



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
POSGRADO EN CIENCIAS E INGENIERÍA DE LA COMPUTACIÓN

**INSTRUMENTACIÓN DE ALGORITMOS DE VISIÓN ESTEREOSCÓPICA
PARA UN ROBOT HUMANOIDE**

TESIS

**QUE PARA OPTAR POR EL GRADO DE:
MAESTRO EN INGENIERÍA (COMPUTACIÓN)**

**PRESENTA:
LUIS SERGIO DURÁN ARENAS**

**TUTOR:
DR. JESÚS SAVAGE CARMONA
FACULTAD DE INGENIERÍA**

MÉXICO, D. F. ENERO 2016



Universidad Nacional
Autónoma de México

Dirección General de Bibliotecas de la UNAM

Biblioteca Central



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Resumen

En este trabajo se presenta un sistema de visión estereoscópica para la obtención de una nube de puntos en tiempo real a partir de dos imágenes de una misma escena, utilizando el algoritmo SGBM para el empatado de ambas imágenes, y la posibilidad de implementarlo en un robot humanoide con recursos limitados. Se presenta el sistema dividido en 3 etapas: preprocesamiento, procesamiento y postprocesamiento. La etapa de preprocesamiento comprende todas aquellas técnicas para refinar las imágenes a empatar; la etapa de procesamiento está dada por la implementación del algoritmo SGBM para el empatado de dos imágenes estéreo y con ello obtener un mapa de disparidad; por último, la etapa de postprocesamiento implica la obtención de la nube de puntos usando el mapa de disparidad, aplicando algunas técnicas para mejorar dicho mapa. Se realizaron algunas pruebas de desempeño en varios equipos de cómputo para verificar la viabilidad de implementar el sistema en el robot NimbRo-OP. Estas pruebas fueron realizadas reescalando las imágenes obtenidas originalmente (procedimiento ejecutado en la etapa de preprocesamiento) para disminuir la cantidad de información a procesar, se obtuvieron los tiempos promedio para el cálculo del mapa de disparidad en cada escala. Además, se hicieron algunas pruebas con la nube de puntos para verificar que puede ser utilizada para ciertos fines, en particular, encontrar el plano más sobresaliente de una escena con el método RANSAC.

Agradecimientos

Agradezco al Posgrado en Ciencias e Ingeniería de la Computación en donde fue posible realizar mis estudios de maestría y a su coordinación que nos ayudó en todo momento, también al Consejo Nacional de Ciencias y Tecnología (CONACYT) por el apoyo brindado a lo largo de dichos estudios.

Se agradece a la DGAPA-UNAM por el apoyo proporcionado para la realización de esta tesis a través del proyecto PAPIIT IG100915 "Desarrollo de técnicas de la robótica aplicadas a las artes escénicas y visuales".

Índice general

Índice general	I
Lista de figuras	III
Lista de tablas	V
1. Introducción	1
1.1. Panorama general	1
1.2. Problemática	2
1.3. Objetivo	4
1.4. Descripción y estructura	5
2. Conceptos básicos y estado del arte	7
2.1. Visión Computacional	8
2.1.1. Segmentación	9
2.1.2. Detección y descripción de características	10
2.2. Modelo <i>Pinhole</i>	12
2.3. Visión Estereoscópica	12
2.4. Algoritmos en Visión Estereoscópica	15
2.4.1. Métodos de empatado	16
2.4.2. Función de costo	18
2.5. Robótica	18
2.6. Estado del arte	19
2.7. Discusión	22
3. Marco teórico	23
3.1. Registro de imágenes 2D	25
3.2. Preprocesamiento de imágenes	26
3.2.1. Parámetros de una cámara	27
3.2.2. Distorsión	28
3.2.3. Reescalamiento	30
3.2.4. Filtrado	32
3.2.5. Realzado	33
3.3. Procesamiento de visión estereoscópica	34
3.3.1. Obtención de características	35
3.3.2. Correspondencia entre imágenes	36

3.3.3. Registro de imágenes estéreo	45
3.4. Postprocesamiento de información	50
3.4.1. Corrección del mapa de disparidad	51
3.4.2. Proyección de información en 2D a 3D	51
3.4.3. Profundidad basada en la disparidad	51
4. Sistema de Visión Estereoscópica	53
4.1. Etapa de preprocesamiento.	56
4.1.1. Calibración de las imágenes	58
4.1.2. Filtrado de las imágenes	58
4.1.3. Realce de imágenes	59
4.1.4. Reescalamiento	61
4.2. Etapa de procesamiento	62
4.2.1. Registro de imágenes	62
4.2.2. Mapa de disparidad	64
4.3. Etapa de postprocesamiento	64
4.3.1. Procesamiento sobre el mapa de disparidad	66
4.3.2. Reproyección 2D a 3D	66
5. Pruebas y resultados	69
5.1. Análisis	70
5.1.1. Empatado de imágenes y mapa de disparidad	70
5.1.2. Análisis a diferentes escalas	71
5.1.3. Obtención de nubes de puntos	74
5.2. Comparación	75
5.2.1. Tiempos de ejecución a diferentes escalas	76
5.2.2. Comparación de nubes de puntos	77
5.3. Pruebas de procesamiento de la nube de puntos	79
5.4. Otras pruebas	81
5.4.1. Análisis en superficies con textura y uniformes	81
6. Conclusiones	83
6.1. Trabajo futuro	84
A. Algoritmo SGBM para el empatado de imágenes estéreo.	86
B. Reproyección de una imagen en 2D a una nube de puntos en 3D.	94
Bibliografía	96

Lista de figuras

1.1. Robot NimbRo-OP.	3
2.1. Proyección de perspectiva con modelo <i>pinhole</i>	13
2.2. Modelo de geometría epipolar.	14
2.3. Orientación de las imágenes	15
3.1. Patrón de cuadros para calibración de una cámara	27
3.2. Ejemplo de distorsión radial	29
3.3. Ejemplo de distorsión tangencial	30
3.4. Proceso de decimación	31
3.5. Proceso de interpolación	31
3.6. Proceso de reescalamiento de una señal por un factor I/D	32
3.7. Ejemplo de un histograma	33
3.8. Representación de un punto en el espacio a un sistema de visión estéreo .	47
3.9. Representación de un punto en el espacio a un sistema de visión estéreo con ejes ópticos convergentes	49
4.1. Diagrama de bloques del sistema de visión estereoscópica.	54
4.2. Algunas capturas para la calibración.	55
4.3. Diagrama de bloques de la etapa de preprocesamiento.	57
4.4. Ejemplo de dos imágenes rectificadas.	59
4.5. Aplicación de un filtro Gaussiano.	59
4.6. Implementación del realce de imagen por especificación del histograma . .	61
4.7. Diagrama de bloques de la etapa de procesamiento.	63
4.8. Ejemplo de mapa de disparidad obtenido con el algoritmo SGBM.	64
4.9. Diagrama de bloques de la etapa de postprocesamiento.	65
4.10. Proceso de remoción de huecos en el mapa de disparidad	67
4.11. Mapeo de la imagen RGB sobre el mapa de disparidad.	68
5.1. Ejemplo de captura de una escena con visión estéreo.	70
5.2. Mapa de disparidad	71
5.3. Ejemplo de una nube de puntos obtenida.	72
5.4. Capturas de una misma imagen a escalas diferentes.	73
5.5. Gráfica del cálculo del error de reproyección y error RMS a diferentes escalas de la escena de prueba.	74
5.6. Capturas de una escena con visión estereoscópica.	77
5.7. Gráfica de tiempo promedio para procesamiento del mapa de disparidad .	78
5.8. Nubes de puntos obtenidas con 2 sistemas diferentes	79
5.9. Gráfica de tiempos promedio para procesamiento de la nube de puntos . .	80

5.10. Cálculo del plano en una nube de puntos	81
5.11. Escena con una superficie texturizada	82
5.12. Escena con una superficie uniforme	82
A.1. Diagrama del funcionamiento del algoritmo <i>Semiglobal Matching</i>	87

Lista de tablas

5.1. Características de los equipos de cómputo comparados	69
5.2. Relación escala-resolución de las imágenes	72
5.3. Resultados en diferentes escalas	75
5.4. Nubes de puntos a diferentes escalas	76
5.5. Promedio de tiempo de cálculo para el mapa de disparidad a diferentes escalas	77
5.6. Promedio de tiempo de procesamiento de la nube de puntos para obtener un plano	80

Capítulo 1

Introducción

1.1. Panorama general

Con el fin de desarrollar sistemas robóticos autónomos que puedan realizar acciones propias del ser humano, ya sea para reducir riesgos, para evitar esfuerzos o para auxiliar gente, la investigación en robótica ha crecido en los últimos años. La Robótica, como rama de la tecnología, ha marcado un gran hito en la ciencia, puesto que involucra otras grandes ramas para su desarrollo propio, tales como: la Inteligencia Artificial, la Visión por Computadora, el Procesamiento del Lenguaje Natural, entre otras.

Una manera de llevar a cabo investigación que logre impulsar el crecimiento de la Robótica, a nivel nacional como internacional, es a través de competencias en las que se ponen a prueba sistemas robóticos construidos por seres humanos, que sean capaces de lograr metas fijas y que cumplan con determinados objetivos. En México, una de ellas es el Torneo Mexicano de Robótica (TMR) que es una competencia de robótica anual organizada por la Federación Mexicana de Robótica¹ en la que se pretende promover el desarrollo de robots en nuestro país y motiva a estudiantes de diferentes niveles académicos a ampliar sus conocimientos e imaginación construyendo robots competitivos ante equipos de diferentes partes del país en distintas categorías; por ejemplo, robots de servicio, robots de soccer, robots de rescate, etc. Las categorías anteriores también se realizan en competencias internacionales, como lo es la RoboCup. Ésta surgió como una iniciativa para el desarrollo y la investigación en Robótica, cuyo objetivo principal es que para el año 2050 se cree un equipo de robots de soccer capaz de enfrentar y vencer al equipo de humanos ganador del campeonato mundial de soccer de la FIFA de ese mismo año. Aunque el objetivo principal de RoboCup era construir un equipo de robots que jueguen soccer, la propuesta se ha extendido a otras áreas, primordialmente robots de servicio,

¹<http://femexrobotica.org/>

donde se espera construir robots capaces de asistir a las personas en un ambiente común; y robots de rescate, donde un robot recorre un entorno de desastre con el objetivo de encontrar personas debajo de los escombros.

Si bien como seres humanos contamos con un determinado número de sentidos con los que percibimos el ambiente, lo que se busca en la Robótica es emular dichas percepciones. Una manera de percibirlo es a través de la vista, con ella podemos ver nuestro alrededor. En el caso de la Robótica, específicamente en Electrónica, para adquirir la misma información que obtenemos los seres vivos a través de los ojos existen sensores de visión que comúnmente se utilizan como cámaras. Con este tipo de sensor se adquiere toda una secuencia de bits que puede ser interpretada como una imagen.

El uso de la Visión por Computadora sirve como medio para percibir lo que hay en el entorno de un robot. Con este trabajo de tesis se pretenden involucrar algoritmos de Visión Computacional en el ámbito de la robótica, con el fin de aportar al desarrollo de la misma. Pero cabe mencionar que así como reconocemos objetos y formas en un instante de tiempo, lo ideal es que un robot haga lo mismo y para lograrlo es necesario realizar un sistema que funcione en tiempo real.

1.2. Problemática

La problemática central de este trabajo es la siguiente:

Se tiene un robot humanoide NimbRo-OP, que se muestra en la Figura 1.1, con ciertas características²:

- Altura: 95 cm.
- Peso: 6.6 kg.
- 20 servomotores
- PC: AMD Dual-Core a 1.65 GHz y 2GB de memoria RAM
- Conexión Wi-Fi
- Cámara con lente gran angular
- Acelerómetro de 3 ejes
- Giroscopio de 3 ejes

²Las características descritas del robot, se tomaron de la página web (<http://www.nimbro.net/OP/NimbRo-OP.html>) del equipo NimbRo de la Universidad de Bonn, en Alemania.

- Batería Li-Po
- Estructura: fibra de carbono, aluminio y ABSplus³.



FIGURA 1.1: Robot NimbRo-OP.

El robot debe cumplir con especificaciones definidas para participar, realizar comportamientos propios de un robot jugador de soccer, observar el entorno y reconocer la cancha, la pelota, las porterías y otros robots, y actuar con respecto a esos datos obtenidos.

Dado lo anterior, se pretende incorporar un sistema de visión estereoscópica en este robot que pueda dar información en 3D del entorno para facilitar sus acciones y mejorar la precisión.

La razón por la que se decidió trabajar en este tema en particular es debido a que obtener información en 3D del ambiente en el que se encuentra el robot es útil para localizar los diversos objetos que se requieren reconocer y realizar las diferentes acciones necesarias que cumplan con el propósito específico, particularmente para un robot humanoide de soccer.

En otras palabras, el objetivo de este trabajo es incorporar visión estereoscópica en un robot humanoide, cuyos recursos de hardware son limitados. Este tipo de visión es la que percibimos los seres humanos y que nos da sentido de profundidad en el ambiente. Esto se logra a través de dos imágenes que tienen relación entre sí, al hacer dos capturas del mismo entorno en diferentes orientaciones pero con una escena en común, esta escena debe tener las características necesarias para determinar que ambas imágenes están

³ABSplus es el material utilizado por las impresoras 3D y se utiliza para las manos, y originalmente para la cabeza, en el robot NimbRo-OP.

observando el mismo entorno, de modo que entre ambas imágenes pueda realizarse una triangulación que permita determinar la distancia aproximada a la que se encuentra un punto común a cualquiera de las dos imágenes, conociendo con anterioridad la orientación de la segunda imagen con respecto a la primera. A este método de determinar la distancia de un punto común entre dos imágenes se le llama *geometría epipolar*, cuya definición se detallará más adelante en este escrito.

Algo importante que hay que mencionar es que en categorías de soccer para robots humanoides en RoboCup no se puede hacer uso de sensores activos, es decir, que emitan algún tipo de patrón que le ayude a adquirir o procesar datos, puesto que en un ambiente de soccer se pueden tener varios agentes que interfieran entre sí con dichos patrones. La información de 3D obtenida con algoritmos de visión estéreo no necesita de sensores activos, pues sólo hace uso de los sensores de visión (cámaras), y de ser posible, incorporar algún sensor de inercia para precisar los datos obtenidos, estos sensores se consideran pasivos, al no emitir algún patrón en el ambiente.

1.3. Objetivo

Desarrollar e integrar un algoritmo de visión estereoscópica en un robot humanoide cuyos recursos de cómputo son limitados, ello con la finalidad de realizar la adquisición de información visual del ambiente, así como aplicar técnicas de refinamiento de dicha información, para obtener datos en 3D del entorno a través de nubes de puntos.

Además, dado que existen competencias en el ámbito de robótica, tanto nacionales como internacionales, introducir al robot a ellas, en las que se requieren realizar comportamientos basados en la información proporcionada por el módulo de visión. Específicamente, la idea primordial es incorporarlo a un ambiente de soccer. Sin embargo, es posible que su aplicación se extienda a otros ámbitos, dado que se puede incorporar en otros sistemas que utilicen visión y que requieran información en 3D de un entorno.

Para lograr el objetivo propuesto, se partirá de la siguiente hipótesis: es posible integrar un algoritmo de visión estereoscópica en un equipo de cómputo con recursos limitados para obtener una nube de puntos que pueda proporcionar información tridimensional, y que esta nube pueda procesarse para el posterior reconocimiento de objetos dentro de un entorno.

1.4. Descripción y estructura

En el presente trabajo se analiza un método de adquisición de información en 3D con ayuda de visión estereoscópica para un robot humanoide que sea capaz de formar parte de un equipo de robots que se desempeñen en un ambiente de soccer. Se tiene contemplado que el robot NimbRo-OP pueda tener varias funciones dentro del campo de juego, como portero o delantero, de tal modo que los algoritmos implementados sean acordes a la posición en la que se desempeñe, además de apoyar en la decisión de realizar comportamientos que ayuden a cumplir con su objetivo como tal.

El robot NimbRo-OP fue parte del equipo que representó a México en la categoría Humanoid Kid-Size League en la RoboCup del año 2014, que se realizó en Brasil. El equipo representante fue el Team Mexico, formado por estudiantes y académicos de la Universidad La Salle, el Tecnológico de Monterrey y la UNAM. Ahí se pusieron a prueba algunas capacidades del robot y se analizó su desempeño, identificando aquellos módulos que podrían ser mejorados o que necesitaban especial atención.

A partir de este punto, el presente trabajo está estructurado de la siguiente manera:

En el capítulo 2 se da una explicación breve acerca de los conceptos básicos utilizados a lo largo del trabajo. Al final del capítulo se presenta una sección con el estado del arte que corresponde al tema en cuestión, mencionando algunos trabajos que se han realizado.

En el capítulo 3 se detalla el marco teórico de un sistema de visión estereoscópica, lo que involucra, algunos algoritmos que se utilizan, así como partir de que este sistema puede dividirse en varias etapas. Se explicarán cada uno de los procedimientos que se involucran en cada etapa del sistema.

El capítulo 4 es referente al desarrollo del sistema de visión estéreo y su funcionamiento, utilizando la estructura de un sistema como el que se menciona en el capítulo 3, haciendo un análisis de los métodos vistos para la adquisición de información en 3D a través de visión estereoscópica, para con ello obtener una nube de puntos confiable y, posteriormente, hacer uso de esta información.

El capítulo 5 trata acerca de los resultados obtenidos con el sistema de visión estereoscópica, es decir, la adquisición de una nube de puntos, y analizamos qué tan factible es incorporarlo en el robot humanoide, además, se comparan algunos otros sistemas que existen en el mercado. Para demostrar que la nube de puntos puede utilizarse de una manera práctica, se obtiene el plano más sobresaliente de la nube utilizando el método RANSAC.

Por último, en el capítulo 6 se expresan las conclusiones a las que se llegó en el presente trabajo, incluyendo las posibilidades que se pueden lograr con éste y sus posibles aplicaciones.

Capítulo 2

Conceptos básicos y estado del arte

La Visión Computacional en el área de Robótica es fundamental en cierto sentido, dado que una gran cantidad de sistemas robóticos de hoy en día se encuentran inmersos en ambientes dinámicos, que cambian constantemente. Para ello, dichos sistemas deben estar adecuados con sensores capaces de reconocer este tipo de cambios. La visión en un robot móvil es muy útil al utilizarse para conocer hacia dónde debe moverse, qué es lo que hay en el entorno, qué puede hacer en él y de qué manera puede hacerlo. La visión puede proporcionar suficientes capacidades a un robot para desempeñarse en nuestro entorno dinámico. La visión estereoscópica sirve para determinar la forma en 3D de superficies visibles en una escena estática con imágenes tomadas por dos cámaras o por una sola en diferentes posiciones de esa misma escena [47].

Una ventaja de un sistema de visión estereoscópica es que con él se puede tener noción de profundidad en el espacio, así como funciona el sistema de visión humana, en el que los ojos funcionan como sensores y cada uno proporciona una imagen independiente de una misma escena pero que al relacionarlas entre sí, el cerebro las interpreta y hace una escena tridimensional de la escena vista.

Nuestro objeto de estudio es la visión estereoscópica aplicada en un robot humanoide que tiene las características definidas en el capítulo anterior. A este robot se le adaptará un sistema de dos cámaras con modelo *pinhole*, distribuidas de manera horizontal a una distancia fija para obtener y facilitar el procedimiento de la visión estereoscópica.

A continuación, se definirán los conceptos más importantes concernientes a nuestro problema y que son muy útiles en este escrito.

2.1. Visión Computacional

La Visión Computacional se considera como la rama de la Computación que se encarga de realizar el procesamiento de imágenes obtenidas a través de sensores (cámaras), que perciben el entorno para interpretar lo que hay en él, una manera de simular el sistema de visión humana. Una de las aplicaciones más comunes de la Visión Computacional en la actualidad es en la robótica, en la que se pretende hacer que un robot sea capaz de observar lo que el ojo humano ve, además de interpretarlo de una manera clara.

Según se dice en [47], la filosofía en Visión Computacional es: “la visión puede ser estudiada en 3 niveles:

- A nivel de teoría de la computación: cada tarea de visión podría ser vista en términos de funciones de procesamiento de información que pueden ser comunes e independientes del algoritmo que se utilice para desempeñar el cómputo.
- A nivel de algoritmo: existen varios algoritmos, éstos mismos pueden ser independientes del hardware sobre el cual son implementados. Sin embargo, pueden presentarse casos donde algún algoritmo sólo puede implementarse en algún sistema de hardware.
- A nivel de implementación: depende del sistema en el que se busca desempeñar.

Con base en esta filosofía, la Visión Computacional se ha desarrollado sustancialmente en los últimos años gracias a los sistemas de cómputo que se han mejorado y a la accesibilidad a ellos, incluso con sistemas más potentes y sistemas dedicados, como los sensores RGB-D o los sistemas de tiempo de vuelo, además de las tarjetas gráficas específicas al procesamiento de imágenes. Pero no sólo eso, los algoritmos utilizados en visión también han sufrido cambios y se han mejorado a lo largo de los años, y dependiendo de las necesidades que se presenten.

En la literatura, algunos autores clasifican los algoritmos de Visión Computacional en 3 niveles: bajo, intermedio y alto, aunque depende del autor cómo se clasifica cada algoritmo. Para nuestro caso, la clasificación que se considerará es la hecha por E. R. Davies en [10].

- Procesamiento de bajo nivel: en este nivel se presentan las metodologías utilizadas para obtener información de una imagen. Estas metodologías son básicamente las operaciones que pueden aplicarse sobre una imagen para obtener información de ella, sin que esta información sea interpretada. A este nivel pertenecen algoritmos como: filtrado, umbralización, binarización, detección de bordes, flujo óptico.

- Procesamiento de nivel intermedio: en el nivel intermedio se le da una interpretación más precisa de la información obtenida por un algoritmo de bajo nivel. Este nivel lo ocupan algoritmos como: detección de líneas, de círculos, de polígonos, detectores de esquinas, Transformada de Hough.
- Procesamiento de alto nivel: este nivel, también considerado de aplicación, se compone de todas aquellas aplicaciones que permiten interpretar la información procesada en los niveles anteriores, bajo e intermedio. Por ejemplo: empatado de patrones, reconstrucción tridimensional, reconocimiento de patrones, reconocimiento de objetos.

2.1.1. Segmentación

Este término es muy utilizado en Visión por Computadora, incluso se aplica en varias actividades, como reconocimiento de patrones o imágenes biomédicas. La segmentación podríamos definirla, en el campo de Visión Computacional, como la agrupación de regiones de píxeles que tienen características semejantes y que pueden separarse para su procesamiento [14] [34]. Los métodos de segmentación de imágenes son variados, pues van desde la segmentación por color, por umbralización, incluso por bordes.

En [34], se detallan dos principios fundamentales de segmentación que pueden agrupar las técnicas utilizadas para separar píxeles de una imagen, estos dos principios son: por discontinuidad y por similitud. Por similitud involucra a todos aquellos píxeles de una imagen que tienen propiedades similares, como el color; mientras que por discontinuidad se consideran aquellas regiones que pueden estar separadas por algo que las delimita, por ejemplo un borde, pero que contienen píxeles que no necesariamente tienen propiedades similares.

Los métodos de segmentación más utilizados en el campo de la Visión Computacional son los que están basados en similitudes. La segmentación de color es un método relativamente sencillo y que se basa en buscar todos aquellos píxeles que tienen las mismas propiedades de color en el espacio (de color) correspondiente. También se utiliza la segmentación por intensidad, en la cual se separan todos los píxeles que cumplen con una intensidad específica y se ocupa bastante en imágenes en escala de grises que se usan en imágenes biomédicas.

El método de segmentación de una imagen debe adecuarse respecto a las necesidades de un sistema. Para nuestro caso en particular, se utiliza la segmentación por color para detectar la cancha de soccer, ya que es un objeto de color uniforme y el cual podría considerarse el espacio de trabajo al ser en su mayoría de ese color y en donde

se desarrolla la actividad del robot. Sin embargo, es necesario considerar que alrededor de la cancha pueden presentarse objetos de color similar que puedan interferir en la detección de la misma. Una posible solución es definir que la cancha siempre va a estar en la zona inferior de la imagen, al suponer que el robot está dentro de la cancha y que se encuentra mirando hacia ella. Otra posibilidad es definiendo los bordes de la cancha y detectando las líneas que delimitan el área de juego. Esta última como una solución que hace uso de la segmentación por discontinuidad y por similitud.

En competencias pasadas de RoboCup, la segmentación de color también era útil para detectar la pelota y las porterías en la cancha. Empero, para el año 2015, las reglas oficiales de RoboCup se han modificado afectando el color de la pelota y las porterías, pues ahora el color de ambas debe ser blanco, el de la pelota en una proporción del 50% o más, y el de las porterías totalmente, lo cual obliga a cambiar la técnica para detectarlas.

2.1.2. Detección y descripción de características

En Visión Computacional, una característica podría considerarse como una parte de información relevante. Para reconocer objetos en una imagen es necesario conocer algunas de las características que componen a esos objetos. Si bien, por ejemplo, necesitamos reconocer un objeto redondo, es primordial encontrar en la imagen objetos que sean redondos para después determinar si alguno de los que se captan en la imagen es el que se busca en principio. Las características de un círculo pueden ser: radio, diámetro, centro y excentricidad.

La detección de características en una imagen es considerada una operación de procesamiento a bajo nivel. La detección se trata de encontrar aquellas características de una imagen, ya sea, por la forma de los objetos visibles en la imagen, por su intensidad, inclusive por la cantidad de bordes.

La diferencia principal entre detección y descripción es que la primera sólo se encarga de encontrar las características que definen a un objeto o una imagen. En cambio, la segunda se encarga de procesar la información que define a un objeto para determinar que esa información exclusivamente define a ese objeto o imagen.

Detección

La detección de características es el procedimiento realizado sobre una imagen para encontrar algunos detalles que definan tal imagen, por ejemplo: líneas, puntos, esquinas o regiones. Se considera que es una operación de procesamiento de imágenes a bajo nivel porque sólo obtiene información de una imagen sin interpretarla.

Actualmente, las características más buscadas con un detector son: bordes, esquinas y regiones. También se pueden encontrar círculos, centroides u otros puntos de interés.

Existe una buena cantidad de algoritmos utilizados para detectar características. Los más comunes se utilizan para detectar bordes o líneas, ya que son relativamente simples de detectar y con ellas se puede definir una imagen. Un algoritmo común y eficiente es el propuesto por Canny en [9]. La mayoría de algoritmos para extraer bordes se basan en la primera y segunda derivada en una imagen. Esto es, la primera derivada se observa como un cambio brusco de intensidad de un pixel a otro, lo cual hace que haya un cruce por cero en la función de la imagen, estos cambios bruscos se observan comúnmente cuando se atraviesa por un borde [34].

Otros algoritmos para encontrar bordes, y que están basados en la primera derivada son los operadores de Sobel, Prewitt y Roberts. Son matrices bidimensionales que al operarse con la imagen, la resultante es una imagen en la que se observan los bordes únicamente. Dichos operadores son separables, lo que quiere decir que se pueden separar en el producto de una matriz renglón por una matriz columna, y que cumplen con la propiedad de distributividad. También, el gradiente de una imagen permite reconocer en dónde existen variaciones de intensidad, y en donde se encuentren las mayores hay más probabilidades de encontrar un borde. Existen también los algoritmos de segunda derivada, por ejemplo, el operador Laplaciano o el Laplaciano de la Gaussiana [34].

Un algoritmo muy utilizado para detectar esquinas es el de Harris [17], basado en una detección de bordes, pues el cruce de bordes, por lo regular son esquinas.

Para detectar regiones, el método más conocido es el de segmentación, que ya se mencionó anteriormente. Principalmente, las regiones por color son más fáciles de obtener, pero a partir de estas regiones, se pueden obtener los centroides para definir puntos y la distribución de esos puntos pueden describir una imagen.

Descripción

La descripción de características es el paso en el que se extraen las características detectadas para definir una imagen. En este paso, la descripción de una imagen u objeto de ella se realiza con el fin de darle una interpretación a los datos obtenidos por un detector. Por ejemplo, los códigos de cadena sirven para describir una recta a partir de su dirección; también existe la transformada de Hough, que describe líneas a partir de su pendiente y su ordenada al origen, para conocer su orientación.

Existen otros tipo de descriptores más robustos y que son los más utilizados, como son el descriptor SURF (*Speeded Up Robust Features*) [2] y también SIFT (*Scaled-Invariant Features Transform*) [27]. También existen detectores de características para cada uno

de estos algoritmos. Se considera que SURF es más rápido que SIFT, pero SIFT es más robusto a cambios de escala.

La descripción de características de una imagen también depende si las características son líneas o regiones [34]. Una región se puede describir por su forma o por los puntos que la componen, también por su topología y las conexiones de sus píxeles. En cambio, las líneas pueden caracterizarse por su orientación y su longitud, además de los puntos que la forman.

2.2. Modelo *Pinhole*

Es el modelo de lente más común en las cámaras que se comercializan en el mercado. El modelo *pinhole* es un modelo de proyección de una escena a una imagen que se encuentra en la mayoría de las cámaras. Este modelo de proyección está dado por un punto (centro focal) a través del cual atraviesa la escena para proyectarse en el plano de la imagen. Este modelo se puede ver en la figura 2.1. Supongamos que queremos proyectar un punto $P(X, Y, Z)$ del mundo real como un punto $p(x, y)$ en el plano de la imagen del sensor utilizado. La luz que refleja el objeto pasa a través del centro focal (C) en línea recta al plano de la imagen. Este plano tiene sus propias coordenadas. El plano focal es el plano que contiene a C y que es paralelo al plano de la imagen. La distancia del plano de la imagen al plano focal está dada por f .

El modelo *pinhole* será el que se utilice a lo largo de este trabajo. Existen otros modelos de lentes que modifican los parámetros existentes en una cámara, como puede ser el modelo *fisheye* o de gran angular, que permite visualizar una escena con un amplio ángulo de visión (casi hasta 180°), pero que agrega distorsión a la escena.

2.3. Visión Estereoscópica

Actualmente, la visión estereoscópica (o también denominada estéreo) es quizá uno de los problemas que más se han estudiado en el campo de Visión Computacional en los últimos años, en el cual se utilizan dos sensores de visión para obtener información tridimensional en un entorno. Con ayuda de la visión estéreo, se logra captar la información de profundidad en una escena, útil para conocer posiciones de objetos relativas a la posición actual de los sensores gracias a que los datos de cada una de las dos imágenes obtenidas por los sensores tienen una relación entre sí. Cuando se logra que dos imágenes con diferentes orientaciones capturen un punto en común y conociendo los parámetros de las cámaras, así como la posición de una respecto a la otra, es posible aproximar la

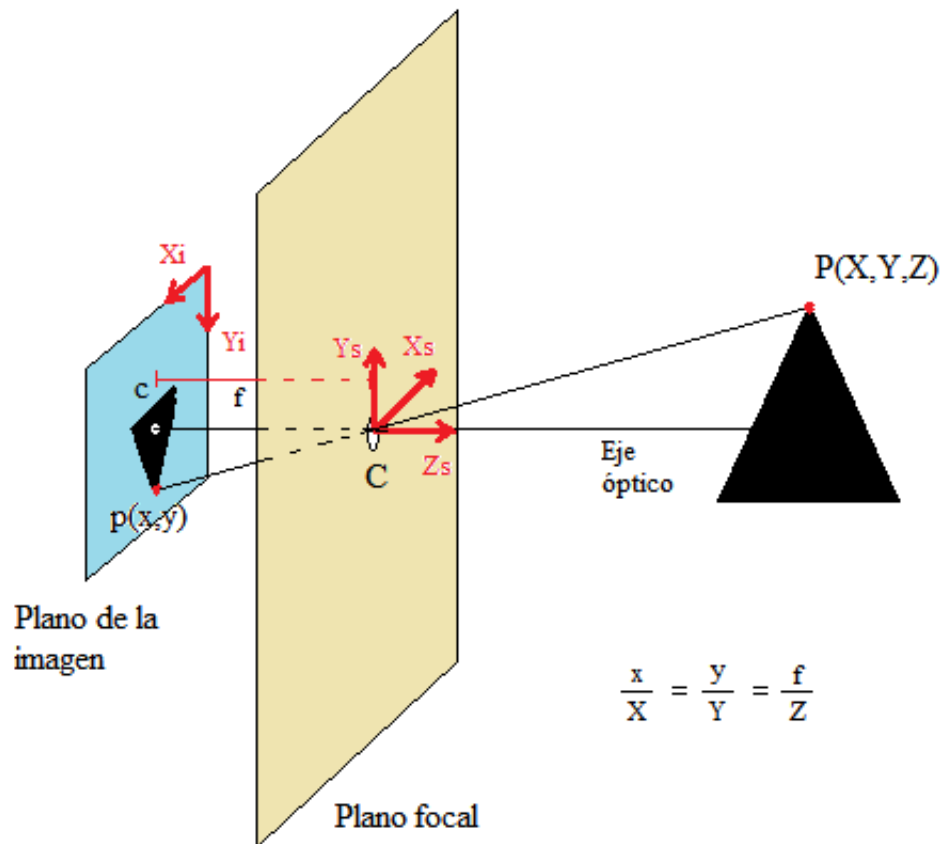


FIGURA 2.1: Proyección de perspectiva con modelo *pinhole*

profundidad a la que se encuentra el punto en las dos cámaras y triangular la posición de la primera cámara con la segunda y el punto común en cuestión. De este modo, se puede obtener información de ese punto común, es decir la profundidad que existe de cualquiera de las dos imágenes obtenidas hacia él. Cabe destacar que ese punto debe aparecer en ambas imágenes.

Este tipo de visión está basado en la percepción visual humana. Con ayuda de visión binocular, los seres humanos podemos percibir profundidad en objetos presentes en el entorno. Esto se debe a que la imagen percibida a través de un ojo tiene una relación con la imagen percibida por el otro, de manera que se está percibiendo el mismo entorno desde perspectivas diferentes, y por ende, se puede observar cierto volumen en objetos vistos.

Un concepto básico para conocer el funcionamiento de la visión estéreo es el de **geometría epipolar** [47]:

Sea I la imagen de un entorno cualquiera e I' una imagen del mismo entorno pero desde una perspectiva distinta. Sea el punto X un punto en el espacio 3D. Sean C y C' los

centros focales de las imágenes correspondientes. Se define como *línea base* a la línea que contiene a los centros focales. Los puntos e y e' son las intersecciones de la línea base con las imágenes I e I' , respectivamente, a éstos se les llama *epipolos*. El punto x en la imagen I es la proyección del punto X en dicha imagen, del mismo modo sucede con x' respecto de I' . Se le denomina *línea epipolar* a la línea que contiene al epipolo y la proyección del punto del espacio en cuestión. En la figura 2.2 l corresponde a la línea epipolar de la imagen I y l' es la línea epipolar de la imagen I' . Por último, por medio de la triangulación entre los centros focales de las imágenes y el punto en el espacio en cuestión, se puede obtener un plano que contenga a estos 3 puntos, además de contener la línea base y las líneas epipolares. A este plano se le denomina *plano epipolar*.

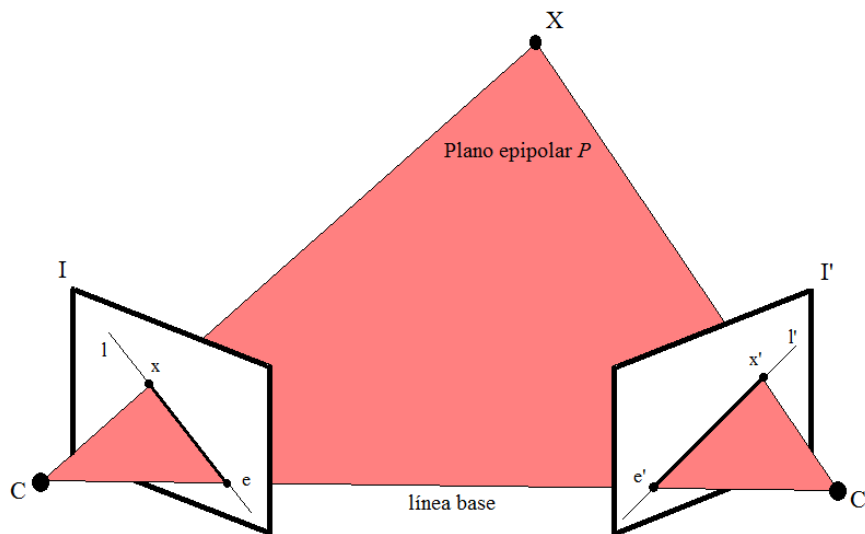


FIGURA 2.2: Modelo de geometría epipolar.

Conocer la geometría epipolar del sistema de visión estereoscópica permite realizar una serie de restricciones con las cuales la búsqueda de correspondencia entre dos imágenes es más sencilla y eficiente. Estas restricciones básicamente reducen el espacio de búsqueda de un pixel, o un bloque de ellos, de una imagen a otra. Dichas restricciones se tratarán más adelante.

La geometría epipolar de un sistema estéreo también depende de la orientación de las cámaras. Si bien, las dos imágenes capturadas deben ser de una escena común, la perspectiva de las cámaras puede marcar cierta diferencia al realizar la triangulación de puntos. Básicamente, pueden existir dos alternativas, que los planos de las imágenes sean coplanares (ejes ópticos paralelos); o que no lo sean (ejes ópticos convergentes). En la figura 2.3 se pueden observar los posibles casos. La figura 2.3(a) muestra que la imagen

I y la imagen I' pertenecen a un mismo plano R , en cambio, la figura 2.3(b) muestra que los planos no necesariamente deben ser coplanares. Para fines prácticos, la geometría de imágenes con ejes ópticos paralelos facilita el cálculo de las correspondencias entre imágenes. En el capítulo 3 se analizan estos casos a detalle.

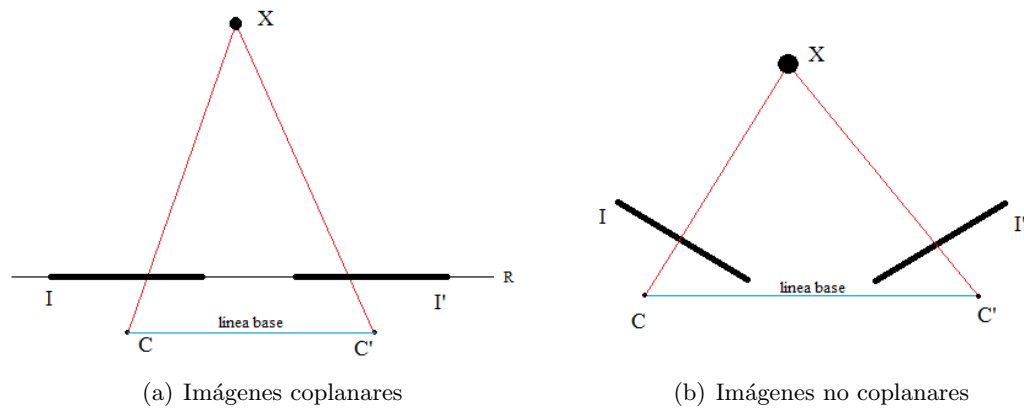


FIGURA 2.3: Orientación de las imágenes

2.4. Algoritmos en Visión Estereoscópica

La visión estereoscópica, como ya se ha mencionado, es la técnica de procesar información de visión a través de un sistema de visión binocular (con dos sensores de visión) y que hace uso de la geometría epipolar para lograr obtener información en 3 dimensiones de un entorno.

El problema de hacer el registro de dos imágenes conlleva toda una secuencia de pasos o procedimientos para llegar al objetivo principal del empatado estéreo, que es el de obtener el mapa de disparidad entre dos imágenes captadas en una escena.

Mapa de disparidad

En primer lugar, en visión estereoscópica, la disparidad es la variación de la posición de un punto (x_1, y_1) de la primera imagen con respecto a las coordenadas de un segundo punto en la segunda imagen (x_1', y_1') que tiene correspondencia con el primero. Esta variación depende de la profundidad a la que se encuentra el punto observado en las dos imágenes, es decir, cuando la variación es ligera implica que la profundidad del punto es significativa, mientras que si la variación aumenta, la profundidad es cada vez menor. En pocas palabras, la variación en la posición de los puntos es inversamente proporcional a la profundidad del punto en cuestión.

El mapa de disparidad es la representación de la información de disparidad de cada uno de los puntos (píxeles) que corresponden en ambas imágenes. Regularmente, el mapa de disparidad se representa como una imagen en escala de grises cuyos valores indican la profundidad de los puntos. Para puntos relativamente cercanos se utilizan píxeles claros y para puntos más lejanos, se utilizan píxeles oscuros.

Para obtener el mapa de disparidad se han desarrollado métodos de empatado que permiten relacionar las imágenes estéreo para calcular sus disparidades. Los métodos de empatado están divididos en dos grupos: métodos locales y métodos globales; los locales se basan en obtener características específicas de una escena que correspondan en las dos imágenes y a partir de ellas realizar el registro; por otra parte, los globales utilizan mayor cantidad de características de la escena, lo que permite obtener un mapa de disparidad aún más amplio y más preciso.

Los métodos locales suelen ser más eficientes, sin embargo, tienen problemas al detectar regiones ambiguas en las imágenes, como las regiones de oclusión o regiones con textura uniforme. Los métodos globales son más robustos a este tipo de problemas, aunque implican un mayor cómputo.

A continuación, se mencionan algunos de los métodos de cada grupo más comunes en la implementación de sistemas de visión estéreo [7].

2.4.1. Métodos de empatado

Métodos locales

- **Empatado por bloques:** Ubica un punto a empatar en la primera imagen sobre la segunda, y realiza la búsqueda a través de bloques, es posible limitar la búsqueda en una pequeña región al conocer la geometría de las imágenes.
- **Optimización basada en el gradiente:** Se determinan pequeñas disparidades locales entre dos imágenes mediante una ecuación diferencial, relacionada al movimiento y al brillo de las imágenes. Se asume que la disparidad existe dentro de una pequeña región de píxeles, pues la geometría de las imágenes se conoce previamente.
- **Empatado por características:** Es un poco más robusto que los anteriores, ya que los métodos de empatado local anteriores tienen el inconveniente de ser sensibles a discontinuidades de profundidad y a regiones de textura uniforme. Esto se debe a que los métodos basados en características limitan la región de búsqueda

a características específicas, aunque limita la densidad de los puntos a los cuales se obtienen sus profundidades.

En [7] se menciona que en los 80's, los métodos basados en características tuvieron un auge debido a su eficiencia. Sin embargo, este método fue perdiendo popularidad, pues las necesidades de las aplicaciones que requerían mapas de profundidad más densos eran mayores, aunado a que los métodos de empatao de bloques tuvieron mejoras.

Métodos globales

- **Programación dinámica:** La ventaja de este método radica en su naturaleza, pues es un método matemático que se encarga de reducir la complejidad computacional descomponiendo un problema en pequeños subproblemas simples. Se utiliza sobre todo para minimizar el cálculo de la función de costo para el empatao de imágenes. Empero, su complejidad temporal es de $O(N^4)$, donde N es el número de pixeles a procesar.
- **Gráficas de corte:** Las gráficas de corte ofrecen al problema de empatao estéreo un método de aplicar la restricción de coherencia bidimensional al encontrar el flujo máximo en una gráfica [7].

El flujo en la gráfica se puede observar como la función de costo de los pixeles vecinos al que se está procesando en ese momento. Cada pixel representaría un nodo, y el flujo es la función de costo al pixel correspondiente. Entonces, resolviendo el problema de corte mínimo de la gráfica, se puede encontrar la trayectoria que debe seguir un pixel para encontrar el que le corresponde en la segunda imagen. La disparidad asociada al corte mínimo no sólo es local, en una trayectoria, sino que también es globalmente consistente con la imagen.

- **Belief Propagation:** Es un método para la resolución de problemas de inferencia a partir del paso de mensajes a través de una red. Fue desarrollado por Judea Pearl en [35] e implementado en visión estéreo por Sun *et al.* en [41] utilizando redes de Markov. *Belief Propagation* es un algoritmo de inferencia iterativo que calcula la distribución de un nodo que no es conocido en la red utilizando suma de productos.

Existe una variante del método de empatao por bloques que trata de ser una combinación entre un método local y un método global. Tal es el caso de *Semiglobal Matching* [18].

2.4.2. Función de costo

Una función de costo es aquella que se utiliza para conocer la similaridad entre la imagen de referencia y la imagen sensada. Actualmente, existe un gran número de funciones de costo como criterios de similaridad para el registro de imágenes estéreo. Algunas funciones de costo pueden ser más eficientes que otras, pero su uso también depende de la necesidad del sistema de visión estereoscópica. A continuación, se mencionarán algunas funciones de costo más utilizadas y que han presentado buenos resultados [19].

- **Suma de diferencias absolutas:** se utiliza comúnmente en métodos de empatado local y compara una ventana de píxeles de la imagen de referencia con otra ventana de la imagen sensada. En esta ventana se calcula la intensidad de cada píxel y se suma para obtener un valor
- **Correlación Cruzada Normalizada:** verifica la correspondencia de diferentes bloques de cada imagen por medio de la correlación cruzada, normalizando y comparándolos. El mayor valor obtenido será en el que la correlación sea mayor para el bloque más parecido.
- **Información Mutua:** este procedimiento se utiliza para comparar la información que proporcionan dos imágenes y es una medida que indica qué tan parecidas pueden ser ambas, haciendo uso de probabilidad [43].

2.5. Robótica

La Robótica es considerada “la disciplina de la ingeniería que se encarga del estudio de los robots” [46]. Según la definición adoptada por la (*Robotic Industries Association* (RIA) de Estados Unidos, y usada por algunos autores [46] [36] [29], un *robot* es “un manipulador reprogramable multifuncional diseñado para mover materiales, partes o dispositivos especializados, a través de movimientos programados, para la realización de diversas tareas”. Si bien, en [36] se menciona que un manipulador es básicamente un brazo y éste es la parte principal de un robot, entonces una definición más general de *robot* es “una máquina reprogramable multifuncional diseñada para manipular materiales, partes, herramientas o dispositivos...”.

La Robótica, como la conocemos actualmente, podría definirse como una rama de la tecnología que se encarga de conjuntar los conocimientos de distintas disciplinas, en software y hardware, aplicadas a robots. En el caso del software, se tiene la Cibernética, el Procesamiento de Imágenes, Procesamiento del Lenguaje Natural, Inteligencia Artificial,

etc. Por otro lado, en el caso del hardware se complementan la Mecánica, la Física, la Electrónica, el Diseño, entre otras. Todas ellas trabajando en conjunto con un mismo fin.

En general, los robots se pueden clasificar en diferentes tipos, de los cuales algunos se mencionan a continuación:

- Móviles

Los robots móviles son aquellos que se apoyan de dispositivos de desplazamiento, como son las llantas y que suelen moverse en dos dimensiones.

- Humanoides

Son robots que simulan tener forma humana y cuya estructura es muy similar. Su funcionamiento se basa en el movimiento natural de las articulaciones de una persona común. Tienen dos extremidades inferiores (piernas) y dos superiores (brazos)

- Insectoides

O también llamados zoomórficos o bioinspirados, son aquellos que tienen forma de insecto, están basados en los movimientos de seres vivos, distintos al ser humano, y tratan de simular el desplazamiento de una animal o insecto. Por ejemplo, los robots cuadrúpedos, los hexápodos, los que tienen desplazamiento de gusano, etc.

- Híbridos

Incorporan movimientos basados en todos los anteriores.

2.6. Estado del arte

Actualmente, la visión computacional aplicada en la Robótica ha tenido un gran auge debido a que la investigación en este ámbito no se ha detenido, pues a nivel mundial existen competencias de robots cuyo fin es ese, fomentar la investigación. Si bien la RoboCup comenzó con el objetivo de crear un equipo de robots de soccer, el campo de investigación es mucho más amplio que eso, a tal grado que se abrieron otras categorías. Incluso existen otras competencias, como el *DARPA Robotics Challenge* en donde los robots deben cumplir objetivos específicos y que van cambiando año con año, esta variedad de actividades permite no centrarse en un problema específico. De igual manera, la NASA realiza competencias para robots exploradores. Cabe mencionar que el más grande desarrollo actual en robots, es el robot ASIMO, creado por Honda, que no necesariamente se utiliza en competencias pero que ha tenido seguimiento a lo largo de los últimos años.

En la mayoría de competencias, los robots se introducen en ambientes totalmente dinámicos con lo cual la visión es un factor muy importante, pero dado que una simple imagen no proporciona información tridimensional sumamente útil para trasladarse sobre el espacio, se han desarrollado técnicas para obtener esta información. Algunos métodos de visión monocular, como el método de Odometría Visual presentado en [33], donde se puede estimar el movimiento de un sistema basado en una cámara o, inclusive, un sistema estéreo; o el mostrado por [31], que desarrolla un método de localización con el desplazamiento de una cámara, obteniendo características con el detector de Harris [17] y utilizando el método de correlación cruzada normalizada de media cero (ZNCC, por sus siglas en inglés) para realizar el empatao. Pero también estos métodos pueden utilizarse para la reconstrucción de escenas en 3D, pues la relación de varias imágenes permitiría calcular la geometría epipolar.

En lo que a este trabajo respecta, el estudio de la visión estereoscópica, en Computación, se remonta a la década de 1960 por Bela Julesz, en [22], donde se hace un análisis de la percepción de profundidad en imágenes estereoscópicas y cómo se pueden obtener patrones que empaten en ambas imágenes.

En la década de 1970, este problema fue un poco más estudiado, al punto que se desarrollaron algoritmos para determinar disparidad entre dos imágenes, por ejemplo, en [28] se explica un algoritmo cooperativo que aplica la utilización de las restricciones de unicidad y de continuidad (que se mencionan más adelante en este trabajo escrito) para procesar estereogramas y obtener lo que se puede ver a través de ellos.

A partir de los años 80's, el problema de visión estéreo comenzó a ser abarcado más a detalle. En [1] se menciona el paradigma del cómputo estéreo, en el que se presenta una metodología para la obtención de la disparidad entre dos imágenes, que va desde la adquisición de las imágenes hasta el proceso de interpolación para obtener la disparidad. En [11] se hace una recopilación de los algoritmos más importantes hasta esa época, diviendo el proceso de estereocopia en 3 etapas: Preprocesamiento, Correspondencia y Reconstrucción de la profundidad, donde para cada etapa existen diferentes algoritmos. A partir de esta década comenzaron a surgir implementaciones de sistemas estéreo, tal es el caso de la presentada por Grimson en [16] donde se hace la implementación de lo que fue presentado por Marr y Poggio en [28] algunos años antes.

Desde entonces, ha surgido una buena cantidad de soluciones para el procesamiento de visión estereoscópica, sobre todo para resolver los problemas más comunes, como similitud y oclusión. Además, se comenzaron a desarrollar los primeros sistemas de visión estéreo por hardware, por ejemplo, Kanade *et al.* [23] muestran una solución al problema de similitud utilizando Suma de Diferencias Absolutas. En este mismo trabajo se desarrolló un sistema de video con el cual se podía obtener un mapa de disparidad.

También, Konolige [25] propuso un sistema de visión en hardware para obtener un mapa de disparidad que, aunque era de baja resolución (160×120) obtuvo buenos resultados, implementando algoritmos básicos sobre su sistema. En 1996, Benschhair *et al.* [3] presentaron un algoritmo de empatado estéreo basado en programación dinámica, que es muy utilizada en la actualidad.

Como un trabajo de recopilación de algoritmos de visión estéreo, en [38] se presenta una taxonomía de algunos de ellos. Sin embargo, existe una base de datos, que sirve como *benchmark*, que mantiene información de esa recopilación y ha ayudado a que otros investigadores conozcan y traten de mejorar los algoritmos. En esta misma base se proporcionan los datos necesarios para probar el desempeño de un algoritmo y se presentan los resultados obtenidos de estas pruebas. Se encuentra disponible a través del sitio web <http://vision.middlebury.edu/stereo/>.

Todos (o casi todos) los algoritmos que se encuentran referenciados en la base de datos de Scharstein y Szeliski son para obtener mapas de disparidad densos. Algunos son de los más utilizados en estos días, como el algoritmo ELAS (*Efficient Large-Scale Stereo Matching*) [13], que utiliza modelo generativo [40] para el empatado de puntos e implementa puntos de soporte que se utilizan para hacerlo más robusto y, aunque este método es de empatado local, utiliza una mayor cantidad de puntos de soporte ubicados en las esquinas de la imagen para obtener mayor densidad en la disparidad; otro de los algoritmos de empatado que se encuentran en la base de datos es SGM (*Semiglobal Matching*) [18] que está basado en la búsqueda de similitudes de una imagen base en una imagen de empatado, haciendo uso de Información Mutua como función de costo. Existe una variante de este último algoritmo, *Semiglobal Block Matching*, que utiliza una métrica de subpíxeles propuesta por Birchfield y Tomasi en [4], en lugar de Información Mutua, la cual permite calcular el empatado con más precisión; además, hace un empatado por bloques, que lo hace más veloz a comparación de SGM.

Existen algunos algoritmos que tienen implementación en las bibliotecas de OpenCV. Tal es el caso del algoritmo *Block Matching* desarrollado por Konolige en [25]. Hace empatado por bloques de píxeles utilizando Laplaciano de la Gaussiana y utilizando correlación de área con intensidades normalizadas, dicha correlación se realiza con la suma y el cuadrado de diferencias absolutas. Uno más de los algoritmos implementados en OpenCV es el ya mencionado SGM, y su variante SGBM. Por último, el algoritmo presentado por Kosov *et al.* en [26] que ejecuta un empatado utilizando métodos variacionales, que son muy útiles para flujo óptico, gracias a su precisión.

A finales de la primera década del 2000, comenzó a desarrollarse el sistema que hoy conocemos como Kinect, que es un sistema llamado de *visión estéreo activa*, pues utiliza dos sensores, el primero es una cámara de video VGA a una resolución de 640×480 ;

y el segundo es un sensor infrarrojo activo que se compone de un proyector de luz infrarroja sobre el ambiente y éste es captado por un sensor CMOS monocromático, para obtener información 3D del ambiente. Kinect revolucionó la manera en que la Visión Computacional es percibida. A partir de que salió al mercado, Kinect comenzó a utilizarse para obtener información 3D del ambiente y se incrementó la investigación en ese sentido, a pesar de que los sistemas estéreo permitían algo similar. El punto más importante fue la precisión con la que el sensor de Kinect trabaja, pues el error de profundidad es milimétrico. Actualmente está en el mercado la versión 2 de Kinect, que presenta mucho mayor precisión al utilizar un sensor de tiempo de vuelo para información de profundidad y una cámara HD con resolución de 1920×1080 .

2.7. Discusión

Dados los algoritmos presentados, la mejor opción dependerá de las necesidades del robot, ya que el objetivo principal es incorporar el algoritmo sobre ese robot. En principio, siguiendo las reglas para competencias de soccer en RoboCup para robots humanoides, categoría en la que participa el robot NimbRo-OP, no es posible utilizar un sistema como Kinect, por lo que el sistema debe estar compuesto de sólo sensores pasivos, por lo tanto son dos cámaras las que se usarán.

Las cámaras que se tienen contempladas para este trabajo son cámaras que tienen una resolución de 720p (1280×720), las cuales se consideran HD, esto con la intención de que se obtenga información lo suficientemente precisa para lograr buenos resultados.

Por otra parte, el algoritmo a utilizar debe poder ejecutarse en tiempo real. Se decidió utilizar el algoritmo SGBM porque implementa un empatado semiglobal, que permite obtener un mapa de disparidad suficientemente denso para obtener información en 3D, además de sacar provecho de las capacidades de la función de costo, que lo hace más preciso.

Capítulo 3

Marco teórico

Para entender con mayor detalle el funcionamiento de un sistema estéreo, y en particular, el desarrollado en este trabajo, en este capítulo se detallarán varios conceptos que se usarán. En principio, un sistema de visión estereoscópica consta de diferentes etapas. Dependiendo de la estructura del sistema y del tipo de imágenes capturadas por las cámaras que lo componen pueden existir variantes entre dichas etapas, aunque la base del sistema siempre es la misma.

Con base en el tipo de cámaras utilizadas, se puede realizar o no una etapa de preprocesamiento que permita obtener mejores resultados al momento de ejecutar el procesamiento principal del sistema de visión estéreo, esto es, cuando se realiza la correspondencia de las imágenes se pueden obtener mejores puntos característicos si se tienen imágenes con una resolución alta o con mayor nitidez.

La etapa de procesamiento del sistema de visión estéreo es en la que se realiza el registro de imágenes para obtener la disparidad entre ellas respecto a la correspondencia de los diferentes puntos analizados en cada una. Es decir, se hace un empatado, o *matching*, de las características obtenidas, ya sean puntos, líneas, o regiones, en las dos imágenes y se procede a revisar qué tanto varían de una imagen a otra. Cabe mencionar que para esto es necesario conocer los parámetros intrínsecos y extrínsecos de las cámaras utilizadas. Estos parámetros tienen información acerca del sensor y de las variaciones que produce el tipo de lente utilizado en la cámara.

Por último, una etapa de postprocesamiento puede incluirse para refinar la información obtenida en el registro de imágenes. Además, es en esta etapa donde se realizará la proyección de las imágenes 2D a 3D que contiene las correspondencias de las imágenes en forma de puntos en el espacio, lo que llamamos nube de puntos. Posterior a esta etapa, cualquier procesamiento de una nube de puntos podría aplicarse.

El objetivo principal de la visión estereoscópica es obtener una medida de rangos, ésta contiene información de la profundidad a la que se puede encontrar un objeto tomando como referencia el sistema de coordenadas de las cámaras estéreo, basada en la proyección de los objetos en dos o más imágenes. El problema más importante a resolver en este tipo de visión es realizar la correspondencia de las múltiples imágenes. Esta medida de rangos puede calcularse conociendo la geometría de las imágenes, como se menciona en [25].

Los métodos de empatado pueden clasificarse en dos grandes grupos:

- Locales: realiza el empatado de pequeñas regiones de una imagen en otra, basándose en las características intrínsecas de la región. Además, éstos mismos pueden clasificarse por la manera de ejecutar el empatado, pues algunos lo realizan por características discretas entre imágenes, y otros por correlación de áreas pequeñas.
- Globales: aunado a lo que hacen los métodos locales, también se consideran algunas restricciones físicas, como puede ser la continuidad de la superficie.

En cuanto a los métodos locales, en el caso de los que están basados en características, estos son robustos al hacer la correspondencia, dado que pueden realizar el empatado de manera rápida y precisa al compensar las diferencias en los puntos de vista y en las cámaras, sin embargo, el proceso de extraer las características es muy costoso. En cambio, los métodos que están basados en correlación de área realizan la búsqueda de un segmento de una imagen en la otra, siendo el costo del algoritmo inversamente proporcional al tamaño del segmento de la imagen, es decir, si el tamaño del segmento es pequeño, el costo de la correlación de área es mayor, pero si el tamaño del segmento se va incrementando, entonces el costo del proceso de correlación disminuye, aunque cuanto más grande es el segmento de imagen, mayor la relación señal a ruido (SNR). [25]

Un método común de correlación de área tiene 5 pasos, de acuerdo con [25]:

- Corrección de geometría: eliminar distorsión en las imágenes de entrada.
- Transformación de imagen: utilizando un operador local aplicado sobre cada pixel para convertirlo a una forma más adecuada.
- Correlación de área: cada segmento de área de la imagen de referencia se compara con áreas de la imagen sensada.
- Extracción de extremos: se calcula el valor extremo de la correlación en cada pixel, lo cual producirá una imagen de disparidad en la que cada valor del pixel es la disparidad entre las imágenes izquierda y derecha, donde mayor coincidencia existe.

- Post-filtrado: en donde uno o más filtros eliminan el ruido de la imagen de disparidad.

3.1. Registro de imágenes 2D

El registro de imágenes es el proceso de determinar la correspondencia punto a punto entre dos imágenes de una misma escena. Gracias al registro de dos imágenes, la relación de información entre imágenes puede obtenerse, además de poder determinar el mapa de profundidad y los cambios que pueden existir en la escena y, al final, que los objetos presentes en la escena puedan ser reconocidos. [15]

Tomando en cuenta la nomenclatura que usa [15], y con la que se trabajará a lo largo de este escrito, para lograr el registro de imágenes se utilizan:

- Imagen de referencia (imagen fuente).
- Imagen sensada (imagen objetivo).
- Función de transformación, que mapea la imagen sensada a la imagen de referencia.

La imagen de referencia, en la mayoría de los casos, es la imagen izquierda de un sistema de visión estéreo cuyas cámaras se encuentran separadas horizontalmente. Esto debe ser irrelevante para el cálculo siempre y cuando se mantenga una relación fija entre las cámaras.

Los pasos para lograr el registro de imágenes, mencionados en [15], son:

Preprocesamiento: Preparar la imagen para obtener características de ella. Escalar, eliminar ruido y/o segmentar. Escalar la imagen sensada a la imagen de referencia, aplicar un filtro suavizador para eliminar el ruido y/o segmentar para extraer características.

Selección de características: Se seleccionan el tipo y un número de características para obtener una correspondencia entre las características en ambas imágenes. Se puede obtener una función de transformación para realizar un remuestreo de la imagen sensada a la geometría de la imagen de referencia. Algunas características que se pueden obtener son esquinas, líneas, curvas y regiones. El tipo de características seleccionadas dependen del tipo de registro. Comúnmente, para registro de imágenes en 2-D se utilizan líneas y para el registro de imágenes en 3-D se utilizan superficies y regiones.

Correspondencia de características: Se seleccionan características de ambas imágenes para determinar si existe correspondencia entre ellas. Los puntos de correspondencia se utilizan para determinar los parámetros de transformación.

Función de transformación: Cuando se conocen las coordenadas de los puntos de correspondencia en ambas imágenes, se calcula una función de transformación para el remuestreo de la imagen sensada respecto a la de referencia. Se debe aplicar una función de transformación que permita adaptar geoméricamente ambas imágenes.

Remuestreo: Una vez conociendo la función de transformación, la imagen sensada se remuestrea respecto a la de referencia. Esto permite unir información o captar cambios en la escena.

Como se menciona en [15], esta metodología se sigue para sistemas de registro de imágenes en 2D, utilizados ampliamente en imágenes aéreas y satelitales.

3.2. Preprocesamiento de imágenes

La etapa de preprocesamiento del registro de imágenes es aquella donde se preparan las imágenes para su posterior procesamiento y obtener los mejores resultados posibles. Cuando implementamos un preprocesamiento a una imagen nos referimos a que aplicamos métodos de mejora sobre dicha imagen para adquirir detalles de ella, como pueden ser puntos específicos, formas, regiones, etc. En algunos casos, esta etapa de preprocesamiento puede no ser necesaria, debido a la precisión de los sensores de captura de imágenes utilizados, por ejemplo cámaras de alta resolución o HD.

Una manera de mejorar las imágenes es aplicando filtros que permitan eliminar el ruido que afecta a dichas imágenes. El ruido en una imagen se presenta como una señal indeseada que afecta la percepción de la imagen.

Para este trabajo, en esta misma etapa se hará uso del reescalamiento de una imagen. Al considerar que el sistema de visión estéreo implementado en el robot humanoide debe funcionar en tiempo real, el tamaño de las imágenes procesadas es un factor determinante para la rapidez con la que se ejecuta tal sistema. Si procesamos imágenes de alta resolución, esto aumenta la cantidad de datos a procesar y, por consecuencia, el tiempo de procesamiento de los algoritmos necesarios en el sistema es mayor. Para resolverlo se propone reescalar las imágenes obtenidas a un tamaño más pequeño, el proceso de reescalar la imagen sin que pierda características conlleva dos operaciones sobre la imagen: decimación e interpolación. Estos dos términos se mencionarán en este capítulo, sin embargo, no se profundizarán a detalle. Para una mejor revisión y su análisis matemático, referirse a [37].

3.2.1. Parámetros de una cámara

Las cámaras son los sensores con los que se capta una imagen para realizar procesamiento digital sobre ella. Al momento de capturar una imagen con una cámara, existen diferentes factores que deben considerarse. Uno de ellos, por ejemplo, es el tipo de lente utilizado. Dependiendo del lente, el modelo de la cámara puede variar y la imagen puede presentar algún tipo de alteración indeseada. Otros factores que pueden existir son los parámetros que definen la geometría de la cámara respecto a un sistema de referencia, que bien puede ser el sistema de coordenadas espacial o del entorno. Estos parámetros contienen detalles acerca de la posición de la cámara y su orientación en el sistema de coordenadas definido [34].

Los parámetros de una cámara se usan para obtener la relación entre el sistema de referencia de una imagen con el sistema de referencia del mundo. Estos parámetros pueden obtenerse realizando un proceso de calibración. Para realizar dicha calibración, por lo regular se utiliza un patrón de cuadros negros y blancos (figura 3.1) para encontrar las esquinas y su ubicación en coordenadas de la imagen, o bien, existen patrones que utilizan puntos negros sobre fondo blanco, con el fin de encontrar la posición de los puntos en la imagen. Conociendo las dimensiones de las figuras en el patrón, y que las mismas figuras se ubican en una posición conocida, entonces se pueden determinar los parámetros de la cámara. Los parámetros pueden ser intrínsecos y extrínsecos.



FIGURA 3.1: Patrón de cuadros para calibración de una cámara

Existen diferentes métodos de calibración, por ejemplo, el método de Tsai [42], el cual está basado en cámaras de modelo *pinhole* y es utilizado para calcular los parámetros de la cámara y la distorsión que puede presentar el lente que utiliza.

Parámetros intrínsecos

Los parámetros intrínsecos son las características internas de una cámara que por lo regular son fijas, y permiten relacionar su sistema de coordenadas de imagen (2D) con el sistema de coordenadas de la cámara (3D).

b : distancia principal.

k_u, k_v : número efectivo de píxeles por unidad de longitud horizontal y vertical, respectivamente.

(u_0, v_0) : coordenadas del punto principal en el plano de la imagen.

θ : ángulo de inclinación de píxeles.

Actualmente, la mayoría de sensores de imagen tienen $\theta = 90^\circ$ y píxeles de forma cuadrada con $k_u = k_v$.

Parámetros extrínsecos

Los parámetros extrínsecos de una cámara son aquellos que expresan su posición y orientación respecto al sistema de referencia del mundo. Esto implica que se obtiene la relación entre el sistema de referencia de la cámara con respecto al sistema de referencia del mundo.

Para fines prácticos, la matriz R y el vector t son los que describen la orientación y la posición de la cámara en el mundo, respectivamente. [47]

En visión estéreo, la matriz esencial es aquella que contiene información de los parámetros extrínsecos de las dos cámaras con modelo *pinhole* que observan una misma escena desde distintos puntos de vista. La matriz fundamental es una generalización de la matriz esencial que contiene información acerca de los parámetros intrínsecos. [44]

3.2.2. Distorsión

La distorsión en imágenes es el efecto producido por los sistemas de lentes reales, en el que los puntos de una escena en las coordenadas de la imagen pueden desviarse con respecto a la imagen percibida naturalmente [44]. Este efecto está representado por el siguiente modelo:

$${}^I p_d = (1 + k_1 r^2 + k_3 r^4 + k_5 r^6) {}^I p + d_t \quad (3.1)$$

En la cual:

$${}^I p = (\hat{u}, \hat{v})^T$$

$$r^2 = \hat{u}^2 + \hat{v}^2$$

$$d_t = \begin{bmatrix} 2k_2\hat{u}\hat{v} + k_4(r^2 + 2\hat{u}^2) \\ k_2(r^2 + 2\hat{v}^2) + 2k_4\hat{u}\hat{v} \end{bmatrix}$$

$[k_1, k_2, k_3, k_4, k_5]$: coeficientes de distorsión.

Existen dos tipos de distorsiones: la distorsión radial y la distorsión tangencial. En el modelo 3.1, los coeficientes $[k_1, k_3, k_5]$ corresponden a la distorsión radial, mientras que los coeficientes $[k_2, k_4]$ a la distorsión tangencial.

La distorsión radial se presenta como el efecto de hacer curvas aquellas líneas de la escena real en la imagen. También se le conoce como efecto barril. Las líneas rectas que cruzan el eje óptico siguen apareciendo rectas en la imagen distorsionada, pero la distancia de un punto en la imagen al punto principal varía de la distancia esperada [44]. Un ejemplo de este tipo de distorsión se observa en la figura 3.2.

El modelo de distorsión radial es el siguiente:

$$\begin{aligned} x_c &= x(1 + k_1r^2 + k_3r^4 + k_5r^6) \\ y_c &= y(1 + k_1r^2 + k_3r^4 + k_5r^6) \end{aligned}$$

Donde:

(x, y) : son las coordenadas de un punto p .

(x_c, y_c) : son las coordenadas del punto p corregidas, es decir removiendo la distorsión.



(a) Imagen original.

(b) Imagen con distorsión radial.

FIGURA 3.2: Ejemplo de distorsión radial

Por otra parte, la distorsión tangencial se observa cuando el lente captura una imagen que no está perfectamente paralela al plano de la imagen. Es decir, cuando observamos una línea recta en el entorno real que cruza por el eje óptico de la cámara, dicha línea se observa doblada en la imagen, en una dirección diferente a la real [44]. La distorsión tangencial puede verse en la figura 3.3. Para este tipo de distorsión, el modelo es el

siguiente:

$$x_c = x + [2k_2xy + k_4(r^2 + 2x^2)]$$

$$y_c = y + [k_2(r^2 + 2y^2) + 2k_4xy]$$



(a) Imagen original.

(b) Imagen con distorsión tangencial.

FIGURA 3.3: Ejemplo de distorsión tangencial

Ahora bien, el efecto de distorsión afecta la precisión de la información obtenida del entorno. Para ello es necesario aplicar la corrección de la distorsión como procedimiento del preprocesamiento de las imágenes estéreo. Para determinar los valores de distorsión de la imagen en la cámara se utiliza el patrón de cuadros blancos y negros.

3.2.3. Reescalamiento

El reescalamiento de una imagen implica dos operaciones importantes, que son la decimación y la interpolación. La primera sirve para reducir el número de muestras, y la segunda sirve para incrementarlas. A continuación se mencionará a groso modo cómo funciona cada una de ellas.

a) Decimación

La decimación, o también llamada submuestreo, es la reducción de la tasa de muestreo de una señal [37]. Decimar una imagen implica reducir la cantidad de muestras (píxeles) procurando que la imagen no pierda sus características. La decimación se aplica como un factor entero D a una señal $x(n)$, de tal manera que se puede definir como:

$$y(n) = x(Dn) \quad (3.2)$$

Sin embargo, si se reduce la tasa de muestreo a múltiplos de D , la señal obtenida presentará efecto de *aliasing*, para lo cual, en principio es necesario reducir el ancho de banda de la señal $x(n)$ a $f_{max} = \frac{f_x}{2D}$, aplicando un filtro paso-bajas [37].

En la figura 3.4 se muestra el proceso de una decimación. La entrada es una señal digital $x(n)$ y a la cual se le aplica un filtro paso-bajas $h(n)$, para obtener $v(n) = x(n) \otimes h(n)$. De la señal $v(n)$, entonces, se obtienen aquellas muestras que se presentan cada periodo múltiplo de D .

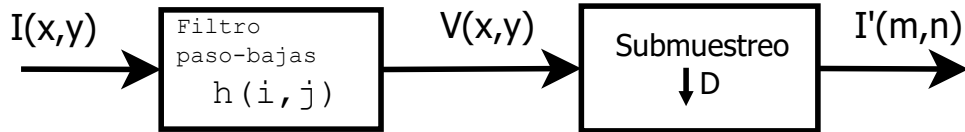


FIGURA 3.4: Proceso de decimación

La frecuencia de $x(n)$ es $F_x = \frac{1}{T_x}$. A la salida, la frecuencia de la señal está dada por $F_y = \frac{F_x}{D}$

Algo que es muy importante mencionar es que sólo se puede hacer un submuestreo con un valor de D entero [37], por lo que si se quiere realizar una decimación de una señal en periodos fraccionarios, será necesario aplicar antes una interpolación a la señal.

b) Interpolación

La interpolación, en señales, es el proceso de incrementar la tasa de muestreo de la señal por un factor I entero [37], este proceso es el inverso de la decimación y sigue una estructura similar. En imágenes, la interpolación de una imagen se puede observar como un “agrandamiento” de ésta, sin que pierda características en la medida de lo posible.

Para este caso, la interpolación se logra aplicando dos operaciones sobre la señal. En principio, se aplica un sobreescalamiento, superior al de la señal actual, es decir, considerar un mayor número de muestras en dicha señal; posteriormente, se aplica un filtro paso bajas. En la figura 3.5 se muestra el paso de la señal en el proceso de interpolación. Ahora, la frecuencia de la señal de entrada, que bien consideramos F_x es proporcional a la de salida F_y en un factor I , es decir, $F_x = IF_y$.

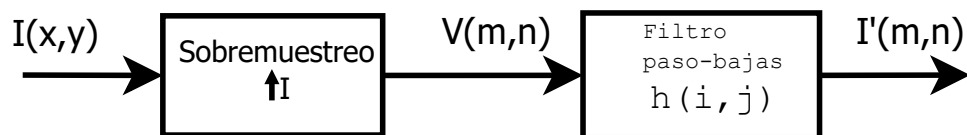


FIGURA 3.5: Proceso de interpolación

Ahora bien, como ya se mencionó, para reescalar una imagen en un factor fraccionario, es necesario que ésta pase por una interpolación, seguida de una decimación. Este proceso se representa en la figura 3.6. La escala de salida de la señal será I/D respecto de la señal de entrada del sistema.

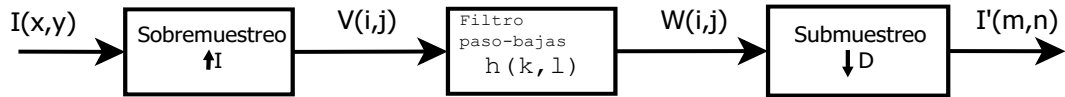


FIGURA 3.6: Proceso de reescalamiento de una señal por un factor I/D

3.2.4. Filtrado

El ruido se puede definir como una señal indeseada que se encuentra en un entorno. En imágenes, son artefactos no deseados que afectan la descripción de una imagen y que se pueden presentar como pixeles alterados o con valores inesperados.

Cuando se obtiene una imagen del entorno real con un sensor, por ejemplo una cámara, la imagen podría observarse con algunos pixeles que no son acordes al entorno captado, es ahí donde se hace presente el ruido.

Regularmente para eliminar artefactos ruidosos de la imagen se utiliza el suavizado en la misma que está dado por la convolución de una imagen por un filtro. Se basa en calcular un valor promedio en cada pixel respecto a sus pixeles vecinos, de tal manera que el pixel que contiene el valor ruidoso reduzca esa diferencia con los demás pixeles.

El suavizado de una imagen pretende reducir el ruido que hay en ella, funciona como un filtro paso-bajas, el cual reduce la presencia de frecuencias altas en una imagen. El suavizado puede obtenerse a partir de procesar la imagen a través de una convolución o una máscara de filtro. Algunos tipos de filtros, por ejemplo, son el filtro Gaussiano que ayuda a reducir el ruido blanco, o el filtro mediana que permite reducir el ruido conocido como “sal y pimienta”. [15]

El ruido es una función en el tiempo y el suavizado es una función en el espacio. El suavizado se utiliza para reducir el ruido, sin embargo, la manera “correcta” de eliminar el ruido sería promediar la intensidad de cada pixel en cada intervalo de tiempo, según se menciona en [15].

Dado que los filtros Gaussiano y mediana no son robustos a orientación, se implementan ventanas circulares que permiten hacerlos independientes a este inconveniente [15]. Además, una ventaja de los filtros gaussianos es que su transformada de Fourier también es una gaussiana.

En la mayoría de aplicaciones de procesamiento de imágenes, se utilizan filtros Gaussianos dado que su implementación es sencilla. Además, estos filtros poseen la propiedad de ser separables, puesto que un filtro en 2-D equivale al producto de un filtro renglón por un filtro columna y éstos pueden aplicarse a la imagen por separado obteniendo los mismos resultados que aplicar el filtro 2-D directamente sobre la imagen. [15]

3.2.5. Realzado

En imágenes, el realzado es una técnica para mejorar la calidad visual de una imagen, disminuyendo efectos indeseados como sombras y/o reflejos, además de aumentar el contraste de la imagen [34].

Actualmente, hay una buena cantidad de técnicas para realzar una imagen, algunas de ellas están basadas en su histograma. Un histograma se considera un método que puede representar el número de píxeles en una imagen en función de los niveles de intensidad presentes en ella. Por así decirlo, se trata de una función que tiene como objetivo calcular la frecuencia con la que aparecen cada uno de los niveles de intensidades de los píxeles presentes en una imagen en escala de grises. En la figura 3.7 se muestra un ejemplo de cómo se representa un histograma, la imagen 3.7(a) está en escala de grises y contiene niveles de intensidad que se encuentran en el rango [0-255], donde 0 pertenece al nivel de intensidad para negro y 255 es el nivel para blanco.

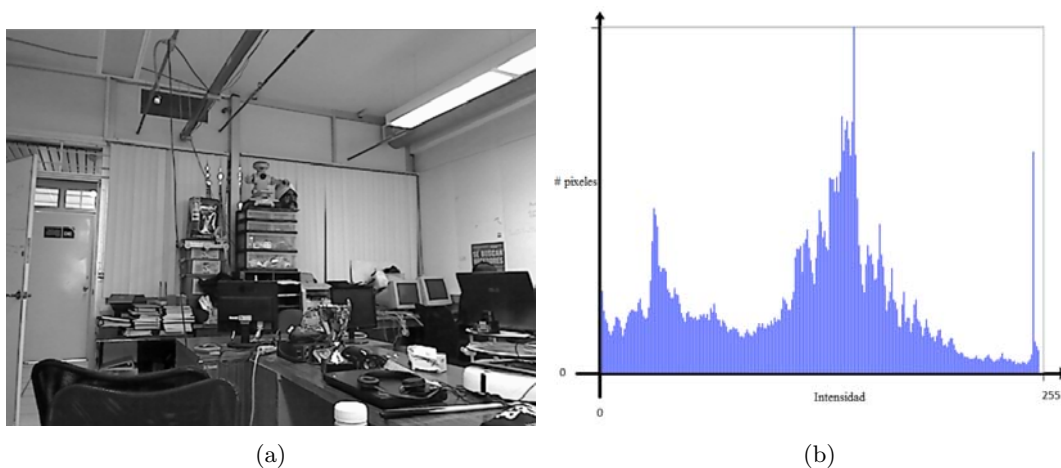


FIGURA 3.7: Ejemplo de un histograma. (a) es la imagen en escala de grises y (b) es su respectivo histograma

Ahora bien, la operación del histograma de una imagen permite manipular algunas de las características que tiene, por ejemplo, el brillo, el contraste, la corrección gamma y la nitidez [34]. En este escrito sólo se hará énfasis en la nitidez de la imagen, que es lo que nos concierne. Para los demás casos, existen suficientes fuentes en donde se pueden consultar, comenzando con [34].

La técnica utilizada para la mejora de la calidad de una imagen a partir de modificar el contraste de ella para hacerla más nítida se le denomina ecualización del histograma. La idea principal de esta técnica es la de intentar igualar el número de píxeles en cada uno de los niveles de intensidad que hay en ella, en otras palabras, tratar de generar un histograma uniforme de la imagen en escala de grises, con lo cual, los píxeles más brillantes pierden brillo y los más oscuros lo ganan, tratando de que la distribución de píxeles sea equivalente, o lo más aproximado, en cada uno de los niveles de intensidades. La ecualización del histograma se logra con una función que realce el contraste de la imagen.

La especificación del histograma es otra técnica de realzado de una imagen basada en histograma. En ella también se trata de distribuir las intensidades de los píxeles de una imagen, sin embargo, esta distribución se hace a partir del histograma de una segunda imagen con mejor calidad de brillo. Esta técnica es muy común en el registro de imágenes aéreas y satelitales. La ecualización del histograma se considera un caso particular de la especificación del histograma, al hacer que el histograma de una imagen sea especificado por otro que está distribuido de manera uniforme.

Otra forma de realzado es a partir del filtrado de una imagen, que se mencionó con anterioridad. Los diferentes filtros: paso-bajas, paso-altas, paso-banda, así como el suavizado, son técnicas de realzado de una imagen por bloques, al operar un bloque de píxeles.

3.3. Procesamiento de visión estereoscópica

En esta etapa es donde radican los métodos principales que conforman el sistema. La etapa de procesamiento se refiere a aquella donde se aplican los diferentes algoritmos para obtener los detalles o características necesarias de cada imagen, hablando de visión estéreo, para lograr su registro.

Esta etapa es la principal del sistema de visión estereoscópica. La parte fundamental del cálculo del mapa de disparidad se encuentra en el procesamiento de las imágenes estéreo. Partiendo de estas imágenes, es necesario comenzar a calcular la relación que existe entre la imagen sensada con respecto a la imagen de referencia.

Uno de los pasos más importantes, y que se abarcará más adelante, es la correspondencia estéreo de dos imágenes. Este paso es primordial para calcular la variación, y después la disparidad, del sistema estereoscópico. El problema de correspondencia no es trivial y requiere una parte importante del procesamiento.

Para el procesamiento de imágenes estéreo, existen algunos métodos separados en dos grupos [34]: los métodos basados en área y los métodos basados en características.

3.3.1. Obtención de características

Las características de una imagen son propiedades únicas de ella y, en lo que nos concierne, son utilizadas para determinar o establecer una correspondencia entre dos imágenes. Las características más deseables son los puntos, debido a que sus coordenadas pueden ser utilizadas directamente para determinar los parámetros de una función de transformación. [15]

Cuando se realiza una extracción de líneas o regiones de una imagen, éstas se pueden convertir en puntos dependiendo de la escena. Por ejemplo, las líneas extraídas se pueden intersecar entre ellas y esa intersección se da en un punto; por lo tanto, en cada intersección se pueden encontrar los puntos correspondientes y al final obtener sólo un conjunto de ellos. Lo mismo ocurre con las regiones, cuando cada región tiene un conjunto de puntos en ella, se puede obtener un punto medio de cada conjunto, que se denomina centroide, ese centroide puede definirse como un punto promedio de todos los puntos que forman parte de una región; así, por cada región se obtiene un centroide y el conjunto de centroides es un grupo de puntos que se pueden utilizar para determinar la función de transformación [15].

En la búsqueda de características, específicamente esquinas, el tamaño de la ventana debe ser proporcional a la desviación estándar del filtro suavizador gaussiano que es utilizado para detectar bordes y los gradientes de la imagen. La detección de bordes puede hacerse por ajuste de curvas. Mientras el tamaño de la ventana utilizada sea pequeño, la medida es local. Conforme el tamaño de la ventana es mayor, la medida es más global. [15]

Operador Laplaciano

Un método para detectar líneas en las imágenes es a través del operador laplaciano.

$$\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \tag{3.3}$$

$$T = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 2 \\ -1 \end{bmatrix} * \begin{bmatrix} -1 & 2 & -1 \end{bmatrix}$$

Este operador puede aplicarse sobre una imagen para lograr enfatizar las discontinuidades que hay, dichas discontinuidades son líneas en una imagen.

3.3.2. Correspondencia entre imágenes

Existen varios métodos para calcular la correspondencia de dos imágenes, toda vez que de éstas se obtuvieron las características de cada una, dichas imágenes son la imagen de referencia y la imagen sensada en un sistema de visión estéreo.

Al obtener características de las imágenes pueden aparecer *outliers*, que son todos aquellos puntos que aparecen en sólo uno de los dos conjuntos, y por consiguiente, no tienen correspondencia en el otro. Estos *outliers* pueden deberse a ruido u otros factores [15].

La correspondencia de las imágenes es un punto determinante en visión estéreo. Con ella se puede obtener la función que relaciona a una imagen con la otra. Cabe mencionar que existen diferentes métodos de empatado de características, dependiendo el tipo de características, un método puede ser mejor que otro. El objetivo del empatado de imágenes es comparar la similaridad que existe de un grupo de características respecto de otro, es decir, examinar la semejanza o diferencia de estos grupos [8]. A continuación, se presentan algunos métodos de empatado de características, descritos en [15].

a) Empatado por puntos

Dado un conjunto de puntos $P = \{p_i; i = 1, \dots, n_r\}$ de la imagen de referencia, y otro conjunto $Q = \{q_j; j = 1, \dots, n_s\}$ en la imagen sensada; tal que $p_i = (x_i, y_i)$ y $q_j = (x_j, y_j)$, se buscan todos los pares de índices (i, j) de los puntos, de tal manera que p_i y q_j representen el mismo punto en el entorno. [15]

Hay algunos atributos de las imágenes que pueden utilizarse para encontrar correspondencias de puntos, como la información de la posición de dichos puntos, distancia relativa entre ellos, incluso la orientación de líneas, según se menciona en [15]. Sin embargo, el uso de estos atributos dependerá de las necesidades de la aplicación.

Los puntos correspondientes en las imágenes podrían relacionarse por una transformación lineal o afín [15]:

$$X = ax + by + c \tag{3.4}$$

$$Y = dx + ey + f \tag{3.5}$$

En otro caso, si las imágenes de referencia y sensada están relacionadas por la transformación proyectiva, la relación de transformación será la siguiente [15]:

$$X = \frac{ax + by + c}{dx + ey + 1} \quad (3.6)$$

$$Y = \frac{fx + gy + h}{dx + ey + 1} \quad (3.7)$$

b) Empatado por coherencia de escena

Utiliza la distancia de Hausdorff para saber la diferencia de similaridad entre dos conjuntos de puntos. La distancia de Hausdorff entre dos conjuntos de puntos está definida como la máxima de las distancias mínimas entre los puntos de ambos conjuntos [21]:

$$h(P, Q) = \max[\min[\text{dist}(p, q)] \mid p \in P, q \in Q] \quad (3.8)$$

P y Q son conjuntos de puntos y, p y q son puntos que pertenecen a P y Q , respectivamente. Esta distancia suele ser un elemento confiable cuando no existen *outliers* [15].

En el empatado por coherencia de escena, si se conoce la correspondencia de 3 puntos no colineales en ambos conjuntos de puntos, entonces la correspondencia de los demás puntos puede ser conocida alineando ambos conjuntos en esos 3 puntos conocidos. Si la distancia de Hausdorff es pequeña implica que los puntos más cercanos en ambos conjuntos probablemente correspondan, sin embargo, una distancia grande significa que los puntos muy probablemente no correspondan entre ellos. Para verificar que existe correspondencia de los demás puntos, se encuentra el número de puntos en los conjuntos que tienen una distancia pequeña respecto a otro utilizando la transformación obtenida de las primeras 3 correspondencias. Los 3 pares de puntos asumidos con correspondencias que produzcan la mayor cantidad de correspondencias con los demás puntos de ambos conjuntos determinarán los parámetros de la transformación y los puntos con menor distancia uno de otro obtenidos por la transformación dada serán puntos que corresponden [15].

Para eliminar *outliers* con la ayuda de la distancia de Hausdorff, se puede aplicar un umbral de distancia; es decir, si existe algún punto de una imagen que se encuentra a una distancia mayor al umbral, con respecto al punto más cercano a éste en la segunda imagen, entonces se puede considerar que ese punto está aislado y no tiene correspondencia.

c) Empatado utilizando agrupamiento (*clusters*)

Se implementa para calcular los parámetros a , b , c , d , e , y f , de la transformación afín. El algoritmo, que se explica más a detalle en [15], utiliza arreglos de acumuladores $A[]$,

$B[]$, $C[]$, $D[]$, $E[]$ y $F[]$, inicializados en cero y de tamaño N . Para cada combinación de tripletas de pares de puntos, se calculan a , b , c , d , e y f , y sus correspondientes entradas en $A[]$, $B[]$, $C[]$, $D[]$, $E[]$ y $F[]$, se incrementa en 1. N es el número de tripletas de pares de puntos.

De manera eficiente, los parámetros: a , b , c , d , e , y f , se pueden calcular con las siguientes ecuaciones:

$$X = (x - x_0)S \cos \theta + (y - y_0)S \sin \theta + h + X_0 \quad (3.9)$$

$$Y = -(x - x_0)S \sin \theta + (y - y_0)S \cos \theta + k + Y_0 \quad (3.10)$$

Donde:

h y k : valores de traslación horizontal y vertical, respectivamente.

θ : rotación.

S : escala.

(x_0, y_0) : Coordenadas del centro de la imagen de referencia.

(X_0, Y_0) : Coordenadas del centro de la imagen sensada.

Para calcular los valores mínimos y máximos de los parámetros a , b , c , d , e y f , se utilizan rangos en las ecuaciones anteriores para el algoritmo de empatado por agrupamiento. El rango de escala S puede variar entre 0.5 y 2, por ejemplo. El rango rotacional θ podría estar entre $(-\frac{\pi}{4}, \frac{\pi}{4})$. El rango traslacional (h, k) puede variar horizontal y verticalmente, $\frac{\text{ancho}}{2}$ y $\frac{\text{alto}}{2}$, respectivamente. Los extremos mínimos y máximos de los rangos producirían los mínimos y máximos, respectivamente, de los parámetros [15].

d) Empatado mediante invarianza

Conociendo la posición de 3 puntos no colineales de la imagen sensada, q_1 , q_2 , y q_3 , se tiene la siguiente relación de cualquier otro punto de dicha imagen con respecto a esos 3 puntos [15]:

$$q = q_1 + \alpha_1(q_2 - q_1) + \alpha_2(q_3 - q_1) \quad (3.11)$$

Si la relación entre la imagen de referencia y la imagen sensada puede ser definida por una transformación afín, se expresa:

$$\underbrace{\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}}_q = \underbrace{\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix}}_A \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_p \quad (3.12)$$

$$\Rightarrow q = Ap \mid q \in Q \text{ y } p \in P$$

$$\Rightarrow Ap = A[p_1 + \alpha_1(p_2 - p_1) + \alpha_2(p_3 - p_1)]$$

Por lo tanto, existe una relación directa entre la imagen de referencia y la imagen sensada:

$$p = p_1 + \alpha_1(p_2 - p_1) + \alpha_2(p_3 - p_1) \quad (3.13)$$

e) Empatado de líneas

Utilizando las características de las líneas: posición, orientación y longitud. La orientación de una línea no se ve afectada por el ruido, pero sí su longitud. El modo de empatar líneas provenientes de dos imágenes es a través de sus diferencias traslacionales, rotacionales y de escala. En primer lugar, se determina la diferencia rotacional, posteriormente las diferencias de traslación y de escala [15].

La relación entre dos líneas es la siguiente [15]:

$$X = S(x \cos \theta + y \sin \theta) + t_x \quad (3.14)$$

$$Y = S(-x \sin \theta + y \cos \theta) + t_y \quad (3.15)$$

(X, Y) : posición de la línea en la imagen sensada.

(x, y) : posición de la línea en la imagen referencia.

S : diferencia de escala.

θ : diferencia rotacional.

(t_x, t_y) : diferencia traslacional.

Sean dos conjuntos de líneas $K = \{k_i : i = 1, \dots, n_r\}$ y $L = \{l_j : j = 1, \dots, n_s\}$, dos líneas en un conjunto pueden corresponder a una sola del segundo conjunto, o bien, algunas líneas pueden aparecer en un solo conjunto [15].

Se eligen dos líneas, k_i y l_j , para determinar la diferencia rotacional, y se asume que ambas líneas corresponden entre ellas. Obteniendo los ángulos $\angle k_i$ y $\angle l_j$ respecto al eje x , entonces se determina que $\theta_{ij} = \angle l_j - \angle k_i$, donde θ_{ij} es la diferencia rotacional entre las dos líneas k_i y l_j .

Para verificar la correspondencia se utiliza información de otras líneas en ambos conjuntos. Si k_i y l_j realmente corresponden, entonces aplicando una transformación rotacional de $-\theta_{ij}$ a L , todas las líneas de correspondencia en ambos conjuntos tendrían la misma orientación [15].

f) Empatado por regiones

En general, el empatado por regiones es similar al empatado por puntos utilizando los centroides de tales regiones, aunque se pueden utilizar las propiedades de tamaño y forma de cada región para optimizar el empatado. El empatado de forma se realiza utilizando la transformada de Fourier [15].

g) Empatado por plantilla

Éste se realiza localizando una plantilla en una imagen. La plantilla puede ser una subimagen de la de referencia y buscarla en la imagen sensada. Una plantilla es un modelo o diseño predefinido que está hecho para desarrollar otros modelos [8]. Es decir que en el empatado por plantilla se tiene un modelo definido por un conjunto de características de la imagen de referencia y se analiza la imagen sensada con el objetivo de encontrar el mismo modelo dentro de ella.

El método más simple de empatado por plantilla es aquel donde se tiene la plantilla y ésta se recorre a través de una ventana sobre una imagen, buscando la mínima diferencia entre la plantilla y la ventana de la imagen [8]. El proceso de empatado por plantilla implica recorrer la imagen sensada, por ventanas, con una plantilla y determinar la posición de desplazamiento en la que la plantilla y la ventana de la imagen sensada tienen mayor similitud. Esto sólo aplica cuando las dos imágenes tienen diferencia traslacional [15]. Algunas técnicas de empatado por plantilla se explican a detalle en [8].

Las plantillas pueden presentar variaciones, no todas son iguales. Inclusive, un ejemplo de ello está relacionado al ruido aditivo. Otros tipos de variaciones se dan en cambios de iluminación, el sensor de imagen o su configuración, además del punto de vista del cual un objeto se observa [47].

h) Empatado por plantilla invariante a rotación

Básicamente, utilizando una plantilla circular, al igual que la ventana y utilizando el centro de ambos como base y usa el método de disimilitud *K-S test* (Kolmogorov-Smirnov) [12], que usa propiedades estadísticas. Igualmente se puede usar la densidad de probabilidad obtenida del histograma de la plantilla. Por lo tanto, la similitud entre los histogramas de la plantilla y la ventana puede ser usada como función de similitud. Otra medida de similitud es a través de momentos invariantes [20] tomando el centro de cada uno como correspondencias y siendo circulares.

Son varios los métodos que existen para calcular la disimilitud entre dos imágenes, o, en nuestro caso, la plantilla y la ventana de la imagen. [15]

Similitud utilizando intensidades de imágenes

Se puede calcular mediante la Suma de Diferencias Absolutas [15]:

$$D(x, y) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} |f(i, j) - f_w(i + x, j + y)| \quad (3.16)$$

$x = 0, \dots, M - m + 1$

$y = 0, \dots, N - n + 1$

f : función de la plantilla.

f_w : función de la ventana.

$m \times n$: dimensiones de la plantilla / ventana.

El objetivo es minimizar D para encontrar el mejor empatao. Un estudio de Svedlow encontró que con los gradientes de intensidad se obtienen mejores resultados. Existen maneras de optimizar lo anterior y una de ellas es a través de la correlación cruzada.

$$S(x, y) = \frac{\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} g_t(i, j) g_w(i + x, j + y)}{\left[\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} g_t^2(i, j) \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} g_w^2(i + x, j + y) \right]^{1/2}} \quad (3.17)$$

Donde:

$$g(x, y) = f(x, y) - \bar{f}$$

\bar{f} : es el promedio de la función f .

Este procedimiento es mucho más lento que la Suma de Diferencias Absolutas. Dado que el numerador es una convolución, puede optimizarse utilizando la Transformada Rápida de Fourier. La idea principal de este método es maximizar S para encontrar el mejor empatao. [15]

Métodos de afinación de empatao

Este tipo de métodos se denominan *coarse-to-fine* debido a que es una manera de lograr que la información obtenida sea más precisa, en otras palabras, es un término que se refiere a afinar un algoritmo. El objetivo de estos métodos en el empatao de dos imágenes es el de acelerar el proceso de empatao, o lograr un incremento en la precisión del registro de imágenes

Dependiendo del procedimiento en el que se encuentre o el tipo de algoritmo implementado, la afinación se puede lograr a diferentes niveles: a nivel de imagen, a nivel de plantilla o en la búsqueda de correspondencias [15]:

A nivel de imagen: cuanto más grande es la escala entre imágenes, mejor es el registro entre ellas. Esto se debe a que al aumentar la escala de las imágenes, se obtiene una mayor cantidad de diferencias geométricas que pueden utilizarse para obtener un registro de imágenes más preciso.

A nivel de plantilla: cuando es imposible o poco probable que la escala de las imágenes a registrar se pueda aumentar, entonces una manera de mejorar el registro entre las imágenes es utilizando una plantilla más grande, que abarcaría más espacio respecto de la imagen sensada y es posible obtener mayores diferencias geométricas, además de una mayor cantidad de características entre la plantilla y la ventana que puedan definir mejor el registro de ambas imágenes.

A nivel de búsqueda: mientras se realiza la búsqueda de correspondencia entre la plantilla y la ventana deslizante, ésta pudiera hacerse en intervalos grandes, es decir, cuando las diferencias de intensidades en una ventana respecto a la plantilla son grandes, la ventana debe desplazarse en grandes intervalos. Sin embargo, cuando se obtiene una diferencia cada vez menor, entonces el intervalo de desplazamiento disminuye.

Cálculo de la función de transformación

Una función de transformación permite encontrar la relación de correspondencia de todos los puntos de las imágenes, a partir de las coordenadas de los puntos que se sabe corresponden entre ellos. [15]

“Una función de transformación contiene información acerca de las diferencias geométricas entre dos imágenes” [15]. Las diferencias geométricas conocidas, junto con la función de transformación obtenida pueden ser utilizadas para eliminar imprecisiones (objetos que sobresalen) en las correspondencias. Si las imágenes sólo tienen diferencias geométricas lineales, entonces se dice que $f_x(x, y)$ y $f_y(x, y)$ son planares. [15]

Función de transformación

El problema de encontrar la función de transformación es el siguiente [15]: Dadas las coordenadas de N puntos que tienen correspondencia en las imágenes de referencia y sensada:

$$\{(x_i, y_i), (X_i, Y_i) : i = 1, \dots, N\}$$

La idea es determinar una función de transformación $f(x, y)$ que satisfaga lo siguiente:

$$X_i \cong f_x(x_i, y_i)$$

$$Y_i \cong f_y(x_i, y_i)$$

La función de transformación depende de la diferencia geométrica entre imágenes, la precisión de la correspondencia de los puntos, la densidad y organización de los puntos. [15]

Una característica que presentan las funciones de transformación es la linealidad. La función de transformación permite que todos los puntos de la imagen de referencia tengan

correspondencia con los puntos de la imagen sensada, esto implica que la imagen sensada puede tener una relación lineal respecto a la imagen de referencia. Sin embargo, “debido a la naturaleza no lineal del proceso de adquisición de imágenes y la deformabilidad de una escena, las imágenes al ser registradas tienen diferencias geométricas no lineales” [15], Cuando esa diferencia no lineal es pequeña, entonces ésta se vuelve despreciable y se puede trabajar tal cual fuera lineal.

Una función de transformación podría ser utilizada para mapear de manera precisa los puntos de control en las dos imágenes, conociendo dicho conjunto de puntos de control. Una función apropiada alineará el resto de puntos en las imágenes. [15]

Se necesita una transformación que suavice el ruido y las pequeñas imprecisiones en las correspondencias. Si las hay, es preferible hacer aproximaciones en lugar de interpolación para el registro de imágenes. [15]

Transformación de similaridad

$$X = S[x \cos \theta + y \sin \theta] + h \quad (3.18)$$

$$Y = S[-x \sin \theta + y \cos \theta] + k \quad (3.19)$$

Esta transformación representa las diferencias de traslación (h, k) rotación (θ) y de escala (S) en las imágenes, y aplica cuando las coordenadas de dos puntos que corresponden en ambas imágenes son conocidas. [15]

La diferencia rotacional puede calcularse con el ángulo de las líneas que se forman con los dos puntos que corresponden de cada imagen. [15]

La diferencia de escala es calculada por el radio de distancia que hay entre los puntos que corresponden. La relación que existe entre la distancia del segundo par de puntos con respecto al primero. [15]

Para conocer la diferencia rotacional es necesario conocer las dos anteriores, de manera que para calcular los parámetros de traslación, se despejan h y k en la relación de similaridad. [15]

Cuando existen correspondencias que presentan ruido o pueden ser imprecisas, no basta con hacer la relación de similaridad con sólo un par de puntos que corresponden, En estos casos, para calcular los parámetros de traslación, rotación y de escala será necesario utilizar más de dos correspondencias, para que de alguna manera se logren filtrar el ruido y las imprecisiones. Cuando se presenta ruido, el método que se usa para reducirlo es el de mínimos cuadrados. Por otro lado, si las correspondencias presentan imprecisiones,

es decir, existe una gran cantidad de *outliers*, entonces el método más común para deshacerse de ellas es a través de la clusterización. [15]

La transformación proyectiva es una transformación lineal y está definida por las ecuaciones 3.6 y 3.7.

En ésta existen 8 parámetros: a, b, c, d, e, f, g y h , que pueden obtenerse si se conocen las coordenadas de 4 puntos no colineales que corresponden en las dos imágenes. Si hay ruido, se usan más de 4 correspondencias para aplicar el método de mínimos cuadrados. [15]

Cuando la escena se encuentra lejos de las cámaras, la transformación proyectiva puede aproximarse por la transformación afín de las ecuaciones 3.4 y 3.5. Los 6 parámetros de la transformación afín (a, b, c, d, e, f) pueden obtenerse con al menos 3 puntos no colineales que corresponden en las imágenes. El registro de imágenes utilizando transformación afín es más común en imágenes aéreas y satelitales. [15]

La transformación proyectiva requiere que las líneas rectas en la imagen de referencia permanezcan rectas en la imagen sensada. Si las líneas paralelas permanecen así de la imagen de referencia a la imagen sensada, entonces se utiliza transformación afín. Si los ángulos entre líneas que corresponden permanecen, entonces la transformación de similaridad o del sistema de coordenadas cartesianas puede ser utilizada. Por lo tanto, si las imágenes pueden ser registradas por la transformación de similaridad, entonces pueden ser registradas por transformación afín, y si lo último es posible, podrán ser registradas por transformación proyectiva. [15]

Si una línea recta en la imagen de referencia mapea una curva en la imagen sensada, se dice que existe una relación no lineal, la cual requerirá una transformación no lineal para lograr el registro. Algunas de las transformaciones no lineales que existen y que se pueden consultar en [15] son: *Thin-plate spline*, *Multiquadric*, *Weighted mean*, *Piece-wise linear*, *Weighted linear*, etc.

La complejidad temporal del registro de imágenes depende del tiempo que tarda en ser computada la función de transformación y el tiempo necesario para remuestrear la imagen sensada al sistema de coordenadas de la imagen de referencia [15]. En el caso de las transformaciones lineales (afín, proyectiva y de similaridad), la complejidad temporal está dada por $O(N) + O(n^2)$. En cuanto a las transformaciones no lineales, tienen una complejidad que va desde $O(n^2)$ hasta $O(N^2) + O(n^2N)$ [15]. Donde N es el número de puntos que corresponden entre imágenes y $n \times n$ es el tamaño de cada imagen.

Remuestreo

Una vez que la función de transformación ha sido calculada, ahora lo que se busca es mapear la imagen sensada a la de referencia.

Remuestreo de imágenes [8]:

$$x \rightarrow x' = Ax$$

$x = (x_1, x_2)^T$: coordenadas del punto de una imagen.

$A \in \mathbb{R}^{2 \times 2}$: matriz de transformación:

$$A = \begin{bmatrix} s \cos \theta & -s \sin \theta \\ s \sin \theta & s \cos \theta \end{bmatrix}$$

$$I'(x') = A^{-1} = I(x)$$

3.3.3. Registro de imágenes estéreo

En el caso de la visión estéreo, las dos imágenes son obtenidas por dos cámaras que tienen características iguales (o similares) y que son desplazadas horizontalmente, incluso puede ser de manera vertical. También, existe un método descrito por Ramakant Nevatia en [32] en el que se obtiene una relación estéreo de una sola cámara, pero que se va desplazando y obtiene diferentes capturas de una escena en determinados instantes de tiempo. La relación estéreo entre dos capturas de una escena hechas con una cámara se puede obtener si se conoce el desplazamiento de la cámara de un instante a otro. Desde otro punto de vista, se podría obtener el desplazamiento de la cámara calculando la relación entre capturas y obteniendo las matrices esencial y fundamental del desplazamiento, además de conocer los parámetros intrínsecos de la cámara utilizada.

Algo muy importante para que se pueda obtener información en 3D de una escena con visión estéreo es que las dos imágenes (o más) obtenidas deben pertenecer a la misma escena. Si alguna imagen no observa la misma escena que la otra, entonces se pierde la información y no es posible obtener una relación entre esas imágenes.

Al registro de imágenes estéreo también se le denomina correspondencia estéreo y su objetivo es recuperar la geometría en 3D de una escena [15]. Por lo regular, en el registro de imágenes estéreo separadas horizontalmente, la imagen izquierda es la imagen de referencia y la imagen derecha es la sensada.

En imágenes estéreo, la oclusión es un problema muy común, dado que las distancias de la escena con las cámaras son mucho menores que en el registro de imágenes aéreas o satelitales. La oclusión se presenta cuando en dos imágenes (estéreo) de una escena que

se está reconstruyendo en 3D, existen discontinuidades de profundidad en partes de la escena que sólo aparecen en una de las imágenes. [15]

En el caso de visión estéreo, no es necesario obtener una función de transformación, pues no se necesitan sobreponer ambas imágenes, basta con una función de interpolación para calcular los valores de profundidad no determinados, ya sea por oclusión u otros problemas. [15]

Asumiendo que dos imágenes tomadas por dos cámaras de condiciones iguales (o lo más similares posible) que están separadas de manera horizontal y sus focos se encuentran en un solo eje (son colineales), lo cual se representa en la figura 3.8, podemos deducir:

(x_r, y_r) : coordenadas de la imagen de referencia (izquierda).

(x_s, y_s) : coordenadas de la imagen sensada (derecha).

f : longitud focal.

B : distancia entre los ejes ópticos.

(X, Y, Z) : coordenadas espaciales del punto P de correspondencia estéreo.

Suponiendo las coordenadas del punto real con respecto al sistema coordenado de la cámara izquierda (X_L, Y, Z) y con respecto al de la cámara derecha (X_R, Y, Z) , se dice que el sistema de coordenadas de las cámaras estereoscópicas se encuentra en medio de los dos sistemas de coordenadas de imagen (izquierdo y derecho). En general, se asume que la geometría de las cámaras estéreo es conocida, de esta manera las líneas epipolares también lo son [47]. De igual modo, se asume que el modelo de las cámaras es *pinhole*.

La relación entre el punto real con respecto al sistema de coordenadas de la imagen izquierda y al sistema de coordenadas de la cámara estéreo quedaría de la siguiente manera:

$$X_L = X + \frac{B}{2}$$

$$Y_L = Y$$

$$Z_L = Z$$

De igual manera para la cámara derecha:

$$X_R = X - \frac{B}{2}$$

$$Y_R = Y$$

$$Z_R = Z$$

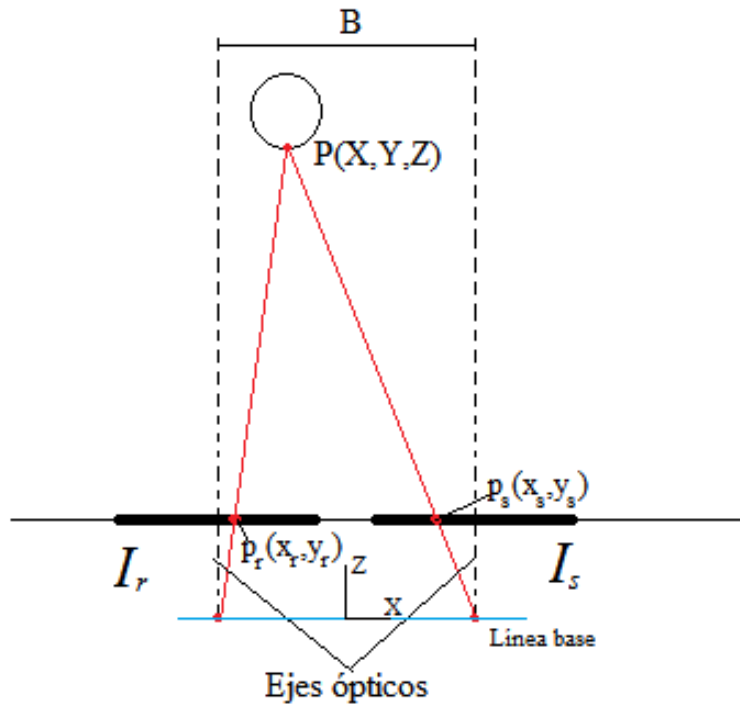


FIGURA 3.8: Representación de un punto en el espacio a un sistema de visión estereó [15].

Por semejanza de triángulos, en la cámara izquierda:

$$\frac{x_L}{f} = \frac{X_L}{Z_L} = \frac{X + \frac{B}{2}}{Z}$$

Por lo anterior, se deduce que:

$$\begin{aligned} X + \frac{B}{2} &= x_L \frac{Z}{f} \\ \Rightarrow X &= x_L \frac{Z}{f} - \frac{B}{2} \end{aligned} \tag{3.20}$$

Repetiendo el mismo procedimiento para la cámara derecha:

$$\begin{aligned} \frac{x_R}{f} &= \frac{X_R}{Z_R} = \frac{X - \frac{B}{2}}{Z} \\ X - \frac{B}{2} &= x_R \frac{Z}{f} \\ \Rightarrow X &= x_R \frac{Z}{f} + \frac{B}{2} \end{aligned} \tag{3.21}$$

Igualando X en las ecuaciones 3.20 y 3.21:

$$\begin{aligned}
 x_L \frac{Z}{f} - \frac{B}{2} &= x_R \frac{Z}{f} + \frac{B}{2} \\
 \Rightarrow x_L \frac{Z}{f} - x_R \frac{Z}{f} &= B \\
 \Rightarrow \frac{Z}{f} (x_L - x_R) &= B \\
 \Rightarrow Z &= \frac{Bf}{x_L - x_R}
 \end{aligned} \tag{3.22}$$

La diferencia $x_L - x_R$ se considera disparidad y es la distancia entre los puntos correspondientes, recordemos que se asume que el sistema de visión estéreo sólo está desplazado horizontalmente. [15]

La ecuación 3.22 muestra que la profundidad es inversamente proporcional a la disparidad. Cuando la profundidad es tal que los puntos que corresponden están muy cerca, implica que la distancia a la que se encuentra el punto en la escena con respecto a las cámaras estéreo es grande, y si la disparidad es mayor, indica que el punto está más cerca. [15]

Cuando la disparidad es cero, el punto del objeto está muy alejado del sistema estéreo. Si las cámaras se alejan entre ellas permitiría tener un mayor rango de percepción de profundidad, sin embargo, los problemas de oclusión podrían presentarse para distancias más cortas, pues podría impedir que un objeto sea visible en alguna de las dos imágenes. Una medida de disparidad mayor permite mejores medidas de profundidad y más precisas. Otra manera de incrementar la disparidad es aumentar la distancia focal, pero esto es más complicado para nuestro caso. [15]

Relación para cámaras cuyos ejes son convergentes

Existen casos de visión estéreo para cámaras cuyos ejes ópticos son convergentes y que se muestra en la figura 3.9.

En estos casos lo que se busca es que las imágenes pasen de tener ejes ópticos convergentes a que tengan estos ejes paralelos. El procedimiento para realizar tal acción se denomina *rectificación*.

Como se menciona en [15], la rectificación “es el proceso de transformar la geometría de las imágenes obtenidas por cámaras con ejes convergentes a imágenes obtenidas por cámaras con ejes paralelos”. Esto permite facilitar el cálculo de la disparidad entre imágenes y aplicar el procedimiento descrito anteriormente.

Restricciones en correspondencia estéreo

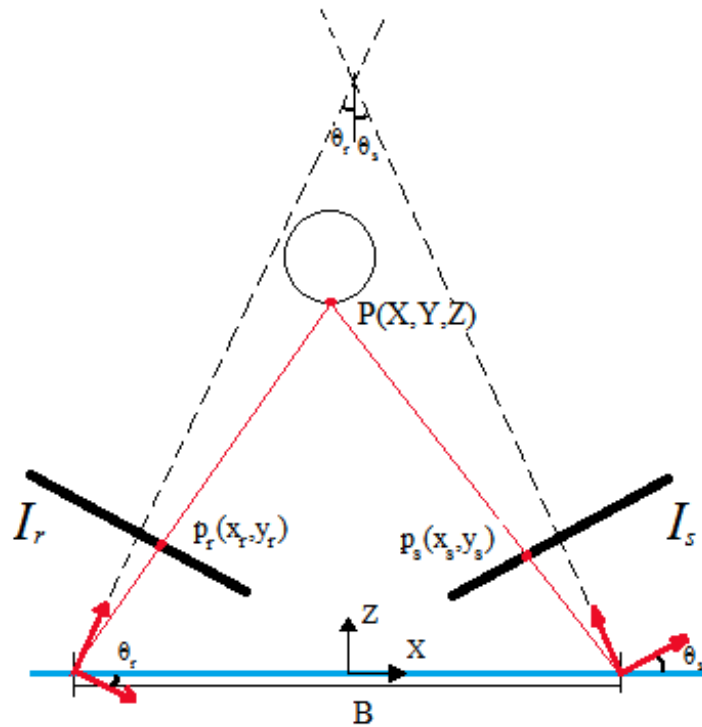


FIGURA 3.9: Representación de un punto en el espacio a un sistema de visión estéreo con ejes ópticos convergentes [15].

Con el conocimiento de la geometría epipolar del sistema de visión estéreo, se pueden determinar también una serie de restricciones que permiten que el emparejo de dos imágenes pueda ser más eficiente, dado que se restringe la búsqueda de correspondencias. En [15] se mencionan las siguientes restricciones:

- Restricción epipolar: buscar la correspondencia de un punto de la imagen de referencia en la imagen sensada, dado que dicho punto se encuentra en la línea epipolar.
- Restricción de unicidad: cada punto en una imagen tiene máximo una correspondencia en la imagen opuesta. Sólo se deberá seleccionar una correspondencia en caso de que se encuentren más. La oclusión puede producir huecos en el mapa de profundidad. En [15], se menciona que “un algoritmo de correspondencia estéreo robusto debería detectar los puntos ocluidos y evitar encontrar su correspondencia en la otra imagen”.
- Restricción de ordenamiento: “si los puntos a lo largo de una línea epipolar pertenecen a un objeto rígido, el orden de los puntos a lo largo de las líneas epipolares en las dos imágenes será el mismo” [15]. “La restricción de ordenamiento puede ser usada para eliminar algunos *mismatches*” [15], ya que se pueden encontrar

puntos correspondientes a lo largo de las líneas epipolares, si éstos no respetan el mismo orden, entonces no son correspondientes, aunque si el orden es el mismo, no necesariamente las correspondencias son correctas (condición necesaria pero no suficiente). Puede utilizarse para eliminar aquellos *mismatches* potenciales.

- Restricción de continuidad: “si la escena es rígida, los puntos en la escena que están cerca uno de otro aparecen cerca uno del otro en ambas imágenes” [15].
- Restricción de compatibilidad fotométrica: es el principio que se utiliza al realizar una correspondencia a partir de la correlación. Se busca que dos bloques de imágenes se comparen entre ellos buscando intensidades similares. Si las intensidades no tienen similitud, quiere decir que no hay correspondencia. Por otro lado, si las intensidades son similares, no necesariamente los bloques coinciden.
- Restricción de compatibilidad geométrica: Se busca que haya formas similares entre imágenes, dado que las formas no cambian (o no deberían) de una imagen a otra en la misma escena. En este sentido, se puede realizar un empatado a partir de regiones o de líneas. También se denomina *Restricción de compatibilidad de características*.

3.4. Postprocesamiento de información

Esta es la última etapa del sistema de visión. En esta se aplican diferentes métodos de mejora o interpretación de los resultados obtenidos de la etapa de procesamiento.

El postprocesamiento es la etapa considerada como aquella en la que se realiza una refinación a la información obtenida por los elementos principales del procesamiento central. Para nuestro caso, será el procesamiento posterior hecho a la información proporcionada por el algoritmo de visión estéreo que produce, como salida del sistema, un mapa de disparidad en el que se especifica la profundidad a la que se pueden encontrar los objetos en la escena respecto al sistema de cámaras estéreo.

En esta etapa, el tratamiento que se le da a la información se realiza con la idea de calcular una nube de puntos a partir del mapa de disparidad. Esto puede lograrse con la información de la reproyección en 2D de una escena en 3D, además de algunos de los parámetros definidos por las cámaras utilizadas, pues se utiliza como referencia el sistema de coordenadas de las cámaras estéreo.

En esta misma etapa se abarcará el procedimiento necesario para calcular una nube de puntos tomando el mapa de disparidad. Por último, se refinará la profundidad de la nube de puntos tomando como base el mapa de disparidad.

3.4.1. Corrección del mapa de disparidad

La corrección del mapa de disparidad implica que aquellas regiones del mapa en las que no se pudo obtener una disparidad, ya sea por errores en el cálculo de la disparidad o por la oclusión, deben ser corregidas con el fin de que este mapa considere la mayor cantidad de puntos posibles para reconstruir la escena en 3D y sea lo más fiel posible.

Una manera de realizar esta corrección es a través de una interpolación para calcular los valores intermedios en los huecos que existen en el mapa de disparidad.

3.4.2. Proyección de información en 2D a 3D

La reproyección de una imagen en 2D necesita de ciertas características para realizarse. Si bien, el proceso de pasar de una escena en 3D a una imagen 2D depende del modelo de proyección de la cámara y resulta sencillo cuando se trata de una cámara con modelo *pinhole*, el proceso inverso también lo es, salvo que se necesitan saber los parámetros intrínsecos de tal cámara. En el apéndice B se explica un poco de cómo funciona la reproyección.

La simple imagen en 2D no es suficiente para proyectar un conjunto de puntos, ya que ésta sólo proporciona la información de las coordenadas (x, y) , lo cual hace notar que se necesita un dato más para proyectar el punto correspondiente en la coordenada de z . En visión estereoscópica el mapa de disparidad es el que contiene esta información, debido a que este mapa incluye la información aproximada de la profundidad a la que se encuentra cada punto proyectado respecto al sistema de referencia de las cámaras. Es decir, que con el mapa de disparidad, podemos conocer la información (x, y) , correspondiente a cada pixel de la imagen de disparidad, más la intensidad del pixel en una escala de grises que indica la profundidad denotada por d (valor de disparidad). Por lo tanto, con la información (x, y, d) de cada pixel y los parámetros intrínsecos de la cámara, es posible hacer una reproyección de la información en 2D, dada por el mapa de disparidad, para obtener un conjunto de puntos en 3D.

3.4.3. Profundidad basada en la disparidad

Como ya se ha hecho mención, el mapa de disparidad representa la información de profundidad de una escena capturada por un sistema de visión estereo, este mapa es derivado del proceso de hacer esa captura y obtener la correspondencia de los puntos en dos (o más) imágenes de una misma escena. A partir de la triangulación hecha por la geometría epipolar del sistema estereo, se puede estimar la profundidad de los objetos

vistos en las cámaras; cuando un objeto se encuentra muy cerca a las cámaras, se puede notar un cambio notable en la posición del objeto de una cámara a otra, respecto a sus coordenadas de imagen correspondientes; en cambio, si un objeto está alejado, la diferencia de posición entre las cámaras se va haciendo menor.

Capítulo 4

Sistema de Visión Estereoscópica

Una vez que se conocen las bases de un sistema de visión estereoscópica y cómo es que éste puede obtener información en 3D a partir de dos imágenes en 2D, ahora hablaremos de la elaboración de un sistema de este tipo que pueda ser apto para un robot humanoide con recursos de cómputo limitados.

Como ya se había hecho mención anteriormente, uno de los principales motivos por los que se decidió trabajar en un sistema de visión estéreo es para obtener información tridimensional de una escena en la que pudieran existir una cantidad N de objetos, pero que en la misma pueden haber otros sistemas interactuando entre sí. Además, el hecho de tener información tridimensional facilita el desplazamiento y el reconocimiento de lo que existe en el entorno. De igual manera, es más sencillo obtener la distancia a la que se encuentra un objeto si se tiene información espacial.

A lo largo de este capítulo se expondrá el método utilizado para la implementación del sistema de visión estéreo en el robot NimbRo-OP. Este sistema, mostrado en la figura 4.1, cuenta con las etapas de procesamiento del capítulo anterior. En principio, como entrada al conjunto de operaciones, se deberá contar con un par de imágenes de una misma escena obtenidas por un sistema de cámaras estereoscópicas. Asumiremos que las capturas necesarias para realizar la calibración del sistema ya fueron obtenidas previamente.

A continuación, se detallan los pasos del método a seguir para el sistema de visión estéreo a implementar en el robot humanoide:

- Montar cámaras.
- Ejecutar la calibración de las cámaras para obtener parámetros extrínsecos e intrínsecos.

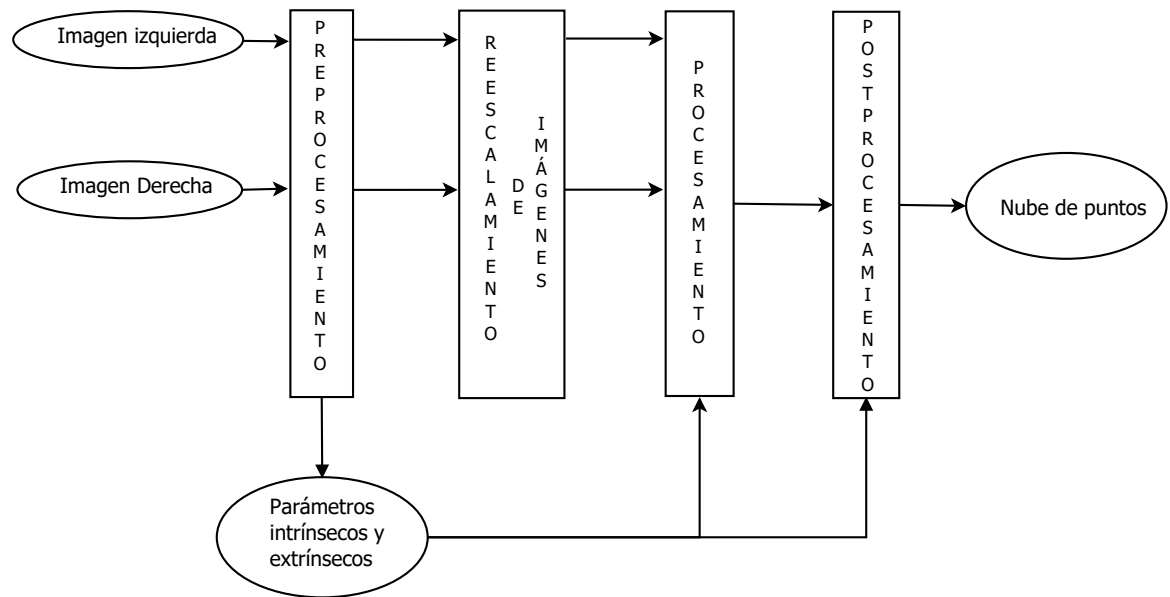


FIGURA 4.1: Diagrama de bloques del sistema de visión estereoscópica.

- Rectificar imágenes.
- Aplicar técnicas de mejora a las imágenes para optimizar resultados. Se aplican filtros promediadores para eliminar ruido y se ejecuta la especificación del histograma para equilibrar los histogramas de las imágenes y de ese modo, los niveles de intensidad.
- Realizar el empatao de las imágenes, conociendo su geometría y los parámetros de las cámaras.
- Obtener mapa de disparidad.
- Aplicar técnicas de refinamiento sobre el mapa de disparidad. Se utiliza filtro mediana para eliminar cierta cantidad de ruido y resaltar los bordes, además se aplican operaciones de cerradura para rellenar algunos huecos con valores uniformes.
- Realizar la reproyección del mapa de disparidad a un espacio 3D para obtener una nube de puntos.

Procedimiento con OpenCV

En primer lugar, es necesario realizar una calibración de las dos cámaras utilizadas, esta calibración se realiza con un patrón de cuadros blancos y negros (como un tablero de ajedrez). El objetivo de la calibración es calcular los parámetros intrínsecos y extrínsecos de las cámaras, los patrones cuadrículados son utilizados porque los cuadros blancos

y negros intercalados producen esquinas suficientemente notorias para que se puedan obtener en las imágenes de una manera precisa. Por lo tanto, conociendo la posición en las coordenadas de la imagen de cada esquina y conociendo las dimensiones del cuadrículado, asumiendo que la cámara usada es de modelo *pinhole* y el patrón de cuadros está sobre un plano, es decir, todas las esquinas son coplanares en el espacio, entonces se pueden calcular los parámetros de las cámaras. En OpenCV, la función que obtiene los parámetros de las cámaras se basa en el método de Zhang presentado en [48] y en la herramienta de calibración de cámaras para MATLAB, creada por Jean-Yves Bouguet¹.

En la figura 4.2 se muestran algunos ejemplos de capturas del patrón de cuadros para realizar la calibración. Las imágenes del lado izquierdo fueron captadas por la cámara izquierda, mientras que las del lado derecho por la cámara derecha. El par superior de imágenes fueron tomadas en el mismo instante de tiempo, al igual que el par medio y el par inferior.

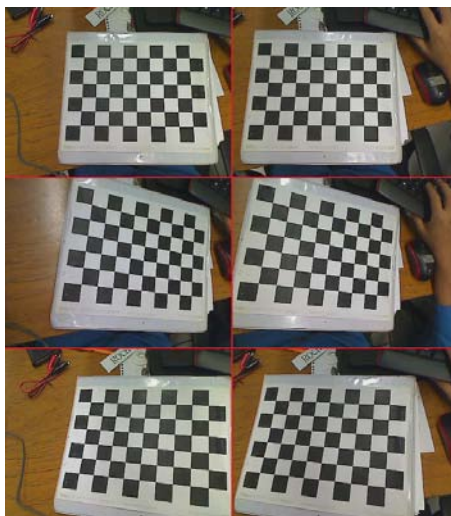


FIGURA 4.2: Algunas capturas para la calibración.

Ahora, debido a que el patrón de cuadros es capturado en las dos cámaras en el mismo instante, se pueden conocer también las coordenadas de imagen de las esquinas en cada una de las capturas, con esto se puede obtener una relación entre las dos cámaras acerca del desplazamiento y la rotación de la segunda cámara respecto a la primera.

Al final de la calibración, se obtienen los parámetros intrínsecos y extrínsecos del sistema de visión estéreo. Los parámetros intrínsecos contienen información de la distorsión de cada cámara del sistema y las matrices de cámara; los parámetros extrínsecos contienen la información de rotación y la traslación de una cámara respecto a la otra. Además, en OpenCV se puede obtener, con ayuda de la calibración, el error RMS de reproyección

¹Disponible en la página web http://www.vision.caltech.edu/bouguetj/calib_doc/

de una imagen 2D a 3D. Este error permite determinar el margen que tiene la correspondencia de puntos en la calibración. Se obtuvieron diferentes resultados para distintas escalas, éstos serán presentados más adelante.

La matriz de cámara contiene la longitud focal y centro focal, expresados en píxeles. La matriz de parámetros extrínsecos contiene la traslación y rotación.

Una vez calculados los parámetros de las cámaras, se obtiene la relación entre ambas imágenes y se aplica la rectificación de las imágenes, obteniendo las matrices esencial y fundamental.

Con las dos imágenes rectificadas, ahora se puede obtener el mapa de disparidad, que en OpenCV es posible con ayuda del algoritmo SGBM que se encuentra implementado en este conjunto de bibliotecas. Este algoritmo implementa una comparación por bloques comparando las dos imágenes, toma un bloque de la imagen de referencia y lo va comparando con bloques de la imagen sensada utilizando como criterio de costo la métrica de Birschfield-Tomasi [4]. SGBM está basado en [18] y que se describe en el *Apéndice A*, con algunas variantes descritas en la documentación de OpenCV.

Teniendo el mapa de disparidad de las dos imágenes comparadas, ahora es posible determinar la nube de puntos, donde el mapa de disparidad nos ayuda a conocer la profundidad aproximada de cada uno de los píxeles, por lo cual, utilizando la matriz de parámetros extrínsecos y el mismo mapa de disparidad, se realiza la reproyección bidimensional a tridimensional, y así obtener un conjunto de puntos que posteriormente será la nube a procesar. El procedimiento de reproyección se explica en el *Apéndice B*.

A la nube de puntos resultante del proceso anterior se le aplica un posprocesamiento que básicamente se trata de un filtro que promedia los valores de profundidad del mapa de disparidad. Esto permite que la nube no contenga demasiadas variaciones y las superficies sean lo más continuas posible.

Otro factor a considerar en el posprocesamiento es que el mapa de disparidad tiende a tener agujeros (secciones en color negro), ya sea por la misma disparidad o por oclusión. Para ello, se pueden utilizar métodos de interpolación que ayudan a rellenar esos huecos.

4.1. Etapa de preprocesamiento.

La etapa de preprocesamiento está dada por todos los pasos que preceden al procesamiento principal del sistema de visión estéreo. Como ya se ha hecho mención anteriormente, en esta etapa se preparan las imágenes izquierda y derecha para procurar que los resultados del procesamiento sean lo más óptimo posible.

Las diferentes sub-etapas del preprocesamiento, y que se muestran en la figura 4.3, son:

- Calibración de imágenes: obtener los parámetros intrínsecos y extrínsecos de las cámaras, necesarios para eliminar la distorsión y, posteriormente, rectificar las imágenes estéreo. En esta etapa se utilizan las capturas del patrón de cuadros almacenadas en memoria, y se requiere especificar el factor de escala que se utilizará.
- Filtrado de las imágenes: para eliminar ruido de las imágenes que pueda alterar sus características. Se aplica un filtro promediador gaussiano.
- Realzado: mejorar la definición de cada imagen para adquirir características más precisas, utilizando especificación del histograma.
- Reescalamiento: reducir el tamaño de las imágenes al factor de escala especificado en la etapa de calibración, para disminuir el cómputo de dichas imágenes y ejecutar el sistema en tiempo real.

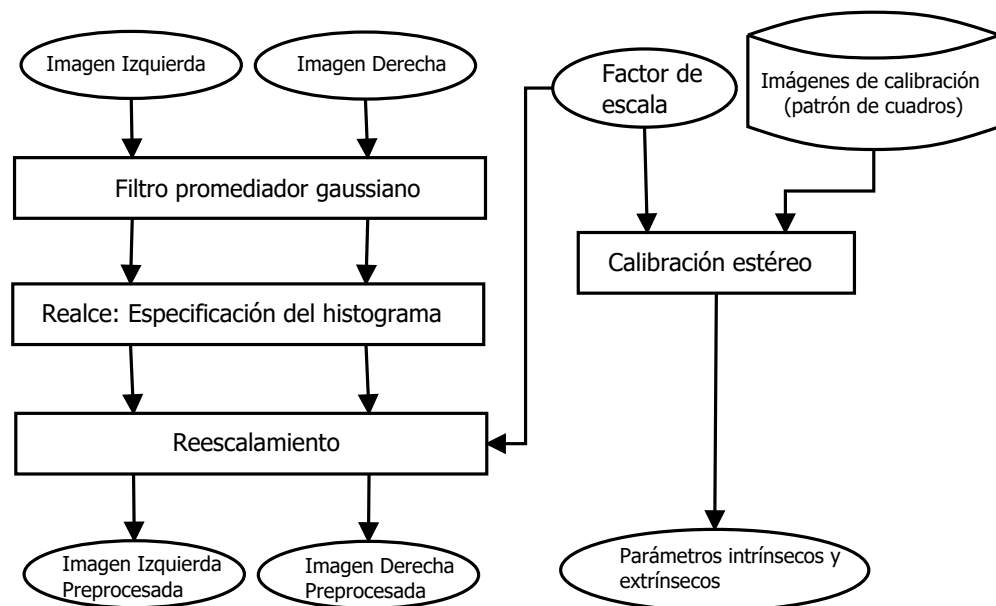


FIGURA 4.3: Diagrama de bloques de la etapa de preprocesamiento.

4.1.1. Calibración de las imágenes

El procedimiento de calibración del sistema de visión estereoscópica es el siguiente:

- Montar el sistema de visión estéreo con ambas cámaras y mantenerlas a una distancia fija.
- Utilizando un patrón de cuadros blancos y negros, hacer capturas con el sistema de visión estereoscópica de manera que por cada captura se tengan dos imágenes (una de la cámara izquierda y la otra de la derecha), realizar una cantidad considerable de capturas.
- En cada par de imágenes del mismo instante de tiempo, calcular las esquinas del patrón de cuadros, realizar el mapeo de los puntos de una imagen a la otra para obtener la relación de los puntos en ambas imágenes. De esta manera, se pueden calcular los parámetros intrínsecos y extrínsecos de las cámaras. Asimismo, se obtienen las matrices de rotación, traslación, la matriz esencial y la fundamental, de una cámara con respecto a la otra.
- Con los parámetros de las cámaras y las matrices obtenidas, realizar la rectificación de imágenes.

Con ayuda de la calibración podemos obtener los parámetros intrínsecos y extrínsecos de las cámaras. Además, es posible que se calcule un error de reproyección de las imágenes, utilizando el método de optimización de Levenberg-Marquardt [30], para minimizar el error de reproyección. Este error se calcula obteniendo la información de los puntos de interés, en este caso, las esquinas del patrón de cuadros, comparada con la información estimada de la posición de esos puntos, con base en la información de los parámetros de la cámara que se han calculado y que se conoce la posición espacial del patrón. Es por eso que se ocupa un patrón definido de esquinas.

El error de reproyección indica qué tan precisos son los parámetros calculados. El objetivo de esta calibración es minimizar ese error. Sin embargo, el error de reproyección dependerá de la capturas hechas al patrón de cuadros y la cantidad.

4.1.2. Filtrado de las imágenes

En principio, lo primero que se requiere hacer es aplicar filtros a las imágenes para eliminar ruido en ellas. Se aplica un filtro promediador para suavizar la imagen y eliminar la presencia de ruido gaussiano. Un filtro promediador ayuda a eliminar el ruido blanco.

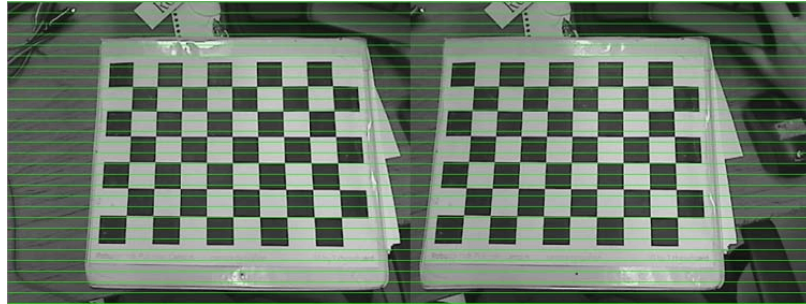


FIGURA 4.4: Ejemplo de dos imágenes rectificadas.

El efecto que produce este filtrado es un efecto de suavizado de la imagen. En la figura 4.5 se puede observar cómo funciona un filtro promediador para eliminar ruido gaussiano.



(a) Imagen original.

(b) Imagen original + ruido gaussiano.

(c) Imagen + ruido, filtrada.

FIGURA 4.5: Aplicación de un filtro Gaussiano.

4.1.3. Realce de imágenes

Después de conseguir que las imágenes pierdan artefactos indeseados que puedan afectar los resultados, ahora buscamos que los detalles puedan percibirse mejor realzando las imágenes. Algo que nos interesa es que los niveles de intensidades en ambas imágenes sea similar, por ello se decidió utilizar la especificación del histograma como medio para tratar de equilibrar las intensidades en las dos imágenes y así obtener un mejor empatado de píxeles, pues el empatado se realiza basado en las intensidades de las imágenes. Debido a que al empatar un punto de la imagen de referencia con un punto de la imagen sensada se busca aquel que se le asemeje en intensidad, entonces la especificación del histograma es útil en este sentido.

Lo que se busca es aplicar especificación del histograma a la imagen derecha (sensada), utilizando la información del histograma de la imagen izquierda (referencia). El primer paso es conocer los histogramas de las dos imágenes capturadas. Debido a que la imagen derecha está desplazada horizontalmente respecto de la imagen izquierda, la imagen izquierda contiene datos que no se encuentran en la imagen derecha, por ello se hace

un recorte de la imagen izquierda únicamente para calcular el histograma de la parte restante, pues esa parte es correspondiente con la imagen derecha.

Para la implementación de la especificación del histograma, en principio se debe calcular la función de densidad de probabilidad de los diferentes niveles de intensidad del histograma, por lo que se obtiene de la siguiente manera:

$$p(i) = \frac{h(i)}{N}; i = 0, \dots, N \quad (4.1)$$

$h(i)$: valor del histograma en la muestra i -ésima.

N : número de niveles de intensidad.

Una vez que se tiene la función de densidad de probabilidad de cada histograma, debe calcularse la función de distribución acumulada:

$$CDF(n) = \sum_{i=0}^n p(i) \quad (4.2)$$

Esto se aplica para cada histograma, h_r (histograma de la imagen de referencia) y h_s (histograma de la imagen sensada)

Ya que se han obtenido las funciones de distribución acumulada, ahora se aplica una tabla de búsqueda de tamaño N , y se llena aplicando el siguiente criterio:

$$LUT[i] = j; |h_r[i] - h_s[j]| \quad (4.3)$$

La tabla de búsqueda contiene las intensidades especificadas para cada valor de intensidad original, por así decirlo, donde:

$$I_e[i] = LUT[I[i]] \quad (4.4)$$

El valor de intensidad original en la imagen I es la posición de la tabla de búsqueda LUT para el nuevo valor especificado en I_e . Por ejemplo, supóngase que se tiene una imagen I de 8 niveles de intensidad, y que contiene 8 pixeles [4 5 3 4 0 1 2 4]; y una tabla de búsqueda de tamaño 8 que contiene la siguiente información [2 2 4 5 6 6 7 7]; entonces, la imagen I especificada estará compuesta por los pixeles [6 6 5 6 2 2 4 6].

La especificación del histograma es útil para realzar los niveles de intensidad de una imagen con base en otra. En nuestro caso en particular, se busca que la imagen sensada tenga una distribución de niveles de intensidad semejante a la imagen de referencia para que las intensidades en la imagen sean lo más similares posible. En la figura 4.6 se puede

observar la especificación sobre la imagen sensada y los resultados sobre el mapa de disparidad.

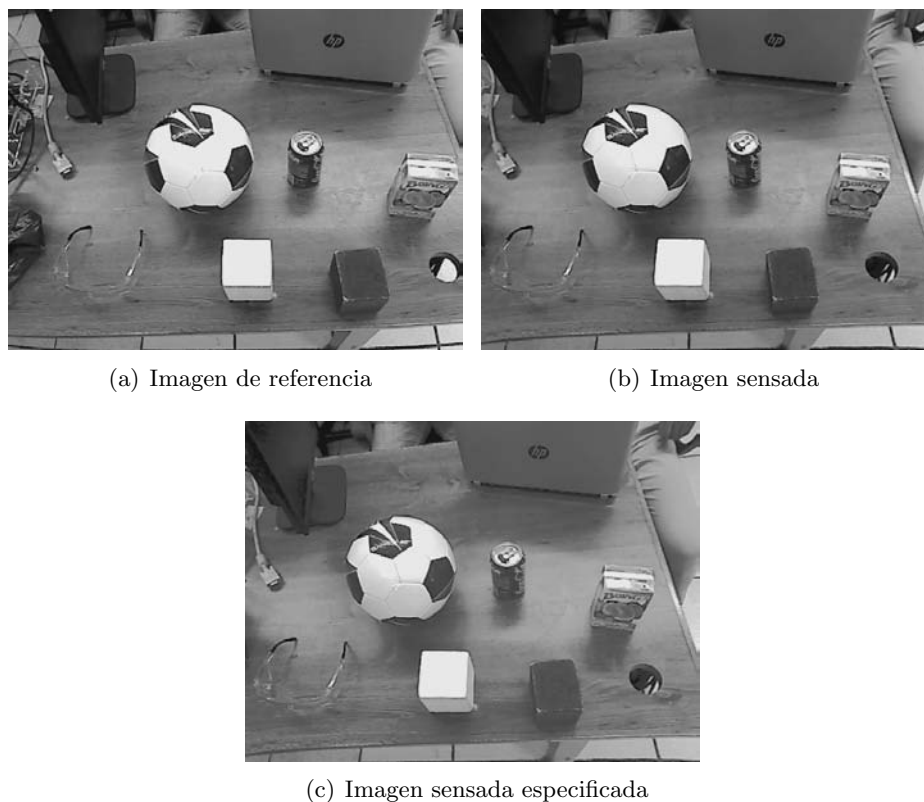


FIGURA 4.6: Implementación del realce de imagen por especificación del histograma. (a) y (b) muestran una captura estéreo, siendo las imágenes izquierda y derecha, respectivamente. (c) es la imagen sensada después de aplicar especificación del histograma.

4.1.4. Reescalamiento

Ahora bien, algo que queremos en el sistema de visión es que los resultados se obtengan en tiempo real. Cuanto mayor es la resolución de las imágenes, mayor es el tiempo de procesamiento que ocupa el algoritmo para buscar características, y posteriormente, hacer el *matching* de la imagen de referencia con la imagen sensada. Para ello, probaremos diferentes resoluciones de pares de imágenes para analizar con cuáles de ellas se obtienen mejores resultados en menores intervalos de tiempo.

Las cámaras utilizadas trabajan a una resolución de 720p (1280×720). Aunque la resolución original de las imágenes capturadas es 640×480 píxeles, puesto que es la resolución predeterminada en OpenCV. Se utilizará la resolución de 640×480 y a partir de ésta se harán los reescalamientos pertinentes para obtener resoluciones al 10 %, 20 %, 30 %, 40 %, 50 %, 60 %, 70 %, 80 % y 90 %, con base en la resolución predeterminada.

Como se puede notar, los factores de resolución de cada una de las escalas a las que se trabajará son factores fraccionarios, por ejemplo, el factor de escala de una resolución al 90 % es de 0,9, o lo que es lo mismo, $9/10$. Entonces, esto implica que el reescalamiento de una imagen original a un factor de escala de 0,9 no es posible con una decimación directa, sino que es necesario aplicar también una interpolación previa, ya que ambas operaciones (decimación e interpolación) requieren de factores enteros para realizarse. Por lo tanto, para obtener un factor de escala fraccionario se implementan ambas operaciones. Siguiendo con el ejemplo anterior, para obtener una escala de imagen a un 90 % será necesario aplicar una interpolación con un factor de 9, y posteriormente, una decimación con factor de 10.

4.2. Etapa de procesamiento

Este es el proceso más pesado del algoritmo del sistema de visión estereoscópica, pues una vez que se prepararon las imágenes ahora se pueden procesar para realizar el empatado de ellas. En la figura 4.7 se muestra el diagrama de bloques de esta etapa.

Se analizará un método basado en área para realizar la correspondencia estéreo, puesto que este tipo de métodos son mejores para obtener un mapa de disparidad denso y es lo que nos interesa. Los métodos de correspondencia basados en características hacen el registro empatando las características en ambas imágenes que deben coincidir al ser imágenes de una misma escena, sin embargo, el mapa de disparidad no es denso, al empatar únicamente algunos puntos de las imágenes.

4.2.1. Registro de imágenes

El registro de imágenes se hará con el algoritmo *Semi-Global Block Matching*, que utiliza un método local de búsqueda de correspondencias en conjunto con uno global para hacerlo más robusto.

En principio, es necesario que se conozcan los parámetros intrínsecos, como los extrínsecos, del sistema estéreo. También es necesario asegurar que las imágenes se encuentran rectificadas, lo cual simplifica el proceso de búsqueda de correspondencias para hacerlo unidimensional y aplicar las restricciones posibles.

La función de costo que utiliza el algoritmo SGBM es una medida de disimilitud por píxeles propuesta por Birchfield y Tomasi en [4], la cual se encarga de encontrar la correspondencia de píxeles a través de funciones de intensidad interpoladas alrededor de

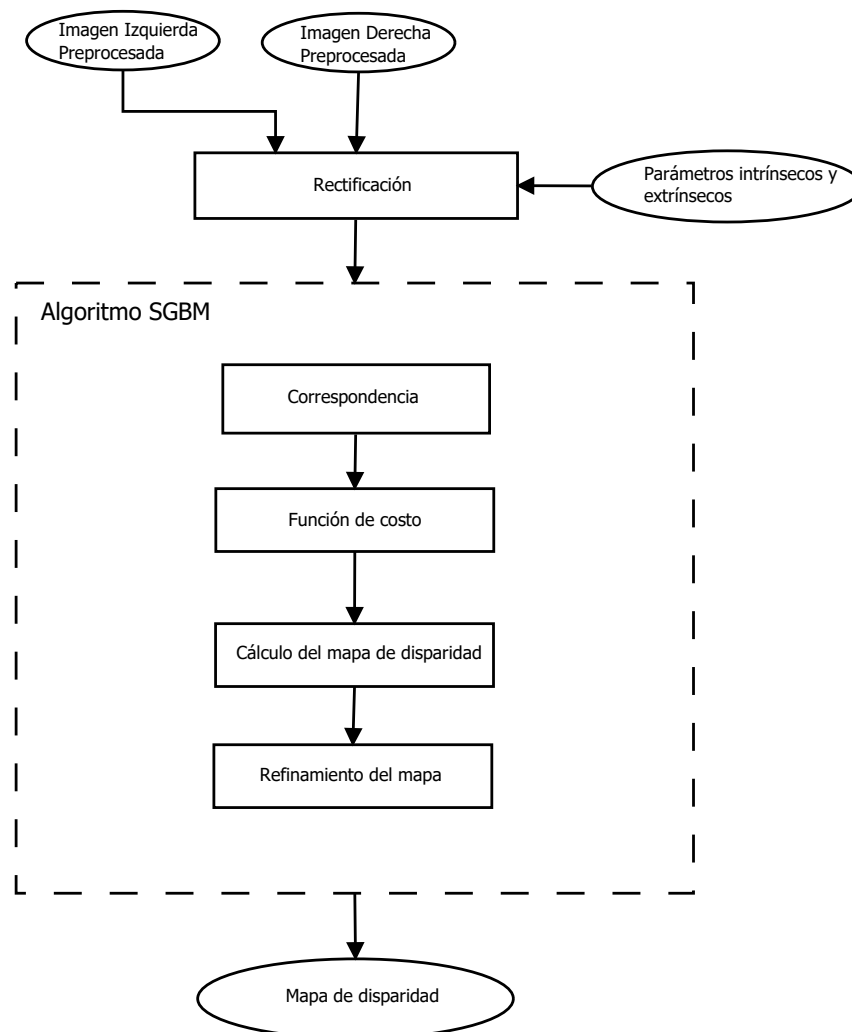


FIGURA 4.7: Diagrama de bloques de la etapa de procesamiento.

los píxeles que, aseguran, es insensible al muestreo de la imagen, es decir, no importan las condiciones.

Esta disimilitud de Birchfield y Tomasi es un método local de empatado, ahora bien, el método global que se implementa es utilizando una función de agregación de costo basada en una restricción de suavidad, expresada como función de energía.

4.2.2. Mapa de disparidad

El algoritmo SGBM obtiene el mapa de disparidad a partir de la información obtenida de los costos calculados. Es decir, al comparar píxeles de la imagen de referencia con la imagen sensada, se determina que aquellos que tengan el menor costo son los que tienen correspondencia, utilizando la información epipolar del sistema.

Además, para eliminar puntos indeseados (*outliers*) se calculan dos disparidades, una tomando como imagen de referencia la del lado izquierdo, y como imagen sensada la del derecho; la segunda disparidad se obtiene invirtiendo el papel de cada imagen, es decir que para la segunda disparidad, la imagen de referencia será la derecha, y la imagen sensada, la izquierda. Comparando ambas disparidades es posible determinar algunos errores de disparidad y oclusiones. En la figura 4.8 se muestra un ejemplo de un mapa de disparidad obtenido con el algoritmo mencionado.

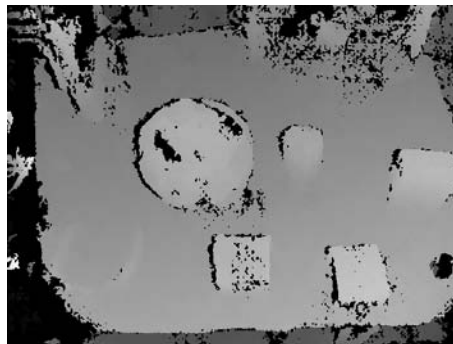


FIGURA 4.8: Ejemplo de mapa de disparidad obtenido con el algoritmo SGBM.

4.3. Etapa de postprocesamiento

En este último proceso, la obtención de la nube de puntos puede lograrse a partir de los datos que ya se han obtenido hasta ahora. El mapa de disparidad contiene la información de la profundidad a la que se encuentra cada píxel de la imagen de referencia, esto representado como una imagen de intensidades. Los valores de la imagen de intensidades se encuentran en el rango $[0 - 255]$, lo cual quiere decir que para los píxeles que se van alejando, su intensidad en el mapa de disparidad converge a 0, mientras que los que se encuentran más cerca, tienden a un valor de 255. Regularmente, la intensidad de un píxel en el mapa de disparidad es diferente de 0, a menos que éste no contenga información de la escena, ya sea, por oclusión o por un error en el empatao. El diagrama de bloques de la etapa de postprocesamiento puede observarse en la figura 4.9.

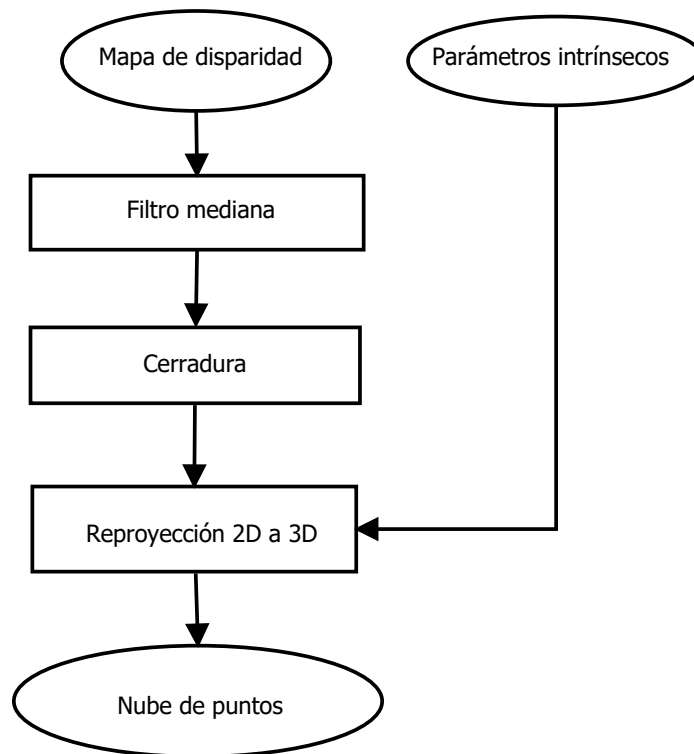


FIGURA 4.9: Diagrama de bloques de la etapa de postprocesamiento.

Para la obtención de la nube de puntos, es necesario tener la información del mapa de disparidad, además deben conocerse los parámetros intrínsecos de la cámara, específicamente la información del foco de la cámara y su centro, esto para tener una referencia.

Por último, una vez que se ha procesado el mapa de disparidad para lograr una nube de puntos, ésta puede tener algún procesamiento posterior con el objetivo de refinar los resultados y mostrar una nube de puntos más confiable. Para este caso, se propone un suavizador aplicado en el mapa de disparidad. Esto permitirá que las intensidades del mapa de disparidad se difuminen, considerando que la intensidad será el valor que indica la profundidad a la que se encuentra un objeto. Entonces, el suavizar la intensidad de los píxeles en el mapa de disparidad implicaría que se está suavizando la profundidad a la que se encontraría cada punto.

4.3.1. Procesamiento sobre el mapa de disparidad

A pesar de realizar la corrección de disparidades y evitar también las oclusiones, el mapa de disparidad obtenido puede seguir presentando huecos. Para corregir este problema, se propone aplicar un filtro de la mediana para eliminar los puntos específicamente negros que se encuentran dentro de una vecindad, sin perder de vista los contrastes de las intensidades, además, operaciones morfológicas de cerradura, para eliminar aquellos huecos más grandes.

El filtro de la mediana permite también eliminar aquellos pixeles que sobresalen de una mayoría, al hacer que el valor de éste sea el valor de alguno de la mayoría. Este filtro, además de eliminar esos puntos, permite mantener los contrastes de los pixeles que se encuentran alrededor de un borde, de manera que el borde no se pierde como sería el caso de un filtro promediador, que suaviza los valores del borde promediando los valores de intensidad de los pixeles que se encuentran alrededor. Se aplica un filtro de la mediana para mantener la intensidad de los bordes y así diferenciar profundidades. En la figura 4.10 se muestra el proceso que se le aplica al mapa de disparidad, la figura 4.10(b) es el resultado de aplicar un filtro mediana sobre el mapa de disparidad mostrado en la figura 4.10(a).

A continuación, se pueden emplear operaciones morfológicas sobre el mapa de disparidad para eliminar los huecos o reducir su tamaño. La operación que permite hacer la acción requerida es la *cerradura*. En primer lugar, se ejecuta una operación de dilatación que permite que los contornos se expandan, de tal manera que si hay huecos suficientemente pequeños, desaparecerán; o, si son grandes, sólo se reducirán. Para culminar la cerradura, se aplica una operación de erosión que reduce los contornos generados por la dilatación, pero los huecos rellenos se mantienen así. En la figura 4.10(c) se observa la imagen resultante.

Se aplicó la cerradura en 3 ocasiones para lograr el resultado mostrado. Se pueden observar superficies más uniformes y con menor cantidad de huecos. En la figura 4.10(d) se aplicó una ecualización del histograma para observar mejor el resultado de remover los huecos. Se puede notar que los contornos son ligeramente afectados por la cerradura.

4.3.2. Reproyección 2D a 3D

Hasta este punto se asume que el mapa de disparidad ha sido calculado y refinado, de tal modo que pueda ser utilizado para obtener la información en 3D de su reproyección. En este punto también se conocen los parámetros necesarios para realizar la transformación de la imagen. El mapa de disparidad contiene la información de profundidad, con la

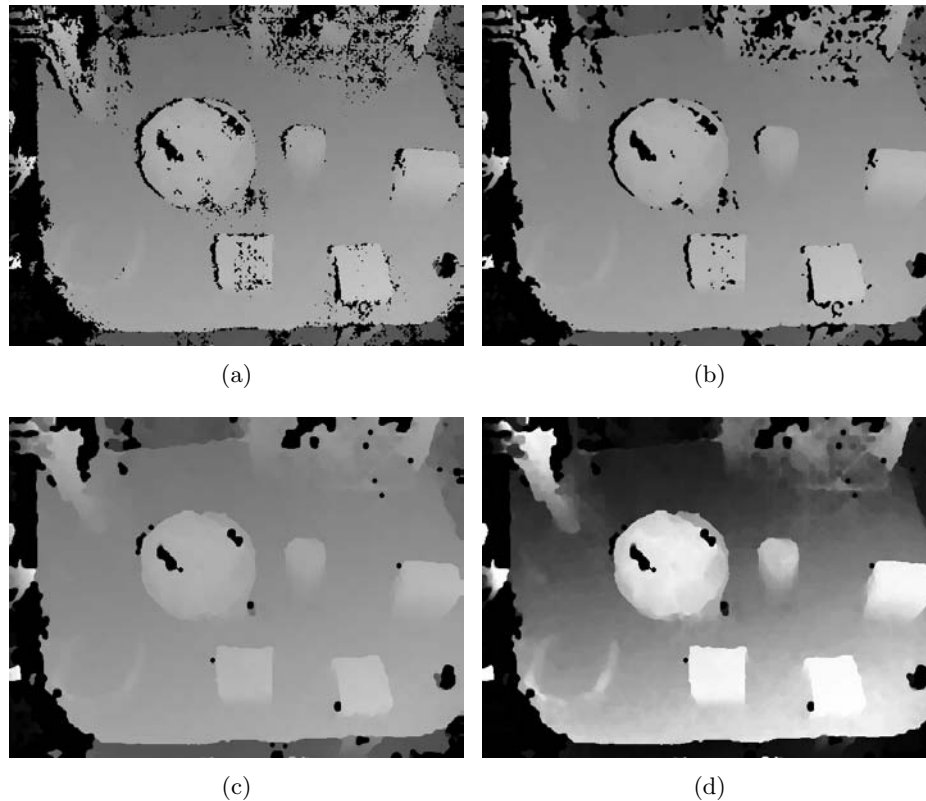


FIGURA 4.10: Proceso de remoción de huecos en el mapa de disparidad. En la imagen (a) se observa el mapa obtenido con el algoritmo de empatado, se puede notar que presenta huecos. A este mapa se le aplica un filtro mediana con un kernel de 5×5 y el resultado se puede ver en la imagen (b). Por último, se aplica una operación de cerradura para reducir aún más aquellos huecos. Al final del proceso, se obtiene la imagen (c). A esta última se le aplica una ecualización de histograma para observar los contrastes, que se puede ver en la imagen (d).

cual se obtienen los puntos en el espacio, al relacionar cada coordenada (x, y) con su correspondiente intensidad d . El proceso para calcular la reproyección se detalla en el *Apéndice B*.

Para visualización de la nube de puntos, se realiza un mapeo del mapa de disparidad con la imagen de referencia del sistema estéreo, de este modo, podemos obtener puntos en 3D con componentes RGB, como se puede ver en la figura 4.11.

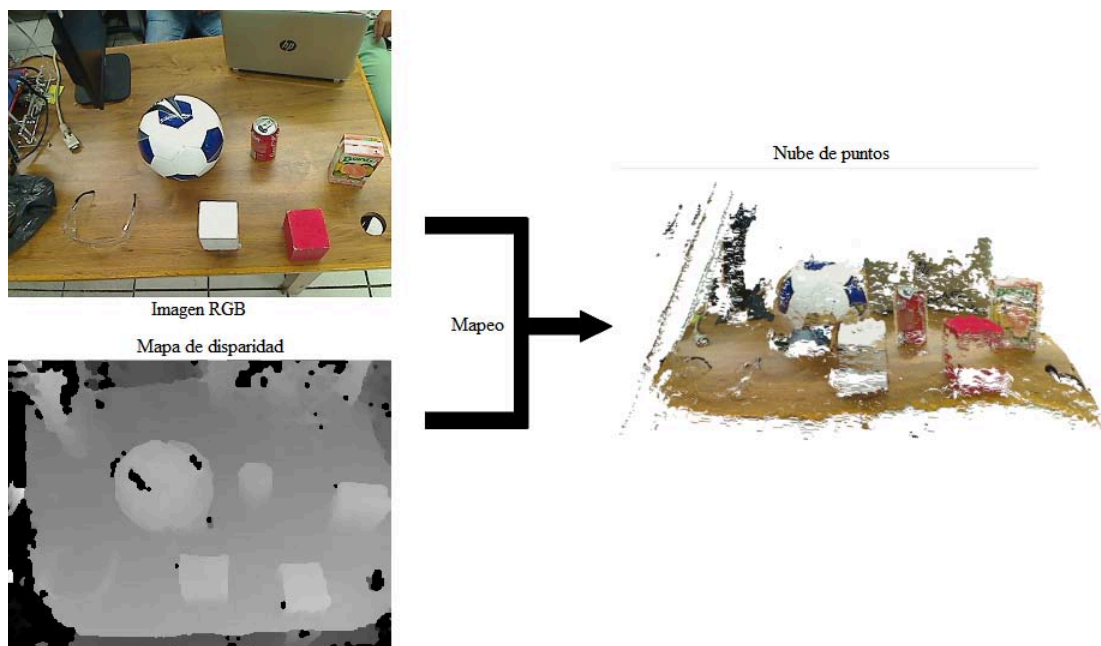


FIGURA 4.11: Mapeo de la imagen RGB sobre el mapa de disparidad.

Capítulo 5

Pruebas y resultados

En este capítulo se mostrarán los resultados obtenidos con el sistema de visión estéreo realizado. Las respectivas pruebas se realizaron con 3 diferentes equipos de cómputo capaces de ejecutar el programa hecho. Cabe mencionar que este programa se realizó en el lenguaje de programación C++ y con ayuda de las bibliotecas de OpenCV y PCL (*Point Cloud Library*), esta última para la visualización de la nube de puntos obtenida por el sistema estéreo.

Las características relevantes de los equipos comparados se muestran en la tabla ???. Por comodidad y simplicidad, se enumeraron en 1, 2 y 3. El equipo 1 es el que corresponde al robot NimbRo-OP de fábrica, el equipo 2 puede ser adaptado al robot y, por último, el equipo 3 corresponde a una computadora personal que sirve como referencia para la ejecución del sistema.

	Equipo 1	Equipo 2	Equipo 3
Procesador	AMD E-450 Dual-Core a 1.65 GHz	AMD A4 Quad-Core a 1.6 GHz	Intel Core i7 Quad-Core a 2.4 GHz
Memoria RAM	2 GB	4 GB	8 GB
Gráficos	AMD Radeon HD	AMD Radeon R3	NVIDIA GTX 770
Sistema Operativo	Linux KUBUNTU 12.10	Linux LUBUNTU 12.10	Windows 8.1/10

TABLA 5.1: Características de los equipos de cómputo comparados

5.1. Análisis

Cabe mencionar que este sistema de visión estéreo está hecho para que funcione en tiempo real, para lo cual necesitamos un equipo de cómputo capaz de ejecutar el algoritmo de manera rápida y eficiente. El resultado que se obtuvo fue un algoritmo que lograra calcular una nube de puntos en tiempo real para procesamiento de la misma.

Para fines prácticos, se realizaron las siguientes pruebas:

- Análisis del empatao de imágenes y obtención del mapa de disparidad a diferentes escalas.
- Tiempo de obtención del mapa de disparidad en cada una de las escalas comparadas.
- Comparación de los tiempos calculados en diferentes equipos de cómputo.
- Análisis de nubes de puntos obtenidas con 2 sistemas diferentes: sistema de visión estéreo presentado sensor RGB-D Microsoft Kinect.

5.1.1. Empatado de imágenes y mapa de disparidad

Para comenzar con el empatao de las imágenes es necesario preparar las imágenes para realizarlo, esto con el fin de que se obtengan buenos resultados. En la figura 5.1 se observa un ejemplo de una escena que fue capturada con el sistema de visión estéreo.

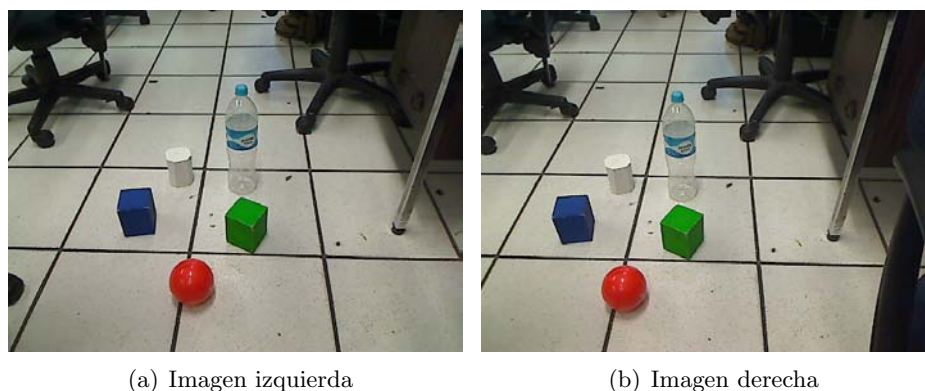


FIGURA 5.1: Ejemplo de captura de una escena con visión estéreo.

El empatao de imágenes se realiza con el algoritmo SGBM de la biblioteca de OpenCV, descrito en el *Apéndice A*. En OpenCV se implementa una variante del algoritmo SGM de Hirschmüller, en [18], sin embargo, el algoritmo de Hirschmüller tiene mejor precisión al realizar el empatao pixel a pixel, puesto que SGBM lo realiza por bloques para

optimización. Aunque la búsqueda pixel a pixel es más tardada computacionalmente, presenta mejores resultados. Adaptando el algoritmo SGBM a los parámetros necesarios para realizar empatado pixel a pixel, se puede presentar el mapa de disparidad de la figura 5.2.

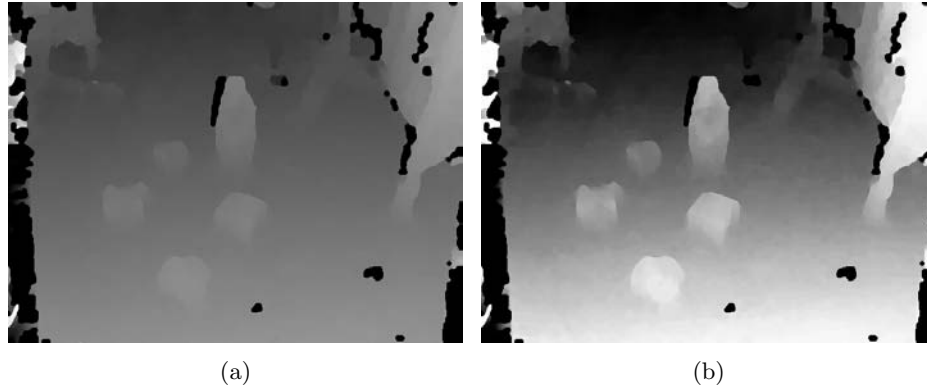


FIGURA 5.2: Mapa de disparidad. (a) es el mapa de disparidad obtenido de la escena 5.1, (b) es el mismo mapa de (a) ecualizado a 64 niveles de cuantización.

El mapa de disparidad ecualizado que se observa en la figura 5.2(b) muestra, con mayor contraste, las variaciones que existen en el mapa de disparidad de la figura 5.2(a). Se puede observar que el plano que contiene los objetos no es totalmente uniforme (en intensidad).

Utilizando la reproyección de 2D a 3D del mapa de disparidad, considerando la intensidad de cada pixel de éste como el valor de profundidad, y teniendo en cuenta los parámetros intrínsecos y extrínsecos del sistema, lo cual se explica en el *Apéndice B*, se puede llegar a obtener una nube de puntos, como la que se muestra en la figura 5.3.

5.1.2. Análisis a diferentes escalas

Las capturas realizadas con las cámaras se obtienen a una resolución de 640 pixeles de ancho por 480 pixeles de alto. Sin embargo, cabe mencionar que las cámaras utilizadas permiten una resolución de 1280×720 pero, cuanto más grande sea la resolución, mayor es el tiempo de procesamiento del sistema, por ello se ha decidido utilizar una resolución inicial de 640×480 , que es el más común en OpenCV. La relación de escalas analizadas se observa en la tabla 5.2. Esta será la relación que se ocupará durante el proceso de las pruebas del sistema de visión estéreo.

Reducir la escala de las imágenes implica que el procesamiento de ellas puede ser más rápido, pues se reduce la cantidad de información. Pero también implica que se van reduciendo el número de características detectables en ellas, pues al pasar por un proceso

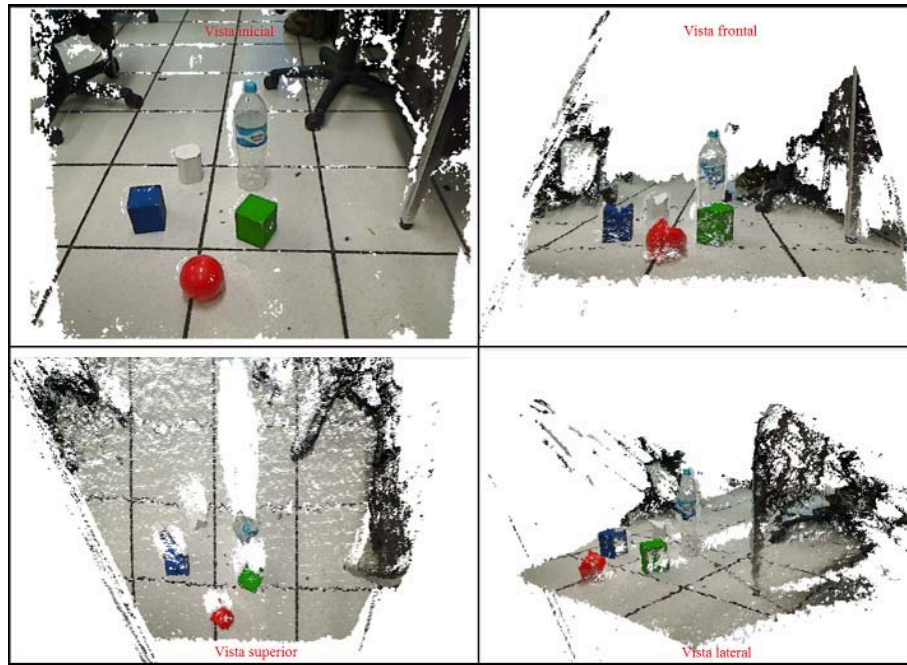


FIGURA 5.3: Ejemplo de una nube de puntos obtenida.

Porcentaje de escala	Factor de escala	Resolución <i>ancho</i> × <i>alto</i> [píxeles]
100 %	1,0	640 × 480
90 %	0,9	576 × 432
80 %	0,8	512 × 384
70 %	0,7	448 × 336
60 %	0,6	384 × 288
50 %	0,5	320 × 240
40 %	0,4	256 × 192
30 %	0,3	192 × 144
20 %	0,2	128 × 96
10 %	0,1	64 × 48

TABLA 5.2: Relación escala-resolución de las imágenes

de reescalamiento, hay pérdida de información al reducir la tasa de muestreo. En la figura 5.4 se puede observar la imagen de una escena en las diferentes escalas a analizar, nótese que algunas imágenes no perciben cambios, pero esto es debido a que la impresión en este documento redimensiona las imágenes y la mayoría de ellas tienen más resolución que la mostrada aquí.

Ahora bien, una vez definidas las escalas que se usarán, se deben obtener los parámetros intrínsecos y extrínsecos para cada una de las escalas, pues estos parámetros dependen de esas escalas. En el caso de los parámetros intrínsecos, sólo es suficiente calcular una vez para la escala de 100 % pues el cálculo es proporcional a la escala. Es decir, si tenemos los parámetros intrínsecos que corresponden a las imágenes de escala a 100 %, entonces sólo podemos multiplicar esos parámetros por un factor equivalente a la escala que se



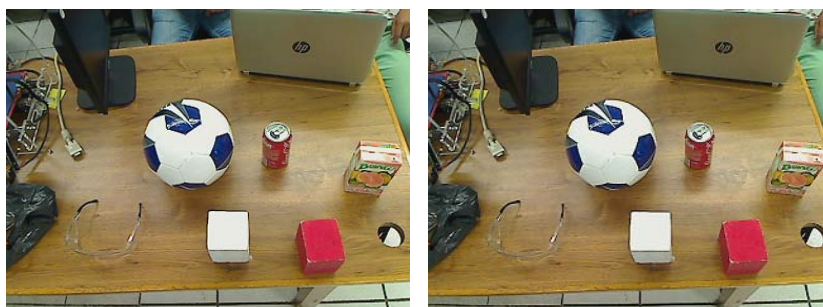
(a) Imagen a escala 100 %.

(b) Imagen a escala 90 %.



(c) Imagen a escala 80 %.

(d) Imagen a escala 70 %.



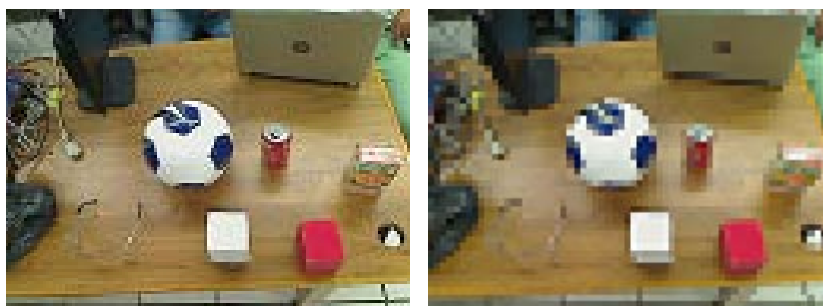
(e) Imagen a escala 60 %.

(f) Imagen a escala 50 %.



(g) Imagen a escala 40 %.

(h) Imagen a escala 30 %.



(i) Imagen a escala 20 %.

(j) Imagen a escala 10 %.

FIGURA 5.4: Capturas de una misma imagen a escalas diferentes.

quiera procesar. Para los parámetros extrínsecos, es más complicado, así que es preferible realizar la calibración con el patrón de cuadros a la escala necesaria. Una vez calculados los parámetros de las cámaras, ahora es posible obtener el mapa de disparidad.

Algo importante que hay que destacar es el cálculo del error de reproyección para cada escala determinada. Se hizo un análisis de éste y en la gráfica que se muestra en la figura 5.5 se puede ver su comportamiento. Se hicieron 30 capturas del patrón de cuadros y las 30 fueron válidas.

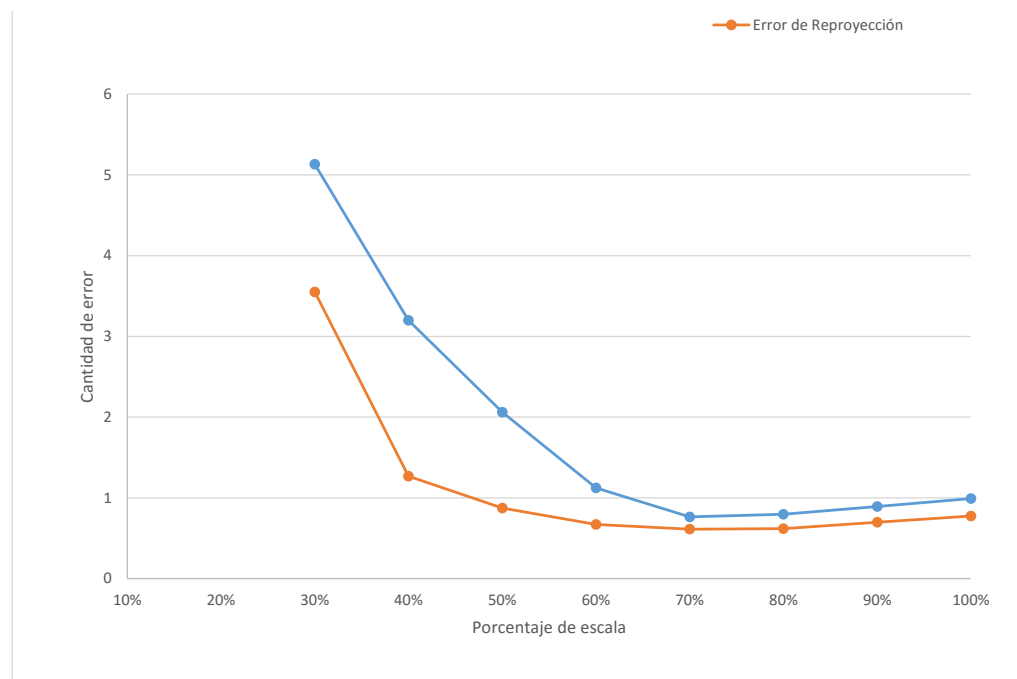


FIGURA 5.5: Gráfica del cálculo del error de reproyección y error RMS a diferentes escalas de la escena de prueba.

Obteniendo los errores de reproyección en la calibración del sistema a diferentes escalas, podemos determinar la escala en la que se pueden presentar mejores correspondencias. Para este caso en particular, se obtuvo que el menor error se presenta en la escala de 70%, con un error RMS de 0,764751 y un error de reproyección de 0,612085.

5.1.3. Obtención de nubes de puntos

Una vez que se han obtenido los mapas de disparidad a diferentes escalas, es posible calcular la nube de puntos a partir de la reproyección del mapa a coordenadas en 3D. Como ya se ha mencionado, para obtener la reproyección es necesario conocer los parámetros de las cámaras, que evidentemente son conocidos, puesto que ya se realizó un mapa de disparidad. En la tabla 5.4 se observan diferentes nubes de puntos obtenidas de la escena a diferentes escalas. Nótese que conforme disminuye la escala, la nube de puntos se va


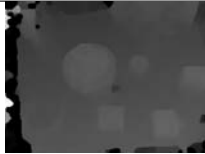








Factor de escala	Mapa de disparidad	Factor de escala	Mapa de disparidad
1.0		0.5	
0.9		0.4	
0.8		0.3	
0.7		0.2	
0.6		0.1	

TABLA 5.3: Resultados en diferentes escalas. Para escalas menores a 40 % no se obtiene un mapa de disparidad bien definido.

degradando, esto se debe a que la cantidad de puntos obtenidos disminuye conforme la escala lo hace. Es decir, en el caso de una resolución del 100 % se tendrán, máximo, $640 \times 480 = 307,200$ puntos en la nube. Para una resolución del 70 % se tendrán $448 \times 336 = 150,528$.

5.2. Comparación

Los puntos de comparación serán los diferentes equipos de cómputo utilizados y el tiempo de procesamiento que toma el calcular el mapa de disparidad, que es el proceso más pesado del sistema. Además se busca hacer un análisis de comparación entre diferentes nubes de puntos obtenidas con ayuda de diferentes sistemas, lo cuales son: sistema de visión estéreo presentado y *Kinect*








Escala	Nube de puntos	Máx. puntos	Escala	Nube de puntos	Máx. puntos
100 %		307,200	50 %		76,800
90 %		248,832	40 %		49,152
80 %		196,608	30 %	_____	_____
70 %		150,528	20 %	_____	_____
60 %		100,592	10 %	_____	_____

TABLA 5.4: Nubes de puntos a diferentes escalas. Para escala menor a 40 % no se muestra nube porque no se obtiene un mapa de disparidad bien definido. La densidad de puntos es proporcional a la escala. La cantidad de puntos es máxima, pero no necesariamente es la que hay en la nube, pues no existen puntos en las oclusiones o en donde hubo errores de disparidad.

5.2.1. Tiempos de ejecución a diferentes escalas

Se analizó el tiempo de procesamiento del mapa de disparidad a diferentes escalas. Considerando que las capturas obtenidas por el sistema tienen una resolución de 640 píxeles de ancho por 480 píxeles de alto.

En la tabla 5.5 se pueden observar los tiempos promedio de ejecución en cada uno de los equipos de cómputo comparados para obtener el mapa de disparidad de una escena. En este caso, se analiza el tiempo promedio de 12 muestras para calcular el mapa de disparidad de la escena de la figura 5.6. En algunas celdas de dicha tabla, se muestra la leyenda N/A, esto quiere decir que los datos percibidos del mapa de disparidad en estos casos no produjeron una nube de puntos legible, puesto que en esas escalas no es posible observar un mapa de disparidad bien definido. Esto se debe a que a escalas más pequeñas la información de las imágenes se acumula en un menor grupo de píxeles.

Como se puede ver, los tiempos de procesamiento del mapa de disparidad son los esperados. Aunque, en el equipo de cómputo 2 no se obtuvo un mapa de disparidad definido

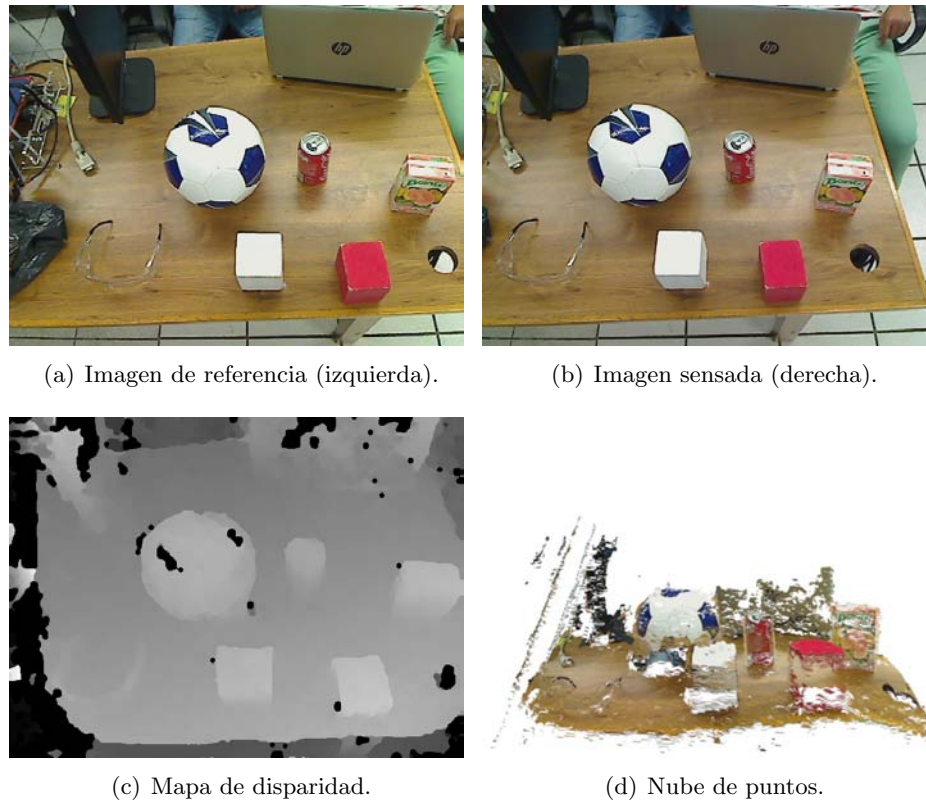


FIGURA 5.6: Capturas de una escena con visión estereoscópica.

	Tiempo promedio por escala [ms]									
	100 %	90 %	80 %	70 %	60 %	50 %	40 %	30 %	20 %	10 %
Equipo 1	1493	1185	920	663	469	295	153	N/A	N/A	N/A
Equipo 2	1167	863	639	463	344	234	N/A	N/A	N/A	N/A
Equipo 3	246	199	157	119	87	61	37	N/A	N/A	N/A

TABLA 5.5: Promedio de tiempo de cálculo para el mapa de disparidad a diferentes escalas

para una escala de 40 %. Esto puede deberse a la precisión de los datos obtenidos en la ejecución de la calibración para una escala de 40 %.

En la figura 5.7 se puede ver el comportamiento de los tiempos de ejecución para calcular el mapa de disparidad en cada uno de los equipos de cómputo. Se observa que el que mejor desempeño tiene es el equipo 3. Los equipos 1 y 2 tienen un comportamiento similar. En todos los casos, el tiempo de procesamiento del mapa de disparidad es exponencial de acuerdo a la resolución de las imágenes utilizadas.

5.2.2. Comparación de nubes de puntos

Como punto de referencia, el sistema de visión estéreo elaborado puede compararse a otros sistemas que proporcionan información de nubes de puntos. Por ejemplo, existe un

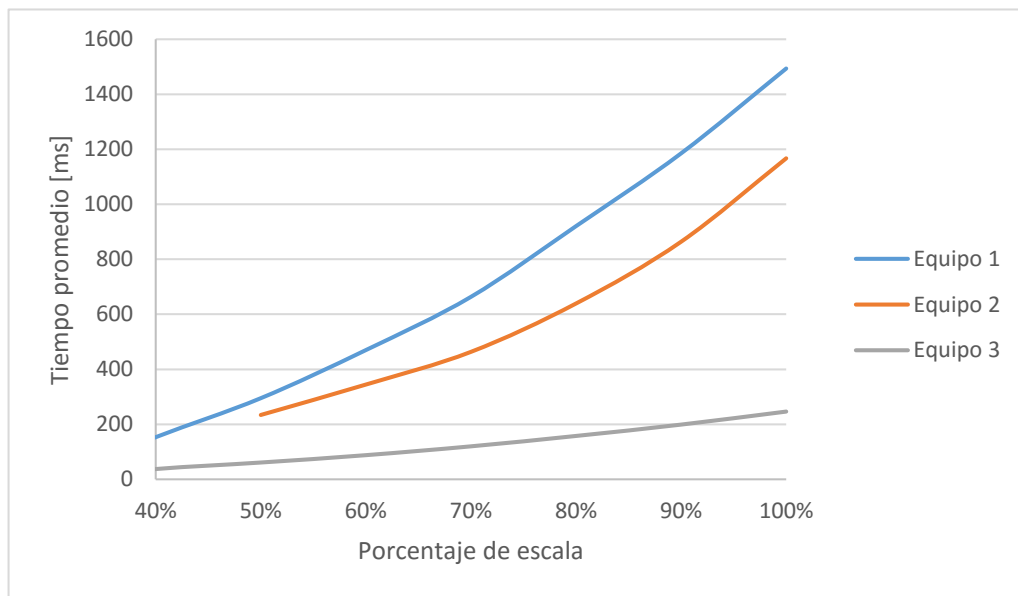


FIGURA 5.7: Gráfica de tiempo promedio para procesamiento del mapa de disparidad. Se observan los tiempos promedios por cada escala a partir de las que se obtienen datos de disparidad definidos.

sistema de visión estéreo comercial que se llama *Minoru 3D Webcam*¹ que es un sistema de dos cámaras fijas, separadas horizontalmente a una distancia de alrededor de 6 cm. y que puede utilizarse para obtener una nube de puntos de la misma manera que se presenta en este trabajo escrito. Sin embargo, la resolución máxima de las imágenes que se pueden obtener con este sistema es de 320×240 .

Otro ejemplo de un sistema para obtener nubes de puntos es el sistema *Kinect* de la empresa Microsoft, se considera un dispositivo RGB-D pues contiene un sensor RGB (cámara) y funciona con un sensor de luz infrarroja que permite calcular la distancia a la que se encuentra un objeto, ya que proyecta un patrón de luz infrarroja y se calcula el tiempo que tarda en regresar al dispositivo. Con este sensor, es posible obtener información de profundidad de casi cada pixel de la imagen RGB con una buena precisión. Sin embargo, éste último se considera un sensor activo, pues proyecta un patrón sobre la escena. En la figura 5.8 se pueden ver las nubes obtenidas por dos sistemas distintos: la figura 5.8(a) es la nube del sistema estéreo realizado y la figura 5.8(b) fue obtenida con un sistema Kinect, ambas imágenes son de la misma escena.

¹Para más detalles, referirse a la siguiente página web <http://www.minoru3d.com/>



(a) Sistema de visión estéreo realizado.



(b) Sistema Kinect de Microsoft.

FIGURA 5.8: Nubes de puntos obtenidas con 2 sistemas diferentes.

5.3. Pruebas de procesamiento de la nube de puntos

En esta etapa de las pruebas se realizó un procesamiento de la nube de puntos con un algoritmo relativamente sencillo para obtener el plano más grande que se encuentra en dicha nube. Es decir, encontrar aquellos puntos que pertenecen a un plano sobresaliente de la nube. Se utilizó el algoritmo RANSAC (*RAN*d*om* *S*A*m*ple *C*onsensus) para encontrar el plano que contenga la mayor cantidad de puntos de la nube procesada.

Las pruebas realizadas se hicieron sobre la escena mostrada en la figura 5.6. Se puede notar a simple vista que el plano más sobresaliente y el que se busca en esa escena es el plano de la mesa, y donde posan los objetos situados. Entre los argumentos de entrada de la función que realiza el algoritmo RANSAC en el código creado destacan:

- Un umbral de distancia de los puntos al modelo del plano de 0.5 unidades.
- El número de iteraciones del algoritmo para encontrar el mejor modelo del plano es de 30.

Ahora, se analiza el tiempo de procesamiento en cada equipo de cómputo revisado a diferentes escalas, pues la escala define la cantidad de puntos en la nube y los cuales serán procesados, este análisis de tiempos se muestra en la tabla 5.6. Al igual que en la tabla 5.5, se analizaron 12 muestras y la leyenda N/A indica que los datos de la

disparidad obtenida a estas escalas no fueron lo suficientemente legibles para procesar una nube bien definida.

	Tiempo promedio por cada factor de escala [ms]									
	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
Equipo 1	212	175	140	103	79	52	28	N/A	N/A	N/A
Equipo 2	1948	1539	1193	947	729	521	N/A	N/A	N/A	N/A
Equipo 3	73	61	48	38	32	21	13	N/A	N/A	N/A

TABLA 5.6: Promedio de tiempo de procesamiento de la nube de puntos para obtener un plano. Utilizando el método RANSAC que se aproxime al plano más grande de la escena, con un umbral de 0.5 unidades y 30 iteraciones.

La figura 5.9 muestra los tiempos promedios de la ejecución del algoritmo RANSAC para calcular el plano que contenga la mayor cantidad de puntos posibles en la nube de puntos procesada. Aquí ocurre un fenómeno inesperado, pues el equipo de cómputo número 2, que se asemeja en características al equipo 1, pero en teoría con un mejor hardware, tiene problemas al procesar la nube de puntos. Los tiempos que tarda en obtener el plano con RANSAC son 10 veces más tardados que el equipo de cómputo 1. En principio, se piensa que puede deberse a la arquitectura del sistema operativo, ya que el mismo algoritmo ha sido ejecutado en los 3 equipos de cómputo, utilizando las mismas entradas.

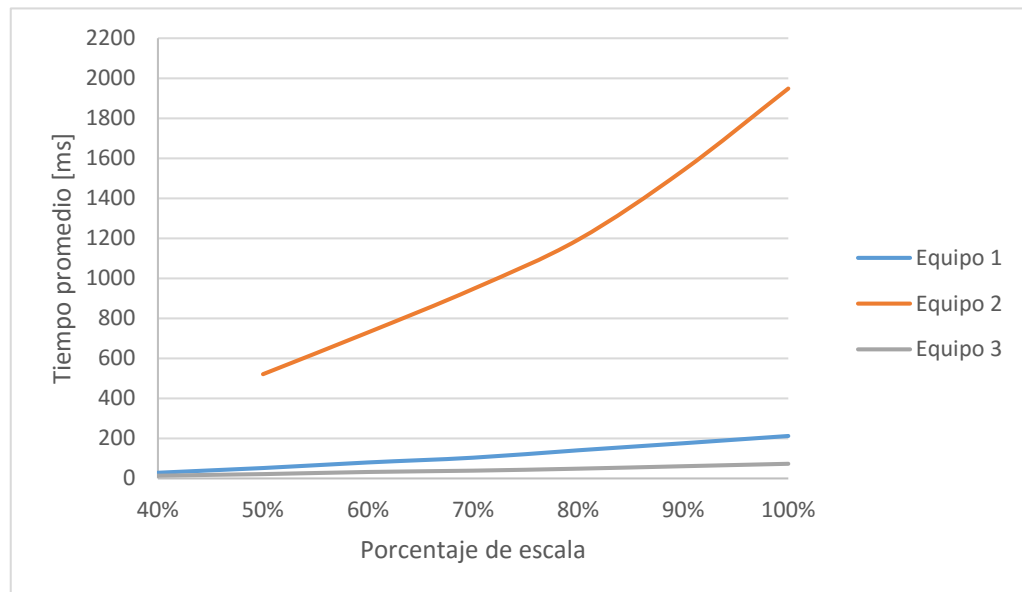


FIGURA 5.9: Gráfica de tiempos promedio para procesamiento de la nube de puntos. Se observan los tiempos promedio por cada escala a partir de las que se obtienen datos de disparidad definidos. Puede notarse que en el Equipo de cómputo 2 el tiempo de procesamiento es mucho mayor respecto a los otros dos.

El resultado de una nube de puntos a la cual se ha extraído el plano se puede ver en la figura 5.10. Utilizando RANSAC se puede obtener el plano. Sin embargo, dado que el plano de la nube no es completamente uniforme, existen falsos positivos al calcularlo.

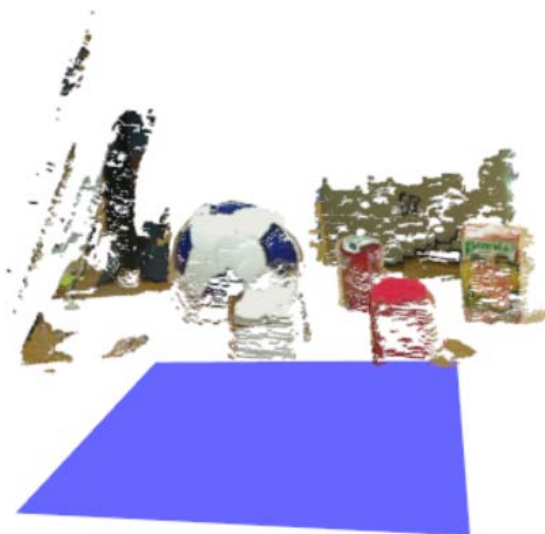


FIGURA 5.10: Cálculo del plano en una nube de puntos. Al calcular el plano, se extraen todos los puntos que cumplen con el modelo del plano y que están a una distancia de él. Además, se eliminan todos los demás puntos que se encuentran por debajo de ese plano, pues los puntos de interés son los que se encuentran por encima.

5.4. Otras pruebas

En este apartado, se realizan algunas pruebas más de importancia. Principalmente, la prueba que aquí se realiza es con el fin de determinar la robustez del algoritmo de empatado en diferentes tipos de superficie.

5.4.1. Análisis en superficies con textura y uniformes

En esta prueba se realiza un análisis del mapa de disparidad obtenido sobre una superficie con textura y otro mapa con una superficie uniforme. La prueba consiste en colocar objetos sobre ambas. La obtención del mapa de disparidad confirmará la presencia de la superficie. Dado que el empatado se realiza pixel a pixel, en una superficie texturizada no debería haber ningún problema, ya que la textura permite que la probabilidad de encontrar un pixel de una imagen en otra es mayor debido a las intensidades de la textura. Pero si la superficie es uniforme, entonces la intensidad también lo es y eso afecta los resultados obtenidos del empatado. En la figura 5.11, se observa la escena con

una superficie con textura y su mapa de disparidad. Claramente se observa que no hay ningún problema para encontrar la superficie y las disparidades de una imagen a otra.

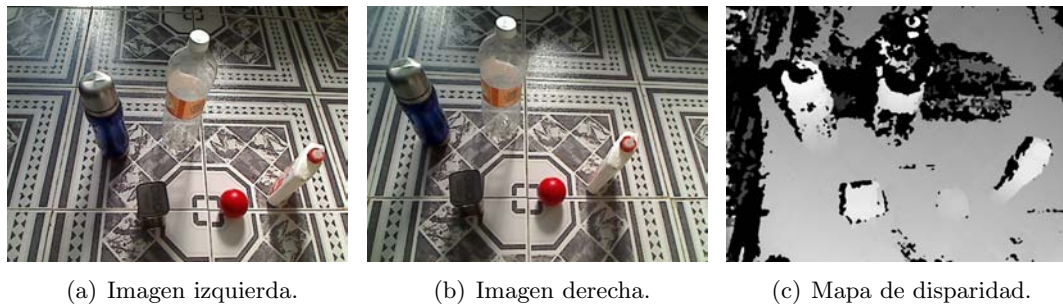


FIGURA 5.11: Escena con una superficie texturizada.

Sin embargo, en la figura 5.12, se tiene una escena en donde se presenta una superficie uniforme, que a pesar de contener objetos sobre ella, el algoritmo de empatado tiene conflictos para encontrar la disparidad en la zona uniforme.



FIGURA 5.12: Escena con una superficie uniforme.

Capítulo 6

Conclusiones

El uso de sistemas de visión estéreo en la robótica tiene ciertas ventajas y desventajas. A diferencia de sistemas de visión en 3D como las cámaras RGB-D o las de tiempo de vuelo, un sistema de visión estéreo no necesita emitir algún patrón que deba ser captado por algún sensor independiente de las cámaras, sino que es suficiente con procesar ambas imágenes y relacionarlas entre sí.

La mayor ventaja reside en que un sistema estéreo es más robusto a efectos de luz ambiental, esto es que los sistemas RGB-D suelen tener conflictos cuando se exponen al aire libre, pues la luz solar produce luz con frecuencias infrarrojas, al igual que las cámaras de tiempo de vuelo, que perciben el ruido generado por la luz ambiental.

Las principales desventajas con respecto a los sistemas mencionados están en que un sistema de visión estéreo pasivo requiere un procesamiento extra al obtener la disparidad entre las imágenes para conocer la profundidad a la que se encuentran los objetos. Utilizar una función de costo para la correspondencia de las imágenes, comparar ambas imágenes, calcular el mapa de disparidad y reproyectar las imágenes 2D a 3D, son los pasos del proceso para obtener una nube de puntos con visión estéreo; mientras que con cámaras RGB-D o tiempo de vuelo se obtiene la información de profundidad únicamente con los datos capturados de los sensores pertinentes y con ellos se calcula la nube de puntos. Otra desventaja se presenta cuando en un sistema de visión estéreo se observa un patrón uniforme y que puede ser periódico, este efecto puede producir que haya una descripción de características inadecuada o una disparidad imprecisa y que la correspondencia de imágenes devuelva falsos positivos.

En general, la hipótesis inicial se cumplió, salvo por algunas inconvenientes. Aunque fue posible realizar un sistema de visión estéreo para conseguir una nube de puntos, que funcione en tiempo real y con recursos de cómputo limitados, el sistema no se

implementó en el robot jugando soccer debido a la falta de tiempo y algunos recursos materiales. Además, puesto que el cómputo del robot no está centrado en la visión, esto ralentizaría la ejecución de cada módulo. Aún así, podría resaltar el hecho de que se logró realizar el sistema utilizando cámaras comunes, sin necesidad de un sistema especializado de visión estereoscópica.

Otra observación es que la nube de puntos resultante puede procesarse, como se mostró en una de las pruebas, para obtener el mejor plano del conjunto de puntos, aunque un sistema como Kinect puede obtener información aún más precisa de las superficies del entorno. Por ejemplo, en una nube de puntos de Kinect es posible calcular el mejor plano en ella con una menor cantidad de iteraciones del algoritmo RANSAC y con un umbral más pequeño. La ventaja del sistema estéreo realizado sobre el sistema Kinect es que el primero es pasivo, de modo que permitiría la interacción de varios agentes con el mismo sistema, lo cual no sucedería con Kinect, pues utiliza sensores activos que interferirían en el entorno y podrían afectar la lectura de otro sistema similar.

6.1. Trabajo futuro

La visión estereoscópica es un tema que sigue en desarrollo en el área de Visión Computacional. El hecho de intentar simular la visión humana para que un robot pueda lograr observar lo que nosotros podemos, da pie a que en un futuro habrá sistemas aplicados que puedan lograrlo. Sin embargo, sistemas de cámaras RGB-D o de tiempo de vuelo han tenido mejores resultados incluso que la visión estéreo, que ha hecho que este tipo de visión quede un poco de lado y se opte por utilizar los dos primeros, sobre todo en competencias como RoboCup.

Muchos métodos para lograr un buen sistema de visión estereoscópica se han realizado en los últimos años. Varias técnicas como las basadas en áreas o en píxeles han sido optimizadas para conseguir un sistema que funcione en tiempo real, incluso sistemas de hardware dedicado a obtener información 3D con visión estereoscópica se han desarrollado. Tal es el caso del sistema de hardware creado por Konolige [25] el cual muestra buenos resultados y se utiliza como base para la fabricación de otros. También es el caso del sistema de visión *Tyxx DeepSea G2*, presentado en [45], que funciona en tiempo real utilizando una arquitectura creada con un FPGA y un DSP. Incluso la versión G3 del mismo se dio a conocer en 2009.

Una mejora sustancial en la visión estéreo será el empatado de imágenes con mayor resolución. Esto permitirá que la correspondencia de imágenes sea más precisa. Sin embargo, esto está limitado por el hardware utilizado. Si bien, casi cualquier equipo de

cómputo es capaz de emparar dos imágenes, esto se vuelve un problema cuando se trata de implementar en tiempo real. Por un lado, las necesidades de procesamiento de las imágenes; y por otro, la complejidad de los algoritmos utilizados. Ambos factores se ven perjudicados cuando la resolución de una imagen es mayor. Tal vez el uso de sistemas de hardware dedicados, como GPU, DSP o FPGA permitan acelerar el procesamiento en tiempo real.

Apéndice A

Algoritmo SGBM para el empatado de imágenes estéreo.

El algoritmo SGBM (*Semi-Global Block Matching*) implementado en las bibliotecas de OpenCV y que está basado en [18], es un algoritmo que se encarga de empatar dos imágenes, combinando métodos locales y globales (de empatado). A continuación se explica a grandes rasgos el funcionamiento del algoritmo SGBM comenzando con el algoritmo SGM descrito en el artículo citado y, posteriormente, las variantes que hay respecto a la implementación en OpenCV. Para una revisión más a detalle del algoritmo SGM o de su implementación en OpenCV, referirse al artículo o a la documentación de las bibliotecas¹, respectivamente.

El algoritmo SGM (*Semiglobal Matching*) es un algoritmo que implementa un método de empatado local combinado con uno global. Está basado en un empatado por píxeles utilizando como medida de costo de empatado la información mutua [43].

El proceso se explica más o menos en la figura A.1. Se basa en la estructura de un empatado de imágenes propuesta en [38], en la que se dividen los métodos en 4 pasos importantes: costo del empatado, agregación del costo, obtención de la disparidad y refinamiento de la misma. Sin embargo, Hirschmüller explica que algunos pasos podrían ser opcionales dependiendo de la aplicación. Se da por hecho que la etapa de preprocesamiento para mejorar las imágenes fue hecha o no es necesaria, también que la geometría epipolar se ha calculado y que las imágenes se encuentran rectificadas.

a) Costo de empatado

En primer lugar, para el costo del empatado se asume que la geometría epipolar de las imágenes es conocida. El costo de empatado se calcula tomando como base un

¹<http://docs.opencv.org/>

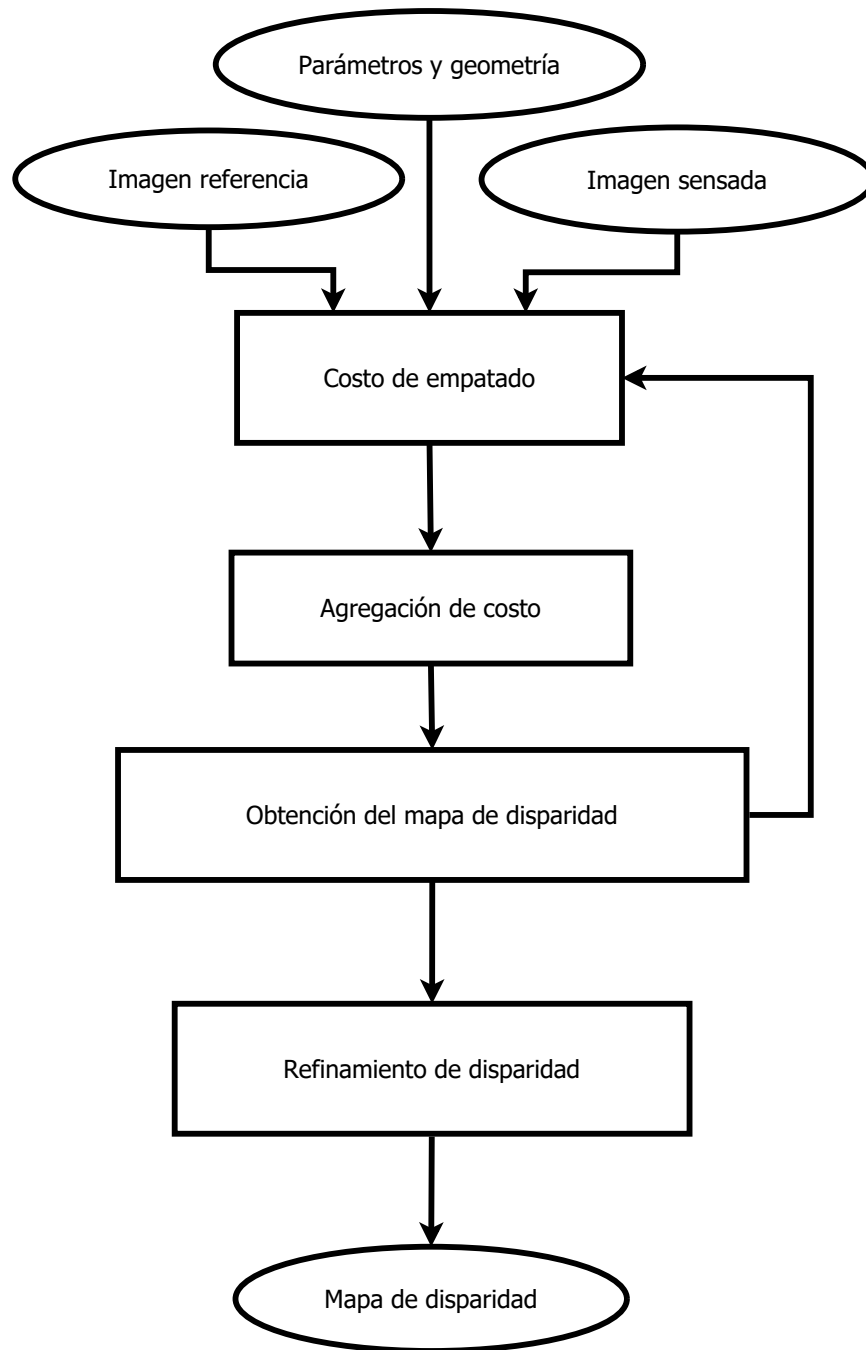


FIGURA A.1: Diagrama del funcionamiento del algoritmo *Semiglobal Matching*.

pixel p de la imagen de referencia I_r , denotado como I_{rp} , que debe ser empatao con un pixel q de la imagen sensada I_s , denotado a su vez por I_{sq} , de tal modo que $q = e_{rs}(p, d)$, donde d es la medida de disparidad que existe entre I_r e I_s . La función $e_{rs}(p, d)$ es la función dada por la línea base, de la geometría epipolar del sistema, proyectada en I_s e indica que en la línea base de I_s debe encontrarse el pixel p a una distancia d , respecto a la imagen de referencia.

El cálculo del costo de empatao se realiza con la información mutua la cual es robusta contra cambios de registro y de iluminación [18]. En teoría de la información, la información mutua está definida como la medida de comparación de información a partir de dos variables aleatorias y dada por sus probabilidades [5]. Para este caso, las imágenes estéreo fungirán como las variables aleatorias en cuestión.

Algunos autores mencionan que la información mutua fue inventada por Shannon en [39], y posteriormente, popularizada por Viola y Wells en el campo de Visión Computacional en [43]. Como se define en [18], la información mutua se expresa como:

$$MI_{I_r, I_s} = H_{I_r} + H_{I_s} - H_{I_r, I_s} \quad (\text{A.1})$$

Donde H_{I_r} y H_{I_s} son las entropías individuales de la imagen de referencia y de la imagen sensada, respectivamente; mientras que H_{I_r, I_s} es la entropía conjunta de las dos imágenes. Cabe mencionar que las imágenes están definidas como imágenes de intensidades.

Cuando la información de una imagen es independiente de la otra, se puede decir que la primera imagen no proporciona ninguna información de la segunda. Por el contrario, si las imágenes dependen entre sí, la información de una imagen puede determinar la de la otra [5].

La entropía de una imagen se obtiene calculando las distribuciones de probabilidad de las intensidades de los pixeles, por lo tanto, la entropía máxima se da cuando todas las intensidades están distribuidas uniformemente.

$$H_I = - \int_0^1 P_I(i) \log P_I(i) di \quad (\text{A.2})$$

La entropía conjunta se expresa como:

$$H_{I_r, I_s} = - \int_0^1 \int_0^1 P_{I_r, I_s}(i_r, i_s) \log P_{I_r, I_s}(i_r, i_s) di_r di_s \quad (\text{A.3})$$

De la ecuación A.2, se puede derivar la siguiente, utilizando el procedimiento de [24] de expansión de series de Taylor para expresar la entropía como una suma sobre

pixeles:

$$H_I = \sum_p h_I(I_p) \quad (\text{A.4})$$

De igual manera, se puede utilizar el mismo procedimiento para la entropía conjunta, quedando de la siguiente manera:

$$H_{I_r, I_s} = \sum_p h_{I_r, I_s}(I_{rp}, I_{sp}) \quad (\text{A.5})$$

De las ecuaciones anteriores, el término h está dado por:

$$h_I(i) = -\frac{1}{n} \log (P_I(i) \otimes g(i)) \otimes g(i) \quad (\text{A.6})$$

$$h_{I_r, I_s}(i, k) = -\frac{1}{n} \log (P_{I_r, I_s}(i, k) \otimes g(i, k)) \otimes g(i, k) \quad (\text{A.7})$$

En las ecuaciones A.6 y A.7, la variable n representa el número de pixeles que tienen correspondencia en I_r y en I_s , mientras que la distribución de probabilidad P_{I_r, I_s} se obtiene:

$$P_{I_r, I_s}(i, k) = \frac{1}{n} \sum_p T [(i, k) = (I_{rp}, I_{sp})] \quad (\text{A.8})$$

En esta última ecuación (A.8), T es un operador en cuyo argumento se realiza una operación booleana; cuando la intensidad del pixel ubicada en un arreglo bidimensional (i, k) corresponde a un pixel p que tiene correspondencia en las imágenes estéreo, entonces el resultado es 1, de lo contrario 0. Al final se obtiene la distribución de probabilidad de las intensidades dadas por las correspondencias en ambas imágenes. Al hacer que la distribución de probabilidad sólo se calcule para las correspondencias en las imágenes, se asegura que las oclusiones no se están considerando [18].

Para el caso de la ecuación A.6, la distribución de probabilidad P_I se puede obtener a partir de la ecuación A.8:

$$P_{I_r}(i) = \sum_k P_{I_r, I_s}(i, k) \quad (\text{A.9})$$

Por lo tanto, la información mutua se obtiene de:

$$MI_{I_r, I_s}(i, k) = \sum_p mi_{I_r, I_s}(I_{rp}, I_{sp}) \quad (\text{A.10})$$

De la que deriva el término mi definido como:

$$mi_{I_r, I_s}(i, k) = h_{I_r}(i) + h_{I_s}(k) - h_{I_r, I_s}(i, k) \quad (\text{A.11})$$

Y por último, el costo de empatao por información mutua queda de la siguiente manera:

$$C_{MI}(p, d) = -mi_{I_r, f_D(I_s)}(I_{rp}, I_{sq}) \quad (\text{A.12})$$

Todo el procedimiento para llegar a la ecuación A.12 se detalla en [18], inclusive algunas mejoras que se pueden hacer al algoritmo para optimizar la complejidad temporal del cálculo del costo de empatao.

Dado que en la ecuación A.12 f_D es una función sobre I_s que hace una deformación basándose en una imagen de disparidad D para empatarla con I_r , dicha disparidad debe ser conocida *a priori*. En [24] se menciona que una solución puede obtenerse iterativamente con una imagen de disparidad aleatoria para obtener C_{MI} , entonces ese mismo costo se puede utilizar para empatar las dos imágenes, I_r e I_s , y calcular una nueva imagen de disparidad, usando ésta última para la siguiente iteración.

b) Agregación de costo

De acuerdo con lo que menciona Hirschmüller, el costo de empatao basado en pixeles puede ser ambiguo cuando existe ruido, pues el costo de empatar dos pixeles que no corresponden podría ser menor a los que sí debido a la presencia de éste. Por lo tanto, él propone una agregación al costo que está basada en una restricción de suavidad, expresada en una función de energía que hace más robusta la búsqueda de pixeles que correspondan. Esta restricción se aplica como una función de energía expresa de la siguiente manera:

$$E(D) = - \sum_p \left(C(p, D_p) + \sum_{q \in N_p} P_1 T[|D_p - D_q| = 1] + \sum_{q \in N_p} P_2 T[|D_p - D_q| > 1] \right) \quad (\text{A.13})$$

Donde, en el primer término, C es la función de costo de un pixel p con una imagen de disparidad D , que se conoce *a priori*; el segundo término es la suma de los pixeles q que se encuentran en la vecindad N de p a una disparidad absoluta de 1 pixel, multiplicada por una constante de penalización P_1 ; y el tercer término es la misma suma de los pixeles q que se encuentran en la vecindad N_p pero que tienen un mayor cambio de disparidad, multiplicada por una penalización mayor P_2 . La penalización P_1 permite que los cambios pequeños en la disparidad se adapten a superficies inclinadas o curvas. La penalización P_2 permite conservar las discontinuidades. P_2 suele ser mayor que P_1 .

Ahora bien, el problema se trata de encontrar la imagen de disparidad D que minimice la función de energía $E(D)$. La minimización se realiza por renglón de imagen, lo que permite que la búsqueda se haga en 1D, y utilizando Programación Dinámica, lo cual se hace eficiente. Se realiza en 1D, por renglón de la imagen, debido a

que una minimización global en 2D es un problema NP-completo, según menciona Hirschmüller.

c) Obtención de la disparidad

En este paso, se obtiene la imagen de disparidad D_r correspondiente a la imagen de referencia I_r , seleccionando las disparidades de cada pixel p cuyo costo es mínimo, es decir, $\min_d S(p, d)$. En este mismo paso se puede calcular una imagen de disparidad D_s que corresponde a la imagen sensada I_s utilizando los mismos costos y, además, haciendo uso de la información de la línea epipolar que corresponde al pixel q de I_s . Ahora, se selecciona el valor d que corresponde al costo mínimo $\min_d S[e_{sr}(q, d), d]$, o bien, para obtener mejores resultados, se obtiene D_s realizando el procedimiento del costo de empatao y la agregación de nueva cuenta, ahora considerando a I_s como la imagen de referencia y a I_r como la imagen sensada. Se aplica un filtro mediana de 3×3 para eliminar *outliers* en D_r y D_s .

El cálculo de las dos disparidades sirve para determinar la oclusión y los falsos positivos que se puedan presentar. Se realiza una comparación de las disparidades en D_r a su correspondiente en D_s y se obtiene D_p , de tal manera que:

$$D_p = \begin{cases} D_{rp}, & \text{si } |D_{rp} - D_{sq}| \leq 1, \\ D_{inv}, & \text{en cualquier otro caso.} \end{cases} \quad (\text{A.14})$$

Es decir, siempre que exista un valor en D_p , éste será obtenido de la imagen de disparidad D_{rp} . La disparidad D_{inv} es más bien un valor inválido arbitrario que podría ser un valor fuera del rango de intensidades para diferenciarlo de éstos mismos.

d) Refinamiento de la disparidad

Éste es el último paso del algoritmo y en el cual se realiza un refinamiento para obtener una mejor disparidad de los valores obtenidos en el paso anterior. Se realiza debido a que la disparidad a este punto puede incluir errores, o bien, puede contener valores inválidos que pueden ser recuperados. El postprocesamiento a la imagen de disparidad D_p permite eliminar los errores y recuperar los datos perdidos. Diferentes métodos se presentan para obtener mejores resultados.

d.1) Remover picos

Los picos se presentan como *outliers* debido a reflejos, ruido, poca textura de la escena, etc. Para eliminarlos simplemente se utiliza una segmentación de vecindades de disparidades y, utilizando un umbral para el tamaño de la vecindad, se pueden definir qué vecindades son válidas. Estas vecindades de disparidades se obtienen considerando que los pixeles de una disparidad están *4-conectados*.

d.2) Selección de disparidad consistente en intensidad

Este procedimiento se utiliza para eliminar los problemas debidos a superficies sin textura. Se realiza una corrección en la función de energía $E(D)$ adaptando el costo de la penalización P_2 . Cuando en la escena se presenta una superficie sin textura, entonces el algoritmo revisa lo que existe alrededor de esa superficie, ya que otros objetos en la escena pueden hacer que la imagen tenga una textura más allá de la superficie que no la tiene. Si la superficie que no tiene textura sobrepasa los límites de la imagen, por ejemplo, paredes, pisos, etc., entonces debe considerarse la trayectoria por la que se encontró esta inconsistencia. Para solucionarlo, en principio, se asumen 3 cosas [18]:

- No existen discontinuidades dentro de áreas sin textura dentro en la imagen de disparidad.
- Existe alguna textura física visible, incluso en superficies sin textura.
- La superficie de un área sin textura puede aproximarse a un plano.

La identificación de las áreas sin textura se hace sobre la imagen de intensidades I_r .

d.3) Interpolación conservando discontinuidad

Este procedimiento se realiza con el fin de rellenar aquellos huecos que quedan después del procesamiento de la imagen de disparidad, incluso eliminando *outliers*. Los huecos que aparecen en la imagen de disparidad, como ya se mencionó, son debidos a oclusión o a errores de empatado. En la interpolación, estos dos casos deben tratarse por separado. Las oclusiones y los errores de empatado pueden distinguirse en el procedimiento de la obtención de disparidad, al hacer la relación de D_r con D_s , con ayuda de la geometría epipolar.

En OpenCV existen algunos métodos de empatado estéreo implementados. La clase *StereoSGBM* implementa el algoritmo *Semi-Global Block Matching* que está basado en el algoritmo de Hirschmüller. A diferencia del algoritmo SGM, el de OpenCV tiene algunas variantes de éste:

- Al ejecutar la Programación Dinámica en el algoritmo, el de OpenCV sólo considera 5 direcciones de búsqueda, en lugar de las 8 que considera el SGM.
- En OpenCV, el algoritmo empara bloques en lugar de hacerlo por pixeles.
- No se implementa la información mutua como función de costo, en su lugar se utiliza la métrica de subpixeles de Birchfield-Tomasi, descrita en [4].

- Algunos pasos de preprocesamiento, así como del postprocesamiento, son utilizados del algoritmo *Block Matching* propuesto por Kurt Konolige en [25], e igualmente implementado en las bibliotecas de OpenCV. Sobre todo, el prefiltrado y el postfiltrado.

La clase *StereoSGBM* en OpenCV contiene los suficientes atributos necesarios para calcular la imagen de disparidad. Cuando se crea un objeto de dicha clase, éste contendrá todos sus atributos. Por ejemplo, se puede manipular el tamaño del bloque utilizado por el algoritmo para empatar las imágenes estéreo y obtener la imagen de disparidad, incluso el tamaño puede ser 1 para realizar el empatao pixel por pixel; también puede decidirse si el algoritmo de empatao utiliza las 8 direcciones del algoritmo de Programación Dinámica; ambas consideraciones son útiles para ejecutar el algoritmo SGBM tal cual aparece en [18].

Apéndice B

Reproyección de una imagen en 2D a una nube de puntos en 3D.

En OpenCV existen métodos muy utilizados para lograr la reproyección de una imagen 2D a una imagen 3D, esto se utiliza para obtener una nube de puntos a partir de un mapa de disparidad, por ejemplo. Dependiendo las necesidades y los recursos con los que se cuenten, se implementan dos métodos que pueden ayudar a obtener dicha tarea. Para reproyectar un conjunto de puntos obtenidos de una cámara y que están en dos dimensiones a un espacio en tres dimensiones, deben conocerse sus coordenadas de imagen y sus parámetros intrínsecos [6]. Esta información se encuentra en la matriz de reproyección Q , que está definida como:

$$Q = \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 1 & f \\ 0 & 0 & -\frac{1}{T_x} & \frac{c_x - c'_x}{T_x} \end{bmatrix} \quad (\text{B.1})$$

El primero de los métodos es el de *projectiveTransform*. En este caso, la información de cada punto está dada como un vector con 3 componentes (x, y, d) , donde x y y es la posición del punto en las coordenadas de la imagen, mientras que el valor de d es la disparidad correspondiente a dicho punto (x, y) , en el mapa de disparidad. Esta información se opera con la matriz Q con el fin de obtener los puntos en 3D para representar un conjunto de puntos de la escena capturada por dos imágenes estéreo y a

la que se le ha calculado el mapa de disparidad.

$$Q \begin{bmatrix} x \\ y \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} \quad (\text{B.2})$$

Es así que las coordenadas en 3D de un punto son $(\frac{X}{W}, \frac{Y}{W}, \frac{Z}{W})$.

El segundo método implementado en OpenCV se llama *reprojectImageTo3D*. Básicamente, este método funciona igual que el anterior, salvo que los argumentos de entrada del método cambian, pero el cálculo es el mismo, al partir del mapa de disparidad y la matriz de reproyección Q . En este caso, es suficiente con el mapa de disparidad, a diferencia del primer método, que requiere como argumento el conjunto de puntos 2D, expresados en vectores de 3 componentes, a reproyectar en el espacio 3D. Sin embargo, el mapa de disparidad contiene la misma información, pues es una imagen con coordenadas (x, y) que tiene asociado un valor de intensidad d que representa la profundidad a la que se encuentra un objeto en la imagen de referencia, respecto a la imagen sensada.

Bibliografía

- [1] Barnard, Stephen T. y Martin A. Fischler: *Computational Stereo*. Computing Surveys, 14(4):553–572, Diciembre 1982.
- [2] Bay, Herbert, Tinne Tuytelaars y Luc Van Gool: *SURF: Speeded Up Robust Features*. En *Proceedings of the Ninth European Conference on Computer Vision*, páginas 404–417, 2006.
- [3] Benschraier, A., P. Miché y R. Debric: *Fast and automatic stereo vision matching algorithm based on dynamic programming method*. Pattern Recognition Letters, 17(5):457–466, 1996.
- [4] Birchfield, Stan y Carlo Tomasi: *A Pixel Dissimilarity Measure That is Insensitive to Image Sampling*. IEEE. Transactions on Pattern Analysis And Machine Intelligence, 20(4):401–406, Abril 1998.
- [5] Bose, Ranjan: *Information Theory, Coding and Cryptography*. Tata McGraw-Hill, 2008.
- [6] Bradski, Gary y Adrian Kaehler: *Learning OpenCV*. O’Reilly Media Inc., 2008.
- [7] Brown, Myron Z., Darius Burschka y Gregory D. Hager: *Advances in Computational Stereo*. IEEE. Transactions on Pattern Analysis And Machine Intelligence, 25(8):993–1008, Agosto 2003.
- [8] Brunelli, Roberto: *Template Matching Techniques in Computer Vision*. John Wiley & Sons, 2009.
- [9] Canny, John: *A Computational Approach to Edge Detection*. IEEE. Transactions on Pattern Analysis And Machine Intelligence, PAMI-8(6):679–698, 1986.
- [10] Davies, E. R.: *Machine Vision: Theory, Algorithms, Practicalities*. Academic Press, Inc., EE.UU., 1990.

-
- [11] Dhond, Umesh R. y J. K. Aggarwal: *Structure from Stereo - A Review*. IEEE Transactions on Systems, Man and Cybernetics, 19(6):1489–1510, Noviembre/Diciembre 1989.
- [12] Eghbali, Hassan J.: *K-S Test for Detecting Changes from Landsat Imagery Data*. IEEE Transactions on Systems, Man and Cybernetics, 9(1):17–23, Enero 1979.
- [13] Geiger, Andreas, Martin Roser y Raquel Urtasun: *Efficient Large-Scale Stereo Matching*. En Kimmel, Ron, Reinhard Klette y Akihiro Sugimoto (editores): *Computer Vision – ACCV 2010*, volumen 6492 de *Lecture Notes in Computer Science*, páginas 25–38. Springer Berlin Heidelberg, 2011, ISBN 978-3-642-19314-9. http://dx.doi.org/10.1007/978-3-642-19315-6_3.
- [14] González, R. C. y R. E. Woods: *Digital Image Processing*. Prentice Hall, 2008.
- [15] Goshtasby, A. Ardeshir: *2-D and 3-D Image Registration for medical remote sensing and industrial applications*. Wiley - Interscience, New Jersey, 2005.
- [16] Grimson, W. E. L.: *A Computer Implementation of a Theory of Human Stereo Vision*. Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences, 292(1058):217–253, Mayo 1981.
- [17] Harris, Chris y Mike Stephens: *A Combined Corner and Edge Detector*. En *Proceedings of Fourth Alvey Vision Conference*, páginas 147–151, 1988.
- [18] Hirschmüller, Heiko: *Stereo Processing by Semiglobal Matching and Mutual Information*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(2):328–341, Febrero 2008.
- [19] Hirschmüller, Heiko y Daniel Scharstein: *Evaluation of Cost Functions for Stereo Matching*. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Junio 2007.
- [20] Hu, Ming Kuei: *Visual Pattern Recognition by Moment Invariants*. IRE Transactions on Information Theory, páginas 179–187, 1962.
- [21] Huttenlocher, Daniel P., Gregory A. Klanderman y William J. Rucklidge: *Comparing Images Using the Hausdorff Distance*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(9):850–863, Septiembre 1993.
- [22] Julesz, Bela: *Binocular Depth Perception of Computer-Generated Patterns*. Bell System Technical Journal, 39(5).
- [23] Kanade, T., H. Kano, S. Kimura, A. Yoshida y K. Oda: *Development of a video-rate stereo machine*. En *IEEE/RJSJ International Conference on Intelligent Robots*

- and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings.,* volumen 3, páginas 95–100, Agosto 1995.
- [24] Kim, Junhwan, Vladimir Kolmogorov y Ramin Zabih: *Visual Correspondence Using Energy Minimization and Mutual Information*. En *Proceedings of the Ninth IEEE International Conference on Computer Vision*, 2003.
- [25] Konolige, Kurt: *Small Vision Systems: Hardware and Implementation*. En *Proceedings of the Eighth International Conference on Robotics Research*, páginas 203–212, 1997.
- [26] Kosov, Sergey, Thorsten Thormählen y Hans Peter Seidel. En *Advances in Visual Computing*, volumen 5875 de *Lecture Notes in Computer Science*, páginas 796–807. Springer Berlin Heidelberg, 2009.
- [27] Lowe, David G.: *Distinctive Image Features from Scale-Invariant Keypoints*. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [28] Marr, D. y T. Poggio: *Cooperative Computation of Stereo Disparity*. *Science*, 194(4262):283–287, 1960.
- [29] Mittal, R. K. y I. J. Nagrath: *Robotics and Control*. Tata McGraw-Hill, 2003.
- [30] Moré, Jorge J.: *The Levenberg-Marquardt algorithm: Implementation and theory*, volumen 630 de *Lecture Notes in Mathematics*, páginas 105–116. Springer Berlin Heidelberg.
- [31] Mouragnon, E., M. Lhuillier, M. Dhome, F. Dekeyser y P. Sayd: *Real Time Localization and 3D Reconstruction*. En *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.
- [32] Nevatia, Ramakant: *Depth Measurement by Motion Stereo*. *Computer Graphics and Image Processing*, (5):203–214, 1976.
- [33] Nistér, David, Oleg Naroditsky y James Bergen: *Visual Odometry*. En *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.
- [34] Pajares, Gonzalo y Jesús De la Cruz: *Visión por computador. Imágenes Digitales y Aplicaciones*. Alfaomega, México, 2008.
- [35] Pearl, Judea: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., EE.UU., 1988.
- [36] Poole, Harry H.: *Fundamentals of Robotics Engineering*. Van Nostrand Reinhold, 1989.

- [37] Proakis, John G. y Dimitris G. Manolakis: *Digital Image Processing. Principles, Algorithms, and Applications*. Prentice-Hall, Inc., EE.UU., 1996.
- [38] Scharstein, Daniel y Richard Szeliski: *A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms*. International Journal of Computer Vision, páginas 7–42, Abril-Junio 2002.
- [39] Shannon, Claude E.: *A mathematical theory of communication*. Bell Systems Technical Journal, (27):379–423, 1948.
- [40] Snyder, John M.: *Generative Modeling for Computer Graphics and CAD: Symbolic Shape Design Using Interval Analysis*. Academic Press, Inc., 1992.
- [41] Sun, Jian, Heung Yeung Shum y Zheng Nan-Ning: *Stereo Matching Using Belief Propagation*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 25:787–800, Julio 2003.
- [42] Tsai, Roger Y.: *A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses*. IEEE Journal of Robotics and Automation, RA-3(4):323–344, Agosto 1987.
- [43] Viola, Paul y William M. Wells III: *Alignment by Maximization of Mutual Information*. International Journal of Computer Vision, páginas 137–154, 1997.
- [44] Wöhler, Christian: *3D Computer Vision. Efficient Methods and Applications*. Springer-Verlag, Berlín, Alemania, 2009.
- [45] Woodfill, John Iselin, Gaile Gordon, Dave Jurasek, Terrance Brown y Ron Buck: *The Tyzx DeepSea G2 Vision System, A Taskable, Embedded Stereo Camera*. En *Proceedings of the IEEE Computer Society Workshop on Embedded Computer Vision. Conference on Computer Vision and Pattern Recognition*, páginas 1–7, 2006.
- [46] Xie, Min: *Fundamentals of Robotics: Linking Perception to Action*. World Scientific, 2003.
- [47] Xu, Gang y Zhang Zhengyou: *Epipolar Geometry in Stereo, Motion and Object Recognition. A Unified Approach*. Kluwer Academic Publishers, Dordrecht, Holanda, 1996.
- [48] Zhang, Zhengyou: *A Flexible New Technique for Camera Calibration*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(11):1330–1334, 2000.