VNIVERSIDAD NACIONAL
AVTNOMA DE
MEXICO

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

"Desarrollo de un Sistema de Realidad Aumentada Basado en la Colocación de Objetos Virtuales en Escenas Reales a Partir de la Obtención de Nubes de Puntos en una Secuencia de Imágenes"

T E S I S

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN INGENIERÍA (COMPUTACIÓN)

P R E S E N T A:

JOSÉ ISRAEL FIGUEROA ANGULO

DIRECTOR DE TESIS: Ana Luisa Solís González-Cosío

México, D.F. **2009.**

Desarrollo de un Sistema de Realidad Aumentada Basado en la Colocación de Objetos Virtuales en Escenas Reales a Partir de la Obtención de Nubes de Puntos en una Secuencia de Imágenes

Autor: Ing. José Israel Figueroa Angulo
Director de Tesis: Mat. Maria Concepción Ana Luisa Solís González Cosío

22 de enero de 2009

Agradecimientos

A mis padres. Quienes me apoyaron y vieron por mí durante mi maestria, a pesar de que no estaban conmigo en persona... No vayan a pensar mal, ellos viven en Guadalajara.

A mi tutora, la Mat. Ana Luisa Solís González Cosío. Por el apoyo y confianza que depositó en mí durante este proyecto.

Al PhD. Rob Hess, de la Universidad del Estado de Oregon, EE. UU. Por haberme dado información esencial de su implementación del algoritmo SIFT.

A los Doctores María Elena Martínez Pérez y Yann Frauel. Por la información que me proporcionaron sobre Visión Computacional.

A mi amigo, Iván Christian Cervantes Coronado.

Índice general

Resumen

Con los avances en Gráficos Computarizados que se han producido desde la década de 1950, hoy en día es posible crear imágenes que sólo existían en de forma imaginaria, así como extender nuestro sentido de la vista, mostrando cosas más allá del alcance de nuestros ojos.

Hoy en día, se ha vuelto cada vez más común sobreponer objetos virtuales sobre imágenes grabadas del mundo real, con el objetivo de mostrar información adicional al espectador; estos objetos se colocan de forma que parezcan que están sobre la escena, aunque éstos no existan en el lugar de la grabación. Esta integración de objetos virtuales en escenas reales es producto de los desarrollo de la Realidad Aumentada, una de las innovaciones de la rama de Cómputo Visual, resultado de tecnologías de la Graficación y Visión Computacional, que estudia las técnicas para integran información adicional, en forma de objetos virtuales, sobre escenas del mundo real.

El **objetivo** de esta tesis es desarrollar un sistema de Realidad Aumentada que utilice elementos del escenario como puntos de referencia para orientar los objetos virtuales. El motivo para desarrollar esto radica en que muchos sistemas de Realidad Aumentada disponibles para el público utilizan marcas artificiales que destacan sobre el escenario, las cuales son fáciles de captar y analizar por un sistema computarizado. El problema con esto es que no es deseable en situaciones donde no está permitido alterar el escenario que se está grabando, por lo que debemos usar otro enfoque para obtener puntos de referencia para los objetos virtuales.

La **contribución** de este trabajo es tener un sistema de Realidad Aumentada que funcione en cualquier clase de escenario, utilizando los elementos existentes en el mismo como puntos de referencia para los objetos virtuales. Su relevancia es que sentará las bases para nuevas aplicaciones de Realidad Aumentada, tales como: aplicaciones multimedia o videojuegos que funcionen en espacios abiertos, integración de objetos virtuales en producciones audiovisuales (TV, cine).

La tesis está organizada de la siguiente forma: el **primer capítulo** muestra la definición de la Realidad Aumentada y una breve historia sobre ésta, seguido por las aplicaciones civiles y militares de la Realidad Aumentada, después de eso, se habla sobre los detalles de la Realidad Aumentada con marcas y, por último, se habla de la Realidad Aumentada sin marcas y su funcionamiento básico. El **segundo capítulo** hace un análisis de las bibliotecas de Realidad Aumentada más comunes y disponibles al público, el análisis está centrado en

las técnicas que usan para el rastreo y el registro. El **tercer capítulo** describe los componentes del sistema que se desarrolla para esta tesis, los algoritmos utilizados en los componentes centrales del sistema de rastreo y la teoría tras los algoritmos usados en el sistema. En el **cuarto capítulo** se presentan los detalles del diseño y la implementación del software de Realidad Aumentada, enfatizando la configuración del rastreador y el sistema gráfico. El **quinto capítulo** trata sobre las pruebas hechas en el sistema para analizar su funcionalidad y, por último, en el **sexto capítulo** se presentan sobre las conclusiones obtenidas en esta tesis.

Capítulo 1

Realidad Aumentada

La Realidad Aumentada (AR, por sus siglas en inglés) es una tecnología que integra imágenes de objetos virtuales en imágenes del mundo real [BKM99], es decir, el usuario ve el mundo real con objetos virtuales superpuestos o compuestos dentro del mismo, en lugar de presentarle un entorno completamente generado por computadora como sucede en la realidad virtual (VR, por sus siglas en inglés).

Un sistema de AR tiene las siguientes características [Azu95]

1. Combina información real y virtual.
2. Es interactivo en tiempo real.
3. El espacio tridimensional virtual y el espacio tridimensional real están enlazados para que ambos concuerden en posición y orientación, a esta relación espacial se le llama registro [BR05]. Mantener un registro correcto y consistente entre los objetos virtuales y el mundo real es importante para que la AR sea creíble por el usuario y el sistema pueda funcionar para usuarios que se estén moviendo, para lo cual también se necesita que el sistema esté rastreando constantemente la posición y punto de vista del usuario para que los cambios de orientación y posición se reflejen en la AR.

1.1. Historia

Los inicios de la realidad aumentada datan de la década de 1960, cuando Morton Heilig, un cinematógrafo, crea un simulador de motocicleta llamado Sensorama en 1962, el cual daba al espectador la experiencia de montar una motocicleta por las calles de Brooklin. El espectador sentía el viento en su cara, la vibración del asiento de la motocicleta, una vista tridimensional de la ciudad y hasta olores de la ciudad [Wik07]. En 1966, Ivan Sutherland y Robert Sproull crearon la primer pantalla de vídeo montada en la cabeza (Head-Mounted Display, HMD) para el sistema Sketchpad, el cual permitía ver al

mismo tiempo el mundo real y gráficos vectoriales, proyectados sobre de los lentes del mismo [Wik02].

Durante las décadas de 1960 y 1970, la fuerza aérea de los EE. UU., bajo la dirección del ingeniero eléctrico Thomas Furness, diseñó los primeras Pantallas proyectadas a la altura de la cabeza (Head-Up Displays, HUD) y miras montadas en cascos (Helmet-Mounted Sights, HMS) para mostrar información de vuelo, que también aparecía en en tablero de controles.

En la década de 1980, Thomas Furness dirige el Proyecto Supercockpit de la Fuerza Aérea de los EE. UU. para crear un sistema de presentación de datos que tomara ventaja de los mecanismos perceptuales del ser humano. Utilizando el HMD para lograrlo, diseñó un sistema que proyectaba información como mapas generados en 3-D, imágenes generadas por radar y radiación infrarroja e información de los instrumentos del avión en un entorno tridimensional virtual inmersivo que el piloto podía ver y oír en tiempo real [Bri08].

A inicios de la década de 1990, Thomas Caudell, ingeniero de la Boeing, acuña el término "Realidad Aumentada" mientras ayuda a los trabajadores a ensamblar los paneles de cables de las aeronaves [Wik02]. También durante esa década surgen los primeros sistemas médicos que permiten visualizar información de ultrasonido directamente sobre el cuerpo del paciente, los cuales fueron creados por la Universidad de North Carolina. En 1996, el MIT crea las primeras computadoras que pueden llevarse sobre la persona (Wearable Computer). A finales de esta década surgen los primeros sistemas de colaboración remota y las transmisiones deportivas se benefician con los primeros sistemas que colocan información virtual sobre la cancha.

1.2. Aplicaciones

La combinación de lo real y lo virtual permite mejorar la percepción e interacción del usuario con el mundo real. Los objetos virtuales proporcionan información que no puede detectarse directamente a través de los sentidos, además la información provista por los objetos ayuda a ejecutar tareas del mundo real [Azu95].

1.2.1. Medicina

Los médicos pueden usar aplicaciones de AR como ayuda para visualización tanto en entrenamiento como en cirugía. Se pueden obtener datos tridimensionales sobre el paciente en tiempo real, usando técnicas de imagenología no invasivas como Resonancia Magnética, Tomografía Computarizada o imágenes de Ultrasonido; estos datos pueden combinarse y representarse como un modelo tridimensional que es colocado sobre el paciente, lo cual proporcionaría al médico una vista del interior del paciente sin abrirlo. Esta clase de vista sería muy útil para cirugía no invasiva, la cual reduce el trauma de una operación mediante el uso mínimo o nulo de incisiones. El problema con las técnicas de cirugía no

invasiva es que el médico no puede ver el interior del paciente con facilidad, lo cual hace más difícil la ejecución de la cirugía.

También la tecnología de AR puede ser útil para tareas de visualización dentro de la sala de operaciones. Los cirujanos pueden detectar a simple vista algunas características que no pueden verse en imágenes obtenidas por técnicas de imagenología. La AR permitiría a los médicos ver las diferentes tipos de imágenes capturadas por imagenología al mismo tiempo. Esta tecnología también ayudaría en operaciones que requieren alto grado de precisión, como biopsias o cirugía cerebral, mostrando sobre el paciente los lugares por donde debe orientar los instrumentos.

Otra aplicación de la AR es para entrenamiento de procedimientos quirúrgicos, el sistema de AR mostraría las instrucciones del procedimiento directamente sobre el sujeto de entrenamiento, de esta forma se evita retirar la vista para consultar un manual o instructivo; los objetos virtuales también puede servir de indicación para identificar zonas que no deben ser tocadas durante el entrenamiento [Azu95].

1.2.2. Manufactura y reparación

Tradicionalmente, la capacitación en manufactura y reparación de productos industriales se ha hecho mediante manuales impresos con instrucciones detalladas y un modelo de práctica del producto. Los problemas que hay con esta clase de capacitación son: el aprendiz tiene que desviar su atención del modelo de práctica para leer el manual y memorizar una o varias acciones antes de continuar con el entrenamiento; se debe fabricar un modelo de práctica por cada producto, aún si se trata de variaciones del mismo; toda la información que aparece en el manual de capacitación es complementada con imágenes de dos dimensiones, así que no es posible ver la ejecución de una acción desde varios puntos de vista.

Utilizando AR como medio para presentar los pasos de un procedimiento directamente sobre el entorno de trabajo resulta en una capacitación más directa e intuitiva, independientemente de la cantidad de información de capacitación que se genere, además de que las instrucciones se presentaran en el lugar y tiempo adecuado, no habría que hacer una imagen mental de la ejecución de la instrucción y la acción podría verse desde diferentes puntos de vista para que el usuario comprenda las relaciones visuales implicadas en la instrucción [BKM99].

1.2.3. Anotación y visualización

La AR puede usarse para mostrar o visualizar la imagen de anotaciones sobre objetos y ambientes usando información pública o privada, en el caso de la información pública se supone la existencia de bases de datos públicas y la información privada puede venir de anotaciones del usuario.

Como ayuda para visualización, la AR puede utilizarse para mostrar información que no está a simple vista, como los sistemas de tuberías que hay dentro de un edificio, información que en ese momento no existe, como la forma

y ubicación de un futuro edificio sobre un terreno, y también como ayuda visual para navegación de un vehículo en condiciones adversas de visibilidad, como niebla o bajo el agua [Azu95].

1.2.4. Planeación de movimientos de robots

Un problema con la operación remota de robots es que no es posible operarlos en tiempo real debido a los retrasos en las comunicaciones entre el operador y el robot. Una forma de resolver este problema es que el operador controle localmente una versión virtual del robot (la cual está sobrepuesta en el mundo real con AR), en la cual el usuario especifica sus acciones en tiempo real. Los resultados son mostrados en el mundo real. Una vez que el plan ha sido probado y determinado, el usuario indica al robot que ejecute el plan especificado. El entorno virtual también sirve para predecir los efectos de las acciones en el entorno, así que el sistema de AR sirve como herramienta de planeación para ayudar a ejecutar tareas [Azu95].

1.2.5. Entretenimiento

Los sistemas de AR se pueden usar para generar escenarios virtuales que permiten interactuar a actores reales con fondos y objetos virtuales, en tiempo real y en 3-D. Los actores se paran frente a una cámara controlada por computadora, la cual registra sus movimientos. Una vez que se ha rastreado la posición de la cámara y se han capturado los movimientos de los actores, es posible hacer una composición del actor real dentro del mundo virtual en 3-D [Azu95].

1.2.6. Aeronaves militares

Desde la década de 1960, los pilotos de aviones y helicópteros militares, sobre todo del ejército de los E.E.U.U, han usado pantallas proyectadas a la altura de la cabeza (Head-Up Displays) y m montadas en cascos (Helmet-Mounted Sights) para sobreponer gráficos vectoriales sobre los ojos del piloto. Además de proveer información básica de navegación y vuelo, a estos gráficos frecuentemente se les registra con los blancos en el entorno, proporcionándoles una forma de apuntar las armas de la aeronave [Azu95].

1.3. Realidad Aumentada con marcas fiduciarias

Una de las técnicas de registro y rastreo para AR basada en visión computacional más fáciles de usar es mediante el uso de marcas artificiales que destacan sobre el escenario, llamadas marcas fiduciarias. Estas marcas tienen una forma y color conocidos para que el sistema de AR las reconozca con facilidad y así sea posible calcular con precisión el punto de vista del usuario en tiempo real.

Uno de los estilos de marca más comunes es un patrón a blanco y negro dentro de un cuadro negro con un borde de ancho constante, este tipo de marca destaca muy bien sobre cualquier clase de fondo, es fácil de detectar mediante umbralización a blanco y negro y la estimación de pose se basa en la detección de la forma del cuadro y en obtener la matriz de proyección a partir de la orientación del patrón que está dentro del cuadro.

Los ejemplos más conocidos de software de AR que usa esta clase de estructuras de estimación de pose son ARToolkit (HITLab, Nueva Zelanda), el cual usa cuadros que contienen patrones de forma arbitraria, los cuales deben ser aprendidos por el software antes de iniciar la detección; ARToolkitPlus —ahora Studierstube Tracker— (Graz University of Technology, Austria) y ARTag (Institute of Information Technology, NRC Canada) usan patrones preestablecidos, que son semejantes a códigos de barras bidimensionales; esta técnica tiene la ventaja de que no hay que aprender nuevos patrones y la detección de los mismos es muy rápida.

1.4. Realidad Aumentada sin marcas fiduciarias

La siguiente tendencia en Realidad Aumentada es prescindir de las marcas fiduciarias como puntos de referencia para el registro y el rastreo. Los problemas con el uso de estas estructuras son que éstas no se integran perfectamente en el escenario, además de que deben registrarse en el programa, lo cual puede resolverse con un conjunto grande de marcas predeterminadas; también se requiere calibrar la cámara para que el registro sea adecuado y, por último, estas estructuras son sensibles a la oclusión y a los cambios de iluminación, es decir, cualquier factor físico o fotogramétrico que altere la continuidad de la marca hace imposible que ésta sea registrada.

Las alternativas al uso de marcas fiduciarias se basan en el uso de otras técnicas de visión computacional para obtener el registro y rastreo del punto de vista del usuario, tales como el rastreo de características y la reconstrucción tridimensional de objetos.

1.4.1. Sobre las características

Una característica es un tipo de estructura particular en una imagen, tales como bordes, esquinas o regiones. Las características se pueden rastrear en una secuencia de imágenes; de acuerdo con el criterio de Jianbo Shi y Carlo Tomasi, una buena característica es aquella que pueda ser rastreada bien. Teniendo buenas características para rastrear se pueden ubicar elementos virtuales en el espacio de imagen y también obtener una reconstrucción tridimensional de un objeto a partir de una serie de imágenes.

1.4.2. Problemas en el rastreo de características

Los dos principales problemas cuando se rastrean características son la *variación del color,* es decir, los colores de una característica pueden variar por la orientación y color de la luz; y la *variación de la forma*, en la cual las características pueden modificar su tamaño, posición y ángulo dependiendo del punto de vista.

La solución a estos problemas son los siguientes: para la variación de color se calcula el cambio de color promedio de cada característica entre cuadros, siempre y cuando que la variación sea uniforme, ya que las propiedades ópticas de los objetos en la escena influyen en la forma en que varía el color de la imagen, pudiendo resultar en características falsas; para solucionar la variación de la forma se aplica a cada característica las propiedades afines: rotación, translación, escala y corte, lo cual no funciona en los casos donde la imagen es deformada no uniformemente, como sucede en el caso de las superficies flexibles y superficies no planas.

Capítulo 2

Sistemas de Realidad Aumentada

En esta sección, se va a hacer un análisis de los métodos que usan los sistemas de Realidad Aumentada más conocidos y disponibles al público.

2.1. ARToolkit

ARToolkit es un conjunto de módulos para Realidad Aumentada basada en marcadores, desarrollada por Hirokazu Kato, de la Universidad de Hiroshima, y Mark Billinghurst, de la Universidad de Washington; los antecedentes de este software datan de 1999 [KBP+00, KB99], el cual corría en equipos Silicon Graphics sobre Irix. A partir de la versión 2.52, se vuelve disponible para otros sistemas operativos, Windows, Linux y MacOS X. Su popularidad entre los desarrolladores de software de Realidad Aumentada se basa en que es sencillo, relativamente robusto y está disponible de forma gratuita.

Su funcionamiento se basa en el procesamiento de imágenes de marcadores cuyo tamaño ya es conocido. Cada marcador tiene un patrón que debe ser almacenado en la computadora para comparar posteriormente.Cuando se captura un cuadro, se conviertela imagen a color en una imagen binaria a blanco y negro, mediante umbralizado adaptativo en escala de grises [Wel93], el cual se utiliza para detectar los componentes conexos cuyo nivel de gris es menor al parámetro del umbral (Fig. 2.1-1). El siguiente paso consiste en extraer los contornos del cuadro para después encontrar sus esquinas (Fig. 2.1-2 y 3), las cuales se utilizan para eliminar la distorsión de perspectiva y obtener la vista canónica frontal del patrón [Fia05]. Después la región es normalizada y se hace corresponder el contenido de la marca con alguno de los patrones registrados mediante correlación, una vez identificado el patrón, se posicionan y orientan los objetos con respecto a la posición y orientación del marcador.

A pesar de que el sistema no funciona bien en condiciones de oclusión o iluminación variable, ha servido como base para desarrollos más avanzados de

Realidad Aumentada.

2.2. ARTag

ARTag es una biblioteca de software para Realidad Aumentada desarrollada por Mark Fiala del Grupo de Vídeo Computacional, del Consejo Nacional Canadiense de Investigación en el año 2004 [Fia05].

La biblioteca está basada en ARToolkit y contiene una serie de mejoras en el sistema de detección de patrones que lo hace más robusto que éste: dos mil dos patrones de 36 bits, acomodados en una cuadricula de 6x6, que eliminan el aprendizaje previo de patrones, detección de contornos y patrones basada en detección de bordes (Fig 2.2-2), esto permite detectar patrones en condiciones de iluminación variable (Fig. 2.3); códigos de redundancia y corrección de errores hacia adelante para seguir detectando los códigos de los patrones, detección de patrones aún en condiciones de oclusión gracias a un sistema de heurística de segmentos de línea, detección de cuadros de patrón a 3 diferentes escalas para reducir la tasa de falsos positivos [Fia05].

2.3. ARToolkitPlus

ARToolkitPlus es una biblioteca para Realidad Aumentada desarrollada por Daniel Wagner y Dieter Schmalstieg, del Instituto para la Graficación y Visión Computacional de la Universidad de Tecnología Graz, en Austria, como parte del proyecto "Handheld AR". Su última versión data de Junio del 2006, ya que el proyecto fue sucedido por el Studierstube Tracker.

El código de detección de patrones está basado en ARTag e incorpora una serie de características que están enfocadas para su uso en dispositivos móviles: las funciones de estimación de la pose de la cámara fueron reescritas para que utilizaran aritmética de punto fijo, ya que muchos dispositivos móviles carecen de una Unidad de Punto Flotante (FPU); soporte para formatos de píxeles existentes en las cámaras de dispositivos móviles, como RGB565 e YUV12, un conjunto fijo de 4096 marcadores que eliminan la necesidad de memorizar patrones en el programa y su detección requiere menos recursos, ya que no necesitan identificación por correlación; control automático del umbral de intensidad mediante ajuste de intensidad en base al último patrón detectado (Fig. 2.4), compensación del viñeteado del lente de la cámara y un algoritmo de estimación robusta de pose (RPP, por sus siglas en inglés) para estimación de pose con menos inestabilidad de imagen [WS07].

2.4. BazAR

Esta biblioteca de software fue desarrollado por Julien Pilet, Andreas Geiger, Vincent Lepetit y Andre Mazzoni del Laboratorio de Visión Computacional de la École Polytechnique Fedéralé de Laussane, en Suiza.

El enfoque de BazAR es muy diferente al de las bibliotecas mencionadas anteriormente, ya que no utiliza marcas para la estimación de pose y rastreo de objetos. Para tal objetivo, utiliza una selección de puntos claves (Fig. 2.6), obtenidos a partir de un conjunto de imágenes de entrenamiento que son generadas a partir de parches de imágenes, centrados sobre puntos característicos de la imagen, a los que se les ha aplicado transformaciones afines al azar y ruido blanco para mejorar su robustez [LLF05, LF06]. Para reconocer qué puntos clave de la imagen de referencia aparecen en la imagen que está captando la cámara, se utilizan árboles aleatorios, los cuales contienen los niveles de intensidad de puntos clave obtenidos a partir de la etapa de entrenamiento [LLF05].

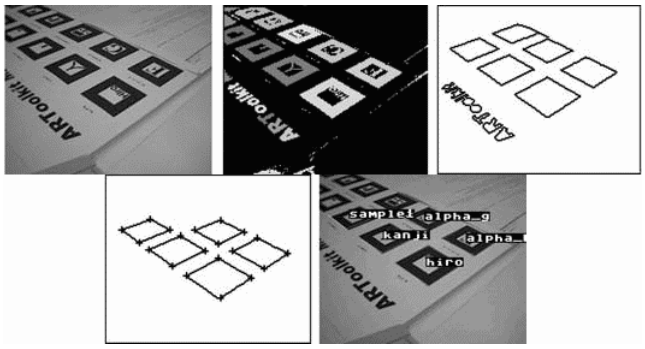


Figura 2.1: Funcionamiento de ARToolkit (imagen obtenida de [Fia05])De izquierda a derecha y de arriba a abajo: 1) Imagen original, 2) Componentes conexos dentro de la imagen umbralizada, 3) Contornos externos extraídos de los componentes conectados, 4) Cuadros identificados y sus esquinas, 5) Marcadores obtenidos etiquetados y colocados sobre al imagen. Imagen tomada de [Fia05].

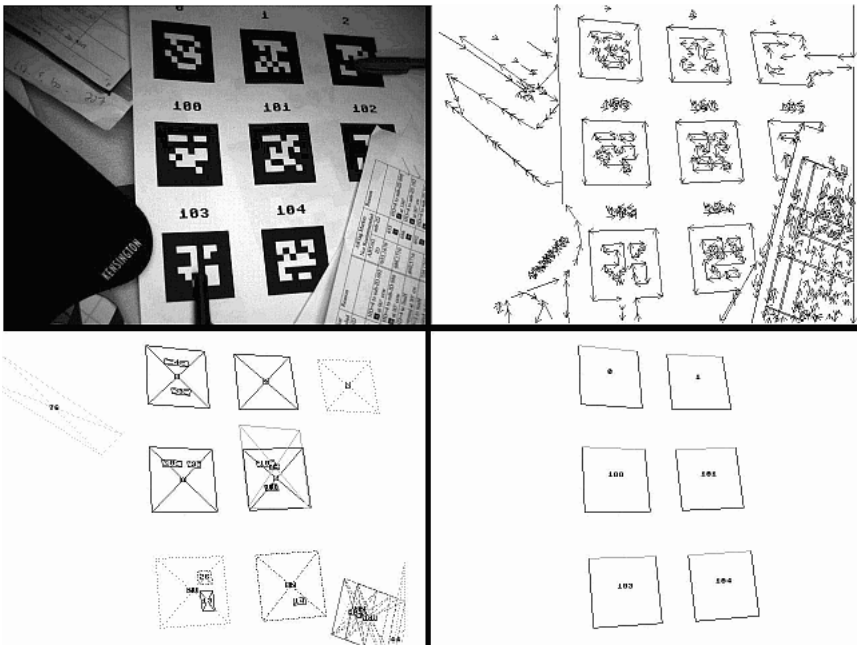


Figura 2.2: Funcionamiento de ARTag De izquierda a derecha y de arriba a abajo: 1) Imagen original, 2) Segmentos de líneas rectas encontrados en la imagen, 3) Segmentos de líneas agrupados en cuadriláteros, 4) Marcadores de ARTag localizados en el interior de cuadriláteros con códigos ARTag válidos. Imagen tomada de [Fia05].

Figura 2.3: ARTag: Detección de marcas en condiciones de iluminación variable. Imagen tomada de [Fia05].

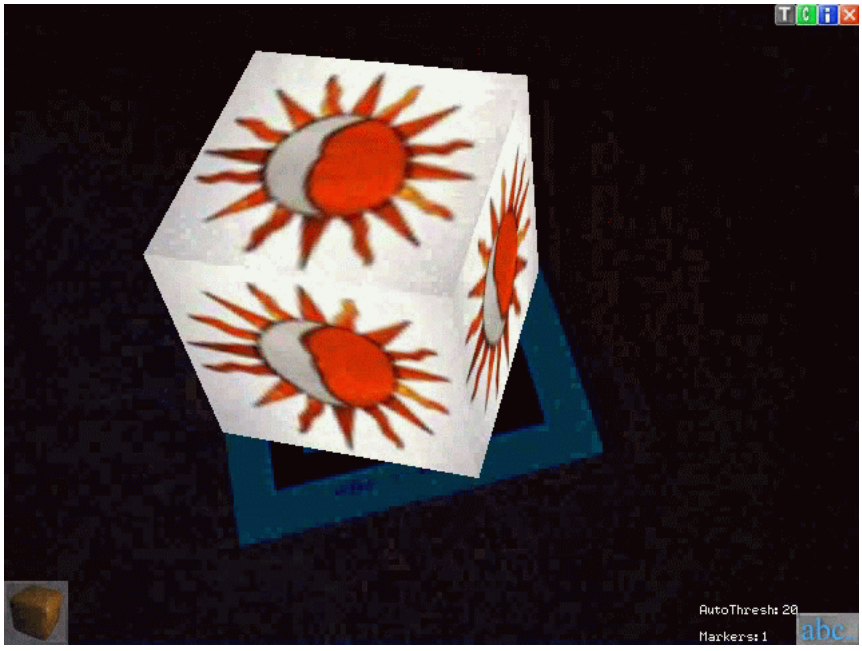


Figura 2.4: ARToolkit Plus: Ajuste automático de umbral para rastreo en entornos muy oscuros. Imagen tomada de [WS07].

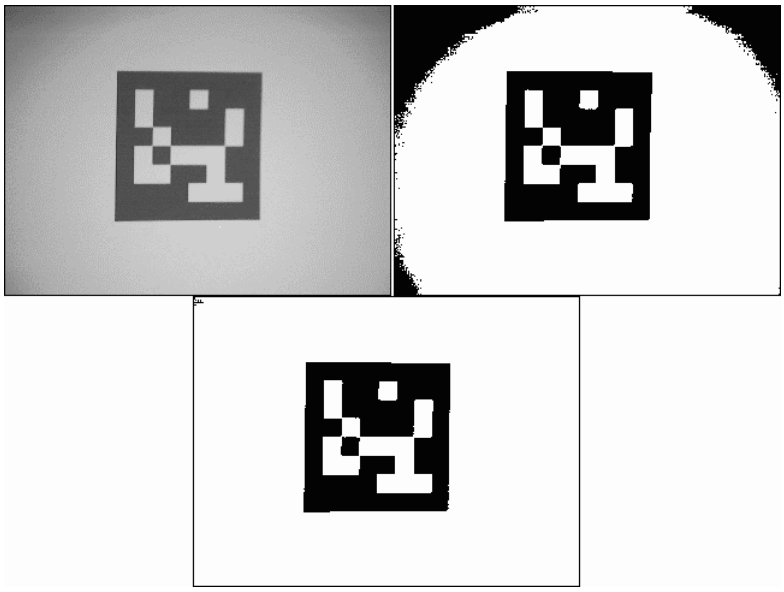


Figura 2.5: ARToolkit Plus: Compensación de viñeteado. Imagen tomada de [WS07].

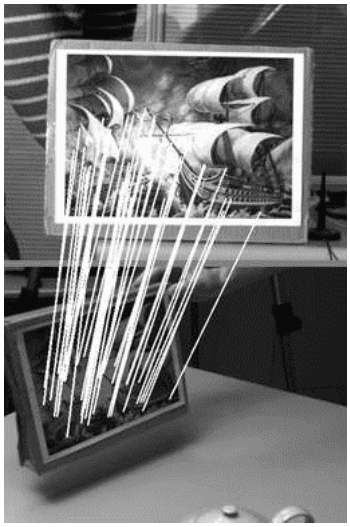


Figura 2.6: BazAR: Puntos clave en una imagen. Imagen tomada de [Pil].

Capítulo 3

Arquitectura del sistema

El sistema de AR propuesto en esta tesis utiliza un rastreador de características basado en SIFT [Low04], que es un algoritmo de detección y caracterización de características locales en una imagen y genera puntos clave que son invariantes a la translación y la rotación.

El sistema de AR estará integrado por los módulos mostrados en la figura 3.1 y que a continuación se describen

3.1. Módulo de vídeo

Este módulo se encargará de capturar los cuadros del mundo real a través de una cámara de vídeo. Cada cuadro obtenido será pasado a los demás módulos para su procesamiento.

Para tal objetivo se utilizará la biblioteca OpenCV (Open Source Computer Vision) [Int], que es una biblioteca de funciones de programación enfocadas principalmente a la visión computacional en tiempo real. Algunos ejemplos de aplicaciones de esta biblioteca son: Interacción Humano–Computadora, Identificación de Objetos, Segmentación y Reconocimiento, Reconocimiento de Rostros, Reconocimiento de Gestos, Rastreo de Movimiento, Robótica Móvil.

OpenCV proporciona una función para capturar cuadros de vídeo a partir de cámaras o archivos de vídeo Audio Video Interleave, cada cuadro es capturado como una imagen que es compatible con las funciones de la Interfaz de Programación de Aplicaciones de OpenCV. En el caso de este proyecto, cada cuadro será pasado al módulo de detección de características para hallar y caracterizar las regiones invariantes de la imagen.

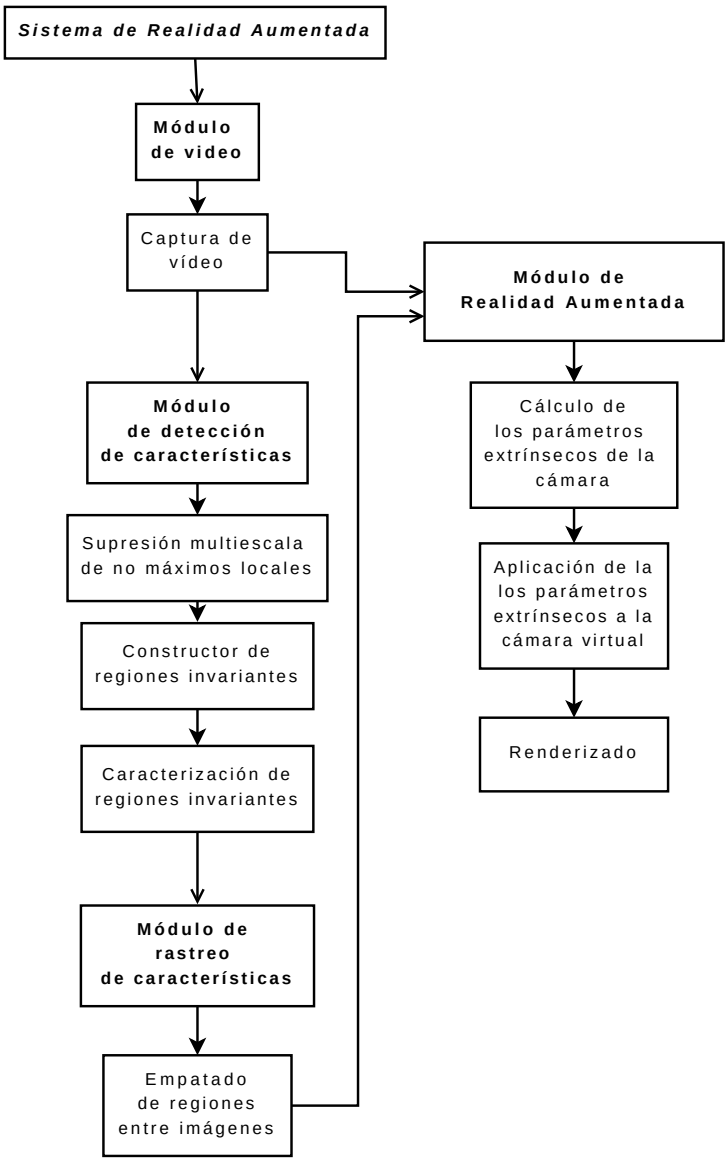


Figura 3.1: Diagrama de Arquitectura del sistema

3.2. Módulo de Detección de Características

La nube de puntos que servirá como zona de referencia para colocar los objetos virtuales sobre la escena del mundo real, se obtendrá por medio de este módulo, el cual procesará cada cuadro capturado por el módulo de vídeo para detectar las características que contenga. Para ello se usará un método para extraer características invariantes distintivas de las imágenes que puedan ser usadas para ejecutar una correspondencia confiable entre diferentes puntos de vista de un objeto o escena [Low04], al cual se le llama SIFT (Scale-Invariant Feature Transform) y el algoritmo fue publicado por David Lowe, de la Universidad de British Columbia, en 1999 [Low99].

3.2.1. SIFT

3.2.1.1. Panorama general

SIFT es un algoritmo de detección y descripción de características locales en imágenes, las cuales son invariantes a la escala y rotación y parcialmente invariantes a cambios en la iluminación y punto de vista tridimensional de la cámara. Las características están bien localizadas en los dominios espacial y de frecuencia de la imagen, con lo que se reduce la probabilidad de interrupciones por oclusiones, ruido o elementos amontonados en el escenario y son altamente distintivas, por lo que su probabilidad de ser correctamente correspondidas en una gran base de datos de características es alta [Low04].

El costo de extraer estas características se minimiza usando un enfoque de filtrado en cascada, en el cual las operaciones más caras se aplican a las locaciones que pasan una prueba inicial. Las etapas para generar el conjunto de características son las siguientes [Low04]:

1. Detección de máximos locales en el espacio de escala mediante diferencias de Gaussianas, para hallar los puntos de interés invariantes a escala y orientación.

2. Localización de puntos clave, en donde a cada ubicación candidata se les determina su ubicación y escala mediante el ajuste de un modelo detallado y son seleccionados en base a medidas de estabilidad.

3. Asignación de la orientación a cada punto clave, mediante las direcciones de los gradientes locales.

4. Descriptor del punto clave, los gradientes locales son medidos en la escala elegida dentro de la región alrededor del punto.

El punto clave se representa de tal forma que permite cambios en la distorsión de la forma y cambios en la iluminación. Una vez que los puntos clave se extraen de la imagen, se guardan en una base de datos para que pueden ser usadas en reconocimiento de imágenes.

La correspondencia entre imágenes se logra comparando los puntos clave de la nueva imagen con los puntos clave de las imágenes de la base de datos.

Los candidatos se hallan mediante la distancia Euclidiana entre los vectores de características de los puntos clave en la imagen de la escena y los vectores de características de los puntos clave de las imágenes de la base de datos. En el caso de las escenas con objetos amontonados, la identificación se hace mediante subconjuntos de los puntos clave.

El algoritmo no es completamente invariante a deformaciones afines, pero tiene un enfoque el cual permite que las posiciones relativas de la característica se desplacen significativamente con solamente unos pequeños cambios en el descriptor. Este enfoque no solamente permite que las características sean empatadas confiablemente a través de un intervalo considerable de deformación afín, sino que también hace que las caracteristicas sean más robustas a cambios en el punto de vista tridimensional en superficies no planas.

3.2.1.2. Detección de máximos locales en el espacio de escala

La detección de puntos clave en una imagen se hace mediante filtrado en cascada, en donde el primer paso consiste en identificar las ubicaciones y escalas que puedan ser asignadas repetidamente bajo diferentes vistas del mismo objeto. La detección de ubicaciones invariantes a la escala se logra buscando características dentro de todas las posibles escalas, aplicando una función continua de escala conocida como espacio de escala.

La representación $L(x, y, \sigma)$ de una imagen es el espacio de escala se obtiene mediante la convolución de una función Gaussiana de escala variable sobre una imagen.

$$L(x, y, \sigma) = G(x, y, \sigma) * L(x, y) \tag{3.1}$$

La detección eficiente de puntos se hace sobre imágenes de diferencias de Gaussianas (DoG, por sus iniciales en inglés), las cuales se obtienen restando dos imágenes convolucionadas con Guassiano con escalas cercanas (Fig. 3.2), separadas por un factor multiplicativo constante. La DoG se acerca bastante al Laplaciano de Gaussiano normalizado en escala.

El siguiente listado muestra una implementación eficiente de la DoG:

- Se genera una pila de imágenes convolucionando con Gaussianas separadas por un factor constante k en el espacio de escala, para ello se divide cada octava de espacio de escala (es decir, el doble de σ) en un número entero de intervalos, s, tal que $k = 2^{1/s}$,para cada octava se producen s+3 imágenes.

- Las imágenes de espacio de escala adyacentes se restan para obtener una DoG (Fig. 3.2).

- Al procesar una octava, se muestrean otra vez las imágenes Guassiannas cuyo valor sea el doble del original de σ, que son las dos imágenes a partir del tope de la pila de imágenes Gaussianas, el muestreo se hace tomando

cada segundo pixel de cada fila y cada columna, en otras palabras, se obtiene una imagen nueva cuyo tamaño es la mitad de la imagen original.

3.2.1.3. Detección de máximos locales

Los máximos locales se detectan usando tres imágenes Gaussianas, usando la segunda imagen como zona de inicio; por cada pixel de la segunda imagen se hace una comparación con sus ocho vecinos en la misma escala y contra los nueve vecinos en las resoluciones inmediata superior e inmediata inferior, dando un total de veintiseís vecinos (Fig. 3.3); los puntos a seleccionar son aquellos que son los más grandes o los más chicos entre sus vecinos [Low04]. Los máximos locales que quedan muy cerca entre sí son inestables a pequeñas perturbaciones.

3.2.1.4. Frecuencia del muestreo en escala

La repetibilidad no sigue mejorando conforme se muestrean más escalas, ya que los máximos locales se vuelven menos estables y no son tan fáciles de detectar cuando la imagen es transformada, ya sea de forma geométrica (transformaciones afines, deformaciones) o de forma fotométrica (cambios de brillo y contraste, adición de ruido). El éxito del reconocimiento depende de la cantidad de puntos clave que sean correspondencias correctamente. Se podrían usar más muestras de escala, pero incrementa el costo del cálculo, así que Lowe eligió tres muestras por escala para su algoritmo SIFT [Low04].

3.2.1.5. Frecuencia de muestreo en el dominio espacial

Dado que los máximos pueden estar muy cerca entre sí, debe calcularse el punto medio entre la frecuencia de muestreo y la tasa de detección. La repetibilidad se incrementa con σ, pero en términos de eficiencia hay costos al usar una σ grande, la elección de Lowe fue de 1.6. Si se suaviza la imagen de la primera octava antes de procesarla, se eliminan las altas frecuencias espaciales. Para aprovechar la imagen de entrada, se duplica su tamaño por interpolación bilineal para tener más muestras, lo cual conduce a una implementación más eficiente. Este cambio de tamaño incrementa la cantidad de puntos clave estables por un factor cercano a 4 [Low04].

3.2.1.6. Localización precisa de puntos clave

Por cada punto clave candidato, ahora se ejecuta un ajuste detallado a los datos cercanos para su localización, escala y radio de las curvaturas principales, con esto se rechazan los puntos con bajo contraste o los que están pobremente localizados en un borde. El mejor método para hacerlo es con una función cuadrática tridimensional de ajuste aplicado a los puntos locales de la muestra, para determinar la ubicación interpolada del máximo. Este enfoque, desarrollado por Brown y Lowe en el 2002, usa la expansión de Taylor de la función del espacio de escala, $D(x, y, \sigma)$, desplazada para que el origen esté en el sitio de la muestra:

$$D(\hat{\mathbf{x}}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x} \tag{3.2}$$

Donde D y sus derivadas son evaluadas en el punto de muestra y en $\mathbf{x} = (x, y, \sigma)^T$ es el desplazamiento desde ese punto. La ubicación del extremo, $\hat{\mathbf{x}}$, se determina con la derivada de esa función con respecto a $\mathbf{x}$ y poniéndola a cero. Como sugiere Brown, el Hessiano y la derivada de la DoG son aproximadas mediante diferencias de muestras vecinas, se obtiene un sistema lineal de 3×3. Si el desplazamiento de $\hat{\mathbf{x}}$ es mayor que a 0.5 en cualquier dimensión, quiere decir que ese máximo está cerca de otro punto. El desplazamiento final $\hat{\mathbf{x}}$ se agrega a la ubicación del punto de muestra para obtener el estimado interpolado para la ubicación del extremo, obteniéndose la siguiente función

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1}\frac{\partial D}{\partial \mathbf{x}} \tag{3.3}$$

La función de valor en el extremo es útil para rechazar máximos locales inestables con bajo contraste [Low04]. Se puede obtener sustituyendo la ecuación (3.3) en (3.2), dando como resultado

$$D(\hat{\mathbf{x}}) = D + \frac{1}{2}\frac{\partial D^T}{\partial \mathbf{x}}\hat{\mathbf{x}} \tag{3.4}$$

En los experimentos en [Low04], todos los extremos con un valor de $|D(\hat{\mathbf{x}})|$ menor a 0.03 fueron descartados, considerando que los valores de los pixeles estén en el intervalo de $[0, 1]$.

3.2.1.7. Eliminando las respuestas de los bordes

La DoG tiene una respuesta fuerte en los bordes aún si la ubicación de los puntos claves en el borde está pobremente determinada, por lo tanto es inestable a pequeñas cantidades de ruido. Los picos pobremente definidos en la función de DoG tienen una gran curvatura principal a lo largo del borde pero es pequeña en la dirección perpendicular. La curvatura se obtiene con una matriz Hessiana, H, calculada con la ubicación y escala del punto clave, las derivadas se estiman por diferencias entre muestras cercanas [Low04].

Los eigenvalores de la matriz Hessiana son proporcionales a las curvaturas principales de la DoG, hay dos eigenvalores extremos: α, que es la magnitud más grande, y β, que es la magnitud más pequeña; a partir de ellos se obtienen el determinante (Ec. 3.6) y la traza de la matriz Hessiana $\mathbf{H}$(Ec. 3.5) [Low04].

$$Tr(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta \tag{3.5}$$

$$Det(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta \tag{3.6}$$

Cuando aparece un determinante negativo, se rechaza el punto clave porque las curvaturas tienen diferente signo y no se trata de un máximo [Low04].

Utilizando los eigenvalores, es posible eliminar los puntos clave que estén más allá de cierto radio de curvatura, $r = |\alpha - \beta|$, verificando si el cociente del rastro entre la determinante es menor al mínimo donde los eigenvalores son iguales (Ec. 3.7) [Low04].

$$\frac{Tr\,(H)^2}{Det\,(H)} < \frac{(r+1)^2}{r} \tag{3.7}$$

3.2.1.8. Asignación de la orientación

La escala del punto clave sirve para seleccionar la imagen Gaussiana más cercana, L, por cada muestra de esa imagen, $L\,(x,y)$, se obtienen la magnitud $m\,(x,y)$ y la orientación $\theta\,(x,y)$ mediante diferencias de pixeles.

$$m\,(x,y) = \sqrt{(L\,(x+1,y) - L\,(x-1,y))^2 + (L\,(x,y+1) - L\,(x,y-1))^2}$$

$$\theta\,(x,y) = \tan^{-1}\left((L\,(x+1,y) - L\,(x-1,y)) \,/\, (L\,(x,y+1) - L\,(x,y-1))\right)$$

Se genera un histograma de orientaciones de los puntos en una región alrededor del punto clave. El histograma tiene 36 clases, de 10° cada una, las cuales cubren 360° de orientaciones. Cada muestra que se agrega al histograma se pondera por magnitud, $m\,(x,y)$, y una función de ventana circular con peso Gaussiano con una $\sigma = 1{,}5x$ la escala del punto clave. Los picos en la orientación pertenecen a direcciones dominantes de las gradientes locales, se detecta el pico más alto en el histograma, así como los puntos que estén en el 80 % de la altura del pico más alto. Si hay múltiples picos de la misma magnitud, se crean múltiples puntos clave en el mismo lugar pero con diferentes orientaciones, ya que éstos ayudan mucho a la estabilidad del correspondencia de puntos clave. Al final, se ajusta una parábola en los 3 puntos más cercanos a cada pico.

3.2.1.9. El descriptor de imagen

Una vez asignada la ubicación, la escala y la orientación de cada punto clave, los cuales imponen un sistema coordenado local repetible en dos dimensiones, ahora hay que calcular un descriptor que sea altamente distintivo e invariante a los cambios en la iluminación y el punto de vista tridimensional.

El enfoque más simple sería muestreando las intensidades alrededor del punto clave en la escala adecuada y corresponderlas entre sí mediante correlación normalizada. El problema con la correlación simple de parches de imagen es su alta sensibilidad a los cambios que alteran la forma de la imagen. Edelman, Intrator y Poggio [SE97] sugirieron una representación basada en la visión biológica, usando como base las neuronas de la corteza visual primaria, las cuales responden a un gradiente en una orientación y frecuencia espacial particular, pero su ubicación en la retina permite que sea desplazada sobre un pequeño campo receptivo. Edelman et al. hipotetizan que la función de esas neuronas es al de corresponder y reconocer objetos tridimensionales en un rango de puntos de vista.

3.2.1.10. Representación de los descriptores

Para generar el descriptor de una región invariante, primero hay que muestrear las magnitudes y orientaciones del gradiente alrededor del punto clave usando la escala del mismo para elegir el radio de desenfoque Gaussiano. A continuación, las coordenadas del descriptor y la orientación del gradiente se rotan relativos al centro del punto clave, después, se usa una función de ponderación Gaussiana con σ igual a la mitad del ancho del descriptor para dar peso a cada punto, el propósito de esta función es evitar cambios abruptos en el descriptor cuando hay pequeños cambios en la posición de la función de ventana y dar una menor énfasis a los gradientes alejados del centro [Low04].

El descriptor se forma con los histogramas de orientación en regiones de 4×4, contiene 8 direcciones por cada histograma de orientación, donde cada flecha indica la magnitud de esa entrada del histograma. Para evitar los efectos en los que el descriptor cambia abruptamente cuando una muestra cambia suavemente entre histogramas u orientaciones, se aplica interpolación trilineal para distribuir cada muestra del gradiente en espacios adyacentes del histograma, para tal efecto, se multiplica el valor de cada espacio del histograma por $1 - d$ en cada dimensión, donde d es la distancia entre la muestra y el valor del espacio central del histograma [Low04]. En la representación del descriptor se utiliza interpolación trilineal porque para generar el descriptor se trabaja con tres o más imágenes de espacio de escala.

El componente principal del descriptor es un vector que contiene los valores de todas las orientaciones del histograma, los mejores resultados se obtienen con una matriz de 4×4 con un histograma de orientación de 8 espacios en cada elemento de la matriz, lo que da un vector de 128 dimensiones [Low04].

3.2.1.11. Invarianza a la iluminación

Al final del cálculo del descriptor, su vector de características es normalizado, para reducir los efectos de los cambios de iluminación. Los cambios de contraste obtenidos por la multiplicación de una constante, también afectan al gradiente, este cambio es cancelado por la normalización del vector. Los cambios de iluminación por adición no afectan al gradiente, porque éste se calcula por diferencias entre muestras de la imagen. Los cambios no lineales de iluminación afectan grandemente a la magnitud relativa del gradiente, pero tienen poco impacto en la orientación; el cambio se compensa umbralizando los valores del vector unitario de caracteristicas y volviendo a normalizar al vector unitario.

3.3. Módulo de Rastreo de Características

En este módulo se rastreará el conjunto de imágenes capturadas por la cámara en tiempo real, a partir de las características detectadas por el módulo de detección de características, para ello se harán pruebas de continuidad geométrica para verificar que una región detectada en un cuadro pertenezca a una región detectada en cuadros anteriores, así como una correspondencia

de regiones entre imágenes para ver cuáles regiones concuerdan entre varias imágenes.

3.3.1. Correspondencia de puntos clave

El mejor punto clave candidato se encuentra identificando al vecino más cercano en una base de datos de puntos clave de imágenes de entrenamiento, mediante la mínima distancia Euclidiana del vector característico.

Una forma eficiente de localizar vecinos en espacios multidimensionales es con una variación del árbol de busqueda k-d (k-d tree, en inglés) llamada BBF (Best Bin First), en el cual los compartimientos del espacio de características se buscan en orden de la distancia más cercana a la ubicación de búsqueda [Low04].

3.3.1.1. Árbol k-d

Un árbol k-d (k-d tree, en inglés) es un árbol de búsqueda multidimensional que sirve como estructura de datos para almacenar información que puede ser recuperada mediante búsquedas asociativas. Una ventaja de esta estructura de datos es que puede manejar diferentes tipos de consultas muy eficientemente. Los algoritmos de búsqueda están datos para búsquedas de correspondencia parcial especificando t claves y para búsquedas del vecino cercano [Ben75].

Dentro de un árbol k-d, los datos se almacenan como nodos, cada uno de éstos tiene dos apuntadores, los cuales pueden ser nulo o apuntar a otro nodo en el árbol. Definamos una notación para los elementos de un árbol k-d: el nodo raíz se denomina $ROOT$, el conjunto de k claves de un nodo P se denominarán $K_0(P) \ldots K_{k-1}(P)$, los apuntadores del nodo serán $LOSON(P)$ y $HISON(P)$y el discriminador será $DISC(P)$. El orden que se impone para el árbol será el siguiente: para cualquier nodo P en un árbol k-d, sea el discriminador $DISC(P)$, entonces para cualquier nodo Q en $LOSON(P)$, será verdadero que $K_j(Q) < K_j(P)$; de la misma manera, para cualquier nodo R en $LOSON(P)$, será verdadero que $K_j(R) > K_j(P)$. Todos los nodos en el mismo nivel tienen el mismo discriminador. El nodo raíz tiene el discriminador 0, sus nodos hijo tienen el discriminador 1 y así hasta el *k-ésimo* nivel, donde el discriminador vale $k-1$ [Ben75].

Cuando buscamos un nodo Q en el árbol, un problema que surge es saber cuál es el siguiente nodo por el que se va a descender para continuar la búsqueda, para solucionar eso, vamos a definir una función $SUCCESOR(P,Q)$ que regresa $LOSON$ o $HISON(P)$. Si $K_j(Q) \neq K_j(P)$, se regresa el nodo hijo adecuado en base a las propiedades descritas anteriormente; en el caso de las claves iguales, la decisión se toma en base a las claves restantes [Ben75].

El algoritmo para insertar elementos en un árbol k-d también sirve para encontrar elementos en el mismo. A este algoritmo se le pasa un nodo P, el cual no está en el árbol. En caso de haber un nodo con claves iguales, se regresa la dirección en memoria de ese nodo; de otra manera, se inserta el nodo y se devuelve el elemento nulo, Λ (Alg. 1) [Ben75].

Algoritmo 1 Inserción y búsqueda en un árbol k-d

I1.[Revisar si hay un árbol nulo.]
Si $ROOT = \Lambda$,
entonces asignar $ROOT \leftarrow P$,$LOSON\,(P) \leftarrow \Lambda$,$HISON\,(P) \leftarrow \Lambda$, $DISC\,(P) \leftarrow 0$ y regresar Λ.
De otra forma,
asignar $Q \leftarrow ROOT$ (Q recorrerá en el árbol en descenso).
I2.[Comparar.]
Si $K_i\,(P) = K_i\,(Q)$para $0 \leq i \leq k-1$ (es decir, todas las claves son iguales),
entonces regresar Q.
De otra forma,
asignar $SON \leftarrow SUCCESOR\,(Q, P)$ (SON será $LOSON$ o $HISON$)
Si $SON\,(Q) = \Lambda$ entonces ir al paso I4.
I3.[Descender en el árbol]
Asignar $Q \leftarrow SON\,(Q)$e ir al paso I2.
I4.[Insertar nuevo nodo en el árbol.]
Asignar $SON\,(Q) \leftarrow P$, $LOSON\,(P) \leftarrow \Lambda$,$HISON\,(P) \leftarrow \Lambda$, $DISC\,(P) \leftarrow NEXTDISC\,(DISC\,(Q))$ y regresar Λ.

Las consultas que se pueden hacer sobre un árbol k-d son las siguientes [Ben75]:

- Consulta de intersección, los registros a recuperar son aquellos que intersectan un subconjunto de registros válidos.
- Consulta de correspondencia exacta, aquí se busca un registro en específico en la estructura de datos. Un árbol k-d no debería usarse para almacenar registros si solamente se usa este tipo de consulta.
- Consulta de correspondencia parcial, el criterio para recuperar registro está dado por los valores que son especificados a un subconjunto de claves.
- Consultas de región, el tipo de consulta de intersección más común es en el que cualquier región puede ser especificada como el conjunto que debe intersectar los registros que quieren ser recuperados.
- Consulta de vecino más cercano, en base a una función de distancia D, una colección de puntos B (en el espacio k-dimensional) y un punto P (en ese espacio), se desea encontrar el vecino más cercano de P dentro del conjunto B. El vecino más cercano es una Q tal que:

$$(\forall R \leq B)\,\{(R \neq Q) \Longrightarrow [D\,(R, P) \geq D\,(Q, P)]\}$$

Para propósitos de esta tesis, los puntos clave de una imagen que desean corresponderse para el rastreo, se buscan mediante el algoritmo de búsqueda "Best Bin First", el cual se describe a continuación.

3.3.1.2. Algoritmo Best Bin First

El algoritmo "Best Bin First" es un algoritmo de búsqueda aproximada para árboles k-d que encuentra el vecino más cercano para una gran fracción de las búsquedas, y un vecino muy cercano en los casos restantes. Una característica particular de un árbol k-d es que los compartimientos son más pequeños en las regiones con alta densidad de puntos y son más grandes en las áreas con baja densidad de puntos; debido a esto, hay alta probabilidad de que una consulta termine en el compartimiento buscado o en uno adyacente [BL97].

La búsqueda del vecino más cercano es mediante un algoritmo de vuelta atrás con ramificación. Primero se recorre el árbol para hallar el compartimiento con el punto a consultar. Durante la vuelta hacia atrás, se podan las ramas que estén lejos de D_{curr}, la distancia del punto que se consulta al vecino más cercano en ese momento; la busqueda termina cuando se completa la poda del árbol k-d. Con este procedimiento, la búsqueda en espacios multidimensionales provoca una baja en el rendimiento del programa, porque mucho tiempo de la busqueda se invierte examinando compartimientos en los que una fracción de su volumen podria contener al vecino más cercano. Si hay disposición a usar una búsqueda aproximada del vecino más cercano, el tiempo de busqueda puede reducirse al limitar la cantidad de nodos hoja a revisar; además, la búsqueda por vuelta hacia atrás en un árbol k-d es ineficiente porque los puntos se revisan en base a su posición en el árbol y no consideran la posición del punto a consultar [BL97].

Una idea simple para buscar el vecino más cercano es mediante la búsqueda del compartimiento a distancia creciente del punto a consultar, esto puede implementarse fácilmente mediante una cola de prioridad, al tomar la decisión en una ramificación, las opciones no tomadas se agregan a la cola para futura referencia, el contenido de la cola incluye la posición actual en el árbol, y la distancia del punto a consultar al nodo. Una vez revisado un nodo hoja, el tope de la cola se remueve y se usa para seguir la busqueda en la zona con el siguiente compartimiento más cercano [BL97].

3.3.1.3. RANSAC

El algoritmo RANSAC, iniciales en inglés de Random Sample Consensus, se usa en análisis de escenas porque los detectores de características locales suelen cometer errores y se requieren de características correctas para interpretar el contenido de la escena. Hay dos tipos de errores cuando se detectan características de una imagen –errores de clasificación y errores de medición. Los errores de clasificación ocurren cuando el detector identifica erróneamente una porción de la imagen como una instancia de una característica. Los errores de medición ocurren cuando una característica es detectada correctamente pero se calculan erróneamente sus parámetros. Los errores de medición siguen una

distribución normal y por lo tanto se puede suponer que la desviación máxima de cualquier dato es una función directa del tamaño del conjunto de datos y que siempre habrá suficientes datos buenos para corregir cualquier desviación grande. Los errores de clasificación son errores brutos, los cuales tienen un efecto mayor en el conjunto de datos y no pueden ser promediados.

RANSAC funciona de la siguiente forma: en lugar de usar todo el conjunto de datos para obtener la solución inicial y luego empezar a eliminar los datos inválidos, se empieza con un conjunto pequeño y se aumenta el tamaño del conjunto con datos consistentes cuando es posible.

Dado que el modelo necesita de un mínimo de n puntos de datos para instanciar sus parámetros libres y un conjunto de puntos de datos P cuyo número sea mayor que n, se seleccionan aleatoriamente un subconjunto S1 de n puntos de P para instanciar el modelo. Se usa el modelo instanciado M′ para determinar un subconjunto S1* de puntos que están dentro de cierta tolerancia de error, a este subconjunto se le llama conjunto consenso de S1.

Si la cantidad de elementos de S1* es mayor a un umbral t, la función de estimado de errores brutos en P, se usa S1*para calcular el modelo M1.

Si la cantidad de elementos de S1* es menor al umbral t, aleatoriamente se elige un nuevo conjunto S2 y se repite el proceso. Si tras un número predeterminado de intentos, no se llega a un conjunto de consenso con t o más elementos, o se resuelve el modelo con el conjunto más grande encontrado o se termina en falla.

3.3.1.4. GTM

Un problema con RANSAC es que se desempeña pobremente cuando el número de correspondencias correctas es menor al 50 %, a lo cual David Lowe sugiere usar la transformada de Hough para crear cúmulos de características en espacio de pose [Low04].

Para este trabajo se probó un algoritmo de empatado de correspondencias basado en grafos, llamado *Graph Transform Matching* (GTM) –Empatado por Transformación de Grafo–, el cual fue desarrollado por Wendy Elizabeth Aguilar Martínez, de la Universidad Nacional Autónoma de México [AFE+08]; este algoritmo empata las imágenes mediante un grafo formado a partir de las características de las mismas.

Asumiendo que tenemos dos imágenes, a las cuales se les extrajeron sus puntos clave, el objetivo es encontrar las correspondencias entre los dos conjuntos de puntos. Se empieza construyendo un conjunto inicial de correspondencias entre los puntos clave de ambas imágenes. Cada punto es empatado al punto correlacionado más fuerte en la otra imagen. Debido a que el empatado inicial no es muy preciso, muchas correspondencias tienden a ser incorrectas [AFE+08].

El algoritmo [AFE+08] está diseñado para remover las correspondencias erróneas, mediante la aplicación de relaciones espaciales coherentes entre las dos imágenes, las cuales remueven las correspondencias que afectan las relaciones de las vecindades. Para lograrlo, se construye un grafo de la mediana de los

K-vecinos más cercanos (K-nearest-neighbour o K-NN) $G_p = (V_p, E_p)$ de la siguiente forma: se define un vértice v_i que para cada uno de los N puntos p_i, de tal forma que $V_p = v_1 \dots v_N$. Una arista no dirigida (i, j) existe cuando p_j y también $\|p_i - p_j\| \leq \eta$, donde η es la mediana de las distancias entre pares de vértices, la cual se define como:

$$\eta = \underset{(l,m)\in V_p \times V_p}{median} \|p_l - p_m\| \tag{3.8}$$

La primera condición 3.8 especifica que un vértice sólo puede validar la estructura de sus vecinos más cercanos. La segunda condición restringe la proximidad de la validación, si no hay K vértices que apoyan la estructura de p_i, entonces este vértice está completamente desconectado hasta el final de la construcción del grafo K-NN. El grafo G_p tiene una matriz de adyacencia A_p de $N \times N$, donde $A_p(i, j) = 1$ cuando $(i, j) \in E_p$ y $A_p(i, j) = 0$ en caso contrario [AFE+08].

El algoritmo GTM se apoya en la hipótesis de que la transformación que ocurre entre dos imágenes es razonablemente suave, asi que los puntos de la primera imagen corresponden a puntos en la segunda imagen. Si todas las correspondencias son correctas, los grafos $G(P)$y $G(P')$ serán isomórficos. Las correspondencias erróneas resultan en diferencias estructurales entre los grafos, los cuales pueden ser eliminados –con alta probabilidad– aplicando iterativamente un criterio estructural simple. El algoritmo GTM itera de la siguiente forma: (i) se selecciona una correspondencia errónea en M; (ii) se remueven los vértices que pertenecen a la correspondencia errónea, así como las referencias a esos vértice, y (iii) se recalculan los dos grafos K-NN. La disparidad estructural se aproxima por una matriz de adyacencia residual $R = |A_P - A'_P|$ y seleccionando la columna j^{out} que dé el mayor número de aristas diferentes en ambos grafos. La correspondencia errónea seleccionada es el par $\left(V_{j^{out}}, V'_{j^{out}}\right)$, entonces se remueven todas las referencias a esos vértices y los grafos K-NN se calculan a partir de los vértices restantes. El algoritmo termina cuando se alcanza la matriz residual nula [AFE+08].

3.4. Módulo de Realidad Aumentada

En este módulo se utilizarán las regiones de características detectadas en todas las imágenes para calcular la matriz fundamental, la cual sera utilizada por una Interfaz de Programación de Aplicaciones de gráficos tridimensionales para orientar y posicionar uno o varios objetos virtuales en relación al punto de vista del mundo real que está capturando la cámara.

3.4.1. Estructura del Módulo de Realidad Aumentada

En literatura reciente, ha surgido el siguiente esquema como el metodo de eleccion preferido para realizar una correspondencia efectiva [Mar06]:

1. Detección de características en la imagen, por ejemplo [Low04].
2. Aplicación de una correlación entre las caraterísticas seguida de un proceso de umbral para obtener un primer conjunto de correspondencias candidatas.
3. Uso de un metodo robusto, tal como RANSAC o LMedS (Least Median of Squares), para estimar la geometría epipolar o trifocal del sistema de la cámara.
4. Rechazo de las parejas candidatas que son incompatibles con la geometría estimada de la cámara.
5. Realización de un proceso de correspondencia guiado que hace uso de la geometría estimada para encontrar más correspondencias.

El módulo de Realidad Aumentada para el sistema que se desarrolla en esta tesis funciona de la siguiente manera:

1. El módulo de vídeo captura un cuadro con la cámara, con lo cual se obtiene una imagen (Fig. 3.4-1).
2. Se usa el algoritmo SIFT sobre la imagen y el patron de referencia para extraer sus puntos clave (Fig. 3.4-2).
3. Se buscan correspondencias entre los puntos clave de la imagen y del patrón de referencia (Fig. 3.4-3).
4. A partir de las correspondencias obtenidas en el paso anterior, se obtienen los vectores de rotación y traslación de la cámara (Fig. 3.4-4).
5. Se dibuja la escena de Realidad Aumentada en el siguiente orden (Fig. 3.4-5):
   a) Primero, se coloca la imagen de la cámara sobre un polígono.
   b) Después, se aplican los vectores de traslación y rotación para modificar la posición del objeto virtual.
   c) Al final, se dibujan los objetos virtuales que son visibles en escena.
6. Se repite el ciclo mientras la cámara siga capturando cuadros.

3.4.2. Estimación de la posición de la cámara

Es posible estimar la posición de una cámara mediante el análisis de dos imágenes, el resultado de este análisis es una matriz que describe a la cámara, la cual se usa en el sistema gráfico para orientar la cámara virtual y modificar la perspectiva de los objetos virtuales que sera colocados sobre la escena real.

El sistema de Realidad Aumentada que se propone en esta tesis tendrá un funcionamiento similar al de las bibliotecas de software descritas en el capítulo

2, es decir, los objetos virtuales se van a colocar sobre las coordenadas del objeto real que corresponde con el sistema de coordenadas virtuales y la orientación de los mismos estará dada por la orientación de la imagen.

3.4.2.1. Parámetros extrínsecos de la cámara

El marco de referencia del cuadro de la cámara se introduce para el propósito de escribir de una forma más simple las ecuaciones fundamentales de la proyección en perspectiva. Sin embargo, este marco de referencia es desconocido, y un problema común es determinar la ubicación y orientación del cuadro de la cámara con respecto a un marco de referencia conocido, usando solamente información de la imagen [TV98].

Los parámetros extrínsecos se definen como un conjunto de parámetros geométricos que identifican de forma única la transformación entre un marco de referencia de cámara desconocido y un marco de referencia conocido, llamado *marco de referencia del mundo* [TV98].

Una elección tipica para describir la transformación entre los cuadros de la cámara y el mundo real es usando [TV98]:

- un vector de traslación, T, que describe las posiciones de los origenes de los cuadros de referencia, y
- una matriz de rotación de 3×3, R, que es una matriz ortogonal $R^{\mathrm{T}}R = RR^{\mathrm{T}} = I$, el cual traslada los ejes correspondientes de cada cuadro hacia el otro cuadro para hacer coincidir los dos sistemas de coordenadas [TV98].

Las relaciones de ortogonalidad reducen los grados de libertad de R a tres [TV98].

La relación entre las coordenadas de un punto P en los cuadros del mundo y la cámara ubicada en el mundo real (Fig. 3.5), P_W y P_c respectivamente, son [TV98]:

$$\mathrm{P_c} = \mathrm{R}\,(\mathrm{P_w} - \mathrm{T})$$

con

$$\mathrm{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

3.4.2.2. Transformación de marcos de referencia

Para que una imagen de referencia sirva como zona para colocar un objeto virtual, es necesario saber su ubicación espacial en la escena. Sobre la imagen de referencia se van a mapear sus puntos clave o características en términos de las coordenadas del mundo, para ubicar los objetos en relación a la cámara.

Los puntos clave de un objeto son mapeados en el plano de la cámara usando la siguiente ecuación

$$\overrightarrow{V}_c = M_{int} M_{ext} \overrightarrow{V}_W \tag{3.9}$$

donde $\overrightarrow{V}_c = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ es un vector de 3×1 que representa las coordenadas de un punto clave en el espacio de coordenadas del plano de la cámara, $M_{int} = \begin{bmatrix} -f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$ es una matriz de 3×3 con los parámetros intrínsecos de la cámara – distancia focal y centro de la cámara –, $M_{ext} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{23} & r_{33} & t_z \end{bmatrix}$ es una matriz de 3×4 con los parámetros extrínsecos de la cámara – rotación y traslación en los ejes x, y y z – y $\overrightarrow{V}_W = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$ es un vector de 3×1 que representa las coordenadas de un punto clave en el espacio de coordenadas del mundo, o sea, la escena que se está capturando.

A fin de obtener la ubicación de la imagen de referencia con respecto a la cámara, primero hay que saber la ubicación de los puntos clave en el sistema de coordenadas del mundo. Esta información se obtiene despejando $\overrightarrow{V}_W$ de la ecuación 3.9, la cual que es un sistema lineal de la forma $Ax = b$, donde $b = \overrightarrow{V}_c$, $A = M_{int} M_{ext}$ y x=$\overrightarrow{V}_W$, quedando el sistema lineal de esta forma

$$\overrightarrow{V}_W = \left(M_{ext}^{\mathrm{T}} M_{int}^{\mathrm{T}} M_{int} M_{ext}\right)^{+} M_{ext}^{\mathrm{T}} M_{int}^{\mathrm{T}} \overrightarrow{V}_c \tag{3.10}$$

Donde los valores de la matriz intrínseca son inicializados mediante funciones de calibración de la cámara de OpenCV [Int] y la matriz extrínseca es inicializada con los valores $M_{ext} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & z \end{bmatrix}$, que representan a una cámara que está alejada a una distancia z, paralela al eje Z, y que no ha sido rotada en ningún eje (Fig. 3.6). Una vez obtenidas las coordenadas de los puntos clave del patrón, se guarda esta información junto con los parámetros intrínsecos de la cámara para estimar nuevos valores de los parámetros extrínsecos.

Estos valores se obtienen usando las coordenadas $\overrightarrow{V}_W$, los parámetros intrínsecos de la cámara y las coordenadas $\overrightarrow{V}_c$ de los puntos clave obtenidos en cada cuadro capturado por la cámara. A fin de saber cuáles puntos de los conjuntos $\overrightarrow{V}_W$ y $\overrightarrow{V}_c$ sirven para obtener los parámetros extrínsecos de la cámara, es necesario obtener las correspondencias entre ambos conjuntos de puntos, aquí es donde los algoritmos RANSAC o GTM entran en acción para eliminar entradas que sean un falso positivo.

Una vez obtenidas las correspondencias de ambos conjuntos de puntos, se pasan a una función para obtener los parámetros extrínsecos de la cámara,

la cual usa como valores de entrada el conjunto de puntos en espacio de coordenadas del mundo $\overrightarrow{V_W}$, el conjunto de puntos en espacio de coordenadas de la imagen $\overrightarrow{V}_W$, los parámetros intrínsecos de la cámara M_{int} y los valores de distorsión de la óptica de la cámara y el resultado es un vector de traslación $t = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$ y un vector de rotación $r = \begin{bmatrix} \theta \\ \omega \\ \phi \end{bmatrix}$, los cuales se pasan al sistema de gráficos tridimensionales para ubicar los objetos virtuales con relación a la cámara.

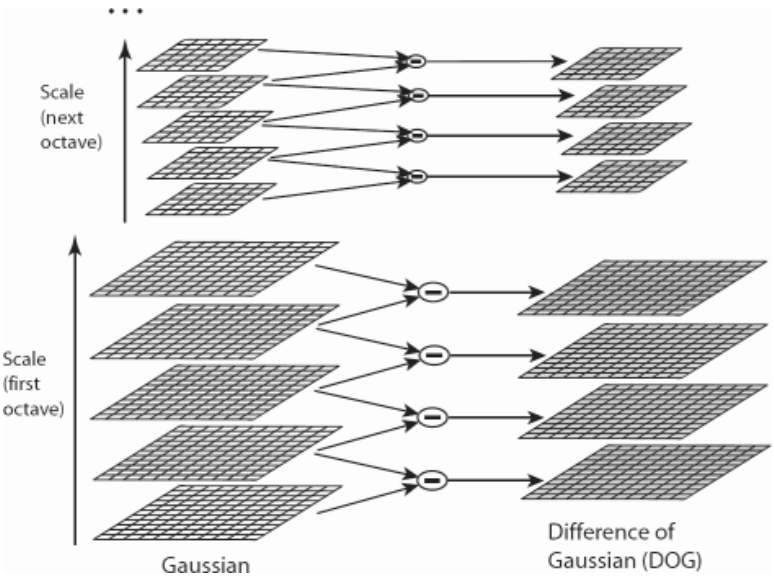


Figura 3.2: Diferencias de Gaussianas en espacio de escala., imagen obtenida de [Low04]

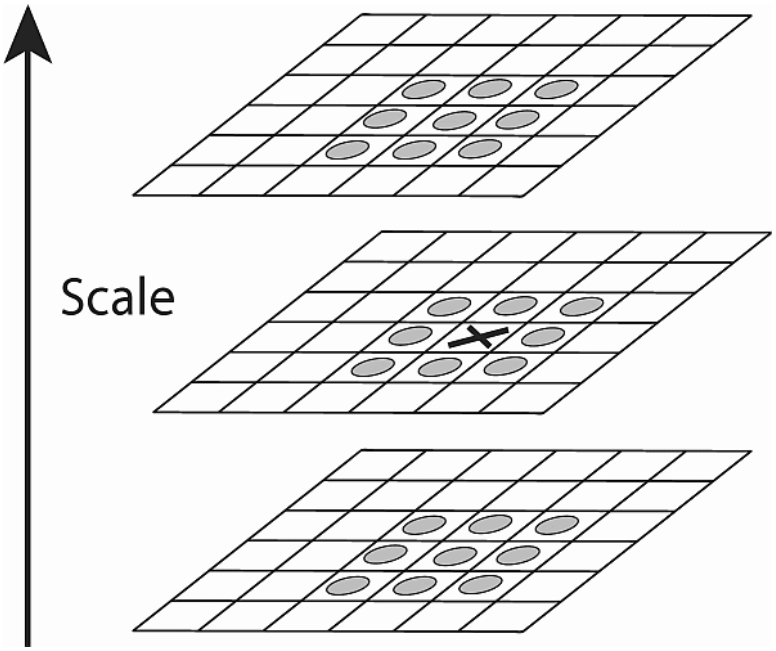


Figura 3.3: Máximos locales en espacio de escala, imagen obtenida de [Low04].

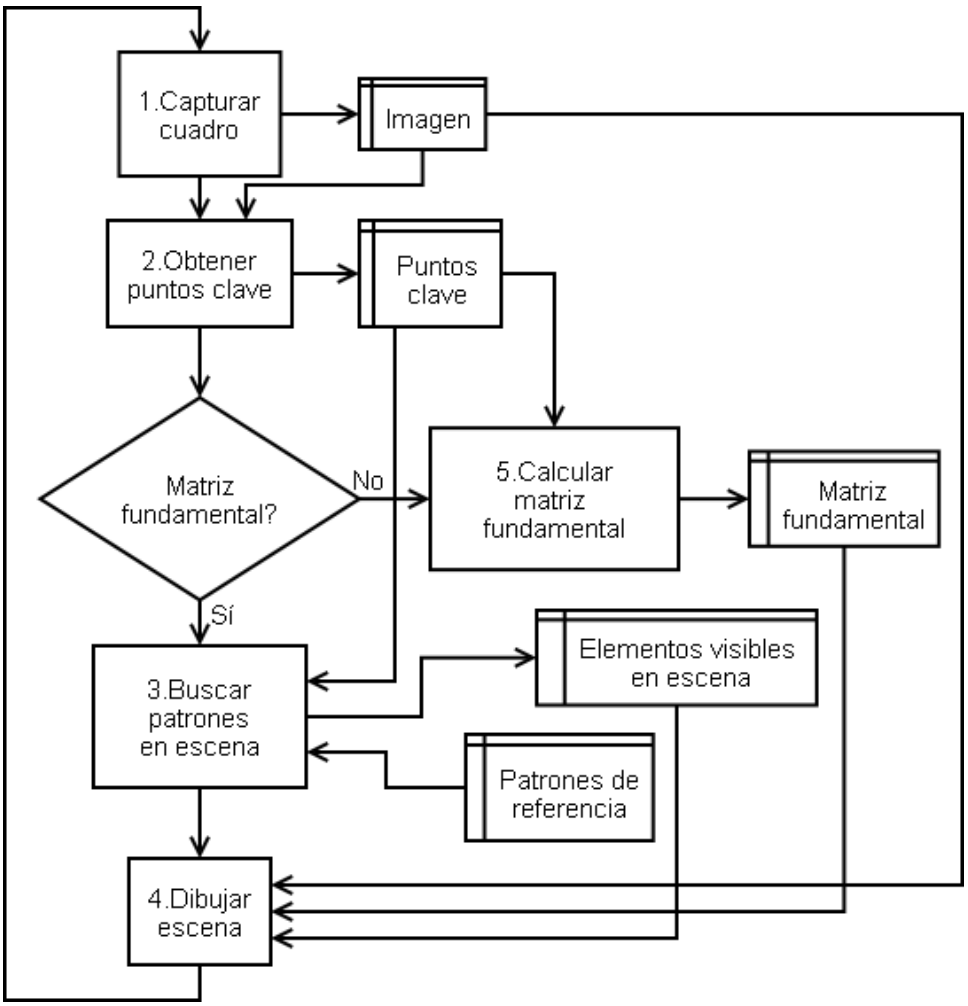


Figura 3.4: Estructura del Módulo de Realidad Aumentada.

Figura 3.5: Relación entre las coordenadas de los cuadros de la cámara y el mundo.

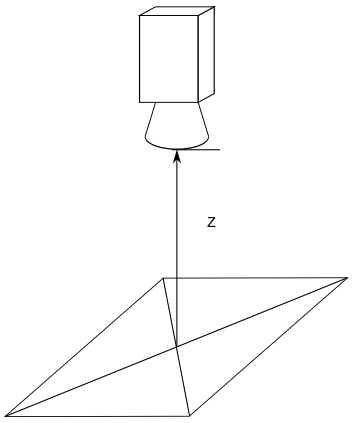


Figura 3.6: Obtención de las coordenadas del mundo de los puntos de la imagen de referencia.

Capítulo 4

Diseño e Implementación

En este capítulo se dará cuenta de los detalles del diseño y la implementación del sistema de Realidad Aumentada, al cual se le puso el nombre de **ColadeRA**. Estos detalles están relacionados con la configuración e integración del rastreador de características basado en SIFT con el sistema gráfico en OpenGL.

4.1. Diseño

4.1.1. Seleccionando un nombre para el software

Para el sistema de Realidad Aumentada se quería seleccionar un nombre que tuviera alguna relación con las iniciales de Realidad Aumentada (RA) y, si era posible, que indicara de qué clase de sistema de rastreo se estaba utilizando, a lo cual el primer nombre nombre consideramos fue **ColadeRA**.

4.1.1.1. Sobre ColadeRA

La elección de ese nombre se debe a que la palabra en inglés "sift", la cual tiene las mismas letras que el acrónimo SIFT, significa colar, o sea, separar cosas usando un colador o coladera. Elegimos la palabra "coladera" porque ésta es una palabra relacionada con la traducción de "sift" al Español y contiene las letras de las iniciales de Realidad Aumentada. La forma de nombrar al software es "ColadeRA", con las letras RA en mayúscula, para destacar que se trata de un sistema de Realidad Aumentada. Esta forma de nombrar al sistema está derivada de los nombres de algunas implementaciones de software para RA hechas por desarrolladores angloparlantes, los cuales destacan las iniciales AR (Augmented Reality) en el nombre, unos ejemplos son: ARToolkit, ARToolkit Plus, ARTag, BazAR, HandyAR.

4.1.2. Estructura del Sistema

El sistema está formado por las siguientes clases (Fig. 4.1), las cuales no tienen herencia entre si, la interacción se hace mediante paso de parámetros:

Clase ConfigReader: Esta clase sirve para leer archivos de configuración mediante las funciones para almacenamiento de datos de OpenCV. En el caso del sistema ColadeRA, el archivo de configuración tiene atributos con valores numéricos enteros, valores numéricos de punto flotante y de cadena de carácteres, por lo que solamente se están llamando las funciones de OpenCV para leer entradas de configuración con esos tipos de datos. Todos los valores se recuperan con el nombre que tienen dentro del archivo de configuración, el cual se pasa como parámetro a la función para recuperar datos, en forma de una cadena de carácteres. Toda clase que necesite leer valores contenidos en un archivo de configuración debe obtenerlos a partir de esta clase.

Clase VideoCapture: Esta clase se encarga de capturar cuadros de un flujo de video en una cámara o un archivo de video Audio Video Interleave, mediante las funciones de OpenCV para captura de vídeo; los cuadros capturados por esta clase se pasan al rastreador SIFT para extraerle sus características invariantes.

Clase SIFTFeature: Esta clase contiene una implementación del algoritmo SIFT [Low04], el cual sirve para extraer las características que son invariantes a la escala de una imagen. Al rastreador se le pueden configurar los parámetros del radio de desenfoque gaussiano, el umbral de contraste y la capacidad de duplicar el tamaño de la imagen para encontrar más características invariantes.

Clase SIFTMatcher: Esta clase sirve para hallar correspondencias entre las coordenadas de las características obtenidas por dos rastreadores SIFT, utiliza el algoritmo RANSAC para extraer las correspondencias y el resultado son dos conjuntos de puntos que pertenecen a las correspondencias entre ambos rastreadores.

CameraPosition: Esta clase calcula la posición de la cámara a partir de las correspondencias obtenidas por la clase SIFTMatcher mediante las funciones de OpenCV para calcular los parámetros extrínsecos de la cámara utilizando dos conjuntos de puntos, los parámetros intrínsecos de la cámara y los valores de distorsión de la cámara, el resultado es un vector de rotación y un vector de traslación, que pueden ser utilizados por una bibliotecas de gráficos tridimensionales, como OpenGL.

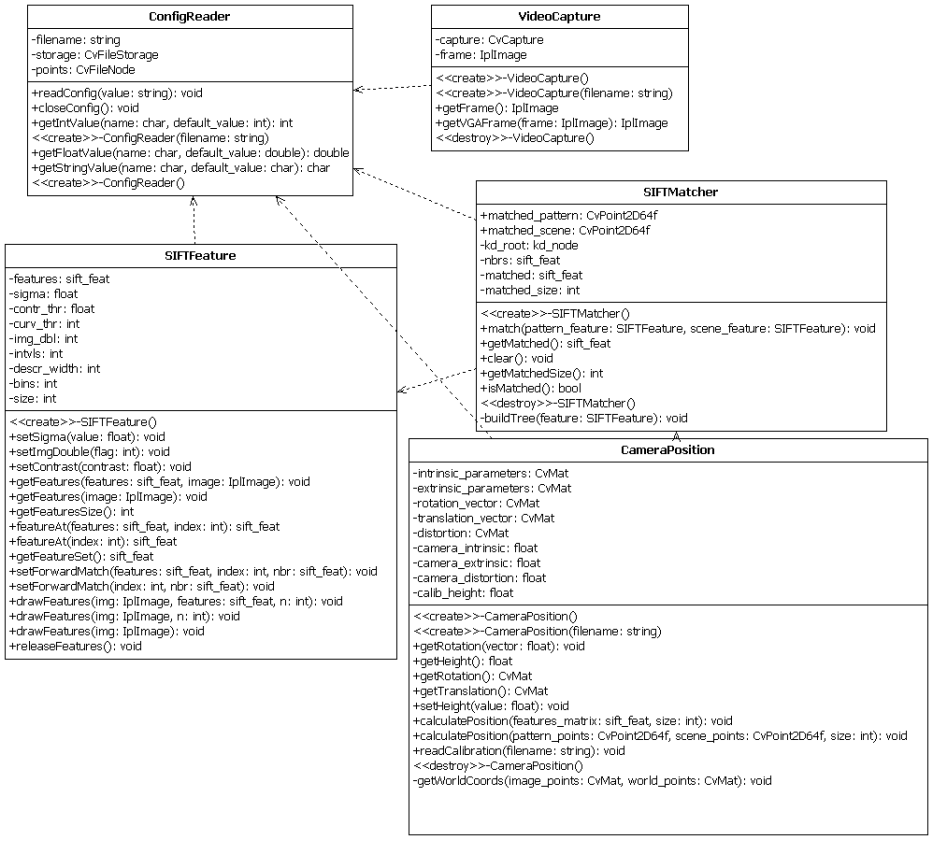


Figura 4.1: Diagrama de Clases de ColadeRA

4.2. Implementación

Para crear la aplicación de Realidad Aumentada, hay que integrar los módulos de Vídeo y Rastreo de Características (Fig. 3.1) usando los datos que genera cada uno. El módulo de Vídeo genera imágenes bidimensionales, mientras que el módulo de Rastreo de Características genera la matriz fundamental que describe la pose de la cámara real. La imagen bidimensional se utiliza para crear la imagen de fondo de la aplicación de Realidad Aumentada, la cual se dibuja sobre un plano, el cual debe ser dibujado antes de que se aplique la matriz fundamental a los demás objetos tridimensionales, de lo contrario, el plano del fondo se moverá de la misma forma que la cámara, con lo cual se rompería la integración entre la escena real y la escena virtual.

El procedimiento para colocar un objeto virtual sobre la escena real requiere

definir una imagen que servirá como marca de referencia, la cual puede ser un objeto que ya esté en la escena o un objeto que se añadirá después. Una vez seleccionada la imagen de referencia, se le aplica el rastreador SIFT para obtener sus descriptores invariantes, los cuales se corresponderán con los descriptores invariantes en cada cuadro del flujo de vídeo para obtener su orientación con relación a la cámara.

4.2.1. OpenGL

La biblioteca OpenGL es una interfaz de software para hardware de gráficos. Consiste en cerca de 150 comandos distintos que pueden usar para especificar los objetos y operaciones necesarias para producir aplicaciones tridimensionales interactivas [BSW+05].

OpenGL está diseñada como una interfaz independiente de hardware para ser implementada en diferentes plataformas de computadoras. Para obtener esas cualidades, no hay comandos para ejecutar tareas de manejo de ventanas ni para interactuar con el usuario en OpenGL; en su lugar, se debe trabajar a través de los controles del sistema de ventanas del hardware en uso. De la misma forma, OpenGL no proporciona comandos de alto nivel para describir modelos tridimensionales. Tales comandos podrían permitir la especificación de formas relativamente complicadas como automóviles, partes del cuerpo, aeroplanos o moléculas. Con OpenGL, se debe construir el modelo a partir de un conjunto de primitivas geométricas: puntos, líneas y polígonos [BSW+05].

4.2.1.1. GLUT

GLUT son las iniciales de OpenGL Utility Toolkit (Juego de Herramientas de Utilidad para OpenGL), se trata de un juego de herramientas independiente del sistema de ventanas para escribir programas de OpenGL. Implementa una interfaz de programas de aplicación (Interfaz de Programación de Aplicaciones) simple para aplicaciones con ventanas para OpenGL. GLUT hace que la programación en OpenGL sea considerablemente más fácil de aprender y explorar. GLUT provee una Interfaz de Programación de Aplicaciones portátil, así que es posible escribir un solo programa en OpenGL que funcionen en todas las plataformas de sistemas operativos para Computadoras Personales y Estaciones de Trabajo.

GLUT está diseñado para construir programas de OpenGL de tamaño pequeño a mediano. Mientras que GLUT es adecuado para el aprendizaje de OpenGL y el desarrollo de aplicaciones simples, GLUT no es un juego de herramientas sofisticado, así que para las aplicaciones grandes que requieran de interfaces de usuario sofisticadas es mejor que utilicen los juegos de heramientas para ventanas nativos para cada plataforma. GLUT es sencillo, fácil y pequeño [Kil97].

La biblioteca GLUT tiene enlaces de programación para C, C++ (el mismo que C), FORTRAN y ADA. El código fuente de GLUT es portátil para prácticamente todas las implementaciones y plataformas OpenGL [Kil97].

El juego de herramientas soporta:

- Ventanas múltiples para renderizado en OpenGL
- Procesamiento de eventos dirigido por retrollamadas
- Dispositivos de entrada sofisticados
- Una rutina para "tiempo muerto" y contadores de tiempo
- Menúes en cascada
- Rutinas de utilidad para generar objetos sólidos y de malla de alambre
- Soporte para fuentes de bitmap y vectoriales
- Funciones misceláneas para administración de ventanas

4.2.1.2. Integrando OpenCV con OpenGL

La biblioteca OpenCV tiene funciones básicas para crear ventanas en las cuales se muestran los resultados del procesamiento de las imágenes. Sin embargo, las funciones para dibujar contenido en las ventanas están limitadas a dos dimensiones, lo cual no es adecuado para aplicaciones de Realidad Aumentada, que trabajan con gráficos tridimensionales. Por otra parte, OpenGL no tiene funciones para Visión Computacional ni para capturar imágenes de cámaras y se requieren funciones de Visión Computacional para estimar la posición de la cámara real y convertirla en una matriz que pueda usar OpenGL para modificar la posición de la cámara virtual en la aplicación tridimensional interactiva.

Para integrar OpenCV en OpenGL, debemos buscar algo que tengan en común ambas bibliotecas para hacer la conexión e integrar en contenido de la primera biblioteca en el flujo de trabajo de la segunda. Ese factor común son las imágenes bidimensionales, las cuales OpenCV puede obtener a partir de un archivo de imagen, un archivo de video o una cámara, cuyo contenido en memoria puede copiarse a un objeto de textura en OpenGL, el cual es usado posteriormente para aplicarse sobre la superficie de un polígono [Mil08].

Los pasos para integrar imágenes de OpenCV en OpenGL son los siguientes (Fig. 4.2):

- En el primer paso, se utiliza OpenCV para capturar una imagen, cuyo origen puede ser un archivo de imagen o un archivo de video o una cámara.
- La siguiente operación consiste en preguntar si OpenGL ha creado un objeto de textura, para almacenar el contenido de la imagen capturada.
- Si no sea ha creado el objeto de textura, se indica a OpenGL que cree un objeto de textura.

- El siguiente paso consiste en copiar el contenido de la imagen al objeto de textura.
- A continuación, se aplica la textura sobre un plano que cubra toda la ventana de la aplicación, para que sirva como imagen de fondo.
- Luego se dibujan todos los objetos tridimensionales en frente del plano con la imagen del fondo.

Este proceso se repite hasta que termine el programa; en el caso de las imágenes capturadas de archivos de video y cámaras, el resultado de este ciclo de integración es una imagen de fondo que cambia a intervalos de la velocidad de cuadro del flujo de video, dando como resultado una película. Por supuesto, este ciclo se puede usar para poner textura a cualquier objeto tridimensional, pero en el caso de las aplicaciones de Realidad Aumentada, las imágenes capturadas se colocan como fondo para representar la escena real en la que se sobreponen los objetos virtuales.

4.2.2. OpenCV

Esta biblioteca [Int] proporciona las funciones y tipos de datos necesarios para los módulos del sistema relacionados con el procesamiento de la imagen y cálculos aritméticos, tales como captura de video, operaciones con matrices; además de proporcionar funciones de ayuda para la aplicación, como el almacenamiento e interpretación de archivos de configuración.

4.2.3. Rastreador SIFT

Para ColadeRA se utilizó la implementación de SIFT hecha por el Dr. Rob Hess de la Universidad del Estado de Oregon [Hes07], la cual fue hecha en lenguaje C, tiene proyectos de compilación para que pueda ser construido en Visual Studio 2003 y en Linux y está diseñado alrededor de las bibliotecas GSL (GNU Scientific Library) y OpenCV.

Algunas de las funciones más importantes de esta implementación son:

- Extracción de las características SIFT de una imagen.
- Construcción del árbol k-d para ejecutar búsquedas de vecino más cercano.
- Búsqueda del vecino más cercano mediante el algoritmo Best Bin First (BBF).
- Detección y correspondencia de características mediante RANSAC.

No se utilizó el algoritmo GTM para la correspondencia de características debido a que su tiempo de procesamiento hacia que el sistema fuera demasiado lento para ser usado en aplicaciones de imágenes en tiempo real, veáse la Sección 5.4 para más información.

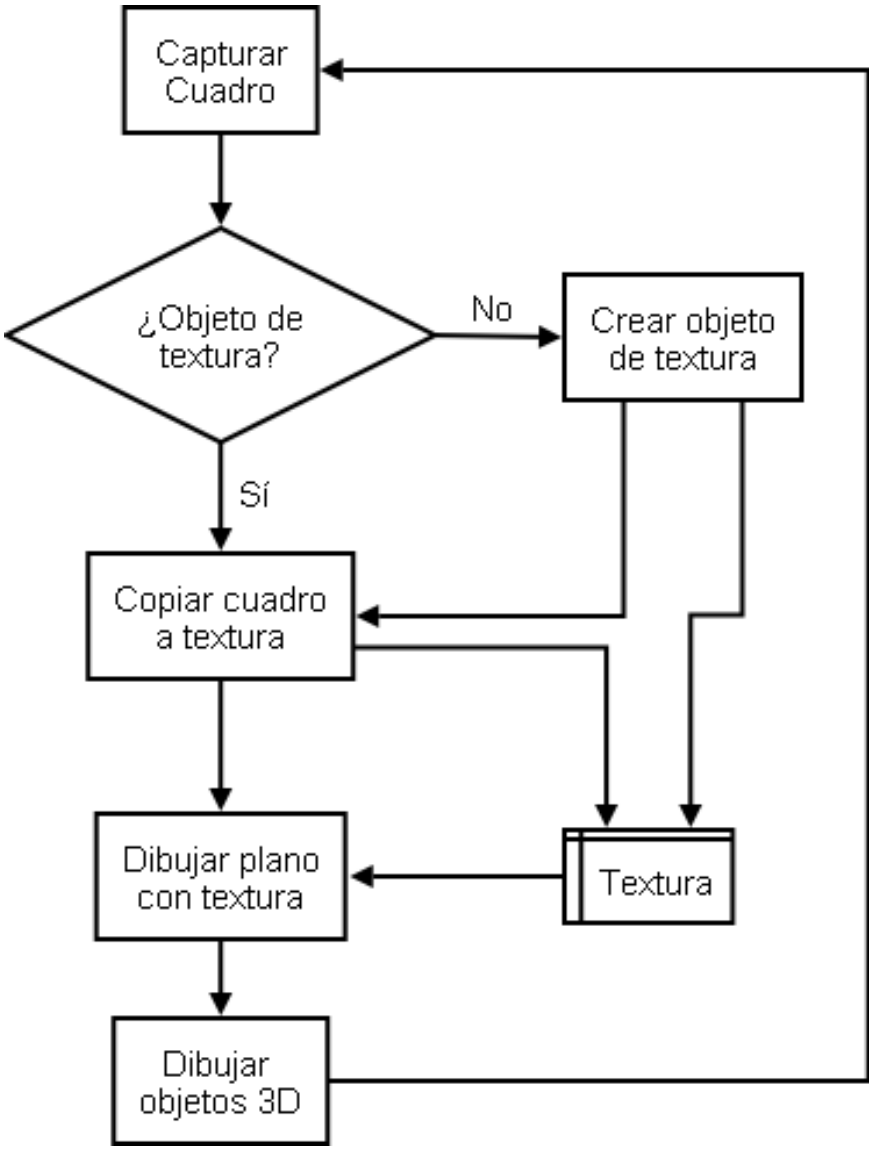


Figura 4.2: Integración de OpenCV en OpenGL.

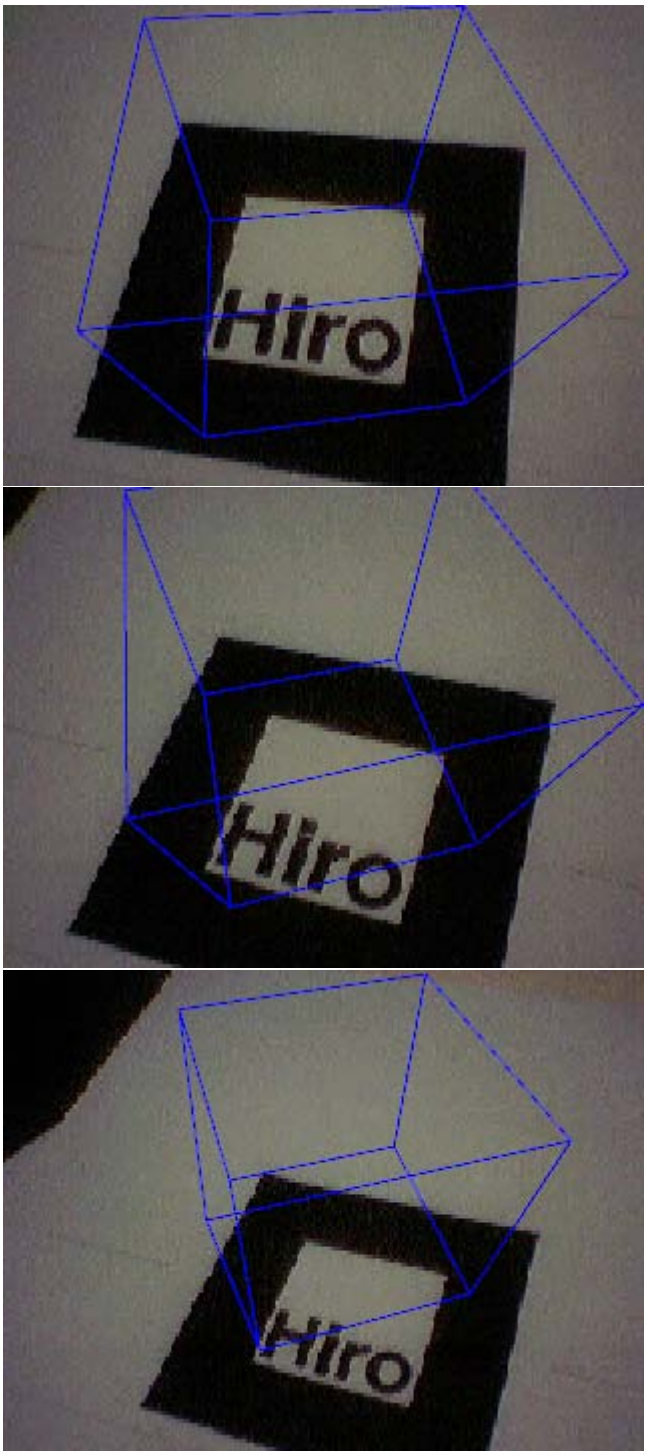


Figura 4.3: Sistema ColadeRA.

Capítulo 5

Pruebas

5.1. Pruebas aplicadas al sistema

Al sistema se le aplicaron pruebas para evaluar su comportamiento con diferentes tipos de superficie y puntos de vista de la cámara.

5.1.1. Propiedades ópticas de la superficie

El objetivo de esta prueba era averiguar qué clase de superficies son las que presentan menos conflictos cuando se les extrae sus características invariantes. Hay que hacer notar que si la cámara tiene control automático de exposición, hay que esperar a que termine el ajuste antes de continuar con el rastreo.

5.1.1.1. Superficie brillosa o semimate

Para esta prueba, se usó la portada de una revista, la cual tenía un acabado brilloso, sobre ésta se hicieron varias tomas con la cámara en diferentes ángulos y la revista fue rotada para modificar la ubicación del reflejo del foco.

No hubo problemas en el rastreo cuando el cambio en la intensidad de la iluminación sobre la superficie de la revista era uniforme (Fig. 5.1a), una situación donde el rastreo no funcionaba era cuando el foco generaba una zona brillante que alteraba la apariencia de la imagen de la revista (Fig. 5.1b), aunque a veces una zona brillante generaba características que eran empatadas erróneamente (Fig. 5.1c).

5.1.1.2. Superficie reflejante

Para esta prueba, se usó la parte grabada de un disco compacto, la cual tenía un acabado reflejante, sobre ésta se hicieron varias tomas con la cámara en diferentes ángulos.

En el caso de las superficies reflejantes, las características se generan a partir de los reflejos de los objetos y las fuentes de luz (Fig. 5.2a). No hay mucho rango

de movimiento para la cámara o el objeto, la más minima desviación modifica la organización de las características y se pierde el rastreo (Fig. 5.2b).

5.1.1.3. Superficie transparente/brillosa

En esta prueba se utilizó la bolsa de un disco compacto que estaba pegada a la portada de una revista, la cual tenía un acabado reflejante,sobre ésta se hicieron varias tomas con la cámara en diferentes ángulos.

Al tratarse de una superficie transparente, las características obtenidas pertenecen al objeto que está detrás de la superficie, además de obtenerse características adicionales por los brillos que aparecen sobre la superficie (Fig. 5.3a). Esta clase de superficie no es adecuada para rastrear porque los cambios de orientación, ya sean de la cámara o el objeto, modifican la forma de los brillos y ya no se generan características que puedan empatarse con el patrón de referencia (Fig. 5.3b).

5.1.1.4. Superficies no relacionadas con el patrón de referencia

Durante las pruebas de propiedades ópticas, se encontraron superficies que no se parecian mucho al patron de referencia y que generaban un número pequeño de características que eran correspondidas con las del patrón de referencia (Fig. 5.4).

5.1.2. Pruebas de puntos de vista de la cámara

Este conjunto de pruebas se hizo para determinar cuáles eran los límites en los grados de libertad de la cámara que permiten tener un rastreo estable. Un detalle a tomar en cuenta es que el rastreo con algoritmo SIFT se está haciendo con una configuración que extrae un número bajo de características de las imágenes, para que el programa tuviera un desempeño lo más cercano poisible al tiempo real.

Las pruebas de punto de vista se hicieron entre un sistema basado en ARToolkit corriendo sobre OpenSceneGraph y el sistema ColadeRA corriendo sobre GLUT; el subsistema de gráficos, sea OpenScenegraph o GLUT, no afecta el rendimiento del sistema de Realidad Aumentada. Como elemento de prueba para ambos sistemas de software,se utilizó el patrón Hiro de ARToolkit (Fig. 5.6).

En todos los casos, la inestabilidad en el rastreo se manifiesta con cambios abruptos en la posición y orientación del objeto virtual, debido a que no hay suficientes puntos empatados correctamente para obtener los parametros extrinsecos de la cámara.

5.1.2.1. Distancia máxima

En esta prueba se alejó la cámara hasta una distancia en el que el rastreo era inestable; hay que tomar en cuenta que la cámara usada es de enfoque fijo, asi que se capturaban imágenes cada vez más borrosas conforme la cámara se

alejaba del patrón de referencia, así que las características extraídas eran de los espacios de escala más grande; este efecto es similar a modificar el nivel de desenfoque gaussiano del algoritmo SIFT (Tab. 5.1).

5.1.2.2. Desplazamiento lineal máximo

En esta prueba se movió la cámara hacia los lados, para ver qué tanto puede desplazarse la imagen de referencia dentro de la escena antes de que el rastreo empiece a presentar inconsistencias, las cuales empezaron cuando las correspondencias se daban entre subconjuntos de puntos de la imagen de referencia dentro de la escena y características que no pertenecen a la imagen de referencia, el efecto era que el área donde se colocaba el objeto virtual se desplazaba junto con la cámara (Tab. 5.2).

5.1.2.3. Ángulo máximo

En esta prueba de movió la cámara en órbita circular con respecto a la imagen de referencia hasta un punto donde el rastreo fuera inestable; esto se debe a que el algoritmo SIFT es parcialmente invariante al punto de vista tridimensional de la cámara., así que las características en las escalas más bajas se pierden con el cambio de punto de vista (Tab. 5.3).

5.1.2.4. Conclusiones

ColadeRA es capaz de funcionar a distancias mucho más cercanas a las que trabaja ARToolkit, pero su alcance, tanto a larga distancia como el ángulo de punto de vista, es mucho más corto que el de ARToolkit.

5.1.2.5. Soluciones

A partir del codigo disponible, una forma de incrementar la tolerancia a la inestabilidad es incrementando el número de características extraídas por imagen, lo cual se logra habilitando la opción del algoritmo SIFT para que duplique el tamaño de la imagen antes de pasar a extraer caracerísticas; el problema con esta configuración es que el desempeño del programa se reduce en una gran medida debido a que algoritmo para incrementar el tamaño de imagen siempre tiene una complejidad de $m \times n$ o n^2, dependiendo de la forma de la imagen. No hay solución simple para el problema de las superficies no relacionadas, porque no es fácil distinguir las características entre una superficie relacionada parcialmente ocluida y una superficie no relacionada.

5.2. Factores de impacto en el desempeño

5.2.1. Rastreador SIFT

El algoritmo de detección de características SIFT tiene que ser adaptado para aplicaciones de procesamiento de imágenes en tiempo real, ya que hay

parámetros que, al tener ciertos valores, incrementan el tiempo de cómputo del algoritmo:

- Duplicación de tamaño de la imagen (image doubling), esta operación tiene una complejidad base de $m \times n$, a lo cual hay que agregarle las operaciones adicionales asociadas al algoritmo de interpolación.
- Sigma, este parámetro controla el radio del suavizado gaussiano; a mayor radio, las características a escalas grandes son las únicas que se conservan.
- Umbral de contraste, este parámetro es el que tiene más peso en la detección de características y el que incrementa en mayor medida el tiempo de computación del algoritmo para cada cuadro, ya que selecciona los máximos locales en base al contraste con sus vecinos. Entre más bajo sea este umbral, más máximos locales son caracterizados, aunque se trata de máximos locales que se pierden en cambios de contraste.

Una cámara mal enfocada equivale a aplicar un desenfoque de tipo gaussiano en la imagen, por lo que el rastreador SIFT solo va a encontrar las características que tengan las escalas más grandes.

Para reducir la complejidad de cómputo del rastreador SIFT hay que deshabilitar la duplicación del tamaño de la imagen, la cual es utilizada por en la definición original del algoritmo SIFT para obtener una mayor cantidad de características y sirve para hacer más robusto el empatado entre la imagen de referencia y la escena, asi como ajustar los valores del radio de suavizado gaussiano y el umbral de intensidad para modificar la cantidad de características que extraen en cada imagen. Otra alternativa para incrementar el rendimiento sería una implementación en arquitecturas paralelas, tales como Graphics Processing Units o clusters.

5.2.2. Búsqueda con árbol k-d

Otro posible factor de impacto en el desempeño del programa va a ser la construcción del árbol k-d, ya que su construcción es a partir de puntos al azar, por lo que hay posibilidad de que algunas ramas tendrán apariencia de listas lineales. La búsqueda en árboles k-d usando BBF no representa un gran impacto en el desempeño del programa porque fue diseñado para ejecutarse rápidamente, ya que una búsqueda por backtracking es computacionalmente intensiva. El algoritmo BBF tiene límites modificables para reducir tiempos de busqueda, tales como el número de nodos a buscar, el cual sirve para terminar rápidamente la búsqueda en zonas con alta densidad de puntos.

5.3. Problemas con el rastreador SIFT

La implementación de SIFT usada en esta tesis tiene limitaciones en la generación de características con ciertos tipos de imagen.

5.3.1. Múltiples instancias

La primera limitación surge cuando en la escena hay dos o más instancias del mismo patrón de referencia, dependiendo de su orientación se dan los siguientes casos: 1) que las correspondencias entre patrón y escena se repartan entre las dos instancias del patrón, el cual sucede cuando éstas tienen diferentes orientaciones (Fig. 5.5b) o 2) que no haya correspondencias en lo absoluto, el cual sucede cuando las dos instancias tienen la misma orientación (Fig. 5.5c).

5.3.2. Características ópticamente inconstantes

Hay objetos que por sus propiedades ópticas generan características que no son constantes cuando cambia el punto de vista de la cámara; el primer caso se da con los objetos con superficies brillantes, los cuales generan características que desaparecen al cambiar a un ángulo donde la fuente la luz donde la fuente de luz incide sobre otro objeto; otro caso se da con los objetos transparentes, ya que los patrones que puedan captarse en ellos pertenecen a los objetos que estén detrás de ellos y, por último, los objetos con superficies reflejantes, a menos que tengan irregularidades que impidan que se refleje la luz, siempre van a tener las características de los objetos que se reflejan sobre su superficie.

Los objetos con cualquiera de estas propiedades ópticas no son adecuados para usarse como patrones de referencia para el sistema de Realidad Aumentada, por lo que se recomienda usar superficies opacas con acabado semi-mate o mate.

5.4. Rendimiento de GTM

Cuando se probó el algoritmo GTM [AFE+08] para empatado de características en un flujo de video, los resultados fueron que GTM no funciona adecuadamente para aplicaciones de visión computacional en tiempo real en base a los siguientes hechos:

1. Cuando se pone un valor de conectividad grande y hay muy pocos puntos para empatar, el algoritmo falla por que no se pueden llenar las estructuras de datos internas del programa.

2. En una situación donde hay muchos puntos para empatar y un valor de conectividad pequeño, el tiempo de procesamiento es grande.

Aún cuando se hallara un punto medio de conectividad, éste usualmente seria un valor pequeño, por lo que aún existe impacto en el rendimiento de programa por el incremento en el tiempo de procesamiento.

(a) Empatado correcto

(b) Carencia de correspondencias por cambio irregular de intensidad

(c) Positivos falsos generados por zona brillante

Figura 5.1: Superficie brillosa/semimate

(a) Empatado correcto (b) Carencia de correspondencias por cambio de posición

Figura 5.2: Superficie reflejante

(a) Empatado correcto (b) Carencia de correspondencias por cambio en punto de vista

Figura 5.3: Superficie transparente

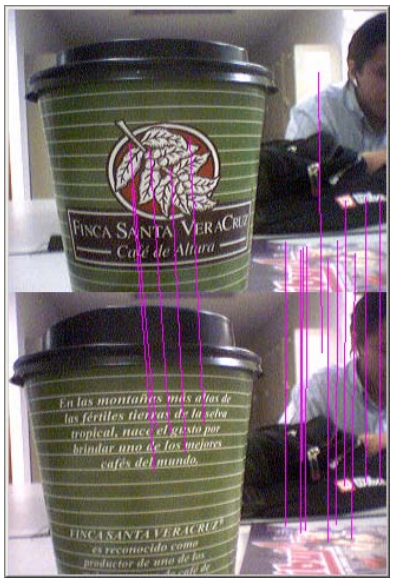
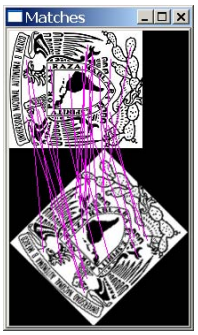
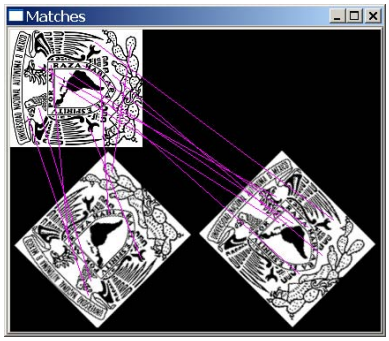


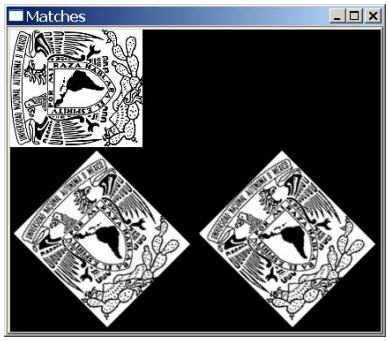
Figura 5.4: Superficies no relacionadas con el patrón de referencia



(a) Correspondencia de un patrón



(b) Correspondencia de patrones con diferentes orientaciones



(c) Correspondencia de patrones con la misma orientación

Figura 5.5: Problemas de correspondencia con múltiples instancias

Figura 5.6: Patrón de referencia para las pruebas.

Distancia (cms)	ARToolkit	ColadeRA
5	-	OK
10	-	OK
15	OK	OK
20	OK	OK
30	OK	OK
40	OK	OK
50	OK	-
60	OK	-
70	OK	-
80	OK	-
90	Hay objeto virtual, empieza la inestabilidad.	-
100	Hay objeto virtual, No se puede empezar a rastrear desde esta distancia	-
110	Hay objeto virtual, No se puede empezar a rastrear desde esta distancia	-
120	Hay objeto virtual, No se puede empezar a rastrear desde esta distancia	-

Cuadro 5.1: Pruebas de distancia entre ARToolkit y ColadeRA.

Distancia (cms)	ARToolkit	ColadeRA
5	-	1.5
10	1.6	3.5
15	3.35	5.5
20	5.1	7.5
30	8.6	11.5
40	12.1	-
50	15.6	-
60	19.1	-
70	22.6	-
80	26.1	-
90	29.6	-
100	33.1	-
110	36.6	-
120	40.1	-

Cuadro 5.2: Pruebas de desplazamiento lateral entre ARToolkit y ColadeRA, distancias en centímetros.

Ángulo (grados)	ARToolkit	ColadeRA
90	OK	OK
80	OK	OK
70	OK	OK
60	OK	OK
50	OK	OK
40	OK	-
30	OK	-
20	OK	-
10	OK	-
5	OK	-
0	-	-

Cuadro 5.3: Pruebas de ángulo de punto de vista entre ARToolkit y ColadeRA.

Capítulo 6

Conclusiones

El objetivo de esta tesis es desarrollar un sistema de Realidad Aumentada sin marcas utilizando detalles visuales, también llamados características, de cualquier objeto disponible en la escena como marco de referencia. El objeto del mundo real, que es usado como punto de referencia, es capturado, se le extraen sus características y se almacenan en términos de coordenadas tridimensionales. Después, se extraen las características de los objetos que se ven en la cámara y se trata de hallar correspondencias entre las coordenadas tridimensionales del objeto de referencia y las coordenadas bidimensionales del objeto de referencia en la escena. A partir de estas correspondencias, se calcula la posición de la cámara en términos de rotación y traslación, estos valores son utilizados por un sistema de gráficos tridimensionales para ubicar un objeto virtual sobre el marco de referencia que representan los puntos.

Gracias al uso de un rastreador de características invariantes a la escala y parcialmente invariante a cambios fotogramétricos, es posible seguir las características de la escena con un grado de libertad aceptable. El uso de un algoritmo de búsqueda espacial probabilístico permite localizar y empatar características con tiempos de búsqueda bajos, que es lo que se requiere para procesamiento en tiempo real.

Los resultados de las pruebas arrojaron que el sistema puede funcionar a distancias mucho más cortas que las que pueden manejar los sistemas de Realidad Aumentadas basados en marcas fiduciarias, los cuales dejan de rastrear cuando el borde del cuadro del patrón no aparece completo en la imagen de cámara.

Las aportaciones del sistema son que es posible generar el punto de referencia para la cámara mediante una imagen capturada por la misma y que los parámetros de sensibilidad del rastreador de características invariantes se pueden ajustar para que el desempeño del sistema se ajuste a computadoras con diferentes velocidades de CPU. El sistema desarrollado en esta tesis es muy básico, ya que el rastreo y empatado de características funciona con un patrón a la vez.

Las limitaciones de este sistema es que funciona en un intervalo de distancias

y puntos de vista más limitado con relación a sistemas de Realidad Aumentada basados en marcas fiduciarias; esto se debe a que, con el objetivo de que el sistema tuviera desempeño en tiempo real, se modificaron los parámetros del rastreador de características invariantes para que extrajera un número pequeño de características, lo cual genera inestabilidad cuando se llega a los límites ópticos de la cámara y al experimentarse cambios en la estructura de color de la imagen. Otro problema que se encontró es que el marco de referencia se desplaza con la cámara cuando hay correspondencias entre subconjuntos de las coordenadas de la imagen, esta situación surge cuando una parte del conjunto de las características del objeto de referencia en escena es visible en la cámara.

Los detalles que se pueden agregar en futuras implementaciones son crear una capa de abstracción que permita usar el sistema en diversos softwares de visualización tridimensional y motores gráficos para juegos, utilizar un rastreador SIFT que se ejecute los procesadores de tarjetas gráficas, lo cual permitiría generar y rastrear grandes cantidades de características y programar el rastreo y empatado de caracteristicas de múltiples patrones en la escena.

Índice de figuras

Índice de cuadros

Lista de algoritmos

Glosario

Á

árbol En teoría de grafos, un árbol es un grafo en el que dos vértices están conectados por exactamente un camino, pág. III.

A

arista Uniones entre nodos o vértices, pág. 25.

aritmética de punto fijo Aritmética en la que se usan números enteros para simular números reales, usando un número fijo de bits tanto para la parte entera y la parte fraccionaria, pág. 8.

Audio Video Interleave (AVI) Formato de archivo contenedor de audio y vídeo lanzado por Microsoft, pág. 13.

C

cola de prioridad Estructura de datos en la que los elementos se atienden en el orden indicado por una prioridad asociada a cada uno. Si varios elementos tienen la misma prioridad, se atenderán de modo convencional según la posición que ocupen, pág. 23.

convolución Es un operador matemático que transforma dos funciones f y g en una tercera función que en cierto sentido representa la magnitud en la que se superponen f y una versión trasladada e invertida de g, pág. 16.

correlación Es la fuerza y la dirección de una relación lineal entre dos variables aleatorias, pág. 7.

correspondencia Relación que realmente existe o convencionalmente se establece entre los elementos de distintos conjuntos o colecciones, pág. 15.

D

distancia Euclidiana Distancia ordinaria entre dos puntos de un espacio euclídeo, se calcula a partir del teorema de Pitágoras, pág. 16.

E

eigenvalor Es un conjunto espacial de escalares asociados a un sistema lineal de ecuaciones, pág. 18.

F

falso positivo Error estadístico que surge cuando se rechaza una hipótesis nula cuando en realidad es cierta, pág. 29.

fiduciaria Que depende del crédito y confianza que merezca, pág. II.

función de ventana Función que aplica suavizado gaussiano en intervalos, pág. 19.

G

grafo Conjunto de nodos relacionados con aristas, pág. 1.

Graphics Processing Unit (GPU) Unidad de Procesamiento de Gráficos, pág. 44.

H

heurística Algoritmo que abandona cualquiera de los objetivos de hallar buenos tiempos de ejecución y buenas soluciones, pág. 8.

histograma Representación gráfica de una variable en forma de barras, donde la superficie de cada barra es proporcional a la frecuencia de los valores representados, pág. 19.

I

imagenología Estudio de las imágenes bidimensionales para extraer información útil, pág. 2.

Interfaz de Programación de Aplicaciones (API) Es el conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción, pág. 13.

interpolación Construcción de nuevos puntos partiendo del conocimiento de un conjunto discreto de puntos, pág. 17.

M

matriz de adyacencia Matriz que contiene las relaciones entre los nodos de un grafo, pág. 25.

multimedia Que utiliza conjunta y simultáneamente diversos medios, como imágenes, sonidos y texto, en la transmisión de una información, pág. v.

N

normalización Operaciónque cambia el rango de un conjunto de valores, pág. 20.

O

oclusión Manera en que un objeto más cercano a la cámara enmascara u obstruye a un objeto que está más lejos de la cámara, pág. 5.

octava Duplicación o partición en dos de una frecuencia, pág. 16.

ortogonalidad Propiedad de dos vectores en el que su producto escalar es cero, pág. 27.

P

patrón Un objeto o substancia que se emplea como muestra para medir alguna magnitud o para replicarla, pág. iii.

polígono Figura geométrica plana limitada por al menos tres segmentos rectos consecutivos no alineados, llamados lados, pág. 26.

R

redundancia Es la duplicidad de componentes críticos de un sistema para incrementar su confiabilidad, pág. 8.

registro Acción y efecto de mirar, examinar algo con cuidado y diligencia, pág. vi.

T

textura En computación gráfica, una imagen de mapa de bits aplicada a una superficie, pág. 37.

traza Suma de los elementos de la diagonal principal de una matriz cuadrada, pág. 18.

U

umbralizado Proceso en el cual los pixeles de una imagen son marcados como fondo si su valor es menor a un valor mínimo de registro, pág. 7.

V

ventana Un área visual de un interfaz de usuario en un sistema informático, pág. 36.

vértice Punto donde concurren las dos semirrectas que conforman un ángulo, pág. 25.

viñeteado Pérdida de claridad hacia las esquinas y lados de una imagen, pág. 8.

vista canónica Vista sujeta a restricciones ortogonales, pág. 7.

Bibliografía

[AFE+08] Wendy Aguilar, Yann Frauel, Francisco Escolano, M. Elena Martinez-Perez, and Arturo Espinosa-Romeroand Miguel Angel Lozano. A robust graph transformation matching for non-rigid registration. *Image and Vision ComputingIn Press*, 2008.

[Azu95] R. Azuma. A survey of augmented reality, 1995. http://citeseer.ist.psu.edu/azuma95survey.html.

[Ben75] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.

[BKM99] Reinhold Behringer, Gudrun Klinker, and David W. Mizell, editors. *Augmented Reality:Placing Artificial Objects in Real Scenes: Proceedings of IWAR 1998.* A K Peters Ltd., 1999.

[BL97] J. Beis and D. Lowe. Shape indexing using approximate nearest-neighbor search in highdimensional spaces, 1997.

[BR05] Oliver Bimber and Ramesh Raskar. *Spatial Augmented Reality Merging Real and Virtual Worlds.* A K Peters LTD, 2005.

[Bri08] Encyclopedia Britannica. virtual reality :: Education and training, 2008. http://www.britannica.com/eb/article-253104/virtual-reality [Online, accessed 07-April-2008].

[BSW+05] OpenGL Architecture Review Board, Dave Shreiner, Mason Woo, Jackie Neider, and Tom Davis. *OpenGL Programming Guide.* Addison-Wesley Professional, 5th edition, 2005.

[Fia05] Mark Fiala. Artag, a fiducial marker system using digital techniques. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2:590–596 vol. 2, June 2005.

[Hes07] Rob Hess. Sift feature detector – a c implementation of a sift image feature detector., 08 2007. http://web.engr.oregonstate.edu/ hess/.

[Int] Intel. Open source computer vision library. http://opencvlibrary.sourceforge.net/.

[KB99] Hirokazu Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *IWAR '99: Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality*, Washington, DC, USA, 1999. IEEE Computer Society.

[KBP+00] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual object manipulation on a table-top ar environment. *ISAR*, page 111, 2000.

[Kil97] Mark Kilgard. Glut - the opengl utility toolkit, 1997. http://www.opengl.org/resources/libraries/glut/.

[LF06] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(9):1465–1479, Sept. 2006.

[LLF05] V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2:775–781 vol. 2, June 2005.

[Low99] D. Lowe. Object recognition from local scale-invariant features. pages 1150–1157, 1999.

[Low04] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[Mar06] Wendy Elizabeth Aguilar Martínez. Reconocimiento de objetos basado en la correspondencia estructural de características locales. Master's thesis, Universidad Nacional Autónoma de México, Noviembre 2006.

[Mil08] David Millan. Opencv & opengl, Mayo 2008. http://www.artresnet.com/david/tutorial.jsp?id=6.

[Pil] Julien Pilet. Bazar: A vision based fast detection library. "http://cvlab.epfl.ch/software/bazar/".

[SE97] Tomaso Poggio Shimon Edelman, Nathan Intrator. Complex cells and object recognition, 1997. Unpublished manuscript, http://kybele.psych.cornell.edu/ edelman/Archive/nips97.ps.Z.

[TV98] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[Wel93] Pierre Wellner. Interacting with paper on the digitaldesk. *Communications of the ACM*, 36:87–96, 1993.

[Wik02] Wikipedia. Augmented reality - wikipedia, the free encyclopedia, 2002. http://en.wikipedia.org/wiki/Augmented reality [Online; accessed 06-April-2008].

[Wik07] Wikipedia. Morton heilig - wikipedia, the free encyclopedia, 2007. http://en.wikipedia.org/wiki/Morton Heilig [Online, accessed 07-April-2008].

[WS07] Daniel Wagner and Dieter Schmalstieg. Artoolkitplus for pose tracking on mobile devices. In Helmut Grabner Michael Grabner, editor, *Computer Vision Winter Workshop*, 2007.