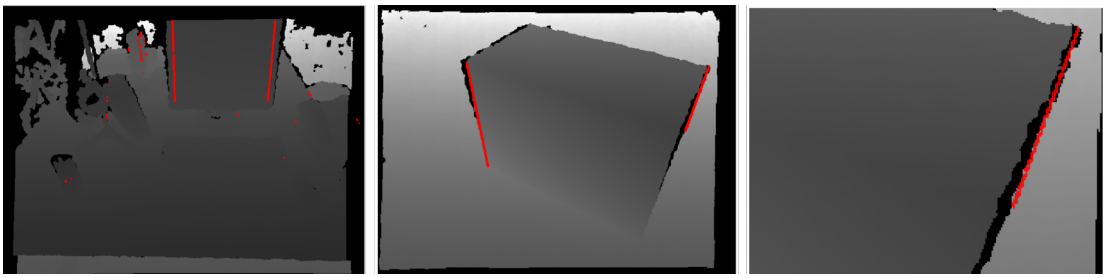
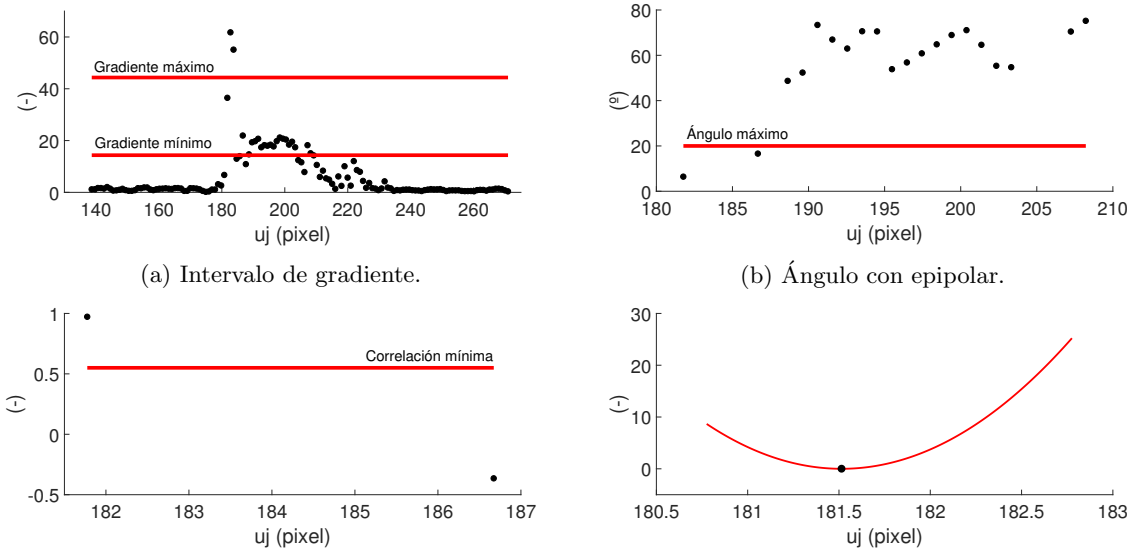


Figura 9: Reproyección de puntos en las imágenes de profundidad.

5.3. Experimentos para la evaluación de procesos

Para la implementación en tiempo real del algoritmo un elemento a tener en cuenta es el grado de restricción que aportan las condiciones para el emparejamiento de píxeles (sección 3.3.1). Con ello se pueden ordenar de la más a la menos restrictiva haciendo el algoritmo más eficiente. La figura 10 muestra como se organizan las restricciones e ilustra un ejemplo de la función de error minimizada en el proceso de precisión subpíxel (figura 10d).



(a) Intervalo de gradiente.

(b) Ángulo con epipolar.

(c) Coeficiente de correlación.

(d) Precisión subpíxel.

Figura 10: Proceso de emparejamiento de píxeles.

En el proceso de cálculo de la profundidad inversa se han considerado tres mecanismos para mejorar la precisión. La tabla 1 incluye los resultados de precisión obtenidos en la reconstrucción de una escena.

El método de interpolación bilineal propuesto para la aproximación del gradiente y la intensidad de puntos de la imagen disminuye de forma muy importante el número de puntos además de ser costoso computacionalmente. Dado que los puntos de alto gradiente de la imagen se encuentran en zonas de fuertes cambios la aproximación bilineal no funciona correctamente para estos píxeles. Por ello se decide no incluir este método en la versión final del programa.

Método	Nºpuntos	Media	% Media	Mediana	% Mediana
Reconstrucción completa	741	4,2826	+1,1057	4,2184	+0,4182
Precisión subpíxel	741	4,2897	+0,9431	4,2249	+0,2655
Optimización	742	4,3237	+0,1569	4,2295	+0,1575
Interpolación bilineal	653	4,2725	+1,3339	4,2192	+0,4013
Reconstrucción básica	742	4,3305		4,2362	

Tabla 1: Caracterización del error de precisión (mm) respecto de diversos procedimientos.

La optimización propuesta para el cálculo de la profundidad inversa converge claramente a valores cercanos a los de la media ponderada. Si bien esta forma de calcular la profundidad mejora la precisión, depende en mayor medida de los valores de las incertidumbres lo que la hace susceptible de empeorar los resultados.

La implementación adecuada de la precisión subpíxel mejora de forma importante los resultados confiando en los valores de gradiente e intensidad de los píxeles vecinos. Sin embargo una implementación correcta requiere el cálculo de la distancia que la línea epipolar deja en el interior del píxel así como la obtención de las características de los píxeles siguiente y anterior lo que incrementa la cantidad de cálculos.

5.4. Experimentos para la evaluación de parámetros

La figura 11 representa la cantidad de puntos de alto gradiente que se extraen y los puntos 3D que finalmente se obtienen variando el límite de selección del gradiente a un número entero de veces la desviación típica. Se puede observar como aumentar la cantidad de puntos extraídos no significa obtener la misma variación en el número de resultados. Ello significa que incrementar el número de puntos de forma exagerada hace el algoritmo ineficiente y sugiere la elección de una proporción adecuada. Además la precisión de los resultados es bastante constante a pesar de experimentar una cierta mejora para un volumen menor de puntos.

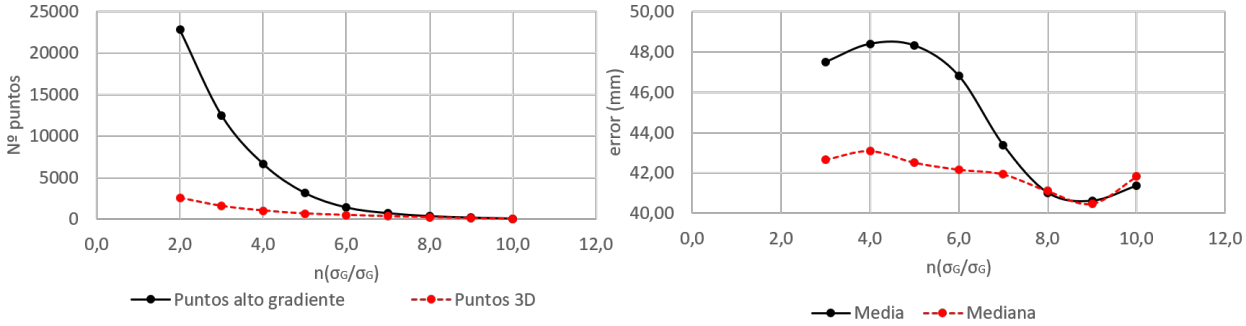


Figura 11: Influencia del gradiente límite en la precisión.

Una posible variación para mejorar la precisión sería discretizar la línea epipolar con valores inferiores al píxel. No obstante, la figura 12 muestra como no se produce variación ni en la cantidad de puntos 3D ni en la precisión al intensificar la discretización. Lo que sí es reseñable es como el hecho de implementar la profundidad subpíxel otorga una mejora de los resultados. En el caso de querer hacer el algoritmo más rápido en detrimento de la precisión, con pasos de discretización mayores, la profundidad subpíxel se convierte en un mecanismo de interpolación entre dos medidas consecutivas. Se puede observar como este proceso suaviza el empeoramiento de los resultados al reducir la discretización.

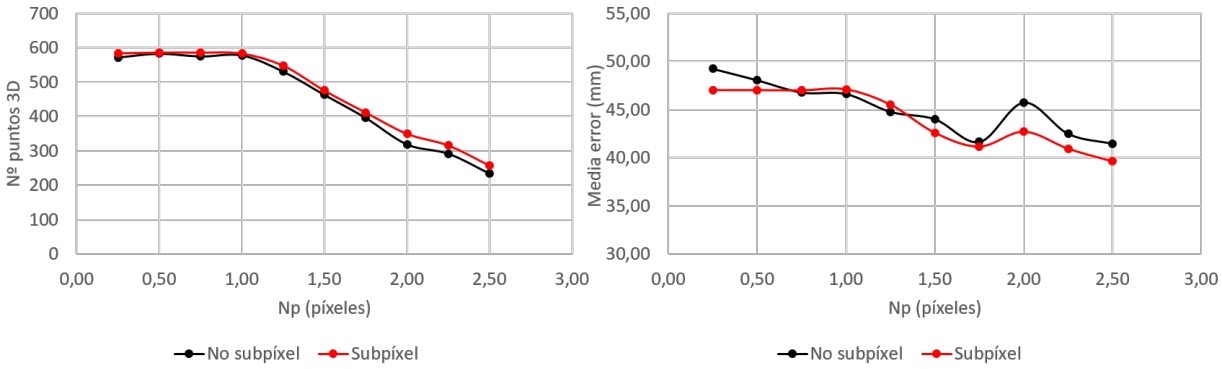


Figura 12: Influencia del paso de recorrido de la epipolar en la precisión.

5.5. Experimentos para la evaluación de filtros

El procedimiento de fusión de hipótesis explicado incluía un número mínimo de compatibilidades. En la figura 13 se enseña como a partir de cuatro ya no se produce una mejora de la precisión aunque sí una reducción del volumen de resultados. La influencia del filtro de mediana en este experimento es menor ya que el número de puntos de alto gradiente alimentado es pequeño.

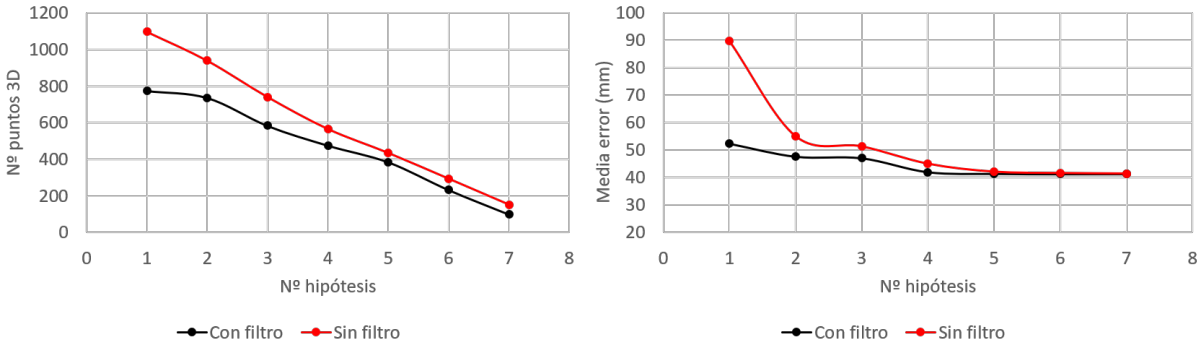
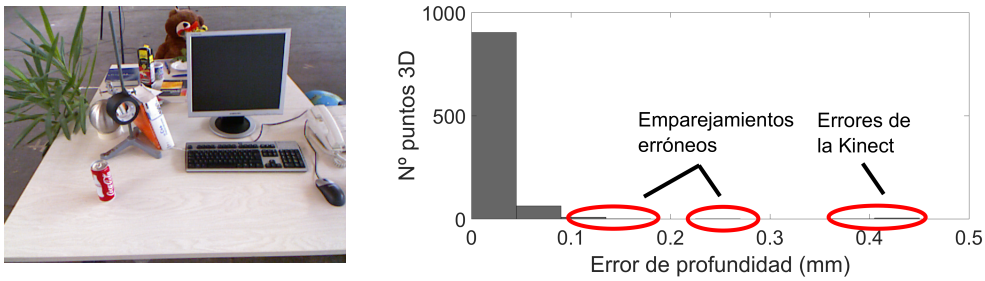


Figura 13: Influencia del número de compatibilidades en la precisión.

La escena reconstruída se muestra en la figura 14 junto con el histograma del error de profundidades. Se aprecian algunos errores excesivamente grandes que podrían estar asociados a la falta de información proporcionada por la Kinect.

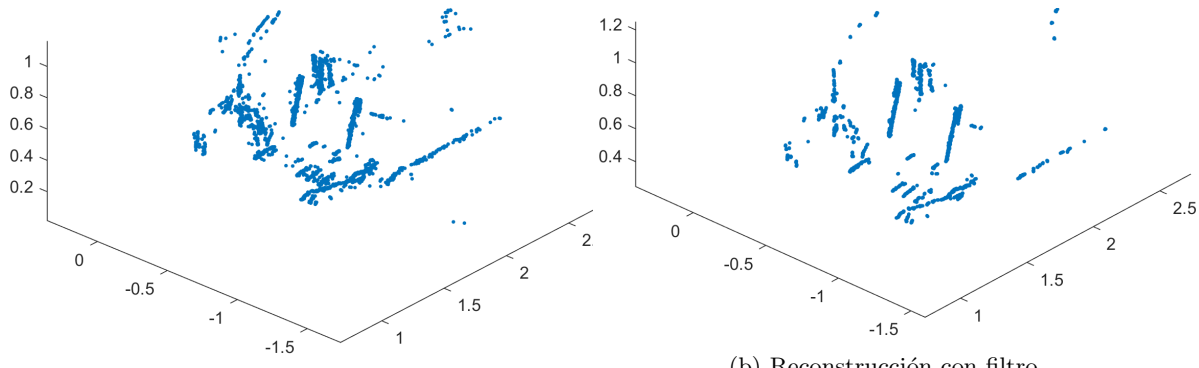


(a) Imagen de *fr2/xyz*.

(b) Histograma del error.

Figura 14: Precisión, emparejamientos erróneos e influencia de la Kinect.

La figura 15 corresponde a un experimento con un mayor caudal de puntos y deja apreciar como el filtro de mediana reduce sustancialmente los emparejamientos erróneos y el ruido.



(a) Reconstrucción sin filtro.

(b) Reconstrucción con filtro.

Figura 15: Reconstrucción de *fr2/xyz*.

6. Implementación y resultados en C++ y ROS

La implementación adecuada del código en C++ permite repetir el proceso ejecutado en Matlab con un número amplio de imágenes clave que permitan la reconstrucción completa de la escena. Sin embargo, ello genera la necesidad de mejorar algunos procesos muy influyentes en la precisión y velocidad del algoritmo: selección de píxeles, elección de imágenes auxiliares y actualización de la profundidad.

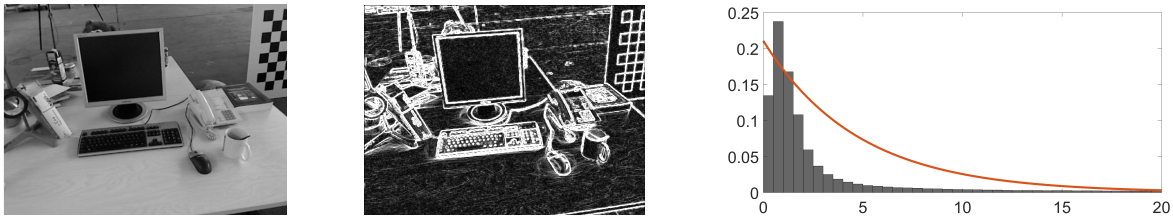
6.1. Selección de puntos de alto gradiente

Como se ha demostrado en secciones anteriores la elección inteligente de los puntos de alto gradiente influye directamente en la precisión y eficiencia de la reconstrucción. El objetivo de esta sección es sistematizar el proceso de selección de manera que se haga en el menor tiempo posible (de cara a la implementación en tiempo real), ajustándose a un número máximo y mínimo de puntos con los que alimentar al sistema, que tengan un gradiente suficiente y procedan de todas las regiones de la imagen.

Se plantea la observación de la distribución probabilística que presenta el gradiente, la implementación de una malla que divida las imágenes en regiones y el cálculo de un coeficiente de relación entre las desviaciones típicas de la intensidad y el gradiente.

6.1.1. Aproximación exponencial del gradiente

Como primera aproximación se va a representar el gradiente de los puntos de la imagen con una distribución exponencial (figura 16). Esto permite por una parte describir el proceso por el que en una escena los píxeles aparecen en regiones de intensidades similares y por lo tanto el número de puntos de alto gradiente disminuye exponencialmente.



(a) Imagen procesada. (b) Gradiente de la imagen. (c) Histograma del gradiente.

Figura 16: Resultado del cálculo del gradiente.

A continuación se muestran las expresiones de la media, la varianza y la función de probabilidad acumulada de la distribución exponencial:

$$\mu = \frac{1}{\chi} \qquad \sigma^2 = \frac{1}{\chi^2} \qquad P\left(X \leq x,\right) = \begin{pmatrix} 0 & ,x < 0 \\ 1 - e^{-\chi x} & ,x \geq 0 \end{pmatrix} \tag{18}$$

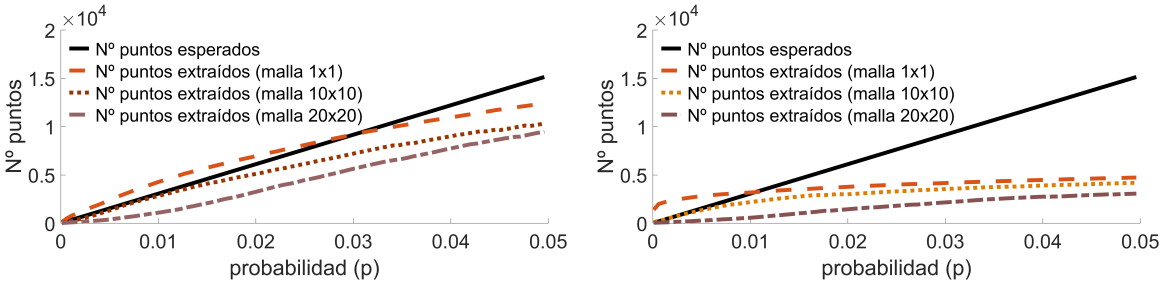
Dada su simplicidad se puede calcular fácilmente el valor límite del gradiente asociado a una cantidad de puntos de la imagen despejando de la función de probabilidad acumulada.

$$x_{th} = -ln(p)\sigma_G \tag{19}$$

6.1.2. Malla de distribución

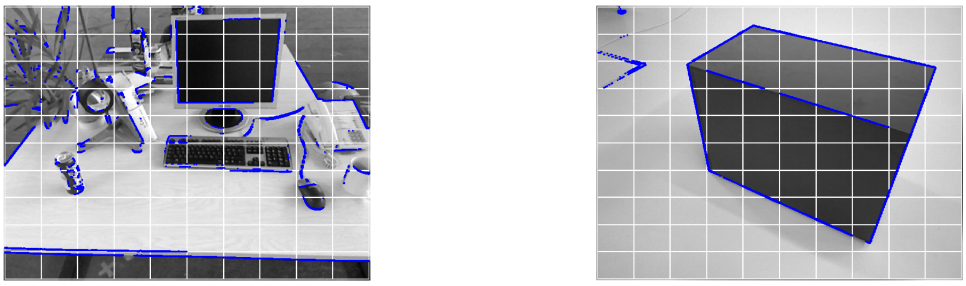
En la figura 17 se muestra la relación entre la cantidad de puntos que se querían extraer y los que la aproximación exponencial finalmente obtiene. Se puede observar que se generan errores de más de mil puntos entre el número estimado y el obtenido.

Para desarrollar mejor la extracción se propone calcular el límite del gradiente respecto de la desviación típica de cada zona de la imagen. Para ello se implementa una malla que divide la imagen en regiones y se establece un límite de gradiente mínimo para extraer puntos de cada una.



(a) Extracción (*fr2/xyz*).

(b) Extracción (*fr3/cabinet*).



(c) Malla (*fr2/xyz*).

(d) Malla (*fr3/cabinet*).

Figura 17: Proceso de extracción de puntos.

Con esta herramienta se consigue mejorar por una parte la estimación de puntos extraídos con una tendencia a obtener siempre una cantidad conservadora de cara a la implementación en tiempo real. Por otra parte se logra una distribución más uniforme de puntos en la imagen. Se espera con este método reducir el tiempo dedicado a la extracción de puntos de alto gradiente ya que no hará falta buscarlos en zonas de baja desviación típica.

6.1.3. Correlación del operador gradiente

Para evitar el cálculo de la desviación típica del gradiente de cada imagen necesaria en este proceso y en otros durante la reconstrucción y ya que el gradiente es una función de la intensidad se trata de relacionar ambas magnitudes mediante un coeficiente de relación [12].

$$\sigma_G^2 = \nu\sigma_I^2 \tag{20}$$

Para la estimación de este coeficiente se ha calculado su valor en una serie de imágenes de los distintos 'datasets' utilizados en este trabajo [15] con los operadores Scharr y Sobel. La figura 18 representa los resultados obtenidos en el experimento.

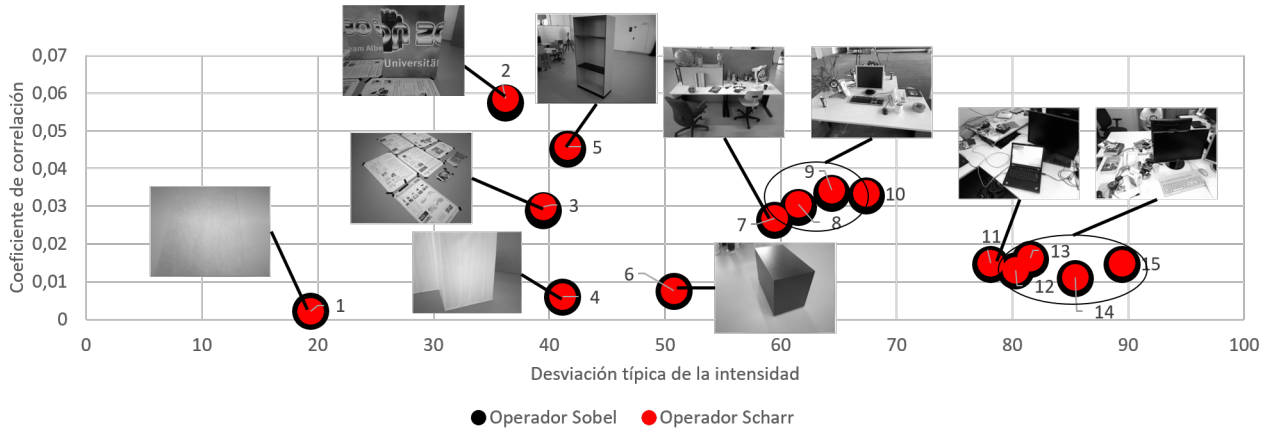


Figura 18: Coeficiente de correlación en los distintos datasets.

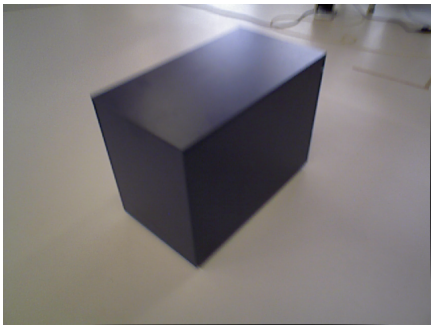
De este experimento se extraen varias conclusiones (los datos se pueden consultar en el Anexo C).

- El coeficiente de relación varía dependiendo de la textura de la imagen, dando valores bajos a vídeos con tramas regulares (como 1,4 y 6) y presentando valores comparables del coeficiente para vídeos de características semejantes.
- La diferencia entre utilizar el operador Scharr o Sobel se encuentra en que el primero da unos valores mayores de gradiente con una mayor dispersión lo que facilita el emparejamiento de los puntos.

6.2. Resultados de la reconstrucción en C++

Con el objetivo de valorar el comportamiento del algoritmo en diferentes situaciones se ha reconstruido una escena de escritorio (figura 19a: *fr3/long office household*) y un objeto cúbico (figura 19b: *fr3/cabinet*).

(a) *fr3/long office household.*



(b) *fr3/cabinet.*

Figura 19: Imágenes de las escenas reconstruidas.

En la tabla 2 se recogen los resultados de ambas reconstrucciones. Estas reconstrucciones se hicieron eliminando del algoritmo el cálculo del coeficiente de correlación ya que debido a una mala

implementación perjudicaba los resultados temporales. Las imágenes son desdistorsionadas al principio del proceso con lo que se pueden eliminar las ecuaciones de distorsión del código acelerando el funcionamiento.

	fr3/long office household	*fr3/cabinet*
Tiempo reconstrucción (s)	126,000	28,946
Tiempo vídeo (s)	87,090	38,580
Nº imágenes clave	92	40
Nº imágenes vídeo	2585	1147
Nº puntos de alto gradiente	1018782	227111
Nº puntos 3D sin densificación	34745	19145
Nº puntos 3D con densificación	81475	40748

Tabla 2: Resultados de las reconstrucciones en C++.

Por una parte se observa como el tiempo consumido por la reconstrucción de la escena de escritorio es bastante superior al tiempo del vídeo. También se aprecia como el proceso de densificación prácticamente duplica el número de puntos 3D obtenidos.

Temporización del algoritmo

Con la finalidad de poder obtener una implementación en tiempo real, la tabla 3 engloba las duraciones de los procesos de la reconstrucción de la escena de escritorio.

Ya que el proceso de emparejamiento consume el 86 % del tiempo total, el modo más efectivo de reducirlo es la disminución del número de iteraciones que realiza. Por otro lado el proceso de densificación consume un tiempo prácticamente irrelevante del algoritmo.

Proceso	Subproceso	$t_{subproceso}(s)$	% Subproceso	$t_{global}(s)$	% Global
Preparación de variables				0,416	0,33 %
Cálculo del gradiente	Scharr	1.438	74,0 %		
	Magnitud	0,505	26,0 %		
	Total		100,0 %	1,943	1,54 %
Extracción de puntos				2,197	1,74 %
Emparejamiento de puntos	Epipolar	105,871	97,4 %		
	Subpíxel	2,788	2,6 %		
	Total		100,0 %	108,659	86,24 %
Fusión de hipótesis	Cálculo profundidad	3,190	27,9 %		
	Fusión	6,608	57,7 %		
	Optimización	1,650	14,4 %		
	Total		100,0 %	11,447	9,09 %
Filtro de hipótesis	Filtro de mediana	0,776	58,0 %		
	Densificación	0,562	42,0 %		
	Total		100,0 %	1,338	1,06 %
Proceso total				126,000	100,0 %

Tabla 3: Temporización de la reconstrucción de la escena *fr3/long office household*.

Visualización de resultados

Las figuras 20 y 21 muestran los resultados de las reconstrucciones ejecutadas. De forma cualitativa se observa que ambas son semidensas con pocos puntos 3D erróneos en comparación a la cantidad global. Sin embargo dada la densidad de las reconstrucciones sería conveniente implementar un filtro de puntos realizado en este caso entre imágenes clave para eliminar los últimos errores [12].

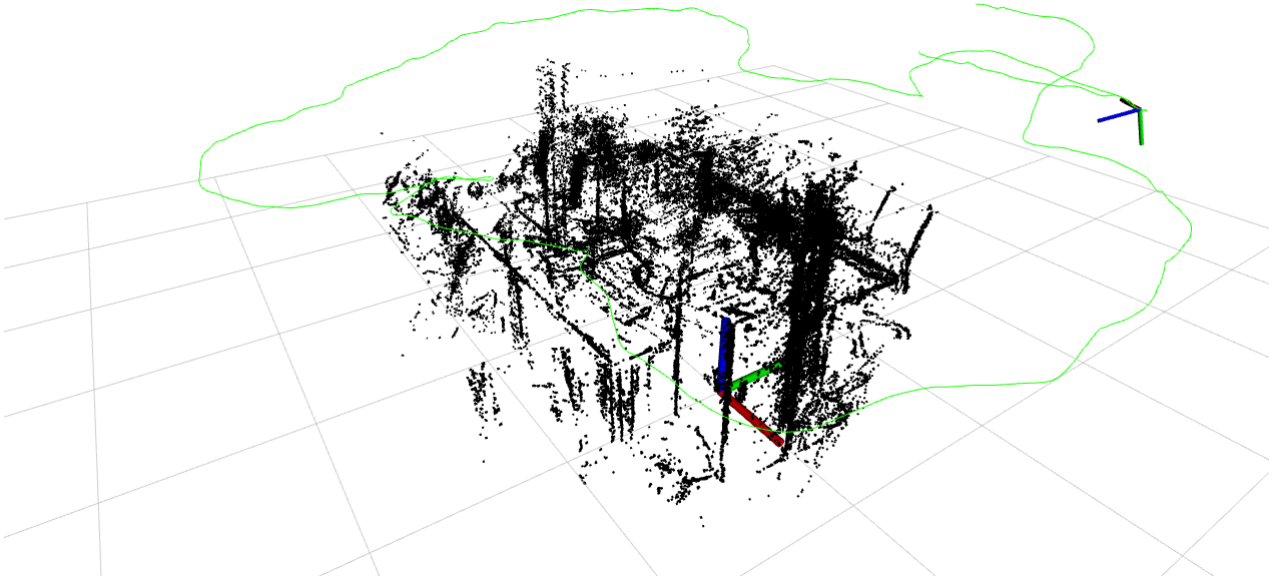


Figura 20: Reconstrucción *fr3/long office household*.

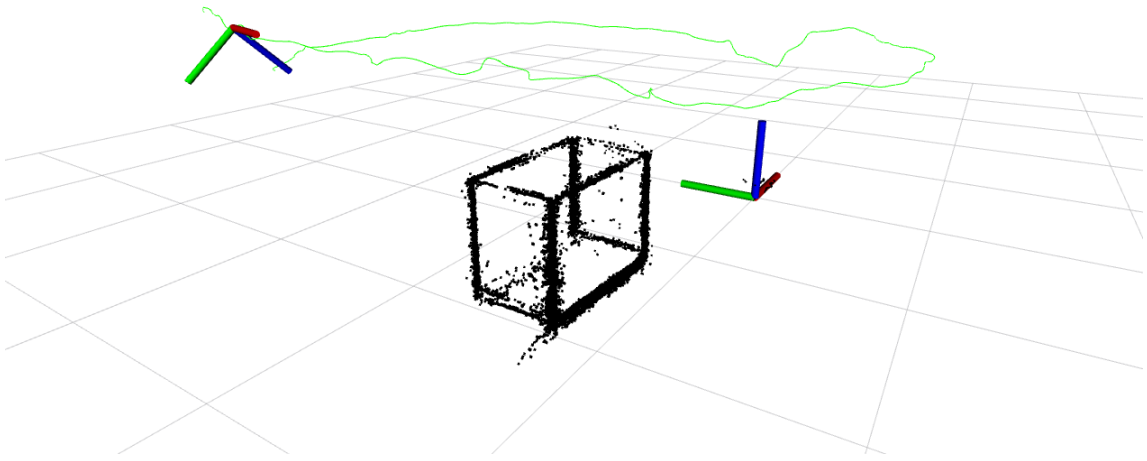


Figura 21: Reconstrucción *fr3/large cabinet*.

7. Aplicación del algoritmo

Un algoritmo de reconstrucción de estas características sería el complemento en la generación del mapa de un sistema instantáneo de localización o en el entendimiento del entorno de una aplicación de realidad aumentada. Es por ello que a partir de este punto surgen dos nuevos problemas: el acoplamiento de la posición como señal de entrada del algoritmo desde un sistema externo (SLAM o "Instant Tracking") y la implementación en tiempo real.

7.1. Implementación en tiempo real

Los resultados de la implementación en tiempo real han sido obtenidos en un portátil con procesador Intel i7-7500U. La reconstrucción se ha simulado sin ninguna paralelización.

Robot Operating System (ROS)

Robot Operating System (ROS) es un entorno orientado a la generación de software para robots. Es una colección de herramientas, librerías y convenciones cuyo objetivo es la simplificación de las tareas a la hora de crear comportamientos complejos y robustos para robots. Está basado en un sistema de nodos independientes que se comunican con un nodo general. Este sistema se basa en la publicación y la suscripción de los nodos a todo tipo de mensajes diferentes de forma asíncrona [6].

Para simular el comportamiento en tiempo real del algoritmo de reconstrucción se han utilizado dos nodos. Uno simula un sistema que provee en tiempo real las imágenes y la posición de una cámara (SLAM o "Instant Tracking"). El otro contiene el algoritmo de reconstrucción, que suscrito a los mensajes del primero utiliza la información que recibe para el cálculo.

Adaptaciones del algoritmo

A continuación se describen las modificaciones realizadas para simular la reconstrucción en tiempo real:

- Las imágenes clave se seleccionan tratando de seguir el vídeo pero dejando margen para seleccionar las auxiliares en un entorno de 60 cm.
- Para acelerar el algoritmo se implementa un proceso de actualización del intervalo de profundidades de búsqueda en la epipolar, de manera que una vez obtenido el primer valor de profundidad posible para un punto, el resto de hipótesis se realizan alrededor de ese valor con un margen de cinco veces la incertidumbre de la medición.
- Se incrementa el valor del límite del gradiente disminuyendo la cantidad de puntos procesados hasta que se puede realizar la reconstrucción a una frecuencia de 3 Hz.

Resultados del tiempo real

La tabla 4 contiene los resultados de la reconstrucción del vídeo *fr3/long office household* en tiempo real. Dado que se ha reducido la cantidad de puntos alimentados a unos 2500 por imagen clave ha aumentado la frecuencia de selección de éstas.

	fr3/long office household	*fr3/long office household* (tiempo real)
Tiempo reconstrucción (s)	126,000	69,66
Tiempo vídeo (s)	87,090	87,090
Nº imágenes clave	92	215
Nº imágenes vídeo	2585	2585
Nº puntos de alto gradiente	1018782	537502
Nº puntos 3D sin densificación	34745	14212
Nº puntos 3D con densificación	81475	25056

Tabla 4: Resultados de la reconstrucción en tiempo real.

La figura 22 enseña los resultados de la reconstrucción en tiempo real que como se puede ver ha disminuido su densidad.

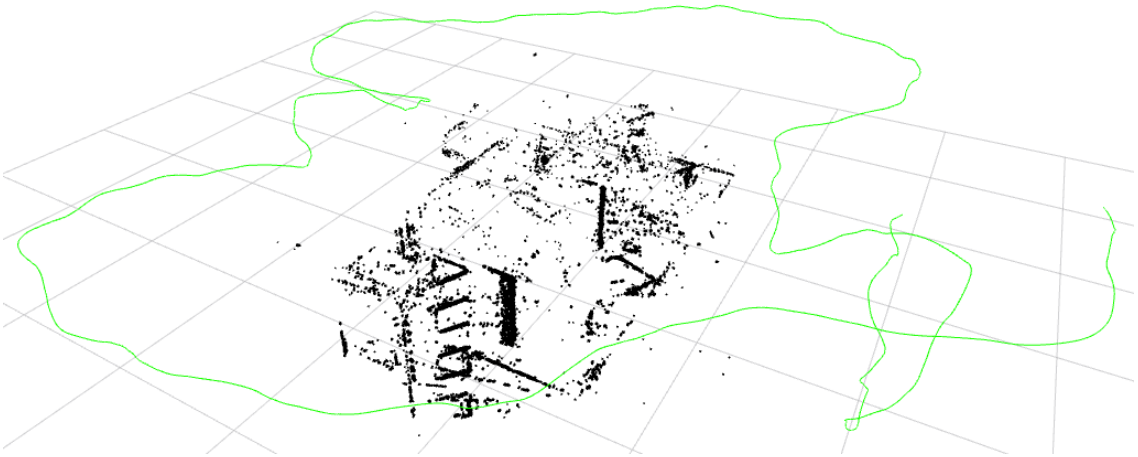


Figura 22: Reconstrucción en tiempo real de *fr3/large cabinet*.

7.2. Image Tracking

La compañía austríaca Wikitude GmbH [8] provee software de realidad aumentada para dispositivos móviles. Sus productos permiten a marcas, agencias y desarrolladores conseguir sus objetivos de AR. Entre sus herramientas se encuentran el reconocimiento de objetos e imágenes, "Instant Tracking","Image Tracking", contenido 3D aumentable y geolocalización. La implementación del "Image Tracking" en un dispositivo móvil permite al usuario conocer la posición de la cámara respecto de una imagen en la escena. Con esta herramienta se pueden añadir elementos fácilmente sobre la imagen.

El acoplamiento del algoritmo de reconstrucción con este método permite no solo conocer la posición de la cámara de forma precisa sino averiguar la escala de la escena (gracias a que se conocen las dimensiones de la imagen). En este experimento se trataba de utilizar un dispositivo móvil Samsung S7 para grabar los datos de posición e imágenes de una escena y alimentar el algoritmo de reconstrucción. Sin embargo dado que las imágenes son tratadas internamente por el SDK (Software Development Kit) de Wikitude, el modelo utilizado para la calibración de la cámara es el del campo de vista y no el de cámara pinhole, por lo que no se puede realizar el experimento apropiadamente a pesar de extraer con precisión la trayectoria de la cámara.

7.3. SLAM Visual-Inercial.

La tecnología de localización y construcción simultánea de mapas (SLAM: Simultaneous Localization and Mapping) consiste en localizar un sensor en un mapa que se construye en línea. El SLAM es una tecnología que se adapta perfectamente a las aplicaciones de realidad aumentada por dos motivos. Por una parte se diseña para funcionar en entornos desconocidos utilizando solo la información de los sensores del dispositivo (cámara e inerciales en el caso de un teléfono móvil). Además un mapa permite localizarse en todo momento sin acumular deriva lo que posibilita que los contenidos aumentados aparezcan siempre en la misma posición.

El sistema de SLAM que se propone para su implementación en aplicaciones de AR es el que combina la información obtenida de la cámara monocular del móvil con la de los sensores inerciales llamado, SLAM Visual-Inercial (VI). Las precisas mediciones a corto plazo estimadas por giróscopos y acelerómetros (IMU) son el complemento perfecto para la rica representación de la estructura proyectada en una imagen.

La tabla 5 [12] contiene una relación de ventajas e inconvenientes del VI frente al SLAM monocular. Obviamente no todas las diferencias son igual de críticas. Las contribuciones más importantes del VI son sin duda la recuperación de la escala y la estimación del movimiento entre imágenes. El problema de la calibración de parámetros extrínsecos entre la cámara y la IMU en el VI se vuelve crítico cuando ésta ha de hacerse en línea, debido especialmente al desempeño ruidoso de los sensores inerciales de los dispositivos móviles.

	Ventajas		**Inconvenientes**
Monocular	Construir modelos 3D. Localizar la cámara. Reconocer lugares visitados.	Baja demanda computacional. Barato. Calibración mínima.	*Escala métrica inobservable.* *Deriva en la escala.* 3D solo a partir de múltiples vistas. No hay mapeo ante rotaciones puras. Inicialización no trivial.
Visual-Inercial	*Estimación del movimiento entre imágenes.* *Conseguir la escala métrica.* Estimar la dirección de la gravedad. Observabilidad del pitch y el roll.		Variabilidad de los sesgos de los sensores. Compensación de la gravedad. Calibración visual-inercial. Sincronización.

Tabla 5: Ventajas e inconvenientes del SLAM monocular frente al Visual-Inercial.

Se van a utilizar dos sistemas VI disponibles en el ámbito científico: OKVIS (Open Keyframe-based Visual-Inertial SLAM [11]) y VINS (Versatile Monocular Visual-Inertial State Estimator [13]). Ambos sistemas construyen una aproximación fuertemente acoplada basada en optimización no lineal (en vez de usar filtrado), es decir, la estimación del movimiento y de los parámetros del modelo de la IMU se realiza mediante la minimización de una función de coste que incluye términos visuales e inerciales. Además presentan fundamentos similares utilizando la extracción de características y procesos de marginalización para hacer el problema tratable. VINS incorpora además un sistema de cierre de bucle (solo basado en información visual) y un proceso de autocalibración de extrínsecos.

De las diferentes pruebas realizadas a los algoritmos en los datasets del grupo [3] se comprobó que ambos se comportaban de forma similar con la diferencia de que VINS mejoraba los resultados al eliminar la deriva con el cierre de bucle. El sistema de autocalibración de VINS reportaba errores

entre 29 y 77 mm en la posición de la IMU y menos de 2º en la orientación, si bien es cierto que tanto la cámara como la IMU utilizadas en los datasets son dispositivos de muy alta calidad.

Reconstrucción de una trayectoria

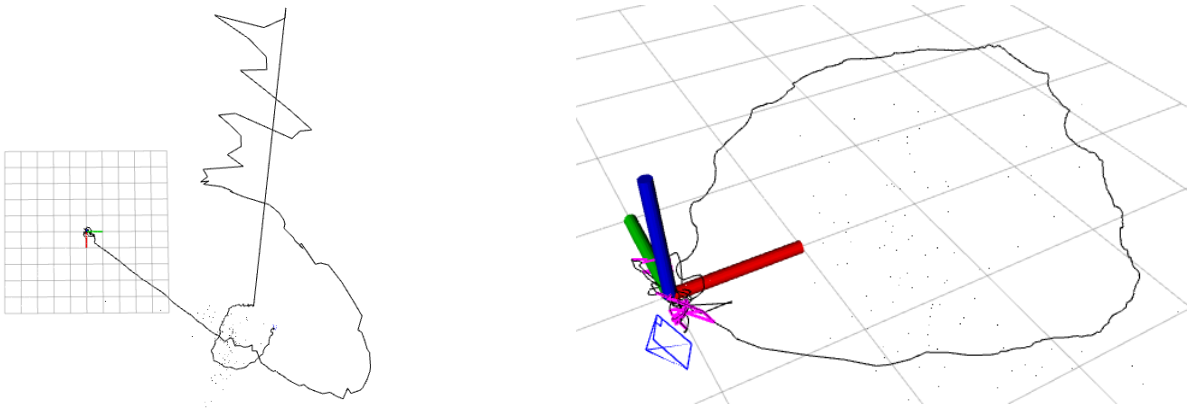
El experimento que se describe a continuación tiene por objetivo reconstruir la trayectoria con ambos sistemas VI en un ordenador con los datos recogidos por un teléfono móvil (Samsung S7). Los datos del acelerómetro y del giróscopo (100 Hz) se recogen con sus etiquetas temporales sincronizadas con las de las imágenes grabadas (30 Hz). La librería ROS permite generar un fichero "ROS bag" con los que alimentar ambos sistemas y simular el tiempo real en el ordenador.

La calibración de los parámetros intrínsecos de la cámara del móvil se realizó con la herramienta Kalibr [7] y la IMU con un experimento basado en la "varianza de Allan". Por último se realizó una estimación de los parámetros extrínsecos cámara-IMU con el mapa de la estructura del móvil.

VI	Distancia recorrida (m)	Error de posición (m)	% Error/distancia
OKVIS	12,724	0,88	6,91
VINS (sin cierre de bucle)	-	-	-
VINS (con cierre de bucle)	15,177	0,11	0,72

Tabla 6: Resultados de la reconstrucción de una trayectoria con VI.

A falta de valores reales para comparar, se realiza una trayectoria circular y se mide el error acumulado al final sobre la distancia recorrida (tabla 6). Junto con la figura 23 se observa que VINS es más preciso que OKVIS cuando implementa el cierre de bucle pero sin él es menos robusto ya que como muestra la figura 23b llega a perder la localización. Finalmente añadir que el mecanismo de autocalibración de VINS no ha convergido en ningún experimento para los datos recogidos por la IMU de un teléfono móvil.



(a) Sin cierre de bucle.

(b) Con cierre de bucle.

Figura 23: Reconstrucción de la trayectoria con VINS.

Para finalizar se ha alimentado el algoritmo de reconstrucción con la trayectoria reconstruída con los sistemas VI (figura 24).

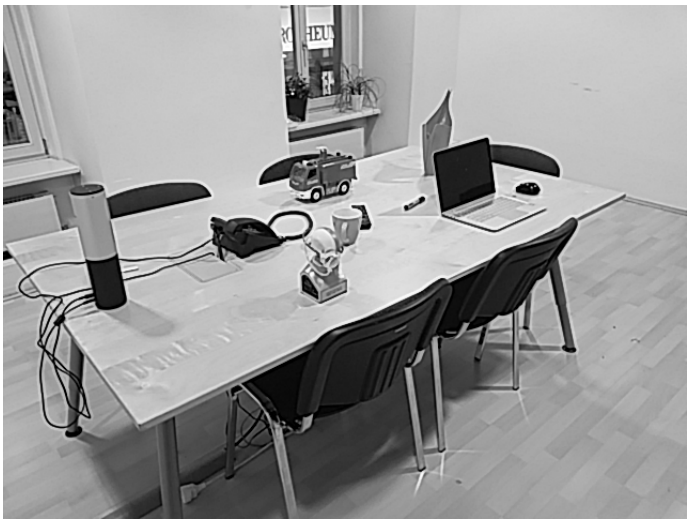


Figura 24: Escena grabada con Samsung S7.

Solo la trayectoria proporcionada por VINS con cierre de bucle permite obtener los resultados que se muestran en la figura 25.

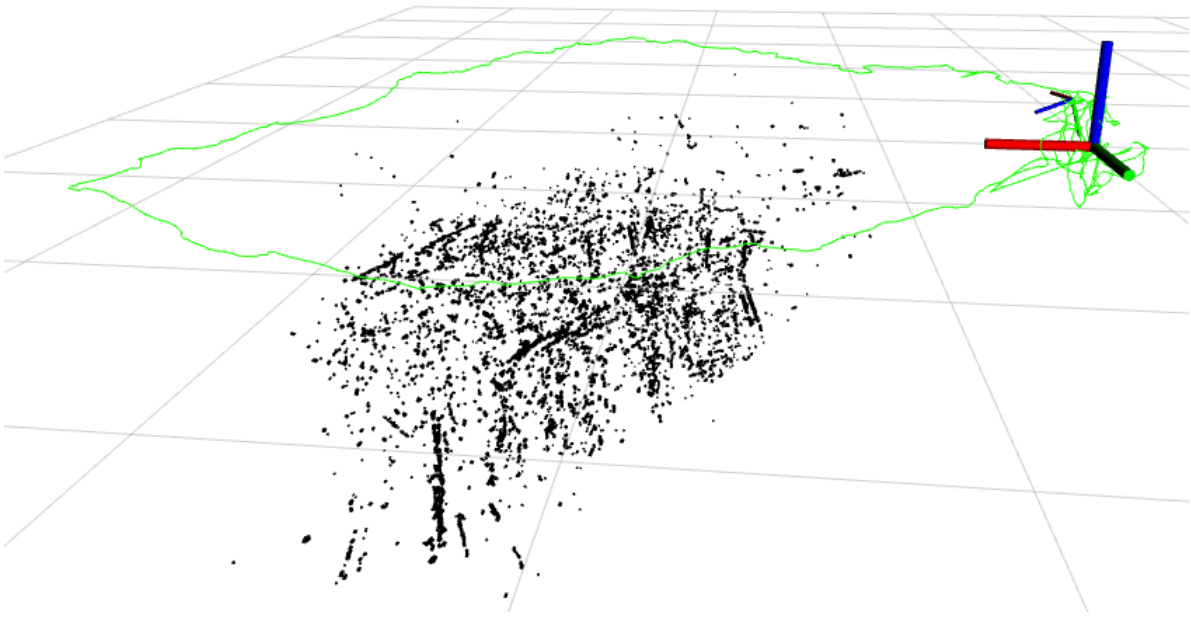


Figura 25: Reconstrucción de la escena con VI.

8. Conclusiones y trabajo futuro

A continuación se exponen las fortalezas y puntos débiles del funcionamiento del algoritmo así como las posibles mejoras y líneas de investigación.

Algoritmo de reconstrucción.

Desde el punto de vista del planteamiento teórico del algoritmo se puede decir que el proceso de emparejamiento directo con búsqueda epipolar de los píxeles supone una disminución computacional respecto de los métodos basados en características. Sin embargo, en este caso, el proceso es fuertemente dependiente de la precisión del sistema de medición de la posición y deja de ser robusto ante cambios en la intensidad o la dirección de la luz.

Los mecanismos de precisión subpíxel y optimización de la profundidad suponen mejoras en la precisión sin afectar excesivamente al coste computacional. Se han descartado el uso de la interpolación bilineal por el empeoramiento que supone de la precisión y del coeficiente de correlación hasta desarrollar una implementación optimizada.

El proceso de fusión de hipótesis basado en la incertidumbre generada por la posición de la cámara así como el filtro de mediana y la densificación redundan en una reconstrucción semidensa con bajo número de errores. Se espera que la implementación de un filtro de los puntos 3D entre las imágenes clave termine de limpiar las reconstrucciones.

La elección de los parámetros del sistema (límites de las restricciones, número de imágenes auxiliares, número de compatibilidades, rangos válidos de incertidumbres...) es todavía estrechamente dependiente de la escena, por eso debería plantearse un estudio de autoajuste de estos valores.

Implementación en tiempo real.

Todos los procesos descritos en la memoria han podido ser implementados en tiempo real a excepción del cálculo del coeficiente de correlación. El tiempo de cálculo depende intensamente del número de puntos de alto gradiente procesados, de lo que se deduce la importancia de una selección inteligente. La selección basada en la distribución exponencial, la construcción de la malla y el cálculo del coeficiente de relación de desviaciones típicas ofrecen resultados que podrían ser una solución que mejorase el rendimiento.

Otro factor determinante en la calidad de la reconstrucción es la elección de imágenes auxiliares con el paralaje y la distancia adecuadas. Un perfeccionamiento exhaustivo de este proceso redundaría en un claro progreso de la reconstrucción.

La actualización del intervalo de profundidad en la búsqueda epipolar se traduce en una avance de los resultados en términos de precisión, errores y tiempo de cálculo con tan solo una pequeña pérdida del número de puntos 3D obtenidos. Un desarrollo más en detalle de esta mejora sería relevante para una implementación en tiempo real.

Si la densidad de la reconstrucción fuera un factor determinante en el desempeño de la aplicación, debería valorarse la paralelización de los procesos. Dado que en el código la proyección de cada píxel se plantea de forma independiente hasta el filtrado la implementación de la paralelización es inmediata.

Aplicación del algoritmo.

Tanto con el sistema de "Image Tracking" como con el SLAM Visual-Inercial se ha conseguido obtener una reconstrucción en tiempo real de la trayectoria de la cámara. El sistema de "Image Tracking" es obviamente más robusto ya que depende de una imagen patrón en la escena lo que limita el número de aplicaciones posibles.

Los sistemas de SLAM Visual-Inertial han demostrado ser solución para el problema de la reconstrucción escalada de la trayectoria resolviendo, aunque no de forma definitiva, la inicialización del sistema, resistiendo movimientos bruscos o sorteando la ausencia de características en la escena. Son sin embargo muy dependientes de una buena calibración tanto de los parámetros intrínsecos de la IMU como de los extrínsecos cámara-IMU. El desempeño de OKVIS con datos procedentes de un teléfono móvil es más robusto que el de VINS que sin embargo es más preciso cuando implementa el proceso de cierre de bucle. El proceso de autocalibración que ejecuta VINS ha demostrado ser ineficaz ante la señal ruidosa de la IMU de un teléfono móvil.

Finalmente se ha demostrado que se puede acoplar el algoritmo de reconstrucción desarrollado a un sistema de SLAM VI, en este caso la versión de VINS con cierre de bucle.

Referencias

[1] Bouguet, Jean-Yves. *Camera calibration toolbox for matlab*. 2004.

[2] Gary Bradski y Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. .ºReilly Media, Inc.", 2008.

[3] Michael Burri y col. «The EuRoC micro aerial vehicle datasets». En: *The International Journal of Robotics Research* (2016).

[4] Javier Civera, Andrew J Davison y JM Martinez Montiel. «Inverse depth parametrization for monocular SLAM». En: *IEEE transactions on robotics* 24.5 (2008), págs. 932-945.

[5] Jakob Engel, Vladlen Koltun y Daniel Cremers. «Direct sparse odometry». En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).

[6] Open Source Robotics Foundation. *ROS (Robotic Operating System)*. 2017. URL: `http://www.ros.org/` (visitado 20-11-2017).

[7] Paul Furgale, Joern Rehder y Roland Siegwart. «Unified temporal and spatial calibration for multi-sensor systems». En: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE. 2013, págs. 1280-1286.

[8] Wikitude GmbH. *Wikitude documentation*. 2017. URL: `https://www.wikitude.com/documentation/` (visitado 20-11-2017).

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

[10] Departamento de Informática e Ingeniería de Sistemas. «60822 – Visión y Robótica». En: Máster en Ingeniería Industrial, Universidad de Zaragoza, 2016. Cap. Visión 3D y fotogrametría, Adquisición de Imágenes.

[11] Stefan Leutenegger y col. «Keyframe-based visual–inertial odometry using nonlinear optimization». En: *The International Journal of Robotics Research* 34.3 (2015), págs. 314-334.

[12] Raúl Mur Artal y Juan Domingo Tardós Solano. «Real-Time Accurate Visual SLAM with Place Recognition». Tesis doct. 2017.

[13] Tong Qin, Peiliang Li y Shaojie Shen. «VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator». En: *arXiv preprint arXiv:1708.03852* (2017).

[14] Hauke Strasdat. «Local accuracy and global consistency for efficient SLAM». Tesis doct. 2012.

[15] J. Sturm y col. «A Benchmark for the Evaluation of RGB-D SLAM Systems». En: *Proc. of the International Conference on Intelligent Robot Systems (IROS)*. Oct. de 2012.

A. Anexo: Revisión del cálculo

A.1. Diferenciación multivariable

Campo escalar: Función $F : \mathbb{R}^n \to \mathbb{R}$ que mapea un vector en un escalar.

Campo vectorial: Función $f : \mathbb{R}^n \to \mathbb{R}^m$ que mapea un vector en otro vector.

Gradiente de un campo escalar: Primera derivada de un campo escalar $\nabla F : \mathbb{R}^n \to \mathbb{R}^n$ que lo mapea en un campo vectorial y se define como:

$$\nabla F(x) = \left(\frac{\partial F(x)}{\partial x_1}, \quad \dots \quad , \frac{\partial F(x)}{\partial x_n} \right)^T$$

Hessiano de un campo escalar: Segunda derivada de un campo escalar $H_F : \mathbb{R}^n \to \mathbb{R}^{n \times n}$ que mapea un n-vector en una matriz $n \times n$:

$$\mathbf{H}_F(x) = \begin{pmatrix} \frac{\partial^2 F(x)}{\partial x_1^2}, & \dots & , \frac{\partial^2 F(x)}{\partial x_1 x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 F(x)}{\partial x_n x_1}, & \dots & , \frac{\partial^2 F(x)}{\partial x_n^2} \end{pmatrix}$$

Jacobiano de un campo vectorial: Primera derivada de un campo vectorial $J_f : \mathbb{R}^n \to \mathbb{R}^{m \times n}$:

$$\mathbf{J}_f(x) = \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1}, & \dots & , \frac{\partial f_1(x)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m(x)}{\partial x_1}, & \dots & , \frac{\partial f_m(x)}{\partial x_n} \end{pmatrix}$$

Hessiano de un campo vectorial: Segunda derivada de un campo vectorial $H_f : \mathbb{R}^n \to \mathbb{R}^{n \times m \times n}$ que mapea un n-vector en una matriz de tres dimensiones $n \times m \times n$.

A.2. Introducción a la optimización

Método de Gauss-Newton

Existen problemas de optimización complicados dimensionalmente en los que el cálculo del Hessiano se vuelve costoso. El método de Gauss-Newton es una variante eficiente del método de Newton que requiere un campo escalar $F : \mathbb{R}^n \to \mathbb{R}$ del siguiente tipo:

$$F(x) = a \cdot \mathbf{d}(x)^T \Lambda \mathbf{d}(x) \tag{21a}$$

dónde $a > 0$, $\mathbf{d} : \mathbb{R}^n \to \mathbb{R}^m$ es un campo vectorial doblemente diferenciable y $\Lambda \in \mathbb{R}^m$ es una matriz simétrica semidefinida positiva. Sin pérdida de generalidad se considera $a = \frac{1}{2}$ ya que el escalado de F no cambia la posición de su mímino.

Aplicando la regla de derivación del producto y considerando que Λ es una matriz simétrica la primera derivada del campo escalar F resulta:

$$\nabla F(x) = \frac{1}{2}(\mathbf{d}(x)^T \Lambda \mathbf{J}_d(x))^T + \frac{1}{2}(\mathbf{J}_d^T(x)\Lambda \mathbf{d}(x)) = \mathbf{J}_d^T(x)\Lambda \mathbf{d}(x) \tag{21b}$$

Del mismo modo se calcula la segunda derivada obteniendo:

$$H_F(x) = \mathbf{J}_d^T(x)\Lambda \mathbf{J}_d(x) + \mathbf{H}_d(x)\Lambda \mathbf{d}(x) \tag{21c}$$

siendo H_f el tensor Hessiano de $\mathbf{d}$. El método de Gauss-Newton aproxima el Hessiano de F cómo:

$$H_F(x) \approx \mathbf{J}_d^T(x)\Lambda \mathbf{J}_d(x) \tag{21d}$$

$$\mathbf{J}_d^T(x)\Lambda \mathbf{J}_d(x)\boldsymbol{\delta} = -\mathbf{J}_d^T(x)\Lambda \mathbf{d}(x) \tag{21e}$$

Una aplicación directa del método de Gauss-Newton es la resolución de problemas de mínimos cuadrados.

B. Anexo: Optimización de la profundidad

Derivadas de la transformación en el espacio proyectivo.

$$x_n = \frac{R_1 + \lambda_1 t_1}{R_3 + \lambda_1 t_3} \qquad \frac{\partial x_n}{\partial \lambda_1} = \frac{t_1(R_3 + \lambda_1 t_3) - (R_1 + \lambda_1 t_1)t_3}{(R_3 + \lambda_1 t_3)^2} \tag{22}$$

$$y_n = \frac{R_2 + \lambda_1 t_2}{R_3 + \lambda_1 t_3} \qquad \frac{\partial y_n}{\partial \lambda_1} = \frac{t_2(R_3 + \lambda_1 t_3) - (R_2 + \lambda_1 t_2)t_3}{(R_3 + \lambda_1 t_3)^2} \tag{23}$$

Derivadas del modelo de distorsión.

$$r_d^2 = x_n^2 + y_n^2 \qquad \frac{\partial r_d^2}{\partial \lambda_1} = 2x_n \frac{\partial x_n}{\partial \lambda_1} + 2y_n \frac{\partial y_n}{\partial \lambda_1} \tag{24}$$

$$c_d = 1 + k_1 r_d^2 + k_2 r_d^4 + k_5 r_d^6 \qquad \frac{\partial c_d}{\partial \lambda_1} = \frac{\partial r_d^2}{\partial \lambda_1}(k_1 + 2k_2 r_d^2 + 3k_5 r_d^4) \tag{25}$$

$$dx = \begin{pmatrix} 2k_3 x_n y_n + k_4(r_d^2 + 2x_n^2) \\ 2k_4 x_n y_n + k_3(r_d^2 + 2y_n^2) \end{pmatrix} \qquad \frac{\partial dx}{\partial \lambda_1} = \begin{pmatrix} 2k_3(y_n \frac{\partial x_n}{\partial \lambda_1} + x_n \frac{\partial y_n}{\partial \lambda_1}) + k_4(\frac{\partial r_d^2}{\partial \lambda_1} + 4x_n \frac{\partial x_n}{\partial \lambda_1}) \\ 2k_4(y_n \frac{\partial x_n}{\partial \lambda_1} + x_n \frac{\partial y_n}{\partial \lambda_1}) + k_3(\frac{\partial r_d^2}{\partial \lambda_1} + 4y_n \frac{\partial y_n}{\partial \lambda_1}) \end{pmatrix} \tag{26}$$

$$x_d = c_d x_n + dx_1 \qquad \frac{\partial x_d}{\partial \lambda_1} = c_d \frac{\partial x_n}{\partial \lambda_1} + \frac{\partial c_d}{\partial \lambda_1} x_n + \frac{\partial dx_1}{\partial \lambda_1} \tag{27}$$

$$y_d = c_d y_n + dx_2 \qquad \frac{\partial y_d}{\partial \lambda_1} = c_d \frac{\partial y_n}{\partial \lambda_1} + \frac{\partial c_d}{\partial \lambda_1} y_n + \frac{\partial dx_2}{\partial \lambda_1} \tag{28}$$

Derivadas del modelo de calibración.

$$u_2 = f_x x_d + c_x \qquad \frac{\partial u_2}{\partial \lambda_1} = fx \frac{\partial x_d}{\partial \lambda_1} \tag{29}$$

$$v_2 = f_y y_d + c_y \qquad \frac{\partial v_2}{\partial \lambda_1} = fy \frac{\partial y_d}{\partial \lambda_1} \tag{30}$$

Derivada del error.

$$e = \sqrt{(u_2 - u_p)^2 + (v_2 - v_p)^2} \qquad \frac{\partial e}{\partial \lambda_1} = \frac{(u_2 - u_p)\frac{\partial u_2}{\partial \lambda_1} + (v_2 - v_p)\frac{\partial v_2}{\partial \lambda_1}}{\sqrt{(u_2 - u_p)^2 + (v_2 - v_p)^2}} \tag{31}$$

Optimización de Gauss-Newton.

$$e^k = \begin{pmatrix} e_1 \\ \vdots \\ e_n \end{pmatrix} \qquad W = \begin{pmatrix} \frac{1}{\sigma_1^2} & 0 & \dots & 0 \\ 0 & \frac{1}{\sigma_2^2} & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \frac{1}{\sigma_n^2} \end{pmatrix} \qquad J = \begin{pmatrix} \frac{\partial e_1}{\partial \lambda_1} \\ \vdots \\ \frac{\partial e_n}{\partial \lambda_1} \end{pmatrix} \tag{32}$$

$$(J^T W J)\lambda_{k+1} = -J^T W e \tag{33}$$

C. Anexo: Coeficiente de relación del gradiente

Dataset	Operador	c	σ_c	σ_G	σ_{σ_G}	σ_I	σ_{σ_G}	num im.
Freib.1 xyz	Sobel	0,0146	0,0048	10,6287	1,7401	89,4425	5,9143	133/798
	Scharr	0,0149	0,0050	10,7421	1,7795			
Freib.1 rpy	Sobel	0,0108	0,0048	8,6775	1,9514	85,4221	5,6857	121/723
	Scharr	0,0111	0,0050	8,7614	1,9965			
Freib.2 xyz	Sobel	0,0328	0,0117	11,9007	1,4210	67,3729	6,0083	119/3369
	Scharr	0,0334	0,0118	12,0065	1,4290			
Freib.2 rpy	Sobel	0,0334	0,0151	11,4612	2,5993	64,4162	7,1392	118/3290
	Scharr	0,0343	0,0155	11,6170	2,6449			
Freib.1 desk1	Sobel	0,0159	0,0058	10,0494	2,0066	81,5264	9,6693	123/613
	Scharr	0,0162	0,0060	10,1580	2,0396			
Freib.1 desk2	Sobel	0,0128	0,0057	8,8478	2,0443	80,2598	7,7927	128/640
	Scharr	0,0131	0,0059	8,9372	2,0761			
Freib.1 room	Sobel	0,0146	0,0067	9,1120	2,0800	78,1301	11,9221	124/1362
	Scharr	0,0149	0,0069	9,1863	2,1034			
Freib.2 desk	Sobel	0,0297	0,0073	10,4623	1,5063	61,4955	7,7906	119/2965
	Scharr	0,0305	0,0075	10,5956	1,5343			
Freib.3 long office	Sobel	0,0263	0,0078	9,4486	1,7253	59,4696	9,5295	118/2585
	Scharr	0,0268	0,0079	9,5349	1,7517			
Freib.3 cabinet	Sobel	0,0074	0,0020	4,3045	0,8369	50,7574	8,9455	128/1147
	Scharr	0,0075	0,0021	4,3318	0,8433			
Freib.3 large cabinet	Sobel	0,0452	0,0224	8,5879	2,3755	41,6131	5,0150	127/1011
	Scharr	0,0458	0,0228	8,6391	2,4013			
Freib.3 nostruct. texture far	Sobel	0,0288	0,0086	6,7439	2,1741	39,4728	8,9259	117/465
	Scharr	0,0299	0,0091	6,8724	2,2311			
Freib.3 struct. notexture far	Sobel	0,0059	0,0032	3,0798	0,8241	41,1577	1,5181	117/814
	Scharr	0,0060	0,0031	3,1027	0,8172			
Freib.3 nostruct. notexture far	Sobel	0,0021	0,0007	0,8773	0,1406	19,4136	0,5475	119/474
	Scharr	0,0022	0,0007	0,8961	0,1399			

Tabla 7: Coeficiente de correlación entre las desviaciones de gradiente e intensidad.