Universidad de las Ciencias Informáticas


Facultad 5

# Localización de objetos virtuales en el mundo real con técnicas de Realidad Aumentada.

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas.

**Autores:** Mileydi Moreno Mirabal.

Ernesto de la Cruz Guevara Ramírez.

**Tutor:** Lic. Lidiexy Alonso Hernández.

Ciudad de La Habana

Julio 2008

*Ciencia y libertad son llaves maestras que han abierto las*

*puertas por donde entran los hombres a torrentes,*

*enamorados del mundo venidero.*

***José Martí***

**DECLARACIÓN DE AUTORÍA**

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

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

**Mileydi Moreno Mirabal**

_____

Firma del Autor

**Ernesto de la Cruz Guevara Ramírez**

_____

Firma del Autor

**Lidiexy Alonso Hernández**

_____

Firma del Tutor

**DATOS DE CONTACTO**

Lidiexy Alonso Hernández

Licenciado en Ciencias de la Computación.
Profesor Instructor.
3 años de experiencia en la Realidad Virtual.

Correo Electrónico: lidiexy@uci.cu

**AGRADECIMIENTOS**

Agradecemos a la Revolución por haber creado este maravilloso proyecto futuro. A Fidel por su genialidad y por permitir hacer realidad nuestros sueños. A nuestro tutor Lidiexy por iniciarnos en el maravilloso mundo de la Realidad Aumentada y por el apoyo incondicional que nos ha brindado. Un especial agradecimiento a Yalice y Yoan por ser más que amigos, por sus consejos útiles y por guiarnos hacia el camino correcto, siendo ejemplos a seguir. A las profesoras Brigit y Cristina por tendernos su mano amiga cuando más falta nos hizo. A Karel y Maylin por ayudarnos dedicándonos parte de su tiempo. A Dayren y Yunieski por ayudarnos y soportarnos. A Siovel y a William por creer en nosotros y en nuestro trabajo. A Alexis, Lorenzo y Leonardo por ayudarnos con el diseño de los modelos 3D. A nuestros compañeros de aula y de proyecto. A nuestros profesores. A todas aquellas personas que de una forma u otra hicieron posible la realización de este trabajo y que por ser este un momento difícil se nos haya olvidado mencionar.

**Mileydi**

A Ada y Miguel por ser los mejores padres del mundo, por todo el amor que me han dado, por apoyarme siempre en mis decisiones, en los momentos de alegría y tristeza, por ser mi inspiración. Si alguien me preguntara cual es el mayor tesoro del mundo, sin duda alguna respondería mis padres. Por hacerme la persona más feliz del mundo, por el simple hecho de que existen, a ustedes gracias. A mi hermano Ariel por ser mi amigo, por el amor que me ha dado, por los sentimientos y valores que me ha inculcado, por regalarme tres bellos sobrinos y por ser el mejor hermano que mis padres me hubieran podido regalar. A Rodo en especial por ser mí hermano mayor, porque su ejemplo despertó en mí el amor al estudio. A mis abuelos mima y pipo por ser personas maravillosas, por la sabiduría, valores y principios revolucionarios que han inculcado a toda la familia. A mis abuelos paternos por el amor que me brindaron. A mis tíos Alicia, Haide, Roberto, Raúl y Rodolfo por el cariño y apoyo que siempre me han brindado, y el compromiso de convertirme en una profesional útil a la Revolución. A Yuliet por su bondad y apoyo incondicional. A la Gallega porque siempre tiene un gesto de cariño para

conmigo. A Isabel por ser mi madrina y a sus tías por quererme como a una nieta. A Zeyda por ser mi hermana y a su familia de la cual me siento miembro. Por estar siempre presente en los buenos y en los malos momentos, por su amistad incondicional a las personas que más que amigos han sido como hermanos en estos años de universidad a Yinet, Leonel, Yaily, Wendy, Enrique y Alexis. A mi amor Ernesto por ser mi novio, amigo y compañero, por sus palabras de aliento en los momentos más difíciles, a su familia por ser tan buenos conmigo.

**Ernesto**

Primero que todo agradecer a ese ser tan especial que me regaló la vida. A ese ángel que nunca se rindió y continuó su instinto maternal hasta lograr dar vida a dos semillas que supo cultivar y ya comienzan a dar los frutos que satisfacen el alma. A ti mami por ser a la misma vez mi mamá y mi papá, por estar siempre presente en los momentos más difíciles, porque Dios pensó en la ternura y no pudo hacer nada mejor que plasmarla en ti, por hacer de mi lo que hasta hoy he sido y seré, a ti mami, muchas gracias. A Naty por ser más que mi hermana, por ser mi compañera de mil batallas y por sacarme una sonrisa aún en los momentos más inesperados. A mama, mi viejita linda, que decir de ti si has sido mi otra madre, que siempre tienes los brazos abiertos para protegerme, porque si hoy soy un hombre de bien es gracias a ti, por todo esto muchas gracias. A mi papá aunque no esté con nosotros, por ser un ejemplo de profesional a seguir y por inculcarme el amor al estudio. Agradezco a Bere por preocuparse siempre por mí y por estar siempre presente aun en los momentos difíciles. A mi familia que siempre me ha ayudado, este documento no alcanzaría para mencionarlos a todos, pero quisiera que les llegaran mis más sinceros agradecimientos. A Mileydi por ser mi amor, mi compañera y amiga, a sus padres y familia por acogerme como un miembro más. A todos mis amigos, el hecho de que no los mencione aquí no significa que los he olvidado, al contrario, son tantos que no quisiera ser injusto al olvidar plasmar aquí el nombre de alguno. A todos ustedes muchas gracias.

**DEDICATORIA**

A mami y papi, a mis hermanos, a mis abuelos y en general a toda mi familia.

A mis sobrinos y primos para que les sirva de ejemplo.

Mileydi.

A mami, a mis hermanas, a mama y a toda mi familia.

A mi papá aunque ya no esté.

Ernesto.

## RESUMEN

La facultad 5 de la Universidad de las Ciencias Informáticas (UCI) estudia la posibilidad de utilizar la Realidad Aumentada (RA) con el objetivo de ampliar su perfil e iniciar una nueva línea de investigación que apoye los proyectos de su Polo de Realidad Virtual. Su aplicación exitosa, sobre todo en la educación, ha conducido a la realización de estudios para que la Realidad Aumentada pueda formar parte de los procesos formativos de los estudiantes de la UCI.

Esta investigación responde a la necesidad de localizar los elementos virtuales en una escena aumentada, con respecto a la cámara real que filma la escena, como punto de partida para la elaboración de un sistema de realidad aumentada. Los autores hacen un estudio para conocer cómo ha evolucionado la RA; se presentan los principales frameworks existentes para desarrollar aplicaciones de RA y algunos algoritmos de tracking que se estudiaron. Se exponen también los dispositivos existentes más importantes, utilizados para hacer aplicaciones de RA y los lenguajes, metodologías y herramientas de desarrollo para elaborar la solución. Se propone una solución para localizar los objetos virtuales, en una escena aumentada con respecto a la cámara, que involucra la utilización de ARToolKit como biblioteca de desarrollo de software para aplicaciones de RA. Finalmente se describe la solución propuesta y se implementa para demostrar su validez, cumpliendo así con el objetivo planteado.

## PALABRAS CLAVE

Realidad Aumentada, Realidad Virtual, tracking, localización, elementos virtuales, dispositivo de realidad aumentada.

**ÍNDICE**

## ÍNDICE DE FIGURAS Y TABLAS

## INTRODUCCIÓN

Desde el surgimiento de los ordenadores, la forma predominante de intercambio de información entre estos y las personas ha sido mediante el tecleado texto y la recepción respuestas visualizadas en pantalla. Debido a la necesidad de su uso en las empresas, se comenzó a interactuar con los ordenadores a través de menús y formularios. Con el auge del ordenador personal, se impone la interacción con interfaces gráficas; en lugar de teclear texto, son manipulados objetos visuales, y modificados utilizando representaciones gráficas de estos objetos. Posteriormente surgen nuevas formas de interactuar con los ordenadores: la realidad virtual (RV), el uso del lenguaje natural, la migración de la interacción de las pantallas y el teclado al entorno que nos propone la realidad aumentada (RA).

La RA permite al usuario permanecer en contacto con su entorno de trabajo de forma que su foco de atención no esté en el ordenador, sino en el mundo real, entendido como *mundo real aumentado*. La RA traslada información adicional al mundo real explotando las habilidades visuales y espaciales de los usuarios, en vez de introducirlos por completo en el mundo virtual del ordenador.

La RA está convirtiéndose en una plataforma de entretenimiento educativo para una variada gama de campos de aplicación a nivel mundial, al igual que la RV. Su uso se ha extendido a museos, a la medicina, educación, construcción y diseño. Muchos artistas han comenzado a utilizar esta tecnología en exhibiciones semipermanentes de sus obras en varios escenarios. El uso industrial de la RA ha tenido un incremento considerable en los últimos años. En la rama industrial, a nivel mundial, se han hecho grandes esfuerzos y se ha llevado a cabo un amplio espectro de investigaciones para producir aplicaciones que permitan acelerar los procesos industriales. Son muchas las instituciones que investigan sobre la Realidad Aumentada y la emplean a nivel mundial.

En Cuba una de las instituciones líderes en el desarrollo de la informática es la Universidad de las Ciencias Informáticas (UCI). La facultad 5, con el perfil de trabajo Entornos Virtuales, es la encargada de llevar a cabo los estudios de RV dentro de esta institución. Cada día se están logrando mayores avances con respecto al tema de la RV, aplicada a toda una serie de esferas de la vida cotidiana en el marco social y empresarial. Por ejemplo, se están desarrollando programas de RV para juegos, simuladores de conducción, quirúrgicos y tiro, por solo mencionar algunos.

La facultad 5 de la UCI estudia la posibilidad de utilizar RA para ampliar el campo de su perfil e iniciar una nueva línea de investigación para apoyar a los proyectos de su Polo de Realidad Virtual. El estudio de esta posibilidad se debe al auge que ha tomado en los últimos años la RA a nivel mundial; los buenos resultados que han sido obtenidos por la aplicación exitosa de esta tecnología en varias áreas del conocimiento humano y el surgimiento de nuevas alternativas para el desarrollo de aplicaciones de diversa índole.

En el evento UCIENCIA 2007, el licenciado Lidiexy Alonso y la máster Mayra Durán [Alonso and Durán, 2007] publican un artículo en el cual muestran la aplicación de la Realidad Aumentada dentro del proceso formativo de los estudiantes de la Universidad de las Ciencias Informáticas y las ventajas que esta tecnología puede brindar. Este artículo deja el camino abierto para la utilización de un sistema de Realidad Aumentada (SRA) que permita llevar a cabo lo que en él se expone.

Sin embargo los SRA existentes en el mundo generalmente son privados, requieren del uso de tecnologías avanzadas que no están al alcance de la Universidad o los dispositivos que utilizan reportarían un costo muy alto. Esto obstaculiza la utilización de estos sistemas por parte de la facultad y la conduce a desarrollar sus propias alternativas en el campo de la RA. Lo expresado anteriormente conlleva a plantearse la siguiente interrogante ¿Cómo mantener localizados espacialmente los objetos virtuales en una escena real filmada con una Webcam, apoyándose en técnicas de Realidad Aumentada? Este constituye nuestro **Problema Científico.**

De aquí que el **Objeto de Estudio** fuese la Realidad Aumentada y el **Campo de Acción** la localización de objetos virtuales en SRA. Esta investigación tuvo como **Objetivo** localizar espacialmente los objetos virtuales introducidos en una escena real con técnicas de RA. Para darle cumplimiento al objetivo en pos de resolver el problema científico, se plantearon las siguientes **tareas de investigación**:

- Revisar la bibliografía existente sobre la Realidad Aumentada, para conocer cómo ha evolucionado.
- Analizar los dispositivos usados por los SRA, para seleccionar el más adecuado para la investigación.
- Estudiar los frameworks existentes para los SRA.
- Estudiar los algoritmos de tracking existentes.
- Seleccionar framework a utilizar.
- Analizar formatos para la representación de modelos virtuales 3D.

- Seleccionar lenguajes, metodología y herramientas de desarrollo.
- Elaborar la propuesta de solución.

Para llevar a cabo la investigación se tuvo en cuenta los siguientes métodos científicos:

**Métodos Teóricos**

- Histórico-lógico: Este método permitió realizar la primera parte de la investigación concerniente al análisis bibliográfico del tema, lo cual permitió evaluar la bibliografía para determinar los conceptos necesarios de la temática y así obtener un conocimiento del estado actual en que se encuentra el fenómeno en cuestión.
- Analítico-sintético: Mediante este método se pudo analizar y estudiar el objeto de la investigación, determinando los componentes significativos que forman parte de él. También permitió relacionar sus componentes de manera tal que se pueda ver el funcionamiento del objeto de investigación como un todo al integrar sus partes.
- Modelación: Este método permitió hacer un modelo sustituto del objeto de estudio, creando abstracciones con el objetivo de lograr su mejor entendimiento.

**Métodos Empíricos**

- Revisión de documentos: Este método permitió determinar el estado del arte del objeto de investigación.

El presente trabajo de diploma está estructurado de la siguiente forma: Resumen, Introducción, cuatro capítulos de contenido, Conclusiones, Recomendaciones, Referencias bibliográficas, Bibliografía consultada, Glosario de abreviaturas, Glosario de términos y los Apéndices. A continuación se hace una breve descripción del contenido de cada uno de los capítulos:

- Capítulo 1 “Fundamentación Teórica”: Se hace un análisis de los principales conceptos que se encuentran involucrados con la Realidad Aumentada, se presenta la relación existente entre la Realidad Aumentada y la Realidad Virtual, se muestra el estado del arte relacionado con el tema de la investigación, los algoritmos de localización más utilizados, así como los dispositivos y librerías utilizadas para el desarrollo de aplicaciones de Realidad Aumentada.

- Capítulo 2 “Solución Propuesta”: En este capítulo se muestran las características de la solución que se propone para resolver la problemática planteada en la investigación.
- Capítulo 3 “Descripción de la Solución Propuesta”: Se hace una conceptualización a través del modelo de dominio, un levantamiento de requisitos, se obtienen los casos de uso del sistema y una descripción de estos.
- Capítulo 4 “Diseño e implementación del sistema”: Muestra los diagramas de clases de diseño, los diagramas de secuencia, las descripciones de las clases de diseño, así como el diagrama de componentes y de despliegue de la solución.

# CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA

En este capítulo se exponen conceptos de la Realidad Aumentada, así como su relación con la Realidad Virtual. Son presentadas las características de las herramientas que los autores consideraron más importantes para elaborar aplicaciones de RA. Se hace un estudio de las principales técnicas y dispositivos de visualización y además se abordan los retos fundamentales y las tendencias actuales de la RA en el mundo, en Cuba y en la UCI, lo cual constituyó la base teórica necesaria para llevar a cabo la presente investigación.

## 1.1 Conceptos de Realidad Aumentada.

Al igual que sucede con la Realidad Virtual, existen muchas definiciones formales y clasificaciones acerca de la RA. Algunos definen la RA como un caso especial de la RV; otros argumentan que la RA es un concepto más general y ven a la RV como un caso especial de la RA. A continuación se muestran conceptos que han dado diferentes autores sobre el tema.

"Por una definición operacional de RA, tomamos el término para referirnos a cualquier caso en el cual el otrora ambiente real es *aumentado* por medio de objetos virtuales (gráficos por computadora)" [MILGRAM and KISHINO, 1994].

La Realidad Aumentada "…es una variación de los Entornos Virtuales (EV), o Realidad Virtual (RV) como se conoce más comúnmente. La Realidad Virtual sumerge al usuario dentro de un ambiente sintético (generado por la computadora). Mientras está inmerso, el usuario no puede ver el mundo real alrededor de él. En contraste, la Realidad Aumentada le permite al usuario ver el mundo real, con objetos virtuales superpuestos sobre el mundo real, o compuestos con él. De ahí que la Realidad Aumentada actúe como complemento de la realidad, tanto mejor que la Realidad Virtual que reemplaza completamente la realidad" [AZUMA, 1997].

Según lo expresado por Azuma en 2001"…la idea fundamental de la Realidad Aumentada es sobreponer gráficas, audio y otras mejoras a los sentidos en tiempo real, sobre un ambiente real y modificar las gráficas de forma tal que se acomoden al movimiento de la cabeza del usuario, de modo que las gráficas siempre estén en la perspectiva correcta" [Azuma *et al*, 2001].

Otra definición de RA la brinda Jong Seung Park cuando expresa que esta es"… una combinación de la escena real vista por el usuario y una escena virtual generada por la computadora que aumenta la escena real" [Park, 2005].

Luego de analizar los conceptos de Realidad Aumentada enunciados por diferentes autores en varios momentos, y a medida que se ha ido profundizando en este campo de investigación, puede identificarse que todos coinciden al señalar el aumento de una escena real con objetos virtuales 3D de tal forma que éstos ofrezcan información complementaria a la realidad. En lo adelante los autores de la investigación adoptarán por concepto de RA el enunciado por Azuma en 1997, por ser el más completo de los analizados.

### 1.1.1 Realidad Aumentada vs. Realidad Virtual.

La Realidad Aumentada y la Realidad Virtual están estrechamente relacionadas debido a que tienen elementos en común. La Realidad Virtual es un entorno tridimensional, interactivo y generado por computadoras en el cual una persona está inmersa completamente. Un entorno virtual se puede ver como una escena tridimensional generada por computadoras, que además requiere alto rendimiento en cuanto al procesamiento de los gráficos por computadoras para proveer un nivel de realismo tal, que pueda simular la realidad de forma efectiva. La inmersión total del usuario dentro de un mundo completamente virtual conduce a que este se divorcie completamente del ambiente real. El mundo virtual es interactivo y el usuario requiere una respuesta en tiempo real del sistema para ser capaz de interactuar con él. Una característica fundamental de la Realidad Virtual es que lo visual, y en algunos sistemas lo auditivo y la percepción, están bajo el control del sistema.

Por otra parte en la Realidad Aumentada, los objetos virtuales son anexados a la vista real para crear una presentación aumentada. En algunas aplicaciones, pudiera ser deseable utilizar tanto como sea posible el mundo real, en vez de crear una nueva escena virtual utilizando gráficos generados por computadora. Por ejemplo, en las aplicaciones médicas, el terapeuta debe ver al paciente para llevar a cabo la cirugía, en tele-robótica el operador debe ver la escena remota para llevar a cabo sus tareas. La RA puede mantener un alto nivel de detalle debido a los matices realistas que se encuentran en el mundo real. Aunque la RA atenúa las náuseas que provocan los simuladores desarrollados en RV también podría estar en desventaja cuando se utiliza un HMD para ver el mundo virtual, debido al error de coincidencia de los sentidos que puede suceder dentro del entorno de la pantalla, produciendo vértigo y mareo.

A continuación se muestran algunas semejanzas y diferencias entre la RA y la RV que evidencian la relación existente entre ambas:

**Semejanzas**

- Se emplean modelos 3D.
- Requieren interactividad y respuesta en tiempo real.
- Lo esencial es la percepción visual; los otros sentidos pueden estar bajo control del sistema.

**Diferencias**

- En la RV el usuario ve el mundo completamente artificial. La computadora genera todo el ambiente virtual y reemplaza el mundo real.
- En la RA el usuario ve el mundo real aumentado. La computadora aumenta el sentido de percepción del mundo real.

### 1.1.2 Continuo de Realidad Mezclada de Milgram.

En 1994 Milgram y Kishino describen una taxonomía donde identifican cómo están relacionadas la Realidad Virtual y la Realidad Aumentada. También define el continuo Realidad-Virtualidad como muestra la figura 1, el cual llamó Continuo de Realidad Mezclada.

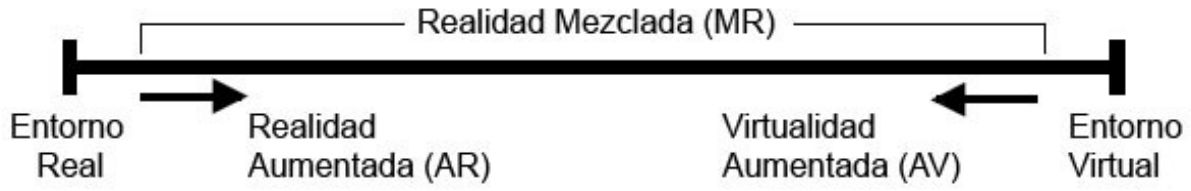



*Figura 1. Continuo Realidad – Virtualidad de Milgram.*

El mundo (entorno) real y los entornos virtuales puros son los extremos del continuo entrelazados por una región media llamada Realidad Mezclada. La Realidad Aumentada se acerca más al extremo del mundo real generando objetos complejos de alta calidad visual y la Virtualidad Aumentada es un término creado por Milgram para identificar sistemas que son sintéticos, pero que adicionan imágenes

y videos del mundo real, como por ejemplo, texturas basadas en fotos y videos capturados de la realidad que se incorporan a los objetos que existen en el entorno virtual.

### 1.1.3 Retos actuales de los sistemas de Realidad Aumentada.

"Mientras muchos investigadores ampliaron la definición de la RA más allá de nuestra visión, nosotros definimos un sistema de RA como aquel que tiene las siguientes propiedades:

- Combina los objetos reales y virtuales en un ambiente real.
- Se ejecuta interactivamente y en tiempo real.
- Registra (alinea) los objetos reales y virtuales unos con otros." [Azuma, 2001]

El mayor reto que enfrentan hoy los sistemas de realidad aumentada, es dar una solución precisa a la problemática de cómo combinar el mundo real con el virtual dentro de un solo ambiente aumentado, lo cual requiere un registro consistente del mundo virtual junto con el mundo real para mantener, en el usuario, la ilusión de que los objetos virtuales son una parte verdadera del mundo real.

La naturaleza de esta problemática de la RA se reduce a lograr una correspondencia precisa entre el sistema de coordenadas local de cada objeto virtual, con el sistema de coordenadas global del mundo virtual, que a su vez debe tener correspondencia con el sistema de coordenadas del mundo real, como se ilustra en la figura 2.

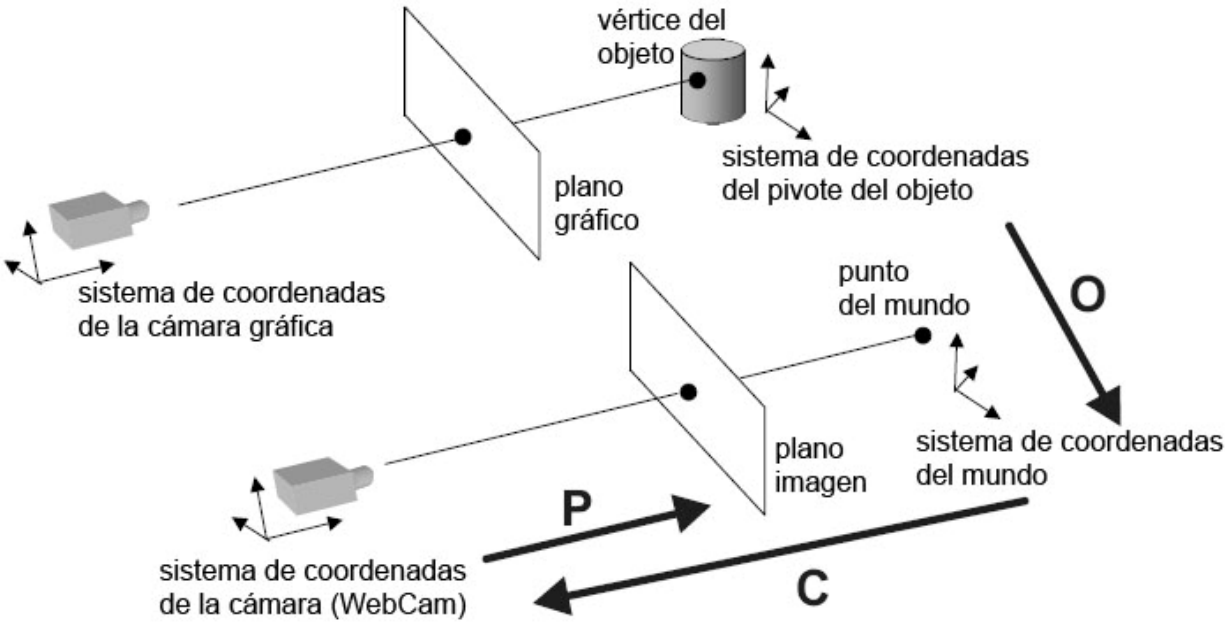



*Figura 2. Sistemas de coordenadas en la Realidad Aumentada.* [Alonso and Duran, 2007]

"Los requisitos para crear un entorno de Realidad Aumentada con alta fidelidad están dados por las relaciones entre los sistemas de coordenadas del mundo y los sistemas de coordenadas del entorno virtual que están implícitas en la figura anterior. Estas relaciones están representadas por las transformaciones **O (**objeto – mundo**)**, **C (**mundo – cámara**)**, **P (**cámara – plano de imagen**)**. La transformación **O**, significa la posición y orientación del objeto virtual con respecto al sistema de coordenadas del mundo real. La transformación **C**, representa la postura o posición de la cámara que captura la escena del mundo real y por otra parte la transformación **P**, constituye la proyección de la escena 3D del mundo real visualizado en una imagen 2D en el plano de proyección" [Alonso and Duran, 2007]. Se puede tener una información más amplia en [Vallino, 1998].

A partir de lo expresado por Vallino, se considera que se tendría una representación inequívoca, debido a la combinación perfecta entre los objetos reales y los generados por la computadora, si se hace coincidir de forma exacta las coordenadas de la cámara real con la cámara gráfica ilustradas en la figura 2. Para lograr esta combinación entre objetos reales y virtuales se ha llevado a cabo un gran número de trabajos e investigaciones que denotan las tendencias actuales.

## 1.2 Trabajos relacionados y tendencias actuales.

Los comienzos de la Realidad Aumentada, como ésta se define, se remontan a los trabajos hechos por Sutherland en la década de 1960, el cual utilizó un HMD con pantalla transparente para presentar gráficos tridimensionales. Sin embargo no fue hasta la pasada década que hubo suficiente trabajo como para referirse a la Realidad Aumentada como un campo de investigación en sí mismo.

En el año 1997, el investigador Ronald Azuma publica un artículo llamado "A Survey of Augmented Reality", en el cual hace un estudio detallado del estado del arte de la RA. En este artículo describe el trabajo llevado a cabo en diferentes lugares y explica los conflictos y problemas encontrados a la hora de construir sistemas de RA. Además resume los esfuerzos y estrategias tomadas para superar estos problemas y especula en cuanto a las futuras direcciones que merecen ser exploradas cuando expresa "Este artículo no presenta nuevos resultados de investigación. Su contribución se debe a la consolidación de la información existente que proviene de muchas fuentes y a la publicación de una bibliografía extensiva de artículos en este campo" [Azuma, 1997].

En 1998 G. Trogemann, B. Graffmann y J. Piesk [Trogemann *et al*, 1998] publican un trabajo en el cual describen cómo reconstruir los parámetros de la cámara basados en imágenes para la postproducción de Realidad Mezclada (RM), teniendo como resultado el seguimiento de puntos de referencia ubicados en la escena real.

Debido al rápido avance que va llevando la RA, Ronald Azuma y otros autores publican un estudio acerca del estado el arte de la RA en 2001. Este estudio no es más que un complemento del artículo anteriormente mencionado y que fue publicado en 1997. En ese estudio los autores definen de una forma más completa las características que deben tener los sistemas de RA.

Una de las características más importantes de un sistema de RA es la estimación de la posición de la cámara o de los objetos en el mundo real donde tiene lugar el aumento. En el año 2002, con la llegada de procesadores más poderosos se facilita aún más la estimación de la posición de objetos basada en marcadores. Aunque en ese mismo año Y. Genc describe un sistema general para rastrear la posición y la orientación de la cámara solamente observando la escena sin ningún marcador visual.

En el Simposio Internacional para la Realidad Aumentada y Mezclada (ISMAR por sus siglas en inglés), en el año 2002, Jong Weon Lee, Suya You y Ulrich Neumann presentan un artículo en el que describen cómo determinar la posición de una cámara en movimiento a través de imágenes 2D del

mundo utilizando visión por computadoras. En ese mismo año y evento Bernd Schwald, Helmut Seibert y Tanja Weller presentan una estrategia para usar pantallas semitransparentes como un tipo de ventana dentro de un paciente en el contexto de las aplicaciones de RA para la medicina, en él hacen una combinación de rastreo óptico y electromagnético.

En el curso de doctorado en "Metrología con sensores de array" impartido en la universidad de Alcalá a partir del año 2003 el profesor Juan Carlos García García [García, 2003] propone un método para recuperar el posicionamiento 3D (2D) y la orientación mediante una sola imagen de una marca artificial. En este mismo año Andrew J. Davison [Davison, 2003] presenta un framework para la localización de una cámara por medio del mapeo de un conjunto de escasas características naturales usando modelación de movimiento apoyado en la visión por computadoras.

En el año 2004 Bernd Schwald, Helmut Seibert y Michael Schnaider [Schwald *et al*, 2004] publican un artículo donde presentan un concepto para la composición de sistemas de rastreo en el contexto de las aplicaciones de Realidad Aumentada y Realidad Virtual. El concepto es definido considerando como ejemplo un sistema médico de RA que combina las tecnologías de rastreo óptico y electromecánico mejorado. En este mismo año Hanhoon Park y Jong-Il Park introducen un nuevo sistema de seguimiento basado en marcadores invisibles dibujados con un lápiz fluorescente infrarrojo. Para más información consultar [Park and Park, 2004].

En la Conferencia Internacional de Adquisición de Información (ICIA por sus siglas en inglés), del año 2005, Yangbin Chen, Yimin Chen, Zhenpeng Yuan, Yunhua Zhang describen [Chen *et al*, 2005] una arquitectura para un clúster de PC que utiliza RA. Proponen un amplio rango de métodos de registro basados en visión estéreo y seguimiento basado en visión multi-estéreo, para resolver los problemas de registro más importantes.

Como parte de la Conferencia Internacional de Realidad Artificial y talleres de tele-existencia (ICAT por sus siglas en inglés) en el año 2006 Yongtian Wang, Yu Li, Jing Chen, Wenze Hu, Xiaojun Zang [Wang *et al*, 2006] presentan un algoritmo para el registro de la posición y orientación de la cámara en tiempo real usando características naturales para las aplicaciones de RA. El sistema usa una sola cámara para el seguimiento visual de las características naturales extraídas de la escena real. Pascal Fua y Vincent Lepetit en 2007 discuten algunas de las estrategias más prometedoras para hacer tracking donde analizan sus fortalezas y debilidades, presentando la visión por computadora como una de las alternativas más poderosas, y también proponen la combinación de varios métodos para hacer tracking, aprovechando las potencialidades individuales de cada método.

En Cuba existen instituciones que incursionan en el trabajo con la RV como es el caso del Centro de Investigación y Desarrollo de Simuladores (SIMPRO) y la UCI. También existen instituciones como el Centro de Aplicaciones de Tecnología Avanzada (CENATAV) y los proyectos del Grupo de Procesamiento de Imágenes de la UCI (GPI), dedicadas al reconocimiento de patrones y análisis de imágenes. Aunque los trabajos desarrollados por estas instituciones y proyectos, constituyen elementos necesarios para hacer aplicaciones de RA, hasta la fecha de la investigación no se tiene conocimiento sobre trabajos relacionados con la RA, ni de aplicaciones de RA de origen cubano que hayan sido utilizadas con algún propósito.

## 1.3 Tecnologías y dispositivos de visualización.

Para lograr la representación del aumento del mundo real y sobreponerse a los retos que se plantea la RA es necesario disponer de dispositivos capaces de llevar a cabo tal mezcla. Estos dispositivos han ido avanzando tecnológicamente a medida que los investigadores se han adentrado en el campo de la RA. En la taxonomía que hace Milgram en 1994 sobre la realidad mezclada, también describe varias clases de dispositivos usados para la RA, los cuales pueden ser representados en un continuo.

Azuma en su artículo “A Survey of Augmented Reality” en 1997 menciona dos posibles alternativas a la hora de construir sistemas de RA, las tecnologías ópticas y basadas en video, cada una de estas tecnologías incluye sus respectivos dispositivos utilizados para llevar a cabo la mezcla del mundo real con el virtual. Además hace una comparación para determinar cuál tecnología y dispositivo seleccionar de acuerdo al área de aplicación que tenga la solución a desarrollar. Aunque los dispositivos utilizados para desarrollar aplicaciones de RA han alcanzado mayor potencia y mejor acabado en sus diseños, básicamente la esencia de su estructura no ha cambiado. A continuación se hace un análisis de cada uno de ellos.

### 1.3.1 Head Mounted Display (HMD) óptico transparente.

Este es un dispositivo que combina el mundo real con el virtual ofreciendo una vista directa del mundo real con objetos virtuales superpuestos por medio de tecnologías ópticas. Este dispositivo trabaja situando mezcladores ópticos frente a los ojos del usuario. Los mezcladores son parcialmente transparentes, de forma tal que el usuario puede mirar a través de ellos para ver el mundo real; también son parcialmente reflexivos, permitiéndole al usuario ver las imágenes virtuales reflejadas desde una especie de proyectores. En la figura 3 se muestra un diagrama conceptual de un HMD óptico transparente.

Los mezcladores actúan como espejos semitransparentes que solo dejan penetrar una parte de la luz proveniente del mundo real y además pueden reflejar una porción de la luz que parte de los proyectores hacia los ojos del usuario. Cuando se le retira el suministro de energía al HMD óptico transparente, los mezcladores actúan como un par de gafas solares.

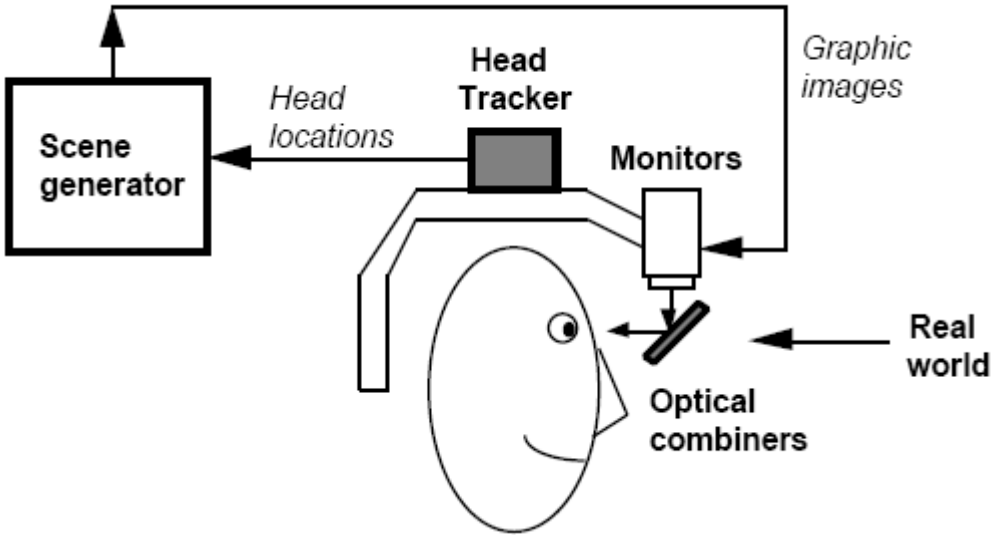



*Figura 3. Diagrama conceptual de un HMD óptico transparente.* [Azuma, 1997]

*Figura 4. HMDs ópticos transparentes.*

### 1.3.2 Head Mounted Display (HMD) con visión a través de video.

Este tipo de dispositivo trabaja combinando un HMD donde el usuario no ve el mundo real directamente con una o dos cámaras montadas en la cabeza. Las cámaras de video ofrecen la vista del mundo real al usuario, es decir, el usuario ve el mundo real de forma indirecta. El flujo de video

proveniente de las cámaras es mezclado con las imágenes gráficas creadas por el generador de la escena. El resultado es enviado a los monitores que están situados frente a los ojos del usuario como se muestra en la figura 5.

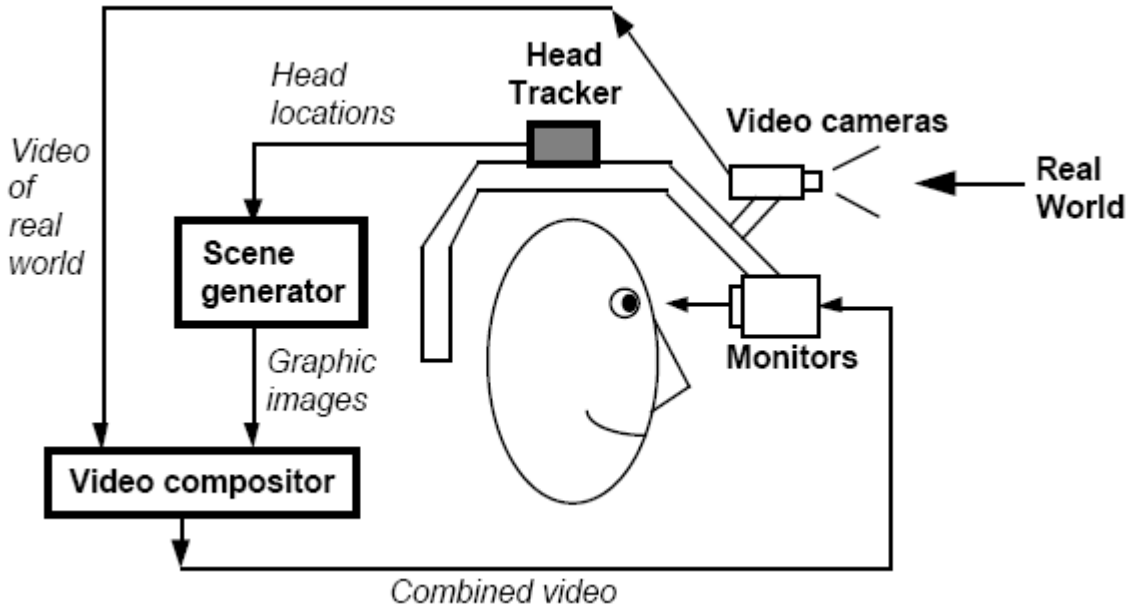



*Figura 5. Diagrama conceptual de un HMD con visión a través de video.* [Azuma, 1997].

Una razón determinante por la cual no se seleccionó ninguno de los dispositivos anteriores, es el precio al que se venden en el mercado internacional actualmente (por encima de los $1000 USD). En los últimos años no se encuentran productos HMD de bajo costo, el mercado de estos dispositivos es muy pequeño para vender esta tecnología. Se pueden encontrar dispositivos de bajo costo de segunda mano con revendedores de equipos de realidad virtual o en foros específicos.

### 1.3.3 Dispositivos alternativos basados en monitores.

Los dispositivos alternativos basados en monitores son una configuración de sistemas de RA que surge como alternativa a los HMD con visión a través de video. Esta configuración consiste en una o dos cámaras de video que captan las escenas del mundo real, un generador de escena, un mezclador y el monitor para visualizar la mezcla del mundo real con el virtual. Las cámaras pueden ser estáticas o móviles, en el caso de que sean móviles éstas pueden ir adjuntas a un robot conociéndose en todo momento su posición y en el caso de que sean estáticas se predefine su posición. Las imágenes creadas por el generador de la escena son combinadas de la misma forma que en el caso del HMD

con visión a través de video y mostradas en el monitor que se encuentra frente al usuario. Esto se aprecia gráficamente en la figura 6.

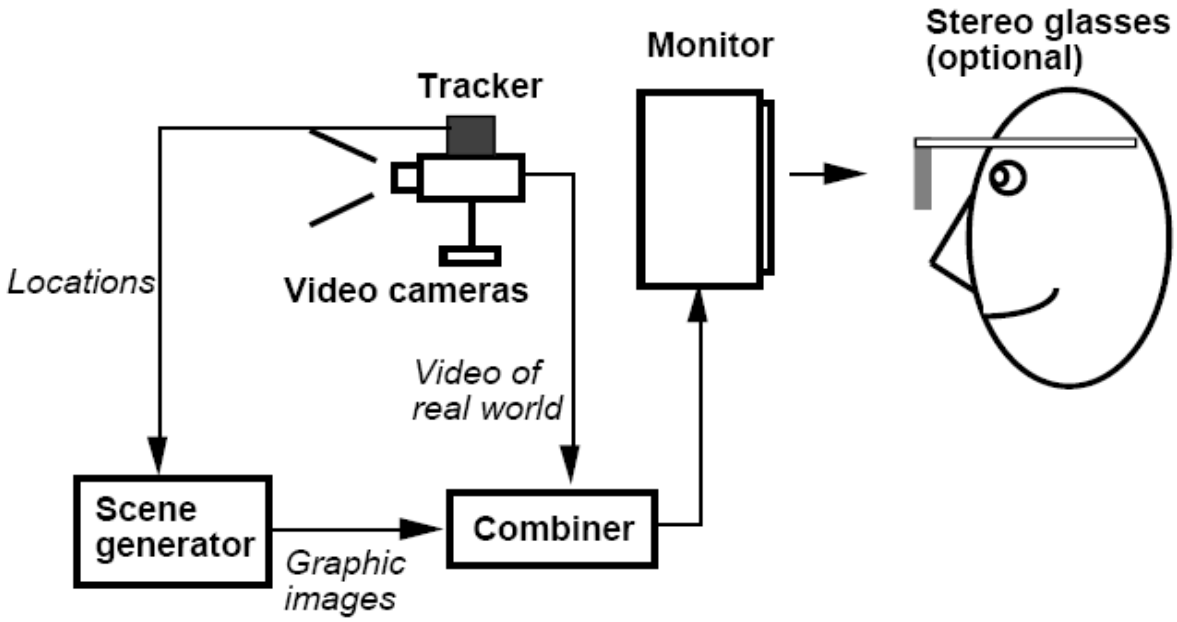



*Figura 6. Diagrama conceptual de un dispositivo basado en monitor.* [Azuma, 1997]

### 1.4 Frameworks para desarrollar aplicaciones de RA.

Existen dos variantes a seguir para elaborar aplicaciones de Realidad Aumentada, una es realizarlas totalmente partiendo del inicio, mediante la implementación de cada una de las herramientas que se necesitan para desarrollar la aplicación. La otra variante es utilizar herramientas existentes que agilizan el proceso de desarrollo. En esta investigación se asumió la segunda variante para obtener un resultado acorde con el objetivo. A continuación se analizan algunas de las herramientas más importantes para el desarrollo de aplicaciones de Realidad Aumentada.

#### 1.4.1 Designers' Augmented Reality Toolkit (DART).

La Herramienta de Realidad Aumentada para Diseñadores (DART por sus siglas en inglés) está diseñada para dar soporte a la construcción rápida de prototipos de aplicaciones de RA que usan dispositivos de visión a través de la pantalla (dispositivos transparentes o de mezcla de video) para superponer gráficos y audio en la vista del mundo del usuario.

DART está constituido por una colección de extensiones del entorno de desarrollo de multimedia Macromedia Director, el estándar de-facto para la creación de contenido multimedia. Su diseño está pensado para apoyar la potencia de Director y asume que los desarrolladores están familiarizados con esta aplicación. El seguimiento de los marcadores en el video en tiempo real se hace a través de la ARToolKit.

Inicialmente el desarrollo de la herramienta DART estuvo motivado por el interés de sus autores en las experiencias para la educación informal, arte digital y entretenimiento. Sin embargo, puede ser usado para crear aplicaciones de RA en cualquier dominio (en la industria, en lo militar y en aplicaciones científicas) y es especialmente útil en la exploración. Su diseño hace que pueda ser usado por todo el que lo desee, tanto diseñadores de oficio como aficionados, artistas e investigadores.

**Requerimientos**

- **Hardware**
  - Windows 2000 o XP, o Mac OSX, No soportado en Linux.
  - Procesador: 1.0 GHz o mejor.
  - Memoria: Al menos 512 MB de RAM.
  - Tarjeta de gráficos de gama alta.
- **Software**
  - Macromedia Director 8.5 o más reciente.
  - Direct X 9.0b Runtime.
  - Shockwave Player – Incluye la versión más reciente de Havok Physics para la física de DART. Havok no es distribuido dentro de Director.

**Licencia**

Está liberado bajo una licencia pública, que impone una única restricción en la que se expresa que no se puede vender DART en sí mismo o con pequeños cambios, pero se pueden crear todas las aplicaciones deseadas con cualquier propósito. Este modelo de licencia, aunque parece factible, no lo es, porque para poder utilizar DART en la elaboración de cualquier tipo de aplicación es necesario tener las herramientas propietarias sobre las que él está sustentado.

Los autores de esta investigación decidieron no utilizar esta herramienta, debido a los requerimientos de software y hardware que exige, el modelo de licencia empleado por las herramientas sobre las que está sustentada, además de que no es multiplataforma.

### 1.4.2 Distributed Wearable Augmented Reality Framework (DWARF).

Desde el punto de vista de la arquitectura de software, para los creadores de la Plataforma de Realidad Aumentada Distribuida para Dispositivos Portátiles (DWARF por sus siglas en inglés), los sistemas de RA pueden ser divididos en un conjunto de componentes, de forma tal que cada uno contribuye con el sistema completo con una funcionalidad dedicada. Los autores de esta herramienta se centraron en encontrar una solución donde fuera posible la construcción rápida de prototipos de aplicaciones con varios investigadores en diferentes computadoras. Originalmente fue acomodado a una infraestructura heterogénea consistiendo en varias PC basadas en arquitectura Intel; laptops con Windows y Linux; estaciones de trabajo y laptops Apple; PDAs iPAQ. Además, adicionalmente debería ser posible incorporar componentes de terceros como *trackers* externos, etc. La solución a estos requerimientos fue un framework basado en componentes para sistemas punto a punto, llamado DWARF.

**Arquitectura**

DWARF está basado en el concepto de colaboración de servicios distribuidos. Los servicios son independientes y exponen sus requerimientos y ofertas (estos requerimientos y ofertas son llamados necesidades y habilidades respectivamente) con la ayuda de administradores de servicios. En cada nodo de la red hay un administrador de servicios; no hay un componente central [Bruegge and Klinker, 2003]. Para llevar a cabo esta arquitectura, los autores de DWARF utilizan el estándar CORBA (Common Object Request Broker Arquitecture), que establece una plataforma de desarrollo de sistemas distribuidos facilitando la invocación a métodos remotos bajo un paradigma orientado a objetos.

Debido al uso de la infraestructura de comunicación CORBA para su construcción, este framework es independiente de sistema operativo y de lenguaje de programación, porque generalmente para cada plataforma existe una implementación para CORBA.

A pesar de que este framework ofrece muchas ventajas, mencionadas anteriormente, no se ajusta a los propósitos de la investigación, porque originalmente está construido para sistemas distribuidos y

para el uso de dispositivos móviles. Además, el servicio encargado del subsistema de tracking (ServiceARTTracker) toma los datos del sistema DTrack, éste a su vez necesita de 2 a 16 cámaras trabajando para capturar la luz emitida por marcadores luminosos a frecuencias cercanas al infrarrojo, añadiendo un costo excesivo. Los servicios de detección y configuración de marcadores están basados en **ARToolKit,** para más información ver [München, 2005] y [GmbH, 2005].

### 1.4.3 StudierStube

El punto de partida para la construcción del sistema Studierstube fue la creencia en que la RA, la prima menos intrusiva de la RV, tiene mejores oportunidades para convertirse en una interfaz de usuario factible para aplicaciones que requieren de manipulación de información tridimensional compleja como una rutina diaria [Schmalstieg *et al*, 2002]. En esencia el propósito perseguido a la hora de concebir este proyecto fue la búsqueda de una metáfora de interfaz de usuario tridimensional tan poderosa como la metáfora del escritorio para 2D.

En el corazón del sistema Studierstube, la RA colaborativa es usada para embeber imágenes generadas por computadora en el entorno de trabajo real. El sistema Studierstube fue uno de los primeros sistemas de Realidad Aumentada colaborativa. Una de sus funcionalidades es que múltiples usuarios se congregan en una habitación y pueden experimentar un espacio virtual compartido poblado de datos tridimensionales.

En 1996, Studierstube era el framework líder para el desarrollo de aplicaciones de Realidad Aumentada móviles, colaborativas y ubicuas. Las tecnologías de dispositivos de pantalla utilizadas por Studierstube son HMDs con transparencia o pantallas de proyección (usando proyectores) para combinar los gráficos con la vista del mundo real del usuario. La interacción con los elementos virtuales se lleva a cabo a través de un panel de interacción personal (PIP por las siglas en inglés de Personal Interaction Panel). El PIP es una interfaz usada a dos manos para controlar las aplicaciones de Studierstube, está compuesto por dos sostenes de mano ligeros, un lapicero y un panel, los dos equipados con trackers físicos. El reconocimiento de marcadores se hace a través de *stbTracker*, una evolución de la biblioteca *ARToolKitPlus,* y ésta es, a su vez, una extensión de *ARToolKit*. Los requerimientos de hardware específicos para este framework y su licencia privativa constituyeron las razones principales que limitaron su utilización.

### 1.4.4 Augmented Reality Toolkit (ARToolKit).

ARToolKit es una biblioteca de software utilizada para la construcción de aplicaciones de RA. Utiliza algoritmos de visión por computadoras para resolver el problema del tracking de los marcadores. Las bibliotecas de ARToolKit calculan la posición real de los marcadores físicos y su orientación relativa a la cámara en tiempo real para hacer tracking. Esto permite el desarrollo fácil de un amplio rango de aplicaciones de RA. La biblioteca ARToolKit presenta un conjunto de características que la hacen un fuerte adversario frente a las demás analizadas anteriormente, destacando que es utilizada por estas últimas para realizar las labores de tracking. A continuación se exponen las principales funcionalidades y facilidades que ofrece ARToolKit para obtener una mejor comprensión de la misma:

- Tracking de orientación/posición de una sola cámara.
- Posee la habilidad de utilizar cualquier marcador o patrón. (Siempre que sea cuadrado).
- Código de calibración de cámara muy fácil.
- Suficientemente rápida para ser usada en aplicaciones de RA.
- Superpone objetos virtuales 3D en marcadores reales (basado en algoritmos de visión por computadoras).
- Es multiplataforma, soportada en distribuciones SGI IRIX, Linux, MacOS y Windows OS.
- Soporta bibliotecas de video multiplataforma:
    - Soporta múltiples fuentes de entrada (USB, FireWire, Tarjeta de captura).
    - Soporta múltiples formatos (RGB/YUV420P, YUV).
    - Soporta tracking para múltiples cámaras.
    - Interfaz de inicialización GUI (Interfaz Gráfica de Usuarios).
- Tracking de marcadores en 6 DOF (grados de libertad) rápido y barato (Detección de planos en tiempo real).
- Estrategia de patrones de marcadores extensible (El número de marcadores es directamente proporcional a la eficiencia).
- Utiliza GLUT para el aspecto de la manipulación de ventanas y eventos, además para la biblioteca de video dependiente de hardware y el API estándar en cada plataforma.
- Utiliza OpenGL para las tareas de dibujado (rendering).
- Ofrece soporte 3D vía VRML.
- Interfaz de programación de aplicaciones (API) simple y modular, escrita en lenguaje C.
- Soporta otros lenguajes (Java, Matlab).
- Ofrece un conjunto completo de utilidades y ejemplos.

- Librería liberada bajo licencia GPL distribuida con el código fuente completo para uso no comercial.
- Tiene una comunidad de desarrollo muy activa.

Para más información ver [HITLab, 2006].

## 1.5 Estrategias de tracking.

El tracking (rastreo) de objetos en 3D es el proceso mediante el cual se trata de calcular la posición del objeto (matriz de transformación) en cada fotograma (frame). Según lo expresado por Borro y Lardizábal, cuando se hace tracking "… se consigue disponer permanentemente de un sistema de referencia virtual alineado con el sistema de referencia del objeto real, y que por lo tanto realiza la misma transformación sólido rígida que este último (traslación y rotación) [Borro and Lardizábal, 2005].

Existen varias estrategias para llevar a cabo el proceso de tracking. Fua and Lepetit [Fua and Lepetit, 2007] discuten algunas de las estrategias más prometedoras, sus fortalezas y debilidades, haciendo un análisis profundo del tracking basado en marcadores, tracking usando características naturales y tracking por detección. A continuación se tratan de manera general los métodos discutidos por los autores citados para conocer en que se basa cada uno de ellos.

### 1.5.1 Tracking basado en marcadores o fiduciarios.

El tracking 3D basado en visión por computadoras puede ser descompuesto en dos pasos principales; el primero es el procesamiento de imagen, para extraer alguna información de las imágenes, y el otro la estimación de la posición en sí mismo. La adición de marcadores en la escena tributa considerablemente a estos dos pasos, porque se convierten en características de las imágenes fáciles de extraer y proveen medidas fáciles de explotar y confiables para la estimación de la posición. Los marcadores se pueden encontrar en dos grupos:

**Marcadores en forma de puntos.**

Los marcadores en forma de puntos han sido usados por fotogrametristas por muchos años. Se diseñan de forma tal que puedan ser detectados fácilmente e identificados con métodos de soluciones rápidas. Presentan una apariencia de patrones circulares y son relativamente invariantes a la distorsión perspectiva. Se asume que su posición 3D en el sistema de coordenadas del mundo real es conocida con precisión y esto se logra de forma manual con un láser o con un algoritmo llamado Estructura a

partir del Movimiento (Structure-From-Motion nombre del algoritmo en inglés). Para facilitar su identificación se diseñan en forma de patrones geométricamente distintos y una vez que son identificados en una imagen, proveen un conjunto de correspondencias que pueden ser utilizadas para recuperar la posición de la cámara.

Este tipo de marcador necesita ser distribuido en varios lugares de la escena y sus posiciones necesitan ser medidas con precisión para poder extraer la posición de la cámara, lo cual constituye una desventaja del mismo.

**Marcadores extendidos.**

[Koller *et al*, 1997] Introduce los marcadores cuadrados en blanco y negro, los cuales contienen pequeños cuadrados rojos para su identificación. Los marcadores rectangulares planos también son usados por otros autores mencionados por Pascal Fua y se demuestra que con un simple marcador es suficiente para estimar la posición de la cámara. Durante el proceso de detección de los marcadores, la estimación de la posición se ejecuta en tiempo real, de ahí que puede ser aplicado en cada imagen dentro del flujo de video.

El sistema de tracking 3D para los marcadores extendidos no requiere de ningún tipo de inicialización manual, al contrario de los marcadores en forma de puntos. “Los marcadores rectangulares se han hecho muy populares porque conducen a una solución robusta, de bajo costo para el tracking 3D y además existe una librería públicamente disponible llamada ARToolKit...” [Fua and Lepetit, 2007] que utiliza este tipo de marcadores, por estas razones se decidió utilizar los marcadores extendidos o rectangulares.

### 1.5.2 Tracking usando características naturales.

El uso de marcadores simplifica la tarea del tracking, aunque a veces es imposible llevar a cabo esta técnica sobre todo en entornos abiertos. Mientras sea posible es mucho mejor utilizar las características naturalmente presente en las imágenes. Para aplicaciones de RM esto no constituye un problema porque se puede disponer de modelos 3D de las escenas del mundo real. Existen dos familias de estrategias dependiendo de las características de las imágenes que se utilizan.

**Métodos basados en aristas.**

Los primeros métodos para hacer tracking estaban basados en aristas en su mayoría, porque son eficientes computacionalmente, relativamente fáciles de implementar y estables frente a cambios de iluminación, incluso para materiales que reflejan la luz. Esto no necesariamente se cumple para los métodos que consideran el uso de los píxeles internos. La estrategia más popular dentro de estos métodos es la búsqueda de gradientes fuertes en la imagen, alrededor de una primera estimación de la posición del objeto, sin sacar explícitamente los contornos, lo cual es rápido y general.

**Métodos basados en texturas.**

Este método suele ser utilizado cuando un objeto 3D real está suficientemente texturizado. La información puede ser derivada a partir del flujo óptico, de la correspondencia de plantillas (template matching) o correspondencias de puntos de interés (interest-point correspondences). Sin embargo, la técnica de correspondencias de puntos de interés probablemente es la más efectiva para aplicaciones de RM, porque depende de establecer una correspondencia entre las características locales de la imagen. Dadas tales correspondencias, la posición de la cámara puede ser estimada por cuadrados mínimos (least-square minimization), o incluso mejor, por estimación robusta. “Estos métodos son relativamente insensibles a oclusiones parciales o a los errores de correspondencias y, al contrario de los métodos basados en aristas, no llegan a confundirse por desórdenes de fondo y explotan más de la información de la imagen” [Fua and Lepetit, 2007].

### 1.5.3 Tracking por detección.

La detección tiene una larga historia en la visión por computadora, ésta se ha apoyado en la detección en 2D incluso para objetos 3D. Sin embargo ha habido un interés sostenido en la detección simultánea de objetos y la estimación de su posición 3D. Según [Lowe, 1991] y [Jurie, 1998] los primeros métodos fueron basados en aristas, pero los métodos basados en la correspondencia de puntos característicos (feature point matching) se han vuelto populares, a partir de que las regiones invariantes locales en la imagen funcionan mejor para este propósito.

La técnica basada en puntos característicos se acerca a ser la más robusta frente a los cambios de escalado, punto de visión e iluminación, así como a las oclusiones parciales. Operan bajo el siguiente principio: durante una etapa de entrenamiento, se construye una base de datos de puntos de interés pertenecientes al objeto y de los cuales se puede calcular la posición que ocupan en su superficie. En

tiempo de ejecución, los puntos característicos son primeramente extraídos de cada una de las imágenes de forma individual y luego se busca la correspondencia contra la base de datos. La posición del objeto puede ser estimada a partir de esta correspondencia. Tomando en cuenta lo expresado por [Fua and Lepetit, 2007] para estimar la correspondencia, es muy conveniente utilizar los algoritmos de tipo "RANdom SAmple Consensus" (RANSAC) o la transformada de Hough debido a que eliminan falsas correspondencias evadiendo problemas combinatorios.

## 1.6 Formatos de ficheros.

En este epígrafe se tratan los formatos en los que se suele representar los objetos virtuales 3D utilizados para aumentar la realidad en aplicaciones de RA.

### 1.6.1 Lenguaje de modelado de realidad virtual (VRML).

El lenguaje de Modelado de Realidad Virtual (VRML por sus siglas en inglés), surgió gracias a Mark Pesce. VRML es un estándar abierto de Realidad Virtual para Internet elaborado por el consorcio Web3D. El VRML no es un lenguaje de programación como pudiera parecer a simple vista sino, como su nombre sugiere, se trata de un lenguaje de modelado de mundos virtuales, consiste en un lenguaje textual que permite describir objetos tridimensionales y su comportamiento dentro de entornos animados. Desde sus inicios ha estado muy ligado a la tecnología basada en Internet, y se puede emplear en conjunto con otros lenguajes como los son Java, JavaScript y HTML para desarrollar aplicaciones que se montan en páginas Web.

VRML contiene una de las semánticas más comunes que se pueden encontrar en las aplicaciones de 3D de hoy en día, tales como jerarquía de objetos, luces, animación, niebla, materiales y mapeo de textura. VRML es capaz de representar objetos estáticos y animados, y también puede tener hipervínculos a otros medios como sonidos, vídeo e imágenes. Algunas de sus ventajas más significativas son:

- VRML es un estándar internacional respaldado por el Consorcio web 3D.
- Cuenta con especificaciones del dominio público por lo que es gratuito, al igual que la mayoría de los visualizadores de VRML disponibles en el mercado.
- Es un lenguaje sencillo y claro que es soportado por la mayoría de las herramientas desarrollo de ambientes virtuales, así como por los visualizadores de Realidad Virtual.

- Permite el control absoluto del mundo virtual, tanto en el desarrollo cómo durante la ejecución del mismo.
- Cuenta con una integración total a tecnologías basadas en Internet cómo HTML, Java, JavaScript, etc.
- Permite reproducir dentro de los mundos virtuales clips de audio y de video.
- No se requiere de grandes recursos para desarrollar ambientes virtuales en VRML, sólo se necesita un editor de textos, un navegador de Internet y un visualizador VR.

Como se trata de un lenguaje interpretado (esto quiere decir que es "leído" por un visualizador de VRML que se encuentra en la computadora del usuario), se corre el riesgo de que presente más fallas o problemas en tiempo de ejecución.

### 1.6.2 Extensible 3D (X3D).

El consorcio web3D expresa en su página web que el X3D es un formato de archivo estándar abierto y una arquitectura de tiempo de ejecución para representar y comunicar escenas y objetos 3D utilizando XML. Es un estándar ratificado por la International Standards Organization (ISO) que provee un sistema de almacenamiento, recuperación y reproducción de contenidos gráficos en tiempo real embebidos en aplicaciones, todo dentro de una arquitectura para soportar un amplio rango de dominios y escenarios de usuarios. El formato X3D posee un conjunto sustancioso de características empaquetadas en componentes que se ajustan bien para su uso en las visualizaciones de carácter científico e ingenieriles, el diseño asistido por computadoras, visualizaciones médicas, entrenamiento y simulación, multimedia, entretenimiento, educación y más.

El formato X3D discute las limitaciones de VRML, este se especifica totalmente, por lo que el contenido es totalmente compatible. Es extensible, lo que significa que X3D puede usarse para hacer una animación 3D pequeña y eficiente o puede usarse para soportar lo último en extensiones Streaming o de Renderizado. Este formato también soporta codificaciones múltiples y APIs, para que pueda integrarse fácilmente con navegadores Web a través de XML o con otras aplicaciones. Además, según [SABIA, 2007], para ir cerrando el círculo con XML, X3D es la tecnología detrás del soporte 3D del MPEG-4. Utilizando X3D se dejarían de aprovechar características que no son objetivo de la investigación y sin embargo VRML cumple muy bien con la representación de los modelos virtuales necesarios para demostrar el resultado de la investigación.

## 1.7 Breve descripción de la metodología de desarrollo de software utilizada.

Una metodología es el conjunto ordenado de pasos a seguir para cumplir un objetivo. Dicho objetivo, en la ingeniería de software, es el desarrollo de software de alta calidad que cumpla con las necesidades del cliente dentro de un plan y un presupuesto predecible. Para esto es necesario proveer un enfoque disciplinado para asignar tareas y responsabilidades dentro del desarrollo del sistema, determinar un camino metódico y sistemático para desarrollar, diseñar y validar una arquitectura y reducir en gran medida los riesgos que representa la construcción de sistemas de software.

Para el desarrollo del prototipo funcional se usó la metodología del Proceso Unificado de Rational (RUP por sus siglas en inglés). Este proceso de ingeniería de software se basa en la modelación de sistemas informáticos usando la tecnología orientada a objetos, lo que provee un enfoque disciplinado para asignar tareas y responsabilidades dentro de una organización desarrolladora o cualquier proyecto de software.

Entre las características de RUP se encuentran:

- El software se desarrolla de forma iterativa e incremental, lo que posibilita que se vayan eliminando los errores cometidos en las iteraciones previas.
- Se presenta un modelo visual del software.
- Usa arquitecturas basadas en componentes.
- Verifica continuamente la calidad del producto.
- Propone la creación de artefactos como herramientas visuales con el uso del Lenguaje Unificado de Modelado (UML).

UML es "… un lenguaje gráfico para especificar, construir y documentar los artefactos de un sistema con gran cantidad de software. UML proporciona una forma estándar de escribir los planos de un sistema, cubriendo tanto las cosas conceptuales tales como procesos del negocio y funciones del sistema; como las cosas concretas tales como las clases escritas en un lenguaje de programación específico, esquemas de bases de datos y componentes de software reutilizable"[Booch, 1998].

Entre las principales ventajas de RUP se destacan:

- La percepción de los malos entendidos al inicio del proceso de desarrollo.
- Siempre se conoce el estado del proyecto.

- Las inconsistencias entre análisis, diseño e implementación se detectan tempranamente.
- El cliente obtiene resultados a corto plazo.
- Las pruebas se concentran en los aspectos de mayor riesgo.

RUP se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo consta de cuatro fases: inicio, elaboración, construcción y transición, y concluye con una versión del sistema, que está lista para ser entregada a los clientes.

## 1.8 Fundamentación de los lenguajes y las herramientas utilizadas.

A la hora de elaborar un programa informático es necesario utilizar uno o varios lenguajes de programación y herramientas de desarrollo para que la PC tenga las funcionalidades deseadas a través de ese programa. A continuación se tratan brevemente cada uno de estos lenguajes y herramientas necesarias para hacer una aplicación de Realidad Aumentada.

### 1.8.1 Lenguajes C y C++.

Existen muchas razones por las que C y C++ tienen gran aceptación entre los programadores dedicados a los gráficos por computadoras. A continuación se mencionan algunas de las características por las que se seleccionó este lenguaje y no otro, para la implementación del sistema.

El lenguaje C es un lenguaje muy potente y rápido, solo superado por el ensamblador. Es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas operativos, aunque también se utiliza para crear otro tipo de aplicaciones. Puesto que es posible utilizar sus características de bajo nivel para realizar implementaciones óptimas, es catalogado como un lenguaje muy eficiente. Esta característica hace que sea ideal para desarrollar programas de gráficos por computadoras que requieren un alto grado de rapidez y optimización. A pesar de su bajo nivel es el lenguaje más portado en existencia, habiendo compiladores para casi todos los sistemas operativos conocidos. Proporciona facilidades para realizar programas modulares y/o utilizar código o bibliotecas existentes.

C++ es una extensión de C, por tanto hereda las características anteriores y añade otras más. Inicialmente se le conoció como "C con clases", porque soporta el paradigma orientado a objetos. Debido a que permite trabajar tanto a alto como a bajo nivel, es un lenguaje muy potente. El C++ junto

a C son los lenguajes por excelencia para desarrollar gráficos por computadoras. Además C/C++ proporcionan un acceso a bajo nivel de hardware sólo igualado por el ensamblador.

### 1.8.2 OpenGL

[Neider and Davis, 1997] expresan en su "Guía de programación con OpenGL", que esta es una interfaz que consiste en alrededor de 150 comandos distintos usados para especificar objetos y operaciones necesarias para producir aplicaciones interactivas tridimensionales. OpenGL está diseñada como una interfaz de software perfilada e independiente de hardware, con el objetivo de ser implementada en varias plataformas de hardware diferentes entre sí.

Esta interfaz gráfica es independiente del sistema de ventanas utilizado y del sistema operativo, no incorpora rutinas para el manejo de ventanas, este manejo debe realizarse a través del API del entorno de ventanas elegido; se necesita, por lo tanto, un conjunto limitado de rutinas que pongan en contacto al sistema operativo y al entorno de ventanas con OpenGL. Este conjunto de rutinas es diferente para cada sistema operativo y no pertenece a OpenGL. Existen algunas herramientas que facilitan la labor de unir OpenGL con un entorno de ventanas. Ejemplo de estas son: WGL (biblioteca para la familia de sistemas MS-Windows), GLX (biblioteca para los sistemas X Windows) y GLUT (biblioteca que tiene versiones tanto para MS-Windows como para X Windows).

### 1.8.3 Microsoft Visual Studio.NET 2003.

Microsoft Visual Studio.NET 2003 proporciona a los programadores un lenguaje orientado a objetos de probada eficacia para generar aplicaciones de alto rendimiento. Gracias a plantillas avanzadas, acceso a plataformas de bajo nivel y un compilador compatible con los estándares ANSI C y ANSI C++, que optimiza las compilaciones, ofrece funcionalidad para generar componentes sólidos, utilizando el lenguaje de programación C++. Soporta las bibliotecas gráficas OpenGL, DirectX que son las utilizadas por excelencia en la industria de los gráficos por computadoras y video-juegos. No se utilizó una versión más reciente, como Visual Studio.NET 2005, porque utiliza muchos recursos de hardware y hace que el proceso de desarrollo de la aplicación sea más lento. Es válido destacar que no se trabajó con el framework de .NET, esencialmente se utilizó el Visual Studio como Ambiente Integrado de Desarrollo (IDE por sus siglas en inglés) y no como plataforma de desarrollo.

### 1.8.4 Rational Rose Enterprise Edition 2003.

Existen muchas herramientas de ingeniería de software asistida por computadoras (CASE por sus siglas en inglés Computer-Aided Software Engineering) de modelado visual, Rational es una de las mejores desarrollada por Rational Corporation y forma parte de un conjunto más amplio de herramientas que juntas cubren todo el ciclo de vida de desarrollo de software. Dicha herramienta para la definición de sistemas permite que el equipo de desarrollo entienda mejor el espacio problema, que identifique las necesidades del cliente en forma más efectiva y comunique la solución propuesta en forma más clara.

Rational ofrece la capacidad de modelar y visualizar los procesos de negocio, además, cuando se modelan los casos de uso, se asegura que la solución sea creada con el usuario en mente. Los desarrolladores de RUP son miembros de Rational Corporation, por lo que Rational es completamente compatible con la metodología usada.

Otra de las ventajas que ofrece esta herramienta, es que brinda muchas facilidades en la generación de la documentación del software que se está desarrollando. Posee una ayuda muy completa sobre la metodología RUP y los estereotipos de UML que facilita el proceso de desarrollo del software. Rational permite llevar a cabo gran parte de los flujos de trabajo fundamentales de RUP tales como:

- Modelado del negocio.
- Captura de requisitos.
- Análisis y diseño.
- Implementación.
- Prueba

Entre las características principales de Rational se pueden destacar:

- Admite como notaciones: UML, COM, OMT y Booch.
- Realiza chequeo semántico de los modelos.
- Ingeniería “de ida y vuelta”: Rational permite generar código a partir de modelos y viceversa.
- Permite desarrollo multiusuario.
- Permite la integración con modelado de datos.
- Genera documentación del sistema.
- Tiene un lenguaje de script para poder ampliar su funcionalidad.

- Soporta OLE.
- Disponible en múltiples plataformas: HP Unix, Linux, Windows.

### 1.9 Conclusiones parciales del capítulo.

La RA emerge como un campo de investigación muy amplio aplicado a muchas esferas de la vida cotidiana. En este capítulo se analizaron conceptos de RA enunciados por varios autores, reconocidos investigadores en el campo de la RA y se seleccionó uno por el cual se guió la investigación. Se hizo un estudio de los trabajos relacionados con el tema, con el objetivo de ver los aspectos más importante sobre lo que existe actualmente en el mundo y en nuestro país. Se analizaron además los dispositivos de visualización más utilizados en la RA, las características más importantes de los principales framework (plataformas de desarrollo) que se emplean para crear aplicaciones de RA. Con el desarrollo de este capítulo se trataron los aspectos teóricos necesarios para llevar a cabo la confección de la solución que propone la investigación.

# CAPÍTULO 2 SOLUCIÓN PROPUESTA

En este capítulo se propone una solución al problema de la investigación. También se define la estructura principal del prototipo funcional elaborado y se explica de forma detallada la solución.

## 2.1 Soluciones técnicas.

La fundamentación teórica planteada en el capítulo 1 sirvió de sustento para seleccionar las soluciones técnicas. Se eligió la variante de dispositivos alternativos basados en monitor. Este tipo de dispositivo resultó ser el indicado para la investigación, porque puede ser fácilmente reproducido con los recursos que posee actualmente la UCI.

Se seleccionó ARToolKit por las características tratadas en el epígrafe 1.4.4 del capítulo 1 y además porque se ajusta perfectamente al modelo de dispositivo de captura y visualización elegido previamente. Una vez seleccionada ARToolKit, se eligió el tipo de marcador que se utilizaría, el cual resultó ser el estilo de marcador cuadrado por las ventajas que ofrece y por último el *tracking* estuvo dado por técnicas de reconocimiento de patrones utilizadas por ARToolKit.

Para el proceso de desarrollo de la aplicación prototipo se seleccionó la metodología RUP, el lenguaje de modelado UML y la herramienta Rational Rose Enterprise Edition 2003 (Rational). Se optó por el lenguaje C++ y el compilador de Visual C++ que ofrece Microsoft Visual Studio.NET 2003 (VS.NET 2003) para la implementación y para manejo de los gráficos por computadora la biblioteca OpenGL.

Por último, para la representación de los modelos virtuales en la escena real se prefirió emplear VRML para aprovechar las funcionalidades de ARToolKit en el trabajo con este formato y para no añadir complejidad a la solución utilizando otras librerías. Todo lo planteado posibilitó tener las herramientas necesarias para comenzar a desarrollar un prototipo funcional demostrativo de una aplicación de Realidad Aumentada, el cual fue nombrado Localización de Elementos Virtuales con Técnicas de Realidad Aumentada (LETRA).

## 2.2 Estructura principal.

El proceso de desarrollo de aplicaciones de RA se divide en dos partes, la primera es la escritura del código del programa y la otra comprende el entrenamiento de las rutinas de procesamiento de imágenes con los marcadores del mundo real que se usarán en el programa. La estructura principal del programa se rigió por el principio de desarrollo que propone ARToolKit para las aplicaciones de RA

que son elaboradas utilizando su Interfaz de Programación de Aplicaciones (API por sus siglas en inglés). La estructura principal definida para el programa comprende los siguientes pasos propuestos por ARToolKit:

| | |
|---|---|
| **Inicialización** | 1. Se inicia la captura de video, se leen los patrones de marcadores y los parámetros de la cámara de sus respectivos ficheros. |
| **Ciclo Principal del programa**  | 2. Se captura el fotograma de entrada. |
| | 3. Se detecta(n) el(los) marcador(es) y patrón(es) reconocido(s) en el fotograma capturado. |
| | 4. Se calcula la transformación de los marcadores detectados relativa a la cámara. (Traslación, Rotación, Escala). |
| | 5. Se dibujan los elementos virtuales en la posición de los marcadores detectados. |
| **Salida** | 6. Fin de la captura de video y otras acciones que requieran ser tratadas en esta parte del programa. |

*Tabla 1. Estructura principal de la aplicación.*

En la Tabla 1 se aprecia la estructura principal de la aplicación, la misma básicamente comprende 6 pasos distribuidos en 3 partes, inicialización, ciclo principal y salida. La parte de inicialización contiene el primer paso donde se inicia la captura de video, se leen los parámetros de la cámara web que filma la realidad y cada uno de los patrones correspondientes a los marcadores que se van a reconocer en la escena. Además se leen los modelos virtuales 3D que tienen asociados los marcadores. Todo esto se hace a partir de la información guardada en sus respectivos ficheros de configuración.

La segunda parte comprende desde el paso 2 hasta el 5, aquí se va capturando cada fotograma del video (paso 2). A través de una rutina que ofrece ARToolKit se van detectando los marcadores presentes en cada fotograma del video (paso 3). Luego, utilizando técnicas de análisis de imágenes, se calcula la transformación de cada marcador detectado con respecto a la cámara. Esto quiere decir que se calcula para cada marcador su traslación, rotación y escala con respecto a la cámara (paso 4) y finalmente se dibujan los modelos virtuales 3D aplicándole a cada uno la transformación de cada marcador asociado respectivamente.

En la tercera parte y final se lleva a cabo el paso 6 donde se realizan las labores de liberación de la memoria y recursos ocupados por las estructuras de datos; se detiene la captura de video y cualquier otra acción que esté asociada a la terminación del programa.

Los pasos del 2 al 5 van a ser repetidos continuamente siempre que el programa esté en ejecución, mientras que los pasos 1 y 6 se van a ejecutar al inicio y al final de la ejecución del programa, respectivamente. Además de los pasos especificados anteriormente, el programa responde a eventos de teclado y mouse según se ejecuten.

Para la realización de los pasos antes mencionados es necesario conocer determinada información que se obtiene a partir de ficheros de configuración. Es en la inicialización donde el programa obtiene esta información, que está en correspondencia con el tipo de dispositivo utilizado, los parámetros de la cámara que forma parte del dispositivo, los patrones de los marcadores que intervienen en la aplicación y los modelos virtuales 3D que estarán presente en la escena y de los cuales es necesario conocer su posición. Estos elementos son discutidos a continuación, porque son de suma importancia para la solución que se propuso.

### 2.3 Dispositivo de visualización.

El dispositivo de visualización utilizado es una variante de los dispositivos de RA basados en monitor. El monitor puede ser reemplazado por el monitor de una PC. El generador de escena virtual y el mezclador de la escena real con la virtual se juntan en la CPU. Se decidió hacer el tracking por medio de software y para captar el flujo de video proveniente del mundo real se seleccionó una cámara web conectada a la PC. Este consiste en una cámara web USB, conectada a la computadora, mediante la cual se captura la escena real que contiene los marcadores necesarios para llevar a cabo el proceso de inserción de los elementos virtuales que aumentan la escena.

En el caso de la investigación la cámara puede mantenerse estática o en movimiento. Es necesario aclarar que aunque la cámara esté en movimiento, en el contexto del programa su sistema de coordenadas se tomó como un sistema de referencia global estático, es decir, las posiciones de los objetos son calculadas relativas al sistema de coordenadas de la cámara. El flujo de video producido por la cámara es analizado y aumentado antes de ser enviado a la pantalla (display) que es el monitor de la PC. Para realizar este proceso se necesita conocer la configuración del video y los parámetros de la cámara.

### 2.3.1 Configuración de video.

Al comenzar el proceso de captura es preciso conocer de dónde proviene el flujo de video, porque existen dos posibles fuentes de donde obtenerlo: la primera es tomarlo en tiempo real desde una cámara y la otra es desde un archivo de video; además es preciso conocer el formato de píxeles en que se va a recibir el flujo de video. ARToolKit exige que esta información esté disponible para los programas que la utilizan. Debido a esto se tomó en cuenta la especificación que exige ARToolKit en su versión 2.72 para darle a conocer al programa el tipo de dispositivo que se va a utilizar, en caso de que sea desde un dispositivo de captura (cámara) o la dirección de un archivo de video en caso de que la captura no sea en tiempo real y el formato de píxeles que tendrá el video.

Lo anterior se le especifica al programa mediante una cadena de caracteres que varía de acuerdo a la plataforma en la cual se ejecute. Como la plataforma de desarrollo elegida resultó ser Windows, la cadena de caracteres provista es la dirección de un archivo de XML bien formado, conformado de acuerdo al esquema definido por el módulo de captura de video de ARToolKit para Windows, Direct Show Video Library (DSVL).

Este esquema se encuentra documentado en el fichero "DsVideoLib.xsd" dentro del paquete DSVL. Para modificar el fichero de configuración y editar uno propio según se estime conveniente, se puede utilizar un visor de esquemas de XML y un editor de XML. El archivo de configuración quedó especificado como se muestra en el Recuadro 1 donde se especificó que la captura se realiza desde una cámara y el formato de píxeles es RGB24.

```
<?xml version="1.0" encoding="UTF-8"?>
<dsvl_input xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\Librerias\ARToolKit\DSVL \DsVideoLib.xsd">
      <camera show_format_dialog="true" friendly_name="PGR">
            <pixel_format>
                  <RGB24 flip_h="false" flip_v="true"/>
            </pixel_format>
      </camera>
</dsvl_input>
```

*Recuadro 1. Configuración para la captura de video.*

### 2.3.2 Parámetros de la cámara.

Los parámetros de la cámara se obtienen a partir de un proceso de calibración que se realiza antes de que se ejecute el programa. Este proceso de calibración permite conocer los valores intrínsecos principales y necesarios para la representación de la cámara. El modelo de cámara usado es el

"modelo pinhole" con parámetros estándares, los cuales son imprescindibles para generar la matriz de proyección necesaria para dibujar la escena con OpenGL. Los parámetros son guardados en un fichero que se carga al inicio de la aplicación y los datos contenidos son los siguientes:

- Ancho de la imagen en píxeles.
- Alto de la imagen en píxeles.
- Matriz de perspectiva.
- Factor de distorsión.

### 2.4 Marcadores

Para los marcadores se utilizó el estilo de marcador cuadrado propuesto por ARToolKit. Los marcadores utilizados se definen de la siguiente forma: un cuadrado externo negro con un cuadrado interno blanco que a su vez contiene una figura que debe ser lo más asimétrica posible como se muestra en la figura 7. La elección de este tipo de marcador evitó resultados indeseados, los cuales se producen por ambigüedades introducidas a la hora del reconocimiento por parte del algoritmo de visión por computadoras. El marcador es una imagen como se muestra en la figura 7, pero impresa en papel, es decir, el marcador es el papel con la imagen impresa.

*Figura 7. Marcador reconocido dentro de la escena filmada.*

#### 2.4.1 Patrón de marcador.

Como se mencionó en el capítulo 1, ARToolKit hace una combinación de métodos de tracking basados en visión por computadora y entre ellos se encuentra la correspondencia de plantillas, que es el que más peso tiene. El programa creado utiliza esta técnica para reconocer los marcadores en la escena apoyado en el patrón representado por cada marcador. Primeramente, durante un proceso de entrenamiento previo a la ejecución del programa, se generan los patrones respectivos en forma de

mapa de bits utilizando los parámetros de la cámara y se almacenan en un fichero binario. Cada patrón es cargado desde su respectivo fichero al inicio de la ejecución del programa, constituyendo la plantilla con la que se va comparar cada marcador reconocido en la escena, para identificar la correspondencia patrón-marcador e insertar su modelo 3d asociado.

### 2.4.2 Detección de los marcadores.

La estrategia de detección y rastreo de los marcadores estuvo determinada por el algoritmo que utiliza el framework de realidad aumentada ARToolKit en su versión 2.72. El algoritmo está basado en un método de detección de esquinas básico, con un algoritmo de estimación de la pose muy rápido. En una entrevista realizada a Philip Lamb (ver Anexo1), desarrollador líder del proyecto ARToolKit, expresó que este framework utiliza una combinación de métodos de tracking, entre ellos el de tracking por detección y correspondencias de plantillas (template matching), al cual en su conjunto llaman algoritmo de visión por computadoras.

La utilización de este algoritmo permitió reconocer en tiempo real que marcadores están presentes en la escena, a través de un análisis del flujo de video proveniente de la cámara. Además del reconocimiento de los marcadores, también fue posible obtener en tiempo real la pose de cada marcador con respecto a la cámara. Esto significa que de cada marcador se obtuvo una matriz de transformación que contenía adentro información de rotación, traslación y puesta en escala del marcador referente al sistema de coordenadas de la cámara.

## 2.5 Modelos virtuales.

Introducir la información complementaria, en este caso los elementos virtuales, es primordial para lograr el aumento de la realidad. Se utilizaron modelos diseñados previamente en la herramienta de diseño 3D Studio Max, debido a las potencialidades que ofrece y porque es compatible con el formato de representación de modelos elegido. El formato elegido fue VRML 2.0 o VRML 97 como también se le conoce, consiste en un lenguaje de modelado de la realidad virtual (Virtual Reality Modeling Language, VRML por sus siglas en inglés). El VRML es un estándar muy utilizado en la descripción de entornos virtuales para la web, una de las proyecciones futuras de esta investigación, aunque se este no es su único fin. Además VRML tiene la ventaja de ser un formato libre y ARToolKit tiene un módulo para el trabajo con este formato.

## 2.6 Fichero de Configuración de patrones y modelos virtuales.

Buscar una alternativa, para que el usuario pudiera especificar que patrones intervendrán en la ejecución del programa, así como el modelo que tiene asociado cada patrón y otra información relacionada con el patrón sin tener que entrar en el código del programa, condujo a la definición de un fichero de configuración. Primeramente hay que aclarar que se tomó una convención para poner comentarios en el cuerpo del fichero. Aquellas líneas que comiencen con el símbolo '#' son consideradas como un comentario y sirven para documentar la información del fichero, por lo tanto estas líneas, al igual que las líneas en blanco, no son tomadas en cuenta a la hora de leer el contenido del fichero. El recuadro 2 muestra la estructura del fichero de configuración definido.

```
<Cantidad de asociaciones>

<Comentario del bloque>
<Nombre "nombre arbitrario">
<Dirección del Patrón "patt.nombre">
<VRML> <Dirección de fichero configuración de VRML "fich_wrl.dat">
<Ancho del marcador "mm">
<Centro del marcador "x y">
```

*Recuadro 2 Estructura del fichero de configuración de patrones y modelos.*

Lo primero que se debe especificar en el fichero de configuración es el número de asociaciones (entiéndase patrón-marcador-modelo) que estarán involucradas en la ejecución del programa. A continuación siguen tantos bloques como asociaciones se especificaron, de la siguiente forma: <Comentario del bloque> aquí se pone en tantas líneas como se estime conveniente una descripción de la información relacionada con este bloque de forma legible; <Nombre arbitrario> como se indica en ésta línea se le da un nombre arbitrario al bloque, se recomienda asignar un nombre intuitivo; <Dirección del patrón> aquí se detalla la dirección del fichero donde se encuentra el patrón que será utilizado para hacer el reconocimiento; <VRML> indica que lo que se va a leer a continuación es la dirección de un fichero VRML especificada por <Dirección del fichero VRML> que contiene la representación de un modelo 3D, esta especificación de tipo de formato y dirección se definió así con la intención de en un futuro, si es necesario, cargar otro formato de modelo especificando que tipo de modelo se va a cargar para darle un tratamiento diferenciado; a continuación en *<Ancho del marcador>* se especifica el ancho que tiene cada marcador físico en milímetros para que el algoritmo de localización pueda estimar de forma correcta las transformaciones de traslación y rotación del marcador en la escena; finalmente en < *Centro del marcador "x y"*> se debe especificar donde se

quiere ubicar el centro de coordenadas del marcador para luego en la ejecución del programa ubicar correctamente el modelo virtual asociado.

### 2.7 Sistema de referencia.

Difícilmente en una aplicación de gráficos por computadoras, que incluye modelación 3D, se trabaja con un único sistema de coordenadas (SC), como mínimo se utilizan 3: el SC del mundo virtual, el SC de la cámara virtual y el SC de la pantalla. En este caso se trabajó a partir de los que propone ARToolKit que son básicamente 3 SC como se observa en la figura 8. Estos son utilizados principalmente por el algoritmo de visión y en las tareas de dibujado en pantalla. Aunque estos SC son los básicos, la realidad es que adicionalmente se tendrán tantos como marcadores se encuentren distribuidos en la escena.

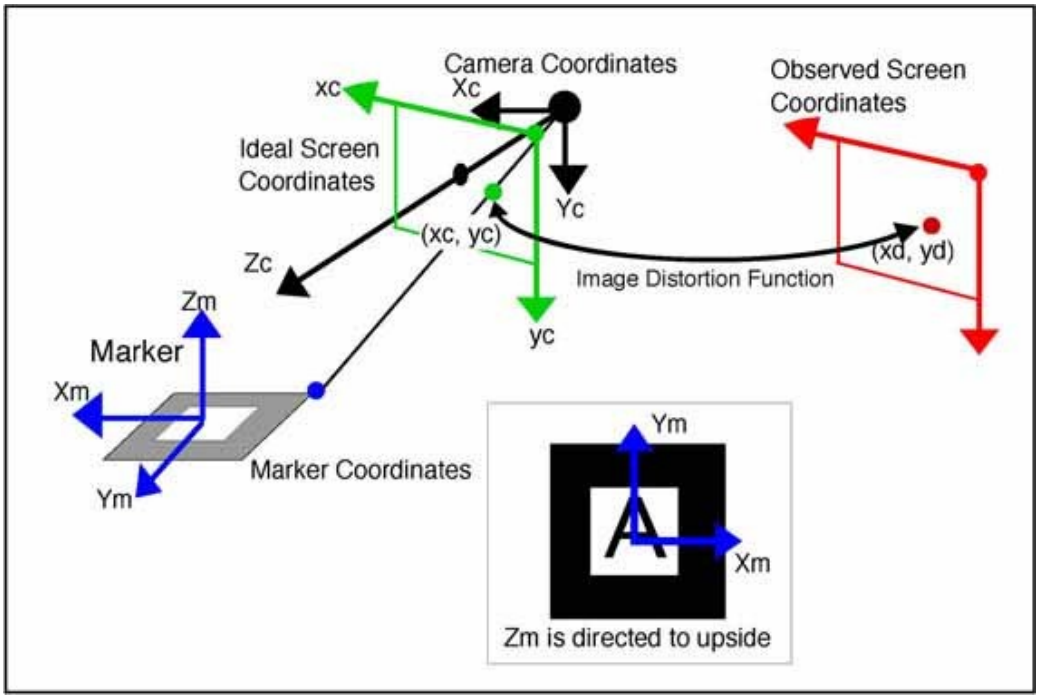



*Figura 8. Sistema de coordenadas de ARToolKit.*

La figura 8 muestra una correspondencia entre ARToolKit y OpenGL, la primera utiliza una matriz de perspectiva generada a partir de la información de los parámetros de la cámara, contenida en el fichero de calibración y que resulta útil en la construcción de una matriz de proyección para la cámara de OpenGL. Esta matriz de proyección no pudo ser creada por las rutinas habituales de OpenGL (*gluPerspective*), sino que una mejor alternativa fue crear la matriz de proyección manualmente

utilizando parámetros del volumen de visión (*Frustum*). Como resultado se obtuvo la matriz de proyección y se cargó, situando así el origen del sistema de coordenadas en la cámara de OpenGL, la cual tiene la misma ubicación que la cámara real, respecto a los modelos. Con esto se logró que los elementos virtuales se vieran desde la perspectiva en que la cámara real observa los marcadores, cumpliendo con uno de los principales objetivos de la realidad aumentada.

### 2.7.1 Relación cámara marcador.

La salida del algoritmo de visión utilizado es la matriz de transformación de cada marcador relativa al SC de la cámara real. Mediante esta matriz de transformación se obtuvo la posición de cada marcador en el SC de la cámara real, dicha información es necesaria para insertar los elementos virtuales. El SC del marcador tiene la misma orientación que del SC de OpenGL, por tanto esto hizo posible utilizar el sistema de matrices de OpenGL para posicionar los elementos virtuales en el SC del marcador y hacerle las transformaciones pertinentes. Es válido aclarar que como el SC de cada marcador está alineado según el según OpenGL, todas las transformaciones aplicadas deben respetar las reglas de transformación de OpenGL. Los SC de la cámara y el marcador se encuentran ubicados como se muestra en la figura 9.

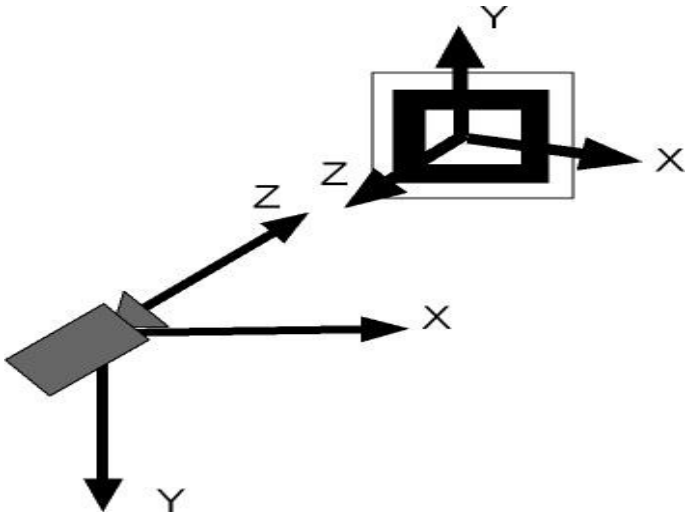



*Figura 9. Relación cámara marcador.*

### 2.7.2 Relación entre marcador y elemento virtual.

Cada marcador tiene asociado un elemento virtual que se inserta en la escena cada vez que es reconocido. Como se pudo apreciar en el epígrafe anterior cada marcador posee un sistema de

coordenadas independiente del sistema de coordenadas de la cámara. El elemento virtual que se introduce en la escena a aumentar se ubica en función del SC de su marcador correspondiente. A partir de aquí, el elemento puede ser trasladado según sea conveniente, pero su ubicación siempre va a ser relativa al SC del marcador, para expresarlo con respecto a otro SC es necesario aplicarle una transformación.

### 2.7.3 Relación entre la cámara y el elemento virtual.

Una vez que se detecta el marcador y se introduce el objeto en su SC. Es preciso poner la posición del objeto que está expresada en el SC del marcador en función del SC de la cámara para cumplir con el objetivo principal de la investigación. Se necesita aplicar una transformación (multiplicar por una matriz de transformación) al punto donde está ubicado el elemento virtual para obtener sus coordenadas con respecto a la cámara.

### 2.7.4 Cambio entre sistemas de coordenadas.

En este y los subsiguientes epígrafes se explica brevemente el procedimiento algebraico necesario, el cual permitió en gran medida dar cumplimiento al objetivo de la investigación.

Un vector expresado en función de un sistema de referencia puede expresarse en función de otro cualquiera. Solamente se tendrá que aplicar una matriz de transformación que genere las componentes del mismo vector en función del segundo sistema de referencia. Es válido aclarar que con esta transformación el vector no cambia, sigue estando en el mismo lugar, solo que se referencia desde otro lugar.

Asumiendo que se tiene una base generadora origen y $S=\{v_1,\ v_2,\ v_3\}$ una base generadora destino $S'=\{u_1,\ u_2,\ u_3\}$ ambas compuestas por 3 vectores y pertenecientes a $\Re^3$, entonces todos los vectores de una base se pueden expresar como combinación lineal de la otra, porque pertenecen al mismo espacio vectorial. A continuación se puede observar lo enunciado anteriormente donde se expresan los vectores de la base $S'$ como combinación lineal de los vectores de la base $S$.

$$\begin{aligned} u_1 &= a_{11} * v_1 + a_{12} * v_2 + a_{13} * v_3 \\ u_2 &= a_{21} * v_1 + a_{22} * v_2 + a_{23} * v_3 \\ u_3 &= a_{31} * v_1 + a_{32} * v_2 + a_{33} * v_3 \end{aligned} \qquad (1)$$

A partir de la representación anterior se obtiene una matriz de 3x3 con los coeficientes $a_{mn}$ la cual constituye la matriz de cambio de los vectores de la base $S$ a la base $S'$.

$$M = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \quad (2)$$

Luego teniendo un vector $w = (c_1 \quad c_2 \quad c_3)$ referente a la base $S$, sus componentes se pueden expresar en función de la base $S'$ de la siguiente forma:

$$w' = M * w \quad (3)$$

Luego sustituyendo (2) en (3) se obtiene:

$$\begin{pmatrix} c'_1 \\ c'_2 \\ c'_3 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} * \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} \quad (4)$$

El vector $w' = (c'_1, \quad c'_2, \quad c'_3)$ obtenido es el mismo vector $w$, solamente que ahora está referido a la base $S'$[1].

### 2.7.5 Coordenadas homogéneas y transformaciones afines.

Generalmente cuando se quieren representar los vértices que definen una geometría se utilizan coordenadas cartesianas. Un punto suele ser representado por $P = (x, y, z)$, lo cual significa una localización en el espacio 3D. En gráficos por computadoras no suele suceder así. En el mundo de los gráficos por computadoras se habla de puntos y vectores. Esto hace que su representación se torne un poco confusa, teniendo en cuenta que un vector es una resta entre dos puntos como se muestra a continuación.

$$v = (x_1, y_1, z_1) - (x_2, y_2, z_2) \quad (5)$$

[1] Para una mejor información consultar bibliografía relacionada con el álgebra lineal y matricial.

$$v = (a,b,c) \tag{6}$$

Como se observa en (6) la representación de $v$ no difiere de la de $P$ por lo tanto para no confundir su representación se decidió utilizar la notación de coordenadas homogéneas, ampliamente utilizadas en los gráficos por computadoras. Para transformar la geometría se utilizaron transformaciones de rotación, escalado y traslación, normalmente conocidas como transformaciones afines porque conservan las líneas (curso aula macedonia). Se recurrió a este tipo de transformaciones porque se pueden aplicar varias de estas a una geometría solo multiplicando matrices y además por ser el utilizado en OpenGL. Esto constituyó una razón más para utilizar coordenadas homogéneas dado que para efectuar estas transformaciones se utilizaron matrices, además de que no es correcto alterar de igual forma un vector y un punto.

Para convertir un punto o un vector en coordenadas cartesianas a su representación homogénea se añadió una cuarta coordenada $w$ a las clásicas $(x,y,z)$. De esta forma un punto $P_1 = (x_1, y_1, z_1)$ en coordenadas cartesianas se convierte a $P_1 = (x_1, y_1, z_1, w_1)$ en coordenadas homogéneas y un vector $v_1 = (a_1, b_1, c_1)$ se convierte a $v_1 = (a_1, b_1, c_1, w_1)$ en homogéneas. Los valores de esta nueva coordenada son:

- $w = 0$ para un vector.
- $w = 1$ para un punto.

Dentro de las transformaciones existentes los autores de esta tesis consideraron que la transformación de traslación con respecto al marcador era suficiente para demostrar la solución. “La traslación es una transformación afín imposible de realizar en coordenadas cartesianas si no se incluye una suma de matrices. La traslación como transformación de instancia es muy utilizada en las aplicaciones gráficas para mover objetos o entidades geométricas situándolas en una posición determinada” [Iznaga and Pérez, 2006]. Para trasladar los modelos con respecto al marcador se utilizó una matriz de traslación formada a partir coordenadas homogéneas como se observa a continuación.

$$\begin{pmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{7}$$

Las variables $T_x, T_y, T_z$ denotan la traslación que se le aplica a un punto a partir de su ubicación. Suponiendo que se tiene una superficie triangular denotada por tres puntos $P_1 = (x_1, y_1, z_1, 1)$, $P_2 = (x_2, y_2, z_2, 1)$ y $P_3 = (x_3, y_3, z_3, 1)$. Para trasladar esa superficie $T_x', T_y', T_z'$ unidades en los respectivos ejes de coordenadas utilizando la matriz de traslación, se realiza una operación de multiplicación de la matriz por cada uno de los puntos como se observa en (8), el resultado se ilustra en la figura 10:

$$\begin{pmatrix} 1 & 0 & 0 & T_x' \\ 0 & 1 & 0 & T_y' \\ 0 & 0 & 1 & T_z' \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ z_1 & z_2 & z_3 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} x_1' & x_2' & x_3' \\ y_1' & y_2' & y_3' \\ z_1' & z_2' & z_3' \\ 1 & 1 & 1 \end{pmatrix} \quad (8)$$

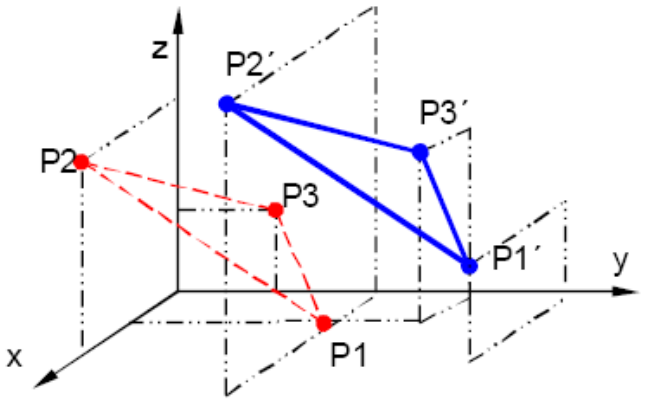



*Figura 10. Traslación de una superficie triangular.* [Iznaga and Pérez, 2006]

### 2.8 Localización de objetos virtuales en la escena real.

Luego de un análisis de la base algebraica necesaria se procedió a dar solución al problema planteado. Una vez determinada la pose del marcador, utilizando la librería ARToolKit, se introduce el elemento virtual correspondiente en su centro de referencia. A partir de este momento el elemento virtual puede ser trasladado con respecto al marcador tanto como sea necesario, gracias a la matriz de traslación (7) que se le asoció al elemento virtual. Utilizando la ecuación (8) se ubicó el elemento virtual con respecto al marcador luego de una traslación y utilizando la ecuación (4) se obtuvo la posición del elemento virtual con respecto al sistema de coordenadas de la cámara.

La matriz de cambio de coordenadas del sistema de referencia del marcador al sistema de referencia de la cámara se obtuvo por medio de ARToolKit. La localización del elemento virtual con respecto a la cámara se obtuvo multiplicando la posición del elemento por la matriz de transformación devuelta por ARToolKit. Esto constituyó el paso principal para dar solución al problema de la investigación.

## 2.9 Áreas de aplicación.

Esta investigación constituye la base para la elaboración de un sistema de Realidad Aumentada, porque tuvo como resultado la localización de patrones y elementos virtuales en tiempo real. Una de las razones principales del desarrollo de esta investigación fue el gran campo de aplicación que tiene su resultado, pudiendo utilizarse en casi todas las esferas de la sociedad. Es imposible mencionar y dar una explicación detallada de todas las áreas donde tiene aplicación el resultado de esta investigación debido a que sería muy extenso. A continuación se detallan solamente algunas aplicaciones que puede tener en determinadas áreas.

### 2.9.1 Aplicaciones en la industria

**Manuales virtuales.**

En la industria se suele trabajar con maquinarias complejas que requieren del conocimiento y experiencias de técnicos calificados a la hora de reparar alguna de sus piezas cuando ocurre algún fallo, o cuando se necesite ensamblar algún equipo. En este caso se puede construir una representación virtual de estos equipos y su funcionamiento, así como la forma en que estos se ensamblan y desensamblan. Con este tipo de representación creada, solo bastaría asociar este modelo a un patrón cualquiera y el usuario, que puede carecer de experiencia en las labores de reparación de la maquinaria o el ensamblaje de los equipos, puede tener una idea más real y más concisa de lo que debe realizar. Esto permite reducir el riesgo de accidentes laborales o mal funcionamiento de los equipos, por mala manipulación, que puede resultar en pérdidas de recursos humanos y materiales para la industria.

### 2.9.2 Educación.

**Proyectos educativos.**

Se conoce que en un museo siempre se exhiben objetos interesantes, generalmente únicos porque pertenecieron a épocas pasadas, prendas que pertenecieron a personalidades, obras de arte hechas

por famosos artistas, entre otras cosas más que podemos encontrar en un museo. Utilizando el resultado de esta investigación se podrían lograr exhibiciones virtuales recreando un museo en un aula aumentada. Se puede acondicionar un aula y distribuir una serie de marcadores en todo el local con sus respectivos elementos virtuales asociados, representativos de piezas únicas que solo pueden ser vistas en los museos más famosos del mundo. Así por ejemplo se podría tener la representación virtual 3D de la mona Lisa, dentro del aula, sin tener que visitar el Louvre.

**Asignaturas en la UCI.**

En la UCI puede ser aplicado en la asignatura Máquinas Computadoras. Debido a la ausencia de recursos materiales para la enseñanza de la arquitectura interna de una computadora, cómo se compone y mostrar a los estudiantes cómo ubicar sus componentes. Para su aplicación serían necesarios modelos virtuales de los diferentes componentes internos de las computadoras. En el aula el profesor utilizaría la Realidad Aumentada para mostrar a los estudiantes cómo ensamblar una computadora e incluso mostrar el funcionamiento de cada uno de los componentes por separado; algo imposible de lograr con las piezas reales por si solas. Esto contribuiría a una mejor comprensión de los temas de hardware.

En la asignatura de matemática puede utilizarse para mostrar a los estudiantes las representaciones geométricas de cuerpos y figuras generados por funciones que se estudian en clases. Con esta solución los profesores podrían preparar clases más interactivas e interesantes. En investigaciones realizadas en otros países se han obtenido buenos resultados de la aplicación de la Realidad Aumentada en esta área.

### 2.9.3 Medicina.

En la medicina tiene gran aplicación sobre todo en la enseñanza, donde los estudiantes necesitan conocer como se componen el cuerpo humano y su funcionamiento. A menudo resulta muy difícil obtener determinados órganos para su estudio, por diferentes motivos. Los órganos pueden no estar disponibles debido a que resulta difícil encontrar algún donante o en caso de que estén disponibles, la cantidad puede no ser la suficiente como para hacer las prácticas necesarias, además de los trámites que son necesarios para su obtención.

En este caso la elaboración de modelos virtuales que representen los órganos del cuerpo humano apoyaría el aprendizaje de los estudiantes de medicina. El empleo de la RA permite a los estudiantes

examinar estos órganos virtuales desde cualquier ángulo o perspectiva y así en dependencia del detalle del modelo representado será el nivel de realismo que se obtendrá. Además esto resulta en una reducción de los esfuerzos realizados a la hora de hacer los trámites para la adquisición de los órganos, los cuales suelen ser muy complejos.

Otra aplicación muy importante dentro del marco de la facultad 5 de la UCI, es en el proyecto de simulador quirúrgico. Aquí sería posible utilizarlo para la manipulación de elementos virtuales en un entorno real. Esto reportaría un mayor nivel de realismo al simulador, además de ser una tecnología que se está utilizando en países desarrollados.

### 2.9.4 Arquitectura.

En la arquitectura resulta muy útil sobre todo para resucitar virtualmente edificios históricos destruidos, así como proyectos de construcción que todavía están bajo planeación. Sobre todo en la planeación puede resultar muy útil, debido a que si se quiere llevar a cabo una construcción, el equipo de trabajo encargado de realizarla puede utilizar muchos modelos, cada uno con sus respectivos patrones. En este caso se produciría un ambiente de trabajo colaborativo en el que los elementos a construir son mostrados como si fueran una maqueta virtual a la ves que los marcadores son ordenados sobre una mesa de trabajo real o específicamente en el lugar donde se quiere realizar el proyecto. Esto constituye un ahorro grande de esfuerzos y costos porque el equipo de trabajo y el cliente tiene una idea concreta de lo que se va a realizar.

## 2.10 Conclusiones parciales del capítulo.

En este capítulo se detalló la solución propuesta exponiendo la estructura que tiene la misma. Luego se hizo hincapié en como debe ser la configuración de video, el dispositivo de visualización, la captura de los parámetros de la cámara, el tipo de marcador, su conversión a patrón, su detección y relación con los modelos virtuales a través del fichero de configuración. Estos aspectos constituyeron la base para procesar la relación cámara-marcador, marcador-modelo virtual y modelo virtual-cámara a través de métodos algebraicos como cambios de sistemas de coordenadas, coordenadas homogéneas y transformaciones afines que permitieron lograr el objetivo principal de la investigación, la localización de objetos virtuales en una escena real. Además quedan trazadas las pautas para la descripción de la solución propuesta a través de la metodología de desarrollo de software seleccionado.

# CAPÍTULO 3 DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA.

## 3.1 Introducción

En este capítulo se realiza la descripción de la solución propuesta a través de la metodología de desarrollo de software utilizada y los diagramas que propone UML. Se precisan las reglas del negocio y el modelo conceptual, se especifican los requisitos funcionales y no funcionales del prototipo funcional. Además se describen los procesos obtenidos a partir de los requerimientos funcionales.

## 3.2 Reglas del Negocio

1. Se utilizará solo una cámara.
2. La cámara se conectará al PC mediante USB.
3. Cada objeto virtual estará dentro de un directorio que tiene su mismo nombre.
4. En el directorio del objeto estarán al mismo nivel el fichero que lo representará y un directorio con sus texturas si las tiene.
5. Los directorios de los objetos virtuales estarán dentro de un directorio llamado *Simple*.
6. El directorio *Simple* estará dentro de otro llamado *Data*.
7. El fichero de configuración de los objetos, que aparecerán en la escena, tiene que ser en texto plano.
8. El directorio *Data* estará al mismo nivel que el ejecutable de la aplicación.
9. El fichero de configuración con los parámetros de la cámara, el de la configuración de video y el de configuración de los objetos virtuales estará en el directorio *Simple*.
10. Para añadir o quitar objetos virtuales, el usuario tendrá que editar el fichero de configuración en un editor de texto ASCII.
11. Los ficheros de los patrones de marcadores estarán dentro de un directorio.
12. El directorio de los patrones estará al mismo nivel de *Data*.
13. Cada fichero de patrón se nombrará como "patt.nombre" por ejemplo el “patrón 1” se llamará "patt.1".
14. En la escena aumentada se visualizarán tantos modelos virtuales como se especifique en el fichero de configuración cuando aparezcan sus respectivos marcadores.
15. Se mostrará la información de un solo marcador y su objeto virtual correspondiente cuando esté solo en la escena.
16. Para demostrar la localización de los objetos solo se hará uso de la traslación.
17. La interacción con la escena se hará por medio del teclado.

18. La información se mostrará en la consola o en la escena aumentada.
19. Los objetos se trasladarán según el usuario estime conveniente y mediante teclas específicas.
20. La aplicación tendrá una pequeña guía de las funcionalidades que presenta.

### 3.3 Modelo de Dominio

El objetivo principal del modelo de dominio es identificar los conceptos más importantes y relacionarlos con el propósito de comprenderlos, estos se representan en el siguiente diagrama:

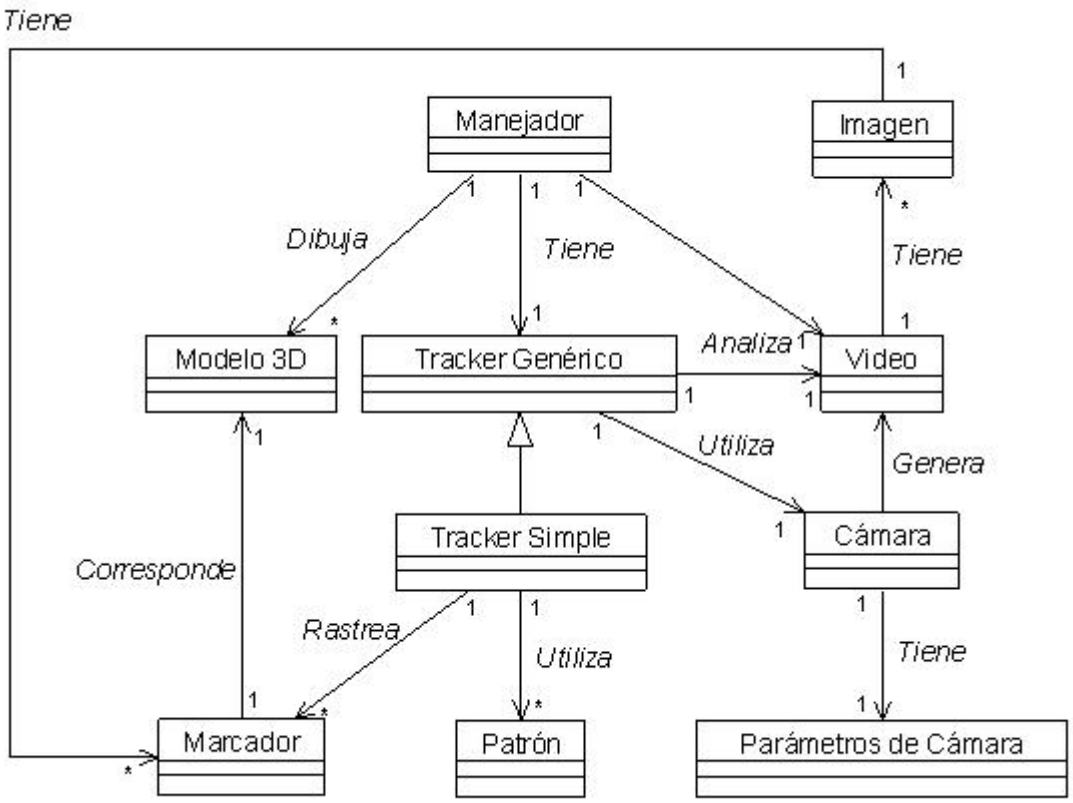



*Figura 11. Modelo del dominio.*

### 3.4 Glosario de Términos del Dominio.

En aras de lograr un mayor entendimiento del modelo de dominio, a continuación se enuncia el significado de los conceptos identificados en este:

**Manejador:** Este es un concepto abstracto el cual expresa que es el encargado de manipular otros conceptos. El contiene un *Tracker Genérico*, un *Video* y dibuja los *Modelos 3D*.

**Imagen:** es cada fotograma del video.

**Video:** Está compuesto por un conjunto de imágenes de la escena que se suceden. Este es generado por la cámara.

**Tracker Genérico:** Es una interfaz a un rastreador, que busca marcadores en el video.

**Tracker Simple:** Es un rastreador que busca marcadores en la escena sin asociarlos entre ellos. Cada marcador es tratado de forma independiente.

**Patrón:** Mapa de bits generado a partir de la región que ocupa un marcador determinado en cada fotograma del video capturado por la cámara.

**Marcador:** Figura impresa en un papel. Consiste en un cuadrado negro externo con uno en blanco interno y dentro del cuadrado blanco una figura con la que se identifica el marcador.

**Modelo 3D:** Representación de un elemento real a través de materiales y herramientas. En este caso los modelos son virtuales, creados utilizando 3D Studio Max u otra herramienta de diseño para la modelación de objetos virtuales.

**Parámetros de la Cámara:** Valores intrínsecos de la cámara, estos son: centro de distorsión (X, Y), radio de distorsión y matriz de perspectiva con respecto a la cámara. Esta matriz es utilizada para calcular la posición del objeto 3D con respecto a la cámara real.

**Cámara:** Dispositivo de captura de video, genera un video con la escena capturada.

**Flip:** Propiedad del video para voltear sus fotogramas tanto vertical como horizontalmente.

**Fotograma:** Ver concepto **imagen**.

### 3.5 Captura de Requisitos

La captura de requisitos es una de las actividades del proceso de desarrollo de requisitos. En ésta actividad se identifican las capacidades, condiciones y restricciones que debe tener un software. A continuación se muestran los requisitos funcionales y no funcionales identificados.

#### 3.5.1 Requisitos Funcionales.

RF1. Configurar modo de visualización del video.
- RF1.1 Mostrar ventana de configuración de video.
- RF1.2 Elegir número de fotogramas por segundo.
- RF1.3 Elegir tamaño de ventana de visualización (Resolución del video).
- RF1.4 Elegir formato de color de píxeles.
- RF1.5 Elegir flip horizontal de video.

RF2. Cargar configuración de marcadores y elementos virtuales.
- RF2.1 Cargar fichero de texto plano (ASCII).
- RF2.2 Abrir fichero de configuración en modo solo lectura.
- RF2.3 Descartar comentarios en el fichero de configuración.
- RF2.4 Leer configuración de objetos y marcadores por bloques de información.
- RF2.5 Cerrar fichero de configuración luego de la lectura.

RF3. Detectar marcadores en la escena filmada.
- RF3.1 Detectar marcadores cuadrados.
- RF3.2 Detectar marcadores en blanco y negro.
- RF3.3 Detectar uno o varios marcadores en la escena.

RF4. Calcular transformaciones de los marcadores.
- RF4.1 Calcular transformaciones de uno o varios marcadores con respecto a la cámara real.
- RF4.2 Hacer el cálculo de transformaciones de los marcadores en tiempo real.

RF5. Mostrar escena aumentada en colores.

RF6. Mostrar escena aumentada en blanco y negro.

RF7. Introducir elementos virtuales 3D en la escena real.

RF8. Calcular posición del elemento virtual 3D respecto a la cámara real.

RF9. Trasladar elementos virtuales 3D con respecto a sus marcadores correspondientes.
- RF9.1 Trasladar elementos virtuales en x,y,z.

RF10. Mostrar información de un elemento 3D que aumenta la escena real.

- RF10.1 Mostrar localización respecto a la cámara.
- RF10.2 Mostrar nombre del modelo.
- RF10.3 Mostrar distancia hasta la cámara.
- RF10.4 Mostrar distancia hasta el marcador.

RF11. Mostrar información de su marcador correspondiente

- RF11.1 Mostrar nombre del marcador.
- RF11.2 Mostrar localización del marcador respecto a la cámara.

RF12. Cambiar vista de la escena aumentada.

- RF12.1 Poner vista de la escena en blanco y negro.
- RF12.2 Poner vista de la escena en colores.

RF13. Terminar ejecución del programa.

- RF13.1 Liberar memoria.
- RF13.2 Liberar recursos.

### 3.5.2 Requisitos No Funcionales.

**Usabilidad:** La aplicación elaborada podrá ser utilizada por cualquier usuario que esté en capacidad de asociar los modelos virtuales, que desea se encuentren en la escena real, con sus respectivos marcadores, editando el fichero de configuración.

**Rendimiento:** La aplicación debe ejecutarse y mostrar los elementos virtuales en tiempo real, debe tener alta velocidad de procesamiento o cálculo.

**Soporte:** En su versión inicial deberá ser compatible con la plataforma Windows, pero debe estar preparado para migrar a GNU\Linux haciéndole pocas o ninguna modificación.

**Hardware:** Debe funcionar sobre microprocesador Intel Pentium (4) 2.8 GHz o superior, 256 MB RAM. Tarjeta de video onboard (recomendada tarjeta de video para juegos NVidia o ATI).

**Software**: La aplicación se debe desarrollar en ANSI C++, utilizar OpenGL específicamente GLUT y para su versión en Windows debe utilizar DirectX 9.0c para el manejo del video. Se hará uso de Programación Orientada a Objetos.

### 3.6 Casos de uso del sistema.

En este epígrafe se muestra el diagrama de casos de uso, en la figura 12, elaborado a partir de las funcionalidades que debe tener el prototipo funcional desarrollado. En este diagrama se puede apreciar el actor que interactúa con la aplicación el cual se detalla en el próximo epígrafe.

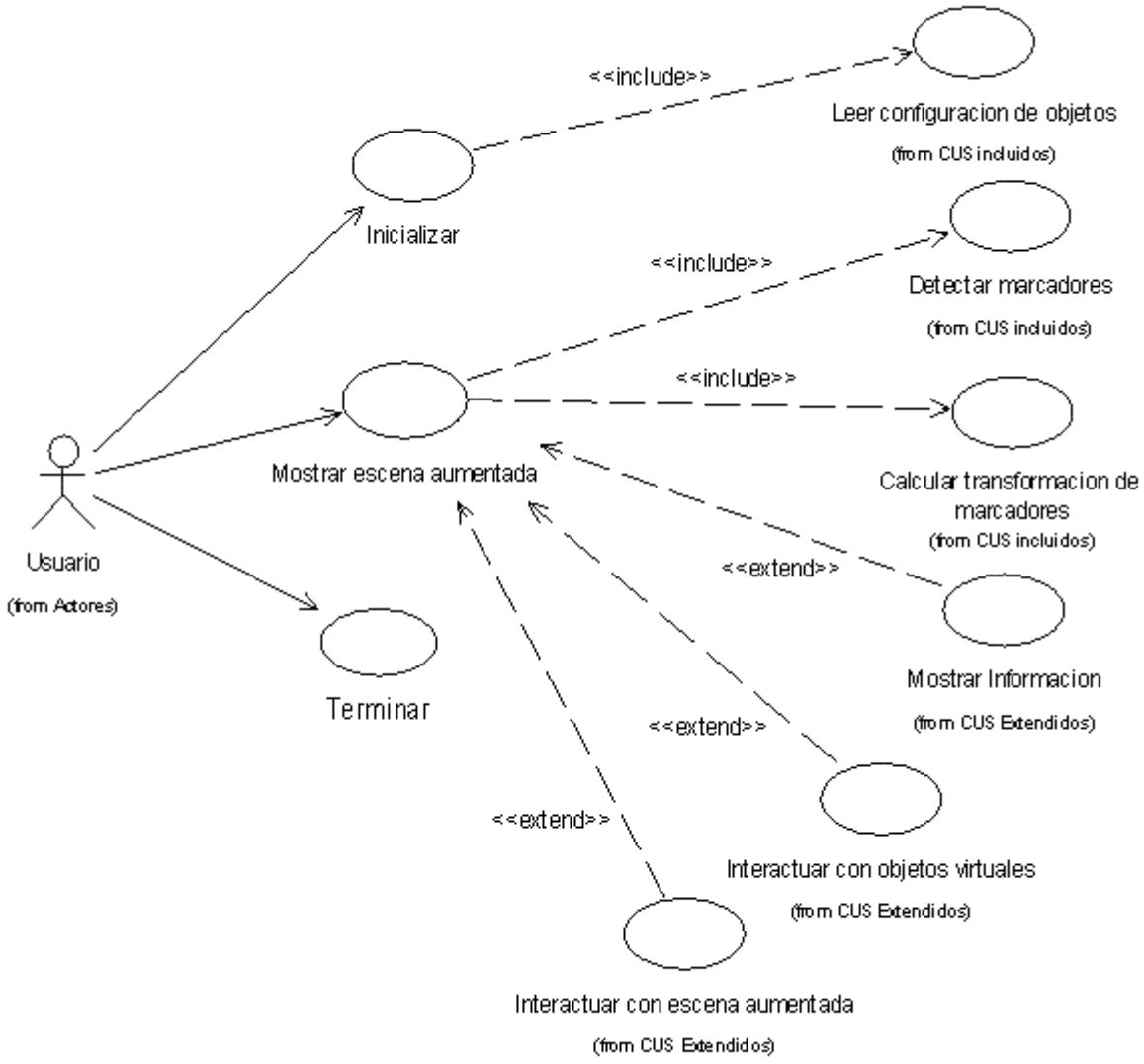



*Figura 12. Diagrama de casos de uso del sistema.*

### 3.7 Definición del actor del sistema.

La aplicación elaborada puede ser utilizada por personas pertenecientes a diferentes áreas de trabajo. Por ejemplo un estudiante, un profesor, un médico o un obrero de la industria la puede utilizar, solamente tendrá que cambiar los modelos virtuales para que se ajusten a su contexto de trabajo. Debido a esto se decidió nombrar al actor como "Usuario", el cual se representa en la tabla 2.

| Actores | Justificación |
|---|---|
| Usuario | Es el que interactúa con la aplicación, solamente tendrá que decidir los modelos necesarios para aumentar la realidad. |

*Tabla 2. Definición del actor.*

### 3.8 Expansión de casos de uso.

| | |
|---|---|
| **Caso de uso:** | Inicializar |
| **Actor (es):** | Usuario |
| **Propósito:** | Brindar la información necesaria para la correcta ejecución de la aplicación. |
| **Resumen:** | El CU comienza cuando el actor ejecuta la aplicación, el sistema le pide la información de la cámara que necesita, carga la otra de sus respectivos ficheros de configuración y culmina cuando toda esta información es obtenida. |
| **Referencias:** | RF 1, RF 1.1, RF 1.2, RF 1.3, RF 1.4, RF 1.5, CU Leer Configuración de Objetos. |
| **Pre-condiciones:** | La cámara está conectada a la PC.<br><br>El programa conoce la dirección del archivo de configuración de video.<br><br>El programa conoce la dirección del archivo que contiene lo parámetros de la cámara.<br><br>El programa conoce la dirección del archivo de configuración de marcadores y elementos virtuales. |
| **Curso Normal de los Eventos** | |
| **Acción del Actor** | **Respuesta del Sistema** |

| | |
|---|---|
| 1. Ejecuta la aplicación.<br><br><br><br>2. Selecciona fotogramas por segundo y formato de píxeles. | 1.1- Lee archivo de configuración de video.<br>1.2- Lee archivo de configuración de la cámara.<br>1.3- Abre ventana de configuración de video.<br><br>2.1- Calcula resolución de video.<br>2.2- Calcula configuración de los parámetros de la cámara en función de la resolución de video.<br>2.3- Inicializa parámetros de la cámara y los muestra.<br>2.4- Comienza captura de video.<br>2.5- Invoca CU Leer Configuración de Objetos. |
| **Curso Alternativo de los Eventos** | |
| | 1.1- Si el archivo de configuración de video no está disponible lanza un mensaje alertando su ausencia y termina el caso de uso.<br>1.2- Si el archivo de configuración de la cámara no está disponible lanza un mensaje alertando su ausencia y termina el caso de uso.<br>1.3- Si la cámara no está conectada lanza un mensaje alertando que falta el dispositivo de captura de video y termina el caso de uso. |
| | 2.4- Si ocurre algún error en la captura de video detiene la ejecución y termina el caso de uso. |
| **Post-condiciones:** | Se inicializaron los parámetros de la cámara.<br><br>Se inició la captura de video. |
| **Prioridad:** | Crítico. |

*Tabla 3. Expansión del caso de uso Inicializar.*

| | |
|---|---|
| **Caso de uso:** | Leer configuración de objetos. <Inclusión> |
| **Actor (es):** | |

| | |
|---|---|
| **Propósito:** | Cargar la información referente a los marcadores que van a estar presentes en la escena capturada por la cámara y sus modelos virtuales correspondientes. |
| **Resumen:** | El CU es inicializado por el sistema, comienza cuando el sistema necesita conocer la información de los marcadores y elementos virtuales. El sistema lee la cantidad de asociaciones que tendrá que crear y las construye con información que lee a continuación. Termina cuando son leídos todos los bloques de asociaciones o cuando se produce algún error en la lectura de la información. |
| **Referencias:** | RF 2, RF 2.1, RF 2.2, RF 2.3, RF 2.4, RF 2.5 |
| **Pre-condiciones:** | Se conoce la dirección del fichero de configuración.<br><br>El fichero de configuración está editado.<br><br>Está creado manejador de elementos virtuales y el manejador de patrones de los marcadores.<br><br>Están disponibles los elementos virtuales y los ficheros de los patrones. |
| **Curso Normal de los Eventos** | |
| | **Respuesta del Sistema** |
| | 1.1- Abre el fichero de configuración de los elementos en modo “solo lectura”.<br>1.2- Lee la cantidad de bloques de asociaciones que tiene el fichero.<br>1.3- Crea una colección de entidades según la cantidad de bloques leídos.<br>1.4- Le asigna a cada entidad el valor de los datos que lee.<br>1.5- Cierra el fichero de configuración de los elementos. |
| **Curso Alternativo de los Eventos** | |
| | 1.1- Si no se puede abrir el fichero de configuración muestra un mensaje en consola diciendo que no se pudo abrir el fichero. |
| **Post-condiciones:** | Se creó una colección de entidades. |
| **Prioridad:** | Crítico. |

*Tabla 4. Expansión del caso de uso Leer configuración de objetos.*

| | |
|---|---|
| **Caso de uso:** | Mostrar escena aumentada. |
| **Actor (es):** | Usuario |
| **Propósito:** | Mostrar la escena capturada por la cámara con los elementos virtuales insertados. |
| **Resumen:** | El CU es inicializado por el actor, comienza cuando el actor pone en el área de captura de la cámara un marcador, si este es reconocido entonces se inserta el modelo virtual con sus transformaciones. Termina cuando el usuario quita el marcador del área de captura de la cámara. |
| **Referencias:** | RF 5, RF 6, RF 7, RF 8, CU Detectar marcadores, CU Calcular transformación de marcadores. |
| **Pre-condiciones:** | Se conocen los marcadores detectados en la escena.<br><br>Las transformaciones de cada marcador con respecto a la cámara están calculadas.<br><br>Se conoce la vista activa (aumentada a color, información o blanco y negro). |

| **Curso Normal de los Eventos** | |
|---|---|
| **Acción del actor.** | **Respuesta del Sistema** |
| 1. El usuario pone uno o varios marcadores en el área de captura de la cámara. | 1.1- Muestra el video capturado por la cámara.<br>1.2- Captura el próximo fotograma de video.<br>1.3- Activa la perspectiva de la cámara.<br>1.4- Calcula la posición de cada objeto virtual con respecto a la cámara.<br>1.5- Muestra cada objeto virtual en la escena real.<br>1.6- Si está activo el modo de abscisas las muestra para cada elemento virtual. |
| **Curso Alternativo de los Eventos** | |
| | 1.2- Si no está visible la entidad no calcula la posición del objeto virtual con respecto a la cámara.<br>1.3- No muestra el objeto virtual en la escena real.<br>1.4- Si no está activo el modo de abscisas no muestra las abscisas. |

| | |
|---|---|
| **Post-condiciones:** | Se localizó cada objeto virtual con respecto a la cámara.<br><br>Se mostraron los objetos virtuales correspondientes a los marcadores reconocidos. |
| **Prioridad:** | Crítico. |

*Tabla 5. Expansión del caso de uso Mostrar escena aumentada.*

| | |
|---|---|
| **Caso de uso:** | Detectar marcadores. <Inclusión> |
| **Actor (es):** | |
| **Propósito:** | Dado el fotograma actual del video capturado por la cámara, reconocer los marcadores que están en ese fotograma. |
| **Resumen:** | El CU es iniciado por el sistema, el sistema obtiene cada fotograma del video capturado por la cámara, busca los marcadores que están presentes en el fotograma y extrae información de ellos. |
| **Referencias:** | RF 3, RF 3.1, RF 3.2, RF 3.3 |
| **Pre-condiciones:** | Se capturó el fotograma actual del video.<br><br>Se conocen los patrones de los marcadores que pueden estar en la escena. |
| **Curso Normal de los Eventos** | |
| | **Respuesta del Sistema** |
| | 1.1- Obtiene el fotograma actual del video capturado por la cámara.<br>1.2- Detecta los marcadores. |
| **Curso Alternativo de los Eventos** | |
| | 1.2- Si ocurre algún error en la detección de los marcadores, muestra un mensaje y termina el CU. |
| **Post-condiciones:** | Se obtuvo fotograma actual.<br><br>Se detectaron los marcadores. |
| **Prioridad:** | Crítico. |

*Tabla 6. Expansión del caso de uso Detectar marcadores.*

| | |
|---|---|
| **Caso de uso:** | Calcular transformación de marcadores. <Inclusión> |
| **Actor (es):** | |
| **Propósito:** | Calcular la transformación de cada marcador que está presente en la escena filmada por la cámara y asignársela a su entidad correspondiente. |
| **Resumen:** | El CU es iniciado por el sistema. Conociendo la información de cada marcador detectado en la escena, se calcula la transformación de cada uno con respecto a la cámara y se le asigna a su entidad correspondiente. |
| **Referencias:** | RF 4, RF 4.1, RF 4.2 |
| **Pre-condiciones:** | Se conoce la información de cada marcador detectado en la escena. |
| **Curso Normal de los Eventos** | |
| | **Respuesta del Sistema** |
| | 1.1- Busca a que entidad pertenece cada marcador detectado.<br>1.2- Encuentra a que entidad pertenece cada marcador detectado.<br>1.3- Para cada entidad encontrada, calcula la transformación de su marcador correspondiente.<br>1.4- Informa a cada entidad la transformación de cada marcador respectivamente.<br>1.5- Activa la visibilidad de cada entidad, correspondiente a cada marcador detectado respectivamente. |
| **Curso Alternativo de los Eventos** | |
| | 1.2- No encontró a que entidad pertenece cada marcador detectado.<br>1.5- Desactiva la visibilidad de todas las entidades. |
| **Post-condiciones:** | La transformación de cada marcador detectado se le asignó a su entidad correspondiente. |
| **Prioridad:** | Crítico. |

*Tabla 7. Expansión del caso de uso Calcular transformación de marcadores.*

| | |
|---|---|
| **Caso de uso:** | Interactuar con escena aumentada. <extensión> |

| | |
|---|---|
| **Actor (es):** | Usuario |
| **Propósito:** | Cambiar la vista a color de la escena a blanco y negro o viceversa. Cambiar el valor de umbral. |
| **Resumen:** | El CU es iniciado por el usuario. Al oprimir una tecla, de acuerdo a la tecla que oprime, el sistema llevará a cabo una acción que repercute directamente en la forma en que se muestra la escena aumentada. |
| **Referencias:** | RF 12, RF 12.1,RF 12.2 |
| **Pre-condiciones:** | La escena aumentada tiene que haberse mostrado. |

| **Curso Normal de los Eventos** | |
|---|---|
| **Acción del actor.** | **Respuesta del Sistema** |
| 1. Oprime una tecla. | 1.1- Capturar tecla oprimida.<br>1.2- Cambiar a vista aumentada en blanco y negro. |
| **Curso Alternativo de los Eventos** | |
| | 1.2- Cambiar a vista aumentada en colores. |
| **Post-condiciones:** | |
| **Prioridad:** | Opcional. |

*Tabla 8. Expansión del caso de uso Interactuar con escena aumentada.*

| | |
|---|---|
| **Caso de uso:** | Interactuar con objetos virtuales. <extensión> |
| **Actor (es):** | Usuario |
| **Propósito:** | Cambiar la posición del objeto virtual con respecto al sistema de referencia del marcador. |
| **Resumen:** | El CU es iniciado por el usuario. Al oprimir una tecla, de acuerdo a la tecla que oprime, se determinara en que dirección y sentido se desplazara el objeto virtual con respecto al marcador. |
| **Referencias:** | RF 9, RF9.1 |
| **Pre-condiciones:** | La escena aumentada tiene que haberse mostrado. |

| **Curso Normal de los Eventos** | |
|---|---|
| **Acción del actor.** | **Respuesta del Sistema** |
| 1. Oprime una tecla. | 1.1- Capturar tecla oprimida.<br>1.2- Comprueba que sea una de las teclas especificadas para desplazar el objeto virtual.<br>1.3- Desplaza el objeto en la dirección y sentido seleccionado. |
| **Curso Alternativo de los Eventos** | |
| | 1.2- Si no es una tecla para desplazar el objeto virtual, comprueba que sea para activar o desactivar el modo de abscisas y lo activa o desactiva según sea el caso.<br>1.3- Si no es la tecla especificada, no se desplaza el objeto virtual. |
| **Post-condiciones:** | Se desplazó el objeto virtual.<br><br>La posición del objeto virtual con respecto al marcador se guardó en su entidad correspondiente. |
| **Prioridad:** | Opcional. |

*Tabla 9. Expansión del caso de uso Interactuar con objetos virtuales.*

| **Caso de uso:** | Mostrar información. <extensión> |
|---|---|
| **Actor (es):** | Usuario |
| **Propósito:** | Mostrar la información del marcador con respecto a la cámara y la información del objeto virtual con respecto a la cámara. |
| **Resumen:** | El CU es iniciado por el usuario. Al oprimir una tecla, se muestra información del objeto virtual y el marcador que estén en la escena. |
| **Referencias:** | RF, 10, RF 10.1, RF 10.2, RF 10.3, RF 10.4, RF 11, RF 11.1, RF 11.2 |
| **Pre-condiciones:** | Existe en la escena un marcador con un objeto virtual asociado. |
| **Curso Normal de los Eventos** | |

| **Acción del actor.** | **Respuesta del Sistema** |
|---|---|
| 1. Oprime una tecla. | 1.1- Capturar tecla oprimida.<br>1.2- Comprueba que sea la tecla especificada para mostrar la información.<br>1.3- Calcula la posición del objeto virtual con respecto a la cámara.<br>1.4- Muestra la información del marcador y el objeto virtual correspondiente. |
| **Curso Alternativo de los Eventos** | |
| | 1.3- Si no es la tecla especificada, no se calcula la posición del objeto respecto a la cámara.<br>1.4- No se muestra la información del marcador y el objeto virtual correspondiente. |
| **Post-condiciones:** | Se mostró la localización del objeto virtual y su marcador correspondiente respecto a la cámara. |
| **Prioridad:** | Secundario. |

*Tabla 10. Expansión del caso de uso Mostrar información.*

| **Caso de uso:** | Terminar |
|---|---|
| **Actor (es):** | Usuario |
| **Propósito:** | Salir de la aplicación.. |
| **Resumen:** | El CU es iniciado por el usuario. Al oprimir la tecla Esc como respuesta se ejecuta la funcionalidad que libera la memoria y los recursos ocupados por la aplicación y se termina su ejecución. |
| **Referencias:** | RF 13, RF 13.1,RF 13.2 |
| **Pre-condiciones:** | La escena aumentada tiene que haberse mostrado. |
| **Curso Normal de los Eventos** | |
| **Acción del actor.** | **Respuesta del Sistema** |
| 2. Oprime la tecla.Esc. | 1.3- Capturar tecla oprimida. |

| | |
|---|---|
| | 1.4- Termina la aplicación.. |
| **Curso Alternativo de los Eventos** | |
| | |
| **Post-condiciones:** | |
| **Prioridad:** | Critico. |

*Tabla 11. Expansión del caso de uso Terminar.*

### 3.9 Conclusiones parciales del capítulo.

En este capítulo se describió lo que el sistema debe ser capaz de hacer. Para ello se modelaron los principales conceptos, se establecieron las capacidades o funcionalidades que el sistema debe cumplir y las propiedades o cualidades que el producto debe tener, es decir, los requisitos funcionales y no funcionales. Se agruparon los requisitos funcionales en casos de uso y estos últimos se describieron para un mejor entendimiento de los procesos que tendrán lugar en el producto desarrollado.

## CAPÍTULO 4 DISEÑO E IMPLEMENTACIÓN

En este capítulo se tratan los aspectos fundamentales del diseño e implementación de la solución para la elaboración del prototipo funcional.

### 4.1 Diagrama de clases de diseño.

En la figura 13 se muestra como quedó el diagrama de clases del diseño el cual resultó ser el punto de partida para comenzar a elaborar la solución en términos del lenguaje de programación seleccionado. Para restar un poco de complejidad al diagrama se decidió tener en cuenta solamente los atributos de las clases. Una descripción un poco más detallada de las clases que resultaron ser más importantes para la solución se encuentra en el epígrafe 1.3.

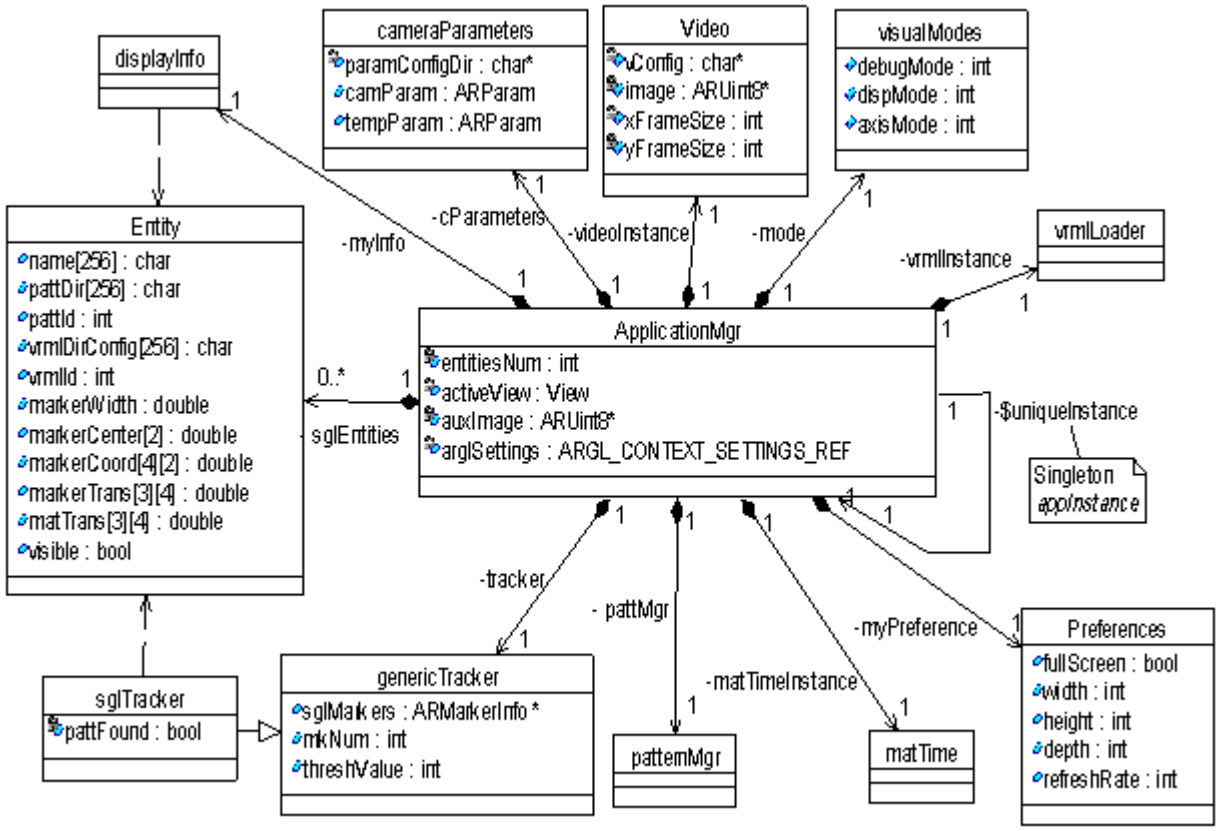



*Figura 13. Diagrama de clases del diseño.*

### 4.2 Descripción de clases de diseño

Una vez visto el diagrama de clases, a continuación se proporcionará una descripción de los atributos y métodos más significativos de algunas clases que se consideran importantes para la elaboración de la solución.

| **Nombre de la Clase:** patternMgr | |
|---|---|
| **Tipo de clase:** Controladora. | |
| **Responsabilidades de la clase:** | |
| Nombre: | loadPattern() |
| Descripción: | Carga un patrón de marcador en memoria. Para ello se le debe dar la dirección del fichero que contiene el mapa de bits que representa la región ocupada por el marcador en una imagen, sin aplicarle ningún tipo de transformación. Devuelve un identificador del patrón cargado. |

*Tabla 12. Descripción de la clase de diseño patternMgr.*

| **Nombre de la Clase:** matTime | |
|---|---|
| **Tipo de clase:** Controladora. | |
| **Responsabilidades de la clase:** | |
| Nombre: | inverseMatrix() |
| Descripción: | Dada una matriz de 3x4 devuelva la inversa. |
| Nombre: | mulMatrix() |
| Descripción: | Dadas 2 matrices de 3x4 las convierte en matrices de 4x4 añadiendo coordenadas homogéneas, las multiplica y devuelve su resultado sin las componentes homogéneas en una matriz resultante de 3x4. |
| Nombre: | getDistance() |
| Descripción: | Devuelve la distancia de un punto (x,y,z) al centro de coordenadas, como la norma del vector representado por los puntos del origen y (x,y,z). |
| Nombre: | sleep() |
| Descripción: | Dado un tiempo en milisegundos, manda a dormir el hilo de ejecución actual por ese tiempo. |

*Tabla 13. Descripción de la clase de diseño matTime.*

| **Nombre de la Clase:** applicationMgr | |
|---|---|
| **Tipo de clase:** Controladora | |
| **Atributo** | **Tipo** |
| entitiesNum | Int |
| activeView | View |
| auxImage | ARUint8* |
| arglSettings | ARGL_CONTEXT_SETTINGS_REF |
| sglEntities | Entity* |
| ideoInstance | Video |
| cParameters | cameraParameters |
| pattMgr | patternMgr |
| tracker | genericTracker* |
| vrmlInstance | vrmlLoader |
| matTimeInstance | matTime |
| myInfo | displayInfo |
| mode | visualModes |
| myPreference | Preferences |
| **Responsabilidades de la clase:** | |
| Nombre: | appInstance() |
| Descripción: | Es un método de clase, cuya función es devolver una instancia única de esta clase, de forma tal que no se creen múltiples instancias de la clase principal. Este es el método que implementa el Singleton |
| Nombre: | init() |
| Descripción: | Inicializa la aplicación con la información del video, los parámetros de la cámara, los marcadores que se van a reconocer y lo modelos que se van a dibujar. También inicia la captura de video. |
| Nombre: | idleSimulation() |
| Descripción: | Ejecuta constantemente la detección de los marcadores, así como el cálculo de sus transformaciones con respecto a la cámara. |
| Nombre: | display() |
| Descripción: | Se encarga de las tareas de dibujado según OpenGL. |
| Nombre: | keyboardEvent() |

| | |
|---|---|
| Descripción: | Controla los eventos de teclado. |
| Nombre: | readSingleData() |
| Descripción: | Se encarga de leer el fichero de configuración donde se encuentra la especificación de los marcadores y los objetos 3D que van a aumentar la escena. |
| Nombre: | setUpOpenGLContext() |
| Descripción: | Inicializa el contexto de OpenGL para dibujar la escena aumentada. |
| Nombre: | drawAgumentedScene() |
| Descripción: | Dibuja la escena aumentada. |
| Nombre: | drawImage() |
| Descripción: | Dibuja las imágenes que componen el video capturado porla cámara. |
| Nombre: | run() |
| Descripción: | Método principal de la clase, es donde se ejecutan las inicializaciones y se definen las funciones callbacks de OpenGL. |

*Tabla 14. Descripción de la clase de diseño applicationMgr.*

| **Nombre de la Clase:** Entity | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributo** | **Tipo** |
| name[256] | Char |
| pattDir[256] | Char |
| pattId | Int |
| vrmlDirConfig[256] | Char |
| vrmlId | Int |
| markerWidth | double |
| markerCenter[2] | double |
| markerCoord[4][2] | double |
| markerTrans[3][4] | double |
| matTrans[3][4] | double |
| Visible | Bool |

*Tabla 15. Descripción de la clase del diseño Entity.*

A la clase especificada en la tabla 15 no se le implementó ninguna funcionalidad debido a que su único propósito es el de guardar datos que van a utilizar otras clases para aumentar la escena. Esta clase

está relacionada directamente con los bloques de información que se definen en el fichero de configuración de los modelos y marcadores que van a aumentar la escena. En los diagramas de secuencia que se verán en epígrafes posteriores, se le colocaron mensajes como funciones de esta clase con el objetivo de esclarecer los dichos diagramas.

| **Nombre de la Clase**: sglTracker | |
|---|---|
| **Tipo de clase**: Controladora | |
| **Atributo** | **Tipo** |
| pattFound | bool |
| sglMarkers | ARMarkerInfo * |
| mkNum; | Int |
| threshValue | Int |
| **Responsabilidades de la clase:** | |
| Nombre: | detectMarkers() |
| Descripción: | Es el encargado de detectar los marcadores que están presentes en el flujo de video proveniente de la cámara, recoger la información de cada uno de ellos y guardar la en sglMarkers. Este último es un arreglo de estructuras para guardar información de los marcadores detectados. |
| Nombre: | fastDetectMarkers() |
| Descripción: | Igual que el método anterior pero la detección se hace con mayor rapidez afectando la precisión de la detección. |
| Nombre: | getTransMat() |
| Descripción: | Calcula la matriz de transformación (rotación y traslación) de un marcador con respecto a la cámara. |
| Nombre: | getTransMatCont() |
| Descripción: | Igual que el anterior, pero el cálculo se realiza con mayor precisión teniendo en cuenta que el marcador haya sido detectado en el fotograma anterior. |
| Nombre: | getPattFound() |
| Descripción: | Devuelve si se ha detectado, al menos, un marcador en la escena. |
| Nombre: | setPattFound() |
| Descripción: | Cambia el estado de la detección de los marcadores: verdadero si al menos se detecto un marcador y falso en caso contrario. |

*Tabla 16. Descripción de la clase de diseño sglTracker.*

| **Nombre de la Clase:** vrmlLoader | |
|---|---|
| **Tipo de clase:** Controladora. | |
| **Responsabilidades de la clase:** | |
| Nombre: | loadVrmlFile() |
| Descripción: | Dada la dirección de un archivo VRML se carga su contenido para después ser dibujado. Devuelve un Id del modelo que se cargó. |
| Nombre: | FreeVrmlModel() |
| Descripción: | Dado el Id de un modelo que se haya cargado, el modelo se elimina de la lista de modelos disponibles y se libera la memoria que ocupaba. |
| Nombre: | drawVrmlModel() |
| Descripción: | Dibuja un modelo 3D dado el Id. |
| Nombre: | updateVrmlTimer() |
| Descripción: | Actualiza el contador interno de tiempo del modulo VRML de ARToolKit. |
| Nombre: | SetVrmlInternalLight() |
| Descripción: | Cambia la iluminación interna de los modelos VRML. |

*Tabla 17. Descripción de la clase de diseño vrmlLoader.*

| **Nombre de la Clase:** Video | |
|---|---|
| **Tipo de clase:** Controladora | |
| **Atributo** | **Tipo** |
| vConfig | char* |
| image | ARUint8* |
| xFrameSize | Int |
| yFrameSize | Int |
| **Responsabilidades de la clase:** | |
| Nombre: | setConfigFile() |
| Descripción: | Se utiliza para informarle a las instancias de la clase la dirección del fichero de configuración de donde deben tomar los datos. |
| Nombre: | openVideoInput() |
| Descripción: | Habilita la fuente de captura de video desde el dispositivo de captura, en este caso la cámara web. |
| Nombre: | closeVideoInput() |
| Descripción: | Deshabilita la fuente de captura de video. |
| Nombre: | startCapture() |
| Descripción: | Inicia la captura de video. |
| Nombre: | stopCapture() |
| Descripción: | Detiene la captura de video. |
| Nombre: | captNextFrame() |
| Descripción: | Actualiza la captura al próximo fotograma grabado. |
| Nombre: | calcVideoSize() |
| Descripción: | Calcula la resolución del video capturado por la cámara. |
| Nombre: | getXSizeFrame() |
| Descripción: | Devuelve el ancho del fotograma capturado en píxeles. |
| Nombre: | getYSizeFrame() |
| Descripción: | Devuelve el alto del fotograma capturado en píxeles. |
| Nombre: | getCurrentFrame() |
| Descripción: | Devuelve el fotograma de video más reciente capturado por la cámara. |
| Nombre: | getSavedFrame() |
| Descripción: | Cuando se llama al método anterior ese fotograma más reciente se guarda |

| | |
|---|---|
| | como un atributo de la clase. Este método devuelve ese fotograma, aún después de seguir capturándose el flujo de video, hasta que no se vuelva a llamar el método anterior. |

*Tabla 18.Descripción de la clase de diseño Video.*

| **Nombre de la Clase:** cameraParameters | |
|---|---|
| **Tipo de clase:** Controladora. | |
| **Atributo** | **Tipo** |
| paramConfigDir | char* |
| camParam | ARParam |
| tempParam | ARParam |
| **Responsabilidades de la clase:** | |
| Nombre: | setConfigFile() |
| Descripción: | Se utiliza para informarle a las instancias de la clase, la dirección del fichero de configuración de donde deben tomar los datos de la cámara. |
| Nombre: | changeParamSize() |
| Descripción: | Pone los parámetros de la cámara en función de la resolución del video. |
| Nombre: | loadParam() |
| Descripción: | Carga los parámetros de la cámara. |
| Nombre: | showParam() |
| Descripción: | Muestra los parámetros de la cámara. |
| Nombre: | initCParam() |
| Descripción: | Inicializa los parámetros de la cámara. |

*Tabla 19. Descripción de la clase de diseño camaraParameters.*

### 4.3 Diagramas de secuencia.

En este epígrafe se muestran los diagramas de secuencia correspondientes a los casos de uso descritos anteriormente en el capítulo 3. Estos diagramas representan los eventos generados por el actor, su orden y los eventos internos del sistema. Para lograr claridad en los diagramas se detallaron los eventos de mayor importancia.

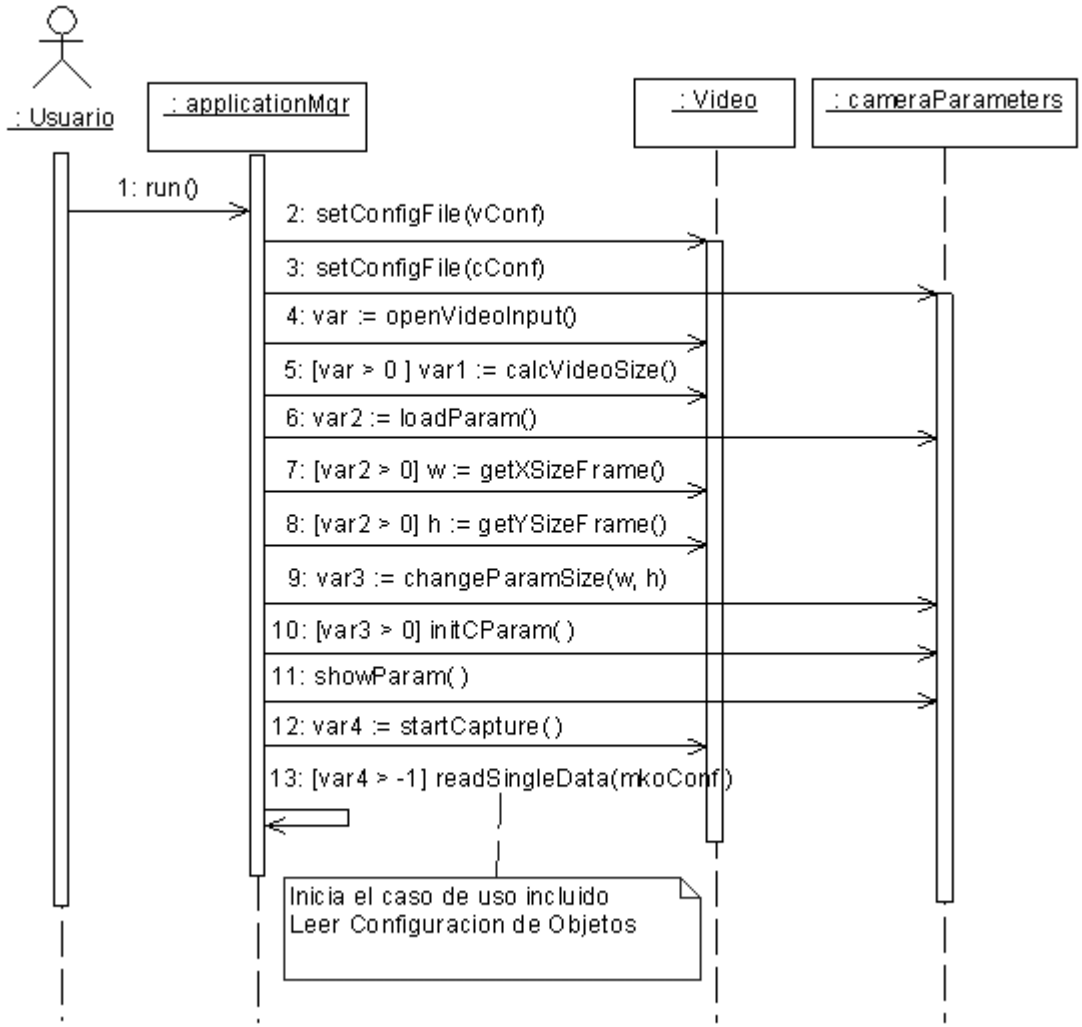



*Figura 14. Diagrama de secuencia del diseño. Inicializar.*

La figura 14 muestra el diagrama de secuencia del caso de uso Inicializar en el cual se habilita la fuente de captura de video, es decir, la cámara; si esta acción se efectúa satisfactoriamente se calcula la resolución del video y se cargan los parámetros de la cámara. Se adaptan los parámetros de la cámara según la resolución del video, se muestran y comienza la captura de video. Al final se lee el fichero de configuración de objetos (Ver descripción de la figura 15).

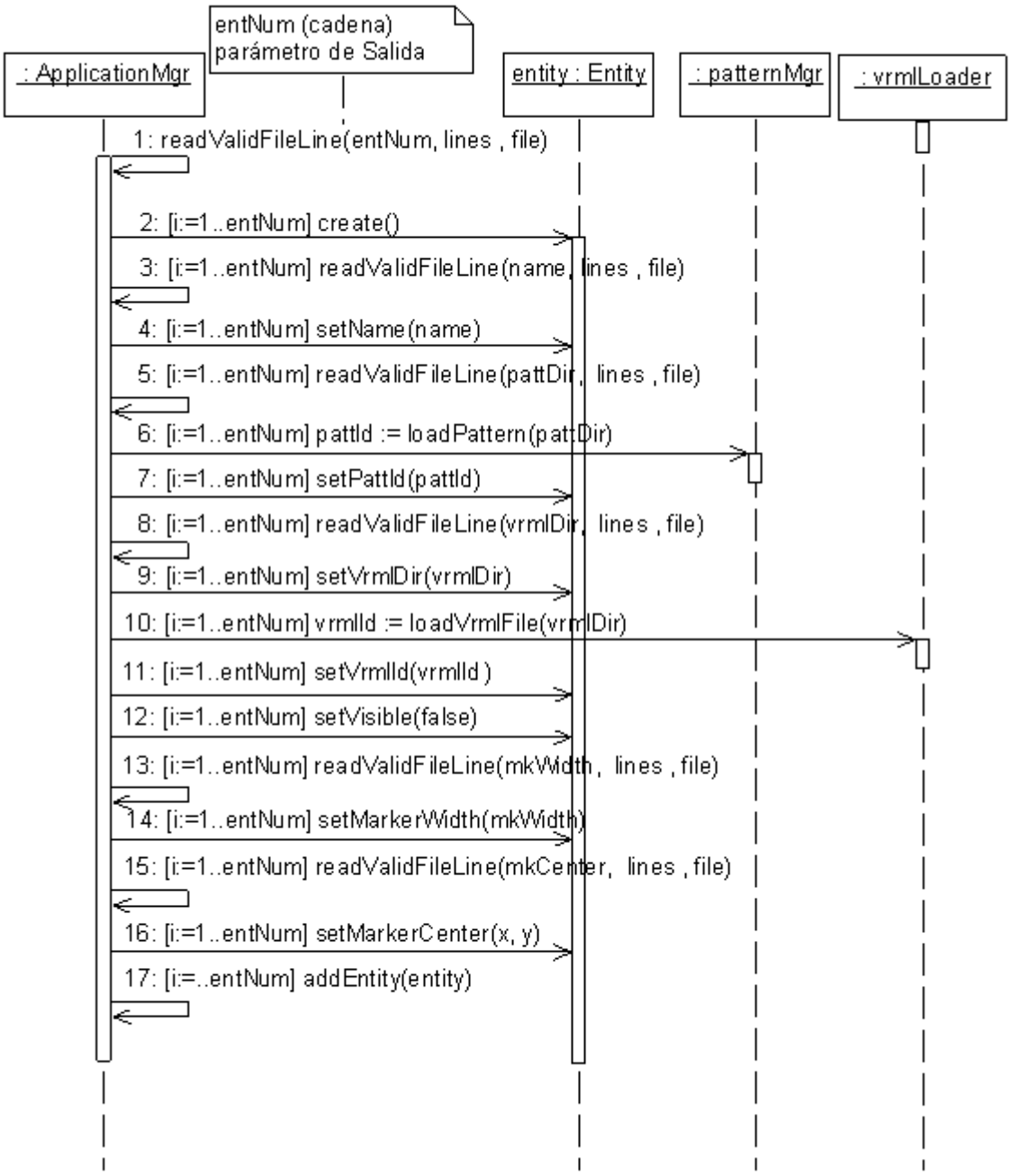



*Figura 15. Diagrama de secuencia del diseño. Leer configuración de objetos.*

En el diagrama de la figura 15 se muestra el flujo de los eventos del caso de uso Leer configuración de objetos, en el cual se lee la información relacionada con los objetos 3D que va a aparecer en la escena aumentada y los marcadores que van a ser reconocidos. Primeramente se lee la primera línea que

contiene el número de bloques de asociaciones patrón-objeto 3D. Luego por cada uno de esos bloques se lee el nombre del bloque; se lee la dirección del patrón y se carga el patrón en memoria; se lee la dirección del modelo VRML y se carga en memoria; se lee el tamaño del patrón y la ubicación de su centro de coordenadas. Con los datos leídos anteriormente se crean tantas entidades como bloques se hayan leído y se ponen todas invisibles inicialmente.

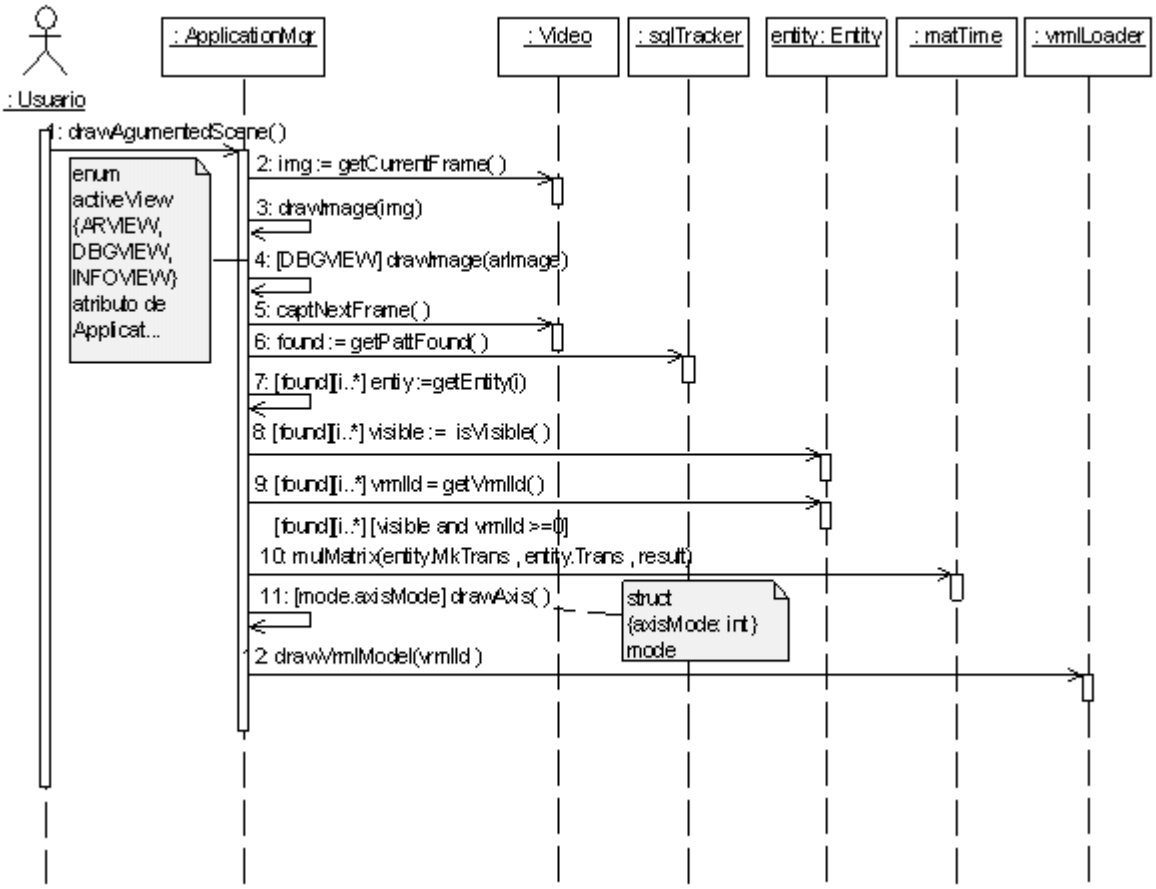



*Figura 16. Diagrama de secuencia del diseño. Mostrar escena aumentada.*

La figura 16 muestra el diagrama de secuencia del caso de uso Mostrar escena aumentada, en el cual se dibuja el fotograma de video obtenido y si la vista activa está en blanco y negro se dibuja el fotograma en blanco y negro. Luego se manda a capturar el próximo fotograma de video, si se detectó al menos un marcador en la escena entonces por cada entidad visible correspondiente al marcador detectado se obtiene su matriz de transformación con respecto a la cámara, si está activo el modo de las abscisas estas se dibujan y finalmente se dibuja el modelo virtual 3D.

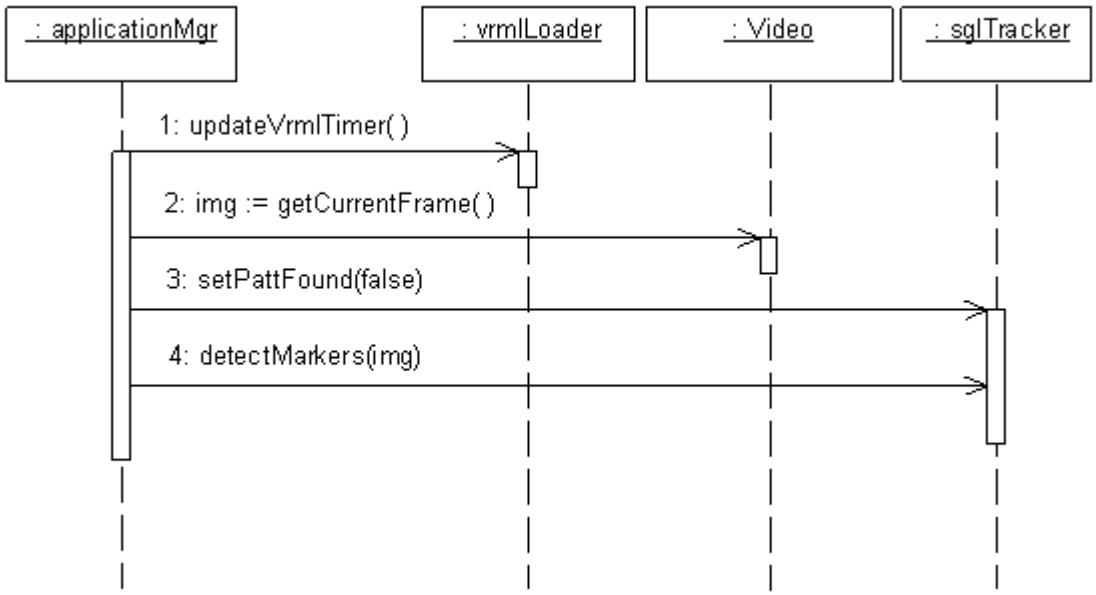



*Figura 17. Diagrama de secuencia del diseño. Detectar marcador.*

En la figura 17 se observa el diagrama de secuencia correspondiente al caso de uso detectar marcador. Primeramente para la detección se actualiza el temporizador de VRML, luego se obtiene el fotograma actual, se le dice al *tracker* que no se ha detectado ningún marcador y se efectúa la detección de los marcadores.

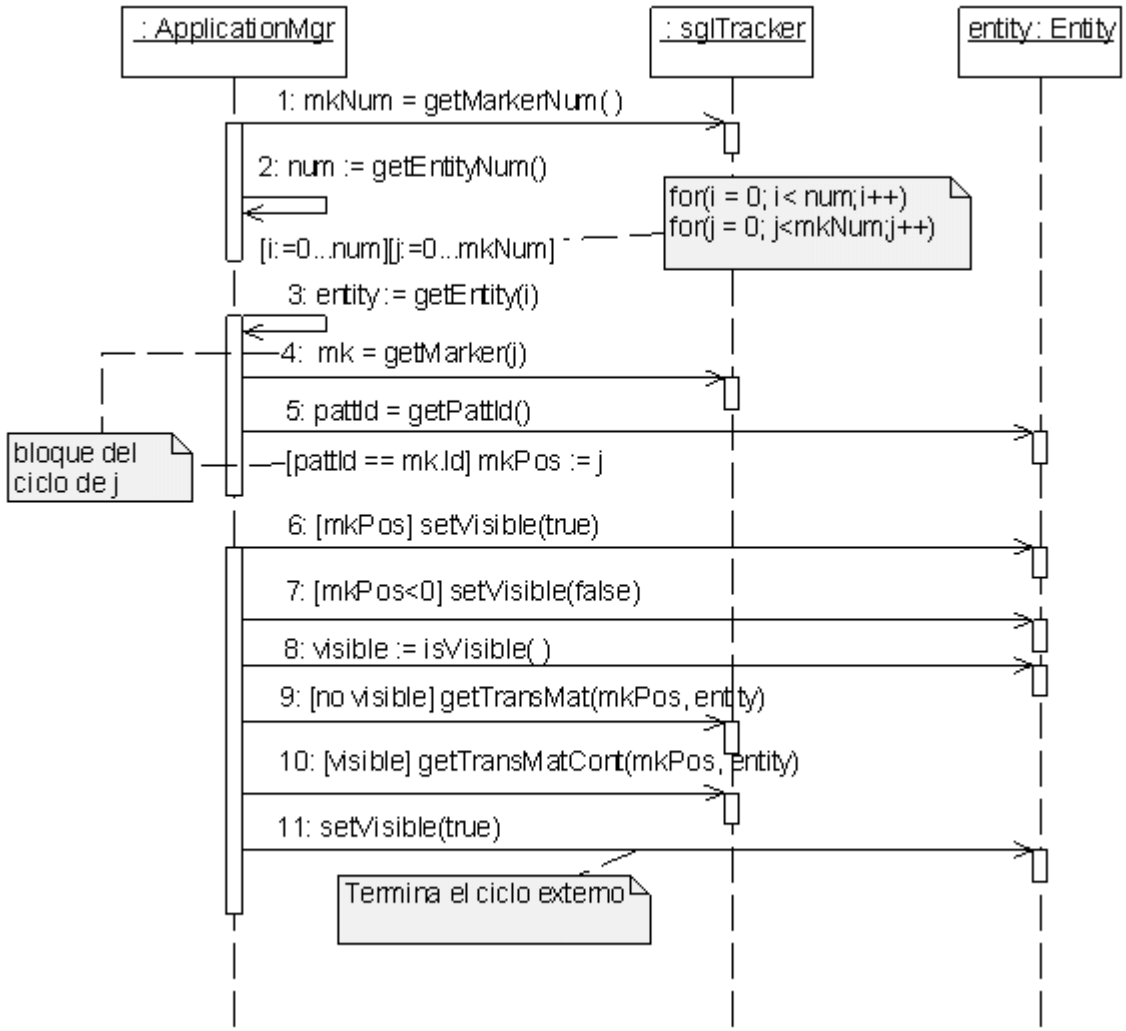



*Figura 18. Diagrama de secuencia del diseño. Calcular Transformación de marcadores.*

En el diagrama de secuencia de la figura 18 se representa el proceso donde se calculan las transformaciones de los marcadores. Por cada entidad se busca entre los marcadores detectados cual le corresponde. Una vez determinado el marcador "j" correspondiente a la entidad "i" se le pregunta a la entidad si ese mismo marcador estuvo visible en el fotograma anterior, en caso de que no estuviera visible se calcula su transformación sin utilizar la transformación anterior, en caso contrario se utiliza la información anterior para un mayor grado de certeza y se pone la entidad visible. En caso de que no se encuentre el marcador correspondiente a la entidad, esta se pone invisible y se pasa a la siguiente entidad.

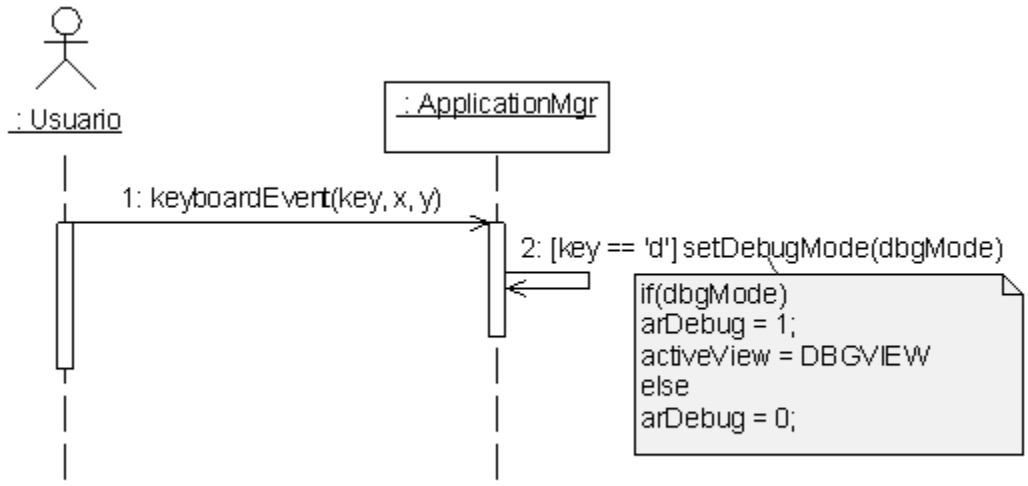



*Figura 19. Diagrama de secuencia del diseño. Interactuar con escena aumentada.*

El diagrama de secuencia representado en la figura 19 corresponde al caso de uso Interactuar con escena aumentada. En este el usuario en este si el usuario oprime la tecla ‘d’ para mostrar la vista en blanco y negro, se pasa de la vista en blanco y negro a la vista en colores y viceversa, en dependencia del valor de ‘dbgMode’.

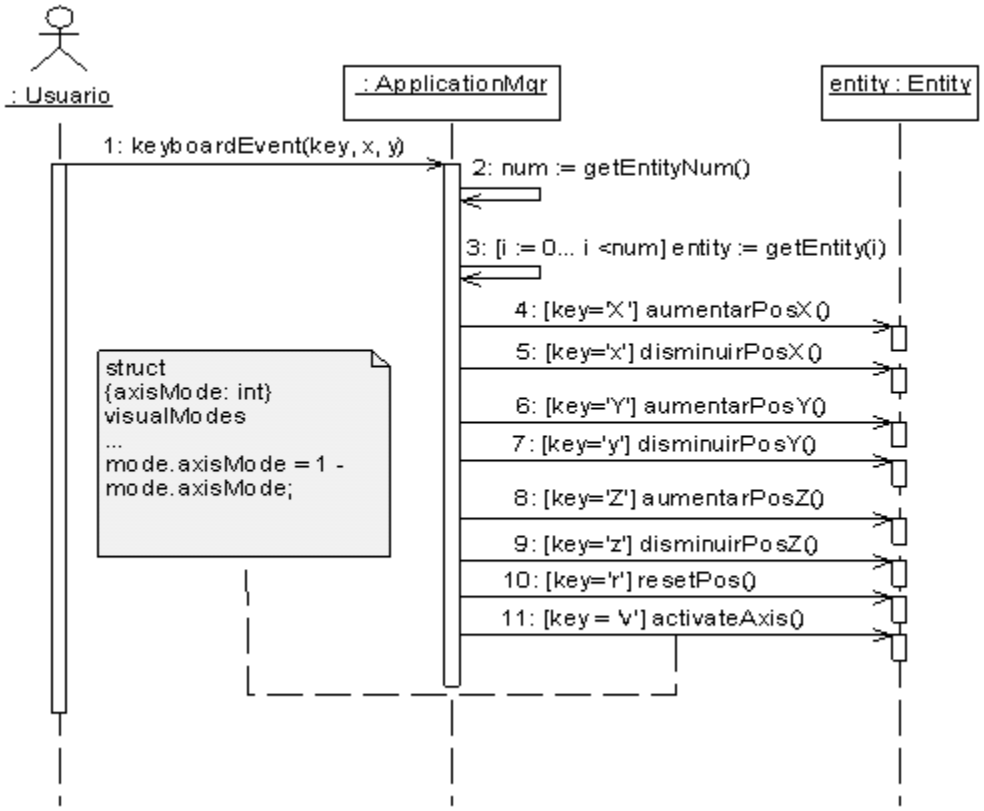



*Figura 20. Diagrama de secuencia del diseño. Interactuar con objetos virtuales.*

En el diagrama de secuencia representado en la figura 20 se observa el flujo de los eventos del caso de uso Interactuar con objetos virtuales. El usuario oprime una tecla y genera un evento de teclado. En dependencia de la tecla que se especifica (‘x’, ’y’ o ’z’) será el eje en el que se trasladaran los objetos virtuales presentes en la escena. También en dependencia de la capitalización de la tecla que se oprimió (mayúscula o minúscula) será el sentido del movimiento (positivo o negativo respectivamente).

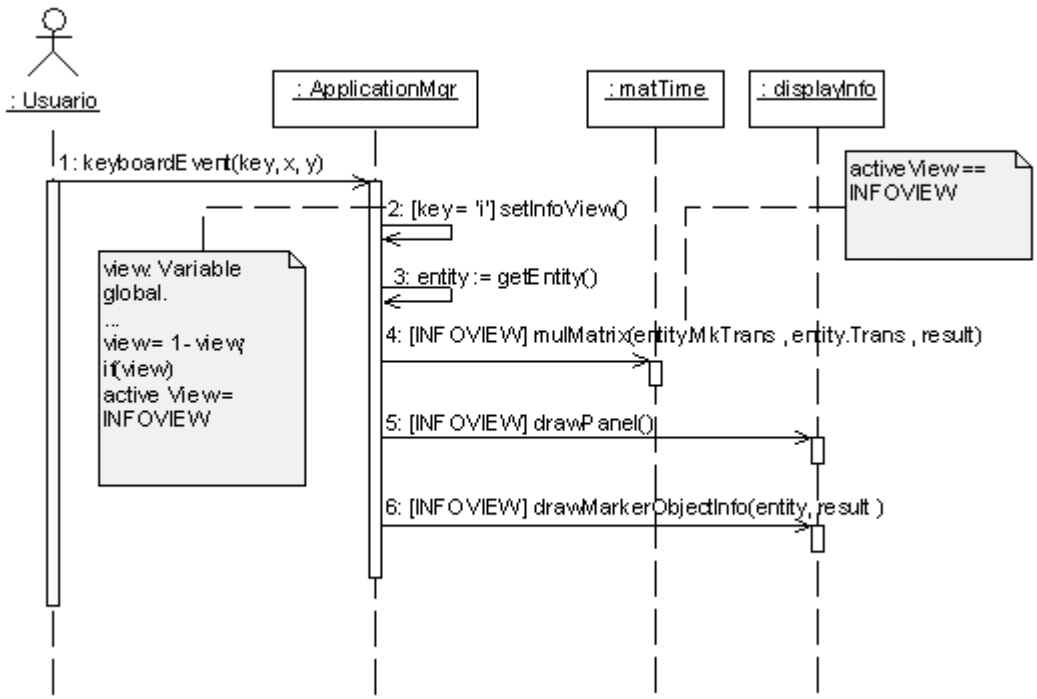



*Figura 21. Diagrama de secuencia del diseño. Mostrar información.*

La figura 21 muestra el diagrama de secuencia del caso de uso Mostrar información. El actor oprime una tecla, en caso de que sea la tecla 'i' se activa la vista de información se le piden los datos del marcador y el elemento virtual que se está dibujando en la escena y se muestran.

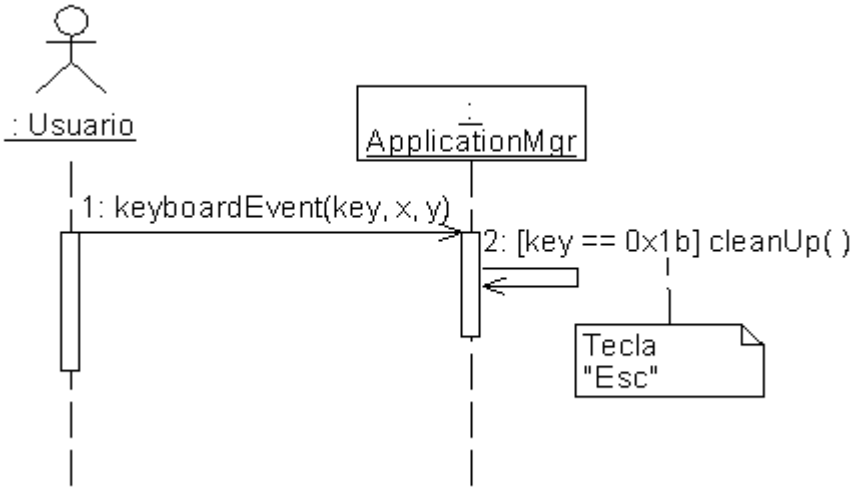



*Figura 22. Diagrama de secuencia del diseño. Terminar*

En la figura 22 se observa la secuencia de las acciones que tienen lugar cuando el usuario decide terminar la ejecución de la aplicación. Primeramente el usuario genera un evento de teclado presionando la tecla “Esc”, como respuesta se ejecuta la funcionalidad que libera la memoria y los recursos ocupados por la aplicación y se termina su ejecución.

### 4.4 Diagrama de componentes.

Los componentes utilizados para la elaboración de la aplicación se agruparon en paquetes como se observa en la figura 23. El paquete LETRA es el que agrupa los componentes definidos para la elaboración del prototipo funcional, en el paquete ARToolKit están los componentes específicos de dicha librería y en el paquete GLUT están los componentes utilizados para los gráficos.

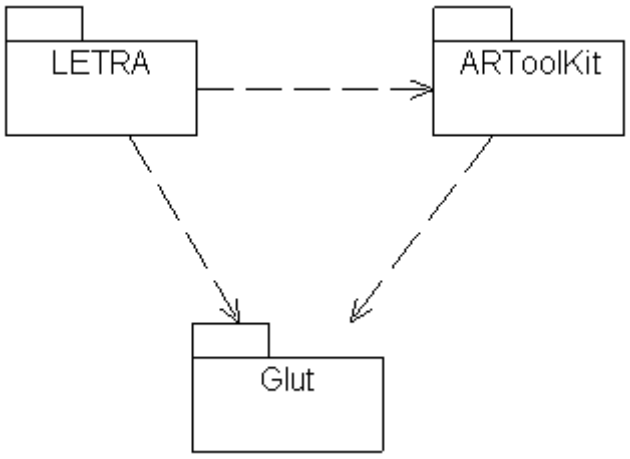



*Figura 23. Diagrama de paquetes.*

La transformación de las clases de la lógica a su forma física se hizo mediante la utilización de componentes. Los componentes especificados en el lenguaje de programación C++ quedaron como se muestra en la figura 24.

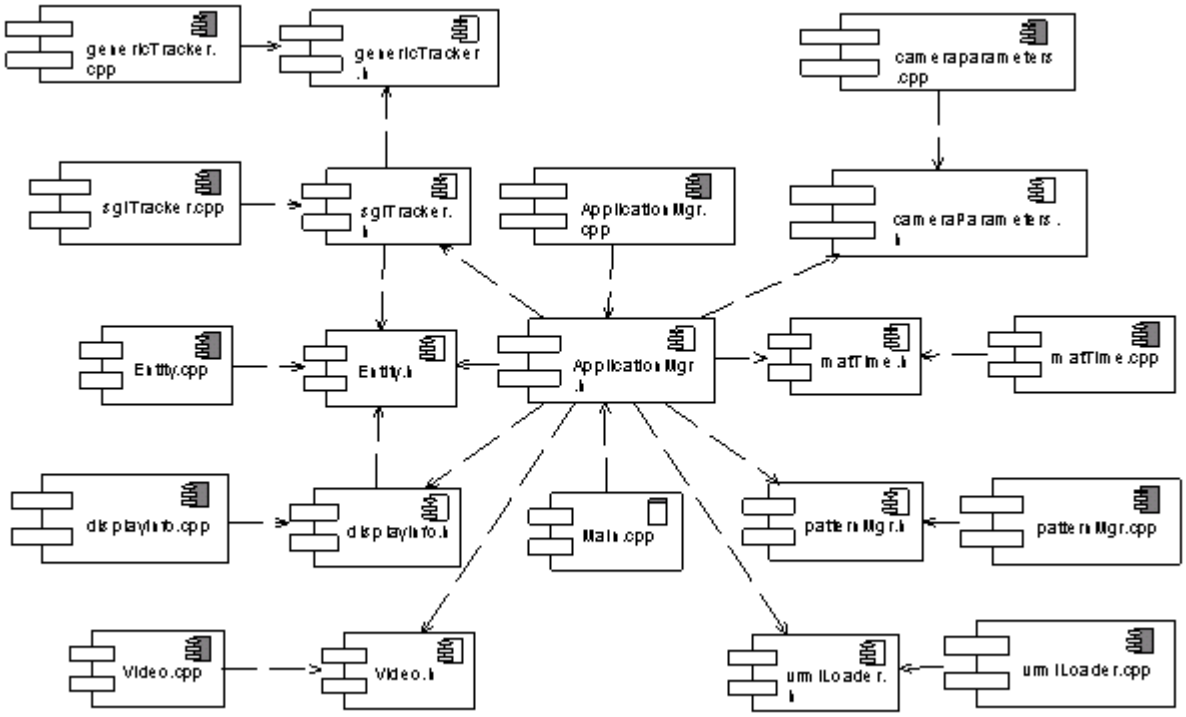



*Figura 24. Diagrama de componentes.*

### 4.5 Diagrama de despliegue.

En la figura 25 se muestra el diagrama de despliegue. Aunque este es un diagrama sencillo los autores de esta tesis decidieron incluirlo para enfatizar en la necesidad del uso de una cámara conectada a la computadora para el correcto funcionamiento del producto elaborado.

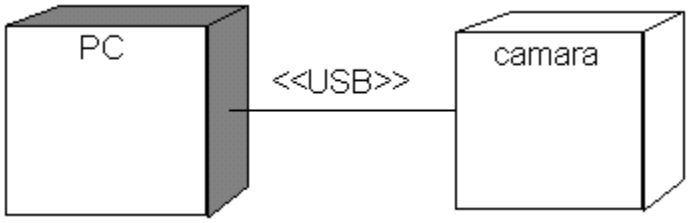



*Figura 25. Diagrama de despliegue.*

### 4.6 Conclusiones parciales del capítulo.

En este capítulo se realizó el diseño e implementación del prototipo funcional. Se obtuvo el diagrama de clases del diseño, las cuales fueron descritas posteriormente con sus atributos y responsabilidades principales para lograr un mayor entendimiento. Por último se obtuvieron los diagramas de componente y de despliegue correspondiente al flujo de trabajo de implementación, lo cual facilitó la codificación de la solución.

## CONCLUSIONES

Durante la realización de esta tesis la revisión de la bibliografía existente sobre la Realidad Aumentada, permitió a sus autores conocer el estado del arte de esta tecnología y como ha evolucionado. El análisis de los principales dispositivos existentes para realizar aplicaciones de Realidad Aumentada, condujo a que se planteara la utilización de un dispositivo de Realidad Aumentada basado en monitor, el cual pudo ser reproducido con los recursos existentes en la Universidad de las Ciencias Informáticas. También se estudiaron diferentes frameworks y algoritmos de tracking, lo cual permitió seleccionar los más adecuados respectivamente. Estos, conjuntamente con los lenguajes de programación, metodología y herramientas de desarrollo utilizadas, permitieron a los autores de este trabajo elaborar una propuesta de solución al problema planteado al inicio de la investigación. Con la implementación de la propuesta a través de una aplicación demostrativa, se pudo comprobar su validez, mostrando a los objetos virtuales localizados con respecto a la cámara real con una precisión milimétrica, lo cual cumple con el objetivo de la investigación.

## RECOMENDACIONES

Una vez concluida la investigación con un resultado satisfactorio, se recomienda que este resultado se utilice para la elaboración de un sistema de realidad aumentada, debido a que constituye el punto de partida para su realización. También se recomienda:

- Migrar la solución propuesta a la plataforma libre GNU\Linux.
- Incorporar otros formatos para la representación de los objetos virtuales como el X3D.
- Elaborar una interfaz gráfica de usuario para facilitar la edición del fichero de configuración de los marcadores y elementos virtuales.
- Generalizar el trabajo con la Realidad Aumentada.
- Realizar investigaciones con vista a elaborar algún producto que utilice la Realidad Aumentada.

## REFERENCIAS BIBLIOGRÁFICAS


[ALONSO and DURÁN, 2007]: Alonso and Mayra Duran Benejam. *Aplicación de la Realidad Aumentada en el proceso de formación del universitario*. UCIENCIA 2007, (2007) [Consultado en: 18-3-2008].

[AZUMA, 1997]: Azuma, Ronald. *A Survey of Augmented Reality*. Presence: Teleoperators and Virtual Environments [En línea] Vol. 6 no. 4 (1997) [Consultado en: 12-12-2007]. Disponible en: http://www.cs.unc.edu/~azuma/ARpresence.pdf

[AZUMA *et al,* 2001]: Azuma ,Ronald,Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, Blair MacIntyre. *Recent Advances in Augmented Reality*. IEEE Computer Graphics and Applications [En línea] Vol. 21 no. 6 (2001) [Consultado en: 16-11-2007]. Disponible en: http://www.cs.unc.edu/~azuma/cga2001.pdf

[BOOCH 1998]: BOOCH. *El Lenguaje Unificado de Modelado*. Addison Wesley, 1998. p.

[BORRO and LARDIZÁBAL 2005]:Borro, Diego, Manuel Lardizábal. *Introducción a la Realidad Aumentada: Aplicaciones en la Industria*. [En línea] (2005) [Consultado en: 8-3-2008]. Disponible en: www.ceit.es/mechanics/people/cv/dborro/Papers/RA%20(Colombia).pdf

[BRUEGGE and KLINKER 2003]: Bruegge, Bernd and Gudrun Klinker. *DWARF Distributed Wearable Augmented Reality Framework* . Technische Universitat Munchen, Chair for Applied Software Engineering [En línea] (2003) [Consultado en: 8-3-2008]. Disponible en: http://campar.in.tum.de/Chair/DwarfWhitePaper

[CHEN *et al*, 2005]: CHEN; CHEN; YUANandZHANG. *Multi-Stereo Vision Tracking for AR System*. IEEE International Conference on Information Acquisition (ICIA) [En línea] Vol. 00 (2005) [Consultado en: 26-2-2008]. Disponible en: http://csdl.computer.org/dl/proceedings/icia/2006/0528/00/04097953.pdf

[DAVISON, 2003]: Davison, Andrew J., *Real-Time Simultaneous Localization and Mapping with a Single Camera*. Ninth IEEE International Conference on Computer Vision (ICCV'03) [En línea] Vol. 2 (2003) [Consultado en: 26-2-2008]. Disponible en: http://csdl.computer.org/dl/proceedings/iccv/2003/1950/02/195021403.pdf

[FUA and LEPETIT 2007]: Fua, Pascal and Vincent Lepetit. *Vision Based 3D Tracking and Pose Estimation for Mixed Reality*. en: *Emerging Technologies of Augmented Reality, Interfaces and Design.* HALLER, M.*et al*, Idea Group Publishing, 2007. 1-22.p.

[GARCÍA 2003]: García García, Juan Carlos. *Posicionamiento 3D (2D) mediante una sola imagen de una Marca Artificial*. [En línea] (2003) [Consultado en: 21-02-2008]. Disponible en: http://www.depeca.uah.es/docencia/doctorado/cursos04_05/82858/Recupera3D.pdf

[GMBH 2005]: GMBH 2005. *Advanced Realtime Tracking*, 2005. [2008]. Disponible en: http://ar-tracking.eu/A-R-T-setup.143.0.htm

[HITLAB, 2006] *ARToolKit Augmented Reality ToolKit.*[Consultado en: 4-3-2008]. Disponible en: http://www.hitl.washington.edu/artoolkit

[IZNAGA and PÉREZ, 2006]. Iznaga Benítez, Arsenio M. and Ivan Pérez Mallea. *Fundamentos de la Gráfica por Computadora.*, 2006.

[JURIE, 1998]: Jurie, F. *Tracking objects with a recognition algorithm. Pattern Recognition Letters.* Vol. 3-4 no. 19 (1998) [Consultado en: 8-3-2008].

[KOLLER *et al*,1997]: KOLLER,D. G. Klinker; E. Rose; D.E. Breen, R.T Whitaker and M. TUCERYAN. *Real-time Vision-based camera tracking for augmented reality applications*. Proceedings of the ACM Symposium on Virtual Reality Software and Technology [En línea] Vol. no. (1997) [Consultado en: 10-3-2008].

[LOWE, 1991]: Lowe, D.G. *Fitting parameterized three-dimensional models to images*. IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 13 no. 5 (1991)

[MILGRAM and KISHINO, 1994]: Milgram, Paul., Kishino, Fumio. A Taxonomy of Mixed Reality Visual Displays. IEICE Transactions in Information Systems [En línea] Vol. E77-D no. 12 (1994) [Consultado en: 5 de febrero de 2008]. Disponible en: http://web.cs.wpi.edu/~gogo/hive/papers/Milgram_IEICE_1994.pdf

[MÜNCHEN 2005]: MÜNCHEN 2005. *Tracking Subsystem ARTTracker*, 2005. [2008]. Disponible en: http://www.ar-tracking.de/viewtopic.php?t=9

[NEIDER and DAVIS, 1997]: Neider, Jackie and Tom Davis. *OpenGL Programming Guide*. II. Addison-

Wesley Publishing Company, 1997. 616 p.

[PARK 2005]: Park, Jong Seung 2005. *Augmented Reality*, 2005 [2008]. [Disponible en: http://ecl.incheon.ac.kr/courses/ar5/index.html

[PARK and PARK 2004]: Park, Hanhoon. and Jong-Il Park. *Invisible Marker Tracking for AR*. Third IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'04) [En línea] Vol. 00 no. (2004) [Consultado en: 26-2-2008]. Disponible en: http://csdl.computer.org/comp/proceedings/ismar/2004/2191/00/21910272.pdf

[SABIA, 2007]: SABIA 2007. *Conociendo X3D*, Dpto. Tecnologías de la Información y las Comunicaciones. Universidad da Coruña., 2007. [2007]. Disponible en: http://sabia.tic.udc.es/gc/Contenidos%20adicionales/trabajos/3D/x3d/Conociendo%20X3D.htm

[SCHMALSTIEG *et al*, 2002]: Schmalstieg, D., A. Fuhrmann, G. Hesina, Z. Szalavári, LM, Encarnação, M. Gervautz and W. Purgathofer. *The Studierstube Augmented Reality Project*. [En línea] Vol. 2 no. 1 (2002) [Consultado en: 6-3-2008]. Disponible en: http://www.ims.tuwien.ac.at/media/documents/publications/schmalstieg_studierstube.pdf

[SCHWALD et al, 2004]: Schwald, Bernd., Helmut Seibert and Michael Schnaider. *A Flexible Tracking Concept Applied to Medical Scenarios Using an AR Window*. International Symposium on Mixed and Augmented Reality (ISMAR'02), IEEE Computer Society, 2002. 261-262 p. 0-7695-1781-1

[TROGEMANN *et al,* 1998]: Trogemman, G., B. Graffmann and J. PIESK. 3D-Tracking: Image Based Reconstruction of Camera Parameters for Mixed Reality Postproduction. [En línea] (1998) [Consultado en: 12-2-2008]. Disponible en: http://www.khm.de/

[VALLINO 1998]: Vallino, James. *Interactive Augmented Reality*. Department of Computer Science. New York, University of Rochester, [En línea] (1998) [Consultado en: 7-3-2008], [Disponible en: www.se.rit.edu/~jrv/publications/VallinoThesis.pdf ]. 108. p.

[WANG *et al,* 2006]: Wang, Yongtian., Yu Li, Jing Chen; Wenze Hu and Xiaojun Zang. An Improved Real-Time Natural Feature Tracking Algorithm for AR Application. 16th International Conference on Artificial Reality and Telexistence--Workshops (ICAT'06) (2006) [Consultado en: 26-2-2008]. Disponible en: http://csdl.computer.org/dl/proceedings/icat/2006/2754/00/27540119.pdf

## BIBLIOGRAFÍA CONSULTADA


[BIMBER and RASKAR 2005]: Bimber, Oliver and Ramesh RASKAR. *Spatial Augmented Reality*. A K Peters, Ltd., 2005. p. 1-56881-230-2

[CENATAV, 2004]: CENATAV 2004. *Centro de Aplicaciones de Tecnología Avanzada*, 2004. [2008]. Disponible en: http://www.cenatav.co.cu

[CONSORTIUM, 2008]: Web 3D CONSORTIUM. *What is X3D?*,[2008]. Disponible en: http://www.web3d.org/about/overview

*Designers' Augmented Reality Toolkit*.[Consultado en: 7-3-2008]. Disponible en: http://www.cc.gatech.edu/dart/aboutdart.htm

[GENC *et al*, 2002]: GENC, Y. S. Riedel; F. Souvannavong, C. Akinlar and N. Navab. *Marker-less Tracking for AR: A Learning-Based Approach*.(2002) [Consultado en: 20-01-2008]. Disponible en: http://ieeexplore.ieee.org/iel5/8184/24594/01115122.pdf?tp=&isnumber=24594&arnumber=1115122

[JACOBSON *et al*, 1999]: JACOBSON, Ivar., Grady Booch and James Rumbaugh. *El proceso unificado de desarrollo de software*. 1999. p. *Object Technology Series.*

[LEE *et al*, 2002]: LEE, Jong Weon. Suya YOU and Ulrich Neumann. *Tracking with Omni-Directional Vision for Outdoor AR Systems*. International Simposium of Mixed and Augmented Reality (ISMAR'02), IEEE Computer Society, 2002. 47 p. 0-7695-1781-1

[MÜNCHEN, 2005]: MÜNCHEN 2005. *DWARF Documentation*, 2005. [2008]. Disponible en: http://campar.in.tum.de/view/Chair/DwarfDocumentation#Tracking_Subsystem

[RATIONAL 2003]: RATIONAL 2003. *IBM - Rational Rose Enterprise - Software*, 2003. [2007]. Disponible en: http://www-306.ibm.com/software/awdtools/developer/rose/enterprise

[SCHWALD *et al*, 2004]: Bernd Schwald; Helmut Seibert and Michael Schnaider. *Composing 6 DOF Tracking Systems for VR/AR*. Computer Graphics International 2004 (CGI'04) [En línea] Vol. 00 (2004) [Consultado en: 26-2-2008]. Disponible en: http://csdl.computer.org/dl/proceedings/cgi/2004/2171/00/21710411.pdf

[SUTHERLAND 1968]: SUTHERLAND, I. *A Head-Mounted Three-Dimensional Display*. en: *Fall Joint Computer Conf.,Am. Federation of Information.Processing Soc. (AFIPS)*. Washington DC, Thompson Books, 1968. 757-764.p.

[SZALAVÁRI *et al*, 1998]: Szalavári, Z., A. Fuhrmann, D. Schmalstieg and M. Gervautz. *Studierstube: An environment for collaboration in augmented reality*. Virtual Reality Systems, Development and Applications [En línea] Vol. 2 no. 1 (1998).

## ANEXOS

### Anexo 1: Entrevista a Philip Lamb.

**P: What kind of vision algorithm is used?**
If it is inside of edge-based method (RAPID or other); Texture Based Method (optical flow, template matching or interest points); tracking by detection or just a combination. (if there is any mistake, please correct me).

**R:** It uses a combination of approaches, including feature detection and template matching. You can read about ARToolKit's methods in these papers:
http://www.hitl.washington.edu/artoolkit/Papers/IWAR99.kato.pdf
http://www.hitl.washington.edu/artoolkit/Papers/ART02-Tutorial.pdf

**P: Why you decided to use this algorithm?**

**R:** ARToolKit was originally developed by Professor Kato, so you would have to ask him. It might be useful to also read some of the early archives of the ARToolKit mailing list. (See http://www.hitl.washington.edu/artoolkit/community/)

**P: Do you recommend the use of ARTK in my final college project over other AR Systems like StudierStube, DWARF, DART? Why?**

**R:** Yes. Firstly, because it is easy to learn, with a simple C-based interface. Secondly, it is well documented. Thirdly, it has a path to a commercial version if you should choose to make a product out of it. Fourthly, it is open-source GPL so that (provided you also release your source code) it costs nothing to use for this kind of project. Fifthly, it has large and active community of users who can give help and advice.

**P: (I'm new in AR research possibly the only one, in my college, with few months on this area. I plan to demonstrate that AR is a good alternative choice to VR making a little software, and I want to use Open Source software over all restrictive software because I'm a student and I have no money to pay any license fee, and it is my philosophy.)**

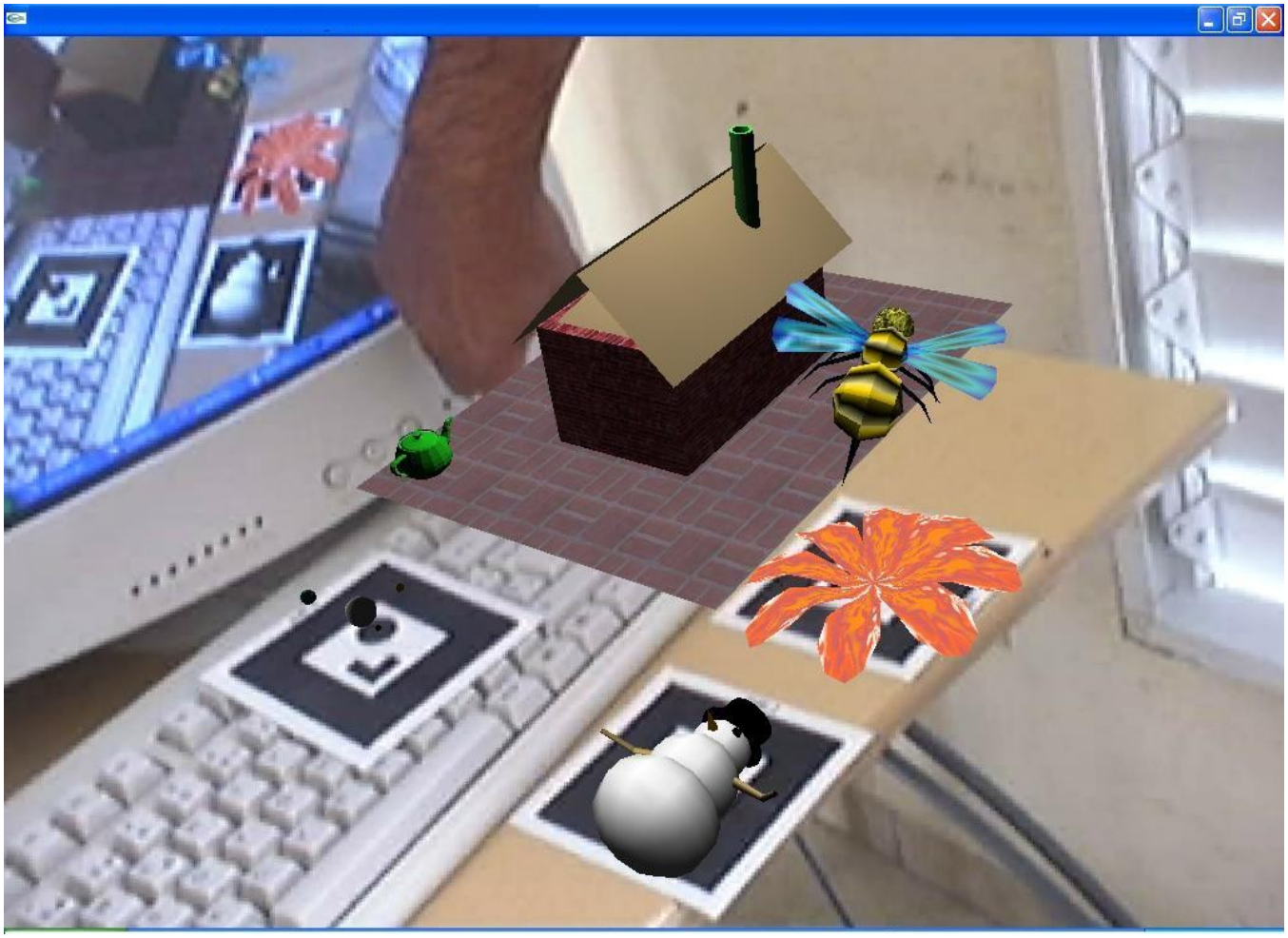
**I'm considering making an AR application to help the learning of computer architecture using internal components' virtual 3D models over a real time video streaming using a Logitech QuickCam webcam. Based on above comment what you recommend to me?**

**R:** This sounds like an interesting application. Of course, ARToolKit only does the tracking, so you would also need to do rendering of the components. That can be done using VRML.

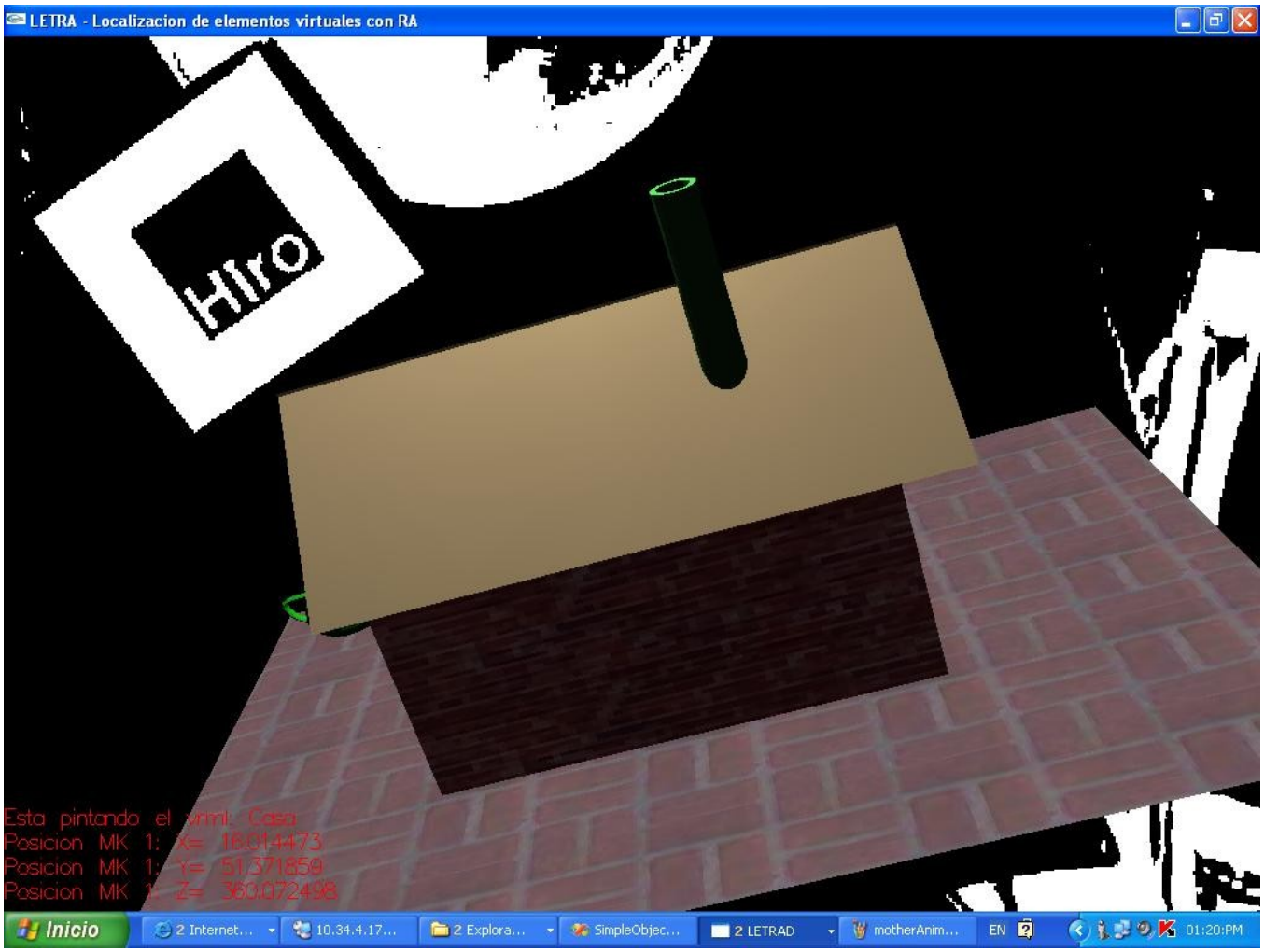
**P: What do you think about the program I want to develop? Is it an important choice? Why?**

**R:** It sounds interesting. Best of luck with it.

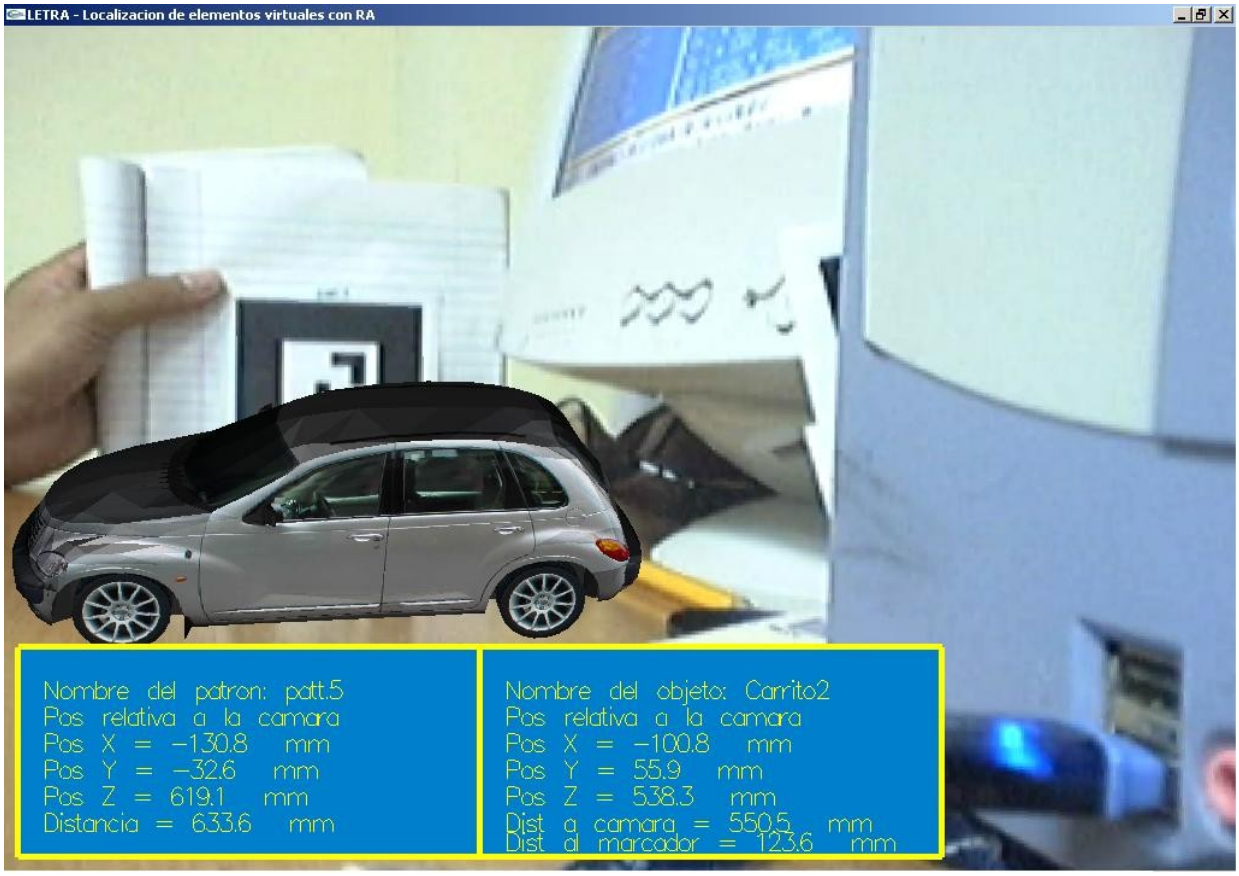
**Anexo 2: Escena Aumentada.**



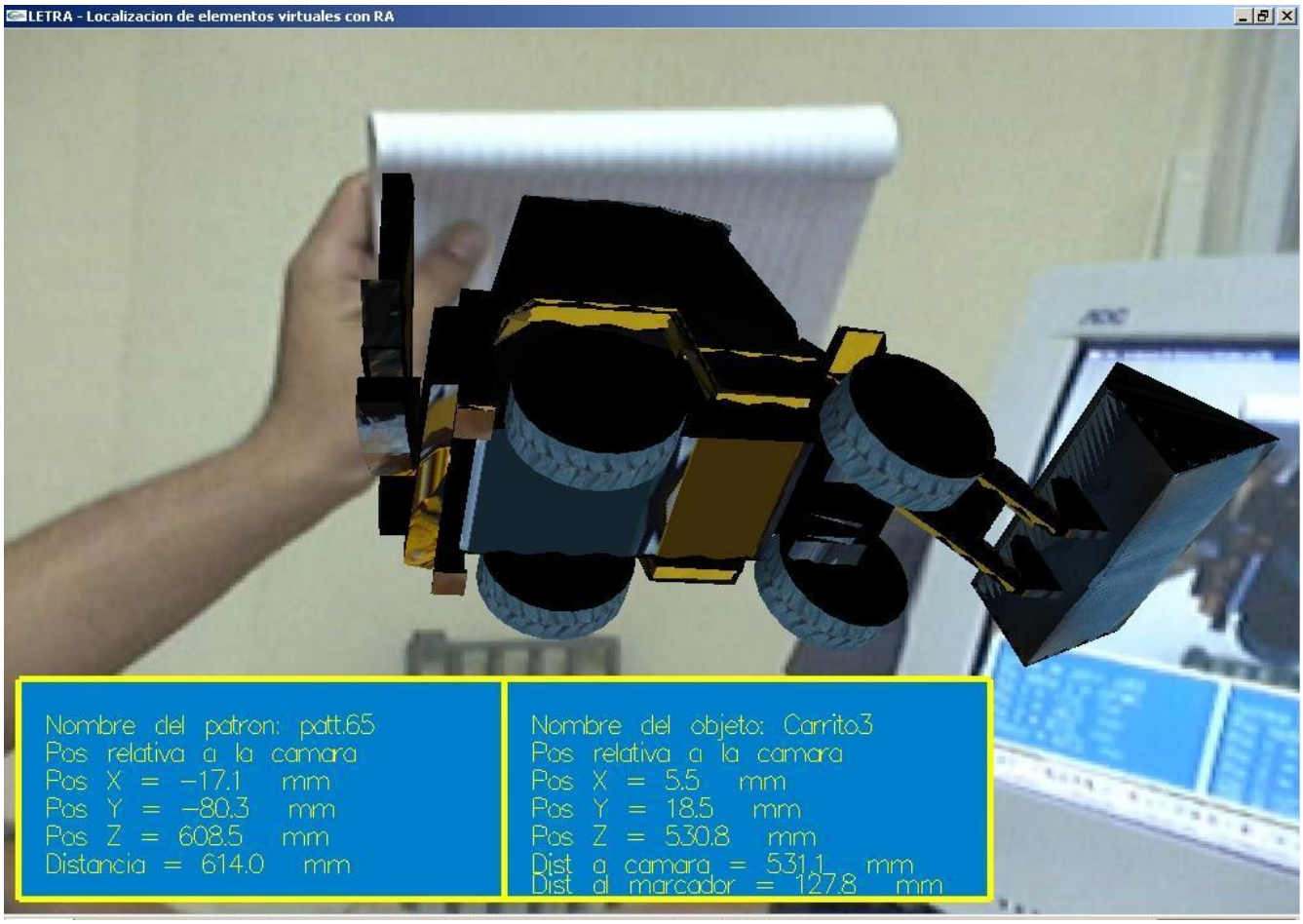
**Anexo 3: Escena aumentada en blanco y negro.**



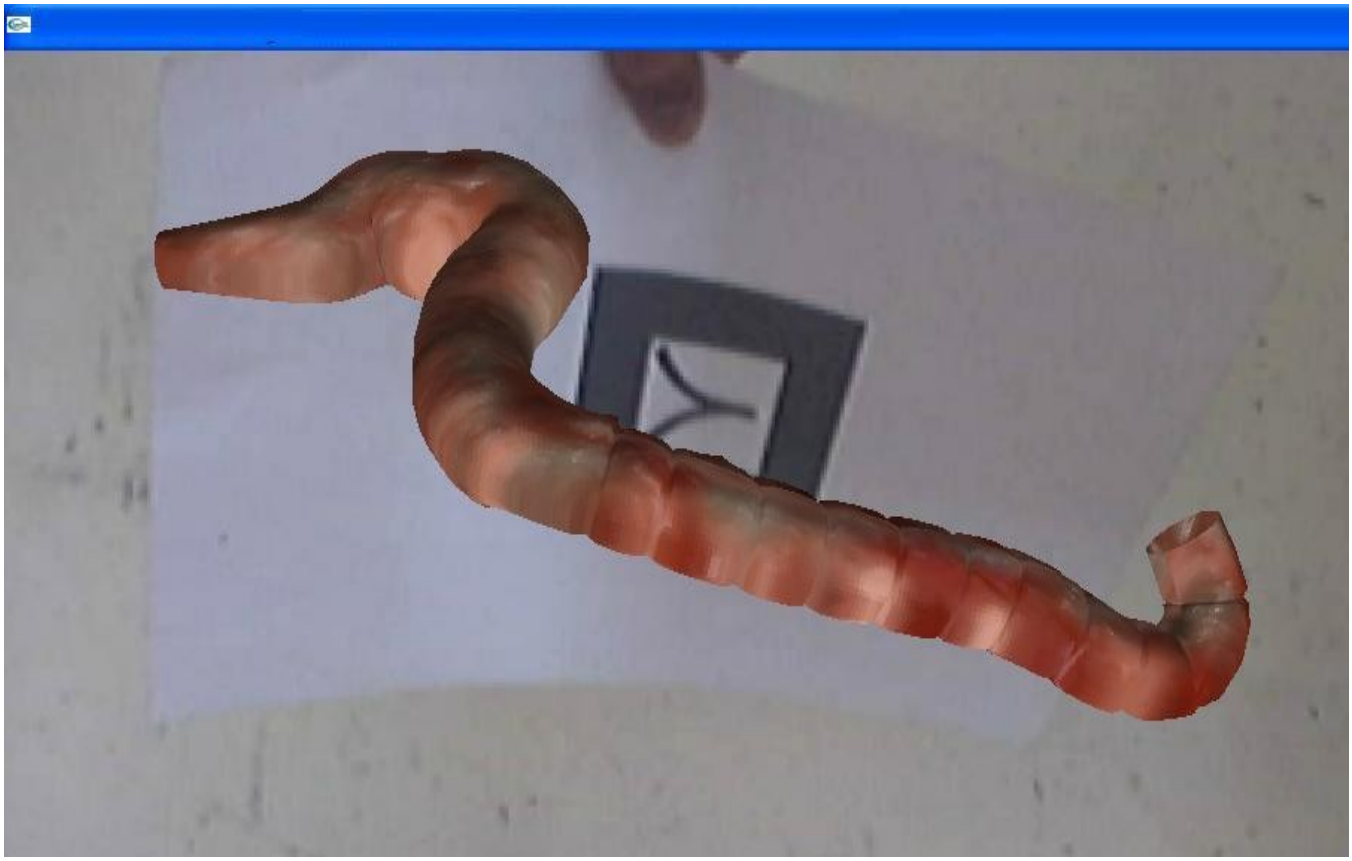
**Anexo 4: Localización de elemento virtual en escena real con respecto a la cámara real.**

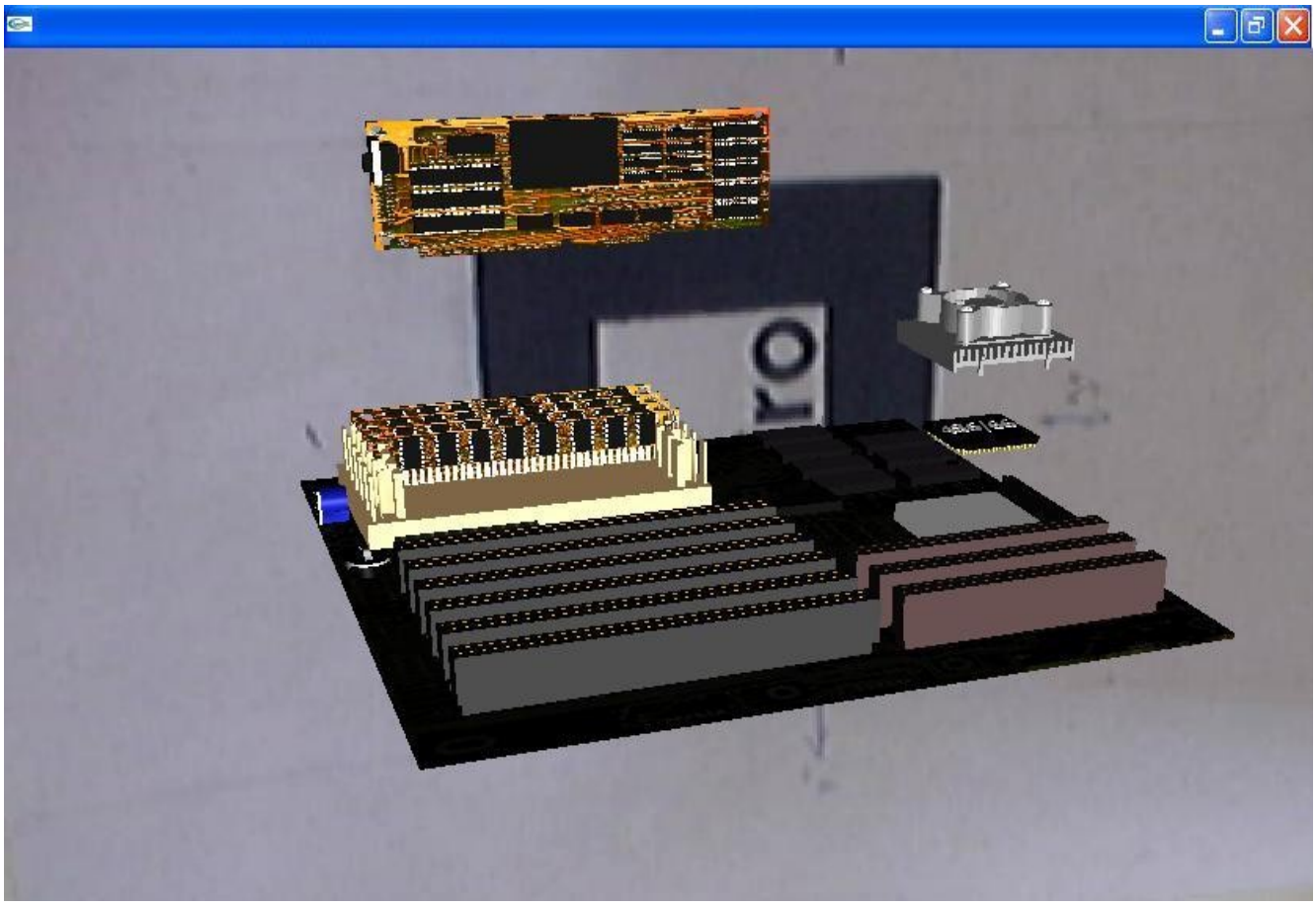
**Anexo 5: Localización de elemento virtual en escena real con respecto a la cámara real.**



**Anexo 6: Visualización de un intestino.**



**Anexo 7: Visualización de una MotherBoard.**



## GLOSARIO DE TÉRMINOS

**Fotogrametrista:** Persona que trabaja la fotogrametría.

**Fotogrametría:** Procedimiento para obtener planos de grandes extensiones de terreno por medio de fotografías, tomadas generalmente desde una aeronave.

**Framework:** En el desarrollo de software, un *framework* es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un *framework* puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

**HMD:** (Head Mounted Display) Dispositivo montado en la cabeza. Como su nombre lo indica este es un dispositivo muy parecido a un casco de realidad virtual, pero el usuario en vez de mirar un entorno totalmente generado por la computadora, observa una parte del mundo real con elementos virtuales insertados en la escena que esta visualizando.

**Modelo pinhole:** Modelo de cámara básico, puede ser elaborado haciéndole un orificio muy pequeño a una caja a través del cual pasa la luz que se proyecta sobre papel fotográfico para obtener una imagen del exterior de la caja. Este modelo es utilizado en la visión por computadoras por lo simple que es y la implementación de su modelo matemático ha dado muy buenos resultados.

**Tracking:** Este término se utiliza para denominar el seguimiento de objetos a través de visión por computadoras.

**Tracker:** Rastreador, se utiliza para hacer seguimiento de objetos en imágenes o videos.