Realidad virtual y realidad aumentada. Interfaces avanzadas.

Cristina Manresa Yee | María José Abásolo
Ramón Más Sansó | Marcelo Vénere

Realidad virtual y realidad aumentada. Interfaces avanzadas.

cacic2011 XVII CONGRESO ARGENTINO DE CIENCIA DE LA COMPUTACIÓN

XV ESCUELA INTERNACIONAL DE INFORMATICA

FACULTAD DE INFORMÁTICA
Universidad Nacional de La Plata

Realidad virtual y realidad aumentada. Interfaces avanzadas / Cristina Manresa Yee ... [et.al.]. - 1a ed. - La Plata : Universidad Nacional de La Plata, 2011.
150 p. ; 24x15 cm.

ISBN 978-950-34-0765-3

1. Aplicaciones Informáticas. I. Manresa Yee, Cristina
CDD 005.3

Realidad virtual y realidad aumentada. Interfaces avanzadas.

Cristina Manresa Yee | María José Abásolo | Ramón Más Sansó | Marcelo Vénere

Diseño y diagramación: Andrea López Osornio

Editorial de la Universidad Nacional de La Plata (Edulp)
47 N° 380 / La Plata B1900AJP / Buenos Aires, Argentina
+54 221 427 3992 / 427 4898
editorial@editorial.unlp.edu.ar
www.editorial.unlp.edu.ar

Edulp integra la Red de Editoriales Universitarias (REUN)

Primera edición, 2011
ISBN Nº 978-950-34-0765-3

Queda hecho el depósito que marca la Ley 11.723

Impreso en Argentina

Agradecimientos

Este trabajo ha sido parcialmente financiado por el proyecto FRIVIG D/ 031964/10. Formación de Recursos Humanos e Investigación en el Área de Visión por Computador e Informática Gráfica, del Programa de Cooperación Interuniversitaria e Investigación Científica entre España e Iberoamérica del MAEC-AECID.

Colaboran:

Índice

Parte I: Realidad virtual y realidad aumentada

Parte II: Interfaces avanzadas

Prefacio

El presente libro es el resultado de la docencia e investigación en las áreas de Realidad Virtual, Realidad Aumentada e Interfaces basadas en visión, llevadas a cabo en tres instituciones universitarias, dos de ellas de Argentina –Universidad Nacional de La Plata (UNLP) y Universidad Nacional del Centro de la Provincia de Buenos Aires (UNICEN) – y una española –Universidad de las Islas Baleares (UIB)–.

El texto se estructura en dos partes principales: la primera parte relacionada con Realidad Virtual (RV) y Realidad Aumentada (RA) y la segunda parte relacionada con las denominadas Interfaces avanzadas o Basadas en Visión (VBI).

La primera parte consta de tres capítulos. El capítulo 1 presenta una introducción a conceptos y tecnología compartidos por las aplicaciones de realidad virtual y realidad aumentada. El capítulo 2 presenta los desafíos actuales para el desarrollo de simuladores de entrenamiento que utilizan realidad virtual, y describe los simuladores desarrollados por el Instituto Pladema de la UNICEN. El capítulo 3 presenta el tema Realidad Aumentada, sus fundamentos, algoritmos de tracking y librerías utilizadas para el desarrollo de aplicaciones. Lo incluido en este capítulo es utilizado como material de docencia en un curso del Doctorado de Ciencias Informáticas de la UNLP, dictado en la actualidad por una docente de dicha institución e investigadora del III-LIDI.

La segunda parte, Interfaces Avanzadas, consta de dos capítulos. El material incluido es resultado de la docencia e investigación de dos investigadores de la Unidad de Gráficos y Visión por Ordenador e Inteligencia Artificial de la UIB. El capítulo 4 realiza una introducción a las interfaces basadas en visión, así como explica el proyecto SINA desarrollado en la UIB. El capítulo 5 presenta los sistemas de interacción multitáctil, y además explica un caso de estudio del diseño de una mesa multitáctil.

CAPÍTULO 1

Introducción a realidad virtual y aumentada

María José Abásolo, Universidad Nacional de La Plata, Argentina

1. Qué es Realidad Virtual

La Realidad Virtual (RV) es un término que se aplica a un conjunto de experiencias sensoriales sintéticas, es decir generadas por computador, comunicadas a un operador o participante.
La mayoría de las aplicaciones de realidad virtual son experiencias visuales donde el participante se ve inmerso e interactúa en un ambiente o escena virtual. La escena virtual se visualiza mediante algún dispositivo de visualización, en algunos casos utilizando visualización estereoscópica la cual brinda la sensación del ambiente tridimensional.

Una aplicación de RV tiene 4 características:
- Mundo Virtual
- Inmersión (mental y física)
- Retroalimentación –en inglés, feedback– sensorial
- Interactividad

La retroalimentación sensorial puede aludir a los diferentes sentidos, comúnmente la vista (objetos virtuales animados), pero también al oído (sonidos) y el tacto o hápticas –del término en inglés *haptics*–.

Uno de los primeros simuladores conocidos que utilizaron realidad virtual es el denominado "Sensorama" realizado en 1962 por Morton Heilig, un simulador de moto con imágenes, sonidos, vibraciones e incluso olores.
Año 1965 Ivan Sutherland escribe el artículo "The ultimate display" donde introduce por primera vez el concepto de realidad virtual y describe un dispositivo de visualización que es un casco que se coloca en la cabeza –en inglés, "Head-Mounted Display" (HMD)–.

Una de las principales aplicaciones actuales de RV son los simuladores de entrenamiento. Puede consultarse el capítulo 2 del presente libro el cual describe los desafíos actuales para el desarrollo de simuladores con RV, y presenta los simuladores desarrollados por la UNICEN.

Los sistemas de RV más habituales, según los dispositivos de visualización más utilizados en la actualidad son:
- Los cascos o HMD
- Las cuevas –en inglés *caves*– (figura 1.a)
- Las *icon*, que son pantallas gigantes
- Los *WorkBench* (figura 1.b)
- El monitor

Estos dispositivos se detallan en la sección 3.3.

(a) (b)

Figura 1. Dispositivos de salida de RV: cuevas o *caves* (a) y *workbench* (b)

En todo momento se debe mostrar el mundo virtual coherente de acuerdo al punto de vista que tenga el participante. Por esto se necesita saber en todo momento la posición y orientación del participante en el mundo virtual. El proceso de conocer en cada instante la posición y orientación de un cierto objeto se denomina seguimiento, en inglés “tracking” (ver sección 3.2.).

Existen diversos libros de realidad virtual entre los cuales se citan [4] [5] [6] [9] [12].

2. Qué es Realidad Aumentada

En el año 1992 Tom Caudell crea el termino Realidad Aumentada (RA) para describir una pantalla que usarían los técnicos electricistas

de Boeing que mezclaba gráficos virtuales con la realidad física, este sistema les permitiría aumentar la eficiencia de su trabajo al facilitarles de alguna forma la operativa sobre las tareas a realizar
La RA agrega información sintética a la realidad. La diferencia principal entre Realidad Virtual y Realidad Aumentada es que por una parte RV implica inmersión del participante en un mundo totalmente virtual y por otra parte la RA implica mantenerse en el mundo real con agregados virtuales.
Siguiendo la definición del autor Ronald Azuma [1] [2] un sistema de RA tiene 3 requerimientos:

- Combina la realidad con información sintética
- Los objetos virtuales están registrados en el mundo real
- Es interactivo en tiempo real

La información virtual tiene que estar vinculada espacialmente al mundo real de manera coherente, lo que se denomina registro de imágenes o en inglés "registration". Por esto se necesita saber en todo momento la posición del usuario, tracking, con respecto al mundo real.
A diferencia de las aplicaciones de RV las aplicaciones de RA generalmente necesitan la movilidad del usuario, incluso hacia ambientes externos- en inglés se denominan aplicaciones *outdoor*. En dichas aplicaciones de realidad aumentada puede ser necesaria conocer la posición global del participante utilizando dispositivos como GPS y brújulas digitales.
Dependiendo del dispositivo de visualización usado las aplicaciones de realidad aumentada pueden basarse en:

- Gafas de video *see-through*[1]
- Gafas de óptica *see-through*
- Proyector
- Monitor
- Dispositivos móviles o *HandHeld*[2]

Estos dispositivos se detallan en la sección 3.3.

Hasta la fecha no son demasiados los libros dedicados a RA, entre los cuales se pueden citar [3] [7] [8]. Para una introducción detallada a la RA, sus fundamentos, técnicas de tracking y librerías, puede consultarse el capítulo 3 del presente libro.

[1] La traducción de "see-through" significa "ver a través", ya que las gafas dejan ver la realidad a diferencia de los dispositivos de realidad virtual.
[2] La traducción de "hand-held" significa "sostenido con la mano".

3. Partes de un sistema de RV y RA

Las necesidades de un sistema de RV y RA pueden dividirse:
- Dispositivos de entrada
    - Para tracking
    - Mecanismos de entrada de comandos y datos
- Dispositivos de salida
    - Salida visual
    - Salida auditiva
    - Salida táctil o *haptics*
- Aplicación en tiempo real
    - Visualización del mundo virtual (en RV) o mixto (en RA) de acuerdo a la posición y orientación del participante
    - Interactividad

En las siguientes secciones se detallan los dispositivos de entrada de comandos (sección 3.1), los dispositivos de tracking (sección 3.2.) y por último los dispositivos de salida (sección 3.3.)

3.1. Dispositivos de entrada de comandos y datos

Con respecto a los mecanismos de entrada de comandos pueden utilizarse:
- Controles físicos
- Control del habla
- Control por gestos

Los controles físicos más utilizados en aplicaciones de RV son:
- mouse 3D de 6 grados de libertad o DOF[3]
- *Props*
- Plataformas

Los *props* son objetos físicos utilizados para representar algún objeto dentro del mundo virtual. Al ser reales, permiten al usuario una manipulación fácil del objeto. Esto permite que el mundo virtual sea un poco más real.
Por su parte, las aplicaciones de RA que requieren más movilidad utilizan otro tipo de mecanismos de entrada. Por ejemplo, las aplicaciones de RA basadas en dispositivos móviles como PDA o teléfonos celulares suelen utilizan la pantalla táctil o el lápiz para dar entrada a comandos.

[3] DOF es el acrónimo de *Degree Of Freedom*.

El control del habla es un método natural de comunicar la información. No es adecuado si se necesita una respuesta inmediata, pero resulta muy adecuado si se tienen las manos ocupadas.
El control por gestos es un método natural de comunicar la información en el que se basan nuevas metáforas de interacción. Requiere de dispositivos especiales para el tracking o en su lugar de la aplicación de técnicas de visión por computador. Los capítulos 4 y 5 del presente libro se dedican a este tipo de interfaces denominadas basadas en visión.
Las interfaces multimodales, es decir las que combinan diferentes formas de dar entrada a comandos, suelen ser las más utilizadas actualmente. Kölsch et al [10] analizan que tipo de control o combinación de ellos – mediante dispositivos, mediante habla y/ mediante gestos- resulta más adecuado según los parámetros que requiera el tipo de tarea que se quiera realizar. Los autores concluyen que en su aplicación:

- Aquellas tareas que requieren parámetros adimensionales, es decir, no se debe indicar nada en el espacio, pueden indicarse adecuadamente mediante el habla
- Las tareas que requieren parámetros espaciales de una dimensión- como un desplazamiento en un eje (por ejemplo, indicar un zoom) - pueden indicarse adecuadamente mediante un dispositivo como un trackball unidireccional.
- Las tareas que requieren parámetros espaciales de dos dimensiones y tres dimensiones (posicionamiento, escalado y orientación de objetos en el mundo virtual) requieren utilizar diferentes tipos de entrada combinando gestos, movimientos de cabeza y trackball unidireccional.

3.2 Tracking

En aplicaciones de RV y RA se debe realizar el seguimiento o tracking del participante, para determinar su posición y orientación en el mundo virtual (en aplicaciones de realidad virtual) o en el mundo real (en el caso de aplicaciones de realidad aumentada).
Para realizar el seguimiento del usuario pueden usarse dispositivos específicos o puede analizarse una imagen capturada de la realidad para deducir la posición del usuario en base a elementos del entorno, lo que se denomina tracking basado en visión.
Generalmente en aplicaciones de realidad virtual y algunas aplicaciones de realidad aumentada se realiza el tracking de la cabeza para realizar la visualización del mundo virtual o mixto de acuerdo al punto de vista del participante.

En algunas aplicaciones puede necesitarse el tracking de objetos que sostiene el participante. Por ejemplo, en una aplicación de medicina de entrenamiento para cirugía en la que el participante sostiene un bisturí.
Dependiendo del tipo de aplicación será la precisión que se necesite para determinar la posición del usuario y en consecuencia el tipo de tracking que se realice. Por ejemplo, en aplicaciones de medicina el tracking puede resultar crítico y por tanto debe ser muy preciso.
El tracking de la mano y de los dedos se utiliza para proporcionar al usuario un mecanismo de interacción con el mundo. Puede montarse sobre la mano (guante) o sobre algún tipo de puntero. Por ejemplo el DataGlove tiene además de un sensor de posición diferentes sensores para detectar la flexión de los dedos (5° de resolución).

Considerando que existirá por una parte un cierto sensor capaz de recolectar datos y por otra parte una fuente emisora de los datos pueden distinguirse diferentes tipos de sistemas según donde se localicen las fuentes y sensores:

Tipo de sistema	**Sensores**	**Fuentes**
Inside-out	En el objeto en movimiento.	En el ambiente.
Inside-in	En el objeto en movimiento.	En el objeto en movimiento.
Outside-in	En el ambiente.	En el objeto en movimiento.

El Tracking puede clasificarse según la tecnología con la que se realice en electromagnético, mecánico, inercial, ultrasónico, óptico y videométrico. Puede consultarse la publicación de Rolland (2001) [11] para mayor detalle de cada tecnología de tracking. En la tabla siguiente se detallan las ventajas y desventajas de cada tipo de tecnología.

Tracking	**Tipo de sistema**	**Ventaja**	**Desventajas**
Electromagnético	Es un sistema inside-out ya que los sensores se localizan en el cuerpo del participante que se mueve en un campo electromagnético exterior que actúa como fuente.	No se necesita una línea de visión directa del participante. Si se colocan múltiples sensores se puede realizar el seguimiento de diferentes articulaciones.	Normalmente funciona con cables. Los metales pueden crear interferencias. El espacio de trabajo es reducido.

Tracking	Tipo de sistema	Ventaja	Desventajas
Mecánico	Es un sistema inside-in donde los sensores son potenciómetros que se colocan en las articulaciones que son las fuentes que ejercen fuerza sobre los potenciómetros.	Muy rápido y muy preciso.	Muy incomodo para el usuario. Movimientos muy limitados. Se puede integrar respuestas de fuerzas (*force-feedback*).
Óptico	Es un sistema outside-in ya que los sensores externos, cámaras, se posicionan en el ambiente exterior para recolectar datos de fuentes localizadas en el cuerpo (puntos reflectantes).	Utiliza cámaras de video u otros sensores de luz.. No necesita cables. Con varias cámaras se puede obtener la posición 3D.	Necesita visión directa y buena iluminación.
Videométrico	Es un sistema inside-out ya que sitúa una cámara sobre el participante y se utilizan puntos de referencia en el mundo real para determinar la posición del mismo.	Es económico.	Necesita elementos de cálculo adicionales o cableado.
Ultrasónico	A través de múltiples fuentes de sonidos de alta frecuencia y micrófonos se puede medir el tiempo que tarda el sonido para ir de un lugar al otro, y determinar la posición de cada micrófono por triangulación.		No es bueno en ambientes ruidosos. Necesita visión directa.
Inercial	Utiliza instrumentos que pueden detectar y medir cambios de fuerzas giroscópicas (aceleración e inclinación).	Son baratos. No precisan punto de referencia.	Pierden precisión a lo largo del tiempo. Funcionan bien combinados con otros sistemas.

3.3. Dispositivos de salida

Los sistemas de realidad virtual y aumentada han de proporcionar información del mundo virtual con respuestas:
- Visuales
- Auditivas
- Táctiles

Para cada una de las diferentes salidas puede analizarse:
- Los medios de interacción
- Sus propiedades
- Sus cualidades logísticas

3.3.1. Salidas visuales

Según los medios de interacción de los dispositivos de visualización se clasifican en estacionarios o estáticos, móviles oclusivos, móviles no oclusivos.

Las propiedades de los dispositivos de visualización son: resolución, color, contraste, brillo, número de canales, distancia focal, campo de visión y campo de mirada, opacidad, latencia y velocidad de refresco. El campo de visión mide la amplitud y la altura angular de la visión del usuario, y se mide en porcentaje o ángulos. La visión humana es de 200 grados en total, con 130 grados de solapamiento; la visión túnel es de 60 grados o menos; y los cascos de RV suelen cubrir 100 grados, con 60 de solapamiento. El campo de mirada mide la cantidad de espacio en que se visualiza el mundo virtual y se mide en porcentajes. En los dispositivos móviles es del 100% y en los dispositivos estáticos es mucho menor a excepción de las caves de 6 lados. La opacidad es el nivel de aislamiento visual del mundo real. Las caves no son opacas; los cascos a menudo son opacos; y la realidad aumentada requiere que los dispositivos permitan ver la realidad. La latencia es el lapso de tiempo entre el movimiento del participante y la sensación que recibe. Se hace muy notoria al mover la cabeza con el casco de RV, y casi no se nota en los dispositivos estáticos. La velocidad de refresco es el número de veces que se actualiza la imagen por segundo medida en *Frames Per Second* (FPS) o Hz. Una velocidad de 30 Hz se considera óptima; 15 Hz se considera la mínima aceptable; y 10 Hz o menos hace que el cerebro detecte una secuencia de imágenes fijas.

Las cualidades logísticas de los dispositivos de visualización son: la movilidad del usuario, la interfaz con el tracking, los requisitos del entorno, la portabilidad, la seguridad y el coste.
La movilidad del usuario tiene un gran efecto en la inmersión mental y en la utilidad del sistema. Los requisitos del entorno son las condiciones necesarias para una buena experiencia de RV. Las imágenes proyectadas requieren que haya poca luz; las grandes pantallas estáticas necesitan mucho espacio; y los dispositivos móviles trabajan mejor en lugares pequeños.

Dispositivo de visualización	**Aplicación**	**Descripción**
Cave	RV	Estáticos. Coste elevado. Campos de visión y de mirada amplios.. Inmersivos. Mayor resolución. Mayor campo de visión. Mayor tiempo de inmersión. Más seguridad. Mejor para grupos.
Monitor	RV y RA	Estáticos. Pocos componentes/bajo coste. Fácil instalación. Dispositivos de interacción disponibles. Campos de visión y de mirada limitados. Poco inmersivos.
HMD Video see-through	RA y RA	Oclusivos. Pantallas pequeñas y transportables. La inmersión fundamental se basa en la orientación de la cabeza. El campo de mirada es del 100%. Tiene un importante rango de resoluciones. Normalmente, le falta campo de visión. Fatiga visual.
HMD Optical see-trough	RA	No oclusivos. Simplicidad. Alta seguridad. El *tracking* debe estar bien sincronizado. Difícil tratamiento de las oclusiones.
Proyector	RV y RA	Estáticos. Pocos componentes/alto coste. Difícil instalación. Dispositivos de interacción disponibles. Campo de mirada limitado. Poco inmersivos.
HandHeld	RA	Menor coste. Campo de mirada del 100%. Más portabilidad. Menos espacio. Pueden ocultar el mundo real. No importa la iluminación.

Azuma [1] resume las diferentes ventajas y desventajas de los dispositivos *optical see-through* y los *video-see-through.* Por un lado los dispositivos de video *see-through* presentan la gran ventaja de la

flexibilidad en la composición o mezcla (*blending*) del video real con el sintético. Un problema básico de los sistemas ópticos es que los objetos virtuales no tapan completamente los objetos reales, por la forma en que funcionan los combinadores ópticos que permiten luz del mundo real y del virtual. Hacer un sistema óptico que selectivamente apague la luz del mundo real aumenta la complejidad del diseño. Los objetos virtuales aparecen semitransparentes y esto puede dañar la ilusión de realidad si se requiere. En contrastes, el video *see-through* es más flexible en cómo mezcla el mundo real y el virtual, ya que ambos están disponibles en forma digital. Se puede componer pixel a pixel tomando o bien el mundo real o el virtual, o mezclándolos si se quiere sensación de transparencia.

Los sistemas ópticos ofrecen una vista instantánea del mundo real pero una vista retrasada del mundo virtual, y esta incoherencia temporal puede causar problemas. Otra de las ventajas de los sistemas de video es que es posible retrasar el video del mundo real para coincidir con el retraso del *stream* del mundo virtual.

En los sistemas ópticos, la única información acerca de la ubicación de la cabeza del usuario viene del dispositivo de tracking de cabeza. Por otra parte, en los sistemas de video *blending*, se tiene otra fuente de información: la imagen digitalizada del mundo real, con lo cual pueden usarse técnicas de visión para el registro de imágenes que no estaría disponible en los sistemas ópticos

Entra las ventajas de los sistemas ópticos frente a los de video pueden enumerarse: la simplicidad, la seguridad, la mayor resolución. Los sistemas ópticos tienen solo un *stream* de video para tratar: los gráficos sintéticos. El mundo real se ve directamente a través de los combinadores ópticos. Esto los hace más simples, ya que los sistemas de video *blending* deben tratar con dos *streams* de video separados para el mundo real y el virtual. Ambos *streams* tienen retrasos del orden de las décimas de milisegundos, y deben ser adecuadamente sincronizados para evitar la distorsión temporal.

Los sistemas de video limitan la resolución de lo que el usuario ve, tanto real como virtual, a la resolución del dispositivo Por otra parte, los sistemas ópticos también muestran los gráficos con la resolución del dispositivo, pero el usuario ve la realidad sin degradarse.

Con respecto a la seguridad los sistemas de video usados en RA son HMD modificados (dado los usados en RV que solo muestran mundo virtual). Si se corta la corriente el usuario estará ciego y esta es una cuestión concerniente a la seguridad en algunas aplicaciones. En contraste, utilizando un sistema óptico el usuario tiene la vista del mundo real y por esto se considera más seguro.

Ambas tecnologías, óptica y video, tienen sus roles y la elección depende de los requerimientos de la aplicación. En aplicaciones de reparación se usan generalmente los sistemas ópticos, posiblemente por cuestiones de seguridad. Por otra parte, en aplicaciones médicas suelen usarse los sistemas de video, probablemente por la flexibilidad en la mezcla del mundo real y virtual y por las estrategias adicionales de registro de imágenes.

3.3.2. Salidas auditivas

Según los medios de interacción de los dispositivos de salida auditiva se clasifican en: estacionarios o estáticos (altavoces) y móviles (auriculares).

Las propiedades de los dispositivos de salida auditiva son: el número de canales, el entorno de sonido, la localización, la oclusión y la amplificación.
Con respecto al número de canales si se tienen diferentes sonidos en cada oído se tiene información sobre la procedencia del sonido. Se denomina sonido monofónico si se tiene la misma señal en cada oído, mientas que sonido estereofónico implica diferentes señales para cada oído.
Con respecto al entorno del sonido hace alusión a la fuente de donde parece originarse el sonido. La localización es la habilidad de nuestro cerebro para determinar la procedencia del sonido. La espacialización es la habilidad tecnológica para hacer que el sonido parezca venir de un punto determinado del espacio.
Con respecto a la oclusión puede decirse que los altavoces no pueden ocultar de una manera efectiva los sonidos del mundo real, mientras que los auriculares cerrados son el mejor mecanismo para experiencias en las que el participante únicamente debe oír sonidos del mundo virtual.

Las cualidades logísticas de los dispositivos de salida auditiva son: la contaminación por ruido, la movilidad del usuario, los requisitos del entorno, la portabilidad, la seguridad y el coste.
Con respecto a los requisitos del entorno las salas cuadradas pueden ser un problema para los altavoces. Los campos magnéticos de los altavoces y los auriculares pueden causar interferencias con los sensores. Los sonidos altos pueden interferir con los *trackers* de ultrasonidos.

Entre las ventajas de los altavoces se pueden señalar las siguientes:

- Trabajan bien con los visualizadores estacionarios
- No requieren el procesado del sonido para crear sonidos referenciados al mundo (estables en el mundo virtual)
- Permiten más movilidad al usuario (cables)

Entre las ventajas de los auriculares se pueden señalar las siguientes:

- Trabajan bien con los visualizadores dinámicos
- La implementación de sonidos referenciados a la cabeza es más fácil de implementar
- Es más sencillo eliminar los ruidos de la habitación
- Son más portables
- Privados

3.3.3. Salidas táctiles

Según los medios de interacción de los dispositivos de salida táctil se clasifican en: percepción kinestética y percepción táctil propiamente dicha. La percepción kinestética está relacionada con la percepción de fuerzas.

Las propiedades de los dispositivos de salida táctil son: pistas kinestéticas, pistas táctiles, fidelidad, latencia, contacto, resolución, grados de libertad, refresco, forma y tamaño.
Las pistas kinestéticas se relacionan con los ángulos de las articulaciones, los músculos, la tensión y resistencia. Las mismas ayudan a determinar la solidez, forma aproximada y fuerzas físicas.
Las pistas táctiles son receptores sensoriales en la piel: Existen mecanoreceptores (forma y textura), termoreceptores (transferencia calor), electroreceptores (flujo de corriente) y nociceptores (daño en los tejidos).
Con respecto al contacto los sensores de fuerza y resistencia necesitan un punto de contacto. Pueden limitar tanto el movimiento del propio cuerpo del participante, como el movimiento del participante respecto al mundo.
Los grados de libertad están asociados al tipo de movimiento que permiten: 6 DOF si el movimiento no está limitado; 1 DOF si permite abrir o cerrar un dedo; 2 DOF igual que el anterior más giro permitido; 3 DOF si nos da la posición (x,y,z) de la punta del dedo.

Las cualidades logísticas de los dispositivos de salida táctil son: la movilidad del usuario, los requisitos del entorno, la portabilidad, la seguridad y el coste.
Con respecto a los requisitos del entorno a veces son necesarisas habitaciones especiales para los dispositivos hidráulicos o de aire

comprimido. Los dispositivos más pequeños pueden funcionar sobre escritorios o incluso en la mano del mismo participante.
Con respecto a la seguridad, se debe ir con mucho cuidado especialmente con actuadores de gran fuerza. Los dispositivos de temperatura son peligrosos y suelen incorporar interruptores de seguridad.
Con respecto al coste normalmente son dispositivos muy costosos. Existen excepciones que son los dispositivos utilizados para juegos como volantes con fuerza, joysticks y controladores de juegos, los cuales pueden adquirirse a precios muy razonables.

4. Referencias

[1] Azuma, R. (1997). "A Survey of Augmented Reality". *Presence: Teleoperators and Virtual Environments*, 6(4), pp 355-385.
[2] Azuma, R.; Baillot, Y.; Behringer, R.; Feiner, S.; Julier, S. and MacIntyre, B. (2001). "Recent Advances in Augmented Reality". *IEEE Computer Graphics and Applications*, 21(6), 34-47.
[3] Bimber, O. and Rascar, R. (2005). *Spatial Augmented Reality Merging Real and Virtual Worlds*, A. K. Peters Ltd, ISBN 1-56881-230-2 [en línea] Consultado el 27 de septiembre de 2011 en <http://140.78.90.140/medien/ar/SpatialAR.download.php>.
[4] Burdea, G. and Philippe Coiffet, P. (2003). *Virtual Reality Technology, Second Edition*, John Wiley & Son ISBN 0-471-36089-9.
[5] Craig, A.; Sherman, W. and Will, J. (2009). *Developing Virtual Reality Applications: Foundations of Effective Design*, Morgan Kaufmann, ISBN 978-0-12-374943-7.
[6] Gutierrez, M.; Vexo, F. and Thalmann, D. (2008). *Stepping into Virtual Reality*, Springer, ISBN 978-1-84800-116-9.
[7] Hainich, R. (2009). *The End of Hardware, 3rd Edition: Augmented Reality and Beyond*, BookSourge Publishing, ISBN 978-1-4392-3602-4.
[8] Haller, M; Billinghurst, M. and Thomas, B. *Emerging Technologies of Augmented Reality. Interfaces and Design*, 1-22, Idea Group Publising, ISBN 1-59904-067-0.
[9] Kim, G. (2005). *Designing Virtual Reality Systems: The Structured Approach*, Springer-Verlag, ISBN 978-1-85233-958-6.
[10] Kölsch, M.; Bane, R.; Höllerer, T. and Turk, M. (2006). "Multimodal Interaction with a Wearable Augmented Reality System", *IEEE Computer Graphics and Applications*, 26(3), 62-71.
[11] Rolland, J.; Davis, L. and Baillot, Y. (2001). "A survey of tracking technology for virtual environments". En Barfield, W. and

Caudell, T. *Fundamentals of Wearable Computers and Augmented Reality*, 1st ed, pp 67-112, Eds. Mahwah.
[12] Sherman, W. and Craig, A. (2003). *Understanding Virtual Reality: Interface, Application, and Design*, The Morgan Kaufmann Series in Computer Graphics, ISBN 1-55860-353-0.

CAPÍTULO 2

Realidad Virtual aplicada al entrenamiento

Marcelo J. Vénere, Cristian García Bauzá, Juan P. D'Amato y Marcos Lazo, Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina
María José Abásolo, Universidad Nacional de La Plata, Argentina

1. Qué es un simulador para entrenamiento

La utilización de simuladores para entrenar operarios es una práctica anterior al concepto de realidad virtual. De hecho podría decirse que la disciplina que primero impulsó el desarrollo de los modelos de visualización en tres dimensiones, que luego condujeron a la idea de realidad virtual, fueron los simuladores de vuelo que requerían las Fuerzas Armadas Americanas.
En general estos simuladores son instalaciones bastante costosas, que utilizan hardware específico y modelos de comportamiento muy precisos. Este concepto sin embargo ha estado cambiando y en la actualidad se ve la utilización de los mismos en todo tipo de áreas y con herramientas de mucho menor costo.
Hoy podríamos hacer una clasificación de simuladores de entrenamiento según la interfaz que utilicen para interactuar con el usuario:

- Sistemas "stand alone": la interfase primaria son los dispositivos clásicos de un sistema de computación convencional, como por ejemplo el monitor, mouse y teclado. Obviamente son de bajo costo y no pretenden que el usuario "sienta" que está operando en situación casi real, sino que aprenda las reacciones y el comportamiento del sistema que opera.
- Sistemas inmersivos: se utilizan dispositivos especiales de realidad virtual tanto de salida (cascos, habitaciones inmersivas, etc.), como de entrada (guantes, trackers, etc.). Aquí si se pretende que el usuario se sienta en la situación real, pero al no utilizar hardware específico se puede hacer funcionar distintos tipos de simuladores en un mismo sistema.
- Sistemas mixtos: son los simuladores convencionales; se utiliza el mismo hardware de la realidad siempre que sea posible y en caso

de no serlo se utiliza simulación por software. De esta forma se logrará el máximo grado de realismo posible, pero las instalaciones son costosas y específicas.

La metodología actual para capacitar operarios es utilizar masivamente los primeros sistemas a modo de formación y posteriormente cumplir una cierta cantidad de horas en simuladores del tercer tipo. En países como Argentina, este último tipo de simuladores no existen y se debe mandar a capacitar al exterior (es el caso de simuladores de vuelo de algunos aviones) o hay una sola instalación para todo el país con fuerte requerimiento (navegación en grandes embarcaciones, submarinos, aviones de guerra y otros). El segundo tipo de simuladores por ahora está poco difundido y en la región solo lo usa Petrobras en Brasil, pero no cabe duda que en la medida que estos sistemas se mejoren, pasarán a marcar la tendencia.

2. Desafíos actuales en el desarrollo de simuladores de entrenamiento

Sin duda los simuladores de entrenamiento son la primera aplicación del concepto de realidad virtual, pero una vez se logre mejorar el grado de realismo que siente el usuario y disminuir los costos de un sistema de este tipo, no cabe duda que las aplicaciones se diversificarán y en especial llegarán al rentable negocio de los juegos. Como expresó uno de los creadores de la primer "cave", "hoy es una instalación de millones de dólares, mañana lo compraremos en el supermercado"

Actualmente los desafíos a resolver están en tres líneas de investigación:

- Lograr ambientes inmersivos a bajo costo
- Realizar renderizado complejo en tiempo real
- Desarrollar modelos de comportamiento precisos

En las siguientes secciones se detallan cada uno de estos problemas.

2.1. Ambientes inmersivos

Como se dijo al comenzar el capítulo, en los sistemas de realidad virtual puede hablarse de diferente grado de inmersión: desde los sistemas no inmersivos "stand-alone" basados en interfaces convencionales, luego los sistemas de realidad virtual totalmente inmersivos, y por último aquellos sistemas híbridos que brindan una

inmersión media ya que constan de interfaces reales (como por ejemplo el simulador de periscopio que se describe más adelante). El punto a mejorar sin duda es el grado de inmersión de los sistemas intermedios.
En esta línea, los lentes de RV actuales por ahora no son una solución a este problema, ya que si bien se pueden generar imágenes estereoscópicas razonables, en ningún caso el usuario siente que está inmerso en ese ambiente.
Una solución mucho mejor son las habitaciones inmersivas o "caves", donde se utilizan múltiples pantallas (piso, techo, frente y costados) sobre las que se proyectan doble imagen para lograr estereoscopía, sincronizado con lentes que debe utilizar el usuario. En este caso si existe la sensación de inmersión, pero una instalación completa cuenta cerca de un millón de dólares y requiere de mucho espacio. Además su empleo no es trivial, ya que deben sincronizarse múltiples imágenes y procesos de simulación.
Las líneas de trabajo en este tema están en mejorar los lentes RV para que las imágenes no se vean solo hacia adelante, y por otro lado abaratar las instalaciones tipo "cave" y achicar los requerimientos de espacio.

2.2. Renderizado complejo en tiempo real

El proceso de renderizado, en inglés *rendering*, es el proceso por el cual el modelo 3D de la escena (representado usualmente por superficies trianguladas) se transforma en imagen considerando la posición de la cámara y las condiciones de iluminación. En una aplicación de realidad virtual, este proceso debe realizarse a un ritmo no inferior a 30 cuadros por segundo y si se trabaja con estereoscopia, el proceso debe realizarse para dos cámaras simultáneamente.
Para modelar correctamente la iluminación de los objetos debe considerarse los tres fenómenos que ocurren cuando la luz incide en el mismo: absorción y emisión difusa; reflexión especular y transmisión (en realidad es refracción, ya que la luz al cambiar de medio cambia su dirección).
El modelo clásico de renderizado soportado en las placas gráficas está basado en el tratamiento por triángulos. Es decir, se toma cada triángulo de la escena y se computa un modelo de iluminación simple (ecuación i) que considera poco más que la componente difusa. Es decir, actualmente es posible renderizar escenas muy complejas (millones de triángulos) en tiempo real solo si se considera la componente difusa de la iluminación. Incluso se puede utilizar texturas para los triángulos sin demasiada pérdida de performance.

Si bien con esta metodología se pueden obtener imágenes muy buenas (ver por ejemplo en la figura 1 la representación del puerto de Bariloche utilizada en un simulador desarrollado en la UNICEN), el ojo de un experto notará múltiples defectos, que a la larga impactan en la sensación de realismo que tiene el usuario.

$$R = I_a.Kr + \Sigma_i I_i.((1\text{-}Ks).Kr.\ \mathbf{L_i.n} + Ks.\ (\mathbf{V.r_i})^q)$$

$$G = I_a.Kg + \Sigma_i I_i.((1\text{-}Ks).Kg.\ \mathbf{L_i.n} + Ks.\ (\mathbf{V.r_i})^q)$$

$$B = I_a.Kb + \Sigma_i I_i.((1\text{-}Ks).Kb.\ \mathbf{L_i.n} + Ks.\ (\mathbf{V.r_i})^q)$$

Donde **R, G y B** son las componente Red, Green y Blue que se debe asignar al triángulo, **Ia** es una intensidad de luz ambiente, **Ii** y **Li** las intensidades y posiciones de las fuentes de luz, **n** y **r** la normal y el rayo reflejado al triángulo, **V** la posición de la cámara y **Kr, Kg, Kb, Ks** y **q** propiedades materiales del triángulo.

El problema es que el tratamiento de las otras dos componentes (reflexión y transmisión) no está bien modelado en la arquitectura actual de las placas gráficas, y las soluciones ofrecidas por ahora no son más que "parches", tales como realizar un z-buffer desde cada fuente de luz para definir sombras, ordenar los triángulos para simular transparencias, pintar como textura la imagen reflejada en una superficie, etc.

Para que las escenas se visualicen con un mejor grado de realismo, estas otras dos componentes deben ser correctamente modeladas, y en esta dirección deben ir los próximos desarrollos. Básicamente, las placas deberán trabajar ya no pintando polígonos, sino pintando píxeles siguiendo los rayos correspondientes. Este modelo se lo conoce como *ray-traycing*, y consigue tratar perfectamente las tres componentes y representar bien las sombras, reflexiones e incluso la refracción. Su costo computacional es sin embargo mucho más elevado y por ahora solo se consiguen unos pocos cuadros por segundo a resoluciones superiores a 1024x780.

Esto se logrará sin duda en el futuro inmediato con un impacto importante en la calidad de las aplicaciones de RV, sin embargo quedarán aún temas por resolver, como son el tratamiento de luces no puntuales y más complejo aun, el tratamiento de la iluminación indirecta (la que proviene de otros objetos) con modelos de tipo *Radiosity.*

(a)

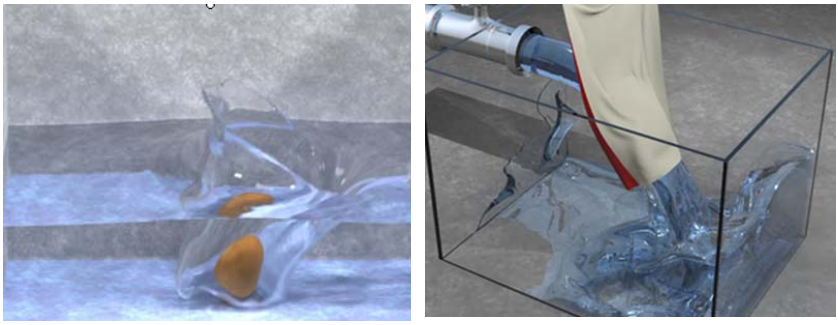
(b)

Figura 1. Fotografía real del Puerto de Bariloche (a) e imagen sintética obtenida a partir de un modelo virtual de dicho puerto (b)

2.3. Modelos de comportamiento precisos

Otro factor que actualmente no está correctamente resuelto en las aplicaciones de realidad virtual es el modelado del comportamiento de los distintos actores de la escena.

Es aun corriente ver por ejemplo que el encuentro entre una embarcación y el agua en la que navega es muy insatisfactorio y no engaña a nadie. También el modelado de un fuego en una hoguera, explosiones, la salpicadura al ingresar un objeto en el agua, son todos fenómenos mal simulados en RV. En la figura 2 se muestran algunos ejemplos involucrando fluidos, los cuales fueron modelados mediante la técnica de Lattice Boltzmann.



(a) (b)

Figura 2. Modelado de fluidos mediante la técnica de Lattice Boltzmann.

Ya hace unos años que se está poniendo mucho esfuerzo en mejorar estos factores, básicamente mediante el desarrollo de "motores físicos" que pueden invocarse desde cualquier aplicación (juegos, simuladores o RV). Los primeros ejemplos son los motores que permiten modelar la interacción de varios objetos en un ambiente con gravedad. Básicamente se simulan colisiones elásticas que resultan en un sistema de ecuaciones diferenciales ordinarias (ODE) que puede resolverse en tiempo real.

Motores como ODE o Newton consiguen modelar cientos de objetos interactuando en una escena (en pruebas que realizamos se mantuvo un buen ritmo de actualización hasta para 4000 objetos). Si bien los resultados cuantitativos (precisión en la posición de cada objeto) pueden dejar algo que desear, el comportamiento global es muy convincente, por lo que es corriente ver su empleo en todo tipo de aplicaciones. La figura 3 muestra un ejemplo de un vehículo colisionando un conjunto de cajas utilizando a Newton como motor de simulación.

Figura 3. Simulación con el modelo de Newton.

El modelado de agua ya es un problema más complejo y costoso en términos computacionales, ya que para su simulación se deben utilizar métodos como Elementos Finitos, Diferencias Finitas o más recientemente Lattice Boltzmann.
En la UNICEN hemos desarrollado un motor físico que modela adecuadamente superficies de agua y su interacción con objetos utilizando el método de Lattice Bolzmann. La superficie es representada mediante una grilla y los puntos de la misma se actualizan a cada paso de tiempo a un ritmo aceptable para una aplicación interactiva. En la figura 4 se muestra el tipo de imágenes que se puede generar de esta forma.

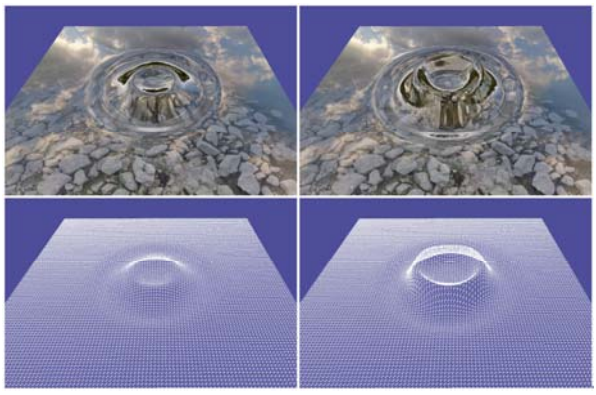


Figura 4. Modelado de agua mediante Lattice Bolzmann

Este caso se logra simular en tiempo real gracias a una programación muy eficiente y a la implementación en dispositivos de alta performance, como por ejemplo GPUs. Además debe tenerse en cuenta que este problema es en realidad bi-dimensional. Modelar problemas tridimensionales como el de la figura anterior por ahora está fuera del alcance de aplicaciones en tiempo real.

3. Simuladores desarrollados en la UNICEN

Desde el año 2001 hasta la actualidad en el Instituto Pladema de la Universidad Nacional del Centro de la Provincia de Buenos Aires se ha realizado diversos desarrollos de simuladores de entrenamiento utilizando distintas tecnologías de realidad virtual.
En orden cronológico, los simuladores desarrollados son:

- Simulador de radar arpa MELIPAL-R (2001)
- Simulador de submarino SIPER (2002)
- Simulador de embarcación pesquera MELIPAL-P (2003)
- Simulador de radar terrestre RASIT (2005)
- Simulador de operación de oleoductos (SIMOPT) (2006)
- Simulador de radar aéreo ATLAS 2 (2007)
- Simulador de maniobras MELIPAL-M (2008)

En las siguientes secciones se describen algunos de estos simuladores y se incluye además la descripción del desarrollo de un editor de escenarios virtuales que fue parte importante para la implementación de algunos de los simuladores anteriores.

3.1. Simulador de radar arpa MELIPAL-R (2001)

Melipal-R incluye el desarrollo e implementación de un simulador radar que cumple satisfactoriamente con los requerimientos de realismo y tiempo real, cuya primera versión se encuentra en operación en un esquema de cuatro puestos alumnos y un instructor en instalaciones de la Escuela Nacional de Náutica, y forma parte de los cursos obligatorios que deben cumplir los capitanes de embarcaciones. En primer lugar se describe el modelo propuesto y las aproximaciones realizadas, a continuación se discute brevemente su implementación computacional y finalmente se presentan los algoritmos con que se consigue obtener el costo deseado.

El problema y su discretización

El principio de funcionamiento de un radar se basa en la escucha de los ecos generados por la emisión de un corto pulso electromagnético. Este pulso debe ser colimado de forma que los ecos provengan mayoritariamente de la dirección que se está observando, sin embargo los pulsos que se logran en la realidad tienen el aspecto que se muestra en la figura 5, donde se observa un lóbulo principal con un cierto ancho en la dirección deseada y una serie de lóbulos laterales. La calidad del radar estará asociada a cuan pequeños son los lóbulos laterales y cuan estrecho es el lóbulo principal, ya que los blancos que se encuentren en las zonas alcanzadas por estos lóbulos podrán generar ecos y es claro que aquellos que no provengan de la dirección principal serán interpretados erróneamente.

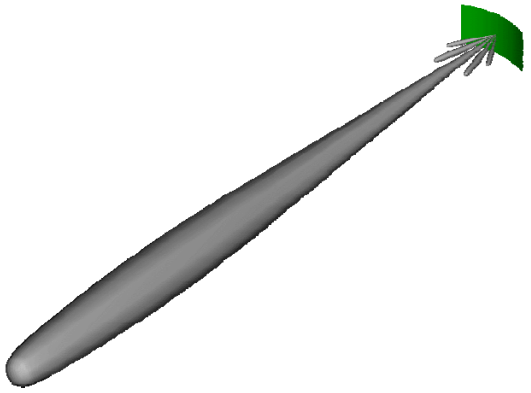


Figura 5: Esquema de un pulso del radar. La zona gris representa la región donde un blanco generará un eco

En el caso que nos interesa, radares en barcos, este pulso es emitido horizontalmente en distintas direcciones, girando la antena a razón de 14 a 20 revoluciones por minuto, con aproximadamente 4000 procesos emisiones-escuchas cada 360 grados.
La discretización natural de este problema puede realizarse considerando al lóbulo como un haz de rayos y a los blancos como un conjunto de polígonos en un espacio tridimensional, tal como se representa en la figura 6. Un proceso de emisión-escucha se transforma entonces en encontrar las intersecciones de este haz de rayos con todos los polígonos, graficando luego en la pantalla del radar la primera intersección para cada rayo. El costo computacional para este algoritmo seria $\boldsymbol{O(N_{pol}.N_{ray})}$, donde $\boldsymbol{N_{pol}}$: Número de polígonos empleados para describir el escenario y los blancos, $\boldsymbol{N_{ray}}$: Número de rayos utilizados para discretizar el lóbulo.
Normalmente la representación de un escenario tridimensional

(topografía de la costa, edificaciones, otros barcos y boyas) se realiza utilizando triángulos y aún para discretizaciones groseras será necesario utilizar del orden de 10^4 a 10^5 triángulos. Si se tiene en cuenta que el cálculo de la intersección entre una recta y un triángulo en el espacio requiere más de cincuenta operaciones de punto flotante, además de cuatro raíces cuadradas, rápidamente se llega a tiempos de cálculo incompatibles con una aplicación en tiempo real, al menos en computadoras estándar.

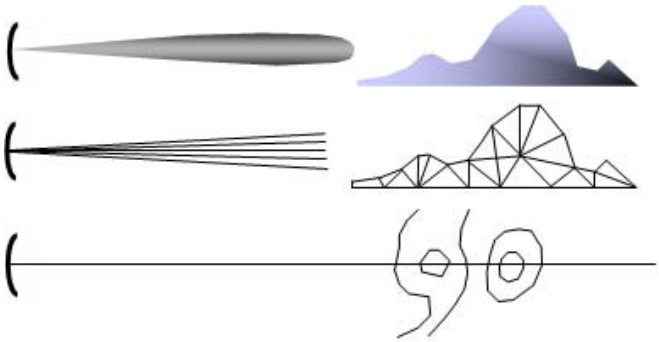


Figura 6: Discretización y aproximación del problema.

Por este motivo hemos propuesto un modelo en donde la tercera dimensión es condensada y el lóbulo es considerado como un único rayo paralelo a la superficie. Para eliminar la tercera dimensión y no perder la información de las pendientes de terreno, representamos la topografía y construcciones mediante líneas de nivel, recurriendo a la interpolación de alturas entre las mismas. De esta forma se obtienen imágenes de la línea de costa muy similares a la que se observa en un radar real y el problema se redujo a evaluar intersecciones entre una semi-recta y el conjunto de segmentos que definen las líneas de nivel.
Para definir la cantidad de puntos a interpolar entre dos líneas hemos propuesto una dependencia proporcional a la diferencia de altura entre las mismas; de esta forma un acantilado se verá nítidamente, mientras que una playa aparecerá en forma difusa. Por otro lado la intensidad de estos ecos en la pantalla es una función del blanco (vegetación produce un eco muy distinto que un edificio por ejemplo) y para considerar también este efecto hemos incorporado una propiedad "intensidad" en cada segmento utilizado para definir las líneas de nivel.

En la figura 7 se muestran las intersecciones detectadas para una dada posición del haz de barrido. Para encontrar los puntos que deberán ser pintados en la pantalla del radar resulta conveniente escribir la

ecuación de la recta en forma paramétrica y ordenar las intersecciones en orden creciente del parámetro ***t***,

$X(t) = cos(\theta).t + Px$

$Y(t) = sin(\theta).t + Py$

(Px,Py): Posición del barco

θ: Angulo donde apunta la antena

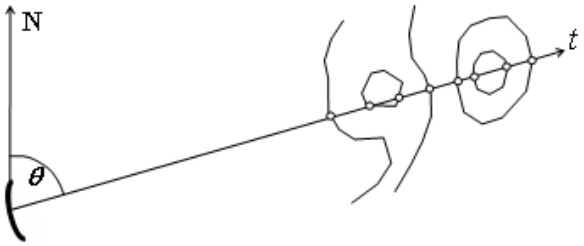


Figura 7: Intersecciones detectadas para una dada posición del haz de barrido.

De esta forma el algoritmo de pintado de píxeles en la pantalla del radar consiste en recorrer la lista ordenada de intersecciones t_i, y en el caso de que la altura de la misma sea mayor que la anterior t_{i-1}, se pintará al píxel con la intensidad indicada en el segmento y se generan **n** puntos adicionales entre t_{i-1} y t_i, siendo **n** proporcional a la diferencia de alturas. En el caso que la altura en t_i sea menor que en el punto anterior, no se pinta el píxel ni se interpolan nuevos puntos, pero la lista debe seguirse recorriendo para considerar el caso que una intersección más lejana posea una altura suficiente para ser detectada.

Resumiendo entonces, los blancos (móviles o fijos) son representados como un conjunto de segmentos. Cada segmento tiene asociado una altura y una intensidad. Esta intensidad es en general un valor positivo, por lo que valores negativos pueden reservarse para indicar segmentos que no ocultan los objetos que están detrás, como puede ser el caso de puentes.

La figura 8 muestra una imagen radar virtual generada con esta metodología correspondiente a una zona cerca del Golfo San Matías. Obsérvese como se visualizan en forma diferente las zonas de playa al sur comparadas con las costas más abruptas al oeste. La figura 9 muestra otra imagen correspondiente al puerto de Buenos Aires donde además de las dársenas se pueden ver algunos edificios y la sombra que generan los mismos. También se pueden ver otros dos navíos, uno dentro y otro fuera del puerto.

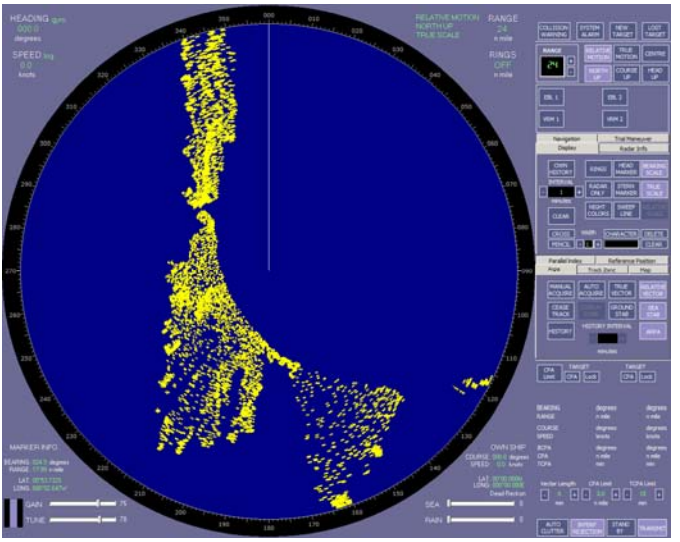


Figura 8: Imagen radar virtual generada para la costa en la zona del Golfo San Matías.

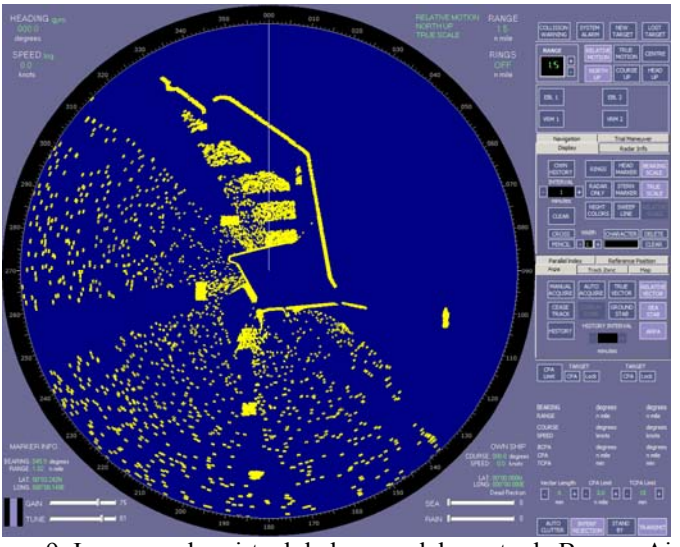


Figura 9: Imagen radar virtual de la zona.del puerto de Buenos Aires

Optimización de performance

Como ya se mencionó, un radar como el que nos interesa debe poder girar a 14 revoluciones por minuto como mínimo, y teniendo en cuenta los 4000 procesos de emisión cada 360 grados, se necesitaran evaluar aproximadamente 1000 de estos procesos por segundo. Si bien con la aproximación realizada al eliminar la tercera dimensión se

reduce notablemente el costo computacional, en caso de escenarios muy complejos pueden ser necesarios miles de segmentos y la performance del algoritmo puede no ser la requerida. De hecho en las pruebas iniciales realizadas sobre los escenarios más complejos, procesando sobre equipos Pentium III de 1.0 GHz, el rendimiento obtenido era de apenas cinco revoluciones por minuto.
Del análisis realizado se detectó que el costo computacional principal estaba asociado al cálculo de las intersecciones rayo-segmentos, ya que para cado rayo se analizaban la totalidad de los segmentos. Se optimizó en lo posible esta operación, dejando incluso precalculados aquellos coeficientes que no varían para cada una de las 4000 semirectas, pero aún así se estaba lejos de los tiempos requeridos.

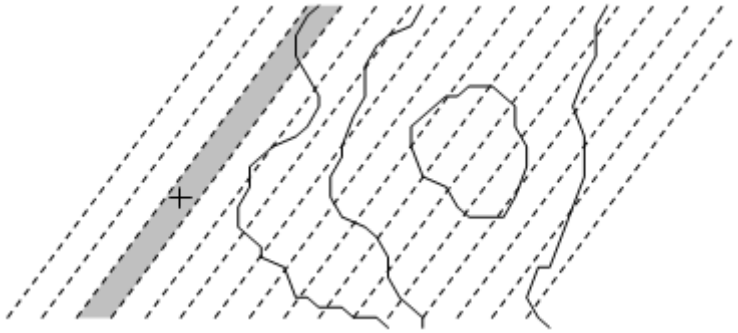


Figura 10: Clasificación de los segmentos en bandas equiespaciadas

La solución clásica a este tipo de problemas es recurrir a técnicas de clasificación geométrica utilizando estructuras auxiliares como quadtrees o grillas regulares. Sin embargo el caso que nos interesa presenta particularidades que vuelven poco adecuados a todos estos métodos.
Dado que se buscan todas las intersecciones de segmentos con una semi-recta, el método de clasificación deberá permitir encontrar fácilmente los segmentos que se ubican en una banda que contiene a la semi-recta. Esto no resulta sencillo utilizando una grilla y mucho menos con un quadtree. La estructura ideal sería clasificar en bandas paralelas a la semi-recta, tal como se muestra en la figura 10. La implementación más sencilla es tomar un ancho de banda fijo con un tamaño tal que ninguna tenga más de una cierta cantidad dada de segmentos. Esta condición puede resultar muy exigente para casos en que una gran cantidad de segmentos están en línea, en cuyo caso debería utilizarse bandas de tamaño variable (será necesario un árbol binario para encontrar la banda que corresponde a la posición del barco).

En nuestro caso se realizó una implementación con bandas fijas en 360 direcciones diferentes, consiguiéndose una mejora notable en el tiempo de ejecución, permitiendo cumplir holgadamente con el requerimiento de más de 14 revoluciones por minuto, aún para los escenarios más complejos.
El cálculo de las intersecciones para un dado rayo se realiza entonces en tres pasos: en primer lugar, dado el ángulo de la semirecta se selecciona la clasificación en bandas correspondiente; luego dada la posición del barco se evalúa cual es la banda que le corresponde y se recuperan los segmentos asociados a la misma y a la vecina (si se realizaran 4000 clasificaciones no sería necesario tomar la vecina inmediata, pero eleva considerablemente el costo de almacenamiento); finalmente se calculan las intersecciones entre la semirecta y los segmentos.
El simulador debe barrer la pantalla a un ritmo constante independientemente de la complejidad del escenario, por lo cual el cálculo de cada una de las 4000 direcciones se realiza con un proceso independiente que se ejecuta a intervalos regulados por el proceso principal del radar.

Sistema de entrenamiento

Para que el simulador radar desarrollado pueda utilizarse con fines de entrenamiento, debe también simularse que el mismo se encuentra sobre un barco que se desplaza en un dado escenario y con la presencia de otros barcos, también en movimiento. El sistema completo, llamado Melipal-R y esquematizado en la figura 11, comprende varios puestos de entrenamiento distribuidos en una LAN monitoreados por un instructor, donde cada puesto esta formado por tres equipos: el radar, los controles del barco y la cartografía.
La computadora que presenta los controles del barco (básicamente el timón y el control de potencia) realiza también una simulación del desplazamiento del barco considerando modelos simplificados para embarcaciones que van desde supertanques, transportador de contenedores a embarcaciones livianas). La cartografía es presentada con un sistema comercial estándar el cual permite además ingresar una señal de equipo GPS (simulada también por el sistema) para una ubicación automática sobre el mapa.
Por último el puesto del instructor dispone de herramientas para seleccionar escenarios, asignar embarcaciones, crear situaciones, monitorear desplazamientos e incluso visualizar en tiempo real, sobre un equipo auxiliar, el estado de cualquier radar de los puestos de entrenamiento a efectos de analizar si el mismo esta siendo utilizado

correctamente. También cuenta con la posibilidad de grabar completamente un ejercicio para una posterior reproducción y estudio.

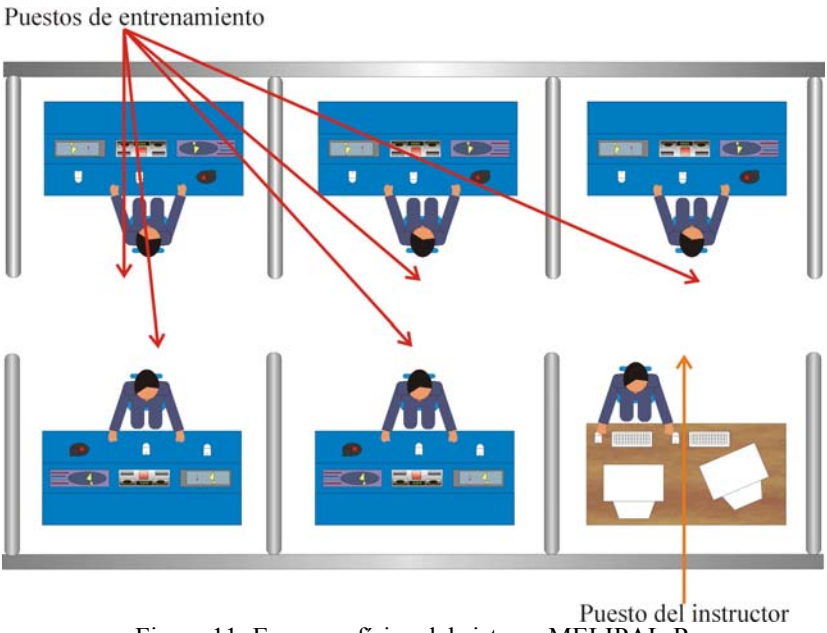


Figura 11: Esquema físico del sistema MELIPAL-R

3.2. Simulador de submarino SIPER (2002)

El proyecto SIPER fue un encargo de la Escuela de Submarinos y Buceo SIAG de la Armada Argentina. En la figuras 12 y 13 puede verse el interior de un submarino real.
El simulador se divide en distinguen dos partes principales (figura 14):

- ADITACSUB
- SIPER

Para el desarrollo de SIPER se contó con un periscopio real el cual se adaptó integrando unos visores de realidad virtual en el visor real. La figura 15 muestra el periscopio sobre una mesa de las instalaciones del Pladema mientras se realizaba la adaptación del visor.
Entre los requisitos del sistema de RV SIPER se encontraba el modelar escenas virtuales 3D que incluyeran:

- el mar en diferentes estados (calmo, oleaje suave, oleaje de tormenta, etc.)
- el cielo en diferentes condiciones climáticas
- las costas
- navíos y aeronaves

Figura 12. Fotografía del interior de un submarino

Figura 13. Interior de un submarino

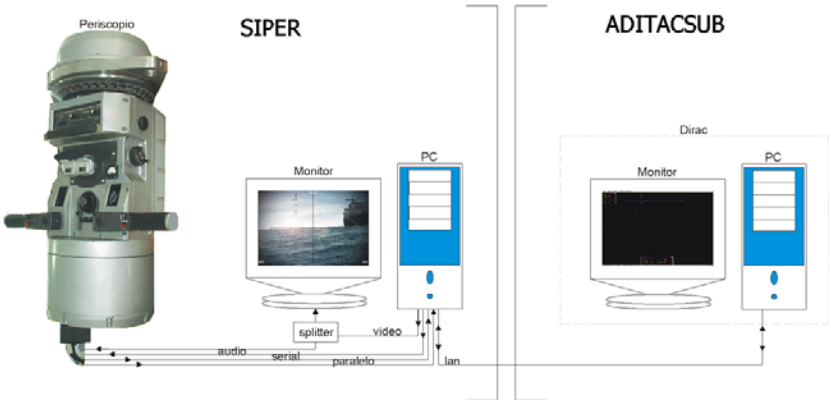


Figura 14. Simulador de submarino: SIPE y ADITACSUB

Figura 15. Adaptación de un periscopio real en las instalaciones de Pladema

Se requirió también la incorporación de los siguientes efectos adicionales:

- tiempo de izado y visión submarina
- bigote de los barcos circundantes
- escurrimiento de calota
- precipitaciones
- sol, luna y estrellas
- luces de navegación

Las funcionalidades con las que debía contar el periscopio en tiempo real fueron:

- giro 360º
- control de elevación
- tres niveles de ampliación
- filtros solares
- desfasaje retículo estadimétrico
- iluminación del retículo

La implementación fue realizada utilizando el lenguaje VRML bajo Cortona para la representación del escenario tridimensional y se agregó funcionalidad mediante JavaScript. El hardware utilizado fue un procesador Athlon 2.0 GHz con 512 MB de RAM y una placa gráfica GeForce4 MX440 AGP8X. Como resultado se obtuvo una

visualización de 10 imágenes por segundo. Para poder alcanzar esta velocidad hubo necesidad de interpolar posiciones del periscopio.
Se logró implementar una herramienta para entrenamiento de operarios de calidad superior a los productos disponibles, basados en plataformas de muy bajo costo y lenguajes de alto nivel.
En la figura 16 puede verse una imagen donde se visualiza un navío en las proximidades. Puede observarse el oleaje, el efecto del reflejo de la luz del sol. La figura 17 muestra la opción de duplicación de imagen la cual permite estimar la distancia a la que se encuentran los barcos de acuerdo a la diferencia existentes entre ambas imágenes. Para ello fue necesario realizar un renderizado doble de la escena desde dos posiciones diferentes de la cámara virtual. La figura 18 muestra la visualización de la costa cercana, un cielo soleado, y un mar calmo.

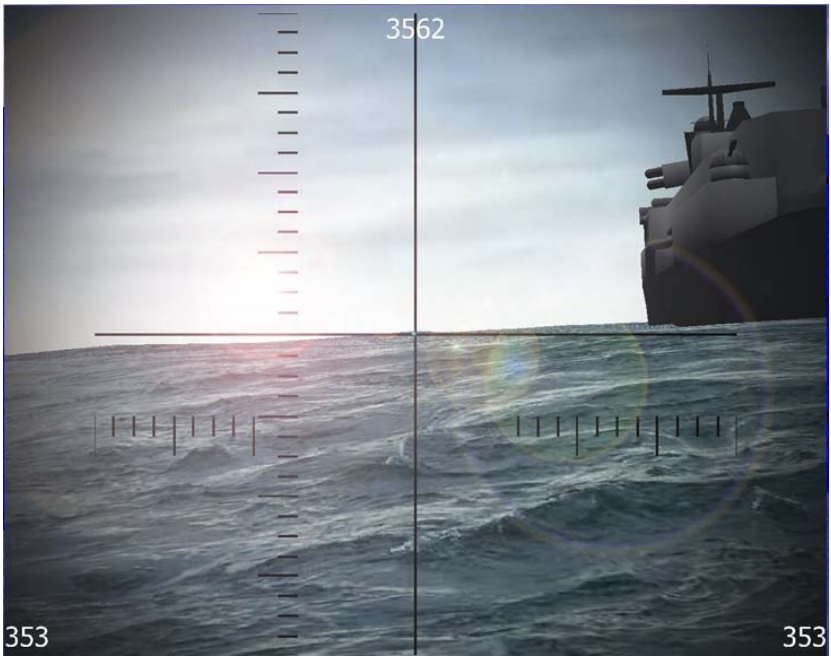


Figura 16. Visualización 3D del entorno del simulador SIPER

Figura 17. Visualización 3D del entorno del simulador SIPER: renderizado doble para estimación de distancias

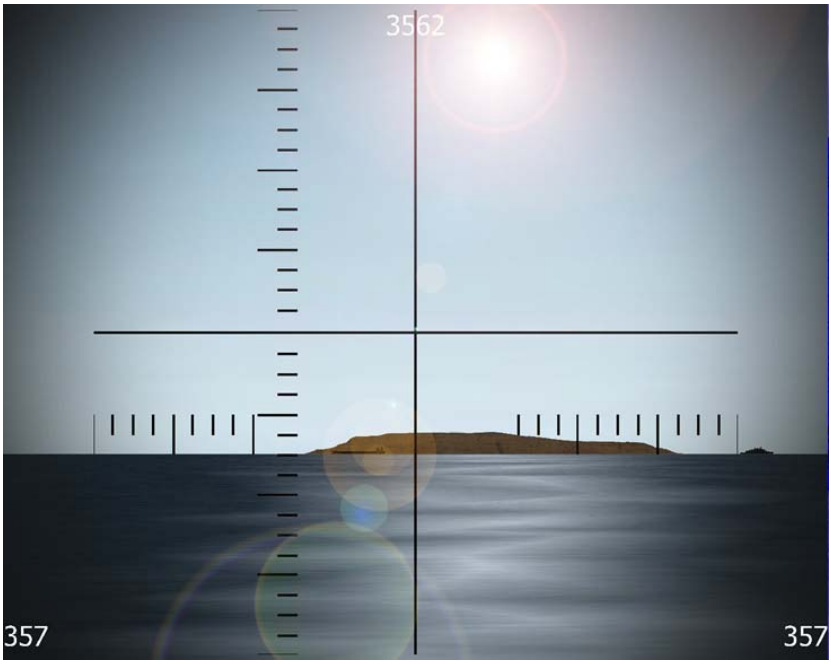


Figura 18. Visualización 3D del entorno del simulador SIPER: visualización de la costa

3.3. Simulador de embarcación pesquera MELIPAL-P (2003)

El sistema Melipal-P es un simulador desarrollado para el entrenamiento de operarios de buques de pesca que se encuentra instalado en la Escuela Nacional de Pesca Luis Piedrabuena. Simula el comportamiento de una ecosonda vertical, un sonar de exploración y una ecosonda de red.
Está basado en el simulador de radar arpa Melipal R. El módulo cuenta demás con un monitor de red que ayuda a brindar un soporte pedagógico al alumno que está siendo entrenado como puede verse en la figura 19 una imagen de la consola de los operarios y del instructor. La figura 20 muestra la interfase de la aplicación para el instructor. La figura 21 muestra la interfase de la aplicación para los operarios a quienes se le presentan los controles del barco y el radar.
Las componentes a modelar en el desarrollo del simulador fueron:

- Radar
- Ecosonda
- GPS
- Sextante
- Cartografía
- Indicadores del barco
- Comunicaciones
- Visual frontal y laterals

Figura 19. Operarios e instructores en el simulador Melipal-P

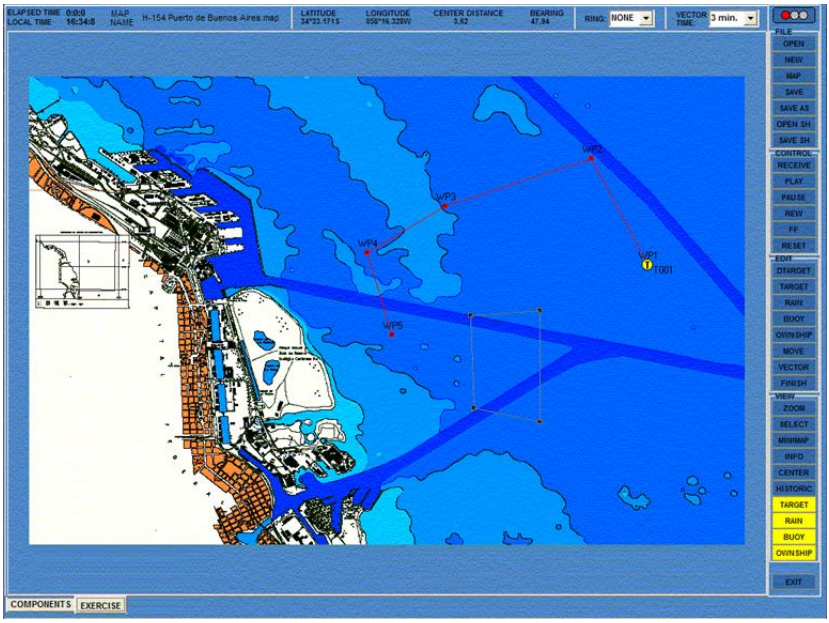


Figura 20. Interfase del instructor en el simulador Melipal-P

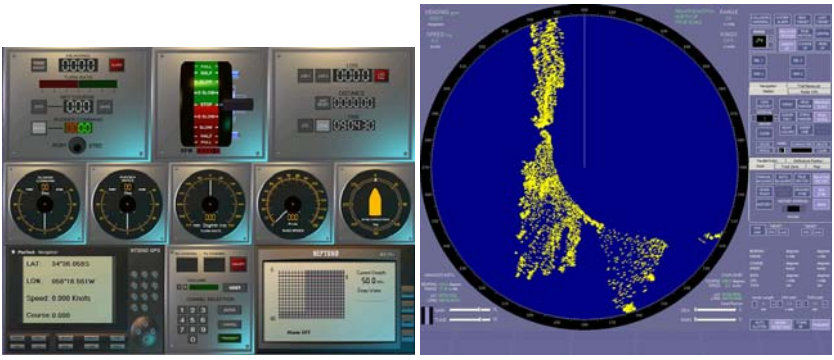


Figura 21. Interfase de los operarios en el simulador Melipal-P

En la figura 22 puede verse un esquema del sistema completo.
Adicionalmente, se trabajó en un módulo visual de la vista del puente del buque, recreando el escenario marítimo; si bien este módulo no fue incluido en la versión final del simulador, fue evaluado por marinos idóneos de la E.N.P. Pueden encontrarse más detalles en la publicación de D'Amato et al (2004) [5].
En la figura 23 puede verse el módulo de instrumentos de pesca del simulador de entrenamiento Melipal-P en la Escuela Nacional de Pesca de Mar del Plata.

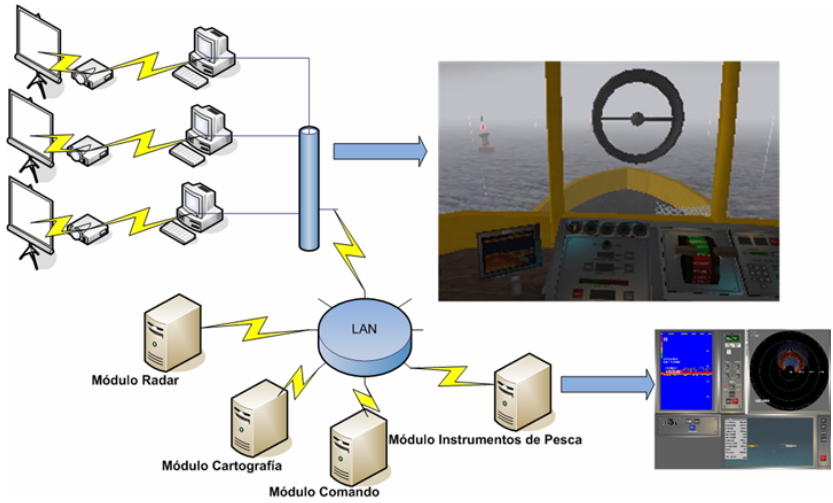


Figura 22. Esquema del sistema del simulador Melipal-P

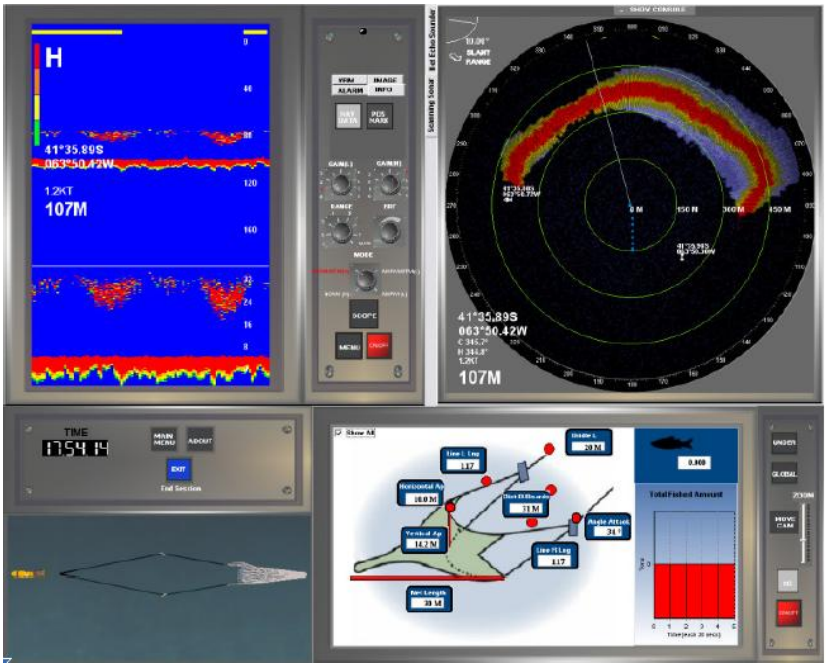


Figura 23. Módulo de instrumentos del sistema del simulador Melipal-P

3.4. Editor de escenarios virtuales MELIPAL-ED (2005)

Melipal-Ed es un editor con capacidades GIS y soporte 3D para crear escenarios georeferenciados, con la posibilidad de importar distintos

modelos 3D desde diversas aplicaciones del mercado y exportar estos escenarios a un formato reconocible por los entornos de visualización de la familia de simuladores Melipal.
El trabajo realizado estructura un editor en una jerarquía de módulos, utilizando georeferenciación para todos los objetos 3D colocados en los escenarios. Como base del escenario virtual se usan modelos topográficos digitales de alta precisión, los cuales presentan una gran cantidad de polígonos. En el editor, el usuario coloca puntos de referencia de acuerdo a la posición verdadera de latitud y longitud de un objeto y le asocia un modelo 3D virtual.
Los escenarios creados son luego utilizados en módulos visuales de simuladores existentes logrando representaciones realistas. Para permitir una navegación fluida se creó un algoritmo de remallado para recorrer modelos digitales densos (millones de triángulos) en tiempo real.
Fue necesario también implementar distintos tipos de algoritmos no triviales para el resto de los módulos del editor, por ejemplo, la simulación de clima, definición de objetos estáticos y móviles, incorporación de objetos con comportamiento (fuego, agua, etc.).
En las publicaciones de D' Amato et al (2005)[6] (2007) [8] se detalla el diseño del editor de modelos virtuales. Dado que los modelos topográficos constan de miles de polígonos, y no siempre es necesario tener la máxima resolución del modelo ya que en las zonas más alejadas no suele visualizarse el detalle, es necesario realizar la simplificación de dichos modelos. En la publicación de Cifuentes et al (2005) [1] se detalla el método de remallado interactivo el cual posibilita la navegación eficiente de los modelos topográficos. No solo la geometría del modelo puede simplificarse sino que también pueden simplificarse las texturas añadidas. En las publicaciones de Cifuentes, V. et al (2006) [2] (2007) [4] se presenta un análisis multicriterio para la simplificación conjunta de geometría y textura de los modelos de terreno. Aquellas zonas de interés, en particular aquellas zonas a las que el usuario dirige su visión, deben detallarse mientras que las zonas más alejadas pueden mostrarse con una menor resolución. En la publicación de Cifuentes et al (2006) [3] se describe el algoritmo de *Ray Casting* utilizado para la definición de zonas de interés dentro del modelo de terreno simplificado.
En la figura 24 puede verse el editor de escenarios virtuales para módulos de visualización de los simuladores de entrenamiento Melipal.

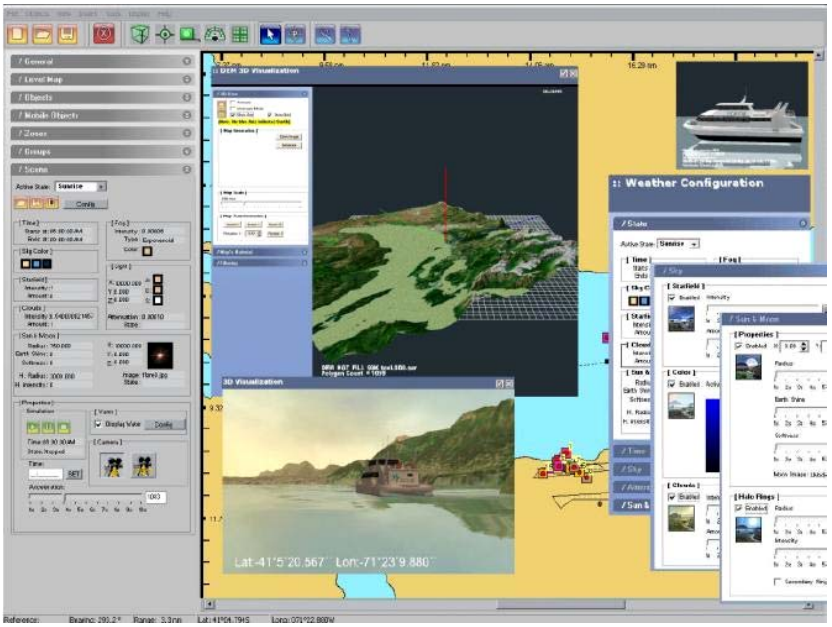


Figura 24. Editor de escenarios virtuales Melipal-Ed

3.5. Simulador de radar terrestre RASIT (2005)

Simulador para el entrenamiento en la utilización de un dispositivo radar RASIT. Se implementó una aplicación que permite ejecutar simulaciones de un dispositivo radar en escenarios sintéticos. Las imágenes obtenidas fueron evaluadas por idóneos de las Fuerzas Armadas, los cuales dieron sus opiniones y brindaron aportes sobre características de color, forma y representación de los objetos en la pantalla radar.

El algoritmo utilizado para la simulación presentada en pantalla, funciona básicamente bajo el concepto de *raycasting*, en donde un rayo con distancia y acimut (representando la señal de radar) es proyectado sobre modelos digitales de elevación –conocido también por su sigla *DEM* (*Digital Elevation Model*)–, representando el terreno real en el que se desarrolla el ejercicio).

Para que este algoritmo funcione de manera fluida fue necesario trabajar con modelos optimizados del DEM original, para los cuales se implementaron y testearon distintos algoritmos de simplificación.

El radar RASIT es un radar terrestre, utilizado en campaña por el ejército, el simulador realizado de este dispositivo será utilizado para entrenar operarios.

Continuando con el proyecto RASIT iniciado en el año 2005, se diseñó e implementó una aplicación, denominada Consola de

Operación, que permite operar un radar real remotamente y mostrar al operador la información recogida de forma vectorial y georeferenciada. A diferencia de las herramientas GIS comerciales, el usuario tiene la posibilidad rotar la información y todas las capas de información sobre un punto de manera de poder orientar la visualización hacia los 4 cuadrantes E, O, N, S. También permite la conversión a sonido del resultado de la escucha doppler.
En la figura 25 puede verse un usuario operando la consola de radar RASIT y en la figura 26 se muestra una imagen obtenida con el simulador de radar terrestre RASIT.

Figura 25. Operador del simulador de radar RASIT

Figura 26. Interfase del simulador de radar RASIT

3.6. Simulador de radar aéreo ATLAS 2 (2007)

Simulador del radar de una aeronave para ser utilizado en el sistema de entrenamiento militar ATLAS 2 (Adiestrador Táctico de Lucha Antisubmarina). Contempla cuatro modelos de radar para diferentes aeronaves (APS 88, RDR 1500, APS 80, APS 705). A diferencia de un radar de embarcación, este radar funciona completamente en tres dimensiones, visualizando topografía y objetos tanto sobre la superficie con en el aire.
El simulador permite recrear el comportamiento de un radar real y mostrar al operador la información recogida de forma vectorial y georeferenciada. Tal simulador, es un módulo de un sistema distribuido, denominado ATLAS 2, por lo que se requirió manejar información de distintas fuentes, interactuar con otras partes del sistema y con el operador. Para una descripción más detallada se puede consultar la publicación D'Amato et al (2007) [7].

Figura 27. Cabina del sistema ATLAS 2 (SIAG, Puerto Belgrano), en la esquina superior derecha se encuentra el simulador de radar aéreo

La topografía como los objetos del escenario son descriptos mediante polígonos El lóbulo de emisión se modeló discretizando el mismo en un conjunto de rayos, por lo que la imagen se genera evaluando las intersecciones de estos rayos con los polígonos presentes en el escenario. Se propusieron optimizaciones basadas en simplificación poligonal de los objetos en el escenario y clasificación geométrica para resolver rápidamente las intersecciones.

En la figura 27 puede verse la cabina del sistema ATLAS 2 (SIAG, Puerto Belgrano), en la esquina superior derecha se encuentra el simulador de radar aéreo.

3.7. Simulador de maniobras MELIPAL-M (2008)

El proyecto denominado Melipal-M busca materializar un simulador de maniobra para buques de superficie, con todos los componentes necesarios para la capacitación efectiva del personal naval.
Para este proyecto se utiliza como base el sistema Melipal R, el cual fue desarrollado por nuestro instituto en el año 2001. El sistema Melipal-R es un prototipo de simulador de entrenamiento, el cual fue cedido en dicho año a la escuela Nacional de Náutica de la Armada Argentina (actualmente se encuentra en Puerto Belgrano, en el Instituto de Servicio de Análisis Operativo Armas y Guerra Electrónica), y que hasta el día de hoy ha sido utilizado como presentación de futuros desarrollos.
La ejecución de este proyecto implica realizar la reingeniería completa del sistema Melipal-R y agregar dos módulos nuevos:

- Implementar un módulo de visualización 3D encargado de simular y visualizar efectos climáticos, tipos de agua marítima, estados de mar y las distintas fases del día con realismo.
- Desarrollar un modelo matemático discreto y simplificado, para predecir la dinámica de maniobra y la cinemática de movimiento de barcos en base a sus características y a los valores de velocidades y aceleraciones instantáneas.

En la figura 28 puede verse una práctica de entrenamiento utilizando el sistema Melipal-M.

Figura 28.Práctica de entrenamiento utilizando el sistema Melipal-M. (Escuela de Oficiales de la Armada, Puerto Belgrano)

4. Conclusiones

De los tres frentes en que actualmente se está trabajando en investigación y nuevos desarrollos, sin duda el de lograr renderizado de calidad es el que está más cerca de una solución. No cabe duda que en un par de años estaremos viendo juegos y aplicaciones de RV que utilicen como herramienta de rendering un algoritmo de ray-traycing en tiempo real. De hecho es posible que la arquitectura de las placas gráficas empiecen a contemplar los requerimientos de esta técnica (solo estaría faltando contar con algún medio para mantener clasificados en el espacio los triángulos que describen la escena, de forma de resolver rápidamente la intersección rayo-triángulo).
El lograr sensación inmersiva, cae más en el área de hardware que en la de software, mientras que por el contrario, el modelado de comportamiento requiere de nuevos desarrollos en el área de métodos numéricos, en especial en la implementación eficiente de los mismos en dispositivos de alta performance económicos, como placas GPUs u otros más específicos como la línea Tesla que sacó NVidia.

5. Referencias

[1] Cifuentes, V.; D'Amato, J.; García Bauza, C.; Vénere, M. y Clausse, A. (2005). "Remallado Interactivo como Método para la Navegación Eficiente de Modelos Topográficos". *Mecánica Computacional,* XXIV, pp. 2385-2395, ISSN1666-6070.
[2] Cifuentes, V.; D'Amato, J.; García Bauza, C.; Lotito, P.; Vénere, M. y Clausse, A. (2006). "Análisis Multicriterio para la simplificación conjunta de geometría y textura de terrenos".*CACIC 2006. Actas del XII Congreso Argentino de las Ciencias de la Computación.*
[3] Cifuentes, V.; D'Amato, J.; García Bauza, C.; Lotito, P. ; Vénere, M. y Clausse, A. (2006). "Ray Casting para la Definición de Zonas de Interés en Simplificación Topográfica". *Mecánica Computacional,* XXV, pp. 1177-1186. ISSN 1666-6070, 2006.
[4] Cifuentes, V.; D'Amato, J.; García Bauza, C. y Vénere, M. (2007). "Técnicas de simplificación de modelos topográficos". *WICC 2007. Actas del IX Workshop de Investigadores en Ciencias de la Computación*, pp. 299-303.
[5] D'Amato, J.; García Bauza, C. y Vénere, M. (2004). "Simulación del Entorno de una Embarcación Pesquera". *33 JAIIO, 2004. Actas de las 33 Jornadas Argentinas de Informática e Investigación Operativa.*

[6] D'Amato, J.; García Bauza, C. y Vénere, M. (2005). "Editor de Escenarios para Aplicaciones de Realidad Virtual". *34 JAIIO, 2005. Actas de las 34 Jornadas Argentinas de Informática e Investigación Operativa.*
[7] D' Amato, J.; García Bauza, C.; Truchi, M. y Vénere, M. (2007) "Algoritmos geométricos para la simulación de un radar de aeronave". *Mecánica Computacional*, XXVI, pp 962-974. ISSN 1666-6070.
[8] D' Amato, J.; García Bauza, C.; Cifuentes, V. y Vénere, M. (2007) "Herramienta para el desarrollo de escenarios virtuales". *JIDIS 2007. Actas de las 2das Jornadas de Investigación y Desarrollo en Ingeniería de Software.*

CAPÍTULO 3

Realidad Aumentada

María José Abásolo, Universidad Nacional de La Plata, Argentina

1. Introducción

Según la definición que aparece en una de las primeras publicaciones relevantes sobre realidad aumentada que debe ser de las más citadas, Azuma (1997) [4], una aplicación de realidad aumentada es una aplicación interactiva que combina la realidad con imágenes sintéticas 3D registradas en tiempo real.
De acuerdo a esta definición se pueden citar como ejemplo de lo que no se consideraría realidad aumentada a los efectos especiales en películas (ya que no se calculan en tiempo real), o a una imagen fija aumentada con imágenes sintéticas (ya que no es interactiva).
Desde un punto de vista más amplio la realidad aumentada es una aplicación interactiva que combina la realidad con información sintética –tal como imágenes 3D, sonidos, videos, texto, sensaciones táctiles– en tiempo real y de acuerdo al punto de vista del usuario.
Según el tipo de aplicación será el dispositivo de visualización adecuado a la misma. En particular las aplicaciones de realidad aumentada pueden utilizar diferentes dispositivos de visualización según sea el tipo de aplicación: monitor, proyector, dispositivos de visualización específicos de realidad aumentada como son las gafas de video "see-through"[1] y las gafas de óptica "see-through", o dispositivos móviles- en inglés "hand held", como los teléfonos celulares de última generación.
En la sección 2 se describen las partes de una aplicación de realidad aumentada. La sección 3 detalla los fundamentos del proceso de generación de una imagen virtual de manera de dar un marco teórico a los algoritmos de realidad aumentada.
En Azuma (1997) [4] pueden encontrarse ejemplos de los primeros dominios de aplicación típicos de la realidad aumentada como son:

[1] La traducción de "see-through" significa "ver a través", ya que las gafas dejan ver la realidad a diferencia de los dispositivos de realidad virtual.

medicina, aplicaciones militares, asistencia a la fabricación, asistencia en la realización de tareas (por ejemplo, montaje, reparaciones, etc.), inspección anotación de escenas, planificación de la trayectoria de un robot, entretenimiento.

La aparición de la librería de código abierto ARToolkit [7] en el año 1999 dio lugar a un gran crecimiento en el desarrollo de aplicaciones de realidad aumentada. Azuma et al (2001) [5] presenta una actualización de la publicación anterior donde distingue tres grandes áreas de nuevas aplicaciones clasificadas en aplicaciones móviles, aplicaciones colaborativas y aplicaciones comerciales, que desde la fecha han ido en auge creciente.

Mientras que las primeras aplicaciones de realidad aumentada se desarrollaban en un entorno conocido y controlado, el interés en aplicación de realidad aumentada para un ambiente externo –en inglés, *outdoor*– fue creciendo Dentro de las aplicaciones móviles pueden encontrarse aplicaciones para asistencia a la navegación, la recuperación de información geográfica, aplicaciones de arquitectura, museística, juegos, etc. Dentro de las aplicaciones colaborativas pueden encontrarse aplicaciones de diseño, entretenimiento y educación. En los últimos diez años hubo una explosión de aplicaciones comerciales –publicitarias– que hacen uso de la realidad aumentada. En particular, pueden encontrarse muchos ejemplos de aplicaciones web donde el usuario interactúa con una escena virtual por medio de un marcador impreso (figura 1).

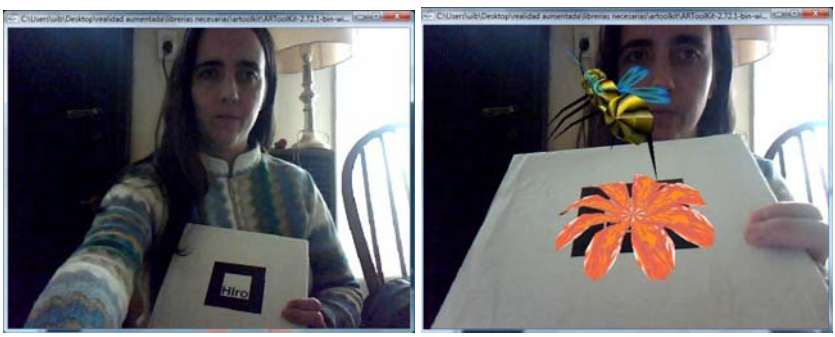


Figura 1. Aplicación de realidad aumentada basada en el reconocimiento de marcadores utilizando la librería ARToolkit

En las diferentes aplicaciones de realidad aumentada, se puede distinguir aplicación con diferentes requerimientos con respecto al tracking del usuario, es decir, el seguimiento. Por una parte, se tienen aplicaciones "serias" como sería el caso de aplicaciones médicas, donde el tracking es crítico ya que requiere imágenes reales y virtuales perfectamente registradas. Por otra parte, se tienen aplicaciones "no

serias" como pueden ser las aplicaciones de publicidad o los juegos, donde el tracking no es crítico dado que se permiten pequeños errores de registro de imágenes aunque sea deseable buenos resultados de integración. En la sección 4 se verán diferentes formas de realizar el tracking en aplicaciones de realidad aumentada. Como se verá el tracking basado en la extracción de características naturales en la escena es una de las áreas de investigación dentro de la realidad aumentada con más auge actualmente.

La gran masificación de los celulares así como el aumento reciente de su capacidad de procesamiento y memoria y la integración de dispositivos como GPS, brújulas, acelerómetros y giroscopios, hace que exista un gran campo de investigación y desarrollo de algoritmos y aplicaciones de realidad aumentada para este tipo de dispositivos. En la sección 5 se resumen algunos aspectos del desarrollo de aplicaciones de realidad aumentada para dispositivos móviles.

En la sección 6 se presentan las librerías de tracking así como frameworks y herramientas de autor de realidad aumentada más conocidos.

Recientemente aparecieron una serie de nuevas aplicaciones que son consideradas por el público como aplicaciones de realidad aumentada si bien los más puristas no las consideran así. Los denominados servicios basados en la geolocalización son aplicaciones para teléfonos móviles que brindan información al usuario de acuerdo a su posición geográfica obtenida vía GPS. Recientemente los servicios basados en la localización se vincularon con el campo de realidad aumentada dando lugar a la denominada realidad aumentada gravimétrica (*Gravimetric AR*) [36]. Los denominados browsers de realidad aumentada entre los cuales ofrecen servicios basados en la localización superponiendo información de acuerdo a la localización del usuario. Utilizan tanto el GPS y los sensores integrados para la localización de usuario, como también algoritmos de visión para el tracking, reconocimiento y visualización de información sobre los objetos reales. Dentro de la sección 6 se detallan tres browsers de realidad aumentada creados recientemente como son Layar [33], Wikitude [61], junaio [28]. El reconocimiento de imágenes u objetos para la recuperación de información también se asocia con realidad aumentada. Como ejemplo, de esto es la aplicación Google Goggles[2] la cual permite capturar un objeto y en base a la imagen capturada se realiza una búsqueda en la base de datos de Google para visualizar información sobre el mismo. Otro ejemplo es una aplicación sorprendente para teléfonos móviles –cuyo SDK perteneciente a la

[2] http://www.google.com/mobile/goggles.

empresa Viewdle[3] se anuncia como disponible– que realiza el reconocimiento de la cara de la persona capturada por la cámara para recuperar información del perfil social (Twitter, Facebook, LinkedIn, etc.).
Por último en la sección 7 presenta las conclusiones y la sección 8 la bibliografía referenciada.

2. Diagrama de una aplicación de RA

Una aplicación de realidad aumentada tiene tres partes principales que son:
- Tracking
- Recuperación de información
- Salida de información

En adelante se hará referencia al tipo de aplicación de realidad aumentada donde las salidas son visuales. La figura 2 muestra un diagrama de las diferentes partes involucradas en una aplicación de realidad aumentada visual:
- Captura de la escena real: la escena real se captura mediante una os de visión. El video capturado puede utilizarse para tracking basado en visión, es decir basado en el análisis de la imagen mediante algoritmos de visión. También es necesaria la captura si la visualización se realiza de forma indirecta en dispositivos de video *"see-trough"*, monitor o dispositivos móviles *"hand held"*
- Tracking del participante: puede realizarse mediante dispositivos específicos o puede realizarse tracking basado en visión para lo cual es necesaria la captura de la escena real.
- Generador de la escena virtual: se tiene un mundo virtual, con la información de la posición y orientación del participante se genera una vista acorde del mismo
- Combinación del mundo virtual y la escena real: en los casos en los que existe una visión directa del mundo real la combinación se realiza directamente en el ojo del participante. En otro caso, se realiza un video resultado de la escena capturada y la escena sintética proyectada.

[3] http://www.viewdle.com.

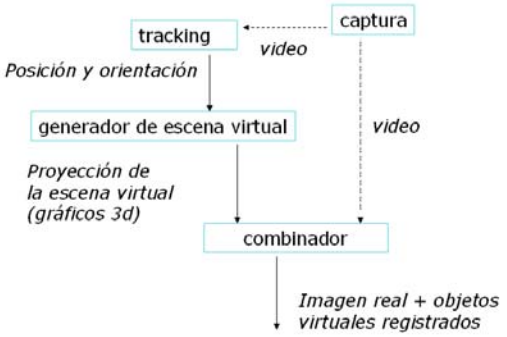


Figura 2 Diagrama de las partes de una aplicación de realidad aumentada

El dispositivo de visualización influye en las partes de la aplicación de realidad aumentada según muestra la siguiente tabla.

Dispositivo	**Captura**	**Tracking**	**Generador escena**	**Combinador**
Optical s-t	Visión directa.	De la cabeza. mediante sensores (generalmente electromagnético).	Software para PC.	Proyección de la escena sintética en las gafas transparentes. Combinación la realiza el ojo.
Video s-t	Cámara captura el video real.	De la cabeza.o cámara mediante sensores o basado en visión.	Software para PC.	Generación y visualización de video combinando la captura y la proyección de la escena sintética.
Proyector	Visión directa. Captura si se usa tracking basado en visión.	Opcional según la aplicación.	Software para PC.	Proyección de la escena sintética sobre la escena real, Combinación la realiza el ojo.
Monitor	Cámara captura el video real.	De la cámara. Generalmente basado en visión.	Software para PC.	Generación y visualización de video combinando la captura y

				la proyección de la escena sintética.
Hand-held	Cámara captura el video real.	De la cámara. Basado en visión o mediante sensores (GPS y brújula digital; sensores inerciales como acelerómetros y giroscopios).	Software para móviles.	Generación y visualización de video combinando la captura y la proyección de la escena sintética.

3. Generación de la escena virtual

3.1. Sistema de coordenadas

En el proceso de formación de imágenes, reales o sintéticas, existe:
- Una escena compuesta por diversos objetos (en diferentes posiciones y orientaciones)
- Una cámara, con una cierta posición y orientación en la escena
- Una imagen resultante de proyectar la escena de acuerdo a la cámara

Dado que el modelado de la escena virtual escapa de los objetivos de este texto no entraremos en detalles de este proceso.
Cuando se habla de posición y orientación de un objeto o de una cámara deber existir un sistema de referencia en base al cual se expresan. En realidad aumentada, al hablar de imágenes registradas significa que tanto las imágenes sintéticas como el mundo real estén en referencia al mismo sistema de coordenadas.
En adelante nos referiremos a una serie de sistemas de coordenadas que se detallan a continuación.
- Sistema de coordenadas local

Es un sistema de coordenadas 3D utilizado para referenciar los puntos de un objeto. Para entenderlo mejor puede pensarse que al modelar un objeto se utilizará un sistema de coordenadas local al objeto, situado en algún punto del mismo. Por ejemplo si se modela un árbol seguramente lo haremos situando un sistema de coordenadas centrado en el árbol o en la base del mismo, con un eje alineado con el tronco.
- Sistema de coordenadas mundo

La posición y orientación de los objetos de una escena suele expresarse en relación a un sistema de coordenadas 3D situado en

algún lugar del mundo. Al modelar una escena completa se utiliza un sistema de coordenadas único situado en algún lugar de la escena, denominado sistema de coordenadas mundo. Todos los objetos que integran una escena se posicionan, orientan y escalan en relación al mismo sistema de coordenadas mundo.

- Sistema de coordenadas cámara

El sistema de coordenadas cámara 3D tiene el origen en el centro óptico de la cámara. El eje de proyección de la cámara pasa por el centro óptico y es perpendicular al plano de formación de la imagen. El eje Z del sistema de coordenadas cámara está alineado con el eje de proyección, con Z+ hacia donde se ubica la escena. El eje Y tiene dirección vertical y el eje X dirección horizontal.

El sistema de coordenadas mundo 3D puede expresarse en relación al sistema de coordenadas cámara 3D mediante transformaciones geométricas. De forma equivalente, el sistema de coordenadas cámara puede expresarse en relación al sistema de coordenadas mundo.

- Sistema de coordenadas de la imagen

Los puntos de la imagen se expresan en relación a un sistema de coordenadas 2D con los ejes alineados con los bordes de la imagen.

3.2. Transformaciones geométricas

En computación gráfica tienen especial importancia las transformaciones afines. Una transformación afín f es una transformación lineal ya que

$f(\alpha p + \beta q) = \alpha f(p) + \beta f(q)$, siendo α y β escalares

En palabras, la transformación de cualquier combinación lineal de dos vértices puede obtenerse mediante la combinación lineal de las transformaciones de los vértices

En informática gráfica la ventaja de las transformaciones lineales radica en que si se quieren transformar todos los puntos de un segmento basta con transformar los extremos y luego trazar el segmento transformado uniendo los extremos transformados.

Las siguientes transformaciones geométricas son transformaciones afines: traslación, rotación y escalado. Ya que la traslación de un punto P es desplazarlo una distancia fija especificada por un vector **D**, una primera representación de la traslación es una suma como la de la figura 3.d.

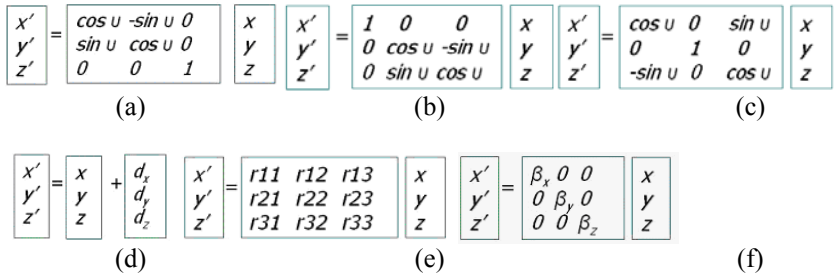


Figura 3. Expresión matricial de la rotación con respecto al eje Z (a), X(b) e Y (c), traslación (d), rotación general (e) y escalado en los tres ejes (f)

Por otra parte, la rotación de un cierto punto P es hacerlo girar un ángulo de rotación μ en relación a un eje de rotación determinado por un vector **E** que pasa por un centro de rotación C. Por ejemplo, la rotación de un punto P con respecto al eje principal **Z** y centro de rotación en el origen puede expresarse como muestra la figura 3.a. De forma similar pueden expresarse las rotaciones con respecto a los ejes principales **X** e **Y** (figura 3.b y 3.c). De una forma general la rotación puede expresarse como muestra la figura 3.e.

Para escalar un punto P se especifica un factor de escala β y un cierto vector **E** que indica la dirección de escalado que pasa por un cierto centro C. La forma más común de escalado de un punto P resulta de utilizar factores de escala βx, βy, βz para los ejes principales **X**, **Y**, **Z** respectivamente. El escalado puede expresarse como se muestra en la figura 3.f.

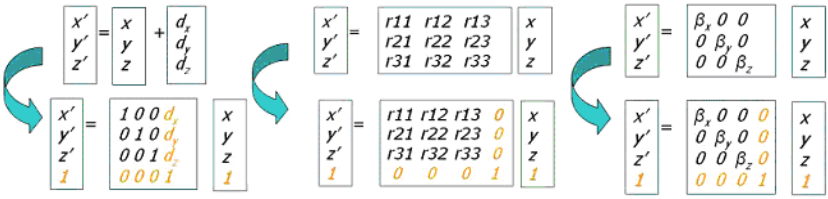
Por cuestiones de eficiencia en la implementación de la concatenación de transformaciones se necesita una representación homogénea de la traslación, la rotación y el escalado. Hasta el momento la representación no es homogénea ya que la traslación se expresa como una suma mientras que la rotación y el escalado se representan como multiplicaciones de matrices.

Para obtener una representación homogénea se recurre a las denominadas coordenadas homogéneas. Se considera que los puntos y vectores 3D se representan con 4 dimensiones de la siguiente forma:

Vector (x,y,z) es (x,y,z,0)

Punto (x,y,z) es (x,y,z,1)

Cada transformación afín 3D, tanto la rotación y el escalado como también la traslación, se representa por una matriz de 4x4. De izquierda a derecha, puede verse en la figura 4 la nueva representación homogénea de la traslación, la rotación y el escalado.



(a) (b) (c)

Figura 4. Expresión matricial homogénea de la traslación (a), rotación general (b) y escalado en los tres ejes (c)

Esta forma matricial uniforme permite la implementación eficiente de transformaciones sucesivas mediante la multiplicación de las matrices que las representan. Se debe cuidar el orden en que se multiplican las matrices ya que muchas combinaciones de transformaciones no son conmutativas.

Un punto P cualquiera del mundo 3D expresado en un cierto sistema de coordenadas local puede expresarse en un sistema de coordenadas mundo si se multiplica por la matriz de transformaciones geométricas T_{cm} (la cual expresa las rotaciones, escalados y traslaciones del sistema de coordenadas local en relación al sistema de coordenadas mundo):

$P_m = T_{cm} P$

Un punto P_m expresado en el sistema de coordenadas mundo puede expresarse en el sistema de coordenadas cámara si se multiplica por la matriz de transformaciones geométricas T_{cc} (la cual expresa las transformaciones geométricas del sistema de coordenadas mundo en relación al sistema de coordenadas cámara):

$P_c = T_{cc} P_m$

3.3. Transformación de proyección

Las imágenes se forman mediante una proyección que transforma un punto 3D del mundo a un punto 2D de la imagen. Cada punto P_c expresado en el sistema de coordenadas cámara, se proyecta según el modelo de la cámara y tiene su correspondiente en la imagen proyectada 2D.

Existen dos tipos de proyecciones más conocidas y utilizadas en informática gráfica: proyección ortogonal y proyección perspectiva. La proyección perspectiva puede describirse por un modelo matemático sencillo denominado “modelo de Pinhole” (figura 5) El centro de proyección c es el punto en el cual convergen los rayos de luz trazados desde cada punto P_c; el plano de proyección en el cual se forma la imagen, está situado a una cierta distancia focal f desde el centro de proyección. El rayo que une P_c con el centro de proyección

interseca al plano de proyección en el punto de la imagen p. El sistema de coordenadas de la cámara tiene como origen a c. El eje Z está alineado con el eje óptico y es perpendicular al plano de la imagen. En la figura 5 puede verse que la proyección de un punto O perteneciente al eje óptico se proyecta en el denominado centro de la imagen o.

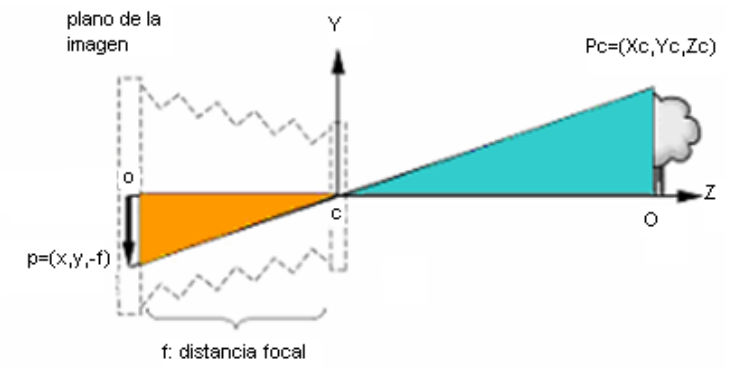


Figura 5. Modelo de Pinhole

Considerando que los triángulos Δcop y ΔcOP_c son similares, el mapeo de un punto del modelo P_c al punto p en el plano imagen situado en –f es:

$x = X_c \cdot f / Z_c$

$y = Y_c \cdot f / Z_c$

$z = -f$

En la bibliografía generalmente se toma una distancia f positiva, por esto y dado que el razonamiento anterior no varía, en adelante se considera:

$z = f$

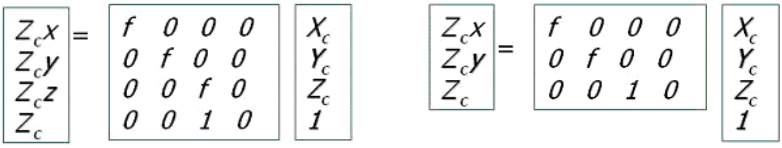
La proyección no es una transformación afín, pero pese a esto puede expresarse en coordenadas homogéneas como matriz de 4x4. Esto es una ventaja ya que resulta muy eficiente en la implementación utilizar la multiplicación de matrices al igual que se hizo con las transformaciones geométricas. Para llegar a una expresión matricial se expresan las ecuaciones anteriores de la siguiente forma:

$Z_c\, x = X_c\, f$

$Z_c\, y = Y_c\, f$

$Z_c\, z = Z_c$

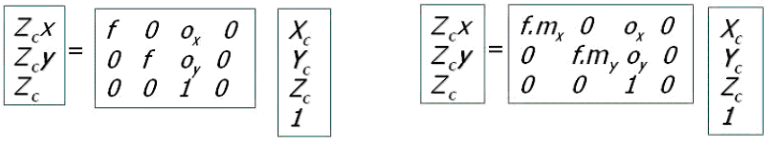
La forma matricial se muestra en la figura 6.a.



(a) (b)

Figura 6. Expresión matricial de la proyección perspectiva con parámetro f, mediante una matriz de 4x4 (a) y su forma más frecuente mediante una matriz de 3x4 (b)

Para obtener el punto p = (X_c f/Z_c, Y_c f/Z_c, f, 1), a continuación de la operación anterior debe dividirse por Z_c. Esta última operación es la que hace que la transformación de proyección sea no lineal.
Dado que el punto resultado es de una imagen puede ignorarse la componente Z que es para todos los puntos igual a la distancia focal f.
Por esto la proyección se escribe utilizando una matriz de 3 x 4 según muestra la figura 6.b.
Dado que el modelo de Pinhole considera solo el parámetro distancia focal f puede resultar demasiado sencillo en algunas aplicaciones. Dicho modelo puede completarse considerando otros parámetros de la cámara.
El origen de coordenadas en el plano de la imagen puede no estar en el centro de la misma. Sea o = (o_x,o_y) las coordenadas del punto principal, el punto p se calcula:
$x = X_c \cdot f / Z_c + o_x$
$y = Y_c \cdot f / Z_c + o_y$
Para llegar a la notación matricial las ecuaciones anteriores se reescriben de la siguiente forma:
$Z_c\, x = X_c \cdot f + Z_c\, o_x$
$Z_c\, y = Y_c \cdot f + Z_c\, o_y$
En forma matricial puede expresarse según muestra la figura 7.a.



(a) (b)

Figura 7. Expresión matricial de la proyección perspectiva con parámetros f y centro de la imagen o=(o_x, o_y) (a); agregando los parámetros de escala del píxel m_x y m_y (b)

Las cámaras CCD pueden producir píxeles no cuadrados. Esto puede modelarse considerando, además de la distancia focal y el centro de la imagen, diferentes escalas en los ejes X e Y de la imagen. Sea m_x y m_y el número de píxeles por unidad de distancia en coordenadas imagen en los ejes X e Y, el punto p se calcula:

$x = X_c \, . \, f \, . \, m_x / Z_c + o_x$

$y = Y_c \, . \, f \, . \, m_y / Z_c + o_y$

Para llegar a la notación matricial las ecuaciones anteriores se reescriben de la siguiente forma:

$Z_c \, x = X_c \, . \, f \, . \, m_x + o_x \, Z_c$

$Z_c \, y = Y_c \, . \, f \, . \, m_y + o_y \, Z_c$

En forma matricial se expresa según muestra la figura 7.b.

Según la calidad de la lente, la cámara puede producir distorsiones o deformaciones. Por ejemplo, la distorsión radial produce una curvatura de las rectas más notoria para los puntos más alejados del centro de la imagen. La distorsión radial puede se puede modelar con un factor no lineal. Denominando (x, y) a las coordenadas ideales sin distorsión, las coordenadas (x',y') con distorsión son:

$x' = x \, (1 + k_1 \, r^2 + k_2 \, r^4)$

$y' = y \, (1 + k_1 \, r^2 + k_2 \, r^4)$

donde $r = x^2 + y^2$, y k_1, k_2 son los parámetros de la distorsión.

A partir de unas coordenadas (x', y') distorsionadas, se puede aplicar la corrección obteniendo unas coordenadas ideales:

$x = x' / (1 + k_1 \, r^2 + k_2 \, r^4)$

$y = y' / (1 + k_1 \, r^2 + k_2 \, r^4)$

Los parámetros intrínsecos de una cámara, es decir aquellos inherentes a sus características internas, influyen en la proyección y por ende es necesario conocerlos. Como se verá en la sección 3.5 los parámetros intrínsecos se pueden estimar mediante un proceso denominado calibración.

3.4. Formación de la imagen sintética

Dado un punto P_m expresado en coordenadas mundo, se expresa en coordenadas cámara mediante transformaciones geométricas, y a continuación se proyecta para obtener el punto 2D de la imagen. Estas sucesivas transformaciones pueden expresarse como multiplicaciones de matrices sucesivas (figura 8).

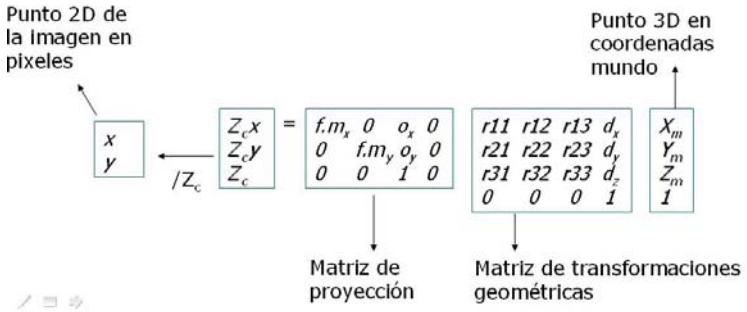


Figura 8.Formulación del proceso de formación de la imagen

En una aplicación de realidad aumentada se tiene una escena sintética modelada en un cierto sistema de coordenadas mundo. Existe un participante que visualiza la escena desde un punto de vista, el de la cámara con la que captura la escena real. A medida que el participante se mueve en el mundo real se debe producir un cambio en el punto de vista del mundo virtual de acuerdo al cambio de posición y orientación del participante. Por esto, para poder combinar los objetos virtuales con la realidad de forma coherente (imágenes registradas) se debe conocer en cada momento la relación entre el sistema de coordenadas mundo y el sistema de coordenadas cámara. Esto significa conocer la matriz de transformaciones geométricas del sistema de coordenadas del mundo con respecto a la cámara, o de forma equivalente la matriz de transformaciones de la cámara en relación al sistema de coordenadas mundo. Pese a que la matriz de transformaciones tiene 12 entradas a estimar, se debe notar que la traslación y la orientación de la cámara se determinan con 6 parámetros o 6 grados de libertad (DOF) –3 para la traslación en cada eje T_x, T_y, T_z; y 3 para los ángulos de rotación con respecto a cada eje μ_x, μ_y, μ_z– denominados parámetros extrínsecos.

Para poder obtener la imagen sintética debe conocerse además la matriz de proyección. A diferencia de la matriz de transformaciones geométricas la matriz de proyección no cambia mientras la cámara se mueve dado que está basada en características intrínsecas de la misma. Por esto se calcula una sola vez con un procedimiento a priori denominado calibración.

Conocidas la matriz de transformaciones geométricas y la matriz de proyección puede obtenerse la proyección de la escena virtual aplicando las operaciones de la figura 4. Dicha proyección es la imagen sintética que debe combinarse con la realidad. Dado que la matriz de transformaciones cambia a medida que el participante se mueve este cálculo debe realizarse en cada momento. Para producir una animación en tiempo real este proceso debería realizarse cerca de

30 veces por segundo o cuadros por segundo –en inglés, fps–. Esta tasa se considera óptima aunque en la práctica se obtienen resultados aceptables por debajo de la misma.

3.5. Calibración de cámara

Según el modelo matemático elegido, la cámara se describe mediante una serie de parámetros. Se denominan parámetros intrínsecos aquellos que describen la geometría y óptica del conjunto cámara y tarjeta de adquisición de imágenes. Estos parámetros afectan al proceso que sufre un rayo luminoso desde que alcanza la lente del objetivo, impresiona el elemento sensible y se convierte en píxel iluminado. Por otra parte, los parámetros extrínsecos describen la posición y orientación de la cámara en un sistema de coordenadas mundo.
Generalmente el conjunto cámara y tarjeta de adquisición se describe mediante los siguientes parámetros intrínsecos:

- Distancia focal: f
- Centro de la imagen: o_x, o_y
- Relación de aspecto del píxel: $\alpha = m_x/m_y$
- Distorsiones: k_1, k_2

Si se requiere determinar la posición de una cámara durante el proceso de calibración pueden estimarse parámetros extrínsecos:

- Traslación: T_x, T_y, T_z
- Rotación: μ_x, μ_y, μ_z

Según la definición de Tsai (1987) [50] la calibración es el proceso mediante el cual se calculan los parámetros intrínsecos y/o extrínsecos de la cámara, a partir de un conjunto de puntos de control, conocidas las coordenadas 3D de esos puntos y midiendo las correspondientes coordenadas 2D en una imagen obtenida con dicha cámara.
Según la aplicación, el proceso de calibración se realiza para conocer solamente los parámetros intrínsecos o los parámetros extrínsecos, o ambos tipos. En particular en las aplicaciones de realidad aumentada se necesita realizar la calibración de la cámara utilizada para conocer los parámetros intrínsecos, y de esta forma definir la matriz de proyección y en caso de ser necesario corregir las imágenes capturadas compensando distorsión que produce la cámara.
Los denominados puntos de control consisten en una plantilla de puntos de los cuales se conocen las coordenadas 3D en un sistema de coordenadas mundo local. Por ejemplo, puede usarse un tablero de ajedrez, los puntos de una cuadrícula, etc. La figura 9 muestra algunos ejemplos de posibles puntos de control.

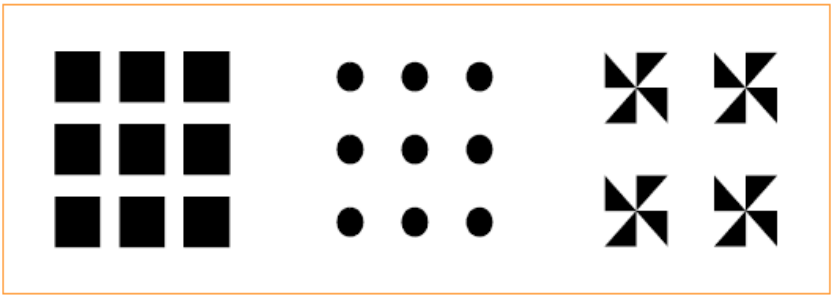


Figura 9. Diferentes configuraciones posibles de puntos de control a utilizar para realizar la calibración

Para un algoritmo detallado de calibración puede consultarse cualquier libro de visión 3D como por ejemplo el de Trucco y Verri (1998) [49] o el de Cyganek y Siebert (2009) [14].
El primer paso de un proceso de calibración es capturar diferentes imágenes de los puntos de control con la cámara en distintas posiciones y ángulos.
Como se dijo en la definición de calibración, se conocen: las coordenadas 3D en un sistema de coordenadas mundo de los diferentes puntos de control Pi_m.
Analizando con técnicas de visión las diversas imágenes capturadas de los puntos de control, pueden extraerse en cada una de ellas las coordenadas 2D de los correspondientes puntos de control pi.
Los puntos Pi_m en coordenadas mundo y sus correspondientes proyecciones pi se relacionan mediante la ecuación del proceso de formación de imágenes (figura 8).
Existen diversos métodos de resolución de las ecuaciones. Tres de los métodos más conocidos están disponibles como implementaciones públicas: el algoritmo de Tsai (1987) [50]; el algoritmo de Zhang (2000) [62] y algoritmo de Heikkilä (2000) [25]. Pueden encontrarse implementaciones para Matlab [12] y versiones en C dentro de la Open Source Computer Vision Library [37].

4. Tracking para aplicaciones de realidad aumentada

4.1. Introducción

Como se definió en el capítulo 1, el tracking es proceso de seguimiento de un objeto en movimiento, es decir, la estimación de la posición y orientación del mismo en cada instante.

En una aplicación de realidad aumentada se necesita el tracking del participante para conocer la matriz de transformaciones geométricas y así realizar el registro de imágenes sintéticas y reales.
En la mayoría de aplicaciones de realidad aumentada se trata de realizar un tracking de la cámara que captura la escena. Aunque también puede tratarse del tracking de la cabeza del usuario o de algún objeto manipulado por este.
El tracking completo estima los 6 parámetros o grados de libertad (DOF) –3 para la traslación en cada eje T_x,T_y,T_z; y 3 para los ángulos de rotación con respecto a cada eje μ_x, μ_y, μ_z–.
Según la aplicación puede interesar conocer:
- La posición y orientación de la cámara en un sistema de coordenadas global
- La posición y orientación de la cámara en relación a un objeto de la escena real en cuestión
- La posición y orientación de la cámara en relación a la posición y orientación del cuadro de video anterior

El tracking en una aplicación de realidad aumentada puede hacerse:
- Mediante dispositivos físicos específicos (sección 4.2.)
- Mediante el análisis de la imagen capturada, denominado tracking basado en visión (sección 4.3.)
- Tracking híbrido, que combina las salidas de dispositivos físicos con el análisis de la imagen (sección 4.4.)

Según Zhou et al (2008)[64], que presenta una recopilación de las publicaciones realizadas en el período 1998-2008 en eventos internacionales relacionadas con la investigación científica en el área de realidad aumentada como el actual *ISMAR* (*Internacional Symposium on Mixed and Augmented Reality*)[4] y los anteriores *ISAR*, *ISMR* e *IWAR*, el tracking ha sido el tópico más popular en la investigación, siendo los artículos de tracking los más citados en la bibliografía. En dicha publicación puede encontrarse la evolución de la investigación en este campo.
En la sección 4.5 se presenta un resumen de las diferentes técnicas de tracking.

4.2. Tracking mediante dispositivos físicos

Según Zhou et al (2008) [64], el tracking basado puramente en sensores fue utilizado en las primeras aplicaciones de realidad aumentada, encontrándose muy pocas publicaciones recientes que no utilicen combinaciones con tracking basado en visión.

[4] http://ismar.net.

Según la tecnología el tracking puede realizarse utilizando sensores mecánicos, magnéticos, ultrasónicos, inerciales u ópticos. Cada uno de estos dispositivos presenta sus ventajas y desventajas (para más detalles consultar capítulo 1).
Actualmente pueden encontrarse en el mercado teléfonos móviles con dispositivos integrados cuyo uso resulta muy apropiado para aplicaciones de realidad aumentada móvil. A continuación se detallan los dispositivos más comunes que pueden usarse para tracking en una aplicación de realidad aumentada en ambiente externo:

- GPS (Global Positional System): nos da la latitud y longitud en el sistema de coordenadas global que puede usarse para consultar un Sistema de Información Geográfica (SIG). El GPS brinda los 3 DOF de la traslación. Para mejorar la exactitud del valor que brinda puede usarse lo que se denomina GPS diferencial, donde una estación base de la cual se conoce su localización con exactitud computa y transmite el error introducido –por ejemplo por los efectos atmosféricos– el cual es utilizado por el receptor GPS para corregir su posición. Para obtener los 6 DOF se necesita combinarlo con otro dispositivo que brinde los 3 DOF de la rotación como una brújula.

- Brújula digital: al igual que una brújula convencional brinda la orientación en un sistema global (3 DOF). Las brújulas que se encuentran en los teléfonos móviles son denominadas brújulas de estado sólido, generalmente construidas mediante sensores de campo magnético que envían señales a un microprocesador. Generalmente se combina con el GPS para obtener los 6 DOF del movimiento del dispositivo. La ventaja con respecto al uso de sensores inerciales es que brindan un resultado con error constante, el cual puede precalibrarse durante la instalación del sistema.

- Sensores inerciales: los acelerómetros y giroscopios permiten conocer aceleración y velocidad de rotación, a partir de las cuales puede conocerse los 6 DOF de la pose. Las ventajas de este tipo de dispositivos radica en su rapidez y su buena respuesta a cambios bruscos. Sin embargo, su principal desventaja suele ser la acumulación de errores debido al ruido, y por esto cada cierto tiempo deben recalibrase. Algunos modelos de teléfonos móviles de última generación poseen integrados acelerómetros y giroscopios con tecnología *MEMS* (*MicroElectroMechanical Systems*).

4.3. Tracking basado en visión

4.3.1. Flujo de trabajo de la aplicación

En lugar de utilizar dispositivos físicos específicos para tracking, la posición y orientación de la cámara en la escena real puede estimarse analizando el video capturado por una cámara utilizando técnicas de visión por computador. En resumen se trata de buscar en la imagen elementos de la escena real que permitan deducir la posición y orientación de la cámara en relación a una cierta referencia –o de forma equivalente la posición y orientación de los objetos en relación a la cámara–. Según la aplicación puede tratarse de una cámara integrada al participante en movimiento, o puede tratarse de una cámara fija en una escena con objetos en movimiento.
En dispositivos de visualización video *see-through* (aquí incluimos también al monitor y los dispositivos *hand-held*), el video capturado por la cámara se usa simultáneamente:

- Como el fondo (video background) de la escena que ve el usuario en la pantalla
- Para realizar el tracking

El flujo de trabajo de una aplicación de RA con tracking basado en visión es el siguiente:

a. Preprocesamiento:
    a.1. Calibración de la cámara
    a.2. Modelado de la escena virtual
b. Procesamiento:
    b.1. Inicializaciones:
        b.1.1. Definir matriz de proyección.
        b.1.2. Carga de la escena virtual
    b.2. Procesamiento de cada cuadro:
        b.2.1. Captura de cuadro de video
        b.2.2. Estimación de la posición y orientación de la cámara
        b.2.3. Visualización de la escena aumentada

La etapa de preprocesamiento se realiza *off-line* y consiste en dos partes bien diferenciadas e independientes: por una parte la calibración de la cámara (paso a.1) y por otra el modelado de la escena virtual (paso a.2).
El proceso de calibración (paso a.1) (ver sección 3.5) permite estimar los parámetros intrínsecos de la cámara que captura la escena.

Conocidos ciertos parámetros –como la distancia focal, la relación de aspecto del píxel, el centro de la imagen– puede definirse la matriz de proyección a utilizar para la formación de la imagen aumentada, de forma que la cámara virtual tenga las mismas características que la cámara real. Además mediante la calibración puede conocerse la distorsión que realiza la cámara en las imágenes, y de esta forma puede realizarse un procedimiento de anulación de la distorsión si así se desea. En algunas aplicaciones de realidad aumentada donde la precisión de la visualización aumentada no es crítica, como por ejemplo aplicaciones web o juegos, el proceso de calibración a veces se omite. En este caso se utilizan parámetros por defecto, y por ende el resultado puede tener sus deficiencias. En resumen, el proceso de calibración es un proceso que se realiza una sola vez por cada cámara utilizada y permite agregar precisión al resultado final. Independientemente de haberse realizado o no la calibración, tanto con parámetros estimados como con parámetros por defecto, durante la inicialización de la aplicación (paso b.1.1) se define la matriz de proyección que se utilizará para la visualización de la escena aumentada (paso b.2.3).

Dado que se quiere aumentar la escena real con objetos virtuales se debe modelar previamente una escena 3D (paso a.2) utilizando para ello un editor de gráficos 3D. Mediante este software se generará un fichero con algún formato adecuado de gráficos 3D, como por ejemplo VRML, X3D[5], etc. Como parte de la inicialización de la aplicación se realizará la carga de este archivo conforme a las librerías gráficas que se utilicen en la aplicación (paso b.1.2). Pueden encontrarse algunos ejemplos de aplicaciones que, dado que utilizan escenas sencillas estas se generan directamente desde la aplicación mediante la invocación directa de las librerías gráficas.

El proceso de la aplicación en tiempo real es un ciclo donde se capturan cuadros de video (paso b.2.1), se analiza para estimar la posición y orientación de la cámara con respecto a la escena real (paso b.2.2) y se visualiza (paso b.2.3) la escena aumentada.

La parte específica del tracking basado en visión es la estimación de la posición y orientación de la cámara, o de forma complementaria la estimación de la posición y orientación de los objetos de la escena en relación a la cámara. Como resultado de esta estimación se define la matriz de transformaciones geométricas que se aplicará a la escena virtual en la visualización.

La visualización de la imagen aumentada consiste en el video capturado de fondo con la proyección de la escena virtual superpuesta.

[5] http://www.web3d.org/x3d.

Existen diferentes formas de implementar el fondo: en caso de tener acceso directo al buffer de la imagen –cuando se realiza *rendering* por *software*– el video se copia antes del renderizado de la escena virtual. En el caso de no tener acceso al buffer –cuando se realiza *rendering* por *hardware*– se debe cargar la imagen como una textura aplicada a un polígono que emula una pantalla. Para que la pantalla no sufra deformaciones debido a la proyección perspectiva se renderiza primero utilizando una proyección ortogonal, y a continuación se proyecta el resto de la escena utilizando la matriz de proyección definida en la inicialización y la matriz de transformaciones geométricas estimada en el paso anterior (se aplica las ecuaciones de formación de imagen de la figura 8).

4.3.2. Clasificación de los algoritmos de tracking basado en visión

Existen diferentes estrategias usadas para tracking basado en visión, que pueden clasificarse en dos grandes ramas:
- Tracking de marcadores
- Tracking basado en características naturales

El tracking de marcadores fue la primera estrategia utilizada y consiste en introducir uno o más marcadores conocidos en la escena real para superponer objetos virtuales o utilizarlos como interfaz tangible. Esta estrategia tuvo mucho auge desde la publicación de Kato y Billinghurst (1999)[29] –según Zhou et al (2008) [64] uno de los artículos más referenciados en realidad aumentada– y la disponibilidad del código abierto de su desarrollo ARToolkit [7]. Es muy usada actualmente, hecho fácilmente comprobable navegando en la web donde pueden encontrarse infinidad de aplicaciones de publicidad. Zhang et al. (2002) [63] presenta un estudio comparativo de diferentes algoritmos de tracking de marcadores, y según Zhou et al (2008) [64] desde la fecha no existen investigaciones significativas en esta área. Dado que puede resultar molesto o indeseable el agregar marcadores visibles a la escena recientemente la investigación se derivó a encontrar otros algoritmos de tracking basado en visión que exploren la presencia de características naturales en la misma (puntos, líneas, bordes, texturas) evitando la introducción de marcadores. Zhou et al (2008) [64] afirma que hasta la fecha de su publicación esta ha sido el área más activa en la investigación de tracking basado en visión, y se podría afirmar que actualmente lo sigue siendo.
Entre los algoritmos que exploran las características naturales de la escena existe la siguiente gran división:
- Tracking con conocimiento de la escena
- Tracking sin conocimiento de la escena

En las siguientes secciones se detalla cada uno de los enfoques de tracking basado en visión.

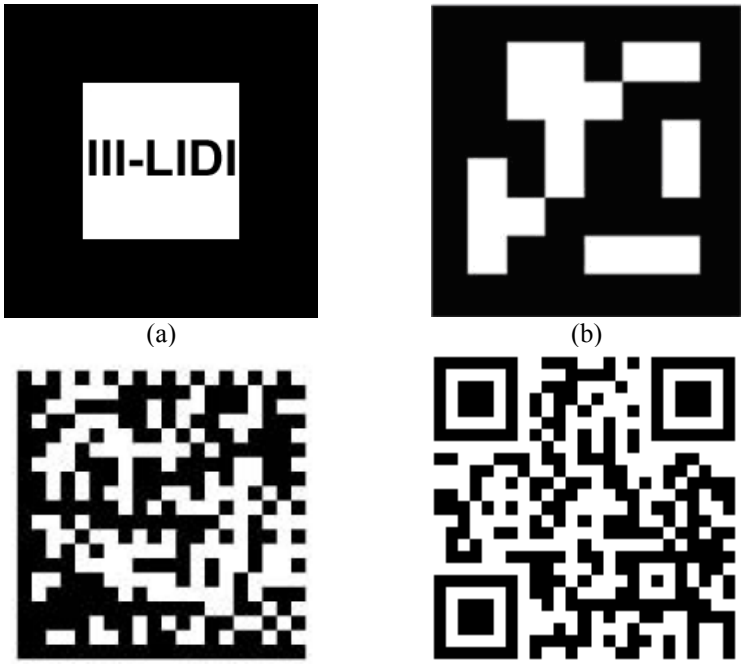
4.3.3. Tracking de marcadores

Un marcador, denominado en inglés "fiducial markers", es una imagen 2D impresa con un formato específico conocido por la aplicación de tracking.
En la figura 10 pueden verse diferentes tipos de marcadores de los que existen lectores de código abierto.
Los marcadores *template* son muy conocidos dado que son los utilizados por la librería de realidad aumentada ARToolkit [7], que fue la primera librería que popularizó las aplicaciones de realidad aumentada. El formato es un cuadrado negro y dentro un cuadrado blanco que tiene dentro una imagen asimétrica en negro. Como se ve en la figura 10.a, donde se visualiza la sigla del Instituto de Investigación y Desarrollo en Informática III-LIDI de la UNLP, es posible crear nuevos marcadores respetando este formato y entrenando previamente a la aplicación para incorporar el marcador a la base de datos de marcadores conocidos.
Los marcadores "ID-marker" (figura 10.b) codifican un número de 9-bits (hasta 512 diferentes) en un patrón de 6 x 6, repitiendo los 9 bits 4 veces completando los 36 bits. Una variante de estos marcadores son los denominados BCH (Bose, Ray-Chaudhuri, Hocquenghem), los cuales son más robustos que los anteriormente descriptos, ya que usa un algoritmo avanzado de chequeos de redundancia cíclica (CRC) que permite restaurar marcadores dañados. Se incrementa el número de marcadores disponibles a un total de 4096. Este tipo de marcadores o una variante de los mismos son los utilizados por las librerías de realidad aumentada ARTag [6] y ARToolkitPlus [8].
Los marcadores "DataMatrix" y marcadores "QRCode" (figuras 9.b y 9.c respectivamente) no fueron diseñados específicamente para realidad aumentada, sino que su propósito inicial es codificar una serie de caracteres ASCII. Uno de los usos más comunes es la codificación de una URL de forma que una aplicación al leerlos y decodificarlos pueda derivar al sitio web codificado. Por esto su uso principal se asocia a los hipervínculos y no a realidad aumentada. Mientras que los *DataMatrix* pueden almancenar hasta 2335 caracteres, los *QRCode* almacenan 4296 caracteres. Una diferencia entre ellos radica que el *QRCode* incluye además símbolos japoneses. Ambos códigos son abiertos y pueden descargarse de forma gratuita aplicaciones lectoras para los diversos teléfonos celulares del mercado. Desde un punto de vista amplio algunos autores los consideran una forma de realidad aumentada. Además, existen algunas aplicaciones de realidad aumentada que utilizan dichos códigos

no solo con el propósito mencionado sino como marcadores para tracking.
Según la aplicación podrá tratarse de una cámara fija con un marcador en movimiento, o de un marcador fijo y una cámara en movimiento. En cualquiera de los casos el cálculo no varía ya que se trata de una posición relativa entre la cámara y el marcador.
Básicamente el algoritmo de tracking basado en marcadores consta de dos pasos principales:

- Detección e identificación de marcadores
- Estimación de la matriz de transformaciones geométricas de cada marcador detectado

Para un estudio comparativo de tracking de marcadores consultar la publicación de Zhang (2002) [63].
En cada cuadro capturado se aplica un algoritmo de detección de marcadores seguido de una identificación de cada marcador. En esta fase puede ser necesario compensar la distorsión de la imagen capturada.



(a) (b)

(c) (d)

Figura 10. Diferentes tipos de marcadores: *template* (a); *ID-Marker* (b); *DataMatrix* (c), *QRCode* (d) los dos últimos codifican el texto "weblidi.info.unlp.edu.ar"

A continuación se ejemplifica la localización de marcadores en la librería ARToolkit [7], que se realiza con los siguientes pasos:

1. Umbralización (*thresholding*) y detección de bordes.
2. Búsqueda de cuadriláteros: Se extraen las regiones cuyo contorno son 4 segmentos. Las regiones muy grandes o pequeñas se rechazan. Se extraen los parámetros de los 4 segmentos y las coordenadas de los 4 vértices.
3. Se normaliza el interior de cada marcador usando una transformación perspectiva que lo convierte en un cuadrado.
4. Las imágenes normalizadas se chequean contra una serie de patrones conocidos. Se realizan cuatro comparaciones por correlación con cada marcador de la base de datos para considerar las posibles rotaciones de 90º.

Si se detectaron marcadores, se aplica para cada uno un algoritmo de estimación de la matriz de transformaciones geométricas que expresa la traslación y orientación del marcador con respecto al sistema de coordenadas cámara.
En la figura 11 se señalan los datos conocidos y la incógnita en la formulación del proceso de formación de imágenes (figura 8).
El marcador es un objeto conocido por construcción, es decir se conocen diferentes puntos Pi_m del marcador en un sistema de coordenadas 3D local al marcador (figura 12)
Dada la imagen del video capturado, pueden obtenerse por medio de técnicas de visión por computador los puntos 2D pi de la imagen del marcador correspondientes a los puntos 3D conocidos.
Mediante la calibración de la cámara, se conocen los parámetros intrínsecos de la cámara, por ende se conoce la matriz de proyección. Además se conoce la distorsión que tiene la imagen capturada y por ende puede corregirse para obtener la imagen ideal.

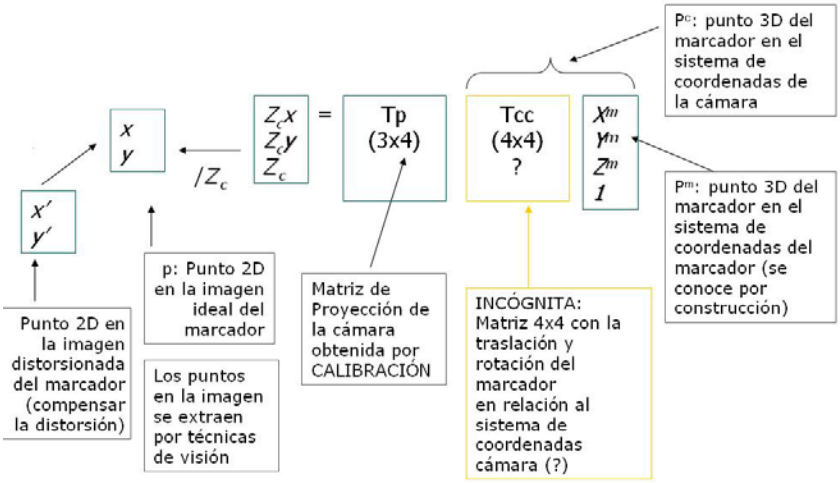


Figura 11. Formulación del proceso de formación de la imagen en una escena con marcadores, indicando datos conocidos e incógnita

Figura 12. Sistema de coordenadas local al marcador utilizado por ARToolkit

A partir del conocimiento de los diversos pares de puntos correspondientes 3D-2D se puede estimar la posición y orientación del marcador en relación a la cámara que capturó la imagen (6 grados de libertad o DOF).
Para estimar la matriz T_{cc} puede realizarse el proceso iterativo de optimización de Newton el cual que minimiza una función de error. El algoritmo se describe de la siguiente forma (para más detalle consultar [49]):

1. Inicializar la matriz de transformación T_{cc}
2. Usar la estimación actual T_{cc} para calcular las proyecciones qi de cada vértice Pi del marcador, con i de 1 a 4
3. Calcular el error entre los puntos detectados en la imagen $pi=(pi_x,pi_y)$ y las proyecciones $qi=(qi_x,qi_y)$, con i de 1 a 4, de la siguiente forma:

$$¼ \sum_{i=1..4} \text{distancia}(pi,qi)$$

Donde

$$\text{distancia}(pi,qi) = (pi_x - qi_x)^2 + (pi_y - qi_y)^2$$

4. Calcular las correcciones ΔT_x, ΔT_y, ΔT_z de los parámetros de traslación y las correcciones $\Delta \mu_x$, $\Delta \mu_y$, $\Delta \mu_z$ para los ángulos de rotación con respecto a los ejes principales. Con estas correcciones actualizar la matriz de transformaciones geométricas Tcc
5. Repetir los pasos 2 a 4 hasta converger, es decir, obtener un error debajo de un cierto límite inferior.

La ventaja del tracking de marcadores es que resulta robusto y computacionalmente eficiente. La desventaja más obvia es la presencia de marcadores visibles que interfieren con la escena real. Otra de las desventajas de algunos algoritmos de tracking de marcadores, como por ejemplo el que implementa ARToolkit, es que exige la visibilidad total del marcador en cada cuadro. Si en determinado cuadro el marcador no ha sido capturado en su totalidad el tracking se pierde, y en consecuencia la escena virtual no se visualiza. Wagner et al. (2008) [54] presenta dos técnicas de tracking de marcadores incrementales, basadas en el seguimiento de características y en el flujo de píxeles, que se aplican cuando los marcadores se pierden o son tapados parcialmente para evitar perder el tracking.
Como se verá en la siguiente sección estos dos problemas se solucionan aplicando las técnicas de tracking basadas en características naturales.

4.3.4. Tracking basado en características naturales con conocimiento de la escena

Con el objetivo de evitar invadir una escena con marcadores surgieron las técnicas de tracking basado en características naturales, es decir basadas en la localización de características como puntos, líneas, bordes o texturas.
Dentro de los diferentes enfoques basados en características naturales se pueden distinguir:
- Métodos basados en bordes
- Métodos basados en texturas

Los dos enfoques anteriores relacionan puntos de cada cuadro de video con el cuadro anterior, y dado que el punto de vista de la cámara varía ligeramente de un cuadro a otro, se dice que son técnicas de

correspondencia de desplazamiento de cámara corto o en inglés "short-baseline-matching".
Un enfoque diferente también basado en características naturales son los denominados:

- Métodos basados en detección

A diferencia de los enfoques anteriores que exploran las relaciones entre cuadros vecinos, este tipo de técnicas relaciona las características de un cierto cuadro con cuadros almacenados en una base de datos que corresponden a puntos de vista diferentes. Estas son técnicas se denominan de correspondencia de desplazamiento de cámara ancho, o en ingés "wide-baseline-matching".
Generalmente todos estos métodos tienen en común que se debe tener algún tipo de conocimiento del objeto o escena real. Por esto se denominan métodos basados en el modelo, o en inglés "model-based".
A continuación se resumen cada uno de los diferentes estos enfoques. Para una descripción más detallada y una ampliación de bibliografía se recomienda consultar a Fua y Lepetit (2010) [22], los cuales discuten las ventajas y desventajas de este tipo de técnicas de tracking que pertenecen a un área de investigación muy activa actualmente.

Métodos basados en bordes

Los métodos de tracking basados en bordes fueron los primeros enfoques ya que son eficientes y fáciles de implementar. *RAPiD* (*Real-time Attitude and Position Determination*), presentado por Harris (1992), fue uno de los primeros trackers en tiempo real, y desde esa fecha se han desarrollado diversas mejoras para hacerlo más robusto. La idea básica de este algoritmo es considerar un conjunto de puntos de control 3D en el objeto a seguir que pertenezcan a bordes del objeto. El movimiento 3D del objeto entre dos cuadros consecutivos puede recuperarse a partir del desplazamiento 2D de los puntos de control en las imágenes.
El algoritmo exige una inicialización donde se registra el modelo con la imagen para comenzar con un valor válido de la matriz de transformaciones geométricas T^0.
En cada cuadro, se realiza lo siguiente:

1. Predecir la matriz de transformaciones geométricas T^j que representa la posición y orientación de los puntos de control P_i del objeto en relación a la cámara, a partir de la matriz de transformaciones geométricas del cuadro anterior T^{j-1}.
2. Usar la estimación actual T^j para calcular las proyecciones q_i de cada punto de control P_i .

3. Localizar en la imagen los puntos de control efectivos p_i (suelen ser parámetros que definen los bordes).
4. Calcular el error que existe entre los puntos de control efectivos y los puntos de control proyectados (error=distancia (q,p)).
5. Ajustar la posición y orientación predichas resolviendo un sistema lineal minimizando las distancias al cuadrado entre los puntos de control efectivos y los puntos de control proyectados.

Un problema de este algoritmo es que pueden existir bordes, y en consecuencia puntos de control, extraídos erróneamente. Por esto se desarrollaron diversas modificaciones del algoritmo original para agregar robustez a RAPID. Una de las primeras mejoras fue introducida por Armstrong y Zisserman (1995) [5] quienes utilizan la metodología conocida como *RANSAC* (*RANdom SAMple Consensus*), el cual es un método iterativo que se usa para estimar parámetros de un modelo matemático a partir de un conjunto de muestras que contienen datos erróneos que se alejan del conjunto (en inglés "outliers") Otras mejoras que pueden consultarse son las introducidas por Drummond y Cipolla (2002) [17] y Vaccetti et al (2004) [51].
Los algoritmos basados en bordes son eficientes y relativamente fáciles de implementar, y resisten frente a cambios de iluminación.

Métodos de tracking basados en texturas

Si se cuenta con escenas donde el o los objetos a seguir están suficientemente texturados puede extraerse información a partir de la textura de los mismos. La información puede derivarse del flujo óptico o de correspondencias entre puntos de interés. Los métodos basados en flujo óptico tratan de estimar el movimiento entre dos cuadros de video consecutivos. Por otra parte, los métodos basados en correspondencias entre puntos de interés, calcula el movimiento de la cámara mediante una minimización de cuadrados mínimos o mediante estimación robusta a partir de los pares de puntos correspondientes en cuadros sucesivos. Estos últimos son relativamente más insensibles a oclusiones parciales, a cambios de iluminación, fondos complejos, cambios de escala y puntos de vista.
Estos métodos seleccionan puntos de interés o puntos clave mediante un "operador de interés". Los puntos de interés deben cumplir una serie de condiciones:
- Los vecinos deben ser diferentes (por esto los puntos de bordes quedan excluidos).
- Los puntos de patrones repetitivos deben rechazarse.

- La selección debe repetirse, es decir, en varias vistas de la escena debe seleccionarse el mismo punto de interés independientemente de la perspectiva o el ruido.
- La extracción de puntos de interés debe ser insensible a cambios de escala, puntos de vista e iluminación.

Dados dos cuadros de video con puntos de vista similares, un procedimiento clásico para este tipo de tracking consiste en:

1. Seleccionar puntos de interés en la primera imagen, de la cual la posición y orientación de la cámara ya ha sido estimada previamente.
2. Para cada punto seleccionado en la primera imagen, buscar el punto correspondiente en una región alrededor del punto en la segunda imagen.
3. A partir de las correspondencias encontradas deducir el cambio de posición y orientación en 3D a partir de sus distancias en la imagen.

Si las coordenadas 3D de los puntos de interés no se conoce a priori para ningún cuadro los métodos están sujetos a errores que producen fallas de tracking. Una solución consiste en inicializar "a mano" comenzando con uno o varios cuadros clave donde se hacen corresponder los puntos en el modelo 3D con los puntos correspondientes en cada imagen. En tiempo real, las nuevas imágenes pueden compararse con los cuadros clave para proveer una posición y orientación en base la posición y orientación de los mismos. Genc et al (2002) [23] utiliza esta solución, que tiene la desventaja de que al relacionar cuadros no consecutivos puede haber una la gran diferencia en los puntos de vista de los mismos ("wide-baseline-matching"). Vacchetti et al (2004) [52] realiza un entrenamiento donde se extraen puntos de interés de cada cuadro clave, computa las posiciones 3D de estos puntos de interés. En tiempo real, para cada nueva imagen el sistema escoge el cuadro clave cuyo punto de vista es más cercano al punto de vista del último cuadro procesado. Utilizando este cuadro clave se realiza una estimación de la posición y orientación de la cámara minimizando los errores de reproyección de los puntos de interés. El problema de "wide-baseline-matching" es que el resultado es menos exacto que el obtenido cuando se aplica correlación entre cuadros vecinos ("short-baseline-matching"). Otro problema es que puede acarrear una inconsistencia temporal ya que cada cuadro se computa independientemente del anterior. En Vacchetti et al (2004) [52] la inconsistencia temporal trata de evitarse realizando una combinación entre la solución obtenida con los cuadros clave y la obtenida en los cuadros precedentes.

Métodos de tracking basados en detección

En visión por computador, el problema de detección se enfoca en la localización de un cierto objeto en una imagen comparándolo con los objetos conocidos de una base de datos. Existen algoritmos que solucionan a la vez tanto el problema de detección como la estimación de la posición y orientación 3D del objeto detectado. Por tanto se dice que el tracking se realiza mediante la detección.
Cada punto de interés se describe mediante un vector de características denominado descriptor local ya que caracteriza la vecindad del punto.
Los métodos de tracking basados en la detección en base a características constan de los siguientes pasos:

a. Preprocesamiento off-line o entrenamiento:
   Se tienen imágenes de entrenamiento. Por cada imagen de entrenamiento:
   a.1. Localizar puntos de interés en la imagen de entrenamiento
   a.2. Calcular el vector de características de cada punto de interés
   a.3. Vincular cada punto de interés en la imagen con su correspondiente en el modelo 3D
   a.4. Almacenar en la base de datos el vector de características con la ubicación 3D

b. Procesamiento:
   Por cada cuadro del video capturado:
   b.1. Localizar los puntos de interés en la imagen
   b.2. Describir cada punto de interés mediante un vector de características
   b.3. Hacer corresponder cada vector de características con los vectores almacenados en la base de datos.
   b.4. Estimar la posición y orientación a partir de las correspondencias más cercanas.

Según la implementación será el descriptor utilizado para cada punto de interés. También según la implementación variará la forma de vincular cada punto de interés con su correspondiente en el modelo, que en su forma más sencilla puede realizarse manualmente.

Entre los descriptores de puntos de interés más eficientes se encuentra el SIFT (Scale-invariant feature transform) introducido por Lowe (2004) [34] y utilizado para tracking por Skrypnyk & Lowe (2004) [45]. Este descriptor es invariante a cambios de escala, orientación, distorsión y parcialmente

invariante a cambios de iluminación. Además permite identificar objetos con oclusiones parciales.

El vector de características SIFT se calcula de la siguiente manera:

1. Dividir la vecindad o región del punto de interés en subregiones (3x3 o 4x4), cada una define una parte del vector de características. En cada subregión calcular el histograma del gradiente en 4 u 8 orientaciones diferentes y tomar como características el pico de cada histograma. Por ejemplo: computando el histograma de 8 orientaciones en 4x4 subregiones da un vector de 128 características
2. Normalizar el vector de características para reducir los efectos de cambios de iluminación

Veamos a continuación como se implementan en el algoritmo de tracking presentado por Skrypnyk y Lowe (2004) [45] algunas partes del esquema general de tracking basados en la detección en base a características.
La localización de puntos de interés (utilizada en a.1 y b.1) se realiza de la siguiente forma:

1. Obtener la imagen a diferentes resoluciones filtrando la misma con Gaussianas a diferentes escalas.
2. Para aproximar el operador Laplaciano (derivada segunda) calcular una pirámide de Diferencias de Gaussianas (DoG).
3. Localizar los máximos y mínimos en las pirámides DoG como puntos de interés

La correspondencia del vector de características con los vectores de características de la base de datos (b.3) intenta identificar el vecino más cercano. El problema radica en el gran tamaño de la base de datos. Para optimizar la búsqueda se utiliza un algoritmo denominado BBF (*Best-bin-first*) basado en el algoritmo de búsqueda *k-d tree* modificado.
Como resultado del mapeo entre puntos de interés con la base de datos se tienen correspondencias 2D-3D. Las correspondencias 2D-3D se usan para estimar la posición y orientación de la cámara vía un enfoque robusto basado en RANSAC.
Pueden encontrarse en internet implementaciones del descriptor-detector SIFT como por ejemplo la realizada por Hess [26] que está implementada en C utilizando la librería OpenCV [37].
Posteriormente a la presentación del descriptor SIFT, Bay et al (2008) [10] presentan un descriptor de características denominado SURF (Speeded Up Robust Features) que está inspirado en el descriptor SIFT y se presenta como más robusto que éste frente a cambios de transformaciones y también más rápido. SURF usa como característica básica sumas de respuestas a la transformada *Wavelet* de Haar. Evans

(2009) presenta en [18] los detalles del algoritmo descriptor-detector SURF así como detalles de implementación de la librería de código abierto OpenSURF. Para una lista de vínculos a implementaciones existentes consultar [60].

4.3.5. Tracking sin conocimiento de la escena

Los enfoques descritos en la sección anterior son enfoques basados en el conocimiento del modelo. Un problema de gran interés reciente en aplicaciones de realidad aumentada es realizar el tracking en escenas desconocidas total o parcialmente. Este es un problema más complejo ya que no solo debe estimarse la posición de la cámara sino que también debe realizarse un mapa del ambiente desconocido.
Los algoritmos utilizados para tracking en escenas desconocidas se basan en un algoritmo denominado SLAM (*Simultaneous Localisation and Mapping*) que fue desarrollado por Thrun et al (2004) [48] en la comunidad de robótica. Este algoritmo no solo deduce la estructura del ambiente sino que al mismo tiempo establece una correlación de la misma con la posición y orientación de la cámara. Reitmayr et al (2010) [40] presenta posibles usos de SLAM en aplicaciones de realidad aumentada.
Tradicionalmente este problema se resuelve usando técnicas de minimización no lineal que son muy exactas pero difícilmente implementables en tiempo real.
Davison et al (2007) [16] presenta una implementación en tiempo real de SLAM, que se ejecuta a 30Hz en una Pentium laptop y una cámara convencional, y ejemplifica su aplicación a realidad aumentada.
Klein y Murray (2007) [31] presentan una implementación paralela de SLAM denominada PTAM (*Parallel Tracking and Mapping*). Divide el algoritmo en dos *threads*, uno que realiza el tracking robusto del movimiento de la cámara y otro que realiza el mapa de la escena a partir de los puntos de interés de los cuadros previos.
Posteriormente Sánchez et al (2010) [42] presentan una implementación de SLAM altamente paralelizable basada en simulaciones de Monte Carlo. Esta diseñada para ejecutarse en la GPU en tiempo real usando CUDA, y funciona tanto en escenas de interior como exterior.
Recientemente, Klein y Murray (2009) [32] y por su parte Wagner et al (2010) [59] presentan implementaciones de tracking y mapping para dispositivos móviles en tiempo real.

4.4. Tracking híbrido

En algunas aplicaciones de realidad aumentada las técnicas de visión por computador no proveen una solución de tracking robusta, por esto se han desarrollado métodos de tracking híbrido que consisten en combinar las salidas de dispositivos físicos y el análisis de video. Este tipo de tracking resulta efectivo para aplicaciones que requieren estimar la posición de la cámara con respecto a una escena estática, pero no se aplica al tracking de objetos en movimiento con una cámara estática.
Por un lado, las técnicas de tracking basado en visión son estables a largo plazo. Sin embargo, pueden resultar lentas y además los movimientos bruscos suelen causar fallas en el tracking con el consecuente consumo de tiempo en la recuperación. Por otra parte, el tracking basado en sensores, en particular el tracking inercial ofrece características complementarias Es rápido y robusto y puede usarse para predecir el movimiento cuando ocurren cambios bruscos. A partir de la aceleración y la velocidad de rotación puede estimarse la pose pero su desventaja radica en que tienden a ir a la deriva debido a acumulación de ruido. En Ribo et al (2002) [41] y en Klein y Drummond (2003) [30] se utiliza un sistema de tracking híbrido que utiliza sensores inerciales para apoyar las fallas del tracking basado en visión cuando se producen movimientos rápidos. En Reitmayr y Drummond (2006) [39] se introduce un sistema de tracking híbrido que combina diferentes enfoques conocidos: un tracking basado en bordes para una localización exacta, medidas de giroscopio para tratar los movimientos rápidos y medidas de campo gravitatorio y magnético para evitar ir a la deriva.
Por otra parte, Bleser et al (2006) [11] presenta un sistema híbrido que combina técnicas basadas en visión: la técnica de recuperación de la estructura de una escena a partir de movimiento –en inglés, *SFM* (*Structure From Motion*)–, SLAM (sección 4.3.5) y un tracking basado en el modelo.

4.5. Resumen de las técnicas de tracking

A continuación se presenta una tabla resumen con las diferentes alternativas de tracking en aplicaciones de realidad aumentada.

Tracking mediante…	**Aplicables a tracking de**		Posición y orientación	Conocimiento necesario	Modifica la escena
	cámara en mov. escena estática	**objetos en mov. con cámara estática**			
Sensores inerciales	SI	NO	relativos	NO	NO
GPS+brujula	SI	NO	Absolutos en sist. de coord.global	NO	NO
marcadores	SI	SI	Marcador en relación a la cámara	SI, del marcador	SI
Bordes	SI	SI	Relativos al cuadro anterior	SI, modelo 3D	NO
Texturas	SI	SI	Relativos al cuadro anterior	NO	NO
Detección	SI	SI	Absolutos	SI, base de datos de imágenes registradas	NO
SLAM/PTAM	SI	SI	Mapa de puntos registrados	NO	NO
Híbrido	SI	NO		NO	NO

5. Realidad aumentada en dispositivos móviles

5.1. Características de hardware y software

Las primeras aplicaciones de realidad aumentada móvil, y para ambientes externos (*outdoor*), utilizaban una mochila para cargar el conjunto de hardware incluyendo la PC y las fuentes de alimentación, y un visor en la cabeza o *HMD* (*head mounted display*). Las aplicaciones de realidad aumentada móvil siguen el modelo de dispositivo sostenido con la mano (*hand-held*) utilizando una tablet PC, una PDA o un *smartphone* o teléfono celular de última generación. Estos dispositivos cuentan con cámara integrada y opcionalmente dispositivos para tracking integrados como GPS, acelerómetro, giroscopio, brújula.

En general, los dispositivos móviles como PDAs y teléfonos móviles tienen unas características diferentes a los PC en relación al hardware y software. Dichas características influyen en el desarrollo de

cualquier algoritmo para este tipo de dispositivos, particularmente de realidad aumentada. Wagner y Schmalltieg (2007) [52] (2009) [57] señalan que:

- La mayoría usa CPUs ARM compatibles, de velocidad de procesador entre 100-600 MHz, optimizados para bajo consumo de corriente
- Generalmente no tienen FPUs (floating-point units) y las operaciones de punto flotante son emuladas por software, lo que las hace 50 veces más lentas que las operaciones con enteros
- Para reducir costes y conservar batería poseen poca memoria y el acceso es lento
- Las cámaras incorporadas tienen baja calidad comparadas con las usadas en PC para visión por computador, lo que causa que la luminancia radial decaiga hacia las esquinas).
- El ancho de banda interno limita el video a resoluciones típicas de 320x240. El formato del video suele ser propietario o formatos inusuales como YUV12.

La tecnología cambia muy rápido y recientemente (2010-2011) hubo un gran salto en los teléfonos celulares. Entre los avances más notorios relacionados con el campo de realidad aumentada se pueden enumerar el aumento de velocidad de la CPU y la inclusión de procesador gráfico. Por ejemplo, NVIDIA® Tegra™ 2 es el primer superchip destinado al mercado de los dispositivos móviles, ofrece alta capacidad multitarea gracias a la primera CPU de doble núcleo del sector CPU ARM Cortex-A9 dual core 1,2 GHz, el rendimiento de juego de una consola gracias a su GPU NVIDIA® GeForce® ULP de muy bajo consumo. Como ejemplos de celulares que cuentan con una tecnología apropiada para estas aplicaciones enumeramos a: Samsung Galaxy SII con procesador Exynos 4210 doble núcleo 1.2 GHz, 16 Gb de memoria, e integra un chip gráfico Mali400; Iphone 4 con procesador Apple A4 fabricado por Samsung ARM Cortex A8 Core de 1 GHz, memoria de 16 o 32 Mb, procesador gráfico PowerVR, sensores como giroscopio de 3 ejes, acelerómetro y localización vía AGPS y brújula digital
Entre los sistemas operativos más importantes se enumeran Symbian, Windows Mobile, Android, iOS (para Iphone). Pese a que las plataformas anteriores son programables son mutuamente incompatibles, lo que hace el diseño de software para varias plataformas más dificultoso. Aún entre diferentes modelos de dispositivos que soportan el mismo sistema operativo puede haber pequeñas incompatibilidades del hardware de bajo nivel que requiere

recompilación para cada modelo. Actualmente puede encontrarse una amplia variedad de aplicaciones, incluso de realidad aumentada, desarrolladas tanto para Android como para iOS, los dos sistemas operativos más dominantes.

5.2. Tracking

El tracking en dispositivos "hand-held" fuerza ciertas restricciones no presentes en otras configuraciones basadas en PC. Los sensores externos generalmente no son posibles ya además de su alto coste, los dispositivos móviles son pequeños y no tienen la interfaz necesaria para conectarlos. Actualmente existen móviles con GPS y sensores como acelerómetros y giroscopios incorporados (ver sección 4.2.).
El software de tracking tiene que diseñarse específicamente para correr bajo estas plataformas restringidas. El tracking basado en cámaras es una buena opción para estos dispositivos ya que es una alternativa que da resultados buenos con bajo coste.
El tracking de marcadores es una de las estrategias más usadas ya que es robusta y computacionalmente eficiente. Existen librerías de tracking de marcadores de código abierto para dispositivos móviles como ARToolKitPlus, creada por Wagner [52], aunque su desarrollo fue abandonado hace varios años. Wagner y Schmalstieg (2009) [56] presentan Studierstube ES, un framework de desarrollo de aplicaciones de realidad aumentada para dispositivos móviles. Pese a que la versión para PC es abierta, la versión para móviles no está disponible.
Por otra parte el tracking basado en características naturales resulta más atractivo dado que no se necesita la invasión de marcadores. Entre los algoritmos de tracking basado en características naturales, donde se tiene conocimiento de la escena, se encuentran la implementación de presenta un algoritmo de tracking basado en SIFT presentada por Wagner et al. (2008)[55] (2010) [58] el cual permite el seguimiento de un objeto plano texturado con una imagen cualquiera.
Por su parte Takacs et al (2008)[47] presenta una implementación de tracking basado en el descriptor SURF para ser ejecutada en teléfonos móviles. Según Wagner et al. (2010) [58] dicho algoritmo tiene un tiempo de ejecución 10 veces mayor al presentado por ellos.
Más recientemente, cobra interés el tracking basado en características naturales en una escena totalmente desconocida. Klein y Murray (2009) presentan PTAM (*Parallel Tracking and Mapping*) una implementación de SLAM para dispositivos móviles.
En la siguiente sección se enumeran algunas librerías de realidad aumentada para dispositivos móviles.

6. Software de realidad aumentada

6.1. Introducción

En 1999 Hirozaku Kato [29] desarrolla Artoolkit [7] en el hitLAB, una de las librerías de software libre más potentes hasta la fecha para la creación de aplicaciones en realidad aumentada. Desde entonces el desarrollo de plataformas de aplicaciones de realidad aumentada no ha parado de crecer con especial relevancia en aplicaciones para teléfonos móviles de última generación
Siguiendo el esquema de la figura 2 las tres partes principales en una aplicación de RA se trasladan a la necesidad de contar con:

- Librerías de captura de video
- Librerías de tracking
- Librerías de gráficos 3D

6.1.1. Librerías para PC

Entre las librerías de tracking de marcadores más conocidas y disponibles en código abierto se encuentran: ARTookit, ARTookitPlus y ARTag.
Existen ejemplos de librerías de tracking de características naturales, sin embargo hasta la fecha no existen disponibilidad de código abierto (hasta donde el autor de esta obra tiene conocimiento).
Por otra parte, entre el software disponible se pueden clasificar en:

- Librerías de tracking de bajo nivel, destinados a expertos que programen (ARToolkit, ARToolkitPlus, ARTAG)
- Frameworks y herramientas de autor: algunos destinados a los no programadores (BuildAR, DART, ATOMIC) y otros para proveer abstracción de algunas partes fundamentales de la aplicación (StudierStube, AMIRE)

6.1.2. Librerías para aplicaciones web

También algunas librerías se han adaptado para realizar aplicaciones web como es el caso de ARToolkit que fue implementada en Adobe Flash™ dando lugar a FLARToolKit [21]. IN2AR [27] es una librería basada también en Adobe Flash™, la cual que permite utilizar cualquier imagen en lugar de un típico marcador, pero no es de código abierto aunque está disponible de forma gratuita (bajo pena de tener siempre el logo de la empresa creadora incorporado en las aplicaciones desarrolladas). Entre las herramientas de autor se

encuentra Atomic Web [9] que permite crear aplicaciones web basadas en FLARToolKit de una forma muy rápida.

6.1.3. Librerías para móviles

El software mencionado hasta el momento está desarrollado para PC. Entre las librerías de bajo nivel disponible existe una implementación de código abierto distribuido bajo licencia GPL de ARTookitPlus la cual puede compilarse para dispositivos móviles. Según señala Wagner y Schmalstieg (2009) [56] [57] no es eficiente dado que fue una adaptación de la versión para PC y no una librería específicamente para dispositivos móviles. Ellos utilizan la librería StbTracker, desarrollada para el framework Studierstube ES, para aplicaciones que se ejecutan en teléfonos móviles. Sin embargo dicha librería no es de código abierto y la versión para móviles tampoco se distribuye libremente.
Recientemente se han dado a conocer una serie de SDK disponibles, aunque no son de código abierto, para desarrollo de aplicaciones de realidad aumentada para dispositivos móviles que incorporan tracking de una imagen cualquiera sin necesidad de utilizar los típicos marcadores. Se pueden citar los SDK de String [46] y Qualcomm AR [38]. Hasta la fecha String permite crear aplicaciones de RA para el sistema operativo iOS con seguimiento de imágenes enmarcadas y Qualcomm AR permite crear aplicaciones para teléfonos con sistema operativo Android y también iOS.
Un tipo de aplicaciones desarrolladas muy recientemente y que seguramente irá en crecimiento son las aplicaciones de localización que brindan información del entorno acorde a la posición global de usuario. Los denominados browsers de realidad aumentada- entre los cuales encontramos a Layar [33], Wikitude [61], junaio [28] –proveen su SDK para el desarrollo de aplicaciones para móviles basadas en la posición global como también utilizando tracking basado en visión (ver sección 6.4)–.
En las siguientes secciones se describen algunas de las librerías y herramientas más conocidas, para terminar el capítulo con una tabla resumen de las mismas.

6.2. Librerías de tracking para realidad aumentada

6.2.1. ARToolkit

La librería ARToolkit [7] presentada por Kato y Billinghurst (1999)[29] es una librería para tracking de marcadores enormemente

popular dado que es una librería C de distribución gratuita para uso en aplicaciones no comerciales y distribuida como código abierto bajo licencia GPL. Está muy bien documentada y hay muchos ejemplos disponibles.
Los ejemplos disponibles utilizan para captura de video llamadas a la librería DirectShow. Para los gráficos 3D incluye llamadas a la librería OpenGL y también permite la carga de modelos VRML.
Los marcadores utilizados son de tipo "template" (ver sección 4.3.3 y figura 10.a). Se necesita cargar un fichero donde está definido el marcador y por cada marcador nuevo se debe entrenar a ARToolkit para que lo reconozca.
El rango de distancia del marcador a la cámara está limitado, y naturalmente cuanto más largo sea el patrón físico se detectará a mayor distancia. Los patrones que tienen grandes áreas blancas y negras (baja frecuencia) son los más efectivos. A medida que el marcador se visualiza más inclinado menos se visualiza el centro del marcador y por tanto el reconocimiento es menos confiable.
Los resultados del tracking se ven afectados por las condiciones lumínicas. Para reducir el reflejo los marcadores puede ser de un material no reflectivo como papel-terciopelo.
Entre las limitaciones de ARToolkit la más importantes es que para reconocer la posición y orientación de un marcador el mismo debe ser totalmente visible. El algoritmo para detectar rectángulos se basa en una umbralización de la imagen y el seguimiento (*scan-line*) de un primer borde encontrado hasta que se completa la detección de los 4 bordes. Este algoritmo no detecta el marcador si uno de los 4 bordes está incompleto. En una escena pueden detectarse múltiples marcadores. Estos pueden tener posiciones y orientaciones independientes pero también puede ARToolkit permite utilizar una configuración espacial fija de muchos marcadores como solución al problema de detectar marcadores que se capturan incompletos.
Debido a que el algoritmo de detección de rectángulos trabaja con una precisión de subpíxeles, aún cuando el marcador permanece fijo en una posición y orientación la matriz estimada puede variar. En su forma de trabajo por defecto ARToolkit analiza cada cuadro de video de forma independiente. Una posible solución al problema de parpadeo es la posibilidad que brinda ARToolkit de habilitar un modo en el que se considera el resultado estimado del cuadro anterior y en caso de que el resultado del cuadro actual no difiera demasiado se tomar la estimación del cuadro anterior.
Provee dos algoritmos de calibración, uno rápido, con resultados menos precisos pero suficiente para superposición de imágenes, y otro más lento pero más preciso, necesario si se quiere realizar mediciones

en 3D. El primero utiliza imágenes de una cuadrícula de puntos para estimar de una sola vez tanto los parámetros de la matriz de proyección como la distorsión. El segundo utiliza dos fases, en la primera utiliza imágenes de la cuadrícula de puntos para estimar los parámetros de distorsión, y a continuación imágenes de una cuadrícula de líneas y en base a los parámetros de distorsión estima los parámetros de la matriz de proyección. Mediante cualquiera de los dos métodos de calibración se genera un nuevo archivo con parámetros de calibración. En caso de no realizar la calibración pueden usarse una calibración por defecto.
Existen otras librerías que se crearon a partir de modificaciones de ARToolkit como ARTag y ARToolkit plus.
Existen además versiones de ARToolkit en otros lenguajes como Java o Processing. En particular, FLARToolKit [21] es la versión ActionScript 3, lenguaje utilizado en aplicaciones web realizadas en el entorno Adobe Flash™. Es de hacer notar que la aparición de esta versión de la librería ha dado lugar a una explosión de páginas web que incluyen una aplicación de realidad aumentada sobre todo con fines publicitarios.

6.2.2. ARTag

ARTag [6], creada por Fiala (2005) [19] y (2010) [20] se inspiró en ARToolkit. Al igual que ésta es una librería de tracking basado en marcadores que usa un procesamiento de imágenes más complejo para lograr más alta confiabilidad e inmunidad a cambios de iluminación.
Los marcadores utilizados son de tipo "ID Marker" (ver sección 4.3.3 y figura 10.b). Estos marcadores poseen algunas ventajas con respecto a los marcadores "template" utilizados por ARToolkit. La ventaja de este tipo de marcadores es que pueden utilizarse cientos de marcadores diferentes sin necesidad de entrenamiento. ARTag no necesita ficheros de marcadores como ARToolkit sino que tiene una librería de 2002 marcadores que pueden identificarse de 0 a 2047 (46 ilegales). Se usa la decodificación digital para identificar el marcador en lugar de la correlación necesaria con ARToolkit. Esto resulta en más eficiencia y evita la falsa detección o la confusión entre marcadores. También utiliza los marcadores BCH (Bose, Ray-Chaudhuri, Hocquenghem).
Al aumentar los marcadores presentes en una escena resultan más rápido detectarlos dado que no es necesario buscar y comparar en una base de datos de marcadores puesto que pueden decodificarse.

Con respecto a los gráficos 3D soporta carga de objetos 3D con formato WRL (VRML), OBJ (Wavefront, Maya), ASE (3D-Studio export), y tiene soporte OpenGL.
El algoritmo de detección de cuadriláteros se basa en detección de bordes en lugar de usar umbralización como utiliza ARToolkit, el cual permite la detección de marcadores parcialmente ocluidos. Otra de las ventajas de ARTag en relación a ARToolkit es que la primera funciona mejor bajo condiciones de luz variables.
ARTag se abandono en 2009 porque expiró el contrato del autor con NRC de Canadá donde se desarrolló.

6.2.3. ARToolkitPlus

ARToolkitPlus [8] es el sucesor de ARToolKit, implementado por Wagner [53] como un API de clases C++.
El código fuente está disponible aunque no está suficientemente documentado y no fue actualizado desde junio 2006 ya que sus autores se dedicaron desde esa fecha al proyecto Studierstube [43] [44].
Esta librería optimiza el código de ARToolkit y puede usarse tanto en PC como también existe una versión para dispositivos móviles.
Inspirado en los marcadores de ARTag, usa marcadores de tipo "ID markers". Utiliza hasta 512 marcadores diferentes, los cuales codifican un número de 9-bits en un patrón de 6x6, y los 9 bits se repiten 4 veces para llenar los 36 bits.
Al igual que con ARTag, con estos marcadores no se necesita entrenamiento y resulta más rápido que los marcadores "template" utilizados por ARToolkit sobre todo al aumentar el número de marcadores presentes en una escena. Al igual que ARTag, también utiliza los marcadores BCH (Bose, Ray-Chaudhuri, Hocquenghem).
Entre las características incorporadas se encuentran:

- Umbralización automática: se realiza una umbralización basada en la mediana de todos los píxeles del último marcador detectado
- Compensación de "vignetting"
- La compensacion de distorsion es computacionalmente cara, ARTK+ puede habilitarse el uso de una lookup table para acelerar el proceso

Para la versión PC, se puede elegir entre el algoritmo de estimación utilizado en ARToolkit y otro algoritmo de estimación de posición y orientación denominado RPP (*Robust Planar Pose*) de Schweighofer y Pinz (Inst.of l.Measurement and Measurement Signal Processing, Graz University of Technology), que da como resultado un tracking más estable.

6.3. Frameworks de realidad aumentada

6.3.1. Studierstube

Studierstube [43][44] es un framework completo para el desarrollo de aplicaciones de realidad aumentada. Incluye:
- StbTracker: tracking de marcadores
- Muddleware: comunicación multiusuario
- Stb SG: maneja el grafo de la escena virtual

El framework está disponible para PC aunque el proyecto fue abandonado en el año 2008.
Studierstube ES [56], un framework de desarrollo de aplicaciones de realidad aumentada para dispositivos móviles. Pese a que la versión para PC es abierta, la versión para móviles no está disponible.

6.3.2. AMIRE

Grimm et al (2002) [24] presentan AMIRE [1], acrónimo de "Authoring Mixed Reality", una herramienta de autor que permite la creación de aplicaciones de realidad aumentada.
El proyecto AMIRE establece la denominadas "MR GEM" que es una colección de técnicas, algoritmos, biblioteca pública de códigos con soluciones eficientes a problemas de programación comunes en aplicaciones de realidad aumentada.
El framework de realidad aumentada AMIRE incluye:
- Componentes 2D y 3D que pueden configurarse por medio de propiedades ("properties")
- Comunicación entre componentes basada en "slot" donde pueden intercambiar datos
- Convenciones para las componentes 2D y 3D tal como mecanismo de "picking"
- Persistencia de la aplicación en un formato de archivo basado en XML que contiene una lista de dependencias de librerías, instancias de componentes y las conexiones entre componentes

6.3.3. DART

MacIntyre et al (2003) [35] presentan DART [15], acrónimo de "Designers Augmented Reality Toolkit" Es una colección de extensiones del ambiente de programación multimedia Macromedia Director.

Soporta *streaming* de video, tracking de marcadores por medio de ARToolkit y entrada a un amplio rango de trackers y sensores físicos vía el "VRPN sensor package".

6.3.4. ATOMIC

ATOMIC Authoring Tool [9] es una herramienta que permite la creación de aplicaciones de realidad aumentada desarrollada especialmente para no-programadores. Fue creado como un "front-end" o interfaz gráfica para la usar librería ARToolkit sin tener que saber programar. Está escrito en el lenguaje de programación Processing y se distribuye bajo licencia GNU GPL.
Su variante ATOMIC WEB es una herramienta que permite la creación de aplicaciones de realidad aumentada para exportarlas a cualquier sitio web. Fue creado como un "front-end" para la usar librería FLARToolKit sin tener que saber programar. El núcleo está escrito en ActionScript3.

6.3.5. BuildAR

BuildAR [13] es una aplicación desarrollada en el laboratorio HITLabNZ por los mismos creadores de ARToolkit que permite crear una escena de realidad aumentada con tracking de marcadores utilizando ARToolkit sin necesidad de programar.
Puede descargarse una versión de prueba del sitio web

6.4. Browsers de realidad aumentada

Los denominados browsers de realidad aumentada son aplicaciones que proveen contenidos relevantes del entorno dependiendo tanto de la ubicación del usuario en un cierto lugar en el mapa (información de lugares, eventos, ofertas, objetos de alrededor) como también dependiendo de la imagen que está observando. Están pensados para ejecutarse en un dispositivo móvil como los teléfonos celulares y pueden encontrarse versiones para los sistemas operativos iOS y Android. Algunos de ellos también están disponibles para otros sistemas operativos.
Los browsers de RA pueden proveer dos tipos de servicios:

- Basados en la localización: a través del GPS y otros sensores del teléfono móvil: se conoce la posición del usuario y en que dirección apunta la cámara (denominada Gravimetric AR[36]). De esta forma se puede visualizar información relevante a la posición

mostrando que hay alrededor dependiendo de los intereses del perfil.
- Basados en imágenes: la aplicación reconoce imágenes capturadas por la cámara del teléfono (por ejemplo objetos, sitios, cuadros, revistas). Una vez reconocidas se puede superponer dinámicamente en el visor contenido multimedia de forma que un objeto 3D esté pegado virtualmente al objeto reconocido.

Un browser de RA se compone de las siguientes partes visuales:
- Radar
- Burbujas de información
- Barra de información
- Rango
- Mapa
- Tienda de compras (App Store)

El radar provee una ubicación visual de los puntos de interés, y en base a esta información el usuario sabe a donde debe apuntar el dispositivo.
Las burbujas de información son íconos usados para brindar información del tipo de punto de interés.
La barra de información muestra una descripción del punto de interés una vez que el usuario selecciona una burbuja de información.
El rango permite establecer al usuario a que distancia máxima se encuentran los puntos de interés que quiere visualizar.
En lugar de visualizar los puntos de interés superpuestos con la realidad, también es posible visualizarlos en un mapa tradicional.
Los browsers de RA más conocidos hasta el momento fueron creados entre 2008-2010 y son los siguientes:
- Wikitude [61], creado en 2008 por la empresa Mobilizy
- Layar [33], creado en 2008 por la empresa LayarMet
- junaio [28], creado en 2009 por la empresa Metaio Augmented Reality Solutions

Madden (2011) [36] realiza una muy buena descripción de cada uno de estos browsers de AR e introduce a la programación utilizando el SDK disponible para cada uno.

6.4.1. Wikitude

Wikitude es el browser de AR para el que más fácilmente se puede crear contenido sin necesidad de escribir una línea de código. En la jerga de Wikitude los desarrolladores construyen mundos (worlds), una vez creados se envían al servidor de Wikitude quien realiza el

hosting de los mismos. Los mundos pueden crearse usando la interfaz de Google Maps, el lenguaje de marcas KLM o el lenguaje de marcas ARML.
KLM (Keyhole Markup Language)es un lenguaje basado en XLM que describe información geográfica y es el lenguaje usado por Google Earth para describir ubicaciones, coordenadas, etc.
ARML (Augmented Reality Markup Language) [2] es una especificación creada por Mobilizy GmbH que permite a los desarrolladores crear contenido que puede visualizarse en diferentes AR browsers.
Dado que KLM es un estándar común los desarrolladores Wikitude tienen acceso a un gran rango de contenido sin necesidad de formateos particulares de los datos. Tanto el archivo KLM o ARML describen puntos de interés del mundo y como se dijo se envían al servidor Wikitude el cual los pone a disposición de los clientes.
Wikitude está basado en datos Wikipedia, se usó en principio como guía de ciudad que permitía encontrar lugares de interés turístico. Actualmente cuenta con más de quinientos mundos desarrollados por todo el mundo. Permite visualizar videos de Youtube, acceso a webcams de todo el mundo, imágenes Fliker geocodificadas, búsquedas de negocios mediante el Google Local, etc.

6.4.2. Layar

Fue el primer browser de AR aparecido en sólo dos años consiguió un millón de usuarios y 2000 *layers* creados y una comunidad activa construyendo herramientas de código abierto para los desarrolladores. El contenido creado se denomina *layer*, al igual que Wikitude denomina *worlds*. Los layers son ordenados de acuerdo a categorías y al igual que los sitios de Internet pueden agregarse a los favoritos. A diferencia de Wikitude, el desarrollador debe encargarse del *hosting* del contenido. Los desarrolladores pueden utilizar MSQL y PSP. Layar tiene una serie de características adicionales:

- En lugar de íconos simples se pueden agregar objetos 3D.
- *Triggers* que definen acciones que ocurrirán cuando el usuario se aproxime a una cierta ubicación.
- Los puntos de interés pueden tener asociado audio.
- Autenticación con usuario y clave

Con estas características los desarrolladores están construyendo un amplio rango de contenido incluyendo juegos que requieren que los usuarios se desplacen e interactúen con el entorno usando la cámara.

6.4.3. Junaio

Desarrollada por Metaio, es el último de los tres browsers, apareció a finales de 2009, en lugar de anunciarse como un browser AR fue anunciado como el primer browser de red social de AR.
Junaio utiliza tecnología de AR desarrollada previamente por Metaio, incluido su SDK de tracking de características naturales. Actualmente es un browser AR con 150 canales (*channels*) – así es como se denomina el contenido– y otros 650 en desarrollo. Permite compartir contenido con amigos y visualizar contenido cercano a la localización del usuario.
Las funcionalidades incluidas son:

- Puntos de interés simples con sonido y video,
- Objetos 3D y animaciones
- Detección de proximidad
- Tracking de características naturales (reconoce imágenes y se visualizan un objeto 3D superpuesto o un video)
- LLA (latitud, longitud, altitud): Dado que GPS no es siempre exacto, particularmente en interior de edificios LLA consiste en marcadores especiales que ayudan a establecer la posición del usuario mediante el tracking del marcador.

7. Conclusiones

No hay duda que el campo de aplicaciones de realidad aumentada seguramente interesa a cualquier persona. En la actualidad la realidad aumentada es un área muy activa de desarrollo e investigación.
Más allá de las primeras aplicaciones en medicina, aplicaciones militares, asistencia a la fabricación, asistencia en la realización de tareas, etc., el gran auge de aplicaciones actuales está centrado en publicidad, juegos y aplicaciones móviles relacionadas con servicios basados en la localización.
Mientras que las primeras aplicaciones de RA realizaban un tracking del participante puramente mediante dispositivos especiales para tracking, las técnicas de tracking basado en visión y las técnicas de tracking híbrido son las de más uso en la actualidad. En los últimos 10 años el tracking basado en marcadores tuvo un gran auge. El hecho de la disponibilidad pública de diferentes librerías de tracking basado en marcadores, tales como ARToolkit, ARTag y ARToolkitPlus, permitió la implementación de gran variedad de aplicaciones de RA. Actualmente existe especial interés en el tracking basado en la extracción de características naturales en la escena siendo esta una de

las áreas de investigación más activas. Las técnicas de tracking basadas en detección de bordes y texturas, así como el tracking mediante detección han sido ampliamente estudiadas en los últimos años. Mientras que estas técnicas necesitan conocimiento de un objeto del mundo real, las últimas investigaciones se basan en el uso de algoritmos de robótica como SLAM para realizar tracking a la vez que se realiza un mapa del entorno desconocido.

La gran masificación de los celulares así como el aumento reciente de su capacidad de procesamiento y memoria y la integración de dispositivos como GPS, brújulas, acelerómetros y giroscopios, hace que exista un gran campo de investigación y desarrollo de algoritmos y aplicaciones de realidad aumentada para este tipo de dispositivos. Recientemente aparecieron los denominados browsers de realidad aumentada, tales como Layar, Wikitude y junaio, los cuales ofrecen servicios basados en la localización superponiendo información de acuerdo a la localización del usuario. Además incorporan algoritmos de visión para el tracking, reconocimiento y visualización de información sobre los objetos reales. Cada uno de los browsers cuenta con un SDK disponible para el sistema operativo Android y el sistema operativo iOS, lo que seguramente dará lugar a un gran crecimiento de este tipo de aplicaciones.

8. Referencias

[1] AMIRE Authoring Tool [en línea] Consultado el 27 de septiembre de 2011 en <http://www.amire.net/>.

[2] ARML (Augmented Realit.y Markup Language) [en línea] Consultado el 27 de septiembre de 2011 en <http://www.openarml.org/wikitude4.html>.

[3] Armstrong, M. and Zisserman, A. (1995). "Robust object tracking". *Proceedings of the Asian Conference on Computer Vision,* pp 58-62.

[4] Azuma, R. (1997). "A Survey of Augmented Reality". *Presence: Teleoperators and Virtual Environments*, 6(4), pp 355-385.

[5] Azuma, R.; Baillot, Y.; Behringer, R.; Feiner, S.; Julier, S. and MacIntyre, B. (2001). "Recent Advances in Augmented Reality". *IEEE Computer Graphics and Applications*, 21(6), 34-47.

[6] ARTag [en línea] Consultado el 27 de septiembre de 2011 en <http://www.artag.net>.

[7] ARToolkit [en línea] Consultado el 27 de septiembre de 2011 en <http://www.hitl.washington.edu/artookit>.

[8] ARToolkitPlus [en línea] Consultado el 27 de septiembre de 2011 en <http://handheldar.icg.tugraz.at/artoolkitplus.php>.
[9] ATOMIC Authoring Tool [en línea] Consultado el 27 de septiembre de 2011 en
<http://www.sologicolibre.org/projects/atomic/en/>.
<http://www.sologicolibre.org/projects/atomicweb/es/>.
[10] Bay, H.; Ess, A.; Tuytelaars, T.; Van Gool, L (2008). "SURF: Speeded Up Robust Features", *Computer Vision and Image Understanding (CVIU)*, 110 (3), pp 346-359.
[11] Bleser, G.; Wuest, H. and Stricker, D. (2006). "Online camera pose estimation an partially known and dynamic scenes" *Proceedings of 5th IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2006).*
[12] Bouguet, J. *Camera Calibration Toolbox for Matlab.* [en línea] Consultado el 27 de septiembre de 2011 en
<http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/links.html.
[13] BuildAR [en línea] Consultado el 27 de septiembre de 2011 en <http://www.buildar.co.nz>.
[14] Cyganek, B. and Siebert, J. (2009) *An Introduction to 3D Computer Vision Techniques and Algorithms.* John Wiley & Son Ltd., ISBN 978-0-470-01704-3.
[15] DART Designers Augmented Reality Toolkit [en línea] Consultado el 27 de septiembre de 2011 en
<http://www.cc.gatech.edu/dart/applications.htm>.
[16] Davison, A., Reid, I., Molton, N.D., Stasse, O.: MonoSLAM: (2007). "Real-time single camera SLAM" *IEEE Transaction on. Pattern Analysis and Machine Intelligence,* 29, pp 1052–1067.
[17] Drummond, T. and Cipolla, R. (2002). "Real time visual tracking of complex structures". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7), pp 932-946.
[18] Evans. C. "Notes on the OpenSURF Library" (2009). University of Bristol. [en línea] Consultado el 27 de septiembre de 2011 en <http://www.cs.bris.ac.uk/Publications/pub_master.jsp?id=2000970>.
[19] Fiala, M. (2005). "ARTag, a Fiducial Marker System Using Digital Techniques" *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* pp 590-596.
[20] Fiala, M. (2010). "Designing Highly Reliable Fiducial Markers", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 32 (7).
[21] FLARToolKit [en línea] Consultado el 27 de septiembre de 2011 en <http://www.libspark.org/wiki/saqoosha/FLARToolKit/en>.
[22] Fua, P. and Lepetit, V. (2010). "Vision Based 3D Tracking and Pose Estimation for Mixed Reality". En Haller, M; Billinghurst, M.

and Thomas, B. *Emerging Technologies of Augmented Reality. Interfaces and Design*, 1-22, Idea Group Publising, ISBN 1-59904-067-0.
[23] Genc, Y.; Riedel, S.; Souvannavong, F. and Navab, N. (2002). "Marker-less tracking for augmented reality: a learning-based approach", *Proceedings of 1st IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2002)*.
[24] Grimm P., Haller M., Paelke V., Reinhold S., Reimann C., Zauner J. (2002). "AMIRE Authoring Mixed Reality" *Proceedings of the first IEEE International Augmented Reality Toolkit Workshop.*
[25] Heikkilä, J. (2000). "Geometric camera calibration using circular control points". *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 22 (10), pp 1066-1077.
[26] Hess, R. "Open Source SIFT Library" [en línea] Consultado el 27 de septiembre de 2011 en <http://blogs.oregonstate.edu/hess/code/sift>.
[27] IN2AR Augmented Reality Engine for Adoble Flash™ Player [en línea] Consultado el 27 de septiembre de 2011 en <http://www.in2ar.com>.
[28] junaio Developer Portal [en línea] Consultado el 27 de septiembre de 2011 en <http://dev.junaio.com>.
[29] Kato, H. and Billinghurst, M. (1999). "Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System." *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*, pp 85-94.
[30] Klein, G. and Drummond, T. (2003). "Robust visual tracking for non-instrumented augmented reality" *Proceedings of 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2003),* pp 113-122.
[31] Klein, G. and Murray, D. "Parallel Tacking and Mapping for small AR workspaces" (2007). *Proceedings of 6th IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2007).*
[32] Klein, G. & Murray, D. (2009). "Parallel Tracking and Mapping (PTAM) on a Camera Phone" *Proceedings of 8th IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2009).*
[33] Layar Browser [en línea] Consultado el 27 de septiembre de 2011 en <http://www.layar.com browser/>.
[34] Lowe, D. G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints", *International Journal of Computer Vision*, 60 (2), pp 91-110.
[35] MacIntyre, B.; Gandy, M.; Bolter, J.; Dow, S. and Hannigan, B. (2003). "DART: The Designer's Augmented Reality Toolkit."

Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR03), pp 329-339.
[36] Madden, L. (2011). *Professional Augmented Reality Browsers for Smartphones. Programming for junaio, Layar and Wikitude.* John Wiley & Sons, ISBN 978-1-119-99281-3.
[37] *Intel Open Source Computer Vision Library* [en línea] Consultado el 27 de septiembre de 2011 en <http://software.intel.com/sites/oss/pdfs/OpenCVreferencemanual.pdf>.
[38] Qualcomm AR Developer Network [en línea] Consultado el 27 de septiembre de 2011 en <http://developer.qualcomm.com/develop/mobile-technologies/augmented-reality>.
[39] Reitmayr, G. and Drummond, T. (2006). "Going out: robust model-based tracking for outdoor augmented reality", *Proceedings of the 5th IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR06)*, pp 109-118.
[40] Reitmayr, G.; Langlotz, T.; Wagner, D.; Mulloni, A.; Gerhard, S. and Schmalstieg, D. (2010). "Simultaneous Localization and Mapping for Augmented Reality", *Proceedings of International Symposium on Ubiquitous Virtual Reality.(ISUVR 2010)*.
[41] Ribo, M.; Lang, P.; Ganster, H.; Brandner, M., Stock, C. and Pinz, A. (2002). "Hybrid Tracking for outdoor augmented reality applications." *IEEE Computer Graphics and Applications,* 22 (6), pp 54-63.
[42] Sánchez, J.; Álvarez, H. and Borro, D (2010). "Towards real time 3D tracking and reconstruction on a GPU using Monte Carlo simulations", *Proceedings of 9th IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2010)*, pp 185-192.
[43] Studierstube Project [en línea] Consultado el 27 de septiembre de 2011 en <http://www.icg.tugraz.at/project/studierstube>.
[44] Schmalstieg, D.; Fuhrmann, A.; Hesina, G.; Szalavari, Z.; Encarnacao, L.; Gervautz, M. and Purgathofer, W. (2002). "The Studierstube augmented reality Project". *Presence: Teleoperators and Virtual Environments,* 11(1).
[45] Skrypnyk & Lowe (2004). "Scene modelling, recognition, and tracking with invariant image features". *Proceedings of 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2004)*, pp 110-119.
[46] String [en línea] Consultado el 27 de septiembre de 2011 en <http://www.poweredbystring.com/product >.
[47] Takacs, G.; Chandrasekhar, V.; Gelfand, N.; Xiong, Y.; Chen, W.-C.; Bismpigiannis, T.; Grzeszczuk, R.; Pulli, K.; Girod, B. (2008). "Outdoors Augmented Reality on Mobile Phone using Loxel-Based Visual Feature Organization" *Proceedings of the 1st International*

conference on Multimedia information retrieval (MM' 08), pp 427-434.
[48] Thrun, S.; Liu, Y.; Koller, D.; Ng A. and Durrant-Whyte, H. (2004). "Simultaneous Localisation and Mapping with Sparse Extended Information Filters", *International Journals on Robotics Research,* 23(7-8), pp 693-716.
[49] Trucco, E. and Verri, A. (1998). *Introductory Techniques for 3D Computer Vision.* Prentice Hall, ISBN:0-13-261108-2.
[50] Tsai, R (1987). "A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses", *IEEE Journal on Robotics and Automation,* RA-3 (4), pp 323-344.
[51] Vaccetti, L.; Lepetit, V. and Fua, P. (2004). Combining edge and texture information for real time accurate 3D camera tracking. *Proceedings of 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2004).*
[52] Vacchetti, L.; Lepetit, V. and Fua, P (2004). "Stable real-time tracking using online and offline." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10), 1385, 1391.
[53] Wagner, D. and Schmalstieg, D. (2007). "ARToolkitPlus for Pose Tracking on Mobile Devices". *Proceedings of 12th Computer Vision Winter Workshop (CVWW 07),* pp 139-146.
[54] Wagner, D.; Langlotz, T. and Schmalstieg, D. (2008). "Robust and Unobtrusive Marker Tracking on Mobile Phones". *Proceedings of 7th IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2008)*, pp 121-124.
[55] Wagner, D.; Reitmayr, G.; Mulloni, A.; Drummond, T. and Schmalstieg, D. (2008). "Pose Tracking from Natural Features on Mobile Phones", *Proceedings of 7th IEEE/ACM International Symposium on Mixed and Augmented Reality(ISMAR 2008)*, pp 125-134.
[56] Wagner, D. and Schmalstieg, D. (2009). "Making Augmented Reality Practical on Mobile Phones, Part I" *IEEE Computer Graphics and Applications*, 29 (3), pp 12-15.
[57] Wagner, D. and Schmalstieg, D. (2009). "Making Augmented Reality Practical on Mobile Phones, Part II" *IEEE Computer Graphics and Applications*, 29 (4), pp 6-9.
[58] Wagner, D.; Reitmayr, G.; Mulloni, A.; Drummond, T. and Schmalstieg (2010). "Real-Time Detection and Tracking for Augmented Reality on Mobile Phones", *IEEE Transactions on Visualization and Computer Graphics,* 16 (3), pp 355-368.
[59] Wagner, D.; Mulloni, A.; Langlotz, T. and Schmalstieg, D. (2010). "Real-time Panoramic Mapping and Tracking on Mobile

Phones", *Proceedings of IEEE Virtual Reality Conference 2010 (VR'10)*.
[60] Wikipedia, "SURF" [en línea] Consultado el 27 de septiembre de 2011 en <http://en.wikipedia.org/wiki/SURF>.
[61] Wikitude [en línea] Consultado el 27 de septiembre de 2011 en <http://www.wikitude.com/developer/getting-started>.
[62] Zhang, Z. (2000). "A flexible new technique for camera calibration" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22 (11), pp 1330-1334.
[63] Zhang, X.; Fronz, S. and Navab, N. (2002). "Visual marker detection and decoding in AR systems: a comparative study". *Proceedings of 1st IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2002)*, pp 97-106.
[64] Zhou, F.; Dhu, H. and Billinghurst, M. (2008). "Trends in Augmented Reality Tracking, Interaction and Display: A Review of Ten Years of ISMAR" *Proceedings of 7th IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2008)*, pp 193-202.

CAPÍTULO 4

Introducción a las Interfaces Basadas en Visión

Cristina Manresa-Yee, Universidad de las Islas Baleares, España
Ramón Mas Sansó, Universidad de las Islas Baleares, España

1. Introducción

Diariamente y de forma continua y ubicua trabajamos con sistemas informáticos ya sea en forma de teléfono móvil, de cajero automático o de ordenadores.
Desde la aparición de los primeros ordenadores, la comunicación entre éstos y las personas ha sido un tema de investigación que ha hecho evolucionar los dispositivos desde las incómodas tarjetas perforadas hasta las intuitivas Interfaces de Usuario Naturales (en inglés, NUI).
En los primeros días de la informática, la comunicación persona-ordenador se conseguía a través de tarjetas perforadas o interruptores. Eran sistemas poco amigables, poco intuitivos y difíciles de utilizar. Los usuarios tenían que adaptarse al sistema.
Preece [45] definió los estilos de interacción como un término genérico para agrupar las diferentes maneras en que los usuarios se comunican o interaccionan con el ordenador. En los 70 aparecieron los sistemas basados en línea de comandos, primero a través de terminales de teletipo y luego con teclados electrónicos y monitores basados en texto. El ordenador recibe órdenes de forma directa a través de teclas de función, caracteres, abreviaturas o palabras. Fue el primer estilo de interacción que se popularizó y que aún hoy en día se utiliza por su velocidad y la flexibilidad que ofrece la parametrización de sus comandos, su uso en tareas repetitivas o su inclusión en lenguajes de script. Ahora bien, su utilización y aprendizaje no es fácil ya que utiliza unos protocolos rígidos y hay que conocer y memorizar los comandos [14].
En los 80, nacieron las interfaces gráficas de usuario (IGU) junto con las WIMP (Ventanas, Iconos, Menús y Dispositivos de puntero.

Windows, Icons, Menus and a Pointing device en inglés) en Xerox PARC. Los elementos tienen una representación continua de los objetos y de las acciones de interés. Las IGU se basan en "la manipulación directa del objeto de interés" [46]. Se tiene una representación del objeto y el usuario realiza acciones directas sobre el objeto que causan un efecto visible sobre el objeto seleccionado.

A pesar de que hoy en día las IGU basadas en WIMP son las más populares y que el paradigma de escritorio es el más extendido, existen otros paradigmas de interacción que empiezan a estar presentes en todas las esferas de nuestra vida diaria: la computación ubicua, la realidad virtual y la realidad aumentada

Por ello los estudios de investigación para obtener nuevos sistemas de interacción basados en otras fuentes de percepción de información como el sonido, el tacto o la visión siguen siendo importantes y se han convertido en un campo en auge que pretende desarrollar interfaces más naturales, intuitivas, no invasivas y eficientes.

Para los humanos, la información visual forma parte de la comunicación y de la interacción entre personas. Dentro de un contexto, el señalar un objeto con el dedo, puede decir tanto o más que el describirlo. Cuando vemos una persona, sin que ésta diga nada, por su expresión facial también podemos conocer su estado e inferir mucha más información de la que nos puede decir oralmente: si le vemos con el ceño fruncido, probablemente signifique que está contrariado; si le vemos sonriendo, significará que está de buen humor o feliz. Las personas interaccionamos a través del habla, pero también a través del lenguaje corporal (gestos, poses, miradas, expresiones faciales) que ayudan a expresar emociones, humor, atención, etc. o a interaccionar con el entorno y lo generamos de forma natural y a veces inconscientemente (ver figura 1).

Si al ordenador se le da la capacidad del sentido de la vista a través de una cámara, el ordenador podrá utilizar este canal como entrada de información de interacción. Hoy en día, las cámaras de bajo coste y con suficiente resolución se encuentran en gran parte de las estaciones de trabajo y en dispositivos frecuentemente utilizados como móviles.

Figura 1. Gestos, signos, miradas y expresiones faciales

Además el uso de cámaras como sensores de entrada hacia los sistemas está presente en interfaces de usuario tangibles (en inglés TUI), realidad virtual y aumentada, computación ubicua o computación pervasiva.
Para ello, será necesario extraer y procesar la información proveniente de la cámara utilizando técnicas de visión por computador para intentar simular lo que harían los ojos y obteniendo las denominadas **Interfaces basadas en visión** (ver figura 2).
El concepto de **Interfaces basadas en visión** reúne dos disciplinas: visión por ordenador (VPO) y la interacción persona-ordenador (IPO).
Las ventajas de este tipo de sistemas pueden resumirse en:

- Interacción a distancia: mientras el objeto de interés esté visible y con suficiente resolución para trabajar con él, este tipo de interfaces permiten al usuario comunicarse con el sistema a cierta distancia.
- No intrusivos: al utilizar una cámara para permitir al ordenador utilizar el canal visual para extraer información, el usuario no precisa tener contacto directo con ningún dispositivo. Además, el usuario no tiene que ponerse encima ningún sensor con lo que se le ofrece una interacción flexible, natural y más cómoda.
- Bajo coste: actualmente se pueden encontrar en el mercado gran variedad de cámaras con características y costes diferentes. Además, muchos sistemas se distribuyen ya con cámaras incorporadas como por ejemplo en móviles o portátiles.

- Entornos ruidosos: este tipo de interfaces sólo utilizan información capturada por el canal visual, por lo que el ruido no le afecta y pueden ser una solución para entornos ruidosos.
- Interacción natural: los humanos nos comunicamos en gran parte con lenguaje corporal, por lo que se naturaliza nuestra interacción con el ordenador.

Las dificultades con las que se encuentran son:

- Sobrecarga de información: el ojo humano filtra gran parte de la imagen que le llega a la retina haciendo sólo caso a lo que considera importante. Además, a diferencia del ordenador, la persona cuenta con una experiencia previa que le ayuda en esta tarea. Computacionalmente este proceso es más difícil.

Figura 2. Entorno: Gestos recogidos por una cámara

- Entorno: la iluminación y fondo variables pueden dificultar la robustez y la precisión de los sistemas de entrada basados en visión.
- Independencia de personas: lo ideal es que este tipo de sistemas sean funcionales a pesar de la gran variabilidad física de las personas, posturas que pueden realizar, la ropa que llevan, etc.

- Independencia de cámaras: las aplicaciones desarrolladas deberían ser independientes a los parámetros de las cámaras.

En las siguientes secciones se describirán conceptos relacionados con las dos disciplinas que envuelven las interfaces basadas en visión, para finalmente describir exhaustivamente un caso específico.

2. Interacción persona-ordenador: conceptos

El área de IPO nació como una disciplina que estudia aspectos de diseño, evaluación e implementación de los sistemas de computación para el uso humano y todos los fenómenos vinculados [17]. En IPO, hay tres elementos: la persona o usuario, el ordenador y la interacción. La persona es un usuario individual o un grupo de personas que se encuentran en un contexto e interaccionan con el sistema y de las cuales hay que conocer factores como sus habilidades cognitivas y perceptuales, sus capacidades físicas, las tareas que tienen que realizar y su contexto de uso. El ordenador es cualquier sistema o tecnología utilizada que cuenta con un conjunto de propiedades como su movilidad o su sistema operativo. Por último está la interacción, es decir, todos los intercambios que suceden entre la persona y el ordenador [3].
Los objetivos de la IPO son desarrollar o mejorar la seguridad, utilidad, efectividad, eficiencia y usabilidad de la comunicación de la persona con sistemas que incluyan computadoras [11].
La IPO depende de estudios realizados tanto en la parte humana como en la parte de máquina, por tanto, tiene que estar alimentada de forma multidisciplinar y englobar a profesionales de diferentes campos tales como por ejemplo la informática, la psicología, la ergonomía, la sociología, la antropología o el diseño industrial.
El término interfaz se ha definido como:

"Un conjunto de dispositivos, tanto lógicos como físicos que permiten interactuar de una manera precisa y concreta con un sistema informático" [18].

"La interfaz es una superficie de contacto entre dos entidades. En la interacción persona-ordenador estas entidades son la persona y el ordenador. La interfaz de usuario es el principal punto de contacto entre el usuario y el ordenador; es la parte del sistema que el usuario ve, oye, toca y con la que se comunica" [19].

Shneiderman [47] dice que cualquier sistema cuya intención de uso sea la interacción persona ordenador tiene que decidir que es aceptable para los siguientes requerimientos:

- Tiempo de aprendizaje: es el tiempo necesario para aprender a utilizar la interfaz.
- Velocidad de rendimiento: es la velocidad de la interfaz de usuario, no la velocidad del software. Es el número de caracteres a teclear, botones a apretar, clics de ratón o movimientos del ratón a ejecutar para llevar a cabo una operación. Este factor entra normalmente en conflicto con el tiempo de aprendizaje ya que las interfaces más rápidas suelen ser más complicadas de aprender.
- Ratio de errores: el ratio de errores producido por un usuario puede ser debido a la mala estructura de la interfaz de usuario. Se ve afectada por factores como la consistencia o la organización de las pantallas en las IGUs.
- Retención a lo largo del tiempo: cuanto más cerca esté la sintaxis de las operaciones a la comprensión del mundo por parte del usuario más fácil le será recordar la interfaz. Ahora bien, si el tiempo de aprendizaje es rápido, la retención se vuelve algo de menor importancia.
- Satisfacción subjetiva: hace referencia a cómo de cómodo se siente el usuario con el software. Esto es un criterio a medir, y depende del gusto individual del usuario y su experiencia.

En algunos casos, como es el caso de sistemas de interacción persona máquina, además se debería medir la respuesta en tiempo real del ordenador por cuestiones de retroalimentación: un sistema opera en tiempo real si el usuario no percibe un retraso entre su acción y la respuesta del sistema.

Además hay que tener en cuenta que todo sistema de interacción debería ser usable. La usabilidad, según la ISO 9241-11 "Requisitos ergonómicos para trabajos de oficina con pantallas de visualización de datos - Guías en usabilidad", define la usabilidad como

"La medida en la que un producto se puede usar por determinados usuarios para conseguir objetivos específicos con efectividad, eficiencia y satisfacción en un contexto de uso especificado."

La efectividad es el alcance de los objetivos especificados en la tarea: la precisión, la completitud y la corrección de la realización de las

tareas. Por eficiencia se entenderán los recursos empleados en relación con que los usuarios alcanzan los objetivos especificados como por ejemplo esfuerzo físico o mental. Y por último, la satisfacción es un factor subjetivo que se entiende como la ausencia de incomodidad y la actitud positiva del usuario en el uso del sistema. Son los usuarios los que deciden qué producto es fácil de usar y fácil de aprender.
Estos tres parámetros se ven afectados por los usuarios, los objetivos y el contexto de uso. Hay que considerar que los usuarios que utilizarán el sistema tienen habilidades físicas, cognitivas y preceptúales diferentes, y disparidad de niveles de experiencia, culturales, de edad, de personalidad etc. La usabilidad significa centrarse en los usuarios, por lo que hay que comprender los factores –psicológicos, ergonómicos, organizativos y sociales– que determinan cómo la gente trabaja y hace uso de los ordenadores [45].
Además, el usuario tiene unos objetivos que se personalizan en tareas a realizar con el sistema de interacción, es decir, el sistema de interacción suele ser un medio para poder realizar las tareas, no es la tarea en sí. Como bien dice Dumas y Redish [12], los usuarios son personas atareadas intentando realizar tareas y que utilizan los sistemas para ser productivos.
Y por último está el contexto de uso, eso significa que hay que conocer el dónde, el cómo y bajo que condiciones se va utilizar el sistema de interacción. Por ejemplo, un sistema de reconocimiento del habla que puede ser muy útil en ambientes silenciosos y en situaciones donde tenemos las manos ocupadas, puede ser muy poco útil en la calle con gente.
La definición de la ISO, menciona tres parámetros a medir: la eficiencia, la eficacia y la satisfacción. Sin embargo, otros autores incluyen más parámetros como la accesibilidad, la experiencia de usuario o más conocido como *user experience* o UX, la fiabilidad, la seguridad, la resistencia, la privacidad, la confianza [36] [37], la utilidad [22] o la estética [51].
Para el diseño y desarrollo de sistemas interactivos, Gould y Lewis [21] describen cuatro principios para incluir la usabilidad en todo el proceso:

1. Concentrarse pronto y de forma continua en los usuarios
2. Integrar consideraciones en todos los aspectos de usabilidad
3. Probar el funcionamiento del sistema de forma continua con prototipos.
4. Iterar el diseño

El uso de estas consideraciones nos ayudará a desarrollar sistemas interactivos y funcionalidades adaptadas al usuario. El prototipado rápido

es un sistema de trabajo apropiado para el diseño de interfaces persona-ordenador [38]. Trabajar con prototipos en etapas tempranas del diseño y desarrollo, donde el usuario puede opinar, permite obtener sistemas finales que necesitan menos coste en mantenimiento, los usuarios requieren menos tiempo de aprendizaje y la calidad del sistema es mayor.

3. Visión por computador: conceptos

La visión por computador (VPO) o visión artificial es el proceso por el cual se extrae información a partir de imágenes. La VPO intenta emular la capacidad visual del ser humano, ya que se pretende efectuar tareas que el ojo humano es capaz de realizar como por ejemplo detectar un objeto en particular o interpretar un gesto de una persona. Nevatia [35] hacía referencia al término visión de máquina, como la construcción de máquinas capaces de percibir e interpretar el entorno.

La visión artificial se ha utilizado normalmente en campos como el de la industria, la medicina, la defensa militar o la tele-medición para realizar tareas de inspección, clasificación, guiado o imágenes médicas. En los últimos años, los campos de aplicación de la VPO se han ampliado y entre ellos está teniendo mucho auge la interacción, sobre todo para videojuegos.

La VPO puede verse afectada por la gran variabilidad existente en la apariencia humana o de los objetos, por las distorsiones de la cámara y las condiciones del entorno como es la iluminación o el fondo, aunque tiene el potencial de ofrecer una interacción natural e intuitiva con el ordenador.

El análisis y reconocimiento del movimiento humano y los gestos en tiempo real puede ser muy útil en un rango muy amplio de aplicaciones, desde controlar una silla de ruedas [26] a interaccionar con videojuegos.

Turk and Kölsch [52] comentan las posibles funcionalidades que pueden ofrecer las VBIs:

- Presencia y localización: para saber si hay alguien en el ángulo de vista, cuanta gente y su situación (en 2D o 3D).
- Identidad: para reconocer quienes son las personas que aparecen en la imagen.
- Expresión: detecta y entiende la expresión de la cara que pone el usuario. A esto se le considera como un campo más especializado dentro de las VBI denominado reconocimiento de gestos de la cara.

- Foco de atención: detecta hacia donde mira una persona, sobre donde tiene su foco de atención.
- Postura del cuerpo y movimiento: controla la postura general y el movimiento del cuerpo.
- Gestos: comprende la semántica significativa de los movimientos de la cabeza, extremidades y cuerpo del usuario
- Actividad: se responsabiliza del análisis de la imagen para saber que tipo de actividad está llevando a cabo el usuario.

Cuando se utilizan gestos o movimiento humano para interacciones, la VBI se centra en un conjunto de tareas que tienen como objetivo detectar, hacer el seguimiento o reconocer partes del cuerpo tales como la cara, las manos, los ojos o cualquier otra parte del cuerpo.
Las fases de procesamiento de las imágenes son comunes en las interfaces que utilizan visión, independientemente del tipo de aplicación de interacción que resultará del uso de técnicas de visión por computador. Estas fases son las siguientes [10][52]:

1. Detección: determinará una salida binaria que significa la presencia o ausencia del objeto buscado. Por ejemplo, detectar si hay una cara o una mano en la imagen.
2. Identificación o reconocimiento: que significará detectar un modelo de un objeto en los datos observados, teniendo en cuenta posición y orientación del objeto, y consiguiendo unos resultados con un cierto rango de confianza que define cómo de buena es la correspondencia. Por ejemplo entender un gesto específico del usuario como abrir la mano. Un caso especial en el reconocimiento es la verificación o autenticación, que juzga si un dato de entrada corresponde a una identidad particular con una gran confianza.
3. El tracking o seguimiento significa localizar objetos y determinar el cambio de pose a lo largo del tiempo. El tracking normalmente se considera una iteración fotograma a fotograma de la detección de un objeto, pero normalmente implica más que un procesado discontinuo. Para mejorar el tracking y añadir una continuidad temporal, una predicción para limitar el espacio posible de soluciones y aumentar la velocidad de procesado se utilizan filtros como regresiones lineales o filtros de Kalman para modelar la progresión temporal de un objeto.

Una postura es una configuración estática de las articulaciones y músculos –localización y orientación– del cuerpo humano como por ejemplo estar sentado. Los gestos son posiciones dinámicas producidas

por el movimiento de las articulaciones y músculos del cuerpo o de partes del cuerpo y pueden ser consideradas secuencias consecutivas temporales de posturas. A los gestos faciales también se les conoce como expresiones faciales y su reconocimiento pasa normalmente por la detección y el seguimiento de las características faciales. Los gestos suelen tener dos intencionalidades: ofrecer información significativa o interaccionar con el entorno.

La apariencia de un objeto describe características como su color y su brillo en cada punto de su superficie, además tiene en cuenta la textura, la estructura de la superficie, la iluminación y el punto de vista.

Dependiendo de la aplicación, del usuario y de las características del entorno, puede interesar trabajar con el cuerpo humano como un todo, con una extremidad o con una parte más específica como una mano, un dedo o algún objeto. Para ello se utilizamos tres posibles estrategias [53]:

- Estrategia *coarse-to-fine*: proceso que va desde una región mayor a una zona de interés más detallada. Cada paso de la secuencia se beneficia del paso anterior. Por ejemplo, para detectar las fosas de la nariz para el seguimiento, primero se podría detectar la cara, después la región de la nariz y por último detectar las fosas nasales.
- Estrategia *fine-to-coarse*: secuencia de trabajo que detecta zonas detalladas para encontrar un objeto o una región más extensa. Un ejemplo es la detección de caras, donde se pueden localizar características como los ojos, la boca o las fosas, y relacionarlas para detectar una zona mayor como la cara.
- Métodos basados en el conocimiento: modelando la escena en la imagen es más fácil conocer las relaciones entre elementos. Por ejemplo, realizar el seguimiento de la cara y de las manos cuando sólo nos interesa la información de la cara. En este caso, las oclusiones de manos tapando la cara no tendrían consecuencias no controladas.

De forma más detallada, en Mitra y Acharya [33] se puede encontrar un recopilatorio de técnicas para reconocer gestos.

Se han propuesto diferentes formas de clasificar las interfaces basadas en técnicas de visión por computador. Si utilizamos el sistema de clasificación a través de la intensidad del movimiento [40], primero se encuentran los sistemas de seguimiento de los pequeños movimientos como lo que pueden generar los ojos.

La mirada tiene un gran poder comunicativo entre las personas. Con los ojos se ve el entorno y lo que nos rodea. Además dan señales a los demás de donde está nuestro foco de atención, apuntamos con ellos, y

dan nociones de nuestro comportamiento (de si pensamos, mentimos, etc.).

El seguimiento de ojos hace referencia a un conjunto de tecnologías que permiten monitorizar hacia donde mira y pone su atención el usuario [24]. Ejemplos de su aplicación en la disciplina de la interacción pueden ser su utilización como dispositivo de entrada para personas con discapacidad, dentro de entornos de realidad virtual, enriquecer la interacción del usuario con avatares –permitiendo al avatar conocer el foco de atención del usuario– o reconocer comportamientos del usuario.

La mayoría de los sistemas actuales utilizan iluminación infrarroja para detectar los ojos del usuario, aunque también existen soluciones con cámaras web estándar. Además, la cámara puede estar colocada sobre la cabeza del usuario por lo que se le ofrece al usuario mayor flexibilidad de movimiento (por ejemplo para moverse en un mundo virtual) o en la estación de trabajo, lo cual ofrece mayor comodidad al usuario ya que son menos intrusivos.

Surakka et al. [49] resumen las principales ventajas de la mirada como entrada a un sistema:

- Ancho de banda: se amplía el ancho de banda de la entrada
- Esfuerzo: no requieren un esfuerzo consciente importante
- Velocidad: la mirada es un sistema más rápido que el uso de las manos para señalar
- Manos-libres: ofrecen un sistema que no requiere el uso de las manos
- Diversión: aporta una vertiente divertida
- Aprendizaje: la interacción es fácil de aprender

Este tipo de sistemas también presentan una serie de dificultades como con el llamado toque de Midas, es decir, que el usuario esté mirando algo que realmente no quiere seleccionar.

Además, Edwards [15], recomendaba que estos sistemas tuvieran una calibración automática, que no fuera necesario movimientos no naturales y que reconociera modos del usuario como el estar pensado o prestando atención a algo en particular.

Si ampliamos el movimiento, se encuentran los sistemas basados en el movimiento de manos o cabeza. Por ejemplo, las caras y las cabezas ofrecen gran cantidad de información en la interacción persona-persona de por ejemplo el foco de atención o la identidad de los interlocutores. El uso de la cabeza o de alguna región de la cara tiene aplicaciones muy diversas como:

- Dispositivo de entrada al ordenador para reemplazar todos los eventos del ratón [54] [5].
- Jugar a videojuegos [6]
- Control del foco de la aplicación según el monitor y la aplicación activa en sistemas multipantalla [2]
- Reconocimiento de expresiones faciales [9] (ver Figura 3) o para ejecutar ciertas operaciones basándose en la interpretación de algún movimiento de cabeza como afirmar o negar [44]
- Control de dispositivos como sillas de ruedas [26]
- Biometría, reconocer o identificar de forma unívoca una persona

El uso de las manos se puede utilizar en gestos como apuntar y en diferentes aplicaciones como por ejemplo [40]:

- Dispositivo de entrada al ordenador para reemplazar sistemas tradicionales: reemplazar un ratón [1], para navegar en una presentación, mando de control para televisiones, equipos estéreo o luces etc.
- Jugar con videojuegos [30] (ver Figura 4)
- Computación ubicua para controlar sistemas
- Manipulación de objetos: de forma colaborativa
- Reconocimiento de lenguaje de signos [56]
- Sistemas médicos [56]

Figura 3. Reconocimiento de expresiones faciales

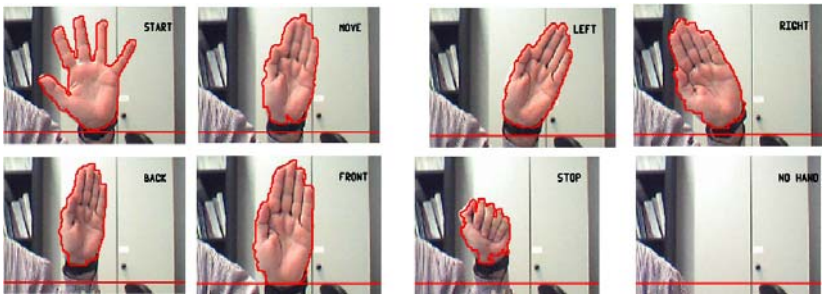


Figura 4. Ejemplo de navegación en un mundo 3D

Y por último tenemos los sistemas de interacción que utilizan movimientos más amplios de todo el cuerpo o de las extremidades superiores o inferiores para su uso en videojuegos, rehabilitación [8] o análisis del movimiento humano para entrenamiento deportivo.
Además, hoy en día en la industria del entretenimiento, las interfaces gestuales se están popularizando e importantes empresas están invirtiendo en producir videoconsolas y videojuegos que trabajan con este sistema de interacción sin ningún tipo de mando: por ejemplo, Microsoft Kinect o Sony Eye-Toy.

Unas librerías que nos pueden ayudar a programar interfaces basadas en visión son las de Open Source Computer Vision, OpenCV. Es una librería con funciones de programación para visión por computador y descargable en http://sourceforge.net/projects/opencvlibrary/ [7]. Contiene gran cantidad de algoritmos implementados que pueden ser utilizados en un abanico amplio de aplicaciones.

4. Estudio de caso: SINA (Sistema de Interacción Natural y Avanzado)

El SINA es un sistema de acceso que tiene como objetivo permitir la interacción con el ordenador a personas con un movimiento funcional limitado en manos o brazos, y que no pueden hacer un uso efectivo de los dispositivos de entrada tradicionales al ordenador. Se buscaba el ofrecer un sistema que llegase al máximo número de personas, por lo que el sistema tenía que ser:

- Un sistema de bajo coste
- Un sistema no intrusivo
- Un sistema normalizado
- Un sistema que pueda trabajar con condiciones de entorno normales

Para obtener un sistema de bajo coste se optó por desarrollar una aplicación que utilizase una cámara web USB estándar. Son periféricos asequibles y cuya presencia cada vez es más frecuente en las estaciones de trabajo. Además, la aplicación de software desarrollada es gratuita para el usuario final y se puede descargar a través de Internet para llegar a mayor público (http://sina.uib.es).
El uso de sensores, marcas o elementos sobre el usuario como pueden ser los licornios, las varillas de boca o sistemas de seguimiento de ojos con soporte cefálico, son incómodas para el usuario. Para obtener un sistema no intrusivo, se optó por utilizar técnicas de visión por computador que no necesitasen ningún elemento sobre el usuario.
Además, este hecho, favorecía la normalización del usuario delante del ordenador. La normalización significa permitir a las personas con algún tipo de discapacidad llevar una vida tan próxima a los colectivos considerados "normales" como sea posible [4]. El usuario con discapacidad se sienta delante del ordenador como cualquier otro usuario sin discapacidades.
Cuando una persona se sienta delante del ordenador, se asume que la cara está visible para una cámara web colocada sobre la mesa o el monitor. Es por ello, que SINA detecta automáticamente la cara y características sobre la región de la nariz, realiza un seguimiento los movimientos de la nariz y los transforma en el movimiento del cursor del ratón. Para realizar acciones con el ordenador existe una botonera gráfica que incluye los eventos del ratón estándar: clic del botón izquierdo, doble clic del botón izquierdo, clic del botón derecho y arrastre. Para ejecutar estas acciones se utiliza el denominado "clic en espera": el usuario selecciona una acción situándose en el botón de evento apropiado y mantiene el cursor parado en esta posición hasta que se selecciona. A partir de la selección, en cualquier sitio de la pantalla (por ejemplo un botón o una ventana) donde se quede el cursor parado durante un tiempo predefinido se ejecuta la acción seleccionada.

4.1 Trabajos relacionados

Existen trabajos que buscan el mismo objetivo final, y entre ellos difieren las técnicas de visión utilizadas, las partes de la cara utilizadas y los parámetros de configuración. Los primeros trabajos eran sistemas de visión que entre otras funcionalidades comentaban la interacción como una posibilidad [6] [50] y utilizaban características globales de la cabeza o cara como la geometría o el color.

Existen otros sistemas basados en el seguimiento de características de la cara como la nariz o la boca y cuyos sistemas no han sido testeados con usuarios finales [23] [16] [39] [34] [13]. Ahora bien, es totalmente diferente diseñar una interfaz para un conjunto de usuarios específicos que diseñar un sistema de uso generalizado. El sistema Camera Mouse de Betke et al. [5] fue el primer sistema que se probó con usuarios con discapacidad. El Camera Mouse aprende una zona de la cara –nariz, boca, ceja– y busca regiones parecidas en cuanto a brillo y color en fotogramas posteriores. Después otros sistemas como el de Gorodnichy [20] que hace el seguimiento de la punta de la nariz, Mauri et al. [32] que funciona por color o Kjeldsen [27] y Perini et al. [42] que utilizan correspondencia de patrones, también realizaron pruebas con usuarios con discapacidades.

4.2 Diseño

El diseño se realizó utilizando prototipos. Partiendo de la base de los cuatro requerimientos iniciales: bajo coste, entorno normalizado –para el usuario y las condiciones del entorno– y no intrusivo, se analizaron, se observaron y se trabajó con usuarios finales para refinar el conjunto de prototipos que se iban presentando.
Se colaboró con una asociación de usuarios con parálisis cerebral durante dos años para obtener el resultado final del sistema. En Manresa et al. [31] se detalla el proceso de mejora del sistema en base al trabajo con los usuarios finales. A continuación se listan los requerimientos del sistema incorporados en la versión final:

- R1. No es intrusivo.
- R2. Es de bajo coste.
- R3. Trabaja con el movimiento de cabeza del usuario.
- R4. Trabaja bajo condiciones normales de entorno.
- R5. Puede ejecutar todos los eventos del ratón.
- R6. La posición de la cámara web es flexible.
- R7. Es totalmente automático.
- R8. Se tiene en consideración el control cefálico del usuario: rango de movimiento y capacidad de mantener la cabeza fija en una posición durante un tiempo predeterminado. Existe un fichero de configuración por cada usuario con las siguientes características:
    - Tiempo de click: tiempo necesario para mantener la cabeza fija para ejecutar un evento.
    - Rango de click: área alrededor del puntero del ratón donde se considera que el puntero está quieto.

    - Desplazamiento en x e y. Constantes utilizadas en la función de transformación que modelan el rango de movimiento de cabeza del usuario.
- R9. La imagen mostrada en la ventana de procesamiento tiene que ser coherente con el movimiento del usuario (imagen espejo) para no confundir al usuario.
- R10. Tiempo real: la retroalimentación tiene que ofrecerse en tiempo real.
- R11. La interfaz tiene que informar al usuario de su estado continuamente.
- R12. La barra gráfica de eventos contiene iconos gráficos (metáforas) (ver Figura 5).
- R13. El uso del color es correcta según la percepción del color: evitando colores contrarios. Este requerimiento aún tiene que desarrollarse.
- R14. La barra gráfica de eventos puede adaptar diferentes ubicaciones en la pantalla.
- R15. La ventana de inicio donde se detecta al usuario tiene que desaparecer tan pronto como se haya detectado el rostro del usuario. Esto es debido a que los usuarios se centran más en su imagen que en las tareas a realizar.
- R16. Tiene que existir "información de contacto para el usuario". El trabajar con sistemas de visión donde el usuario no toca físicamente ningún dispositivo puede hacer que se pierda el enlace usuario-sistema.
- R17 Se añaden parámetros de configuración al perfil del usuario: posición de la botonera y evento inicial seleccionado (para aquellos usuarios que sólo utilizan un único evento).

Figura 5: Interfaz gráfica de usuario. A la derecha se muestra la botonera gráfica de selección de Eventos.

- R18. La ventana de selección de usuario sólo aparecerá al inicio de la aplicación si está configurado (para su uso en asociaciones, centros etc.).
- R19. Se deberá hacer el seguimiento sólo de características en la región de la nariz. Esto es para aportar al sistema robustez debido a los posibles fallos que pueden generar los sistemas de visión.

4.3 Implementación (aspectos técnicos)

La aplicación del SINA se desarrolló en C++, las librerías de visión por computador OpenCV y las DirectX para el control de la cámara.
Desde un punto de vista técnico, descrito en Varona et al. [54] y Manresa-Yee et al. [29], para conseguir una interfaz de usuario fácil y amigable, el sistema es totalmente automático y está compuesto por dos módulos principales: Inicialización y Proceso, que se relacionan entre sí para poder recuperarse de posible errores (ver Figura 6).
El módulo de Inicialización es el responsable de extraer las características faciales del usuario. Esta fase localiza su cara y extrae las mejores características faciales sobre la región de la nariz para realizar el seguimiento. La fase de Proceso se encarga de hacer el seguimiento de las características faciales, recuperarse de la pérdida de éstas y de enviar el evento y la posición del ratón al sistema operativo [41].

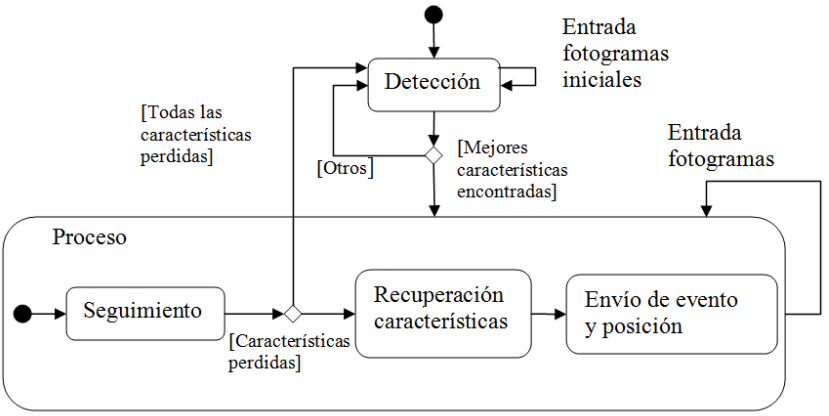


Figura 6. Funcionamiento del SINA

4.3.1. Inicialización

Para desarrollar una interfaz fácil para el usuario, ésta tiene que ser lo más automática que se pueda. Por tanto, el requerimiento para iniciar el proceso es que el usuario se encuentre parado mirando hacia el frente durante un segundo más o menos para que el algoritmo de Viola y Jones [55] le detecte la cara. Necesitamos que el usuario esté parado durante un tiempo predefinido para entender que realmente quiere activar el sistema (ver figura 7a).
Una vez detectada la cara, por medidas antropométricas de los humanos, ésta se puede dividir en tres regiones: ojos y cejas, nariz y boca. Nos centramos en la región de la nariz porque es la parte más centrada de la cara, siempre está visible en todas las posiciones de la cara mirando hacia la pantalla (incluso permite leves rotaciones), y no está ocluida por ninguna barba, bigote o gafas. Sobre la región de la nariz, buscamos las mejores características, aquellas con un cambio importante de intensidad para realizar el seguimiento [48]. Idealmente estas características deberían ser las fosas nasales y los bordes de la nariz, ya que nos interesa tener puntos a ambos lados de la nariz para que su media esté centrada en el centro de la nariz. Debido a la iluminación es posible que se seleccionen puntos inestables (ver figura 7b), por ello se seleccionaran pares de características verticalmente simétricas (ver figura 7c). Finalmente el punto que se transformará en posición del puntero es el dado por la media de las características seleccionadas (ver figura 7d). De esta forma, un movimiento de la cabeza hacía la derecha provocaría un movimiento del cursor hacia la derecha de la pantalla, por tanto, el usuario solo debe mover su cabeza en la dirección en la que quiere desplazar el cursor por la pantalla.

4.3.2 Proceso

En el módulo de Proceso se hace el seguimiento de las características detectadas a través del algoritmo de Lucas y Kanade [28]. Ahora bien, debido a cambios de iluminación o movimientos bruscos del usuario puede ser que las características de las cuales se hace el seguimiento se desplacen o se pierdan. Si se desplazan una distancia mayor a un valor predefinido, entonces esa característica se elimina.

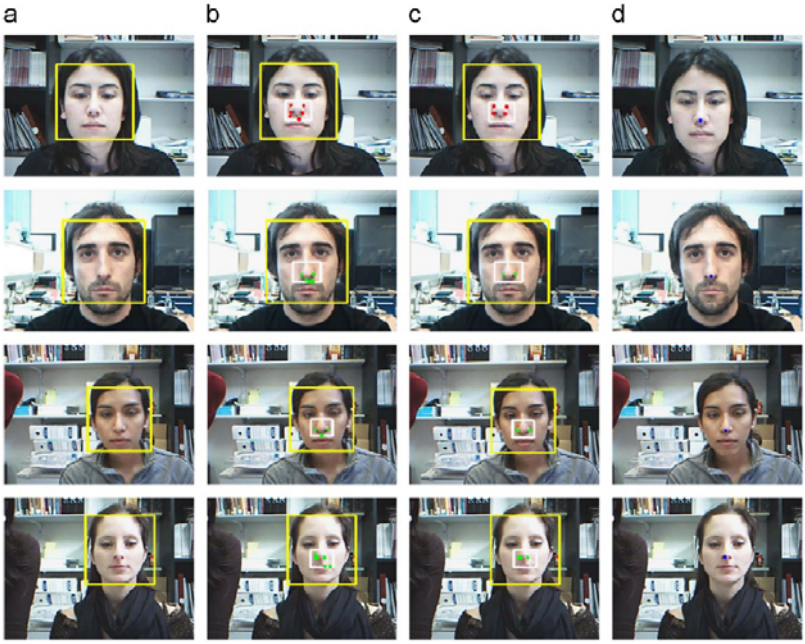


Figura 7. (a) Detección automática de la cara. (b) Conjunto inicial de características. (c) Punto final considerado: punto de referencia.

En el momento en que se pierdan todas las características –por ejemplo, por la no presencia del usuario– el sistema se reiniciará automáticamente esperando detectar el rostro del usuario. Puede ocurrir que las características se desplacen levemente pero no lleguen a perderse. En estos casos, el resultado es que el punto medio no estará centrado sobre la nariz, por lo que el mover el puntero del ratón se dificulta. Para solventar este problema, el sistema detecta cuando el usuario está mirando hacia el frente, detecta características nuevas sobre la región de la nariz, y modifica un porcentaje predefinido de características seguidas por nuevas. Esto provoca que si el punto medio se ha desplazado en algún momento, se recupere el centrado sobre la nariz de forma totalmente transparente al usuario para prevenir y corregir este tipo de errores. Esta característica es muy importante porque garantiza un uso continuo de la aplicación sin provocar la frustración del usuario por tener que realizar una continua inicialización del sistema.

La aplicación es flexible a pequeñas rotaciones y desplazamientos de la cabeza. En la Figura 8 se muestra una secuencia de funcionamiento de la aplicación, donde el punto de referencia (en azul) se mantiene correctamente localizado en todo momento.

Figura 8. Ejemplo de funcionamiento de la aplicación.

4.3.3 Envío de eventos y posición

A través del seguimiento de los movimientos de la nariz, el sistema controla el movimiento del cursor. La precisión necesaria tiene que ser suficiente para controlar el movimiento del cursor y posicionarlo sobre el lugar deseado de la pantalla. La transformación de la posición de la nariz de la imagen capturada por la cámara web a la posición del cursor en la pantalla se puede hacer de dos formas: absoluta o relativa.

Si se trabaja con posiciones absolutas, la posición se traduciría directamente sobre la pantalla, pero esto requeriría un seguimiento muy exacto y preciso, ya que un error pequeño de seguimiento en la imagen supondría un error magnificado en pantalla. Por esta razón, se utiliza el movimiento relativo para controlar el movimiento del ratón, que no es tan sensible a la precisión del seguimiento. Así, cuando el usuario quiere mover la posición del ratón a un lugar en particular, simplemente ha de mover la cabeza en la dirección deseada.
Esta función de transformación tiene en cuenta el desplazamiento en píxeles del la nariz en fotogramas con instantes de tiempo consecutivos (n_t , n_{t-1}), la posición actual del puntero del ratón (s_{t-1}) y unos parámetros configurables (α) que dependen de las capacidades de control cefálico del usuario.

$$s_t = s_{t-1} + \alpha(n_t - n_{t-1}) \text{ (Eq.1)}$$

Para conseguir una trayectoria del movimiento del puntero más suave, se utiliza el método del ajuste lineal.
La ejecución de eventos del ratón se hace a través de una botonera gráfica que contiene todos los eventos del ratón: el clic del botón izquierdo, el doble clic del botón izquierdo, el clic del botón derecho, el arrastre, y además las opciones de desactivar todos los eventos del ratón y de salir de la aplicación. Como hemos explicado anteriormente, el modo de funcionamiento es por medio de lo que se llama "clic en espera". El usuario tiene que posicionarse sobre uno de los eventos durante unos instantes, éste se selecciona y a partir de allí en cualquier sitio donde el cursor se quede parado se ejecuta el evento. La única excepción de funcionamiento es el evento de "Arrastre". Para ejecutar la operación de arrastre, el primer sitio donde se mantenga el cursor durante unos segundos, sería como pulsar el botón izquierdo del ratón y mantenerlo. La operación de soltar el botón izquierdo se realizaría cuando se mantuviera el cursor nuevamente sobre otra zona de la pantalla.

4.4 Evaluación

El sistema ha sido evaluado en diferentes entornos y con diferentes tipos de usuarios.

4.4.1 Evaluación del punto seleccionado

Para evaluar la eficacia de la detección de nariz, se utilizó la base de datos de caras BioID que contiene 1521 imágenes de 23 personas –

caras y hombros– en entornos naturales –diferentes iluminaciones y fondos [25]–. Las imágenes tienes puntos de la cara marcados manualmente, y entre ellos la punta de la nariz. Calculamos la distancia entre nuestros puntos de la nariz con los marcados manualmente y se observó un 95.79% de caras detectadas y en ellas, la nariz se detectó un 96.08% de las veces con precisión. El error en la detección de caras es debido a que algunos rostros de algunos participantes no están totalmente visibles o tienen una fuente de iluminación en un lado de la cara que causa fuertes sombras en el otro. Se observa un mayor error de pixeles en la distancia vertical (m: 4.98, sd: 4.86) respecto a la distancia horizontal (m: 2.34, sd: 2.05). Esto es debido a que en nuestro algoritmo las fosas nasales tienen una gran influencia en lo que consideramos la nariz, por lo que éstas se encuentran por debajo de la punta de la nariz.

4.4.2 Evaluación inicial en condiciones de laboratorio con usuarios sin discapacidad

Se realizó una evaluación en laboratorio con usuarios sin discapacidad para comprobar el funcionamiento del sistema antes de llevarlo a los usuarios finales. Se presentó a los usuarios una matriz de 5x5 círculos. Tenían un único intento por círculo y no había ningún orden. Si el usuario no acertaba al realizar el click izquierdo, se computaba la distancia en píxeles al círculo más cercano –dando a suponer que era el círculo que intentaba seleccionar–. Se realizó la prueba con 13 usuarios noveles y 9 usuarios que habían tenido una fase de entrenamiento. Los resultados de los usuarios con entrenamiento fueron del 97.3 de selecciones correctas y del 85.9 de los participantes sin entrenamiento. La media de distancia de error fue de 2 y 5 pixeles para entrenados y noveles.

4.4.3. Evaluación del sistema externa y junto con otros sistemas

Un grupo externo al equipo de desarrollo del SINA evaluó el sistema utilizando una guía de diseño, GEDIS, para analizar el diseño de la interfaz valorando factores como la calibración de la interfaz, la barra de eventos, el rango de movimiento o la retroalimentación para mejorar aspectos como el uso del color, la visibilidad o la ubicación de los elementos. Las recomendaciones fueron posteriormente incluidas en los manuales de uso y en la interfaz.
Además, el grupo externo combinó el uso del SINA con una interfaz de control domótico obteniendo resultados positivos en cuanto a su uso conjunto [43].

4.4.4. Evaluación inicial con usuarios con discapacidad

Se realizó una observación de seis usuarios con parálisis cerebral en sus tareas diarias. Los usuarios continuaban realizando actividades de sus currículos escolares, por lo que cada uno de ellos tenía tareas adaptadas a sus necesidades.
Se realizó un registro de cada sesión donde se guardaban datos del estado físico, el humor y la motivación del usuario, condiciones del entorno, los problemas surgidos, el funcionamiento del SINA y la interacción del usuario con el sistema.
Se evaluó durante un año escolar el funcionamiento y los problemas que iban surgiendo se solucionaban en nuevas versiones que se implantaban directamente en el centro.
Se hizo una evaluación cualitativa según lo observado por los terapeutas. Inicialmente el nivel de motivación de los usuarios era alto para todos ellos, y aunque no todos los usuarios presentaban dificultades con el uso del sistema, algunos problemas se presentaron en algunos de ellos como:

- Necesidad de ayuda verbal y física para que el usuario siguiera las instrucciones
- Movimientos cefálicos bruscos y sin coordinación
- Pérdida del foco debido a la falta de atención o debido a movimientos involuntarios
- Postura incómoda por tener el cuello muy flexionado
- Dificultad en activar eventos por no ser capaz de mantenerse estático
- Postura natural del cuello flexionada
- Fatiga

Al final del año de entrenamiento y junto con las modificaciones que se habían realizado al sistema, todos los usuarios eran capaces de dirigir el puntero hacia el lugar deseado y ejecutar el evento de click izquierdo. Todos trabajaban con una postura más adecuada y sin presentar un cansancio importante. Además se hizo una comparativa de una misma actividad entre el sistema de acceso anterior y el SINA realizando una misma actividad. Exceptuando las tareas de escritura –que se realizan con un teclado virtual y con al menos 1 segundo para el clic en espera– y cuando el usuario utilizaba un pulsador –que no es un dispositivo de puntero sino que escanea las opciones en pantalla y el usuario pulsa cuando está sobre la correcta– el SINA ofrecía una efectividad mayor. Para los detalles de la experiencia de cada usuario, el lector puede dirigirse a [31].

4.4.5 Evaluación final

Se hicieron dos evaluaciones finales con el SINA. La primera era con usuarios sin discapacidad donde se realizó la prueba recomendada por la ISO 9241-9 "Requisitos ergonómicos para trabajos de oficina con pantallas de visualización de datos- Requerimientos para dispositivos de entrada que no son teclados" donde se mide el *througput* (rendimiento) que es una medida que incluye la precisión y la velocidad en una prueba donde el usuario sólo tiene una oportunidad para pulsar sobre cada círculo (ver figura 9). El sistema se comparó con otro ratón comercial, el Crea Ratón [32]. Se obtuvieron resultados similares con ambos ratones, aunque el valor del *throughput* estaba muy alejado del ratón convencional. Hay que comentar que los ratones faciales requieren cierto tiempo para ejecutar eventos, el tiempo de click, y además los usuarios eran noveles en el uso de este tipo de sistema de interacción.
Finalmente los usuarios con discapacidad realizaron todos una prueba común y adaptada a todos los niveles cognitivos. Los resultados obtenidos demuestran la eficacia, la eficiencia y la satisfacción de los usuarios pero no nos permiten extraer ninguna estadística debido a que la mayoría de los usuarios realizaban la prueba sin ninguna estrategia, según sus habilidades y sin presión de tiempo.

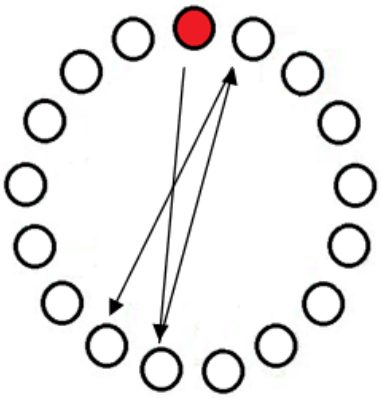


Figura 9. Prueba recomendada por la ISO 9241-9

5. Conclusiones

En este capítulo se ha hecho una introducción a las interfaces basadas en visión. Con más frecuencia encontramos este tipo de sistemas de interacción en diferentes productos comerciales, en trabajos de investigación y en gran diversidad de aplicaciones. No nos hemos adentrado en ninguna tecnología de visión en detenimiento debido a las diferentes posibilidades que se nos presentan, aunque se espera que la descripción del diseño, desarrollo y evaluación de un sistema en particular ayude a pensar en las dificultades y consideraciones a tener en cuenta con estos dispositivos de entrada.

6. Referencias

[1] A.A. Argyros, M.I.A. Lourakis (2006). Vision-Based Interpretation of Hand Gestures for Remote Control of a Computer Mouse. En: T.S. Huang et al. (Eds.): *Proceedings HCI/ECCV 2006*, LNCS 3979, Springer-Verlag Berlin Heidelberg, pp. 40–51.

[2] M. Ashdown, K. Oka, Y. Sato (2005). Combining head tracking and mouse input for a GUI on multiple monitors. En: *Extended Abstract CHI'05*, pp. 1188-1191.

[3] R.M. Baecker, W.A.S. Buxton (Eds.) (1987). Readings in human-computer interaction: A multidisciplinary approach. San Mateo, CA: Morgan Kaufmann Publishers.

[4] N. E. Bank-Mikkelsen (1975). El principio de normalización. Siglo Cero, 37 pp.16-21.

[5] M. Betke, J. Gips, P. Fleming (2002). The camera mouse: visual tracking of body features to provide computer access for people with severe disabilities. En: *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 10(1) pp.1-10.

[6] G. R.Bradski (1998). Computer vision face tracking for use in a perceptual user interface. En*: Intel Technology Journal*, Vol. 1, Num. Q2, pp. 1-15.

[7] G. Bradski, A. Kaehler (2008). Learning OpenCV Computer vision with the OpenCV library. O'Reilly.

[8] J.W. Burke, P.J. Morrow, M.D.J. McNeill, S.M. McDonough y D.K. Charles (2008). Vision Based Games for Upper-Limb Stroke Rehabilitation. En: *Conference Proceedings International Machine Vision and Image Processing Conference. IMVIP '08*. pp. 159-164.

[9] E. Cerezo, I. Hupont, C. Manresa-Yee, J. Varona, S. Baldassarri, F.J. Perales, F. Serón (2007). Real-Time Facial Expression

Recognition for Natural Interaction, En: *Lecture Notes in Computer Science 4478, IbPRIA 2007*, pp. 40-47.
[10] J. L. Crowley, J. Coutaz, F. Berard (2000). Things that See. En: *Communications of the ACM* 43 (3) pp. 54-64.
[11] D. Diaper (1989). The discipline of human-computer Interaction. En: *Interacting with computers*, núm. 1, vol. 1, Butterworth-Heinemann Ltd., Guildford, Reino Unido.
[12] J. Dumas, J. Redish (1999). A Practical Guide to Usability Testing. Intellect Books.
[13] G. C. De Silva, M. J. Lyons, S. Kawato, N. Tetsutani (2003). Human factors evaluation of a vision-based facial gesture interface. En: *Proceedings of the 2003 Conference on Computer Vision and Pattern Recognition Workshop*, 5 pp. 52-59.
[14] A. Dix, J. Finlay, G.D. Abowd, R. Beale (2004). Human-Computer Interaction. Third Edition. Pearson Prentice Hall.
[15] G. Edwards(1998). A tool for creating eye-aware applications that adapt to changes in user behaviours. En: *Assets '98: Proceedings of the third international ACM Conference on assistive technologies*. ACM Press, pp.67-74.
[16] L. El-Afifi, M. Karaki, J. Sorban (2004). Hands-free interface. a fast and accurate tracking procedure for real time human computer interaction. En: *Proceedings of the Fourth IEEE International Symposium on Signal Processing and Information Technology*, pp 517-520.
[17] T.T. Hewett, R. Baecker, S. Card, T. Carey, J. Gasen, M. Mantei, G. Perlman, G. Strong, W. Verplank (1992). "Acm sigchi curricula for human-computer interaction". http://sigchi.org/cdg/index.html. Último acceso: Septiembre 2011.
[18] Fundación Vodafone España (2005). Tecnologías de la Información y las Comunicaciones y Discapacidad. Dependencia y Diversidad.
[19] T. Granollers, J. Lorés, (2005). Introducción a la IPO en Ed. Granollers, T., Lorés, J., Cañas, J.J. Diseño de sistemas interactivos centrados en el usuario. Editorial UOC.
[20] G. Gorodnichy, R. Malik, G. Roth. (2004). Nouse 'use your nose as a mouse' perceptual vision technology for hands-free games and interfaces. En: *Image and Video Computing*, 22(12) pp. 931-942.
[21] J.D. Gould, C. Lewis (1985). Designing for usability: Key principles and what designers think. En: Communications *of the ACM*, 28(3):300-311.
[22] J. Grudin (1992). Utility and usability: Research issues and development contexts. En*: Journal Interacting with Computers*, 4(2) pp. 209-217.

[23] J. Hannuksela, J. Heikkilä, M. Pietikäinen (2004). Human-computer interaction using head movements. En: *Proceedings Workshop on Processing Sensory Information for Proactive Systems* (PSIPS 2004), pp 30-36.
[24] R.K.J. Jacob, (1990). What you look at is what you get: Eye movement-based interaction techniques. En: *Proceedings CHI 1990*, pp. 11-18.
[25] O. Jesorsky, K. Kirchberg, and R. Frischholz (2001). Robust face detection using the hausdorff distance. En: *Lecture Notes in Computer Science*, 2091 pp.90-95.
[26] J. S Ju, Y.Shin, E.Y. Kim (2009). Intelligent wheelchair (IW) interface using face and mouth recognition. En: *Proceedings of the 13th international conference on Intelligent user interfaces*, ACM, New York, pp. 307-314.
[27] R. Kjeldsen (2006). Improvements in vision-based pointer control. En: *Proceedings of ACM SIGACCESS conference on Computers and accessibility*, ACM Press, pp. 189-196.
[28] B.D. Lucas, T. Kanade (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. En: *International Joint Conference on Artificial Intelligence*, pp. 674-679.
[29] C. Manresa-Yee, J. Varona, F.J. Perales, (2006). Towards hands-free interfaces based on real-time robust facial gesture recognition. En: *Lecture Notes in Computer Science 4069, AMDO'06*, pp. 504-513.
[30] C. Manresa-Yee, J. Varona, R. Mas, F.J. Perales, (2005). Hand Tracking and Gesture Recognition for Human-Computer Interaction, En: *Electronic Letters on Computer Vision and Image Analysis* 5(3) pp.96-104.
[31] C. Manresa-Yee, P. Ponsa, J. Varona, F.J. Perales (2010). User experience to improve the usability of a vision-based interface. En: *Interacting with Computers* Volume 22, Issue 6, pp. 594-605
[32] C. Mauri, T. Granollers, J. Lores, M. García (2006). Computer vision interaction for people with severe movement restrictions. En: *Human Technology: An Interdisciplinary Journal on Humans in ICT Environments*, 2(1) pp.38-54.
[33] S. Mitra, T. Acharya (2007). Gesture recognition: a survey. En: *IEEE transactions on systems, man, and cybernetics-Part C: applications and reviews*, vol. 37, No.3, pp.311-324
[34] T. Morris, V. Chauhan (2006). Facial feature tracking for cursor control. En: *Journal of Network and Computer Applications*, 29 pp. 62–80.
[35] R. Nevatia (1982). Machine perception. Prentince-Hall Inc.
[36] J. Nielsen (1993). Usability Engineering.

[37] J. Nielsen, J. Levy (1994). Measuring usability: Preference vs. performance. En: *Communications of the ACM*, 37(4):66-75
[38] S.P. Overmyer (2002). Revolutionary vs. evolutionary rapid prototyping: Balancing software productivity and hci design concerns. En: *Proceedings of the Fourth International Conference on Human-Computer Interaction*, pp. 303-307
[39] T. Palleja, W. Rubion, M. Teixido, M. Tresanchez, A. Fernandez del Viso, C. Rebate, J. Palacin, J., (2009). Using the optical flow to implement a relative virtual mouse controlled by head movements. En: *Journal of Universal Computer Science*, 14(19) pp. 3127-3141.
[40] M. Pasch, N. Bianchi-Berthouze, E.M.A.G. van Dijk, A. Nijholt, (2009). Immersion in movement-based interaction. En: *LNCS Intelligent Technologies for Interactive Entertainment Third International Conference*. Volume 9, Part 2, Springer Verlag pp. 169-180
[41] F.J. Perales, J. J. Muntaner, J. Varona, F. Negre, C. Manresa-Yee (2008). SINA, Sistema de interacción natural avanzado, el ordenador al alcance de todos. Premi d'investigació del Consell Econòmic i Social 2008. CES Illes Balears
[42] E. Perini, S. Soria, A. Prati, R. Cucchiara (2006) Facemouse: A human-computer interface for tetraplegic people. En: *ECCV Workshop on HCI, volume 3979 of Lecture Notes in Computer Science*, Springer, pp 99-108
[43] P. Ponsa, M. Díaz, C. Manresa-Yee, B. B. Amante (2008). Diseño ergonómico de interfaz gráfica y uso de interfaz de manos libres en simulación de tareas domóticas. En: *Proceedings Interaccion 2008*, pp. 197-206.
[44] M. Porta (2002). Vision-based user interfaces: methods and applications. En: *International Journal of Human-Computer Studies*, 57(1) pp. 27-73
[45] J. Preece (1994). Human-computer interaction. Addison-Wesley, Reading, MA
[46] B. Shneiderman (1982). The Future of Interactive Systems and the Emergence of Direct Manipulation. En: *Behaviour and Information Technology*, 1 (3) pp. 237-256.
[47] B. Shneiderman (1998). Designing the user interface: strategies for effective human-computer interaction. Addison-Wesley, 3rd ed edition
[48] J. Shi, C. Tomasi (1994). Good features to track. En: *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. Seattle, pp. 593–600.

[49] V. Surakka, M. Illi, P. Isokoski (2004). Gazing and frowning as a new human-computer interaction technique. En: *ACM Transactions on applied perception*, 1 (1), pp.40-56
[50] K. Toyama (1998) Look, ma–No hands! Hands-free cursor control with real-time 3D face tracking. En: *Proceedings of the workshop on perceptual user interfaces (PUI)*. San Francisco, pp. 49-54.
[51] N. Tractinsky, A. S. Katz, D. Ikar (2000). What is beautiful is usable. En: *Interacting with Computers*, 13(2) pp.127-14.
[52] M. Turk, M. Kölsch (2005). Perceptual interfaces. En: G. Medioni and S.B Kang, editors, *Emerging Topics in Computer Vision*. Pearson Education
[53] M. Turk, M. Kölsch (2003) Perceptual interfaces. *UCSB Technical report* 2003-33, pp.1-43
[54] J. Varona, C. Manresa-Yee, F.J Perales (2008). Hands-free Vision-based Interface for Computer Accessibility, *Journal of Network and Computer Applications* Volume 31 , Issue 4, pp.357-374
[55] P. Viola, M. J. Jones (2004). Robust real-time face detection. *Int. Journal Computer Vision*, 57(2) pp.137-154
[56] J.P. Wachs, M. Kölsch, H. Stern, Y. Edan (2011) Vision-based hand-gesture applications. *Communications of the ACM*. Vol 54, Num.2, pp. 60-71

CAPÍTULO 5

Introducción a los Sistemas de Interacción Multitáctiles

Ramón Mas Sansó, Universidad de las Islas Baleares, España
Cristina Manresa-Yee, Universidad de las Islas Baleares, España

1. Introducción

A lo largo de la historia, los dispositivos de entrada, a diferencia de los dispositivos de visualización y a pesar de la indiscutible importancia que tienen en el campo de la interacción persona-ordenador, han pasado prácticamente desapercibidos en la literatura. Se han referenciado superficialmente y se ha descrito las características principales de estos dispositivos [1] [2], pero es difícil encontrar compendios que ofrezcan una visión detallada de estas tecnologías [3].

Para hacernos una idea de lo que han supuesto los avances en los mecanismos de entrada de información en un sistema informático basta con hacer un rápido recorrido por la historia.

La simple entrada de texto a partir de teclados de todo tipo se vio complementada con la aparición de los lápices ópticos (Whirlwind computer, MIT) usados a principios de los años cincuenta [4] que permitían seleccionar puntos en una pantalla de rayos catódicos (CRT). Estos lápices ópticos fueron posteriormente sustituidos por el ratón, creado en el Stanford Laboratory of Research en 1965 y comercializado por Xerox y Apple a principios de los años ochenta. Éste dispositivo de entrada fue crucial para el triunfo de las interfaces basadas en ventanas múltiples superpuestas (Macintosh 1984).

Complementando al ratón, caracterizado por proporcionar un posicionamiento relativo, se introdujeron las tabletas digitalizadoras, que permitían obtener coordenadas precisas de los puntos introducidos y que eran, por tanto, más preciadas por las aplicaciones que requerían mayor precisión.

Evoluciones lógicas del ratón fueron el paso de 2 a 6 grados de libertad, con versiones para obtener tanto coordenadas absolutas como relativas, y el trackball.
Para las primeras aplicaciones de ocio se introdujo el joystick y posteriormente, promovidos por el auge incipiente de la realidad virtual surgieron mecanismos de seguimiento (tracking) de múltiples partes del cuerpo humano con muy diferentes tecnologías (mecánicas, electromagnéticas, ópticas, videométricas, de ultrasonidos, inerciales, neuromusculares...). Los más conocidos por su rápida entrada en el ámbito familiar son el mando de la Wii, el playstation Move y la Kinect de la Xbox 360.
Con la introducción de los smartphone también se dio un fuerte empujón a las pantallas táctiles, una tecnología iniciada en los años setenta (Elograhp y AccuTouch) y que ha experimentado importantes avances en los últimos años.
En este capítulo nos centraremos precisamente en este último punto, en el estudio y desarrollo de las técnicas que permiten la interacción táctil con los sistemas informáticos, por el gran impacto que han supuesto en la sociedad como mecanismo alternativo de interacción persona-ordenador.

2. Los dispositivos de entrada táctil

En el campo de la interacción persona-ordenador, se pueden encontrar dos tipos de dispositivos para la entrada táctil, el panel táctil (o touchpad) y la pantalla táctil (o touchscreen). Aunque a veces la diferencia entre ambas es difusa, podemos establecer un criterio claro de separación, las pantallas táctiles están superpuestas a un dispositivo de visualización, mientras que los paneles táctiles no. Las confusiones aparecen cuando los paneles táctiles están montados sobre dispositivos de visualización secundarios.
El panel táctil lo creó George E. Gerpheide en 1988 y originariamente se conoció como Glidepoint y se extendió comercialmente cuando Apple lo adoptó para las series 500 de los powerbook, en el año 1995. El dispositivo se introdujo con el nombre comercial de trackpad y hasta hace poco sólo permitía la utilización de un punto de contacto. Las últimas versiones de paneles táctiles ya permiten la detección simultánea de diferentes puntos de contacto (11 en los powerbook) y hacer clic directamente sobre el hardware.
El sistema de funcionamiento se basa en la detección de variaciones en la capacitancia de los circuitos que incorpora debidas al contacto con los dedos. La detección de contactos múltiples ha permitido

ampliar enormemente la capacidad del panel táctil con la utilización de gestos para interactuar con el ordenador.
La pantalla táctil, como ya hemos comentado anteriormente, añade a la funcionalidad del panel la posibilidad de interactuar directamente con el display, por lo que el usuario tiene una visión global de sus gestos. La primera pantalla táctil se remonta a mediados de los años sesenta, y fue desarrollada por E.A. Johnson [5].
Las pantallas táctiles se han usado durante muchos años, sobre todo para paneles informativos (turismo, museos, divulgación...) y últimamente aparecen en multitud de dispositivos tecnológicos (teléfonos móviles, PDAs, tabletas, gps, electrodomésticos...).
La tecnología usada para la construcción de las pantallas táctiles es muy variada, en la siguiente sección las detallaremos y comentaremos las diferencias, ventajas e inconvenientes de cada una de ellas.

3. Tecnologías de construcción

En el mercado podemos encontrar pantallas táctiles de muy diversa tecnología, a continuación vamos a resumir las diferentes técnicas de construcción de este tipo de dispositivos.

- **Pantallas resistivas**: están formadas por un panel de cristal cubierto por dos capas conductoras ligeramente separadas. Cuando se activa el display, una corriente eléctrica circula por las capas y si el usuario toca la pantalla se produce un contacto entre ellas en este punto. Una circuitería específica detecta el cambio en el campo eléctrico y se calculan las coordenadas que se enviaran al sistema operativo del dispositivo para ser convertidas en una acción específica. La facilidad de construcción hace que sean las pantallas táctiles más asequibles y durante mucho tiempo han sido las más utilizadas. Tienen la ventaja de que pueden detectar contactos de cualquier tipo (funcionan por presión) y permiten la utilización de punteros o lápices para contactos de gran precisión. El gran inconveniente es la perdida de brillo debido a las múltiples capas. Pueden llegar a perder una cuarta parte de la luminosidad. Este tipo de pantallas se ha visto habitualmente en las ya casi extintas PDA, en muchos tablets-PC y en multitud de teléfonos móviles. El tipo de construcción dificulta también la detección de múltiples puntos de contacto, lo cual incrementa su precio y por consiguiente anula una de sus principales ventajas.

- **Pantallas capacitivas**: junto con las resistivas son, sin ningún tipo de duda, las pantallas táctiles más utilizadas actualmente. Se construyen a partir de un material aislante, habitualmente cristal, cubierto por un conductor (normalmente se utiliza óxido de indio y estaño ITO por su alto grado de transparencia, aproximadamente un 90%, comparado con el 75% de las pantallas resistivas). Cuando una parte del cuerpo humano toca la superficie, se produce una alteración del campo electrostático debido a nuestra conductividad. Esta alteración se puede medir como un cambio en la capacitancia y traducirse en la posición del contacto. El principal problema de este tipo de pantallas es que el contacto debe ser con un objeto conductor, no funciona con un lápiz ni con una mano con guantes. La principal ventaja es la claridad de la imagen, apenas alterada por la capa de material conductivo. Las pantallas capacitivas pueden detectar la capacitancia de superficie o la proyectada. En el caso de la capacitancia de superficie, la más habitual en los últimos años, introducida en los años ochenta por MicroTouch Systems, se aplica un campo electrostático uniforme y los sensores determinan las variaciones en la capacitancia a partir de las medidas que se toman en las cuatro esquinas del panel. Se obtiene una resolución limitada y debe calibrarse durante la fabricación. En la capacitancia proyectada se forma una malla de filas y columnas y la capacitancia puede cambiarse en cada intersección. Se puede medir para determinar exactamente la posición de contacto. Tienen más resolución y permiten la detección de múltiples puntos de contacto.
- **Superficies de ondas acústicas** (Surface acoustic wave SAW): ondas de ultrasonidos recorren la superficie de la pantalla y si se toca el panel se altera la amplitud y la velocidad de la onda, lo cual permite la detección del punto de contacto.
- **Pantallas de infrarrojos**: en este caso se montan parejas de emisores y receptores de luz infrarroja alrededor de la pantalla, una interrupción del flujo normal de la luz permite la detección del punto de contacto. La gran ventaja de este tipo de pantallas táctiles es que la interacción se puede hacer con cualquier dispositivo, sea el dedo, un puntero o un guante.
- **Pantallas dispersivas**: el contacto con la pantalla se detecta a partir de la energía mecánica que se ejerce al tocar la pantalla. El problema que tienen este tipo de pantallas, dejando a un lado la complejidad de los algoritmos de detección, es que una

vez detectado el punto de contacto inicial, el sistema no puede detectar un dedo estático.

4. Entrada multitáctil

Ya hemos visto que en algunos casos, las pantallas táctiles permiten la detección simultánea de varios puntos de contacto. Esto ha hecho posible el desarrollo de metodologías de interacción completamente diferentes, permitiendo al usuario una comunicación más natural e intuitiva con el ordenador. Los habituales gestos utilizados para seleccionar, arrastrar, rotar o escalar objetos visuales han evolucionado a versiones en las que el usuario utiliza ambas manos o varios dedos para realizar las mismas acciones, proveyendo de mecanismos que se han convertido en gestos estándares. Recuerdo que recientemente en una comida familiar, mi sobrino, de apenas 2 años, intentaba pasar las fotos de un marco digital (sin pantalla táctil) arrastrando el dedo, al estar el marco programado para cambiar de fotografía cada 5 segundos, era asombroso ver cómo el pequeño arrastraba el dedo hasta que la fotografía cambiaba, alternaba estos movimientos con otros para aumentar el tamaño de la imagen a partir de gestos que obviamente había aprendido a través del smartphone de sus padres y que no tenían ningún efecto sobre la estática imagen del marco. Me quedó bien patente que para un nativo digital la adopción de este lenguaje de gestos es más que intuitiva.

Está claro que el campo abierto por estos avances en la tecnología ha hecho posible un nuevo paradigma donde la naturalidad e intuición de los gestos está reemplazando los tradicionales sistemas de comunicación con el ordenador.

Los continuos avances en el hardware y la imparable expansión de este tipo de dispositivos han llevado a una importante reducción en los costes de producción y por tanto de comercialización. Aún así, estos costes continúan siendo prohibitivos cuando el tamaño de la pantalla aumenta. Por otra parte, los displays de grandes dimensiones también se han hecho más y más necesarios por el auge del trabajo cooperativo, donde dos o más personas interactúan en un mismo entorno. Han surgido nuevos términos para describir estos dispositivos, palabras como superficies (surfaces), mesas multitáctiles (multi-touch tables), *tabletops*, se han hecho habituales en este ámbito, de hecho han surgido nuevas técnicas que permiten implementar superficies multitáctiles a un menor coste y con una mayor versatilidad.

Con ligeros matices, se pueden utilizar tecnologías muy similares a las vistas anteriormente para construir mesas multitáctiles, pero debido a las nuevas circunstancias (mayor tamaño y por tanto mayor espacio) y necesidades (detección de un elevado número de puntos, detección de proximidad, detección de fiduciales) la utilización de cámaras y el posterior tratamiento de las imágenes ha hecho que la visión por ordenador pase a formar parte de este nuevo entorno.
Está claro que los sistemas multitáctiles han pasado a formar parte de nuestra vida cotidiana (teléfonos móviles, tabletas tipo ipad, consolas de videojuegos, cámaras fotográficas digitales...). Sin embargo, las superficies multitáctiles de grandes dimensiones (sean verticales u horizontales) continúan siendo caras para su uso doméstico, ya que las soluciones comerciales disponibles son de coste elevado.
Las primeras incursiones en interacción multitáctil utilizando cámaras se remontan a principios de los años 80 [6] y desde entonces no han dejado de evolucionar. La principal ventaja de la tecnología óptica es su bajo coste comparado a otros sistemas, que requieren instalaciones de calidad industrial para su construcción.

5. Superficies multitáctiles y educación

En esta sección se describirán experiencias de superficies multitáctiles y la educación, tanto para centros educativos como para centros con usuarios con necesidades especiales, que es el campo que se tratará en secciones posteriores.
En un contexto educativo, las mesas multitáctiles permiten a los estudiantes interactuar con objetos digitales en tareas colaborativas. La mesa comercial SMART [7] fue diseñada especialmente para la educación y permite a los profesores crear sus propias actividades. [8] presentaron la MET (Multitouch Education Table) para evitar los problemas de portabilidad, falta de modularidad y coste elevado de los sistemas comerciales (Smart table o Microsoft Surface [9]).
El proyecto SynergyNet tiene como objetivo el desarrollo de entornos docentes multitáctiles para mejorar los procesos de enseñanza-aprendizaje [10] [11] [12]. Otros artículos de investigación analizan las aplicaciones educacionales para las superficies multitáctiles [13] y presentan guías de diseño para aplicaciones de aprendizaje colaborativo para estos entornos [14] [15] utilizan juguetes para la interacción tangible.

En el caso de la educación especial, la tecnología puede proporcionar a los usuarios oportunidades de aprender, compartir información y

ganar independencia [16]. Se dedican esfuerzos para adaptar y desarrollar aplicaciones a las superficies multitáctiles para usuarios con limitaciones cognitivas y dificultades sociales. En [17] utilizan mesas multitáctiles para usuarios con síndrome de Down. [18] hacen lo propio para gente con síndrome de Asperger, en terapias de grupo para ayudar a mejorar las habilidades de trabajo en grupo.

Por otra parte, al trabajar con usuarios con necesidades especiales, a menudo en su rutina diaria en los centros de día o escuelas, las actividades cognitivas se alternan con actividades físicas. Revisando la bibliografía podemos encontrar aplicaciones para la rehabilitación, [19] presentan ejercicios en una Surface de Microsoft para usuarios con parálisis cerebral que necesitan estiramientos y mejorar su coordinación. La primera versión de tabletop de Microsoft era una superficie cerrada que no permitía el acercamiento de las sillas de ruedas. Una de las sugerencias de los autores para un trabajo futuro es permitir la inclinación y adaptación de la superficie al usuario.

En [20], los autores desarrollan un conjunto de aplicaciones basadas en mesas multitáctiles para la rehabilitación de los miembros superiores para gente mayor en un sistema AIR. La protección del espejo interior evita que se acerque una silla de ruedas y la postura de trabajo es más apropiada para una persona que trabaje de pie.

Hemos podido identificar trabajos pasados para construir mesas interactivas multitáctiles, para diseñar y desarrollar aplicaciones interactivas para educación, para necesidades especiales y para rehabilitación. Sin embargo, no hemos encontrado ninguna solución que se ocupe específicamente del diseño y desarrollo de este tipo de dispositivos accesibles para usuarios discapacitados. El diseño debería tener en cuenta un conjunto de requisitos que sobrepasen las barreras que encuentran habitualmente en los tabletops los usuarios con problemas de movilidad y accesibilidad.

A continuación vamos a describir una aproximación para la construcción de una mesa multitáctil de bajo coste pensando sobre todo en la seguridad de uso y en la accesibilidad y confort para usuarios discapacitados.

Este desarrollo forma parte de un proyecto conjunto con un centro para gente con parálisis cerebral, ASPACE [21] y previamente vamos a introducir las circunstancias en que se han construido para poder entender el contexto y las características de las mesas. ASPACE ofrece un centro de día y una escuela de educación especial donde los usuarios reciben actividades curriculares personalizadas. El término parálisis cerebral describe un grupo de desórdenes en el desarrollo motriz y postural que limitan las actividades y que se atribuyen a complicaciones ocurridas en el desarrollo fetal o en el cerebro infantil.

Estos desórdenes van frecuentemente acompañados por problemas cognitivos, de comunicación, de percepción, de comportamiento... [22].

Los terapeutas seleccionaron once usuarios con parálisis cerebral (algunos de ellos con parálisis cerebral profunda); seis adultos del centro de día y cinco niños de la escuela de necesidades especiales de edades comprendidas entre los 4 y los 28 años. Los usuarios que participan en el proyecto presentan limitaciones físicas y cognitivas importantes. Muchos utilizan sillas de ruedas y tienen posturas de trabajo diferentes: están derechos, inclinados hacia adelante o inclinados hacia atrás. La postura tiene que tenerse en cuenta puesto que se pretende proporcionar una experiencia de interacción confortable.

Algunos de los niños apenas tienen fuerza en sus miembros superiores y algunos de los adultos no la controlan, por lo que a veces pueden realizar movimientos descontrolados y provocar fuertes impactos sobre la mesa. Cognitivamente están aislados de lo que pasa en el entorno (situación más prevalente en los adultos). Algunos usuarios se ponen continuamente las manos en la boca, por lo que frecuentemente están húmedas. También debemos considerar estos factores a la hora de elegir los materiales para las mesas multitáctiles.

Los usuarios adultos también presentan comportamientos de autoestimulación, es decir, movimientos repetitivos del cuerpo que no tienen ninguna utilidad aparente, como movimientos repetitivos y balanceos. Estos movimientos, combinados con episodios de ansiedad, pueden conducir a las autoagresiones, mucho más graves, puesto que implican comportamientos destructivos que pueden tener consecuencias sociales y ponen en peligro la integridad física de la persona (golpearse duramente o morderse). Todos los usuarios muestran interés, en mayor o menor grado, cuando hay un estímulo significante en el entorno.

Es sabido que la estimulación temprana es un tratamiento útil y necesario para desarrollar el potencial social de cualquier persona en un entorno con riesgos [23]. Uno de los objetivos que perseguimos en nuestros trabajos es el desarrollo de un entorno interactivo multisensorial que permita una generación autónoma y controlada de estímulos significativos para este tipo de usuarios. Se quiere generar un espacio que agrupe equipos que permitan estimular los sentidos en un entorno controlado, tranquilo y seguro que son de gran beneficio para personas con necesidades especiales, ya que les permite divertirse, relajarse e inhibir los desordenes del comportamiento.

Los terapeutas, profesores y cuidadores quieren estimular a los usuarios para:

- Mejorar su relación con el entorno.
- Incrementar y controlar el movimiento intencionado (acción-reacción) en los miembros superiores cuando están sentados.
- Desarrollar habilidades motoras, perceptivas y cognitivas.
- Desarrollar habilidades sociales y de comportamiento cuando se producen cambios en el entorno.
- Contribuir con estimulación visual o auditiva adaptada a las necesidades e intereses del usuario cuando éste participa activamente.
- Contribuir con estimulación visual o auditiva adaptada a las necesidades e intereses del usuario para centrar su atención y reducir el aislamiento
- Reducir la autoestimulación

[24] [25] [26] ya establecieron que cuando estos usuarios reciben un estímulo significativo del entorno, las autoestimulaciones y las autoagresiones se reducen.
Estos estímulos puede proceder de la interacción autónoma (sin la ayuda de otras personas) con un ordenador para visualizar una presentación, videos u otras actividades.
Los terapeutas resaltan la importancia de ofrecer una retroalimentación al contacto de los usuarios puesto que en sus procesos de aprendizaje se sienten más cómodos cuando tocan directamente la pantalla que cuando usan otros dispositivos de entrada. Más aún, algunos usuarios no son capaces de utilizar ni de entender dispositivos de entrada como joysticks, trackballs o teclados.
Por esto, como parte del entorno multisensorial, que explicaremos en la próxima sección, hemos desarrollado una superficie multitáctil. Esta mesa multitáctil autónoma se instaló en las dependencias de ASPACE y se controla desde un teclado y un ratón inalámbricos. A pesar de que la mesa funciona perfectamente y cumple con los requisitos iniciales del proyecto, la retroalimentación de los terapeutas nos confirmó que algunos de los usuarios, que tenían problemas motores, fueron incapaces de usar correctamente la mesa, bien por accesibilidad o porque problemas posturales les impedían ver la pantalla y por tanto los estímulos.
Para solucionar estas limitaciones, estamos diseñando un segundo prototipo a partir de una mesa adaptada a las necesidades de los usuarios. Una de las mesas que habitualmente utilizan para actividades cotidianas, en este caso comer, fue el punto de partida.
Para alcanzar los objetivos particulares de cada usuario, la tabletop debe incorporar aplicaciones de acción/reacción. El feedback se obtiene en forma de efectos visuales impactantes y efectos sonoros

agradables, que dependen del tipo de contacto del usuario. Para usuarios con movimientos descontrolados y abruptos, los terapeutas buscan recompensar contactos largos, continuos y suaves. Para usuarios con reducidas habilidades motoras en los miembros superiores, los terapeutas definen regiones en la superficie que desencadenaran la reacción del ordenador. La retroalimentación debe motivar a los usuarios, por eso las aplicaciones deberán ser configurables según las preferencias en la música, imágenes, videos y demás contenidos de los programas.

6. Un entorno de estimulación sensorial

En el entorno interactivo que se ha diseñado [27] se recurre a medios tecnológicos para facilitar al terapeuta profesional herramientas alternativas para su trabajo en los momentos de atención indirecta, es decir, cuando el terapeuta no atiende únicamente a un solo usuario. Estos medios tecnológicos conforman un espacio controlado y seguro, con un equipamiento diseñado para la estimulación y la calma, ideales para la terapia de este tipo de tratamientos.

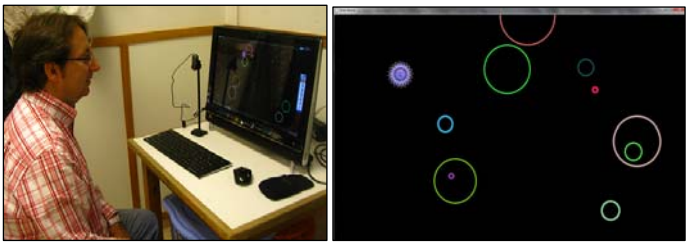
Las aplicaciones diseñadas ofrecen un alto componente de interactividad controlada por los movimientos de alguna parte del cuerpo del usuario recogidas por una cámara web. La respuesta en el entorno será en forma de estímulo sensorial (auditivo, visual y háptico) adaptado a las necesidades y capacidades de cada uno de los usuarios.

Los usuarios con discapacidad profunda pueden presentar dificultades a tres niveles diferentes en la interacción con el ordenador: durante la entrada, durante el proceso o durante la salida, sin que necesariamente sean exclusivos entre ellos. Es por ello, que en todos los casos se utilizan técnicas de visión por computador para reducir la intrusión de elementos sobre el usuario y minimizar las dificultades de accesibilidad cognitivas y físicas de la interacción.

Además de la mesa multitáctil se ha diseñado y desarrollado un entorno multimedia de interacción. El objetivo es común en ambos sistemas: la motivación y/o desmotivación de movimientos corporales localizados para establecer una relación causa/efecto. El objetivo de algunos usuarios es aumentar su movimiento corporal para mejorar su comunicación con el entorno. En cambio, el objetivo para un grupo importante de usuarios es el de reducir sus movimientos relacionados con las autoestimulaciones o autoagresiones.

En el entorno multimedia (figura 1) se utiliza un sistema basado en visión para motivar o desmotivar el movimiento de determinadas

partes del cuerpo. A través de una biblioteca de visión de código abierto y multiplataforma (OpenCV o Open Computer Vision) se capturan y tratan las imágenes obtenidas a través de una cámara web estándar.
En algunos casos puede requerirse un paso previo para la identificación, mediante una banda de color, de la parte del cuerpo del paciente con la que se quiere trabajar para poder detectarla y realizar su seguimiento con un algoritmo probabilístico.
A partir de las acciones que realiza el paciente y según el ejercicio que se trabaje, el programa ofrece una retroalimentación en forma de imágenes y/o sonidos. Si el sistema detecta que no se mueve una parte del cuerpo cuyo movimiento se pretende penalizar o que se mueve una parte cuyo movimiento se pretende motivar, se gratifica al usuario con imágenes o sonidos que le supongan un estímulo significativo positivo.



(a) (b)

Figura 1. (a) Configuración de los dispositivos hardware. El usuario interactúa a través de los movimientos de la cabeza capturados por la webcam. (b) Imagen que sigue una trayectoria, música que suena y círculos de grosores y colores diferentes que van apareciendo ante el movimiento o ausencia del movimiento del usuario

7. Caso de estudio: Diseño de una mesa multitáctil

Los requisitos iniciales son construir una mesa multitáctil que produzca estímulos significantes a los usuarios y que se engloba dentro del proyecto más ambicioso de construcción de un entorno interactivo multisensorial comentado en la sección anterior.
En los primeros contactos con los terapeutas se describieron algunos casos interesantes de cómo utilizar los estímulos y que ya han sido comentados en apartados anteriores: los estímulos podrían generarse de acuerdo con el tipo de contacto del usuario. También debían

considerarse algunas restricciones iniciales, entrada intuitiva para facilitar la tarea a los usuarios, bajo coste debido a las restricciones presupuestarias y seguridad debido al poco control del movimiento y presión de los gestos.
Aunque hay dispositivos multitáctiles comerciales que permiten estos usos, hay dos razones fundamentales que justifican la necesidad de un dispositivo adaptado. Primero, los dispositivos táctiles de mediano tamaño todavía requieren importantes inversiones y segundo, al basarse en dispositivos de salida tradicionales (como monitores LCD) son demasiado frágiles y en consecuencia poco seguros para trabajar en estos entornos. Podrían llevar a situaciones peligrosas y comprometer la seguridad de los usuarios.
En nuestro contexto, una mesa multitáctil de tecnología óptica encaja perfectamente.
Para diseñar un dispositivo de este tipo, deben tomarse algunas decisiones de diseño. Las tecnologías ópticas necesitan fundamentalmente una fuente de luz, un sensor óptico y un dispositivo de salida. Según se elijan estos componentes, optaremos por alguna de las técnicas de construcción más conocidas [28]:

- FTIR (Frustated Total Internal Reflection), desarrollada en 2005 por Jeff Han [29] y que se basa en un principio físico bien conocido: la luz que pasa de un medio a otro con un índice de refracción mayor y con un determinado ángulo queda atrapado y rebotando en su interior. Cuando un dedo contacta con el material, los rayos de luz se frustran en esa zona y se libera una cantidad de luz que puede ser detectada por una cámara. El problema principal de este tipo de tecnología es que se necesita una superficie especial para poder seguir correctamente el desplazamiento de los dedos. Otro inconveniente es que si bien los dedos se detectan perfectamente, no se puede trabajar con objetos, ya que no se pueden distinguir unos de otros.
- DI (Diffuse Illumination) o iluminación difusa, consiste en distribuir la luz de forma difusa sobre la superficie. Cuando un objeto contacta la superficie, se detecta la sombra que produce. La iluminación puede ser frontal o trasera, dependiendo de donde se sitúe la fuente de luz. Al detectarse las imágenes directamente, pueden distinguirse diferentes objetos sobre la superficie con lo cual es posible el uso de fiduciales. El problema con el que podemos encontrarnos es la dificultad de distribuir de forma homogénea la luz sobre toda la superficie y la posible influencia de la luz ambiental.

- LED-LP (LED-Light Plane), ilumina la superficie colocando emisores justo encima, haciendo que los objetos que se aproximan queden iluminados y sean fácilmente detectados. En este caso se pueden detectar objetos y dedos dentro de unos ciertos rangos de distancia.
- LLP (Laser Light Plane), funcionan de forma idéntica al LED-LP pero en este caso se recurre a la iluminación láser. La ventaja del láser es que produce una luz plana y no cónica y esto hace que únicamente se iluminen y detecten objetos muy cercanos al plano. El inconveniente es que trabajar con láser implica un cierto riesgo con el que deben tomarse las precauciones necesarias.

Las dos primeras son las técnicas más habituales aunque hay circunstancias que pueden llevarnos a utilizar alguna de las otras. Para una revisión exhaustiva de la construcción de mesas multitáctiles se pueden consultar [30] [31].

8. Construcción de una mesa multitáctil con tecnología óptica

La mesa multitáctil se ha construido utilizando un proyector DLP como dispositivo de salida para permitirnos la visualización de las imágenes sobre la superficie. El uso de proyectores es habitual, ya que con el mismo coste puede generar imágenes de tamaños muy diferentes y por tanto diseñar la mesa del tamaño que se requiera, lo único que debe tenerse en cuenta es que cuanto mayor queramos la imagen, más deberemos alejar el proyector y más espacio necesitaremos. Lo que se hace normalmente es utilizar espejos para aumentar el recorrido de la luz y reducir el espacio necesario. La imagen se proyectará sobre un panel de metacrilato transparente lo suficientemente grueso (1 cm) para garantizar uno de nuestros requisitos iniciales, la seguridad. Sobre el metacrilato colocaremos una lámina de papel vegetal que actuará como difusor y además servirá como pantalla de proyección, ya que sino la luz del proyector atravesaría el panel y no se vería la imagen. El papel vegetal está protegido por una capa de plástico transparente muy delgado que facilita su limpieza y como efecto secundario aumenta el brillo de las imágenes proyectadas.

La idea es que el usuario interactúe directamente sobre la imagen generada por el proyector para así tener una retroalimentación directa sobre la misma pantalla.
Esto nos lleva a tener que detectar las sombras de los dedos sobre una pantalla en la que proyectamos una imagen. Lógicamente, la imagen generada por el proyector interferirá con las sombras producidas por las manos y dificultará la posterior detección del punto de contacto. Esto no sucedería si estuviéramos construyendo un panel táctil, ya que no necesitaríamos la proyección de ninguna imagen. La solución que se adopta en estos casos es iluminar la superficie con luz infrarroja y detectar las sombras con una cámara que pueda captar este tipo de luz.
Las tradicionales webcams son capaces de detectar un determinado rango de luz infrarroja pero se venden con un filtro que la bloquea. Adaptar la webcam para poder detectar la luz infrarroja es relativamente sencillo, aunque depende de la cámara. El primer paso es desmontarla y comprobar si el filtro infrarrojo puede quitarse directamente o sino puede ser necesario romperlo cuidadosamente con un fino punzón, como fue en nuestro caso, ya que el filtro infrarrojo estaba fijo sobre la lente (figura 2). Una vez podemos detectar la luz infrarroja y para impedir las interferencias de la luz visible (del proyector) que todavía puede detectar la webcam, tenemos que usar un filtro pasabandas que únicamente deje pasar la luz infrarroja y bloquee la luz visible del espectro electromagnético. Este tipo de filtros puede comprarse, aunque su precio es elevado. Lo más sencillo es utilizar un filtro pasa altas utilizando simplemente un negativo fotográfico velado. Como es muy fino, debemos superponer varias capas y colocarlas en lugar del filtro de infrarrojos. También puede usarse el material flexible de un diskette, aunque los resultados son de peor calidad. Para comprobar si la cámara modificada funciona podemos utilizar el mando a distancia de cualquier dispositivo, al apuntar con el directamente a la cámara y pulsar cualquier botón del mando deberíamos ver los destellos producidos por la luz infrarroja.
En esta mesa hemos utilizado una iluminación difusa frontal, usando directamente la iluminación infrarroja del entorno. Nos dimos cuenta que el entorno de trabajo de la mesa tenía un componente IR muy difusa, los dedos se detectaban perfectamente antes de iluminar con nuestra luz infrarroja la superficie, por ello no fue necesario añadir ningún tipo adicional de iluminación.

Figura 2. El filtro IR se ve claramente de color rojo

La estructura de la mesa está construida en madera de un centímetro de gruesa, de forma que su montaje/desmontaje es sencillo (figura 3), recubierta de pintura acrílica.
Internamente, la mesa contiene un ordenador con sistema operativo, un sistema de refrigeración para contrarrestar las altas temperaturas producidas por el proyector, un espejo que refleja la imagen y reduce considerablemente la distancia a la que debe estar el proyector y la webcam modificada (figura 4).

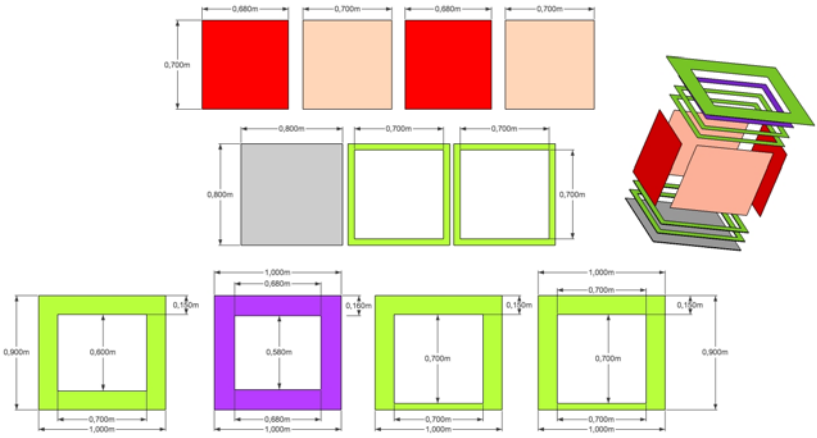


Figura 3. Esquema de la estructura de madera

Para minimizar la dependencia del usuario, sólo sobresalen de la mesa un enchufe y el interruptor de encendido/apagado. Si se requiere un acceso alternativo al sistema se dispone de un teclado y un ratón inalámbrico.
Se puede ver la configuración final tanto exterior como interior de la mesa multitáctil en la figura 4.

Figura 4. La tabla multitáctil. Diferentes etapas durante la construcción (a), detalle del proyector, espejo y sistema de refrigeración (b), ordenador (c), vista exterior (e), en funcionamiento (e)

Una vez terminada la mesa, se hicieron las pruebas de detección utilizando la librería touchlib (http://nuigroup.com/touchlib/), los blobs o regiones de la imagen que representan cada uno de los dedos que contactan la pantalla se ven claramente y se pueden seguir (figura 5). También probamos el software Reactivision [32] y después de ajustar el tamaño de los blobs deseados, el sistema también se comportó de forma estable.

Después de los primeros ensayos, se nos informó que la mayor parte de los usuarios tenían problemas de acceso a pesar de los 20 cm de margen que dejamos en los lados. No era suficiente para una aproximación cómoda de las sillas de ruedas, los usuarios con mayor

movilidad podían acceder. Los usuarios tenían que trabajar desde demasiado lejos, causando problemas de fatiga impropios de una terapia de relajación. De hecho no había ningún espacio desaprovechado en el interior de la mesa por consiguiente el diseño no podía mejorar de ninguna manera el acceso. Cualquier cambio en la distribución interferiría con el campo de visualización de la cámara interior.

El problema de accesibilidad debía afrontarse desde otro punto de vista completamente diferente y por ello observamos a los usuarios en su rutina diaria.

Una de las actividades principales en las que los usuarios discapacitados se sientan delante de una mesa es a la hora de comer. Utilizan mesas adaptadas a las sillas de ruedas, que pueden ajustarse en altura para los diferentes modelos y tamaños de sillas permitiendo a los usuarios posturas cómodas. Estas mesas permiten la aproximación del usuario gracias a una hendidura semicircular en la superficie (figura 6). El nuevo diseño se está diseñando y desarrollando sobre esta premisa.

Figura 5. Detección por visión de los blobs correspondientes a cada dedo

Figura 6. Mesa accesible en silla de ruedas

9. Conclusiones y trabajo futuro

La interacción multitáctil se ha convertido en uno de los medios más intuitivos y utilizados en los últimos años. Su presencia en la sociedad es más que evidente en campos como la telefonía móvil, las tabletas o los lectores de libros. Hemos visto cómo, a partir de técnicas de visión por ordenador, se puede construir una mesa multitáctil de bajo coste y que cumpla requisitos adicionales como la seguridad del usuario. La mesa que hemos visto, junto con un sistema multimedia, se incluye dentro de un entorno interactivo de estimulación multisensorial que a través de programas de reconocimiento de imágenes ofrecen al terapeuta profesional herramientas alternativas para su trabajo en los momentos de atención indirecta haciendo posible la estimulación autónoma y controlada de los pacientes. El entorno ha sido diseñado para generar un ambiente adecuado para la terapia de los comportamientos autoestimulantes y es una muestra palpable de la utilidad de la visión por ordenador en el campo de la interacción (VBI).

A partir de este momento corresponde al terapeuta la evaluación de la validez de las herramientas que se ofrecen a partir de este entorno y que deberán llevar al diseño/programación de nuevas aplicaciones que permitan disminuir o eliminar los comportamientos autoestimulantes de los pacientes.

Al haberse detectado en una evaluación preliminar que el diseño de la mesa multitáctil presenta, para algunos de los usuarios, dificultades de acceso, se está rediseñando a partir de una mesa adaptada.

10. Referencias

[1] J. Foley, A. van Dam (1982). *Fundamentals of Interactive Computer Graphics*. Reading, MA, USA: Addison-Wesley (IBM Systems Programming Series).

[2] W.Newman, R.Sproull (1979). *Principles of Interactive Computer Graphics*, Mcgraw-Hill College, ISBN 0-07-046338-7.

[3] Buxton, B. "http://www.billbuxton.com/multitouch Overview. html" Last updated March 2011. Last accessed July 2011.

[4] A. Goldberg, ed. (1988). *A History of Personal Workstations.* Addison-Wesley Publishing Company: New York, NY. 537.

[5] E.A. Johnson (1965). "Touch Display - A novel input/output device for computers". En: *Electronics Letters* 1 (8): 219-220.

[6] Mehta, N. (1982). *A Flexible Machine Interface.* En: *M.A.Sc. Thesis*, Department of Electrical Engineering, University of Toronto supervised by Professor K.C. Smith.
[7] Smart technologies [en línea] Consultado el 01 de Julio de 2011 en <http://smarttech.com/>.
[8] George, J., de Araujo, E., Dorsey, D., McCrickard, D.S., and Wilson, G. (2011). Multitouch tables for collaborative object-based learning. En: *Proceedings of HCII*.
[9] Microsoft Optimizing the Microsoft Surface Location. [en línea] Consultado el 01 de Julio de 2011 en <http://technet.microsoft.com/en-us/library/ee692028(Surface.10) .aspx>.
[10] AlAgha, I., Hatch, A., Ma, L., Burd, L. (2010). Towards a Teacher-centric Approach for Multi-touch Surfaces in Classrooms. En: *ACM International Conference on Interactive Tabletops and Surfaces 2010*, Saarbrucken, Germany, ACM.
[11] Burd, E., Elliott, J., Hatch, A., Higgins, S., Kyaw P., Smith, S. (2009). Multi-touch desks for learning: developing pedagogy through technology. En: *Proceedings of the 13th Biennial Conference of the European Association of Research into Learning and Instruction*, Amsterdam, EARLI.
[12] Hatch, A., Higgins, S & Mercier, E. (2009). SynergyNet: Supporting Collaborative Learning in an Immersive Environment. En: *STELLAR Alpine Rendez-Vous Workshop 2009: Tabletops for Education and Training*, December 2009, Garmisch-Partenkirchen.
[13] Rick, J., Rogers, Y. (2008). From digiquilt to digitile: Adapting educational technology to a multi-touch table. En: *Proceedings TABLETOP '08*, 73-80.
[14] Kharrufa, A., Leat, D., Olivier, P. (2010). Digital mysteries: designing for learning at the tabletop. En: *Proceedings of ITS '10*, 197-206, New York, ACM Press.
[15] Marco, J., Baldassarri, S., Cerezo, E. (2010). Bridging the gap between children and tabletop designers. En: *Proceedings of IDC'10*. ACM, NY, 98-107.
[16] Seegers, M. (2001). Special Technological Possibilities for Students with Special Needs. En: *Learning & Leading with Technology*, vol. 29, number 3, 32-39.
[17] Roldán, D., Martín, E., Haya, P.A., García-Herranz, M. (2011). Adaptive Activities for Inclusive Learning using Multitouch Tabletops: An approach. En: *Diana Perez-Marin, Milos Kravcik, Olga C. Santos (Eds.), Proceedings of the International Workshop on Personalization Approaches in Learning Environments* (PALE-2011) at 19th International Conference on User Modeling, Adaptation and Personalization. CEUR Workshop Proceedings, Vol. 732, 42-47.
[18] Piper, A. M., O'Brien, E., Morris, M. R., Winograd, T. (2006). SIDES: a cooperative tabletop computer game for social skills development. En: *20th Anniversary Conference on Computer Supported Cooperative Work.*
[19] Dunne, A.; Son Do-Lenh; Laighin, G.O., Chia Shen; Bonato, P. (2010). Upper Extremity Rehabilitation of Children with Cerebral Palsy using

Accelerometer Feedback on a Multitouch Display. En: *Engineering in Medicine and Biology Society* (EMBC), 2010 Annual International Conference of the IEEE, 1751-1754.
[20] Annett, M., Anderson, F., Goertzen, D., Halton, J., Ranson, Q., Bischof, W. F., Boulanger, P. (2009). Using a Multi-touch Tabletop for Upper-Extremity Motor Rehabilitation. En: *OZCHI 2009 Proceedings*, 261-264.
[21] Aspace Baleares, http://aspaceib.org/ last accessed July 2011.
[22] Bax, M., Goldstein, M., Rosenbaum, P., Leviton, A., Paneth, N. (2005). Proposed definition and classification of cerebral palsy. En: *Developmental Medicine & Child Neurology*, 47: 571-576.
[23] García-Navarro, M.E., Tacoronte, M., Sarduy, I., Abdo, A., Galvizú, R., Torres, A., Leal, E. (2000). Influence of early stimulation in cerebral palsy. En: *Rev Neurol*. 16-31; 31 (8): 716-9.
[24] Reisman, J. (1993. Using a sensory integrative approach to treat self-injurious behavior in an adult with profound mental retardation. En: *American J. Occupational Therapy*, 47, 403-411.
[25] Smith, S. A., Press, B., Koenig, K. P., Kinnealey, M. (2005). Effects of sensory integration intervention on self-stimulating and self-injurious behaviours. En: *American J. Occupational Therapy*, 59, 418-425.
[26] Singh, N., Lacioni, G.E., Winton, A. S. W., Molina, E.J., Sage, M., Brown, S., Groeneweg, J. (2004). Effects of Snoezelen room, Activities of Daily Living skills training, and Vocational skills training on aggression and self-injury by adults with mental retardation and mental illness. En: *Research in Developmental Disabilities* 25, 3, 285-293.
[27] C. Manresa-Yee, R. Mas, G. Moyà, M.J. Abásolo, J. Giacomantone (2011). Interactive multi-sensory environment to control stereotypy behaviours. En: Actas del congreso CACIC 2011.
[28] NUI Group authors (2009). Multitouch Technologies, V1.0 [en línea] Consultado el 01 de Julio de 2011 en
<http://nuicode.com/attachments/download/115/Multi-Touch_Technologies_v1.01.pdf>.
[29] Han, J. Y. (2005). Low-cost multi-touch sensing through frustrated total internal reflection. En: *UIST '05 Proceedings of the 18th annual ACM symposium on User interface software and technology*, 115-118, New York, NY, USA, ACM Press.
[30] Schöning, J., Brandl, P., Daiber, F., Echtler, F., Hilliges, O., Hook, J., Löchtefeld, M., Motamedi, N., Muller, L., Olivier, P., Roth, T., von Zadow, U. (2008). Multi-Touch Surfaces: A Technical Guide. En: *Technical Report TUM-I0833.*
[31] Schöning, Hook, J., Bartindale, T., Schmidt, D., Olivier, P., Echtler, F., Motamedi, N., Brandl P. (2010). Building interactive multi-touch surfaces. En: *C. Müller-Tomfelde, editor, Tabletops-Horizontal Interactive Displays*. Springer.
[32] Kaltenbrunner, M., Bencina, R. (2007). ReacTIVision: a computer-vision framework for tablebased tangible interaction. En*: Proceedings of the 1st international conference on Tangible and embedded interaction*, 69-74.

Esta publicación se terminó de imprimir
en el mes de octubre de 2011,
en la ciudad de La Plata,
Buenos Aires,
Argentina.