



Universidad de Castilla-La Mancha

Escuela Superior de Informática

Grupo ORETO

ANTEPROYECTO

M.A.R.S

Sistema de *tracking* híbrido modular para Realidad Aumentada.

Autor: Sergio Pérez Camacho

Director: Carlos González Morcillo

22 de diciembre de 2009

Índice

1. Objetivos	3
1.1. Introducción	3
1.2. <i>Registro</i> de la realidad y <i>tracking</i>	4
1.3. Representación en tiempo real.	5
1.4. La problemática del <i>registro</i>	6
1.5. Objetivos de este <i>Proyecto de Fin de Carrera</i>	7
2. Métodos y fases de trabajo.	9
3. Medios a utilizar	11
4. Bibliografía	12

1. Objetivos

1.1. Introducción

“La *Realidad Aumentada* (AR) es”, según Klein [Kle06], “*la síntesis de imágenes virtuales sobre imágenes reales. Así, mientras que la Realidad Virtual sumerge al usuario en un mundo completamente artificial, la realidad aumentada completa escenas reales con información sintética superpuesta*”. Normalmente, las imágenes generadas por ordenador se superponen sobre el campo de visión del usuario para ofrecer información visual extra del entorno, o para proporcionar una guía de ayuda en la realización de alguna tarea. La complejidad de esa información puede variar, desde una simple flecha indicando una dirección, a un objeto 3D completamente integrado en la escena real, siendo éste indistinguible de uno no sintetizado.

Las aplicaciones de la realidad aumentada son casi innumerables. Imaginemos un cirujano que contase con la ayuda de una imagen de rayos-x del paciente superpuesta al cuerpo del paciente mientras le opera, un restaurador que pudiera contar con una recreación aproximada de un monumento sobre el vídeo real, o un empleado de seguridad con información extra sobre autenticaciones, sobre protocolos de actuación y sobre localización, todo ello en tiempo real.

La realidad aumentada dota al usuario de sentidos que no tiene. Gracias a ella se pueden representar (de forma resumida y procesable por un humano) datos capturados de varios sensores. Por ejemplo, se podría visualizar la temperatura de una serie de tuberías de una fábrica como un conjunto de cuadrados de colores: verde para una temperatura correcta, amarillo para una temperatura elevada pero aceptable, naranja para una temperatura elevada y no aceptable, y rojo para una temperatura críticamente elevada. El usuario visualizaría las temperaturas superpuestas al vídeo real, con la información integrada dentro del contexto adecuado, con los cuadrados sintetizados y compuestos encima del vídeo real.

Además de para representar datos que existen (pero que de otra forma no serían visibles), la realidad aumentada también se puede utilizar para visualizar entidades que no existen. Una aplicación clara sería la arquitectura o el diseño: se podría tener una vista previa del aspecto de una fachada con un retoque determinado o del aspecto que tendría una vivienda con unos muebles o una decoración específica sin tener que disponer de los muebles y sin tener que pintar ni una sola pared. Muchas retransmisiones de eventos deportivos hacen uso de la realidad aumentada para incorporar publicidad

a modo de objetos que no existen (por ejemplo, una botella gigante en un campo de fútbol) o para anotar de forma precisa una acción irreglamentaria (por ejemplo, mostrando una línea ficticia que indica el *fuera de juego* sobre el campo).

Según [ABB⁺01], un sistema de *AR*:

- combina objetos reales y virtuales en un entorno;
- se ejecuta de forma interactiva y en tiempo real; y
- alinea (compone de manera creíble) los objetos reales y virtuales entre sí.

1.2. *Registro de la realidad y tracking.*

La parte más compleja de un sistema de realidad aumentada es el registro de la realidad, entendiéndose por *registro* la captación del entorno y la transformación de esos datos en información útil que podrá ser utilizada para:

- Localizar al usuario (posición y orientación) dentro de un contexto determinado (campo de visión).
- Conocer la posición de un objeto que se encuentre dentro de ese contexto.
- Identificar un objeto y/o recuperar características únicas del mismo.

La localización automática del usuario y de su orientación es conocida como *tracking*. Teniendo en cuenta que el posicionamiento se puede calcular de forma global o relativa se distinguen varias aproximaciones para la obtención del mismo. Para el posicionamiento global se suele emplear algún sistema de marcas, donde una o varias texturas especiales, bien conocidas y situadas de forma estratégica en el entorno, actúan como marcas de posición global. Cuando el sistema capta esa marca, se puede conocer, o al menos estimar, la posición global del usuario en un contexto determinado. Otra aproximación no visual puede ser el posicionamiento global por triangulación y análisis de potencia de emisores de radiofrecuencia (redes *Wifi*, emisores *bluetooth*,...). Otro tipo de cálculo de posicionamiento es el local, que consiste en obtener los incrementos posicionales respecto a la localización anterior. Hay que tener en cuenta que en la realidad aumentada el posicionamiento se podría determinar con dos puntos en el espacio, uno representaría la posición del usuario y otro hacia dónde mira.

Esos dos puntos se pueden calcular de forma conjunta o de forma independiente, dependiendo del sistema de registro que se utilice. Por ejemplo, si utilizamos un sistema de marcas visuales, al detectar una de ellas podremos calcular la distancia relativa a la misma (y así conocer la posición global del usuario), y también de forma casi directa, hacia dónde se dirige el campo de visión. Sin embargo, un sistema inercial (una brújula digital, con aceleración, inclinación y rotación añadidas) no puede proporcionar un posicionamiento global completo por sí misma. Aun así, puede resultar muy útil para contrastar el registro realizado por otro tipo de sensores.

Conocer e identificar un objeto (incluida su posición relativa al observador), así como recuperar algunas de sus características, es una labor que normalmente recae sobre la disciplina conocida como *visión por computador*. La realidad aumentada se basa en gran medida en esta disciplina, y si bien la visión no es el único sentido para ser *aumentado*, sí que es el más importante¹ en lo que atañe a la misma.

Existen librerías que solucionan ciertos problemas relativos a la visión por computador, como puede ser *OpenCV*[Bra00]. Algunas otras, apoyan la creación de aplicaciones de *realidad aumentada* de forma directa, por ejemplo *ARToolKit* [KBBM99], que es una librería que procesa imágenes (de un vídeo) para buscar una marca y devolver el plano asociado y su vector superficie.

1.3. Representación en tiempo real.

Otra parte indispensable de un sistema de *Realidad Aumentada* es su capacidad de sintetizar y representar en tiempo real cualquier objeto dentro del sistema. Para ello, se puede hacer uso de algún motor de representación gráfico o incluso si se quiere tener un control mayor, de alguna librería gráfica de bajo nivel. Algunos ejemplos de motores gráficos de representación son:

- Crystal Space (GPL).
- Ogre 3D (GPL y licencia comercial).
- Blender Game Kit (GPL).
- Unreal Engine 3 (comercial).

¹Welch [Wel78] habla de un fenómeno conocido como *captura visual*. Consiste en la tendencia del cerebro a creer en lo que se ve antes de en lo que se siente, se oye,...

- ID Engine (comercial).

Las librerías gráficas apoyadas por hardware más conocidas son:

- Microsoft DirectX (gratuita).
- OpenGL (libre).

La diferencia entre un motor de representación y una librería gráfica son las facilidades que brindan. Mientras que una librería gráfica sólo se encarga de la representación en sí, una *engine* proporciona una serie de utilidades adicionales muy deseables. Estas utilidades tienen que ver con la carga, conversión, optimización y texturizado de objetos 3D; con la creación de texturas, de mapas navegables, de automatización de esqueletos asociados a una malla, y muchas otras partes que habría que programar desde cero si sólo se utilizasen librerías gráficas. Como contrapartida, la integración de un motor en un sistema de *Realidad Aumentada* requiere un trato especial, ya que los dispositivos utilizados en estos sistemas tienen ciertas limitaciones de tamaño y capacidad de proceso, impuestas por la necesidad de que sean fácilmente transportables y de una alimentación independiente. Los motores gráficos ofrecen características de representación avanzadas que suelen requerir alta capacidad de proceso, tanto de la CPU como de la GPU, características que no corresponden con las restricciones de hardware de los equipos portátiles de *Realidad Aumentada*.

Cabe mencionar la librería *SDL*², que proporciona, además de soporte de integración de *OpenGL*, la gestión de periféricos de entrada como ratón, teclado y *joystick*. Se podría clasificar como un *framework*.

1.4. La problemática del registro.

Aunque algunas formas de *tracking* son más precisas que otras³, ninguna está libre de errores. La distorsión óptica, la precisión o la necesidad de ahorrar algunos cálculos, entre otros factores conllevan un error, que si bien se puede acotar y atenuar, no se puede eliminar por completo. En un sistema de tiempo real, un error pequeño puede convertirse en uno grande en muy poco tiempo. Esto se debe a que en sistemas con cálculo relativo (o local) la posición actual se calcula a partir de la anterior. Por

²Simple DirectMedia Layer

³Se puede ver un resumen de las características de diferentes tecnologías en [Bai01]

cada fotograma procesado (de 15 a 60 por segundo dependiendo de la cámara) se calcula una nueva posición, lo que conlleva que el error se acumule en muy poco tiempo.

Para que un sistema de *AR* sea creíble y utilizable los objetos sintéticos deben estar alineados de forma precisa con la realidad [Azu93]. Un cálculo impreciso o tolerante con los errores propiciará una representación errónea y confusa para el usuario. En algunos ámbitos de aplicación esto puede ser crítico, como por ejemplo en una cirugía apoyada por uno de estos sistemas.

Algunos autores como Azuma ([ABB⁺01] [Azu93]) defienden un *tracking* híbrido como una posible solución de los problemas de registro. Utilizar varias formas de *tracking* puede ser muy beneficioso para la minimización del error. Hay que remarcar que un mismo tipo de sensor puede implementar varias formas de *tracking*. Por ejemplo, con una cámara se podrían detectar marcas para obtener un posicionamiento global y se podrían detectar puntos de interés para el cálculo de posicionamiento relativo ([CMPC06]) usando la técnica de *optical flow* [BB95]. Un *tracking* híbrido se apoyaría en las partes más fuertes de las técnicas implementadas y de los sensores utilizados. Del mismo modo, descartaría o daría poca importancia a los módulos que ofrezcan peores resultados en determinados ámbitos de aplicación.

1.5. Objetivos de este Proyecto de Fin de Carrera.

El proyecto de fin de carrera propuesto en este documento pretende construir un sistema de *tracking* híbrido que ayude a reducir en gran medida el error cometido por un sistema con un sólo tipo de *tracking*. Para ello se proponen los siguientes objetivos específicos:

- **Diseñar y desarrollar un sistema modular que explote las capacidades de varios sistemas de *tracking* de manera simultánea.**

Se diseñará un sistema robusto y mantible que acepte diferentes métodos de *tracking*. Se desarrollará de manera extensible, para que el hecho de añadir nuevos métodos de *tracking* no suponga realizar cambios en todo el sistema.

- **Integrar algunos métodos de *tracking* en este sistema.**

Se implementarán algunos métodos de *tracking* y serán incorporados al sistema.

- **Diseñar y desarrollar un sistema modular que trabaje con diferentes entradas de vídeo.**

Teniendo en mente que las fuentes de vídeo pueden ser cámaras con diferentes *drivers* o incluso

archivos de video se diseñará y se desarrollará un subsistema modular que controle esa entrada, la manipule y devuelva datos procesables por el resto del sistema. Añadir un nuevo tipo de fuente no debería suponer cambiar nada en el resto del sistema.

■ **Diseñar y desarrollar, o integrar, un pequeño *framework* para la representación de objetos sintetizados sobre vídeo real.**

Se diseñará e implementará un subsistema que se encargue de la representación de los objetos sintetizados. Este subsistema se encargará de componer estos objetos con el vídeo real. Para realizar este objetivo se integrará algún motor gráfico o se creará uno básica que sirva como base para el sistema de *Realidad Aumentada* .

2. Métodos y fases de trabajo.

Para llevar a cabo los objetivos propuestos en este documento se dividirá el proyecto en las siguientes fases:

1. Estudio del estado del arte.

En esta fase se repasarán las técnicas y sistemas de *tracking* más utilizados. Se analizará tanto el hardware que apoya estos sistemas, como el software que le da funcionalidad.

En esta etapa se estudiará también la problemática del registro de la realidad, tratando de averiguar a qué se deben los errores de los sistemas de *tracking* más usados. Se recopilarán datos experimentales sobre el tema que se contrastarán con pruebas propias. Para ello, previo a la realización de los experimentos, se analizarán algunas librerías que faciliten los mismos. Así, se considerará el uso de librerías ya mencionadas, como *OpenCV* o *ARToolKit*. Se analizarán las posibilidades que brindan y se realizarán varias pruebas de cálculo de la posición (global y relativa) basadas en ellas.

Paralelamente se revisarán varios motores. Se aceptará o descartará el uso de alguno de ellas para la sintetización y representación de los objetos. En el caso de que se descartar su uso, se estudiarían las librerías estándar para la representación gráfica, principalmente *OpenGL* y algún tipo de apoyo como pudiera ser *SDL*. Se estudiarán las estructuras de datos más relevantes para el almacenamiento de datos en la *Realidad Aumentada* y los diferentes algoritmos que faciliten la representación de los mismos en tiempo real.

2. Estudio de la viabilidad.

En esta fase se analizará la posibilidad de llevar a cabo la integración del proyecto en alguna arquitectura hardware probada para la *Realidad Aumentada*.

3. Análisis de la arquitectura del sistema.

En esta etapa se identificarán los componentes del sistema: se analizarán sus requisitos y se modelará utilizando alguna notación estándar para dicho propósito.

4. Diseño de la arquitectura del sistema.

Partiendo del modelo anterior se descompondrán dichos componentes en módulos concretos. Se

estudiarán estos módulos y a raíz de ello se decidirán las tecnologías a utilizar para su desarrollo: lenguajes de programación, librerías, ...

5. Desarrollo del sistema.

Se desarrollarán los módulos y se integrarán en sus componentes, que formarán el sistema.

6. Pruebas del sistema.

Durante el desarrollo, que será incremental, se llevarán a cabo las pruebas de los módulos que se vayan desarrollando. Se utilizará algún método fiable para tal propósito.

Cuando se termine un módulo, se harán las pruebas de integración pertinentes.

Cuando se termine el sistema, se harán pruebas de integración de los componentes y de su correcto funcionamiento.

7. Evaluación de los resultados.

Se evaluará la ganancia de precisión del sistema al compararlo con uno de *tracking* simple. También se analizará la penalización de rendimiento que supone la inclusión de varios métodos de *tracking* en un mismo sistema.

Se compararán los resultados de utilizar distintos métodos de *tracking* para construir el sistema híbrido.

8. Documentación del proyecto.

Paralelamente a los puntos anteriores se escribirá la documentación asociada al proyecto. Se realizará también un manual de usuario del sistema.

3. Medios a utilizar

El propósito del proyecto es realizar un sistema robusto, modular y altamente extensible para que pueda ser utilizado y actualizado tras la finalización del mismo.

Teniendo este objetivo en mente, la distribución del proyecto se realizará mediante alguna licencia libre como GPLv3.

Se seguirá el *proceso unificado de desarrollo de software* [JBR00] durante la realización del proyecto. Para la notación del modelado se usará UML.

Para el control de versiones se utilizará *subversion*. El código fuente será alojado en un repositorio situado en los servidores de *ORETO*.

Para la elección de los lenguajes de programación se estudiarán las ventajas que proporcionan a la hora de portar sistemas y de trabajar en tiempo real. Este, junto con el prototipado rápido de algunos de los módulos, será el criterio de elección de los mismos. Se estudiarán distintas alternativas para llevar a cabo la persistencia, como ficheros de textos planos, jerárquicos (XML) o bases de datos. . . . Se optará por la opción más eficiente, siempre que permita la continuación futura del proyecto, esto es, la escalabilidad y mantenibilidad del mismo.

Se utilizará alguna librería gráfica libre para la creación del módulo encargado de sintetizar objetos, o incluso, si se probase oportuno, una *engine* libre que apoyase varias de las tareas a realizar por el sistema.

Como plataforma hardware del sistema se utilizarán unas gafas 3D, varias cámaras USB, y un equipo móvil basado en *Intel Atom* con una alimentación independiente basada en dos baterías de litio. Todo este material es propiedad del grupo *ORETO*. Así, la mayor parte del desarrollo del proyecto se llevará a cabo en las instalaciones de dicho grupo en la E.S.I. de Castilla-La Mancha, en Ciudad Real.

4. Bibliografía

Referencias

- [ABB⁺01] R. Azuma, Y. Baillet, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. Recent Advances in Augmented Reality. *IEEE Computer Graphics and Applications*, 21(6):34–47, 2001.
- [Azu93] R. Azuma. Tracking requirements for augmented reality. *Communications of the ACM*, 36(7):51, 1993.
- [Bai01] Y. Baillet. A Survey of Tracking Technology for Virtual Environments. *Fundamentals of Wearable Computers and Augmented Reality*, page 67, 2001.
- [BB95] SS Beauchemin and JL Barron. The computation of optical flow. *ACM Computing Surveys (CSUR)*, 27(3):466, 1995.
- [Bra00] G. Bradski. The opencv library. *Doctor Dobbs Journal*, 25(11):120–126, 2000.
- [CMPC06] A.I. Comport, E. Marchand, M. Pressigout, and F. Chaumette. Real-time markerless tracking for augmented reality: the virtual visual servoing framework. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):615–628, 2006.
- [JBR00] I. Jacobson, G. Booch, and J. Rumbaugh. El proceso unificado de desarrollo de software. *Addison Wesley. Madrid*, 2000.
- [KBBM99] H. Kato, M. Billinghurst, B. Blanding, and R. May. ARToolKit. *Hiroshima City University*, 1999.
- [Kle06] Georg Klein. *Visual Tracking for Augmented Reality*. PhD thesis, University of Cambridge, 2006.
- [Wel78] R.B. Welch. *Perceptual modification: Adapting to altered sensory environments*. Academic Press, 1978.