TRABAJO FIN DE MÁSTER

(Máster en Tecnologías Informáticas Avanzadas)

Arquitectura para la Computación Contextual: Aplicación a la Realidad Aumentada

30 de Septiembre de 2009

Autor

José Ángel Mateos Ramos

(Ingeniero en Informática)

Directores

Dr. Luis Jiménez Linares - Dr. Carlos González Morcillo

Este documento fue maquetado con LaTeX. Imágenes compuestas con Gimp y OpenOffice.

Resumen

En entornos dinámicos que requieren el tratamiento y gestión de un alto número de variables y datos resulta imprescindible disponer de mecanismos que permitan seleccionar un subconjunto relevante de éstos. La computación contextual facilita el tratamiento de la complejidad inherente de algunos entornos de representación interactivos, como es el caso de los sistemas de Realidad Aumentada.

La Realidad Aumentada es una disciplina que enriquece la percepción que tiene el usuario del entorno que lo rodea superponiendo objetos generados de forma virtual sobre imágenes capturadas del mundo real. Debido a la rápida evolución de la tecnología, este tipo de aplicaciones está tomando más y más importancia en nuestros días.

Actualmente, existe una gran variedad de frameworks para el desarrollo de aplicaciones de Realidad Aumentada, pero la mayoría de ellos sólo proporcionan un conjunto de servicios básicos imprescindibles y no utilizan el conocimiento disponible (localización, rol, percepciones, etc) para proporcionar funcionalidad avanzada, como la información dependiente del contexto. Estos valores serán modelados y utilizados para seleccionar y modificar la información superpuesta sobre la realidad capturada, adaptándola a las necesidades específicas de cada usuario.

En este trabajo se presenta una nueva aproximación que utiliza los conceptos del área de los sistemas multi-agente para el desarrollo y el despliegue de aplicaciones de Realidad Aumentada. Las características de autonomía, reactividad, proactividad y habilidad social de dichos sistemas permiten un manejo inteligente del contexto que representa al usuario y se pueden aprovechar para gestionar los datos relacionados de una forma eficiente y adecuada.

Índice general

Índice de figuras

1

Introducción

La Realidad Mezclada [41] (Mixed Reality) es una técnica que consiste en combinar información visual del mundo real y elementos generados por computador con el objetivo de obtener nuevos entornos y visualizaciones, de tal forma que los objetos reales y virtuales comparten representación y son capaces de interactuar los unos con los otros. Dentro de la disciplina de la Realidad Mezclada se pueden distinguir dos aproximaciones diferentes: la Virtualidad Aumentada (VA) [40] y la Realidad Aumentada (en adelante RA) [6]. La Virtualidad Aumentada consiste en enriquecer mundos virtuales generados por computador con objetos reales. La Realidad Aumentada consiste en enriquecer el mundo real con objetos generados mediante un computador.

La Realidad Aumentada mejora la percepción y la interacción del usuario con el mundo real, debido a la información proporcionada por los objetos virtuales. Estos objetos superpuestos sobre la escena aportan información que el usuario no puede percibir de forma independiente. Según Azuma [6] un sistema de realidad aumentada debe conseguir los siguientes objetivos:

1. Combinar mundo real y virtual.
2. Ser interactivo en tiempo real.
3. La información en 3D debe estar correctamente alineada con la imagen del mundo real.

Se puede decir que la Realidad Virtual y la RA están relacionadas. La principal diferencia entre estas dos disciplinas es que la Realidad Virtual sitúa al usuario dentro de un mundo completamente generado de forma virtual. Las reglas (físicas) que rigen este mundo virtual pueden o no coincidir con las del mundo real. En el año 1996 Milgram y Kishino [41] sitúan la RA dentro del continuo Realidad-Virtualidad como puede comprobarse en la figura 1.1.

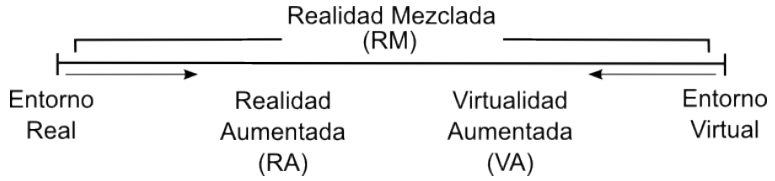

Figura 1.1: Realidad-Virtualidad

A la izquierda de la figura se encuentra representado el mundo real, definido por un conjunto de experiencias reales. A la derecha se encuentra el entorno virtual, en el que las acciones y percepciones reales se sustituyen por un mundo generado artificialmente. Todo el espectro entre estos dos extremos es lo que se conoce como Realidad Mezclada. La RA se sitúa más cerca del extremo izquierdo ya que está constituida en su mayor parte por las percepciones reales.

Para implementar una aplicación de RA efectiva es necesario desarrollar e integrar diferentes factores [63] como son:

- Hardware y software capaz de renderizar gráficos para generar el contenido virtual que se superpondrá sobre el mundo real.
- Técnicas de tracking para que los cambios en la posición del usuario se reflejen de forma correcta en el dibujado de los gráficos.
- Calibración de los dispositivos y de las herramientas de posicionamiento para alinear de forma precisa los mundos reales y virtuales.
- Hardware de representación adecuado para proporcionar la información al usuario.
- Hardware de procesamiento que soporte dispositivos de entrada/salida y además nos permita ejecutar la simulación de RA.
- Técnicas de interacción que permitan al usuario manipular el contenido virtual presentado.

Según la enumeración anterior, se puede llegar a la conclusión de que una aplicación de RA se apoya en dos tecnologías fundamentales, el tracking y los sistemas de representación de información. El proceso de tracking consiste en proporcionar la localización, en cada instante de tiempo, de personas u objetos en movimiento. Para proporcionar un alineamiento correcto de la información mostrada, se hace necesario un buen proceso de tracking que proporcione en cada momento el punto de vista del usuario. La precisión y la velocidad con la que se proporciona esta información es crítica para el correcto funcionamiento del sistema. Si el punto de vista no se obtiene de forma estable y correcta, los objetos no se colocarán de forma adecuada y se perderá la sensación de inmersión, como consecuencia de esto, la información presentada será inservible. Además, esta información debe mostrarse utilizando un dispositivo portátil que no obligue al usuario a permanecer estático dentro del entorno en el que se desarrolla la acción.

Durante el principio de los años 90, mientras la disciplina de la realidad aumentada estaba creciendo, Weiser [60] definió la idea de computación ubicua. Se trata de un concepto en el que la tecnología está oculta en los objetos cotidianos. Weiser define un entorno en

el que la tecnología pasa desapercibida para los usuarios. Actualmente, los dispositivos móviles proporcionan más que una simple funcionalidad de comunicación. Muchos de ellos contienen elementos que permiten ejecutar aplicaciones de RA. El punto de encuentro entre la RA y la computación ubicua se denomina Realidad Aumentada Móvil (en adelante RAM).

Los sistemas de RAM necesitan manejar una gran variedad de dispositivos y sensores (cámaras, acelerómetros, giróscopos, HMD...), por tanto se hace necesaria una arquitectura flexible que proporcione métodos y técnicas que permitan la gestión de todos estos dispositivos de una forma adecuada. Hace relativamente poco tiempo, la gran mayoría de las aplicaciones de realidad aumentada eran monolíticas, pero con la evolución de la disciplina han aparecido otras aproximaciones distribuidas (como [7]). Desde entonces han aparecido una gran cantidad de propuestas de arquitecturas para desarrollar aplicaciones de RAM. Estas arquitecturas proporcionan los componentes básicos para construir prototipos de aplicaciones descentralizadas. El principal inconveniente es que ninguna de las arquitecturas estudiadas aprovechan la información de contexto del usuario para proporcionar servicios más avanzados.

En este trabajo proponemos una arquitectura Multi-Agente para realizar aplicaciones de RA que aprovechen al máximo el **contexto** del usuario para proporcionar algunos servicios más avanzados y que permitan integrar y mejorar diferentes técnicas de tracking existentes.

Debido a la cantidad de información manejada por las aplicaciones de RA (modelos, usuarios, entornos ...), al dinamismo de dichas aplicaciones y a los rápidos cambios del contexto del usuario se hace necesaria una rápida adecuación de la información virtual acorde a estos cambios. La arquitectura propuesta se presenta como una serie de agentes que colaboran entre si para realizar las tareas de gestión de contexto. Esto facilita el reparto de responsabilidades, la heterogeneidad, la concurrencia y la distribución. Para que todo el sistema se comporte correctamente se hace necesaria una configuración previa tanto del entorno como de la sensorización disponible.

1.1. Marco de trabajo

El presente trabajo se encuadra en el Proyecto de Investigación Hesperia: Homeland Security: Tecnologías para la Seguridad Integral en Espacios Públicos en Infraestructuras, financiado por el CDTI (UCTR060182, 2006-2010).

El proyecto Hesperia tiene por objeto el desarrollo de tecnologías que permitan la creación de sistemas punteros de seguridad, vídeo vigilancia y control de operaciones de infraestructuras y espacios públicos. Uno de los paquetes que forman parte de este proyecto es el de la representación de información. El objetivo de este paquete es el de mejorar la productividad de los empleados de seguridad mediante nuevos interfaces de usuario y paradigmas de interacción.

Muchas veces se hace necesario un proceso de entrenamiento de los empleados para la ejecución de ciertas tareas. El uso de la RA puede ayudar a reducir el tiempo empleado en la preparación de los guardias, mediante el guiado y asistencia de las funciones a realizar durante el periodo de preparación. Además, un sistema de RA maduro y puesto en explotación se puede utilizar como un instrumento de trabajo que facilite la labor de mantenimiento y seguridad.

Finalmente, el mismo equipo de RA podrá ser utilizado por personal con diferentes ro-

les (empleado de seguridad, encargado de mantenimiento de red eléctrica, encargado de fontanería...) y dependiendo de dicho rol, la información presentada será distinta en cada caso.

1.2. Temas de Investigación

Los principales temas (topics) de investigación involucrados en el trabajo realizado son los siguientes:

- Arquitecturas de Realidad Aumentada.
- Inteligencia Artificial Distribuida.
- Sistemas Multi-Agente.
- Sistemas de Tracking.
- Sistemas de representación 3D interactivos.
- Visión por Computador.

1.3. Estructura del documento

El presente documento consta de dos capítulos principales en los que se presenta el contenido de la Tesis de Master. El primero de ellos, se estudia el estado del arte de las arquitecturas de realidad aumentada, los sistemas de tracking y de las aplicaciones dependientes del contexto. Para ello, se proporciona una amplia visión en orden cronológico de las propuestas en cada una de las categorías expuesta anteriormente. En el capítulo de descripción de la propuesta se detalla nuestra definición de contexto, los métodos utilizados para la adquisición del contexto y la arquitectura que se propone. Por último, se exponen las conclusiones y las propuestas de trabajo futuro.

El documento contiene dos anexos. El primero de ellos, Anexo A, contiene un resumen de las asignaturas cursadas en el Master de Tecnologías Informáticas Avanzadas, detallando los trabajos efectuados para cada una de ellas. En el Anexo B se presenta el Curriculum Vitae del autor.

2

Estado del Arte

2.1. Frameworks para aplicaciones de RA

Son muchos los elementos (visualización, tracking, redes de comunicaciones...) que juegan un papel importante en lo que a aplicaciones de realidad aumentada se refiere. Hasta hace relativamente poco tiempo las arquitecturas de las aplicaciones de RA eran monolíticas, pero con la proliferación de esta disciplina han aparecido nuevas aproximaciones que proporcionan los elementos necesarios para construir prototipos de diferentes aplicaciones de RA descentralizadas. La gran mayoría presentan un núcleo para la arquitectura y una serie de dispositivos que se pueden añadir a la aplicación en forma de *plugins*. A continuación se presentan brevemente los frameworks más trascendentales utilizados para el prototipado de las aplicaciones de RA.

Uno de los primeros frameworks que permitían la modularización de las aplicaciones de realidad aumentada es el especificado por Feiner et al. [22]. En este trabajo los autores proponen una arquitectura para la implementación de un sistema de navegación. Para el desarrollo de la infraestructura se ha utilizado COTERIE [37] que es un campo de pruebas para el prototipado rápido de entornos virtuales distribuidos. Está diseñado para soportar la creación de entornos virtuales en los que múltiples usuarios simultáneos interactúan con varios displays y dispositivos de entrada heterogéneos. Las principales ventajas de la arquitectura son el tratamiento uniforme de la información, ya sea local o remota. Además se utiliza un modelo de programación orientado a objetos multihilo. Otra de las ventajas destacables es la propagación de información de forma asíncrona. El principal inconveniente de la arquitectura presentada es que es una arquitectura específica para aplicaciones de navegación y la limitación al lenguaje de programación Modula-3.

DWARF, propuesto por Bauer et al. [7] en el año 2001, es un framework consistente en un conjunto de servicios distribuidos específicos para las necesidades de las aplicaciones de RA, el middleware que los combina y una arquitectura de software extensible. La descripción de la arquitectura se centra básicamente en la ingeniería del software aplicada a los sistemas de RA.

El framework tiene tres aspectos fundamentales, los servicios, el middleware y la arquitectura. DWARF está compuesto por una serie de servicios y cada uno de éstos proporciona una determinada funcionalidad al usuario o a otros servicios. DWARF también especifica el middleware necesario para componer estos servicios de forma dinámica y establecer la comunicación entre ellos. Por último, la arquitectura define la estructura básica de los sistemas de RA que se pueden construir con ella. Ésta es fácilmente entendible y puede ser configurada, añadiendo los módulos hardware adecuados, para que se adapte a las necesidades del usuario.

Los servicios básicos aportados por DWARF para la construcción de aplicaciones de realidad aumentada son:

- **Subsistemas de tracking**: El subsistema de tracking consiste en una serie de servicios software que pueden ser combinados dinámicamente. Se trata de una arquitectura jerárquica consistente en una serie de dispositivos de bajo nivel que suministran información acerca de la posición y la orientación de un objeto, además de un *manager* que combina la salida de estos dispositivos. Con esta aproximación resulta sencillo agregar nuevos servicios de tracking. Todo lo que se ha de hacer es registrar el nuevo servicio frente al *manager*. Otra de las ventajas es la posibilidad de añadir o eliminar dispositivos de forma dinámica.
- **Modelo del mundo**: La información acerca de los objetos reales y virtuales se almacena en una pequeña base de datos especializada denominada modelo del mundo. Para facilitar el modelado de objetos se ha añadido la posibilidad de parsear descripciones textuales en XML.
- **Motor de flujo de tareas**: Muchos de los sistemas de RA requieren de ayuda a la navegación y de asistencia para tareas de mantenimiento. Estos aspectos se pueden representar haciendo uso del motor de flujo de tareas de DWARF. Un flujo de tareas es una secuencia de acciones que tiene que ser ejecutada. Para la definición de flujos de tareas se utiliza un lenguaje (TDL) basado en XML. Mediante TDL se define una máquina de estados finita de actividades conectadas mediante transiciones.
- **Acceso a servicios dependientes del contexto**: La idea básica de este componente (CAP) es la de encapsular información acerca del usuario para poder servirla cuando sea necesario. Por ejemplo para tareas en las que sea necesaria una identificación o una especificación de preferencias.
- **Motor de interfaz de usuario**: Este componente es el encargado de proporcionar la interacción con el usuario. Para desarrollar interfaces de usuario genéricas se utiliza UIML.

DWARF es una arquitectura que contiene todos los componentes básicos para las aplicaciones de realidad aumentada, pero está muy enfocada a la ingeniería del software. Los autores no dan demasiados detalles acerca de la implementación del sistema.

Otro framework ampliamente utilizada es la de Studierstube propuesta por Scmalstieg et al. [52]. El objetivo de Studierstube es el de proporcionar una interfaz de usuario en 3D tan poderosa como la metáfora de escritorio en 2D. Se trata de un conjunto de clases implementadas en C++ construidas sobre el toolkit OpenInventor(OIV) [58]. Se han desarrollado dos componentes relacionados:

- Un conjunto de extensiones de la jerarquía de clases del OpenInventor.
- Un framework de ejecución que constituye el entorno adecuado para la ejecución de las aplicaciones de Studierstube.

Las aplicaciones se escriben y se compilan, de forma separada, como objetos compartidos que son dinámicamente cargados en el framework de ejecución. Un mecanismo de seguridad se asegura de que sólo una instancia de cada objeto se carga en el sistema. La principal ventaja que aporta esta arquitectura es la programación independiente de las aplicaciones, ya que esto permite que un desarrollador utilice las partes necesarias según sus necesidades. El principal inconveniente es la limitación a un solo lenguaje de programación. Además se centra básicamente en la interacción con el usuario dejando de lado otros aspectos importantes de las aplicaciones de RA.

VHD++ es una framework ideado por Ponder et al. [44]. Los autores plantean una arquitectura básica orientada a componentes que puede ser fácilmente extendida y personalizada añadiendo nuevos componentes que siguen unos acuerdos predefinidos para poder ser integrados en el framework. El objetivo perseguido es el de permitir la reutilización de código y ocultar la complejidad, proporcionando unidades funcionales independientes que se agregan para una composición rápida de aplicaciones. El framework consiste en un conjunto de, aproximadamente, 500 clases escritas en C++ organizadas en 35 paquetes.

Los componentes principales del framework son tres:

- **vhdRuntimeEngine**: Se trata del núcleo de cualquier aplicación basada en VHD++. Es el módulo encargado de gestionar la colaboración de un conjunto finito de servicios inherentes a cualquier aplicación de VR/AR. Un servicio se puede seleccionar, cargar y activar en tiempo de ejecución, lo que permite una rápida composición de prototipos de aplicaciones funcionales. El *vhdRuntimeEngine* juega el papel de middleware separando la aplicación del sistema operativo y el hardware específico.
- **vhdServices**: Se trata de componentes software independientes que se pueden añadir a las aplicaciones. Cada uno de estos componentes tiene una función bien definida, por ejemplo detección de colisiones, animación de esqueletos, seguimiento de la cámara, dispositivos de entrada/salida etc. Cualquier servicio seleccionado puede ser utilizado según se proporciona o pueden ser adaptados a las necesidades particulares a partir de la herencia y la sobrescritura de los métodos virtuales.
- **vhdProperties**: Se trata de objetos que contienen la información acerca del estado del sistema y de la simulación. Por ejemplo, una instancia de *vhdProperties* puede representar la configuración de un servicio determinado. Cada clase *vhdProperty* tiene asociado un *vhdPropertyLoader* que interpreta los parámetros estructurados en forma de fichero XML para crear una instancia *vhdProperty* de una clase determinada. Los desarrolladores pueden extender el conjunto de clases *vhdProperty* y *vhdPropertyLoader* para añadir nuevos estados del sistema y tipos de datos de simulaciones.

Utilizando VHD++ se pueden desarrollar prototipos rápidos de aplicaciones de realidad aumentada. Además permite la reutilización de los componentes previamente desarrollados para nuevas aplicaciones. Está principalmente orientado a aplicaciones de realidad virtual y totalmente desarrollado en C++.

La siguiente es una arquitectura Software orientada a objetos para aplicaciones de Realidad Mixta propuesta por Piekarski y Thomas [43]. La base de la que parten los autores es

que los sistemas de realidad mixta están dirigidos por sensores por lo que la arquitectura software está basada en el flujo de información. El modelo de flujo de información está soportado por el uso de objetos que realizan acciones específicas como pueden ser el procesamiento de la información obtenida de un dispositivo de tracking, combinar los resultados y renderizar gráficos en 3D. La aproximación a objetos permite dividir los problemas en subproblemas independientes para simplificar el desarrollo software. Así, una vez desarrollados los objetos deseados, se conectan en forma de grafo dirigido y según van entrando nuevos valores en el sistema, se procesan a través del gráfico como un flujo de información y se ajusta el estado para renderizar de forma correcta la información.

Las clases utilizadas en la arquitectura se pueden dividir en cuatro categorías diferentes:

- **Clases de información**: aquellas clases que se utilizan para representar valores.
- **Clases de procesamiento**: aquellas clases que se utilizan para procesar valores de entrada y producir algún tipo de valor de salida.
- **Clases de núcleo**: aquellas que implementan las características principales y que otras clases pueden heredar o utilizar.
- **Clases de ayuda**: aquellas que implementan las interfaces y sirven como ayuda al desarrollo.

El sistema completo ha sido estructurado haciendo uso de un modelo basado en sistema de archivos. Todos los objetos que procesan información en el sistema se almacenan en este repositorio haciéndolos accesibles al resto de objetos en el sistema a través de su descubrimiento en tiempo de ejecución. Esta estrategia permite el manejo de grandes colecciones de objetos por parte de múltiples módulos sin la necesidad de utilizar variables globales ni el problema de las colisiones en el espacio de nombres. La base de la implementación de este tipo de direccionamiento es el sistema de archivos UNIX. Los autores utilizan las rutas para la obtención de los recursos del sistema y los valores de los inodos son punteros a direcciones de memoria en las que están alojados los objetos.

El principal problema de esta arquitectura es que es completamente dependiente del sistema de archivos UNIX. Además existen otras aproximaciones más adecuadas para el descubrimiento de servicios que el uso de un sistema de archivos.

Otro de los framework orientados a componentes es MORGAN propuesto por Ohlenburg et al. [42]. Se trata de un framework que se apoya en CORBA y permite el fácil acceso a dispositivos externos como pueden ser los de tracking y esconde los detalles del sistema distribuido. MORGAN utiliza un conjunto de patrones bien conocidos para ser fácilmente entendible y extensible.

Para la comunicación entre componentes se utiliza el patrón publicador-suscriptor. Cada uno de los dispositivos es un publicador. Los componentes que estén interesados en la información proporcionada por alguno de los publicadores se suscriben a este servicio. El patrón factoría se utiliza para implementar la creación y la instanciación de componentes remotos. La clase factoría de un componente es la responsable de la instanciación de dicho componente y de informar al Broker acerca de la existencia del mismo.

Además MORGAN, a diferencia de otros framework propuestos para aplicaciones de RA y VR, utiliza su propio motor de render, diseñado para satisfacer los requerimientos de las aplicaciones multiusuario de VR y RA. Con esto se asegura un estado consistente en los gráficos generados, ya que si varios fuentes los manipulan, existe un hilo independiente de actualización el cual aplica los cambios propuestos por cada uno de los hilos de renderizado.

2.2. Sistemas de tracking

En la sección anterior se han estudiado algunas de las arquitecturas y frameworks más importantes para el despliegue de aplicaciones de RA. En este apartado se estudiarán los principales métodos de tracking empleados por todos ellos para obtener la correspondencia entra la imagen de las cámaras reales y el punto de vista virtual.

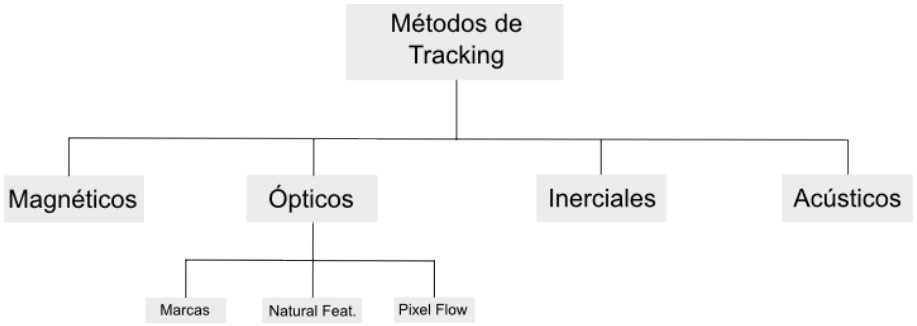

Figura 2.1: Métodos de tracking

En la figura 2.1 se puede ver una clasificación de los distintos mecanismos de tracking existentes. A continuación se explicarán las características principales de cada uno de ellos, así como los inconvenientes que presentan y las aproximaciones de tracking híbrido.

Los sistemas de tracking magnéticos utilizan la alteración de una serie de campos magnéticos para la implementación de aplicaciones encargadas de medir la posición y la orientación en el espacio 3D [45]. Estos sistemas obtienen la posición y la orientación de los sensores que los componen analizando las variaciones de los campos en el sistema de coordenadas y actualizando las medidas obtenidas anteriormente. El problema principal que presentan estos sistemas es que la presencia de metales en el entorno alteran su correcto funcionamiento, produciendo errores significativos en las mediciones.

El principio de los sistemas de tracking inerciales es el de determinar la posición de un cuerpo mediante una doble integración de la aceleración lineal experimentada por éste. Además, también es posible conocer la orientación mediante la integración de la velocidad angular [36]. El problema de estos sistemas es que debido a la necesidad de integrar los valores cualquier error, por pequeño que sea, se propaga en el tiempo y como consecuencia las mediciones cada vez serán menos fiables.

Los sistemas de tracking acústicos se basan en las ondas de sonido ultrasónicas para obtener la posición y la orientación del objetivo [10]. La mayoría de ellos miden el tiempo que tarda en llegar una onda sonora a un determinado sensor. Por regla general, los sensores se encuentran fijos en el entorno y el objeto transporta emisores de ultrasonidos. Este tipo de sistemas presentan gran cantidad de desventajas debido a que el sonido se propaga a una velocidad muy lenta, como consecuencia, la tasa de refresco de los datos acerca del usuario es muy lenta también. Además, la velocidad de propagación del sonido se puede ver afectada por la temperatura, la humedad y la presión existentes en el ambiente.

Por último, los sistemas de tracking ópticos [29] utilizan técnicas basadas en visión por computador para detectar puntos de interés dentro del entorno en el que el usuario está desplazándose. Estos puntos de interés se pueden obtener haciendo uso de diversas

técnicas como la detección de puntos clave en la imagen y el uso de marcas estáticas. El problema de estas técnicas de posicionamiento es que dependen en gran parte de la iluminación de la escena. Además, la detección de marcas estáticas [32] requiere de una preparación previa del entorno además estas marcas deben estar visibles en todo momento, de lo contrario el proceso de tracking falla.

Como se puede comprobar, cualquiera de las técnicas de tracking expuestas anteriormente presenta una serie de inconvenientes asociados a la propia naturaleza del sistema utilizado. Los sistemas de tracking híbridos aparecen como una solución a estos problemas y su principio consiste en utilizar dos o más técnicas de tracking, de tal forma que cada una se complemente con el resto y así obtener sistemas más fiables y robustos [57].

A continuación se presentan con un mayor grado de detalle las características de los sistemas de tracking óptico así como los sistemas de tracking híbrido que son los que resultan de mayor interés para el presente trabajo.

2.2.1. Sistemas de tracking Óptico

Las aplicaciones de realidad aumentada requieren del conocimiento preciso de la posición y la orientación relativas de la cámara con respecto a la escena. Esto significa que aunque la cámara se desplace es necesario mantener conocimiento en tiempo real de los seis grados de libertad que definen la posición y la orientación de dicha cámara con respecto a la escena. Son varias las tecnologías utilizadas con este fin pero todas y cada una de ellas tiene sus debilidades. Los sistemas de tracking magnéticos son vulnerables a las distorsiones producidas por los metales próximos además de limitar también el rango de desplazamiento. Los sistemas de tracking ultrasónicos sufren de ruido y suelen ser inexactos para rangos amplios debido a la temperatura ambiente. Por último, los sistemas de tracking inercial pierden precisión con el tiempo.

Una de las aproximaciones más prometedora es la visión por computador ya que consiste en métodos precisos y de bajo coste. Entre las técnicas más utilizadas se encuentran las basadas en la detección de marcas. Para aplicar esta técnica es necesario añadir una serie de marcas en el entorno de trabajo, las cuales al ser detectadas permiten calcular la posición relativa de la cámara con respecto a las mismas. Cuando no se detecta ninguna de estas marcas el cálculo de la posición de la cámara falla.

Una de las primeras aproximaciones basadas en marcas es la que exponen Hoff et al. en [29]. Dicha aproximación utiliza marcas circulares concéntricas consistentes en un anillo negro sobre un fondo blanco o viceversa. Dentro de la misma línea de investigación State et al. que utilizan marcas circulares de colores para obtener una estimación más exacta [57]. Para la codificación de las marcas se utilizan cuatro colores y cada marca consiste en dos círculos concéntricos de distinto tamaño por lo que se pueden obtener doce marcas diferentes.

Otra aproximación interesante es la utilizada por Claus y Fitzigibbon [15] que, a diferencia de los métodos expuestos anteriormente que utilizan soluciones específicas, utilizan un método basado en aprendizaje para la detección de marcas con la que se obtienen importantes mejoras en la fiabilidad. Para su funcionamiento se hace necesaria la adquisición de imágenes de ejemplo tomadas desde distintas perspectivas, escalas y condiciones de iluminación, además de imágenes de entrenamiento negativas. A partir del conjunto de imágenes obtenidas se entrena un clasificador que posteriormente será utilizado para la correcta estimación de la posición y la orientación de la cámara. Los autores utilizan un tamaño de

ventana de 12x12 píxeles para las imágenes de entrenamiento. Posteriormente se procesa cada uno de los frames realizando dos pruebas sobre ellos:

- Para la primera prueba se aplica un clasificador bayesiano para un conjunto de dos píxeles en cada una de las subventanas de 12x12 píxeles del frame que está siendo analizado.
- Para la segunda prueba se aplica un clasificador de vecinos más cercanos específico del problema para todas las subventanas que han pasado la primera clasificación.

La primera de las etapas debe ser muy eficiente y obtener una tasa cercana a cero de falsos negativos y pasar un número reducido de falsos positivos. La segunda etapa consigue una alta precisión pero va asociada a un mayor coste computacional.

Hasta el momento todas las marcas utilizadas han sido circulares y solamente se utilizaba su centro por lo que se hacen necesarias varias marcas para estimar la posición y la orientación de la cámara. Koller et al. proponen el uso de marcas cuadradas basadas en un cuadro negro que contiene pequeños cuadrados rojos utilizados para la identificación de la marca [34]. Esta aproximación sigue necesitando de varias marcas para realizar los cálculos deseados.

En 1999 Rekimoto presenta un método en el que sólo hace falta el uso de una marca para la estimación de la posición y la orientación de la cámara [46]. Para ello propone un algoritmo que calcula las cuatro esquinas correspondientes a la marca. A partir de estos cuatro puntos, pertenecientes al mismo plano, se calcula una matriz que representa la traslación y la rotación de la cámara en el sistema de coordenadas global. Se trata de una aproximación robusta de bajo coste para seguimiento en tiempo real y ha dado lugar a una famosa librería software llamada ARToolkit [32]. El proceso de detección de marcas y la estimación de la posición se realiza en tiempo real y se puede aplicar a cualquier frame. El funcionamiento de la librería es adecuado siempre y cuando haya unas buenas condiciones luminosas y se puedan colocar las marcas en el entorno de trabajo.

El uso de marcas requiere de una previa preparación del entorno. Hay ocasiones en las que esto no es posible, por ejemplo en aplicaciones que funcionen en el exterior. Por tanto, siempre que sea posible es mucho mejor utilizar las características naturales del entorno para realizar la tarea de tracking. El uso de estas características hacen que este proceso sea notablemente más complejo. Existen dos técnicas aplicables para aprovechar las características naturales del entorno:

- Métodos basados en la detección de bordes o aristas.
- Métodos que utilizan la información proporcionada por los píxeles que se encuentran dentro de la proyección del objeto.

Las primeras aproximaciones en aparecer para la realización de tareas de tracking son las basadas en detección de bordes debido a que estos métodos son computacionalmente eficientes y son relativamente sencillos de implementar. Además se comportan bastante bien frente a los cambios en la iluminación.

RAPiD [26] fue uno de los primeros sistemas de tracking 3D en obtener buenos resultados en tiempo real y muchas mejoras se han propuesto desde entonces. La idea principal en la que se basa consiste en considerar una serie de puntos 3D pertenecientes al objeto,

denominados puntos de control, que se encuentran situados en los bordes de alto contraste de la imagen. Una vez obtenidos los puntos de control, el movimiento 3D del objeto entre dos frames consecutivos puede ser recuperado a partir de los desplazamientos en 2D de los puntos de control.

Una vez inicializado, el sistema ejecuta un bucle. Para cada frame, la predicción de la posición, que se puede obtener a partir de la posición obtenida en el frame anterior, se utiliza para predecir qué puntos de control serán visibles y cual debería ser su nueva localización.

El principal inconveniente de RAPiD es la falta de robustez. Se han llevado a cabo muchas investigaciones que pretenden hacer de RAPiD un método mucho más robusto. Muchas se basan en estimación robusta en lugar de en el método de la minimización de diferencias al cuadrado [38][21][56].

Además de los métodos basados en puntos de control, expuestos anteriormente, existen otras aproximaciones que utilizan la información aportada por las texturas de los objetos. Si el objeto está suficientemente texturizado la información se puede obtener a partir del flujo óptico [16], comparación de plantillas [35] o correspondencias entre puntos de interés [23]. Aunque el método más efectivo para aplicaciones de realidad mixta es el último debido a que se basa en las características locales del objeto. Además son insensibles a oclusiones parciales o errores de correspondencia.

En las aproximaciones basadas en puntos de interés en lugar de comprobar todos los píxeles de la imagen solo algunos de ellos se seleccionan, utilizando un operador de interés antes de realizar la correspondencia entre los puntos. Las características deseables para un operador de interés fueron descritas por Forstner [23]. Los puntos elegidos deben ser diferentes de sus puntos vecinos y además la selección de puntos debería ser repetible, esto es, que los mismos puntos deberían ser elegidos en diferentes imágenes de la misma escena. La precisión y la fiabilidad de la correspondencia entre puntos depende directamente de la invariancia de los puntos de selección. Algunos detectores de puntos de interés exitosos [23][55] se basan en la matriz de autocorrelación calculada para cada posición de los píxeles. Se trata de una matriz cuadrada de dimensión dos cuyos coeficientes son sumas a partir de la primera derivada de la intensidad de la imagen con respecto a las coordenadas del pixel. Con esto se miden las variaciones locales en la imagen.

En ausencia de puntos cuyas coordenadas son conocidas a priori, todos los métodos están ligados a una acumulación de error, lo que se traduce en fallos en el proceso de tracking. Una de las soluciones a este problema es el uso de una serie de *keyframes*, es decir, imágenes del objetivo o escena para las que la cámara ha sido registrada de antemano. En tiempo de ejecución, cada una de las imágenes se puede comparar con cada uno de los keyframes para proporcionar una estimación acertada. Aunque esta aproximación es más compleja que comparar un frame con los inmediatamente anteriores, debido a que la diferencia en los puntos de vista suele ser mucho más amplia. Por tanto el algoritmo encargado de calcular las correspondencias entre los puntos debe ser rápido e insensible a las distorsiones de perspectiva, lo que no es habitual en los algoritmos que necesitan manejar pequeñas distorsiones entre frames consecutivos.

Los sistemas de tracking recursivos se basan en las estimaciones realizadas sobre los frames anteriores y hacen que la identificación de las características de la imagen sea relativamente fácil. Aunque también presenta varios inconvenientes.

- El sistema debe ser inicializado a mano o requiere que la cámara se encuentre muy cerca de una posición especificada.

- Se trata de sistemas realmente frágiles. Si existe algún problema entre dos frames consecutivos, como por ejemplo la oclusión completa del objeto analizado, el sistema dejará de funcionar correctamente y tendrá que ser reiniciado.

Estas deficiencias hacen que los sistemas de tracking recursivos no sean usables a efectos prácticos y son las que han hecho que ARToolkit[1] sea un sistema muy popular entre la comunidad. ARToolkit es el primer sistema basado en visión por computador que supera estas limitaciones, siendo capaz de detectar las marcas en cada uno de los frames sin poner ninguna restricción sobre la posición de la cámara.

De todas formas obtener el mismo nivel de rendimiento sin tener que preparar el entorno todavía sigue siendo un objetivo deseable.

2.2.2. Sistemas de Tracking Híbrido

Los sistemas de tracking híbrido aparecen para paliar los problemas presentados por los métodos de tracking tradicionales, basados en una sola aproximación. Dichos sistemas usan dos o más métodos de los anteriormente descritos (ver sección 2.2) con el objetivo de superar las limitaciones existentes al utilizar un solo método de tracking.

Uno de los primeros sistemas de tracking híbrido en aparecer fue el propuesto por State et al. [57]. Los autores plantean un sistema que combina la precisión de los métodos de tracking ópticos y la robustez de los métodos de tracking magnéticos. El principal elemento de tracking son las marcas dispuestas en el entorno, que además se utilizan para calibrar, continuamente, el dispositivo de tracking magnético. Además el sistema magnético se utiliza con los siguientes objetivos:

- **Aceleración del análisis de la imagen**: El dispositivo magnético ayuda a reducir el área de búsqueda en las imágenes capturadas, acelerando el proceso de búsqueda de las marcas.
- **Selección entre varias soluciones**: Cuando la ecuación que obtiene la posición del usuario tiene varias soluciones, el método de tracking magnético se utiliza para elegir la más adecuada.
- **Tracking de apoyo**: Cuando no se detecta ninguna marca en el entorno, el sistema de tracking magnético proporciona la posición y la orientación del usuario.
- **Comprobación de los resultados conseguidos con el sistema óptico**: Los datos obtenidos por el tracking magnético se comparan con los proporcionados por el óptico. Así se evitan diferencias sustanciales y como consecuencia se obtiene un método mucho más preciso, sin grandes variaciones.

Posteriormente, aparecieron otros sistemas de tracking híbrido que combinan los métodos de tracking óptico con los de tracking magnético como el planteado por [4]. El método que se propone no guarda gran diferencia con lo expuesto anteriormente. Los autores utilizan el sistema de tracking magnético para reducir las zonas de búsqueda en la imagen capturada y para evitar los errores del sistema óptico. Además, solamente se analizan 10

[1] Página oficial de ARToolkit: http://www.hitl.washington.edu/artoolkit/

frames/segundo, que no es suficiente para que el usuario tenga una visión fluida del entorno.

Debido a que los dispositivos magnéticos pueden producir medidas erróneas a causa de la presencia de metales cercanos, se comenzaron a investigar otras aproximaciones de tracking híbrido. Las dos aproximaciones más utilizadas son las de tracking óptico en conjunción con las de tracking inercial. La gran mayoría de los sistemas analizados para este trabajo combinan ambas tecnologías para obtener la posición y la orientación del usuario.

Uno de los primeros sistemas de este tipo fue el propuesto por You et al. [62][5]. Está compuesto un sistema óptico y un sistema inercial compuesto por tres giróscopos. Los giróscopos aumenta la robustez y la eficiencia del sistema de visión proporcionando una estimación de la orientación de la cámara. Debido a la integración de las velocidades angulares obtenidas a partir de los giróscopos, el sistema inercial está sujeto a un error que crece según aumenta el tiempo (este error recibe el nombre de *drift*), para corregirlo se utiliza el sistema de visión. Además, se utiliza una brújula que proporciona en todo momento la orientación del usuario con respecto al norte, con un error de +/- 0.5 grados. Toda la información recuperada a partir de los sensores se fusiona y se obtiene una estimación de la orientación del usuario. El principal inconveniente de este sistema es que el proceso de tracking no se ejecuta en tiempo real, por lo que es necesario grabar las imágenes previamente para aplicar, posteriormente, el proceso de fusión y de tracking. Además solo se consiguen tres grados de libertad (la orientación de la cámara) cuando lo deseable es obtener seis grados de libertad (posición y la orientación de la cámara).

Satoh et al. [50] plantean un sistema muy parecido pero que consigue un rendimiento adecuado para obtener la orientación de la cámara a una tasa similar al *framerate* del vídeo. Según los autores, utilizar una técnica robusta de visión por computador que calcula la orientación de la cámara para obtener una mayor precisión, se traduce en una significante pérdida de rendimiento. Para depender en menor parte del sistema óptico, se propone el uso de sensores de alta precisión que introducen un error muy pequeño. Además, utilizan un conjunto de acelerómetros para corregir el error que se produce al calcular la orientación a partir de los giróscopos.

Ribo et al. [48] proponen, en 2002, un sistema híbrido que consigue seis grados de libertad, es decir, proporciona la posición y la orientación del usuario en cada uno de los ejes de coordenadas (x, y, z). Además, este sistema es capaz de funcionar en exteriores. Para ello, y como los dos anteriormente descritos, utilizan de forma conjunta un sistema de tracking óptico y otro inercial. Para realizar el tracking con el sistema óptico, los autores disponen de un modelo 3D fotorrealista del que extraen un conjunto de puntos de interés (no colineales) que posteriormente se utilizarán como marcas naturales.

Para el tracking inercial se hace uso de un sistema inercial compuesto por tres acelerómetros y tres giróscopos que miden la aceleración y la velocidad angular en cada uno de los ejes de coordenadas. Además, utilizan un procesador de la señal digital (DSP) que manipula los datos en brutos leídos por los sensores y se comunica con la computadora que ejecuta la aplicación. Para la fusión de la información, servida por el conjunto de sensores, se utiliza un filtro de Kalman extendido (EKF) [31][61]. El vector de estado en el momento de tiempo t contiene la posición (p), la velocidad (v), la aceleración (a), el sesgo de la aceleración (a_b) y la orientación (q). El filtro de Kalman utiliza el vector de estado en el momento anterior (t-1), y las medidas obtenidas en el instante actual (t) para calcular el nuevo vector de estado. Los valores obtenidos a partir de los sensores, se utilizan para obtener el vector estado en el momento de tiempo t a partir del vector anterior. Después, los valores de la posición (p_v) y la orientación (q_v) obtenidos por el sistema óptico se utilizan para corregir el nuevo vector

de estado y obtener el vector definitivo. De esta forma, se consigue depender del sistema inercial mientras se calcula la posición y orientación utilizando el sistema óptico.

Con la proliferación de los sistemas de tracking híbridos se hace necesaria una metodología para integrar los valores obtenidos por los diferentes métodos de tracking utilizados. Esta es la idea de Schwald et al. [54]. En su artículo, los autores presentan una forma genérica para la composición de varios sistemas de tracking diferentes, independientemente de su naturaleza. Según exponen en el trabajo, el hecho de componer la información obtenida por los distintos sistemas se puede dividir en dos:

- Determinar las transformaciones necesarias para toda la información de tracking, de tal forma que toda esta información se pueda representar utilizando un mismo sistema de coordenadas.
- Transformar las estimaciones a dicho sistema de coordenadas.

Para conseguir el objetivo, cada sistema de tracking se trata como un módulo independiente que emite la información generada, junto con un valor de calidad y una marca de tiempo a una unidad de control. Esta unidad de control se encarga de aplicar todas las transformaciones necesarias y proporcionar la información en el sistema común de coordenadas a la aplicación principal. Para que esto sea posible, es necesario establecer una configuración inicial que incluye información acerca de cada uno de los dispositivos de tracking disponibles, el número de sensores de los que consta, la transformación necesaria para cada dispositivo y un identificador único para las comunicaciones.

Además de la configuración, necesaria para las transformaciones entre los sistemas de coordenadas de los dispositivos de tracking y el sistema de coordenadas global, es necesario definir un método de fusión para obtener una sola estimación de la posición y la orientación del usuario. Los autores dan total libertad en la elección del sistema de fusión. Como un ejemplo, proponen la obtención de la media de todos los valores o, si se prefiere dar más preferencia a algunos dispositivos de tracking que a otros, el uso de un filtro de Kalman [61] para realizar una fusión más compleja.

Un ejemplo de aplicación de RA que combina múltiples fuentes de tracking es el que plantean Hallaway et al. [25]. En este trabajo se presenta una aplicación de realidad aumentada para asistir la navegación en espacios desconocidos que combina múltiples fuentes de tracking. El sistema se ha diseñado para proporcionar la información de tracking en tres ambientes diferentes: dentro de un laboratorio, en el pasillo del edificio y otras habitaciones del mismo edificio y en exteriores. Dependiendo del entorno en el que se encuentre el usuario, se emplea una aproximación diferente para el proceso de tracking. Además, si el usuario se encuentra dentro del propio edificio, el tracking se realiza utilizando cierto conocimiento ambiental. En concreto, este conocimiento consiste en un mapa espacial que representa la geometría del edificio y un grafo de accesibilidad que contiene información acerca de los caminos que puede seguir el usuario, partiendo de su posición actual.

Otra aproximación interesante es la propuesta por Bourgeois et al. [9]. Se plantean dos métodos de tracking diferentes. El primero consiste en un sistema de detección de marcas pero con una particularidad: los autores definen dos tipos diferentes de marcas, que son:

- **Marcas de colores**: Las marcas de colores permiten identificar los objetos sobre los que están colocados y obtener la posición y la orientación de la cámara.

- **Marcas simples**: Las marcas simples consiste en una serie de puntos blancos colocados sobre un fondo negro. Estas marcas son fácilmente detectables en la imagen capturada y se utilizan para refinar el proceso de tracking y obtener unos resultados más exactos.

El segundo método de tracking, propone el uso de las marcas anteriormente descritas, junto con un modelo 3D que se utiliza para mejorar la robustez del proceso de tracking. Partiendo de la premisa de que no todas las marcas están visibles en un momento determinado, y que los falsos positivos pueden afectar negativamente al proceso de tracking, los autores utilizan modelos 3D para descartar las marcas que están ocultas. Para esto, es necesario tener una estimación inicial de la posición y la orientación del usuario en el entorno, a partir de aquí, las marcas no visibles se irán descartando. Además, esto acelera el proceso de tracking, porque las marcas que descarte el sistema no se contemplarán a la hora de hacer la detección.

Actualmente, la mayoría de las aproximaciones de sistemas de tracking híbridos utilizan un sistema óptico y uno inercial. Las diferencias surgen en la forma de fusionar la información obtenida por ambos sistemas. Muchas de las aproximaciones utilizan un filtro de Kalman extendido (EKF), aunque comienzan a aparecer otras ideas para la fusión de la información. Una de ellas es la propuesta por Ababsa et al. [1]. Los autores proponen un filtro de fusión basado en un filtro de partículas [20]. Según se puede comprobar en los resultados, el método de fusión aplicado parece tener un rendimiento mejor que el EKF.

2.3. Aplicaciones dependientes del Contexto

El uso del contexto es importante en aplicaciones interactivas, especialmente, en aquellas aplicaciones en las que el contexto del usuario cambia rápidamente. En las aplicaciones interactivas tradicionales el usuario está muy limitado a la hora de introducir información al computador. Como consecuencia de esto, los computadores no pueden obtener el máximo provecho del contexto en el que se encuentra el usuario. Si se mejora el acceso, por parte del computador, al contexto del usuario, se podrán proporcionar servicios computacionales mucho más útiles.

Existe una gran variedad de definiciones de contexto. Por ejemplo, por Schilit y Theimer [51] lo definen como la localización del usuario, identidades de la gente y objetos cercanos y los cambios que sufren dichos objetos. Otra posible definición es la que dan Brown et al. [14] que definen el contexto como la localización, identidad de personas cercanas, hora del día, estación temperatura, etc. Otras definiciones de contexto se pueden encontrar en [49][18]. *En términos generales, el* ***contexto*** *es cualquier tipo de información que se utiliza para particularizar la situación de una entidad. Una entidad es una persona, lugar u objeto que se considera importante para la interacción entre un usuario y una aplicación, ambos incluidos* [19].

Como se puede comprobar, no se llega a un acuerdo a la hora de definir qué es el contexto. La gran mayoría de dichas definiciones incluyen, entre otra información, la **localización** del usuario. Pero el contexto no es solo la localización [53], dependiendo de la naturaleza de la aplicación se podrá definir el contexto de una forma u otra. En las aplicaciones de RA aumentada la localización del usuario es de vital importancia para su correcto funcionamiento. En función de esta localización y de otros factores, una aplicación de RA puede suministrar una serie de servicios avanzados que facilitarán su uso por parte de los usuarios.

Al igual que ocurre con el contexto existe un gran número de definiciones para las aplicaciones dependientes del contexto. Según [19] *un sistema es dependiente del contexto si utiliza el contexto para proporcionar información relevante y/o servicios al usuario, donde la relevancia depende de la tarea que quiera ejecutar el usuario.* Según esta definición un sistema es dependiente del contexto si se adapta de forma automática al contexto particular del usuario.

Para que una aplicación sea dependiente del contexto, ésta tiene que ser capaz de adquirir el contexto en el que se encuentra el usuario en un momento determinado. La adquisición del contexto se puede llevar a cabo de dos formas diferentes:

- **Explícita**: Es el propio usuario el encargado de introducir el contexto. Por ejemplo, el login es una forma explicita de obtener el contexto. El usuario se identifica frente al sistema y sus preferencias serán cargadas de forma automática. El sistema se adapta al contexto pero es el propio usuario el que debe introducir la información.
- **Implícita**: La información se adquiere de forma totalmente transparente al usuario. Por ejemplo, un sistema de tracking proporciona la localización de un usuario de forma automática, en función de esta localización se puede personalizar la información presentada.

Una aplicación de RA maneja una gran cantidad de información y de modelos. Dependiendo del tipo de usuario, de su intención y de su localización en el espacio, la aplicación se puede adaptar de forma automática y proporcionar la información más adecuada en cada momento.

2.3.1. Trabajos previos

El uso del contexto es importante en las aplicaciones interactivas. El contexto es cualquier información que puede utilizarse para clasificar la situación de una entidad [19]. Este tipo de información es particularmente valiosa en las aplicaciones de realidad aumentada, permitiendo la aplicación de roles o la creación de contenido personalizado basado en el contexto. Existen algunos trabajos previos que hacen uso de la consciencia del contexto. A continuación se describen algunos de ellos:

- En el proyecto **CyberGuide** [3] se han desarrollado una serie de guías turísticas que funcionan tanto en interiores como en exteriores. En ellas, según se mueve el usuario, se actualiza su posición en un mapa y además se le proporciona información de los lugares que está visitando en cada momento. También en este proyecto se trabaja con el llamado *contexto histórico* que consiste, en este caso concreto, en estudiar las estadísticas de sitios más visitados por los usuarios, así como las reacciones de los mismos, para poder determinar qué visitas son las más aconsejables.
- Una propuesta distinta es la que aportan los documentos **Stick-e** [13]. Estos documentos contienen, además de un contenido que proporcionará información al usuario, una descripción del contexto. Para modelar este contexto se han creado una serie de variables como la localización, la proximidad a otros objetos o el tiempo. Además como ejemplos de utilización de este tipo de documentos se han implementado varias aplicaciones, como por ejemplo un *tour* guiado por museos, que mediante las variables de localización y proximidad a los objetos muestra una información u otra al usuario.

- **Augment-able Reality** [47] es un sistema de realidad aumentada que muestra diferente información a los usuarios dependiendo del contexto. La principal novedad que aporta este sistema es que los propios usuarios pueden modificar la información que se va a mostrar e incluso incluir nuevas situaciones contextuales con nuevo contenido.
- **The Interactive Office** [28] es un prototipo que proporciona soporte a las actividades diarias en una oficina. El sistema trata de ahorrar al usuario la realización de diversas acciones que podrán interrumpir su trabajo. El contexto de cada uno de los usuarios se obtiene utilizando un conjunto de sensores cuya información se fusiona.
- **Classroom 2000** [2] es una aplicación que utiliza la computación ubicua en la educación. El propósito de este proyecto es utilizar herramientas automáticas para obtener material de clase que, posteriormente, la aplicación se encarga de integrar y hace accesible vía *web*. El uso del contexto en este caso es el momento en el ocurren los distintos eventos, y se utiliza con el objetivo de sincronizar los materiales que se pondrán al alcance de los usuarios.

3

Descripción de la Propuesta

3.1. Análisis de Contexto en Aplicaciones de Realidad Aumentada

Las aplicaciones de Realidad Aumentada manejan una gran cantidad de datos para proporcionar información que enriquezca el mundo real. Con dicho propósito en mente, este trabajo pretende hacer uso del contexto en el que se encuentra sumergido el usuario. Dependiendo del contexto, una misma pregunta puede tener diferentes respuestas. Es necesario realizar un estudio que analice cuál es el contexto a utilizar para mejorar la calidad de los datos que serán manejados en el entorno de RA.

Con este propósito definimos el **contexto** como toda la información relevante acerca de un usuario y del entorno en el que está inmerso [53]. Esta información permitirá la selección en cada instante de tiempo de los modelos e información más adecuados para la situación actual del usuario.

El diseño de un sistema dependiente del contexto debe contemplar seis aspectos fundamentales [30][27]:

- **Detección del contexto**: Hace referencia a la capacidad de capturar el contexto del usuario y adaptar la información presentada al usuario.
- **Representación explícita**: Hace referencia al conocimiento previo sobre el entorno y sus elementos. Se puede representar utilizando lenguajes formales o técnicas específicas diseñadas para una aplicación concreta.
- **Adaptación Contextual**: Capacidad de modificar el comportamiento de forma automática según los cambios experimentados por el contexto.
- **Descubrimiento de recursos del contexto**: Capacidad de encontrar y utilizar recursos relacionados con el contexto actual.

- **Consultas del contexto**: Debido a la gran cantidad de información manejada por los sistemas dependientes del contexto, se deben proporcionar métodos de consulta a dicha información.
- **Aumento Contextual**: Hace referencia a la capacidad de obtener información de más alto nivel a partir del conocimiento actual.

En la mayoría de las aplicaciones no es necesario desarrollar todas las funcionalidades especificadas anteriormente, pero como consecuencia del dinamismo de la información contextual, todas serían deseables. Debido a la heterogeneidad de las fuentes de información es necesario implementar técnicas eficientes de adquisición del contexto. La captura del contexto puede ser muy problemática y además, la información contextual puede no ser completa en todo momento. Es importante tener presente esta limitación a la hora de desarrollar aplicaciones dependientes del contexto y habilitar mecanismos que permitan trabajar con incertidumbre.

En una aplicación de Realidad Aumentada el contexto juega un papel muy importante. Cuanta más información se tenga acerca del usuario de la aplicación y del entorno en el que se está ejecutando, más se personalizarán los servicios que proporciona el sistema. La información que se considera más relevante para el propósito de este trabajo incluye la posición global del usuario, el Rol del usuario, la Intención del usuario y las características técnicas del equipo portado por el usuario entre otras.

3.1.1. Clasificación del Contexto

Debido a la naturaleza heterogénea del contexto es necesario establecer una clasificación que nos permita distinguir entre los diferentes datos adquiridos y su uso para la obtención de información a un mayor nivel que la obtenida inicialmente. Con este propósito en mente, hemos definido dos tipos diferentes de contexto:

- **Contexto Estático**: Se define como contexto estático toda aquella información que permanece invariable durante el tiempo de ejecución de la aplicación. Esta información queda definida a priori y será la misma para todos los usuarios que utilicen el sistema. Todo el contexto estático estará representado de forma explícita.
- **Contexto Dinámico**: Se define como contexto dinámico toda aquella información que cambia durante el tiempo de ejecución de la aplicación. Cada usuario tendrá su propio contexto dinámico, haciendo que el comportamiento de la aplicación se adapte a dichos valores. La información de contexto dinámico debe actualizarse con una frecuencia determinada.

Para obtener el contexto global del usuario es necesario utilizar tanto la información proporcionada por el contexto estático como la proporcionada por el contexto dinámico. Ambas informaciones se complementan mutuamente y permiten la extracción de nuevo conocimiento, como por ejemplo la posición en el sistema global de coordenadas del usuario o el nivel de especialización que tiene el usuario a la hora de realizar una tarea específica.

Tradicionalmente, las aplicaciones de RA sólo tienen en cuenta la información asociada a la posición del usuario. Es necesario conocer este valor a la hora de mostrar información 3D generada por computador, ya que se utilizará para visualizar los modelos desde un punto

de vista adecuado. En este trabajo se considera que, además de la posición, existen muchos más aspectos relevantes que se pueden utilizar para asistir al usuario de una forma mucho más personalizada y precisa:

- Dentro del **contexto estático** se considera relevante la información que hace referencia al **lugar** en el que se desarrolla el proceso completo, el **agente** que lo está desarrollando y toda la **información estructural** referente a la posición en el espacio de las marcas 2D desplegadas.
- Dentro del **contexto dinámico** es importante diferenciar la información asociada al agente de la información asociada al lugar en el que se está desarrollando la acción.

3.1.2. Representación del Contexto Estático

La información estructural que se refiere a la posición en el espacio de una marca se representa mediante una matriz de transformación. Estas matrices representan la posición y orientación relativas de un elemento con respecto a un sistema de coordenadas global establecido. Esta información debe ser calculada a priori e introducida de forma explicita para su posterior uso.

Cuando una marca se detecta, es posible obtener la matriz de transformación de esta marca con respecto a la cámara que la está detectando. Esto hace que el sistema de coordenadas de la cámara actúe como sistema de referencia global. Haciendo uso de estas matrices podemos mostrar la información relacionada con la marca que está siendo detectada en ese momento. Además se pueden realizar transformaciones geométricas sobre los modelos, pero siempre con respecto del sistema de coordenadas de la cámara.

Lo realmente interesante, es la posibilidad de obtener la posición global del usuario tomando como referencia el sistema global de coordenadas establecido a priori. Para tener conocimiento de dicha posición global, se emplea la matriz asociada a cada una de las marcas mediante la composición de una serie de operaciones matriciales. Con estas operaciones, se lleva a cabo un cambio de sistema de referencia y se puede obtener la posición global de la cámara que está detectando la marca. Conociendo ésta, se puede desplegar la información seleccionada de forma absoluta.

En el modelado del contexto estático resulta muy útil definir diferentes niveles de interacción. De esta forma se han definido dos tipos de marcas:

- **Marcas de posicionamiento**: Las matrices proporcionadas cuando se detecte alguna de estas marcas se utilizarán para obtener la posición global de la cámara dentro del entorno controlado en el que se está ejecutando la aplicación. Para ello es necesario procesar la información obtenida inicialmente.
- **Marcas de actuación**: Las matrices proporcionadas cuando se detecta alguna de estas marcas se utilizarán para representar los modelos de forma local, esto es, hacer de la cámara el sistema de referencia global y representar los modelos con respecto a éste.

Toda la información estructural así como la información asociada a cada una de las marcas será serializada y cargada al inicio del sistema.

Por último, para poder utilizar un conjunto de cámaras con el propósito de posicionar al usuario en el espacio 3D, es necesario conocer la relación existente entre las cámaras que

porta el usuario. Una de las cámaras se utilizará como **cámara principal**. Éste dispositivo será el encargado de proporcionar la información sobre el mundo real. El resto de cámaras disponibles se utilizarán como soporte para calcular la posición global del usuario. Para ello, también es necesario conocer la relación existente entre cada una de las cámaras secundarias y la cámara principal. Esta relación también estará representada por una matriz de transformación que nos permite cambiar entre los diferentes sistemas de referencia de cada una de las cámaras.

3.1.3. Representación del Contexto Dinámico

El contexto dinámico está representado por toda aquella información relevante que cambia durante el tiempo de ejecución. Se han definido tres campos para representarlo:

- **Localización**: Este atributo almacena la posición del usuario en el espacio. Esta localización está representada por una matriz de 3x4, denominada matriz de posición.
- **Rol**: Este atributo está compuesto por el tipo de usuario. El rol establece diferentes permisos que se utilizarán para acceder al repositorio de modelos. Por lo tanto, dependiendo de este valor se limitará el acceso a unos modelos u otros.
- **Percepciones**: Este atributo representa la percepción que se tiene, en un determinado momento, sobre el entorno. Estas percepciones se componen a su vez de percepciones físicas y percepciones funcionales. Las físicas son aquellas que representan es estado actual del entorno percibido. Las funcionales representan la información obtenida a partir de las percepciones físicas anteriores.

La **localización** del usuario se calculará a partir de las **percepciones** obtenidas en cada iteración. Se creará una percepción por cada una de las marcas detectadas. En un proceso posterior, todas las percepciones obtenidas se fusionarán para obtener una estimación aproximada de la posición del usuario dentro del espacio controlado. En las iteraciones posteriores, se utilizarán las percepciones sensoriales para estabilizar la localización obtenida (ver subsección 3.1.4).

Todos los campos que están asociados con el contexto dinámico deben ser calculados en cada iteración del sistema, con la información que haya disponible en dicho instante de tiempo. Si no es posible calcular todos los valores de forma adecuada se tendrá una vista parcial o nula del contexto dinámico.

3.1.4. Adquisición del Contexto Dinámico

La adquisición del contexto dinámico se centra, básicamente, en la captura de las percepciones para construir la localización del usuario. Existen varios métodos para obtener la localización del usuario que pueden ser representados mediante una pirámide (ver figura 3.1), en la que las posiciones superiores implican una mayor precisión de la estimación debido a que están sujetas a un menor error dinámico. Los elementos que ocupan la parte inferior son más inexactos pero se pueden utilizar en entornos que no han sido preparados previamente.

Para la composición de las percepciones y de la localización se utilizarán los dos niveles superiores de la pirámide. El sistema óptico proporcionará una estimación muy aproximada

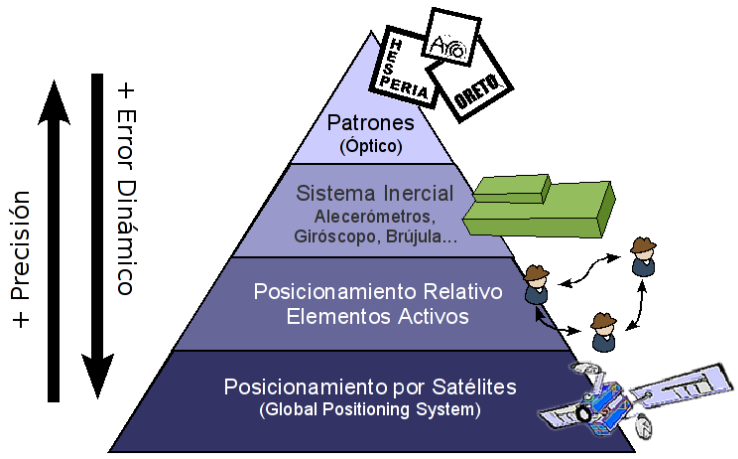

Figura 3.1: Métodos disponibles para el cálculo de la localización

de la localización absoluta del usuario. El sistema inercial sufre mucho error, debido a que la localización se obtiene integrando los valores adquiridos a través de los sensores. Esto hace que un pequeño error se propague en el tiempo, proporcionando valores muy poco precisos.

3.1.4.1. Sistema Óptico

El sistema óptico está compuesto por un conjunto de cámaras, cada una de ellas colocada de forma fija con respecto a las demás. La relación existente entre las cámaras se considera contexto estático y por tanto es necesario conocerla en cada ejecución del sistema. Las cámaras obtienen un frame en cada iteración del sistema y se analiza su contenido en busca de una o varias marcas, independientemente de si se trata de marcas de posicionamiento o marcas de actuación.

Cuando un frame capturado contiene una marca de posicionamiento, se obtiene una matriz de transformación de la marca con respecto al sistema de coordenadas de la cámara (T_{MC}). Para obtener el posicionamiento del usuario dentro del sistema de coordenadas global definido, es necesario conocer la matriz de transformación de la cámara con respecto al sistema de coordenadas de la marca, para ello, basta con invertir T_{MC}. Si conocemos la matriz de transformación de la marca detectada (N) con respecto al sistema global de coordenadas (M_{NG}), basta con multiplicar ésta por la anteriormente obtenida. La matriz resultante se corresponde con la de transformación en el sistema global de coordenadas (T_{CG}):

$$T_{CG} = M_{NG} * T_{MC}^{-1}$$

El método tradicional de representación de información realiza transformaciones sobre los modelos tomando como referencia el sistema de coordenadas de la cámara. Como consecuencia, no es posible representar un modelo global dependiendo del punto de vista del usuario. Obteniendo la matriz de transformación con respecto al sistema de coordenadas

global, en lugar de realizar la transformación sobre los modelos, se realizará sobre la cámara, colocándola en la posición adecuada. Así, se puede mostrar toda la información con respecto al origen global y al colocar la cámara el usuario solo percibirá la parte que corresponda.

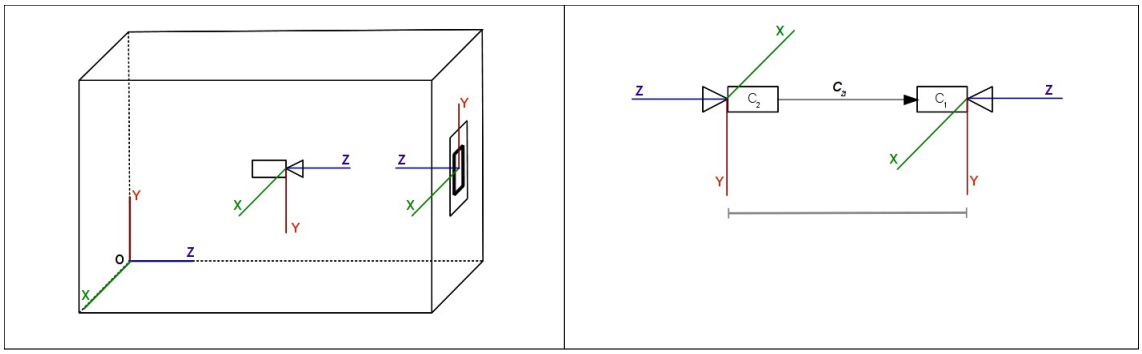

Figura 3.2: Relación entre sistemas de referencia

El problema de los sistemas de *tracking* ópticos basados en marcas, es que cuando la cámara pierde la referencia de la marca, no se puede obtener ninguna matriz de posición y por lo tanto, el usuario deja de percibir la información generada virtualmente. Con el propósito de evitar este efecto en la medida de lo posible, se utilizará más de una cámara para el proceso de *tracking*.

El hecho de disponer de varias cámaras aumenta la probabilidad de que una de ellas detecte alguna de las marcas de posicionamiento. Si esto ocurre, es posible obtener la posición global de la/las cámaras que detecten alguna marca y después realizar una transformación sobre esta matriz y obtener la posición global de la cámara principal. Para poder hacer esta transformación es necesario conocer la matriz que relaciona los sistemas de coordenadas de ambas cámaras (ver figura 3.2), en concreto, de cada una de ellas con respecto de la cámara principal.

Por ejemplo, suponiendo que la cámara 2 detecte una marca se obtendría su matriz de transformación en el sistema de coordenadas global (T_{C_2G}). Si además conocemos la matriz C_{21}, que es la de transformación entre los sistemas de referencia de la cámara 2 con respecto a la cámara 1, es posible obtener la matriz de transformación global de la cámara principal T_{C_1G} mediante la siguiente operación:

$$T_{C_1G} = T_{C_2G} * C_{21}$$

En una aplicación de RA real una cámara puede detectar más de una marca en un mismo instante de tiempo. Esto puede ayudarnos a mejorar el proceso de *tracking* o, por el contrario, introducir ruido en dicho proceso si la detección se realiza de forma errónea. Para descartar las marcas que pueden perjudicar al proceso de *tracking*, solamente se consideran aquellas cuya probabilidad de haber sido detectada correctamente supera un cierto umbral.

Cuando alguna de las cámaras disponibles detecta una marca, con una probabilidad superior al umbral establecido, se crea una **percepción**. Una **percepción** estará formada por los siguientes campos:

Cuadro 3.1: Componentes del sistema inercial.

Sensores	Modelo	Información
3 acelerómetros	ADXL330	(a_x, a_y, a_z)
2 giroscopos	IDG300	(ω_x, ω_z)
Brújula electrónica	CMPS03	º con respecto al norte
Arduino	Diecimila	Datos recolectados de los sensores

- **Identificador de cámara**: El número asociado a la cámara que ha detectado la marca.
- **Identificador de marca**: El identificador de la marca que se corresponde con la percepción.
- **Ángulo**: Este valor representa el ángulo que forman el eje Z del sistema de coordenadas de la cámara con el eje Z del sistema de coordenadas de la marca. Este valor se utilizará posteriormente para ponderar cada una de las percepciones.
- **Transformación relativa**: La matriz de transformación de la marca con respecto al sistema de coordenadas de la cámara.
- **Transformación absoluta**: La matriz de transformación global obtenida a partir de la marca detectada.

Por lo tanto, si una cámara ha detectado N marcas se dispondrá de un vector de percepciones con N instancias. Para obtener la matriz global definitiva, primero es necesario obtener cada una de las matrices globales asociada a cada percepción. Posteriormente, se realiza una ponderación de cada matriz global dependiendo del ángulo que forma con el vector Z del sistema de coordenadas de la cámara (cuanto más de frente se detecten las marcas, más fiables son los valores obtenidos). Por tanto, para una cámara se tiene:

$$M_{C_iG} = \sum_{j=0}^{npercepts} angle_j * absolute_j$$

$$T_{C_iG} = \frac{M_{C_iG}}{\sum_{j=0}^{npercepts} angle_j}$$

Nuevamente, es necesario fusionar las matrices de transformación globales calculadas para cada cámara, obteniendo una matriz de transformación global para la cámara principal. Para ello se transforman al sistema de coordenadas de la cámara principal y se calcula la media de todas ellas.

$$T_{C_1G} = \frac{\sum_{i=0}^{ncameras} (T_{C_iG} * C_{i1})}{ncameras}$$

3.1.4.2. Sistema Inercial

Además del sistema de *tracking* óptico, la arquitectura propuesta cuenta con un sistema inercial que proporciona en todo momento la orientación del usuario (3DOF). Los componentes del sistema están especificados en la tabla 3.1.

Utilizando los acelerómetros y los giróscopos disponibles se puede obtener una estimación de la orientación del usuario con respecto a los ejes x y z. La aproximación utilizada consiste en obtener una estimación a partir de los acelerómetros, otra de los giróscopos y fusionar ambas.

Los acelerómetros que componen el sistema inercial proporcionan la aceleración experimentada en cada uno de los ejes. Debido a que la gravedad es una aceleración se puede utilizar para calcular la desviación con respecto de la vertical de los ejes x (*roll*) y z (*pitch*). Como se puede comprobar el la figura 3.3 el eje local del acelerómetro es perpendicular al objeto. Además el ángulo que forman el eje del acelerómetro y la gravedad se corresponde con la medida del *pitch*. Para calcular el ángulo Θ:

$$Acel = cos(\Theta) * G \qquad \Theta = acos(\frac{Acel}{G})$$

Como $pitch = \Theta + 90^o$ entonces:

$$pitch = asin(\frac{Acel}{G})$$

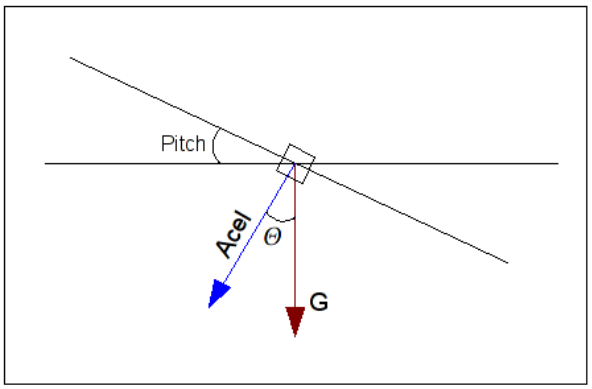

Figura 3.3: Obtención del pitch del cuerpo rígido

El valor de *roll* se obtiene de forma similar. El problema al que están asociados los acelerómetros es que si el objeto está experimentando alguna fuerza externa, las medidas proporcionadas por los acelerómetros no son válidas para calcular los valores de *roll* y *pitch*. Utilizando los giróscopos se puede obtener otra estimación de los grados de giro, para corregir el error cometido por los acelerómetros. Los giróscopos proporcionan velocidades angulares en cada eje (en este caso, los ejes x y z). Integrando estos valores se pueden conocer el *roll* y *pitch*.

$$\int(\omega_x) = pitch$$

$$pitch_i = pitch_{i-1} + \omega_x * \Delta t$$

El problema de utilizar giróscopos para obtener una estimación de la orientación de la cámara es necesario integrar los valores proporcionados. Esto hace, que cualquier error

cometido, por pequeño que sea, se propague en el tiempo, produciendo una desviación de los valores denominada **drift**.

Para integrar las dos estimaciones y obtener una medida de la orientación del usuario se ha utilizado un filtro de Kalman [31]. El filtro diseñado utiliza la velocidad angular calculada por los giróscopos para proyectar la estimación del giro en el instante anterior de tiempo, al instante actual. Después, con la estimación obtenida a partir de los acelerómetros, se corrige la proyección hecha anteriormente, obteniendo así el valor final estimado.

3.2. Arquitectura Orientada a Servicios (SOA)

Las aproximaciones monolíticas para la implementación de aplicaciones de RA presentan una serie de inconvenientes. A continuación se detallan dichos problemas y se propone una arquitectura alternativa basada en los sistemas Multi-Agente que se adapta al ámbito de la aplicación del sistema. Los principales inconvenientes de una arquitectura monolítica son los siguientes:

1. El usuario necesita conocer tanto el contexto estático como el contexto dinámico. Por tanto, aunque el contexto estático es el mismo para todos los usuarios, debe ser configurado para todos y cada uno de ellos.
2. Todos los usuarios deben almacenar los modelos 3D y la información multimedia que se ha de mostrar, incluso aunque un usuario no tenga acceso a una cierta información o dicha información sea irrelevante para el usuario.
3. Si el sistema está centralizado, es imposible obtener de forma externa información acerca de los usuarios y su contexto. También es imposible el envío de información específica al propio usuario.

Además, el hecho de desarrollar aplicaciones monolíticas implica la obtención de sistemas altamente especializados para una tarea particular. Aunque las tareas que se realizan en cada uno de estos sistemas suelen ser comunes, reutilizarlos para diferentes implementaciones puede resultar una tarea muy compleja.

Por último, la escalabilidad y el mantenimiento de la propia aplicación se vería afectada de forma negativa. Debido a esta serie de razones, es conveniente utilizar una arquitectura orientada a servicios, en la que la información común a todos los usuarios esté distribuida y como consecuencia, eliminar la redundancia de dicha información.

La arquitectura propuesta consiste en un conjunto de servicios que proporcionan información, para construir el contexto del usuario, y una serie de agentes que manejan toda esa información para obtener los modelos adecuados y presentar la información al usuario (ver figura 3.4).

Dentro del propio sistema hemos distinguido tres capas diferentes:

- **Capa perceptiva**: Se trata de la capa encargada de recolectar la información acerca del entorno y partir de esta información construir el contexto del usuario. Gran parte del contexto del usuario se establecerá de forma implícita a partir de la información recolectada de los sensores. Toda esta información se utilizará para adaptar la aplicación a las necesidades del usuario de forma completamente transparente.

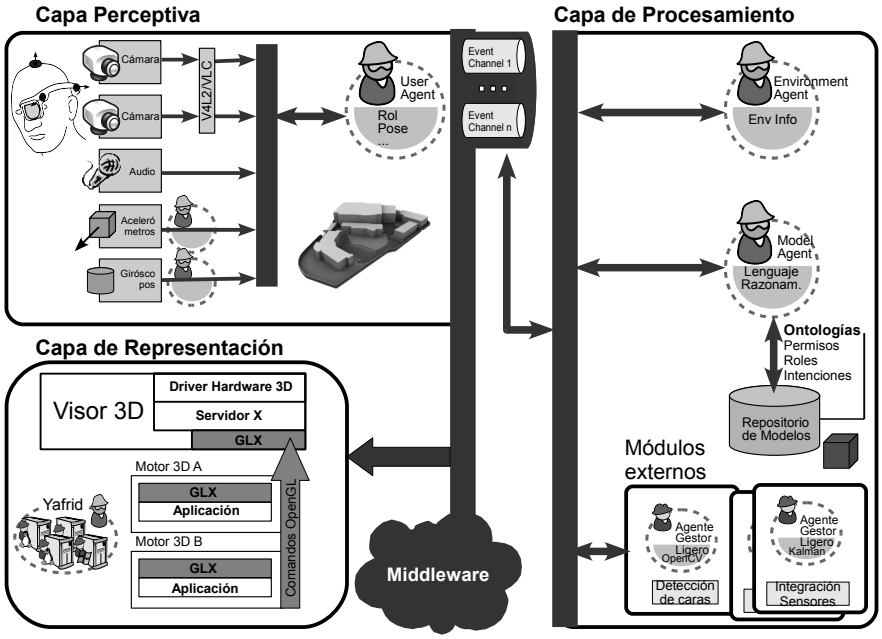

Figura 3.4: Arquitectura general del sistema

- **Capa de procesamiento**: Esta capa es utilizada por la capa perceptiva para establecer el contexto del usuario. En función de dicho contexto, esta capa selecciona los modelos más adecuados para la presentación de información.
- **Capa de representación**: Esta capa está compuesta por los mecanismos de representación disponibles y varios módulos de procesamiento externo distribuidos. Es la encargada de obtener la información procesada y representar los modelos de forma adecuada en los dispositivos que haya disponibles.

3.2.1. Capa perceptiva

La **capa perceptiva** está compuesta por todos los dispositivos que proporcionan información útil para establecer el contexto del usuario. Actualmente, se permiten sensores tales como cámaras, acelerómetros, giroscopios y brújulas electrónicas. Con la información proporcionada por estos dispositivos es posible calcular la posición y la orientación del usuario en el espacio. Para almacenar el estado actual del usuario hemos definido un **contexto de usuario** (ver 3.1.3) que representa el estado interno del usuario y que se utilizará para la representación de información.

Además, también en la capa perceptiva se ha definido un agente llamado **User-Agent**. Este agente representa a cada uno de los usuarios del sistema y es el encargado de recolectar todos los datos proporcionados por la sensorización, que puede ser diferente para cada uno de los usuarios, y componer el contexto en cada instante de tiempo. Este agente es el encargado de comunicarse con la capa de procesamiento proporcionando el contexto en el estado actual y obteniendo la información que debe ser mostrada.

3.2.2. Capa de procesamiento

La **capa de procesamiento** es la encarga de gestionar el contexto del usuario y de seleccionar la información más adecuada en cada momento. Esta capa, básicamente, está compuesta por:

- **Environment-Agent**: Este agente contiene toda la información acerca del entorno en el que se está ejecutando la aplicación. Esta información consiste básicamente en el número de instancias o habitaciones que forman el entorno controlado y la posición global de cada una de las marcas desplegadas. El propósito de este agente es el de colaborar con el User-Agent para proporcionar la localización global del usuario. Para ello, obtendrá un conjunto de percepciones físicas, proporcinadas por la capa perceptiva, y aplicando el conocimiento que tiene del entorno obtendrá la localización global correspondiente. Para su correcto funcionamiento se hace necesita una etapa de configuración en la que se especifique la información que necesita.
- **Model-Agent**: Encargado de seleccionar los modelos más adecuados. Para ello, este agente obtendrá el contexto del User-Agent, lo analizará y recuperará del repositorio la información. Dependiendo del contexto se seleccionarán unos modelos u otros. Además, existe la posibilidad de modificar los modelos en función de dicho contexto.
- **Repositorio 3D**: El repositorio es un componente pasivo del sistema. Su única función es la de almacenar la información relativa a los modelos que se van a desplegar y servirlos bajo demanda. Además proporcionará una interfaz bien definida que permitirá la agregación de nuevos modelos.

Finalmente, también forman parte de la capa de procesamiento una serie de módulos auxiliares que dotan de más funcionalidad a la aplicación. Estos módulos se podrán agregar de una forma fácil proporcionando la especificación de las interfaces que los definen. La comunicación con estos módulos se realizará directamente sin hacer uso de los canales de eventos. Por ejemplo, se puede implementar un módulo que se encargue de detectar las caras contenidas por un frame, en un instante de tiempo concreto. Para ello el user-agent se comunicaría directamente con el módulo encargado de realizar esta función.

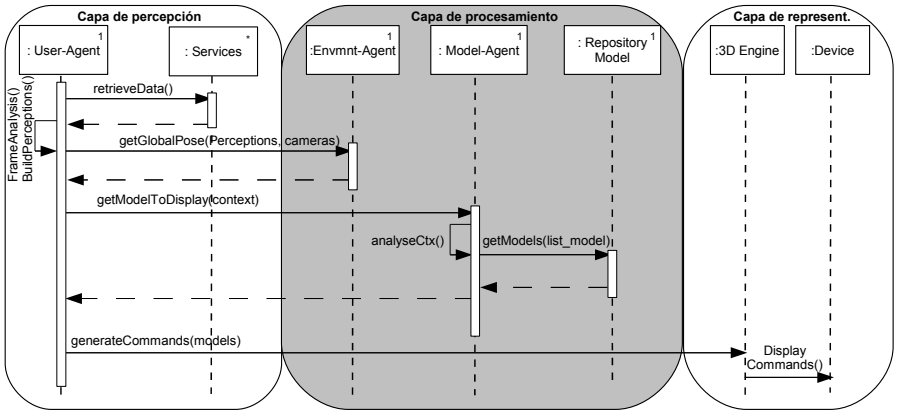

Figura 3.5: Diagrama de interacción típica entre los agentes.

En la figura 3.5 se puede ver el intercambio de información que existe entre los agentes en una iteración del sistema.

3.2.3. Capa de representación

Esta es la capa encargada de representar la información proporcionada por la capa de procesamiento. Aquí se encuentran todos los módulos y dispositivos necesarios para dicha representación. Esta capa genera los comandos necesarios para mostrar la información al usuario. La calidad de estos modelos dependerá del tipo de dispositivo en el que se van a representar.

El objetivo de esta capa es el de proporcionar diferentes módulos de renderizado de la escena y poder seleccionar el más adecuado para su representación en el dispositivo que se esté utilizando. Estos módulos generarán los comandos GLX que se irán proporcionando a la capa superior para su representación 3D.

Por último, indicar que en ciertos contextos puede ser necesaria la generación de imagen 3D realista bajo demanda. Los requisitos de los sistemas de RA hacen necesaria su generación con una tasa de frames por segundos elevada. Para generar imágenes 3D realista en tiempos reducidos existen aproximaciones basadas en computación distribuida que reducen significativamente el tiempo de renderizado. Hemos desarrollado diferentes técnicas para acelerar el proceso de renderizado fotorrealista basadas en sistemas multi-agente [24], en conocimiento experto [59] y en computación Grid [39].

3.2.4. Comunicación entre Agentes

Todo lo relativo a la comunicación estará gestionado por el **middleware**. Esto permite que los agentes solo se centren en recibir la información y procesarla, abstrayéndolos así de las particularidades del proceso de comunicación. El hecho de utilizar un middleware, también facilita la inclusión de nuevos componentes en el sistema. Esto proporciona un alto grado de flexibilidad y escalabilidad.

La comunicación entre los agentes de las diferentes capas se llevará a cabo mediante una serie de canales de eventos. Cada canal proporcionará la información asociada a un determinado tópico. Los agentes interesados en un tópico concreto se suscribirán al canal correspondiente, obteniendo así toda la información publicada en dicho canal. Los agentes que publican en un determinado canal se denomina **agentes productores**, y los que se suscriben a los canales para recibir la información se denominan **agentes consumidores**.

3.3. Implementación del Prototipo Funcional

Para probar la validez de la propuesta se ha implementado un prototipo de la aplicación orientado a las labores que tradicionalmente desempeña un empleado de seguridad. Las principales funcionalidades del sistema son:

- **Desplegar modelos de actuación**: Los modelos de actuación son objetos con los que el usuario puede interactuar para llevar a cabo ciertas acciones. Para desplegar estos

modelos se utiliza la información proporcionada por las marcas de actuación descritas anteriormente en la subsección 3.1.2.

- **Desplegar modelos globales**: Con la información obtenida a partir de las marcas de posicionamiento se pintan modelos de forma global, esto es, con respecto al sistema global de coordenadas. Esto nos permite tener una única representación de los modelos y desplegarlos en función del punto de vista del usuario.
- **Proporcionar la localización del usuario**: Las marcas de posicionamiento también sirven para obtener la posición 2D de un usuario y dibujarla en el mapa de la sala en la que se encuentre. Además, haciendo uso de la brújula electrónica se obtiene la orientación con respecto al norte del mismo usuario.
- **Asistir en la ejecución de tareas**: En ocasiones, es necesario proporcionar información multimedia adicional para la ejecución de tareas complejas, en función del nivel de especialización que tenga el usuario.
- **Agregación de módulos externos**: Los módulos externos utilizados permiten la visualización del vídeo proporcionado por las cámaras de seguridad disponibles, la detección de caras y otras funcionalidades adicionales.
- **Capacidad de interactuar con el usuario**: Además se ha implementado una centralita que permite visualizar lo que el usuario percibe en cada momento, obtener su posición dentro del edificio en el que se está desarrollando la acción y el envío de mensajes o alarmas a través de la propia centralita.

3.4. Conclusiones y Trabajo futuro

Debido al dinamismo y a la gran cantidad de información manejada por las aplicaciones de RA se hace necesaria una especificación arquitectural que de soporte al usuario, proporcionando servicios avanzados en función de toda la información disponible.

En el presente trabajo se presenta una propuesta, basada en los sistemas multi-agente, que en cada iteración recupera el estado del usuario y obtiene la información más relevante en función de dicho estado. El hecho de utilizar esta aproximación proporciona las siguientes ventajas:

- El sistema está compuesto por un conjunto de agentes independientes que colaboran para resolver un problema específico. Este sistema es fácilmente escalable, ya que este tipo de arquitecturas permite la agregación de nuevos elementos con relativa facilidad.
- Las capas definidas son independientes entre si. La comunicación se realiza a través del Middleware, lo que permite que cada agente se encargue de realizar su función dentro del sistema, abstrayéndose completamente de las comunicaciones.
- El mecanismo de comunicación basado en canales de eventos permite que los agentes reciban sólo la información en la que están interesados.
- La definición del **Environment-Agent** y del **Repositorio de modelos** permite el acceso por parte de los usuarios a la información común de la aplicación, evitando que los modelos y la información acerca del entorno tenga que estar replicada para cada uno de los usuarios.

3.4.1. Trabajo futuro

El proceso de *tracking* es una de las partes más importantes de las aplicaciones de RA. La primera línea de trabajo futuro es la de mejorar dicho proceso. Para ello, se especificará una serie de módulos que se encargarán de proporcionar la posición y la orientación del usuario utilizando diferentes técnicas como la extracción de posición a través de las características naturales o el análisis de flujo de los píxeles que componen cada uno de los *frames*. Además, toda esta información se integrará por parte de un **agente de fusión**, situado en un nivel superior, que obtendrá todas las estimaciones, las integrará y proporcionará una estimación óptima de la posición y la orientación del usuario en el espacio. Esta aproximación permitirá que la aplicación sea menos dependiente de las marcas dispuestas, permitiendo así obtener estimaciones de la posición del usuario aunque el *frame* actual no contenga ninguna de las marcas dispuestas en el entorno.

Otra opción para mejorar el proceso de *tracking* es la de filtrar la información proporcionada por el sistema inercial. Según la física clásica, si se integra dos veces la aceleración se obtiene el espacio recorrido en un periodo de tiempo. Esto es imposible con el dispositivo que se utiliza actualmente. Al tratarse de acelerómetros de bajo coste, el error producido es muy significativo cuando se integra doblemente. Se estudiarán los **métodos de filtrado** disponibles para la implementación de un sistema inercial que proporcione 6DOF. Idealmente, este sistema se inicializaría con una posición y una orientación conocidas y a partir de este momento debería ser capaz de funcionar de manera independiente.

Otra mejora del sistema, sería el de proporcionar flujo de video bajo demanda. Con el propósito de visualizar lo que está pasando en diferentes habitaciones o incluso en el exterior del edificio se pretende implementar un consumidor de video que establezca una conexión con una cámara IP y proporcione un **streaming de video** de la cámara solicitada por parte del usuario. Se analizarán distintas QoS proporcionando la que mejor se adapte a las necesidades del usuario.

Además, se desarrollarán herramientas que faciliten el despliegue de la aplicación en entornos complejos, ya que este proceso puede ser bastante tedioso si no se disponen de dichas herramientas. Estas herramientas permitirán la especificación 3D de la escena, la disposición de las marcas en el entorno y la generación de los archivos de configuración necesarios para que la aplicación funcione correctamente.

Bibliografía

[1] F. Ababsa and M. Mallem. Hybrid three-dimensional camera pose estimation using particle filter sensor fusion. *Advanced Robotics*, 21:165–181, 2007.

[2] G.D. Abowd, C.G. Atkeson, A. Feinstein, C. Hmelo, R. Kooper, S. Long, N. Sawhney, and M. Tani. Teaching and learning as multimedia authoring: The classroom 2000 project. In *Proceedings of the ACM Conference on Multimedia*, 1996.

[3] G.D. Abowd, C.G. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. CyberGuide: A mobile context-aware tour guide. *ACM wireless Networks*, 1997.

[4] T. Auer and A. Pinz. The integration of optical and magnetic tracking for multi-user augmented reality. *Computers and Graphics*, 23(6):805–808, 1999.

[5] R. Azuma, J. Weon, B. Jiang, J. Park, J. Park, S. You, and U.Ñeumann. Tracking in unprepared environments for augmented reality systems. *Computers and Graphics*, 23:787–793, 1999.

[6] R. T. Azuma. A survey of augmented reality. *Presence*, 6:355–385, 1997.

[7] M. Bauer, B. Bruegge, G. Klinker, A. Macwilliams, T. Reicher, S. Riß, C. Sandor, and M. Wagner. Design of a component-based augmented reality framework. In *ISAR '01: Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR'01)*, pages 45–54, 2001.

[8] J. Biolchini, P. G. Mian, A. Travassos, and H. Guilherme. Systematic review in software engineering. Technical report, Technical Report RT-ES 679/05, Systems Engineering and Computer Science Department, COPPE/UFRJ, May 2005.

[9] S. Bourgeois, H. Martinsson, Q. Pham, and S.Ñaudet. A practical guide to marker based and hybrid visual registration for ar industrial applications. In *CAIP*, pages 669–676, 2005.

[10] L.R. Breslau, J.B. Hersey, H.E. Edgerton, and F.S. Birch. A precisely timed submersible pinger for tracking instruments in the sea. *Deep-Sea Research and Oceanographic Abstracts*, 9(3-4):137–144, 1962.

[11] L. C. Briand, Y. Labiche, M. Di Penta, and H. Yan-Bondoc. An experimental investigation of formality in uml-based development. *IEEE Transactions on Software Engineering*, 31(10):833–849, 2005.

[12] B. Brown and M. Chalmers. Tourism and mobile technology. In *ECSCW'03: Proceedings of the eighth conference on European Conference on Computer Supported Cooperative Work*, pages 335–354. Kluwer Academic Publishers, 2003.

[13] P. J. Brown. The stick-e document: a framework for creating context-aware applications. *Electronic Publishing*, 8(2-3):259–272, 1995.

[14] P. J. Brown, J. D. Bovey, and X. Chen. Context-aware applications: from the laboratory to the marketplace. *Personal Communications, IEEE [see also IEEE Wireless Communications]*, 4(5):58–64, 1997.

[15] D. Claus and A. W. Fitzgibbon. Reliable fiducial detection in natural scenes. In *In European Conference on Computer Vision*, pages 469–480. Springer-Verlag, 2004.

[16] D. Decarlo and D. Metaxas. Optical flow constraints on deformable models with applications to face tracking. *Int. J. Comput. Vision*, 38(2):99–127, 2000.

[17] M. Delgado, A. F. Gomez-Skarmeta, and L. J. Linares. A regression methodology to induce a fuzzy model. *International Journal of Intelligent Systems*, 16(2):169–190, 2001.

[18] A. K. Dey. Context-aware computing: The cyberdesk project. In *AAAI 1998 Spring Symposium on Intelligent Environments*, pages 51–54. AAAI Press., 1998.

[19] A. K. Dey and G. D. Abowd. Towards a Better Understanding of Context and Context-Awareness. *CHI 2000 Workshop on the What, Who, Where, When, and How of Context-Awareness*, 2000.

[20] A. Doucet, N. Defreitas, and N. Gordon. *An Introduction to Sequential Monte Carlo Methods*. Springer-Verlag,, 2001.

[21] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):932–946, 2002.

[22] S. Feiner, B. MacIntyre, T. Hollerer, and A. Webster. A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environment. *Wearable Computers, IEEE International Symposium*, 0:74–81, 1997.

[23] W. Forstner. A feature based correspondence algorithm for image matching. *From analytical to digital. Proc. symposium, ISPRS, Commission III, Rovaniemi, 1986, vol. 3*, pages 150–166, 1987.

[24] C. Gonzalez Morcillo, G. Weiss, L. Jiménez Linares, D. Vallejo Fernández, and J. Albusac Alonso. A multiagent system for physically based rendering optimization. In *Cooperative Information Agents (CIA)*, pages 149–163, 2007.

[25] D. Hallaway, S. Feiner, and T. Höllerer. Bridging the gaps: Hybrid tracking for adaptive mobile augmented reality. *Applied Artificial Intelligence, Special Edition on Artificial Intelligence in Mobile Systems*, 25:477–500, 2004.

[26] C. Harris. Tracking with rigid objects. In *MIT Press*, 1992.

[27] R. Hervás Lucas. *Modelado de contexto para la visualización de información en ambientes inteligentes*. PhD thesis, Escuela Superior de Informática, February 2009.

[28] S. Hodges and G. Louie. Towards the interactive office. In *In proceedings of Computer Human Interaction Conference*, pages 305–306, 1994.

[29] W. A. Hoff, K.Ñguyen, and T. Lyon. Computer vision-based registration techniques for augmented reality. In *Intelligent Robots and Computer Vision XV*, pages 538–548, 1996.

[30] B. Jiang. Modeling of user profile in context awareness enabled personal network environment. In *6th IEE International Conference on 3G and Beyond*, pages 1–5, 2005.

[31] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[32] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *IWAR '99: Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality*, pages 85–94, Washington, DC, USA, 1999. IEEE Computer Society.

[33] B. Kitchenham. Procedures for performing systematic reviews. Technical report, 2004.

[34] D. Koller, G. Klinker, E. Rose, D. Breen, R. Whitaker, and M. Tuceryan. Real-time vision-based camera tracking for augmented reality applications. In *In ACM Symposium on Virtual Reality Software and Technology*, pages 87–94. ACM Press, 1997.

[35] M. La Cascia, S. Sclaroff, and V. Athitsos. Fast, reliable head tracking under varying illumination: An approach based on robust registration of texture-mapped 3d models. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:322–336, 2000.

[36] P. Lang, A. Kusej, A. Pinz, and G. Brasseur. Inertial tracking for mobile augmented reality. volume 2, pages 1583–1587. IEEE Computer Society, 2002.

[37] B. MacIntyre and S. Feiner. Language-level support for exploratory programming of distributed virtual environments. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 83–94, New York, NY, USA, 1996. ACM.

[38] E. Marchand, P. Bouthemy, and F. Chaumette. A 2d-3d model-based approach to real-time visual tracking. *Image and Vision Computing*, 19(13):941–955, 2001.

[39] J. A. Mateos Ramos, C. González Morcillo, D. Vallejo Fernández, and L. M. López López. Yafrid-ng: A peer to peer architecture for physically based rendering. In *CEIG (09)*, pages 227–230, 2009.

[40] P. Milgram and F. Kishino. A taxonomy of mixed reality visual displays. *IEICE Trans. Information Systems*, E77-D(12):1321–1329, 1994.

[41] P. Milgram, H. Takemura, A. Utsumi, and F. Kishino. Augmented reality: A class of displays on the reality-virtuality continuum. In *Proceedings of the SPIE Conference on Telemanipulator and Telepresence Technologies*, volume 2351, pages 282–292, November 1995.

[42] J. Ohlenburg, I. Herbst, I. Lindt, T. Fröhlich, and W. Broll. The morgan framework: enabling dynamic multi-user ar and vr projects. In *VRST '04: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 166–169, New York, NY, USA, 2004. ACM.

[43] W. Piekarski and B. H. Thomas. An object-oriented software architecture for 3d mixed reality applications. In *ISMAR '03: Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, page 247, Washington, DC, USA, 2003. IEEE Computer Society.

[44] M. Ponder, G. Papagiannakis, T. Molet, N. Magnenat-Thalmann, and D. Thalmann. Vhd++ development framework: Towards extendible, component based vr/ar simulation engine featuring advanced virtual character technologies. *Computer Graphics International Conference*, 1:96–104, 2003.

[45] F.H. Raab, E.B. Blood, T.O. Steiner, and H.R. Jones. Magnetic position and orientation tracking system. *Aerospace and Electronic Systems*, 5:709–718, 1979.

[46] J. Rekimoto. Matrix: A realtime object identification and registration method for augmented reality. *Asia-Pacific Computer and Human Interaction*, 1:63–68, 1998.

[47] J. Rekimoto, Y. Ayatsuka, and K. Hayashi. Augment-able Reality: Situated Communication throug Physical and Digital Spaces. In *Second International Symposium on Weareable Computers*, pages 68–75, 1998.

[48] M. Ribo, P. Lang, H. Ganster, M. Brandner, C. Stock, and A. Pinz. Hybrid tracking for outdoor augmented reality applications. *IEEE Comput. Graph. Appl.*, 22(6):54–63, 2002.

[49] N. S. Ryan, J. Pascoe, and D. R. Morse. Enhanced reality fieldwork: the context-aware archaeological assistant. In *Computer Applications in Archaeology 1997*, British Archaeological Reports. Tempus Reparatum, 1998.

[50] K. Satoh, M. Anabuki, H. Yamamoto, and H. Tamura. A hybrid registration method for outdoor augmented reality. In *ISAR '01: Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR'01)*, page 67. IEEE Computer Society, 2001.

[51] B.Ñ. Schilit and M. M. Theimer. Disseminating active map information to mobile hosts. *Network, IEEE*, 8(5):22–32, 1994.

[52] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavári, L. M. Encarnaçäo, M. Gervautz, and W. Purgathofer. The studierstube augmented reality project. *Presence: Teleoper. Virtual Environ.*, 11(1):33–54, 2002.

[53] A. Schmidt, M. Beigl, and H. Gellersen. There is more to context than location. *Computers and Graphics*, 23:893–901, 1998.

[54] B. Schwald, H. Seibert, and M. Schnaider. Composing 6 dof tracking systems for vr/ar. In *CGI '04: Proceedings of the Computer Graphics International*, pages 411–418. IEEE Computer Society, 2004.

[55] J. Shi and C. Tomasi. Good features to track. Technical report, Ithaca, NY, USA, 1993.

[56] G. Simon and M. O. Berger. A two-stage robust statistical method for temporal registration from features of various type. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, pages 261–266, Washington, DC, USA, 1998. IEEE Computer Society.

[57] A. State, G. Hirota, D. T. Chen, W. F. Garrett, and M. A. Livingston. Superior augmented reality registration by integrating landmark tracking and magnetic tracking. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 429–438, New York, NY, USA, 1996. ACM.

[58] P. S. Strauss and R. Carey. An object-oriented 3d graphics toolkit. *SIGGRAPH Comput. Graph.*, 26(2):341–349, 1992.

[59] D. Vallejo Fernandez, C. Gonzalez Morcillo, and L. Jimenez Linares. A qualitative expert knowledge approach to rendering optimization. In *ICEIS (2)*, pages 336–341, 2007.

[60] M. Weiser. The computer for the twenty-first century. *Scientific American*, 265(3):94–104, 1991.

[61] G. Welch and G. Bishop. An introduction to the kalman filter. White paper, University of North Carolina at Chape Hill., 2006.

[62] S. You, U.Ñeumann, and R. Azuma. Orientation tracking for outdoor augmented reality registration. *IEEE Computer Graphics and Applications*, 19:36–42, 1999.

[63] F. Zhou, H. Been-Lirn, and M. Billinghurst. Trends in augmented reality tracking, interaction and display: A review of ten years of ismar. *Mixed and Augmented Reality, IEEE / ACM International Symposium on*, 1:193–202, 2008.

A

Asignaturas Cursadas

En este capítulo se resumen las áreas de investigación estudiadas y las tareas realizadas en los cursos correspondientes a los módulos I y II del *Máster en Tecnologías Informáticas Avanzadas*.

Metodologías y Técnicas de Investigación en Informática

Profesorado:

- Dr. Mario Piattini
- Dra. Marcela Genero

El objetivo principal de esta asignatura es dotar al alumno de los conocimientos necesarios para llevar a cabo la realización de una Tesis Doctoral. Para ello, el curso abarca cuatro aspectos fundamentales: 1) estudio de la estructura de una tesis doctoral, 2) estudio de las guías más comúnmente aceptadas para escribir una comunicación científica, 3) estudio de una metodología para la revisión sistemática de bibliografía científica y 4) estudio de varios métodos para la validación de hipótesis de investigación.

Como primer trabajo, se realizó el análisis de una comunicación científica[11]. El artículo presenta los resultados obtenidos por dos experimentos, diseñados con el objetivo de analizar el impacto de utilizar OCL junto para el modelado de sistemas software. Finalmente, se llega a la conclusión de que tras una linea de aprendizaje el uso de OCL junto con UML proporciona beneficios a la hora de realizar un modelado preciso utilizando UML.

Por último, se realizó una revisión sistemática de la bibliografía relacionada con Sistemas de Tracking Híbridos. El trabajo ha sido elaborado tomando como base la plantilla de revisión sistemática propuesta por Biolchini et al. [8]. Además se han tenido en consideración

las pautas generales propuestas por Kitchenham [33]. La revisión sistemática sirvió como punto de partida para el estudio de las áreas de investigación relacionadas con el tema de la revisión.

Sistemas Distribuidos Avanzados: Grid e Inteligencia Ambiental

Profesorado:

- Dr. Alfonso Niño
- Dra. Camelia Muñoz
- Dr. José Bravo

El objetivo de este curso fue el estudio en detalle de los fundamentos teóricos y prácticos del *Grid* como aproximación para el diseño e implantación de sistemas distribuidos de propósito general. Se abordaron temas relativos a la evolución del *Grid* como infraestructura software/hardware para proporcionar diversos recursos y tomando diferentes puntos de vista en lo relacionado al control de recursos, consistencia, fiabilidad, robustez y economía.

Como trabajo introductorio, se realizó un análisis de una aplicación particular del *Grid* llamada **The Virtual Lab**. Este primer trabajo sirvió como una introducción al concepto de *Grid* y a los elementos relacionados con su infraestructura. **The Virtual Lab** es un conjunto de herramientas para el modelado molecular distribuido. Mediante esta aplicación se ilustraron numerosos conceptos teóricos relacionados con la arquitectura utilizada, la división de la tarea, la gestión de las bases de datos y el acceso a los recursos distribuidos.

A continuación se realizó un análisis de middlewares alternativos a *Globus Toolkit*, en concreto s estudiaron los middlewares UNICORE y NoruGrid. Con este trabajo se analizó la arquitectura de cada uno de los middleware, analizando las similitudes y las diferencias existentes. La conclusión final obtenida a partir de este trabajo es que aunque existan múltiples middlewares para la gestión grid, todos ellos deben permitir la especificación de un trabajo, la monitorización de los trabajos que están siendo realizados por el *Grid* y proporcionar la seguridad dentro del *Grid*.

Por último, se realizó un análisis sobre los **Entornos de Solución de Problemas** (*Problem Solving Environments*), término acuñado a finales de los años sesenta y que, según la evolución, se comparan a los sistemas *Grid* de propósito específico de hoy en día. En él, se estudiaron las capacidades de este tipo de sistemas para proporcionar todos los recursos computacionales necesarios para resolver los problemas de un dominio de aplicación específico. Se concluyó que las dos características fundamentales de todo Entorno de Solución de Problemas tienen que ver con la posibilidad de dar solución a problemas de un dominio determinado de una forma tal que no podría conseguirse sin la utilización del sistema y con la interacción con el sistema en un lenguaje propio del dominio específico de aplicación.

Sistemas Avanzados de Interacción Persona-Computador: Sistemas Colaborativos y Computación Ubicua

Profesorado:

- Dr. Manuel Ortega
- Dr. Miguel Ángel Redondo
- Dr. Crescencio Bravo

Este curso tuvo como objetivo principal la introducción de los conceptos teóricos y aspectos prácticos fundamentales para el diseño y desarrollo de sistemas colaborativos y/o ubicuos desde una perspectiva metodológica.

Como primer trabajo, se realizó una valoración crítica de [12], donde se presenta un estudio etnográfico de las prácticas realizadas por los turistas en las ciudades que visitan. Se describe el proceso mediante el cual los turistas *colaboran* de forma conjunta utilizando mapas y guías de viaje, planificando sus visitas etc. Todos los conocimientos adquiridos a partir de este estudio inicial se utilizarán para el diseño de tecnologías para el turismo, en especial sistemas que proporcionen el modelo de interacción presentado por los turistas, guías y mapas electrónicos y aplicaciones que los guiarán a través de un tour dentro de la propia ciudad.

Posteriormente, se estudian varios aspectos relacionados con el desarrollo y la implementación de sistemas de colaboración entre usuarios para la ejecución de tareas de forma conjunta. Se ha estudiado el sistema colaborativo **COLLECE**, que es una herramienta que permite realizar programación colaborativa. Haciendo uso de esta herramienta se han estudado aspectos relevantes en este tipo de aplicaciones como el *awareness*. El *awareness* se puede definir como la conciencia que tiene el grupo de las tareas que se están ejecutando en cada instante de tiempo. Finalmente, se introdujo el concepto de evaluación heurística aplicado a sistemas de cooperación.

Finalmente, se estudian dos metodologías para el diseño de interfaces de usuario: ConcurTaskTrees (CTT) y CIAN. CTT es una notación creada por Fabio Paterno para modelar interfaces de usuario aplicadas a sistemas de soporte a la colaboración y cooperación. CIAN, por otro lado, es una metodología para el desarrollo de interfaces de usuario para sistemas Groupware en general. En este bloque se realizó el diseño de una interfaz de usuario mediante la utilización de CTT y CIAN, y un análisis sobre las ventajas y desventajas de utilizar cada aproximación.

Sistemas Heterogéneos en Red

Profesorado:

- Dr. Francisco Moya
- Dr. Fernando Rincón

En esta asignatura se estudiaron diferentes aspectos teóricos y prácticos acerca de los sistemas complejos interconectados, los cuales integran diversos componentes heterogéneos, tanto software como hardware. Se destacan varios aspectos relacionados con diferentes niveles de abstracción en las metodologías de diseño de este tipo de sistemas, considerando especialmente aspectos de software, hardware y comunicaciones.

Como trabajo, se realizó la implementación de un sistema p2p para el renderizado de escenas 3D. El desarrollo se llevó a cabo utilizando el middleware Zeroc Ice, utilizado para la integración de los diversos componentes heterogeneos de la arquitectura.

Cognición y Colaboración

Profesorado:

- Dra. Carmen Lacave
- Dr. Manuel Prieto
- Dr. José Ángel Olivas

Este curso está dividido en tres bloques de conocimiento: i) estudio de los modelos gráficos probabilistas como herramientas de inferencia de conocimiento, ii) mecanismos de representación y gestión del conocimiento y iii) aspectos avanzados de recuperación de la información.

El primer bloque aborda el estudio de los modelos gráficos probabilistas como herramientas para la inferencia de conocimiento. En este bloque, se realiza una introducción general a la teoría probabilística y se definen las Redes Bayesianas como formalismo de asociación entre la probabilidad a priori de hipótesis, la probabilidad condicionada de hipótesis con respecto a evidencias y la probabilidad a posteriori de las hipótesis dadas las evidencias. Como caso de estudio, se analiza la aplicación *Prostanet*, una red bayesiana dedicada al diagnóstico de cáncer de próstata. Fue este bloque el seleccionado para realizar un trabajo práctico, relacionado con la resolución de varios problemas de inferencia de conocimiento mediante la aplicación de modelos gráficos probabilistas.

En el segundo bloque, se ilustran diferentes de mecanismos de representación y gestión del conocimiento mediante su aplicación en sistemas de aprendizaje colaborativo y cooperativo, *e-Learning*.

El tercer bloque está relacionado con la recuperación de la información. En concreto se analizan varios aspectos relativos a la indexación, consulta, evaluación y realimentación en sistemas de recuperación de información.

Técnicas de Softcomputing

Profesorado:

- Dr. Luis Jiménez

- Dr. Juan Moreno
- Dr. José Jesús Castro

En este curso, se plantea el estudio de los sistemas basados en Lógica Difusa como solución al modelado de sistemas artificiales complejos.

El punto de partida en la asignatura consiste en la definición de las limitaciones de los sistemas de razonamiento exacto, las cuáles llevan a la introducción de mecanismos de razonamiento aproximado, más flexibles y robustos para su aplicación en ciertos problemas del mundo real. Así, se analizan diferentes mecanismos de razonamiento aproximado, como los modelos probabilísticos y la teoría de los conjuntos difusos. Este último aspecto fue analizado con gran profundidad e ilustrado mediante los denominados sistemas de razonamiento aproximado de tipo Mamdani. El primer trabajo de la asignatura se realizó bajo este contexto, analizando a nivel general en qué consiste la técnica razonamiento aproximado propuesta por Mamdani. Además se realizó una implementación de los árboles de regresión difusos [17].

Posteriormente, se analizan diferentes aspectos en el desarrollo de sistemas difusos de inferencia y adquisición de conocimiento. En este segundo bloque se estudian diferentes mecanismos para la adquisición de conocimiento, la inferencia de reglas basadas en el conocimiento adquirido y la interpretación de dichas reglas para la inferencia de conocimiento abstracto.

B

Curriculum

CURRICULUM VITAE

Datos Personales

Nombre Completo	José Ángel Mateos Ramos
D.N.I.	71217362-R
Fecha de Nacimiento	3 de Junio de 1981
Lugar de Nacimiento	Manzanares (Ciudad Real)
Nacionalidad	Española
Teléfono	+34652205230

Formación Académica

Actualmente	Cursando último año del Máster de Posgrado Oficial en Tecnologías Informáticas Avanzadas del Dpto. de Tecnologías y Sistemas de Información de la Universidad de Castilla-La Mancha. Realizando la Tesis Doctoral.
12/2007	Ingeniero Informático por la Universidad de Castilla-La Mancha.
9/2004	Ingeniero Técnico Informático por la Universidad de Castilla-La Mancha.

Ocupación Actual

Puesto:	Investigador Contratado
Facultad:	Escuela Superior de Informática

Universidad:	Universidad de Castilla-La Mancha
Dirección:	Paseo de la Universidad 4, 13071 Ciudad Real. ESPAÑA.
Teléfono:	+34926295300 ext. 3746
Fax:	+34926295354
e-Mail:	JoseAngel.Mateos@uclm.es
Inicio:	Noviembre 2007

Experiencia Profesional Pasada

10/2006 - 9/2007	Becario Ayuda a la Investigación. Grupo ISA. Escuela Superior de Informática.
1/2005 - 7/2005	Beca de Colaboración en la UCLM. Centro de Cálculo. Universidad de Castilla-La Mancha.

Formación Complementaria de Interés

Desarrollo e Implantación de Aplicaciones Web con JSP. Curso de enseñanzas propias de 20 horas.

Desarrollo de Componentes Enterprise Java Beans (EJB). Curso de enseñanzas propias de 20 horas.

Programación con ASP.NET. Curso de enseñanzas propias de 20 horas.

Desarrollo de Videojuegos 3D Multiplataforma. Curso de enseñanzas propias de 20 horas.

Aplicaciones de Desarrollo en GNU/Linux. Curso de enseñanzas propias de 20 horas.

Idiomas

Inglés	Nivel Avanzado hablado y escrito.

Participación en Proyectos de I+D Financiados

Título del Proyecto:	CENIT HESPERIA: Homeland Security. Tecnologías para la Seguridad Integral en Espacios Públicos e Infraestructuras.
Financiado por:	Proyecto CENIT (CDTI, MITYC)
Duración:	Enero 2006 - Diciembre 2009
Investigador Principal:	Dr. Luis Jiménez Linares (Grupo ORETO - UCLM)

Publicaciones (CN=Congreso Nacional, CI=Congreso Internacional, J=Revista, B=Capítulo de Libro)

J	D. Vallejo, J. Albusac, *J.A. Mateos*, C. Glez-Morcillo, L. Jiménez. **A Modern Approach to MultiAgent Development**. Journal of Systems and Software (Elsevier). JSS-D-09-00080. Enviado: 28/01/2009 Aceptado: 18/09/2009. (Pendiente de publicación).
CI	*J.A. Mateos Ramos*, D. Vallejo-Fernandez, I. Arriaga Sánchez, C. González Morcillo. **A Multi-Agent Architecture for Augmented Reality Applications**. Proceedings of the 2009 IADIS International Conference Intelligent Systems and Agents. Pages: 159-166. Algarve, Portugal. Jun, 2009.
CI	V. Herrera Tirado, C. González Morcillo, M. A. García Marín, *J. A. Mateos Ramos*, I. Arriaga Sánchez. **GANAS: A Flexible Architecture for 3D Sign Language**. Proceedings of the 2009 IADIS International Interfaces and Human-Computer Interaction. Pages: 61-68. Algarve, Portugal. Jun, 2009.
CN	*J.A. Mateos Ramos*, C. González Morcillo, D. Vallejo Fernández, L.M. López López. **Yafrid-NG: A Peer to Peer Architecture for Physically Based Rendering**. Congreso Español de Informática Gráfica (CEIG'09). Pages: 227-230. San Sebastián, Spain. Sep, 2009.

Estancias en Centros en el Extranjero

09/2005 - 07/2006	Faculty of Computer and Information Science, University of Ljubljana. Ljubljana, Eslovenia.