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

GENERACIÓN DE MAPAS DE PROFUNDIDAD A PARTIR DE IMÁGENES ESTÉREO UTILIZANDO REGISTRO NO RÍGIDO

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELECTRICISTA

DANIEL SEBASTIÁN CALDERÓN SAAVEDRA

SANTIAGO DE CHILE
AGOSTO 2012

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

GENERACIÓN DE MAPAS DE PROFUNDIDAD A PARTIR DE IMÁGENES ESTÉREO UTILIZANDO REGISTRO NO RÍGIDO

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELECTRICISTA

DANIEL SEBASTIÁN CALDERÓN SAAVEDRA

PROFESOR GUÍA:
TAKESHI ASAHI KODAMA

MIEMBROS DE LA COMISIÓN:
CLAUDIO PÉREZ FLORES
JORGE SILVA SÁNCHEZ

SANTIAGO DE CHILE
AGOSTO 2012

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELECTRICISTA
POR: DANIEL CALDERÓN SAAVEDRA
FECHA: 21/08/2012
PROF. GUÍA: TAKESHI ASAHI KODAMA

GENERACIÓN DE MAPAS DE PROFUNDIDAD A PARTIR DE IMÁGENES ESTÉREO UTILIZANDO REGISTRO NO RÍGIDO

El presente trabajo trata sobre la aplicación de registro no rígido al caso de imágenes estéreo con el fin de generar un mapa de profundidad. En particular, la correspondencia entre las dos imágenes se describe como solución de una ecuación diferencial cuyos parámetros son determinados a través de un problema de optimización.

El proceso completo requiere etapas previas de calibración, rectificación y preparación de las imágenes, las que se implementan sobre la librería OpenCV. Se incluyen además los conceptos teóricos que se encuentran detrás de cada subproceso.

Una vez que las imágenes son rectificadas, se procesan fila por fila, adquiriéndose un enfoque unidimensional. El registro no rígido se efectúa mediante el cálculo de una transformación espacial difeomórfica ϕ, capaz de deformar de manera no lineal una de las imágenes para que iguale a la otra. En particular, ϕ es la solución de la ecuación diferencial ordinaria no lineal $d_t\phi\left(x,t\right) = v\left(\phi(x,t)\right)$, donde se desconoce el campo vectorial v, el cual se determina utilizando el método del gradiente para minimizar un funcional con componentes de similitud y regularización. Se adopta el esquema *Forward Euler Method*, es decir, primero se calcula v, y luego ϕ, utilizando la relación $\phi\left(x,t+dt\right) = \phi\left(x,t\right) + v\left(\phi\left(x,t\right)\right)dt$.

Tanto las imágenes como el campo vectorial son tratados de forma continua utilizando B-splines unidimensionales. La elección radica en sus características interpoladoras y de soporte compacto, dado que solo se esperan deformaciones locales. De esta forma, los parámetros a determinar en el problema son los coeficientes $a\left[i\right]$ que definen el campo $v\left(x\right) = \sum_i a\left[i\right]\beta^n\left(x-i\right)$.

Una vez obtenido el difeomorfismo, es directo el cálculo del mapa de disparidad, pues se cumple que $d = \phi - x$. Y posteriormente, al añadir los parámetros de la cámara estéreo, es posible generar el mapa de profundidad con la ecuación $Z = \frac{fT}{d}$ (en el caso de alineación binocular).

La estrategia de solución demuestra ser útil en casos con objetos de superficies suaves y sin necesidad de pronunciados gradientes de intensidad del nivel de gris, destacando sobre alternativas más tradicionales, que se enfocan básicamente en correspondencias de puntos característicos. El caso de imágenes generales, con combinaciones de zonas de gradientes suaves y pronunciados, no es resuelto completamente, restando abordar problemas de oclusiones, diferencias en los bordes de las imágenes, tiempo de cómputo, y exceso de suavidad en la función de deformación. El trabajo constituye un primer acercamiento a un nuevo enfoque continuo, por lo que se proponen variantes que puedan generar mejores resultados.

A mis padres

Agradecimientos

Quisiera agradecer primero a mis padres, Jessica Saavedra y Daniel Calderón, por haberme apoyado incondicionalmente durante todos estos años. Su compañía y fortaleza me han permitido terminar esta etapa, y desarrollar de buena forma todos los proyectos que me he propuesto. Muchas gracias por estar ahí, siempre.

Agradecimientos a todas mis amistades, cercanas y lejanas, por todos los gratos momentos vividos durante estos años. Mención honrosa a Paulina Arellano, Claudio Burgos, Sebastián Fehlandt, Sebastián Fuentealba, Karen Salvatierra, Felipe Valdés y Grace Varela (en orden alfabético para que no se sienta ni más ni menos). No puedo dejar fuera a los cabros del Instituto Nacional; a las amistades de la sección 5 2006, en especial a Alejandro Abarzúa, Jaime Arias, Víctor Bucarey y Alejandro Quezada; Al equipo Eolian 2, por toda esa energía sinérgicamente puesta en un mismo objetivo; y finalmente, a todo el escuadrón eléctrico con quienes nos juntamos una y otra vez.

Muchas gracias a las profesoras María Cecilia Rivara y Nancy Hitschfeld por darme la oportunidad de ser profesor auxiliar en computación gráfica durante 5 semestres.

Por supuesto agradecer a Takeshi Asahi, quien más que un profesor guía, fue un compañero de trabajo. A Alfredo López, por ayudarme a entender varios conceptos relacionados con esta memoria. Junto con ellos, agradezco también a Fernando Padilla y a Jaime Ortega por generar un ambiente de trabajo muy grato en el laboratorio de imágenes del CMM.

Gracias a los profesores Claudio Pérez y Jorge Silva por aceptar ser parte de mi comisión.

Esta memoria fue financiada por el proyecto FONDEF D04I1237.

May the Force be with you

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Descripción del problema

Observar una escena desde distintas tomas permite adquirir conocimiento e información espacial de la misma. El cerebro, procesa las imágenes provenientes de cada ojo y mezcla esta información para determinar que objetos se encuentras más cerca, y cuales, más lejos. En los sistemas biológicos, este mecanismo de visión es extremadamente poderoso, pudiendo trabajar en altas velocidades y con gran precisión. Esta misma idea se pretende simular, es decir, utilizar un par de imágenes estéreo para estimar la distancia a la que se encuentran los distintos objetos de la escena. Con este fin, se necesita identificar puntos correspondientes entre ambas imágenes, interpretándose esto como si cada punto se hubiera movido. Teniendo esta información, es posible estimar cuan lejos de las cámaras se encuentra cada punto del espacio visto.

(a) (b) (c)

Figura 1.1: Imágenes *Cones* y mapa de disparidad asociado.

Una imagen se representa digitalmente como una matriz de píxeles. Teniendo un par de imágenes estéreo, se quiere encontrar una función tal que dadas las coordenadas de un píxel en una imagen, se obtengan las coordenadas del píxel correspondiente en la otra imagen. Se dice que dos píxeles son correspondientes si ambos representan la proyección del mismo punto del espacio tridimensional.

Al disponer de las correspondencias de toda la imagen, se ha determinado el llamado registro de imágenes. En el caso de un par de imágenes estéreo con cámaras dispuestas como binocular, se observan solo desplazamientos horizontales, el registro que asocia a cada píxel la magnitud de cuanto se mueve hacia la izquierda o derecha, se conoce como el *mapa de disparidad*. Conociendo la disparidad asociada a un píxel, y determinadas características de las cámaras, es posible determinar la distancia entre el punto del espacio y las cámaras utilizando triangulación. La función que asocia esta distancia a cada píxel es conocida como *mapa de profundidad*.

Luego, el problema de encontrar un mapa de profundidad equivale a encontrar correspondencias para cada píxel de las imágenes, y de conocer los parámetros que describan las cámaras.

Se han desarrollado muchas estrategias para estimar buenos parámetros de cámara. El trabajo se limita a utilizar funciones de la librería OpenCV para realizar esta tarea.

El problema del cálculo de correspondencias ha sido abordado de múltiples formas, y aun no existe una solución aceptada como final. A pesar de esto, las técnicas pueden ser agrupadas en 2 grandes líneas: aquellas que trabajan sobre una cantidad determinada de puntos característicos; y otras que establecen relaciones entre las intensidades del nivel de gris asociados a cada imagen. Por otro lado, el tratamiento de las imágenes puede ser continuo o discreto. Las cámaras digitales generan una matriz de píxeles, siendo directa la aplicación de teoría de señales discreta. Para efectuar un tratamiento continuo, se requiere algún tipo de interpolación, en esta línea, es posible aplicar ecuaciones diferenciales y distintos métodos para resolverlas.

El objetivo principal de este trabajo es la generación de un mapa de profundidad, interpretando las correspondencias como la deformación continua entre el par de imágenes estéreo. En este sentido, el problema se trata en un dominio continuo y el método de solución se basa en las intensidades del nivel de gris.

1.2. Motivación

Trabajar en base a imágenes convierte, en general, al sistema en un método no invasivo, proporcionando múltiples aplicaciones, dentro de las que destacan:

- En robótica, es posible construir un mapa tridimensional de la realidad, mejorando la percepción y permitiendo formular interacciones más complejas con el entorno.
- Utilizando imágenes provenientes de un vuelo o imágenes satelitales, es posible efectuar un "levantamiento de terreno", lo que es de muchísima utilidad para los Sistemas de Información Geográficos (SIG).
- Añadir la variable "forma" o profundidad, permite mejorar todos los sistemas de seguridad basados en imágenes.
- Cuando se logra un sistema en tiempo real, es posible implementar aplicaciones interactivas controladas solo con movimiento, de manera similar al *Kinect* de Microsoft.

- Un buen mapa de profundidad en tiempo real, es de utilidad para asistir la conducción de un vehículo, analizando el movimiento de los objetos circundantes y previniendo posibles accidentes.
- Los mapas de profundidad son ampliamente utilizados en la industria de computación gráfica, permitiendo generar rápida y automáticamente modelos tridimensionales de la realidad.

Los anteriores constituyen solo una pequeña cantidad de ejemplos de posibles aplicaciones. Cada aplicación requerirá de consideraciones particulares, pero es común a todas ellas la generación de un buen mapa de profundidad.

Si bien se han propuesto múltiples métodos para estimar un mapa de profundidad, aun no se encuentra una solución óptima. La mayoría de estos métodos se basa en enfoques discretos, encontrando solución solo para un conjunto reducido de puntos característicos. En cambio, el enfoque continuo propuesto permitirá encontrar una solución utilizando toda la información presente en la imagen, y para cada píxel de la misma. En esta línea, se debieran evitar los abundantes falsos positivos y problemas en periodicidades de las imágenes que son típicos en los otros métodos. De esta forma, se abre un amplio abanico de posibilidades que puedan lograr mejores resultados.

1.3. Objetivos

El objetivo general es la generación de un mapa de profundidad a partir de la correspondencia estéreo vista como una deformación. Trabajando en esta línea, se plantean los siguientes objetivos específicos:

- Estudio de la teoría y de los conceptos relacionados con las imágenes estéreo y el proceso de calibración.
- Estudio de la teoría y de los conceptos relacionados con el registro no rígido de imágenes.
- Estudio de la teoría y de los conceptos relacionados con las B-splines.
- Formulación de modelo teórico que relacione el campo de deformaciones con la profundidad.
- Diseño, implementación y validación de software de calibración y de generación de mapas de profundidad.

1.4. Estructura

El presente trabajo se subdivide en 5 capítulos, agrupando las distintas partes que se requiere tratar.

El capítulo 2 comprende el marco teórico, donde se describe toda la teoría necesaria como requisito de la etapa de implementación. El trabajo sintetiza contenido desde distintas fuentes bibliográficas.

El desarrollo más importante de la presente memoria se encuentra en el capítulo 3 de implementación. Aquí se describe el sistema implementado; y en particular, se detalla teóricamente el módulo de registro no rígido.

En el capítulo 4 se presentan resultados y análisis particulares de la implementación realizada. Se efectúan pruebas para las etapas de calibración, registro no rígido, y generación de mapas de profundidad. En particular se analizan los distintos parámetros que controlan el proceso de registro.

Finalmente, las conclusiones se presentan en el capítulo 5, donde se analiza el trabajo desde un punto de vista más general, contrastando objetivos y resultados, y a la vez, proponiendo posibles trabajos futuros.

Se añade un anexo con una síntesis de la terminología utilizada en el presente informe, su objetivo es servir de consulta rápida con tal de agilizar y facilitar la comprensión del presente trabajo. Un segundo anexo detalla los parámetros utilizados en cada una de las pruebas ejecutadas en el capítulo 4.

Capítulo 2

Marco Teórico

El presente trabajo reúne varios tópicos, los que pueden encapsularse en 3 grandes temas: Geometría de cámaras, Deformaciones y B-splines. Por comodidad, la geometría de cámaras es dividida en dos, separando la involucrada con una única cámara, y la asociada al caso estéreo. Por supuesto, cada tema puede ser analizado en extenso, y existen libros completos para abordar cada uno de ellos. Por esta razón, el contenido se sintetiza, restringiéndose solo a lo que compete al presente trabajo. Se procede de la siguiente manera: primeramente se describe el modelo de una cámara simple; luego los modelos asociados a una cámara estéreo; dándose paso a las B-splines; y finalmente, acotando el concepto de deformación y dando un procedimiento para calcularlas.

2.1. Cámara simple

2.1.1. Modelo de cámara

2.1.1.1. Parámetros intrínsecos

Para adquirir una imagen del espacio, es necesario que los haces de luz reflejados en los objetos pasen a través de un pequeño orificio, de forma que cada punto del espacio proyecte un único haz de luz hacia el otro lado, proyectándose finalmente en el llamado plano de imagen. La figura 2.1 ilustra este proceso. La óptica involucrada conlleva a tratar la imagen invertida, sin embargo, en la práctica, es más simple trabajar con un sistema matemáticamente equivalente, situando el plano de imagen delante del centro de proyección (ver figura 2.2). Esta nueva configuración respeta las mismas propiedades geométricas originales.

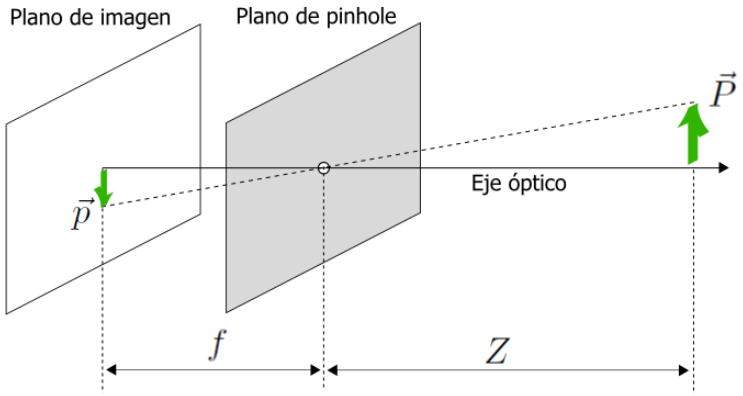

Figura 2.1: Modelo Pinhole.

Basándose en la figura 2.2, se definen algunos conceptos: el *centro de proyección* corresponde al origen del sistema de referencia asociado a la cámara; el *plano de imagen* es el plano donde se proyecta lo observado; y finalmente, el *eje óptico* o *rayo principal*, es el rayo que nace en el centro de proyección e intercepta perpendicularmente en el plano de imagen. Conviene destacar que el eje óptico atraviesa la imagen (en unidades de píxeles) justo en el centro; y que además, la imagen pertenece al plano de imagen, pero trabaja en coordenadas de píxeles y no métricas.

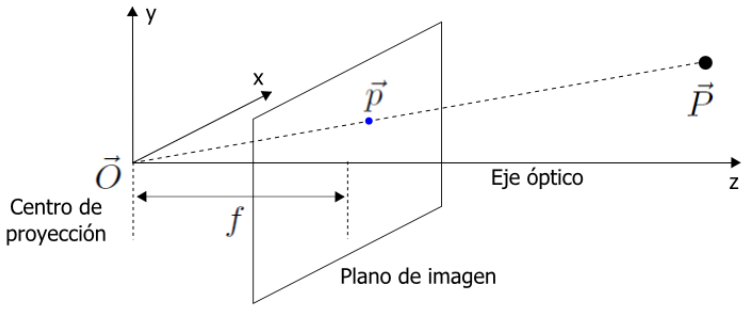

Figura 2.2: Sistema de referencia.

Se tienen 3 sistemas de referencia, con las siguientes características:

- Cámara: Con origen en el centro de proyección y unidades métricas. El eje z atraviesa el plano de imagen en la posición del centro óptico. Ejes x e y van en las direcciones horizontal y vertical respectivamente. Este sistema trabaja sobre el plano $z = f$, por lo que se considera bidimensional. Se utiliza $\vec{p}$ para representar un punto en este sistema.
- Imagen: Corresponde a una transformación del sistema asociado a la cámara de forma tal que permita trabajar en unidades de píxeles. Su origen se localiza en la esquina superior izquierda de la imagen, indexándose con i hacia la derecha y con j hacia abajo. Se utiliza $\vec{q}$ para representar un punto en este sistema.
- Mundo: Este es el sistema de referencia global utilizado, a él pertenecen los puntos del espacio real, por lo tanto, se trabaja en unidades métricas. Usualmente, su origen se

hace coincidir con el centro de proyección de la cámara. Se utilizan las variables X, Y, Z para describir este espacio. Se utiliza $\vec{P}$ para representar un punto en este sistema.

Con el modelo de la figura 2.2 es posible formular las ecuaciones de proyección 2.1 y 2.2. Donde (i, j) corresponde al punto proyectado en la imagen en unidades de píxeles; (X, Y, Z) es el punto del espacio real en unidades métricas; f_x y f_y son las distancias focales horizontal y vertical (en unidades de píxeles); y finalmente, (c_x, c_y) corresponde al centro de proyección en la imagen (también en unidades de píxeles).

$$i = f_x \left(\frac{X}{Z} \right) + c_x \tag{2.1}$$

$$j = f_y \left(\frac{Y}{Z} \right) + c_y \tag{2.2}$$

Utilizando coordenadas homogéneas, es posible plantear la expresión matricial 2.3, donde la matriz $\mathbf{M}$ se conoce como *matriz de cámara* o *matriz intrínseca de cámara*, pues encapsula características de la misma, como las distancias focales y la posición del eje óptico en las imágenes. Las ecuaciones 2.4, 2.5 y 2.6 definen $\vec{q_h}$, $\mathbf{M}$ y $\vec{P_h}$. Usualmente se omite la transformación de proyección $\left[\begin{array}{cc} \mathbf{I} & 0 \end{array} \right]$, obteniéndose la ecuación 2.7.

$$\vec{q_h} = \mathbf{M} \left[\begin{array}{cc} \mathbf{I}_{3\times3} & 0_{3\times1} \end{array} \right] \vec{P_h} \tag{2.3}$$

$$\vec{q_h} = \begin{bmatrix} i_h \\ j_h \\ w \end{bmatrix} \tag{2.4}$$

$$\mathbf{M} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

$$\vec{P_h} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad \vec{P} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.6}$$

$$\vec{q_h} = \mathbf{M}\vec{P} \tag{2.7}$$

El modelo descrito es llamado *Modelo Pinhole* de cámara. Se asume que un único haz de luz logra atravesar dicha barrera. Sin embargo, si el agujero es muy pequeño, no se alcanza a reunir suficiente luz en una rápida exposición, y además pueden llegar a surgir fenómenos de difracción; y si el agujero es muy grande, la imagen se vuelve borrosa, dado que un único punto del espacio se proyecta múltiples veces debido a que muchos haces de luz que atraviesan la barrera. Por lo tanto, un *pinhole* no es suficiente para capturar una buena

imagen. El problema es resuelto utilizando un lente, el cual logra concentrar una cantidad apropiada de luz. La contraparte del lente, es que inevitablemente se introduce distorsión en la imagen.

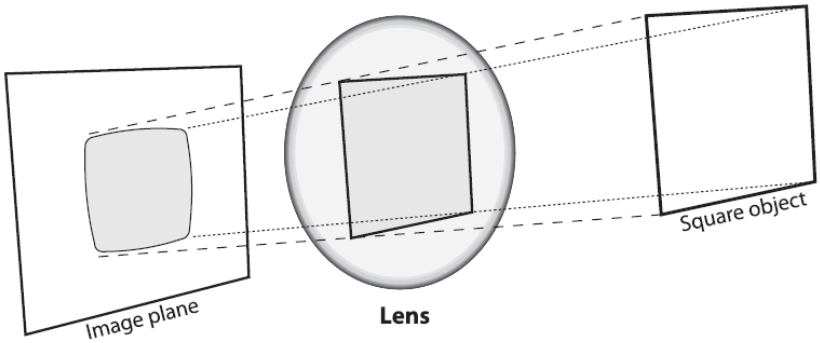

Figura 2.3: Distorsión radial. [6]

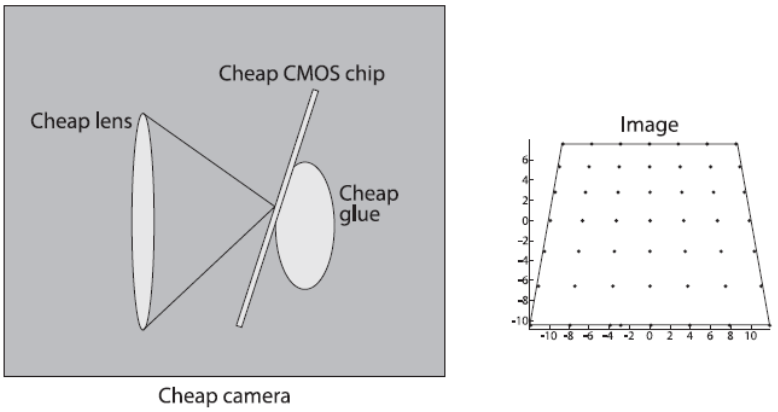

Figura 2.4: Distorsión tangencial. [6]

Dada la geometría del lente de las cámaras, esta distorsión puede ser modelada radial y tangencialmente. La distorsión radial se aprecia en la figura 2.3, se refiere a que los objetos más lejanos al eje óptico tenderán a redondearse, su efecto es modelado por las ecuaciones 2.9 y 2.10, donde r es la distancia del píxel al eje óptico (ecuación 2.8). Por otro lado, la distorsión tangencial ocurre cuando el lente no queda correctamente alineado con el eje óptico, como se muestra en la figura 2.4, su efecto puede ser modelado por las ecuaciones 2.11 y 2.12. Donde se utilizan los subíndices ur y ut para indicar los valores de i y j una vez corregida las distorsiones radial y tangencial.

$$r = \sqrt{(i - c_x)^2 + (j - c_y)^2} \tag{2.8}$$

$$i_{ur} = i\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{2.9}$$

$$j_{ur} = j\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{2.10}$$

$$i_{ut} = i + \left(2p_1 j + p_2(r^2 + 2i^2)\right) \tag{2.11}$$

$$j_{ut} = j + \left(p_1(r^2 + 2j^2) + 2p_2 i\right) \tag{2.12}$$

De esta forma, se tiene un total de 5 coeficientes que modelan la distorsión. En caso de conocer estos parámetros, es posible corregir la distorsión presente en las imágenes, este proceso es llamado *undistortion* en la literatura [6]. Efectuado este proceso, se pueden efectuar medidas de mayor precisión sobre la imagen resultante. La figura 2.5 muestra una imagen cuya distorsión se ha corregido.

Figura 2.5: Ejemplo de imagen corregida.[6]

Todos los parámetros de distorsión pueden ser agrupados en un único vector $\vec{D}$, el cual es llamado *vector de distorsión*.

$$\vec{D} = \begin{bmatrix} k_1 \\ k_2 \\ p_1 \\ p_2 \\ k_3 \end{bmatrix} \tag{2.13}$$

Dado que las características descritas por la matriz de cámara $\mathbf{M}$ y el vector de distorsión $\vec{D}$, son propias de la cámara, estos parámetros son conocidos como los *parámetros intrínsecos*.

2.1.1.2. Parámetros extrínsecos

Según lo descrito, cada cámara posee su propio sistema de referencia, sin embargo, se puede requerir expresar las magnitudes en función de otro sistema, como por ejemplo, para poder relacionar información proveniente de distintas cámaras. En este sentido, es necesario conocer la pose (posición + orientación) de cada cámara.

Si una cámara se encuentra en la posición $\vec{T}$ y orientación $\mathbf{R}$ respecto de un sistema de coordenadas global, las ecuaciones 2.14 y 2.15 permiten intercambiar la descripción del punto $\vec{P}'$ visto por el sistema de la cámara, y el punto $\vec{P}$ visto por el sistema global. La figura 2.6 ilustra el cambio de coordenadas.

$$\vec{P'} = \mathbf{R}\vec{P} + \vec{T} \tag{2.14}$$

$$\vec{P} = \mathbf{R}^T \left(\vec{P'} - \vec{T} \right) \tag{2.15}$$

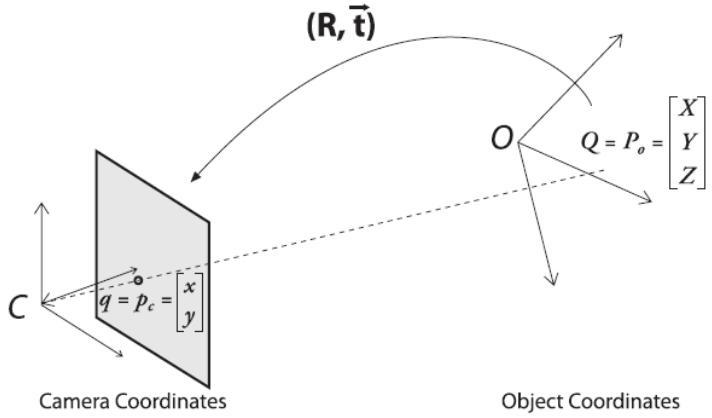

Figura 2.6: Relación entre dos sistemas de referencia. [6]

Los parámetros que definen tanto la matriz de rotación $\mathbf{R}$ como el vector de traslación $\vec{T}$ son conocidos como los *parámetros extrínsecos*, pues no dependen de la cámara misma, sino solo de su disposición espacial.

Conociendo $\mathbf{R}$ y $\vec{T}$, es posible generalizar la expresión 2.3 para considerar una proyección genérica, resultando la ecuación 2.16. En este caso, se define la matriz $\mathbf{K}$ de 3×4 conocida como *matriz de proyección de cámara*. La descomposición formal de $\mathbf{K}$ se describe en la ecuación 2.17.

$$\vec{q_h} = \mathbf{K}\vec{P_h} \tag{2.16}$$

$$\mathbf{K} = \mathbf{M}[\ \mathbf{R} \quad \vec{T}\ ] \tag{2.17}$$

2.1.2. Calibración

El proceso de calibración se refiere a la estimación de los parámetros de los modelos ya explicitados (matriz de cámara y vector de distorsión). Existe una gran variedad de procedimientos para efectuar esta tarea. En la práctica, no es posible acceder a las piezas internas de una cámara, y en general, el fabricante tampoco proporciona estos parámetros. Por estas razones, los métodos más usados para lograr una buena calibración se basan en analizar imágenes de un patrón conocido.

Supongamos como patrón un cuadrado negro en fondo blanco, pues sus vértices pueden ser fácilmente detectados. La imagen captura determinada pose del patrón, de esta forma,

es posible calcular la transformación existente entre un cuadrado descrito y el cuadrilátero realmente visto. Esta transformación se define por los parámetros intrínsecos y extrínsecos. Omitiendo los parámetros de distorsión, se tiene un total de 10 parámetros para cada vista del patrón: los 3 ángulos que describen la orientación de la cámara; los 3 valores que definen su posición; y los 4 parámetros de la matriz $\mathbf{M}$.

Cada vista del cuadrado impone un total de 8 restricciones, esto es, una por cada componente x e y de cada vértice. El sistema tiene 10 incógnitas, por lo que no posee solución para una sola vista. Si se consideran 2 vistas, los 6 parámetros asociados a la pose del patrón cambiarán, pero los 4 parámetros de la matriz de cámara seguirán siendo los mismos. De esta forma, para 2 vistas, se tiene un total de 16 incógnitas $(6 + 6 + 4)$ y 16 restricciones (8 por cada imagen), por lo tanto el sistema sí tiene solución.

Es claro que las imágenes contienen ruido y distorsiones que conducen a obtener distintas soluciones según las vistas consideradas. El problema puede ser resuelto al considerar varias tomas, formulando un sistema sobredimensionado. En esta situación, se puede buscar la solución que minimice el error cuadrático medio del sistema de ecuaciones. En este sentido, es conveniente la adquisición de una gran cantidad de vistas del patrón y que estas abarquen, idealmente, toda el área vista por la cámara (ver figura 2.7).

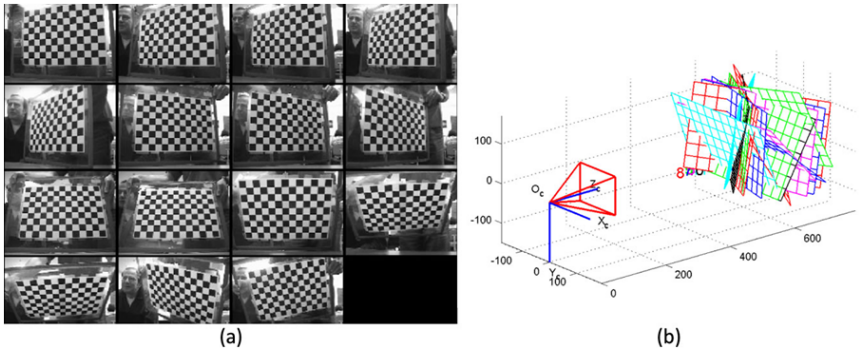

Figura 2.7: Distintas vistas de un tablero de ajedrez.

Conviene mencionar, que es posible utilizar cualquier figura como patrón de calibración, incluyendo objetos tridimensionales de mayor complejidad. En la práctica, se prefiere utilizar patrones planos, como tableros de ajedrez o grillas de círculos[1] (ver figura 2.8) pues su construcción física y transporte son más simples.

En términos más formales, se busca la transformación que realiza un mapeo proyectivo de un plano a otro. En visión computacional, este concepto es llamado *homógrafo*. En [11, páginas 170, 92, 73 y 556] se describe un algoritmo para estimar la matriz de cámara, buscando el homógrafo entre el plano asociado al patrón conocido, y el plano de imagen. En otras palabras, la calibración permite obtener $\mathbf{R}$ y $\vec{T}$ para cada una de las vistas del patrón, y la matriz de cámara $\mathbf{M}$. Para estimar también el vector de distorsión $\vec{D}$, el procedimiento es similar, complejizando levemente el modelo.

[1]En efecto, la librería OpenCV 2.3.0 implementa funciones de calibración para un tablero de ajedrez y una grilla de círculos

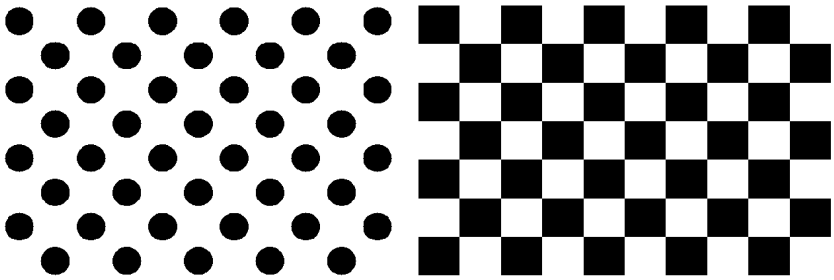

Figura 2.8: Patrones de calibración planos.

2.2. Cámara estéreo

2.2.1. Triangulación

Supongamos que se dispone de 2 cámaras alineadas perfectamente en paralelo y con el mismo plano de imagen, tal como se muestra en la figura 2.9. Cada cámara posee sus propios parámetros y definiciones, por lo que para evitar confusiones, en lo sucesivo, se utilizarán los subíndices 0 y 1 para referirse a las cámaras izquierda y derecha, respectivamente.

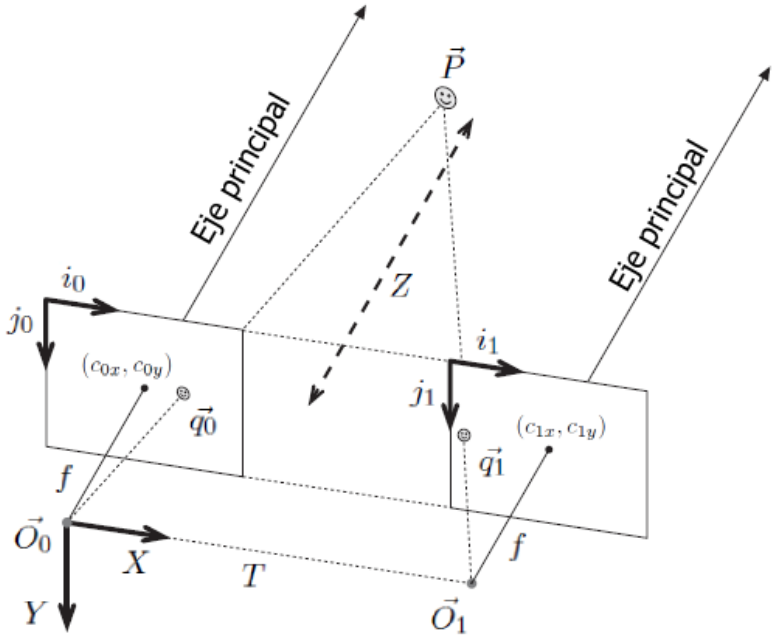

Figura 2.9: Triangulación con cámaras alineadas en paralelo.

Ambas cámaras observan un mismo punto $\vec{P}$ del espacio, el cual se proyecta como $\vec{q_0} = (i_0, j_0)$ y $\vec{q_1} = (i_1, j_1)$ en cada imagen. Conociendo las distancias focales de las cámaras, por simple triangulación (ecuaciones 2.18), es posible calcular la posición en el espacio del punto $\vec{P}$ (ecuaciones 2.19, 2.20 y 2.21).

$$\frac{f}{Z} = \frac{(i_0 - c_{x0})}{X} = \frac{(i_1 - c_{x1})}{X - T} = \frac{c_{y0} - j_0}{Y} = \frac{c_{y1} - j_1}{Y} \tag{2.18}$$

$$X = \frac{(i_0 - c_{0x})\,T}{d - (c_{0x} - c_{1x})} \tag{2.19}$$

$$Y = \frac{-(j_0 - c_{0y})\,T}{d - (c_{0x} - c_{1x})} \tag{2.20}$$

$$Z = \frac{fT}{d - (c_{0x} - c_{1x})} \tag{2.21}$$

Donde $d = i_0 - i_1$ y se conoce como *disparidad*. Esta configuración particular de las cámaras, implica que un mismo punto se proyectará en la misma altura en ambas imágenes, es decir, $j_0 - c_{0j} = j_1 - c_{1j}$. En otras palabras, para determinar la posición en el espacio tridimensional de un punto $\vec{P}$, se requiere conocer los parámetros de la cámara, y a la vez, identificar el mismo punto proyectado en ambas imágenes.

Si ambas cámaras tienen el centro óptico en la misma posición, las ecuaciones se simplifican a 2.22, 2.23 y 2.24. Destaca la proporcionalidad inversa entre Z y d. De esta forma, los objetos cercanos a las cámaras producen grandes disparidades, y pueden ser bien diferenciados; mientras que los lejanos, tienen asociada una pequeña disparidad e inevitablemente se pueden interpretar como un plano (ver figura 2.10).

$$X = \frac{(i_0 - c_{0i})\,T}{d} \tag{2.22}$$

$$Y = \frac{-(j_0 - c_{0j})\,T}{d} \tag{2.23}$$

$$Z = \frac{fT}{d} \tag{2.24}$$

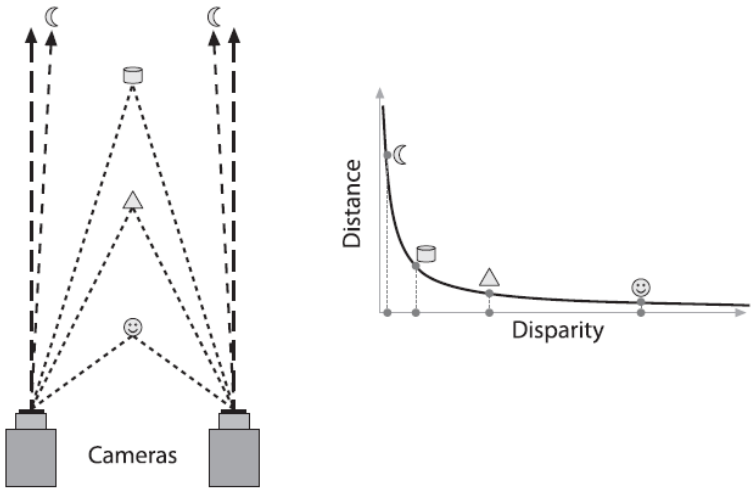

Figura 2.10: Distancia y disparidad son inversamente proporcionales. [6]

Es posible expresar las ecuaciones 2.19, 2.20 y 2.21 en un sistema matricial (ecuación 2.26). Para esto se define la *matriz de re-proyección* $\mathbf{Q}$ (ecuación 2.25). Notar que, en caso de que las imágenes sean del mismo tamaño, el término de la esquina inferior derecha se anula.

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & -c_{0x} \\ 0 & 1 & 0 & -c_{0y} \\ 0 & 0 & 0 & f \\ 0 & 0 & -\frac{1}{T_x} & -\frac{(c_{0x}-c_{1x})}{T_x} \end{bmatrix} \tag{2.25}$$

$$\mathbf{Q} \begin{bmatrix} i_0 \\ j_0 \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ W_0 \end{bmatrix} \tag{2.26}$$

La ecuación 2.26 permite determinar el punto del espacio en coordenadas homogéneas. En coordenadas cartesianas, el punto queda dado por $\left(\frac{X_0}{W_0}, \frac{Y_0}{W_0}, \frac{Z_0}{W_0}\right)$. Conviene mencionar que el punto tridimensional queda referido al sistema de coordenadas propio de la cámara izquierda.

2.2.2. Geometría epipolar

2.2.2.1. Conceptos básicos

Tener dos cámaras perfectamente alineadas es prácticamente imposible. Es natural que haya un pequeño error en la alineación. De esta forma, el procedimiento explicado en la sección 2.2.1 no es válido.

El caso general, donde las cámaras tienen poses arbitrarias pero con cierta región en común, tiene características geométricas interesantes, este campo se conoce como *Geometría*

Epipolar. En esencia, se combinan dos modelos pinhole. La figura 2.11 presenta cámaras trasladadas en un vector $\vec{T}$, y rotadas acorde a la matriz de rotación $\mathbf{R}$.

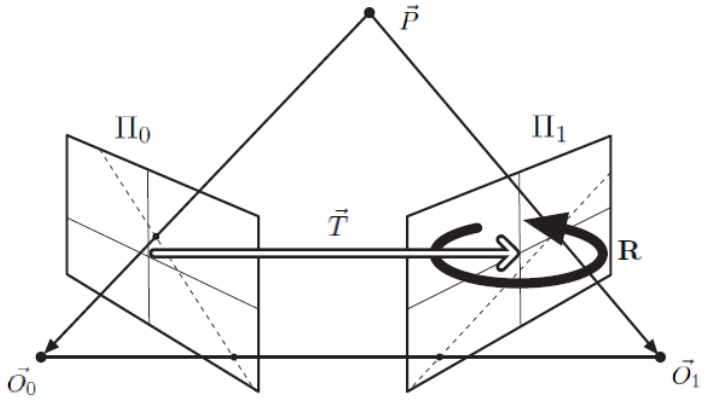

Figura 2.11: Cámaras con poses no paralelas. [6]

Cada cámara tiene un centro de proyección separado $\vec{O_0}$ y $\vec{O_1}$, y sus correspondientes planos de proyección Π_0 y Π_1. El punto $\vec{P}$ en el mundo físico tiene proyección en cada uno de los planos como $\vec{p_0}$ y $\vec{p_1}$. Los nuevos puntos de interés son los *epipolos*. Un epipolo es definido como la proyección en el plano de imagen del centro de proyección de la otra cámara. De esta forma, los epipolos son llamados $\vec{e_0}$ y $\vec{e_1}$. El plano formado por el punto $\vec{P}$ y los epipolos $\vec{e_0}$ y $\vec{e_1}$ (o consecuentemente $\vec{O_0}$ y $\vec{O_1}$) es llamado *plano epipolar*, y las líneas $\overleftrightarrow{p_0 e_0}$ y $\overleftrightarrow{p_1 e_1}$ son llamadas *líneas epipolares*. Estos conceptos son ilustrados en la figura 2.12.

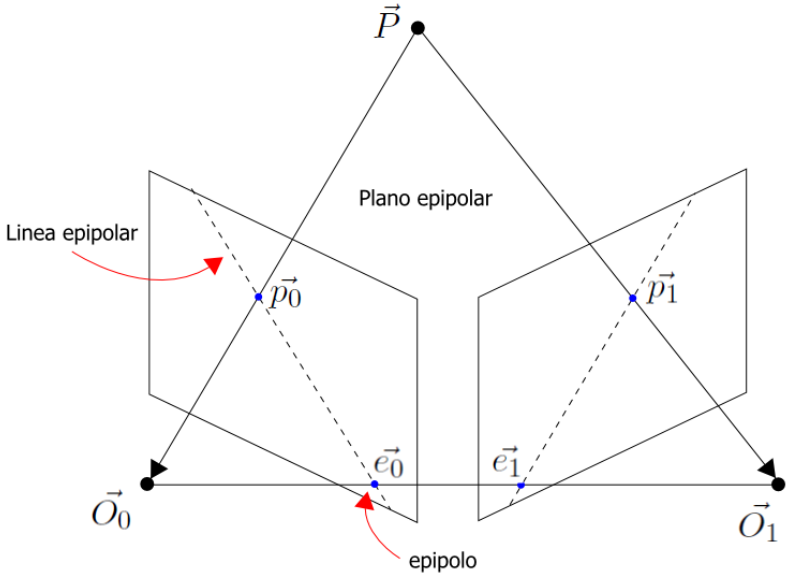

Figura 2.12: Conceptos de geometría epipolar.

Al visualizar la proyección del punto $\vec{P}$ en la cámara izquierda como $\vec{p_0}$, se sabe que $\vec{P}$ se encuentra en alguna posición del rayo que va desde $\vec{O_0}$ a $\vec{p_0}$. La línea epipolar en la imagen derecha corresponde a la proyección del rayo descrito, y se sabe por tanto que $\vec{p_1}$ se encontrará en dicha línea.

Con estos conceptos, es posible enunciar lo siguiente:

- Cada punto 3D visto en las cámaras, está contenido en un plano epipolar que intercepta cada imagen en la línea epipolar.
- Dado un punto en una imagen, su punto correspondiente en la otra imagen debe pertenecer a la línea epipolar asociada. Esto es conocido como *restricción epipolar*.
- La restricción epipolar permite que la búsqueda del punto correspondiente se transforme en una búsqueda unidimensional (línea epipolar), disminuyendo considerablemente la carga de cómputo, y además descarta puntos que produzcan falsos positivos.
- Existe la misma relación de orden entre los puntos vistos por ambas cámaras.

2.2.2.2. Matrices Esencial y Fundamental

La geometría epipolar requiere de un último par de ingredientes, las matrices esencial y fundamental, estas permiten relacionar puntos en los planos de imagen de una cámara a la otra.

La *Matriz Esencial* $\mathbf{E}$ contiene información sobre la traslación y rotación que relaciona a las cámaras en el espacio físico. Mientras que la *Matriz Fundamental* $\mathbf{F}$, contiene la misma información que $\mathbf{E}$, pero agrega los parámetros intrínsecos, permitiendo relacionar ambas cámaras en coordenadas de píxeles.

Estas matrices cumplen las siguientes relaciones:

$$\vec{p_1}^T \mathbf{E} \vec{p_0} = 0 \tag{2.27}$$

$$\vec{q_1}^T \mathbf{F} \vec{q_0} = 0 \tag{2.28}$$

Como ya se ha mencionado, $\vec{p_0}$ y $\vec{p_1}$ son los puntos proyectados en el plano de imagen en coordenadas espaciales, mientras que $\vec{q_0}$ y $\vec{q_1}$ pertenecen al mismo plano, pero utilizan coordenadas de píxeles.

Sobre la matriz esencial, destaca:

- Es de 3×3, pero tiene rango 2, por lo que su determinante es cero.
- Contiene 5 parámetros: 3 para rotación y 2 para traslación. No considera escalamiento.
- Los 2 valores propios son iguales.
- Relaciona los puntos en coordenadas métricas de cámara, no en coordenadas de píxeles.

Por otro lado, la matriz fundamental:

- Al igual que **E**, es de 3×3 y tiene rango 2.
- Trabaja en coordenadas de píxeles, no en coordenadas de cámara.
- Contiene 7 parámetros: 2 para cada epipolo y 3 para el homógrafo que relaciona los 2 planos de imagen.

2.2.2.3. Cálculo de la matriz Esencial

El plano de normal $\vec{n}$ y que pasa por la posición $\vec{a}$, está descrito por todos los $\vec{x}$ que satisfacen la ecuación 2.29.

$$(\vec{x} - \vec{a}) \cdot \vec{n} = 0 \tag{2.29}$$

Situándose en el sistema de referencia de la cámara izquierda, el sistema de la cámara derecha tiene orientación **R** y posición $\vec{T}$ (ver figura 2.11). Se tiene que los vectores $\vec{T}$ y $\vec{P_0}$ pertenecen al plano epipolar, siendo $\vec{P_0}$ el punto del espacio en el sistema $\vec{O_0}$. Luego, es posible describir el plano epipolar utilizando $\vec{a} = \vec{T}$, $\vec{x} = \vec{P_0}$ y $\vec{n} = \vec{T} \times \vec{P_0}$, por lo que al reemplazar en la ecuación 2.29 se obtiene:

$$\left(\vec{P_0} - \vec{T}\right)^T \left(\vec{T} \times \vec{P_0}\right) = 0 \tag{2.30}$$

Por otro lado, el mismo punto $\vec{P}$ en el sistema de referencia de la cámara derecha es $\vec{P_1}$, se cumple la relación 2.30 (ver sección 2.1.1.2).

$$\vec{P_0} = \mathbf{R}\vec{P_1} + \vec{T} \tag{2.31}$$

Luego $\left(\vec{P_0} - \vec{T}\right) = \mathbf{R}\vec{P_1}$ y reemplazando en 2.30 se obtiene:

$$\left(\left(\vec{P_1}\right)^T \mathbf{R}^T\right) \left(\vec{T} \times \vec{P_0}\right) = 0 \tag{2.32}$$

Considerando que es posible escribir el producto cruz como un producto matriz-vector (ecuación 2.33), se reescribe la ecuación 2.32 como la ecuación 2.34.

$$\vec{T} \times \vec{P_0} = \mathbf{S}\vec{P_0} \quad \Rightarrow \quad \mathbf{S} = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \tag{2.33}$$

$$\left(\vec{P_1}\right)^T \mathbf{R}^T \mathbf{S} \vec{P_0} = 0 \tag{2.34}$$

Se define entonces:

$$\mathbf{E} = \mathbf{R}^T\mathbf{S} \tag{2.35}$$

Y por lo tanto:

$$\left(\vec{P_1}\right)^T \mathbf{E}\vec{P_0} = 0 \tag{2.36}$$

Los puntos $\vec{P_1}$ y $\vec{P_0}$ pertenecen al espacio 3D, según coordenadas relativas a cada una de las cámaras. Es posible llevar esta expresión a coordenadas métricas ubicadas en los planos de imagen a través de las expresiones 2.37 y 2.38. Las que al reemplazar en 2.36 generan la expresión para la matriz esencial $\vec{p_1}^T\mathbf{E}\vec{p_0} = 0$.

$$\vec{p_1} = \frac{f_1}{Z_1}\vec{P_1} \tag{2.37}$$

$$\vec{p_0} = \frac{f_0}{Z_0}\vec{P_0} \tag{2.38}$$

2.2.2.4. Cálculo la matriz Fundamental

La matriz de cámara $\mathbf{M}$ relaciona los puntos proyectados desde coordenadas espaciales a coordenadas de píxeles a través de la relación $q_h = MP$ (ecuación 2.16). De esta forma, efectuando un simple reemplazo en 2.27, queda:

$$\overrightarrow{q_1}\left(\mathbf{M}_1^{-1}\right)^T \mathbf{E}\mathbf{M}_0^{-1}\vec{q_0} = 0 \tag{2.39}$$

Donde $\mathbf{M}_1$ y $\mathbf{M}_0$ son las matrices de cámara derecha e izquierda respectivamente. Luego, basta con definir $\mathbf{F}$ según la ecuación 2.40. Obteniéndose la expresión para la matriz fundamental $\vec{q_1}^T\mathbf{F}\vec{q_0} = 0$.

$$\mathbf{F} = \left(\mathbf{M}_1^{-1}\right)^T \mathbf{E}\mathbf{M}_0^{-1} \tag{2.40}$$

El procedimiento aquí descrito permite el cálculo de $\mathbf{E}$ y $\mathbf{F}$ una vez que se disponga de $\mathbf{R}$ y $\mathbf{T}$, es decir, requiere de un sistema estéreo correctamente calibrado. Existen mecanismos que permiten estimar $\mathbf{F}$ directamente a partir de una serie de correspondencias entre las imágenes [11], pues estos puntos validan la ecuación 2.28. Sin embargo, al utilizar correspondencias erróneas, se generan una estimación incorrecta de $\mathbf{F}$, por lo que, de ser posible, se debe optar por un sistema calibrado.

2.2.2.5. Alineación binocular

Los conceptos de geometría epipolar descritos aplican para cualquier par de imágenes estéreo mientras mantengan determinada región en común. Conviene explicitar expresiones

para $\mathbf{E}$ y $\mathbf{F}$ en el caso particular de imágenes tomadas desde cámaras idénticas en disposición binocular, es decir, con ejes ópticos paralelos y con un mismo plano de imagen. En este caso se tiene que no hay rotación y que hay traslación en un solo eje, digamos x. Las matrices $\mathbf{R}$ y $\mathbf{S}$ quedan descritas por las expresiones 2.41 y 2.42 respectivamente.

$$\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.41}$$

$$\mathbf{S} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T_x \\ 0 & T_x & 0 \end{bmatrix} \tag{2.42}$$

Luego, utilizando las ecuaciones 2.35 y 2.40, se tiene que $\mathbf{E} = \mathbf{S}$ y $\mathbf{F} = (\mathbf{M}^{-1})^T \mathbf{E}\mathbf{M}^{-1}$, obteniéndose las expresiones 2.43 y 2.44.

$$\mathbf{E} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T_x \\ 0 & T_x & 0 \end{bmatrix} \tag{2.43}$$

$$\mathbf{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\frac{T_x}{f_y} \\ 0 & \frac{T_x}{f_y} & 0 \end{bmatrix} \tag{2.44}$$

Ambas expresiones son similares, nótese que la restricción epipolar (ecuación 2.27 o 2.28) se traduce en una igualdad en la componente y, es decir, basta con buscar puntos correspondientes en la misma fila. Las ecuaciones 2.45 y 2.46 muestran este resultado para $\mathbf{F}$; el procedimiento para $\mathbf{E}$ es análogo (partiendo de la ecuación 2.27).

$$\vec{q_1}^T \mathbf{F} \vec{q_0} = 0 \iff \begin{bmatrix} i_1 & j_1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\frac{T_x}{f_y} \\ 0 & \frac{T_x}{f_y} & 0 \end{bmatrix} \begin{bmatrix} i_0 \\ j_0 \\ 1 \end{bmatrix} = 0 \tag{2.45}$$

$$j_1 = j_0 \tag{2.46}$$

2.2.3. Calibración estéreo

Calibración estéreo se refiere al proceso de computar la relación geométrica entre dos cámaras en el espacio. En otras palabras, corresponde a la estimación de la matriz de rotación $\mathbf{R}$ y del vector de traslación $\vec{T}$ que relacionan ambas cámaras.

Basándose en la misma expresión para el cambio de coordenadas (ecuación 2.14), es posible formular las ecuaciones 2.47 y 2.48. $\vec{P}$ corresponde a un punto arbitrario del sistema global de coordenadas. Por otro lado, $\mathbf{R}_0$, $\mathbf{R}_1$, $\vec{T_0}$ y $\vec{T_1}$ son las matrices de rotación y los

vectores de traslación que describen a los sistemas de las cámaras izquierda y derecha respecto de un sistema de coordenadas global (ver figura 2.6). Estos valores pueden ser estimados siguiendo el mismo algoritmo que para la calibración de solo una cámara (sección 2.1.2), pues se obtiene una rotación y una traslación para cada vista del patrón utilizado.

$$\vec{P_0} = \mathbf{R}_0\vec{P} + \vec{T_0} \tag{2.47}$$

$$\vec{P_1} = \mathbf{R}_1\vec{P} + \vec{T_1} \tag{2.48}$$

Por otro lado, $\mathbf{R}$ y $\vec{T}$ relacionan ambas cámaras a través de la ecuación 2.49. Conviene precisar que esta rotación y traslación llevan el sistema de referencia propio de la cámara derecha, al de la cámara izquierda.

$$\vec{P_0} = \mathbf{R}\vec{P_1} + \vec{T} \tag{2.49}$$

Reemplazando 2.47 y 2.48 en 2.49.

$$\mathbf{R}_0\vec{P} + \vec{T_0} = \mathbf{R}\left(\mathbf{R}_1\vec{P} + \vec{T_1}\right) + \vec{T} \tag{2.50}$$

$$(\mathbf{R}_0 - \mathbf{R}\mathbf{R}_1)\vec{P} + (\vec{T_0} - \mathbf{R}\vec{T_1} - \vec{T}) = 0 \tag{2.51}$$

$\vec{P}$ es genérico, por lo que en general no será nulo. Se obtienen entonces las siguientes ecuaciones:

$$\mathbf{R}_0 - \mathbf{R}\mathbf{R}_1 = 0 \tag{2.52}$$

$$\vec{T_0} - \mathbf{R}\vec{T_1} - \vec{T} = 0 \tag{2.53}$$

De donde es posible despejar $\mathbf{R}$ y $\vec{T}$.

$$\mathbf{R} = \mathbf{R}_0\mathbf{R}_1^T \tag{2.54}$$

$$\vec{T} = \vec{T_0} - \mathbf{R}\vec{T_1} \tag{2.55}$$

Como se dispone de un valor $\mathbf{R}_0$, $\mathbf{R}_1$, $\vec{T_0}$ y $\vec{T_1}$ para cada una de las vistas del patrón de calibración, se tiene también un valor de $\mathbf{R}$ y $\vec{T}$ para cada una de las vistas. Sin embargo, teóricamente se espera que todos estos valores sean relativamente parecidos, de forma que al considerar el promedio simple, u otra estrategia similar, se obtenga una buena aproximación para $\mathbf{R}$ y $\vec{T}$. Realizado este paso, el cálculo de las matrices $\mathbf{E}$ y $\mathbf{F}$ es natural según lo ya explicado en la sección 2.2.2. Se completan entonces, las principales ecuaciones tras la calibración estéreo.

2.2.4. Rectificación

La *rectificación estéreo* es el proceso de transformar las imágenes individuales de forma tal que parezcan que hubiesen sido tomadas desde cámaras perfectamente alineadas, esto es, que posean el mismo plano de imagen, y consecuentemente, los rayos principales se intercepten en el infinito (figura 2.9). En estricto rigor, es posible definir rectificaciones de forma de obtener imágenes asumiendo cualquier otra configuración. Sin embargo, en el presente trabajo se considerará solo el caso descrito, como si de un binocular se tratase. La ventaja de este caso particular es que permite el cálculo de profundidad mediante simple triangulación, tal como se mostró en la sección 2.2.1.

La matriz $\mathbf{R}$ y el vector $\vec{T}$ encontrados en la calibración, rotan y trasladan la cámara derecha hacia la cámara izquierda. Sin embargo, esto no garantiza que ambas imágenes pertenezcan al mismo plano, pues, ambas cámaras pueden tener distintas distancias focales. $\mathbf{R}$ y $\vec{T}$ simplemente alinean los rayos principales. Es necesario entonces añadir una nueva restricción que permita igualar los planos de imagen y a la vez, que mejore los resultados. El criterio escogido es maximizar las áreas comunes entre ambas imágenes.

El algoritmo de Bouguet [6] simplemente intenta minimizar los cambios de la re-proyección, y a la vez, maximizar las áreas comunes de las vistas. Con el fin de minimizar la distorsión en la re-proyección, la matriz $\mathbf{R}$ es dividida en $\mathbf{r}_0$ y $\mathbf{r}_1$, las que son medias rotaciones para las cámaras izquierda y derecha respectivamente. De esta forma, luego de aplicar estas rotaciones, las cámaras tendrán sus ejes principales alineados con el vector suma de los ejes principales originales. Los planos de imágenes quedan coplanares, sin embargo no necesariamente se quedarán alineados fila por fila, requiriéndose otra rotación.

Por definición, las matrices de rotación pueden ser construidas describiendo un nuevo sistema de coordenadas (dígase sistema $\vec{O_e}$) unitario en función del sistema de referencia original (dígase sistema $\vec{O_o}$). De esta forma, basta con establecer tres vectores orto-normales tales que orienten paralelamente los ejes principales.

De la etapa de calibración, se conoce el vector $\vec{T}$ el cual apunta desde el epipolo izquierdo al derecho. Por tanto, es posible utilizar esta dirección como primera referencia.

$$\vec{e_1} = \frac{\vec{T}}{\left\|\vec{T}\right\|} \tag{2.56}$$

El segundo vector, es calculado a partir del producto cruz entre la posición del eje principal $\vec{C}$ en el plano de imagen y el vector de traslación $\vec{T}$, de forma de garantizar la ortogonalidad.

$$\vec{e_2} = \frac{\vec{C} \times \vec{T}}{\left\|\vec{C} \times \vec{T}\right\|} \tag{2.57}$$

Dado que $\vec{C} = f\hat{z}$, la expresión para $\vec{e_2}$ puede simplificarse a

$$\vec{e_2} = \frac{T_x\hat{x} + T_y\hat{y}}{\sqrt{T_x^2 + T_y^2}} \tag{2.58}$$

Finalmente, último vector base puede calcularse con el producto cruz entre los otros dos vectores.

$$\vec{e_3} = \vec{e_1} \times \vec{e_2} \tag{2.59}$$

Formándose la matriz de rotación:

$$\mathbf{R_{eo}} = \begin{bmatrix} \vec{e_1} & \vec{e_2} & \vec{e_3} \end{bmatrix} \tag{2.60}$$

Luego, la matriz $\mathbf{R}_{eo}$ toma un punto en el sistema de referencia O_e y lo escribe en el sistema de referencia O_o. En este caso particular, se necesita la relación inversa, es decir, aquella que toma un punto descrito en O_o y lo describe en el nuevo sistema O_e. Como la inversa de una matriz de rotación es su traspuesta, la relación buscada queda dada por la ecuación 2.61.

$$\mathbf{R}_{oe} = \begin{bmatrix} \vec{e_1}^T \\ \vec{e_2}^T \\ \vec{e_3}^T \end{bmatrix} \tag{2.61}$$

Con esta matriz, se completan las rotaciones necesarias, por lo que las transformaciones de rotación completa quedan dadas por las ecuaciones 2.62 y 2.63, donde $\mathbf{R}_{0rec}$ y $\mathbf{R}_{1rec}$ dejan las cámaras izquierda y derecha alineadas en paralelo, correspondiendo fila a fila y maximizando el área común de vista.

$$\mathbf{R}_{0rec} = \mathbf{R}_{oe}\mathbf{r}_0 \tag{2.62}$$

$$\mathbf{R}_{1rec} = \mathbf{R}_{oe}\mathbf{r}_1 \tag{2.63}$$

Por otro lado, también son necesarias las matrices de cámaras del sistema rectificado. Las imágenes rotadas deben ser re-proyectadas en nuevas imágenes, dependiendo del tamaño, se deben escalar apropiadamente sus parámetros. Tomando como referencia el sistema asociado a la cámara izquierda, las ecuaciones 2.64 y 2.65 presentan las matrices de proyección completas.

$$\mathbf{K}'_0 = \mathbf{M}'_0\mathbf{K}_0 = \begin{bmatrix} f'_{x0} & 0 & c'_{x0} \\ 0 & f'_{y0} & c'_{y0} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} f'_{x0} & 0 & c'_{x0} & 0 \\ 0 & f'_{y0} & c'_{y0} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.64}$$

$$\mathbf{K}'_1 = \mathbf{M}'_1\mathbf{K}_1 = \begin{bmatrix} f'_{x1} & 0 & c_{x1'} \\ 0 & f_{y1'} & c_{y1'} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} f'_{x1} & 0 & c'_{x1} & T_x \\ 0 & f'_{y1} & c'_{y1} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.65}$$

Estas matrices de proyección, toman un punto tridimensional en coordenadas homogéneas y lo convierten a un punto bidimensional en coordenadas homogéneas de píxeles. La transformación opera acorde a la ecuación 2.66.

$$\mathbf{K} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} i_h \\ j_h \\ w \end{bmatrix} \tag{2.66}$$

Y por tanto, en coordenadas de píxeles, se tendrá: $i = \frac{i_h}{w}$ y $j = \frac{j_h}{w}$.

2.2.5. Correspondencias

Se define por correspondencia a un mismo punto del espacio visto o proyectado en dos imágenes obtenidas desde distintas tomas. Encontrar correspondencias entre dos imágenes constituye un desafío que ha sido abordado de múltiples formas. En general, el problema es llamado registro de imágenes, y se realiza con el fin de obtener una mayor cantidad de información de lo que se observa. Distintas tomas de un mismo objeto, podrían proporcionar, por ejemplo, información sobre la posición del mismo, de la posición de la fuente de luz, de distancias relativas entre distintos objetos, etc.

Para el presente trabajo, en particular se desea establecer un registro entre un par de imágenes estéreo. Una vez efectuado el proceso de rectificación, se espera que dado un punto, su punto correspondiente se encuentre trasladado solo horizontalmente, bastando con realizar una búsqueda unidimensional.

Existen dos grandes líneas a seguir con tal de resolver el problema. La primera radica en un enfoque discreto, encontrando correspondencias solo en puntos determinados, los que pueden ser relativamente fáciles de detectar en ambas imágenes; mientras que la otra busca correspondencias para cada píxel de las imágenes, basándose en la intensidad de los colores.

La ventaja de los métodos basados en intensidades es que permiten encontrar correspondencias para la totalidad de las imágenes mientras que en el enfoque basado en puntos, solo se tiene solución en un subconjunto muy reducido, requiriéndose algún tipo de interpolación.

Por otro lado, es propio del registro de imágenes estéreo la existencia de *oclusiones*, es decir, regiones presentes en una imagen que no son vistas en la imagen vecina. Esta característica produce que los métodos basados en puntos característicos sean más robustos.

Otro nivel de clasificación es como se tratan las imágenes, pudiendo ser vistas como funciones discretas (matrices) o continuas (superficies). Establecer un registro en el caso discreto consiste en una re-indexación de la imagen; mientras que en el caso continuo, se requiere una función completa de deformación. El enfoque discreto en general está basado en teoría de señales, en oposición a las estrategias continuas, planteándose como solución a ecuaciones diferenciales.

En [19] y [6] se presentan esquemas de trabajo en base a puntos característicos. Por otro lado, en [14] y [4] se utilizan esquemas continuos. En [18] se evalúa una gran cantidad de

algoritmos con fines de establecer correspondencias entre imágenes estéreo. La figura 2.13 presenta distintos mapas de disparidad obtenidos para las imágenes *Tsukuba*.

Independiente de cual sea la estrategia utilizada, existe una serie de reglas que se cumplen para el caso de imágenes estéreo.

- Se preserva el orden de los elementos.
- Sólo se esperan variaciones horizontales entre las posiciones de los objetos en las imágenes.
- Cada punto tiene un único punto correspondiente en la otra imagen.

Al disponer de las correspondencias para cada punto de las imágenes, se posee un *mapa de correspondencias*. Ahora, como en general solo se esperan desplazamientos horizontales entre ambas imágenes, es de utilidad disponer del llamado *mapa de disparidad*, donde cada punto tiene asociado un valor escalar de cuanto se ha desplazado hacia la izquierda o derecha de la imagen original para observarse en la otra imagen. Teniendo las correspondencias, es natural el cálculo de disparidades a través de la relación 2.67; donde (x, y) corresponde a un punto en una de las imágenes, $\phi(x, y)$ el punto correspondiente a (x, y) en la otra imagen; y finalmente, $d(x, y)$ es la disparidad asociada al punto.

$$d(x, y) = \phi(x, y) - (x, y) \tag{2.67}$$

Figura 2.13: Mapas de disparidad de las imágenes *Tsukuba* generados con estrategias actuales. [18]

2.2.6. Mapas de profundidad

En un *mapa de profundidad*, cada punto de la imagen tiene asociada la distancia a la que se encuentra de la cámara. Disponiendo de imágenes rectificadas y un mapa de disparidad; el cómputo de un mapa de profundidad es simple y natural, gracias a la maquinaria construida en las secciones anteriores. Basta con pre-multiplicar cada punto por la matriz de

re-proyección (ver ecuación 2.25), acorde a la ecuación 2.26.

Existe una estrecha relación entre disparidad y profundidad, pero en concreto, conviene trabajar en el espacio del mapa de disparidad. De esta forma, se trabaja directamente en coordenadas de píxeles, y se logra una mayor precisión para los objetos que se encuentren relativamente cercanos a la cámara. En la figura 2.14 se observa como términos similares, podrían eventualmente confundirse en el dominio de la profundidad, mientras que en el de la disparidad, permanecen claramente diferentes. La razón es la no linealidad de la curva presentada en la figura 2.25.

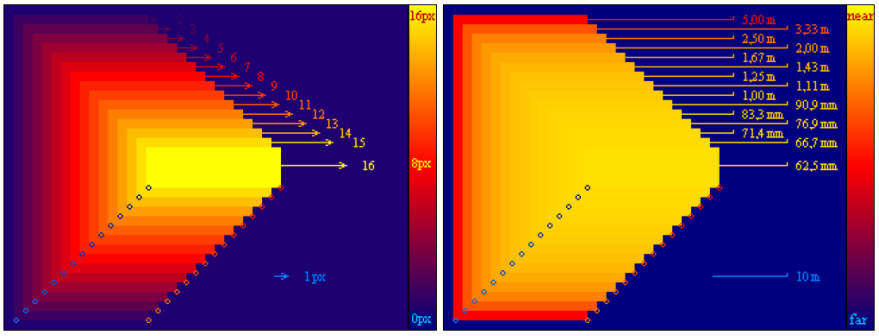

Figura 2.14: Mapa de disparidad vs Mapa de profundidad. [14]

2.3. Deformaciones

2.3.1. Introducción

En primer lugar es necesario precisar matemáticamente el concepto de deformación. Para esto, considere un conjunto abierto Ω en $\mathbb{R}^k$. Una deformación es una función φ que asigna a cada punto $x \in \Omega$ una posición desplazada $y = \varphi(x) \in \Omega$. También es deseable que:

- La deformación no cree agujeros. Cada punto $y \in \Omega$ sea la imagen de algún punto $x \in \Omega$. Es decir, que φ sea epiyectiva.
- Que no se formen pliegues. Dos puntos distintos x y x' en Ω no deberían tener una misma imagen $y \in \Omega$. Es decir, que φ sea inyectiva.

Luego, las deformaciones deben ser biyecciones de Ω. Adicional a esto, se requieren algunas condiciones de suavidad para φ, por lo que se introducen los siguientes conceptos:

- Un *homomorfismo* de Ω es una biyección continua $\varphi : \Omega \to \Omega$ tal que su inversa φ^{-1} es continua.
- Un *difeomorfismo* de Ω es un homomorfismo continuamente diferenciable $\varphi : \Omega \to \Omega$ tal que su inversa φ^{-1} es continuamente diferenciable.

De esta forma, se define *deformación* como un *difeomorfismo* de un conjunto abierto $\Omega \subset \mathbb{R}^k$.

Considérese como ejemplo una imagen $I : \Omega \to \mathbb{R}$ y un difeomorfismo φ de Ω. La deformación creará una nueva imagen I' en Ω permitiendo que $I'(y)$ sea un valor de I en la posición x que apuntó a y a través de la deformación $y = \varphi(x)$, es decir, $I'(y) = I(x = \varphi^{-1}(y))$ ó $I' = I \circ \varphi^{-1}$. Luego, la acción de un difeomorfismo sobre una función equivale a la operación de composición, por lo que puede ser interpretado como un remapeo del dominio de la imagen original.

Por otro lado, el problema inverso, es decir, dado un par de imágenes I e I', encontrar el difeomorfismo φ que mejor las relacione, no es tarea fácil. Este problema es el de establecer correspondencias entre un par de imágenes.

Existe además el problema de construir difeomorfismos, pues una combinación lineal de difeomorfismos, no necesariamente será un difeomorfismo. Ahora, la operación de composición sí preserva la condición de difeomorfismo, por lo que a continuación se formula una estrategia en base a esta característica.

2.3.2. Generando difeomorfismos

Una directa, pero limitada forma de construir difeomorfismos se basa en pequeñas perturbaciones a la identidad. La idea es deformar la imagen lentamente a través de una secuencia de transformaciones difeomórficas.

Proposición 1. *Sea $u \in C^1(\Omega, \mathbb{R}^k)$ y que para algún $\delta > 0$, se tiene que $u(x) = 0$ para cualquier $x \in \Omega$ tal que existe $y \notin \Omega$ que cumpla $|x - y| < \delta$. Entonces, para un ϵ lo suficientemente pequeño, $\varphi : x \to x + \epsilon u(x)$ es un difeomorfismo de Ω.*

Demostración en [25].

□

Utilizando la proposición anterior, construir pequeñas deformaciones es simple, pero estas constituyen una clase muy limitada de difeomorfismos. Sin embargo, es posible utilizar una serie de estas pequeñas deformaciones para generar una gran deformación [3, 25], dado que, como se ha mencionado, los difeomorfismos pueden ser combinados bajo la regla de composición.

Así, sea $\epsilon_0 > 0$ y $u_1, ..., u_n, ...$ campos vectoriales en Ω tales que, para $\epsilon < \epsilon_0$, $id + \epsilon u_i$ es un difeomorfismo de Ω. Considere $\varphi_n = (id + \epsilon u_n) \circ ... \circ (id + \epsilon u_1)$. Se tiene entonces:

$$\varphi_{n+1} = (id + \epsilon u_n) \circ \varphi_n = \varphi_n + \epsilon u_n \circ \varphi_n \tag{2.68}$$

Lo que puede ser reescrito como $(\varphi_{n+1} - \varphi_n)/\epsilon = u_n \circ \varphi_n$. Esto puede ser interpretado como la discretización de una ecuación diferencial, introduciendo una variable de tiempo t:

$$\frac{d\varphi_t(x)}{dt} = u_t(\varphi_t(x)) \tag{2.69}$$

En palabras, la imagen se deforma poco a poco en cada instante. En este sentido, existe un difeomorfismo para cada instante de tiempo. El problema de encontrar una gran deformación, se ha traducido en encontrar la solución a una ecuación diferencial. Es necesario añadir la condición inicial $\varphi_{t=0}(x) = x$, pues se parte de una de las imágenes, y se deforma para convertirse en la segunda. Conviene efectuar un pequeño cambio de notación.

$$\frac{d\phi(x,t)}{dt} = v(\phi(x,t),t) \quad \phi(x,t=0) = x \tag{2.70}$$

La misma ecuación puede ser más clara en su forma integral, pues partiendo de la identidad, el difeomorfismo ϕ evoluciona en el tiempo acorde a lo indicado por el campo v.

$$\phi(x,t) = x + \int_0^t v(\phi(x,\tau),\tau)d\tau \tag{2.71}$$

En este sentido, la incógnita del problema es la función v, una vez encontrada, es natural el cálculo del difeomorfismo ϕ. Ahora, encontrar una función $v(x,t)$ que resuelva el problema, es una tarea compleja. En el presente trabajo se simplificará el problema, asumiendo que el campo vectorial v no depende el tiempo. Por lo que las ecuaciones 2.70 y 2.71 se transforman en 2.72 y 2.73.

$$\frac{d\phi(x,t)}{dt} = v(\phi(x,t)) \qquad \phi(x,t=0) = x \tag{2.72}$$

$$\phi(x,t) = x + \int_0^t v(\phi(x,\tau))d\tau \tag{2.73}$$

Este caso particular se conoce como *caso estacionario* y permite encontrar una solución de manera más simple. Se asume entonces que un campo v estacionario proporciona la generalidad suficiente para resolver el problema. Conviene destacar que la ecuación 2.72 no permite encontrar cualquier deformación, pues se busca solo en aquella familia de difeomorfismos generados a través de un campo vectorial estacionario.

Por otro lado, el caso general, con $v = v(x,t)$, es llamado *dependiente del tiempo* o simplemente *no-estacionario*. Existen distintas estrategias desarrolladas para tratar este problema, pero dada su dificultad, es simplemente omitido del presente trabajo.

En [5] se presentan distintos enfoques para resolver este problema (caso estacionario). En particular, se considerará el llamado método Euler de avance[2], cuya formulación (ecuación 2.74) corresponde a una evolución temporal de la función difeomorfismo.

$$\phi(x,t+dt) = \phi(x) + v(\phi(x,t))\,dt \tag{2.74}$$

Donde $\phi(x,0) = x$ y $dt = \frac{1}{T}$. Este método posee dos grandes ventajas: la precisión puede ser controlada con el número de pasos T (el cual puede ser arbitrariamente grande), y

[2]en literatura, referenciado como *Forward Euler Method.*

segundo, solo es necesario interpolar el campo vectorial v. Otros métodos requieren interpolar el difeomorfismo, lo que añade complejidad en la implementación, y por lo tanto, se está mas propenso a errores [5].

Como se trata de una aproximación numérica, es necesario asegurar que ϕ permanezca en el dominio Ω, por lo cual, se requiere que el campo v sea nulo en la frontera del dominio (ecuación 2.75). La consecuencia directa de esta restricción, es que no habrá deformación en la frontera (ecuación 2.76) y por lo tanto, se asumirá que las imágenes solo presentan variaciones en su interior.

$$v\left(x\right) = 0 \qquad \forall x \in \partial\Omega \tag{2.75}$$

$$\phi\left(x\right) = x \qquad \forall x \ \in \partial\Omega \tag{2.76}$$

Para el caso particular de aplicación en imágenes estéreo, los bordes serán distintos, por lo que este último supuesto, en general no se cumple. De hecho, exclusivamente en los bordes, se visualizarán elementos que no están presentes en la imagen vecina (las cámaras están trasladadas horizontalmente). Para resolver este problema, se propone una etapa previa de recorte, con tal de lograr una vista sólo de las zonas comunes. Sin embargo, debido a la perspectiva, conseguir imágenes con bordes iguales no siempre es posible, por lo que en muchos casos, los resultados serán inválidos en los límites de las imágenes.

2.3.3. Un problema de optimización

Se dispone de una ecuación diferencial (ecuación 2.72) genérica para un campo $v \in C^1(\Omega, \mathbb{R}^k)$ que garantiza la obtención de un difeomorfismo. El problema es encontrar el difeomorfismo ϕ que mejor relaciona un par de imágenes I_1 e I_2, por lo tanto, es necesario determinar un campo v que cumpla este objetivo en la ecuación diferencial. De esta forma, se propone un funcional de energía, cuyo objetivo sea encontrar el campo (y consecuentemente el difeomorfismo), la ecuación 2.72 se interpreta simplemente como una restricción.

$$F = F_{sim}(I_1, I_2, \phi) + \alpha F_{reg}(\phi) \tag{2.77}$$

El funcional F se ha descompuesto en 2 términos con distintos propósitos. F_{sim} o funcional de similitud, intenta efectuar la correspondencia (o *matching*) entre ambas imágenes. Por otro lado, F_{reg} o funcional de regularización, penaliza el comportamiento no deseado del difeomorfismo ϕ, típicamente asegura que ϕ sea lo suficientemente suave y no muy distinto a la identidad [25].

Como funcional de similitud típicamente se escoge una métrica de distancia, un ejemplo natural es el error cuadrático medio o norma L_2 entre una imagen y la segunda imagen deformada. Por otro lado, como funcional de regularización, es posible escoger alguna norma Sobolev sobre el campo vectorial o difeomorfismo. Las tablas 2.1 y 2.2 presentan algunos ejemplos de funcionales de similitud y regularización respectivamente.

Métrica	Funcional
Norma L2	$F_{simL2} = \int_{\Omega} \left(I_1(x) - I_2(\phi(x))\right)^2 dx$
Norma L1	$F_{simL1} = \int_{\Omega} \left\|I_1(x) - I_2(\phi(x))\right\| dx$

Tabla 2.1: Posibles funcionales de similitud.

Métrica	Funcional
Energía de v	$F_{reg} = \int_{\Omega} \left(v\left(x\right)\right)^2 dx$
Energía del laplaciano de v	$F_{reg} = \int_{\Omega} \left(\triangle v\left(x\right)\right)^2 dx$

Tabla 2.2: Posibles funcionales de regularización.

Es posible mezclar varios funcionales, sin embargo, cada uno de ellos debe ser apropiadamente calibrado, es decir, ponderado correctamente de forma tal que se logre un buen efecto de deformación, mezclando similitud y regularización. La idea es estructurar ambos funcionales de forma que se posea un mínimo global. En estas condiciones, es posible resolver el problema de optimización mediante el método del gradiente.

$$v^k = v^{k-1} - \delta \frac{dF^{k-1}}{dv} \tag{2.78}$$

Hecho esto, el cálculo de la deformación, es directo a través de la ecuación 2.74. Solo resta agregar una situación inicial al campo v, como se trata de dos imágenes relativamente similares, no se esperan grandes deformaciones. En este sentido, la condición inicial puede ser una función nula, indicando que en un primer caso, no habrá deformación.

$$v^0 = 0 \tag{2.79}$$

Es interesante notar que se tiene un problema de optimización global, sin embargo, su efecto ataca localmente en cada parte de la curva que define v. Acorde a la magnitud del gradiente del funcional, la curva se desviará en mayor o menos magnitud del valor que posea en la iteración anterior. Destaca además que se vuelve esencial la calibración del parámetro δ, la cual controlará la velocidad de convergencia del algoritmo.

2.4. B-splines

2.4.1. Introducción

Las *splines* son polinomios por trozos, los que se unen suavemente. Los puntos de unión son llamados nodos (o *knots* en inglés). Para una spline de grado n, cada segmento es un polinomio de grado $n-1$, lo que sugiere que se necesitan n coeficientes para definir cada trozo de curva. Sin embargo, existen $n-1$ restricciones de continuidad y suavidad para la spline y sus derivadas, lo que deja solo un grado de libertad por segmento. En este trabajo

solo se considerarán splines con nodos uniformemente espaciados, pensando que se utilizarán para interpolar o representar una imagen.

Una spline cualquiera, puede ser caracterizada de manera única en términos de una expansión B-spline[3].

$$s(x) = \sum_{k \in \mathbb{Z}} c[k] \beta^n (x - k) \tag{2.80}$$

Donde β^n define una base B-spline de orden n (ver ecuaciones 2.81 y 2.82). De esta forma, los coeficientes $c[k]$ ponderan la B-spline de orden n trasladada en k.

$$\beta^n(x) = \beta^1 * \beta^{n-1}(x) \qquad \forall x \in \mathbb{R} \tag{2.81}$$

$$\beta^1(x) = \begin{cases} 1 & x \in \left[-\frac{1}{2}, \frac{1}{2}\right) \\ 0 & \sim \end{cases} \tag{2.82}$$

De las definiciones anteriores[4], destaca que la B-spline solo posee valores no nulos en el intervalo $\left[-\frac{n}{2}, \frac{n}{2}\right)$. Se dice entonces que es de soporte compacto[5] y se nota $supp\beta^n(x) = \left[-\frac{n}{2}, \frac{n}{2}\right)$.

La figura 2.15 muestra las B-splines desde el orden $n = 1$ hasta $n = 4$. Es claro además que presentan continuidad siendo de clase C^{n-1}. Usando un orden superior, se obtienen mejores condiciones de suavidad, pero se agranda el soporte de la B-spline.

Se define además la B-spline ensanchada en m como muestra las ecuación 2.83. Esta definición permite trabajar con muestras equiespaciadas a intervalos de largo m. De esta forma, ya no es necesario disponer de datos a distancia unitaria.

$$\beta_m^n(x) = \beta^n \left(\frac{x}{m}\right) \quad \forall x \in \mathbb{R} \tag{2.83}$$

Por último, se define también la versión discreta de la misma, como una simple evaluación de β_m^n en un punto entero.

$$b_m^n[k] = \beta_m^n(k) \qquad \forall k \in \mathbb{Z} \tag{2.84}$$

[3] I.J. Schoenberg, "Contribution to the problem of aproximation of equidistant data by analytic functions" *Quart. Appl. Math.*, vol. 4, pp. 45-99, 112-141, 1946.

[4] Dependiendo de la bibliografía, algunas veces se define como bspline base (el rectángulo) β^0 en vez de β^1. También pueden definirse de manera no centrada.

[5] Se dice que una función es de soporte compacto, si la adherencia del conjunto donde la función no es nula, es cerrado y acotado.

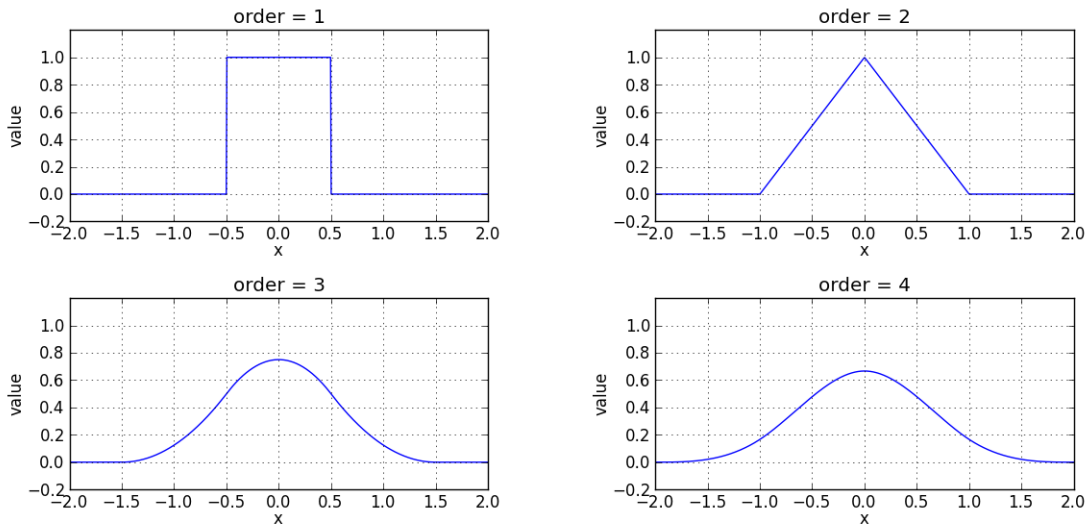

Figura 2.15: Bases B-splines de orden 1 a 4.

2.4.2. Propiedades

El uso de B-splines presenta varias propiedades y características interesantes, a continuación se enumeran algunas de ellas.

1. Derivada de una base B-spline de orden n.

$$\frac{d\beta^n}{dx} = \beta^{n-1}\left(x + \frac{1}{2}\right) - \beta^{n-1}\left(x - \frac{1}{2}\right) \tag{2.85}$$

2. La segunda derivada de una B-spline de orden n, puede ser expresada como la siguiente convolución continua:

$$\frac{d^2\beta^n(x)}{dx^2} = d_2(x) * \beta^{n-2}(x) \tag{2.86}$$

   Donde d_2 se define como el laplaciano discreto.

$$d_2(x) = \delta(x+1) - 2\delta(x) + \delta(x-1) \tag{2.87}$$

3. Integral de una B-spline de orden n.

$$\int_{-\infty}^{x} \beta^n(x)\,dx = \sum_{k=0}^{+\infty} \beta^{n+1}\left(x - \frac{1}{2} - k\right) \tag{2.88}$$

4. Una B-spline de orden n, tiene soporte $\left[-\frac{n}{2}, \frac{n}{2}\right)$.

5. Paridad.

$$\beta^n(x) = \beta^n(-x) \tag{2.89}$$

6. La convolución de 2 B-splines de distinto orden, genera una nueva B-spline de orden igual a la suma.

$$\beta^n(x) * \beta^m(x) = \beta^{n+m}(x) \tag{2.90}$$

2.4.3. Evaluando una B-spline

Con fines de optimizar el tiempo de cómputo del valor de una función definida en base a B-splines, conviene implementar directamente las bases, sin que se requieran pasos intermedios de convolución. La tabla 2.3 presenta expresiones analíticas para las bases B-spline de orden 1 a 4.

Orden	β^n
1	$\beta^1(x) = \begin{cases} 1 & x \in [-0.5, 0.5) \\ 0 & \sim \end{cases}$
2	$\beta^2(x) = \begin{cases} 1+t & x \in [-1, 0) \\ 1-t & x \in [0, 1) \\ 0 & \sim \end{cases}$
3	$\beta^3(x) = \begin{cases} \frac{t^2}{2} + \frac{3t}{2} + \frac{9}{8} & x \in [-1.5, -0.5) \\ \frac{3}{4} - t^2 & x \in [-0.5, 0.5) \\ \frac{t^2}{2} - \frac{3t}{2} + \frac{9}{8} & x \in [0.5, 1.5) \\ 0 & \sim \end{cases}$
4	$\beta^4(x) = \begin{cases} \frac{t^3}{6} + t^2 + 2t + \frac{4}{3} & x \in [-2, -1) \\ -\frac{t^3}{2} - t^2 + \frac{2}{3} & x \in [-1, 0) \\ \frac{t^3}{2} - t^2 + \frac{2}{3} & x \in [0, 1) \\ -\frac{t^3}{6} + t^2 - 2t + \frac{4}{3} & x \in [1, 2) \\ 0 & \sim \end{cases}$

Tabla 2.3: Bases B-spline de orden 1 a 4.

Gracias a la característica de soporte compacto, para efectuar la evaluación de un punto arbitrario x, solo se debe evaluar un número de funciones base igual al orden utilizado, y estos, ponderarlos adecuadamente. En este sentido, para lograr un buen rendimiento, conviene determinar el nodo más cercano a x de forma previa, y así evaluar solo las bases que influyen en el cálculo. Es útil discriminar entre los casos par e impar:

- Si el orden de las B-splines es par, la base posee un soporte que comienza y termina en un número entero. Por esta razón, sirve encontrar el nodo i entero inferior más cercano al x arbitrario, y a la vez calcular un $\epsilon = x - i$ que estará en el intervalo $[0, 1)$.

- Ahora, si se trata de un orden impar, el soporte de la base comienza y termina justo en el punto medio entre dos nodos consecutivo. En estas condiciones, conviene encontrar simplemente el nodo j más cercano al valor de x, y al mismo tiempo, calcular un $\delta = x - j$ perteneciente al intervalo $[-0.5, 0.5)$.

La figura 2.16 muestra la evaluación de funciones B-spline de ordenes 2 y 3, ilustrando la diferencia entre ambos casos. Por ejemplo, observando la figura, si se quiere evaluar $x = 6.7$ en ambos casos, entonces se calcula $(i, \epsilon) = (6, 0.7)$ para el caso de orden 2, y $(j, \delta) = (7, -0.3)$ cuando el orden es 3. La tabla 2.4 presenta analíticamente las expresiones a evaluar para las curvas de distinto orden a partir de los pares (i, ϵ) y (j, δ) ya determinados.

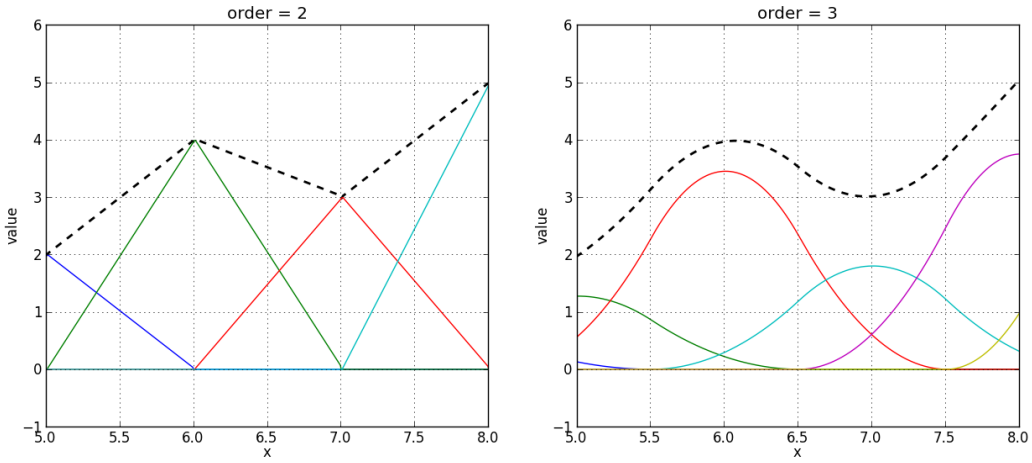

Figura 2.16: Evaluación de funciones B-splines para ordenes 2 y 3.

Orden	Evaluación
1	$f(x) = c[j]$
2	$f(x) = c[i]\,\beta^2(\epsilon) + c[i+1]\,\beta^2(\epsilon - 1)$
3	$f(x) = c[j-1]\,\beta^3(\delta + 1) + c[j]\,\beta^3(\delta) + c[j+1]\,\beta^3(\delta - 1)$
4	$f(x) = c[i-1]\,\beta^4(\epsilon + 1) + c[i]\,\beta(\epsilon) + c[i+1]\,\beta^4(\epsilon - 1) + c[i+2]\,\beta^4(\epsilon - 2)$

Tabla 2.4: Evaluando una función B-spline.

El cómputo de las derivadas también aprovecha la característica de soporte compacto. Utilizando los mismos valores (i, ϵ) y (j, δ), basta con efectuar operaciones en los coeficientes, y luego evaluar una cantidad reducida de bases B-spline de un orden inferior. La tabla 2.5 presenta las expresiones para evaluar las funciones B-spline de órdenes 1 a 4 (por simplicidad, se omiten las discontinuidades para la derivada del orden 1).

2.4.4. Interpolación

Se tiene una secuencia de puntos $s[k]$ equiespaciados en k. Se quiere encontrar una expresión $f(x)$ continua de forma que se cumpla $f(x = k) = s[k]$ y determinadas condiciones de suavidad. El problema a abordar se refiere a encontrar los coeficientes $c[k]$ que ponderan las

Orden	Derivada
1	$\frac{df(x)}{dx} = 0$
2	$\frac{df(x)}{dx} = \left(c\left[i+1\right]_{i+1} - c\left[i\right]_i\right)\beta^1\left(\epsilon - 0.5\right)$
3	$\frac{df(x)}{dx} = \left(c\left[j\right] - c\left[j-1\right]\right)\beta^2\left(\delta + 0.5\right) + \left(c\left[j+1\right] - c\left[j\right]\right)\beta^2\left(\delta - 0.5\right)$
4	$\frac{df(x)}{dx} = \left(c\left[i\right] - c\left[i-1\right]\right)\beta^3\left(\epsilon + 0.5\right) + \left(c\left[i+1\right] - c\left[i\right]\right)\beta^3\left(\epsilon - 0.5\right) + \left(c\left[i+2\right] - c\left[i+1\right]\right)\beta^3\left(\epsilon - 1.5\right)$

Tabla 2.5: Evaluando la derivada de una función B-spline.

bases B-spline de forma tal que la función resultante pase exactamente por lo puntos dados de $s[k]$.

Para el caso de $n = 1$ y $n = 2$, el problema es trivial, dado que los coeficientes pueden ser los mismos valores de $s[k]$. Cuando se considera $n = 1$ se está interpolando con rectángulos, mientras que con $n = 2$ se interpola linealmente entre cada par de puntos. La simpleza de estos casos particulares, se debe a que la amplitud de las bases es menor a 2 y cada B-spline no influye con la B-spline vecina. Sin embargo, esto no ocurre para los órdenes superiores, como se muestra en la figura 2.15.

La ecuación 2.80 debe cumplirse para los valores conocidos $s[l]$, resultando la expresión:

$$s[l] = \sum_{k\in\mathbb{Z}} c[k]\beta^n(l-k) \tag{2.91}$$

Como β^n solo se está evaluando en valores enteros, es posible cambiarla por su versión discreta b_1^n, resultando la ecuación 2.92, lo que corresponde a una convolución discreta.

$$s[l] = \sum_{k\in\mathbb{Z}} c[k]b_1^n[l-k] = (c * b_1^n)[l] \tag{2.92}$$

$$s = c * b_1^n \tag{2.93}$$

Siendo $B(z)$ la transformada Z de $b_1^n[k]$, entonces, $\frac{1}{B(z)}$ es la transformada Z de una función $f[k]$ tal que $b_1^n * f[k] = 1$. Gracias a esta característica, $f[k]$ es llamada $(b_1^n)^{-1}$. Luego, la solución de los coeficientes $c[k]$ puede ser formulada según la ecuación 2.94. Este proceso de cálculo es conocido como filtrado inverso, y es abordado en la sección 2.4.5.

$$c = s * (b_1^n)^{-1} \tag{2.94}$$

Por otro lado, es posible obtener una expresión análoga a 2.80 pero utilizando como coeficientes los $s[k]$ conocidos, y una base distinta de las B-spline β^n. La expresión es:

$$s(x) = \sum_k s[k]\eta^n(x-k) \tag{2.95}$$

Para llegar a esta expresión, se debe proceder como sigue, partiendo desde la ecuación 2.80.

$$s(x) = \sum_k \left(s * (b_1^n)^{-1}\right)[k]\beta^n(x-k) \tag{2.96}$$

$$s(x) = \sum_k \left(\sum_l s[l]\,(b_1^n)^{-1}[k-l]\right)\beta^n(x-k) \tag{2.97}$$

$$s(x) = \sum_l s[l] \sum_k (b_1^n)^{-1}[k-l]\beta^n(x-k) \tag{2.98}$$

Haciendo el cambio de variable discreta $j = k - l \;\Rightarrow\; k = j + l$, se obtiene:

$$s(x) = \sum_{l\in\mathbb{Z}} s[l] \sum_{j\in\mathbb{Z}} (b_1^n)^{-1}[j]\beta^n(x-l-j) \tag{2.99}$$

Finalmente, se define:

$$\eta^n(x) = \sum_{k\in\mathbb{Z}} (b_1^n)^{-1}[k]\beta^n(x-k) \tag{2.100}$$

Con lo que reemplazando en la ecuación 2.99, se obtiene la expresión 2.95. η^n se conoce como función *interpoladora* o *B-spline cardinal [20]* debido a que vale 1 cuando se evalúa en $x = 0$ y vale 0 en cualquier otro número entero. Esta función es similar a la famosa función $sinc\,(x) = \frac{sin(x)}{x}$ (ver figura 2.17). A pesar de la ventaja de no requerir un cómputo de coeficientes, no son muy utilizadas dado que su evaluación es lenta al tener un soporte más amplio.

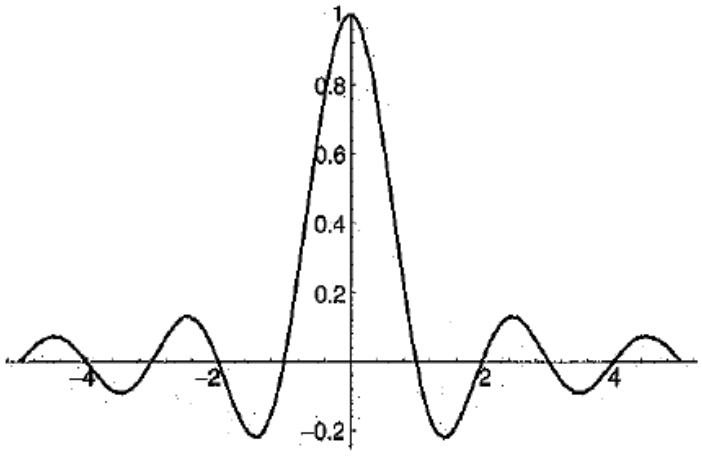

Figura 2.17: B-spline cardinal de orden 4. [20]

2.4.5. Filtrado inverso

La ecuación 2.94 permite calcular los coeficientes de la B-spline para determinados valores de la señal. Sin embargo, esto parte del supuesto donde se conocen los valores de la señal discreta $(b_1^n)^{-1}$.

Como b_1^n es un filtro FIR[6], el llamado filtro directo B-spline $(b_1^n)^{-1}$ es un sistema que puede ser implementado muy eficientemente usando una combinación en cascada de filtros recursivos, siendo uno causal y otro anti-causal [21, 23]. Este algoritmo es estable numéricamente y es más rápido y fácil de implementar que cualquier otra técnica numérica [20].

Supongamos un filtro FIR de segundo orden $a[k]$ cuya transformada Z es $A(z)$. Es posible encontrar la siguiente descomposición:

$$A(z) = \alpha \left(\frac{1}{1 - z_1 z^{-1}} \right) \left(\frac{-z_1}{1 - z_1 z} \right) \tag{2.101}$$

Donde z_1 es la raíz de $A\,(z)$. Basándose en esta descomposición, se extraen las ecuaciones de diferencias 2.102. Luego, efectuando solo 2 multiplicaciones y 2 adiciones por cada muestra, se obtiene la señal resultante del filtrado inverso.

$$\begin{aligned} c^+[k] &= s[k] + z_1 c^+[k+1] \\ c^-[k] &= z_1 (c^-[k+1] - c^+[k]) \end{aligned} \tag{2.102}$$

Dado que un filtro de orden superior puede ser descompuesto como la acción en cascada de varios filtros de orden 2, el procedimiento explicitado aplica en cualquier orden, basta con determinar sus raíces y la constante de multiplicación. De esta forma, se aplica un filtrado inverso de orden 2 para cada una de las raíces. La tabla 2.4.5 presenta las raíces y funciones de transferencia de los filtros B-splines directos hasta el orden 8.

[6]del inglés *Finite Impulse Response*

Orden	Filtro B-spline directo	α	Polos
1	1	1	-
2	1	1	-
3	$\frac{8}{z+6+z^{-1}}$	8	$z_1 = -0.171573$
4	$\frac{6}{z+6+z^{-1}}$	6	$z_1 = -0.267949$
5	$\frac{384}{z^2+76z+230+76z^{-1}+z^{-2}}$	384	$z_1 = -0.361341$ $z_2 = -0.0137254$
6	$\frac{120}{z^2+26z+66+26z^{-1}+z^{-2}}$	120	$z_1 = -0.430575$ $z_2 = -0.0430963$
7	$\frac{46080}{z^3+722z^2+10543z+23548+10543z^{-1}+722z^{-2}+z^{-3}}$	46080	$z_1 = -0.488295$ $z_2 = -0.0816793$ $z_3 = -0.00141415$
8	$\frac{5040}{z^3+120z^2+1191z+2416+1191z^{-1}+120z^{-2}+z^{-3}}$	5040	$z_1 = -0.53528$ $z_2 = -0.122555$ $z_3 = -0.00914869$

Tabla 2.6: Funciones de transferencia y polos de Filtros B-splines directos de orden 1 a 8

2.4.6. Condiciones de borde

Las ecuaciones y definiciones anteriores asumen una señal discreta de entrada de largo infinito, esto es, definida en todo $\mathbb{Z}$. En la práctica, solo se posee un tramo finito de la señal. En este sentido, es necesario extrapolar la señal hacia valores no definidos inicialmente.

Una alternativa sería asumir un valor nulo para todo índice no definido. Otra opción es asumir que la señal se refleja como un espejo en cada extremo, el reflejo puede ser a partir de la primera muestra o desde la segunda, estas condiciones son referenciadas como FSMC[7] y HSMC[8], respectivamente. En [13] se tratan distintas formas de extrapolar los datos presentes en una señal finita.

FSMC permite simular un borde mas suave, pues, dado que hay una muestra repetida, se impone implícitamente una derivada cero en el borde. Por otro lado, HSMC involucra un quiebre algo más abrupto, pero aún así, menor al producido en el caso de rellenar con ceros. Para el caso unidimensional, estas condiciones son ilustradas por la figura 2.18. En el presente trabajo se utilizan las condiciones de borde de tipo FSMC.

[7] del inglés *Full Sample Mirror Condition*
[8] del inglés *Half Sample Mirror Condition*

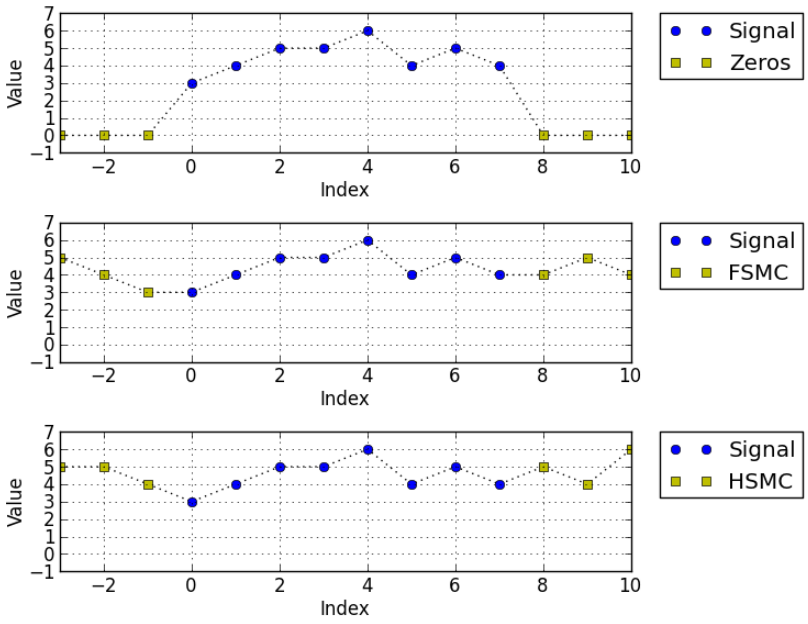

Figura 2.18: Distintas condiciones de borde de extrapolación.

Capítulo 3

Implementación

3.1. Descripción general

El problema completo involucra varias partes, las que van desde la calibración de la cámara estéreo, hasta la generación del mapa de profundidad. De esta forma, se estructuró un sistema de módulos independientes para atender cada una de las tareas que se necesita realizar. En concreto, estos módulos son:

- Calibración de una cámara: Se trata la obtención de los parámetros intrínsecos de cada cámara independientemente. Estos parámetros son guardados en archivos para su posterior uso.
- Calibración estéreo: Se obtienen los parámetros tanto intrínsecos como extrínsecos entre ambas cámaras. Pueden o no utilizarse parámetros de partida obtenidos independientemente con el módulo de calibración anterior.
- Recorte y alineación: Las imágenes rectificadas con los parámetros del módulo de calibración estéreo pueden poseer distintas dimensiones y una traslación natural del fondo presente en la imagen. Este módulo permite al usuario alinear y recortar apropiadamente las imágenes, igualando bordes y preparándolas para la etapa de registro no rígido.
- Filtrado de las imágenes: Este módulo permite aplicar varias opciones filtrado y reducción vertical. El objetivo es facilitar el trabajo del posterior registro.
- Registro no rígido: Tomando como entrada dos imágenes ya pre-procesadas, se determina la deformación fila por fila de la imagen. En particular, se obtiene el desplazamiento horizontal asociado a cada fila de la imagen izquierda.
- Generación de mapa de profundidad: Teniendo el registro, la información de alineación y los parámetros de la cámara estéreo, este módulo es capaz de generar tanto un mapa de disparidad, como de profundidad.

- Visualización de deformación por fila: Un simple pero útil programa que permite visualizar las curvas originales y deformadas de cada fila, así como también observar y contrastar con la función de deformación resultante.

La descripción del sistema completo queda dado por el diagrama de bloques presentado en la figura 3.1. Por otro lado, los módulos de calibración pueden ser esquematizados independientemente, y se presentan en las figuras 3.2 y 3.3.

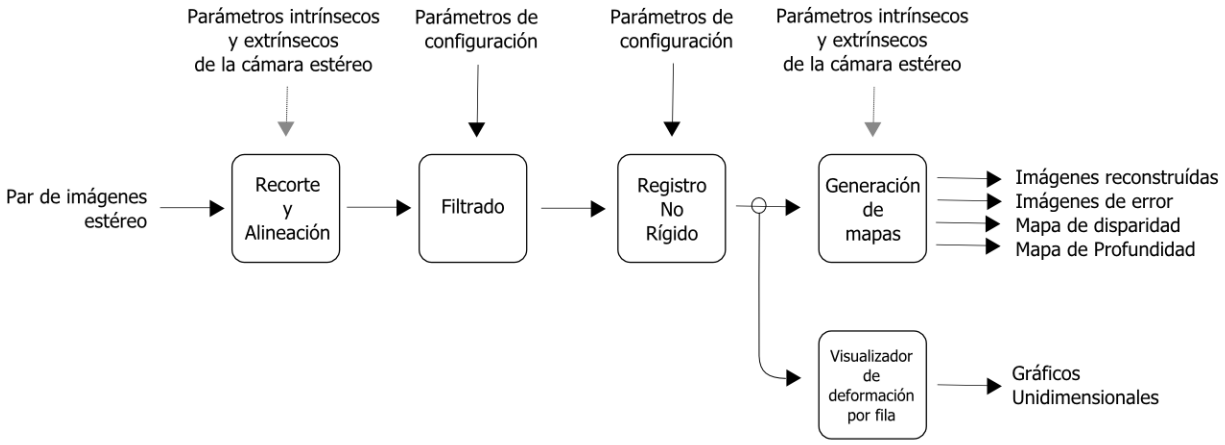

Figura 3.1: Diagrama de bloques del sistema.

Figura 3.2: Diagrama para módulo de calibración.

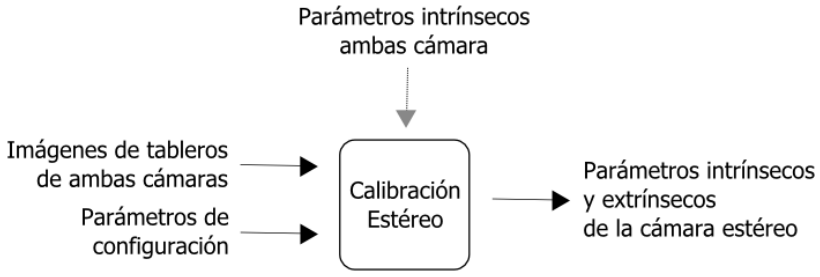

Figura 3.3: Diagrama para módulo de calibración estéreo.

Los distintos módulos hacen uso de distintas componentes de software, las que son citadas en 3.2. El eje principal del trabajo se refiere a la implementación del registro no rígido entre

las imágenes estéreo, por esta razón, para este módulo se desarrolla una sección completa detallando su implementación desde un punto de vista teórico (sección 3.4). Para el resto de los módulos, se hace uso de las funciones provistas por la librería OpenCV (y otras), por lo que no son tratados en profundidad, la teoría tras estas funciones ya se ha descrito en el marco teórico. Por último, en la sección 3.5, describe en detalle las entradas, salidas y funcionalidades de cada uno de los módulos del sistema completo.

3.2. Plataformas utilizadas

Para el desarrollo del presente trabajo se utilizaron varias plataformas de software libre, en particular:

- Lenguaje de programación C++, con las librerías OpenCV 2.3.0[1] y Boost 1.48.0[2].
- Lenguaje de programación Python 2.7, con las librerías Numpy y Matplotlib[3].
- IDE Eclipse Indigo, con los plugins CDT y PyDev[4].

Casi todo el sistema se implementa sobre C++, dejando solo el módulo de visualización por línea en Python/Matplotlib, dado la facilidad para generar gráficos. Todos los módulos en C++ son accesibles a través de un único programa ejecutable, el cual recibe como entrada un archivo con los distintos parámetros necesarios.

3.3. Limitaciones

Al utilizar dos imágenes provenientes desde distintas cámaras, a pesar de que correspondan a un mismo lugar y sean capturadas al mismo tiempo, pueden presentar considerables diferencias: distintas escalas de color, distintas reflexiones especulares, una imagen puede ser más borrosa que la otra, etc. En el presente trabajo se buscan correspondencias basándose exclusivamente en el color de los píxeles, por lo que se vuelve necesario asumir lo siguiente:

- Las imágenes solo presentan objetos opacos.
- Ambas imágenes presentan la misma escala de color.
- Las imágenes son enfocadas de igual forma.

[1] OpenCV (de *Open Computer Vision*) es una librería de visión artificial originalmente desarrollada por Intel.

[2] Boost es un conjunto de librerías preparadas para extender las capacidades del lenguaje de programación C++.

[3] Numpy y Matplotlib permiten trabajar en Python de manera similar a Matlab.

[4] Los plugins CDT y PyDev permiten utilizar Eclipse como IDE de programación para C++ y Python.

El no cumplimiento de estas condiciones puede producir correspondencias erróneas. Ahora, es posible sobrellevar estas dificultades añadiendo un pre-procesamiento de las imágenes, que se encargue, por ejemplo, de normalizar los histogramas de color o de aplicar algún algoritmo de compensación de iluminación.

Por otro lado, se considera un valor en escala de grises, este se calcula como el promedio simple de las 3 capas de color de la imagen. Esta restricción puede ser fácilmente eliminada, pues basta con calcular los funcionales para cada una de las tres capas, y luego sumarlos para obtener un funcional total. Utilizar un mayor número de capas de color proporciona mayor información, lo que en definitiva, generará mejores resultados, pero a un mayor costo de cómputo.

El trabajo se limita a efectuar las etapas de calibración utilizando funciones de la librería OpenCV 2.3.0. Se considerará el caso particular de las cámaras alineadas paralelamente y con el mismo plano de imagen. Se añade esta restricción para estimar de mejor forma los parámetros de calibración, pues resultados previos para configuraciones más generales no han sido satisfactorios. Esta configuración permite centrarse en el registro no rígido de las imágenes binoculares, independizándose de la etapa previa de calibración.

3.4. Implementación teórica de registro no rígido

Se tienen 2 imágenes estéreo, y se quiere encontrar correspondencias entre ambas imágenes. Como se mencionó en el marco teórico (sección 2.2.5), existen múltiples formas de abordar este problema. El presente trabajo interpreta las correspondencias como la deformación de una de las imágenes, en particular, la imagen derecha se deformará con tal de igualarse a la imagen izquierda. Por lo tanto, se debe calcular esta función de deformación, la que en particular será interpretada como un difeomorfismo.

3.4.1. Mecanismo de solución

Para imágenes tomadas desde cámaras perfectamente alineadas binocularmente, las correspondencias se encontrarán en las mismas filas de la imágenes, en otras palabras, solo se esperan variaciones horizontales. En este contexto, el problema es tratado línea a línea, considerándose entonces un enfoque unidimensional.

Acorde a la sección 2.3.2, se debe encontrar un campo vectorial v que garantice que la ecuación diferencial 2.72 genere un difeomorfismo que transforme una imagen en la otra. Lo que se logra formulando un funcional a minimizar (ecuación 2.77), el cual depende tanto de las imágenes estéreo, como del difeomorfismo utilizado. En esta línea, y si el funcional es correctamente definido, es posible utilizar el método del gradiente para encontrar el campo v (ecuación 2.78), y posteriormente calcular el difeomorfismo ϕ (ecuación 2.74). Este mecanismo de solución es llamado método de Euler de avance.

Se vuelve necesario definir una formulación de v que permita representar prácticamente cualquier curva. Como se trata de dos imágenes tomadas desde posiciones cercanas, se espera que las variaciones entre una y otra sean pequeñas, y con variaciones locales. En el problema

de imágenes estéreo, según la distancia a la que el objeto se encuentre desde las cámaras, su posición horizontal variará en mayor o menor medida. De esta forma, el fondo, debiera quedar prácticamente estático.

Para manejar curvas analíticas se dispone de varias técnicas, como la descomposición en series de Fourier y las B-splines. En particular, en este caso se escogen las B-splines gracias a sus propiedades de soporte compacto, por lo que para representar variaciones locales, solo se modificarán los coeficientes de la función B-spline asociados a esa zona, quedándose inmóvil el resto de los nodos.

$$v(a,x) = \sum_{i=0}^{N-1} a\,[i]\,\beta^n(x - x_i) \tag{3.1}$$

Se requiere fijar un orden n y una cantidad N de nodos para las B-splines a utilizar. En la medida de utilizar un mayor orden, se ensancha el soporte de las funciones base, pero se adquieren mejores condiciones de suavidad (acorde a lo discutido en la sección 2.4.1). Por otro lado, una mayor cantidad de nodos permitirá representar curvas de mayor complejidad, pero se añade costo computacional al tener que calcular una mayor cantidad de coeficientes. Para el presente trabajo, se utilizarán B-splines cúbicas, lo que acorde a la definición dada por las ecuaciones 2.81 y 2.82, equivale a utilizar $n = 4$. A su vez, se realizarán pruebas para distintas cantidades de nodos.

Como el campo v se define en términos de un conjunto de coeficientes $\{a\,[i]\}_{i=1..N}$, y luego ϕ se calcula en términos de v, por claridad, se notará esta dependencia en los argumentos de las funciones. De esta forma, se tiene que: $\phi = \phi\,(a, x, t)$ y $v = v\,(a, x)$.

Al definir el campo v en base a B-splines, la ecuación 2.78 se convierte en 3.2. En otras palabras, se deben actualizar iterativamente los coeficientes $a\,[i]$ que definen el campo. La expresión 3.2 involucra la derivada del funcional respecto de los coeficientes, por lo que es necesario calcular dicha derivada en cada caso.

$$a\,[i]^k = a\,[i]^{k-1} - \alpha \frac{\partial F^{k-1}}{\partial a[i]} \qquad \forall i \in \{0, ..., N-1\} \tag{3.2}$$

Primero se describen los funcionales a utilizar (sección 3.4.2), y luego se detallan los mecanismos de discretización y cálculo de sus gradientes (secciones 3.4.3 y 3.4.4). Como la formulación teórica puede ser un tanto compleja, el algoritmo de cómputo completo se sintetiza en 3.4.5.

3.4.2. Formulación de funcionales

Es necesaria la formulación de un buen funcional que permita determinar el campo v de forma correcta. Como se ha explicado en 2.3.3, este funcional puede ser descompuesto en similitud y regularización.

3.4.2.1. Funcionales de similitud

El funcional de similitud intenta minimizar el error existente entre la imagen original y la segunda imagen deformada, en este sentido, puede ser bien descrito basándose en la diferencia $I_1(x) - I_2(\phi(x))$. Por lo tanto, se adopta una formulación general dada por la ecuación 3.3, donde f corresponde a una función de penalización.

$$F_{sim} = \int_0^1 f\left(I_1(x) - I_2(\phi(a,x))\right) dx \tag{3.3}$$

Algunas posibilidades para f son las normas L_2 y L_1 (ecuaciones 3.4 y 3.5). Sin embargo, la norma L_1 no posee una derivada continua, por lo que puede no presentar un buen comportamiento al intentar optimizar utilizando el método del gradiente. De esta forma, se plantea un métrica alternativa, la cual es llamada L_{1A} o L_1 ajustada (ecuación 3.6), donde se ha incorporado un parámetro λ que suaviza el abrupto quiebre que presenta la norma L_1 en cero. Mientras mayor sea el parámetro λ, se tendrá una curva de mayor suavidad en torno a cero. La figura 3.4 presenta curvas características de estos tres funcionales.

$$f_{L2}\left(s\right) = s^2 \tag{3.4}$$

$$f_{L1}\left(s\right) = |s| \tag{3.5}$$

$$f_{L1A}\left(s\right) = \sqrt{s^2 + \lambda^2} \tag{3.6}$$

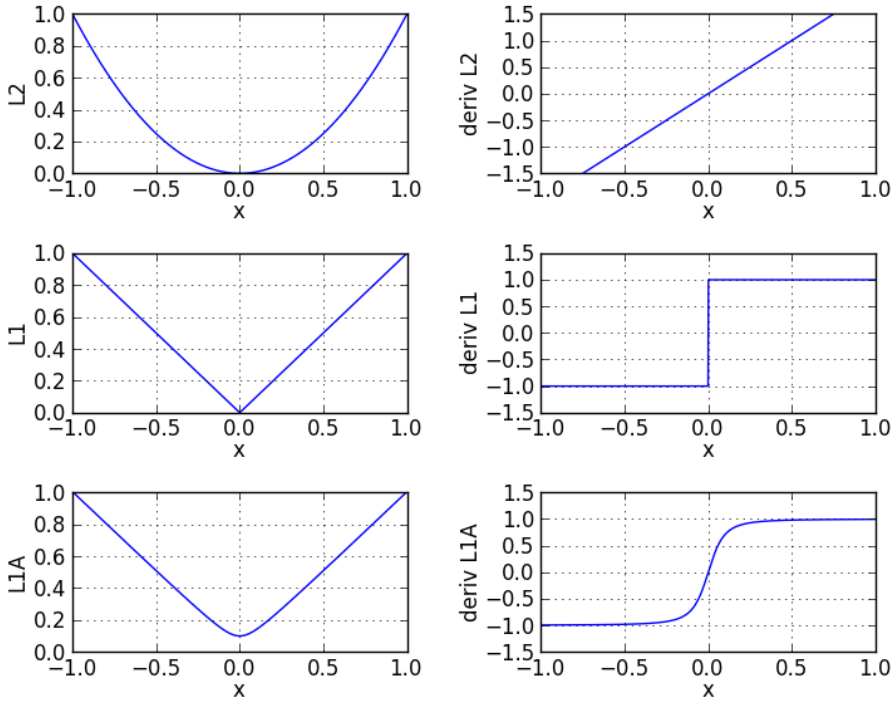

Figura 3.4: Funciones base de los funcionales de similitud y sus derivadas. ($\lambda = 0.1$)

La diferencia fundamental entre L_1 y L_2, es en cuanto se penalizan los valores lejanos a la tendencia general. En L_2, un valor lejano ejerce una gran fuerza sobre el valor resultante (ponderación cuadrática); mientras que en L_1 el valor influye, pero en menor medida que en L_2 (ponderación lineal). En este sentido, la norma L_2 castiga de manera importante los puntos lejanos, moviendo más rápidamente los coeficientes que definen la curva en esa sección. Las desventaja a priori, es que puede alterarse la tendencia general de la curva por casos puntuales, como simples ruidos o pequeñas oclusiones en las imágenes.

3.4.2.2. Funcionales de regularización

Con el fin de lograr el campo vectorial mínimo, un funcional de regularización posible es considerar la energía del mismo. En otras palabras, se buscan campos pequeños, por lo tanto, la deformación resultante también lo será.

$$F_{regE} = \int_0^1 \left(v\left(x\right)\right)^2 dx \tag{3.7}$$

Otro posible funcional es la energía del laplaciano del campo v, en este caso, se minimiza el valor de la segunda derivada, incentivándose solo pequeñas variaciones de la curvatura de la función. En este sentido, se opta por que las distintas partes de la curva sean lo más rectas posibles.

$$F_{regL} = \int_0^1 \left(\frac{d^2 v\,(x)}{dx^2} \right)^2 dx \tag{3.8}$$

3.4.3. Cálculo de los funcionales de similitud

Para los funcionales de similitud, se ha considerado una expresión base dada por la ecuación 3.3, con f una métrica de distancia; por lo tanto, su derivada queda genéricamente descrita por la ecuación 3.9. Donde, para el caso de los funcionales de similitud considerados, el valor de $\frac{\partial f(s)}{\partial s}$ queda dado por las expresiones 3.10, 3.11 ó 3.12.

$$\frac{\partial F_{sim}}{\partial a\,[i]} = \int_0^1 \left. \frac{df\,(s)}{ds} \right|_{s=I_1(x)-I_2(\phi(a,x))} \cdot \left(- \left. \frac{dI_2\,(y)}{dy} \right|_{y=\phi(a,x)} \right) \cdot \frac{\partial \phi\,(a,x)}{\partial a\,[i]} \cdot dx \tag{3.9}$$

$$\frac{df_{L2}\,(s)}{ds} = -2s \tag{3.10}$$

$$\frac{df_{L1}\,(s)}{ds} = sign\,(s) \tag{3.11}$$

$$\frac{df_{L1A}\,(s)}{ds} = \frac{s}{\sqrt{s^2 + \lambda^2}} \tag{3.12}$$

La evaluación de la derivada de los funcionales respecto de los coeficientes de la función B-spline v requiere del cálculo de la integral para todo x en el dominio. De esta forma, se requiere la evaluación en valores continuos de cada fila de la imagen (y de su derivada), requiriéndose una forma de interpolarlas. Existen múltiples técnicas que pueden realizar esta tarea. Las B-splines constituyen una buena alternativa, pues permiten, dependiendo del orden, establecer interpolaciones de rectángulos, lineal, cuadrática, cúbica, etc. Se reutiliza además la maquinaria construida para definir el campo v. El problema a considerar es determinar los coeficientes con tal de ajustarse a una señal equiespaceada discreta conocida.

Las B-splines con fines de interpolación han sido descritas en la sección 2.4.4, por los que los coeficientes se obtienen simplemente por filtrado inverso acorde a las ecuaciones recursivas 2.102 y supuestos para las condiciones de borde de extrapolación. En particular, se utilizarán B-splines de orden 4 y condiciones de borde del tipo *FSMC* (ver definición en sección 2.4.6). Sin perdida de generalidad, se considera $\Omega = [0, 1]$ y con X elementos equiespaciados. Los nodos de la B-spline interpoladora serán los mismos píxeles de la imagen normalizados en el intervalo $[0, 1]$. Del mismo modo se considera un intervalo de tiempo entre 0 y 1 con T elementos.

Teniendo las imágenes representadas por B-splines, como ya se ha visto en la sección 2.4.3, es fácil y rápido el cómputo tanto de la función misma como de su derivada (de cualquier orden). De esta forma, es posible calcular los términos $I_1(x)$, $I_2(\phi(a,x))$ y $\left. \frac{dI_2(y)}{dy} \right|_{y=\phi(a,x)}$ una vez que se conozca el valor de $\phi(a,x)$ en toda la discretización espacial utilizada. Luego,

queda pendiente el cálculo de $\phi(a,x)$ y de $\frac{\partial \phi(x)}{\partial a[i]}$, para cada x de la discretización, y para cada coeficiente $a\,[i]$ del campo vectorial, lo que es tratado en la sección 3.4.3.1.

Disponiendo de todos los elementos presentes en la expresión 3.9, la integral principal es resuelta a través del simple método de los rectángulos. Es decir, acorde a la expresión 3.13.

$$\frac{\partial F_{sim}}{\partial a\,[i]} = \frac{1}{X}\sum_{x=0}^{X}\left\{\left(\left.\frac{df\,(s)}{ds}\right|_{s=I_1(x)-I_2\left(\phi\left(a,\frac{x}{X}\right)\right)}\right)\cdot\left(-\left.\frac{dI_2\,(y)}{dy}\right|_{y=\phi\left(a,\frac{x}{X}\right)}\right)\cdot\frac{\partial\phi\left(a,\frac{x}{X}\right)}{\partial a\,[i]}\right\} \tag{3.13}$$

Si bien la evaluación final de las imágenes es discreta, el haberlas interpolado con B-splines permite calcular la integral con mayor o menor precisión en base a la discretización del dominio Ω considerada. A la vez, al disponer de curvas continuas para cada imagen, es posible efectuar distintos tipos de integraciones numéricas, como el método de los rectángulos, de los trapecios, o simpson. El caso óptimo sería aprovechar propiedades de las B-splines para efectuar un cálculo más eficiente, pero para priorizar tiempo de implementación, simplemente, se considera el método de los rectángulos en una discretización relativamente fina del dominio.

3.4.3.1. Cálculo del difeomorfismo y su derivada.

Para el difeomorfismo ya se dispone de la ecuación 2.74 para establecer el cálculo. Introduciendo la dependencia del conjunto de coeficientes a se obtiene la ecuación 3.14.

$$\phi\,(a,x,t) = x + \int_0^t v(a,\phi(a,x,\tau))d\tau \tag{3.14}$$

Discretizando el tiempo de forma que $t \in \{0,1,...,T-1\}$, se formula la siguiente recursión temporal.

$$\phi\left(a,x,\frac{t}{T}\right) = \phi\left(a,x,\frac{t-1}{T}\right) + \frac{1}{T}v\left(a,\phi\left(a,x,\frac{t-1}{T}\right)\right) \tag{3.15}$$

$$\phi\,(a,x,0) = x \tag{3.16}$$

Derivando la expresión 3.14 según el coeficiente $a\,[i]$.

$$\frac{\partial\phi\,(a,x,t)}{\partial a\,[i]} = \int_0^t \frac{\partial}{\partial a\,[i]}\left\{v\left(a,\phi\left(a,x,\tau\right)\right)\right\}d\tau \tag{3.17}$$

$$\frac{\partial}{\partial a\,[i]}\left\{v\left(a,\phi\left(a,x,t\right)\right)\right\} = \left.\frac{\partial v\,(a,x)}{\partial a\,[i]}\right|_{x=\phi(a,x,t)} + \left.\frac{\partial v\,(a,x)}{\partial x}\right|_{x=\phi(a,x,t)}\frac{\partial\phi\,(a,x,\tau)}{\partial a\,[i]} \tag{3.18}$$

$$\frac{\partial v\left(a,x\right)}{\partial a\left[i\right]}=\beta^{n}\left(x-x_{i}\right) \tag{3.19}$$

Luego, reemplazando.

$$\frac{\partial\phi\left(a,x,t\right)}{\partial a\left[i\right]}=\int_{0}^{t}\left(\beta^{n}\left(\phi\left(a,x,\tau\right)-x_{i}\right)+\left.\frac{\partial v\left(a,x\right)}{\partial x}\right|_{x=\phi(a,x,\tau)}\frac{\partial\phi\left(a,x,\tau\right)}{\partial a\left[i\right]}\right)d\tau \tag{3.20}$$

Discretizando de la misma forma que para ϕ se obtiene la ecuación 3.21, y finalmente, reagrupando términos, queda la ecuación 3.22 con la condición inicial dada en la expresión 3.23. Este último par de expresiones permite calcular por completo $\frac{\partial\phi}{\partial a[i]}$.

$$\frac{\partial\phi\left(a,x,\frac{t}{T}\right)}{\partial a\left[i\right]}=\frac{\partial\phi\left(a,x,\frac{t-1}{T}\right)}{\partial a\left[i\right]}+\frac{1}{T}\left(\beta^{n}\left(\phi\left(a,x,\frac{t-1}{T}\right)-x_{i}\right)+\left.\frac{\partial v\left(a,x\right)}{\partial x}\right|_{x=\phi\left(a,x,\frac{t-1}{T}\right)}\frac{\partial\phi\left(a,x,\frac{t-1}{T}\right)}{\partial a\left[i\right]}\right) \tag{3.21}$$

$$\frac{\partial\phi\left(a,x,\frac{t}{T}\right)}{\partial a\left[i\right]}=\left(1+\frac{1}{T}\left.\frac{\partial v\left(a,x\right)}{\partial x}\right|_{x=\phi\left(a,x,\frac{t-1}{T}\right)}\right)\frac{\partial\phi\left(a,x,\frac{t-1}{T}\right)}{\partial a\left[i\right]}+\frac{1}{T}\beta^{n}\left(\phi\left(a,x,\frac{t-1}{T}\right)-x_{i}\right) \tag{3.22}$$

$$\frac{\partial\phi\left(a,x,0\right)}{\partial a\left[i\right]}=0 \tag{3.23}$$

3.4.4. Cálculo de los funcionales de regularización

3.4.4.1. Regularización con energía del campo v

Primero considérese el caso de F_{regE}. Derivando respecto de $a\left[i\right]$ se obtiene la ecuación 3.24, donde se ha utilizado que $\frac{\partial v(x)}{\partial a[i]}=\beta^{n}\left(x\right)$.

$$\frac{\partial F_{regE}}{\partial a\left[i\right]}=2\int_{\Omega}v\left(a,x\right)\cdot\beta^{n}\left(x-x_{i}\right)dx \tag{3.24}$$

$$\frac{\partial F_{regE}}{\partial a\left[i\right]}=2\int_{\Omega}\sum_{k}a\left[k\right]\beta^{n}\left(x-x_{k}\right)\cdot\beta^{n}\left(x-x_{i}\right)dx \tag{3.25}$$

$$\frac{\partial F_{regE}}{\partial a\left[i\right]}=2\sum_{k}a\left[k\right]\int_{\Omega}\beta^{n}\left(x-x_{k}\right)\cdot\beta^{n}\left(x-x_{i}\right)dx \tag{3.26}$$

Para una función g par es posible demostrar la siguiente propiedad:

$$\int_{-\infty}^{\infty} f\left(x-y\right) g\left(x-z\right) dx = f * g\left(z-y\right) \tag{3.27}$$

Por lo tanto, dada la paridad de las bases B-spline, es posible formular 3.28 y luego 3.29.

$$\frac{\partial F_{regE}}{\partial a\left[i\right]} = 2\sum_{k} a\left[k\right] \beta^{n} * \beta^{n}\left(x_i - x_k\right) \tag{3.28}$$

$$\frac{\partial F_{regE}}{\partial a\left[i\right]} = 2\sum_{k} a\left[k\right] \beta^{2n}\left(x_i - x_k\right) \tag{3.29}$$

Considerando una discretización espacial equiespaciada, se tiene $x_i = \frac{i}{m}$ y $x_k = \frac{k}{m}$, con m la cantidad de elementos en que se divide el intervalo $[0, 1]$. Luego, como β^{2n} se está evaluando solo en valores múltiplos de $\frac{1}{m}$, puede ser intercambiada por b_m^{2n}. Por lo tanto, la expresión resultante se reduce a una convolución discreta entre los coeficientes del campo vectorial y la B-spline discreta del doble del orden.

$$\frac{\partial F_{regE}}{\partial a\left[i\right]} = 2\sum_{k} a\left[k\right] b_m^{2n}\left[i-k\right] \tag{3.30}$$

$$\frac{\partial F_{regE}}{\partial a\left[i\right]} = 2\, a * b_m^{2n}\left[i\right] \tag{3.31}$$

3.4.4.2. Regularización con energía del laplaciano del campo v

Ahora, considerando el caso del funcional F_{regL}. Primero nótese que el laplaciano unidimensional del campo v queda dado por la expresión 3.33, y que la derivada del campo v respecto del coeficiente $a\left[i\right]$ se reduce a evaluar una única base B-spline (ecuación 3.34).

$$\frac{\partial^2 v\left(a, x\right)}{\partial x^2} = \sum_{k} a\left[k\right] \frac{d^2}{dx^2} \beta^{n}\left(x - x_k\right) \tag{3.32}$$

$$\frac{\partial^2 v\left(a, x\right)}{\partial x^2} = \sum_{k} a\left[k\right] \cdot d_2 * \beta^{n}\left(x - x_k\right) \tag{3.33}$$

$$\frac{\partial v\left(a, x\right)}{\partial a\left[i\right]} = \beta^{n}\left(x - x_i\right) \tag{3.34}$$

Donde d_2 es el laplaciano discreto según la ecuación 2.87. Luego, derivando 3.8 respecto de $a\left[i\right]$ y reemplazando los términos conocidos.

$$\frac{\partial F_{regL}}{\partial a\,[i]} = 2\int_{\Omega} \frac{\partial^2 v\,(a,x)}{\partial x^2} \cdot \frac{\partial}{\partial a\,[i]} \left\{ \frac{\partial^2 v\,(a,x)}{\partial x^2} \right\} dx \tag{3.35}$$

$$\frac{\partial F_{regL}}{\partial a\,[i]} = 2\int_{\Omega} \frac{\partial^2 v\,(a,x)}{\partial x^2} \cdot \frac{\partial^2}{\partial x^2} \left\{ \frac{\partial v\,(a,x)}{\partial a\,[i]} \right\} dx \tag{3.36}$$

$$\frac{\partial F_{regL}}{\partial a\,[i]} = 2\int_{\Omega} \sum_k a\,[k] \cdot d_2 * \beta^n\,(x - x_k) \cdot \frac{d^2}{dx^2} \left\{ \beta^n\,(x - x_i) \right\} dx \tag{3.37}$$

$$\frac{\partial F_{regL}}{\partial a\,[i]} = 2\sum_k a\,[k] \cdot \int_{\Omega} d_2 * \beta^{n-2}\,(x - x_k) \cdot d_2 * \beta^{n-2}\,(x - x_i)\, dx \tag{3.38}$$

Haciendo uso de la propiedad definida en la ecuación 3.27.

$$\frac{\partial F_{regL}}{\partial a\,[i]} = 2\sum_k a\,[k] \cdot \left(d_2 * \beta^{n-2}\right) * \left(d_2 * \beta^{n-2}\right)(x_i - x_k) \tag{3.39}$$

$$\frac{\partial F_{regL}}{\partial a\,[i]} = 2\sum_k a\,[k] \cdot d_2 * d_2 * \beta^{2(n-2)}\,(x_i - x_k) \tag{3.40}$$

Al igual que en el caso anterior, se considera $x_i = \frac{i}{m}$ y $x_k = \frac{k}{m}$, y se reemplaza la B-spline por su versión discreta.

$$\frac{\partial F_{regL}}{\partial a\,[i]} = 2\sum_k a\,[k] \cdot d_2 * d_2 * b_m^{2(n-2)}\,[i-k] \tag{3.41}$$

Lo que no es otra cosa que un producto de convolución discreto, lográndose formular la expresión 3.42. Notar que la señal discreta $d_2 * d_2 * b_m^{2(n-2)}$ puede ser calculada una única vez, traduciéndose el proceso de cómputo en una única convolución con los coeficientes del campo vectorial.

$$\frac{\partial F_{regL}}{\partial a\,[i]} = 2\,a * d_2 * d_2 * b_m^{2(n-2)}\,[i] \tag{3.42}$$

La gran ventaja de las expresiones 3.31 y 3.42 es que calculan el valor exacto para $\frac{\partial F_{reg}}{\partial a[i]}$, de forma rápida (una simple convolución) y sin requerir una discretización espacial.

3.4.5. Algoritmo

El algoritmo 3.1 sintetiza la implementación a realizar.

Algoritmo 3.1 Cálculo del difeomorfismo.

ENTRADAS: Imágenes I_1 e I_2 (unidimensionales), parámetros λ, α_E y α_L.
SALIDAS: Difeomorfismo ϕ.

1. Etapa de inicialización.

   Interpolar I_1 e I_2.

   $k = 0$

   $a\,[i]^0 = 0$

2. Cálculo del gradiente del funcional.

   - Cálculo de F^k_{sim} y $\frac{\partial F^k_{sim}}{\partial a[i]}$ utilizando algoritmo 3.2 con a^k, λ, I_1 e I_2.
   - Cálculo de $\frac{\partial F^k_{reg}}{\partial a[i]}$ utilizando algoritmo 3.3 con a^k, α_E y α_L.
   - $\frac{\partial F^k}{\partial a[i]} = \frac{\partial F^k_{sim}}{\partial a[i]} + \frac{\partial F^k_{reg}}{\partial a[i]}$

3. Actualizar valores.

   $k = k + 1$

   $a\,[i]^k = a\,[i]^{k-1} - \alpha \frac{\partial F^{k-1}}{\partial a[i]}$

4. Evaluar $\phi\,(a, x, t) \quad \forall x \in \Omega, \forall t \in \{1, ..., T\}$.

   $\phi\,(a, x, 0) = x$

   $\phi\left(a, x, \frac{t}{T}\right) = \phi\left(a, x, \frac{t-1}{T}\right) + \frac{1}{T} v\left(a, \phi\left(a, x, \frac{t-1}{T}\right)\right)$

5. Evaluar criterios de detención y, si corresponde, volver a paso 2.

Algoritmo 3.2 Cálculo del funcional de similitud y su gradiente.

ENTRADAS: Imágenes I_1 e I_2 (unidimensionales), coeficientes $a[i] \quad \forall i \in \{0, ..., N-1\}$
SALIDAS: Funcional de similitud F_{sim} y su gradiente $\frac{\partial F_{sim}}{\partial a[i]} \quad \forall i \in \{0, ..., N-1\}$.

1. Etapa de inicialización.. Ejecutar $\forall x \in \Omega, \forall i \in \{0, ..., N-1\}$.

   $\phi(a, x, t=0) = x$

   $\frac{\partial \phi}{\partial a[i]}(a, x, t=0) = 0$

2. Cálculo de ϕ y $\frac{\partial \phi}{\partial a[i]}$. Ejecutar $\forall t \in \{1, ..., T\}, \forall x \in \Omega, \forall i \in \{0, ..., N-1\}$.

   $\phi(a, x, \frac{t}{T}) = \phi(a, x, \frac{t-1}{T}) + \frac{1}{T} v(a, \phi(a, x, \frac{t-1}{T}))$

   $\frac{\partial \phi(x, \frac{t}{T})}{\partial a[i]} = \left(1 + \frac{1}{T} \frac{\partial v}{\partial x}(a, x)\big|_{x=\phi(a, x, \frac{t-1}{T})}\right) \frac{\partial \phi}{\partial a[i]}\left(a, x, \frac{t-1}{T}\right) + \frac{1}{T} \beta^n \left(\phi\left(x, \frac{t-1}{T}\right) - x_i\right)$

3. Cálculo del funcional

   $$F_{sim} = \int_0^1 f\left(I_1(x) - I_2(\phi(a, x, 1))\right) dx$$

   - Si se usa norma L_2, entonces $f(s) = s^2$
   - Si se usa norma L_1, entonces $f(s) = |s|$
   - Si se usa métrica L_{1A}, entonces $f(s) = \sqrt{s^2 + \lambda^2}$

4. Cálculo del gradiente del funcional. $\forall i \in \{0, ..., N-1\}$.

   $$\frac{\partial F_{sim}}{\partial a[i]} = -\int_0^1 \frac{df(s)}{ds}\bigg|_{s=I_1(x)-I_2(\phi(a,x,1))} \cdot \frac{dI_2(y)}{dy}|_{y=\phi(a,x,1)} \cdot \frac{\partial \phi}{\partial a[i]}(a, x, 1) \cdot dx$$

   - Si se usa norma L_2, entonces $\frac{df(s)}{ds} = 2s$
   - Si se usa norma L_1, entonces $\frac{df(s)}{ds} = sign(s)$
   - Si se usa métrica L_{1A}, entonces $\frac{df(s)}{ds} = \frac{s}{\sqrt{s^2+\lambda^2}}$

Algoritmo 3.3 Cálculo del gradiente del funcional de regularización.

ENTRADAS: Coeficientes $a[i] \quad \forall i \in \{0, ..., N-1\}$, ponderadores α_E y α_L
SALIDAS: Gradiente del funcional de regularización $\frac{\partial F_{reg}}{\partial a[i]} \quad \forall i \in \{0, ..., N-1\}$.

1. Inicializar.
   $\frac{\partial F_{reg}}{\partial a[i]} = 0$
2. En caso de utilizar F_{regE}. $\forall i \in \{0, ..., N-1\}$.
   $\frac{\partial F_{reg}}{\partial a[i]} = \frac{\partial F_{reg}}{\partial a[i]} + \alpha_E \cdot a * b_m^{2n}[i]$
3. En caso de utilizar F_{regL}. $\forall i \in \{0, ..., N-1\}$.
   $\frac{\partial F_{reg}}{\partial a[i]} = \frac{\partial F_{reg}}{\partial a[i]} + \alpha_L \cdot a * d_2 * d_2 * b_m^{2(n-2)}[i]$

3.5. Descripción de módulos

3.5.1. Calibración

Como se ha descrito en la sección 2.1.2, calibración se refiere al proceso de obtener los parámetros intrínsecos de una cámara, esto es, la matriz de cámara $\mathbf{M}$ (ecuación 2.5) y el vector de coeficientes de distorsión $\vec{D}$ (ecuación 2.13). Para esta tarea, se utiliza la librería OpenCV, la cual incorpora una funciones especiales para calibrar y obtener estos parámetros. La implementación se limita a utilizar y dar la interfaz apropiada a estas funciones, pues el énfasis y núcleo del trabajo se encuentra en la implementación del registro no rígido.

El módulo requiere de una serie de imágenes de un patrón de calibración tipo tablero de ajedrez, cuyos nombres deben estar registrados en un archivo de texto. Es necesario indicar las dimensiones del tablero utilizado, las dimensiones de las imágenes y los parámetros que requiere la función de OpenCV. Si las imágenes poseen distinta resolución que la especificada como argumento, estas son re-dimensionadas de forma previa. Como resultado, se genera un archivo con los valores de $\mathbf{M}$ y $\vec{D}$.

3.5.2. Calibración estéreo

Similar al módulo de calibración, se utiliza OpenCV para realizar las tareas de calibración y rectificación estéreo. En este caso, se requiere de las listas con las imágenes del patrón de calibración asociadas a cada cámara; así como también las dimensiones del tablero; y la resolución de las imágenes utilizadas. Al igual que en el caso anterior, si las imágenes poseen distinta resolución que la especificada como argumento, estas son re-dimensionadas de forma previa.

Las imágenes de los tableros deben ser correspondientes, es decir, la n-ésima imagen de cada lista, se asocia a la misma pose del tablero vista por ambas cámaras. Si por diversas

razones el tablero no es detectado en alguna de las vistas, el par de imágenes es simplemente ignorado.

El módulo dispone además de la posibilidad de utilizar datos de una calibración previa, en particular, de las matrices de cámara y vectores de distorsión. Esta información puede utilizarse como punto de partida para un nuevo cómputo de estos parámetros, en teoría, facilitando la optimización; o también, de forma fija, con tal de calcular solo el resto de los parámetros de calibración y rectificación estéreo.

3.5.3. Recorte y alineación

Sólo una región rectangular de las imágenes rectificadas es útil, pues los bordes se aprecian deformados, dejando zonas son definir. Este módulo resuelve el problema pues recorta las imágenes rectificadas preparándolas para el módulo de registro.

En particular, el registro implementado necesita imágenes del mismo tamaño, por esta razón, el software da la posibilidad de recortarlas con un único rectángulo superpuesto a ambas imágenes. Simultáneamente con la tarea de recorte, el módulo también permite trasladar una imagen sobre la otra, de forma de alinear los fondos. El objetivo de este proceso es eliminar la traslación simple que inevitablemente existe entre ambas imágenes, para que el cómputo del difeomorfismo (el cual es considerablemente costoso), se centre en las diferencias no lineales.

Otro tema a considerar son los bordes de la imagen. Por construcción, el algoritmo de registro asume que los bordes de las imágenes son iguales, imponiendo un cero en los extremos del campo v de de deformación. Es claro que por razones de perspectiva y distancia entre ambas cámaras, en general esto no será así. En este sentido, este módulo viene a dar el paso necesario para validar los supuestos del difeomorfismo que se pretende encontrar. Si bien esta etapa no es automática, constituye una pequeña ayuda para lograr mejores resultados con el algoritmo de cómputo de la deformación.

De esta forma, el módulo de recorte y alineación, para dos imágenes de entrada cualquiera, y junto con los parámetros de calibración/rectificación estéreo, permite generar dos imágenes de igual tamaño, con los fondos alineados, y de ser posible, con bordes idénticos. La traslación inherente entre ambas imágenes es de importancia para la posterior reconstrucción tridimensional, por esta razón, es guardada en un archivo.

3.5.4. Filtrado de las imágenes

En vista de que el algoritmo de registro no rígido es costoso computacionalmente, este módulo viene a simplificar las imágenes para mejorar el proceso de registro. A través de argumentos, el programa puede aplicar o no un filtrado pasa-bajos y una reducción vertical.

La aplicación de filtrado pasa bajos es natural para simplificar las curvas asociadas a cada línea, con este simple paso, se pretende eliminar ruido y pequeñas oclusiones que puedan poseer las imágenes. Por simplicidad, se utiliza un mismo filtro caja una determinada cantidad de veces. Tanto las dimensiones del filtro, como la cantidad de veces que se utiliza, constituyen parámetros de entrada al módulo.

Si las imágenes se encuentran levemente trasladadas verticalmente, se intentará encontrar correspondencias entre líneas erróneas. Para atacar este problema, las imágenes son reducidas verticalmente. Esto consiste en que cada línea de las imágenes finales será el promedio de N líneas consecutivas de las imágenes originales. Donde N es un valor ingresado por el usuario. Esto reduce el número de filas, por lo que en definitiva, se disminuye el tiempo que tardará procesar las imágenes completas. Los resultados, pueden ser interpolados para las líneas restantes.

3.5.5. Registro no rígido

Este módulo encapsula la parte fundamental del presente trabajo, pues resuelve la ecuación de difeomorfismo para cada fila de la imagen. Es importante el uso de una estructura de B-splines, la cual se utiliza tanto para interpolar las filas de las imágenes, como para expresar el campo vectorial incógnito del problema. El algoritmo de solución corresponde al descrito en detalle en la sección 3.4.

El software implementado puede utilizarse para calcular tanto el registro completo de las imágenes, como para solo una parte de ellas, incluyéndose la opción de una única línea. El cómputo de la deformación es considerablemente lento, por esta razón se optó por ir guardando los resultados en archivos separados fila por fila. De esta forma, se tiene un respaldo en caso de necesitar interrumpir proceso. Del mismo modo, esta opción es útil para realizar el registro de la imagen completa en varias etapas.

Como se ha mencionado, el algoritmo de cómputo de deformación está pensado e implementado en una dimensión, pensando en que las imágenes solo presentarán desplazamientos horizontales. En este sentido, el registro no rígido se realiza independientemente fila por fila. Ahora, intuitivamente, es de esperar que la imagen, en la mayoría de los casos, no varíe tanto verticalmente. Por esta razón, los coeficientes calculados en la fila previa debieran constituir un buen punto de partida para el algoritmo de optimización.

El módulo toma como parámetros de configuración: el paso de iteración; la discretizaciones temporal y espacial; la región de las imágenes a procesar; la cantidad de nodos con los que se definirá el campo v; el nombre de los archivos donde se guardará el registro; si se utiliza algún registro anterior como punto de partida; y finalmente, el criterio de detención.

Es posible que al utilizar un paso de iteración del método del gradiente, la solución se aleje del mínimo global, en este sentido, se observará un aumento sostenido del valor del funcional. En caso de detectar este aumento del funcional, el valor del paso de iteración es disminuido a la mitad. Esta característica puede ser habilitada o no y se permite un número máximo de adaptaciones especificada por el usuario.

La implementación realizada define 3 criterios de parada para el método del gradiente, estos son:

- Por número máximo de iteraciones, constituyendo la causal definitiva de término para conseguir resultados en un tiempo finito.
- Por convergencia del funcional, esto es, la variación relativa del funcional es demasiado

pequeña para seguir siendo considerada.

- Número máximo de adaptaciones del paso de iteración.

Los dos últimos criterios de parada pueden ser o no considerados dependiendo de los argumentos dados al programa. El criterio de un número máximo de iteraciones, podría eventualmente ser único y suficiente, sin embargo, no existe forma de conocer a priori cuantas iteraciones se necesitan para conseguir un buen registro. Por esta razón, en virtud de conseguir mejores resultados, habría que dar un límite alto.

Por otro lado, el criterio de convergencia del funcional viene a ser mandatorio. Si el funcional converge, se ha encontrado una solución al problema. Esto puede ocurrir en cualquier momento, es decir, antes o después del número máximo de iteraciones. De esta forma, en caso de converger antes, se ha logrado un ahorro en tiempo de cómputo; por otro lado, si el algoritmo frena por número de iteraciones, se imprime una advertencia de que la deformación calculada es incompleta. En un sentido estricto, la optimización sólo debiera parar por convergencia, pero de ser así, no se tendría una limitante máximo de tiempo. Ahora, si el funcional converge, no necesariamente corresponde al mínimo global, pudiendo haber alcanzado alguna localidad, por lo que la solución encontrada puede ser errónea.

El módulo trabaja con unidades normalizadas, esto es, las imágenes tienen tanto ancho unitario, como un valor normalizado en su escala de grises. El negro se asocia al valor 0.0, y el blanco al 1.0. Como el ancho de la imagen es unitario, la función de deformación queda también normalizada; y por lo tanto, sus valores pueden interpretarse como desplazamientos porcentuales, siendo fácil y directa la conversión a píxeles.

Como salidas del módulo se tiene el registro de cada una de las filas consideradas. Esto es: el valor del difeomorfismo ϕ asociado a cada píxel de la imagen derecha; el valor del coeficiente de cada nodo que caracterice al campo vectorial v; una reconstrucción de v para cada píxel; las imágenes (1D) originales consideradas; y por último, la imagen derecha deformada.

3.5.6. Generación de mapa de profundidad

Como se ha explicado en el marco teórico, la generación del mapa de profundidad es simple y natural una vez obtenido el mapa de disparidad (sección 2.2.6). El módulo de registro no rígido genera la deformación ϕ necesaria para transformar la segunda imagen en la primera.

Para calcular el desplazamiento horizontal (o disparidad), basta con calcular la diferencia $\phi(x) - x$ para cada fila de las imágenes completas. El módulo de registro trabaja las imágenes con un ancho normalizado, por lo que se requiere escalar esta diferencia en un factor s igual al ancho de las imágenes para volver a las unidades de píxeles. Por otro lado, las imágenes fueron trasladadas de forma previa al registro, por lo que es necesario recoger esta información y efectuar el procedimiento inverso, y así contar con el mapa de disparidad real entre las imágenes. Se añade por tanto el *offset* de la traslación inicial t_x a la diferencia recién calculada.

$$d = t_x + s \cdot (\phi(x) - x) \tag{3.43}$$

Teniendo el mapa de disparidad correcto, solo resta proyectar cada píxel al espacio tridimensional utilizando la matriz de re-proyección **Q**, la cual ya fue calculada en el módulo de calibración y rectificación estéreo.

Cuando el desplazamiento horizontal es nulo (por ejemplo en el fondo), el punto tridimensional asignado queda en el infinito, de esta forma, no es posible visualizar el mapa de profundidad en todo el rango. Para obtener una visualización, se añade entonces un parámetro de distancia máxima a considerar, los valores obtenidos mas allá de este límite, se considera que pertenecen a ese plano. Vale recordar que las posiciones de objetos muy distantes poseen menor precisión (ver sección 2.2.1) por lo que establecer un límite máximo no actúa como restricción a las capacidades del sistema implementado.

La salida de este módulo considera los mapas de disparidad y de profundidad, ambos en unidades métricas y en unidades de píxeles. También se reconstruyen las imágenes originales, la segunda imagen deformada, e imágenes del error. Los mapas en unidades métricas son almacenados en un archivos de texto.

Cabe destacar que las unidades métricas quedan definidas en la etapa de calibración, al establecer la arista de cada cuadrado del tablero de ajedrez utilizado. En otras palabras, si la arista del cuadrado mide 1 cm, y al módulo de calibración se le entrega 1, todas las unidades métricas quedarán en centímetros, por otro lado, si se le entrega 10, las unidades quedarán en milímetros.

Capítulo 4

Resultados Experimentales

El sistema completo genera una gran cantidad de resultados, los que pueden ser agrupados de la siguiente forma:

- Resultados de calibración simple
- Resultados de calibración y rectificación estéreo
- Registro no rígido por línea
- Registro no rígido de imagen completa
- Generación de mapa de profundidad

Dado que los módulos de calibración son implementados sobre funciones provistas por OpenCV, la presentación de resultados es más bien reducida. El problema de registro es resuelto fila a fila de la imagen, por tanto, basta con observar una línea para analizar y determinar el desempeño del algoritmo frente a distintos parámetros. En este sentido, en una primera oportunidad se presentan gráficas para líneas claves de distintas imágenes; y luego, se presentan imágenes completamente procesadas. Finalmente, se presentan los resultados de generación de mapas de profundidad.

Las pruebas se realizan fundamentalmente sobre 4 imágenes: *Tsukuba*, *Venus*, *Cones* y *Mouse*. Las 3 primeras provienen de la base de datos de Middlebury [18, 17, 16, 12] y se presentan en la figura 4.1. El par de imágenes *Mouse* son capturadas utilizando una cámara 3D Fujifilm, el modelo particular es *FinePix REAL 3D W1*. La imágenes tomadas con la cámara 3D permiten validar la rectificación y generalizar a imágenes arbitrarias.

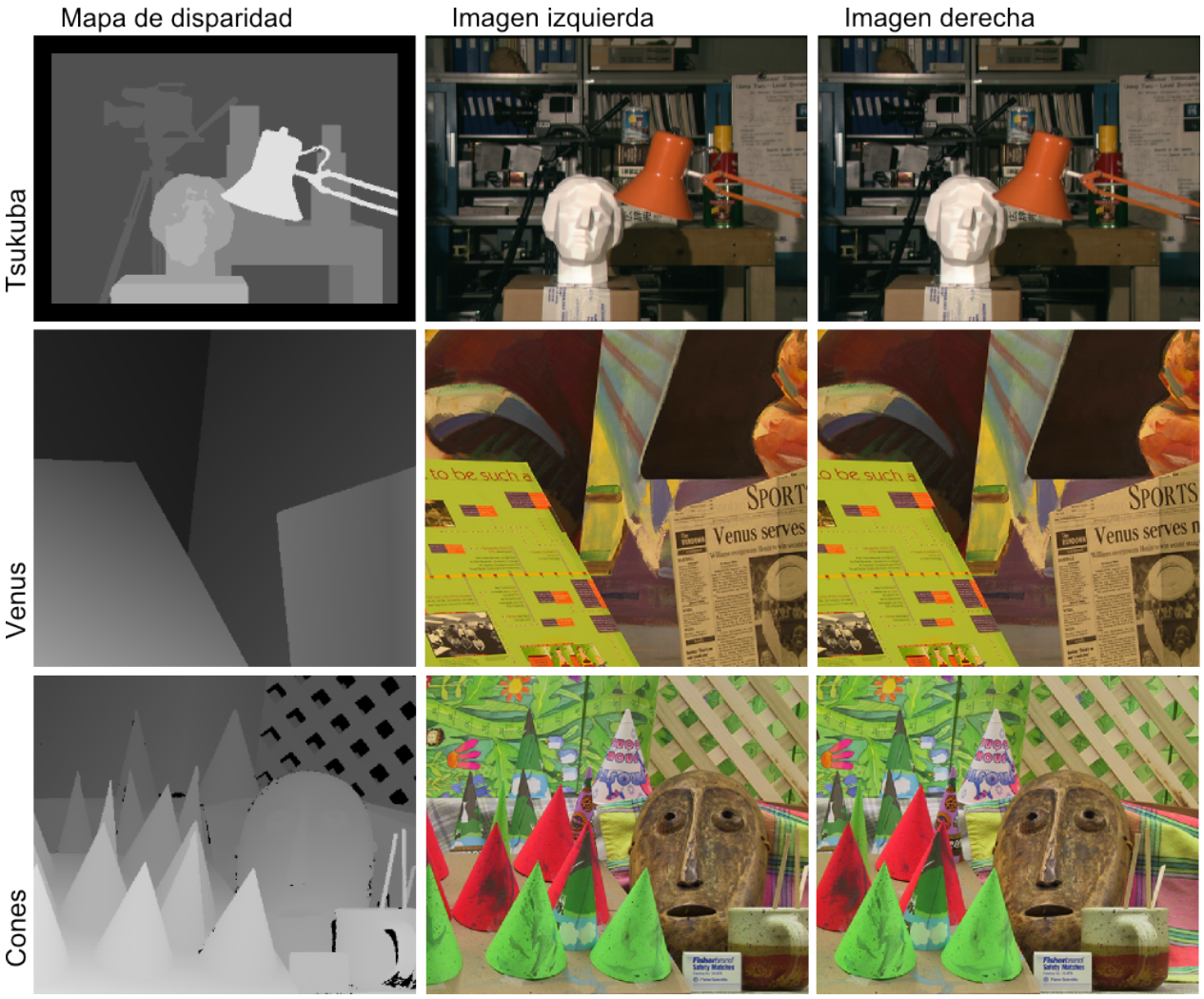

Figura 4.1: Pares de imágenes estéreo y mapa de disparidad asociado.

4.1. Calibración de una cámara

4.1.1. Imágenes desde cámara 3D

Utilizando la cámara 3D, se calibran independientemente las lentes izquierda y derecha, usando como patrón un tablero de ajedrez de 9×7 cuadros, cuya arista es de 30 mm. Aplicando la calibración directamente sobre las imágenes originales de 3584×2016 píxeles, se obtienen los parámetros indicados en la tabla 4.1. Para facilitar la convergencia del algoritmo, se asume que no hay distorsión tangencial, y que el eje principal se encuentra en el centro de las imágenes.

	Cámara izquierda	Cámara derecha
f_x	4205.97	4219.39
f_y	4235.83	4246.12
c_x	1791.5	1791.5
c_y	1007.5	1007.5
k_1	-0.076100	-0.052379
k_2	0.047717	0.116038
Error	1.46164	1.58864

Tabla 4.1: Parámetros de calibración cámara 3D Fujifilm.

Por supuesto, procesar imágenes tan grandes requeriría un tiempo excesivamente elevado en la etapa de registro. Por esta razón, las imágenes son re-escaladas a una resolución de 450×253 píxeles. Tomando estas imágenes, se obtienen los parámetros vistos en la tabla 4.2. La figura 4.2 presenta el par de vistas asociados a un tablero una vez corregida la distorsión. En la figura se observa que, si bien se efectúa una corrección sobre las imágenes, estas son prácticamente despreciables, lo que da cuenta de la buena calidad de las cámaras utilizadas.

	Cámara izquierda	Cámara derecha
f_x	528.00	531.07
f_y	531.53	534.04
c_x	224.5	224.5
c_y	126	126
k_1	-0.071381	-0.036360
k_2	0.007077	-0.021522
Error	0.187093	0.289923

Tabla 4.2: Parámetros de calibración cámara 3D Fujifilm con imágenes re-escaladas.

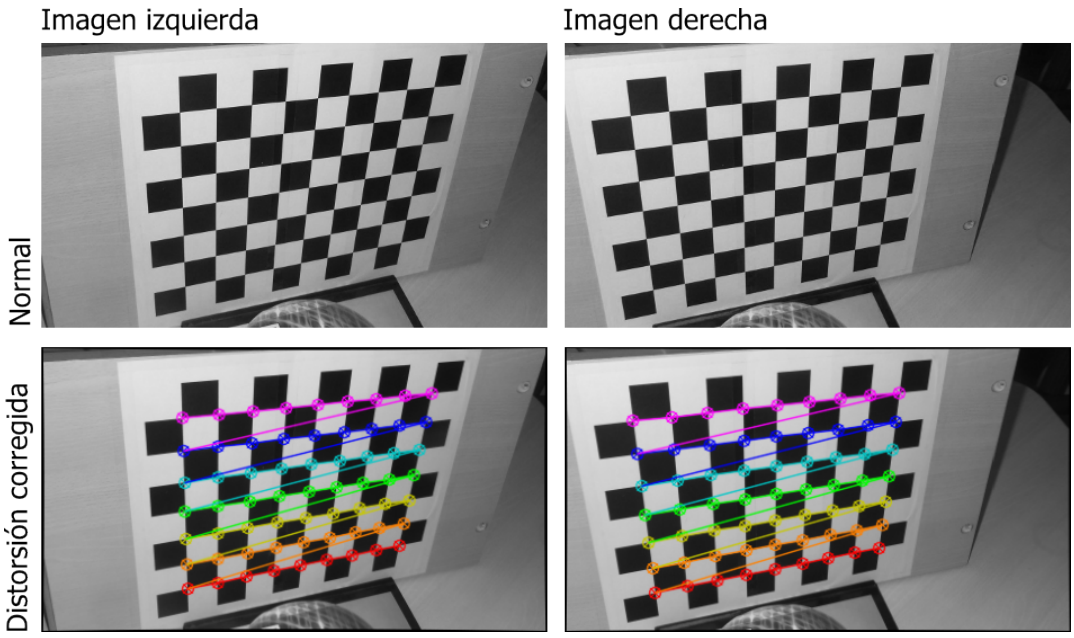

Figura 4.2: Distorsión corregida en imágenes de cámara 3D Fujifilm.

La medida de error presente en las tablas 4.1 y 4.2 corresponde al error cuadrático medio calculado gracias a las esquinas interiores de los tableros. Se utiliza como referencia la posición real de la esquina, y como valor asociado al modelo, la proyección de la misma esquina utilizando las transformaciones estimadas con la calibración. Al tener una buena estimación de los parámetros, ambas posiciones en píxeles debieran ser similares, por lo que la diferencia debiese ser pequeña. Es claro que al utilizar imágenes de mayor resolución, la medida de error aumentará, pues la distancia en píxeles aumenta, lo que se ve reflejado en estos resultados.

4.1.2. Ejemplos de OpenCV

Con fines de prueba del módulo, también se ejecuta la calibración utilizando las imágenes provistas como ejemplos de OpenCV. Estas imágenes poseen una resolución de 640×480 píxeles y son procesadas directamente. Los valores obtenidos en este caso se presentan en la tabla 4.3. De forma análoga al caso anterior, la figura 4.3 presenta un par de tableros con la distorsión corregida.

	Cámara izquierda	Cámara derecha
f_x	540.08	542.37
f_y	540.89	542.36
c_x	319.5	319.5
c_y	239.5	239.5
k_1	-0.295377	-0.281413
k_2	0.122866	0.087632
Error	0.490902	0.455966

Tabla 4.3: Parámetros de calibración para imágenes de ejemplo de OpenCV.

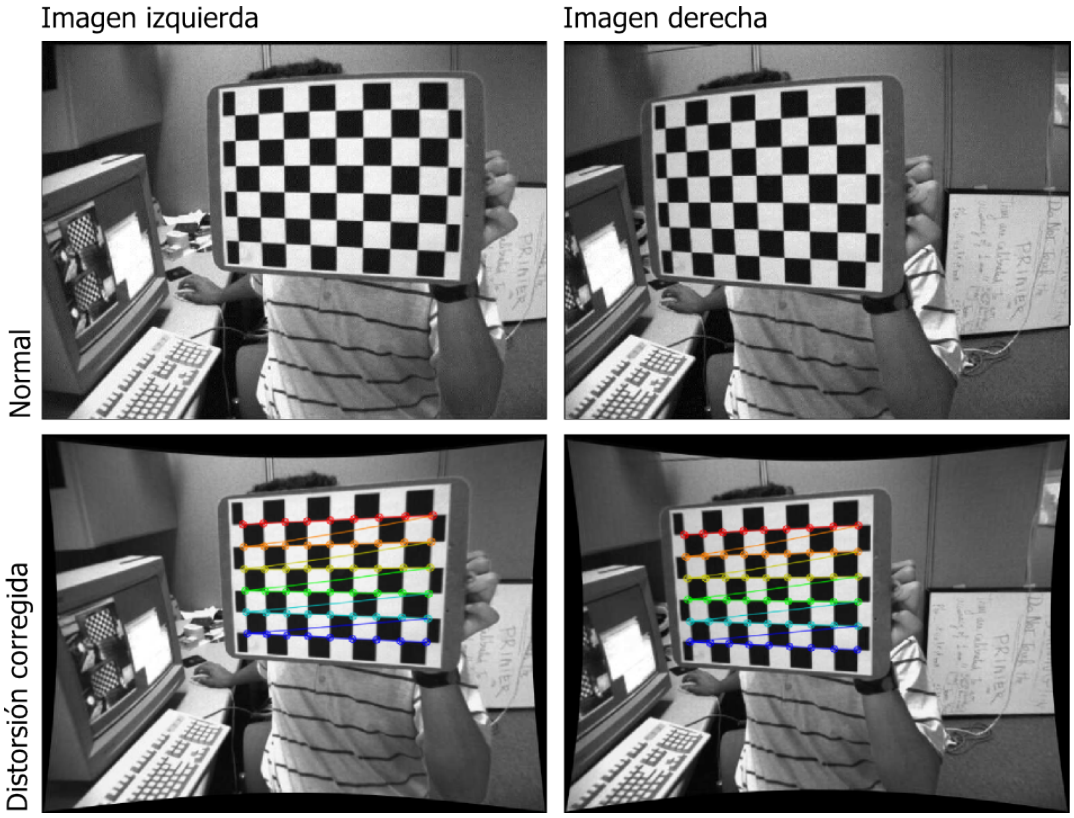

Figura 4.3: Distorsión corregida en imágenes de ejemplo de OpenCV.

Se nota que las imágenes de prueba de OpenCV presentan una distorsión considerable, a diferencia de las obtenidas con la cámara 3D. Este efecto se aprecia tanto en las tablas de parámetros intrínsecos, como en las mismas figuras presentadas. Luego, para la cámara 3D, una aproximación relativamente buena es considerar que las imágenes no presentan distorsión. Por otro lado, es claro que las distancias focales horizontal y vertical (f_x y f_y) dependen de la resolución de las imágenes consideradas, pues corresponden a los factores que en definitiva convertirán desde píxeles a unidades métricas.

Las medidas de error en este caso se encuentran entre los valores estimados en las tablas 4.1 y 4.2, lo que concuerda con lo esperado, pues la resolución utilizada es justamente menor a la asociada con la tabla 4.1 y mayor a la asociada con la tabla 4.2. Luego, dados los resultados obtenidos, el módulo de calibración de una única cámara valida su correcto funcionamiento.

4.2. Calibración estéreo y rectificación

Al igual que el módulo de calibración de cámaras independiente, en este módulo se realizan pruebas sobre las imágenes de ejemplo provistas por OpenCV, y sobre imágenes capturadas por la cámara 3D. Conviene mencionar, que ambos conjuntos de imágenes han sido capturadas desde cámaras aproximadamente paralelas, por lo que los parámetros calculados deben reflejar este hecho (ver sección 2.2.2.5).

4.2.1. Ejemplos de OpenCV

La figura 4.4 muestra las imágenes de ejemplo de OpenCV rectificadas, donde como es de esperar, se observa que las líneas de los tableros de ajedrez se vuelven paralelas. De esta forma, se observa que se han estimado correctamente los parámetros de calibración estéreo y rectificación.

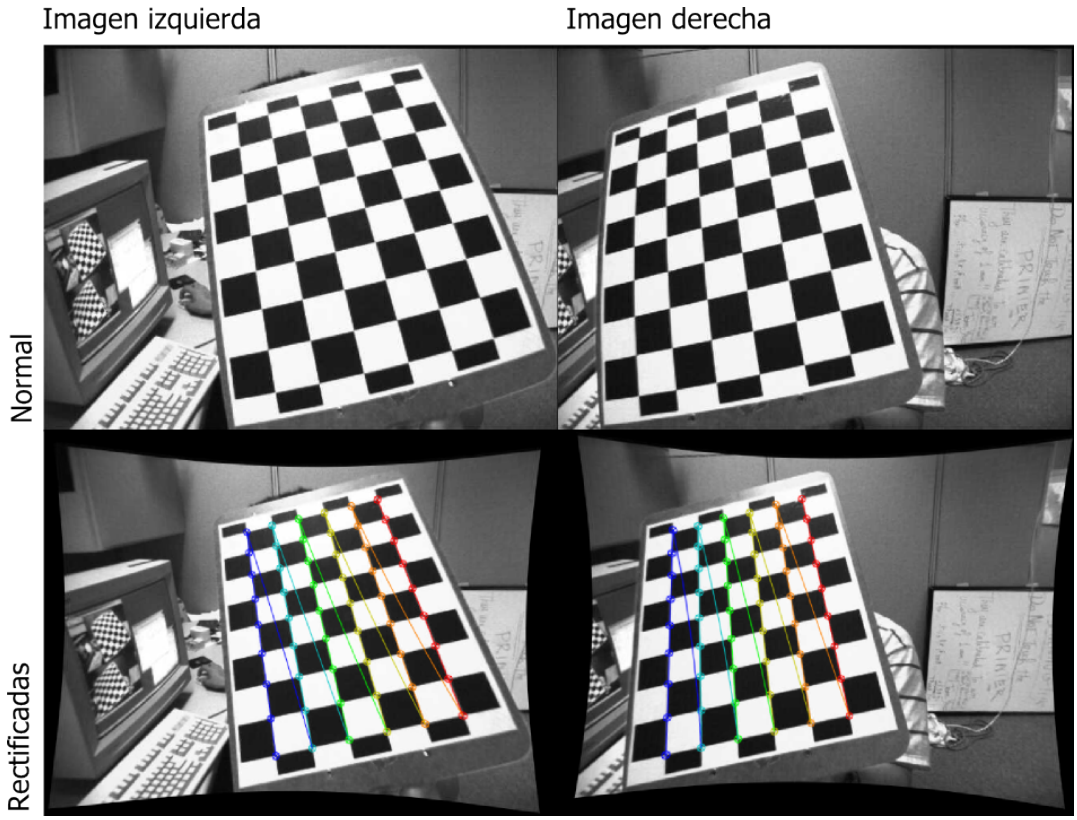

Figura 4.4: Imágenes de prueba de OpenCV rectificadas.

Los parámetros intrínsecos asociados a cada cámara se presentan en las tablas 4.4 y 4.5; en este caso, se añaden los parámetros de rectificación $\mathbf{R}$ y $\mathbf{K}$. Se observa que tanto $\mathbf{M}$ como $\vec{D}$ mantienen valores similares a los obtenidos en la calibración independiente. Como estas imágenes han sido tomadas desde cámaras aproximadamente en disposición binocular, la matriz de rotación de la rectificación tiende a parecerse a la identidad. Finalmente, como al rectificar cambian las dimensiones del área útil de la imagen, este cambio se ve reflejado en las matrices $\mathbf{K}$. En este caso, se consigue un error de 0.479446 al efectuar la re-proyección.

Parámetros	Valores
$\mathbf{M}_0$	$\begin{bmatrix} 534.72 & 0 & 335.77 \\ 0 & 534.74 & 240.38 \\ 0 & 0 & 1 \end{bmatrix}$
$\mathbf{D}_0$	$\begin{bmatrix} -0.288464 & 0.076299 \end{bmatrix}$
$\mathbf{R}_0$	$\begin{bmatrix} 0.9999 & -0.0095 & 0.0089 \\ 0.0095 & 0.9999 & 0.0090 \\ -0.0090 & -0.0089 & 0.9999 \end{bmatrix}$
$\mathbf{K}_0$	$\begin{bmatrix} 426.79 & 0 & 324.71 & 0 \\ 0 & 426.79 & 241.34 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

Tabla 4.4: Parámetros intrínsecos asociados a las imágenes de ejemplo de OpenCV (cámara izquierda).

Parámetros	Valores
$\mathbf{M}_1$	$\begin{bmatrix} 534.72 & 0 & 334.73 \\ 0 & 534.74 & 241.67 \\ 0 & 0 & 1 \end{bmatrix}$
$\mathbf{D}_1$	$\begin{bmatrix} -0.278840 & 0.077077 \end{bmatrix}$
$\mathbf{R}_1$	$\begin{bmatrix} 0.9994 & -0.0142 & 0.0307 \\ 0.0145 & 0.9998 & -0.0087 \\ -0.0306 & 0.0092 & 0.9994 \end{bmatrix}$
$\mathbf{K}_1$	$\begin{bmatrix} 426.79 & 0 & 324.71 & -1425.27 \\ 0 & 426.79 & 241.34 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

Tabla 4.5: Parámetros intrínsecos asociados a las imágenes de ejemplo de OpenCV (cámara derecha).

Por otro lado, los parámetros extrínsecos que relacionan ambas cámaras se pueden ver en la tabla 4.6. Los valores de $\mathbf{R}$ y $\vec{T}$ validan la configuración espacial de las cámaras, pues $\mathbf{R}$ se asemeja a la identidad, y $\vec{T}$ presenta un valor solo significativo en su primera componente. Por otro lado, $\mathbf{E}$ tiene la forma que se espera para este caso (sección 2.2.2.5), sin embargo, $\mathbf{F}$ presenta un 1.0 en la esquina inferior derecha que no corresponde, y se traduce en que las imágenes no están alineadas verticalmente. Salvo este 1.0 erróneo, las imágenes consideradas funcionan adecuadamente con el algoritmo de calibración proporcionado por OpenCV

Parámetros	Valores
R	$\begin{bmatrix} 0.9997 & 0.0052 & -0.0215 \\ -0.0048 & 0.9998 & 0.0181 \\ 0.0216 & -0.0179 & 0.9996 \end{bmatrix}$
T	$\begin{bmatrix} -3.3375 \\ 0.0476 \\ -0.1027 \end{bmatrix}$
E	$\begin{bmatrix} 0.0005 & 0.1019 & 0.0494 \\ -0.0305 & -0.0605 & 3.3384 \\ -0.0314 & -3.3371 & -0.0593 \end{bmatrix}$
F	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -0.0887 \\ 0.0004 & 0.0884 & 1 \end{bmatrix}$
Q	$\begin{bmatrix} 1 & 0 & 0 & -324.71 \\ 0 & 1 & 0 & -241.34 \\ 0 & 0 & 0 & 426.79 \\ 0 & 0 & -0.2994 & 0 \end{bmatrix}$

Tabla 4.6: Parámetros extrínsecos asociados a las imágenes de ejemplo de OpenCV.

4.2.2. Imágenes de la cámara 3D

Para imágenes distintas de las proporcionadas por la librería OpenCV, la función de calibración estéreo no encuentra una buena solución directamente. Se realizaron numerosas pruebas, y una tras otra vez, los errores de re-proyección eran demasiado elevados. Las imágenes se rectificaban erróneamente, visualizándose en la mayoría de los casos solo una mancha. Se ha intentado utilizar como punto de partida los parámetros intrínsecos calculados independientemente, pero los resultados no mejoran significativamente.

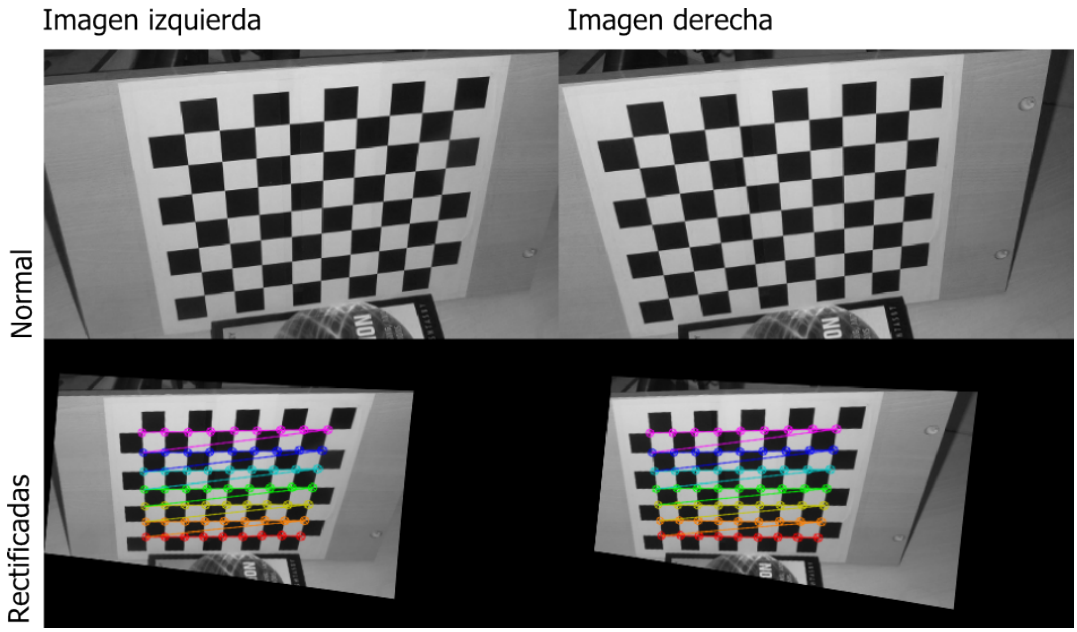

Figura 4.5: Imágenes de cámara 3D rectificadas.

La única opción que genera resultados medianamente buenos, es estimar los parámetros intrínsecos previamente y utilizarlos de forma fija para la calibración estéreo. La figura 4.5 muestra un ejemplo de rectificación; y las tablas 4.7, 4.8 y 4.9 presentan los valores numéricos calculados. Los parámetros se comportan de manera similar a lo visto en los ejemplos de OpenCV, pero dado que el error de re-proyección de esta rectificación es 34.209, son puestos en duda. Del mismo modo, la matriz fundamental presenta el mismo valor 1 inesperado.

La arista del tablero utilizado para calibrar es de $30mm$, y este valor es el utilizado, por lo que las unidades resultantes debieran estar en milímetros. La distancia entre las lentes de la cámara 3D utilizada es de $75mm$, y dicho valor no concuerda con los $2.87mm$ estimados para el vector de traslación $\vec{T}$ en el eje x [6, página 394]. Luego, si bien los resultados obtenidos conservan la tendencia esperada (según análisis en sección 2.2.2.5), la medida dista de ser precisa.

Al observar la figura 4.5, a diferencia del caso anterior (figura 4.4), las imágenes son transformadas en mayor medida, pero manteniéndose lo esperado según el algoritmo de Bouguet (descrito en sección 2.2.4), esto es, la rotación total es equirepartida en ambas imágenes, disminuyendo el error de re-proyección.

Parámetros	Valores
$\mathbf{M}_0$	$\begin{bmatrix} 528.00 & 0 & 224.5 \\ 0 & 531.53 & 126 \\ 0 & 0 & 1 \end{bmatrix}$
$\mathbf{D}_0$	$\begin{bmatrix} -0.07138 & 0.0070 \end{bmatrix}$
$\mathbf{R}_0$	$\begin{bmatrix} 0.9813 & -0.0888 & 0.1703 \\ 0.0889 & 0.9960 & 0.0074 \\ -0.1703 & 0.0078 & 0.9853 \end{bmatrix}$
$\mathbf{K}_0$	$\begin{bmatrix} 344.15 & 0 & 89.47 & 0 \\ 0 & 344.15 & 118.65 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

Tabla 4.7: Parámetros intrínsecos asociados a la cámara 3D (izquierda).

Parámetros	Valores
$\mathbf{M}_1$	$\begin{bmatrix} 531.07 & 0 & 224.5 \\ 0 & 534.04 & 126 \\ 0 & 0 & 1 \end{bmatrix}$
$\mathbf{D}_1$	$\begin{bmatrix} -0.0363 & -0.0215 \end{bmatrix}$
$\mathbf{R}_1$	$\begin{bmatrix} 0.9625 & -0.1016 & 0.2511 \\ 0.1015 & 0.9947 & 0.0131 \\ -0.2512 & 0.0128 & 0.9678 \end{bmatrix}$
$\mathbf{K}_1$	$\begin{bmatrix} 344.15 & 0 & 89.47 & -1026.21 \\ 0 & 344.15 & 118.65 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

Tabla 4.8: Parámetros intrínsecos asociados a la cámara 3D (derecha).

Parámetros	Valores
R	$\begin{bmatrix} 0.9964 & 0.0136 & -0.0828 \\ -0.01347 & 0.9999 & 0.0027 \\ 0.0828 & -0.0016 & 0.9965 \end{bmatrix}$
T	$\begin{bmatrix} -2.8702 \\ 0.3030 \\ -0.7490 \end{bmatrix}$
E	$\begin{bmatrix} 0.0150 & 0.7484 & 0.3040 \\ -0.5085 & -0.0149 & 2.9224 \\ -0.2633 & -2.8741 & 0.0171 \end{bmatrix}$
F	$\begin{bmatrix} 0 & 0.0001 & 0.0158 \\ -0.0001 & 0 & 0.4096 \\ -0.0196 & -0.4156 & 1 \end{bmatrix}$
Q	$\begin{bmatrix} 1 & 0 & 0 & -89.4733 \\ 0 & 1 & 0 & -118.6589 \\ 0 & 0 & 0 & 344.1582 \\ 0 & 0 & -0.3353 & 0 \end{bmatrix}$

Tabla 4.9: Parámetros extrínsecos asociados a la cámara 3D.

De esta forma, el módulo de calibración estéreo no presenta un buen desempeño, no constituyendo una solución apropiada. El problema radica en lo poco robusta de la función asociada de OpenCV 2.3.0. En este sentido, se sugiere implementar manualmente algún algoritmo de calibración, esperar una actualización de esta librería, o buscar otras alternativas.

4.3. Registro no rígido de una línea

En la presente sección se muestran y analizan casos particulares de líneas registradas variando distintos parámetros. Con el fin de generar gráficas comparables entre si, se utiliza en la mayoría de los casos la línea 47 de las imágenes *Venus*[1] reducidas a un cuarto de su tamaño vertical original (promedio simple cada 4 filas consecutivas); solo en la sección 4.3.8, se utiliza la línea 40 de *Tsukuba*[2] (también reducida a un cuarto). Para mantener un análisis focalizado en la acción particular de determinados parámetros, en esta sección no se da información de los otros parámetros utilizados; sin embargo, esta información es provista en el Anexo B.

4.3.1. Interpretación de gráficas

Los resultados para una línea son presentados en 3 gráficas. Tómese por ejemplo la figura 4.6 (a). En la gráfica superior se presenta en verde el desplazamiento horizontal $\phi - x$ reconstruido sobre todo el dominio de x; también se observa en rojo el campo v, destacando en puntos azules cada uno de los nodos. En el gráfico central se visualizan las imágenes

[1] Tamaño original de imágenes Venus: 434×383 píxeles.
[2] Tamaño original de imágenes Tsukuba: 384×288 píxeles.

originales y la deformada; en azul la imagen izquierda (I_0), y en línea punteada verde la imagen derecha (I_1); por último, la línea segmentada roja corresponde a la imagen derecha deformada (I_{1w}), intentando igualarse a su par izquierdo. El gráfico inferior presenta el valor del funcional de similitud sobre el número de iteraciones; conviene mencionar que este valor puede ser interpretado como el error entre I_{1w} e I_0.

Interpretando esta misma figura (4.6 (a)), se nota que un valor negativo en el desplazamiento horizontal se traduce en un movimiento hacia la derecha de la imagen I_1. En este sentido, los bordes de la imagen se deforman en gran medida, mientras que el centro permanece casi inmóvil. La gráfica central permite visualizar, en este caso, un buen calce de las curvas. Por último, en el gráfico del funcional, se aprecia una buena tendencia, la que luego de 250 iteraciones muestra que puede seguir disminuyendo. El caso considerado corresponde a la línea 47 de las imágenes Venus reducidas a un cuarto, es decir, se ubica aproximadamente en la mitad (ver imágenes Venus en figura 4.1); luego, la disparidad calculada concuerda con lo esperado, pues en los bordes se localizan hojas levemente inclinadas, mientras que en el centro se visualiza el fondo.

4.3.2. Distintos funcionales de similitud

La figura 4.6 (a) presenta la acción del funcional de similitud basado en la norma L_2. En 250 iteraciones se nota que puede continuar acercándose al mínimo global. En la gráfica central, se observa como se superponen las curvas I_0 e I_{1w}, lo que da cuenta de un buen resultado. No obstante, en el borde derecho se visualiza un error, producto de que las imágenes consideradas simplemente no son iguales. En particular, la imagen I_1 alcanza a capturar una región no vista por I_0, por lo tanto, cualquier deformación o remapeo que se logre de esta señal, tendrá este mismo problema en dicho borde.

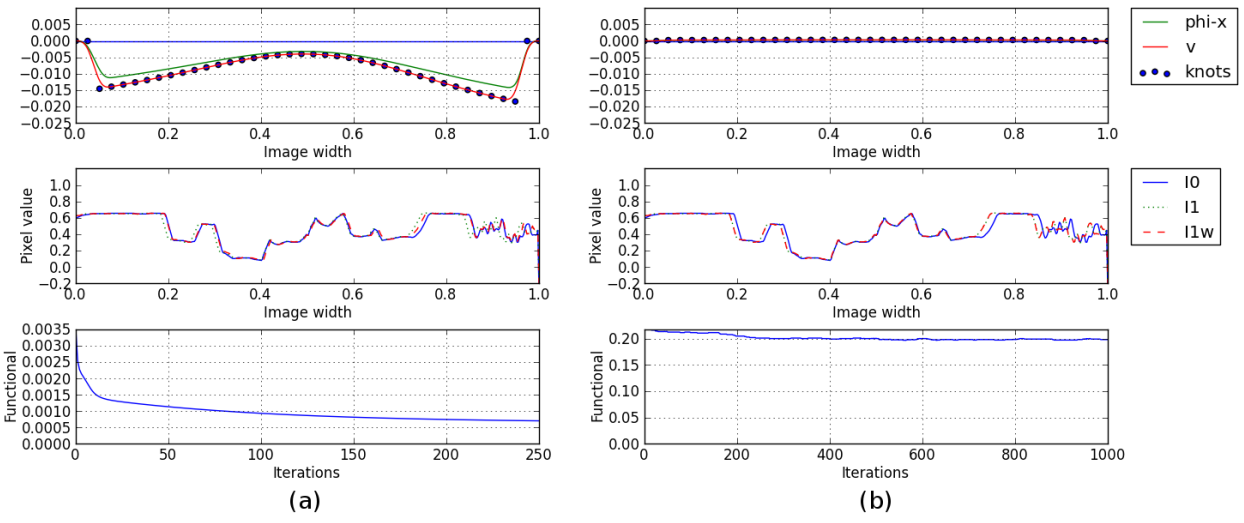

Figura 4.6: Distintos funcionales de similitud. (a) Norma L_2; (b) Norma L_1.

Por otro lado, utilizar la norma L_1 en el funcional de similitud, no genera buenos resultados, el algoritmo simplemente no encuentra la solución. La figura 4.6 (b) constituye un simple

ejemplo. Si en vez de utilizar la norma L_1 como métrica, se utiliza su versión ajustada L_{1A}, sí es posible lograr la convergencia del algoritmo. La figura 4.7 (a) constituye un ejemplo de este caso, dentro de las primeras 100 iteraciones la optimización se presenta relativamente bien, pero luego comienza a oscilar. Para resolver este problema se habilita un paso adaptativo, es decir, reduciendo el paso cuando el funcional incremente su valor, resultando un comportamiento como el descrito por la figura 4.7 (b).

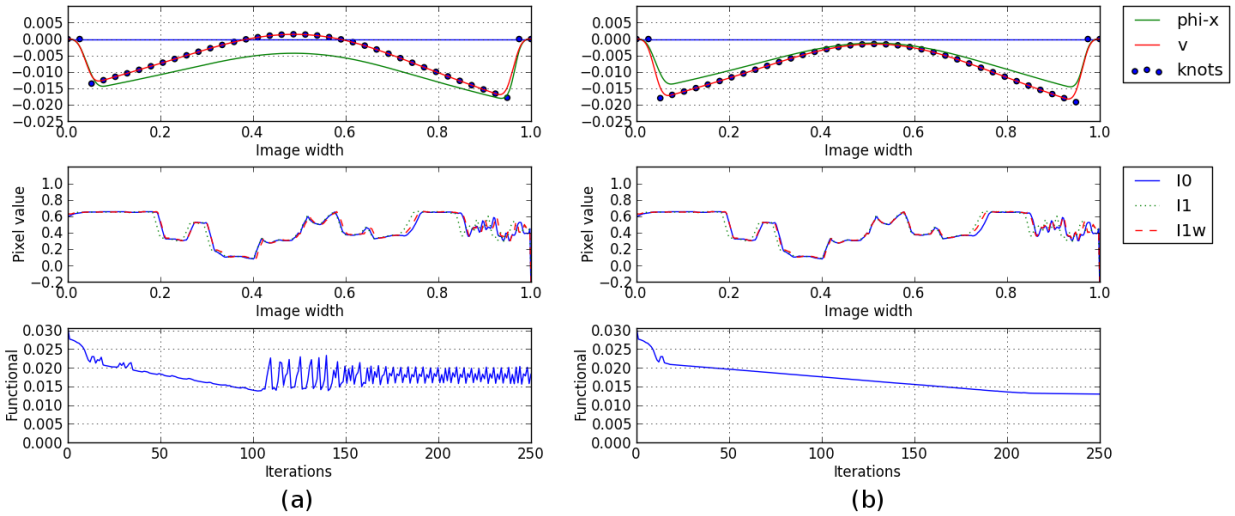

Figura 4.7: Métrica L_{1A} como funcional de similitud. (a) Sin adaptar el paso; (b) Adaptando el paso.

Conviene destacar, que comparar numéricamente los valores de los distintos funcionales carece de sentido, pues son distintas funciones las que se están evaluando.

4.3.3. Distintos funcionales de regularización

Al añadir como regularización la energía del campo vectorial, se intenta disminuir el valor del campo v, y consecuentemente, de ϕ. Este efecto se aprecia claramente al comparar las figuras 4.6 (a) y 4.8 (a), pues luego de 250 iteraciones, el valor, tanto de v como ϕ disminuye en los extremos de la señal, pero aumenta levemente en el centro. El efecto es causado por la excesiva suavidad del difeomorfismo encontrado, no permitiendo modificar una parte de la señal, sin alterar las partes vecinas. El valor del funcional tiende a estabilizarse, por lo que el problema no será resuelto con más iteraciones, la solución simplemente se ha estancado en un mínimo local. Comparándose con el resultado visto en la figura 4.6 (a), se observa que el desplazamiento horizontal en el borde izquierdo de la imagen alcanzaba un valor de alrededor de los 0.011, mientras que al añadir esta regularización, se tiene un valor cercano a 0.007. En este sentido, la regularización evitó la deformación innecesaria de la señal, pues dada la poca textura del sector, un pequeño trozo, puede estirarse, y aun así contabilizar una buena similitud.

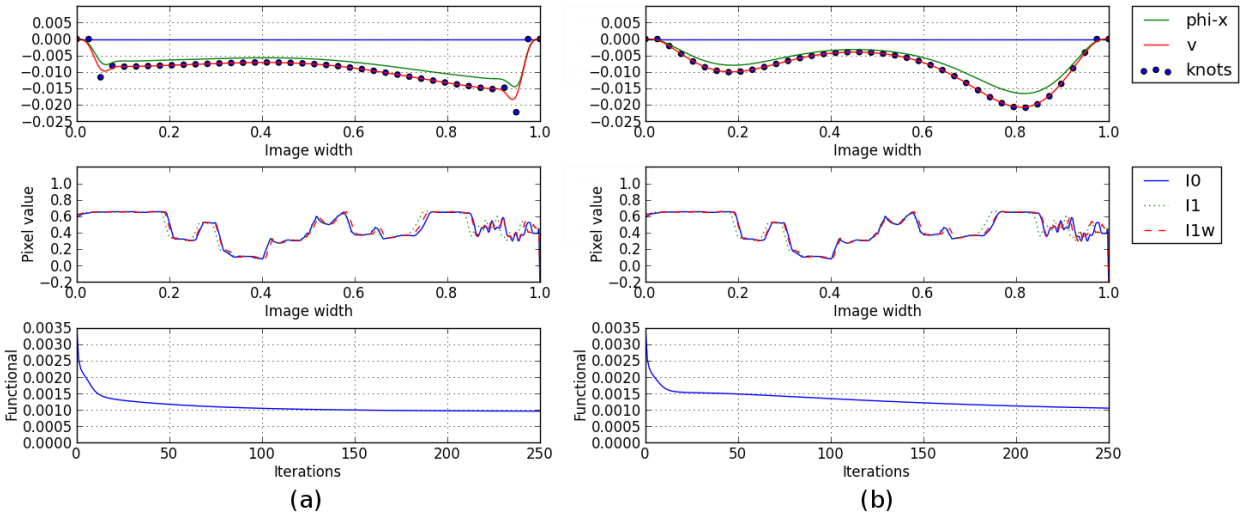

Figura 4.8: Distintos funcionales de regularización. (a) Energía de v; (b) Energía del laplaciano de v.

Por otro lado, al regularizar con la energía del laplaciano del campo vectorial, se minimiza la curvatura de la deformación, por lo que el resultado es sustancialmente más suave. En estas señales en particular, se evitan las abruptas deformaciones presentes en los extremos (ver figuras 4.6 (a),4.7 (b) y 4.8 (a)), sin embargo, esto deja una menor región con resultados significativamente útiles. A diferencia de la regularización previamente descrita, en este caso, sí se logra una buena solución en el centro de la imagen.

Finalmente, el efecto de la regularización es el esperado, la solución se suaviza y evita deformaciones no deseadas. Destaca lo complicado que puede llegar a ser el ajuste del parámetro de ponderación, en las figuras 4.8 (a) y (b), el funcional de regularización se ha ponderado por 0.001 y 0.01 para la energía de v y la energía del laplaciano de v respectivamente. Para valores superiores a estos, la solución simplemente no converge, y para valores menores, su efecto no se puede observar fácilmente. Disminuyendo progresivamente estos ponderadores la solución encontrada se asemeja cada vez más al caso donde no se utiliza regularización (figura 4.6 (a)).

4.3.4. Nodos del campo de velocidades

Es claro que mientras mayor sea la cantidad de nodos con las que se defina el campo vectorial, se podrá representar una mayor variabilidad en las curvas, sin embargo se añade un significativo tiempo de cómputo, volviéndose necesario estimar una cantidad de nodos apropiada para dar con la solución sin excesivos cálculos. Se debe tener presente que una de las motivaciones para utilizar B-splines, es justamente evitar el excesivo cómputo, por lo tanto, la idea es trabajar con una cantidad reducida de nodos.

Las figuras 4.9 (a), 4.9 (b), 4.8 (b), 4.10 (a) y 4.10 (b) presentan soluciones para 10, 20, 40, 80 y 160 nodos del campo v, respectivamente. Conviene recalcar que en cada extremo, se

impone un par de nodos nulos[3], esto es, para mantener la deformación dentro del dominio (ver sección 2.3.2).

En los primeros dos casos, es decir, para 10 y 20 nodos, se observa que las cantidades no son suficientes para representar correctamente al campo v. Las gráficas del valor del funcional versus iteraciones se estabilizan luego de los primeros pasos. En este sentido, la solución encontrada no presenta mayor utilidad.

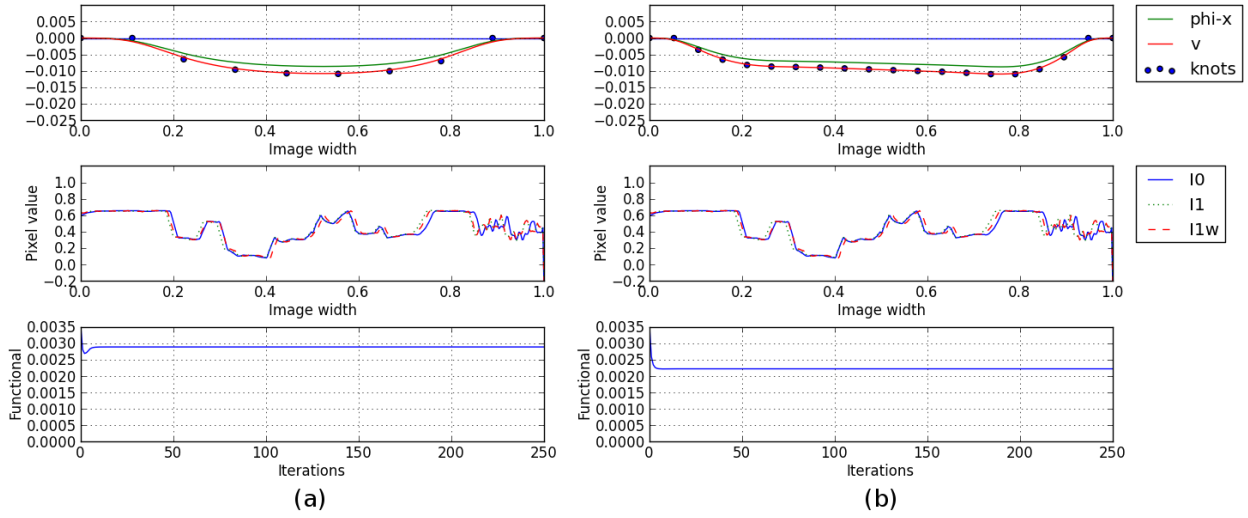

Figura 4.9: Distintas cantidades de nodos. (a) 10 nodos; (b) 20 nodos.

En las curvas de 40, 80 y 160 nodos, se logra un buen calce. Y dado que el funcional no alcanza a estabilizarse, con un mayor número de iteraciones, será posible obtener aun mejores resultados aunque con una lenta convergencia. Por otro lado, una cantidad elevada de nodos, genera mejores resultados en mayor parte de la imagen: el desplazamiento horizontal en el centro de la señal adquiere una tendencia plana, y por otro lado, se logra un mejor calce en el borde derecho.

[3]Solo 2 nodos, pues se utilizan B-splines de orden 4.

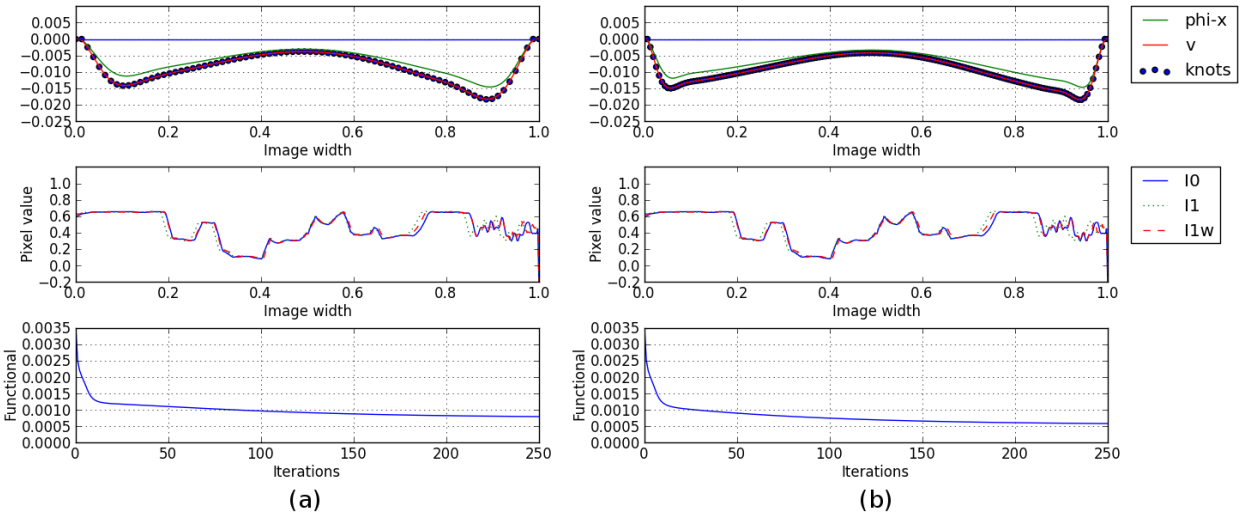

Figura 4.10: Distintas cantidades de nodos. (a) 80 nodos; (b) 160 nodos.

Como se utilizan los mismos funcionales, pueden ser directamente comparados; luego, en cada caso, se logra un menor error en la iteración 250. Sin embargo, tras simple inspección visual, basta con considerar entre 40 y 80 nodos para las imágenes utilizadas.

4.3.5. Discretización temporal

En el detalle de la implementación (ver sección 3.4.3.1), se indicó que el difeomorfismo es calculado según el método de Euler de avance, discretizándose el tiempo en T intervalos. En este sentido, una discretización más fina, debiera lograr una mejor aproximación de la integral, y a la vez, conducir a mejores resultados. Conviene destacar, que el tiempo de cómputo final será fundamentalmente proporcional a la cantidad de intervalos de tiempo considerados.

La tabla 4.10 presenta valores del funcional (norma L_2 o error cuadrático medio), luego de 250 iteraciones, para distintas discretizaciones. Se observa entonces, que utilizar una discretización mayor que 5 o 10 elementos, no mejora en gran medida la convergencia del algoritmo, no justificándose el gran costo que añade su cómputo.

T	Funcional
2	0.000967578
5	0.000700744
10	0.000664089
15	0.000654894
20	0.000650685

Tabla 4.10: Valor del funcional para distintas discretizaciones temporales.

4.3.6. Discretización espacial

Evaluar la integral del funcional de similitud y su gradiente requiere de una discretización espacial, en este caso, para utilizar el método de los rectángulos. Para lograr una buena aproximación, es necesario que al menos la integral considere cada píxel de las imágenes , de esta forma, la implementación realizada considera un multiplicador entero para la cantidad de píxeles de las imágenes originales. Este multiplicador será referenciado como *densidad*. En otras palabras, la densidad indica la cantidad de puntos por píxel que se considerarán para evaluar la integral completa.

Similar al caso de discretización temporal, la tabla 4.11 presenta el valor del funcional tras 250 iteraciones. Similar al caso de la discretización temporal, no conviene realizar una discretización espacial muy fina, pues los resultados no mejoran significativamente, y solo aumenta el tiempo de cómputo.

Densidad	Funcional
2	0.000718679
5	0.000700744
10	0.000696211
15	0.000694875
20	0.000694243

Tabla 4.11: Valor del funcional para distintas discretizaciones espaciales.

4.3.7. Filtrado inicial de las imágenes

El proceso de optimización implementado es muy sensible a la naturaleza de las imágenes utilizadas. Es fácil que en imágenes con rugosidades se llegue a un mínimo local, no lográndose la solución deseada; y por otro lado, si las imágenes presentan amplias zonas de similar intensidad, pueden producirse deformaciones innecesarias. Una estrategia para sobrellevar esta dificultad, es efectuar un filtrado pasa-bajos en mayor o menor medida. A su vez, es posible encontrar una solución para las imágenes suavizadas, y luego utilizar esta solución como punto de partida para procesar las mismas imágenes pero menos suaves.

En la figura 4.8 (b) se presenta una buena convergencia, en particular se ha utilizado un filtro caja horizontal 2 veces de un tamaño 1×7 píxeles. Ahora, si se cambia el tamaño del mismo filtro a uno de 1×3, no se obtiene un resultado erróneo (figura 4.11 (a)).

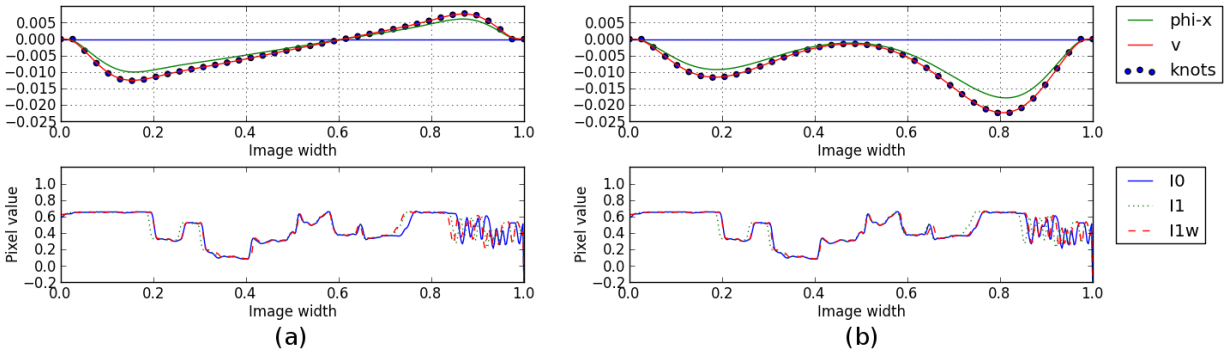

Figura 4.11: (a) Imágenes procesadas directamente; (b) Imágenes procesadas sucesivamente de mayor a menor filtrado.

Es natural entonces formular una estrategia de búsqueda progresiva, partiendo de imágenes muy suavizadas, hasta las imágenes originales. La figura 4.11 (b) muestra el resultado de procesar sucesivamente: 200 iteraciones con filtro de 1×7; 200 iteraciones con filtro de 1×5; y finalmente, 200 iteraciones con filtro de 1×3. En este caso, los resultados sí adquieren buena tendencia.

4.3.8. Oclusiones

Las imágenes *Venus* poseen pocas oclusiones, por lo que constituye un caso relativamente simple para optimizar. Por otro lado, en el caso general, siempre habrán grandes o pequeñas oclusiones.

La figura 4.12, presenta el resultado de procesar una fila de las imágenes de *Tsukuba*. En los rectángulos se indican importantes oclusiones, las que producen que el algoritmo se estanque en un mínimo local, impidiendo un calce correcto entre las curvas I_0 e I_1. De esta forma, acorde a las pruebas realizadas, el método del gradiente no es capaz de encontrar el óptimo en presencia de grandes oclusiones.

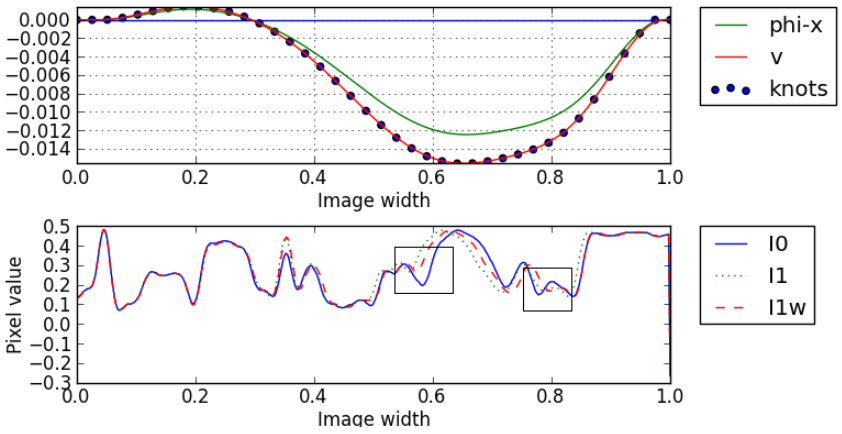

Figura 4.12: Imágenes con oclusiones.

4.4. Registro no rígido de una imagen

En todas las pruebas unidimensionales presentadas en las sección 4.3 se observa que las curvas de deformación son considerablemente suaves. Luego, al repetir el procedimiento para cada fila de la imagen, inevitablemente se mantendrá dicha característica de suavidad. Se espera por tanto visualizar mapas de disparidad más bien borrosos.

La figura 4.13 presenta los resultados obtenidos para las imágenes *Venus*. Del mapa de disparidad real (esquina superior derecha de la misma figura), se sabe que las 2 hojas de periódico se encuentran en un primer plano, y todo el resto constituye fundamentalmente un fondo. En este sentido, el mapa de disparidad calculado visualiza estas hojas como suaves manchas y deja el fondo estático en las regiones superior y central de la imagen. La suavidad de la solución generada por este método conlleva a imprecisiones importantes en los bordes de los objetos, los que son confirmados en la imagen de error presente en la misma figura.

El mapa de disparidad calculado considera una escala de grises centrada y normalizada; esto es, si la imagen derecha se mueve a la izquierda, el nivel de gris se acerca al negro; y por otro lado, si se mueve hacia la derecha, el nivel de gris se acerca al negro; en un segundo paso, la imagen es escalada es su escala de grises de forma que el máximo o mínimo valor se iguale al blanco o al negro. De esta forma, cuando no hay desplazamiento horizontal, se visualiza el color plomo. Se debe recordar que este mapa se obtiene con imágenes cuyos fondos están alineados, por lo que existe un *offset* de traslación entre ambas, y consecuentemente, el mapa no puede ser directamente comparado con el real.

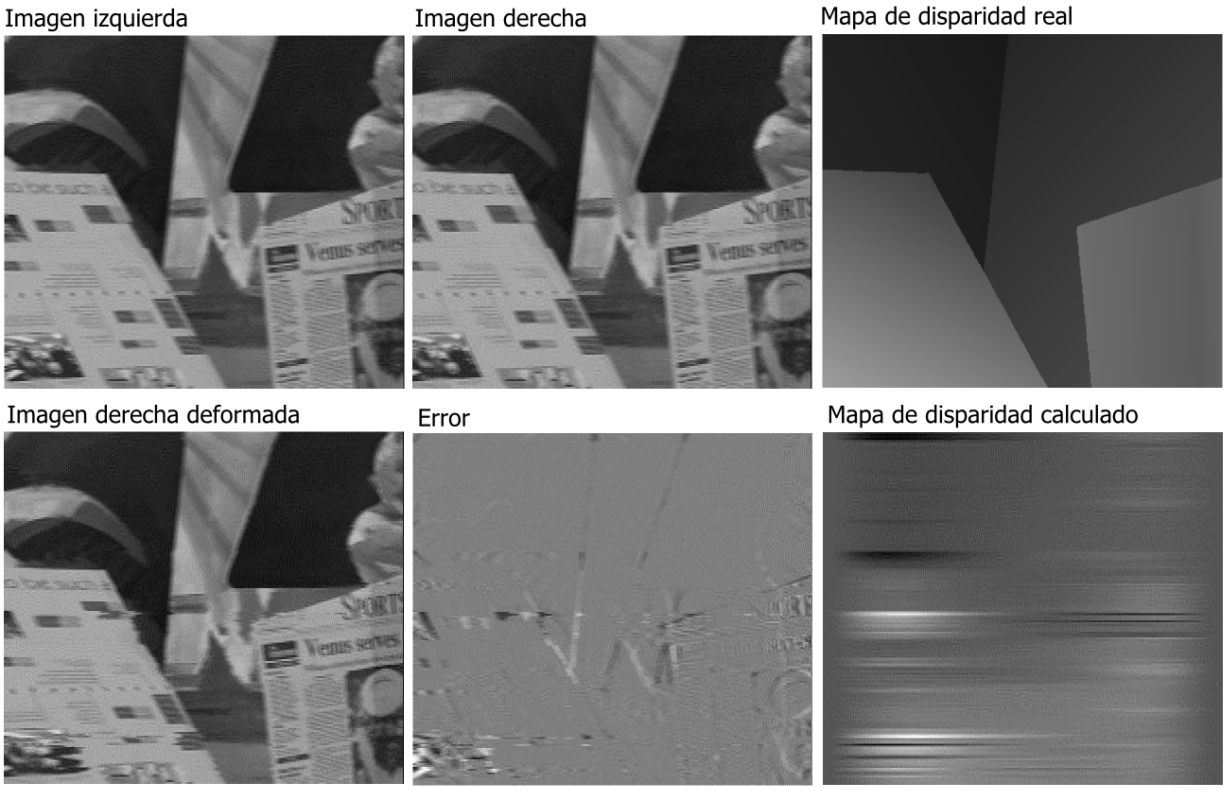

Figura 4.13: Resultados para imágenes *Venus*.

Características similares son observadas en los resultados para las imágenes *Cones*, *Tsukuba* y *Mouse* (figuras 4.14, 4.15 y 4.16, respectivamente). Es decir, se observan manchas blancas en las posiciones de los objetos más cercanos, que por lo tanto, llevan asociada una mayor disparidad. Los resultados muestran cualitativamente una buena tendencia general, sin embargo, no se obtiene la precisión requerida para discriminar correctamente cada elemento, ni mucho menos, generar un adecuado mapa de profundidad.

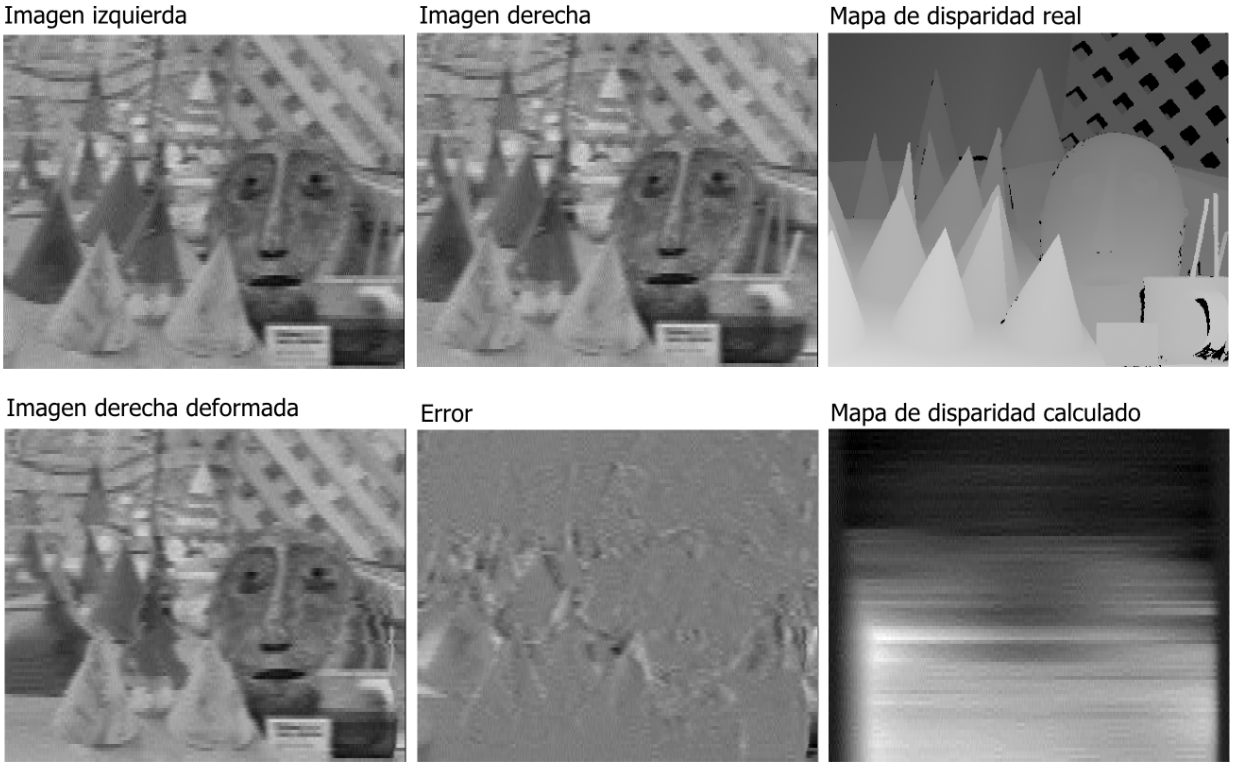

Figura 4.14: Resultados para imágenes *Cones*.

En particular, las imágenes *Mouse* fueron tomadas con la cámara 3D, preparando un escenario simple: sin grandes oclusiones, bordes iguales y una forma suave. Los resultados presentan una mancha clara en el centro, lo que da cuenta del volumen asociado al mismo mouse. La periferia de la imagen es color gris, al cual, como se ha mencionado, se asocia un movimiento horizontal nulo.

Analizando la imagen de disparidad desde arriba hasta abajo, se observan errores al comienzo y final del mouse. Esto sugiere errores de alineación del par de imágenes, dado que al intentar igualar curvas fundamentalmente distintas, la función de deformación no se calcula correctamente.

Se destacan además las pequeñas imperfecciones presentes arriba y abajo del *scroll*, estas líneas no lograron converger a la solución correcta. En comparación con el resto de las líneas, se sugiere la falta de textura o suficientes variación del nivel de gris en esta región, lo que en definitiva produce una desviación equívoca.

Para el resto de la imagen, la tendencia inducida por el mouse se identifica correctamente. Sin embargo, al tratarse de un desplazamiento excesivamente suave, no solo se deforma la

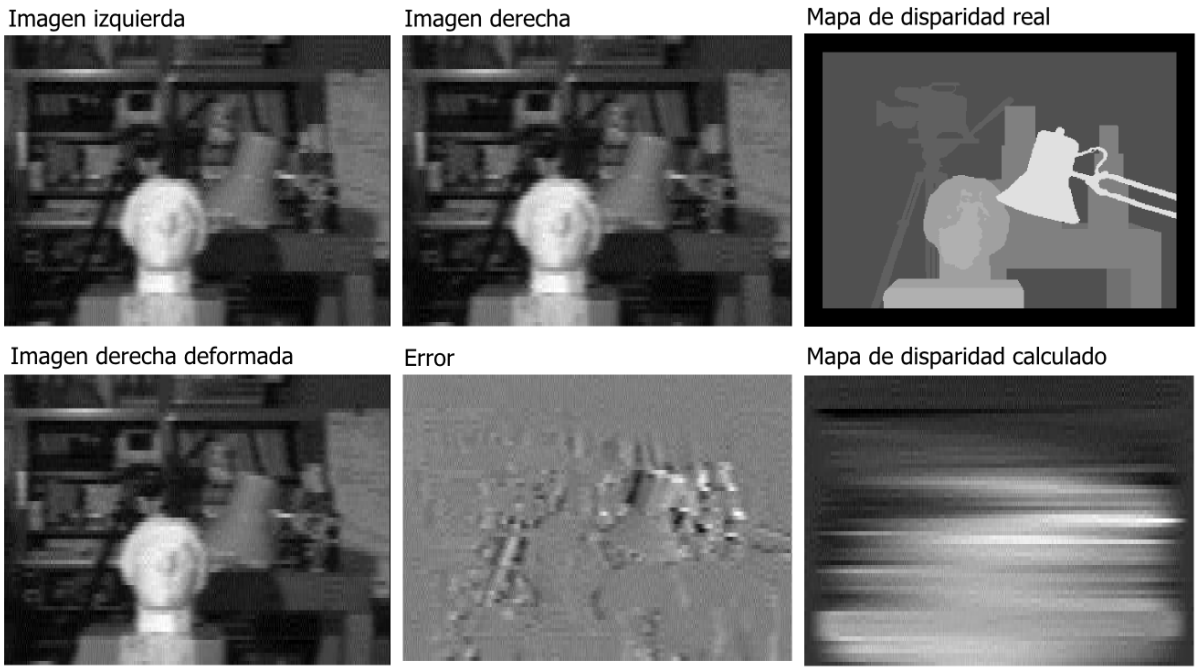

Figura 4.15: Resultados para imágenes *Tsukuba*.

sección asociada al mouse, sino que también se acarrean otras zonas, las que se distinguen en sus bordes.

La excesiva suavidad de la solución encontrada no permite discriminar adecuadamente los objetos presentes en complejas escenas, sin embargo, el mecanismo propuesto permite encontrar correspondencias en imágenes con gradientes suaves. Luego, las imágenes *Mouse* presentan un buen caso de aplicación para el método desarrollado.

Conviene mencionar que justamente para imágenes de estas características, es decir, con suaves variaciones de color, muchos métodos tradicionales no encuentran una buena solución. En la figura 4.17 se presentan mapas de disparidad asociados a las mismas imágenes *Mouse*, pero calculados con los 4 métodos que proporciona la librería OpenCV. Las manchas centrales se asocian directamente al mouse, por lo que estos algoritmos encuentran una solución, sin embargo, en las regiones negras no se proporciona un cálculo de disparidad. Se visualizan además importantes errores en la esquina superior derecha en 3 de los 4 métodos, mientras que en el cuarto, la solución no permite identificar correctamente el mouse.

En este caso particular, el método propuesto se comporta mejor que las funciones provistas por OpenCV. Sin embargo, los tiempos de cómputo son muy diferentes, los métodos de OpenCV encuentran el mapa de disparidad en menos de un segundo, mientras que la implementación realizada tarda alrededor de medio día. Por lo tanto, hay mucho que trabajar en cuanto a la velocidad de procesamiento.

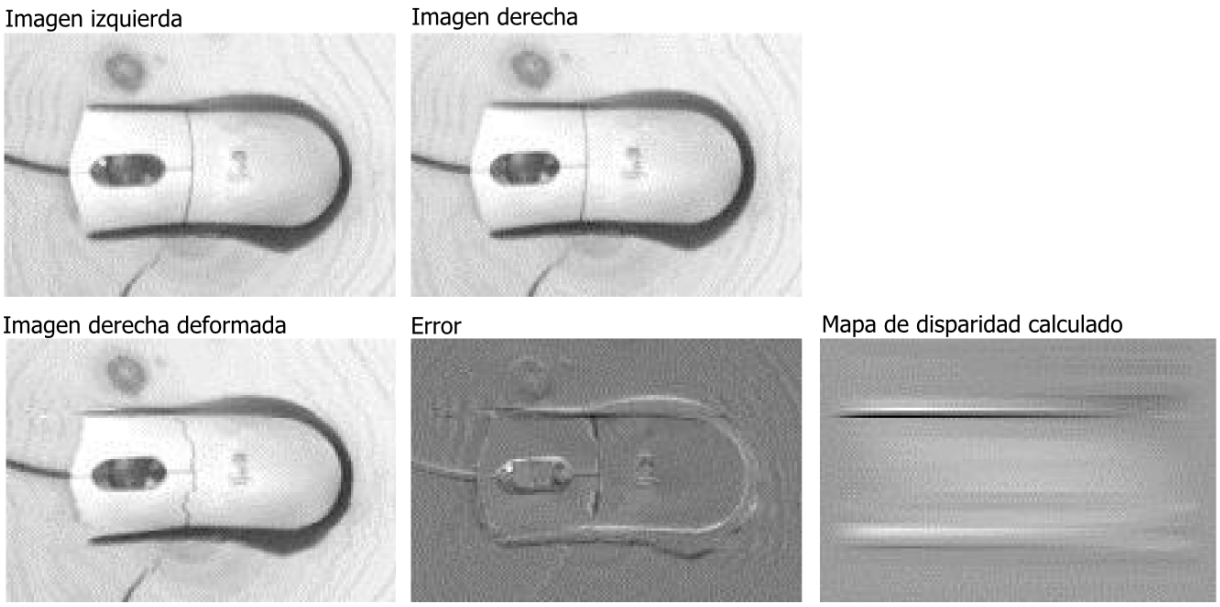

Figura 4.16: Resultados para imágenes *Mouse*.

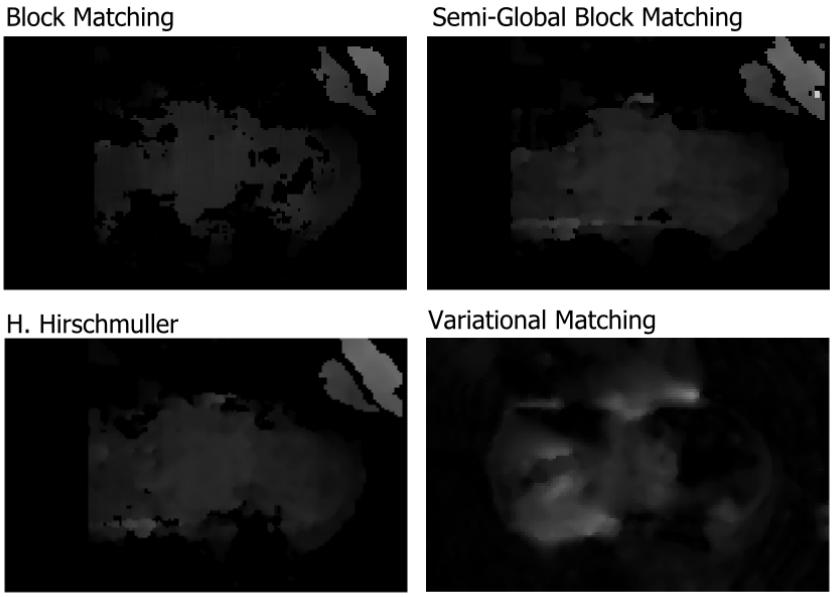

Figura 4.17: Mapas de disparidad para imágenes *Mouse* utilizando OpenCV.

4.5. Generación de mapa de profundidad

Teniendo los parámetros de cámara y el mapa de disparidad, es directa la generación del mapa de profundidad. Los parámetros extrínsecos de la cámara 3D calculados con OpenCV no son lo suficientemente buenos; y dada la alineación binocular de la cámara 3D utilizada, se opta por una estimación manual considerando una distancia entre cámaras de $75mm$ y

los parámetros intrínsecos de la cámara izquierda (tabla 4.2). Con esta información y el registro no rígido de las imágenes *Mouse* presentada en la figura 4.16 se genera el mapa de profundidad presentado en la figura 4.18. En este caso se ha optado por presentar una superficie tridimensional.

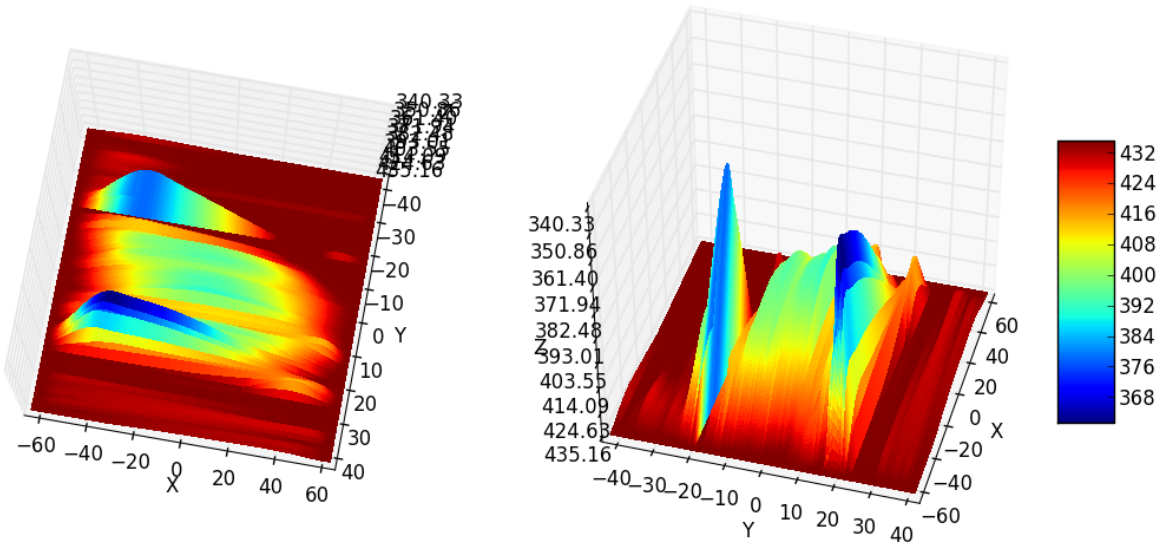

Figura 4.18: Mapa de profundidad para imágenes *Mouse*. Unidades en milímetros.

Es claro que el mapa presenta abundantes errores, pero fundamentalmente provienen de la etapa de cálculo de disparidad anterior. Los bordes quedan bien definidos en rojo, indicando el plano asociado al tablero. En el centro se distingue el mouse; sus medidas se estiman en unos 45, 100 y $35mm$, en las direcciones x, y y z, respectivamente; las que concuerdan, a grandes rasgos, con las medidas reales del mouse ($65 \times 110 \times 40mm$), validando una primera aproximación. La solución se muestra distinta línea a línea, debido a que el problema es abordado de esta forma, se estima que un enfoque bidimensional pueda generar mejores resultados, eliminando los grandes errores al inicio y término del mouse ya mencionados.

Capítulo 5

Conclusiones

A partir del presente trabajo, es posible la generación de un mapa de profundidad a partir de imágenes estéreo de un sistema correctamente calibrado.

La estimación de parámetros intrínsecos de una cámara fue bien lograda gracias a la utilización de OpenCV. Por otro lado, una calibración estéreo robusta quedó fuera del alcance del trabajo, simplificándose a una estimación manual de parámetros para una disposición binocular.

El registro no rígido se abordó de manera unidimensional. Los resultados presentan dificultades en oclusiones, periodicidades y curvas complejas. Posibles soluciones se enmarcan en: cambios de funcionales; uso de estrategias híbridas, mezclándose con estrategias de puntos característicos; y finalmente, cambio del método de optimización. El enfoque de registro no rígido permite tratar el caso puntual de regiones sin grandes gradientes en los niveles de gris, proponiéndose como buena alternativa en estas situaciones.

Se ha completado un primer paso en una forma distinta de abordar la correspondencia entre imágenes estéreo. Como no se ha dado con una solución general, inevitablemente surgen múltiples variantes que podrían presentar mejores desempeños.

Las conclusiones se agrupan en los temas de geometría de cámaras, B-splines y registro no rígido. Finalmente, se dan lineamientos generales para trabajos futuros.

5.1. Geometría de cámaras

Con el fin de generar un mapa de profundidad representativo y preciso, es fundamental contar con una buena estimación de todos los parámetros asociados a las cámaras. El trabajo se limitó a utilizar funciones de la librería OpenCV para abordar esta tarea. Si bien los resultados para la calibración independiente de cada cámara son buenos, no es así en el caso de un par estéreo de cámaras. En esta línea, es necesaria la búsqueda de mecanismos que resuelvan esta tarea eficazmente. Este problema dista de ser nuevo y ha sido tratado en abundante literatura, por lo que basta con encontrar un enfoque teórico e implementación lo suficientemente funcional. Al término del presente trabajo, se ha actualizado la versión de

OpenCV a 2.4.2.; un primer sería revisar si esta implementación proporciona una solución robusta al tema de calibración estéreo.

La aplicación implementada permite generar un mapa de profundidad a partir del mapa de disparidad. En teoría, sería posible para cualquier disposición espacial de cámaras mientras mantengan una amplia zona en común. Sin embargo, dadas las dificultades para dar con parámetros de calibración precisos, solo se ha considerado el caso de cámaras binoculares.

5.2. B-splines

El tratamiento de un algoritmo en un espacio continuo no siempre puede ser directamente implementado. El enfoque básico para abordar estos problemas radica en una discretización lo suficientemente fina, lo que en definitiva se traduce en un excesivo tiempo de cómputo.

En el presente trabajo, se han utilizado B-splines, las que permiten un tratamiento continuo del espacio, pero realizando operaciones discretas. Sus propiedades de soporte compacto, rápida evaluación, interpolación, derivada e integración simples, entre otras, las hacen idóneas para este tipo de aplicaciones. En concreto, las B-splines han permitido implementar de manera rápida y fácil un algoritmo pensado directamente un marco continuo. En cuanto al código C++ implementado, gracias a un objeto B-spline, es posible visualizar directamente las fórmulas analíticas que se requiera implementar.

5.3. Registro no rígido

Considerando un enfoque unidimensional, se logró calcular un difeomorfismo que deforme una señal para convertirla en otra. Los resultados, en las múltiples pruebas realizadas, demuestran que el algoritmo cumple con encontrar una deformación apropiada. Es importante efectuar un buen ajuste de los parámetros involucrados para generar mejores resultados.

A partir del análisis de los parámetros que controlan el registro, se comprobó que ni la discretización temporal ni la espacial requieren ser muy finas para lograr buenos resultados. También se observó que la norma L_1 directamente no sirve para evaluar el funcional, pero si se introduce un pequeño ajuste para "redondear" en torno a cero, se obtienen resultados apropiados. La norma L_2 se comporta bien, pero castiga mucho los casos con oclusiones. El trabajo puede ser fácilmente extendido probando otros funcionales, y a la vez, mezclándolos con los ya considerados.

Si las señales unidimensionales presentan demasiada complejidad, como partes notoriamente diferentes o periodicidades, el algoritmo no encuentra la solución buscada, estancándose en algún mínimo local. El mecanismo de filtrado suavizador de la señal, de forma previa al cómputo del registro, ayuda a encontrar una mejor solución. En este sentido, la estrategia óptima comprende encontrar la solución final en etapas sucesivas, donde la señal se filtra menos cada vez a modo de relajación. Por supuesto, un exceso de filtrado elimina las pequeñas variaciones de color que requiere el algoritmo para determinar la solución buscada, pudiendo producir deformaciones innecesarias.

Considerando la ayuda del proceso de pre-filtrado, los resultados en una línea aparentan ser buenos. Sin embargo, el caso no se generaliza al tratar con un par de imágenes estéreo completas. Se han identificado 3 grandes problemas:

- Oclusiones: en general evitan una convergencia correcta, pues el método del gradiente se estanca en un mínimo local. Las oclusiones son inherentes de este caso particular de aplicación. Se requiere por tanto, diseñar un mecanismo que permita la existencia de discontinuidades en la función de deformación.
- Bordes de las imágenes: se ha asumido que son iguales, cuando este no constituye el caso general dado el fenómeno de perspectiva. En las pruebas realizadas con bordes diferentes, los resultados son, por supuesto, inválidos en estas zonas. Si las diferencias entre bordes son demasiado notorias, a veces incluso se impide la correcta convergencia del algoritmo.
- Suavidad: las funciones de deformación obtenidas en cada caso siempre son suaves, impidiéndose la identificación apropiada de objetos en la escena.

La teoría que fundamenta el procedimiento es correcta, pues los funcionales considerados efectivamente poseen el mínimo global en el punto solución buscado; es decir, en la situación donde la segunda imagen se transforma en la primera imagen original. La raíz del problema se encuentra entonces en el mecanismo de optimización. El método del gradiente demuestra no ser adecuado para resolver este problema. Una alternativa puede ser convexificar aún más el funcional a considerar (para no estancarse en mínimos locales); y otra es directamente cambiar el procedimiento de minimización.

La gran ventaja del método propuesto es que, solo requiere variaciones del nivel de gris, el algoritmo automáticamente efectuará el calce de curvas. Métodos basados en caracterización de puntos requieren de abundante textura y gradientes del nivel de gris pronunciados en las imágenes. En este sentido, se ataca un problema no resuelto. Ahora, en presencia de colores principalmente lisos y sin una adecuada regularización, es posible encontrar deformaciones degeneradas; esto es, estiramientos innecesarios que aún así pueden validar un mínimo funcional, dado que poseen igual intensidad del nivel de gris. Inmediatamente se sugiere entonces la implementación de un mecanismo híbrido, mezclándose con enfoques basados en puntos característicos, permitiendo evadir oclusiones, pero que en el resto de la imagen se efectúe un registro continuo y correcto píxel a píxel.

5.4. Trabajo futuro

Dada la presencia de múltiples mínimos locales (oclusiones y periodicidades), se estima que un algoritmo de optimización que explore una mayor cantidad de alternativas permita obtener mejores resultados. En esta línea, destacan por ejemplo, algoritmos genéticos y optimización por enjambre de partículas[1], aunque se sacrificaría tiempo de cómputo.

[1] También llamados *PSO*, del inglés *Particle Swarm Optimization*

En el escenario donde no se encontró una solución definitiva al problema, surge una infinidad de posibles alternativas. Se pueden distinguir dos líneas principales: aquellas que mantienen un enfoque de registro no rígido exclusivo; y otras que aprovechan particularidades del caso de imágenes estéreo. En la línea de mantener la generalidad, surgen alternativas como la consideración del caso no estacionario, o incluso la búsqueda de otras familias de difeomorfismos. En el caso de considerar un enfoque híbrido, surgen ideas como efectuar primeramente estrategias de *block matching* o identificación de puntos característicos, para luego refinar en el continuo con deformaciones difeomórficas.

Continuando en la misma línea del presente trabajo, es posible abordar el problema en 2 dimensiones. El movimiento de los nodos de una superficie B-spline otorgará la inercia necesaria para evitar caer en soluciones erróneas en líneas particulares; y así, se encontrará un mejor registro para la imagen completa. Como en el caso de imágenes estéreo alineadas como binocular solo se esperan movimientos horizontales; es posible aprovechar esta misma característica en la optimización, restringiendo el movimiento de los nodos solo en una dirección.

Una de las desventajas del método propuesto es la incerteza del tiempo del procesamiento, pues, cada línea es un problema diferente, requiriendo de un número indeterminado de iteraciones para establecer una buena correspondencia. Si bien el cómputo de la deformación es considerablemente lento, es fácilmente paralelizable, pues cada fila es procesada independientemente. De esta forma, la implementación se vuelve natural en una GPU[2]. Disponiendo de cientos de procesadores, cada uno podrá procesar una o varias líneas, pudiendo lograrse una paralelización prácticamente perfecta. En este sentido, el tiempo de cómputo se reduciría en un factor de escala considerable.

Una extensión natural del trabajo realizado, es la integración de distintos mapas de profundidad para construir un modelo tridimensional completo. El problema a tratar aquí es considerablemente distinto; e involucra disciplinas más cercanas a las ciencias de la computación, como geometría computacional y computación gráfica. Sin embargo, la construcción de un completo escáner tridimensional, solo a partir de imágenes, es sin lugar a dudas, atractiva y con múltiples y potenciales aplicaciones.

[2]Del inglés *Graphics Processing Unit*.

Bibliografía

[1] Takeshi Asahi. *1-Dimensional Spline functions*. 1999.

[2] Luis Baumela. *Apuntes de Visión por Computador*. <http://www.dia.fi.upm.es/~lbaumela/doctorado/> [consulta: julio 2012]. Universidad Politécnica de Madrid, Departamento de Inteligencia Artificial, 2005.

[3] M. F. Beg, M. I. Miller, A. Trouvé, and L. Younes. Computing large deformation metric mappings via geodesic flows of diffeomorphisms. *Int. J. Comput. Vision*, 61(2):139–157, 2005.

[4] Jérémie Bigot, Sébastien Gadat, and Jean-Michel Loubes. Statistical m-estimation and consistency in large deformable models for image warping. *J. Math. Imaging Vis.*, 34(3):270–290, July 2009.

[5] M. Bossa, E. Zacur, and S. Olmos. Algorithms for computing the group exponential of diffeomorphisms: Performance evaluation. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on*, pages 1 –8, june 2008.

[6] Gary Bradski. *Learning OpenCV : computer vision with the OpenCV library*. O'Reilly, Sebastopol, CA, 2008.

[7] Middlebury Data Base. <http://vision.middlebury.edu/stereo/> [consulta: julio 2012].

[8] OpenCV Developer Site. <http://code.opencv.org> [consulta: julio 2012].

[9] OpenCV Documentation. <http://docs.opencv.org/> [consulta: julio 2012].

[10] Ardeshir Goshtasby. *2-D and 3-D image registration for medical, remote sensing, and industrial applications*. John Wiley & Sons, Hoboken, N.J, 2005.

[11] Richard Hartley. *Multiple view geometry in computer vision*. Cambridge University Press, Cambridge, UK New York, 2000.

[12] H. Hirschmuller and D. Scharstein. Evaluation of cost functions for stereo matching. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1 –8, june 2007.

[13] Gunnar Karlsson and Martin Vetterli. Extension of finite length signals for sub-band coding. *Signal Processing*, 17(2):161–168, 1989.

[14] Sergey Kosov. *Multi-View 3D Reconstruction with Variational Method.* Saarland University, 2008.

[15] Boost C++ Libraries Documentation. <http://www.boost.org/> [consulta: julio 2012].

[16] D. Scharstein and C. Pal. Learning conditional random fields for stereo. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1 –8, june 2007.

[17] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–195 – I–202 vol.1, june 2003.

[18] D. Scharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *Stereo and Multi-Baseline Vision, 2001. (SMBV 2001). Proceedings. IEEE Workshop on*, pages 131 –140, 2001.

[19] Santiago Sippa. *Construcción de un mapa tridimensional usando visión binocular.* Universidad de Chile, 2005.

[20] M. Unser. Splines: a perfect fit for signal and image processing. *Signal Processing Magazine, IEEE*, 16(6):22 –38, nov 1999.

[21] M. Unser, A. Aldroubi, and M. Eden. Fast b-spline transforms for continuous image representation and interpolation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(3):277 –285, mar 1991.

[22] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing. i. theory. *Signal Processing, IEEE Transactions on*, 41(2):821 –833, feb 1993.

[23] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing. ii. efficiency design and applications. *Signal Processing, IEEE Transactions on*, 41(2):834 –848, feb 1993.

[24] OpenCV Wiki. <http://opencv.willowgarage.com/wiki/> [consulta: julio 2012].

[25] Laurent Younes. *Deformation analysis for shape and image processing.* <http://web.univ-pau.fr/~cgout/viscosite/old/20052006/younesdeformationAnalysis.pdf> [consulta: julio 2012].

[26] Laurent Younes. *Shapes and diffeomorphisms.* Springer, Heidelberg New York, 2010.

Anexo A: Terminología

- **M**: Matriz de cámara.
- f: Distancia focal de la cámara. Se asume igual en las direcciones horizontal y vertical.
- f_x, f_y: Distancias focales horizontal y vertical.
- $\vec{C} = (c_x, c_y)$: Coordenadas en píxeles del eje principal.
- $\vec{D}$: Vector de distorsión.
- **K**: Matriz de proyección 3D-2D.
- **Q**: Matriz de re-proyección.
- $\vec{P_h} = (X_h, Y_h, Z_h, W)$: Punto del espacio tridimensional en coordenadas homogéneas.
- $\vec{P} = (X, Y, Z)$: Punto del espacio tridimensional en coordenadas cartesianas.
- $\vec{p_h} = (x_h, y_h, f)$: Punto en el plano de imagen en unidades métricas y coordenadas homogéneas, proviene de proyectar el punto tridimensional P.
- $\vec{p} = (x, y)$: Punto en el plano de imagen en unidades métricas y coordenadas cartesianas, proviene de proyectar el punto tridimensional P.
- $\vec{q_h} = (i_h, j_h, w)$: Punto en la imagen en unidades de píxeles y coordenadas homogéneas.
- $\vec{q} = (i, j)$: Punto en la imagen en unidades de píxeles y coordenadas cartesianas.
- **R**: Matriz de rotación extrínseca. Se encarga de rotar el sistema de referencia propio de la cámara derecha al sistema de referencia de la cámara izquierda.
- $\vec{T}$: Vector de traslación extrínseca. Se encarga de trasladar el sistema de referencia propio de la cámara derecha al sistema de referencia de la cámara izquierda.
- **E**: Matriz Esencial.
- **F**: Matriz Fundamental.
- $\phi(x, t)$: Función difeomórfica. Se considera: $x \in [0, 1]$ (ancho de la imagen) y $t \in [0, 1]$ (intervalo finito de tiempo).
- $v(x)$: Campo vectorial de deformación.

- $\beta^n(x)$: Función base B-spline de orden n. Equivale al producto de convolución de n funciones $rect(x) = \begin{cases} 1 & |x| < 0.5 \\ 0 & \sim \end{cases}$.

Anexo B: Parámetros de cada prueba

	Valor			
Parámetros	fig. 4.6 (a)	fig. 4.6 (b)	fig. 4.7 (a)	fig. 4.7 (b)
Imágenes	Venus			
Tamaño	423×379 píxeles			
Filtrado	Caja horizontal de 7×1 píxeles			
Reducción vertical	4			
Fila	47			
Paso de iteración	10.0	0.0001	0.001	
Discretización temporal	5			
Densidad espacial	5			
Nodos de v	40			
Orden de B-splines	4			
Máximo de iteraciones	250	1000	250	
Similitud	Norma L_2	Norma L_1	Métrica L_{1A} ($\lambda = 0.0001$)	
Regularización	no			
Adaptar paso	no			si

Tabla 5.1: Pruebas con distintos funcionales de similitud.

	Valor	
Parámetros	fig. 4.8 (a)	fig. 4.8 (b)
Imágenes	Venus	
Tamaño	423×379 píxeles	
Filtrado	Caja horizontal de 7×1 píxeles	
Reducción vertical	4	
Fila	47	
Paso de iteración	10.0	
Discretización temporal	5	
Densidad espacial	5	
Nodos de v	40	
Orden de B-splines	4	
Máximo de iteraciones	250	
Similitud	Norma L_2	
Regularización	Energía de v. $\alpha_E = 0.001$	Energía del laplaciano de v. $\alpha_L = 0.01$
Adaptar paso	si	

Tabla 5.2: Pruebas con distintos funcionales de regularización.

	Valor			
Parámetros	fig. 4.9 (a)	fig. 4.9 (b)	fig. 4.10 (a)	fig. 4.10 (b)
Imágenes	Venus			
Tamaño	423×379 píxeles			
Filtrado	Caja horizontal de 7×1 píxeles			
Reducción vertical	4			
Fila	47			
Paso de iteración	10.0			
Discretización temporal	5			
Densidad espacial	5			
Nodos de v	10	20	80	160
Orden de B-splines	4			
Máximo de iteraciones	250			
Similitud	Norma L_2			
Regularización	Energía del laplaciano de v. $\alpha_L = 0.01$			
Adaptar paso	no			

Tabla 5.3: Pruebas con distintos nodos del campo v.

Parámetros	Valor
Imágenes	Venus
Tamaño	423×379 píxeles
Filtrado	Caja horizontal de 7×1 píxeles
Reducción vertical	4
Fila	47
Paso de iteración	10.0
Discretización temporal	2, 5, 10, 15 y 20
Densidad espacial	5
Nodos de v	40
Orden de B-splines	4
Máximo de iteraciones	250
Similitud	Norma L_2
Regularización	Energía del laplaciano de v. $\alpha_L = 0.01$
Adaptar paso	no

Tabla 5.4: Pruebas con distintas discretizaciones temporales. Asociado a tabla 4.10

Parámetros	Valor
Imágenes	Venus
Tamaño	423×379 píxeles
Filtrado	Caja horizontal de 7×1 píxeles
Reducción vertical	4
Fila	47
Paso de iteración	10.0
Discretización temporal	5
Densidad espacial	2, 5, 10, 15 y 20
Nodos de v	40
Orden de B-splines	4
Máximo de iteraciones	250
Similitud	Norma L_2
Regularización	Energía del laplaciano de v. $\alpha_L = 0.01$
Adaptar paso	no

Tabla 5.5: Pruebas con distintas discretizaciones espaciales. Asociado a tabla 4.11

	Valor	
Parámetros	fig. 4.11 (a)	fig. 4.11 (b)
Imágenes	Venus	
Tamaño	423×379 píxeles	
Filtrado	600 iteraciones 2 cajas horizontales de 3×1 píxeles	200 iteraciones 2 cajas horizontales de 7×1 píxeles
		200 iteraciones 2 cajas horizontales de 5×1 píxeles
		200 iteraciones 2 cajas horizontales de 3×1 píxeles
Reducción vertical	4	
Fila	47	
Paso de iteración	10.0	
Discretización temporal	5	
Densidad espacial	5	
Nodos de v	40	
Orden de B-splines	4	
Similitud	Norma L_2	
Regularización	Energía del laplaciano de v. $\alpha_L = 0.01$	

Tabla 5.6: Pruebas con parámetros de partida.

Parámetros	Valor
Imágenes	Tsukuba
Tamaño	384×288 píxeles
Filtrado	2 cajas horizontales de 7×1 píxeles
Reducción vertical	4
Fila	30
Paso de iteración	10.0
Discretización temporal	5
Densidad espacial	5
Nodos de v	40
Orden de B-splines	4
Máximo de iteraciones	1000
Similitud	Norma L_2
Regularización	Energía del laplaciano de v. $\alpha_L = 0.001$
Adaptar paso	si

Tabla 5.7: Prueba con oclusiones. Asociado a figura 4.12.

Parámetros	Valor
Imágenes	Venus
Tamaño	423×379 píxeles
Filtrado	caja horizontal de 7×1 píxeles
Reducción vertical	2
Paso de iteración	10.0
Discretización temporal	5
Densidad espacial	5
Nodos de v	40
Orden de B-splines	4
Máximo de iteraciones	250
Similitud	Norma L_2
Regularización	Energía del laplaciano de v. $\alpha_L = 0.01$
Adaptar paso	no

Tabla 5.8: Prueba con oclusiones. Asociado a figura 4.12.

Parámetros	Valor
Imágenes	Cones
Tamaño	425×372 píxeles
Filtrado	2 cajas horizontales de 7×1 píxeles
Reducción vertical	4
Paso de iteración	20.0
Discretización temporal	5
Densidad espacial	5
Nodos de v	40
Orden de B-splines	4
Máximo de iteraciones	500
Similitud	Norma L_2
Regularización	no
Adaptar paso	si

Tabla 5.9: Prueba con oclusiones. Asociado a figura 4.14.

Parámetros	Valor
Imágenes	Tsukuba
Tamaño	357×279 píxeles
Filtrado	2 cajas horizontales de 7×1 píxeles
Reducción vertical	4
Paso de iteración	10.0
Discretización temporal	5
Densidad espacial	5
Nodos de v	80
Orden de B-splines	4
Máximo de iteraciones	500
Similitud	Norma L_2
Regularización	Energía del laplaciano de v. $\alpha_L = 0.01$
Adaptar paso	si

Tabla 5.10: Prueba con oclusiones. Asociado a figura 4.15.

Parámetros	Valor
Imágenes	Mouse
Tamaño	155×103 píxeles
Filtrado	no
Reducción vertical	no
Paso de iteración	7.5
Discretización temporal	5
Densidad espacial	4
Nodos de v	40
Orden de B-splines	4
Máximo de iteraciones	250
Similitud	Norma L_2
Regularización	Energía del laplaciano de v. $\alpha_L = 0.01$
Adaptar paso	si

Tabla 5.11: Prueba con oclusiones. Asociado a figura 4.16.