Universidad de las Ciencias Informáticas
Vertex, Entornos Interactivos 3D, Facultad 5

Componente de seguimiento estereoscópico para software de neuronavegación quirúrgica

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: José Lozano Hernández

Tutores: Ing. Mileydi Moreno Mirabal

MSc. Ernesto de la Cruz Guevara Ramírez

La Habana, 2015

Lo que sabemos es una gota de agua; lo que ignoramos es el océano
Isaac Newton

Dedicatoria

A mis padres, a mi hermano y a mis abuelos.

Agradecimientos

Le agradezco a mis tutores Mileydi y Ernesto porque es de ellos la idea inicial de este trabajo y su apoyo ha tenido un valor infinito,
a mis amigos Leiser, Yuriasky, Nelson, Jorge Luis y Alejandro,
a mi novia Tania, por ser mi pilar fundamental todo este tiempo,
a Joaquín y Gabriel por su valiosa ayuda,
a mis amistades que me acompañan desde Cienfuegos, en especial a Yanislay, a Rosali y a Nelson,
a mis amistades Omar, Roberto, Antuannet, Héctor, Yosbani y Marco,
a mis compañeros del aula y de la beca, que fueron parte de mi vida en estos cinco años,
a los compañeros de la banda que me han apoyado en todo momento,
a todos los profesores que me han preparado desde la primaria hasta ahora, en especial a Isnel, Estela, Adián, Yaxiel, Yidián, Adriana y Marvyn,
a mi excelente tribunal de tesis conformado por Zaida, Susej y Andy, y mi oponente José Andrés, por contribuir con la mejora del trabajo,
a Muchi, a Cachita, a Panchito, a Talia y a Iraida por acogerme con calidez en su hogar y tratarme como uno más de la familia,
a las amistades eternas de la familia Monteagudo, Wilfredo y Marín,
y el mayor agradecimiento de todos a mi familia, en especial a mis padres, sin su sacrificio nada de esto fuera posible, ellos son las personas que soportan todo por el bien de sus hijos,
a mis abuelos, qué más sacrificio que el de ellos?, son la raíz de la familia y se sienten responsables por todos,
a mi tío Bohdan, después de mis padres es mi ejemplo a seguir,
y a mi hermano, que me va a superar.

Declaración de autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre esta, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

José Lozano Hernández
Autor

Ing. Mileydi Moreno Mirabal
Tutora

MSc. Ernesto de la Cruz Guevara Ramírez
Co-tutor

Resumen

Con el uso de la cirugía asistida por computadora, surgen tecnologías de asistencia en las intervenciones. El neuronavegador es una de las herramientas que más demanda tiene por los neurocirujanos, para la visualización de las imágenes médicas y el seguimiento de la posición de instrumentos quirúrgicos en una cirugía. La presente investigación comienza a raíz de una propuesta planteada por la Clínica Central Cira García a la Universidad de las Ciencias Informáticas, de construir un sistema de este tipo, alternativo a los existentes en el mercado, que por sus altos costos son imposibles de adquirir. En el año 2012 se desarrolló un prototipo de neuronavegador como parte de un trabajo de tesis en la Facultad 5, que utilizó un sistema de seguimiento monocular, la precisión alcanzada no satisface las necesidades para la neurocirugía, pero demostró que es posible el desarrollo de estos sistemas. Este trabajo se enfoca en la problemática de detectar en tiempo interactivo la posición y orientación espacial de un objeto y se plantea como objetivo implementar un componente de seguimiento estereoscópico para software de neuronavegación quirúrgica. Para su desarrollo se utilizó las bibliotecas IGSTK, ArUco, OpenCV y el marco de trabajo Qt. El componente desarrollado es capaz de adquirir la orientación y posición espacial de varios marcadores visibles para ambas cámaras. Las pruebas realizadas, demuestran que el uso de las técnicas estereoscópicas mejoran considerablemente la precisión.

Palabras clave: neuronavegador, sistema de seguimiento, visión estéreo.

Índice general

Índice de figuras

Índice de tablas

Introducción

El proceso de cirugía en ocasiones es complejo, algunas intervenciones pueden durar todo el día y demandar la presencia de varias personas en el quirófano. La planificación de estas cirugías es un proceso delicado tan importante como la ejecución en sí, en ocasiones el cirujano debe abandonar la acción para observar las imágenes médicas y hacerse un mapa mental del área comprometida. El desarrollo de la ciencia y la técnica ha favorecido a esta rama de la medicina, y la cirugía asistida por computadora es una tendencia que se está generalizando en todos los hospitales por las ventajas que aporta. El neuronavegador es una herramienta de asistencia para los neurocirujanos con la cual pueden visualizar las imágenes médicas y observar la posición de instrumentos quirúrgicos dentro del paciente en una cirugía, lo cual es imposible a simple vista. Estos equipos brindan un elevado margen de seguridad a los cirujanos, lo que facilita el tratamiento quirúrgico de lesiones cerebrales. Dada la evolución de esta tecnología, existe una alta demanda por parte de los cirujanos y los pacientes, lo cual provoca que cada vez existan menos intervenciones que no usen esta tecnología.

La gran desventaja para la adquisición de estos equipos es su alto costo, las empresas privadas que los desarrollan se aprovechan de la suficiencia de esta rama elitista de la medicina y proveen un conjunto completo de hardware y software, que muchos países subdesarrollados no son capaces de adquirir, precio dado mayormente por el software.

El desarrollo alcanzado en las ciencias informáticas en Cuba, posibilita la construcción de un sistema de neuronavegación, que podrá distribuirse por los hospitales del país y en las colaboraciones médicas. Por las carencias económicas existentes, es necesario desarrollar el sistema con materiales económicamente viables y médicamente fiables.

En virtud de lo antes expuesto hasta aquí y debido a la necesidad de utilizar sistemas de navegación guiados por imágenes, la Clínica Central Cira García ha propuesto a la Universidad de las Ciencias Informáticas la fabricación de un neuronavegador guiado por imágenes. Esta decisión está sustentada por la experiencia de algunos neurocirujanos en hospitales europeos, que utilizan estos sistemas como procedimiento de rutina para brindar a los pacientes servicios de elevada calidad.

Un neuronavegador tiene tres componentes fundamentales. El primero se encarga de visualizar las imágenes médicas tomadas por una tomografía, radiología u otra técnica, y brindar un mecanismo de navegación por ellas en el software. El segundo componente, el cual es objeto de la investigación en cuestión, es el sistema de seguimiento tridimensional. Este componente tiene la tarea de rastrear la posición y orientación espacial del instrumento quirúrgico usado por el cirujano, tanto fuera como dentro de las estructuras anatómicas del paciente. Por la complejidad que representa la zona en la que operan los neurocirujanos, donde existen probabilidades de generar secuelas e incluso causar la muerte, el componente debe ser robusto y preciso, pues debe brindar al sistema las coordenadas más cercanamente posibles a las reales del instrumento que se está siguiendo, y en un tiempo interactivo cercano al tiempo real. El tercer componente es el encargado de tomar la información tridimensional adquirida por el componente de seguimiento y correlacionarlas con las imágenes médicas (ENQUOBAHRIE y YANIV, 2009).

En el centro Vertex de la Facultad 5, se tiene conocimiento en áreas involucradas como la visión por computadora y se han realizado trabajos que incluyen seguimiento de objetos en imágenes. En el año 2012, se realizó un prototipo de neuronavegador en la facultad. El sistema de seguimiento utilizado alcanzó un error promedio de 4,17 milímetros aproximadamente. En la neurocirugía un error de 1 milímetro es inadmisible para operar en ambientes reales. Sin embargo, se probó que es posible construir sistemas como éste, pero con la utilización de otras técnicas para alcanzar mayor precisión.

Teniendo en cuenta lo anteriormente expuesto, se plantea el siguiente **problema de investigación**: ¿cómo detectar en tiempo interactivo la posición y orientación espacial de un objeto?

La investigación tiene como **objeto de estudio**: seguimiento 3D de objetos basado en visión por computadora, constituyendo su **campo de acción**: las técnicas de seguimiento 3D de objetos mediante visión estereoscópica.

Para darle solución al problema, se propone el siguiente **objetivo general**: implementar un componente de seguimiento estereoscópico para software de neuronavegación quirúrgica.

Para resolver el problema científico y darle cumplimiento al objetivo general, se plantearon las siguientes **tareas de investigación**:

1. Analizar la bibliografía referente a las técnicas estereoscópicas utilizadas para constituir el marco teórico.
2. Caracterizar las diferentes técnicas de seguimiento utilizadas en los neuronavegadores para seleccionar la más conveniente a la propuesta de solución.
3. Analizar posibles bibliotecas para el desarrollo de software de navegación guiada por imágenes que se puedan utilizar para la implementación de la solución.

4. Implementar un componente de software capaz de realizar el seguimiento de un objeto basándose en una técnica previamente analizada.
5. Realizar pruebas para comprobar la precisión lograda con la solución.

El proceso de investigación y elaboración del presente trabajo involucra los siguientes métodos científicos de investigación:

Métodos teóricos:

- Histórico - lógico: Se estudian los antecedentes y las tendencias de los sistemas de seguimiento estereoscópicos que existen, además de los conceptos, vocabularios y términos del campo, que contribuyen al mejor entendimiento del trabajo.
- Analítico - sintético: Analizar teorías, documentos e información referentes a sistemas de neuronavegación con el objetivo de extraer los elementos esenciales que se relacionan con el objeto de estudio.
- Inductivo - deductivo: Luego de adquirir una serie de conocimientos referentes a la geometría que involucra la estereoscopía, se pueden deducir ideas que pueden ser aplicadas en el problema a tratar.

Métodos empíricos:

- Revisión de la bibliografía en todo tipo de fuentes, para conformar el marco teórico de la investigación.
- Entrevistas a personas especialistas en el tema de seguimiento de objetos y estereoscopía, para obtener información sobre el tema de la investigación, así como para la validación de los resultados esperados.

Esta tesis está estructurada de la siguiente forma:

Capítulo 1: Fundamentación teórica. Se indican las bases teóricas fundamentales de los sistemas estereoscópicos. Se refleja el estado del arte del tema de investigación y se explican conceptos y modelos matemáticos necesarios para el desarrollo de estos sistemas. Se identifican los componentes del sistema de seguimiento estereoscópico basado en imágenes que se utilizarán en la solución.

Capítulo 2: Propuesta de solución. Se explica la elección de los dispositivos y la técnica de seguimiento a usar para la solución propuesta. Se explican los pasos a seguir para realizar el seguimiento estereoscópico. Se realiza la ingeniería del componente y se definen los artefactos necesarios que pertenecen a las fases de planificación y diseño.

Capítulo 3: Implementación y pruebas. Se definen las tareas de ingeniería y las pruebas al componente que corresponden a las fases de implementación y pruebas dentro del desarrollo del software.

CAPÍTULO 1

Fundamentación teórica

En el presente capítulo se abordan los principales elementos teóricos que conforman los componentes de seguimiento de un neuronavegador, centrándose en los dispositivos ópticos basados en grabación de video. Se explican distintos enfoques y técnicas en el seguimiento visual de objetos. Se describen las etapas del proceso de visión estereoscópica y la calibración de las cámaras. Por último se definen las herramientas informáticas necesarias para cumplir con el objetivo de la investigación.

1.1. Cirugía asistida por computadora

La cirugía asistida por computadora (CAS) tiene como objetivo reducir al máximo el trauma terapéutico, al reducirse el tiempo en las operaciones y mejorar en eficacia, seguridad y precisión mediante la introducción de técnicas alternativas a las utilizadas en la cirugía tradicional. La CAS utiliza herramientas informáticas que le permiten al cirujano mejorar su rendimiento, tanto en las operaciones como en el diagnóstico y planificación de las cirugías. En la actualidad la navegación guiada por imágenes es una de las técnicas de última generación usada por cirujanos (MARTÍNEZ, 2012).

1.2. Navegación guiada por imágenes

La navegación, por definición, es el proceso de determinar y mantener un camino o trayectoria hacia una ubicación u objetivo. En medicina, se adopta el término, para expresar la orientación espacial respecto a un volumen anatómico (ibíd.). La navegación guiada por imágenes tiene como objetivo: tomar las coordenadas

de instrumentos quirúrgicos durante una operación y mostrar su posición relativa en las imágenes médicas del paciente para guiar al cirujano en zonas de difícil acceso y visibilidad. El neuronavegador es la herramienta más demandada por los neurocirujanos.

La presente investigación se enfoca en el componente de seguimiento del neuronavegador y se explica detalladamente a continuación.

1.3. Componente de seguimiento de un neuronavegador

Los componentes de seguimiento o de *tracking*, tienen como objetivo la localización tridimensional. En el campo quirúrgico los objetos rastreados son los instrumentos usados por los cirujanos. Estos sistemas progresan constantemente en número, fiabilidad y facilidad de uso. Existen diferentes sistemas de este tipo, pero se clasifican en dos grupos, los ópticos y los no ópticos, cada uno con sus ventajas y desventajas, la diferencia radica en que los dispositivos ópticos requieren una línea de visión directa entre emisores y receptores, por lo que son afectados por la presencia de obstáculos que impidan la visualización del objetivo a seguir (ENQUOBAHRIE y YANIV, 2009) y (MARTÍNEZ, 2012).

A continuación se describen los diferentes componentes de seguimiento usados en la neuronavegación: marco estereotáxico, brazo mecánico, ultrasonidos, magnetismo, luz infrarroja, grabación de video y láser.

1.3.1. Marco estereotáxico

Este es el primer método de seguimiento y el de mejor influencia y longevidad en la historia del desarrollo de la navegación quirúrgica. Este método consiste en una estructura rígida (marco estereotáxico), que es firmemente anclada al cráneo del paciente, lo que impide el movimiento relativo entre el marco y el paciente (ibíd.).

La principal ventaja de este dispositivo es la precisión, se han alcanzado niveles de fracciones de milímetros, pero como desventajas tiene la obstrucción de la cirugía por el marco y los sistemas de guía, solo se puede calcular un número limitado de objetivos quirúrgicos, falta de maniobrabilidad cuando el instrumento avanza en el campo quirúrgico, el paciente debe someterse al procedimiento de toma de imágenes, transportación y cirugía con el marco fijado al cráneo, por lo que aumenta la invasividad del procedimiento, el tiempo y riesgo en cirugía.

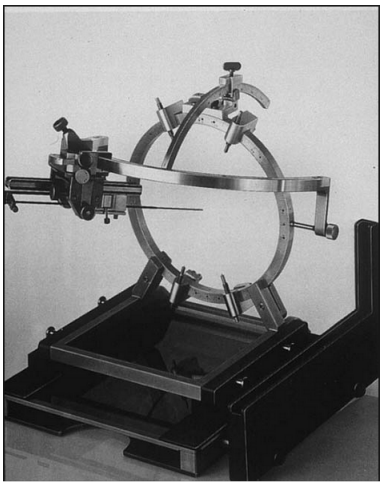


Figura 1.1. Marco estereotáxico

1.3.2. Brazo mecánico

Este consiste en un brazo articulado con medidores de ángulos en las articulaciones, de modo que el componente conoce las coordenadas de un puntero colocado al extremo distal del brazo articulado. Este método mejora la obstrucción del campo quirúrgico causado por el marco estereotáxico, pero su precisión es limitada comparada con el anterior y es difícil manipular el brazo en estructuras profundas (MARTÍNEZ, 2012).

1.3.3. Ultrasonido

Se utilizan sensores de ultrasonidos. Al instrumento quirúrgico se le fija un emisor de ultrasonido y se colocan sensores en el quirófano que determinan la posición del instrumento por medio de triangulación (SEHNERT, 2008), calculando las diferencias de tiempo en que demora en llegar el ultrasonido a los sensores. La ventaja de estos dispositivos, es que no requieren de una línea de visión directa entre emisores y sensores, pero como inconveniente se tienen la susceptibilidad a las interferencias del ruido ambiental y a los cambios de temperatura (BUCHOLZ y SMITH, 1993).

1.3.4. Magnetismo

Estos dispositivos son relativamente nuevos en aplicaciones médicas. Su principal ventaja, al igual que en los dispositivos basados en ultrasonidos, es que no requieren de una línea directa de visión, pero tienen como desventaja la susceptibilidad a la distorsión causada por metales cercanos y la precisión limitada comparada con los dispositivos ópticos (PETERS y CLEARY, 2008).

1.3.5. Ópticos

Los dispositivos ópticos son los que más éxito han tenido en el ambiente clínico por su alta precisión y fiabilidad. Su uso se ha estandarizado en muchas clínicas, aunque tienen como desventaja que precisan de una línea de visión limpia de obstáculos. También se aplican en la terapia de radiación de alta precisión para enfermedades de la retina y corrección de movimientos en la reconstrucción tomográfica (MARTÍNEZ, 2012) y (PETERS y CLEARY, 2008).

Los tipos de sistemas ópticos son:

- **Basados en grabación de video**: estos identifican patrones conocidos en secuencias de video usualmente tomadas por una o varias cámaras. Ejemplo de un componente comercial de este tipo es el Claron Tracker (*Claron Technology Inc.* s.f.). Teniendo en cuenta la pérdida de información que presenta la proyección perspectiva, los dispositivos usados en sistemas de neuronavegación guiados por imágenes, poseen más de una cámara para realizar cálculos más precisos. Estos dispositivos operan en el espectro de luz visible.
- **Basados en infrarrojos**: estos tienen la ventaja que eliminan toda la luz ambiental de otras longitudes de ondas, haciendo la identificación de los marcadores más simple y confiable. Dos tipos de dispositivos de este tipo son los activos y los pasivos. Los activos consisten en la ubicación de dos o más cámaras receptoras de luz infrarroja en el quirófano. Las cámaras receptoras detectan la luz infrarroja emitida por diodos emisores de luz (LED) fijados a los instrumentos quirúrgicos, luego por triangulación es posible determinar la localización espacial del instrumento. Los pasivos funcionan de manera similar, la diferencia radica en que las cámaras infrarrojas ubicadas en el quirófano son las encargadas de emitir la luz, y a las herramientas se fijan marcadores pasivos que reflejan la luz que reciben, lo que permite conocer su posición mediante triangulación (*Northern Digital* s.f.).
- **Basados en láser**: La posición del cuerpo se estima mediante el muestreo de diferentes haces de luz láser y un foto sensor. Este método no es muy utilizado, se prefieren los anteriormente explicados por su facilidad de uso y fiabilidad (PETERS y CLEARY, 2008).

Según las características de todos estos dispositivos, los ópticos basados en grabación de video son los más fáciles de adquirir en el mercado. Una solución que utiliza estos dispositivos puede ser probada con cámaras de baja calidad, pero sigue siendo compatible con cámaras de gama alta. En la siguiente sección se abordan algunos trabajos relacionados con prototipos de navegación guiados por imágenes y los sistemas de seguimiento utilizados respectivamente.

1.4. Trabajos relacionados

Los sistemas de neuronavegación guiados por imágenes se vuelven cada vez más imprescindibles en los quirófanos. Algunas empresas privadas como (*Claron Technology Inc.* s.f.), (*Northern Digital* s.f.) y (*Medtronic* s.f.) proveen esta clase de sistemas, incluyendo hardware y software.

Existen varios trabajos de prototipos de navegadores guiados por imágenes realizados por grupos de desarrollo que buscan una alternativa a los sistemas de estas empresas. En Uruguay está en desarrollo un neuronavegador y se utiliza las bibliotecas libres Imaged-Guided Surgery Toolkit (IGSTK), Insight Segmentation and Registration Toolkit (ITK) y Visualization Toolkit (VTK). Como dispositivo de seguimiento utilizan el *Polaris Spectra* de (*Northern Digital* s.f.). Este dispositivo cuenta con dos cámaras infrarrojas. Las herramientas que son rastreadas son esferas y discos reflectantes (CARBAJAL; GOMEZ; PEREYRA et al., 2010).

Otro trabajo, que no incluye la neurocirugía pero es válido mencionar, es el desarrollado por (NICOLAU; GOFFIN y SOLER, 2005). Este equipo desarrolló un navegador para cirugía laparoscópica. Para realizar el seguimiento utilizaron dos cámaras a color de resolución 768×576 que se calibraban previamente. Las herramientas rastreadas son marcadores extendidos de ARToolKit (*Open Source Augmented Reality SDK* s.f.) impresos en papel.

Una solución similar se presentó por (TREVISAN; NEDEL y MACQ, 2008) para cirugía maxilofacial. Estos usaban cámaras estereoscópicas y marcadores impresos personalizados de forma circular con cuadros en blanco y negro para detectar las esquinas fácilmente.

(GÓMEZ y RANDALL, 2004) construyeron un prototipo de neuronavegador utilizando cámaras estereoscópicas de bajo costo. La herramienta rastreada, es un marcador personalizado que cuenta con varias esquinas formando un hexágono. Este marcador es detectado utilizando los triángulos en su interior.

El trabajo desarrollado por (MARTÍNEZ, 2012) demostró que es posible desarrollar este tipo de sistemas en la universidad. Teniendo en cuenta los trabajos relacionados, se identificó el uso del seguimiento óptico basado en grabación de video, la visión estereoscópica y el uso de marcadores como las técnicas más usadas en los sistemas de seguimiento alternativos. Además se reafirmó el uso de las bibliotecas libres IGSTK, ITK y VTK para obtener una solución robusta.

Existen varias técnicas para realizar el seguimiento o seguimiento visual de objetos.

1.5. Seguimiento visual de objetos

Cuando se tiene un video, a diferencia de imágenes aisladas, se desea a veces seguir algunos objetos en particular que se encuentran en la escena. La detección y clasificación de objetos, son pasos que anteceden al seguimiento del objeto en una secuencia de imágenes o *frames*. El seguimiento de objetos, se realiza monitorizando los cambios espaciales y temporales durante una secuencia de video, incluyendo su presencia, posición, tamaño, forma y orientación. El seguimiento de objetos es usado en video vigilancia, visión robótica, monitorización de tráfico, animación, deportes como el fútbol y el tenis (ENDRESEN, 2010).

Algunos aspectos a tener en cuenta en el seguimiento de objetos son: la pérdida de información con la proyección perspectiva del mundo real 3D en la imagen 2D, ruido en la imagen, movimiento del objeto difícil de seguir, oclusión parcial o total del objeto, estructuras complejas del objeto (PAREKH; THAKORE y JALIYA, 2014). En el artículo de (JACOB y ANITHA, 2012) se comparan varias técnicas de seguimiento de objetos.

Los pasos básicos para el seguimiento de objetos son los siguientes:

1. *Selección de características para el seguimiento*: Seleccionar las características adecuadas es fundamental. La mejor propiedad para las características visuales es la unicidad, así el objeto es distinguido fácilmente en el entorno. Algunas de las características más usadas son: los puntos de interés, el color, las aristas, el flujo óptico, la textura. Es común, que estas sean elegidas por el usuario en dependencia de la aplicación.

2. *Detección del objeto*: Existen diversas técnicas para detectar objetos. Estas son las basadas en puntos, la diferenciación de marco, flujo óptico, substracción de fondo y segmentación. Estas se explicarán en la siguiente sección.

3. *Seguimiento de objetos*: El objetivo del seguimiento es crear la trayectoria del objeto en el tiempo, localizando su posición en cada *frame* del video. Hay dos metodologías distintas para enfocar el problema del seguimiento, de arriba hacia abajo (*hacia adelante*) y de abajo hacia arriba (*de vuelta*). Los métodos de arriba hacia abajo están orientados a encontrar la posición de los objetos en el *frame* actual, usando hipótesis generadas al comienzo del seguimiento basadas en representaciones paramétricas del blanco u objetivo. Estos métodos usan diferentes características de los objetos como color, textura, forma y movimiento. Un método popular en esta categoría es el cambio medio (*mean-shift*). Por otra parte, en el enfoque hacia arriba, los objetos en movimiento son detectados en cada *frame* y se establece una correspondencia con los objetos detectados en *frames* anteriores. Dentro de esta categoría entran el modelado y substracción de fondo (JALAL y SINGH, 2012).

De los dos enfoques usados en el seguimiento de objetos, el de *arriba hacia abajo* es más seguro y robusto.

En el otro enfoque *"de abajo hacia arriba"* se adquiere la posición del objeto, a partir de su trayectoria por los *frames* anteriores, este enfoque es usado para rastrear objetos de difícil reconocimiento. Según (JALAL y SINGH, 2012) puede fallar si hay grandes saltos de los objetos en *frames* consecutivos. Teniendo esto en cuenta, se decide utilizar el primer enfoque y detectar en cada *frame* el objeto usando características de fácil detección.

1.5.1. Técnicas de detección de objetos

1. *Detector de puntos*: son usados para encontrar puntos de interés en imágenes. Esta técnica se usa en el contexto de movimiento, estereoscopía y seguimiento. Una característica deseable en un punto de interés es su invarianza a los cambios en la iluminación y al punto de vista de la cámara. Los algoritmos detectores de puntos o esquinas más usados son el operador de interés de Moravecs, el detector de puntos de interés de Harris, el detector KLT (Shi y Tomasi) y el detector SIFT.

2. *Diferenciación de marco*: la presencia de objetos en movimiento es determinada calculando la diferencia entre dos imágenes consecutivas. Este cálculo es simple y sencillo de implementar. Para una variedad de entornos dinámicos se adapta muy bien, pero es generalmente difícil de obtener un esquema completo del objeto en movimiento, se puede manifestar el fenómeno *vacío* como consecuencia de una detección imprecisa del objeto en movimiento (PAREKH; THAKORE y JALIYA, 2014).

3. *Flujo óptico*: este método puede obtener una información completa del movimiento, lo representa como una serie de vectores de desplazamiento, que definen el desplazamiento de cada píxel en una región entre dos *frames* consecutivos. La mayor desventaja de este es la complejidad algorítmica, requiere de más tiempo que otros métodos y es sensible al ruido, por lo que no es adecuado para ocasiones en que se exige información en tiempo real (JALAL y SINGH, 2012).

4. *Substracción de fondo*: el primer paso en esta técnica es el modelado de fondo, este es el núcleo del algoritmo, se trata de tener un modelo de referencia para poder compararlo en cada *frame* en busca de posibles variaciones, estas variaciones en término de píxeles significan que existen objetos en movimiento. Este método proporciona la información más completa del objeto si el fondo es conocido. Este método tiene dos enfoques: el algoritmo recursivo y el no recursivo, el segundo depende de un *buffer* de almacenamiento (SEN-CHING y KAMATH, 2004).

5. *Segmentación*: estos algoritmos particionan la imagen en regiones perspectivamente similares. Los dos problemas a tener en cuenta son: el criterio de cuando una partición es buena y el método para lograr el particionamiento eficiente. Algunos de los métodos son: agrupamiento de desplazamiento medio (mean-shift clustering), segmentación de la imagen usando cortes de grafos, y contornos activos (YILMAZ; JAVED y SHAH, 2006).

1.5.2. Técnicas de seguimiento basado en marcadores

El uso de marcadores o fiduciarios en una escena, es un modo de simplificar la tarea de detección del objeto. El marcador es un objeto que presenta características conocidas y de fácil rastreo, como son los colores, las aristas y las esquinas. Estas características son fáciles de explotar y confiables para estimar la posición y orientación espacial. Los marcadores se pueden dividir en dos grupos: basados en puntos y extendidos. Los extendidos son muy populares y los más usados son generalmente de forma rectangular, esto reduce el costo computacional cuando son rastreados en una imagen.

Según (MORENO y CRUZ GUEVARA, 2008) existen varias técnicas para realizar el seguimiento utilizando marcadores:

- **Marcadores en forma de puntos**: estos marcadores han sido usados en la fotogrametría por muchos años. Tienen un diseño que permite su rápida y fácil detección. Presentan una apariencia de patrones circulares y son relativamente invariantes a la distorsión perspectiva (ibíd.). Su posición en el mundo real es conocida con precisión, esto se logra de forma manual con láser o un algoritmo llamado *Estructura a partir del movimiento (Structure-From-Motion)*. Se diseñan patrones geométricamente distintos y una vez que son identificados en una imagen se puede adquirir la posición de la cámara a partir de las correspondencias encontradas. Una desventaja es que este tipo de marcador necesita ser distribuido en varios lugares de la escena y sus posiciones necesitan ser medidas con precisión para poder extraer la posición de la cámara.
- **Marcadores extendidos**: estos marcadores cuadrados en blanco y negro son introducidos por (KOLLER; KLINKER; ROSE et al., 1997), estos contienen pequeños cuadrados para su identificación. Los marcadores rectangulares son usados también y según (FUA y LEPETIT, 2005) se demuestra que es necesario solamente un marcador para estimar la posición de la cámara. Durante el proceso de detección de los marcadores la estimación de la posición se ejecuta en tiempo real, por lo que puede ser usado en cada imagen dentro del flujo de video.

El seguimiento 3D de este tipo de marcadores no requiere inicialización manual, su uso se ha extendido porque llevan a una solución robusta y de bajo costo. Existen varias bibliotecas que brindan soporte para marcadores extendidos como ARToolKit (*Open Source Augmented Reality SDK* s.f.) y (*AVA* s.f.), por lo que se selecciona esta técnica para realizar el seguimiento en la solución propuesta.

En el estudio de los trabajos relacionados, se identificó la visión estereoscópica como la técnica más utilizada en los sistemas de seguimiento.

1.6. Visión estereoscópica

La visión estereoscópica es el proceso de entender o analizar superficies de objetos tridimensionales visibles, basado en imágenes. Los sistemas visuales de humanos y animales prueban que la visión estereoscópica funciona en ambientes complejos. Con el análisis de dos imágenes tomadas de un objeto, se puede obtener información que se pierde con su proyección perspectiva en el plano bidimensional, la proyección perspectiva y la afín son transformaciones necesarias dentro del proceso de visión por computador (ENDRESEN, 2010).

1.6.1. Transformaciones afines y perspectivas

Una transformación afín en matemática, se refiere a la transformación que se puede expresar con una multiplicación de una matriz y una adición vectorial. Una transformación afín de un paralelogramo $ABCD$ es otro paralelogramo $A'B'C'D'$, esta transformación mantiene el paralelismo de las rectas, pero puede modificar los ángulos y las longitudes de los segmentos. En la visión por computador, la transformación afín se basa en el escalado, rotación y traslación en el espacio bidimensional. Una transformación afín se puede describir de la siguiente forma: $x' = Ax$ donde A es el elemento básico, una matriz de transformación de dimensiones 2×3, que es la encargada de mapear las coordenadas de un punto cuando se le realiza esta transformación (DÍAZ, 2014).

$$x' = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} x \tag{1.6.1}$$

Por su parte, la transformación perspectiva está estrechamente ligada a la proyección perspectiva, la cual se manifiesta en el proceso de formación de la imagen, donde se forma una representación bidimensional, a partir de una representación tridimensional del mundo real. En esta conversión se pierde una dimensión. La forma usual de representar este proceso es con la proyección central, en la cual un rayo de un punto del espacio atraviesa un punto central e intersecta un plano específico llamado plano de imagen (R. HARTLEY y ZISSERMAN, 2003). La transformación perspectiva u homografía determina una correspondencia entre dos figuras geométricas, de forma que cada uno de los puntos y rectas de una de ellas le corresponden, respectivamente, un punto y una recta de la otra. La matriz $\tilde{H}$ que representa la transformación perspectiva se expresa en coordenadas homogéneas y es de dimensiones 3 por 3.

$$\tilde{x'} = \tilde{H}x \tag{1.6.2}$$

La coordenada resultante es homogénea y debe ser normalizada para obtener un resultado no homogéneo.

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + h_{22}} \qquad y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + h_{22}} \tag{1.6.3}$$

La transformación perspectiva preserva las líneas rectas (SZELISKI, 2010).

Transformación	**Matriz**	**#DoF**	**Preserva**	**Figura**
Traslación	$[I\|t]_{2x3}$	2	orientación	
Rígida (Euclidiana)	$[R\|t]_{2x3}$	3	longitudes	
Semejanza	$[sR\|t]_{2x3}$	4	ángulos	
Afín	$[A]_{2x3}$	6	paralelismo	
Proyectiva	$[\tilde{H}]_{3x3}$	8	líneas rectas	

Tabla 1.1. Transformaciones en el plano 2D. La semejanza preserva no solo los ángulos, también el paralelismo y las líneas rectas. La columna #DoF, se refiere a los grados de libertad.

Muchas veces es necesario reconstruir la dimensión perdida a causa de la proyección perspectiva, esto se puede lograr gracias a la disparidad que se obtiene de dos perspectivas de un punto.

1.6.2. Disparidad

Los humanos, y cualquier otro animal que tenga los ojos de tal forma que pueda ver un punto con ambos a la vez, son capaces de percibir profundidad basándose en las diferencias entre las imágenes que captan el ojo izquierdo y el ojo derecho. Si se prueba un simple experimento, como mantener el pulgar verticalmente delante de la cara y cerrar cada ojo alternativamente, se puede notar que este salta de izquierda a derecha en relación al fondo de la escena.

Esta disparidad o paralaje entre las dos imágenes, es inversamente proporcional a la distancia desde el observador. Medir automáticamente esta disparidad, estableciendo precisas correspondencias entre las imágenes, es un reto (ibíd.).

Cuando el objeto se encuentra relativamente cerca de las cámaras, arroja diferentes posiciones en las fotografías, pero cuando no se detecta ninguna disparidad en el objeto, significa que está a distancia infinita. Esto ocurre generalmente cuando el objeto se encuentra a una distancia lejana de las cámaras y los rayos

homólogos son paralelos.

1.7. Etapas en el proceso de la visión estereoscópica

En la visión estereoscópica se trabaja con la adquisición de imágenes y con la modelación de cámaras. Una descomposición en mayor detalle del proceso de visión estereoscópica fue realizada por (BARNARD y FISCHLER, 1982). Según esta descomposición, el proceso completo contempla seis pasos principales:

1. Adquisición de imágenes.
2. Modelado de la cámara (geometría de cámaras).
3. Extracción de las características.
4. Correspondencia de las imágenes (características).
5. Determinación de la distancia (triangulación).
6. Interpolación, cuando sea necesaria.

Estos pasos se realizan de forma secuencial en el orden que se presentan. De estos pasos el más complicado es hallar la correspondencia, y depende en gran medida de la elección de características (MONTALVO, 2010).

1.7.1. Adquisición de imágenes

La adquisición de imágenes puede realizarse de diversas formas. Pueden ser tomadas simultáneamente o mediante intervalos de tiempo de una duración determinada. Además estas pueden tomarse desde localizaciones y direcciones ligeramente distintas o totalmente diferentes. Las imágenes pueden tomarse con diferencia temporal grande, ejemplo de esto, cuando se usa sólo una cámara móvil y se hacen tomas de la escena desde distintas posiciones, en ese caso influye el momento del día en que fueron capturadas, las condiciones de iluminación y cualquier elemento que haya cambiado la escena (ibíd.). En el presente trabajo, donde se pretende detectar con precisión las coordenadas espaciales de un instrumento quirúrgico, el tiempo de adquisición del par de imágenes debe ser lo más simultáneo posible en cada cámara, pues una imagen con al menos un segundo de diferencia, asocia un punto distinto en el espacio y no calcula la coordenada real en el instante de tiempo.

1.7.2. Modelo geométrico de la cámara

Un modelo de cámara, son los atributos geométricos y físicos más importantes para la visión estéreo. Este modelo puede tener una componente relativa, la cual relaciona el sistema de coordenadas de una cámara con el de la otra, y es independiente de la escena, y también puede tener una componente absoluta, la cual relaciona el sistema de coordenadas de una de las cámaras con un sistema de coordenadas fijo de la escena. El modelo en que se centra este trabajo, utiliza dos cámaras que tienen sus ejes ópticos paralelos, con la distancia que los separa como su línea base. Quedando sus ejes ópticos perpendiculares a la línea base, y sus líneas de exploración o líneas epipolares paralelas a la línea base. Las líneas epipolares son líneas que unen la imagen izquierda y la imagen derecha a partir de las proyecciones de un punto. Cualquier punto del espacio tridimensional unido a los dos centros de proyección de las cámaras define un plano, llamado plano epipolar. La intersección de un plano epipolar con el plano de proyección de una cámara define una línea epipolar. Para todos los puntos, cuyas proyecciones izquierdas estén contenidas en una línea epipolar en la imagen izquierda, sus proyecciones derechas deben estar contenidas también sobre la línea epipolar en la imagen derecha, y viceversa (R. HARTLEY y ZISSERMAN, 2003).

El principal problema al realizar visión estereoscópica es encontrar la correspondencia entre los puntos de las imágenes. Se debe efectuar una búsqueda en dos dimensiones, tanto en el eje X como en el eje Y. Para reducir la complejidad de este proceso se puede llegar a una búsqueda unidimensional. Basta con situar y orientar las cámaras de forma que sólo exista un desplazamiento horizontal entre ellas, los ejes ópticos deben ser paralelos y los ejes de abscisas (eje X) de cada una de las cámaras deben ser coincidentes como sucede en la figura 1.2.

Bajo esta situación, las líneas epipolares que definen el plano epipolar son coincidentes, consiguiendo simplificar la búsqueda del emparejamiento a recorrer las imágenes por filas. Con esta geometría se obtiene la denominada restricción epipolar, de manera que, en el sistema de ejes paralelos convencional todos los planos epipolares originan líneas horizontales al cortarse con los planos de las imágenes. En un sistema con la geometría anterior, se obtiene un valor de disparidad d, para cada par de puntos emparejados $P_I(X_I, Y_I)$ y $P_D(X_D, Y_D)$ dado por $d = X_I - X_D$. Con el valor de disparidad para cada punto de la imagen se construye un matriz o mapa de disparidad, en el que cada punto de la imagen contiene su valor de disparidad, y así se puede calcular la profundidad para cada punto de la escena (ibíd.).

En la figura 1.2 se pueden ver todos los elementos comentados. El sistema absoluto, que pertenece a la escena real tridimensional que se está observando, está dado por el origen de coordenadas O y los ejes $\{X, Y, Z\}$. El sistema de referencia relativo de la cámara izquierda está dado por el origen de coordenadas O_I y los ejes $\{X_I, Y_I, Z_I\}$, y el sistema de referencia de la cámara derecha tiene origen de coordenadas en O_D y ejes $\{X_D, Y_D, Z_D\}$.

En ambos sistemas, el de la cámara izquierda y el de la derecha, se hace coincidir el centro de proyección

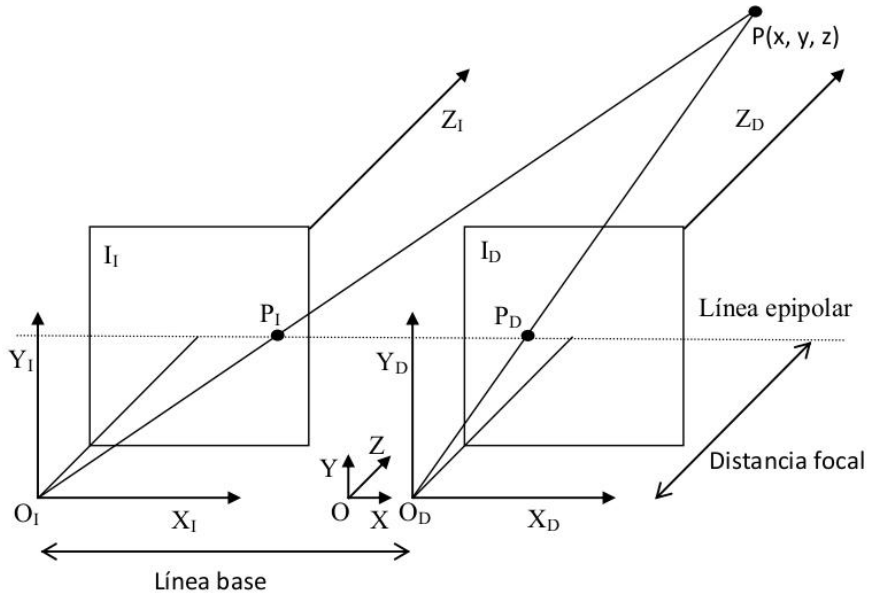


Figura 1.2. Correspondencia de un sistema estéreo (MONTALVO, 2010).

óptico con el origen de coordenadas. Según esta notación, todos los elementos que se refieran a la cámara izquierda y se expresen en su sistema de referencia tendrán el subíndice I, de forma idéntica para la cámara derecha pero con el subíndice D. El punto P en la escena tridimensional se proyecta en la imagen izquierda como el punto P_I y en la imagen derecha tiene como proyección P_D. Para obtener la proyección de un punto en una imagen se hace pasar un rayo por su centro óptico y por el propio punto, en la intersección de este rayo con el plano imagen se formará la proyección del punto. Los rayos de proyección PO_I y PO_D definen el plano de proyección del punto de la escena 3D, (el plano epipolar). Como se puede comprobar en la figura 1.2 se han hecho coincidir los ejes X de los dos sistemas relativos, así las imágenes estarán en correspondencia y las coordenadas Y de los puntos P_I y P_D serán idénticas, siendo las líneas epipolares paralelas y consiguiendo simplificar las búsqueda. La línea epipolar correspondiente a un punto se puede hallar con las matrices esencial y fundamental.

En la figura 1.3 se muestra una cámara junto a su sistema de coordenadas. El origen de coordenadas O coincide con el centro de proyección, y el eje Z está superpuesto con el eje óptico del sistema.

A pesar de la teoría expuesta anteriormente, en la realidad muy pocas veces las imágenes que forman el par estereoscópico se encuentran alineadas horizontalmente. Además en el proceso de adquisición se introducen deformaciones en las imágenes que degradarán la precisión en la medida de profundidad a menos que sean corregidas. Dos ejemplos de estas deformaciones son la distorsión radial y la distorsión tangencial. En este caso, es necesario realizar un proceso previo de calibración de cámaras con el fin de corregir dichas anomalías (ver sección 1.9).

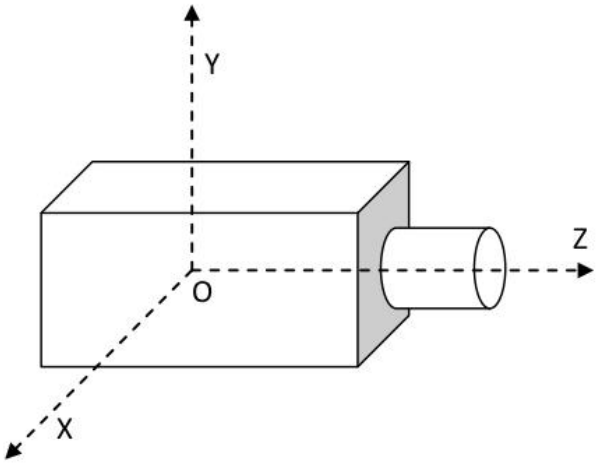


Figura 1.3. Sistema de referencia de una cámara.

Matriz esencial

Las líneas epipolares pueden ser calculadas mediante el uso de dos matrices. La matriz esencial E contiene la información de la traslación y rotación referente a ambas cámaras en el espacio físico. Según (SZELISKI, 2010) con ella se puede mapear un punto x_0 en la imagen de la izquierda dentro de una línea $l_1 = Ex_0$ en la imagen derecha. La matriz esencial no contiene la información intrínseca de las cámaras, y relaciona los puntos en coordenadas físicas o de cámara, no en coordenadas de píxeles (BRADSKI y KAEHLER, 2008).

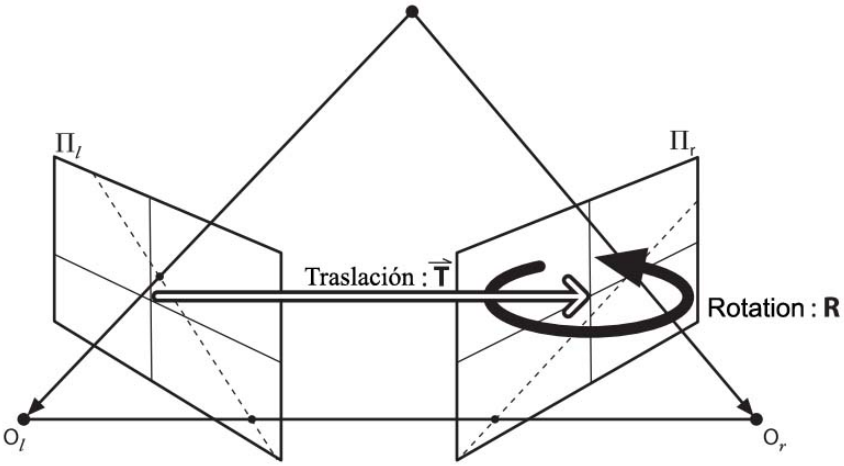


Figura 1.4. Geometría esencial de las imágenes estereoscópicas: se captura en la matriz E, que contiene toda información de rotación R y traslación T que describe la localización de la segunda cámara con respecto a la primera en coordenadas globales (BRADSKI y KAEHLER, 2008).

Matriz fundamental

La matriz esencial contiene toda la información geométrica de las cámaras relativa una con la otra, pero nada de las cámaras en sí. En la práctica, usualmente se trabaja con coordenadas de píxeles. Para encontrar una relación entre un píxel de una imagen y su correspondiente línea epipolar en la otra imagen se debe introducir la información intrínseca de las cámaras. La matriz fundamental es como la esencial pero esta opera en coordenadas de píxeles (BRADSKI y KAEHLER, 2008). Según (SZELISKI, 2010) se puede calcular la línea epipolar l_1 correspondiente al punto x_0 de la imagen izquierda en la imagen derecha con la ecuación $l_1 = Fx_0$. En el artículo de (BAILEY y WOLF, 2007) se explica con claridad la matriz fundamental.

Rectificación de las imágenes

Con el uso de la geometría de las cámaras, se puede usar la línea epipolar de cada píxel en una imagen para restringir su búsqueda en la otra imagen. Un algoritmo más eficiente puede ser obtenido rectificando primeramente las imágenes, de modo que las correspondientes líneas de búsqueda horizontales sean líneas epipolares. Para rectificar las imágenes se utilizan los parámetros obtenidos en la calibración previa, luego cualquier píxel de la imagen izquierda presenta una coordenada "y" idéntica al correspondiente en la imagen derecha (SZELISKI, 2010).

Los pasos para rectificar las imágenes son los siguientes (ver figura 1.5):

1. Rotar ambas cámaras para que sus ejes x coincidan.
2. Dado que existe un grado de libertad en la inclinación, las rotaciones más pequeñas que se obtienen se deben utilizar.
3. Para determinar el giro deseado alrededor de los ejes ópticos, el vector de arriba (el eje y de la cámara) se pone perpendicular a la línea central de la cámara. Esto asegura que las líneas epipolares correspondientes sean horizontales y que la disparidad de los puntos en el infinito es 0.
4. Por último, se escalan las imágenes si es necesario, magnificando la imagen más pequeña y reduciendo al mínimo las últimas distorsiones horizontales.

1.7.3. Extracción de características

En este paso de la visión estereoscópica, se procede a extraer las características a las que luego, en el próximo paso se le buscará la correspondencia en las imágenes. Por lo tanto, este paso está estrechamente ligado al de

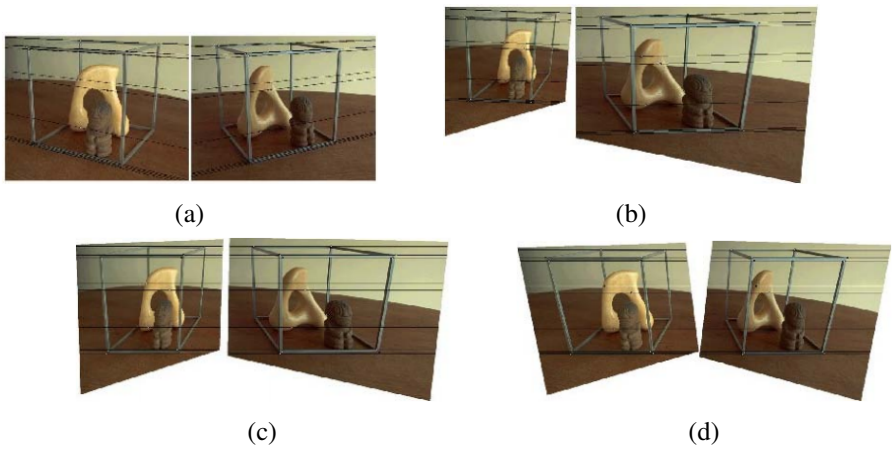


Figura 1.5. Etapas en la rectificación: (a) Par de imágenes originales con varias líneas epipolares; (b) imágenes transformadas, las líneas epipolares quedan paralelas; (c) imágenes rectificadas de manera que las líneas epipolares son horizontales y verticales en correspondencia; (d) rectificación final que reduce al mínimo las distorsiones horizontales (SZELISKI, 2010).

correspondencia. Primero se suele decidir qué método utilizar al realizar la correspondencia entre imágenes y según las características que se empleen, éstas serán las que se extraigan de las imágenes. Existen dos clases de técnicas para determinar la correspondencia entre dos imágenes estereoscópicas: las técnicas basadas en el área (*area-based*) y las técnicas basadas en las características (*feature-based*) (MONTALVO, 2010).

Como esta investigación trata dos áreas del conocimiento: el seguimiento de objetos y la visión estereoscópica, hay conceptos que se interceptan. Este es el caso de la elección de características visto en la sección 1.5. Los puntos de interés son las características escogidas por su bajo coste computacional. Los puntos son invariantes a los cambios de iluminación y a la posición de las cámaras. Cuando sea necesario buscar la correspondencia, solo se buscarán dichos puntos en la otra imagen en lugar de regiones completas, este proceso es llamado *correspondencia escasa*.

Esta decisión está sustentada por la necesidad de obtener una precisión alta en el cálculo de las posiciones espaciales. Es preferible concentrarse en unos pocos puntos y refinarlos, que trabajar con zonas más grandes de la imagen.

1.7.4. Correspondencia

En este paso, dado el método seleccionado, que puede ser basado en área (píxeles) o en características, se procede a localizar para un punto en el espacio tridimensional su proyección en cada uno de las imágenes del par estereoscópico. Los métodos basados en características, restringen la búsqueda a un conjunto disperso de características. Se emplean propiedades simbólicas y numéricas de las características, obteni-

das a partir de los llamados descriptores. En los métodos basados en área, los elementos a comparar son ventanas de la imagen de dimensión fija y el criterio de semejanza dentro de la región de búsqueda. Dentro de este segundo grupo se pueden distinguir métodos como: la Suma de diferencias absolutas (SAD), la Suma de diferencias al cuadrado (SSD), la Correlación cruzada normalizada (NCC) y el coeficiente de correlación de Pearson. La explicación detallada de estos métodos se encuentran en la siguiente bibliografía: (LÓPEZ-VALLES; FERNÁNDEZ-CABALLERO y FERNÁNDEZ, 2005);(KANADE y OKUTOMI, 1994); (OKUTOMI y KANADE, 1993); (MATTHIES; KANADE y SZELISKI, 1989) y (HIRSCHMULLER; INNOCENT y GARIBALDI, 2002).

Ambos procesos de correspondencia utilizan propiedades de la realidad física, que son formuladas en términos de restricciones. Las restricciones utilizadas se enuncian a continuación (PAJARES y CRUZ, 2007) y (SCHARSTEIN y SZELISKI, 2002):

1. *Epipolar*: las imágenes de una entidad 3D deben proyectarse sobre una línea epipolar. Esta restricción se deriva de la geometría de las cámaras y requiere que las cámaras estén alineadas.
2. *Semejanza*: las dos imágenes de una entidad 3D deben tener propiedades o atributos similares.
3. *Unicidad*: para cada característica en una imagen debe haber una única característica en la otra imagen, salvo que se produzca una oclusión y no haya correspondencia de alguna característica.
4. *Orden posicional*: dadas dos características en una determinada imagen, por ejemplo la izquierda, situada una a la derecha de la otra, esta restricción supone que este orden se mantiene en la imagen derecha para sus respectivas características homólogas.
5. *Continuidad de la disparidad*: asume que las variaciones de disparidad en la imagen son generalmente suaves, si se considera un mapa de disparidad, éste se presenta continuo salvo en unas pocas discontinuidades.

La correspondencia obtenida puede ser escasa si se seleccionó el tipo basado en características o densa si se escogió el basado en áreas o píxeles. Los puntos de interés a buscar son las esquinas del marcador previamente detectado. Así el algoritmo encargado de buscar la correspondencia no necesita pasar por toda la imagen para obtener el mapa de disparidad completo, sino solo la de dichos cuatro puntos.

1.7.5. Determinación de la distancia o triangulación

Para hallar la posición de un punto en el espacio, es preciso tener la posición de las dos proyecciones del punto, la calibración de las cámaras estereoscópicas y su pose. Este proceso requiere de la intersección de

dos rayos en el espacio y es comúnmente llamado triangulación. Cuando hay ausencia de ruido, el problema es trivial, pero si no es el caso los dos rayos generalmente no se encuentran y es preciso hallar el mejor punto de intersección.

En (R. I. HARTLEY y STURM, 1997) se explican varios algoritmos usados en la triangulación y realizan a cada uno de ellos pruebas de eficiencia y error cometido. Dentro de las pruebas de error miden el error 3D y 2D para la reconstrucción euclidiana, el error en la reproyección 2D entre otros. Además de esto realizan evaluaciones en imágenes reales.

Los algoritmos comparados fueron el polinomial, polinomial-abs, lineal-valor-propio (*linear-eigen*), iterativo-valor-propio (*iterative-eigen*), punto medio, lineal-LS (*least squares*) e iterativo-LS.

Algoritmo	**Tiempo**
polinomial-abs	60
polinomial	28
iterativo-valor-propio	10
lineal-valor-propio	6
iterativo-LS	6
lineal-LS	4
punto medio	4

Tabla 1.2. Comparación relativa de velocidad entre algoritmos de triangulación. No se utiliza unidad de medida, se expresan números que representan la diferencia relativa entre los algoritmos, los cuales procesan miles de puntos.

Todos los métodos se desempeñan bien para la reconstrucción euclidiana, en términos de error 3D. El polinomial-abs es ligeramente superior que el polinomial en cuanto al error. Ambos métodos tienen como desventajas que no son especialmente fáciles de generalizar para más de dos imágenes y son más lentos que los otros.

El iterativo-LS es un buen método, aunque a veces no converge, es tres veces más rápido que el polinomial y es muy cercano a ser invariante a la proyección. Es ligeramente más rápido que el iterativo-valor-propio.

Según (VITE-SILVA; CRUZ-CORTÉS; TOSCANO-PULIDO y FRAGA, 2007) cuando hay grandes niveles de ruido en las imágenes, y a su vez en las matrices fundamentales y de proyección, el polinomial-abs no se desempeña bien, incluso el iterativo-LS arroja mejores resultados, lo cual es una situación bastante realista.

Tomando en cuenta lo dicho anteriormente, se selecciona el método iterativo-LS para realizar la triangulación, este arroja buenos resultados en ambientes reales donde existe ruido en las imágenes. Además no se corre el riesgo de obtener una no convergencia con algún par de puntos con disparidad cero (en el infinito), puesto que el objetivo no es triangular todos los puntos de la escena sino solo las cuatro esquinas de los marcadores, ninguno de los marcadores estará a una distancia muy grande de las cámaras.

Después de la triangulación es necesario hallar la transformación correspondiente al marcador.

1.7.6. Interpolación

Muchas aplicaciones que utilizan visión estereoscópica calculan mapas de profundidad. Un mapa de profundidad es una imagen que a diferencia de tener valores de colores en sus píxeles, tienen los valores de profundidad calculados con la triangulación. Cuando se utiliza una correspondencia densa, se obtiene un mapa de profundidad sumamente detallado, pero requiere un tiempo de cómputo mayor que al utilizar la correspondencia escasa. La correspondencia escasa se obtiene solamente tomando características de interés de las imágenes. La interpolación es un método utilizado para obtener mapas de profundidad densos a partir de una correspondencia escasa. Se halla una función continua, que aproxime la interpolación, para obtener la profundidad de cualquier punto del espacio tridimensional (MONTALVO, 2010).

1.8. Transformación rígida

Es necesario explicar el proceso final del seguimiento estereoscópico que se propone en el presente trabajo. Una vez que se realizó la triangulación de las esquinas de los marcadores en las imágenes, se tiene un conjunto de puntos en el espacio asociados a cada marcador. Como el objetivo de un componente de seguimiento en la navegación por imágenes es brindar posición y rotación, no basta con conocer la posición de estos marcadores, es decir, las coordenadas (x, y, z) de cada esquina, es necesario obtener una matriz de transformación, que es la unión de una matriz de rotación y una de traslación, asociada a cada marcador. Estas matrices contendrán la información geométrica de los marcadores con respecto al sistema de coordenadas de las cámaras.

Se le llama transformación rígida a la matriz que es capaz de alinear dos conjuntos de puntos en un espacio dado, donde sus pares correspondientes han sido determinados. Se llama *rígida* porque solo aplicando rotaciones y traslaciones se puede lograr ese objetivo. Existen varios algoritmos que hallan los componentes de rotación y traslación. En el trabajo de (A. LORUSSO y FISHER, s.f.) se comparan cuatro algoritmos muy populares que le dan solución al problema. Esos difieren en términos de representación de la transformación y método de solución, utilizando respectivamente la Descomposición en valores singulares de una matriz (SVD), matrices ortogonales (OM), *quaternions* (UQ) y *dual quaternions* (DQ). La comparación presenta los resultados de varios experimentos diseñados para determinar la exactitud en presencia de ruido, estabilidad respecto a juegos de datos degenerados y tiempo de cómputo.

La comparación realizada por (ibíd.) determinó que el SVD provee el mejor promedio de exactitud y estabilidad, pero no es el más eficiente para juegos de datos con grandes cantidades de puntos. El algoritmo que

usa *quaternions* es ligeramente más rápido pero está en desventaja en los otros aspectos. Como los conjuntos de puntos que se tienen no son grandes, la mejor opción es usar el SVD que provee mejor precisión. El algoritmo de Kabsch (KABSCH, 1978) usa la descomposición SVD y es el elegido para la propuesta de solución.

1.8.1. Descomposición SVD

En álgebra lineal, la descomposición en valores singulares de una matriz real o compleja es una factorización de la propia matriz. Esto tiene muchas aplicaciones en estadística y otras disciplinas. Según (MARTÍNEZ FERNÁNDEZ DE LAS HERAS, 2005), una descomposición SVD de la matriz compleja M de dimensiones $m \times n$ es una factorización de la forma $M = UWV^T$, donde U es una matriz unitaria $m \times m$, W es una matriz rectangular $m \times n$ con números no negativos en su diagonal, y V^T (La transpuesta de V) es una matriz unitaria $n \times n$. Los valores de la diagonal de W son conocidos como los valores singulares de M.

1.9. Proceso de calibración

La calibración es esencial en cualquier aplicación de visión por computador que se base en la captura de imágenes o video. En este caso tiene como objetivos fundamentales la obtención de los coeficientes de distorsiones y el establecimiento de la geometría epipolar del sistema de cámaras. Mediante la calibración se calculan los parámetros intrínsecos y extrínsecos de la cámara, a partir de un conjunto de puntos de control. Conocidas las coordenadas tridimensionales de los puntos y midiendo las correspondientes coordenadas en la imagen obtenida con la cámara (FONSECA, 2011) y (SAÍZ, 2010).

Parámetros intrínsecos

- Distancia focal: f
- Desplazamiento del centro de la imagen: (C_x, C_y)
- Coeficiente de distorsión radial: K_1, K_2

Cuando se realiza la calibración, se obtiene una matriz intrínseca correspondiente a una cámara con la siguiente distribución:

$$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1.9.1}$$

Donde:

- f_x y f_y: denotan la distancia focal en la dirección x y dirección y respectivamente.
- c_x y c_y: denotan el punto principal que idealmente debería ubicarse en el centro de la imagen.

En ocasiones estos parámetros son suministrados directamente por el fabricante de la cámara aunque existen multitud de técnicas para su obtención.

En la figura 1.6 se muestran algunas de las representaciones de los parámetros de la cámara más importantes que intervienen en el proceso de adquisición tridimensional (FONSECA, 2011).

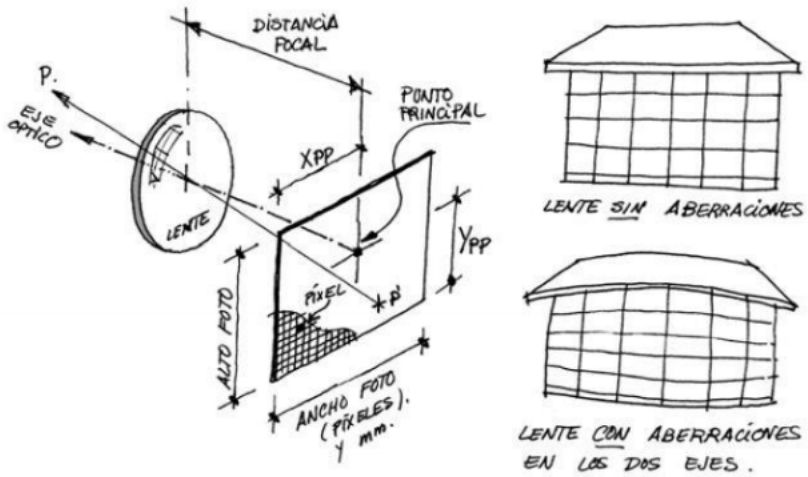


Figura 1.6. Parámetros de calibración de una cámara (FONSECA, 2011).

Distorsiones

La distorsión geométrica existe por la imposibilidad de construir una cámara ideal. Existen dos tipos de distorsiones, la tangencial debido a la no perpendicularidad del eje óptico del lente con el plano de la imagen, y la radial, que hace que los píxeles se alejen (distorsión de barril) o se acerquen (distorsión de acerico, *pincushion distortion*) del centro de la imagen en una cantidad proporcional a la distancia radial. Con la

distorsión radial las líneas rectas aparecen curvadas. La distorsión de ojo de pescado se manifiesta en cámaras de un lente más ancho donde se pueden tomar imágenes de hasta 180° de un lado a otro (SZELISKI, 2010).

(a) Barril (b) Pincushion (c) Ojo de pescado

Figura 1.7. Distorsiones geométricas (SZELISKI, 2010).

Los coeficientes de distorsión radial suelen denotarse con la letra k. Los modelos matemáticos más simples utilizados para corregir estas anomalías son polinomios de bajo orden.

$$\begin{aligned} x'_c &= x_c(1 + k_1 r_c^2 + k_2 r_c^4) \\ y'_c &= y_c(1 + k_1 r_c^2 + k_2 r_c^4) \end{aligned} \tag{1.9.2}$$

Donde $r_c^2 = x_c^2 + y_c^2$, (x_c, y_c) son las coordenadas del píxel obtenidas después de la proyección perspectiva pero antes de escalarlas por la longitud del foco f y correrlas por el centro óptico (c_x, c_y).

Luego del paso de la distorsión radial, las coordenadas finales del píxel se pueden obtener usando:

$$\begin{aligned} x_s &= f x'_c + c_x \\ y_s &= f y'_c + c_y \end{aligned} \tag{1.9.3}$$

Parámetros extrínsecos

Los parámetros extrínsecos definen la posición y orientación de la cámara con respecto al sistema de coordenadas de la escena: traslación T_{ij} y rotación R_j. Existen muchas técnicas para obtener los parámetros extrínsecos de una cámara, aunque los más habituales son los que utilizan conocimiento específico acerca de la escena (FONSECA, 2011).

Las etapas que sigue el proceso de calibración para llegar a determinar dichos parámetros son:

- Obtención del modelo matemático.
- Obtención de los datos de campo: En esta etapa se obtienen los puntos 2D de la imagen que provienen de los puntos tridimensionales de las escenas.
- Determinación de los parámetros tanto intrínsecos como extrínsecos de la cámara: Una vez conocidos todos los puntos de la escena real y la imagen capturada por la cámara, se procede a resolver las ecuaciones que rigen el modelo que mejor se aproxima a la cámara.

1.10. Marco de trabajo

Para el desarrollo de un software es necesario definir un conjunto de herramientas, dentro de las que se encuentran los lenguajes de programación, bibliotecas y herramientas auxiliares para la compilación y la metodología de desarrollo de software.

Lenguaje de programación C++

Este es una optimización del lenguaje de programación C, hereda las características de su predecesor y tiene otras, que lo hacen mucho más flexible, como la inclusión de clases y espacios de nombre, así se puede utilizar el paradigma orientado a objetos, y evitar la programación estructurada de C. Permite trabajar tanto a bajo como a alto nivel y proporciona un acceso a nivel de hardware solo superado por ensamblador. Al ser un lenguaje compilado, es apto para aplicaciones que requieran una alta velocidad de ejecución, como las utilizadas en visión por computadora (STROUSTRUP, 1986).

Bibliotecas para el desarrollo

OpenCV

Es una biblioteca de código abierto con licencia BSD de visión por computador escrita en C y C++. Es compatible con Linux, Windows y Mac OS X. Existen interfaces de desarrollo para Python, Ruby, Matlab y otros lenguajes. OpenCV fue diseñada para eficiencia computacional y aplicaciones de tiempo real, puede aprovechar los procesadores multinúcleos y las funciones de tarjetas gráficas. Posee una infraestructura de visión por computadora de fácil uso y más de 500 funciones incluidas (BRADSKI y KAEHLER, 2008).

Qt

Marco de trabajo multiplataforma escrito en C++ ampliamente usado tanto en aplicaciones con interfaz gráfica como en aplicaciones de consola. Qt es desarrollado como un software libre y de código abierto, presenta licencia GNU *Lesser General Public License*. Cuenta con funcionalidades para la manipulación de archivos, manejo de ficheros, acceso a bases de datos, uso de Extensible Markup Language (XML), gestión de hilos y soporte de red (*Qt Cross-platform application* s.f.).

IGSTK

Es una biblioteca de C++ de código abierto, que provee los componentes básicos necesarios para el desarrollo de aplicaciones de cirugía guiada por imágenes. Esta biblioteca se caracteriza por su robustez usando una arquitectura de máquina de estado. Es multiplataforma y usa CMake para gestionar el proceso de compilación. IGSTK permite cargar y visualizar imágenes médicas en formato DICOM, soporta la mayoría de los dispositivos de seguimiento existentes en el mercado, y provee una vía para extender el soporte a otros tipos de componentes de seguimiento (ENQUOBAHRIE y YANIV, 2009).

ArUco

Biblioteca mínima para aplicaciones de realidad aumentada basada en OpenCV, desarrollada por el grupo de la Universidad de Córdoba, Aplicaciones de visión artificial (*AVA* s.f.). Provee un conjunto de funcionalidades para detectar marcadores conocidos y su posición espacial utilizando una cámara. Los marcadores son cuadrados para facilitar el proceso de detección, además cuentan con un código binario para su identificación que permite el uso de varios de ellos en la escena.

Recursos auxiliares

CMake

Herramienta que ayuda a simplificar el proceso de compilación para el desarrollo de proyectos en diferentes plataformas. Cmake automatiza la generación de archivos para la compilación como *Makefiles* y archivos del *Visual Studio*. Es flexible y extensible, cuenta con módulos para OpenCV, IGSTK, Qt entre muchos otros (*CMake* s.f.).

XML y YML

Lenguajes neutrales para las aplicaciones. Son usualmente procesados por parseadores o procesadores de XML y Formato de serialización de datos legible al humano (YML) respectivamente. Los documentos son fáciles de leer y se guardan en texto plano. Presentan una estructura de jerarquía y permiten elegir las propias etiquetas, a diferencia de HTML (*The Official YAML Web Site* s.f.).

Metodología de desarrollo XP

Extreme Programming (XP) fue desarrollada por Kent Beck, es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios (JOSKOWICZ, 2008). El objetivo de XP es tener grupos pequeños o medianos. Cuando los requerimientos aún son muy ambiguos y/o pueden ser de alto riesgo, se cambian durante el proceso. Esta metodología pretende tener al cliente vinculado directamente con el desarrollo del proyecto. También, se presenta para el incremento de la comunicación entre el cliente y el desarrollador en el entorno de trabajo, con el motivo de mantener una agenda dinámica (LETELIER, 2006).

1.11. Conlusiones parciales

En el presente capítulo se analizaron distintos dispositivos de seguimiento utilizados en los neuronavegadores, identificando los ópticos basados en grabación de video como la solución más viable. Se explicaron varias técnicas de seguimiento visual de objetos y se optó por las basadas en detección de puntos y marcadores. Se identificaron distintos algoritmos para triangular puntos, y se escogió el iterativo-LS por su buen desempeño en ambientes donde existe distorsión, además de tener un bajo costo computacional. Se eligió el algoritmo basado en la descomposición SVD para hallar la transformación rígida entre dos conjuntos de puntos. Por último, basándose en los trabajos relacionados, se decidió utilizar la biblioteca IGSTK que posibilita el desarrollo de robustas aplicaciones de navegación guiadas por imágenes.

CAPÍTULO 2

Propuesta de solución

En este capítulo se explican los pasos para calibrar un sistema de visión estereoscópica. Se explica como se utiliza el proceso estereoscópico y sus etapas en el componente de seguimiento, además de su integración con la biblioteca IGSTK. Se realiza la primera parte de la ingeniería del sistema utilizando la metodología XP, se definen las historias de usuario y el plan de entrega. Se define la arquitectura, las tarjetas CRC y los patrones de diseño para la implementación del sistema.

2.1. Solución técnica

Como dispositivo de seguimiento se seleccionó el óptico basado en grabación de video. Se utilizan dos cámaras de bajo costo para probar el componente de seguimiento. A continuación se muestra el flujo del sistema y los pasos que lo conforman:

- *Adquisición*: Se adquieren las imágenes de ambas cámaras.
- *Calibración*: Se desarrolló una herramienta con OpenCV para calibrar a nivel de software sistemas estereoscópicos basados en video. Esta depende de un tablero similar al de ajedrez.
- *Rectificación*: Se rectifican ambas imágenes (ver subepígrafe 1.7.2).
- *Detección de marcadores*: Los marcadores rastreados son cuadrados en blanco y negro. La biblioteca ArUco implementa funcionalidades de detección de estos marcadores. Estos son colocados en la herramienta quirúrgica como muestra la figura 2.2.

- *Triangulación*: Una vez que se tienen las posiciones del marcador en cada una de las imágenes (sus respectivas proyecciones), se hallan las posiciones de sus cuatro esquinas en el espacio tridimensional mediante triangulación. Estas posiciones son relativas al eje de coordenadas del sistema estereoscópico (ver epígrafe 1.7.2). El algoritmo utilizado para triangular los puntos es el iterativo-LS (ver epígrafe 1.7.5).

- *Matriz de transformación*: Por último se halla la matriz de transformación. Es necesario conocer no solo la posición espacial de los marcadores, sino su orientación con respecto a la geometría de las cámaras. Para esto se define una posición canónica del marcador en el centro de coordenadas del sistema, y se calcula la transformación rígida entre los puntos espaciales del marcador hallados con la triangulación y los puntos canónicos (ver sección 1.8).

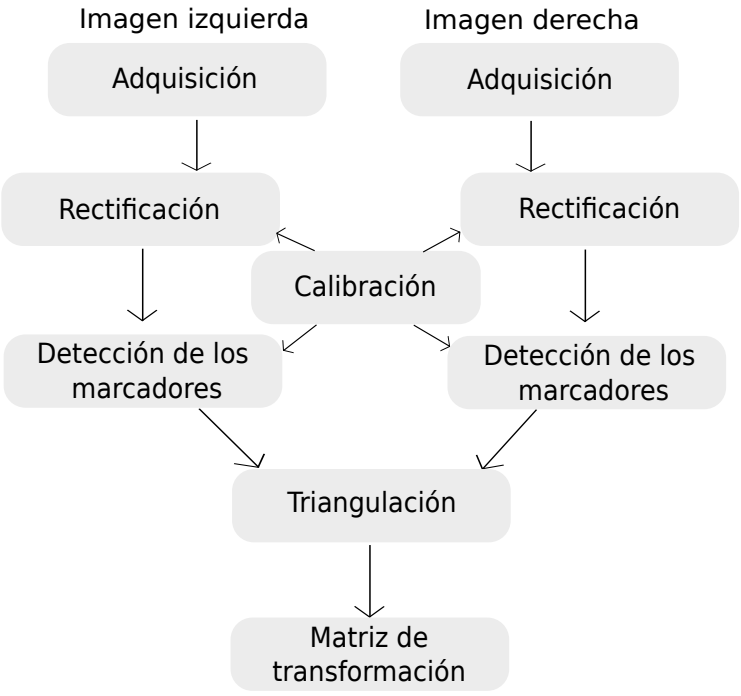


Figura 2.1. Flujo del sistema

El resultado final es un componente de seguimiento compatible con la biblioteca IGSTK. La integración con la biblioteca se realiza al heredar de dos clases: *igstk::Tracker* que es la encargada de realizar el seguimiento, e *igstk::TrackerTool* que es la encargada de representar los objetos seguidos. Este componente además de IGSTK, depende de la biblioteca OpenCV para el procesamiento de las imágenes, ArUco para la detección de los marcadores y Qt como marco de trabajo utilizado.

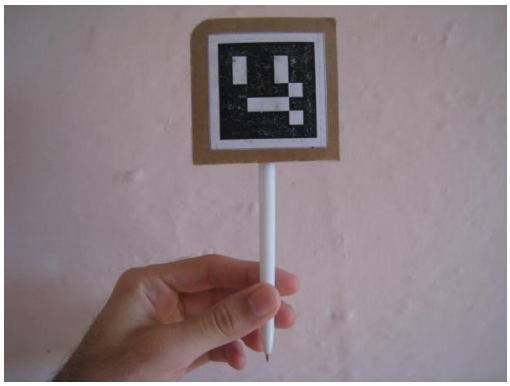


Figura 2.2. Herramienta rastreada (marcador cuadrado en blanco y negro)

2.2. Dispositivo de seguimiento

El componente desarrollado utiliza dos cámaras de bajo costo fabricadas por la compañía (*Logitech International S.A.* s.f.). Estas cámaras cuentan con un cable USB 1.0 para la conexión. La resolución es de 352×288 píxeles y el zoom se puede ajustar manualmente. Estas cámaras cuentan con una frecuencia de captura estándar de 30 *frames* por segundo.

2.3. Adquisición de imágenes

La posición de las cámaras es fundamental, para la solución propuesta, estas se colocan una al lado de la otra, con una separación de aproximadamente 10 centímetros y sus ejes ópticos paralelos, de esta forma ambos ejes quedan perpendiculares a la línea base (ver epígrafe 1.7.2). Esta separación puede variar en dependencia de la distancia promedio a la que se detectarán los marcadores. Con esta separación se puede realizar el seguimiento a una distancia promedio de 30 centímetros a 3 metros, aunque según (WIDMANN; STOFFNER y BALE, 2009), en la navegación guiada por imágenes, el campo para el seguimiento es de $100 \times 100 \times 100$ centímetros aproximadamente.

Con la configuración definida es decir, la posición y el zoom de las cámaras, se aseguran las restricciones para la correspondencia y se obtiene una disparidad horizontal. Mientras mejor sea la disposición de las cámaras, mejor será la calibración, y serán corregidos los pequeños errores inevitables.

Otro punto a tener en cuenta es la **sincronización de las cámaras**. Las cámaras utilizadas no cuentan con sincronización por hardware y es muy difícil que las imágenes sean tomadas en el mismo instante de tiempo. Este efecto negativo se reduce si las peticiones y los *frames* llegan concurrentemente, para eso las cámaras deben contar con su propio bus de conexión USB.

En la captura de imágenes se utilizó la clase *cv::VideoCapture* para conectar las cámaras y obtener los *frames*. Para reducir el tiempo de captura entre el par de imágenes, se utilizaron las funciones *VideoCapture::grab* y *VideoCapture::retrieve* en vez de *VideoCapture::read*. Las dos primeras funciones se utilizan cuando las cámaras no cuentan con sincronización por hardware. Primero se llama la función *grab* para cada cámara, y luego el método más lento *retrieve* que decodifica la imagen. De esta forma ambos *frames* estarán más cerca en el tiempo. Con la adquisición de las imágenes, se puede realizar el proceso de calibración de las cámaras.

2.4. Calibración estereoscópica

La calibración estereoscópica es un procedimiento parecido a la calibración de una cámara. Es necesario realizar varias tomas con ambas cámaras de un objeto conocido, el más utilizado y difundido es una rejilla de cuadrados alternados blancos y negros, que es llamado usualmente (tablero de ajedrez), este no tiene que tener necesariamente 8 filas y 8 columnas ni ser cuadrado, pero mientras más esquinas posea, se comparan más puntos y se puede llegar a calibraciones más exactas. Según (BRADSKI y KAEHLER, 2008) cualquier tablero asimétrico mayor de 3×3 es válido.

Para calibrar el sistema se construyó una herramienta de calibración estereoscópica con interfaz visual (ver figura 2.3). La biblioteca de visión por computadora OpenCV contiene funciones para calibrar estos sistemas como *cv::stereoCalibrate* y *cv::stereoRectify*. Existe otra función que es utilizada para hallar las matrices de calibración de un sistema estereoscópico, *cv::stereoRectifyUncalibrated*, pero ésta no utiliza los parámetros intrínsecos de las cámaras y su posición relativa en el espacio, por lo que se optó utilizar *cv::stereoRectify* en su lugar. Cuando comienza la calibración, se deben realizar tomas ubicando el tablero en varias poses para obtener varias perspectivas y con condiciones lumínicas lo más parecidas posibles, la luz global afecta a ambas cámaras por igual. Todas las esquinas deben ser detectadas en cada iteración, para esto se usó el detector de Harris implementado en OpenCV.

Además de buscar los parámetros intrínsecos y extrínsecos de las cámaras, la herramienta encuentra la matriz de rotación R y el vector de traslación T de la cámara derecha con respecto a la izquierda. Las matrices esencial y fundamental (ver subepígrafe 1.7.2) también son obtenidas, además de las nuevas matrices de reproyección para cada cámara y la matriz de mapeo de disparidad.

Para la solución se utilizó un tablero de 9×6 filas y columnas. Con la herramienta se puede definir el tamaño de la casilla, y todas las medidas calculadas corresponderán a esa métrica, es decir, si se determina que el lado de la casilla tiene 1 cm, entonces todas las unidades corresponderán a un centímetro. Además se agregó la funcionalidad de mostrar el mapa de disparidad y modificar un conjunto de parámetros que intervienen en su representación, con el objetivo de visualizar la calidad de la calibración. Por último se agregaron las funcionalidades de guardar y cargar toda la configuración de la calibración en formato XML o YML.

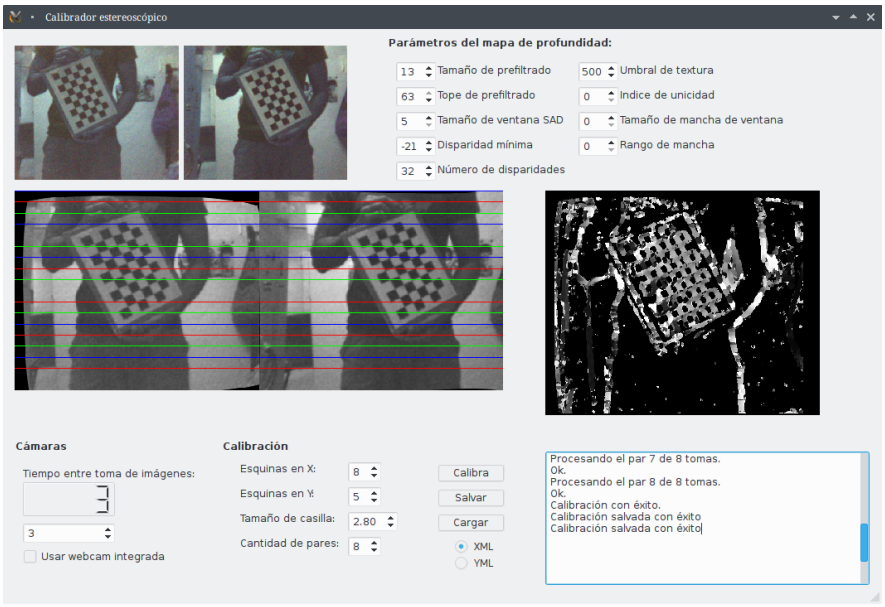


Figura 2.3. Herramienta para la calibración estereoscópica desarrollada.

2.4.1. Distribución del archivo de calibración

La herramienta genera un archivo en formato XML ó YML, a decisión del usuario, con toda la información de calibración de las cámaras. Las etiquetas utilizadas aparecen en el siguiente orden en el archivo de calibración:

1. *frame_size*: tamaño de la imagen.
2. *intrinsics_left*: Matriz intrínseca de la cámara izquierda.
3. *distortion_left*: Coeficientes de distorsión de la cámara izquierda.
4. *intrinsics_right*: Matriz intrínseca de la cámara derecha.
5. *distortion_right*: Coeficientes de distorsión de la cámara derecha.
6. *rotation_matrix*: Matriz de rotación de la cámara derecha respecto a la izquierda.
7. *rotation_matrix1*: Matriz de rotación de la cámara izquierda.
8. *rotation_matrix2*: Matriz de rotación de la cámara derecha.
9. *traslation_matrix*: Matriz de traslación de la cámara derecha respecto a la izquierda.
10. *essential_matrix*: Matriz esencial.

11. *fundamental_matrix*: Matriz fundamental.
12. P1: Nueva matriz de proyección de la cámara izquierda después de la rectificación.
13. P2: Nueva matriz de proyección de la cámara derecha después de la rectificación.
14. Q: matriz 4×4 utilizada para mapear la disparidad y profundidad.
15. mx1: mapa para la rectificación de la coordenada *x* de la primera imagen.
16. my1: mapa para la rectificación de la coordenada *y* de la primera imagen.
17. mx2: mapa para la rectificación de la coordenada *x* de la segunda imagen.
18. my2: mapa para la rectificación de la coordenada *y* de la segunda imagen.

Cuando se tiene el sistema calibrado no se puede cambiar su configuración, pues las matrices que contienen la información geométrica ya no serían válidas. En ese caso se debe recalibrar todo el sistema. Luego de calibrar el sistema se procede a rectificar las imágenes.

2.5. Rectificación de las imágenes estereoscópicas

El objetivo de la rectificación es mantener en una línea epipolar los puntos homólogos de cada imagen. Se utilizó la función *cv::initUndistortRectifyMap*, la cual es capaz de hallar una matriz de igual tamaño que las imágenes a rectificar. Esta matriz es capaz de mapear cada píxel a su nueva posición rectificada y sin distorsión. Estos mapas son utilizados por la función *cv::remap* para rectificar cada imagen adquirida. En la herramienta de calibración se muestran las imágenes rectificadas unidas horizontalmente con algunas líneas epipolares dibujadas (ver figura 2.4). Con las imágenes rectificadas se procede al siguiente paso que es la detección de marcadores.

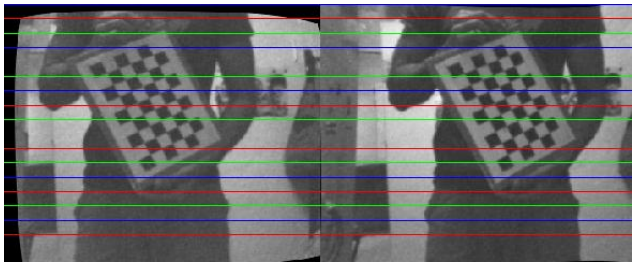


Figura 2.4. Rectificación utilizando OpenCV.

2.6. Detección del marcador

La biblioteca ArUco desarrollada por (*AVA* s.f.) tiene implementado un algoritmo robusto y fiable de detección de marcadores extendidos o fiduciales. Este utiliza marcadores de forma cuadrada, que han sido ampliamente difundidos por la facilidad de detección de su forma y sus cuatro esquinas. Los marcadores de ArUco poseen una codificación en su interior que además de servir para hallar la rotación, posibilita su identificación, por lo que se pueden utilizar varios en una escena. (GARRIDO-JURADO; MUÑOZ-SALINAS; MADRID-CUEVAS y MARÍN-JIMÉNEZ, s.f.).

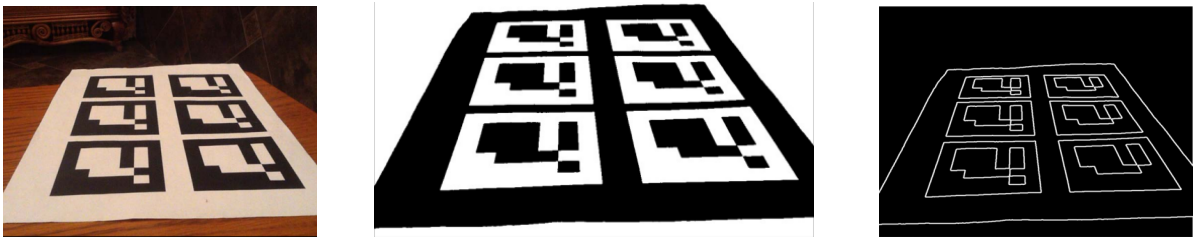
2.6.1. Obtención de los puntos

La detección de marcadores con ArUco se realiza utilizando algunos algoritmos implementados en OpenCV. Estos son los pasos fundamentales que realiza esta biblioteca (BAGGIO, 2012):

1. Se obtiene la imagen que contiene los marcadores.
2. Se le aplica a la imagen original un filtro de umbral adaptativo.
3. Se detectan fácilmente los contornos en la imagen modificada.
4. Se analizan las regiones cuadradas que pueden ser marcadores potenciales. Para esto se le aplican filtros a los contornos, con estos se eliminan los contornos que no tienen cuatro aristas y los que tienen longitudes de aristas menores que las esperadas.
5. Se elimina la proyección perspectiva para identificar el marcador en su forma canónica.

Figura 2.5. Eliminación de la proyección perspectiva

6. Se realiza la decodificación y se analiza el código interno del marcador con una máscara de bits. Si el código es válido se considera que el rectángulo es un marcador.
7. Por último se refinan las esquinas para obtener su ubicación con una precisión de subpíxel, para esto se utiliza la función *cv::cornerSubPix* de OpenCV.



(a) (b) (c)

Figura 2.6. Proceso de detección de marcadores con ArUco: a) imagen original, b) aplicación de un filtro para llevar a blancos y negros la imagen (umbral adaptativo), c) detección de contornos (BAGGIO, 2012).

Codificación del marcador

La codificación del interior del marcador permite el uso de varios de ellos simultáneamente, es una versión modificada del código de Hamming. Es un total de 25 bits divididos en 5 filas de 5 bits cada una, así se obtienen 5 palabras de 5 bits. Cada palabra contiene solo 2 bits de información, el resto es para la detección y correción de errores. Los 10 bits de información real permite obtener $2^{10} = 1024$ marcadores diferentes.

2.7. Triangulación

Con los marcadores detectados en las imágenes se tienen los puntos asociados a sus esquinas, luego se hallan las coordenadas espaciales correspondientes a estos puntos. Como se explicó en el epígrafe 1.7.5, se implementó el algoritmo iterativo-LS, al cual se le pasan dos puntos en el espacio bidimensional (las proyecciones), y devuelve la información correspondiente al punto en el espacio tridimensional.

El método implementado por OpenCV *cv::triangulatePoints* no es el más preciso, esto es comentado por varios desarrolladores en los propios foros de OpenCV. No obstante se utilizaron ambos métodos y se compararon sus resultados. En efecto, el método iterativo-LS fue superior al de OpenCV y se utilizó en la solución (ver figura 3.1).

2.8. Obtención de la matriz de transformación

Una vez realizada la triangulación, es necesario hallar la matriz de transformación asociada a cada marcador con respecto al sistema de coordenadas de las cámaras (ver sección 1.8). Una de las ventajas de utilizar la visión estereoscópica es que se pueden hallar directamente los puntos 3D, a diferencia de si se utiliza solo una cámara. En esos casos cuando se requiere la matriz de transformación, hay que aplicar ciertos

algoritmos para buscar la matriz a partir de un conjunto de puntos 2D (la proyección) y un conjunto inicial 3D. La precisión se puede ver afectada, puesto que una figura puede tener varias proyecciones perspectivas. Al contrario, si se cuenta con los puntos 3D, el objetivo es hallar una matriz de transformación que sea capaz de alinear un conjunto de puntos $P_1 \in \mathbf{R}^3$ en otro conjunto $P_2 \in \mathbf{R}^3$.

2.8.1. Cálculo de la transformación rígida

En el trabajo (SORKINE, 2009) se explica el algoritmo para hallar la transformación rígida que alinea dos conjuntos de puntos utilizando la descomposición SVD.

Teniendo a $P = \{p_1, p_2, ..., p_n\}$ y $Q = \{q_1, q_2, ..., q_n\}$ dos conjuntos de puntos correspondientes del espacio $\mathbf{R}^d$ se desea encontrar la transformación rígida que alinea a ambos conjuntos minimizando la función:

$$(R, t) = \sum_{i=1}^{n} \|(Rp_i + t) - q_i\|^2 \tag{2.8.1}$$

Basado en el trabajo estudiado se definió un algoritmo que cuenta con los siguientes pasos:

1. Se calculan los centroides de cada conjunto de puntos.

$$\bar{p} = \frac{\sum_{i=1}^{n} p_i}{n} \qquad \bar{q} = \frac{\sum_{i=1}^{n} q_i}{n} \tag{2.8.2}$$

2. Se calculan los vectores centrados.

$$x_i = p_i - \bar{p}, \qquad y_i = q_i - \bar{q}, \qquad i = 1, 2, ..., n. \tag{2.8.3}$$

3. Se calcula la matriz de covarianza $d \times d$.

$$S = XY^T, \tag{2.8.4}$$

   donde X y Y son matrices $d \times n$ que tienen a x_i y a y_i como sus columnas respectivamente.

4. Se realiza la descomposición de valores singulares $S = UWV^T$. D es una matriz auxiliar para eliminar posibles reflexiones de la rotación. La matriz resultante R es la rotación que se busca:

$$R = VDU^T, \qquad D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & det(VU^T) \end{bmatrix} \tag{2.8.5}$$

5. Se calcula la traslación óptima.

$$t = \bar{q} - R\bar{p} \tag{2.8.6}$$

2.9. Integración del componente de seguimiento con IGSTK

IGSTK cuenta con dos componentes fundamentales para utilizar dispositivos de seguimiento: *igstk::Tracker* e *igstk::TrackerTool* que son los encargados de establecer, mantener y cerrar la comunicación con el dispositivo; adquirir la información del seguimiento, y verificar la validez del dispositivo en todo momento.

Los componentes de IGSTK funcionan como un sistema síncrono, cada componente se ejecuta según su turno. Debido a esto, no existe garantía de que el procesamiento y la adquisición de una posición sean completados antes de que se haga la siguiente medición de posición. Además no se garantiza que la frecuencia del *Tracker* sea similar a la del componente de visualización de imágenes anatómicas del paciente. Por otra parte, cuando un componente envía un evento a otro componente, el segundo componente controla la ejecución del programa hasta que cede el control nuevamente al primero.

Para solucionar esta situación el componente *igstk::Tracker* utiliza un hilo independiente para la comunicación con el dispositivo de seguimiento, esto evita que el componente tenga que esperar a que se le pase el control de la aplicación para obtener los datos de posición y orientación de las herramientas. Este hilo independiente se utiliza para mantener la comunicación con el dispositivo de seguimiento y almacenar en un *buffer* los datos de medición de las herramientas. Cuando el hilo principal de la aplicación le solicita los datos de medición de las herramientas al componente *Tracker*, este le devuelve las mediciones más recientes del *buffer*. De esta manera se garantiza que no haya retardos en el proceso de obtención de los datos de medición, y que se mantenga una comunicación continua con el dispositivo de seguimiento. La figura 2.7 ilustra el funcionamiento del componente *Tracker* de IGSTK.

Cada una de las herramientas (marcadores en este caso) que van a ser rastreadas por el *Tracker* son representadas por el componente *TrakerTool*. El componente *Tracker* es el encargado de establecer la comunicación con el dispositivo de seguimiento, obtener las medidas de posición y orientación calculadas mediante la transformación rígida (ver sección 1.8) para cada herramienta y almacenarlas en el *buffer* interno.

Como resultado se obtienen los componentes *igstkStereoTracker* e *igstkStereoTrackerTool*, compatibles con aplicaciones basadas en QT, IGSTK y OpenCV.

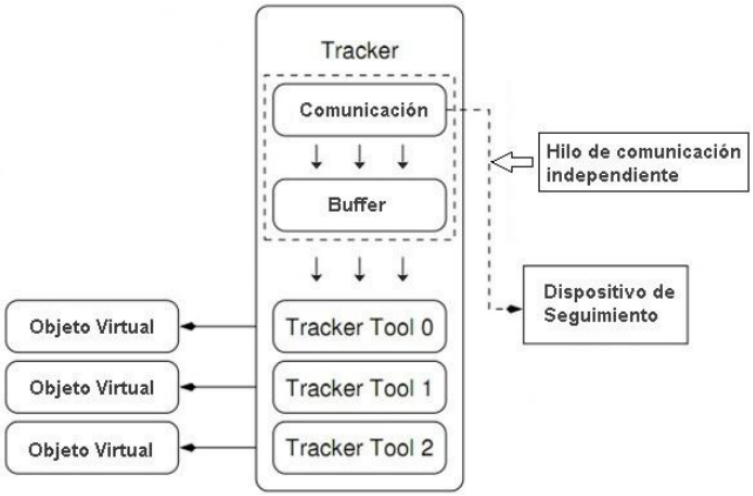


Figura 2.7. El componente *Tracker* de IGSTK (ENQUOBAHRIE y YANIV, 2009)

2.10. Ingeniería del sistema

La metodología que se eligió para organizar el trabajo fue XP, a partir de aquí se aplicarán las cuatro etapas que propone esta metodología, planificación, diseño, implementación y pruebas.

2.10.1. Historias de usuario

Las Historias de usuario (HU) tienen un propósito similar a los casos de uso. Estas se obtienen en la fase de diseño. Las escriben los propios clientes, tal como ven ellos las necesidades del sistema. Estas no se limitan a la interfaz de usuario, también conducirán el proceso de creación de los test de aceptación, que se emplean para verificar que las historias de usuario han sido implementadas correctamente. Según Kent Beck cada HU recoge los siguientes aspectos:

- Número: número asignado a la HU.
- Nombre de HU.
- Usuario: usuario del sistema que protagoniza o utiliza la HU.
- Prioridad en negocio: nivel de prioridad de la HU en el negocio. En caso de ser indispensable es alta, si no afecta el negocio en caso de no realizarse es media y baja si no constituye una prioridad.
- Riesgo en desarrollo: nivel de riesgo en caso de no realizarse la HU. Es alto cuando el riesgo implica en el funcionamiento de la plataforma. En caso de ser medianamente importante es medio el riesgo y bajo en caso de que no se considere un riesgo el hecho de tardar en la realización de la HU.

- Puntos estimados: estimación dada por el equipo de desarrollo del tiempo de duración de la HU, 1 equivale a una semana ideal de trabajo (5 días hábiles, 40 horas, 8 horas diarias).
- Iteración asignada: iteración a la que pertenece la HU.
- Programador responsable: responsable de la implementación de la HU.
- Descripción: describe lo que realizará la HU.
- Observaciones: casos especiales en el funcionamiento de la HU, flujos alternativos.

Tabla 2.1. Historia de usuario # 1

Historia de usuario	
Número: 1	**Nombre:** Calibrar el sistema estereoscópico
Usuario: Especialista	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 3	**Iteración asignada:** 1
Programador responsable: José Lozano Hernández	
Descripción: Permite calibrar el sistema estereoscópico utilizando la técnica del tablero de ajedrez. Primero se establecen todos los parámetros necesarios para la calibración: cantidad de filas y columnas del tablero, el tamaño de la casilla, que será la unidad de medida universal para el sistema y la cantidad de imágenes que se tomarán para la calibración. Luego se procesan todas las imágenes, adquiriendo todas las esquinas internas del tablero y se calibra el sistema.	
Observaciones: En caso de que alguna toma de imagen no se realice satisfactoriamente, es decir, que no se detecten todas las esquinas en alguna de ellas, el sistema emitirá una alerta para repetir la toma.	

Tabla 2.2. Historia de usuario # 2

Historia de usuario	
Número: 2	**Nombre:** Visualizar imágenes estereoscópicas rectificadas
Usuario:	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 2
Programador responsable: José Lozano Hernández	
Descripción: Con el sistema calibrado, es necesario rectificar cada par de imágenes. Una vez rectificadas, se deben mostrar las imágenes una al lado de la otra con las líneas epipolares superpuestas sobre ellas, para evaluar visualmente la rectificación.	
Observaciones: El observador debe fijarse que cada par de puntos correspondientes de las imágenes deben coincidir en una línea epipolar horizontal.	

Tabla 2.3. Historia de usuario # 3

Historia de usuario	
Número: 3	**Nombre:** Salvar la calibración del sistema
Usuario: Especialista	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.4	**Iteración asignada:** 2
Programador responsable: José Lozano Hernández	
Descripción: Cuando se tiene el sistema calibrado, es necesario tener persistencia de la calibración. El usuario escogerá el formato (XML ó YML) y la carpeta para guardar la calibración.	
Observaciones: En caso de error notificar al usuario.	

Tabla 2.4. Historia de usuario # 4

Historia de usuario	
Número: 4	**Nombre:** Cargar la calibración del sistema
Usuario: Especialista	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.6	**Iteración asignada:** 2
Programador responsable: José Lozano Hernández	
Descripción: El usuario selecciona la carpeta y el fichero de calibración en formato (XML ó YML). El sistema debe ser capaz de cargar una calibración previamente realizada.	
Observaciones: En caso de error notificar al usuario.	

Tabla 2.5. Historia de usuario # 5

Historia de usuario	
Número: 5	**Nombre:** Realizar el seguimiento espacial de los marcadores
Usuario:	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 2.8	**Iteración asignada:** 3
Programador responsable: José Lozano Hernández	
Descripción: Para realizar el seguimiento, el sistema debe contar con una calibración previa. Cuando se colocan los marcadores extendidos de la biblioteca ArUco frente a ambas cámaras el sistema debe ser capaz de detectarlos. Una vez halladas las coordenadas de sus esquinas en ambas imágenes, se procede a calcular los puntos en el espacio, asociados a cada esquina del marcador, esto se realiza mediante triangulación. Cuando se tienen todas las coordenadas 3D, se calcula la matriz de rotación y traslación asociada a cada marcador.	
Observaciones: El seguimiento espacial debe realizarse en tiempo interactivo a 30 *frames* por segundos (estándar para la mayoría de las cámaras). Si algún marcador no es detectado, el sistema debe seguir ejecutándose sin cambios.	

Tabla 2.6. Historia de usuario # 6

Historia de usuario	
Número: 6	**Nombre:** Implementar componente de seguimiento estereoscópico utilizando IGSTK
Usuario:	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1.2	**Iteración asignada:** 3
Programador responsable: José Lozano Hernández	
Descripción: Implementar un *tracker* de IGSTK basándose en las clases *igstkTracker* e *igstkTrackerTool*.	
Observaciones: Implementar los métodos virtuales puros de las clases correspondientes. El proceso debe estar desacoplado para poder dividirse en hilos y hallar las mediciones eficientemente.	

2.10.2. Plan de estimación y entrega

Siguiendo la metodología XP, se define la estimación de las iteraciones. Este plan tiene como objetivo mostrar la duración de cada una de las iteraciones en las que está dividida la fase de desarrollo del proyecto.

Tabla 2.7. Estimación de esfuerzo por historia de usuario

Iteración	**Historias de usuario**		**Puntos estimados (semanas)**
1	1	Calibrar el sistema estereoscópico	3
2	2	Visualizar imágenes estereoscópicas rectificadas	2
	3	Salvar la calibración del sistema	0.4
	4	Cargar la calibración del sistema	0.6
3	5	Realizar el seguimiento espacial de los marcadores	2.8
	6	Implementar componente de seguimiento estereoscópico utilizando IGSTK	1.2
Total			**10.0**

El orden de las tareas se define según sus dependencias con otras. En el presente proyecto se definieron tres iteraciones.

Tabla 2.8. Plan de duración de las iteraciones

Iteración	**Historias de usuario**		**Duración (semanas)**
1	1	Calibrar el sistema estereoscópico	3
2	2	Visualizar imágenes estereoscópicas rectificadas	3.0
	3	Salvar la calibración del sistema	
	4	Cargar la calibración del sistema	
3	5	Realizar el seguimiento espacial de los marcadores	4.0

Continúa en la próxima página

Tabla 2.8. Continuación de la página anterior

	6	Implementar componente de seguimiento estereoscópico utilizando IGSTK	
Total			**10.0**

Tabla 2.9. Plan de entrega

Historia de usuario	**Iteración 1 Del 9/2/2015 al 27/2/2015**	**Iteración 2 Del 2/3/2015 al 20/3/2015**	**Iteración 3 Del 23/3/2015 al 17/4/2015**
Calibrar el sistema estereoscópico	1ra entrega	-	-
Visualizar imágenes estereoscópicas rectificadas	-	2da entrega	-
Salvar la calibración del sistema	-	2da entrega	-
Cargar la calibración del sistema	-	2da entrega	-
Realizar el seguimiento espacial de los marcadores	-	-	3ra entrega
Implementar componente de seguimiento estereoscópico utilizando IGSTK	-	-	3ra entrega

2.11. Arquitectura y diseño

La arquitectura y diseño dentro del desarrollo de software es un proceso fundamental y se realizó en la fase de diseño de la metodología XP. Si se diseñó correctamente toda la estructura y el funcionamiento del software, la fase de implementación se ejecutará de una forma eficiente, dando claridad en las ideas y flexibilidad a los cambios. A continuación se presentan las tarjetas Clase - Responsabilidad - Colaborador (CRC), el patrón de arquitectura y los patrones de diseño utilizados en el desarrollo del componente de seguimiento estereoscópico.

2.11.1. Tarjetas CRC

El uso de tarjetas CRC constituye un modelo simple de organizar las clases más relevantes del sistema. Se utilizaron con el objetivo de representar organizadamente las clases. Estas presentan tres secciones:

- Clase: objeto que se analiza.
- Responsabilidad: cualquier función que la clase conoce o realiza.
- Colaborador: Conjunto de clases requeridas para que la clase reciba la información necesaria con el objetivo de completar una responsabilidad.

Tabla 2.10. Tarjeta CRC # 1

Tarjeta CRC	
Clase: StereoCalibrator	
Responsabilidad	**Colaboración**
• Establecer los parámetros iniciales de calibración como tamaño de la casilla, de imagen, y cantidad de casillas del tablero de ajedrez. • Tomar un conjunto N de pares de imágenes donde sea visible el tablero. • Detectar todas las esquinas en ambas imágenes. • Calibrar el sistema y obtener las matrices que arroja la calibración. • Adquirir los mapas de rectificación para cada cámara. • Salvar la calibración en formato YML ó XML. • Cargar la calibración en formato YML ó XML.	cv::findChessboardCorners cv::cornerSubPix cv::stereoCalibrate cv::stereoRectify cv:initUndistortRectifyMap cv::FileStorage

Tabla 2.11. Tarjeta CRC # 2

Tarjeta CRC	
Clase: StereoCapture	
Responsabilidad	**Colaboración**
• Establecer los parámetros iniciales de las cámaras (tamaño de imagen y frecuencia). • Capturar el *frame* correspondiente a cada cámara. • Convertir la imagen a escala de grises.	cv::VideoCapture cv::cvtColor

Tabla 2.12. Tarjeta CRC # 3

Tarjeta CRC	
Clase: StereoProcess	
Responsabilidad	**Colaboración**
• Adquirir una calibración previamente realizada. • Rectificar el par de imágenes utilizando los mapas de rectificación. • Calcular el mapa de disparidad (profundidad). • Se muestran las imágenes rectificadas y se dibujan las líneas epipolares horizontales para visualizar la rectificación.	cv::remap cv::StereoBM

Tabla 2.13. Tarjeta CRC # 4

Tarjeta CRC	
Clase: TrackerStereo	
Responsabilidad	**Colaboración**
• Establecer los parámetros iniciales del componente de seguimiento (Carga la calibración, inicia la captura, los parámetros de *StereoProcess* y la posición inicial del marcador en el espacio). • Detener el seguimiento. • Detectar los marcadores en ambas imágenes. • Triangular los puntos 2D correspondientes en cada imagen para hallar los puntos 3D. • Hallar la matriz de transformación asociada a cada marcador.	StereoCapture StereoProcess StereoCalibrator aruco::Marker aruco::CameraParameters aruco::MarkerDetector

Tabla 2.14. Tarjeta CRC # 5

Tarjeta CRC	
Clase: igstkStereoTracker	
Responsabilidad	**Colaboración**
• Abrir comunicación. • Iniciar seguimiento. • Reanudar seguimiento. • Actualizar estado de la máquina de estados. • Actualizar estado de la máquina de estados (hilo). • Adicionar *TrackerTool*. • Verificar *TrackerTool*. • Eliminar *TrackerTool*. • Detener seguimiento. • Cerrar comunicación.	igstk::Tracker igstk::StereoTrackerTool TrackerStereo

Tabla 2.15. Tarjeta CRC # 6

Tarjeta CRC	
Clase: igstkStereoTrackerTool	
Responsabilidad	**Colaboración**
• Obtener el ID del marcador. • Asignar nombre al marcador. • Reportar nombre inválido del marcador. • Chequear si el *TrackerTool* está configurado. • Reportar pedido inválido del componente de seguimiento.	igstk::TrackerTool igstk::StereoTracker

2.11.2. Arquitectura por capas

Una de las técnicas más comunes utilizadas por los arquitectos de software es el **patrón por capas**. Un sistema en capas permite establecer responsabilidades en cada una de ellas, dependiendo de la capa anterior, pero sin visibilidad para las capas superiores. Cada capa oculta las inferiores a ella a sus superiores. Utilizando esta arquitectura se minimizan las dependencias entre capas, se estandarizan los servicios y se reutilizan funcionalidades (FREEMAN; ROBSON; BATES y SIERRA, 2004).

IGSTK propone una arquitectura por capas (ENQUOBAHRIE y YANIV, 2009) como se muestra en la figura 2.8. La solución propuesta sigue este enfoque.

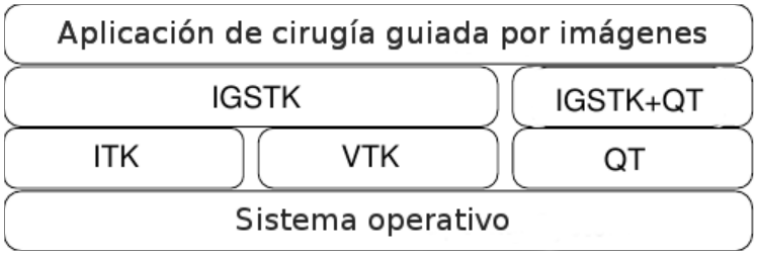


Figura 2.8. Arquitectura por capas de IGSTK (ENQUOBAHRIE y YANIV, 2009)

2.11.3. Patrones de diseño

Los patrones de diseño son considerados soluciones ya probadas a problemas de desarrollo de software sujeto a contextos similares. Los Patrones de software de asignación de responsabilidad general (GRASP) por sus siglas en inglés, se consideran una serie de buenas prácticas para el diseño de software (TABARES, 2011).

El patrón **experto** se usó en cada una de las clases que conforman el TrackerStereo, cada una está encargada de una serie de funcionalidades que solo ella conoce, con esto se asegura el encapsulamiento.

El uso del patrón **creador** se ve de manifiesto en las clases *TrackerStereo* e *igstkStereoTracker*. Estas clases son las encargadas de crear instancias de las subordinadas, no en el otro sentido.

Los patrones **alta cohesión** y **bajo acoplamiento** están estrechamente ligados, las responsabilidades de las clases están asignadas de tal forma, que cada una realice actividades específicas y exista la mínima dependencia con otras clases. Esto se ve de manifiesto en las clases *StereoProcess* y *StereoCapture*, que tienen un conjunto de funciones propias de ellas y dependen muy poco de otras clases.

El patrón **controlador** es utilizado para separar la capa de negocio de la capa de aplicación, este se pone de manifiesto en la clase *MainWindow* de la herramienta de calibración, la cual controla todo el proceso de

negocio asignando responsabilidades a las clases correspondientes.

Dentro de los patrones Gang of Four (GOF) se usó el **patrón estado** en la implementación de las clases *igstkStereoTracker* e *igstkStereoTrackerTool*. El concepto de máquina de estado fue introducido por Alan Turing en 1936. Una máquina de estado es definida por un conjunto de estados, entradas y transiciones de un estado a otro. Puede ser determinista si dada una entrada existe solo una transición a un único estado (FREEMAN; ROBSON; BATES y SIERRA, 2004). Esta biblioteca implementa este patrón para mejorar la seguridad y robustez en sus aplicaciones, limita los posibles comportamientos y asegura que el programa se encuentre siempre en una condición válida. La máquina de estados se definió mediante macros de la IGSTK. Una ventaja de este enfoque es la generación de un código válido y robusto.

2.12. Conclusiones parciales

En este capítulo se explicó qué métodos se utilizaron en la implementación de la herramienta de calibración, eligiendo *cv::stereoRectify* en vez de *cv::stereoRectifyUncalibrated* por la información geométrica añadida. Para la triangulación se decidió comparar el algoritmo iterativo-LS con la función cv::triangulatePoints y se seleccionó el iterativo-LS por ser más preciso. Se explicó como se integró el trabajo con la biblioteca IGSTK y la importancia del uso de sus macros para la construcción de la máquina de estados robusta.

CAPÍTULO 3

Implementación y pruebas

En este capítulo, se describen las tareas de ingeniería en la fase de implementación, con esto se organiza el proceso de implementación. También se realizaron pruebas al componente para calcular el error Root mean square (RMS) de la transformación rígida asociada a los marcadores.

3.1. Fase de implementación

La metodología XP propone que una vez que se realice una HU, es necesario probar y mostrar al cliente si se cumplieron con las especificaciones. Para esto, en la fase de implementación las HU se desglosan en tareas de ingeniería. A continuación se muestran las tablas que representan las tareas de ingeniería.

Tabla 3.1. Tarea de ingeniería # 1

Tarea	
Número de tarea: 1	**Número de Historia de usuario:** 1
Nombre de la tarea: Implementar la clase responsable de realizar la captura de las imágenes.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 9 de febrero de 2015	**Fecha de fin:** 10 de febrero de 2015
Descripción: La clase debe abrir la comunicación con las dos cámaras, proveer los *frames* actuales de cada cámara en formato normal y en escala de grises.	

Tabla 3.2. Tarea de ingeniería # 2

Tarea	
Número de tarea: 2	**Número de Historia de usuario:** 1
Nombre de la tarea: Implementar la clase responsable de calibrar el sistema estereoscópico.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.6
Fecha de inicio: 11 de febrero de 2015	**Fecha de fin:** 20 de febrero de 2015
Descripción: La clase obtiene los pares de imágenes y los parámetros iniciales para calibrar el sistema estereoscópico.	

Tabla 3.3. Tarea de ingeniería # 3

Tarea	
Número de tarea: 3	**Número de Historia de usuario:** 1
Nombre de la tarea: Implementar un calibrador estereoscópico con interfaz gráfica.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 23 de febrero de 2015	**Fecha de fin:** 27 de febrero de 2015
Descripción: Utilizando las clases anteriormente desarrolladas, se puede construir un sistema para calibrar cualquier par de cámaras utilizando un tablero de ajedrez.	

Tabla 3.4. Tarea de ingeniería # 4

Tarea	
Número de tarea: 4	**Número de Historia de usuario:** 2
Nombre de la tarea: Implementar la clase responsable de rectificar las imágenes estereoscópicas.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2
Fecha de inicio: 2 de marzo de 2015	**Fecha de fin:** 13 de marzo de 2015
Descripción: La clase toma los parámetros necesarios para la rectificación y el par de imágenes originales, devuelve las imágenes rectificadas, una unión con las líneas epipolares horizontales y el mapa de disparidad. Se le agregan las funcionalidades implementadas al calibrador con interfaz gráfica, este sistema debe mostrar las imágenes rectificadas y el mapa de disparidad para visualizar la calibración.	

Tabla 3.5. Tarea de ingeniería # 5

Tarea	
Número de tarea: 5	**Número de Historia de usuario:** 3
Nombre de la tarea: Implementar la funcionalidad de salvar la calibración en formato YML ó XML.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4

Continúa en la próxima página

Tabla 3.5. Continuación de la página anterior

Fecha de inicio: 16 de marzo de 2015	**Fecha de fin:** 17 de marzo de 2015
Descripción: Los parámetros adquiridos por el calibrador se deben guardar y así tener persistencia de la configuración. Esta funcionalidad se agrega a la clase CalibradorStereo	

Tabla 3.6. Tarea de ingeniería # 6

Tarea	
Número de tarea: 6	**Número de Historia de usuario:** 4
Nombre de la tarea: Implementar la funcionalidad de cargar la calibración en formato YML ó XML.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 18 de marzo de 2015	**Fecha de fin:** 20 de marzo de 2015
Descripción: Se adquiere la calibración guardada en formato YML ó XML.	

Tabla 3.7. Tarea de ingeniería # 7

Tarea	
Número de tarea: 7	**Número de Historia de usuario:** 5
Nombre de la tarea: Implementar la clase *TrackerStereo*	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.8
Fecha de inicio: 23 de marzo de 2015	**Fecha de fin:** 9 de abril de 2015
Descripción: Esta clase se encarga de detectar los marcadores, hallar las posiciones espaciales mediante triangulación, y calcular la transformación rígida para cada marcador.	

Tabla 3.8. Tarea de ingeniería # 8

Tarea	
Número de tarea: 8	**Número de Historia de usuario:** 6
Nombre de la tarea: Implementar clase *igstkStereoTrackerTool*	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 10 de abril de 2015	**Fecha de fin:** 14 de abril de 2015
Descripción: Esta clase representa una herramienta que es rastreada por el sistema de seguimiento. Se basa en la clase *igstkTrackerTool*.	

Tabla 3.9. Tarea de ingeniería # 9

Tarea	
Número de tarea: 9	**Número de Historia de usuario:** 6

Continúa en la próxima página

Tabla 3.9. Continuación de la página anterior

Nombre de la tarea: Implementar clase *igstkStereoTracker*	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 15 de abril de 2015	**Fecha de fin:** 17 de abril de 2015
Descripción: Esta clase realiza el seguimiento a las distintas herramientas y adquiere las mediciones para cada una de ellas. Se basa en la clase *igstkTracker*.	

3.2. Pruebas

El componente de seguimiento de un neuronavegador debe alcanzar una precisión alta, estos tienen una misión crítica y la vida de los pacientes corre peligro en caso de un funcionamiento incorrecto del software. Para que el cirujano aborde una lesión, debe pasar el instrumento quirúrgico a través de regiones de diferente valor funcional. Una medida con algunos milímetros de error, puede significar el acceso a zonas que no debían ser intervenidas y la ocurrencia de incidentes inesperados.

3.2.1. Pruebas realizadas

Para tener una medida de la precisión del componente implementado, se halló el error asociado a la transformación rígida correspondiente a un marcador. La transformación rígida es capaz de alinear dos conjuntos de puntos del mismo espacio, aplicando solamente rotaciones y traslaciones. Teniendo la matriz de rotación R, el vector de traslación t y los conjuntos de puntos P_a y P_b de N puntos cada uno, se transforma el conjunto de puntos inicial P_a en $P_{a'}$ con la siguiente formula:

$$P_{a'} = RP_a + t \tag{3.2.1}$$

El conjunto de puntos $P_{a'}$ debe estar alineado con P_b, pues se le aplicó la transformación rígida obtenida. En una situación con distorsión cero, no existiría error, o estaría muy cercano a cero. En este caso el conjunto de puntos asociado al marcador fue hallado aplicando técnicas para su detección en las imágenes distorsionadas, y luego se aplicó la triangulación. El error asociado a la transformación rígida se calcula mediante la raíz del error cuadrático medio, conocido por RMS:

$$error = \frac{\sqrt{\sum_{i=1}^{N}(P_{a'i} - P_{bi})^2}}{N} \tag{3.2.2}$$

Para la selección del método de triangulación, se compararon el método propuesto iterativo-LS, y el método implementado por OpenCV *cv::triangulatePoints*. Luego de 1658 iteraciones, se adquirieron los errores de transformación utilizando ambos métodos. La figura 3.1 muestra una comparación donde el error está dado en milímetros y se ve una clara diferencia que favorece al iterativo-LS.

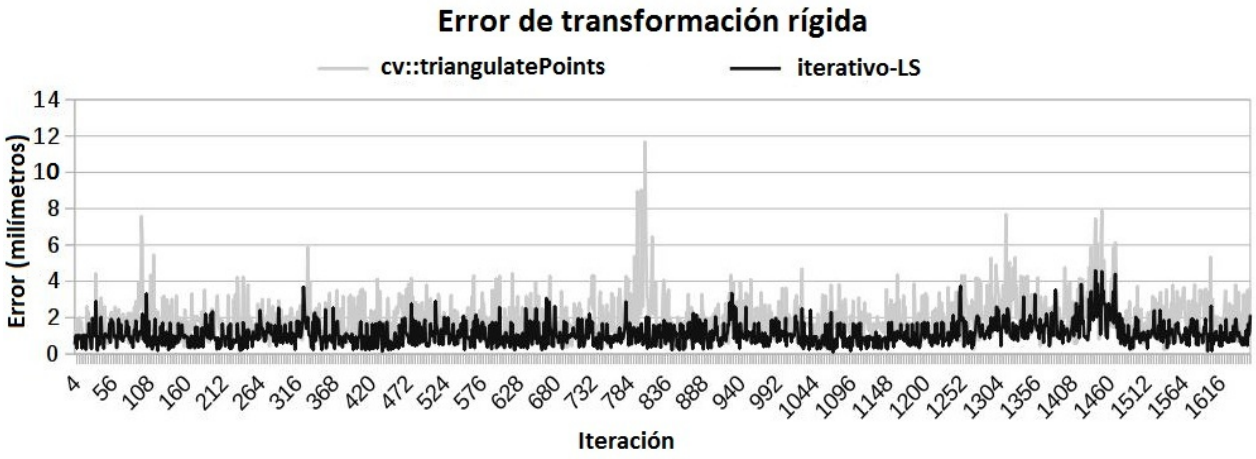


Figura 3.1. Error de transformación rígida utilizando dos algoritmos de triangulación, el *cv::triangulatePoints* de OpenCV y el iterativo-LS

Los resultados de la tabla 3.10 muestran que la media del error para el iterativo-LS fue de 1.11197 milímetros y para el cv::triangulatePoints fue de 1.97020 milímetros. La mediana es menor que la media, esto indica que la mayoría de los valores están por debajo de 1,11197 milímetros. Además los máximos y mínimos muestran también que los errores del método de OpenCV están esparcidos en un rango mayor. Las últimas dos columnas tienen el menor y mayor cuartil. Los cuartiles son los tres valores que dividen al conjunto de datos ordenados en cuatro partes porcentualmente iguales. Esto da una medida de qué por ciento de los datos están cercanos a la media.

Tabla 3.10. Análisis del error utilizando el método de triangulación cv::triangulatePoints y el iterativo-LS.

(medidas en milímetros)	Media	Mediana	Mínimo	Máximo	Menor cuartil	Mayor cuartil
cv::triangulatPoints	1.97020	1.75073	0.20421	11.70320	1.19696	2.51940
son iterativo-LS	1.11197	0.98098	0.07821	4.61566	0.74380	1.38326

Con el error adquirido no se puede realizar una comparación directa con los errores obtenidos en (MARTÍNEZ, 2012), pues en el error de registro para un neuronavegador interviene el error de la calibración de pivote. Con esta calibración se puede calcular la punta de la herramienta a partir de la posición del marcador colocado en ella. El error de registro en (ibíd.) del prototipo de neuronavegador fue de aproximadamente 4.17 milímetros. Aunque en el presente estudio falta el error de pivote, los resultados fueron mejorados considerablemente. Según (ENQUOBAHRIE y YANIV, 2009) el error de pivote depende en su gran mayoría del error de transformación.

Estos resultados son alentadores, teniendo en cuenta que las cámaras utilizadas tienen una resolución de 352×288 píxeles. Esta precisión puede mejorarse si se obtienen cámaras de mejor calidad, ya que la solución no depende del hardware. El componente todavía no cuenta con la precisión adecuada para ser utilizado en ambientes reales de neurocirugía, pero se demostró que el uso de técnicas estereoscópicas es un enfoque acertado.

Conclusiones

Con la realización del presente trabajo pueden ser enunciadas las siguientes conclusiones:

- Se desarrolló un componente de seguimiento estereoscópico utilizando la biblioteca IGSTK logrando un error medio de 1.112 milímetros, lo cual es un buen resultado, considerando que las cámaras utilizadas son de bajo costo.
- Se demostró que el uso de técnicas estereoscópicas en el seguimiento basado en video, puede mejorar considerablemente la precisión al adquirir la posición y orientación espacial de objetos en el mundo real.
- Se desarrolló una herramienta para calibrar sistemas estereoscópicos ópticos basados en grabación de video a nivel de software. Esta herramienta permitió conocer de forma visual e interactiva el modelo de proyección de las cámaras, como una etapa previa a la ejecución del componente estereoscópico.

Recomendaciones

Esta investigación puede contribuir a futuras investigaciones en el área de la visión estereoscópica, el seguimiento tridimensional de objetos y sistemas de seguimiento para la cirugía. Algunas recomendaciones para futuros trabajos son:

- Desarrollar un componente de seguimiento estereoscópico basado en visión infrarroja utilizando algunas funcionalidades implementadas en este trabajo como: la triangulación y el cálculo de la matriz de transformación.
- Probar el componente con un par de cámaras de alta definición y sincronización por hardware y comparar con los resultados obtenidos en esta investigación.

Glosario

buffer Memoria intermedia del sistema. 10, 38

DICOM Digital Imaging and Communications in Medicine. 27

DoF Por sus siglas en ingles, degrees of freedom. 13

fotogrametría Técnica para determinar las propiedades geométricas de los objetos y las situaciones espaciales a partir de imágenes fotográficas. 11

Hamming Código detector y corrector de errores que lleva el nombre de su inventor, Richard Hamming. En los datos codificados se pueden detectar errores en un bit y corregirlos. 36

SIFT Transformación de características invariantes a escalado, SIFT por sus siglas en inglés. 10

triangulación Hallar la posición de un punto en el espacio interceptando dos rayos correspondientes a dos proyecciones del mismo punto. 6, 21

USB Canal para el traspaso de datos universal. 31

Acrónimos

CAS cirugía asistida por computadora. 4

CRC Clase - Responsabilidad - Colaborador. 43

GOF Gang of Four. 48

GRASP Patrones de software de asignación de responsabilidad general. 47

HU Historias de usuario. 39, 49

IGSTK Imaged-Guided Surgery Toolkit. 8, 27, 30, 38

ITK Insight Segmentation and Registration Toolkit. 8

NCC Correlación cruzada normalizada. 20

RMS Root mean square. 49, 52

SAD Suma de diferencias absolutas. 20

SSD Suma de diferencias al cuadrado. 20

SVD Descomposición en valores singulares de una matriz. 22, 37

VTK Visualization Toolkit. 8

XML Extensible Markup Language. 27, 28, 32, 41

XP Extreme Programming. 28, 39, 42, 43

YML Formato de serialización de datos legible al humano. 28, 32, 41

Referencias bibliográficas

A. LORUSSO, D. W. Eggert y FISHER, R. B. A Comparison of Four Algorithms forEstimating 3-D Rigid Transformations. En. *A Comparison of Four Algorithms forEstimating 3-D Rigid Transformations*.

AVA. (Visited on 06/02/2015). Url: ⟨http://www.uco.es/investiga/grupos/ava/node/1⟩.

BAGGIO, Daniel Lélis. 2012. *Mastering OpenCV with practical computer vision projects*. 2012.

BAILEY, Brenton y WOLF, Alexander. 2007. Real Time 3D motion tracking for interactive computer simulations. *Unter: http://www3. imperial. ac. uk/pls/portallive/docs/1/20673697. PDF (abgerufen am 21.07. 2014)*. 2007, vol. 3, n.º 1, págs. 3-1.

BARNARD, Stephen T y FISCHLER, Martin A. 1982. Computational stereo. *ACM Computing Surveys (CSUR)*. 1982, vol. 14, n.º 4, págs. 553-572.

BRADSKI, Gary y KAEHLER, Adrian. 2008. *Learning OpenCV: Computer vision with the OpenCV library*. 2008.

BUCHOLZ, RD y SMITH, KR. 1993. A comparison of sonic digitizers versus light emitting diode-based localization. 1993, págs. 179-200.

CARBAJAL, Guillermo; GOMEZ, Alvaro; PEREYRA, Gabriela et al., 2010. First neuronavigation experiences in uruguay. En. *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*. 2010, págs. 2317-2320.

Claron Technology Inc. (Visited on 06/02/2015). Url: ⟨http://www.clarontech.com⟩.

CMake. (Visited on 06/20/2015). Url: ⟨http://www.cmake.org/⟩.

DÍAZ, Joaquin Alberto. 2014. *Componente de software para Realidad Aumentada, basado en la detección de características naturales en imágenes.* 2014.

ENDRESEN, Kai Hugo Hustoft. 2010. *Tracking objects in 3D using Stereo Vision*. 2010.

ENQUOBAHRIE, Andinet y YANIV, Ziv. 2009. *IGSTK: The Book*. 2009.

FONSECA, Adrián. 2011. *Adquisición de modelos geométricos a partir de objetos del mundo real*. 2011.

FREEMAN, Eric; ROBSON, Elisabeth; BATES, Bert y SIERRA, Kathy. 2004. *Head first design patterns*. 2004.

FUA, Pascal y LEPETIT, Vincent. 2005. Vision based 3D tracking and pose estimation for mixed reality. *Emerging Technologies of Augmented Reality Interfaces and Design*. 2005, págs. 43-63.

GARRIDO-JURADO, S.; MUÑOZ-SALINAS, R.; MADRID-CUEVAS, F.J y MARÍN-JIMÉNEZ, M.J. *Automatic generation and detection of highly reliable fiducial markers under occlusion.*

GÓMEZ, Alvaro y RANDALL, Gregory. 2004. NAVEGACIÓN GUIADA POR IMÁGENES EN NEUROCIRUGÍA. 2004.

HARTLEY, Richard I y STURM, Peter. 1997. Triangulation. *Computer vision and image understanding*. 1997, vol. 68, n.º 2, págs. 146-157.

HARTLEY, Richard y ZISSERMAN, Andrew. 2003. *Multiple view geometry in computer vision*. 2003.

HIRSCHMULLER, Heiko; INNOCENT, Peter R y GARIBALDI, Jon. 2002. Real-time correlation-based stereo vision with reduced border errors. *International Journal of Computer Vision*. 2002, vol. 47, n.º 1-3, págs. 229-246.

JACOB, Ann Maria y ANITHA, J. 2012. Inspection of Various Object Tracking Techniques. *International Journal of Engineering and Innovative Technology*. 2012, vol. 2, págs. 118-124.

JALAL, Anand Singh y SINGH, Vrijendra. 2012. The State-of-the-Art in Visual Object Tracking. *Informatica (Slovenia)*. 2012, vol. 36, n.º 3, págs. 227-248.

JOSKOWICZ, José. 2008. Reglas y prácticas en eXtreme Programming. *Universidad de Vigo*. 2008, págs. 22.

KABSCH, W. 1978. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*. 1978, vol. 34, n.º 5, págs. 827-828. Url: ⟨http://dx.doi.org/10.1107/S0567739478001680⟩.

KANADE, Takeo y OKUTOMI, Masatoshi. 1994. A stereo matching algorithm with an adaptive window: Theory and experiment. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*. 1994, vol. 16, n.º 9, págs. 920-932.

KOLLER, Dieter; KLINKER, Gudrun; ROSE, Eric et al., 1997. Real-time vision-based camera tracking for augmented reality applications. En. *Proceedings of the ACM symposium on Virtual reality software and technology*. 1997, págs. 87-94.

LETELIER, Patricio. 2006. Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). 2006.

Logitech International S.A. Url: ⟨http://www.logitech.com⟩.

LÓPEZ-VALLES, José M; FERNÁNDEZ-CABALLERO, Antonio y FERNÁNDEZ, Miguel A. 2005. Conceptos y técnicas de estereovisión por computador. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*. 2005, vol. 9, n.º 27, págs. 35-62.

MARTÍNEZ FERNÁNDEZ DE LAS HERAS, José Javier. 2005. La descomposición en valores singulares (SVD) y algunas de sus aplicaciones. *Gaceta de la Real Sociedad Matemática Española*. 2005, vol. 8, n.º 3, págs. 796-810.

MARTÍNEZ, Fidel. 2012. *Sistema de Registro para software de Navegación Quirúrgica*. 2012.

MATTHIES, Larry; KANADE, Takeo y SZELISKI, Richard. 1989. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*. 1989, vol. 3, n.º 3, págs. 209-238.

Medtronic. Url: ⟨http://http://www.medtronic.com⟩.

MONTALVO, M. 2010. *Técnicas de visión estereoscópica para determinar la estructura tridimensional de la escena*. 2010.

MORENO, Mileydi y CRUZ GUEVARA, Ernesto de la. 2008. *Localización de objetos virtuales en el mundo real con técnicas de Realidad Aumentada.* 2008.

NICOLAU, SA; GOFFIN, Laurent y SOLER, Luc. 2005. A low cost and accurate guidance system for laparoscopic surgery: Validation on an abdominal phantom. En. *Proceedings of the ACM symposium on Virtual reality software and technology*. 2005, págs. 124-133.

Northern Digital. (Visited on 06/02/2015). Url: ⟨http://www.ndigital.com/medical/products/polaris-family⟩.

OKUTOMI, Masatoshi y KANADE, Takeo. 1993. A multiple-baseline stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*. 1993, vol. 15, n.º 4, págs. 353-363.

Open Source Augmented Reality SDK. (Visited on 06/02/2015). Url: ⟨http://www.artoolkit.org/⟩.

PAJARES, G y CRUZ, JM. 2007. *Visión por Computador: Imágenes digitales y aplicaciones , RA-MA*. 2007.

PAREKH, Himani S; THAKORE, Darshak G y JALIYA, Udesang K. 2014. A Survey on Object Detection and Tracking Methods. *International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3297: 2007 Certified Organization) Vol*. 2014, vol. 2.

PETERS, Terry y CLEARY, Kevin. 2008. *Image-guided interventions: technology and applications*. 2008.

Qt Cross-platform application. (Visited on 06/18/2015). Url: ⟨http://www.qt.io/⟩.

SAÍZ, M. 2010. *Reconstrucción tridimensional mediante visión estéreo y técnicas de optimización*. 2010.

SCHARSTEIN, Daniel y SZELISKI, Richard. 2002. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*. 2002, vol. 47, n.º 1-3, págs. 7-42.

SEHNERT, Shelly. 2008. Triangulation. 2008.

SEN-CHING, S Cheung y KAMATH, Chandrika. 2004. Robust techniques for background subtraction in urban traffic video. En. *Electronic Imaging 2004*. 2004, págs. 881-892.

SORKINE, Olga. 2009. *Least-Squares Rigid Motion Using SVD*. 2009.

STROUSTRUP, Bjarne. 1986. *The C++ programming language*. 1986.

SZELISKI, Richard. 2010. *Computer vision: algorithms and applications*. 2010.

TABARES, Ricardo Botero. 2011. Patrones Grasp y Anti-Patrones: un Enfoque Orientado a Objetos desde Lógica de Programación. *Entre Ciencia e Ingeniería*. 2011, n.º 8, págs. 161-173.

The Official YAML Web Site. (Visited on 06/18/2015). Url: ⟨http://yaml.org/⟩.

TREVISAN, Daniela G; NEDEL, Luciana P y MACQ, Benoit. 2008. Augmented vision for medical applications. En. *Proceedings of the 2008 ACM symposium on Applied computing*. 2008, págs. 1415-1419.

VITE-SILVA, Israel; CRUZ-CORTÉS, Nareli; TOSCANO-PULIDO, Gregorio y FRAGA, Luis Gerardo de la. 2007. Optimal triangulation in 3D computer vision using a multi-objective evolutionary algorithm. En. *Applications of Evolutionary Computing*. 2007, págs. 330-339.

WIDMANN, Gerlig; STOFFNER, Rudolf y BALE, Reto. 2009. Errors and error management in image-guided craniomaxillofacial surgery. *Oral Surgery, Oral Medicine, Oral Pathology, Oral Radiology, and Endodontology*. 2009, vol. 107, n.º 5, págs. 701-715.

YILMAZ, Alper; JAVED, Omar y SHAH, Mubarak. 2006. Object tracking A survey. *Acm computing surveys (CSUR)*. 2006, vol. 38, n.º 4, págs. 13.

Generado con LaTeX: 24 de junio de 2015: 12:12 Noon