Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional.

Unidad Distrito Federal

Departamento de Control Automático

Reconstrucción 3D para Humanoides

Tesis que presenta

Jorge Enrique Lavín Delgado

Para Obtener el Grado de

Doctor en Ciencias

En la Especialidad de

Control Automático

Director de la Tesis: Dr. Juan Manuel Ibarra Zannatha

México, D.F. Mayo 2015

Contenido

Lista de Figuras

A la memoria de mi querida abuelita
Faustina Carreón Uribe.

Agradecimientos

En primer lugar quiero agradecer a Dios por todas sus bendiciones a lo largo de mi vida y por permitirme llegar hasta este inolvidable día.

Gracias a mis padres Ignacia y Rutilo porque este logro es más suyo que mío, por ser mi apoyo y fortaleza en todo momento, por ser para mí ejemplo de fe y amor. Muchas gracias por todo lo que hacen por mí, saben que los amo y siempre están en mi corazón.

También quiero agradecer a mis hermanos Hugo y Sergio porque me han dado lecciones de vida y me han dado su cariño. Sé que siempre estoy en sus oraciones y ustedes están en las mías.

De manera muy especial quiero agradecer a mi asesor, el Dr. Juan Manuel Ibarra Zannatha, por el apoyo incondicional en la realización de este trabajo, por sus enseñanzas y su amistad, pero sobre todo por ser un ejemplo a seguir.

Mis más sinceros agradecimientos a mis revisores de tesis: Dr. Jorge Antonio Torres Muñoz, Dr. Alejandro J. Malo Tamayo, Dra. Zizilia Zamudio Beltrán y Dr. Eduardo Campos Mercado por sus valiosos comentarios que ayudaron a mejorar este trabajo.

Gracias a mis abuelitas Ubaldina y Estela, a mis tíos Marcelino, Ismael, Gabriel, Alfonso, Jesús, Alicia, Beatriz, Jaime, Arminda, Elvia y Maricela por su apoyo y comprensión, pero sobre todo por su cariño y por las palabras de aliento que en algún momento me ayudaron a retomar el camino. Sin ustedes no lo hubiera logrado.

Muchas gracias a mis compañeros de generación y amigos Zizilia, Eric, Ariel, Josué, Rafael, Miguel, Irving, Víctor, Alejandro y Ángel por todos los momentos que pasamos juntos y por ser un ejemplo de perseverancia y esfuerzo constante, en verdad son especiales para mí.

Quiero agradecer a los maestros Gabriel Pedroza Silvar, Oscar Juárez González y Martha Macedo Millán por su apoyo durante los últimos dos años que estuve realizando este trabajo. Siempre les estaré agradecido.

De manera muy especial quiero agradecer al Consejo Nacional de Ciencia y Tecnología (CONACyT) por su apoyo y financiamiento que hicieron posible la realización de mis estudios de posgrado.

Resumen

El problema de la visión estereoscópica ha recibido una atención considerable en los últimos años debido a sus múltiples aplicaciones. Este problema puede enunciarse de forma breve del siguiente modo: dadas dos imágenes del mundo real tomadas desde puntos de vista diferentes, el problema consiste en recuperar la geometría de la escena. Considerando que el sistema de visión estereoscópico está calibrado, esto es, que se conocen los parámetros intrínsecos y extrínsecos de ambas cámaras, el problema consiste esencialmente en definir pares de puntos, uno en cada imagen, tales que sean la proyección del mismo punto en la escena. Este se conoce como problema de correspondencia. Una vez que se ha identificado un par de puntos correspondientes, es posible reconstruir el punto 3D a partir de la intersección de los rayos ópticos correspondientes. El objetivo principal de esta tesis es implementar algoritmos de reconstrucción 3D en diferentes aplicaciones de robótica.

La primera tarea consiste en proponer un algoritmo de reconstrucción que genere un modelo tridimensional virtual a partir de un conjunto de imágenes capturadas con un sistema de visión estereoscópica. De este modo se pretende sustituir las técnicas tradicionales como la telemetría láser o la luz estructurada, las cuales permiten obtener modelos muy exactos y precisos, pero con el inconveniente de utilizar equipos costosos. La segunda tarea consiste en desarrollar un sistema de seguimiento o tracking que permita estimar la pose de las manos de un usuario humano, información que será utilizada para actualizar los parámetros de pose y movimiento de las manos y dedos del modelo tridimensional dentro de un mundo virtual. El enfoque de la solución propuesta consiste en utilizar una cámara instalada que detecta el movimiento de ciertas marcas ubicadas en la mano del usuario para después calcular la matriz de homografía. El sistema de seguimiento que se propone es de bajo costo y confiable. La última tarea es desarrollar una plataforma para fines de rehabilitación de personas con parálisis cerebral o de pacientes apopléjicos, compuesto por un robot humanoide Nao, un sensor Kinect y una PC. El sistema debe ser capaz de detectar los movimientos de un ser humano y reproducirlos en el humanoide antes mencionado. La técnica desarrollada consiste en capturar las coordenadas 3D de ciertos puntos utilizando el sensor Kinect y mediante un solución propuesta, resolver el problema cinemático inverso para calcular las variables articulares que generaron una cierta posición. Esta variables se convierten en consignas que son enviadas al robot humanoide para que imite o reproduzca el movimiento del usuario. De este modo el robot aprenderá ciertos movimientos que después el pueda enseñar a usuarios humanos discapacitados para recuperar parcial o totalmente la movilidad de los miembros superiores.

Abstract

The stereoscopic vision problem has received a considerable attention in the last years due to its wide domain of application. This problem can be shortly explained as follows: Given two images of the real world (3D scene) taken from two different points of view, the problem consists on recovering the geometry of the scene. Considering that the stereoscopic system of vision is calibrated, i. e., intrinsic and extrinsic parameters of both cameras are known, in addition distortions in both images have been previously corrected, the problem consists essentially of defining pairs (or sets) of points, one in each image, so they are projections of the same point in the scene. This is known as the correspondence problem. Once a pair of matching points has been identified, the 3D point can be reconstructed by intersecting the corresponding optical rays. The main objective of this thesis is to implement 3D reconstruction algorithms for different kind of robotic applications.

The first task is to propose a reconstruction algorithm that generates a virtual three-dimensional model from a set of images captured with a system of stereoscopic vision. Thus, it is intended to replace the traditional techniques such as laser telemetry or structured light, which allow to obtain very precise and accurate models , but with the disadvantage of using expensive equipment. The second task is to develop a tracking system to estimate the pose of the hands of a human user , information that will be used to update the parameters of pose and movement of the hands and fingers dimensional model within a virtual world. The approach of the proposed solution is to use an installed camera that detects the movement of certain brands located in the user's hand and then calculate the homography matrix . The tracking system proposed is inexpensive and reliable. The last task is to develop a platform for rehabilitation of people with cerebral palsy or stroke patients , consisting of a humanoid robot Nao , a Kinect sensor and a PC. The system must be able to detect the movements of a human being and then be imitating the humanoid robot. The developed technique is to capture the 3D coordinates of certain points using the Kinect sensor and through a proposed solution , solving the inverse kinematics problem for calculating joint variables that generated a certain position . This variables become slogans that are sent to the humanoid robot to reproduce the movement of the user. With this aim the robot to learn certain movements which then can teach human users with some disabilities to reach partially or completely the mobility of the upper limbs (arms).

Capítulo 1

Introducción

Los seres vivos tenemos la capacidad de interactuar con el entorno que nos rodea, lo cual nos permite satisfacer nuestras necesidades como alimento, protección, aprendizaje, etc. Sin embargo, para que el proceso de interacción se lleve a cabo se requiere de dos etapas, la primera consiste en la obtención de información del exterior (capacidad sensorial) y la segunda en la manipulación de la misma (procesamiento) [10]. Los seres humanos obtenemos información del medio ambiente a través de los sentidos que poseemos: vista, olfato, oido, tacto y gusto. Sin embargo de los cinco sentidos, probablemente del que más información obtenemos de manera consciente es la vista. En efecto, la extracción de información tridimensional a partir de imágenes bidimensionales es una tarea que el ser humano puede realizar con mucha facilidad, pues desde temprana edad tiene la capacidad de observar e identificar los objetos presentes en el entorno que le rodea, lo que nos permite interactuar con el medio ambiente y desenvolvernos en él con un alto grado de eficiencia. Gracias al sistema de visión que poseemos es posible realizar desde actividades básicas diarias que implican la búsqueda y reconocimiento de lugares, la identificación de personas, la manipulación de objetos, hasta actividades intelectuales de entretenimiento como mirar una puesta de sol, leer un libro o una revista, etc. Entre otras características la información visual nos permite recuperar color, textura, forma y ubicación de los objetos sin necesidad de llegar a un contacto físico [23].

Por otro lado, en el mundo donde vivimos e interactuamos los acontecimientos ocurren en tres dimensiones (horizontal, vertical y profundidad), sin embargo la profundidad se pierde al capturar una imagen de la escena, es decir, una sola imagen no proporciona suficiente información a cerca de la profundidad a la que se encuentran los objetos que la conforman [91]. A raíz de esto, se han desarrollado diversas técnicas capaces de recuperar la profundidad utilizando una sola imagen, que funcionan sólo en

ciertos casos especiales cuando se tiene un conocimiento a priori de la escena observada, otras requieren una configuración especial de la escena y/o de la cámara utilizada. Con al menos dos imágenes tomadas desde diferentes puntos de vista es posible recuperar la profundidad mediante un proceso de triangulación. Esta es la razón por la que los humanos son capaces de recuperar la información de profundidad, es decir, su sistema de visión consta de dos ojos con sus ejes paralelos entre sí y separados una cierta distancia, con esta configuración se generan dos imágenes con una pequeña diferencia entre ellas conocida como disparidad binocular. La disparidad permite realizar un proceso de triangulación mediante el cual es posible calcular la profundidad de la escena observada.

Dada la importancia del sistema de visión que poseemos, y considerando las ventajas y las potenciales actividades que se pueden llevar a cabo con la información obtenida de él, resulta interesante plantear la posibilidad de proveer a una máquina, (probablemente un robot) de dicho sentido utilizando cámaras digitales y una computadora para procesar las imágenes capturadas, lo que aunado a otros mecanismos tales como razonamiento y aprendizaje hagan de este una herramienta capaz de interactuar eficientemente con el medio ambiente dinámico en el que se desenvuelve [86]. Debido al deseo de lograr que una máquina sea capaz de ver como lo hacemos los humanos, se han desarrollado importantes investigaciones. Dentro de estas investigaciones, ha tenido una particular importancia el proceso de formación de una imágen, sus propiedades geométricas y la recuperación de la información tridimensional. A este proceso de extraer o recuperar la información tridimensional se le denomina Reconstrucción 3D y de manera formal puede enunciarse como el proceso mediante el cual se calcula la información tridimensional de los puntos que componen la escena a partir de conocer su proyección que tienen dentro de una imagen.

El estudio de las proyecciones bidimensionales de objetos tridimensionales se ha tratado desde hace tiempo en geometría proyectiva y descriptiva, en donde se ha hecho énfasis en la descripción de objetos en un plano bidimensional. El problema contrario de cómo reconstruir la estructura de un objeto 3D mediante su proyección 2D en un plano empezó a llamar la atención a finales de los años 60 con el desarrollo de las computadoras digitales [40]. En efecto, la extracción de información tridimensional a partir de una o varias imágenes es el objetivo principal en Fotogrametría. Aunque en algunos casos especiales es posible utilizar una sola imagen generalmente en Fotogrametría se utilizan dos o más imágenes a modo de emplear técnicas estereoscópicas con las cuales se pueda recuperar la información tridimensional de la escena [98].

A principios del siglo XX se inventaron los primeros dispositivos para realizar

mediciones estereoscópicas y que posteriormente se utilizaron como graficadores analógicos [101]. En 1957 se construyó un graficador analítico que tenía la capacidad de establecer una relación geométrica entre imágenes estereoscópicas. Aproximadamente en el año 1990 se introdujó la primera estación de trabajo fotogramétrica digital en donde se almacenaban imágenes digitales y se realizaba la extracción de información métrica mediante una computadora. En esa misma época se desarrollaron diversas técnicas para extraer información de las imágenes con el propósito de obtener la reconstrucción 3D de la escena de forma automática [84].

El proceso de Reconstrucción 3D se puede dividir en tres pasos:

1. **Procesamiendo digital de imágenes.** Este primer paso involucra la adquisición de las imágenes, el filtrado de ellas para mejorar sus condiciones, la detección de características y la búsqueda de correspondencias.

2. **Reconstrucción 3D.** En esta etapa se realiza la calibración del sistema de visión utilizado y posteriormente con el conjunto de correspondencias obtenido en el paso anterior, se aplica algún algoritmo de reconstrucción (disparidad, triangulación, detección de pose, cosenos directores, etc).

3. **Visualización de la Reconstrucción.** Este último paso involucra la generación de un modelo tridimensional de la escena a partir de los puntos obtenidos en la etapa de reconstrucción, es decir, la nube de puntos se convierte en mallas de polígonos o en mallas triangulares irregulares obteniéndose un modelo tridimensional de la escena observada.

1.1. Motivación

En los ultimos años, la Visión Artificial se ha enfocado en el estudio de técnicas de reconstrucción de objetos y algoritmos capaces de extraer información útil de las imágenes [26]. Siendo éstas últimas las que mayor desarrollo han tenido debido a las múltiples aplicaciones en diversas áreas tales como robótica, medicina, procesos de manufactura, arqueología, entre otras, en las cuales se requiere obtener un modelo tridimensional de la escena observada. Una de las aplicaciones de la Reconstrucción 3D se da en el área de la ingeniería biomédica, donde se realiza la reconstrucción de estructuras anatómicas a partir de imágenes médicas como resonancias magnéticas, lo cual representa un herramienta importante en el diagnostico médico, en la planificación de terapias y en procedimientos quirúrgicos [56]. Otra aplicación consiste en la reconstrucción de ciudades, edificios históricos o museos para visitas virtuales, permitiendo al usuario visualizar el lugar con la sensación de estar allí. La reconstrucción

3D también se utiliza en el ámbito industrial en la generación de modelos CAD para después fabricar prototipos [84].

En la industria, la medicina y la robótica también se han logrado diversas aplicaciones con los avances obtenidos en la Reconstrucción 3D. En la industria la tarea se ha enfocado en ubicar una cámara en una línea de producción, de tal forma que se realiza una inspección visual a los productos en búsqueda de partes faltantes, defectos de fabricación, ensambles, etc. Este tipo de actividades anteriormente podían llevarse a cabo sólo por seres humanos, pero debido a lo repetitivo de la tarea eran propensas a fallas. En medicina las aplicaciones pueden apreciarse en actividades como cirugía asistida por computadora, obtención de imágenes con información tridimensional del interior del cuerpo, rehabilitación de ciertas enfermedades [59]. En robótica se desea dotar a las máquinas de la capacidad de interactuar con el medio ambiente tal como los robots de servicio, futbolistas, exploradores, etc; de manera que se requiere una mayor comprensión de la estructura tridimensional de la escena para lograr resultados satisfactorios, por lo que se ha hecho enfásis en la generación de mapas para dichos robots, en donde tradicionalmente se utilizan sensores ultrasónicos tipo sonar o láser para tal fin, pero debido a la poca fiabilidad en las lecturas y el poco alcance su rango de aplicación su implementación se reduce a espacios cerrados. Estos inconvenientes han sido eliminados utilizando sistemas de visión, los cuales permiten realizar la reconstrucción del entorno y de este modo obtener un mapa del mismo [33]. Por otra parte, los sistemas de visión son los elementos sensoriales más utilizados en la actualidad para incrementar la autonomía de un robot, los cuales proporcionan información importante acerca del entorno del robot. Esto ha propiciado el control de robots, mejor conocido como visual servoing, basado en información visual en el cual el sistema de visión provee las entradas de referencia al controlador de las articulaciones del robot [12].

Con lo anterior se puede decir que la Reconstrucción 3D sigue siendo un campo activo de investigación debido a las diversas aplicaciones en diferentes áreas en las que puede implementarse. Apesar de los avances que hoy en día se tienen con los cuales ya es posible recuperar la información tridimensional de una escena, logrando así una mejor compresión de la misma, es necesario mejorar estos resultados y procedimientos para poder dotar verdaderamente a una máquina de un sistema de visión similar al humano. Este es precisamente el motivo por el cual la reconstrucción 3D es el problema principal que se aborda en este trabajo de tesis.

1.2. Plantamiento de los problemas

El tema principal de este trabajo de tesis consiste en la proposición de una solución a diversas aplicaciones de robótica utilizando técnicas basadas en visión artificial, particularmente la reconstrucción 3D. La reconstrucción 3D se define como el proceso de obtener las coordenadas tridimensionales de las escena a partir de la información contenida en una o varias imágenes. De este modo, el presente trabajo persigue tres objetivos principales:

1. Proponer un algoritmo mediante el cual a partir de un conjunto de imágenes capturadas con un sistema de visión estereoscópico se obtenga un modelo tridimensional de la escena. Con esto se pretende sustituir las técnicas tradicionales como la telemetría láser o la luz estructurada que permiten reproducir modelos muy exactos y precisos, pero con el inconveniente de emplear un equipo costoso. Otras técnicas como la visión estereoscópica densa tienen tiempos de ejecución muy altos, por lo cual se propone una reconstrucción estereoscópica dispersa basada en puntos de interés o característicos, la cual representa una solución robusta y rápida en comparación con el resto de las técnicas existentes [27]. En resumen, un primer problema consiste en generar una aproximación o modelo 3D de un entorno partiendo de un conjunto de imágenes estereoscópicas.

2. Desarrollar un sistema de seguimiento o tracking que permita estimar la pose de las manos de un usuario humano, información que será utilizada para actualizar los parámetros de pose y movimiento de las manos y dedos del modelo tridimensional dentro de un mundo virtual. El enfoque de la solución propuesta consiste en utilizar una cámara instalada que detecta el movimiento de ciertas marcas ubicadas en la mano del usuario. El sistema de seguimiento que se propone es de bajo costo y confiable.

3. Desarrollar una plataforma para fines de rehabilitación de personas con parálisis cerebral o de pacientes apopléjicos, compuesto por un robot humanoide Nao, un sensor Kinect y una PC. El sistema debe ser capaz de detectar los movimientos de un ser humano y reproducirlos en el humanoide antes mencionado. La técnica desarrollada consiste en capturar las coordenadas 3D de ciertos puntos utilizando el sensor Kinect y mediante un solución propuesta, resolver el problema cinemático inverso para calcular las variables articulares que generaron una cierta posición. Esta variables se convierten en consignas que son enviadas al robot humanoide para que imite o reproduzca el movimiento del usuario. Con esto se pretende que el robot aprenda ciertos movimientos que después el pue-

da enseñar a usuarios humanos con una cierta discapacidad para que alcancen parcial o totalmente la movilidad de los miembros superiores (brazos).

Capítulo 2

Estado del arte

A pesar de toda la información contenida en una imagen, la profundidad de la escena no se puede calcular directamente considerando una sola imagen, salvo algunos casos especiales en donde se cuenta con información extra [23]. Con al menos dos imágenes la profundidad se puede obtener mediante un proceso de triangulación. De esta manera, los algoritmos de reconstrucción 3D se clasifican en dos grandes grupos: monoculares y estereoscópicos. Los primeros consideran una sola imagen para extraer la información tridimensional de la escena, mientras que los segundos hacen referencia al uso de más de una imagen para lograr el mismo fin [24].

2.1. Métodos monoculares

En casos particulares en los cuales se tiene información a priori sobre los objetos observados , o cuando la geometría de dichos objetos es tal que la proyección en perspectiva genera una imagen que conserva información sobre la profundidad, pueden implementarse algoritmos de reconstrucción 3D sobre una sola imagen.

En [99] se describe una técnica monocular que tiene como idea principal utilizar los cambios de iluminación para obtener los vectores normales a la superficie de un objeto 3D en la escena, para lo cual se requieren luces puntuales de intensidad conocida y regulable. Se considera que la superficie del objeto a reconstruir es Lambertiana, es decir, la luminosidad reflejada por el objeto es uniforme. Sin embargo en la práctica no existen objetos con tal reflectividad, de manera que la precisión de esta técnica está limitada por el tipo de objeto considerado. A pesar de que esta técnica de reconstrucción considera el uso de una cámara, también puede suponerse el uso de dos cámaras.

Shape from focus es un método monocular basado en la textura de la superficie

del objeto observado para calcular su profundidad [72]. Se captura una secuencia de imágenes del objeto utilizando diferentes niveles de enfoque mediante un operador de enfoque denominado *sum-modified-Laplacian*, con el cual se calcula una medida local de la calidad de enfoque en la imagen. Este operador se aplica a la secuencia de imágenes obteniéndose un sistema de medidas de enfoque para cada uno de los píxeles en las imágenes. Después se aplica un algoritmo de estimación de la profundidad basado en un modelo que describe la variación de la medida de enfoque, en donde se interpola un pequeño numero de valores de enfoque, obteniéndose una estimación precisa de la profundidad.

Texture Gradient es un algoritmo muy similar a Shape from Focus, el cual estima la profundidad utilizando una medida de la variación (gradiente) de la textura del objeto a reconstruir [5]. Dicha variación se basa en alguna de las propiedades de la textura tales como la homogeneidad, isotropía, etc.; comparando su valor original con el valor observado. También se considera que la propiedad observada en la textura es uniforme y que la iluminación no afecta su valor, esto es, las superficies de los objetos son Lambertianas. La variación de la textura aumenta conforme el objeto se aleja, sin embargo, esta variación también puede ser causada por la iluminación de la escena, rugosidad de las superficies y oclusiones, por lo cual en muchos casos esta técnica es poco fiable. En resumen, esta técnica consta de tres etapas: seleccionar alguna propiedad de la textura y estimar su estado inicial, calcular el gradiente o variación de esta propiedad, finalmente determinar la información tridimensional de la escena a partir de este gradiente.

Active Range-finding es un conjunto de técnicas cuya idea principal consiste en cambiar las condiciones de la escena de algún modo, para después medir la reacción del objeto a reconstruir. En [85] se describe un método denominado *Striped Lighting* que consiste en formar un plano de iluminación en la escena y moverlo de modo que recorra el objeto. Esto se puede lograr haciendo pasar un rayo de luz (o láser) en la superficie del objeto, generando una sombra sobre el mismo. Una vez que se percibe el plano, se triangulan los puntos de intersección de éste con el objeto observado. En [96] se muestra una técnica aplicada en áreas planas de poliedros solidos que utiliza un filtro lineal en el dominio de la frecuencia y la transformada de Fourier para estimar la posición y orientación de los planos presentes en la imagen.

La *rotación de Kanatani* es un técnica monocular cuyo enfoque considera una escena formada por un cubo cuya imagen muestra los planos del cubo distorsionados de manera que se observan 3 puntos de fuga hacia los cuales convergen las rectas paralelas de cada uno de los tres planos visibles del cubo [54]. Uniendo dos a dos

dichos puntos de fuga se obtienen tres rectas denominadas rectas de fuga. Ahora bien, Kanatani propone un método de exploración geométrica sobre cada una de las rectas de fuga para encontrar el punto en el cual una rotación, denominada de *Kanatani*, permite llevar ese punto al punto principal de la imagen con lo cual el plano asociado a la recta de fuga en cuestión quedará sin distorsión alguna [48],[47]. Los parámetros de la transformación de rotación obtenida contiene la información tridimensional buscada.

La detección de pose es un método aplicado a una sola imagen que considera el conocimiento a priori de la geometría del objeto observado y además que se tiene un modelo CAD del mismo. La idea principal consiste en ir proponiendo una ubicación y orientación del modelo CAD, proyectarlo virtualmente sobre el plano imagen y comparar esta proyección con la imagen real. El error entre la proyección virtual y la imagen se utiliza en un proceso de optimización en la que va cambiando iterativamente la ubicación del modelo CAD hasta que el error se hace tan pequeño como se desee, en cuyo caso se ha calculado la posición y orientación del objeto [21].

En [45] se describe una técnica denominada *Cosenos Directores*, la cual establece, para cada punto de la escena, un conjunto de tres ecuaciones en donde haya tres incógnitas (las coordenadas de dicho punto) en función de las mediciones disponibles y alguna otra información conocida a priori. Por ejemplo, dado un punto en la imagen, existe un vector que va del origen del referencial asociado a la cámara hasta dicho punto, cuyas coordenadas son tres: las dos coordenadas en la imagen más la distancia focal; mientras que el punto en la escena está representado por un vector en ese mismo referencial cuyas coordenadas son sus tres coordenadas cartesianas. Estos dos vectores son colineales, por lo que los cosenos directores del primero son iguales a los del segundo. Así, igualando las expresiones de sus cosenos directores se obtienen tres ecuaciones con tres incógnitas, con lo cual se construye un método de reconstrucción 3D monocular que requiere, como información a priori, la distancia focal de la cámara
.

En casos similares al anterior (escena plana) se puede dar el caso de tener objetos de forma y tamaño conocidos, en cuyo caso es posible parametrizar su profundidad en términos de la distorsión de algunas de sus dimensiones conocidas. En [23] se muestra un algoritmo de reconstrucción que considera un objeto cuadrado ubicado sobre una pared del entorno y que la imagen producida por la cámara está montada sobre un robot móvil cuyo eje óptico es perfectamente horizontal (perpendicular a la pared mencionada) y se encuentra a la misma altura que el centro del objeto cuadrado. En este caso, la imagen producida se encuentra distorsionada por la posición y orientación del objeto con respecto a la cámara, de modo que las rectas horizontales del cuadrado

se distorsionan dejando de ser paralelas para intersecarse en un punto de fuga, mientras que las rectas verticales si mantienen su paralelismo entre ellas. La diferencia entre la altura de estas dos rectas permite recuperar la orientación del cuadrado, mientras que sus respectivas escalas serán proporcionales a su profundidad.

El trabajo descrito en [83] difiere de las técnicas tradicionales en la forma de estimar la profundidad pues introduce el area de reconocimiento de patrones con el fin de modelar el problema de estimación de la profundidad como un problema de aprendizaje. La idea principal radica en que dada una imagen, esta tiene asociado un mapa de profundidad, la cual se puede extraer o aproximar mediante un mecanismo de aprendizaje. Al plantear el problema de esta forma resulta necesario obtener un conjunto de características que describan de manera significativa la relación de la imagen con su mapa de profundidad, por lo cual se utiliza un algoritmo de detección que considere las diferentes escalas, para así obtener características locales y globales de la imagen. Para obtener el mapa de disparidad se emplea un mecanismo de aprendizaje basado en el campo aleatorio de Markov, es decir, un clasificador de patrones probabilístico.

2.2. Métodos estereoscópicos

Con al menos dos imágenes capturadas desde diferentes puntos de vista es posible recuperar la información de profundidad de los objetos dentro de la escena mediante un proceso de triangulación de los puntos correspondientes en las diferentes imágenes. Según el número de imágenes utilizadas, se habla de visión binocular (dos imágenes), trinocular (tres imágenes) o n-cular (n imágenes) [104]. Las diversas técnicas existentes se clasifican en tres grupos principales: reconstrucción euclidiana o métrica, reconstrucción salvo un factor de escala y reconstrucción salvo una transformación proyectiva [40].

Reconstrucción euclidiana o métrica

Las técnicas que se encuentran en esta clasificación consideran que el sistema de visión se encuentra calibrado, es decir, se conocen los parámetros intrínsecos y extrínsecos de las dos cámaras. En este caso se obtiene una reconstrucción Euclidiana por triangulación de los puntos puestos en correspondencia que tiene la característica de ser única y por consecuencia está bien definida.

Reconstrucción salvo un factor de escala

Considérese el caso en el que sólo se conocen los parámetros intrínsecos de ambas cámaras, a partir de los cuales y de un conjunto de correspondencias se deriva un método para calcular los parámetros extrínsecos y la información tridimensional de

la escena [91]. Este método utiliza la matriz esencial para lograr tal fin por lo que se asume que se cuenta con al menos 8 correspondencias de la forma $\mathbf{w}_i \leftrightarrow \mathbf{w}'_i$.

A diferencia de la triangulación, donde se conoce por completo la geometría del sistema estereoscópico, en este caso no se cuenta con la información suficiente para localizar los puntos 3D de forma inequívoca. Además, puesto que no se conoce la línea base del sistema estereoscópico (distancia entre los centros ópticos de las cámaras), no es posible recuperar la verdadera escala de la escena, por lo tanto, la reconstrucción es única salvo un factor de escala desconocido. Dicho factor puede determinarse si conoce la distancia real entre dos puntos de la escena.

El origen de la ambigüedad en este método claramente se deriva del hecho que se requiere la estimación de la matriz esencial, la cual sólo se puede calcular salvo un factor de escala arbitrario, por lo tanto, es necesario buscar una normalización conveniente de dicha matriz [40].

Reconstrucción salvo una transformación proyectiva

Considere ahora que se tiene un sistema de n correspondencias $\mathbf{w}_i \leftrightarrow \mathbf{w}'_i$, y que no se conoce la pose relativa entre las cámaras y que además no estan calibradas. A pesar de que no se conocen los parámetros intrínsecos y extrísecos, es posible obtener una reconstrucción a partir de conocer sólo un sistema de puntos correspondientes y no se requiere ninguna otra información. El objetivo de la reconstrucción es encontrar las matrices de proyección de las cámaras y los puntos en la escena que satisfagan un modelo determinado [40].

Por otra parte, si se tienen muy pocas correspondencias este objetivo es imposible, sin embargo si existen suficientes correspondencias que permitan estimar la matriz fundamental de manera única, entonces la escena se puede reconstruir salvo una transformación proyectiva . Esta ambigüedad se puede reducir, por ejemplo a una transformación afín o a una transformación de similaridad, si se tiene información adicional bien definida sobre las cámaras o la escena. Este es un resultado muy importante y uno de los principales logros del enfoque no calibrado [40]. Este algoritmo básicamente consta de tres etapas: estimación de la matriz fundamental, estimación de las matrices de proyección y triangulación.

Existen muchas variantes de este método, por ejemplo, si las cámaras están calibradas, entonces puede utilizarse la matriz esencial en lugar de la matriz fundamental para calcular las matrices de proyección. Además, se puede utilizar la información sobre el movimiento de las cámaras, restricciones de la escena o calibraciones parciales de las cámaras para obtener mejoras en la reconstrucción.

2.3. Métodos binoculares

A partir de dos imágenes de una escena puede recuperarse la información de profundidad de los objetos dentro de la escena empleando como principio básico la triangulación de puntos correspondientes en ambas imágenes.

Técnicas basadas en la disparidad

El modelo binocular más simple que puede considerarse consta de un par de cámaras idénticas con sus ejes ópticos paralelos, separados un cierta distancia d y con los planos imagen también alineados (paralelos). Esta configuración permite que solo exista disparidad en uno de los dos ejes del plano imagen. Se conoce como disparidad a la ligera diferencia de posición debido a los dos puntos de vista proporcionados por ambas imágenes. La disparidad representa una forma de percibir la profundidad la cual permite calcular por triangulación la profundidad del objeto observado [43]. Los algoritmos que consideran este modelo obtienen dos imágenes, denominadas mapas de disparidad, correspondientes a cada una de las imágenes de entrada en donde se calcula un valor de disparidad para cada píxel. A partir de esta información se puede obtener la estructura de la escena mediante un proceso de triangulación.

El mapa de disparidad consiste en una nueva imagen, cuyos niveles de gris de los puntos representan la distancia en píxeles que existe desde el punto en una de las imágenes a otra, esto es, se toman las coordenadas del punto en la primera imagen y las de su punto correspondiente en la segunda imagen, la diferencia entre ellas es la intensidad del nivel de gris que se coloca en la nueva imagen. Geométricamente, esta diferencia representa en proporción inversa, la profundidad a la que se encuentra el punto tridimensional.

Una vez determinado el mapa de disparidades de la imagen se debe calcular la profundidad de la escena. El proceso de obtener un punto tridimensional a partir de sus proyecciones en varias imágenes se denomina triangulación. Para que se puede llevar a cabo es necesario conocer los puntos correspondientes en las imágenes y los parámetros de las cámaras obtenidos en el proceso de calibración, de este modo, encontrar la intersección de los rayos ópticos que van del centro óptico de cada cámara a su correspondiente punto imagen sería trivial. Sin embargo, existen errores asociados a la resolución de la cámara de manera que la intersección perfecta entre los puntos correspondientes y sus focos no siempre ocurre.

En [30] se presenta un algoritmo que obtiene mapas de disparidad analizando las discontinuidades en la profundidad mediante la correlación. El proceso de correlación se realiza dos veces, obteniéndose dos mapas de disparidad, los cuales se interpolan

considerando la información de las imágenes originales. Esta técnica genera pocos puntos con disparidad errónea, pero los resultados no son muy precisos y las imágenes resultantes se muestran difusas debido a la interpolación.

[7] describe un modelo basado en la teoría bayesiana que utiliza programación dinámica para encontrar la solución optima. Este modelo calcula los mapas de disparidad a partir de las imágenes de entrada, es decir, por el Teorema de Bayes se obtiene un modelo a posteriori de la escena considerando la información de las dos imágenes de entrada.

En [15] se presenta dos variantes de la técnica descrita en [7] que consisten en efectuar un análisis de la influencia de las oclusiones en el modelo de la escena e implementar tres modelos del mundo de creciente complejidad. Sin embargo, debido a las hipótesis y las consideraciones que se plantean sólo se aplica en un conjunto restringido de escenas.

Un algoritmo para el cálculo de disparidad a partir de una función de costo basado en el criterio de máxima verosimilitud se muestra en [42]. No se requiere conocer un modelo a priori de la escena, a diferencia del enfoque Bayesiano, sin embargo, se considera que las intensidades de los puntos correspondientes en ambas imágenes de entrada tiene una distribución normal centrada en la intensidad del punto real. La función de costo propuesta es minimizada con programación dinámica. También se agrega una serie de restricciones para minimizar el número de discontinuidades, obteniéndose un mapa de disparidad con mayor coherencia vertical. Esta técnica se puede aplicar a más de dos imágenes.

En [46] se describe un algoritmo para el cálculo de disparidad en donde se introduce los conceptos de puntos de control (GCP), que se definen como aquellos puntos relevantes del espacio de disparidad (DSI) en los que puede asegurarse la correspondencia. Este método modela las oclusiones y las integra en el proceso de cálculo con un costo mínimo mediante el uso de los GCP para encontrar el camino optimo a través del DSI en presencia de oclusiones grandes.

Otro de los algoritmos existentes para la obtención de los mapas de disparidad se describe en [57], el cual utiliza técnicas de corte de grafos para hallar la correspondencia entre los puntos de las imágenes, imponiendo una restricción de unicidad. La función a minimizar se basa en la función de energía de Potts, a la cual se le agrega un término que considera las oclusiones en la escena. De este modo, el problema de los puntos ocultos se modela dentro de la minimización de la función de energía.

Se presenta un algoritmo estereoscópico con bajo costo computacional y mapa de disparidad con un alto porcentaje de correspondencia en [80]. Este algoritmo integra técnicas clásicas como programación dinámica mejorada con la incorporación de varianza auto-ajustable para la obtención del mapa de disparidad; con multirresolución y extracción de puntos característicos (bordes y esquinas). Con esto se reduce el tiempo de cómputo y se mejora la calidad del mapa de disparidad final.

En [66] se propone un nuevo modelo estereoscópico que considera que la escena se compone de superficies planas, cuya idea principal consiste en la representación de tales superficies, es decir, donde se asigna cada píxel para una cierta superficie 3D. Esta representación tiene varias contribuciones importantes: se establece una clasificación de alto orden que establece que aquellos píxeles de apariencia similar se consideran pertenecientes a la misma superficie 3D, lo que permite incorporar una restricción basada en la segmentación de color. Por otro lado, se utiliza un algoritmo de reconocimiento y segmentación de objetos conocido como MDL (*Multiphase Dynamic Labeling Model*) para determinar el número de superficies dentro de la escena. Finalmente se utiliza un modelo de oclusión asimétrica mejorado que rechaza píxeles de la misma superficie que causen la oclusión de algún otro píxel.

Un nuevo enfoque basado en la segmentación a modo de optimizar el cálculo del mapa de disparidad en un sistema de visión estereoscópico se decribe en [67], cuya principal contribución es una mejora significativa de la calidad en las correspondencias de las oclusiones y áreas de baja textura, segmentando la imagen de color izquierda o una imagen de textura calculada. Además se utiliza una función local de costo basada en la correlación para determinar correctas correspondencias, que posteriormente se optimizan mediante el enfoque SGM (*Semi -Global Matching*) con una precisión subpíxel. Finalmente, el mapa de disparidad se obtiene sólo para las correspondencias antes calculadas en lugar de considerar las imágenes completas. En resultados experimentales se comprobó que el consumo de memoria se reduce significativamente.

Un método de construcción del mapa de disparidad se propone en [58], en donde se pone de manifiesto que la correlación entre las áreas de las imágenes se ve afectada por variaciones de iluminación, perspectiva y diferencias entre las imágenes (la cámara izquierda percibe la intensidad luminosa de forma ligeramente diferente a la cámara derecha). Para compensar estos efectos, en lugar de utilizar las imágenes originales para encontrar las correspondencias, se propone encontrarlas sobre las imágenes transformadas de alguna manera. La transformación propuesta es el Laplaciano del Gaussiano (LOG) con el fin de normalizar las imágenes, resaltar los bordes y utilizar la diferencia absoluta para calcular la correlación entre las dos ventanas. Para opti-

mizar los cálculos se utilizan dos reestricciones a modo de que la ventana tenga el mismo alto y ancho (ventana cuadrada) y que el valor máximo de disparidad esperado sea múltiplo de 16. Una vez obtenido el mapa de disparidad denso, se aplica un filtro para suavizar el resultado.

En [100] se describe una técnica denominada *Graph Cut* que obtiene una aproximación del mapa de disparidad cuyo enfoque que en lugar de utilizar los valores de intensidad alrededor de un píxel en una ventana de píxeles vecinos, utiliza el gradiente de luminosidad global de la imagen asumiendo explícitamente cierta suavidad sobre el mapa de disparidad y utilizando varias técnicas de minimización para obtenerlo. Por otro lado, el problema de correspondencia estereoscópica se formula como un problema de minimización de energía, donde se incluye energía de suavidad (smoothness energy) que mide la suavidad entre pares de píxeles vecinos y datos de energía (data energy) que mide la diferencia entre píxeles correspondientes en base a las disparidades asumidas. Después se construye un grafo con pesos en el que cada nodo representa un píxel de la imagen, un conjunto de etiquetas representan todos los posibles valores de disparidad y los pesos de las aristas del grafo corresponden con los valores de energía definidos. Este algoritmo se utiliza entonces para aproximar la solución óptima, asignando a cada píxel su valor correspondiente de disparidad.

Un algoritmo que modela las oclusiones y las integra en el proceso de cálculo de un costo mínimo mediante el uso de Programación Dinámica se presenta en [8], en donde se introduce el concepto de *Ground Control Points* y utilizan una imagen en el espacio de disparidad (DSI). El algoritmo se ejecuta para cada una de las filas, denominadas *scanlines*, de manera independiente y no se introduce información de las filas adyacentes. Después, para cada pareja de *scanlines* correspondientes se crea una imagen en el espacio de disparidad donde en cada píxel de esta imagen se asigna una medida de disparidad. Cuando el algoritmo no logra encontrar una pareja de correspondencias asume que el punto está oculto. Una de las principales característixas de este algoritmo es el uso de lo que los autores denominan *Ground Control Points* (GCP), que son puntos relevantes del DSI en los que puede asegurarse la correspondencia. Los mapas de disparidad obtenidos con este método son aproximados a la solución real.

En el trabajo presentado en [79] se aborda la problemática del proceso de correspondencia en imágenes estereoscópicas procedentes de escenas reales. Se utiliza una serie de algoritmos de correspondencia que forman parte del proceso estereoscópico global y cuya finalidad es realizar una reconstrucción 3D para la navegación de robots autónomos en entornos naturales y no estructurados. En primer lugar, se realiza un filtro homomórfico sobre las imágenes de entrada como paso previo a la correspondencia.

Mediante dicho filtro se consigue mejorar considerablemente el mapa de disparidad, logrando un mayor número de correspondencias verdaderas en relación a los procesos de correspondencia sin filtrado. Después, se realiza un filtrado del mapa de disparidad dirigido por clusters y basado en el principio de continuidad espacial con el que se consigue la eliminación de falsos positivos o correspondencias erróneas. Ambos filtrados constituyen la principal aportación del trabajo.

El método de *Corte de Grafos* es anterior a los problemas de Reconstrucción 3D, cuya implementación en esta área se presenta en [93]. El algoritmo se basa en encontrar la correspondencia entre los puntos de las imágenes, imponiendo la restricción de unicidad. Para resolver de forma eficiente el corte del grafo presentan dos algoritmos con los que obtienen tiempos mucho menores que con otros métodos. La función de energía propuesta considera la semejanza entre puntos correspondientes, basada en la diferencia en los niveles de gris de los mismos; un término de suavidad que intenta asignar una disparidad similar a los puntos cercanos, imponiendo una coherencia en todas las direcciones y finalmente un potencial de penalización cuando una pareja de puntos en una vecindad tiene diferentes disparidades.

Algoritmo del punto medio

Un modelo más general es aquel que no considera reestricciones en la colocación de las cámaras, de modo que éstas se pueden colocar de manera arbitraria. En este caso la triangulación se realiza mediante la obtención del *Punto Medio* de la normal común entre los rayos ópticos generados por cada par de puntos imagen [75]. A este método se le conoce como método del punto medio, es sencillo de implementar y debido a su tiempo de cómputo se puede aplicar en aplicaciones en tiempo real [76]. Resulta útil en reconstrucciones euclídeanas, sin embargo genera resultados erróneos si la colocación de las cámaras no es simétrica ya que el punto medio no es la mejor solución.

Existe otra técnica de reconstrucción 3D que basa su enfoque en encontrar una aproximación del punto tridimensional mediante un sistema de ecuaciones lineales [39]. Para resolverlas existen dos alternativas, en la primera se manipulan las ecuaciones obteniéndose un sistema de ecuaciones lineales homogéneas, resueltas mediante el enfoque de mínimos cuadrados, en donde se efectúa una descomposición en valores singulares. El inconveniente que presenta es su alto costo computacional [40].

La segunda alternativa consiste en acomodar las ecuaciones de modo tal que se genera un sistema ecuaciones lineales no homogéneo, el cual se resuelve utilizando el algoritmo de ecuaciones normales. Este algoritmo es sencillo de implementar y su tiempo de ejecución es muy similar al método del punto medio descrito anteriormente [40].

Otros algoritmos existentes

En [14] se presenta un algoritmo que efectúa la reconstrucción de un objeto empleando un sistema de visión binocular. Primero se realiza la correspondencia de puntos entre las dos imágenes capturadas a apartir de una función de correlación cruzada normalizada, aplicada sobre ventanas cuadradas de tamaño variable. Para determinar la correlación, los puntos deben satisfacer la reetricción epipolar. Después, con este sistema de correspondencias se obtiene un mapa de profundidad de donde se extrae una nube de puntos para posteriormente obtener un modelo 3D.

En [36] se describe un método binocular de optimización que calcula la intersección de los dos rayos ópticos que pasan por los puntos imagen y sus correspondientes centros ópticos, partiendo de la idea que los píxeles a triangular no intersecan en el espacio. De este modo, se deben encontrar los píxeles más cercanos a dichos puntos imagen tales que satisfagan la reestricción epipolar. Para refinar los resultados obtenidos se implementa un algoritmo iterativo lineal basado en mínimos cuadrados que reproyecta los puntos antes mencionados.

Hartley propusó un método denominado método *Polinomial Absoluto*, el cual no sólo se aplica a la reconstrucción euclidiana, sino que además se puede implementar en la reconstrucción salvo un factor de escala y en la reconstrucción proyectiva [39]. Por otra parte, si las matrices de proyección de ambas cámaras se calculan con precisión, este método es el óptimo para efectuar la extracción de la información tridimensional. Sin embargo, si los puntos en las imágenes presentan ruido y éstos son utilizados para estimar dichas matrices, los resultados que arroja no son tan buenos.

En [75] se presenta un método basado en las matrices de calibración de las cámaras, cuyas desventajas son: su modelo carece de significado geométrico para el problema, no es muy robusto dado que asigna un peso arbitrario a los parámetros de la cámara, de modo que si se cambia este peso, también cambia la solución de la reconstrucción.

En [52] se describe el diseño e implementación de un sistema basado en una cámara de adquisición de escenas 3D. El sistema usa una matriz de cámaras móviles digitales SLR (*Single Lens Reflex*) para capturar pares estereoscópicos de alto rango dinámico (HDR) sobre campos de 360^o de visión horizontal. Aumentando los pares estereo con patrones de luz proyectada y aplicando técnicas de reconstrucción estereoscópica se genera alta resolución, es decir, rangos de imágenes exactas del entorno con perfecta correspondencia en la textura de color HDR. Este enfoque con frecuencia funciona mejor que los escáneres de láser para áreas con propiedades reflectivas cambiantes. Además la textura de la superficie capturada a partir de la visión estereo es perfectamente registrada con la superficie resultante.

En [81] se presenta una recuperación trigonometrica directa de la geometría de una escena 3D a partir de una secuencia de imágenes. Estas imágenes, son rectificadas verticalmente usando salidas giroscópicas para minimizar la razón relativa de desplazamiento entre cada frame. Para que coincida esta secuencia de imágenes rectificadas verticalmente en tiempo real se pide prestado la fuerza del CNN (*Cellular Nonlinear Network*) y los resultados de coincidencia son trigonometricamente procesados por el rango de estimación 3D.

El trabajo presentado en [13] se enfoca en la representación y generación de vistas arbitrarias de escenas tridimensionales. Esta representación consiste en densos mapas de profundidad correspondientes a varios puntos de vista preseleccionados a partir de una secuencia de imágenes. En lugar de utilizar múltiples cámaras estacionarias calibradas, los mapeos de profundidad se obtienen a partir de una secuencia de imágenes capturadas por una cámara descalibrada. Una vez que los mapas de profundidad son calculados en los puntos de vista preseleccionados, la intensidad y la profundidad en estos sirve para reconstruir vistas arbitrarias de la escena 3D. El enfoque propuesto consiste en procesar un video a través de varias trayectorias de la escena para generar una secuencia de imágenes, mismas que serán usadas en la construcción de los mapas de profundidad. Una vez que la profundidad y la intensidad en los frames de referencia son guardados como una representación compacta de la escena, esta representación es usada para reconstruir vistas arbitrarias localizadas dentro o fuera de las trayectorias escaneadas.

Se ha desarrollado una forma de representar imágenes estereoscópicas a través de hologramas de Fourier que requieren un sólo paso en su generación en [82]. El holograma retiene la información de fase de la transformada de Fourier de la escena de entrada modificada y contiene además una fase lineal para lograr el efecto estereoscópico. La escena modificada consiste en la escena original a la que se agrega una fase aleatoria para garantizar una mejor distribución de la información en la fase de la transformada de Fourier. Para implementar el holograma se ha utilizado un modulador de luz espacial operando en distintas configuraciones. Se muestran resultados experimentales donde se aprecia la reconstrucción estereoscópica y se discuten además las limitaciones del dispositivo.

En [17] se describe un método de reconstrucción 3D que consta de dos partes. En la primera parte, se obtiene el mapa de profundidad con la correspondencia estereoscópica y la geometría de la cámara en cada vista. El algoritmo esta basado en un método de adaptación de ventana con un marco de trabajo jerárquico. En la segunda parte,

se usa la característica SIFT para estimar el movimiento de la cámara. El algoritmo LMEDS reduce el efecto outliers en este proceso.

En las aportaciones más recientes se encuentra el algoritmo descrito en [94] que realiza la reconstrucción basándose únicamente en reglas de geometría. Su método establece la formulación matemática de un sistema de ecuaciones que representa un conjunto de condiciones geométricas que deberá verificar el modelo 3D a partir del análisis de una imagen o de una porción de la misma. Se asume que topológicamente todos los vértices del objeto son triedros y que la imagen representa siempre un poliedro euleriano.

Se propone un método en tiempo real para analizar escenas tridimensionales de objetos a partir de una serie de imágenes monoculares en [53]. Una cámara simple observa algunos puntos sobre el objeto tal que la posición 3D de los puntos puede ser estimada por un filtro de Kalman extendido. El enfoque consta de dos etapas, primero se obtiene la geometría del objeto considerado y después se estima su posición. No se requieren datos del modelo a priori del objeto.

2.4. Métodos trinoculares

En [37] se propone una técnica muy efectiva que considera el uso del tensor trifocal, es decir, un tensor que tiene como base la geometría epipolar extendida a tres imágenes. De aquí se derivan diversos algoritmos para calcular el tensor trifocal, cada uno con ciertas ventajas que dependen de los objetos a reconstruir y de las posiciones de las cámaras [38],[89],[29].

La implementación del sistema de visión de un robot móvil se detalla en [18], el cual consiste de tres cámaras separadas una cierta distancia y distribuidas en un arreglo triangular. Con esta configuración se realizan dos cálculos de la disparidad uno para la cámara que queda arriba con la de la izquierda y otro para la cámara de arriba con la cámara de la derecha, para posteriormente combinar ambos resultados obteniendo un solo mapa de disparidad que contenga un número mínimo de errores. Una vez obtenido el mapa de disparidad se procede a calcular la profundidad de los puntos de la escena que está observando el robot.

En [37] se describe una reconstrucción métrica que requiere un mínimo de tres imágenes y en donde se utiliza una función que relaciona los puntos en la escena con los puntos imagen mediante los parámetros de calibración de las cámaras. Para estimar los valores de cada parámetro del sistema de visión se implementa el algoritmo

del gradiente descendente [36]. Esta técnica también se puede aplicar a dos imágenes, pero en este caso será necesario una estimación inicial de los parámetros del sistema de visión muy cerca de la solución para porder obtener resultados aceptables.

Por otro lado, en [102] se muestra un algoritmo que utiliza tres imágenes para realizar la reconstrucción 3D de una escena cuyo enfoque se deriva de las ecuaciones de Kruppa para su inicialización. Luego, se emplea una matriz de homografía sobre los puntos de las imágenes para recuperar los parámetros de la cámara y así poder obtener la información tridimensional de la escena.

2.5. Métodos n-culares

Conforme aumenta el número de imágenes capturadas de una misma escena, resulta más difícil encontrar correspondencias de puntos en dichas imágenes. Lo anterior se debe a que existen puntos que no estan presentes en todas las imágenes, por lo que resulta conveniente implementar un método que elimine los puntos que no sean visibles en todas las imágenes y emplear métodos iterativos que refinen los resultados obtenidos. Pese a estos inconvenientes, bajo ciertas restricciones, los resultados de la reconstrucción 3D son muy aceptables.

En [90] se propone un algoritmo denominado *Tensor Cuadrifocal* que considera cuatro imágenes. La idea fundamental es utilizar la geometría epipolar extendida a cuatro imágenes, sin embargo existen ciertas posiciones de la cámara en donde no es posible realizar la reconstrucción 3D con este método [28],[41].

Otro método que ha demostrado tener buen desempeño a partir de cuatro o más imágenes se describe en [16]. Con este método iterativo es posible estimar los parámetros de la cámara para cada imagen, al mismo tiempo que mediante un proceso de triangulación se recupera la información tridimensional. Se realiza un ajuste generado a partir de los parámetros de la cámara en relación con los puntos de la escena y los puntos de la imagen. Este ajuste fue propuesto inicialmente por Tomasi y Kanade en [88] para múltiples imágenes (cuatro o más) .

En [55] se presenta un método implementado en el sistema de visión de un robot con dos cámaras con una configuración estereoscópica. Este método requiere 5 o más pares de imágenes para poder efectuar la reconstrucción de una superficie plana. Al tomar las imágenes con ambas cámaras se producen dos vistas de la escena, mismas que se incorporan en dos planos distintos para lograr la reconstrucción .

En [68] se describe un método que emplea estructuras planas dentro de las imágenes para estimar las alineaciones entre ellas, y de este modo obtener los parámetros de la cámara. Se requieren al menos seis imágenes de la escena para obtener resultados favorables.

El método presentado en [71], utiliza un sistema de visión que consta de un cámara que captura imágenes de la rotación de objetos enfrente de ella, la rotación se lleva a cabo de manera constante y a intervalos definidos de modo que a cada imagen se le asocia una matriz de rotación. Una vez obtenida la secuencia de imágenes se efectua la detección de bordes con la técnica descrita en [35]. Posteriormente, se realiza la correlación de las disparidades en dos imágenes sucesivas empleando unicamente los puntos de interés previamente obtenidos. Cuando se han obtenido los puntos donde hay correlación, éstos se agregan a un modelo que considera los efectos de las rotaciones y translaciones. Cabe hacer notar que resulta indispensable que el sistema se encuentre correctamente calibrado para poder obtener resultados aceptables.

El trabajo descrito en [65] emplea redes neuronales para aprender la relación existente entre la imagen capturada, la superficie observada por la cámara y los puntos tridimensionales a los que pertenece. Para implementar este método se emplean sesenta imágenes, de las cuales se estima la forma tridimensional del área observada, para después hacer una composición de las formas obtenidas, generándose como resultado un objeto 3D de a partir del conjunto de imágenes. El costo computacional de esta técnica es relativamente menor, sin embargo, la desventaja que se presenta es que sólo se logra la reconstrucción de un objeto y no de la escena completa.

El trabajo presentado en [1] propone una técnica para la obtención de la profundidad a partir de una sola cámara, remplazando el mecanismo de apertura de la cámara por uno de apertura codificada, que consiste de un filtro al que se le perforan distintos patrones. De esta manera, se obtiene la disparidad dentro de una misma cámara, además las diferentes imágenes obtenidas presentan un grado de borrosidad de acuerdo a la distancia de los objetos. Para estimar el mapa de profundidad se realiza un análisis estadístico de los gradientes de la imagen, con lo cual se caracterizan las regiones borrosas y se estima el kernel (filtro) que produjo una región borrosa dada, de acuerdo al tamaño del kernel estimado se asigna la profundidad en la que se encuentra una determinada región.

A diferencia de los algoritmos de visión estereoscópica, en [25] se presenta un algoritmo para la obtención de la profundidad a partir del enfoque que consiste en tomar diferentes imágenes de una misma escena con un diferente grado de enfoque de

la lente de la cámara, logrando así que la escena sea vista de manera borrosa en un determinado grado. La profundidad de la escena se calcula con la ley de Gauss (ley de la lente delgada) que relaciona la distancia focal y el grado de borrosidad.

Capítulo 3

Marco Teórico

3.1. Modelo de proyección en perspectiva

Es realmente fácil describir los aspectos geométricos de la formación de una imagen. De hecho, estas leyes fueron ya entendidas por los pintores italianos del Renacimiento, los cuales estudiaron la geometría para reproducir correctamente los efectos de la perspectiva en las imágenes del mundo que ellos habían observado y que querían plasmar en sus obras [40]. La transformación de un espacio tridimensional a un plano bidimensional realizada por un cámara puede modelarse utilizando el modelo de proyección en perspectiva, cuya principal ventaja es que la relación entre las coordenadas de la escena y de la imagen es una proyección lineal, siendo esta propiedad independiente de la elección de los referenciales asociados a la imagen y la escena. Así pues, esta resulta ser la razón principal para utilizar el formalismo proyectivo para describir el mapeo de puntos en la escena al plano imagen de una cámara. Además, este modelo ignora efectos no lineales tales como aquellos causados por las distorsiones de las lentes.

Considere la proyección de puntos pertenecientes al espacio $3D$ sobre un plano. Sea el centro de la proyección el origen de un referencial derecho y considere que el plano de proyección se ubica en $Z = f$, el cual se conoce como plano imagen o plano focal mientras que f se conoce como distancia focal, que no es otra cosa que la distancia entre el centro de proyección y el plano imagen. Bajo el modelo de proyección en perspectiva, un punto $\mathbf{M}$ en el espacio se mapea a un punto $\mathbf{m}$ en el plano imagen el cual se localiza donde la línea que une el punto $\mathbf{M}$ con el centro de proyección interseca al plano imagen, tal como se ilustra en la figura 3.1. Este es un mapeo de un espacio euclidiano tridimensional $\mathbb{R}^3$ a un espacio euclidiano bidimensional $\mathbb{R}^2$.

El centro de proyección se conoce como centro de la cámara o centro óptico de la cámara y se denota por $\mathbf{C}$. La línea que pasa por el centro óptico y que es perpendicular

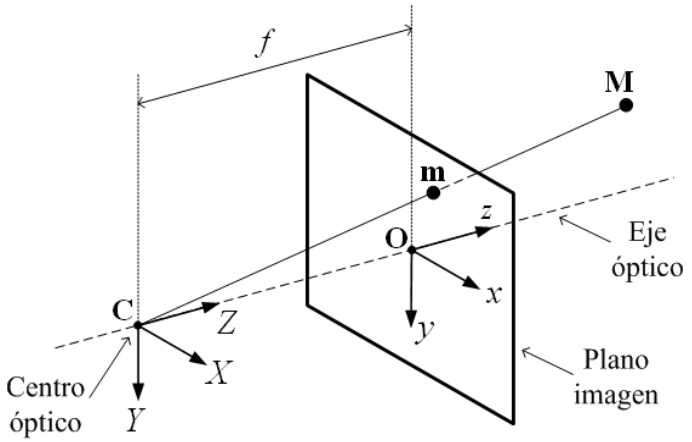


Figura 3.1: Modelo de proyección en perspectiva.

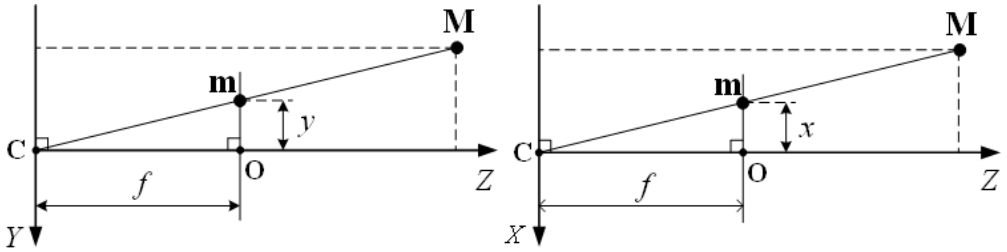


Figura 3.2: Geometría del modelo de proyección en perspectiva.

al plano imagen se conoce como eje principal de la cámara o simplemente eje óptico. Además, el punto donde el eje óptico interseca al plano imagen se conoce como punto principal de la imagen y el cual se denota por **O**. Ahora bien, considere un referencial **O**xyz asociado al plano imagen que tiene su origen en el punto principal de la imagen **O**, este referencial se asigna de modo que su eje z se alinea con el eje óptico. Y además considere el referencial **C**XYZ centrado en el centro óptico y que es paralelo al referencial del plano imagen, que se conoce como referencial de la cámara.

De la figura 3.1 se observa que la proyección de un punto **M** de la escena en el plano imagen forma un esquema basado en triángulos semejantes como se muestra en la figura 3.2. Utilizando el Teorema de Tales se obtiene que:

$$\begin{aligned} Zx &= fX \\ Zy &= fY \end{aligned} \tag{3.1}$$

donde $[x, y]^T$ representan las coordenadas del punto proyectado **m** expresadas en

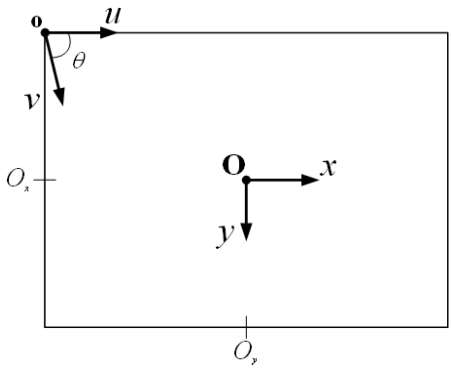


Figura 3.3: Sistemas de coordenadas de la imagen y del captador.

el referencial de la imagen y $[X, Y, Z]^T$ las coordenadas del punto **M** expresadas en el referencial de la cámara. Las dos expresiones en la ecuación (3.1) son no lineales, pero si ambos puntos se representan mediante vectores homogéneos, entonces el modelo de proyección en perspectiva se puede expresar como un mapeo lineal. En particular, la ecuación (3.1) se puede escribir en términos del producto de matrices como:

$$Z \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.2}$$

El modelo descrito anteriormente supone que el origen del referencial de la imagen se ubica en el punto principal de la misma, sin embargo, para algunas aplicaciones resulta más conveniente expresar las coordenadas de los puntos imagen con respecto a un referencial ubicado en el captador, **o***uv*,el cual se ubica en la esquina superior izquierda de la imagen tal como lo muestra en la figura 3.3. Por lo tanto la ecuación (3.2) se convierte en:

$$\lambda \widetilde{\mathbf{m}} = \begin{bmatrix} f & 0 & x_o & 0 \\ 0 & f & y_o & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \widetilde{\mathbf{M}} \tag{3.3}$$

donde x_o y y_o representan los desplazamientos en las direcciones x y y respectivamente entre los dos referenciales. Además $\widetilde{\mathbf{M}} = [X, Y, Z, 1]^T$ y $\widetilde{\mathbf{m}} = [x, y, 1]^T$ representan los vectores homogéneos de los puntos **M** y **m** respectivamente. Además λ es un factor de escala que permite mantener la igualdad y cuyo valor es igual a Z, es decir, la profundidad del punto **M**.

Para poder expresar las coordenadas de la imagen en unidades de píxeles es nece-

sario introducir un factor de escala en cada dirección, los cuales serán diferentes en el caso de que los píxeles no sean cuadrados. Sean m_u y m_v el número de píxeles por unidad de longitud en las direcciones u y v respectivamente se tiene que:

$$\lambda\widetilde{\mathbf{w}} = \begin{bmatrix} \alpha_u & 0 & u_o & 0 \\ 0 & \alpha_v & v_o & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \widetilde{\mathbf{M}} \tag{3.4}$$

donde $\mathbf{w} = [u, v]^T$ es el punto imagen expresado en el nuevo referencial (captador) y en unidades de píxeles, mientras que $\alpha_u = fm_u$ y $\alpha_v = fm_v$ representan la distancia focal de la cámara en píxeles en las direcciones u y v respectivamente. Además $[u_o, v_o]^T$ son las coordenadas del punto principal también en píxeles y en el referencial del captador, con $u_o = m_u x_o$ y $v_o = m_v y_o$.

Para más generalidad, considere el parámetro s el cual se refiere al parámetro de *skew*. Dicho parámetro es simplemente una compensación $s = \alpha_u \cot\theta$ de los posibles errores de ortogonalidad del chip, donde θ es el ángulo entre los ejes del referencial del captador como se observa en la figura 3.3. El parámetro de *skew* será cero para la mayoría de las cámaras, sin embargo, en raras ocasiones, dependiendo de la calidad y el costo de la cámara puede tomar valores distintos de cero [27]. Incluyendo este parámetro resulta:

$$\lambda\widetilde{\mathbf{w}} = \begin{bmatrix} \alpha_u & s & u_o & 0 \\ 0 & \alpha_v & v_o & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \widetilde{\mathbf{M}} \tag{3.5}$$

A los elementos α_u, α_v, u_o, v_o y s se les conoce como parámetros intrínsecos de la cámara, los cuales definen la geometría y la óptica de la cámara.

Se ha considerado hasta ahora que las coordenadas de los puntos $\mathbf{M}$ en la escena se expresan en el referencial de la cámara, pero en muchas aplicaciones resulta más conveniente expresarlas en un referencial asociado a la escena, $\mathbf{O}^w X^w Y^w Z^w$, tal como se muestra en la figura 3.4 [26]. Ambos referenciales se relacionan mediante una matriz de transformación homogénea $\mathbf{T}$ que incluye una rotación $\mathbf{R}$ y una traslación $\mathbf{t}$.

Sea $\widetilde{\mathbf{M}}^w = [X^w, Y^w, Z^w]^T$ las coordenadas de un punto $\mathbf{M}$ en la escena expresadas en el referencial asociado a la misma, entonces sus coordenadas en el referencial de la cámara están dadas por:

$$\widetilde{\mathbf{M}} = \mathbf{T}\widetilde{\mathbf{M}}^w \tag{3.6}$$

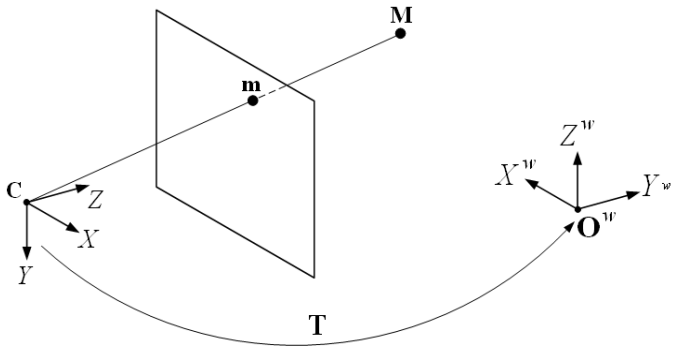


Figura 3.4: Sistema de coordenadas del mundo.

A los parámetros de esta matriz de transformación homogénea se les conoce como parámetros extrínsecos de la cámara, los cuales representan la posición y orientación de la cámara con respecto a un referencial asociado a la escena.

Sustituyendo la ecuación (3.6) en la ecuación (3.5) se obtiene:

$$\lambda\widetilde{\mathbf{w}} = \begin{bmatrix} \alpha_u & s & u_o & 0 \\ 0 & \alpha_v & v_o & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \widetilde{\mathbf{M}}^w \tag{3.7}$$

Haciendo las operaciones adecuadas se tiene:

$$\lambda\widetilde{\mathbf{w}} = \mathbf{K}\,[\mathbf{R} \mid \mathbf{t}]\,\widetilde{\mathbf{M}}^w \tag{3.8}$$

donde a $\mathbf{K}$ se le conoce como matriz de calibración y sus elementos son los parámetros intrínsecos de la cámara, es decir:

$$\mathbf{K} = \begin{bmatrix} \alpha_u & s & u_o \\ 0 & \alpha_v & v_o \\ 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

Se define la matriz de proyección de la cámara como:

$$\mathbf{P} = \mathbf{K}\,[\mathbf{R} \mid \mathbf{t}] \tag{3.10}$$

la cual depende de los parámetros intrínsecos y extrínsecos de la cámara.

3.2. Reconstrucción Estereoscópica

De acuerdo con la expresión que representa al modelo completo de proyección en perspectiva se tiene que a partir de una sola imagen no es posible obtener la profundidad a la que se encuentran los objetos presentes en la escena excepto en algunos casos particulares [59]. Sin embargo, si se cuenta con dos cámaras que produzcan dos imágenes de cada uno de estos objetos, es posible extraer información de profundidad y de esta manera se completaría el modelo tridimensional de la escena que se está observando.

Por otra parte, la reconstrucción 3D estereoscópica es el método más fácilmente asumible por la mente humana debido a su extrema similitud con el proceso de visión humano. En efecto, resulta más sencillo y preciso recuperar la tridimensionalidad de una escena utilizando, al menos, una segunda cámara orientada hacia la misma escena, y que esté mirando desde otra perspectiva cercana pero diferente de la primera. De esta manera, analizando las dos imágenes y utilizando la geometría del sistema formado por las dos cámaras y la escena es como se puede recuperar la profundidad de cada uno de los objetos visibles. Ahora bien, los problemas principales que se presentan en la reconstrucción 3D varían de acuerdo al método seleccionado; pero, en el caso de la visión estereoscópica se presentan tres grandes problemas [26], los cuales son:

Problema de correspondencia: Sea un punto $\mathbf{M}$ en la escena 3D el cual se proyecta como $\mathbf{m}$ en la primera imagen y como $\mathbf{m}'$ en la segunda imagen, el problema de correspondencia también conocido como problema de apareamiento trata de buscar qué par de puntos, $\mathbf{m}$ y $\mathbf{m}'$corresponden a un mismo punto $\mathbf{M}$ del espacio tridimensional. En otras palabras, dado un punto en una de las imágenes se debe averiguar a qué punto corresponde en la otra imagen. De manera implícita este problema considera la detección de características, es decir, la identificación de puntos en la imagen que de algún modo sean de importancia tales como los bordes, esquinas, líneas, blobs, etc.

Problema de calibración: La calibración de una cámara representa una etapa importante en la reconstrucción 3D de la escena, la cual consiste en estimar los parámetros de intrínsecos y extrínsecos de dicha cámara. Dichos parámetros definen las condiciones de formación de la imagen incluyendo la geometría interna y la óptica de la cámara, así como su posición y orientación. En definitiva, la calibración es un procedimiento que trata de conocer cómo se proyecta el mundo 3D en el plano imagen para así poder extraer información métrica de él a partir de imágenes 2D.

Problema de reconstrucción: Si se conocen dos puntos correspondientes $\mathbf{m}$ y $\mathbf{m}'$ en un par de imágenes, el problema de la reconstrucción trata de encontrar las

coordenadas (X, Y, Z) del punto $\mathbf{M}$ en la escena.

La tarea más difícil es sin duda responder al problema de correspondencia. En general existen varias posibilidades para escoger el elemento correspondiente en la segunda imagen dado un elemento en la primera imagen, por lo que se dice que el problema de correspondencia es ambiguo. Debido a esta ambigüedad, es necesario averiguar qué elementos, qué características, qué restricciones y qué consideraciones se pueden aplicar para reducirla al máximo.

3.3. Detección de características y correspondencia de puntos

La detección de características (features, puntos de interés o puntos clave) en una imagen representa un aspecto fundamental en muchos problemas de Visión Artificial tales como la correspondencia de puntos, detección, reconocimiento y seguimiento de objetos, navegación de robots y elaboración de imágenes panorámicas por mencionar algunos [35]. La dificultad de estos problemas se debe en gran parte a la falta de éxito en la detección de características en las imágenes. Por ejemplo, para el reconocimiento de objetos se requiere que las características no se vean afectadas por el desorden de la imagen o por la oclusión parcial de la escena. Además, las características deben ser invariantes a los cambios de iluminación y escala de la escena. Por otra parte, resulta conveniente que las características sean suficientemente distinguibles para poder identificar objetos específicos de entre todos los que están presentes en la escena.

Otro ejemplo que representa la importancia de la detección de características es cuando se tienen dos imágenes de una escena capturadas con un sistema de visión estereoscópico y se desea realizar la Reconstrucción 3D de la misma. En este caso resulta indispensable, entre otras cosas, hacer una correspondencia de puntos en ambas imágenes. Computacionalmente es imposible hacer la correspondencia comparando todos los píxeles en las dos imágenes, por lo cual resulta conveniente tratar de relacionar las imágenes haciendo corresponder sólo puntos que de alguna manera resulten interesantes. Tales puntos se denominan puntos de interés, rasgos, características o puntos clave y se localizan utilizando un detector de características. De esta manera, para reducir considerablemente el tiempo de cómputo, sólo deben utilizarse estos puntos para realizar la correspondencia entre las dos imágenes. Dada su importancia en diversas aplicaciones, la detección de características ha recibido una atención considerable y por consecuencia se han propuesto diversos detectores con diferentes enfoques y con una amplia gama de definiciones acerca de qué puntos en la imagen son de interés. Al-

gunos detectores encuentran puntos con alta simetría local, otros encuentran áreas con una gran variación de textura, mientras que algunos más encuentran bordes, esquinas, líneas, círculos, blobs (manchas). Un buen detector de características debe detectar características reales , ser preciso (características bien localizadas), deber tener un alto índice de repetibilidad (porcentaje del total de características detectadas en una secuencia de imágenes), debe ser robusto al ruido de la imagen y computacionalmente eficiente [35].

3.3.1. Seeded-Up Robust Feature (SURF)

SURF (*Speeded-Up Robust Features*) es un algoritmo de detección de características locales que son invariantes a la escala y rotación de la imagen y parcialmente invariantes a distorsiones afines, cambios de iluminación, cambios en el punto de vista de la cámara, oclusión y ruido en la imagen . Su enfoque se basa en el uso de la matriz Hessiana tal como sucede en el operador SIFT [63], siendo la diferencia entre ambos la manera de aproximar dicha matriz, esto es, SIFT utiliza diferencias de Gaussianas (DoG), mientras que SURF lo hace mediante el uso de un tipo de imágenes denominadas integrales, logrando así una drástica reducción en el costo computacional [60]. Con esta técnica se obtiene un representación de la imagen basada en unos vectores denominados descriptores que contienen la información de luminosidad en la vecindad alrededor de cada característica detectada. Estos vectores son invariantes a las transformaciones antes mencionadas por lo que son altamente distintivos y por consecuencia muy útiles en tareas como la correspondencia de puntos entre imágenes estereoscópicas y el reconocimiento de objetos, entre otras [69]. Los descriptores sólo tienen 64 dimensiones, reduciendo así el tiempo para la detección de características y la correspondencia de puntos, y aumentando al mismo tiempo la robustez del operador. También se agrega una etapa de indexado basada en el signo del Laplaciano para descartar falsas correspondencias y aumentar la velocidad de cómputo.

Imágenes integrales

Gran parte del aumento en el rendimiento (reducción del costo computacional) de SURF se le puede atribuir al uso de una representación de la imagen conocida como imagen integral [95]. Sus principales ventajas son la rapidez con que se obtienen y la disminución del tiempo computacional a la hora de calcular la suma de las intensidades de los píxeles dentro de una región rectangular. Dada una imagen de entrada $I(u, v)$ y un punto $\mathbf{w} = [u, v]^T$, el valor de luminosidad de la imagen integral en la posición correspondiente a dicho punto, denotada como $I_\Sigma(\mathbf{w})$, se calcula como la suma de las intensidades de todos los píxeles dentro de la región rectangular formada por el punto

w y el origen **o** del referencial del captador tal como se muestra en la figura 3.5.

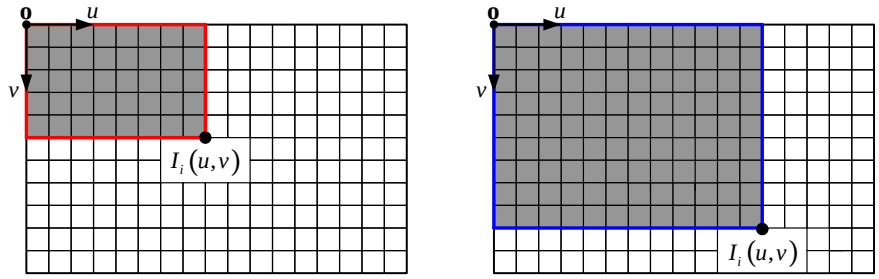


Figura 3.5: Imágenes integrales.

Esta suma se puede escribir como:

$$I_\Sigma(\mathbf{w}) = \sum_{i=0}^{i\leq u}\sum_{j=0}^{j\leq v} I(i,j) \tag{3.11}$$

Una vez calculada la imagen integral sólo se requieren tres adiciones para calcular la suma de las intensidades de todos los píxeles en una región rectangular arbitraria. Por ejemplo, considere el rectángulo acotado por los vértices **A**, **B**, **C** y **D** mostrado en la figura 3.6, la suma de las intensidades luminosas de los píxeles dentro de él se calcula como:

$$\sum = I_\Sigma(\mathbf{A}) + I_\Sigma(\mathbf{D}) - [I_\Sigma(\mathbf{B}) + I_\Sigma(\mathbf{C})] \tag{3.12}$$

donde $I_\Sigma(\mathbf{A})$, $I_\Sigma(\mathbf{B})$, $I_\Sigma(\mathbf{C})$, y $I_\Sigma(\mathbf{D})$ representan los valores de la imagen integral en las posiciones de los vértices del rectángulo. Así pues, el tiempo de cálculo es independiente del tamaño del rectángulo, lo que resulta útil cuando se analizan regiones rectángulares grandes. Esta propiedad se aprovecha para realizar convoluciones rápidas de la imagen con unos filtros conocidos como filtros de caja (*box filters*) de diferentes tamaños.

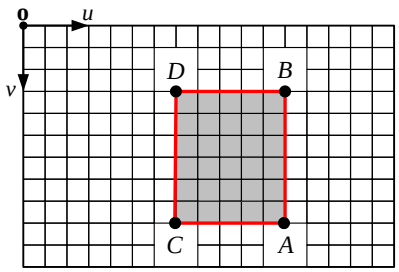


Figura 3.6: Aplicación de las imágenes integrales.

Matriz Hessiana

La matriz Hessiana resulta ser una buena herramienta debido a su buen desempeño para detectar características en una imagen, esto es, las características se detectan como aquellos puntos donde el determinante es máximo [60]. Para más detalle, considere una función continua de dos variables tal que su valor en $[u, v]^T$ está dado por $f(u, v)$. La matriz Hessiana, $\mathbf{H}$, se define como la matriz de las segundas derivadas parciales de la función f:

$$\mathbf{H}(f(u,v)) = \begin{bmatrix} \frac{\partial^2 f}{\partial u^2} & \frac{\partial^2 f}{\partial u \partial v} \\ \frac{\partial^2 f}{\partial u \partial v} & \frac{\partial^2 f}{\partial v^2} \end{bmatrix} \tag{3.13}$$

y su determinante (discriminante o Hessiano) se calcula como:

$$\det(\mathbf{H}) = \frac{\partial^2 f}{\partial u^2}\frac{\partial^2 f}{\partial v^2} - \left(\frac{\partial^2 f}{\partial u \partial v}\right)^2 \tag{3.14}$$

Con el valor del Hessiano se obtienen los máximos y mínimos de la función $f(u, v)$ mediante el criterio de la segunda derivada y su valor se obtiene como el producto de los valores característicos de $\mathbf{H}$, de modo que los píxeles se pueden clasificar mediante el signo del determinante. Si es negativo, los valores característicos tienen signos diferentes y el píxel no es un extremo local, por el contrario, si el determinante es positivo entonces ambos valores característicos son positivos o bien ambos negativos y en cualquiera de los casos el píxel se clasifica como un extremo local. Para aplicar este principio a una imagen, en lugar de considerar una función continua, primero se debe sustituir la función $f(u, v)$ por la imagen $I(u, v)$. Para calcular las entradas de la matriz $\mathbf{H}$ (derivadas parciales de la imagen) se utiliza un filtro Gaussiano dado que las imágenes resultantes son adecuadas para el análisis en el espacio escala, esto es, al usar este filtro es posible variar la cantidad de suavizado (escala), obteniendo así el determinante a diferentes escalas [60], [56]. Por lo tanto, la matriz Hessiana se calcula como una función de los espacios escala e imagen. Dado un punto $\mathbf{w} = [u, v]^T$ en la imagen $I(u, v)$, la matriz Hessiana, $\mathbf{H}(\mathbf{w}, \sigma)$, en el punto $\mathbf{w}$ y en la escala σ se define como:

$$\mathbf{H}(\mathbf{w}, \sigma) = \begin{bmatrix} L_{uu}(\mathbf{w}, \sigma) & L_{uv}(\mathbf{w}, \sigma) \\ L_{uv}(\mathbf{w}, \sigma) & L_{vv}(\mathbf{w}, \sigma) \end{bmatrix} \tag{3.15}$$

donde $L_{uu}(\mathbf{w}, \sigma)$ representa la convolución de la segunda derivada del filtro Gaussiano, $\frac{\partial^2 g(\sigma)}{\partial u^2}$, con la imagen en el punto $\mathbf{w}$. De manera similar se definen $L_{vv}(\mathbf{w}, \sigma)$ y $L_{uv}(\mathbf{w}, \sigma)$, a estas derivadas se les conocen como Laplaciano del Gaussiano. A partir de este enfoque se puede calcular el Hessiano para cada píxel en la imagen y utilizar

su valor para encontrar puntos de interés, sin embargo en la práctica la función Gaussiana debe discretizarse y recortarse (figura 3.7), lo que afecta de manera negativa la detección de características. Para contrarestar este efecto, la aproximación se obtiene mediante filtros de caja (figura 3.7) utilizando las imágenes integrales para reducir el costo computacional. De este modo, el desempeño de la aproximación del Laplaciano mediante filtros de caja resulta ser mejor en comparación con las Gaussianas discretizadas y recortadas.

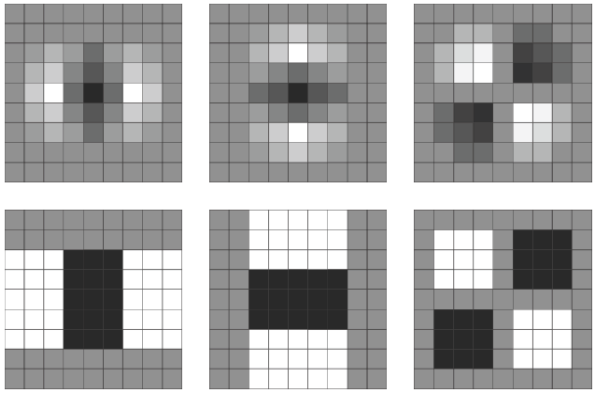


Figura 3.7: Arriba. Segundas derivadas parciales Gaussianas discretizadas y recortadas. Abajo. Aproximación a partir de filtros de caja (box filters).

En la figura 3.7 se muestran filtros de caja 9×9 que representan las aproximaciones de las segundas derivadas de una Gaussiana con desviación estándar $\sigma = 1{,}2$. Estos filtros se denotan como D_{uu}, D_{vv} y D_{uv}. Los pesos (ponderación) aplicados a cada una de las regiones del filtro se mantienen simples para una mayor eficiencia computacional, es decir, para el filtro D_{uv} las regiones negras se ponderan con un valor de 1, las regiones blancas con un valor de -1 y las regiones restantes no se ponderan. Los filtros D_{uu} y D_{vv} se ponderan de manera similar pero ahora las regiones blancas tienen un valor de -1 y las negras un valor de 2. La ponderación simple permite un cálculo rápido de las áreas, sin embargo se requiere balancear la diferencia entre las respuestas de los filtros original y aproximado mediante la siguiente expresión:

$$\det\left(\mathbf{H}_{aprox}\right) = D_{uu}D_{vv} - \left(wD_{uv}\right)^2 \tag{3.16}$$

donde el peso w es precisamente el encargado de lograr el balance y el valor que genera la aproximación más precisa está dado por:

$$w = \frac{\left|L_{uv}\left(1{,}2\right)\right|_F \left|D_{uu}\left(9\right)\right|_F}{\left|L_{uu}\left(1{,}2\right)\right|_F \left|D_{uv}\left(9\right)\right|_F} = 0{,}912\ldots \simeq 0{,}9 \tag{3.17}$$

donde $|x|_F$ representa la norma de Frobenius. Note que para esta corrección teórica los cambios de ponderación dependen de la escala, sin embargo, resultados experimentales demostraron que en la práctica este factor (w) se puede mantener constante debido a que no tiene un impacto significativo en los resultados obtenidos. La pérdida de precisión al utilizar los filtros de caja es superada por el considerable aumento de la eficiencia y la velocidad de cómputo.

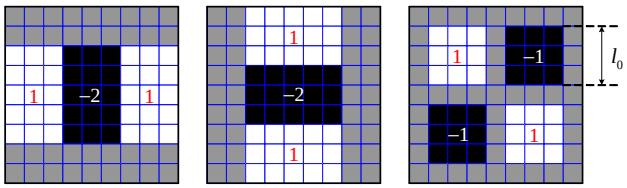


Figura 3.8: Ponderación de los filtros de caja (box filters).

3.3.2. Construcción del espacio escala

Un espacio escala es una función continua que se utiliza para encontrar puntos de interés en todas las escalas posibles y que generalmente se implementa como una pirámide de imágenes convolucionando progresivamente la imagen original con un filtro Gaussiano, depués se efectua un submuestreo (reducción de tamaño) para obtener el siguiente nivel de la pirámide [97]. La convolución se realiza con filtros de caja de diferentes tamaños, permitiendo procesar varios niveles del espacio escala de manera simultánea y elimina la necesidad de submuestrear la imagen. De este modo, el espacio de la escala se forma aumentando la escala (tamaño) del filtro en lugar de reducir progresivamente el tamaño de la imagen tal como se observa en la figura 3.9.

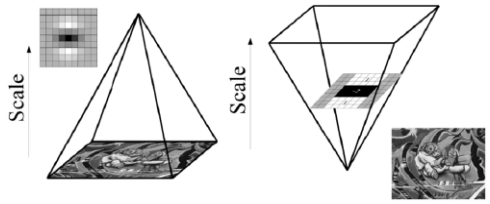


Figura 3.9: Construcción del espacio escala: Forma tradicional (izq). Enfoque SURF (der).

El espacio escala se divide en octavas y cada octava representa un conjunto de imágenes obtenidas al convolucionar la imagen original con un filtro que aumenta su tamaño. Cada octava se subdivide en un número constante de niveles de escala. Se

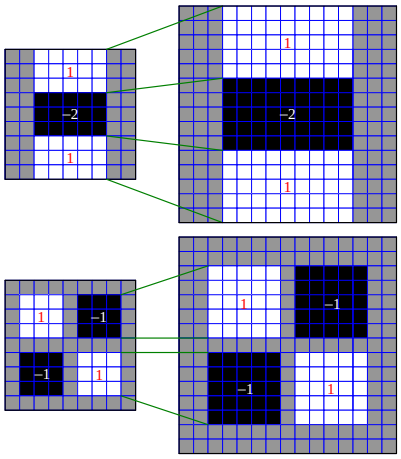


Figura 3.10: Aumento del tamaño del filtro.

inicia con un filtro de 9×9 (escala más pequeña) y posteriormente se aplican filtros de tamaño 15×15, 21×21 y 27×27. Para cada nueva octava, el incremento en el tamaño del filtro se duplica, esto es, en la primera octava el incremento es de 6 píxeles, en la segunda de 12 píxeles, 24 para la tercera, 48 para la cuarta octava y así sucesivamente. Por lo tanto, los tamaños de filtro para la segunda octava son 15, 27, 39, 51. La tercera octava se calcula con los filtros de tamaño 27, 51, 75, 99. Si el tamaño de la imagen original es todavía más grande que el tamaño del filtro entonces se construye la cuarta octava utilizando filtros de 51, 99, 147 y 195. La figura 3.11 muestra una visión general de los tamaños del filtro para las primeras tres octavas.

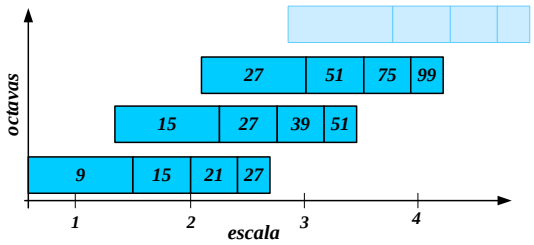


Figura 3.11: Tamaños del filtro para las primeras tres octavas.

Se pueden considerar más octavas, sin embargo tal como puede verse en la figura 3.12 el número de características disminuye drásticamente a medida que la escala aumenta [61].

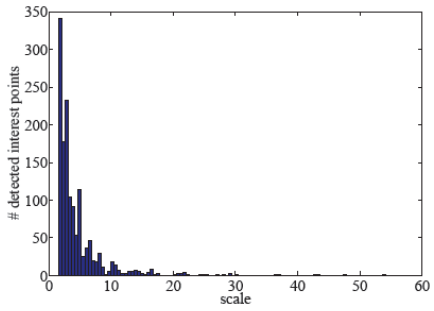


Figura 3.12: Histograma de las escalas analizadas.

3.3.3. Localización de características

La tarea de localizar puntos de interés invariantes a la escala y rotación de la imagen puede dividirse en tres etapas. Primero, una vez construido el espacio escala, se define un umbral t de modo que todos aquellos puntos cuya intensidad luminosa este por debajo del umbral se descartan. Tenga en cuenta que a medida que el umbral aumenta su valor, el número de puntos de interés detectados disminuye, conservándose sólo los más "*fuertes*", mientras que si este disminuye permite detectar muchos más puntos de interés. Por lo tanto, el umbral debe adaptarse de acuerdo a la aplicación y/o al tipo de imágenes utilizadas. Después se realiza una supresión de no-máximos, para lo cual cada píxel se compara con sus 26 vecinos, esto es, con sus 8 vecinos en la escala actual y con sus 9 vecinos en las dos escalas adyacentes (escalas superior e inferior) de manera que si la intensidad luminosa de este es menor o mayor que todos ellos se considera como un punto de interés candidato tal como se observa en la figura 3.13. Así pues, se obtiene un conjunto de características con una intensidad mínima determinada por el umbral t, los cuales representan extremos locales (máximos o mínimos) en el espacio escala.

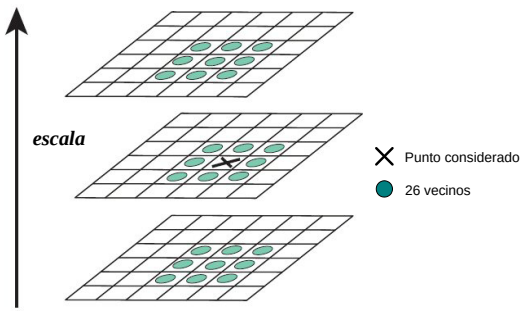


Figura 3.13: Supresión de no-máximos.

Finalmente, deben refinarse la posición y escala de cada punto de interés interpolando el determinante de la matriz Hessiana con una función cuadrática $3D$ [11]. Para realizar esta interpolación es necesario expresar el determinante de la matriz Hessiana, $\mathbf{H}(u, v, \sigma)$, mediante una expansión en series de Taylor centrada en el punto analizado considerando sólo hasta el término cuadrático:

$$\mathbf{H}(\mathbf{x}) = \mathbf{H} + \frac{\partial \mathbf{H}^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 \mathbf{H}}{\partial \mathbf{x}^2}\mathbf{x} \tag{3.18}$$

La posición (precisión subpíxel) y escala interpoladas, $\hat{\mathbf{x}} = (u, v, \sigma)$, se calculan derivando e igualando a cero la ecuación (3.18):

$$\hat{\mathbf{x}} = \frac{\partial^2 \mathbf{H}^{-1}}{\partial \mathbf{x}^2}\frac{\partial \mathbf{H}}{\partial \mathbf{x}} \tag{3.19}$$

Las derivadas se aproximan mediante diferencias finitas de píxeles cercanos [6].

3.3.4. Descriptor de características

Los descriptores son vectores de contienen la información de la distribución de luminosidad de los píxeles dentro de una vecindad alrededor de cada punto de interés. Esta distribución se obtiene a partir de unos filtros conocidos como *wavelets de Haar* (ver figura 3.14) que permiten calcular el gradiente de luminosidad en las direcciones u y v de la imagen. El uso de estos filtros en conjunto con las imágenes integrales permite aumentar la robustez del descriptor, al mismo tiempo que disminuye el costo computacional.

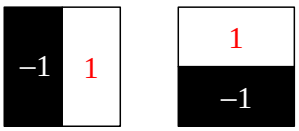


Figura 3.14: Wavelets de Haar.

El cálculo del descriptor se divide en dos etapas:

1. Asignar una orientación a cada característica.
2. Construir una vecindad alrededor de cada punto de interés de donde se extrae un vector $64-$*dimensional* o descriptor.

Todos los cálculos para la obtención del descriptor son dependientes de la escala del punto de interés (escala en la que fue detectado), logrando así la invarianza a la escala de la imagen.

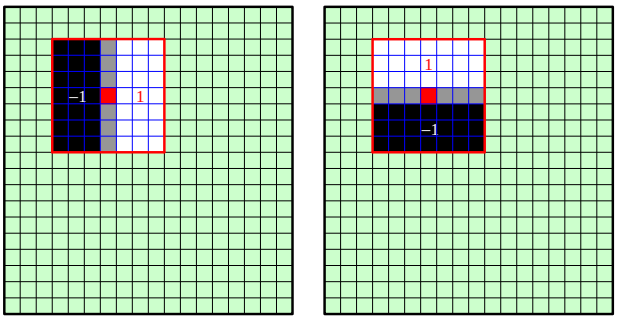


Figura 3.15: Calculo del gradiente utilizando los wavelets de Haar.

Asignación de la orientación

Para obtener un descriptor invariante a la rotación de la imagen se asigna una orientación a cada característica calculando las respuestas a los wavelets de Haar para un conjunto de píxeles dentro de una vecindad alrededor de ella. Primero se construye una vecindad circular de $6s$ de radio, donde s representa la escala del punto de interés. Luego se calculan las respuestas a los wavelets de Haar para cada píxel de dicho conjunto, cuyo tamaño depende de la escala y su longitud lateral es de $4s$. Una vez obtenidas las respuestas a los wavelets de Haar, estas se ponderan usando una Gaussiana con desviación estándar $\sigma = 2{,}5s$. Las respuestas ponderadas se representan como puntos en un espacio bidimensional con la respuesta horizontal a lo largo de la abscisa y la respuesta vertical a lo largo de la ordenada tal como se observa en la figura 3.16.

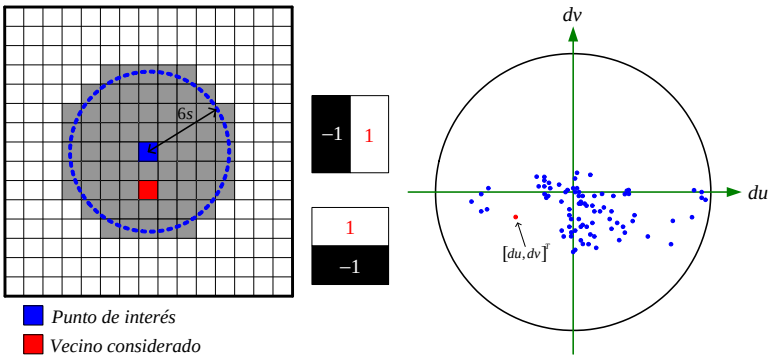


Figura 3.16: Respuestas horizontales y verticales con los wavelets de Haar.

La orientación dominante se calcula rotando alrededor del origen un segmento circular con un ángulo central de $\pi/3$ y sumando las respuestas dentro él, con lo que se obtiene un nuevo vector tal como puede observarse en la figura 3.17. El vector con mayor longitud define la orientación del punto de interés.

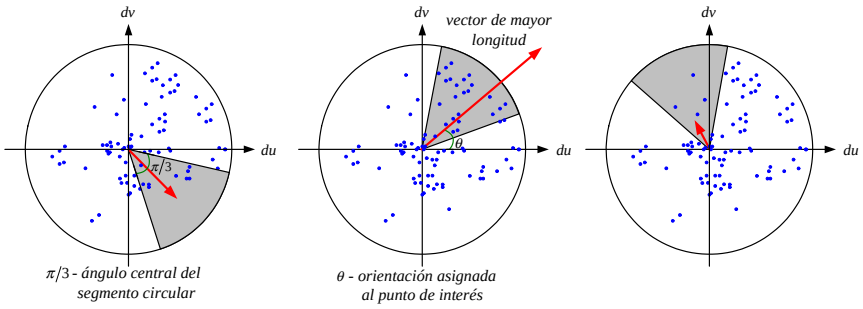


Figura 3.17: Asignación de la orientación.

Calculo del descriptor

Para la obtención del descriptor, primero se construye una ventana cuadrada de tamaño $20s$ alrededor del punto de interés. Esta ventana contiene los píxeles que se utilizarán para obtener las entradas del descriptor. Dicha ventana se orienta de acuerdo a la orientación asignada al punto de interés, tal como se muestra en la figura 3.18. Todos los cálculos subsecuentes son relativos a esta orientación, logrando así la invarianza a la rotación de la imagen.

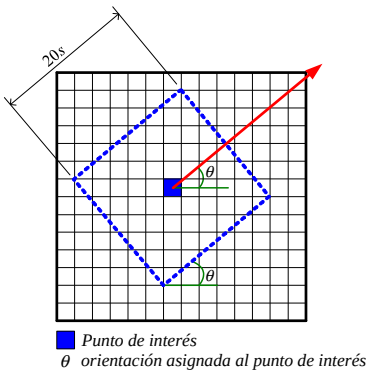


Figura 3.18: Ventana cuadrada con la misma orientación a la asignada al punto de interés.

La ventana del descriptor se divide en 4×4 regiones cuadradas, de donde se seleccionan 5×5 píxeles uniformemente distribuidos o espaciados dentro de cada una de ellas. Después se calculan las respuestas a los wavelets de Haar para este conjunto de píxeles con un tamaño igual a $2s$, siendo s la escala en donde se detecto la característica. Por simplificidad se denotan como d_u y d_v a las respuestas de los wavelets de Haar en las direcciones horizontal y vertical respectivamente, en donde estas direcciones se definen en función de la orientación asignada a cada punto de interés (ver figura 3.19).

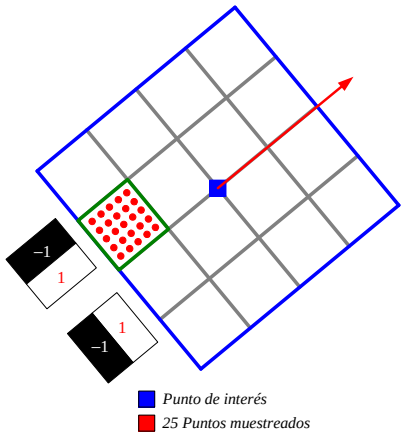


Figura 3.19: División de la ventana del descriptor y muestreo de puntos.

Para incrementar la robustez a las deformaciones geométricas y a los errores de localización, las respuestas d_u y d_v se ponderan con una Gaussiana centrada en el punto de interés con desviación estándar $\sigma = 3{,}3s$. Finalmente, las 25 respuestas d_u y d_v se suman para formar un primer sistema de entradas del descriptor (ver figura 3.20). Para obtener información acerca de la polaridad de los cambios de intensidad, también se calculan las sumas de los valores absolutos de las respuestas, $|d_u|$ y $|d_v|$. Por lo tanto, cada región genera un total de cuatro entradas del descriptor:

$$\mathbf{v}_{región} = \left[\sum_{i=1}^{25} d_{u_i} \quad \sum_{i=1}^{25} d_{v_i} \quad \sum_{i=1}^{25} |d_{u_i}| \quad \sum_{i=1}^{25} |d_{v_i}| \right] \tag{3.20}$$

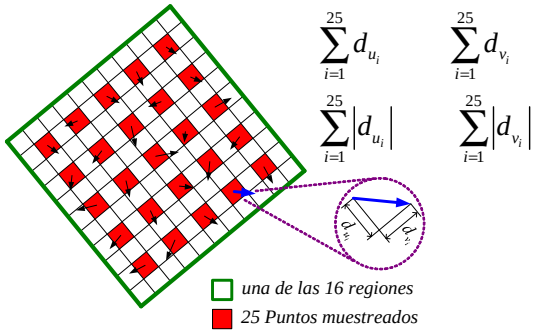


Figura 3.20: Calculo de las entradas del descriptor.

Las 16 regiones generan un total de 16×4 entradas que se agrupan en un vector $64 - dimensional$ conocido como descriptor *SURF*. Debido a que las respuestas a los wavelets de Haar son invariantes a los cambios de iluminación, entonces el descriptor también se vuelve invariante al brillo de la imagen. Luego, el descriptor se normaliza

a una magnitud unitaria logrando así la invarianza al contraste de la imagen.

Correspondencia de puntos

Para lograr la correspondencia de puntos entre dos imágenes utilizando los descriptores SURF se aplica el algoritmo del vecino más cercano a los descriptores de ambas imágenes y agregando una etapa de indexado en donde se considera el signo del Laplaciano (traza de la matriz Hessiana) de cada característica. El signo del Laplaciano distingue características brillantes sobre un fondo oscuro de la situación inversa, es decir, características oscuras sobre un fondo claro, de manera que sólo se comparan las características que tienen el mismo tipo de contraste tal como se observa en la figura 3.21. Esto permite una correspondencia de puntos más rápida sin reducir el desempeño del descriptor.

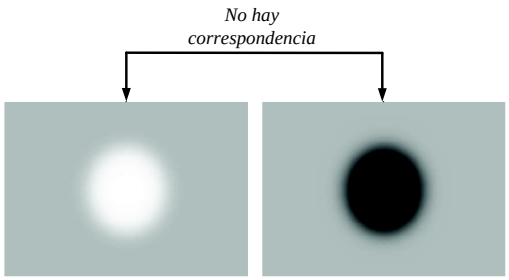


Figura 3.21: Correspondencia de puntos.

3.4. Reconstrucción 3D

El sistema de visión humano es capaz de percibir la profundidad principalmente porque poseemos visión binocular, la cual tiene lugar porque los dos ojos, separados unos centímetros, observan al mismo objeto desde perspectivas ligeramente distintas, obteniendo como resultado dos imágenes muy parecidas, pero no iguales [59]. A esta ligera diferencia entre los dos puntos de vista proporcionados por ambos ojos se le conoce como disparidad binocular. La disparidad binocular es la forma de percibir profundidad utilizada por el cerebro humano, en la cual el cerebro procesa ambos puntos de vista y por triangulación calcula la profundidad de los objetos observados [26], [91].

Basándose en este enfoque, considere el sistema de visión compuesto de "*dos cámaras idénticas*" colocadas de forma que sus ejes ópticos son paralelos entre sí, con

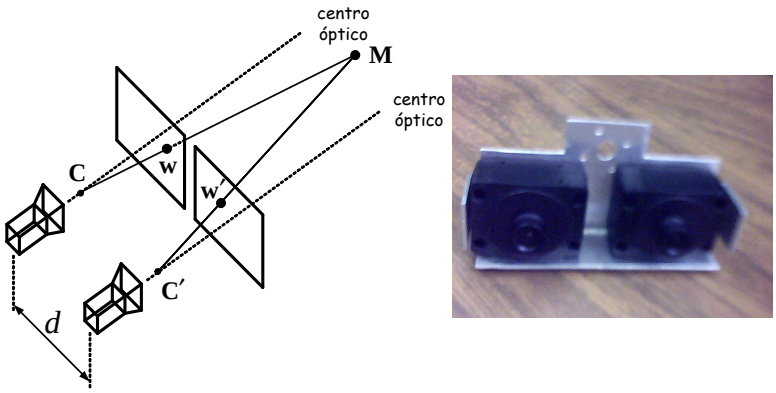


Figura 3.22: Sistema de visión binocular.

sus planos imagen alineados y a la misma altura, separados una distancia d tal como se observa en la figura 3.22.

Considerese ahora un punto **M** en la escena cuyas coordenadas son $[X, Y, Z]^T$ y $[X', Y', Z']^T$ en los referenciales de la primera y segunda cámara respectivamente, que se proyecta como $\mathbf{m} = [x, y]^T$ y $\mathbf{m'} = [x', y']^T$ en la primera y segunda imagen respectivamente. Del modelo de proyección en perspectiva dado por la ecuación (3.1) y de la figura 3.23 resulta:

$$\begin{aligned} x &= f\frac{l + \frac{d}{2}}{Z} \\ x' &= f\frac{l - \frac{d}{2}}{Z} \end{aligned} \tag{3.21}$$

Como ambos planos imagen se encuentran a la misma altura sólo existe disparidad en el plano horizontal tal como se observa en la figura 3.23. De acuerdo con la definición de disparidad, ésta será la diferencia entre las coordenadas imagen:

$$x - x' = f\frac{d}{Z} \tag{3.22}$$

Por lo tanto, la profundidad del punto **M** está dada por:

$$Z = f\frac{d}{x - x'} \tag{3.23}$$

En la imagen sólo se pueden medir las coordenadas $\mathbf{w} = [u, v]^T$ expresadas en el referencial del captador y en unidades de píxeles, de la figura 3.24 se observa que:

$$x = \frac{1}{m_u}(u - u_0) \tag{3.24}$$

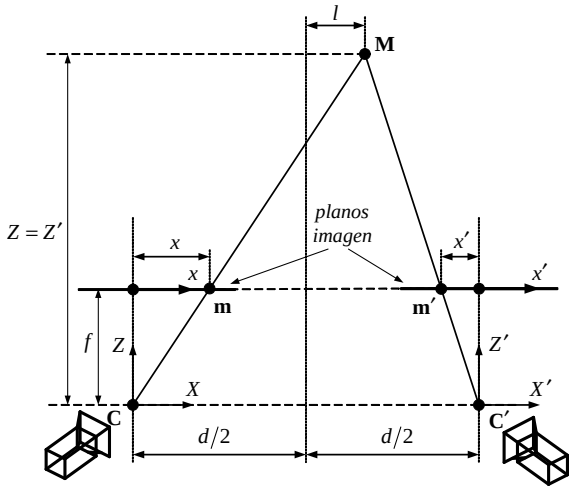


Figura 3.23: Relación entre la profundidad y la disparidad.

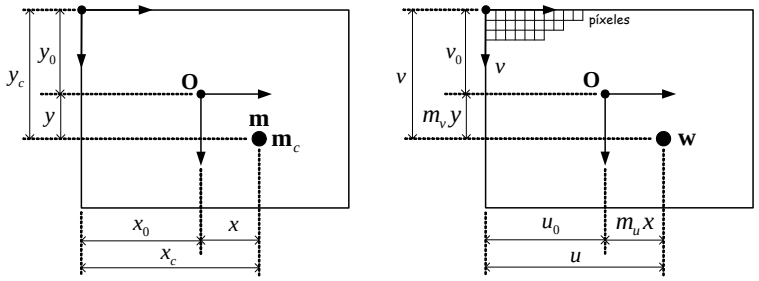


Figura 3.24: Referencial del captador.

Si las cámaras son idénticas $m_u = m'_u$:

$$x' = \frac{1}{m_u}\left(u' - u'_0\right) \tag{3.25}$$

Sustituyendo (3.24) y (3.25) en (3.23) se obtiene:

$$Z = \alpha_u \frac{d}{u - u' - (u_0 - u'_0)} \tag{3.26}$$

Una vez calculada Z, las coordenadas X y Y se calculan como:

$$X = Z\frac{u - u_0}{\alpha_u} \tag{3.27}$$

$$Y = Z\frac{v - v_0}{\alpha_v} \tag{3.28}$$

Note que, para realizar la reconstrucción de la escena es necesario conocer los parámetros α_u, α_v, u_0, u'_0 y d, que se obtienen en el proceso de calibración [91].

Capítulo 4

Generación de modelos 3D utilizando visión estereoscópica

Uno de los principales objetivos de la reconstrucción 3D es conseguir que una computadora llegue a analizar una escena real como lo haría una persona. Por otra parte, cuando se realiza la Reconstrucción 3D se obtiene un conjunto de puntos espaciados irregularmente sin un orden específico, los cuales por sí sólos no son muy representativos y en cierta forma son información en bruto. A este conjunto se le conoce como nube de puntos y representan el conjunto de puntos que ha detectado el dispositivo. Sin embargo, resulta conveniente utilizar esta información obtenida para construir un modelo digital tridimensional que pueda ser utilizado en una amplia variedad de aplicaciones. Una de las representaciones más utilizadas en los últimos años consiste en convertir esta nube de puntos en mallas de polígonos o en mallas triangulares irregulares. A esta nueva representación de la nube de puntos se le conoce como modelo 3D y se define como el proceso completo que comienza a partir de la adquisición de datos y termina con un modelo virtual en tres dimensiones que se pueden visulizar interactivamente en una computadora [1].

El modelado tridimensional tiene varias aplicaciones, por ejemplo en la generación de mapas del entorno para la navegación de robots móviles permite planificar los movimientos del robot sin necesidad de ayuda humana. En los procesos de producción se utiliza para determinar magnitudes como distancias, áreas o volúmenes de los objetos que componen la escena, lo cual se puede aplicar en los controles de calidad para verificar los procesos y superficies de los objetos que se estén fabricando. También ha sido ampliamente utilizado en la digitalización de museos o monumentos históricos para crear visitas virtuales, que consiste en generar una simulación tridimensional de los aspectos del mundo real y en donde el usuario tiene la sensación de pertenecer a

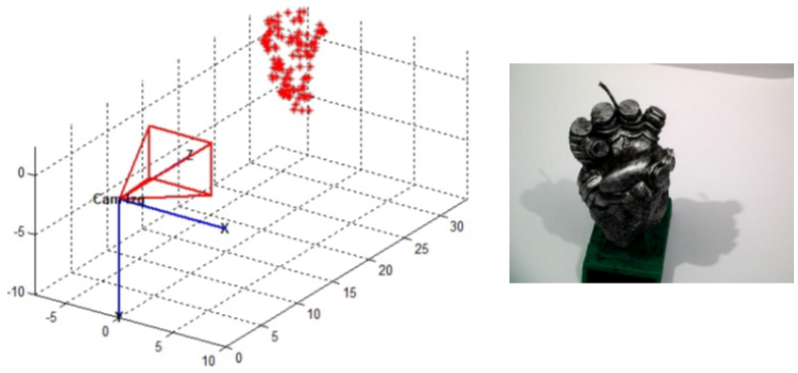


Figura 4.1: Nube de puntos.

ese ambiente ficticio e interactuar con él [10]. En el área de la ingeniería biomédica resulta ser una herramienta importante en la reconstrucción de estructuras anatómicas a partir de imágenes médicas (resonancias magnéticas), para efectuar un diagnostico médico, la planificación de terapias y procedimientos quirúrgicos.

Las principales ventajas de los modelos 3D digitales son:

- Debido a su almacenamieno digital, los modelos pueden ser complementados y reconstruidos en diferentes ambientes por medio de herramientas de importación y exportación de imágenes.
- Se pueden generar representaciones realistas del modelo utilizando algún tipo de animación interactiva y entornos de simulación tales como VRML, X3D, Meshlab, entre otros.
- Los modelos siempre pueden ser actualizados y reconstruidos individualmente o simplemente ser considerados como parte de un modelo más complejo.

La primera etapa de este proceso consiste en la obtención de la información tridimensional y generalmente se lleva a cabo mediante algún dispositivo como un escáner, con el cual se obtiene un conjunto de coordenadas 3D sobre la superficie de la escena observada, generando lo que se conoce como nube de puntos (ver figura4.1). Posteriormente, este conjunto de puntos se procesa en algún software obteniéndose un modelo 3D del objeto considerado. Esta representación se almacena en un archivo de computadora para ser utilizado después en alguna aplicación específica.

Algunas de las técnicas existentes enfocadas en extraer la información tridimensional de la escena como la telemetría láser, la luz estructurada o escáneres de tiempo

de vuelo permiten reproducir modelos muy exactos y precisos, pero con el inconveniente de emplear un equipo costoso y computacionalmente caros. Por el contrario, la visión estereoscópica representa un opción viable debido a su versatilidad, bajo costo computacional y económico, robustez y simplicidad de entendimiento debido a su similitud con el sistema de visión humano. En efecto, en un principio el estudio de la visión estereoscópica se centró en los seres humanos y en su capacidad de percepción del entorno a través de los ojos que, dada su posición, reciben imágenes sensiblemente diferentes de una misma escena. Estas diferencias relativas en la posición reciben el nombre de disparidades y guardan una relación directa con la profundidad a la que se encuentran los objetos respecto al observador, de manera que una vez captadas las imágenes a través de los ojos, el cerebro se encarga de de la reconstrucción tridimensional de la escena. A partir de la década de los 70, el estudio de la visión estereoscópica se extendió al campo de la visión artificial, donde ha sido un tema ampliamente estudiado ofreciendo considerables mejoras en tareas como la navegación de robots, la medicina, la topología, entre otras.

Ahora bien, independientemente de la técnica utilizada para la reconstrucción, el objetivo principal del modelado 3D es obtener un algoritmo que sea capaz de realizar la conexión del conjunto de puntos representativos del objeto en forma de elementos de superficie, de modo que la eficiencia de la técnica utilizada define la calidad final del modelo.

4.1. Solución propuesta

En este trabajo se presenta el desarrollo de un sistema de Reconstrucción 3D de objetos a partir de un conjunto de imágenes o vistas sucesivas de la misma escena capturadas desde diferentes puntos de vista, cada una de las cuales representa una parte del objeto. El sistema considera dos etapas principales, la primera realiza el procesamiento de imágenes siendo su objetivo determinar el mapa de disparidad para cada par de ellas, donde cada par estereoscópico sigue una secuencia de pasos como la detección de puntos de interés, la correspondencia de puntos y la reconstrucción de puntos. Esta secuencia se repite para todos los pares de vistas sucesivas del conjunto. La segunda etapa tiene como objetivo crear el modelo 3D del objeto a partir de determinar un nube de todos los puntos 3D generados. Cada iteración de la etapa anterior agrega nuevos puntos a la nube, de modo que una vez obtenida la nube total de puntos, la superficie del objeto se genera a partir de triángulos obtenidos con los puntos que conforman la malla y del método conocido como triangulación de Delaunay. Los puntos reconstruidos de cada par de imágenes se unen mediante puntos comunes

entre imágenes sucesivas. Los resultados obtenidos del proceso de reconstrucción son modelados en un ambiente virtual Meshlab a modo de obtener una visualización más realista del objeto manteniendo sus características físicas como dimensiones, volumen y forma.

4.2. Triangulación de Delaunay

El resultado de la etapa de reconstrucción 3D mediante la técnica de la obtención de la profundidad es un conjunto de puntos en $\mathbb{R}^3$ que no tienen conectividad alguna. Este conjunto, conocido como nube de puntos, es un conjunto de vértices en un referencial tridimensional que representan la superficie externa del objeto observado y se define como:

$$\mathbf{P}_i = \{\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_i\}$$

donde $\mathbf{P}_i$ es la nube de puntos e i es el número de puntos en la nube. A pesar de que $\mathbf{P}_i$ es una representación de la escena reconstruida, no es la mejor si es que se requiere mostrar en una computadora. Esto se debe entre otras cosas a que al aplicar una transformación como el escalamiento sobre $\mathbf{P}_i$ se obtendrá un conjunto de puntos muy separados entre sí que se asemejan poco a la escena que se quiere reconstruir por lo tanto, resulta conveniente obtener otra representación de la escena que se asemeje más a ella [6]. Una opción confiable es aquella denominada triangulación, que consiste en convertir un conjunto dado de puntos en un modelo poligonal consistente (mallas). Esta operación particiona los datos de entrada generando vértices, aristas y caras, dividiendo el conjunto de puntos en muchos elementos pequeños, por lo general triángulos o cuadriláteros en dos dimensiones y tetraedros en tres dimensiones.

Una triangulación óptima se define midiendo los ángulos, longitudes de los bordes, la altura o el área de los elementos; de modo que el error de aproximación de los elementos finitos generalmente está relacionado con alguna de estas medidas. La triangulación puede ser realizada en 2D o en 3D, de acuerdo con la geometría de los datos de entrada [7]. Por otro lado, resulta lógico pensar que la triangulación que forme los triángulos más regulares generará el modelo más fiel de la escena observada, lo cual puede obtenerse mediante la triangulación de Delaunay que tiene la característica de que cada uno de los triángulos que componen al modelo esta circunscrito en un círculo y cada círculo contiene unicamente un triángulo.

Se le denomina así en honor al matemático ruso Boris Nikolaevich Delone quien la inventó en 1934 y consiste en una red de triángulos que cumple la condición de Delaunay, la cual establece que la circunferencia circunscrita de cada triángulo de la

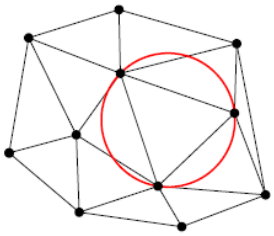


Figura 4.2: Triangulación de Delaunay.

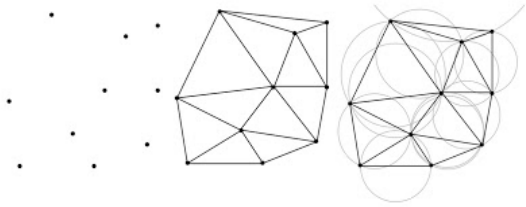


Figura 4.3: Condición de Delaunay.

red no debe contener ningún vértice de otro triángulo. La condición de Delaunay dice que una red de triángulos es una triangulación de Delaunay si todas las circunferencias circunscritas de todos los triángulos de la red son vacías (figura 4.2). Esta es la definición original para espacios bidimensionales, sin embargo es posible extenderla para espacios tridimensionales usando la esfera circunscrita en vez de la circunferencia circunscrita.

La triangulación de Delaunay utiliza redes de polígonos para modelar objetos tridimensionales, juntando los polígonos para imitar la superficie del objeto. Generalmente se usan triángulos porque son los polígonos más simples y tienen muchas propiedades favorables, como que representan una superficie coplanaria. Después de efectuar la Reconstrucción 3D del objeto se obtiene un conjunto discreto de puntos que para fines prácticos hay que transformar en una red de triángulos (figura 4.3). A esta tranformación se le conoce como triangulación. La triangulación de Delaunay maximiza los ángulos interiores de los triángulos, lo que resulta muy práctico porque al usar la triangulación como modelo tridimensional los errores de redondeo son mínimos, generando los triángulos más equiláteros posibles.

Las principales propiedas de esta triangulación son que forma la envolvente convexa del conjunto de puntos, el ángulo mínimo dentro de todos los triángulos está maximizado y la triangulación es unívoca si en ningún borde de circunferencia cir-

cunscrita hay más que tres vértices. Además, el algoritmo para obtener un modelo tridimensional a partir de una nube de puntos se puede llevar a cabo de al menos dos formas:

- **Algoritmo por incrementos.** En este algoritmo se van agregando puntos de manera gradual y se determina si se satisface el criterio de Delaunay. En caso de cumplirse la condición se agrega otro punto, de lo contrario se reagrupa la triangulación a modo de satisfacer dicho criterio, el proceso se repite agregando otro punto. Generalmente se ordenan los puntos con respecto a alguna coordenada, de modo que por cada punto que se agrega hay que reordenar la triangulación [1].
- **Algoritmos divide y vencerás.** La metodología utilizada por esta técnica consiste en dividir recursivamente la nube de puntos y después crear una triangulación en cada subdivisión, cumpliendo con el criterio de Delaunay. Finalmente se unen los resultados y se obtiene el modelo [1].

El algoritmo por incrementos se describe en los siguientes pasos:

1. Introducir la nube de puntos $\mathbf{P}$ de tamaño n en un gran triángulo conteniendo a dicha nube de puntos.
2. Para cada uno de lo n puntos de $\mathbf{P}$, elegir p_r de forma aleatoria y triangular:
   a) Encontrar el triángulo $p_i p_j p_k$ conteniendo a p_r.
   b) Si p_r cae en el interior del triángulo $p_i p_j p_k$, entonces se deben añadir aristas desde p_r hasta los vértices p_i, p_j, p_k.
      - Legalizar las aristas (p_r, $p_i p_j$)
      - Legalizar las aristas (p_r, $p_j p_k$)
      - Legalizar las aristas (p_r, $p_k p_i$)
   c) En caso contrario p_r cae en la arista $p_i p_j$, que es incidente a los triángulos $p_i p_j p_k$ y $p_i p_j p_l$.
      - Legalizar las aristas (p_r, $p_l p_j$)
      - Legalizar las aristas (p_r, $p_j p_k$)
      - Legalizar las aristas (p_r, $p_k p_i$)
3. Eliminar los tres puntos del gran triángulo inicial.

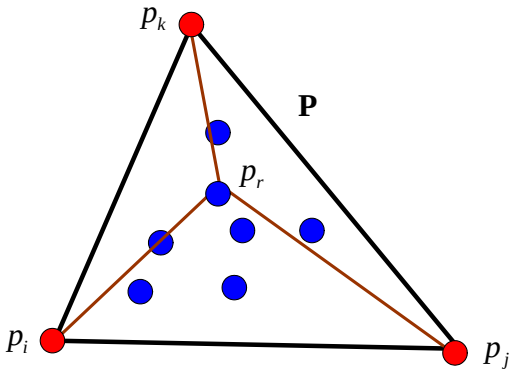


Figura 4.4: Triagulación de Delaunay.

En este contexto, se denomina arista legal, a toda arista de la triangulación que pertenece a dos triángulos tal que la circunferencia circunscrita a uno de los triángulos no contiene al punto restante que pertenece al otro triángulo. Una arista ilegal es aquella que pertenece a dos triángulos tales que forman un cuadrilátero convexo, sin embargo, si se intercambia dicha arista por la otra diagonal del cuadrilátero mejora el vector de ángulos generando una arista legal. A esta operación que consiste en sustituir una diagonal por la otra en un cuadrilátero se le denomina intercambio de aristas o flip.

4.3. Arquitectura del sistema de reconstrucción

El sistema de visión utilizado consta de componentes de bajo costo (ver figura 4.5) tales como:

1. Cámara RGB con resolución VGA (640×480 píxeles)
2. Arduino Uno
3. Mecanismo de pan (movimiento horizontal).

El par de imágenes se obtiene colocando la cámara sobre un mecanismo controlado mediante una tarjeta Arduino que genera un desplazamiento horizontal sobre dicho mecanismo. La primera imagen se captura en el posición inicial, mientras que la segunda imagen se obtiene después de efectuarse el desplazamiento del mecanismo, obtiéndose así un par de imágenes de la misma escena capturado desde diferentes puntos de vista. El sistema presentado es una variación del modelo tradicional pues en lugar de utilizar dos cámaras identicas separadas una cierta distacia y alineadas de

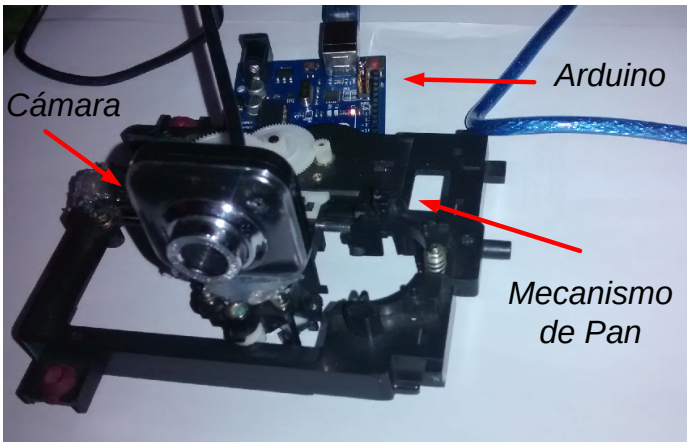


Figura 4.5: Sistema de visión estereoscópico.

modo que sus ejes ópticos sean paralelos, se utiliza una sola cámara que se desplaza horizontalmente a fin de obtener las dos imágenes que conforman el par estereoscópico. Con esta configuración el desplazamiento del centro óptico de la cámara es horizontal, lo que se traduce en el hecho de que las imágenes de una escena arbitraria sólo difieren en la componente horizontal.

Las entradas del sistema están compuestas por una secuencia de pares de imágenes I_1, I'_1, I_2, I'_2,..., I_n, I'_n las cuales son procesadas de vistas sucesivas y la matriz de calibración $\mathbf{K}$ pues se considera que el sistema se encuentra calibrado. El sistema está compuesto por dos módulos principales, el primero realiza el procesamiento de las dos imágenes, mientras que el segundo determina el modelo 3D incorporando los puntos de los pares de vistas que no han sido considerados (ver figura 4.6).

El primer módulo tiene como entrada un par de imágenes y la salida son los puntos 3D reconstruidos, generándose así una nube de puntos. El módulo realiza un proceso iterativo, donde primero detecta los puntos característicos de las imágenes utilizando para tal fin el detector SURF [6]. La salida de este proceso es el conjunto de puntos detectados, a partir de los cuales se realiza la correspondencia de puntos mediante los descriptores obtenidos con el mismo algoritmo. La determinación de la correspondencia esta dada por el algoritmo del vecino más cercano aplicado al conjunto de descriptores en ambas imágenes. Para evitar falsas correspondencias, se agrega una etapa de análisis de dichas correspondencias a partir de la matriz fundamental, esto es, cada correspondencia debe satisfacer la reestricción epipolar con el fin de evitar falsas correspondencias [40]. Posteriormente se aplica el algoritmo de disparidad al conjunto de correspondencias de modo que la salida de este primer módulo es un conjunto de

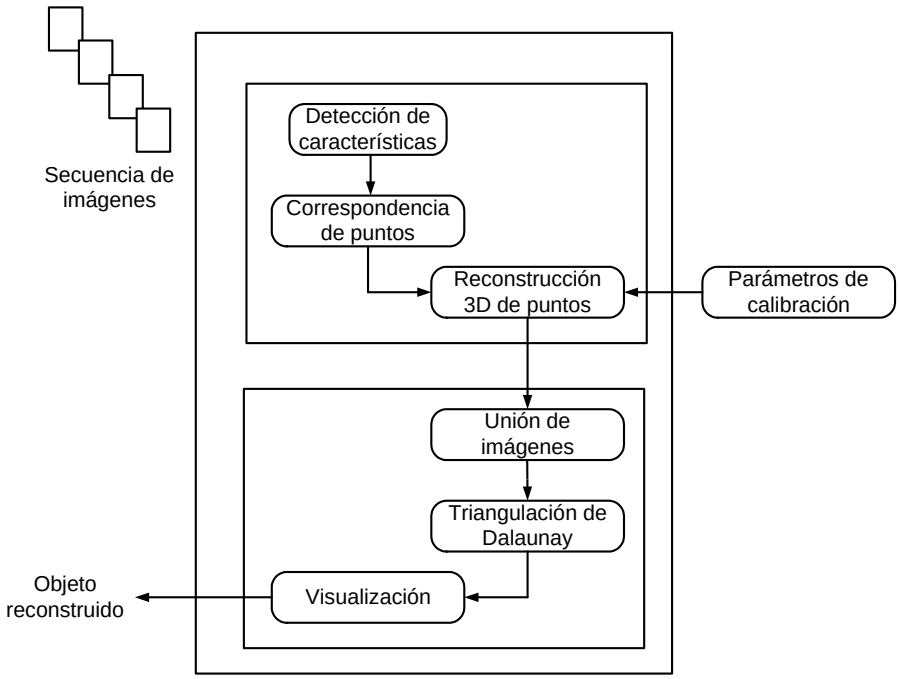


Figura 4.6: Arquitectura sistema de reconstrucción

puntos reconstruidos 3D a partir de dos imágenes sucesivas.

El segundo módulo toma la salida del anterior y realiza como primer proceso la unión de imágenes para formar un mapa de profundidad que represente al conjunto de vistas. Para cada par de imágenes existe un conjunto de puntos correspondiente en coordenadas de píxeles y un conjunto de puntos 3D reconstruidos que relaciona la imagen I_i con I_{i+1}. Para cada tres imágenes, se obtienen los puntos comunes de pares de vistas consecutivas, es decir, las imágenes I_i con I_{i+1}, las imágenes I_{i+1} con I_{i+2}, y sus correspondientes puntos 3D en el referencial de la cámara $k+1$, $\mathbf{P}^{k+1}_{(i+1)(i)}$ y $\mathbf{P}^{k+1}_{(i+1)(i+2)}$ respectivamente. El conjunto de puntos 3D de las imágenes sucesivas están conformados de la siguiente forma:

$$\mathbf{P}^{k+1}_{(i+1)(i+2)} = \mathbf{P}c^{k+1}_{(i+1)(i+2)} \cup \mathbf{P}n^{k+1}_{(i+1)(i+2)} \tag{4.1}$$

donde $\mathbf{P}c^{k+1}_{(i+1)(i+2)}$ es un subconjunto de puntos comunes entre las imágenes I_{i+1} y I_{i+2} que indica los puntos que aparecen en ambas imágenes en la etapa de detección, mientras que $\mathbf{P}n^{k+1}_{(i+1)(i+2)}$ representa el subconjunto de puntos no comunes de las mismas imágenes, los cuales representan los puntos que no aparecen en la vista I_{i+1}. El objetivo es determinar el conjunto de puntos correspondientes 3D del par de imágenes I_{i+1} e I_{i+2}, que serán agregados al mapa de puntos entre las vistas I_i e I_{i+1}. Las

coordenadas de los puntos que serán agregados en la referencia $k+1$ están dadas por:

$$\mathbf{P}a^{k+1}_{(i+1)(i)} = \mathbf{T'P}n^{k+1}_{(i+1)(i+2)}$$

Para determinar la transformación que relaciona un punto común en distintos referenciales se utiliza la siguiente ecuación:

$$\mathbf{P}^{k+1}_{(i+1)(i)} = \mathbf{T}\mathbf{P}^{k+1}_{(i+1)(i+2)} + \mathbf{e} \tag{4.2}$$

En (4.2) la matriz $\mathbf{T}$ representa la matriz de transformación homogénea entre un referencial y otro, mientras que el vector $\mathbf{e}$ es el error debido a la imprecisión de medida de los píxeles y de la estimación de los parámetros de movimiento $\mathbf{R}_i$ y $\mathbf{t}_i$. La representación matricial para la ecuación (4.2) está dada por:

$$\begin{bmatrix} X^{k+1}_{i+1i} \\ Y^{k+1}_{i+1i} \\ Z^{k+1}_{i+1i} \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} X^{k+1}_{i+1i+2} \\ Y^{k+1}_{i+1i+2} \\ Z^{k+1}_{i+1i+2} \\ 1 \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix} \tag{4.3}$$

Por lo tanto, se obtiene un sistema homogéneo de ecuaciones compuesto de 4 ecuaciones y 16 incógnitas, el cual puede escribirse como:

$$\begin{bmatrix} \left(X^{k+1}_{i+1i}\right)_1 \\ \left(Y^{k+1}_{i+1i}\right)_1 \\ \left(Z^{k+1}_{i+1i}\right)_1 \\ 1 \\ \vdots \\ \left(X^{k+1}_{i+1i}\right)_n \\ \left(Y^{k+1}_{i+1i}\right)_n \\ \left(Z^{k+1}_{i+1i}\right)_n \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & a_{21} & a_{22} & a_{23} & a_{24} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & a_{n4} & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & a_{n1} & a_{n2} & a_{n3} & a_{n4} \end{bmatrix} \begin{bmatrix} \left(X^{k+1}_{i+1i+2}\right)_1 \\ \left(Y^{k+1}_{i+1i+2}\right)_1 \\ \left(Z^{k+1}_{i+1i+2}\right)_1 \\ 1 \\ \vdots \\ \left(X^{k+1}_{i+1i+2}\right)_n \\ \left(Y^{k+1}_{i+1i+2}\right)_n \\ \left(Z^{k+1}_{i+1i+2}\right)_n \\ 1 \end{bmatrix} + \begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \\ 1 \\ \vdots \\ e_{n1} \\ e_{n2} \\ e_{n3} \\ 1 \end{bmatrix}$$

La ecuación anterior representa el sistema homogéneo para el conjunto de puntos comunes de la siguiente relación:

$$\mathbf{P}c^{k+1}_{(i+1)(i)} = \mathbf{T'P}c^{k+1}_{(i+1)(i+2)} + \mathbf{e} \tag{4.4}$$

donde $\mathbf{T'}$ es la extensión de la matriz $\mathbf{T}$ creada para ajustar la cantidad de variables y ecuaciones. Para cada punto común $\mathbf{P}c^{k+1}_{(i+1)(i)}$ y $\mathbf{P}c^{k+1}_{(i+1)(i+2)}$ se generan 4 ecuaciones,

una por cada fila de la matriz $\mathbf{T}$, por lo cual el sistema de ecuaciones aumenta 4 veces su tamaño por cada punto incorporado. En consecuencia el tamaño del sistema será $4n$, siendo n es el número de puntos comunes entre las imágenes, cuya solución nos permite determinar la matriz $\mathbf{T}$.

Para resolver el sistema son necesarios 4 puntos comunes, lo cual nos entrega una solución, pero esta no es buena debido a que no se estaría considerando la minimización del error, por lo tanto se utilizan todos los puntos comunes para el cálculo de la matriz $\mathbf{T}$, considerando el método de minimización de error *MSE* (Mean Square Error). La solución está dada por:

$$\mathbf{P}a^{k+1}_{(i+1)(i+2)} = inv(\mathbf{T'}^T\mathbf{T'})\mathbf{P}n^{k+1}_{(i+1)(i+2)} \tag{4.5}$$

Por lo tanto, los puntos de la imagen I_{i+2} son agregados al mapa de puntos de la imagen I_{i+1}, en el referencial $k+1$, por consiguiente:

$$\mathbf{P}^{k+1}_{(i)(i+2)} = \mathbf{P}^{k+1}_{(i+1)(i)}\mathbf{P}\mathbf{a}^{k+1}_{(i+1)(i+2)} \tag{4.6}$$

contiene los puntos de la imagen I_i e I_{i+1} y los puntos agregados de la imagen I_{i+1} e I_{i+2} en el referencial $k+1$. Una vez que se obtienen los puntos en el mismo referencial $k+1$, se aplican las transformaciones de movimiento para llevar los puntos al referencial k:

$$\mathbf{P}^{k}_{(i)(i+2)} = inv\left(\mathbf{R}_i\right)\left(\left(\mathbf{P}^{k+1}_{(i)(i+2)}\right) - \mathbf{t}_i\right) \tag{4.7}$$

donde $\mathbf{R}_i$ y $\mathbf{t}_i$ son los parámetros de movimiento entre las imágenes I_i e I_{i+1}.

El proceso se generaliza para las imágenes posteriores. Por otra parte, para formar la superficie del objeto reconstruido se consideran los puntos de la imagen I_i en el referencial k, es decir todos los puntos que son no comunes deben ser transformados al referencial k, así para todo el conjunto de imágenes se parte de las tres últimas hacia atrás determinando los puntos para llegar finalmente al referencial k.

La nube de puntos resultante del proceso de unión de las imágenes es triangulado para crear el modelo 3D del objeto. Para ello se utiliza la triangulación de Delaunay [1] con el fin de determinar la malla que será visualizada en un ambiente 3D. En esta etapa también se generan triángulos que no pertenecen al objeto, debido a lo cual se deben aplicar restricciones que permitan eliminarlos. La eliminación será por criterio de mayor distancia euclidiana entre los vértices de los triángulos, es decir, para cada triángulo de la malla se calcula la distancia entre sus vértices.

$$d = \sqrt{(\mathbf{v}_i - \mathbf{v}_j)^2}$$

con $i = 1, 2$ y $j = 2, 3$, $i \neq j$, $\mathbf{v}_i = [X, Y, Z]^T$. De este modo, se determina la máxima distancia de cada triángulo y se establece un valor de umbral para eliminar triángulos mayores respecto a la mayor distancia entre sus vértices. La disminución del umbral permite eliminar una mayor cantidad de triángulos, sin embargo se debe tener cuidado de no eliminar triángulos que formen parte de la superficie del objeto. Esto puede ocurrir cuando el conjunto de puntos que conforman la nube es escaso, debido a que en las regiones con pocos puntos los triángulos presentan un área mayor.

La visualización del objeto se realiza con el software Meshlab en un ambiente virtual que considera como entrada la nube total de puntos 3D, la malla triangular obtenida mediante la triangulación Delaunay y la información de color de los píxeles de las imágenes en formato RGB. Con esta información se genera el modelo tridimensional del objeto para su visualización en un ambiente virtual.

4.4. Resultados experimentales

Para validar el sistema se utilizaron imágenes con una resolución de 640×480 capturadas por una sóla cámara que se encuentra sobre una mecanismo que realiza un desplazamiento horizontal a modo que cada imagen tiene una ligera diferencia. Como se mencionó en la sección anterior, el propósito del sistema de reconstrucción propuesto consiste en crear una nube de puntos a partir de una secuencia de imágenes, la cual se utiliza para efectuar un proceso de triangulación a modo de generar el modelo tridimensional deseado. Para comprobar la robustez de la metodología propuesta se utilizaron varios pares de imágenes correspondientes a una escena estática.

En la figura 4.7 se muestra la interfaz gráfica que se desarrolló utilizando utilizando Java y Phyton, misma que permite controlar el desplazamiento del mecanismo y por consecuencia permite la captura de los pares de imágenes estereoscópicas. Para poder controlar el movimiento del mecanismo es necesario activar primero la tarjeta arduino, definiendo el número de puerto en el que se encuentra conectada, tal como se ilustra en la figura 4.7. Una vez seleccionado el puerto, se enviará un mensaje informando si se realizó correctamente, en caso contrario habrá que elegir algún otro puerto. La captura de los pares de imágenes se realiza aprovechando el desplazamiento del mecanismo sobre el cual se encuentra montada la cámara, de este modo en lugar del método tradicional de utilizar dos cámaras alineadas de forma que se sitúen ligeramente separadas en el espacio, se desplaza una única cámara una cierta distancia y se capturan las imágenes en las diferentes posiciones de movimiento. Por otra parte, la interfaz cuenta con una opción de segmentación la cual permite separar un objeto que sea de particular interés de los demás que se encuentren presentes en la escena.

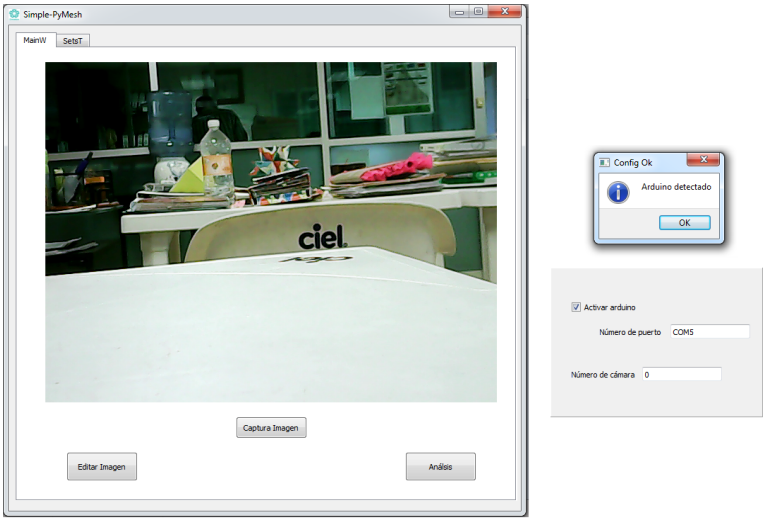


Figura 4.7: Interfaz sistema de visión.

De este modo, es posible efectuar el modelado tridimensional sólo de aquellos objetos que de algún modo resulten importantes, descartando todos los demás.

En la figura 4.8 se observa un par de imágenes del objeto que se desea reconstruir, las cuales fueron procesadas con un filtro Gaussiano con el fin de mejorar la calidad de las imágenes pues suaviza las imágenes y elimina el ruido presente en ellas.

Figura 4.8: Par de imágenes estereoscópicas.

La siguiente etapa consiste en la detección de características del objeto a reconstruir, para lo cual se utilizó el algoritmo SURF descrito en la sección 3.3 que es un detector de características computacionalmente rápido y que además es invariante a la escala y rotación de la imagen y parcialmente invariantes a distorsiones afines, cambios de iluminación, cambios en el punto de vista de la cámara, oclusión y ruido en la imagen; siendo esto los motivos por los cuales proporciona resultados bastante satisfactorios. Otra de las ventajas que presenta SURF es que además de la detección

de características, también transforma la imagen en una representación compuesta de vectores, denominados descriptores, los cuales contienen la información del gradiente de luminosidad en una vecindad alrededor de cada característica detectada. Estos descriptores también son invariantes a las transformaciones antes mencionadas lo que los hace altamente distintivos y muy útiles en tareas como la correspondencia de puntos. El principio es el siguiente, dado un punto característico en una imagen, su punto correspondiente en la otra imagen se encuentra aplicando el algoritmo del vecino más cercano a los descriptores de ambas imágenes descritos. En este enfoque, el vecino más cercano se define como el punto cuyo descriptor tiene la distancia Euclidiana mínima con respecto al descriptor de un punto en la otra imagen. En la figura 4.9 se muestran las correspondencias obtenidas para un par de imágenes estereoscópicas, en donde por simplicidad sólo se muestran algunas de ellas.

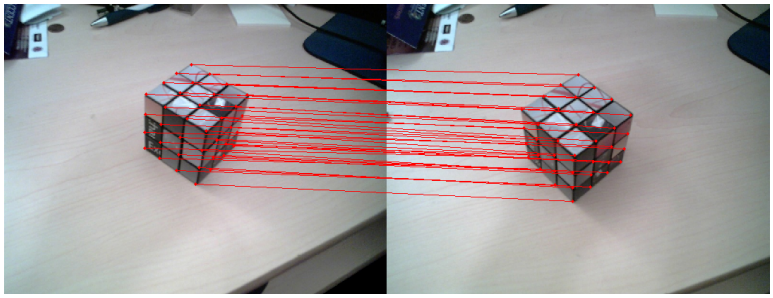


Figura 4.9: Sistema de correspondencias.

El algoritmo de reconstrucción estereoscópico implementado y descrito en 3.2 considera que el sistema de visión está calibrado, es decir, supone que se conocen los parámetros intrínsecos y extrínsecos de la cámara. Para lograr tal fin se utilizó la técnica de calibración propuesta en [9] que tiene como idea principal el uso de un patrón plano, el cual es observado desde distintos puntos de vista. De este modo, al tratarse de una escena plana, existe una matriz de homografía que mapea cada punto de la escena en un punto de la imagen. Esta matriz de homografía genera dos reestricciones que permiten estimar los parámetros de la cámara. En la figura 4.10 se observan las imágenes utilizadas en la etapa de calibración.

Las matriz de calibración obtenida para el sistema de visión es:

$$\mathbf{K} = \begin{bmatrix} 1108.29865 & 0.00000 & 326.02633 \\ 0.00000 & 1107.17951 & 147.73966 \\ 0.00000 & 0.00000 & 1.00000 \end{bmatrix}$$

Una vez obtenido el sistema de correspondencias, se aplica la técnica descrita en

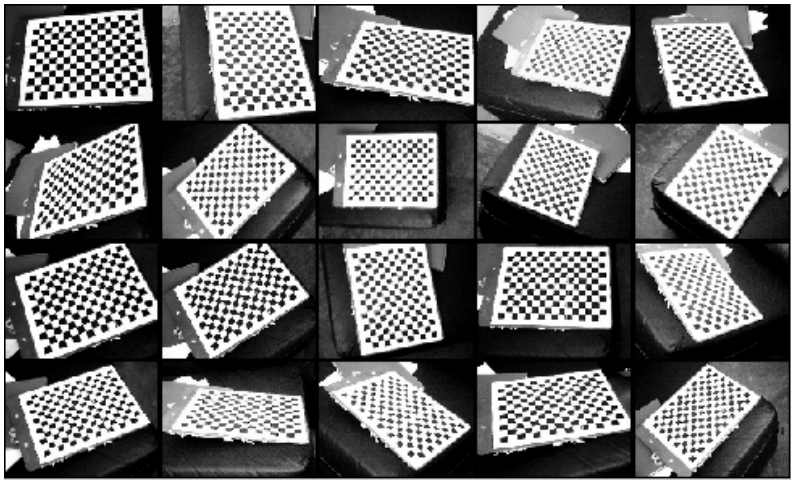


Figura 4.10: Imágenes de calibración.

X	Y	Z
−1699.945	−149.23	634.144
−961.34	−121.584	715.437
−467.112	57.752	763.852
−847.477	75.322	662.934
−724.265	319.322	680.662
−1602.462	−128.565	632.354
−225.055	316.094	1397.989
−97.48	319.552	828.577
−1702.887	−65.5	608.959
−902.554	305.589	672.929
−755.174	197.334	738.402

Tabla 4.1: Nube de puntos

la sección 3.4, generándose la nube de puntos mostrada en la tabla 4.1, en donde por simplicidad sólo se muestran algunos puntos.

En la figura 4.11 se muestra el mapa de disparidad obtenido a partir de un par de imágenes estereoscópicas que representa la diferencia de posición del objeto debido a los dos puntos de vista. Este mapa guarda una relación directa con la profundidad del objeto con respecto a la cámara, de manera que se puede conocer la profundidad a la que se encuentra a partir de obtener dicho mapa.

La siguiente etapa del algoritmo consiste en obtener las coordenadas tridimensionales de los puntos característicos detectados y puestos en correspondencia, obteniéndose así la nube de puntos del par de imágenes. Esta nube de puntos se puede editar y procesar a fin de obtener el modelo 3D de la escena observada, tal como se muestra en la

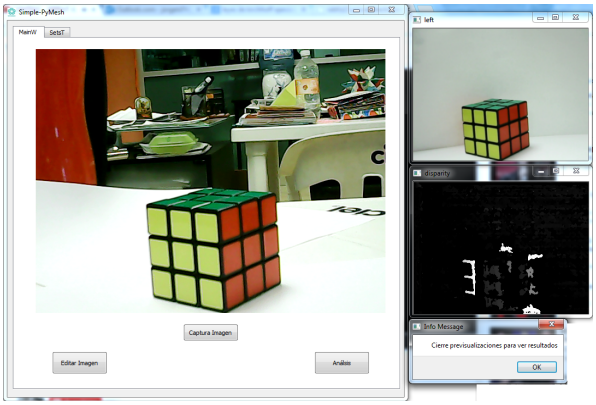


Figura 4.11: Mapa de disparidad.

figura 4.12. MeshLab es una herramienta utilizada ampliamente para generar, editar e incluso manipular modelos tridimensionales a partir de conocer las coordenadas de ciertos puntos que conforman la escena. A este conjunto de coordenadas también se le conoce como nube de puntos o PCD (Point Cloud Data) y a partir de ella se obtiene una malla poligonal y posteriormente se aplica un algoritmo adecuado para generar el modelo 3D de la escena. En general, el problema de encontrar un conjunto de polígonos a partir de un conjunto de puntos está muy restringido, es decir, en el mejor de los casos se considera una nube densa de puntos, en la cual todos los puntos pertenecen a superficies con poca curvatura y en donde las superficies están separadas por varias veces la distancia media entre ellos. En los demás casos, con la mayoría de los algoritmos, existirán errores tales como la generación de polígonos que no corresponden a ninguna característica física. De este modo, la elección del algoritmo utilizado dependerá de la calidad de la nube de puntos (ruido, densidad de puntos y uniformidad espacial) y de la tarea seleccionada. Aplicando la triangulación de Delaunay a la nube de puntos obtenida se genera el modelo 3D que se muestra en la figura 4.12.

Para unir los pares de imágenes y obtener la nube total de puntos es necesario empezar de la última imagen hacia la primera siguiendo el proceso especificado en la sección anterior. Note que, para efectuar la unión de imágenes es necesario expresar todos los puntos con respecto a un mismo referencial de manera que es necesario obtener el conjunto de matrices de transformación $\mathbf{T}_i$ entre un referencial y otro. En el proceso de calibración se pueden obtener tales matrices, las cuales se componen de una matriz de rotación $\mathbf{R}_i$ y de un vector de traslación $\mathbf{t}_i$ que representa la pose de la cámara con respecto al referencial asociado a la escena (ver figura 4.13) en el instante en que fue capturada. La técnica de calibración propuesta en [9] además de estimar la matriz de calibración $\mathbf{K}$, también calcula la matriz de transformación

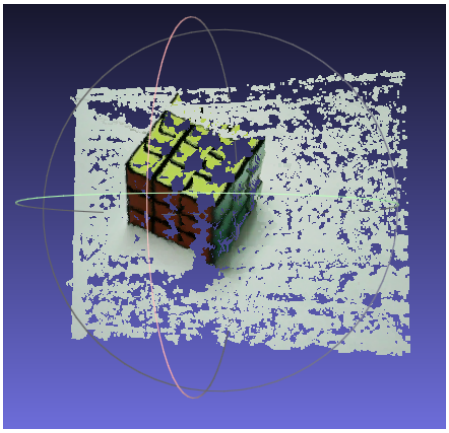


Figura 4.12: Obtención del modelo tridimensional.

para cada imagen, de modo que para obtener la pose relativa entre dos referenciales correspondientes a dos pares estereoscópicos de imágenes sólo se quiere hacer una composición de transformaciones homogéneas:

$$\mathbf{T}_0^2 = \mathbf{T}_0^1 \mathbf{T}_1^2$$

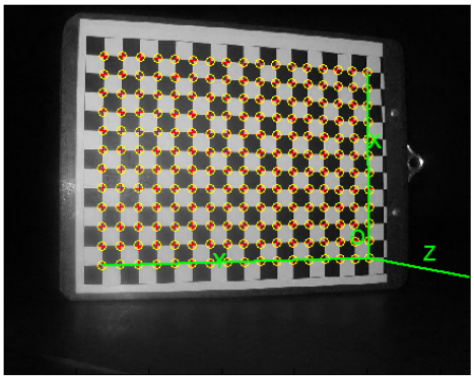


Figura 4.13: Referencial asociado a la escena.

Una vez unidas todas las imágenes es posible obtener la nube total de puntos (ver figura 4.14) que se genera como resultado del proceso de reconstrucción. Con esta nube de puntos puede obtenerse un modelo que se asemeje más al objeto real, sin embargo si se requiere mayor detalle es posible editar la nube de puntos para agregar características como color, textura e incluso efectuar un zoom para ampliarla o reducirla. Cabe mencionar que la nube de puntos se almacena como una matriz que

se compone de tres columnas, una para cada una de las coordenadas $[x, y, z]^T$, pero si se considera el color entonces deben agregarse tres columnas más correspondientes a las componentes *RGB*.

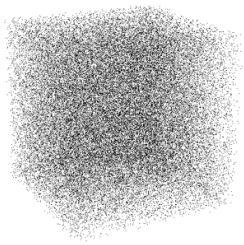


Figura 4.14: Nube total de puntos.

Una vez elegidas las propiedades de la nube de puntos, el siguiente paso consiste en la obtención de un modelo tridimensional del objeto observado, en donde su precisión depende de ciertos factores como la calidad de la nube, la densidad de los puntos (cantidad total de puntos que la conforman), homogeneidad de la nube en base a la geometría de la figura que representa, espaciado promedio entre puntos, entre otros. En la figura 4.15 se observa el modelo obtenido en el cual no se tiene en cuenta el color ni algún otro tipo de propiedades del objeto.

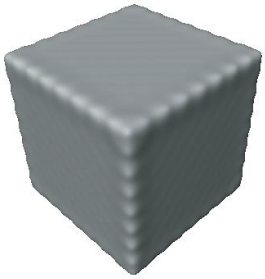


Figura 4.15: Modelo tridimensional

En la figura 4.16 se muestra el modelo tridimensional obtenido al unir todos los pares de imágenes estereoscópicas. Del lado izquierdo se observa el resultado del proceso de triangulación de Delaunay que convierte la nube de puntos en un modelo poligonal que particiona los datos de entrada en un conjunto de tetraedros que se encuentran sólo en caras compartidas (vértices, aristas o triángulos). Ahora bien, en la imagen derecha se agregaron detalles de color, rugosidad y textura a fin de obtener un modelo tridimensional más realista. Note que la mejora es considerable, por lo cual puede decirse que los resultados de la reconstrucción están influenciados directamente por los factores concernientes al tipo de objeto, forma, color, textura, propiedades

geométricas y luminosidad, lo cual se ve reflejado en el resultado final.

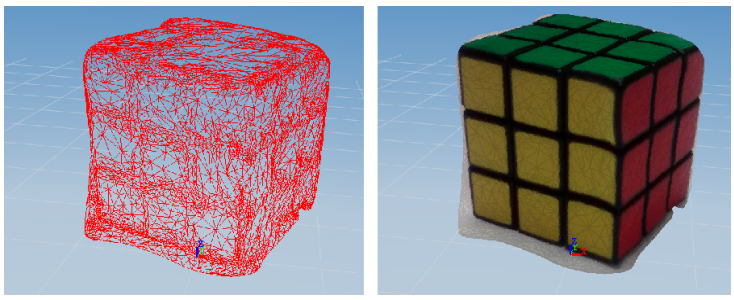


Figura 4.16: Modelo tridimensional.

4.5. Conclusiones

La cantidad de puntos característicos detectados influye directamente en la cantidad de puntos 3D reconstruidos y la calidad de la reconstrucción, es decir, mayores detalles del objeto. Por lo tanto, una mayor cantidad de puntos en el mapa de disparidad define áreas con más detalles. En algunos casos esto no se aplica debido a la geometría del objeto, por ejemplo en un cubo, una mayor cantidad de puntos puede resultar en un objeto distorsionado. En la técnica de reconstrucción propuesta se utiliza puramente un procesamiento basado en visión artificial, el cual restringe el número de correspondencias posibles con los puntos detectados. De este modo, no es conveniente comparar este enfoque de reconstrucción con otros propuestos en donde se utilizan distintos sensores para la captura de datos como sensores láser u ópticos.

La formación del mapa total de punto 3D no es homogénea, ocurriendo áreas con mayor o menor densidad de puntos. La variación se debe a mayor o menor contraste de la imagen, de manera que áreas con mayor cantidad de puntos presentan un mayor detalle del objeto. La mejora potencial a la reconstrucción se realiza necesariamente en las etapas de detección de características y de correspondencia de puntos.

Por otro lado, imágenes con poco contraste dificultan el procesamiento y la obtención de los puntos característicos, lo que influye fuertemente en la calidad de los resultados. Además, los errores observados en los resultados de la reconstrucción son reflejo de los errores en el proceso de calibración, en la etapa de correspondencia y en la estimación de los parámetros de movimiento de la cámara. La densidad de puntos de algunas regiones aumenta a medida que se agregan imágenes al proceso de reconstrucción.

Capítulo 5

Seguimiento visual para la generación de un mundo virtual

En esta sección se presenta el desarrollo de un sistema de seguimiento (tracking) que permite calcular la posición y orientación de las manos de un usuario en el espacio tridimensional así como el movimiento de cada uno de sus dedos. Esta información se utiliza para actualizar los parámetros de pose y movimiento de las manos y dedos del modelo tridimensional dentro de un mundo virtual. El movimiento de los dedos se detecta calculando el flujo óptico en imágenes capturadas por una cámara web instalada en el dorso de cada una de las manos del operador que registran la imagen de los hilos de nylon que transmiten el movimiento de flexión-extensión entre las falanges medial y proximal. Mientras que, el movimiento de las manos se detecta mediante un par de cámaras web colocadas en el espacio de trabajo del operador que permiten la reconstrucción 3D de ambas manos basada en la matriz de Homografía. El sistema de seguimiento visual desarrollado se propone como una alternativa de bajo costo y confiable para el desarrollo de aplicaciones de Realidad Virtual, en particular en los simuladores realistas utilizados en robótica médica.

Desde hace más de una veintena de años se dispone de técnicas de Computación Gráfica que permiten la creación de objetos tridimensionales que pueden ser mostrados en ambientes virtuales con mucho realismo en cuanto a sus propiedades reflectivas y a la interacción entre diferentes objetos en una misma escena a nivel de la iluminación. Estos modelos han sido muy utilizados en Sistemas de Dibujo, de Diseño y de Ingeniería Asistida por Computadora (CAd, CAD y CAE). Más que imágenes se trata de verdaderos mundos virtuales 3D desplegables a través de páginas Web con ayuda de navegadores (Browsers) que permiten navegar visualmente a su interior. A estas alternativas de visualización se les ha incorporado la facilidad para que el usuario

interaccione mecánicamente con los diferentes objetos presentes en este mundo virtual convirtiéndolo así en un MVI, lo que hace de la Computación Gráfica un inmejorable aliado de la Medicina, la arquitectura, la Educación y la Robótica, entre otras áreas de trabajo [92].

Actualmente, la interacción entre el usuario y los MVIs puede hacerse de múltiples maneras:

1. Utilizando las interfaces clásicas de la computadora (teclado, mouse, joystick).
2. A través de programas escritos en algún lenguaje de alto nivel (Java, Python, C++, Visual C++, etc.).
3. De manera bidireccional utilizando dispositivos hápticos (guantes, manipuladores, joysticks, etc.).

De esta manera el usuario puede lograr una inmersión total incluyendo la vista, el tacto y la cinestesia de varias partes de su cuerpo como la cabeza y, aún, de los miembros superiores e inferiores. Las características de interactividad del humano con la computadora a nivel de varios sentidos simultáneamente (tercera alternativa en el párrafo previo), son las que permiten denominar a una metodología de visualización gráfica como Realidad Virtual: "Virtual" porque existe sólo como modelos matemáticos al interior de una computadora y "Realidad" porque el usuario humano tiene acceso a esta información gráfica mediante varios sentidos a la vez, lo que permite producir una sensación de realidad captada por los sentidos (visual y táctil, preferentemente) generando una inmersión sensorial muy realista [44].

Los sistemas de captura de movimientos del cuerpo humano o de alguno de sus miembros tienen mucha importancia en aplicaciones de realidad virtual tanto de la industria del entretenimiento como profesionales. Son de sobra conocidos los sistemas como Kinect, Wii y otros que representan bien el primer tipo de aplicaciones. Mientras, las de tipo profesional se encuentran en una gran cantidad de ambientes, destacando las de la Robótica Médica utilizadas en simuladores y entrenadores de futuros cirujanos expertos en técnicas mínimamente invasivas (laparoscopia). En simuladores de la industria automotriz, principalmente en el ensamble virtual; en películas, para capturar movimientos de seres humanos y posteriormente generar animaciones con los movimientos capturados o bien esquemas de realidad aumentada.

Existen diversos tipos de sistemas de captura de movimiento: mecánicos, ópticos, magnéticos, y en el caso de los guantes también existen de variación de resistencias. El

problema de la adquisición de los dispositivos de captura de movimientos comerciales (Cyberglove^{TM} y otros) es el alto costo de los mismos, ya que pueden variar desde los 4,000 hasta los 200,000 dólares [74],[3]. Es en este contexto que proponemos una solución barata y sin contacto basada en visión para el seguimiento de los movimientos de las manos.

5.1. Solución propuesta

Uno de los temas de interés de este trabajo es la relación entre la Robótica con la Realidad Virtual, en particular en aplicaciones de Robótica Médica. Tal es el caso de los simuladores para el entrenamiento de cirujanos laparoscopistas y los sistemas de rehabilitación de pacientes con apoplejía, entre otras aplicaciones de interés. Es aquí donde aparece la necesidad de contar con un sistema de captura de la posición y orientación de las manos de un usuario, incluyendo el movimiento de sus dedos a fin de manipular un avatar dentro de un MVI. Si bien existen en el mercado sistemas eficientes para realizar la detección de pose en línea, estos son relativamente caros, por lo que se propone un sistema extremadamente simple y barato que, con la ayuda de técnicas de visión artificial (detección de movimiento y reconstrucción 3D), permiten tener una sistema de manipulación manual de MVIs barato, robusto y confiable con una interfaz ligera y cómoda. Además, al haber desarrollado tanto el hardware electrónico y mecánico del sistema así como la metodología (modelos y algoritmos) y la programación correspondiente, es posible hacer la comercialización del prototipo, la adaptación a diferentes tipos de problemas, la mejora del sistema, etc. Una de las primeras aplicaciones en que será utilizado el dispositivo desarrollado será la rehabilitación de pacientes con apoplejía. Para la captura del movimiento de flexión de los dedos se ideó un ingenioso sistema que transforma cada uno de estos movimientos en la traslación de un cable que es detectado mediante algoritmos de visión artificial que obtienen el flujo óptico [92]. Por otra, para hacer el seguimiento de una mano se propone utilizar un sistema de reconstrucción 3D monocular calibrado basado en la homografía entre un plano en la escena con marcas conocidas (dorso de la mano) y su imagen [59],[50].

5.2. Tracking del movimiento de los dedos

En el marco de este trabajo es necesario contar con la medición de la posición en cada instante de los cinco dedos de la mano. Considerando que cada uno de ellos consta de tres falanges (distal, medial y proximal) se tiene un total de 15 gdl a medir para determinar la flexión-extensión de los cinco dedos. Si, además, se quiere medir

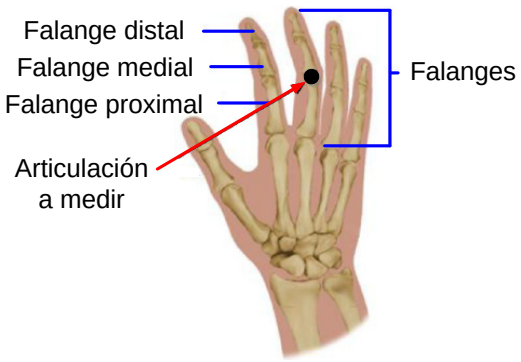


Figura 5.1: Esquema anatómico de la mano.

el movimiento lateral de cada dedo se necesitarían 5 gdl suplementarios. Sin embargo, para las aplicaciones contempladas en este trabajo basta con medir la flexión-extensión de una sola articulación por dedo, la que se encuentra conectando las falanges medial y proximal (5 gdl), tal como se indica en la figura 5.1.

5.3. Transmisión del movimiento de los dedos

La primera parte del sistema desarrollado consiste en la transmisión 5.6 del movimiento de flexión-extensión de la falange distal con respecto a la proximal. En una versión previa se efectuaba la medición del movimiento de la falange medial con respecto a la proximal, lo que complicaba el mecanismo [92]. Ambas soluciones son equivalentes y se inspiraron en un mecanismo muy parecido propuesto en [74]. En su versión actual el prototipo consiste en un guante sin dedos, como los que se utilizan para conducir, en el cual se fijan unos tubos de plástico que van desde la falange medial hasta una caja cuadrada ubicada en el dorso de la mano. Al interior de estos tubos corre un hilo de nylon (parecido a una fibra óptica) cuyo extremo distal está fijo a un anillo ubicado en la falange distal, mientras que su extremo proximal tiene una marca que permite seguir visualmente el desplazamiento de cada hilo al interior de la cajita. Cabe mencionar que, para minimizar la curvatura de estos tubos, los correspondientes a los dedos meñique y pulgar entran por la parte inferior de la cajita, con lo cual los hilos de nylon correspondientes se desplazan en sentido contrario a los otros tres (figura 5.2). Con este mecanismo tan simple como ingenioso se transforma la variable articular a medir (valor angular de la flexión-extensión entre las falanges medial y proximal) en un desplazamiento longitudinal de los hilos de nylon al interior de la cajita, el cual es monitoreado en línea por una cámara de video Creative^{TM}, modelo Creative WebCam Live Ultra (vf-0060), con resolución de 640x480 pixeles y una frecuencia de 30 cuadros

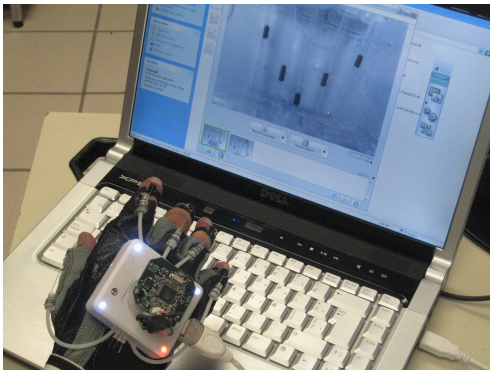


Figura 5.2: Aspecto del guante desarrollado.

por segundo. La cámara se monta sobre la cajita en el dorso del guante.

5.4. Medición del movimiento de los dedos

El movimiento longitudinal de los hilos de nylon produce un campo de movimiento unidimensional en la imagen, gracias a la marca que cada uno de ellos posee. Este campo de movimiento puede evaluarse mediante la medición del flujo óptico correspondiente. Decidimos emplear el algoritmo de Lucas-Kanade [4],[64] provisto en las bibliotecas de OpenCV, el cual permite la extracción de las dos componentes de la velocidad en el plano utilizando una función ventana para tomar los valores de los gradientes espaciales y temporal de una vecindad del punto estudiado, estimando el flujo óptico en cada uno de los bloques de pixeles previamente definido. Así, este algoritmo no necesita procesar toda la imagen capturada, sino solamente una vecindad de pixeles alrededor del punto a seguir. Esto permite un rápido procesamiento de la imagen, con lo cual se puede hacer el tracking del movimiento de los dedos en tiempo real.

La imagen capturada por la cámara presenta 5 marcas oscuras sobre fondo claro (incluyendo los hilos de nylon) cuyo desplazamiento longitudinal a lo largo del eje Y de la imagen representa la flexión-extensión de la articulación entre las falanges proximal y medial de cada dedo. Una vez capturada la imagen, es necesario especificar manualmente (por medio de un clic con el ratón) la ubicación de dichas marcas. Cada punto especificado en la imagen (cinco en total) representa las coordenadas imagen $[u_i, v_i]^T$ a seguir durante el experimento. Este valor de desplazamiento varía entre un valor mínimo y uno máximo propio del operador, los cuales se conocen de un proceso de calibración previo y corresponden a un valor mínimo y máximo del ángulo a medir, también conocidos [62]. Por lo cual basta con aplicar una interpolación lineal entre

dichos valores para calcular el ángulo deseado [92]. Para la captura de la imagen se utiliza la librería de visión por computadora OpenCV, mientras que el entorno de programación es Visual C++ 2008 Express Edition.

5.5. Tracking del movimiento de las manos

En este trabajo se propone un sistema de tracking visual de la mano que no es otra cosa más que un sistema de reconstrucción 3D monocular que debe calcular en cada instante la posición de la mano (3 gdl) y su orientación en el espacio (3 gdl más). Debido a que la mano del usuario puede estar realizando cualquier movimiento que demande la tarea de manipulación, ésta puede adquirir un número infinito de diferentes posturas, por lo que sería prácticamente imposible de reconocer. Para simplificar el problema y como de todos modos es necesario tener iluminación en el guante diseñado para la detección del movimiento de los dedos, se han colocado cuatro LEDs en el dorso de la mano formando un rectángulo de dimensiones conocidas, uno de los cuales es de un color diferente a los otros tres a fin de utilizarlo como referencia. Tratemos ahora de obtener una relación matemática entre la posición y orientación de la mano en el espacio con la posición en que aparecen estos cuatro LEDs dentro de la imagen.

De la ecuación 3.8 se tiene la siguiente relación matemática:

$$\lambda \tilde{\mathbf{w}}_i = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \tilde{\mathbf{M}}_i^w \tag{5.1}$$

Esta igualdad se establece salvo un factor de escala, lo que significa que todos los puntos en la escena ubicados sobre el rayo óptico que va de $\mathbf{M}_i^w$ al centro óptico de la cámara $\mathbf{C}$ proyectan el mismo punto en la imagen $\mathbf{w}_i$, esto imposibilita la reconstrucción 3D mediante una sola imagen. Sin embargo, para escenas 3D en donde es posible asignar un referencial de tal manera que todos los puntos de interés se ubiquen en un plano, tal que $\mathbf{M}_i^w = [X_i^w, Y_i^w, 0]^T$, en el modelo de proyección en perspectiva dado por la ecuación (5.1) la tercera columna de la matriz $\mathbf{R}$ es irrelevante, por lo que puede eliminarse obteniéndose la siguiente expresión:

$$\lambda \tilde{\mathbf{w}}_i = \mathbf{H} \tilde{\mathbf{M}}_i^w \tag{5.2}$$

donde $\mathbf{H} = \mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix}$ es la matriz que representa la homografía entre el plano escena en donde vive $\mathbf{M}_i^w$ y el plano imagen. Dicha homografía es una transformación lineal biunívoca ($\mathbf{H}$ siempre es no singular) sobre vectores homogéneos del espacio proyectivo que tiene 8 grados de libertad, pues se involucra en una igualdad salvo un factor de escala.

Si el referencial del mundo $O^wX^wY^wZ^w$ se ubica en el guante del usuario de modo que el origen sea el LED rojo y sus ejes X^w y Y^w se elijan como se muestra en la figura 5.2, la matriz de homografía $\mathbf{H}$ incluye tanto la matriz de calibración como la posición $\mathbf{t}$ y orientación $\mathbf{R}$ de la cámara con respecto a este referencial. Dada una imagen de la mano con sus cuatro LEDs es posible identificar la matriz de Homografía $\mathbf{H}$ correspondiente y, como la matriz de calibración $\mathbf{K}$ es conocida, basta con despejar el vector $\mathbf{t}$ y la matriz de rotación $\mathbf{R}$ para conocer la pose de la mano con respecto al referencial de la cámara [23],[50]. En un sistema calibrado ($\mathbf{K}$ conocida) la imagen (conjunto de puntos $\mathbf{w}_i$) de un patrón plano conocido (conjunto de puntos escena $\mathbf{M}_i$) permite la identificación de la matriz $\mathbf{H}$ asociada a la homografía si es que contamos con al menos 4 puntos $\mathbf{w}_i$ a fin de formar ocho ecuaciones lineales con 8 incógnitas (parámetros $\mathbf{h}_{ij}$) con base en la ecuación (5.2). Entre los principales métodos con que puede identificarse la matriz $\mathbf{H}$ se encuentran RANSAC y el algoritmo DLT siendo este último el método seleccionado y el cual se describe a continuación.

5.5.1. Algoritmo de transformación lineal directa

Este algoritmo, conocido por las siglas DLT (Direct Linear Transformation) permite la identificación de la matriz $\mathbf{H}$ a partir de un conjunto de n pares correspondientes $\mathbf{M}_i^w \leftrightarrow \mathbf{w}_i$, $i = 1, 2, \ldots, n$ [38]. Reescribiendo la ecuación (5.2) como un producto vectorial se obtiene la siguiente ecuación que relaciona al par correspondiente ($\mathbf{M}_i^w \leftrightarrow \mathbf{w}_i$) con los tres vectores fila $\mathbf{h}_j$ de la matriz de homografía:

$$\begin{bmatrix} \mathbf{0}^T & -\tilde{\mathbf{M}}_i^{wT} & v_i\tilde{\mathbf{M}}_i^{wT} \\ \tilde{\mathbf{M}}_i^{wT} & \mathbf{0}^T & -u_i\tilde{\mathbf{M}}_i^{wT} \end{bmatrix} \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \mathbf{h}_3^T \end{bmatrix} = \mathbf{0} \tag{5.3}$$

Así, cada par de puntos correspondientes ($\mathbf{M}_i^w \leftrightarrow \mathbf{w}_i$) genera dos ecuaciones linealmente independientes, por lo que

serán necesarios 4 pares de puntos correspondientes, pero que cumplan con la restricción de no tener tres o más puntos colineales en la escena. Para cumplir las restricciones del método de identificación de la matriz $\mathbf{H}$, los cuatro LEDs que se pondrán en el dispositivo diseñado formarán un rectángulo y uno de ellos, el origen del referencial, será de color diferente a los otros tres para garantizar, además, que la correspondencia con sus respectivas imágenes será sencilla de implementar.

La ecuación (5.3) es de la forma $\mathbf{A}_i\mathbf{h} = \mathbf{0}$ ($\mathbf{A} \in \mathbb{R}^{2\times 9}$, $\mathbf{h} \in \mathbb{R}^{9\times 1}$) y se obtiene a partir de un par correspondiente, de modo que utilizando 4 correspondencias se genera un sistema de ecuaciones $\mathbf{Ah} = \mathbf{0}$ en donde $\mathbf{A}$ se construye con las cuatro matrices $\mathbf{A}_i$ generadas por los cuatro pares correspondientes considerados. La presencia de

perturbaciones (ruidos, sombras y otros problemas de iluminación) sobre los puntos imagen $\mathbf{w}_i$ impide la existencia de una solución exacta $\mathbf{h}$, que no sea la trivial, de manera que resulta conveniente tratar de de obtener una solución aproximada que minimice la norma $\|\mathbf{Ah}\|$ sujeta a la restricción $\|\mathbf{h}\| = 1$ que evita la solución trivial.

5.5.2. Estimación por Máxima Verosimilitud

Asumiendo que las perturbaciones mencionadas son gaussianas con media cero y matriz de covariancia $\mathbf{\Lambda}_{\mathbf{w}_i}$, se puede estimar $\mathbf{H}$ mediante el criterio de máxima verosimilitud minimizando la función [105]:

$$\sum_i \left(\mathbf{w}_i - \hat{\mathbf{w}}_i\right)^T \mathbf{\Lambda}_{\mathbf{w}_i}^{-1} \left(\mathbf{w}_i - \hat{\mathbf{w}}_i\right) \tag{5.4}$$

donde $\hat{\mathbf{w}}_i = \frac{1}{\mathbf{h}_3 \tilde{\mathbf{M}}_i^w} \begin{bmatrix} \mathbf{h}_1 \tilde{\mathbf{M}}_i^w \\ \mathbf{h}_2 \tilde{\mathbf{M}}_i^w \end{bmatrix}$.

En la práctica, se asume que $\mathbf{\Lambda}_{\mathbf{w}_i} = \sigma^2 \mathbf{I}\ \forall i$, lo cual suena razonable considerando que los puntos se extraen de manera independiente con el mismo procedimiento, entonces el problema se convierte en un problema no lineal de mínimos cuadrados donde la función de costo a minimizar es:

$$\sum_i d\left(\mathbf{w}_i - \hat{\mathbf{w}}_i\right)^2 = \sum_i \left\|\mathbf{w}_i - \hat{\mathbf{w}}_i\right\|^2 \tag{5.5}$$

Esta minimización no lineal se obtiene mediante el algoritmo iterativo de Levenberg-Marquardt, el cual requiere valores iniciales que pueden obtenerse del algoritmo DLT mostrado anteriormente [27].

5.5.3. Reconstrucción 3D

De la ecuación (5.2) resulta $\mathbf{H} = \mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix}$, entonces expresando la matriz de homografía $\mathbf{H}$ por sus vectores columnas:

$$\begin{bmatrix} \mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3 \end{bmatrix} = \lambda' \mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \tag{5.6}$$

En donde λ' es un escalar arbitrario. Suponiendo que el sistema de visión está calibrado ($\mathbf{K}$ conocida) y que ya se ha identificado la Homografía $\mathbf{H}$, esta información puede usarse para obtener la pose del objeto de interés, es decir su posición $\mathbf{t}$ y su orientación $\mathbf{R} = \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 \end{bmatrix}$. Si bien sólo hay manera de calcular los vectores $\mathbf{r}_1$ y $\mathbf{r}_2$, el vector $\mathbf{r}_3$ puede calcularse como el producto cruz de los dos primeros pues forman una matriz ortonormal. Por lo tanto, la matriz de rotación $\mathbf{R}$ (orientación) se

obtiene como:

$$\begin{aligned} \mathbf{r}_1 &= \lambda' \mathbf{K}^{-1}\mathbf{h}_1 \\ \mathbf{r}_2 &= \lambda' \mathbf{K}^{-1}\mathbf{h}_2 \\ \mathbf{r}_3 &= \mathbf{r}_1 \times \mathbf{r}_2 \end{aligned} \tag{5.7}$$

Mientras que el vector de posición $\mathbf{t}$ y el factor de escala λ' se calculan como sigue:

$$\begin{aligned} \mathbf{t} &= \lambda' \mathbf{K}^{-1}\mathbf{h}_3 \\ \lambda' &= \frac{1}{\|\mathbf{K}^{-1}\mathbf{h}_1\|} = \frac{1}{\|\mathbf{K}^{-1}\mathbf{h}_2\|} \end{aligned} \tag{5.8}$$

Ahora bien, debido a la presencia de ruido en las coordenadas imagen, en general $\mathbf{R}$ no satisface las propiedades de una matriz de rotación por lo que es necesario estimar una matriz de rotación que se aproxime a ella [105]. La aproximación es en el sentido de la norma de Frobenius, esto es, la matriz de rotación que mejor se aproxime será aquella que minimice la norma de la diferencia de estas dos matrices:

$$\min_{\hat{\mathbf{R}}} \left\| \hat{\mathbf{R}} - \mathbf{R} \right\|^2 \qquad \text{sujeto a } \hat{\mathbf{R}}^T\hat{\mathbf{R}} = \mathbf{I} \tag{5.9}$$

Dado que:

$$\begin{aligned} \left\| \hat{\mathbf{R}} - \mathbf{R} \right\|^2 &= \text{traza}\left[\left(\hat{\mathbf{R}}^T - \mathbf{R} \right)^T \left(\hat{\mathbf{R}}^T - \mathbf{R} \right) \right] \\ &= 3 + \text{traza}\left(\mathbf{R}^T\mathbf{R} \right) - 2\,\text{traza}\left(\hat{\mathbf{R}}^T\mathbf{R} \right) \end{aligned} \tag{5.10}$$

De esta manera, resolver la ecuación (5.9) equivale a maximizar $\hat{\mathbf{R}}^T\mathbf{R}$. Sea, ahora, $\mathbf{USV}^T$ la descomposición en valores singulares de $\mathbf{R}$, en donde $\mathbf{S} = \text{diag}\left(\sigma_1, \sigma_2, \sigma_3\right)$. Si se define una matriz ortogonal $\mathbf{Z} = \mathbf{V}^T\hat{\mathbf{R}}^T\mathbf{T}$, entonces:

$$\begin{aligned} \text{traza}\left(\hat{\mathbf{R}}^T\mathbf{R} \right) &= \text{traza}\left(\hat{\mathbf{R}}^T\mathbf{USV}^T \right) = \text{traza}\left(\mathbf{V}^T\hat{\mathbf{R}}^T\mathbf{US} \right) \\ &= \text{traza}\left(\mathbf{ZS} \right) = \sum_{i=1}^{3} \mathbf{z}_{ii}\sigma_i \leq \sum_{i=1}^{3} \sigma_i \end{aligned} \tag{5.11}$$

Se observa que el máximo se alcanza fijando $\hat{\mathbf{R}} = \mathbf{UV}^T$ dado que $\mathbf{Z} = \mathbf{I}$, generándose así la solución a la ecuación (5.9).

5.6. Resultados Experimentales

A continuación se reportan los resultados experimentales obtenidos al evaluar los dos sistemas visuales de medición implementados: movimiento de dedos usando flujo

Figura 5.3: Mundo virtual interactivo desarrollado para visualizar una mano.

óptico y posición y orientación de la mano dentro del espacio de trabajo. Para visualizar los movimientos de las manos y dedos del usuario cuando utiliza nuestro sistema de seguimiento visual, se desarrolló un ambiente virtual utilizando las librerías de OpenGL en el entorno de programación Visual C++ 2008 Express Edition. El modelo virtual de la mano fue desarrollado en Blender y su animación se hace mediante interpolación de valores preestablecidos utilizando los datos adquiridos por el sistema de seguimiento visual de dedos y manos (figura 5.3).

Con el fin de validar la propuesta, se realizó un estudio experimental para conocer la frecuencia de lectura de datos y la resolución mínima de dichas lecturas. Así, para obtener la frecuencia de lectura del movimiento de los dedos (flujo óptico), cada vez que el sistema informático realiza una lectura de datos y los procesa, se incrementar un contador y se calcula el tiempo transcurrido en dicha operación. Se realizaron 1000 lecturas con un tiempo total de 29,99 segundos, de modo que, en promedio, cada lectura se hace en 29,99 ms. Con lo cual el tiempo de procesamiento permite aplicaciones en tiempo real con una cadencia imagen de 30 fps. Esto se ilustra en la figura 5.6. Para obtener la frecuencia de cálculo de la posición y orientación de la mano se realizó un experimento de manipulación de un objeto virtual con una sola mano y se hicieron cálculos similares a los descritos en el párrafo previo obteniéndose un tiempo promedio para 1000 lecturas de 34,19 ms. Con lo cual estamos en medida de calcular el movimiento de la mano y dedos a razón de 15 imágenes por segundo. Para conocer la resolución mínima angular que tiene el sistema de seguimiento del movimiento de los dedos se diseñó un experimento que consiste en hacer movimientos aleatorios de los dedos y hacer calcular el incremento de posición entre mediciones consecutivas para cada dedo, guardando en memoria la diferencia actual siempre y cuando sea menor a la registrada anteriormente. Así, al final del experimento se tiene en dicho registro la mínima diferencia calculada durante el experimento, la cual fue de 0,0084°. Esto representa una resolución muy grande en la medición de la posición angular de los

dedos.

A continuación se reportan los resultados experimentales hechos para comprobar la validez de la metodología de reconstrucción 3D basada en homografías utilizada para la medición de la pose de la mano. Para la etapa de calibración se utiliza el Toolbox para Matlab desarrollado en CalTech [9], el cual se basa en la observación de un damero. De esta manera se obtuvieron los parámetros intrínsecos de la cámara representados por la matriz $\mathbf{K}$ y los parámetros extrínsecos $\mathbf{R}_C$ y $\mathbf{t}_C$, que representan respectivamente la orientación y la posición del referencial del guante con respecto al referencial de la cámara:

$$\mathbf{K} = \begin{bmatrix} 651.43268 & 0.00000 & 337.44965 \\ 0.00000 & 650.26989 & 224.91824 \\ 0.00000 & 0.00000 & 1.00000 \end{bmatrix}$$

$$\mathbf{R}_C = \begin{bmatrix} -0.969552 & -0.238094 & 0.057280 \\ -0.185709 & 0.562389 & -0.805748 \\ 0.159630 & -0.791852 & -0.589481 \end{bmatrix} \qquad \mathbf{t}_C = \begin{bmatrix} 156.336617 \\ -53.487722 \\ 517.255786 \end{bmatrix}$$

La figura 5.4 ilustra el proceso de calibración así como los tres referenciales de interés (base, guante y cámara). Sobre cada una de las imágenes del guante capturadas con el sistema de visión, se comienza por hacer una binarización adaptable a fin de obtener una imagen segmentada en donde aparecen solamente las regiones correspondientes a los cuatro LEDs del guante tal como se ilustra en la figura (ver figura 5.5). Una de las regiones obtenidas en esta etapa de binarización, se identifica con la etiqueta rojo, siendo precisamente su centroide el origen del referencial asociado al guante (ver figura 5.6), con sus ejes X y Y apuntando hacia las esquinas del cuadrado detectado, de modo que todos los puntos tendrán su coordenada Z igual a cero, tal como se muestra en la figura 5.7. De este modo, en el referencial asociado al guante, los LEDs

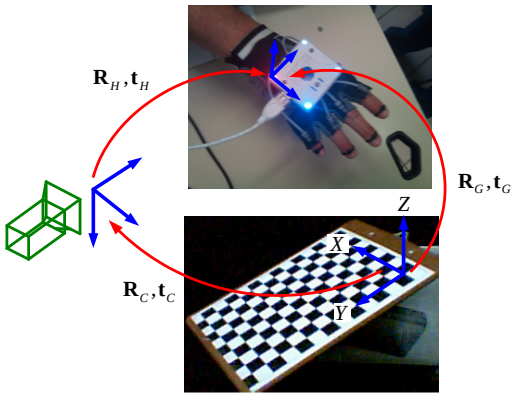


Figura 5.4: Esquema de calibración y relación entre referenciales.

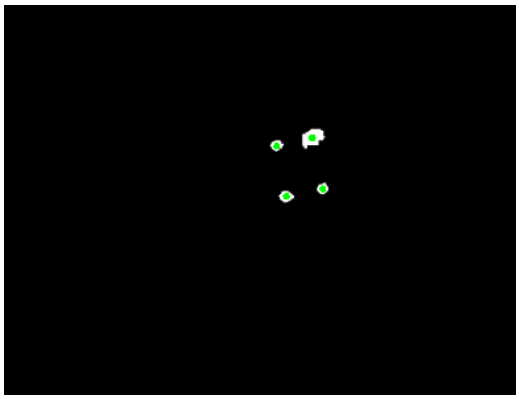


Figura 5.5: Binarización y cálculo de centroides.

tienen las siguientes coordenadas:

$$\mathbf{M}_1^G = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \qquad \mathbf{M}_2^G = \begin{bmatrix} 0 & l_1 & 0 \end{bmatrix}^T$$
$$\mathbf{M}_3^G = \begin{bmatrix} l_2 & l_1 & 0 \end{bmatrix}^T \qquad \mathbf{M}_4^G = \begin{bmatrix} l_2 & 0 & 0 \end{bmatrix}^T$$

En donde l_1 y l_2 son las dimensiones del rectángulo en cuyas esquinas se encuentran los LEDs tal como se ilustra en la figura 5.8. Así, con las coordenadas imagen de dichos puntos, se establece la siguiente correspondencia para el caso aquí considerado (ver

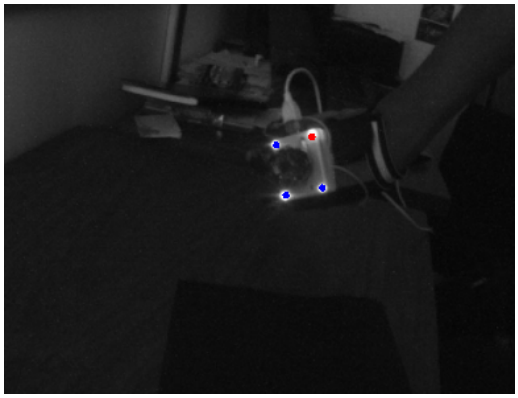


Figura 5.6: Etiquetación de los 4 LEDs.

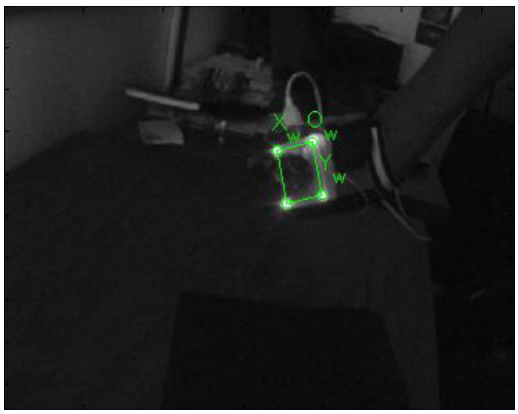


Figura 5.7: Referencial asociado al guante.

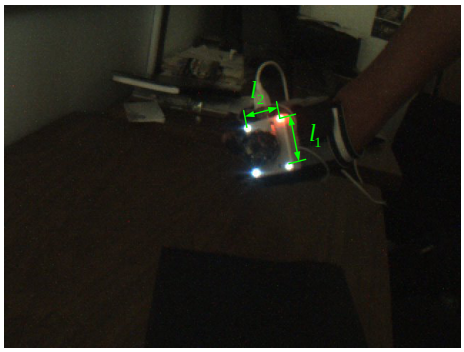


Figura 5.8: Rectángulo de tamaño conocido con LEDs en sus esquinas.

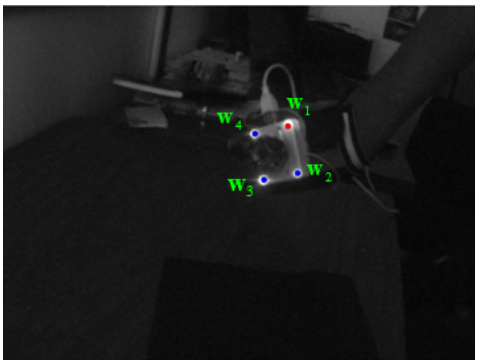


Figura 5.9: Coordenadas imagen de cada Led.

figura 5.9):

$$\begin{aligned}
\mathbf{w}_1 &= \begin{bmatrix} 387.1619 & 163.2451 \end{bmatrix}^T \quad \leftrightarrow \quad \mathbf{M}_1^G = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \\
\mathbf{w}_2 &= \begin{bmatrix} 342.9474 & 172.7566 \end{bmatrix}^T \quad \leftrightarrow \quad \mathbf{M}_2^G = \begin{bmatrix} 0 & l_1 & 0 \end{bmatrix}^T \\
\mathbf{w}_3 &= \begin{bmatrix} 354.6000 & 235.4270 \end{bmatrix}^T \quad \leftrightarrow \quad \mathbf{M}_3^G = \begin{bmatrix} l_2 & l_1 & 0 \end{bmatrix}^T \\
\mathbf{w}_4 &= \begin{bmatrix} 399.5874 & 225.7832 \end{bmatrix}^T \quad \leftrightarrow \quad \mathbf{M}_4^G = \begin{bmatrix} l_2 & 0 & 0 \end{bmatrix}^T
\end{aligned}$$

Aplicando el algoritmo DLT antes descrito se obtiene la siguiente matriz de homografía:

$$\mathbf{H} = \begin{bmatrix} -710.0000 & 140.0000 & 222320.0000 \\ 0.0000 & 650.0000 & 93840.0000 \\ 0.0000 & 0.0000 & 570.0000 \end{bmatrix}$$

A partir de esta Homografía y usando los parámetros intrínsecos $\mathbf{K}$ de la cámara, se obtuvo la matriz de rotación $\mathbf{R}_H$ y el vector de translación $\mathbf{t}_H$, que representan

respectivamente la orientación y posición del referencial del guante con respecto al referencial de la cámara:

$$\mathbf{R}_H = \begin{bmatrix} -0.8033 & 0.1902 & 0.5644 \\ 0.1895 & 0.9800 & -0.0606 \\ -0.5647 & 0.0583 & -0.8233 \end{bmatrix} \qquad \mathbf{t}_H = \begin{bmatrix} 44.4081 \\ -53.9149 \\ 573.0840 \end{bmatrix}$$

Con esta información se construye la matriz de transformación homogénea $\mathbf{T}_C^G$ que permite calcular la posición en el espacio de cada LED en el referencial de la cámara mediante la expresión:

$$\mathbf{M}_i^C = \mathbf{T}_C^G \mathbf{M}_i^G$$

Obteniéndose que las posiciones de los LEDs, expresadas en milímetros, con respecto al referencial de la cámara están dadas por:

$$\begin{aligned} \mathbf{M}_1^C &= \begin{bmatrix} 44.4081 & -53.9149 & 573.0840 \end{bmatrix}^T \\ \mathbf{M}_2^C &= \begin{bmatrix} 54.8701 & -0.0143 & 576.2885 \end{bmatrix}^T \\ \mathbf{M}_3^C &= \begin{bmatrix} 14.7069 & 9.4599 & 548.0546 \end{bmatrix}^T \\ \mathbf{M}_4^C &= \begin{bmatrix} 4.2449 & -44.4407 & 544.8501 \end{bmatrix}^T \end{aligned}$$

Ahora, sólo basta expresar esta posición en el referencial de base utilizando para ello una transformación homogénea $\mathbf{T}_W^C$, de la siguiente manera:

$$\mathbf{M}_i^W = \mathbf{T}_W^C \mathbf{M}_i^C$$

El problema es que el proceso de calibración nos permite obtener la matriz de transformación $\mathbf{T}_C^W$ con los parámetros extrínsecos de la cámara ($\mathbf{R}_C$, $\mathbf{t}_C$), por lo que se debe aplicar la inversa de dicha transformación, de modo que:

$$\mathbf{M}_i^W = \mathbf{T}_W^C \mathbf{M}_i^C = \left(\mathbf{T}_C^W\right)^{-1} \mathbf{M}_i^C$$

con lo cual se obtiene:

$$\begin{aligned} \mathbf{M}_1^W &= \begin{bmatrix} 117.5116 & -17.7985 & -38.9768 \end{bmatrix}^T \\ \mathbf{M}_2^W &= \begin{bmatrix} 97.8699 & 7.4862 & -83.6969 \end{bmatrix}^T \\ \mathbf{M}_3^W &= \begin{bmatrix} 130.5437 & 44.7341 & -76.9879 \end{bmatrix}^T \\ \mathbf{M}_4^W &= \begin{bmatrix} 150.1855 & 19.4494 & -32.2678 \end{bmatrix}^T \end{aligned}$$

Esta información es la que permite dibujar el avatar correspondiente dentro del mundo virtual interactivo diseñado. Así, las coordenadas del punto $\mathbf{M}_1^G$ es el origen del

referencial asociado a dicho avatar, mientras que el eje X_G del referencial del guante es el vector que va de dicho punto al punto $\mathbf{M}_4^G$ y el vector Y_G va de $\mathbf{M}_1^G$ a $\mathbf{M}_2^G$. Finalmente, se trató de calcular la resolución de este sistema de medición visual de la postura de la mano. Tomando en consideración que para una aplicación de ensamble o de manipulación la orientación de la mano cambia muy poco, siendo más notorio el cambio de posición, solamente se midieron los incrementos mínimos en la coordenada z (altura) del vector posición de la mano, obteniéndose una resolución de 2,014 mm., satisfactoria para muchas aplicaciones de rehabilitación médica.

5.7. Conclusiones

En la sección anterior se presentó el desarrollo de un sistema que permite hacer el seguimiento simultáneo de los movimientos de la mano de un usuario y de sus dedos utilizando técnicas de visión artificial sin contacto: reconstrucción 3D basada en homografías para el movimiento de la mano y con flujo óptico (algoritmo de Lucas-Kanade) para el movimiento de los dedos. Los resultados experimentales permiten asegurar que la resolución del sistema es suficientemente buena para el tipo de aplicaciones que pretendemos, mientras que la duración del proceso permite su aplicación a una cadencia de 15 imágenes por minuto.

Los resultados experimentales permiten asegurar que la resolución del sistema es suficientemente buena para el tipo de aplicaciones que pretendemos, mientras que la duración del proceso permite su aplicación a una cadencia de 15 imágenes por segundo, la cual puede mejorarse con un rediseño del sistema informático desarrollado. El mundo virtual interactivo desarrollado para evaluar el sistema de seguimiento visual es muy simple por el momento y aún no permite aplicaciones realistas. Se trata una escena 3D con una mano virtual para representar visualmente los datos capturados en esta propuesta.

Capítulo 6

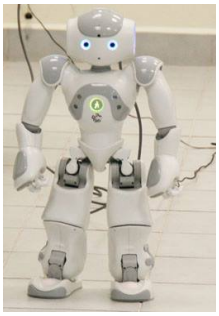
Comando gestual de un humanoide Nao usando visión 3D

El avance de la tecnología a lo largo del tiempo ha permitido a los seres humanos mejorar su calidad de vida. El nacimiento de la Computación y la evolución de la Robótica son los que mayor repercusión ha tenido en las últimas décadas, de modo que la relación entre los humanos y las computadoras ha cambiado mucho desde su nacimiento a la actualidad, pasando del ámbito militar y académico a ser parte indispensable en la vida diaria de la población. En la actualidad existe una computadora (o varias) en cada casa y un smartphone en cada bolsillo, por lo que puede prever que la siguiente etapa es una gran expansión en el uso de robots en todos los ámbitos de la vida [103].

El objetivo de la mayoría de los robots es ayudar o sustituir a un humano en la realización de ciertas actividades, que debido a sus características no son adecuadas para realizarse por personas, son demasiado complicadas o peligrosas, o porque un robot puede realizarlas mejor. En un principio la robótica se utilizó con mucho éxito en la industria, aplicándose en las cadenas de montaje, lo que favoreció su expansión hacia otras aplicaciones. Así pues, en los últimos años se ha hecho hincapié en los robots de servicio [77] los cuales amplian sus funcionalidades, de modo que no se limitan a sustituir a las personas en las líneas de producción, si no que además son capaces de realizar tareas muy diversas y que puedan estar presentes en sectores como la agricultura, minería, sanidad, educación, etc. Dentro de este tipo de robots se encuentran los robots humanoides cuya fisionomía y forma de interactuar con el medio trata de imitar a la de las personas, lo cual los convierte en los más aptos para remplazarnos en la realización de determinadas labores.

El robot Nao es un robot humanoide creado por Aldebaran Robotics, una empresa

francesa con sede en París fundada en el año 2005. Aldebaran Robotics fue creada con la idea de crear robots humanoides que pudieran asistir a las personas, de manera que Nao tiene como objetivo ofrecer una plataforma hardware y software que permita un avance en las investigaciones en este ámbito a un precio razonable.



Nao mide aproximadamente 58 cm de altura y pesa unos 4,8 kg., dispone de una batería de ion de litio que le permite una autonomía de unos 90 minutos y funciona con un procesador *x86 AMD GEODE 500MHz CPU* con 256 MB de memoria SDRAM y 2 GB de memoria flash. Cuenta con 26 articulaciones, que se distribuyen de la siguiente manera:

- 2 grados de libertad en la cabeza
- 5 grados de libertad en cada brazo
- 2 grados de libertad en cada mano
- 5 grados de libertad en cada pierna
- 1 grados de libertad en la pelvis

Cuenta con 4 micrófonos en la cabeza, uno a cada lado, uno en la parte delantera y otro en la parte trasera; también dos altavoces uno a cada lado de la cabeza. Para la visión posee dos cámaras, una que le permite mirar hacia el frente y otra para observar la parte del mundo que tiene más cercana y que esta inclinada hacia abajo, como se puede ver en la figura 6.1. Cuenta con varios sensores de presión en todo el cuerpo: un botón en el pecho, 2 botones tipo bumper en cada pie, 3 sensores táctiles en la cabeza y otros 3 sensores táctiles en cada mano. Cuenta con 8 FSRs (force-sensing resistor), sensores que miden los cambios de resistencia debidos a la fuerza ejercida en un punto, que se encuentran en los pies y son utilizados para que el robot mantenga el equilibrio. Además tiene 2 girómetros de un eje, un acelerómetro de 3 ejes y 2 sónares.

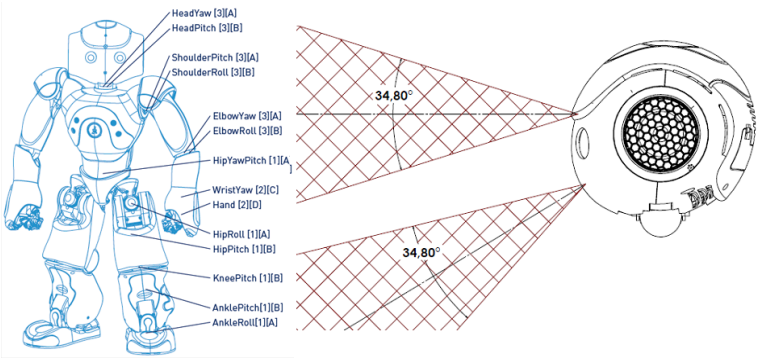


Figura 6.1: Articulaciones y posición de las cámaras.

Desde sus inicios, son diversas las aplicaciones que se han desarrollado para este robot, tales como jugar futbol; reconocer objetos, caras o voces; colaborar con otros Nao para cargar objetos; obedecer ordenes; escribir; realizar coreografías en grupo; tocar un xilófono; ayudar en la cocina entre muchas otras. Por lo tanto, a pesar de sus limitaciones, las posibilidades de expandirse a otras áreas son muy grandes [103].

6.1. Rehabilitación de pacientes ACV

De acuerdo con la OMS (Organización Mundial de la Salud), el accidente cerebrovascular (ACV) es la tercera causa de muerte en los países industrializados y la principal causas de discapacidad en el mundo. Sólo en los EUA el costo de atención de los pacientes que la sufren asciende a 30 billones de dólares.

El ACV es una enfermedad que afecta los vasos sanguíneos que riegan el cerebro y que se conoce con muchos nombres diferentes como: ataque cerebral, apoplejía, derrame cerebral, infarto cerebral, hemorragia cerebral, embolia cerebral o accidente cerebrovascular isquémico. Esto se debe a que su origen es muy variado, aunque puede clasificarse en dos familias: las de tipo isquémico que se refieren al taponamiento de las arterias por un coágulo de sangre y las de tipo hemorrágico debidas al rompimiento de un vaso sanguíneo causando filtración de sangre dentro del cerebro. En ambos casos se pierde la irrigación sanguínea originando un infarto cerebral, cuyos síntomas varían en función del área cerebral afectada. Desde síntomas puramente sensoriales a los puramente motores, pasando por los síntomas sensitivomotores. Los más frecuentemente diagnosticados son: pérdida de fuerza en un brazo o una pierna, o parálisis en la cara (hemiparesia/hemiplejia); dificultad para expresarse, entender lo que se le dice o lenguaje ininteligible (afasia); dificultad al caminar, pérdida de equilibrio o de coordinación; mareos, dolor de cabeza brusco, intenso e inusual, casi siempre acompañado

de otros síntomas; y pérdida de la visión en uno o ambos ojos [31].

Gracias a la plasticidad del Sistema Nervioso Central (SNC) este tipo de padecimientos admiten una rehabilitación que permite la recuperación de la capacidad sensorial y motriz mediante la reconversión de algunas zonas del cerebro. Los programas de rehabilitación recomendados para pacientes con este tipo de padecimientos deben atender tanto los aspectos motores como los relacionados con el habla, los trastornos visuales, las actividades de la vida diaria y las secuelas incapacitantes como la espasticidad, para que el sobreviviente del ACV pueda alcanzar un grado de independencia suficiente como para retomar, al menos parcialmente, sus actividades habituales. Este equipo interdisciplinario debe estar formado por kinesiólogos, neuropsicólogos, fonoaudiólogos, terapeutas ocupacionales, y los relacionados con la medicina, como el médico fisiatra, el psiquiatra y el neurólogo.

Entre las muchas discapacidades que un paciente puede presentar por problemas cerebrales destacan dos por la susceptibilidad que tienen de ser rehabilitadas: la parálisis cerebral en niños y los accidentes cerebrovasculares o apoplejía (stroke). Esto y la gran cantidad de gente que sufre estos padecimientos motivan la realización de este trabajo. La rehabilitación de pacientes con ACV consiste en la repetición de movimientos del miembro afectado tratando de incrementar paulatinamente velocidad, fuerza y precisión de los mismos con ayuda del fisioterapeuta. Muchos esfuerzos se han hecho en el área de Robótica Médica para generar sistemas robotizados de ayuda en estas tareas de rehabilitación [34],[49]. Así, se han propuesto exoesqueletos [78] y robots con cinemática diferente a la del miembro en rehabilitación [22], tanto pasivos como activos, para automatizar estas tareas y para realizar una evaluación en línea de la evolución del paciente, el uso de sistemas de percepción propioceptiva basada en sensores inerciales o de visión artificial [51] para captar los movimientos del miembro en rehabilitación y reproducirlos dentro de un ambiente virtual computarizado que permite al paciente la realización de tareas virtuales comandadas gestualmente mediante dicho miembro [70].

6.2. Solución propuesta

La generación de movimientos en un robot humanoide que opera bajo ciertas limitaciones debido a su arquitectura (dimensiones, articulaciones, etc) es un proceso complicado que consume mucho tiempo computacional y los intentos de crear movimientos más simples ajustando manualmente los parámetros son tediosos y a menudo terminan en un fracaso [19]. En el marco de este trabajo se presenta el desarrollo de una

plataforma formada por un robot humanoide Nao de Aldebaran, un Kinect de Microsoft y una PC capaz de detectar los movimientos de un ser humano y reproducirlos en el humanoide mencionado. En este sistema, el sensor Kinect captura las coordenadas 3D de ciertos puntos de la anatomía del operador (mano, codo, hombro, etc.), datos con los cuales el sistema resuelve el modelo cinemático del operador para calcular el valor de sus coordenadas articulares. Estas coordenadas son escaladas y convertidas a las coordenadas articulares del humanoide mencionado, cuyo modelo cinemático es diferente al del ser humano, pero compatible. Con esta información es posible mover al humanoide ya sea en simulación mediante el software propietario Choregraph o directamente enviando dichos datos al robot o aún a los dos simultáneamente. La motivación de este proyecto es su posible uso en la rehabilitación de niños con parálisis cerebral o de pacientes apopléjicos que hayan sufrido un evento vascular cerebral (stroke).

El problema que se presenta con este enfoque es que los humanos y los robots no comparten capacidades motoras, amplitud de movimiento, dimensiones, masa y otras características físicas. Para los objetivos que persigue este trabajo, se asume que las dimensiones y la masa de los miembros del cuerpo humano y del robot son aproximadamente equivalentes. Esta suposición tiene mucho sentido al considerar que el interés principal se centra en el rango de movimiento, esto es, el objetivo es mapear desde el espacio de movimiento del humano al restringido espacio de movimiento del robot.

Gracias al gran desarrollo que han tenido las Tecnologías de la Información y las Comunicaciones (TICs) durante los últimos años han proliferado los dispositivos de adquisición de información visual 2D y 3D. Esto ha dado un enorme impulso a la investigación en la comunicación hombre-máquina mediante dispositivos visuales, generándose así lo que se conoce como Natural Human Robot Interaction (HRI) [87],[20],[32]. Con base en estas tecnologías ya es posible que la máquina (robot equipado con visión artificial) detecte y comprenda gesticulaciones corporales humanas; de este modo ya puede establecerse una comunicación gestual entre el humano y la máquina (computadora o robot).

Así, ya se cuenta con interfaces de comunicación hombre-máquina que permiten, por ejemplo, que personas con alguna discapacidad que les impida manipular un teclado, puedan comandar un sistema robótico simplemente con el movimiento de su cuerpo, logrando de este modo una interacción más intuitiva para la realización de ciertas tareas. Otra posible aplicación de esquemas de comando gestual de computadoras o de robots es en programas de rehabilitación de pacientes que ven disminuida su movilidad por problemas del sistema nervioso central que son susceptibles de regenerarse como la

Figura 6.2: Seguimiento del movimiento del usuario mediante marcadores reflectantes.

parálisis cerebral en niños o de pacientes de cualquier edad que han sufrido Accidentes Cerebro-Vasculares (ACV, apoplejía o stroke).

Es en ese sentido que se diseño un proyecto a fin de contar con una herramienta simple de utilizar que permite la comunicación gestual entre un ser humano y una computadora con el fin de comandar el movimiento de un robot humanoide, es decir para conseguir que un robot humanoide repita los movimientos que han sido detectados en un ser humano, con fines lúdicos y de entretenimiento o bien para efectos de rehabilitación de los pacientes antes mencionados.

Existen diversos sistemas que realizan un correcto seguimiento del movimiento del cuerpo humano, siendo uno de los más conocido aquel que utiliza marcadores de captura de movimiento óptico, esto es, un usuario lleva normalmente un traje con varios marcadores reflectantes que son observados por varias cámaras aéreas, obteniéndose su posición mediante triangulación [2]. En la figura 6.2 se observa el seguimiento del cuerpo humano utilizando un sistema de marcador óptico con el robot Nao. Una desventaja importante de estos sistemas es que requieren grandes espacios con equipos y software caros. Ahora bien, el enfoque utilizado en estra trabajo considera el uso del sensor Kinect de Microsoft, aunque no es tan preciso como las plataformas más costosas proporciona un nivel suficiente de detalle, es fácilmente accesible y no requiere un espacio o laboratorio especial. El sensor Kinect genera información de color y profundidad imágenes que se pueden utilizar para la captura o seguimiento de movimiento (MoCap) del cuerpo de un usuario.

En resumen, el objetivo es desarrollar un sistema de comunicación gestual entre un ser humano y el robot humanoide Nao de Aldebaran que permita que dicho robot repita simultáneamente los movimiento del humano detectados mediante un sistema comercial de visión 3D (Kinect de Microsoft) y que pueda utilizarse en tareas de rehabilitación de niños con parálisis cerebral o pacientes que hayan sufrido ACV.

Para ello, se desarrolló un sistema compuesto por:

i) Un sistema de visión 3D basado en el sensor el Kinect, utilizado para detectar las coordenadas $[x, y, z]^T$ de ciertos puntos correspondientes a partes del cuerpo del operador tales como manos, codos, hombros, cuello, cabeza, etc.

ii) Un robot humanoide comercial Nao de Aldebarán.

iii) Una computadora personal en la que residen modelos (cinemática del operador humano, cinemática del humanoide Nao) y algoritmos (calibración de cámara, captura y procesamiento de imágenes, cálculo de la cinemática, escalamiento cinemático, simulación del humanoide Nao y controlador de dicho robot).

La incorporación de un robot humanoide en el sistema de rehabiltación de este tipo de pacientes agrega una componente atractiva y lúdica que podría permitir los objetivos de rehabilitación más pronto y, además, permite tener tareas de rehabilitación más variadas, como hacer que el robot realice una cierta tarea de manipulación de objetos o bien su avatar virtual dentro de un simulador.

En este trabajo se propone una opción de este tipo, en donde el paciente se ubica de pie frente a un sistema de visión 3D compuesto por un sensor Kinect que captura los movimientos del brazo a ser rehabilitado como un conjunto de puntos 3D. Estos puntos 3D se utilizan para calcular las variables articulares del paciente siguiendo un modelo cinemático obtenido mediante la metodología de Denavit-Hartenberg (D-H) Dado que la estructura cinemática de los miembros superiores del robot humanoide Nao es equivalente a la del ser humano (bajo ciertas reestricciones de movimiento), estos valores articulares calculados pueden utilizarse directamente como consignas de movimiento a dicho robot. De este modo el robot estará copiando fielmente los movimientos del paciente. Aunque también puede utilizarse Choregraphe®, el software de simulación de este humanoide.

6.3. Biomecánica del brazo humano

A continuación se describe la biomecánica del brazo que representa un aspecto de interés en el marco de este trabajo, en donde también se incluye el cálculo de las cuatro variables articulares del brazo del paciente que debe calcular el sistema de rehabilitación propuesto. Para realizar el análisis de la cinemática del pacientes es necesario definir los planos sobre los cuales se efectúan los movimientos de interés. De este modo, la figura 6.3 muestra los planos transversal, coronal, y sagital en el cuerpo humano que se utilizarán en el análisis de la cinemática del paciente.

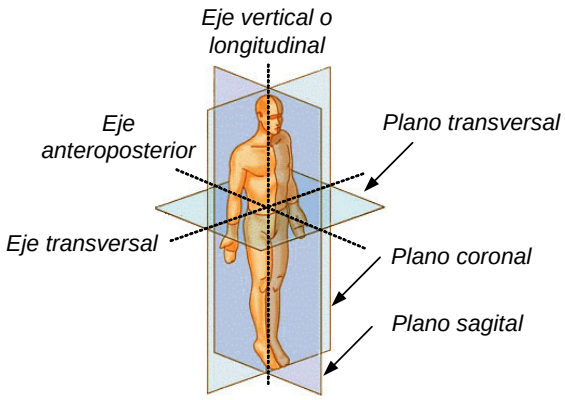


Figura 6.3: Planos y ejes del cuerpo humano.

En términos generales, la biomecánica del hombro es complicada, por lo tanto se requiere un análisis detallado y por separado de cada una de las articulaciones que lo conforman. Sin embargo para los objetivos que persigue este trabajo sólo se requiere de un estudio breve sobre la naturaleza, rangos de movimiento y algunos otros detalles referentes a los movimientos más significativos del hombro y del codo, en particular: flexo-extensión, abducción-aducción de hombro, rotación interna-externa del hombro y flexo-extensión del codo. La elección de estos movimientos se debe a que, para fines de valoración y de terapias de rehabilitación, estos son los movimientos analizados y sobre los cuales se trabaja generalmente en el área de fisioterapia.

Flexo-extensión del hombro

Este movimiento se efectúa en dos sentidos sobre el plano sagital en torno a un eje transversal (figura 6.4):

i) *Flexión:* Se realiza elevando el brazo hacia delante y su amplitud es de 0° a 180°.

ii) *Extensión:* Movimiento que se lleva a cabo en sentido contrario a la flexión su amplitud es 0^o a 50°.

Abducción-aducción del hombro

Este movimiento también se realiza en dos sentidos sobre el plano coronal alrededor del eje anteroposterior (figura 6.5):

i) *Abducción o separación:* Se realiza desplazando el brazo hacia afuera y su amplitud es de 0° a 90°.

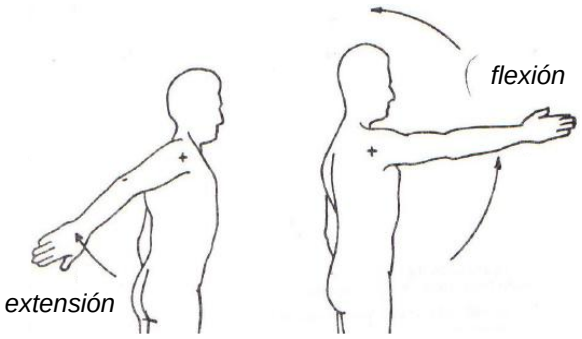


Figura 6.4: Flexión-Extensión del hombro.

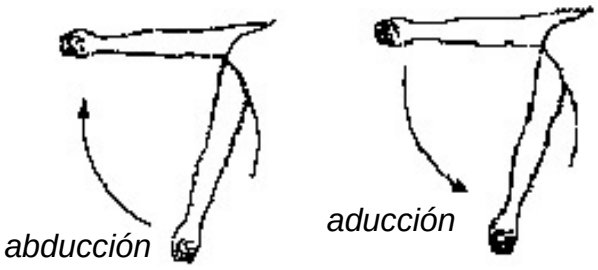


Figura 6.5: Abducción-Aducción del hombro.

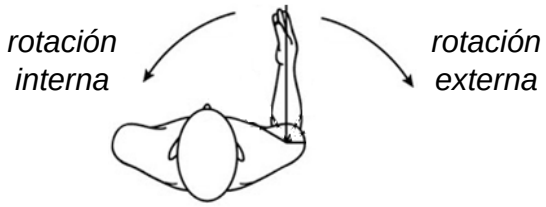


Figura 6.6: Rotación interna-externa del hombro.

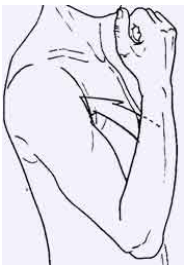


Figura 6.7: Flexión del codo.

ii) *Aducción o aproximación:* Es el movimiento contrario al anterior y tiene igual amplitud.

Rotación interna-externa del hombro

Este movimiento también se realiza en dos sentidos sobre el plano transversal alrededor del eje longitudinal (figura 6.6):

i) *Rotación interna:* Este movimiento puede ejecutarse llevando la mano hacia dentro con el codo en flexión de 90^o, su amplitud es de 100° a 110°.

ii) *Rotación externa:* Inverso al anterior, se realiza llevando la mano hacia afuera con el codo en flexión de 90^o y su amplitud es de 80°, jamás alcanza los 90°.

Flexión del codo

Este movimiento permite acercar las caras anteriores del brazo y del antebrazo. El regreso de la flexión a la posición anatómica se llama extensión del codo (figura 6.7), sin embargo, no es posible la extensión hacia atrás más allá de la posición anatómica.

6.3.1. Modelo cinemático propuesto

Una vez definida la naturaleza de las variables biomecánicas que se pretenden medir, resulta indispensable abordar el cálculo del modelo cinemático inverso cuyo

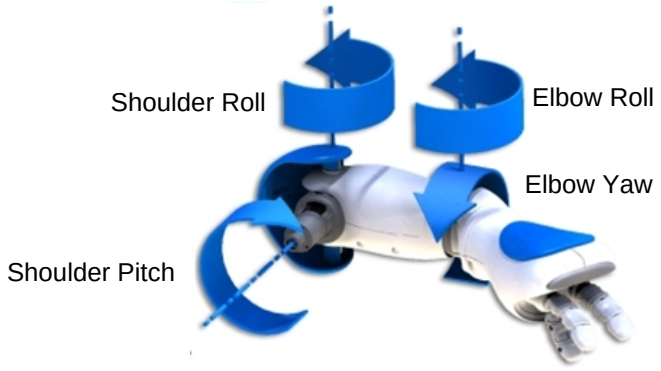


Figura 6.8: Articulaciones del brazo del humanoide.

planteamiento se expresa como: Dadas únicamente las posiciones de 7 puntos de interés del esqueleto obtenido mediante OpenNI/NITE (hombros, codos, manos y torso) se deben calcular las variables articulares asociadas (θ_1, θ_2, θ_3 y θ_4), mismas que corresponden a la abducción-aducción de hombro, flexo-extensión del hombro, rotación interna-externa del hombro y flexo-extensión del codo respectivamente.

Nao es un robot humanoide que cuenta con 25 gdl distribuidos de la siguiente manera: dos en el cuello, cinco en cada pierna (tobillo 2, rodilla 1 y cadera 2), uno en la pelvis y cinco más en cada brazo (hombro 3, codo 1 y muñeca 1), además de tener manos prensiles que pueden considerarse como un grado de libertad más en cada una de ellas. Existe una versión especial para competir en la Liga de plataforma estándar (SPL) de Robocup que cuenta con menos sensores y dos gdl menos en cada brazo (muñeca y mano) y este es justamente el robot con el que se desarrolla este proyecto. En la figura 6.8 se muestran las 4 articulaciones correspondientes al brazo y sus respectivos ejes de rotación, en donde se observa que Shoulder Roll es equivalente al ángulo de Rotación Interna-Externa del hombro, Shoulder Pitch corresponde con el ángulo de Flexión-Extensión del hombro, mientras que Elbow Roll equivale al ángulo de Flexión del codo.

La metodología propuesta se basa en un enfoque totalmente vectorial y trigonométrico que utiliza los vectores entre las articulaciones del esqueleto detectado para determinar las variables articulares del robot, y el cual se puede extender a cualquier robot que contenga articulaciones rotacionales. Ahora bien, la cinemática inversa puede utilizarse como un método alternativo para determinar dichos valores articulares, sin embargo introduce una cierta ambiguëdad, dado que por ejemplo la trayectoria de las articulaciones intermedias en una cadena cinemática no se garantizan para seguir el movimiento del usuario. Además, el objetivo que pretende no es posicionar los efec-

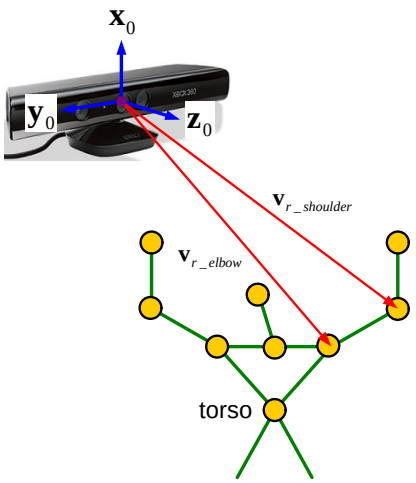


Figura 6.9: Vectores del hombro y codo.

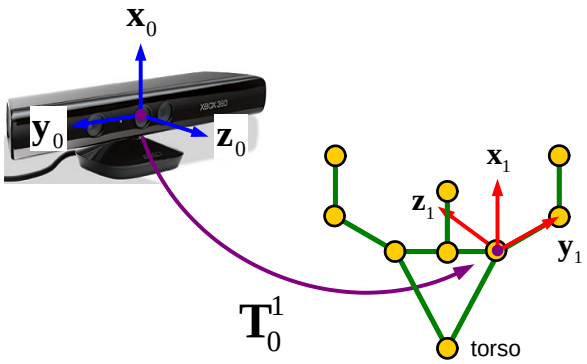


Figura 6.10: Referenciales del Kinect y del usuario.

tores finales del robot sino asegurar que los ángulos del robot relativos a las partes del cuerpo humano sean correctos. Por esta razón, lo más apropiado es hacer un cálculo directo de las variables articulares tal como se considera en el enfoque propuesto.

Por simplicidad se describe el caso del brazo derecho, sin embargo el procedimiento es el mismo para ambos miembros superiores. El esquema de la figura 6.9 muestra el caso particular de los vectores $\mathbf{v}_{r\_shoulder}$ y $\mathbf{v}_{r\_elbow}$ correspondientes a la posición del hombro y codo derechos respectivamente, ambos expresados en un referencial localizado en el Kinect. Ahora bien, resulta conveniente expresar dichos vectores con respecto a un referencial asociado al cuerpo a modo de obtener una idea general y verídica del movimiento del tronco en su conjunto, esto es, al moverse el torso también se mueven ambos hombros en la misma proporción (ver figura 6.10).

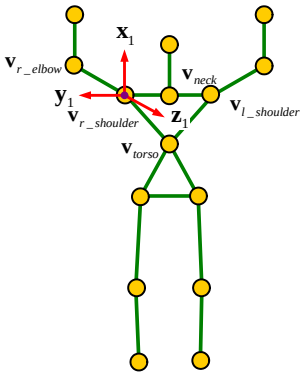


Figura 6.11: Diagrama cinemático.

Los dos referenciales mostrados en la figura 6.10 se relacionan mediante una matriz de transformación homogenéa $\mathbf{T}_0^1$ compuesta por una matriz de rotación y un vector de traslación que modelan la orientación y la posición del torso. Esta representación de los puntos en el nuevo referencial proporciona invarianza ante la pose del paciente, ya que en muchas ocasiones se presentan dificultades para permanecer perfectamente erguido o en una alineación frontal paralela al sensor Kinect. La transformación inversa $\mathbf{T}_1^0$ mapea cualquier vector en el referencial del Kinect a su correspondiente representación en el referencial asociado al torso del usuario. Por otra parte, dado que sólo se requiere la orientación (ángulos) de las articulaciones, la matriz $\mathbf{T}_1^0$ está dada por:

$$\mathbf{T}_1^0 = \mathbf{R}_1^0$$

Sea $\mathbf{v}_0$ un vector arbitrario obtenido con el sensor Kinect, este se expresa como $\mathbf{v}_1$ en el referencial del torso mediante la expresión:

$$\mathbf{v}_1 = \mathbf{R}_1^0 \mathbf{v}_0$$

La figura 6.11 muestra el diagrama del esquelto obtenido con el sensor Kinect y su referencial asociado. Los ejes se determinan a manera de formar un referencial derecho como:

$$\begin{aligned} \mathbf{z}_1 &= \left(\mathbf{v}_{l\_shoulder} - \mathbf{v}_{torso}\right) \times \left(\mathbf{v}_{r\_shoulder} - \mathbf{v}_{torso}\right) \\ \mathbf{y}_1 &= \mathbf{v}_{r\_shoulder} - \mathbf{v}_{neck} \\ \mathbf{x}_1 &= \mathbf{y}_1 \times \mathbf{z}_1 \end{aligned}$$

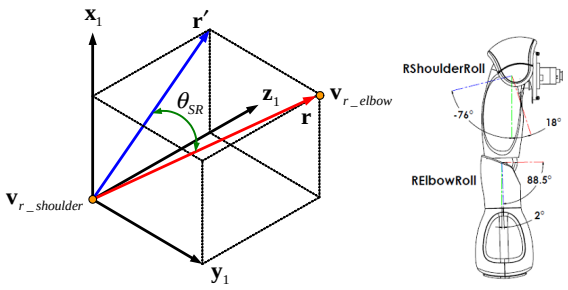


Figura 6.12: Roll del hombro.

Así pues, la transformación que relaciona ambos referenciales está dada por:

$$\mathbf{R}_1^0 = \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ z_1 & z_2 & z_3 \end{bmatrix}$$

Después de la transformación de todos los vectores de posición al nuevo referencial, las variables articulares se pueden calcular fácilmente mediante consideraciones geométricas. En la figura 6.12 se observa el brazo derecho en una configuración arbitraria, donde $\mathbf{r}$ representa el vector que va de $\mathbf{v}_{r\_shoulder}$ a $\mathbf{v}_{r\_elbow}$ y está dado por:

$$\mathbf{r} = \mathbf{v}_{r\_elbow} - \mathbf{v}_{r\_shoulder}$$

Este se proyecta como $\mathbf{r}'$ en el plano formado por $\mathbf{y}_1$ y $\mathbf{z}_1$ como se muestra en la figura 6.12:

$$\mathbf{r}' = (\mathbf{x}_1 \cdot \mathbf{r})\,\mathbf{x}_1 + (\mathbf{z}_1 \cdot \mathbf{r})\,\mathbf{z}_1 \tag{6.1}$$

El ángulo de *roll del hombro* (rotación interna-externa) está dado por:

$$\theta_{Shoulder\_Roll} = -\cos^{-1}\left(\frac{\mathbf{r} \cdot \mathbf{r}'}{|\mathbf{r}|\,|\mathbf{r}'|}\right) \tag{6.2}$$

De acuerdo a la figura 6.13, el ángulo de *pitch del hombro* (flexión-extensión) es:

$$\theta_{Shoulder\_Pitch} = -\cos^{-1}\left(\frac{\mathbf{z}_1 \cdot \mathbf{r}'}{|\mathbf{z}_1|\,|\mathbf{r}'|}\right) \tag{6.3}$$

Sea $\mathbf{s}$ el vector que va de $\mathbf{v}_{r\_elbow}$ a $\mathbf{v}_{r\_wrist}$:

$$\mathbf{s} = \mathbf{v}_{r\_elbow} - \mathbf{v}_{r\_shoulder}$$

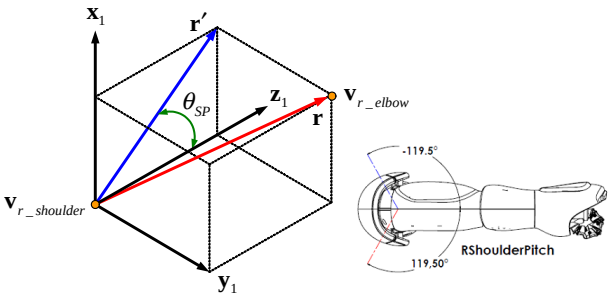


Figura 6.13: Pitch del hombro.

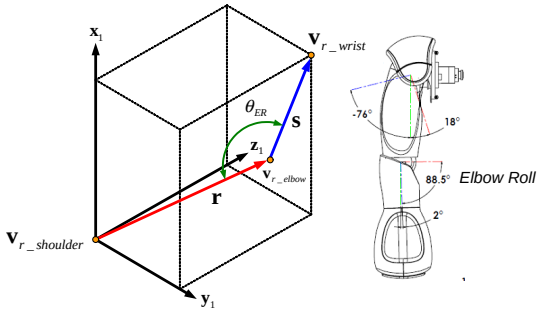


Figura 6.14: Roll del codo.

El ángulo de *roll del codo* (flexión) se calcula como:

$$\theta_{Elbow\_Roll} = \cos^{-1}\left(\frac{\mathbf{r}\cdot\mathbf{s}}{|\mathbf{r}|\,|\mathbf{s}|}\right)$$

Para determinar el ángulo de *yaw del codo* es necesario definir un nuevo referencial ubicado en el codo, cuyos ejes se definen de la siguiente manera (ver figura 6.15):

$$\mathbf{y}_2 = \frac{\mathbf{r}}{|\mathbf{r}|} \qquad \mathbf{z}_2 = \frac{\mathbf{r}\times\mathbf{r}'}{|\mathbf{r}\times\mathbf{r}'|} \qquad \mathbf{x}_2 = \frac{\mathbf{y}_2\times\mathbf{z}_2}{|\mathbf{y}_2\times\mathbf{z}_2|}$$

Ahora bien, $\mathbf{s}$ se proyecta como $\mathbf{s}'$ en el plano formado por $\mathbf{x}_2$ y $\mathbf{z}_2$ tal como se ilustra en la figura 6.16**:**

$$\mathbf{s}' = (\mathbf{x}_2\cdot\mathbf{s})\,\mathbf{x}_2 + (\mathbf{z}_2\cdot\mathbf{s})\,\mathbf{z}_2 \tag{6.4}$$

El ángulo de *Yaw del codo* (*Elbow_ Yaw*) es:

$$\theta_{Elbow\_Yaw} = \cos^{-1}\left(\frac{\mathbf{x}_2\cdot\mathbf{s}'}{|\mathbf{x}_2|\,|\mathbf{s}'|}\right)$$

Un análisis similar se realiza para el brazo izquierdo.

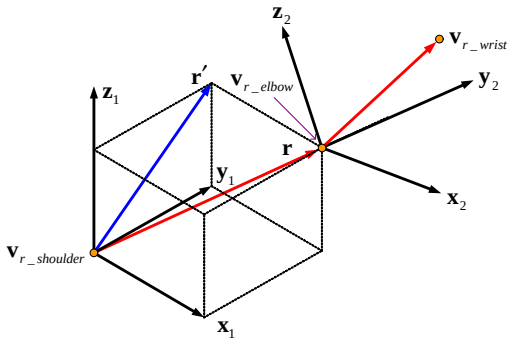


Figura 6.15: Referencial asociado al codo.

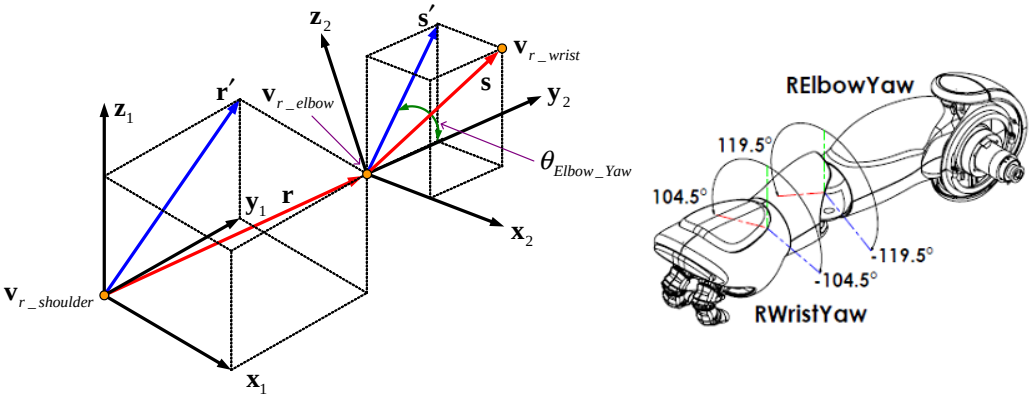


Figura 6.16: Yaw del codo.

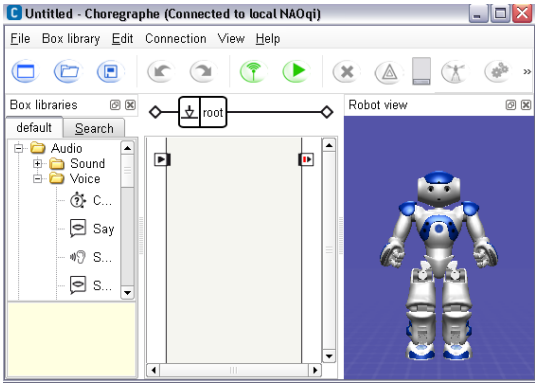


Figura 6.17: Interfaz de Choreographe.

6.4. Resultados experimentales

Nao puede programarse de diversas maneras mediante sencillos pero muy intuitivos métodos interactivos basados en el software Choregraphe cuya interfaz con el usuario se muestra en la figura 6.17, o bien utilizando sofisticados y poderosos sistemas como el Nao SDK y NaoQi que le confiere una gran versatilidad y permite que lo utilicen de manera productiva estudiantes de cualquier nivel y aún investigadores. Choregraphe incluye un simulador cinemático que debe asociarse a algún programa (Java, Python, etc.) para poder comandar directamente el robot físico.

La morfología de este humanoide, al menos a nivel de los primeros 4 gdl de sus extremidades superiores, es equivalente a la del ser humano por lo que el escalamiento necesario entre los modelos cinemáticos de ambos mecanismos consiste sólo en la manera de medir los ángulos y en los límites articulares. La realización de las pruebas entre la interacción de los movimientos humanos y la plataforma Nao contempla la implementación de una interfaz gráfica con la finalidad de facilitar la interacción hombre – humanoide; actualmente existe una gama de software disponible para realizar dicha implementación como son C++, C#, Visual Basic, Java, Qt entre otros; sin embargo se ha optado por la implementación bajo el código de Java. En la figura 6.18 se muestra la implementación de la interfaz, la cual se encarga de gestionar la información entre el Kinect y la plataforma Nao de manera inalámbrica utilizando comandos proxy.

La interfaz cuenta con campos destinados para cada una de las articulaciones de interés de ambo brazos y con la visualización de la imagen en formato RGB, en profundidad o bien sin despliegue de la misma. En los 3 casos se visualiza por defecto el esqueleto de la persona posicionada frente al Kinect. Por otra parte, se encarga de realizar el enlace entre el software Choreograph si no se cuenta con la plataforma Nao de

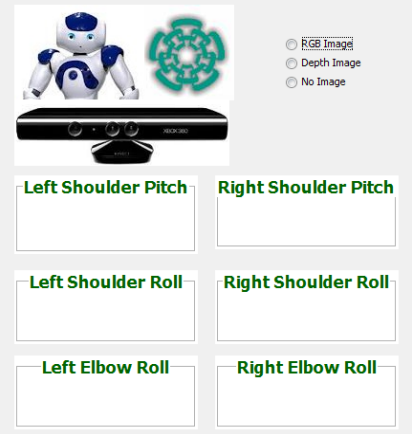


Figura 6.18: Interfaz gráfica.

manera física, sin embargo también es posible enviar las consignas de movimiento a la plataforma Nao sin necesidad de estar en ejecución éste software (Choreograph), o bien, es posible realizar el enlace entre la plataforma Nao y el Choreograph al mismo tiempo. Además, una de sus principales ventajas de utilizar la plataforma de simulación es que permite mantener al humanoide en un estado seguro dado que éste no se encuentra en interacción directa, evitando que existan colisiones entre sus brazos o alguna otra parte de su cuerpo o con objetos de su entorno.

En la figura 6.19 se presentan los resultados obtenidos, en las imágenes superiores se observa en la parte izquierda la imagen RGB en donde se sobrepone el esqueleto detectado de la persona (paciente), mientras que a la derecha se muestra la interacción del Nao en el software Choreograph, reproduciendo los movimientos de la persona. Las imágenes de la parte inferior muestran la interacción de la interfaz gráfica entre el simulador Choreographe y el robot Nao reproduciendo los movimientos del usuario. Note que en dichas imágenes se presenta la visualización a partir de la imagen de profundidad generada por el sensor Kinect.

En las figuras 6.20, 6.21 y 6.22 se muestra una secuencia del seguimiento de los movimientos del usuario por parte del Robot Nao. Note que la reproducción del movimiento humano está limitado por la diferencia entre los grados de libertad del robot y del humano, así como por la manera en como funcionan las articulaciones de cada uno. Por lo tanto, es evidente que los movimientos no pueden ser completamente idénticos, pero tienen la suficiente precisión para utilizarse para fines de rehabilitación.

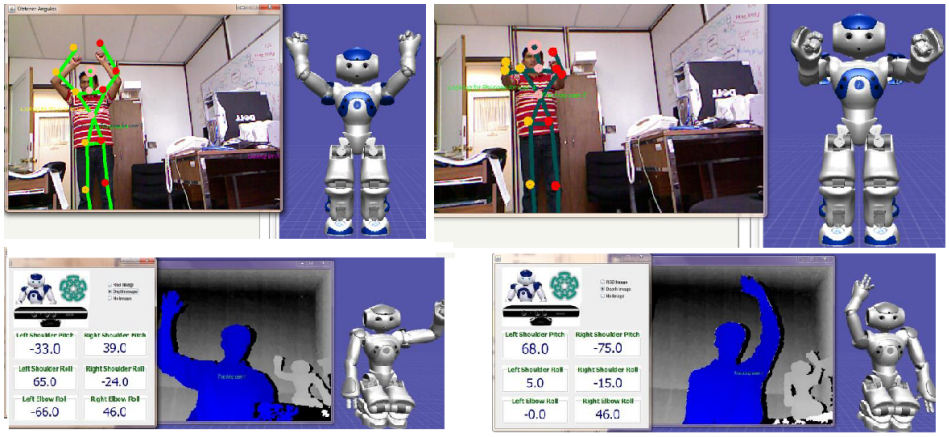


Figura 6.19: Resultados experimentales.

Figura 6.20: Seguimiento del movimiento.

Figura 6.21: Seguimiento del movimiento.

Figura 6.22: Seguimiento del movimiento.

6.5. Conclusiones

El sistema de rehabilitación, compuesto por el sensor Kinect y el robot Nao representa un herramienta muy apropiada para la rehabilitación de funciones de tipo motriz, siendo uno de sus principales beneficios evitar que el paciente tenga que desplazarse hasta el lugar donde se encuentra el equipo de fisioterapeutas. En el caso particular de los niños, los hospitales suelen ser ambiente muy poco familiares e incómodos, por lo que en la medida que un proceso de rehabilitación se puede llevar a cabo en el hogar siempre será positivo. Por otro lado, la solución propuesta ofrece una alternativa viable para la réplica de movimientos humanos gracias al desarrollo de hardware relativamente económico y con buen nivel de precisión como lo es el sensor Kinect de Microsoft.

En lo que concierne a la implementación del sistema de adquisición de movimientos, el principal problema que se presentó es el retardo temporal que presenta los datos entregados por el sensor con respecto a los movimientos reales de una articulación. Para minimizar este inconveniente, se propone una solución que no consume demasiado tiempo computacional pues se basa en un enfoque puramente geométrico y utilizar la librería OpenNI y su clase SkeletonJoint que cuenta con una frecuencia de muestreo de 30 Hz y que efectua un correcto seguimiento de los movimientos del usuario. De esta manera, se logró construir un sistema que presenta un buen tiempo de respuesta, que tiene la capacidad de responder ante cambios bruscos en las posiciones de las articulaciones y que no genera oscilaciones indeseables, lo cual genera una operación agradable e interactiva del sistema. Sin embargo, debido a que el humanoide Nao tiene restricciones mecánicas de movimiento, las variables articulares del operador captadas con el sensor Kinect tienen que ser acondicionadas para evitar colisiones de sus extremidades, pero los movimientos requeridos para realizar la rehabilitación ante Accidentes Cerebro Vasculares son posibles con la metodología propuesta en este

trabajo. Por lo tanto, se concluye que mediante un sensor como el Kinect y el robot Nao es posible imitar el movimiento de un usuario humano de forma eficiente en tiempo real aprovechando la versatilidad del sensor Kinect y al desarrollo de la solución propuesta.

Como trabajo futuro se contempla la interacción de movimientos en extremidades inferiores (piernas), así mismo, movimientos de relacionados con la cabeza. Además de generar una sincronización con otro robot, para realizar una coreografía entre las plataformas.

Apéndice A

Sensor Kinect

Microsoft Research invirtió veinte años en el desarrollo del Kinect. Este dispositivo fue desarrollado por la empresa israelí PrimeSense por encargo de Microsoft y presentado el 1 de junio de 2009 en la Electronic Entertainment Expo 2009 y exhibido como una interfaz hombre-máquina destinada para los juegos de video. El sensor tuvo como objetivo principal revolucionar el mercado de los controles (o mandos) para la entonces nueva generación de consolas de videojuegos. A diferencia de los controles existentes, no es necesario un dispositivo electrónico que el jugador o usuario deba llevar consigo en sus manos, si no que el usuario es el propio mando a distancia, controlando la dinámica del videojuego mediante ademanes, posturas y movimientos del cuerpo en general, incluso en ocasiones mediante comandos de voz.

Las capacidades y flexibilidad de dicho sensor le han permitido encontrar una enorme diversidad de aplicaciones, muchas de ellas en el mundo de la investigación en Robótica, aprovechando al máximo las características que éste ofrece. Uno de los motivos fundamentales de enfocar la atención en este dispositivo radica en la capacidad de contar con un sensor de profundidad de bajo costo, fácil adquisición y una relativa facilidad en cuanto a su uso. El Kinect es una barra horizontal de aproximadamente 23cm de largo conectada a una base circular mediante una articulación de elevación actuada (tilt), tal como se muestra en la figura A.1. Los principales componentes del Kinect son:

1. Cámara RGB (sensor CMOS).
2. Emisor de infrarrojos (IR).
3. Arreglo de 4 Micrófonos.
4. Servomecanismo de elevación (tilt).
5. Cámara IR (Sensor CMOS monocromático IR).

La cámara de captura en color o también llamada cámara RGB está fabricada con un sensor CMOS y tiene tres resoluciones posibles, de 640×480 píxeles a 30 fps, de 640×480 pixeles a 15fps y de 1280×960 pixeles a 12 fps. Su campo de visión horizontal

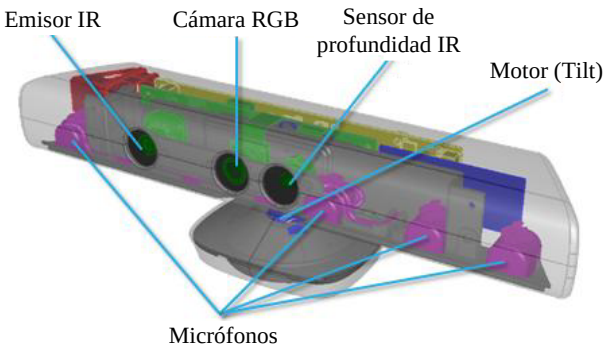


Figura A.1: Elementos del Kinect.

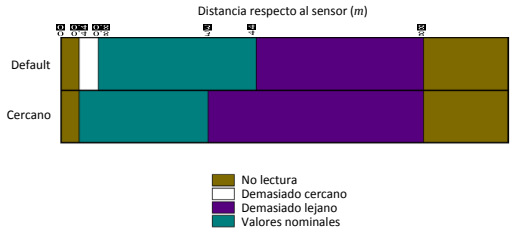


Figura A.2: Rangos de profundidad.

es de 57° y el vertical de 43°, mientras que su ángulo de elevación (tilt) es de ±27° con respecto a la horizontal.

En cuanto al sensor de profundidad, el sensor Kinect cuenta con un rango de 0,4m a 8m, y se ofrece en dos versiones. La primera, conocida como modo por defecto, se caracteriza por tener dos rangos dentro de los cuales la información generada no es del todo confiable: demasiado cerca (entre 0,4m y 0,8m) y demasiado lejos (entre 4m y 8m), quedando como rango óptimo el que va de los 0,8m a los 8m. La segunda versión, conocida como modo cercano, tiene un rango útil que va de 0,4m a 3m, teniendo entonces un rango demasiado lejos entre los 3m y los 8m (ver figura A.2).

El sensor de profundidad esta formado por un emisor de laser infrarrojos, un sensor CMOS monocromo y el módulo encargado de construir la nube de puntos tridimensional. El funcionamiento de este sensor es similar al del sonar de un barco, primero el emisor de infrarrojos proyecta una matriz de puntos de luz infrarroja invisible al ojo humano (figura A.3). Cada uno de estos rayos de luz viaja por el aire hasta rebotar con algún objeto, para posteriormente ser capturado por el sensor CMOS. Los rayos de luz infrarroja van perdiendo intensidad conforme mayor sea la distancia recorrida, por lo tanto, una vez que el sensor CMOS ha capturado toda la imagen, se puede calcular la distancia a la que cada rayo de luz ha rebotado a partir de su intensidad. La principal ventaja de este sistema consiste en que la luz infrarroja funciona bien bajo prácticamente cualquier condición de luminosidad.

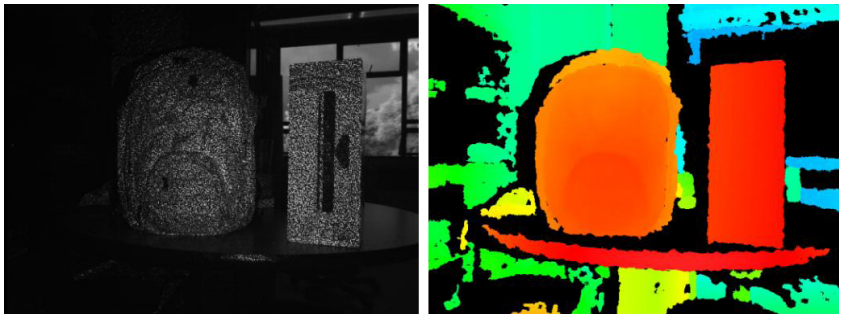


Figura A.3: Sensor de profundidad.

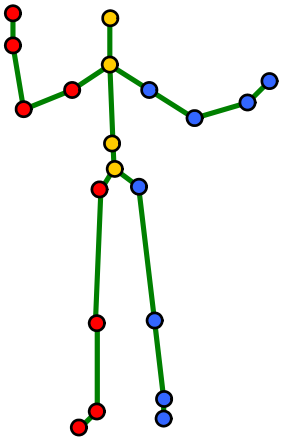


Figura A.4: Puntos del cuerpo que detecta el sistema de captura basado en el Kinect.

A.1. OpenNI

Debido a que el Kinect fue diseñado para el entretenimiento, en donde los jugadores interactúan con video juegos mediante sus movimientos corporales sin utilizar ningún otro dispositivo de control, cuenta con un sistema de seguimiento que detecta hasta 6 personas, 2 jugadores activos y 4 pasivos, siendo posible cambiar de rol, es decir, un jugador activo puede convertirse en pasivo y viceversa. Para cada jugador activo identificado se detectan se asignan 20 puntos cuyas coordenadas son monitoreadas constantemente, tal como se observa en la figura A.4.

En noviembre de 2010, a un año de haberse dado a conocer el sensor Kinect, Adafruit Industries® ofreció una recompensa monetaria a quien fuera capaz de desarrollar un controlador de código abierto para el sensor Kinect. Así, el 10 de noviembre de dicho año, se anunció al español Héctor Martín como el ganador. Su controlador fue desarrollado para trabajar sobre la plataforma GNU/Linux permitiendo el uso de la cámara RGB así como la obtención de un mapa de profundidad. A partir de dicho acontecimiento, desarrolladores de todo el mundo se han dado a la tarea de extraer y explotar las prestaciones que Kinect es capaz de ofrecer más allá de su eventual uso en

juegos de video. Es por esto que en la actualidad se pueden identificar 5 librerías de código abierto para la extracción, acondicionamiento y uso de la información obtenida por el sensor Kinect, dichas librerías son: OpenKinect/libfreenect, CLNUI, Microsoft Kinect SDK, OpenNI/NITE y SimpleOpenNI.

La librería libfreenect funciona para plataformas de sistemas operativos Windows, Linux y MacOS X. Mientras que CLNUI es sólo para Windows, sin embargo, permite la conexión de varios Kinects. En el marco de este trabajo se optó por utilizar el paquete OpenNI versión 1.5.4 debido a su amplia aceptación, documentación disponible, funcionalidad y flexibilidad de uso en diversas plataformas y sistemas operativos, pero principalmente porque con esta librería y su clase SkeletonJoint es posible integrarla con el software Java.

OpenNI (Open Natural Interaction) es una arquitectura de software libre multiplataforma que provee ciertas APIs para el desarrollo de aplicaciones que involucren interacción natural. El propósito principal de OpenNI es constituir una API estándar que permita la comunicación tanto con sensores de visión y de audio, es decir, dispositivos que ven y escuchan el entorno considerado, así como Middleware de visión y de audio, esto es, componentes de software que analizan e interpretan la información visual y auditiva obtenida de la escena o entorno. De este modo, OpenNI provee un conjunto de APIs a ser implementadas por dispositivos sensores y un conjunto de APIs a ser implementadas por componentes middleware.

Con la librería OpenNI y su clase SkeletonJoint es posible identificar, detectar y realizar el seguimiento de 20 nodos del cuerpo, sin embargo, 9 de estos nodos no se encuentran disponibles en la versión empleada, por lo que sólo pueden utilizarse los 15 restantes tal como se observa en la figura A.5. SkeletonJoint lleva a cabo las siguientes funciones:

1. Identificar a una persona del resto de la escena.
2. Desplegar una ventana mostrando la escena observada.
3. Realizar una calibración del usuario manteniendo una pose inicial.
4. Determinar la posición y orientación de los 15 nodos del cuerpo del usuario.

A.2. Seguimiento del esqueleto del usuario

El propósito del algoritmo de segmentación de usuario es el de identificar y rastrear a uno o más usuarios presentes de cuerpo entero en la escena. A cada usuario se le asigna un único y persistente número de identificación (ID). La salida principal del proceso de segmentación de usuario es un mapa de etiquetado, el cual consiste en asignarle a cada pixel de la escena una etiqueta, la cual identifica la pertenencia de dicho pixel a un usuario en particular tal como se ilustra en la figura A.6.

Una de las principales cualidades de OpenNI es su capacidad de obtener una representación virtual de cuerpo completo de una o más personas ubicadas en el campo

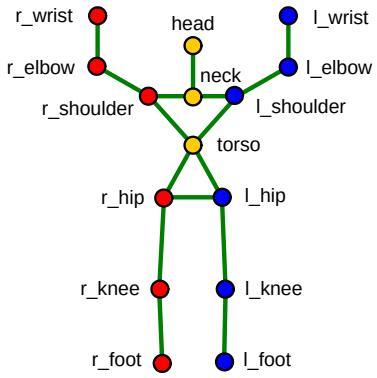


Figura A.5: Detección de 15 puntos de interés en el cuerpo del usuarrio.

Figura A.6: Segmentación de usuarios y mapa de etiquetado generado.

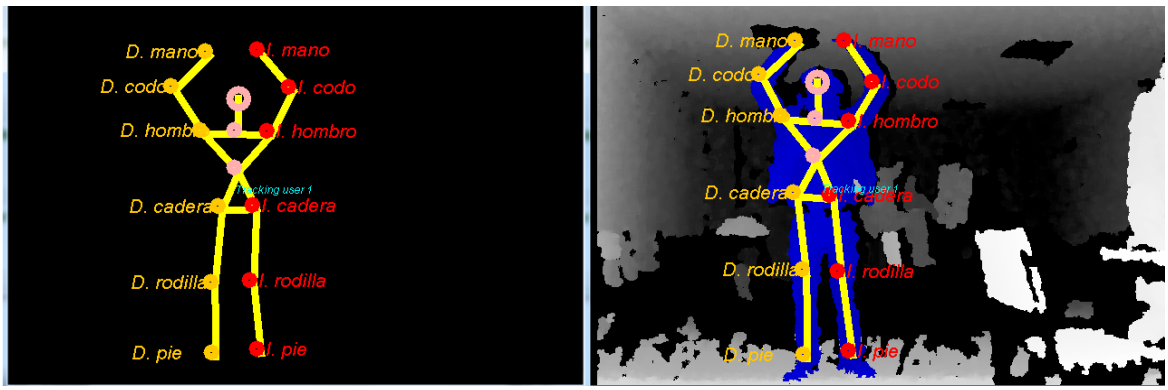


Figura A.7: Puntos detectados para la formación del esqueleto.

de visión del sensor de profundidad (Kinect). Esta representación es en realidad un esqueleto que se puede desplegar superpuesto con la imagen RGB obtenida y que se construye a partir de la unión de los 15 puntos de interés correspondientes a posiciones de algunas articulaciones y partes del cuerpo humano tales como cabeza, cuello, torso, hombros, codos, manos, caderas, rodillas y pies. La obtención de estos puntos se logra mediante el cálculo de un mapa de profundidad usando el concepto de luz estructurada, seguido de la inferencia de la pose del cuerpo humano por métodos de aprendizaje de máquina y de segmentación del cuerpo del usuario [73]. La figura A.7 muestra la distribución de los puntos obtenidos, mismos que se utilizan para crear un esqueleto virtual y que resultan ser de mucha utilidad para determinar los valores de las variables articulares en los miembros superiores que generaron un cierta posición del usuario.

Para efectos de rehabilitación de los miembros superiores sólo es necesaria la captura de 7 puntos (torso, mano izquierda, codo izquierdo, hombro izquierdo, mano derecha, codo derecho y hombro derecho) dados como una terna de coordenadas $[x, y, z]^T$. El Kinect proporciona una estimación de la profundidad de los objetos presentes en la escena, permitiendo obtener la información tridimensional de los puntos de interés con la cual es posible estimar el valor angular de cada una de las articulaciones del paciente que sean de interés en el proceso de rehabilitación. La relación matemática que permite calcular los valores de los ángulos de las articulacuiones a partir de posiciones 3D se conoce como modelo cinemático inverso. Para cadenas cinemáticas en una configuración serie como lo es el brazo o la pierna de una persona se cuenta con metodologías bien conocidas, pero, debido a las particularidades de nuestro problema, se deberá obtener aplicando conocimientos de biomecánica, de cinemática y de geometría a cada una de las articulaciones de interés.

Con base a los puntos detectados con el Kinect es necesario calcular el valor de las cuatro variables articulares de cada brazo del paciente, a fin de poder enviarlas al

humanoide Nao a manera de consignas articulares de movimiento. Esto se realiza en tres pasos:

i) Determinar el modelo de transformación de coordenadas (transformación homogenéa) que permite expresar las coordenadas articulares del modelo cinemático obtenidas con el kinect, con respecto a un referencial asociado al cuerpo del paciente. Esto genera que los cálculos de la cinemática inversa sean invariantes a la postura (pose) del usuario, ya que en muchas ocasiones se presentan dificultades para permanecer perfectamente erguido o en una alineación frontal paralela al sensor Kinect.

ii) Obtener del modelo cinemático inverso del paciente mediante un enfoque vectorial y trigonométrico a fin de identificar las variables articulares de cada brazo y pierna del paciente.

iii) Escalar las coordenadas articulares obtenidas en el paso anterior para obtener las correspondientes coordenadas articulares en el modelo del Robot Nao, las cuales serán enviadas hacia dicho robot como consignas de movimiento.

Referencias

[1] F. Durand A. Levin, R. Fergus and W. Freeman. Image and depth from a conventional camera with a coded aperture. In *International Conference on Computer Graphics and Interactive Techniques ACM SIGGRAPH*, volume 26, 2007.

[2] M. Quinlan & P. Stone A. Setapen. Marionet: Motion acquisition for robots through iterative online evaluative training. In *In 9th International Conference on Autonomous Agents and Multiagent Systems - Agents Learning Interactively from Human Teachers Workshop (AAMAS - ALIHT)*, volume 1, pages 1435–1436, 2010.

[3] N. Abe. A training system using virtual machines for teaching assembling/disassembling operations to novices. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 2096–2101, 1996.

[4] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. In *International Journal of Computer Vision*, volume 56, pages 221–255, 2004.

[5] S. Barsky and M. Petrou. Colour photometric stereo: Simultaneous reconstruction of local gradient and colour of rough textured surfaces. In *8th IEEE International Conference on Computer Vision*, volume 2, pages 600–605, Vancouver, BC, 2001. IEEE.

[6] H. Bay. *From Wide-baseline Point and Line Correspondences to 3D.* PhD thesis, Swiss Federal Institute of Technology, 2006.

[7] P. Belhumeur. A bayesian approach to binocular stereopsis. *International Journal of Computer Vision*, 3(19):237–260, Agosto 1996.

[8] A. Bobick and S. Intille. Large occlusion stereo. In *International Journal of Computer Vision*, volume 3, pages 181–200, 2010.

[9] J. Boughet. Camera calibration toolbox for matlab, california institute of technology. http://www.visioin.caltech.edu/boughetj/calib$_d$oc/*index.html*, 2007.

[10] H. Bowditch and W. Southard. A comparison of sight and touch. In *The Journal of Physiology*, volume 3 of *4*, pages 232–245, 1982.

[11] M. Brown and 2002. D. Lowe. In BMVC. Invariant features from interest point groups. In *British Machine Vision Conference BMVC*, pages 656–665, 2002.

[12] R. Kelly C. Soria, R. Carelli and J. Ibarra. Coordinated control of mobile robots based on artificial vision. In *International Journal of Computers, Communications & Control*, volume 2, pages 85–94, 2006.

[13] N. Chang and A. Zakhor. Arbitrary view generation for three-dimensional scenes from uncalibrated video cameras (1995). In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, pages 2455–2458, 2009.

[14] T. Boubekeur D. Bradley and W. Heidrich. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1063–6919, June 2008.

[15] B. Ladendorf D. Geiger and A. Yuille. Occlusions and binocular stereo. In *European Conference on Computer Vision*, pages 423–435, 1992.

[16] R. Hartley D. Huynh and A. Heyden. Outlier correction in image sequences for the affine camera. In IEEE Computer Society, editor, *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, pages 585–590, Washington, DC, USA, 2003.

[17] D. Min D. Kim, Seoul and K. Sohn. 3d scene reconstruction system with hand-held stereo cameras. In *3DTV Conference*, pages 1–4, 2010.

[18] J. Little D. Murray. Using real-time stereo vision for mobile robot navigation. *Autonomous Robots*, 8(2):161–171, 2000.

[19] S. Kalyanakrishnan Y. Bentor D. Urieli, P. MacAlpine and P. Stone. On optimizing interdependent skills: A case study in simulated 3d humanoid robot soccer. In *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 2, pages 769–776, 2011.

[20] K. Dautenhahn and B. Robins. The aurora project. http://www.aurora-project.com, 2003.

[21] S. Rojas E. Atoche, L. Muñoz and A. Espinosa. Introducción a la inicialización y de pose en visión artificial. *5o. Congreso Mexicano de Robótica COMRob*, pages 159–164, 2003.

[22] N. Garcia R. Morales E. Fernández, J. Sabater and F. Badesa. Aupa project grupo de investigación nbio. http://nbio.umh.es/robot-aupa, 2014.

[23] J. Ibarra E. Hernández, R. Cisneros and J. Lavín. Reconstrucción 3d monocular para el sistema de slam visual de un robot humanoide. In *10o. Congreso Mexicano de Robótica COMRob*, México, D.F., 2008.

[24] M. Trujano J. Ibarra E. Hernández, R. Cisneros and J. Lavín. Proposición de una técnica de análisis de imágenes para generar los vectores de medición clásicos usados en slam. In *6o. Congreso Internacional en Innovación y Desarrollo Tecnológico (CIINDET)*, volume 6, pages 178–186, Cuernavaca, México, 2008.

[25] H. Farid. *Range Estimation by Optical Diferentiation.* PhD thesis, University of Pennsylvania, 1997.

[26] O. Faugeras. *Three-Dimensional Computer Vision : A geometric Viewpoint.* The MIT Press, 1993.

[27] O. Faugeras and Q. Luong. *The Geometry of Multiple Images: The Laws That Govern the Formation of Multiple Images of a Scene and Some of Their Applications.* The MIT Press, 2001.

[28] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. In IEEE Computer Society, editor, *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, pages 951–956, Washington, DC, USA, 1995.

[29] O. Faugeras and T. Papadopoulo. Grassmann-cayley algebra for modeling systems of cameras and the algebraic equations of the manifold of trifocal tensors. Technical report, INRIA Sophia-Antipolis, France, 1997.

[30] P. Fua. Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In *International Joint Conferences on Artificial Intelligence*, pages 1292–1298, 1991.

[31] G. González and H. Arroyo. *Accidente cerebrovascular en la infancia y adolescencia.* Buenos Aires, 2011.

[32] M. Goodrich and A. Schultz. Human-robot interaction: A survey. *Foundations and Trends in Human Computer Interaction*, 1(3):203–275, 2007.

[33] A. Gómez. Control de robots móviles usando visión. Master's thesis, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV), México, D.F., 2008.

[34] M. Aisen H. Krebs, N. Hogan and B. Volpe. Robot-aided neurorehabilitation. *IEEE Transactions on Rehabilitation Engineering*, 6(1):75–87, 1998.

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

[36] R. Hartley. Euclidean reconstruction from uncalibrated views. In Springer Berlin Heidelberg, editor, *Second Joint European - US Workshop on Applications of Invariance in Computer Vision*, Lecture Notes in Computer Science, pages 235–256, Ponta Delgada, Azores, Portugal, 1994.

[37] R. Hartley. Lines and points in three views and the trifocal tensor. *International Journal of Computer Vision*, 2(22):125–140, 1997.

[38] R. Hartley. Minimizing algebraic error. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, pages 469–476, DC, USA, 1998.

[39] R. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 2(68):146–157, 1997.

[40] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2003.

[41] A. Heyden. *Geometry and algebra of multiple projective transformations*. PhD thesis, Lund Institute Technology, Sweden, 1995.

[42] S. Rao I. Cox, S. Hingorani and B. Maggs. A maximum likelihood stereo algorithm. In *Computer Vision and Image Understanding*, volume 3, pages 542–567, May 1996.

[43] J. Ibarra. Notas del curso de visión artificial para robots. Technical report, Centro de Investigación y de Estudios Avanzados, México, D.F., 2005.

[44] J. Ibarra and R. Cisneros. Robótica y realidad virtual proyecto opensurg cyted. Technical report, CINVESTAV IPN, México, D.F., 2010.

[45] J. Ibarra and E. Iturbe. Reconstrucción 3d monocular basada en cosenos directores. In *6o. Congreso Mexicano de Robótica COMRob*, volume 7, pages 789–805, Torreón, Coahuila, 2004.

[46] S. Intille and A. Bobick. Disparity-space image and large occlusion stereo. In *European Conference on Computer Vision*, volume 2, pages 179–186, May 1994.

[47] E. Iturbe. Generación de mapas para la navegación de robots móviles. Departamento de control automático, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV), México, D.F., 2004.

[48] E. Iturbe and J. Ibarra. Generación de mapas virtuales interactivos para robots móviles usando la reconstrucción 3d de kanatani. In *5o. Congreso Mexicano de Robótica COMRob*, pages 182–188, 2003.

[49] E. Perepezko H. Krebs J. Rogers K. Goyal M. Dohring E. Fredrickson J.Ñethery J. Daly, N. Hogan and R. Ruff. Response to upper-limb robotics and functional neuromuscular stimulation following stroke. *Journal of Rehabilitation Research & Development*, 42(6):723–736, 2005.

[50] R. Cisneros J. Ibarra, J. Lavín and E. Hernández. Reconstrucción 3d monocular para el sistema de slam visual de un robot humanoide. *Memorias del X Congreso Mexicano de Robótica AMROB*, 2008.

[51] R. Cisneros U. Zaldivar C. León S. Guerrero X. Zaldivar D. Murillo J. Ibarra, J. Lavin and C. Martínez. Seguimiento visual de las manos y dedos para actualizar un mundo virtual interactivo. In *XIV Convención de Ingeniería Eléctrica*, volume 1, Villa Clara, Cuba, 2011.

[52] D. Koller S. Steuart J. Zhu, G. Humphreys and W. Rui. Fast omnidirectional 3d scene acquisition with an array of stereo cameras. In *In Proc. Sixth International Conference on3D Digital Imaging and Modeling*, pages 217–224, 2009.

[53] S. Kobashi K. Kondo and Y. Hata. A real-time analysis of 3d scene from monocular images by observing known background. In *Intelligent Signal Processing and Communication Systems*, pages 5–8, 2011.

[54] K. Kanatani. *Group Theorical Methods in Image Understanding.* Springer, Verlag, 1990.

[55] J. Knight and I. Reid. Self-calibration of a stereo rig in a planar scene by data combination. In *15th Int Conf on Pattern Recognition ICPR*, pages 1411–1414, Barcelona, Spain, 2000.

[56] J. Koenderink. The structure of images. In *Biological Cybernetics*, volume 50, pages 363–370, 1984.

[57] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions via graph cuts. In *International Conference on Computer Vision*, pages 508–515, 2001.

[58] K. Konolige. Small vision systems: Hardware and implementation. In *In Proceedings of the International Symposium on Robotics Research*, pages 203–212, 2011.

[59] J. Lavín. Reconstrucción 3d. Master's thesis, CINVESTAV IPN, México, D.F., 2009.

[60] T. Lindeberg. Feature detection with automatic scale selection. In *International Journal of Computer Vision IJCV*, volume 2, pages 79–116, 1998.

[61] T. Lindeberg and L. Bretzner. Real-time scale selection in hybrid multi-scale representations. Technical report, Royal Institute of Technology, Stockholm, Sweden, 2003.

[62] J. Lizárraga. Desarrollo y evaluación de una técnica de interacción para el agarrado y manipulación de objetos en ensamble virtual. Master's thesis, Universidad de Occidente, Mazatlán, México, 2010.

[63] D. Lowe. Distinctive image features from scale-invarian keypoints. In *International Journal of Computer Vision*, volume 60, pages 91–110, 2004.

[64] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *7th International Joint Conference on Artificial Intelligence IJCAI '81*, pages 674–679, 1981.

[65] T. Choi M. Asif. Shape from focus using feedfordward neural networks. *Image Processing, IEEE Transactions on*, 10(11):1670–1675, 2000.

[66] C. Rother M. Bleyer and P. Kohli. Surface stereo with soft segmentation. In *Computer Vision and Pattern Recognition CVPR*, pages 1570–1577, 2010.

[67] T. Engelke M. Humenberger and W. Kubinger. A census-based stereo vision algorithm using modified semi-global matching and plane fitting to improve matching quality. In *Computer Vision and Pattern Recognition Workshops CVPRW*, pages 77 – 84, 2010.

[68] E. Malis and R. Cipolla. Camera self-calibration from unknown planar structures enforcing the multiview constraints between collineations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(24):1268–1272, 2002.

[69] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *International Conference on Computer Vision ICCV*, volume 1, pages 525–531, 2001.

[70] J. Márquez. Sistema de realidad aumentada para la rehabilitación después de una apoplejía. Master's thesis, Instituto Tecnologico de Ciudad Madero, 2011.

[71] D. Pizarro N. Pastén and C. Tozzi. Reconstrucción de objeto 3d a partir de imágenes calibradas. *Ingeniare: Revista chilena de ingeniera*, 2(15):158–168, 2007.

[72] S.Ñayar and Y.Ñakagawa. Shape from focus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(8):824–831, August 1994.

[73] OpenNI. Openni official programmer's guide. http://www.openni.org/openni-programmers-guide/, 2010.

[74] H. Ortega and E. Martínez. Wired gloves for everyone. *Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology*, pages 305–306, 2008.

[75] A. Zisserman P. Beardsley and D. Murray. Navigation using affine structure from motion. In Springer-Verlag, editor, *ECCV '94: Proceedings of the Third European Conference on Computer Vision*, volume 2, pages 85–96, Secaucus, NJ, USA, 1994.

[76] A. Zisserman P. Beardsley and D. Murray. Sequential updating of projective and affine structure from motion. In *Applications of Invariance in Computer Vision, Second Joint European-US Workshop*, volume 3, pages 235–259, 1997.

[77] C. Balaguer R. Aracil and M. Armada. Robots de servicio. *Revista Iberoamericana de Automática e Informática Industrial*, 5(2):6–13, 2008.

[78] R. Roldán R. Cisneros, D. Hernández and J. Ibarra. Análisis cinemático y simulación dinámica de una órtesis para la rehabilitación de la apoplejía. *VI Congreso Iberoamericano deTecnologías de apoyo para la discapacidad*, 2011. Palma de Mallorca, España.

[79] G. Pajaresa R. Correala and J. Ruzb. Stereo images matching process enhancement by homomorphic filtering and disparity clustering. In *Revista Científica de América Latina, el Caribe, España y Portugal*, volume 16, pages 485–500, 2013.

[80] A. Botía R. Satorre, P. Compañ and R. Rizo. Estimación de disparidad en visión estereoscópica mediante la integración de diversas técnicas combinadas con multiresolución. Technical report, Universidad de Alicante. Departamento de Ciencia de la Computación e Inteligencia Artificial, 2003.

[81] K. Murao S. Derrouich, K. Izumida and K. Shiiya. The use of cnn models and vertical rectification for a direct trigonometric recovery of 3d scene geometry from a stream of images. In *Signals, Systems and Computers, 2008. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 2, pages 1600–1604, 2009.

[82] C Heman S. Ledesma and I. Sidelnik. Hologramas digitales de fase con información estereoscópica para visión tridimensional. Technical report, Laboratorio de Procesado de Imágenes, Departamento de Física, FCEyN, UBA, 2009.

[83] A. Saxena and S. Chung. 3d depth reconstruction from a single still image. *International Journal of Computer Vision*, 78(23):143–167, July 2008.

[84] Y. Shirai. *Three-dimensional Computer Vision*. Springer-Verlag, 1987.

[85] Y. Shirai and M. Suwa. Recognition of polyhedrons with a range finder. In *2nd. International Joint Conference on Artificial Intelligence*, pages 80–87, London, England, September 1971.

[86] R. Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science, 1st edition, 2010.

[87] K. Dautenhahn T. Salter and R. Boekhorst. Learning about natural human-robot interaction styles. *Robotics and Autonomous Systems*, 54(2):127–134, 2006.

[88] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 2(9):137–154, 1992.

[89] P. Torr and A. Zisserman. Robust parameterization and computation of the trifocal tensor. *Image and Vision Computing*, (15):591–607, 1997.

[90] B. Triggs. Matching constraints and the joint image. In IEEE Computer Society, editor, *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, pages 338–343„ Washington, DC, USA, 1995.

[91] E. Trucco and A. Verri. *Introductory Techniques for 3D Computer Vision*. Prentice Hall, 1998.

[92] S. Guerrero V. Fregoso C. Zaldívar D. Murillo U. Zaldívar, C. León and C. Martínez. Sistema de transmisión y captura de movimientos de los dedos de la mano. *Memorias del XII Congreso Mexicano de Robótica AMROB*, pages 238–242, 2010.

[93] P. Monasse V. Kolmogorov and P. Tan. Kolmogorov and zabih's graph cuts stereo matching algorithm. In *Image Processing On Line*, volume 4, pages 220–251, 2014.

[94] P. Varley and R. Martin. A system for constructing boundary representation solid models from a two-dimensional sketch. geometric finish fronta. 1st Korea-UK Joint Workshop on Geometric Modeling and Computer Graphics, 2010.

[95] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511–518, 2001.

[96] P. Will and Pennington. Grid coding: A preprocessing technique for robot and machine vision. In *2nd. International Joint Conference on Artificial Intelligence*, pages 66–68, London, England, September 1971.

[97] A. Witkin. Scale-space filtering. In *International Joint Conference on Artificial Intelligence*, pages 1019–1022, 1983.

[98] P. Wolf and B. Dewwit. *Elements of Photogrammetry with Applications in GIS*. McGraw Hill, 2000.

[99] R. Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineering*, 1(19):139–144, 1980.

[100] O. Veksler Y. Boykov and R. Zabih. Fast approximate energy minimization via graph cuts. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 11, pages 1222–1239, 2001.

[101] J. Koecka Y. Ma, S. Soatto and S. Sastry. *An Invitation to 3D Vision: From Images to Geometry Models*. Springer Verlag, 2003.

[102] W. Liu Y. Ma and Y. Sun. A robust self-calibration algorithm based on three views. *CIT*, pages 741–746, 2004.

[103] K. Yokoi. La robótica japonesa, presente y futuro. http://portal.uc3m.es/portal/page/portal, 2008.

[104] Z. Zhang. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 2(27):161–195, 1998.

[105] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. *7th IEEE International Conference on Computer Vision, ICCV'99*, 1:666–673, 1999.