APÉNDICES

2/2

María Escriche Andrés. Grado en Diseño Industrial y Desarrollo de Producto.

Índice de los Apéndices.

APÉNDICE 1

Introducción a la Realidad Aumentada.

Índice del Apéndice 1.

1.0 Introducción.

En el Apéndice 1 se describe qué es la Realidad Aumentada, qué se utiliza para implementarla y se presentan algunas de las filosofías con las que puede trabajarse.

1.1 Realidad Aumentada.

La Realidad Aumentada es una tecnología a través de la cual se mezcla el entorno físico con elementos virtuales en tiempo real. Unido a las técnicas de interacción se puede ampliar la información que proporciona el mundo físico, a la vez que se puede mejorar la interactividad del usuario. Ejemplos de su uso podrían ser la representación textual de datos superpuestos sobres escenas u objetos reales y escenas gráficas 3D interactivas integradas en otras reales. Esta realidad mixta se puede mostrar a través de distintos dispositivos (ordenador, móvil, tablet…).

Aplicación de Realidad Aumentada a través de móvil.

Los elementos que intervienen en una aplicación de Realidad Aumentada son la cámara del dispositivo con el que se utiliza la aplicación, los modelos o elementos multimedia generados virtualmente, y los elementos de referencia.

1.1 Realidad Aumentada.

La Realidad Aumentada depende de forma crucial del hardware capaz de capturar información sobre el mundo real, como vídeo, datos de posición y otras formas de datos, y que pueda reproducir una representación que combine medios en directo con contenido virtual de forma que tenga sentido y utilidad para los usuarios. Con la reciente expansión de los teléfonos inteligentes, prácticamente todo el mundo dispone de hardware con potenciall para RA. Esto ha contribuido al interés por el desarrollo de la RA, tanto para plataformas móviles como generales. Gracias al extenido uso de cámaras Web y portátiles, ha aumentado la RA basada en navegadores.

Existen diversas variedades de RA. En algunas aplicaciones usan datos de ubicación (GPS) y orientación (acelerómetro) de un dispositivo móvil para animar o integrar contenidos en una escena en directo. Estas aplicaciones saben lo que percibe la cámara de su teléfono móvil porque saben dónde se encuentra y en qué dirección apunta su teléfono. En función de sus datos, se pueden tranferir anotaciones, desde un servicio centralizado o a través de otros usuarios, y después superponerlas sobre la escena en la cámara.

Aplicación de RA con móvil: Anotaciones superpuestas sobre la escena.

1.1 Realidad Aumentada.

Otro enfoque de RA consiste en usar las imágenes capturadas por una cámara para determinar lo que se está viendo. Esta tecnología se denomina visión informática. El ordenador procesa los píxeles de cada fotograma de vídeo y evalúa su relación con los píxeles circundantes, en el tiempo y en el espacio, para identificar patrones. Entre otros aspectos, incluye la posibilidad de reconocimiento facial, identificación de objetos móviles en vídeos y marcadores o patrones visuales concretos identificados previamente para el algoritmo. La RA basada en visión informática se puede usar en contexto móviles y no móviles. Se puede emplear para mejorar métodos de RA basados en posición y orientación, y también para crear aplicaciones de RA que no estén vinculadas a una ubicación concreta. Se pueden reconocer patrones de paquetes, productos, ropa, ilustraciones y otros muchos contextos.

Aplicaciones de RA a través de visión informática: reconocimiento de un patrón visual (imagen de la izquierda) y de un marcador (derecha) . La imagen de la derecha muestra lo que se vería en la pantalla del ordenador.

1.1 Realidad Aumentada.

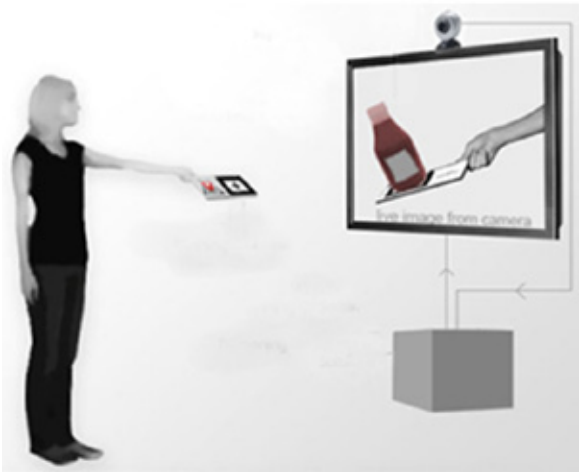
Existen distintas herramientas y tecnologías para trabajar la realidad Aumentada. A continuación se describen tres filosofías para trabajar con la Realidad Aumentada y se citan programas o librerías con las que pueden obtenerse tales aplicaciones.

- Las aplicaciones basadas en marcadores o imágenes.
- Las aplicaciones basadas en creación de mapas.
- Las aplicaciones basadas en sensores.

Aplicaciones basadas en marcadores o imágenes.

Las aplicaciones basadas en marcadores trabajan con recuadros en los que están impresas imágenes simples (marcadores) o complejas, que sirven para que el programa, a través del marcador, determine la posición y orientación del modelo asociado a éste.

Consideramos una situación como la que se muestra en la imagen, en la que una persona sujeta un marcador delante de una cámara conectada a un ordenador. En la pantalla del ordenador se ve la escena que graba la cámara y un modelo superpuesto sobre el marcador impreso.



Ejemplo de uso de una aplicación de RA con marcador.

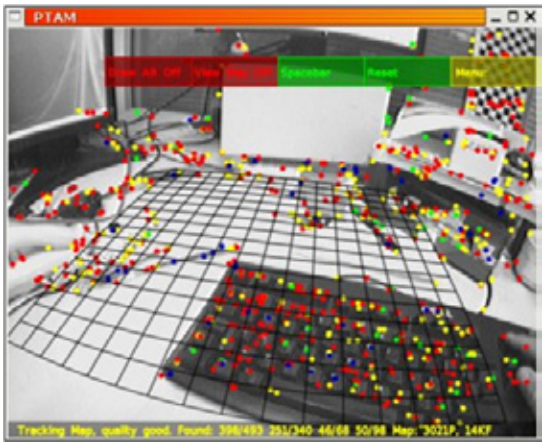
1.1 Realidad Aumentada.

El proceso informacional actúa de la siguiente manera: La cámara captura la imagen del mundo real y lo envía al computador. El software en el computador busca una forma de marcador que coincida que el marcador utilizado y, si lo encuentra, calcula la posición de la cámara relativa al marcador. Una vez reconocida la posición de la cámara, se renderiza un modelo gráfico desde esa posición, se superpone sobre la imagen capturada y se visualiza el conjunto final. Todo el proceso anterior se realiza en tiempo real, de forma que el modelo siempre aparece sobre el marcador en la posición correcta. Un ejemplo de una librería que permite utilizar esta filosofía en ARToolKit.

Aplicaciones basadas en creación de mapas.

Las aplicaciones basadas en creación de mapas no utilizan marcadores o imágenes impresas para calcular la posición del modelo o elementos virtuales. Realizan el seguimiento de la cámara en una escena de la que no se tiene conocimiento previo, creando un mapa rudimentario del entorno . A partir de él, se insertan objetos virtuales en la escena, y estos se posicionan en relación con los objetos reales del entorno. Está técnica está pensada para entornos estáticos. Un ejemplo de libraría útil sería PTAM.

1.1 Realidad Aumentada.



Seguimiento de la escena con PTAM.

Aplicaciones basadas en sensores.

Las aplicaciones basadas en sensores tampoco utilizan marcadores, sino que hacen uso de distintos sensores, como GPS, brújula, giroscopio y acelerómetro, para obtener la posición absoluta y la orientación del dispositivo. A partir de estos datos, se consigue ubicar ciertos puntos de referencia cercanos obtenidos de una base de datos y generar sobre ellos el contenido virtual que se añadirá en la pantalla del dispositivo. Estas aplicaciones están diseñadas generalmente para dispositivos móviles. Un ejemplo de herramienta para este tipo de aplicaciones es Layar.

Aplicación de móvil a partir de Layar.

APÉNDICE 2

3ds MAX: Aprendizaje de modelado 3D.

Índice del Apéndice 2.

2.0 Introducción.

3ds Max es un programa de creación y animación tridimensional. Se utiliza para crear desde juegos de ordenador hasta escenas cinematográficas con espectaculares efectos especiales o anuncios, simulaciones, etc. Se trata, pues de una herramienta completa de modelado, animación y renderización con resultados inmediatos.

3ds Max es el programa con el que se han adquirido los conocimientos básicos de modelado 3D, animación, aplicación de materiales, uso de luces, cámaras y renderizado. A lo largo de este apéndice se muestran algunas imágenes de los resultados obtenidos con los ejercicios de aprendizaje del programa.

Además, se muestran los resultados de otros ejercicios que se realizaron de manera libre, con la ayuda de tutoriales, en los que se ponen en práctica muchos de los conocimientos adquiridos.

2.1 Introducción a 3ds MAX.

*Presentación de 3ds Max 2013.

*Crear un cubo, una esfera, un cilindro y una tetera.

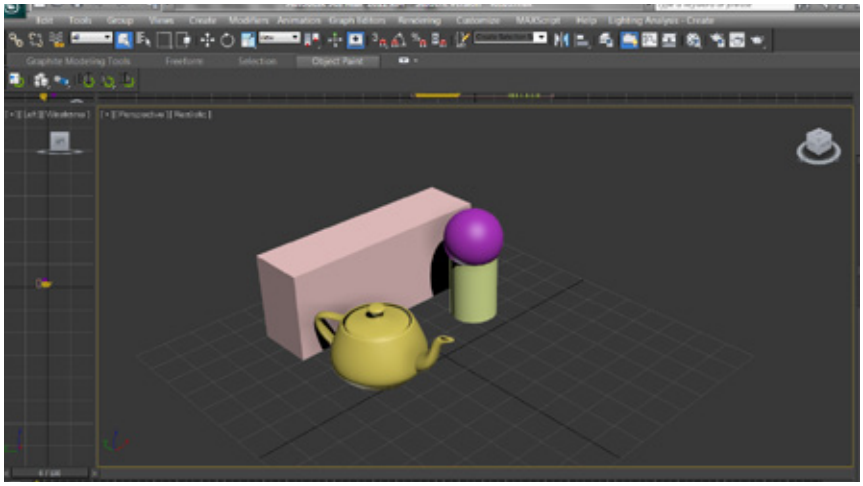
*Mover, rotar y cambiar de tamaño.

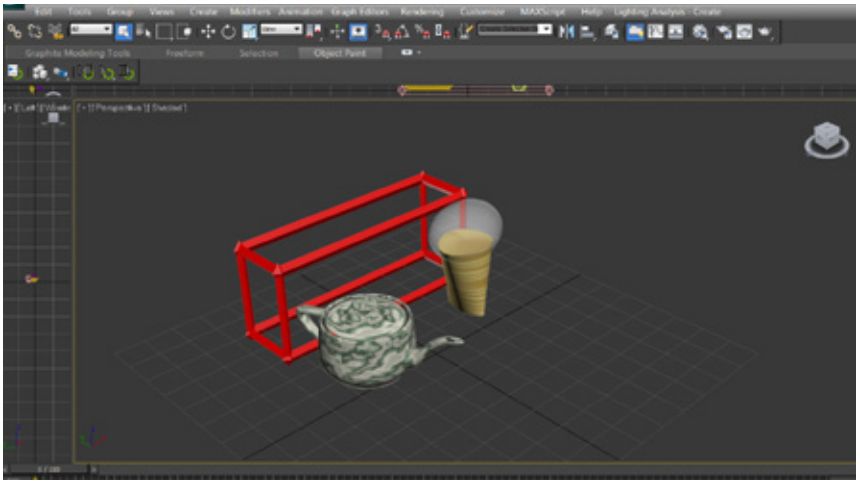
*Deformar objetos, doblarlos y comprimirlos.

*Aplicar materiales a objetos.

*Colocar luces y cámaras.

*Renderización: obtener la primera escena.





2.2 El área de trabajo de 3ds Max.

*La interfaz de 3ds Max.

*Personalizar el aspecto de 3ds Max.

*Crear y gestionar nuevos espacios de trabajo.

*Los menús cuad.

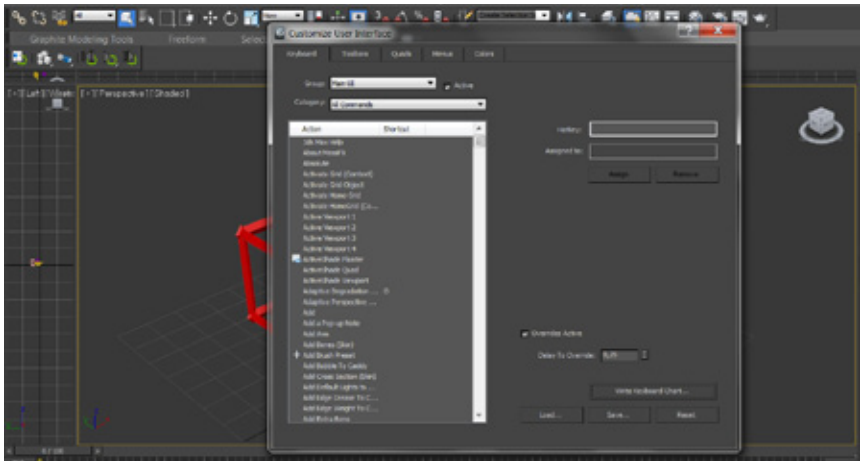
*El sistema de coordenadas de 3ds Max.

*Configurar los visores.

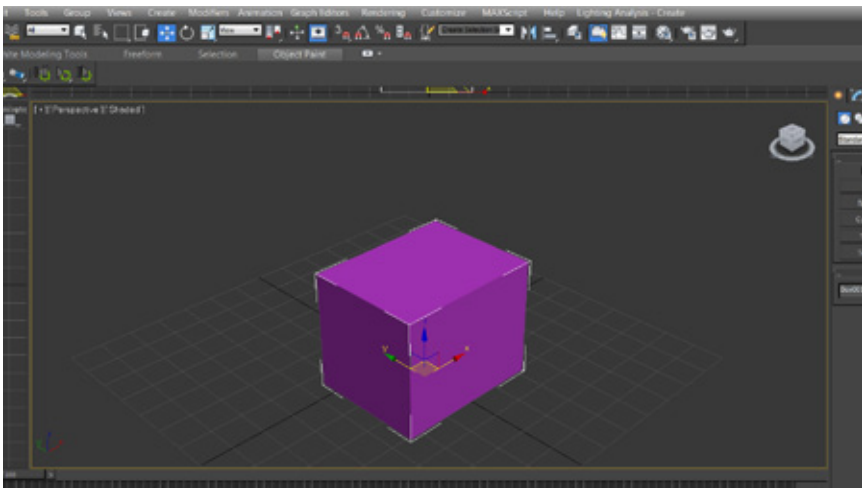
*El control ViewCube.

*El control SteeringWheels.

*La herramienta InfoCenter.



Interfaz, personalización del aspecto, control ViewCube.



2.3 Manipular objetos.

*Abrir y guardar un documento.

*Seleccionar objetos directamente.

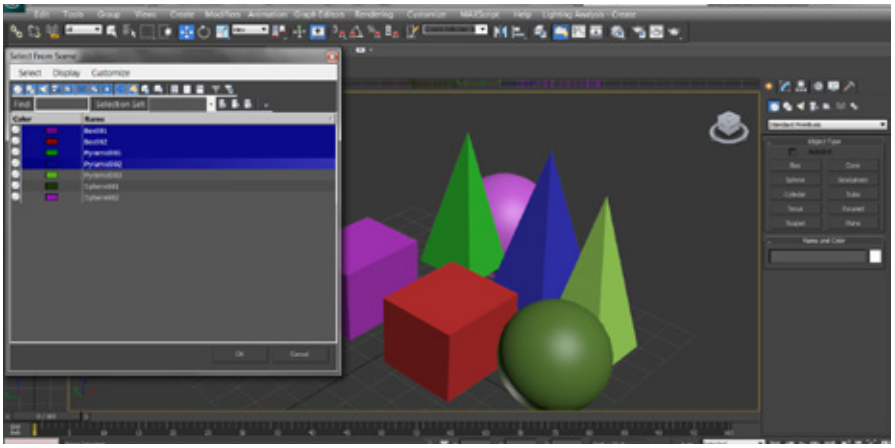
*Seleccionar objetos desde una lista de selección.

*Mostrar y ocultar objetos (I).

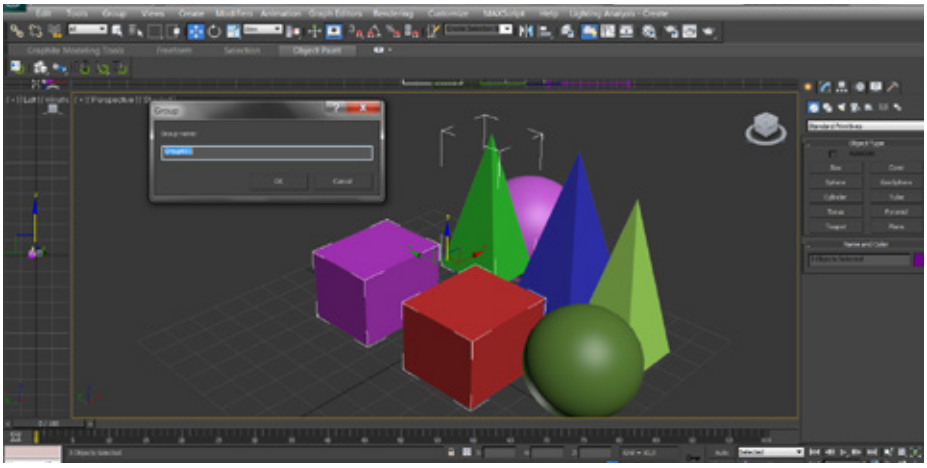
*Mostrar y ocultar objetos (II).

*Agrupar y dar nombre a los grupos.

*Desagrupar, extraer, abrir y cerrar grupos.



Selección de objetos desde la lista de selección.



Agrupar objetos.

2.3 Manipular objetos.

*Trabajar con capas(I).

*Trabajar con capas(II).

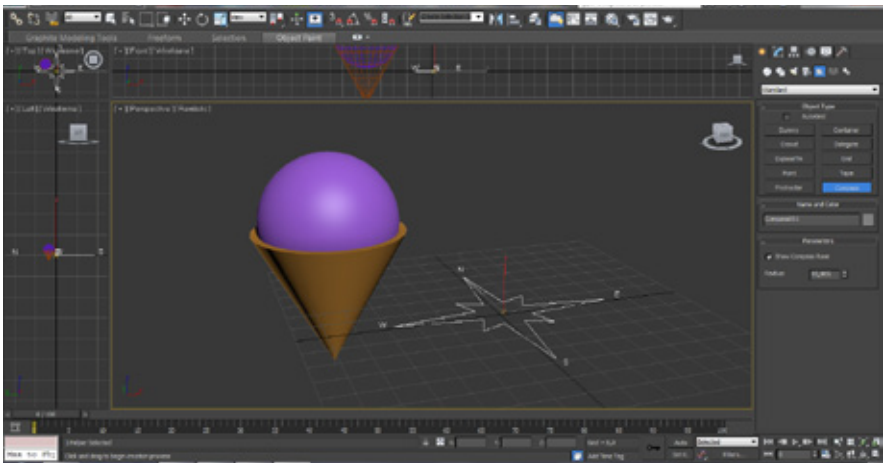
*Unidades y cuadrícula.

*Utilizar ajustes.

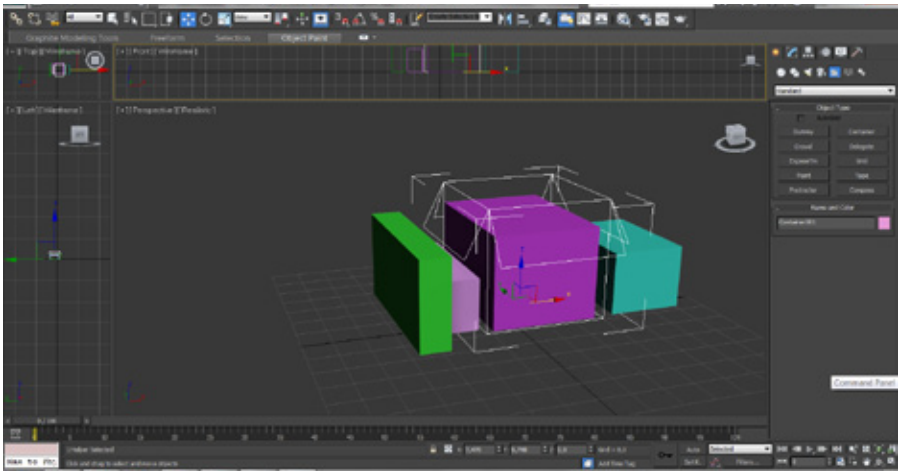
*Los objetos ayudantes.

*La herramienta Container.

*Utilizar la herramienta de alineación.



Utilización de ajustes (Snap Toggle) y ayudantes (rosa de los vientos).



Uso de la herramienta Container.

2.4 Crear y manipular primitivas 3D.

*Crear primitivas estándar.

*Crear primitivas extendidas.

*Crear otros objetos.

*Editar geometría.

*Mover objetos.

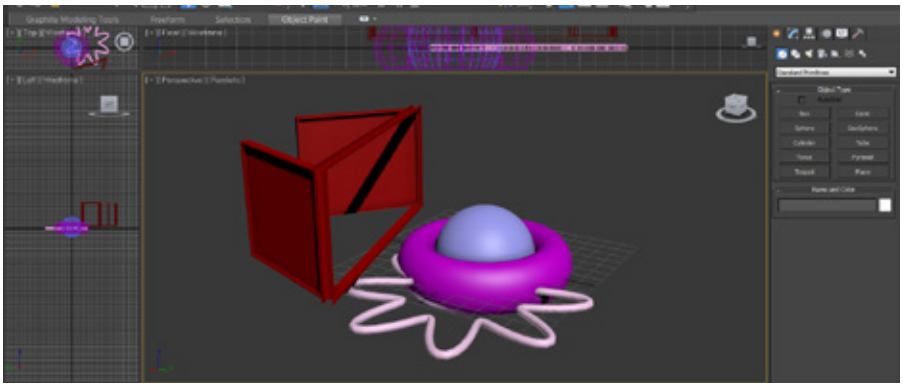
*Rotar objetos.

*Escalar objetos.

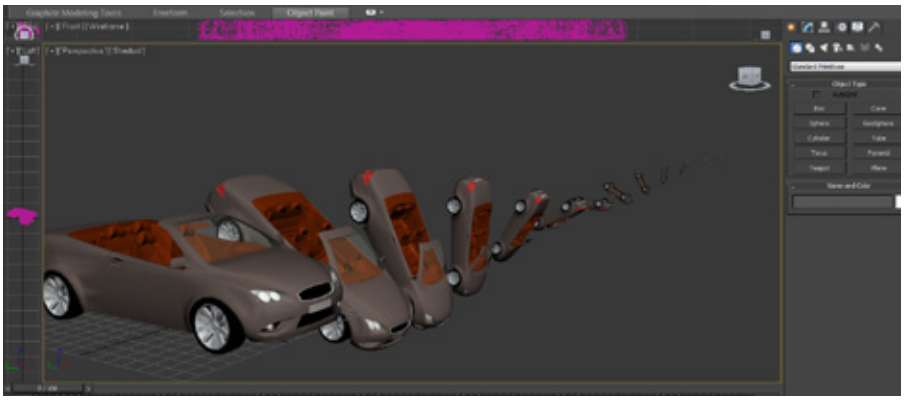
*Clonar objetos.

*El cuadro Transform Toolbox.

*Crear matrices de objetos.



Creación de primitivas estándar (esfera, toroide), extendidas (Torus Knot) y de otros objetos (ventana)



Matriz de objetos (en el cuadro de parámetros de la matriz se coloca un valor de rotación, consiguiendo el efecto de la imagen).

2.5 Crear y manipular primitivas 2D.

*Objetos 2D.

*Texto en 2D.

*Convertir formas en splines editables.

*Trabajar con modificadores de splines.

*Operaciones booleanas con splines.

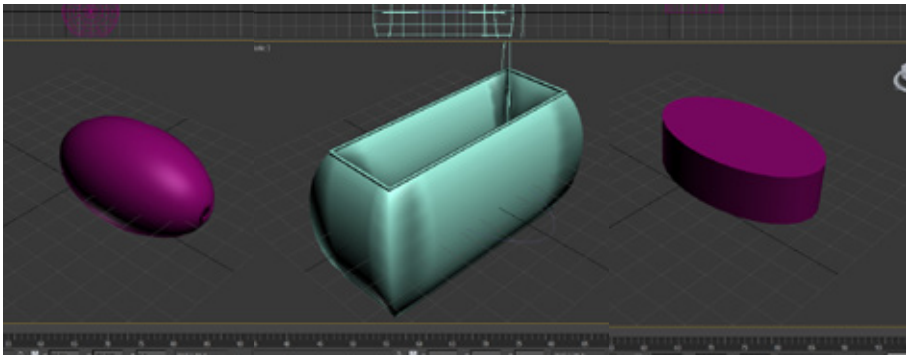
*Crear objetos 3D a paritr de 2D (I): Tornear.

*Crear objetos 3D a paritr de 2D (II): Extruir.

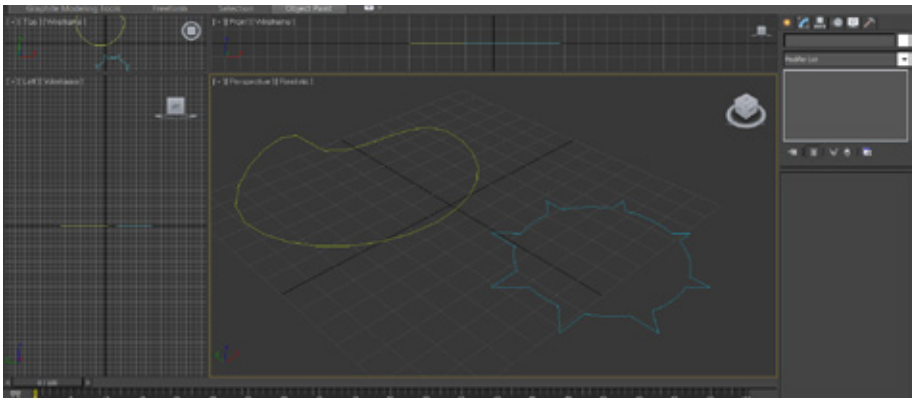
*Crear objetos 3D a paritr de 2D (III): Solevar.

*Las curvas de modificación.

*La herramienta de creación de splines Egg.



A través de la lista de modificadores se ha torneado, solevado y extruido una elipse en 2D, obteniendo las siguientes figuras.



Objeto 2D convertido en spline editable. Operación booleana de unión entre un círculo y una estrella.

2.6 Modelado de mallas.

*Convertir un objeto en malla editable.

*Manipulación de vértices.

*Selección flexible.

*Manipulación de aristas.

*Manipulación de caras y polígonos.

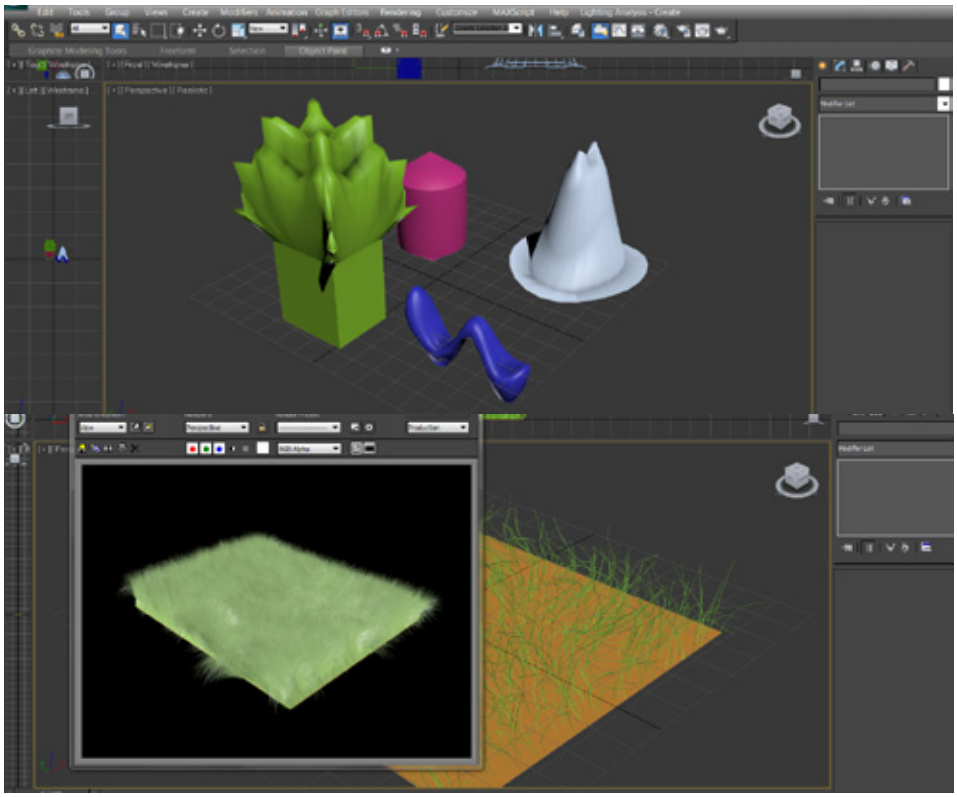
*El analizador de mallas xView.

*Añadir modificadores: Mesh Select, Relax, Edit Mesh, Tessellate, Ripple, Bend, Stretch, Wave y Hair and Fur.

*Acciones con modificadores.

*Cambiar los parámetros de un modificador.

*El gizmo del modificador.



Aplicación de modificadores a disntintos objetos. Aplicación del modificador Hair and Fur para crear césped.

2.7 Modelado de mallas poligonales.

*Bases del modelado poligonal.

*Herramientas de modelado Graphite: su interfaz.

*Herramientas de modelado Graphite: aplicaciones.

*Superficies de subdivisión.

*Edición de mallas poligonales.

*Objetos de composición: BlobMesh y ProBoolean.

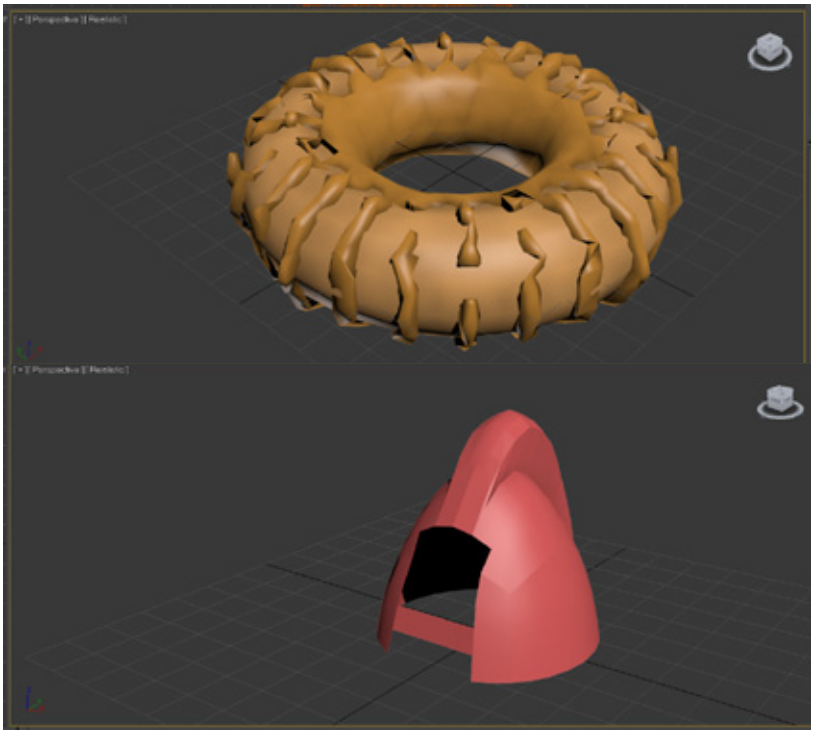
*Añadir modificadores: Face Extrude, Simmetry, Lattice, Taper, Cap Holes, Push, Path Deform y Smooth.

*Copiar modificadores.

*Cambiar el gizmo de un modificador.

*Aplicar varios modificadores sobre un mismo objeto.

*Fijar el catálogo de modificadores.



Creación del objeto de composición BlobMesh. Creación de un casco a través de herramientas de modelado Graphite y modificadores.

2.8 Modelado con superficies.

*Bases del modelado con cuadrículas de corrección.

*Subdividir una cuadrícula de corrección.

*Objetos de composición: Scatter.

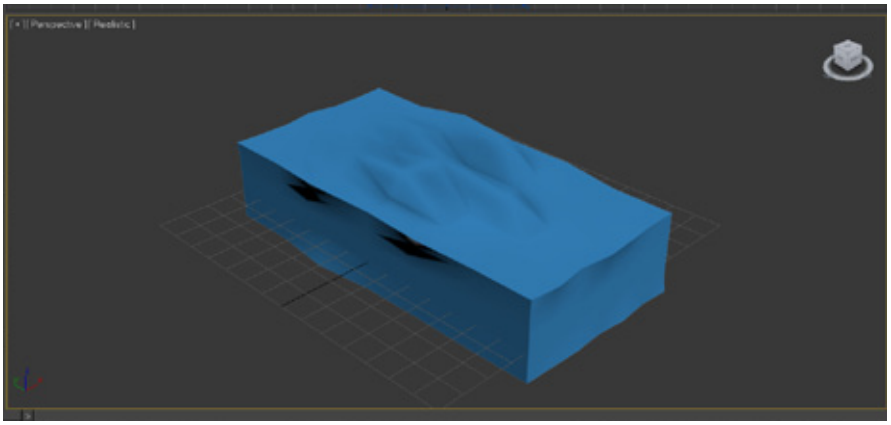
*Añadir modificadores: Noise, Edit Patch, Shell, Displace.

*Contraer el catálogo de subobjetos.

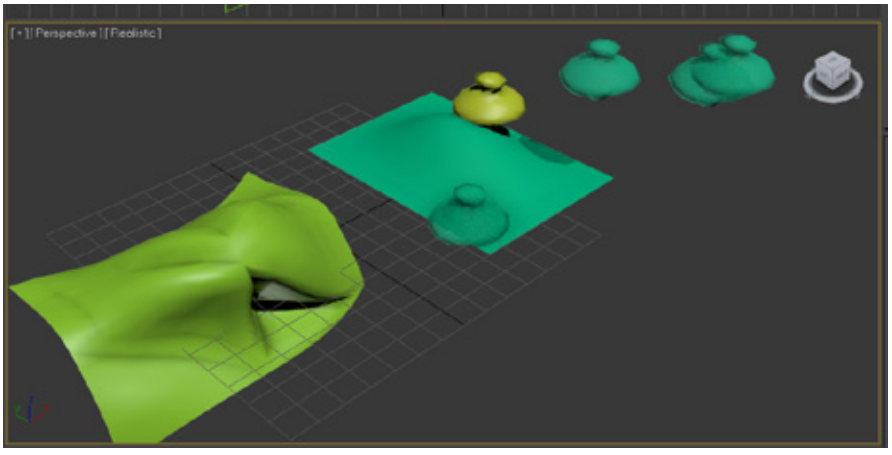
*Usar la selección flexible.

*Trabajo con objetos de composición.

*Eliminar modificadores.



Simulación superficie de agua a través de los comandos Shell y Displace.



Se le han aplicado modificadores tales como Noise y Edit Patch.

2.9 Modelado con NURBS.

*Bases del modelado con NURBS.

*Subdividir una superficie NURBS.

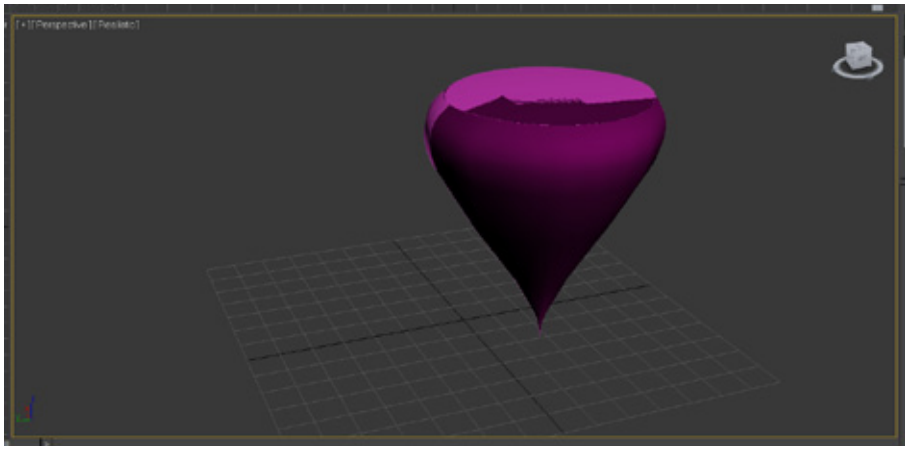
*Otros métodos de creación de NURBS.

*Añadir modificadores: Surface Deform, Twist, Sherify, Melt.

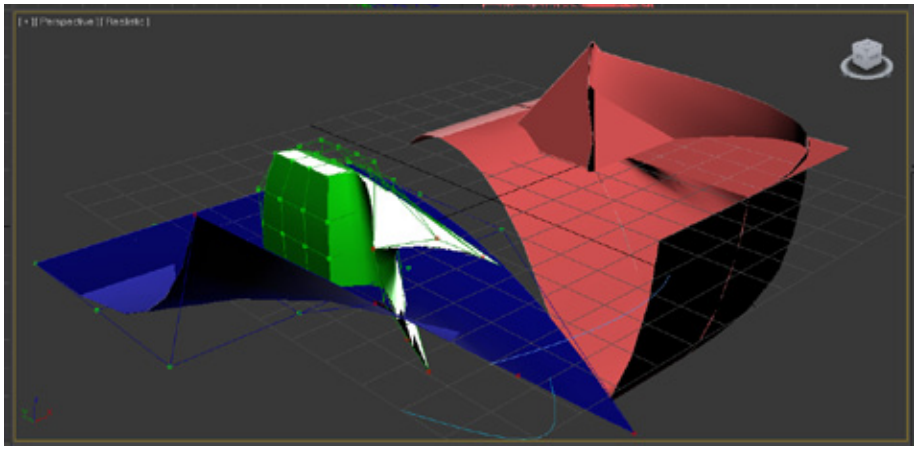
*Mostrar dependencias y convertir en exclusivo.

*Copiar modificadores y por arrastre.

*Trabajar con el modificador Skew.



Aplicación a un cilindro de los modificadores Shperify y Squeeze.



Creación de superficies NURBS, subdivisión y modifiación de las mismas. Aplicación de los modificadores Surface Deform y Skew (al cubo).

2.10 Los materiales.

*Los materiales en 3ds Max.

*El editor de materiales.

*Materiales creados en 3ds Max.

*Materiales con mapas 2D y 3D

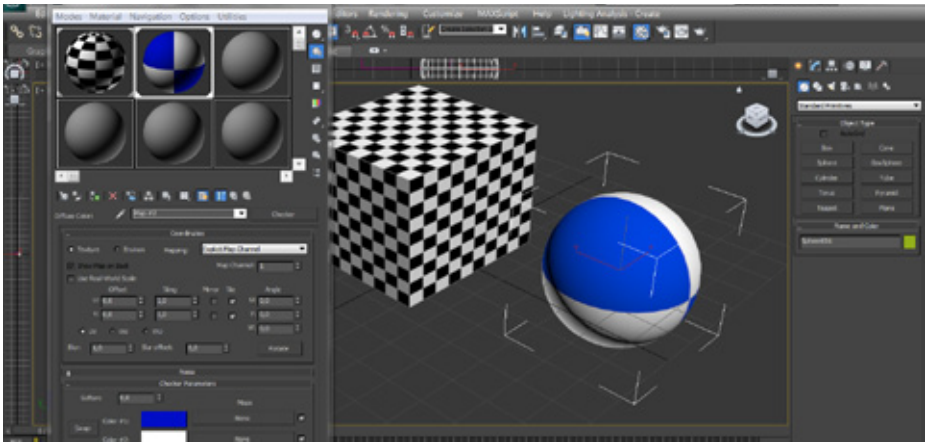
*Efecto mosaico en materiales.

*Materiales Multi/Sub-Object.

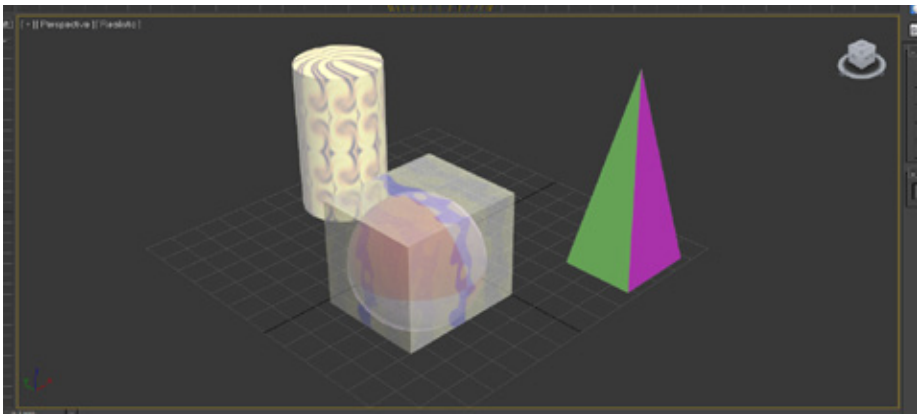
*El material Ink'n Paint.

*El modificador Vertex Paint.

*El modificador Unwrap UVM.



Aplicación de materiales sobre objetos a través del editor de materiales.



Aplicación de materiales con mapas 2D (esfera) y 3D (cubo). Efecto mosaico en materiales (cilindro). Materiales Multi-Object (pirámide).

2.10 Los materiales.

*La función de mapeado de splines.

*El explorador de materiales.

*El Slate Material Editor.

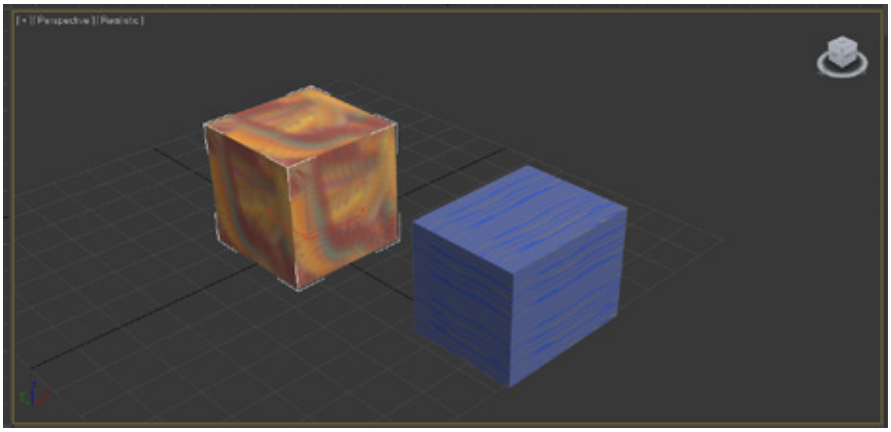
*El renovado acelerador de gráficos Nitrous.

*Los mapas de sustancias.

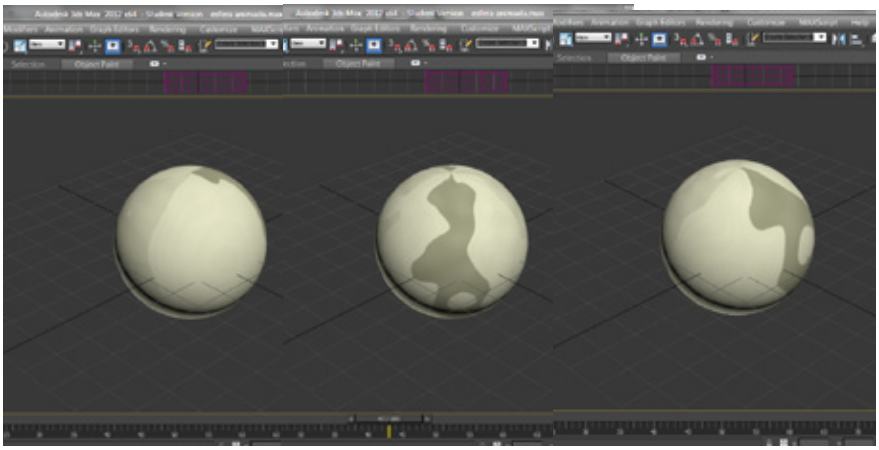
*Usar la herramienta Viewport Canvas para pintar y clonar.

*Los flujos de trabajo Pelt y Relax.

*Crear previsualizaciones de un material animado.



Uso de la herramienta Viewport Cnavas para pintar (cubo de la izquierda).



Previsualización de un material animado. Toma de tres fotogramas durante la reproducción de la animación.

2.11 Aplicar luces.

*La importancia de la iluminación en 3D.

*Las luces libres y objetivo.

*La luces Omni y Skylight.

*Simular la iluminación global.

*Las luces fotométricas.

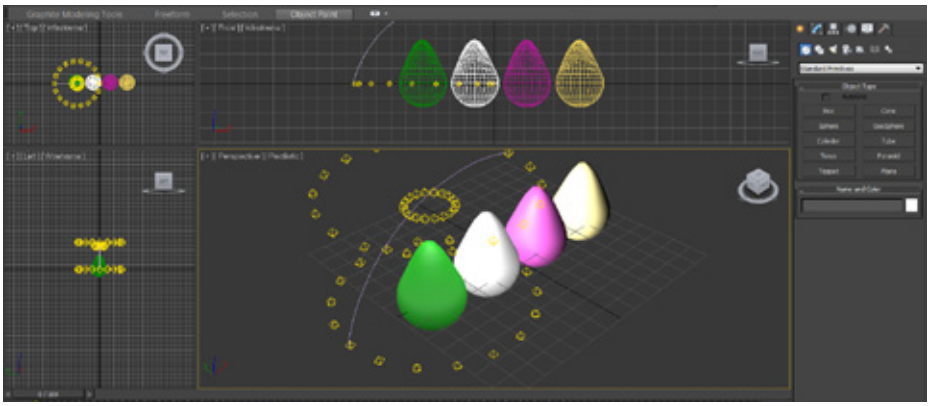
*Otras luces avanzadas: Sunlight y Daylight.

*Añadir sombras por Hardware (I): sombras suaves.

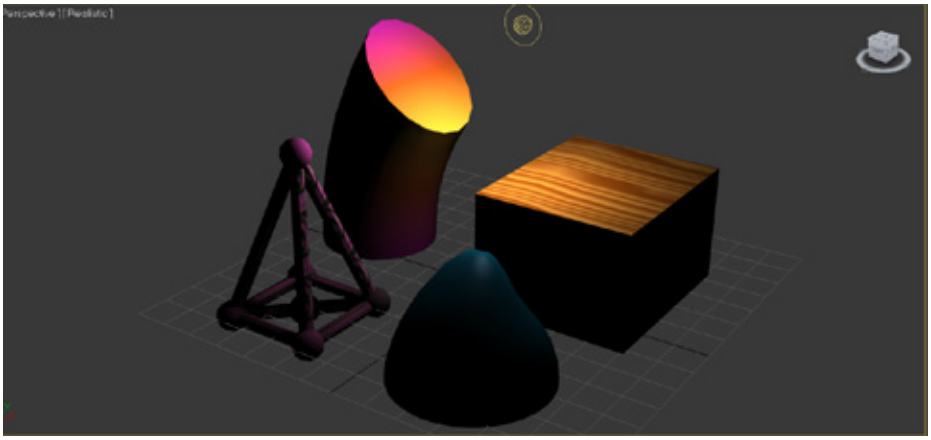
*Añadir sombras por Hardware (II): oclusión ambiental.

*Controlar las luces desde el cuadro Light Lister.

*Analizar la iluminación de una escena.



Simulación de la luz global en la escena.



Uso de luz fotométrica en la escena.

2.12 Aplicar cámaras.

*Colocar una cámara.

*Los controles de la cámara.

*Cámara libre y cámara con objetivo.

*Los parámetros de la cámara.

*El encuadre de la escena.

*La herramienta Walk Throught Viewport Navigation.

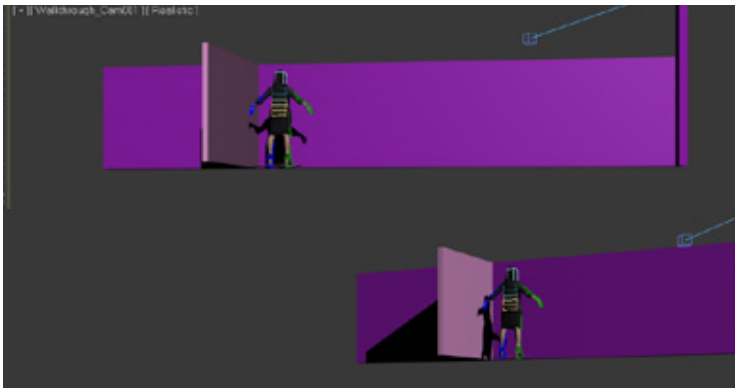
*Animar cámara con WalkThrought Assistant.

*Modificar y reproducir animaciones de cámara.

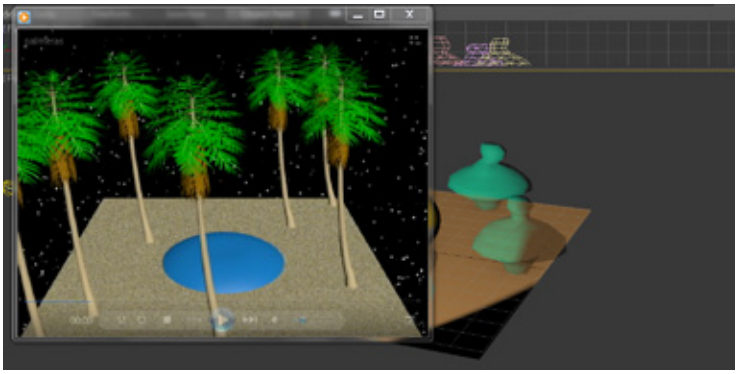
*El modificador Camera Correction.

*El objeto ayudante Camera Point.

*La utilidad Camera Match.



Escena en la que se ha utilizado y animado la cámara con WalkThrought. Dos tomas del movimiento de la



Uso del modificador Camera Correction.

2.13 Animar escenas.

*El modo Auto Key.

*Preferencias de animación.

*Restricciones de animación.

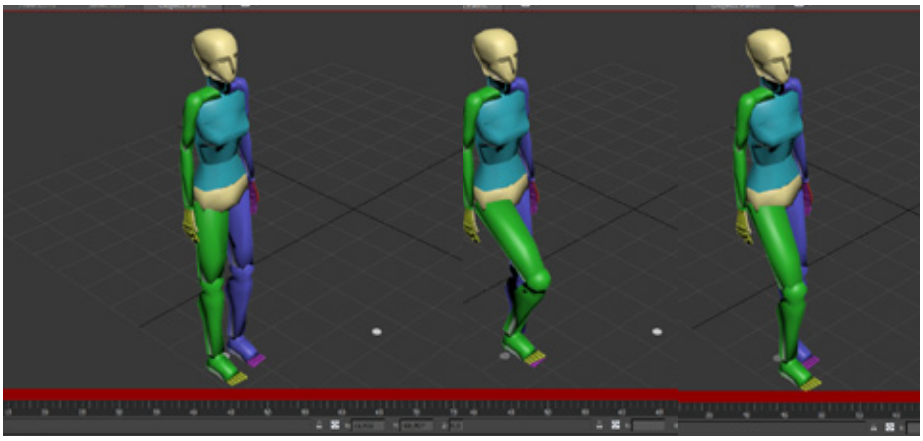
*La barra de pistas y los cuadros clave.

*Configurar el tiempo de animación.

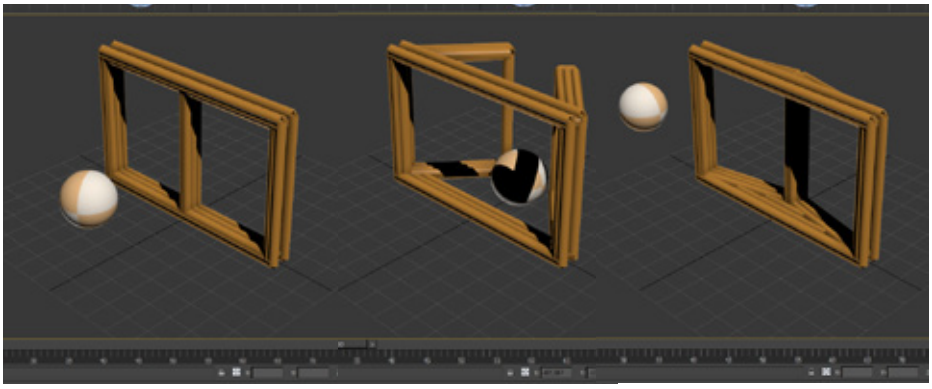
*Los controles de animación.

*Previsualizar la animación

*Track View: Curve Editor.



Creación y modificación de bípedos. Uso del modo Rubber Band. Muestra de animación de un bípedo.



Uso del modo Auto Key para nimar la escena. Uso de la barra de pistas y cuadros clave. Configuración del tiempo de animación y previsualización.

2.13 Animar escenas.

*Track View: Curve Editor.

*Track View: Dope Sheet.

*Crear y modificar un sistema de huesos.

*El modificador Skin.

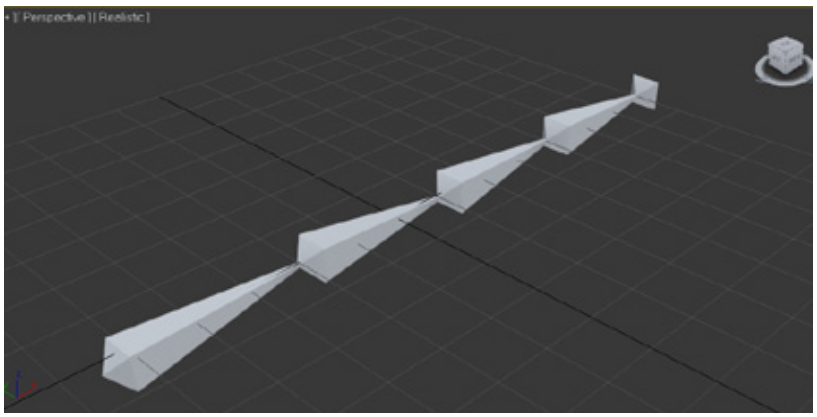
*El punto de Pivote.

*Crear bípedos.

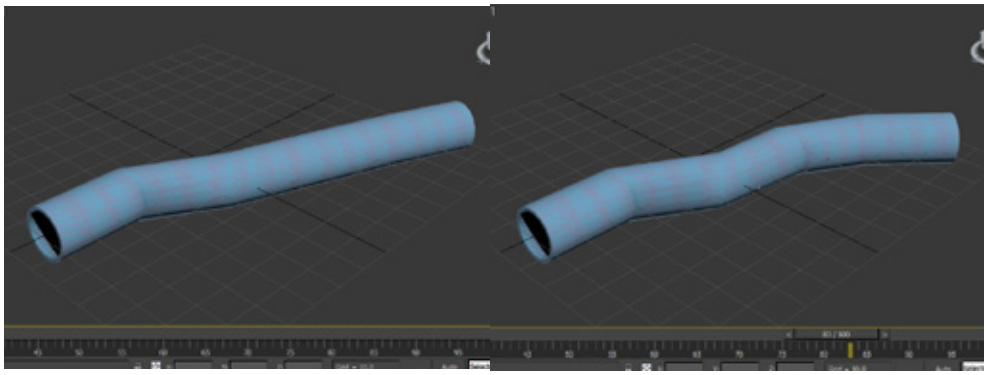
*Modificar parámetros bípedos.

*Usar el modo Rubber Band.

*Guardar y cargar archivos FIG.



Creación de un sistema de huesos.



Uso del modificador Skin. Animación del tubo a partir del sistema de huesos interno.

2.14 Conceptos avanzados de animación.

*Convertir un objeto en un cuerpo rígido.

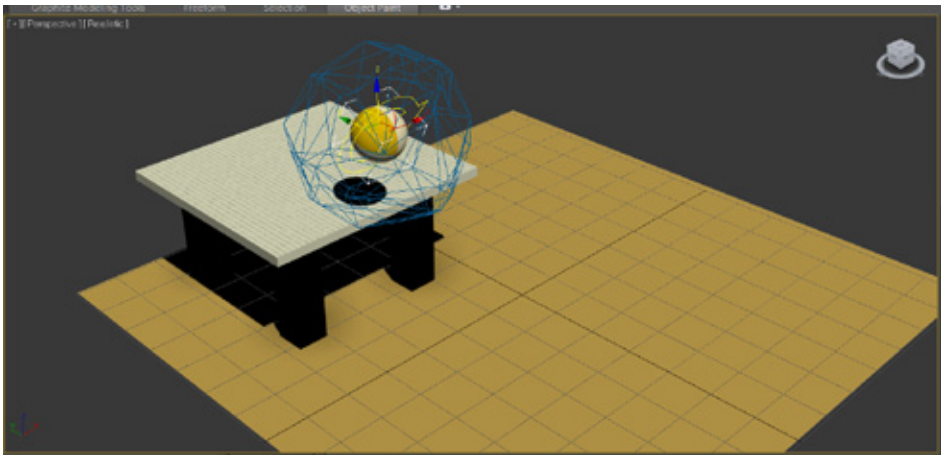
*Modificar el inicio de una simulación.

*Cambiar las propiedades físicas con MassFX.

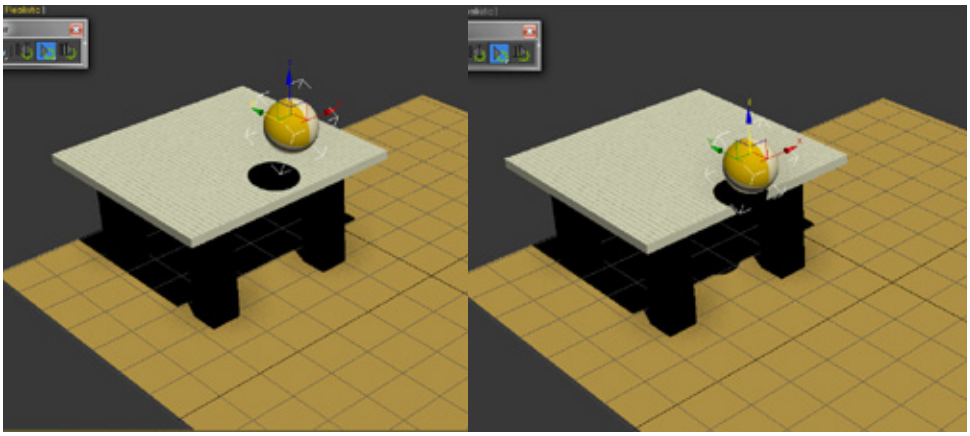
*Restringir movimientos simulados.

*El modificador Rigid Body.

*Exportar cuerpos rígidos y restricciones.



Conversión del objeto en cuerpo rígido y cambio de las propiedades físicas con MassFX. Restringión de los movimientos simulados.



Dos fotogramas del efecto de la pelota al caer tras modificar las porpiedades físicas a través de la herramienta MassFX.

2.15 Renderizar la escena.

*Las opciones del cuadro Render Setup.

*El renderizador mental ray.

*mental ray 3.10.

*Aplicar materiales mental ray.

*Otras opciones de renderización.

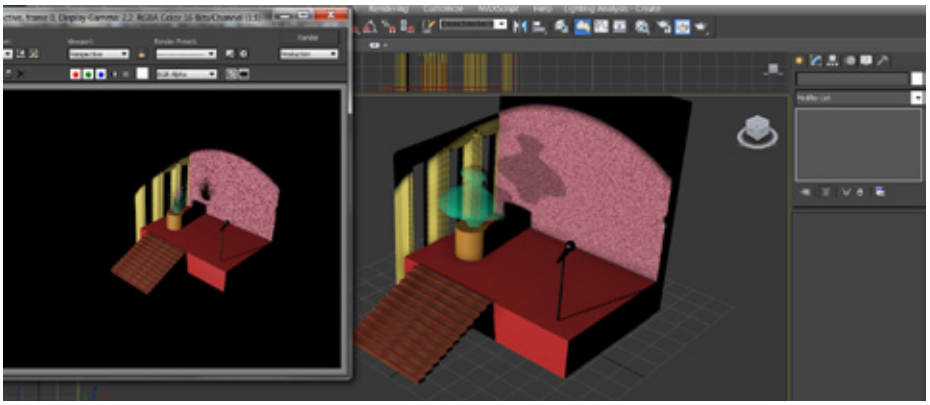
*Renderizar con otros estilos visuales.

*El renderizador iray v2.1.

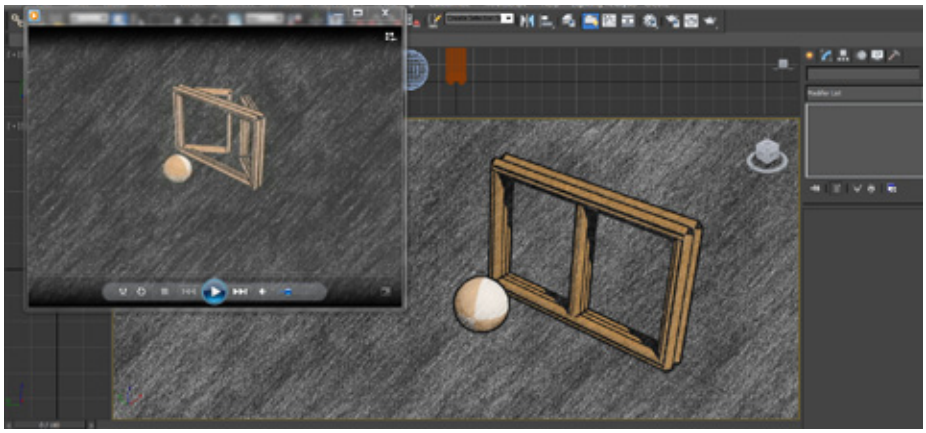
*Guardar una animación renderizada.

*El renderizador Quicksilver.

*El nuevo sistema de renderización State Sets.



Renderizado de la escena con mental ray. Aplicación de materiales con esta herramienta.



Guardar animación renderizada. El renderizador Quicksilver.

2.16 Aplicar efectos a la escena.

*Añadir un efecto de fuego.

*Aplicar un efecto de niebla con volumen.

*Obtener un efecto de niebla.

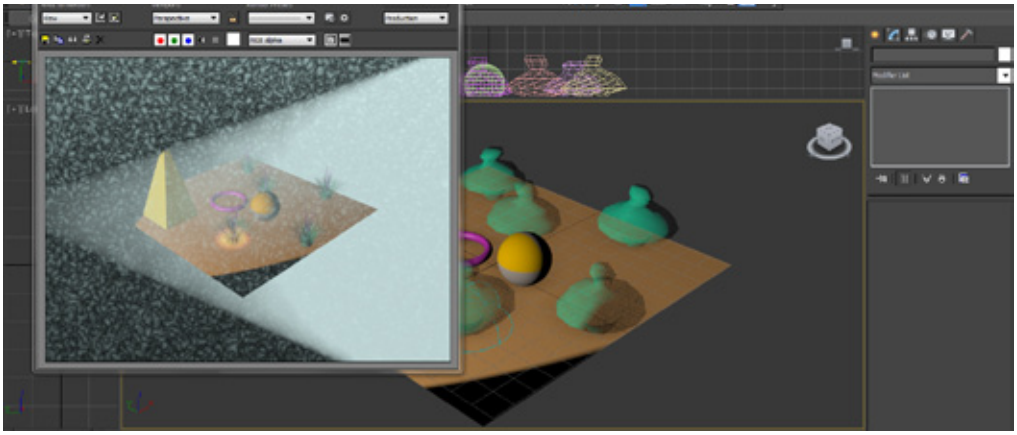
*Volumen luminoso.

*Sistema de partículas.

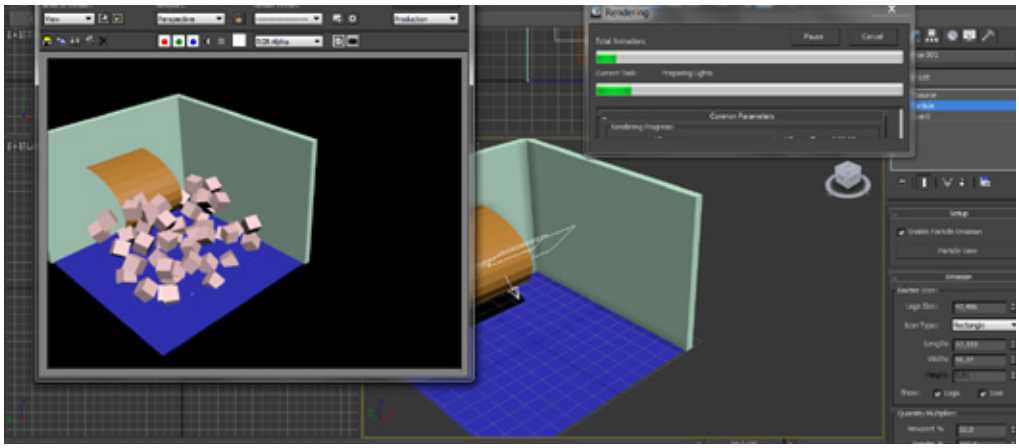
*Sistema de partículas Super Spray .

*Usar la herramienta Partile Flow Source.

*Deformadores de espacio: Creación y enlace a objetos.



Escena a la que se le ha aplicado un volumen luminoso.



Escena a la que se le ha aplicado un sistema de partículas (geometría).

2.16 Aplicar efectos a la escena.

*Dinámica: Efecto de rebote.

*Simular el efecto del viento.

*Simular el efecto de la gravedad.

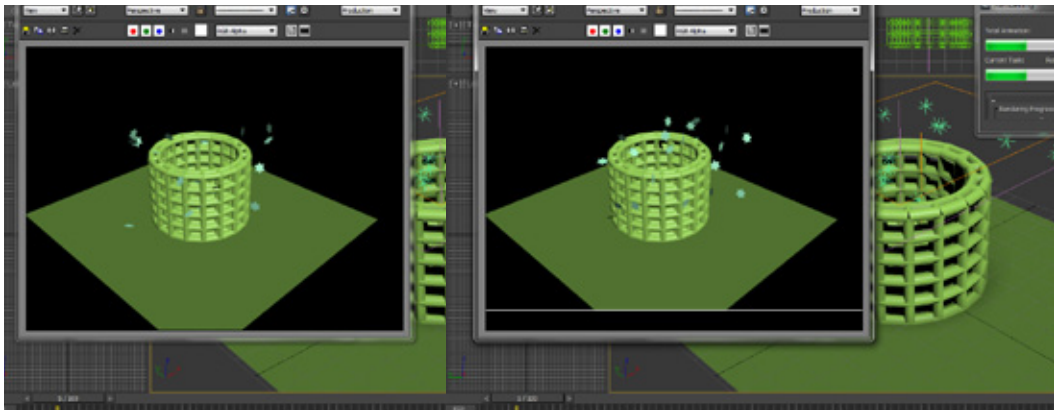
*Cambiar una superficie con la fuerza de movimiento.

*Crear objetos dinámicos.

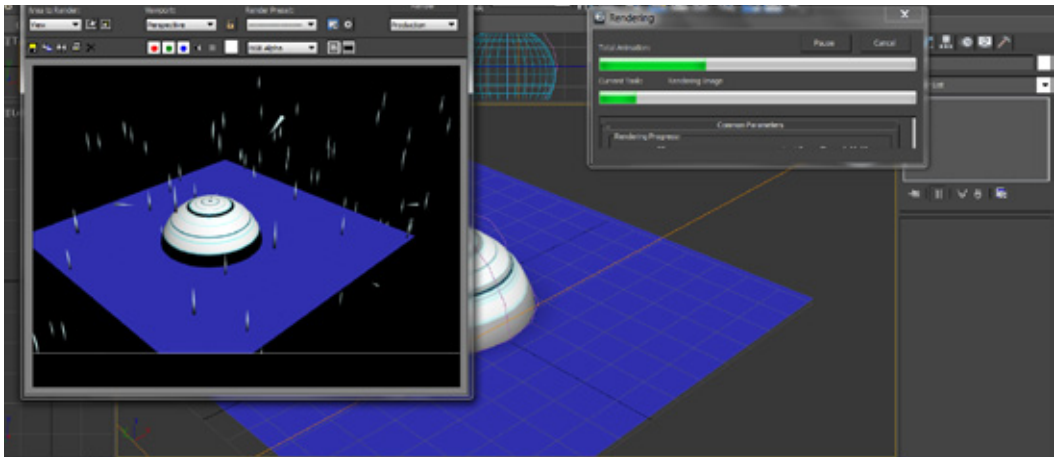
*Modificar objetos dinámicos.

*Simular ropa con el modificador Cloth.

*Aplicar el modificador Morpher.



Escena a la que se le ha aplicado una simulación de la gravedad.



Escena a la que se le ha aplicado un efecto rebote.

2.16 Aplicar efectos a la escena.

*Crear animaciones con el modificador Morpher.

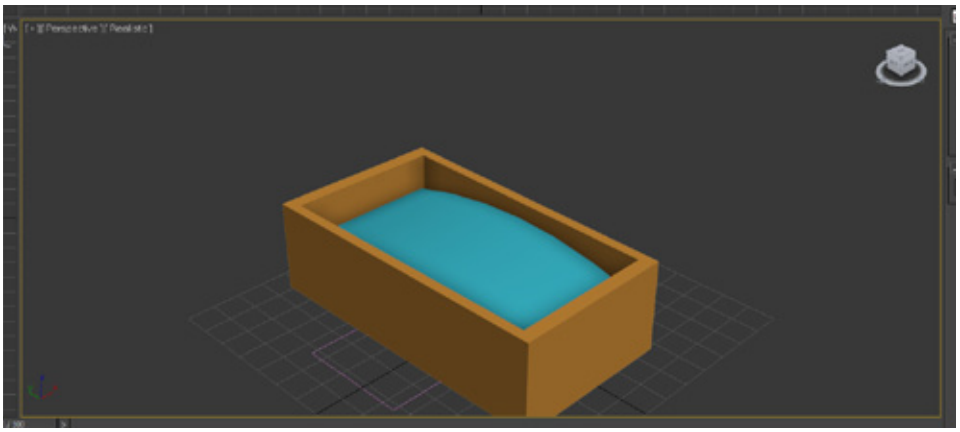
*Añadir objetivos al modificador Morpher.

*Postproducción con Video Post.

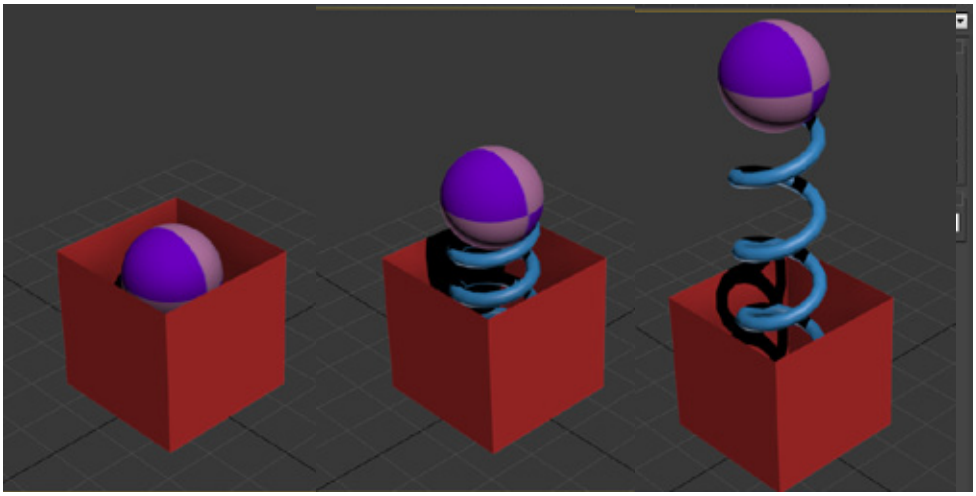
*Configurar los parámetros de un efecto.

*Más efectos Vídeo Post: Cielo estrellado.

*Cambiar el punto de mira con Video Post.



Efecto de cambio de una superficie con la fuerza del movimiento.



Creación de un efecto dinámico.

2.17 Complementos para 3ds MAX.

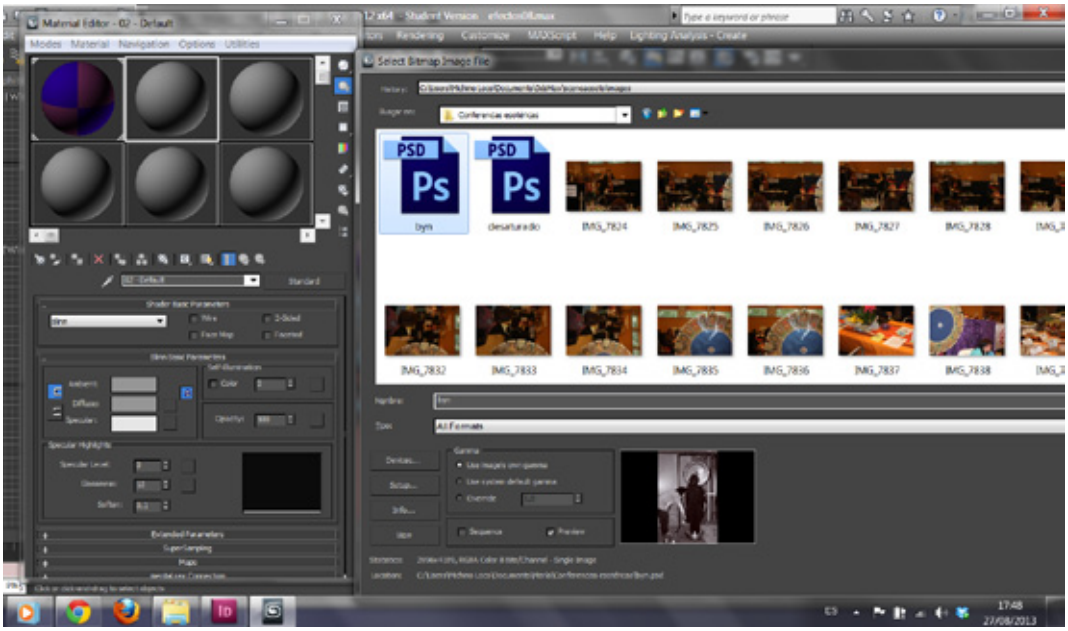
*Añadir audio a una escena.

*Dónde encontrar un plug-in.

*Cómo instalar un plug-in.

*La herramienta Plug-in Manager.

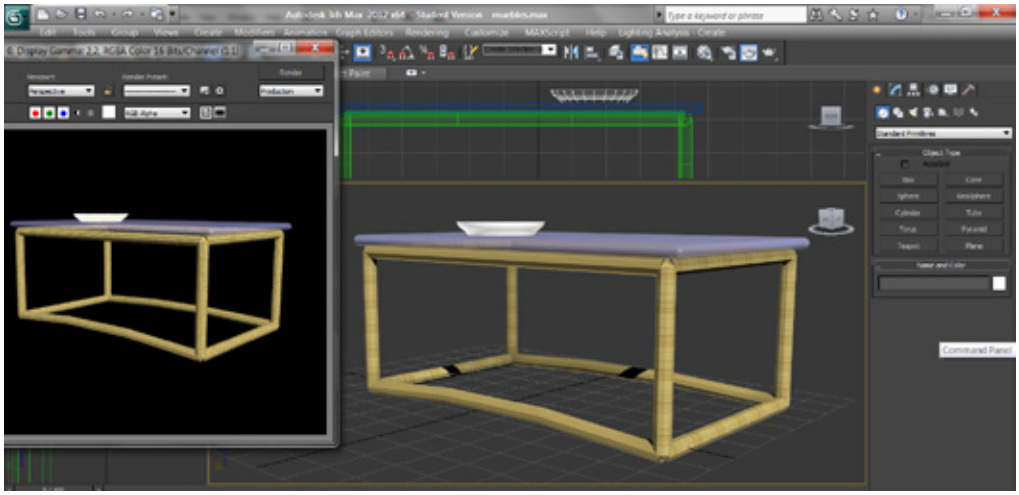
*Enlazar con Photoshop.



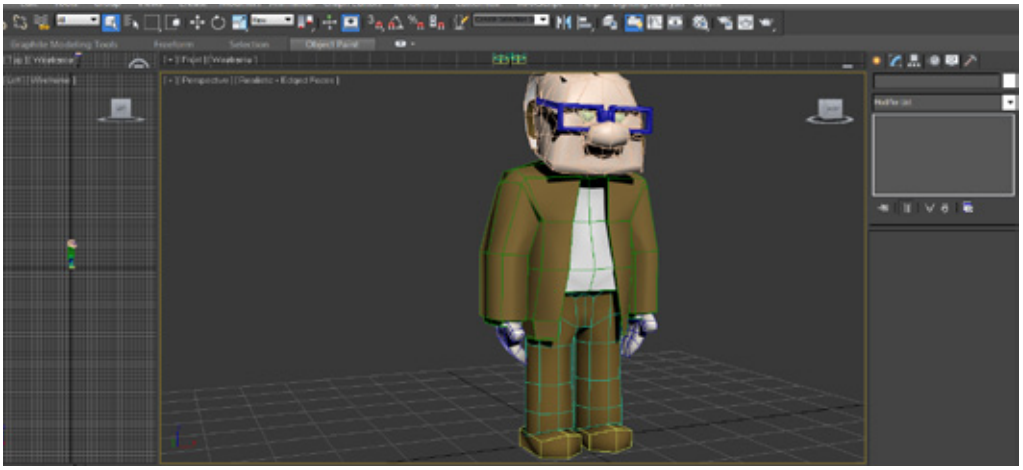
Enlazar con Photoshop (materiales).

2.18 Ejercicios libres 3ds Max.

Una vez terminada la fase de estudio del programa, se hicieron varios ejercicios por libre para practicar con la herramienta. En el primer caso, el modelado y apliación de materiales en mobiliario. En el segundo, el modelado con bajo número de polígonos de un personaje a través de un tutorial encontrado en la Internet.



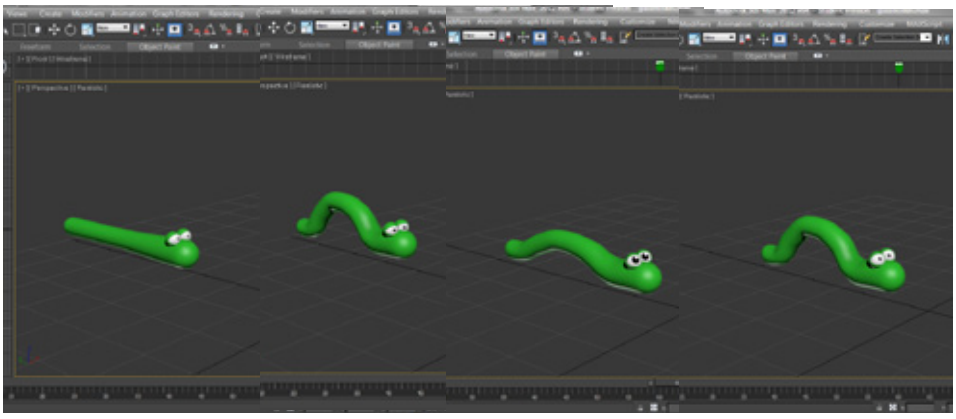
Modelado de una mesa y aplicación de materiales (uno de ellos con tranparencia, el cristal).



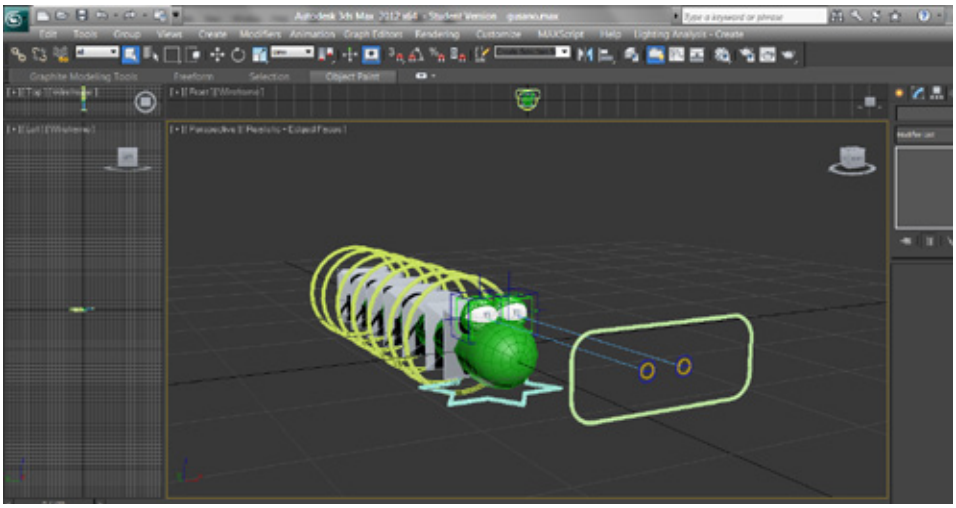
Modelado de un personaje a través de un tutorial en la web. Modelado de bajo número de polígonos.

2.18 Ejercicios libres 3ds Max.

También se realizó el modelado de un personaje simple (un gusano) y se animó a través de un sistema de huesos que se aplicó al modelo a través de Skin y a través del uso de distintos controladores (general, para los huesos, para ojos, párpados y pupila). Esto también se realizó a través de un tutorial en el que se mostraba paso a paso cómo generar la animación correctamente.



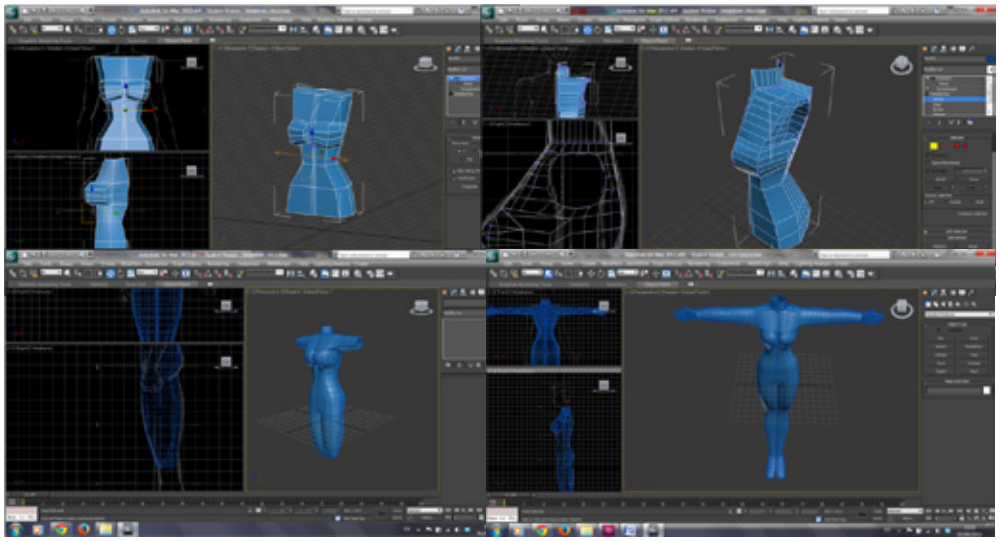
Fotogramas escogidos durante la reproducción de la animación del gusano.



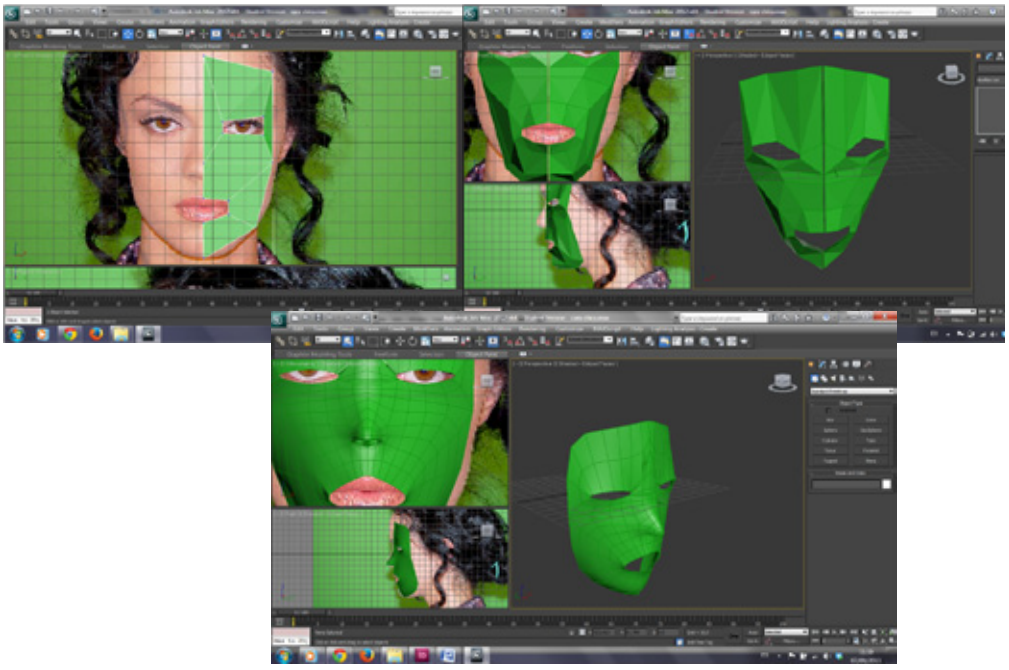
Modelo del gusano con los huesos y los controladores visibles.

2.18 Ejercicios libres 3ds Max.

Se ha profundizado más en el modelado de personajes a través de otros tutoriales en la red, en los que se enseña cómo crear un modelo más realista y detallado.



Cuerpo del modelo a través del modelado poligonal.



Cara del modelo a partir de una imagen a través del modelado poligonal.

APÉNDICE 3

Processing: Aprendizaje sobre la Realidad Aumentada.

Índice del Apéndice 3.

3.0 Introducción.

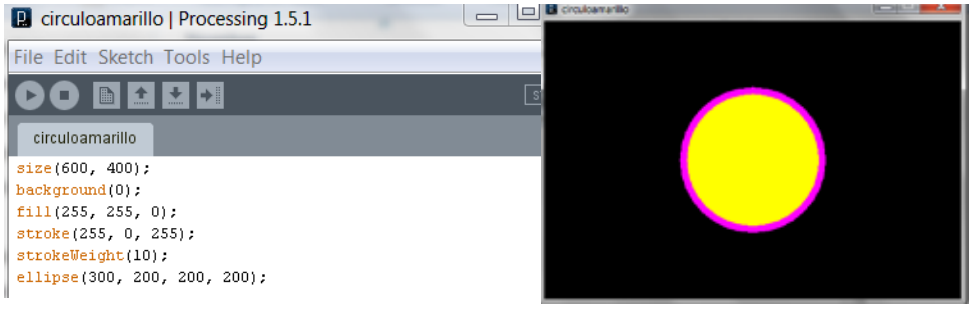
Processing es un entorno de programación de código abierto y basado en Java. Destaca por su facilidad de uso y permite generar sofisticadas aplicaciones gráficas e interactivas al tiempo que minimiza la dificultad asociada a la compilación y generación de software. Por otra parte, tiene acceso a todas las prestaciones de Java.

Processing dispone de bibliotecas para trabajar con Arduino, para implementar aplicaciones RA y para generar proyectos para la plataforma Android, lo que la convierte en un magnífico punto de partida para muchas de las tareas de creación de prototipos.

A lo largo de este apéndice se muestran los ejercicios realizados con este programa y se describe el modo en el que se hicieron.

3.1 Primer programa de Processing.

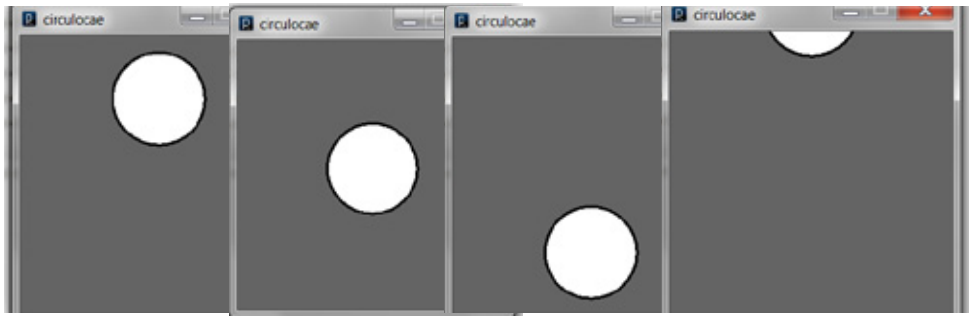
En el primer programa de processing se han aprendido los comandos básicos: crear una ventana con un tamaño y color de fondo y crear una forma geométrica de dterminadas dimensiones, color de relleno y borde con color.



Esfera posicionada en el centro. Se ha indicado el color del borde y del fondo.

3.2 Modo interactivo.

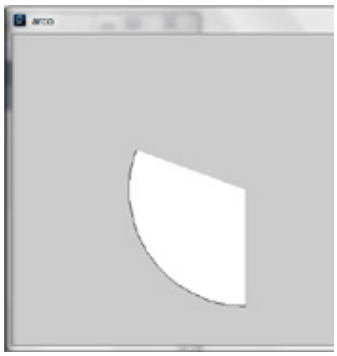
Para que el boceto cambie con el tiempo se utiliza el modo interactivo con las funciones setup(), cuando el código sólo se ejecuta una vez, o draw(). Con esta última, dejando tres valores constantes y uno variable (posición vertical), éste último se va incrementando cada vez en uno hasta un valor máximo, en el que vuelve a su posición inicial. Con esto se consigue una imagen interactiva.



El círculo va incrementando su posición en y hasta llegar al máximo valor, en el que vuelve al valor inicial.

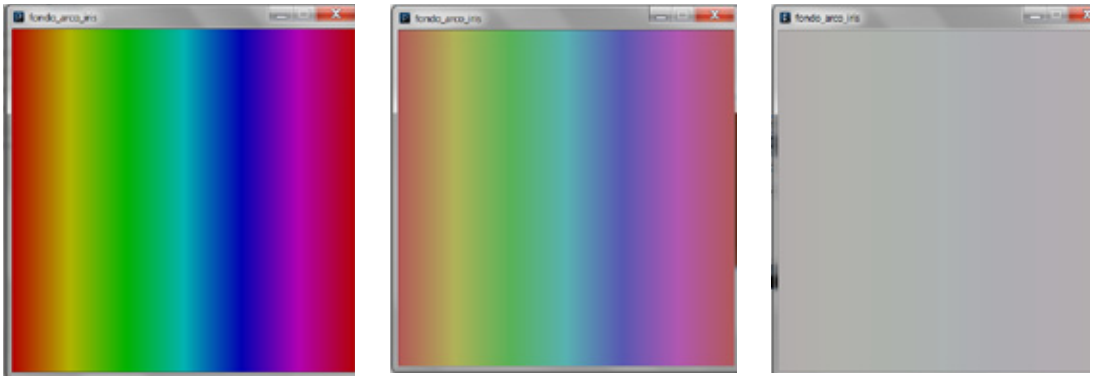
3.3 Dibujar en processing.

Formas primitivas: A través de las siete funciones básicas de formas primitivas 2D (point(), line(), triangle(), quad(), rect(), ellipse() y arc()) se pueden crear dibujos de formas geométricas en la ventan de processing. Cada una de ellas tiene diferentes argumentos para definirlas.



Función arc(), que traza una elipse a través de seis argumentos: posición del arco (dos primeros), anchura y altura de la elipse (los dos siguientes) y ángulos de origen y destino del arco (dos últimos).

Trabajar con color HBS: El modo de color HBS tiene como argumentos el tono, la saturación y el brillo. Esto resulta útil para iterar por los colores del arco iris, ajustar brillo y ajustar saturación. Así, a través de la función stroke, en la que se determina el valor de las tres funciones se obtiene fácilmente el arco iris, así como el cambio en la saturación dependiendo del a posición en x del ratón.



La saturación de los colores va disminuyendo conforme se mueve el ratón por la pantalla de derecha a izquierda. Cuando se llega al extremo de la izquierda (última imagen) los colores se han vuelto completamente grises.

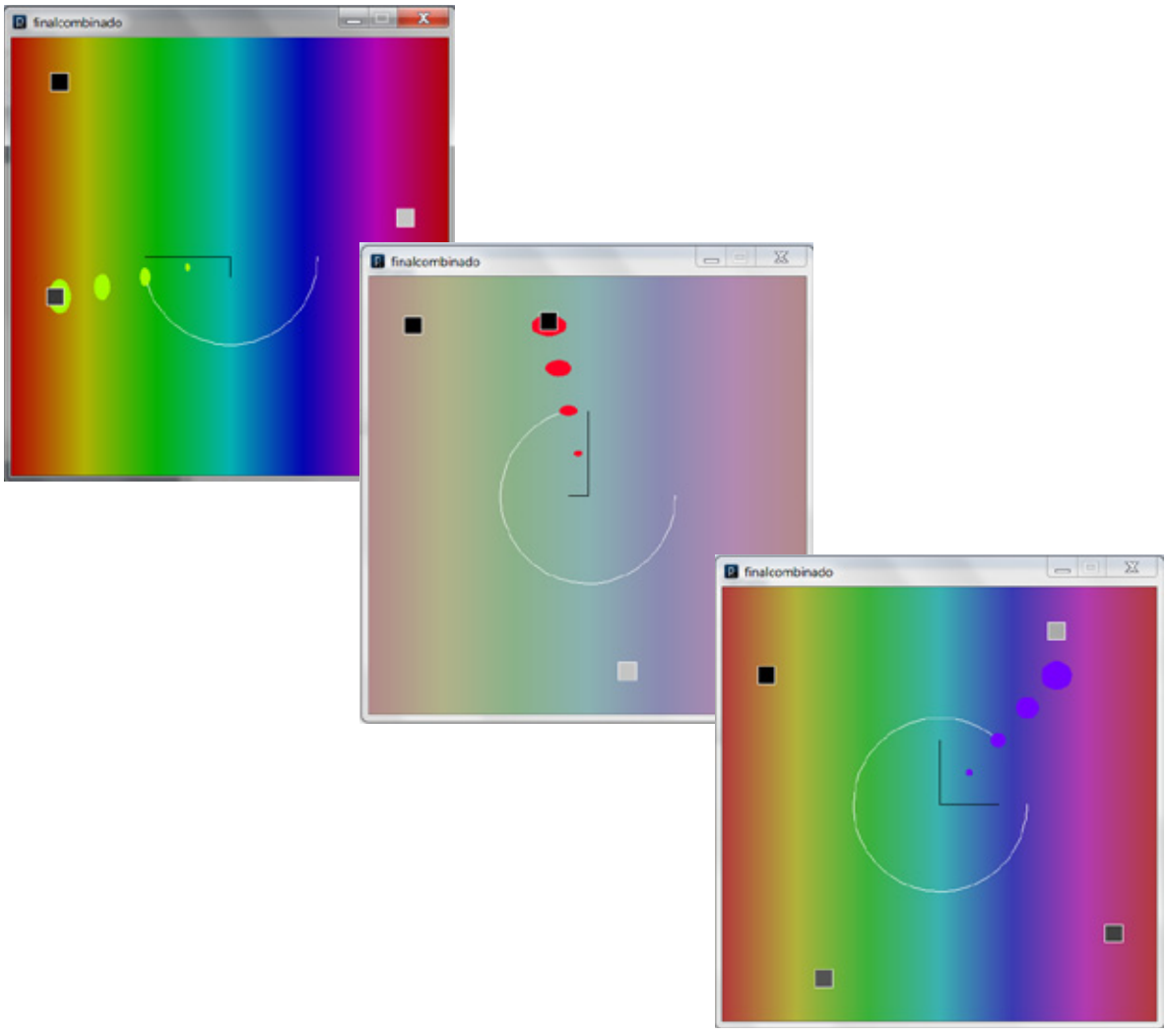
3.3 Dibujar en processing.

Trigonometría sencilla: El conocimiento de las funciones trigonométricas básicas es muy útil para programar con software realacionado con gráficos. En este caso, trabajando con las funciones sin() y cos() se obtiene la siguiente animación: un arco blanco es trazado en ángulos crecientes. Además, se trazan unas líneas negras cuya longitud cambia con respecto al ángulo. Por último, se ha trazado una elipse que gira alrededor de la forma de arco (el uso de senos y cosenos le permite localizar puntos de la trayectoria del círculo en función del ángulo).

En las imágenes se puede ver la progresión del trazado de la circunferencia, junto a las líneas negras marcando el seno y el coseno. La elipse sigue la trayectoria de la circunferencia y va cambiando su tono, también en funión del ángulo.

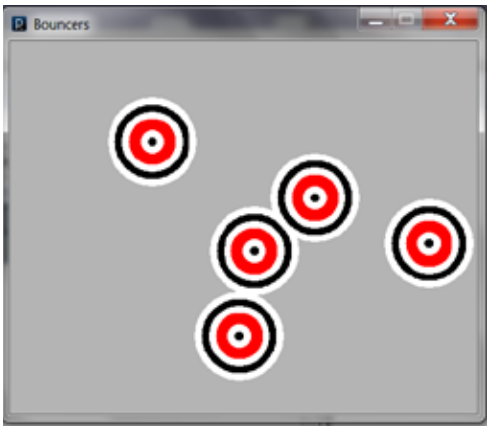
3.3 Dibujar en processing.

Combinar las piezas: Combinando código de los apartados anteriores con otros elementos y usando senos, cosenos y el bucle "for" se ha obtenido el siguiente boceto: una animación de un arco de ángulos cada vez mayores junto a distintos elementos cuya posición y forma cambian en función del ángulo. El fondo es un arco iris que se funde a escala de grises cuando el ángulo es derecho y apunta hacia arriba o hacia abajo.



3.4 Trabajar con clases.

En el boceto se crean puntos que funcionan de forma independiente y esto se controla fácilmente con las clases. Por lo tanto, se ha creado una clase para los puntos con información sobre cómo dibujarlo y variables para almacenar datos de los objetos de la clase. Para ello se ha creado el boceto Spot.



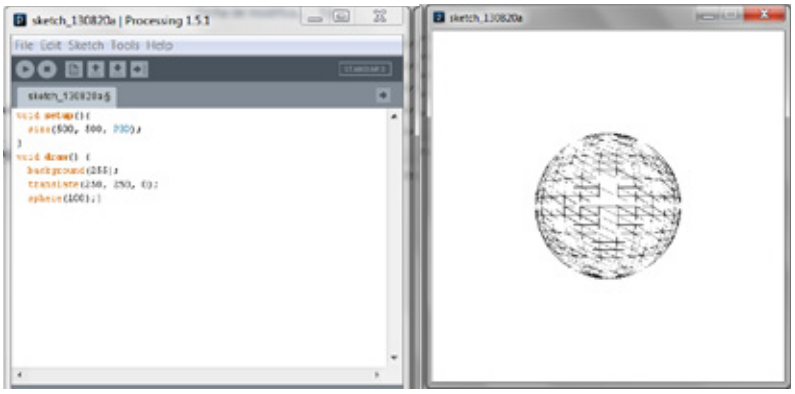
En el boceto se crea un punto con forma de diana cada vez que se hace clic en la ventana, y rebota verticalmente desde el punto en el que se ha creado. Se crean cinco puntos y, a partir de ahí, si se crea otro desaparece el anterior y se crea el nuevo.

Aerógrafo.

A través de un boceto sencillo se puede crear el efecto de un aerógrafo. Se ejecuta una y otra vez la orden de dibujar una elipse en la posición del ratón, con un valor de transparencia en el color de forma que se crea el efecto de estar pintando con un aerógrafo.

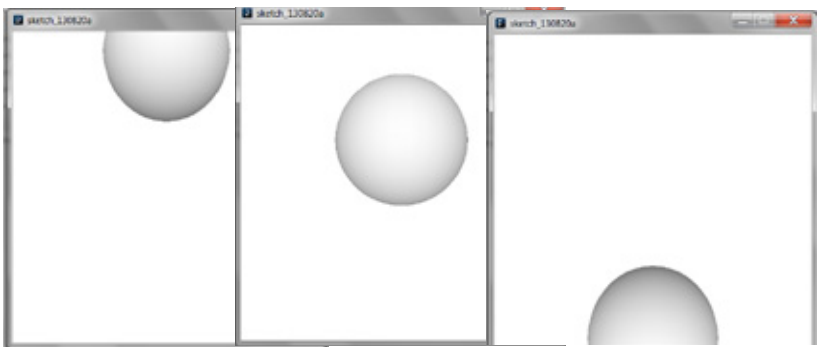
3.5 Primitivas 3D.

Al igual que al programar en 2D, existen formas primitivas para programar en 3D. Existen dos primitivas tridimensionales: sphere y box. En el siguiente ejemplo básico se puede ver el uso de la primitiva sphere. Además, añadiendo el comando translate se puede ubicar la primitiva en el espacio.



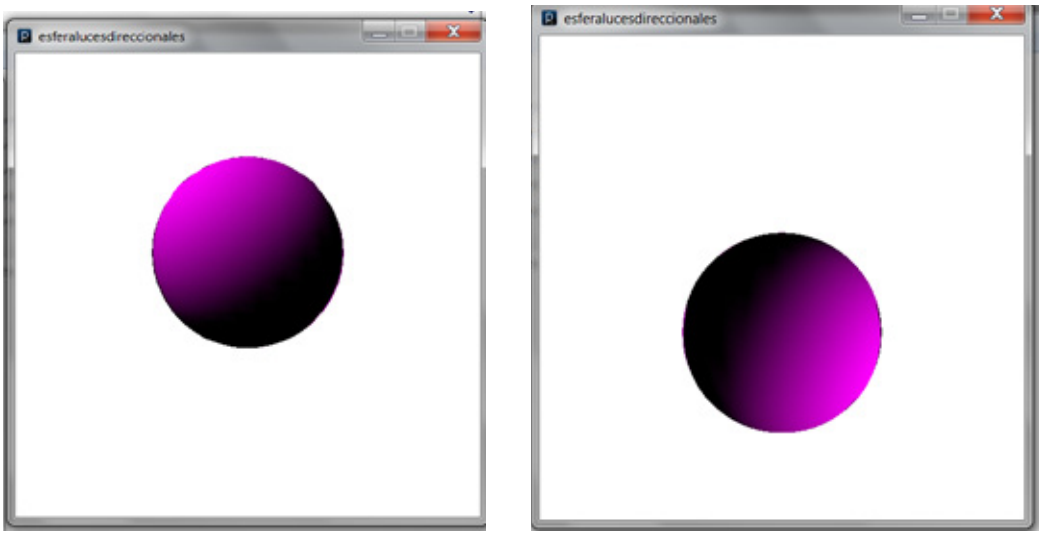
3.6 Luces y animación.

A través del comando lights() se ha añadido la luz a la escena, mientras que el comando noStroke() elimina la malla de la esfera. Además, se le ha dado una posición inicial en "y" fuera de la ventana y se ha creado una función que incremente su valor progresivamente hasta un valor máximo, en el que vuelve al inicial. De eta forma se ha obtenido el efecto de que la esfera cae.



3.7 Luces direccionales.

A través de las luces direccionales se puede tener un mayor manejo de éstas. En este caso, al ejemplo anterior se le ha añadido a través de estas luces la posibilidad de controlar la iluminación a través de la posición del ratón (que funciona como un foco). Este boceto se ha procesado tanto con P3D como con OpenGL.

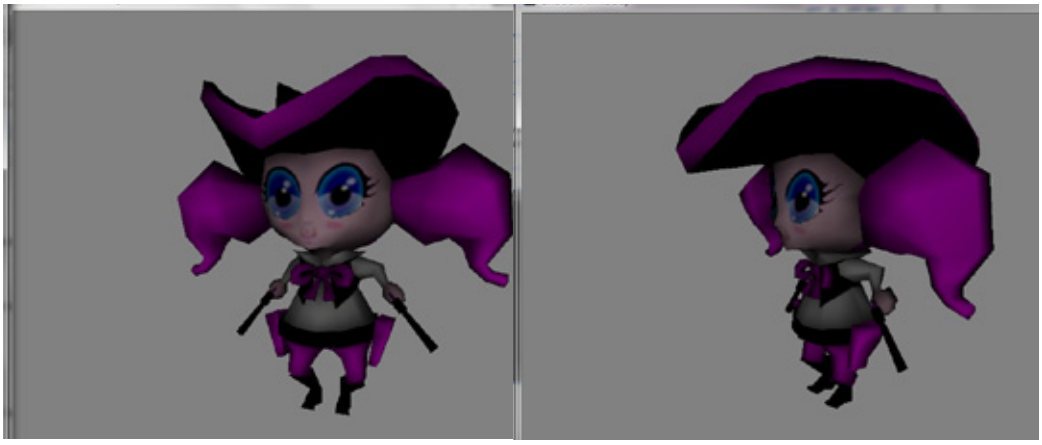


Uso de luces direccionales en una esfera animada. EL ratón es el foco que la ilumina, por lo que la dirección de la luz cambia con su movimiento.

3.8 Trabajar con archivos OBJ.

Utilizando la biblioteca OBJLoader se ha podido realizar el siguiente boceto. A partir de un archivo OBJ con el modelo del personaje, su archivo .mtl y la textura se puede generar un modelo en la ventana del boceto. Para conseguirlo se copian a la carpeta del boceto los tres archivos y se crea el modelo a través del constructor OBJModel.

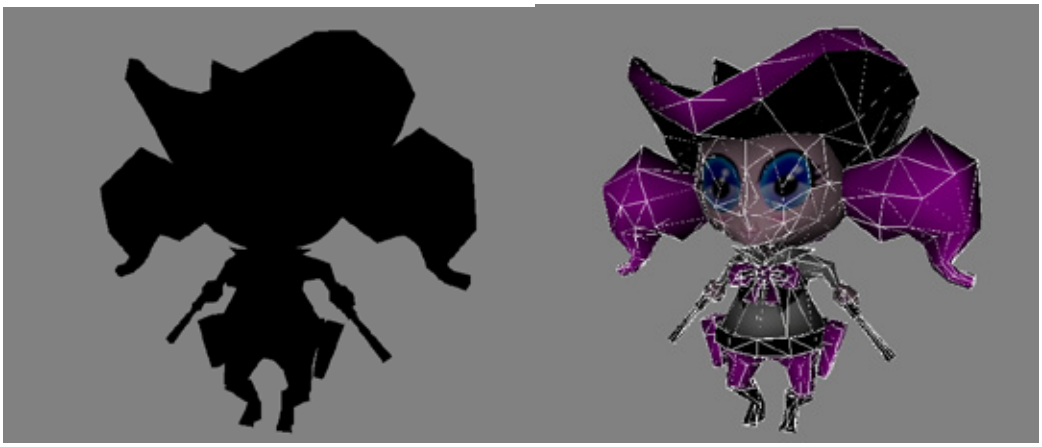
El modelo se ha escalado a través del comando "model.scale()" y se ha garantizado que el centro del espacio se calcula como centro geométrico del modelo a través del comando "translateToCenteer()". Además, se han añadido las líneas de código necesarias para que el modelo pueda ser rotado con el movimiento del ratón.



Modelo creado a partir de los archivos sa.obj (modelo), sa.mtl (materiales) y shoot-inanniecolor.jpg (imagen de la textura de color del material). Haciendo clic con el ratón y moviéndolo por la pantalla se puede rotar el modelo en todas las direcciones.

3.8 Trabajar con archivos OBJ.

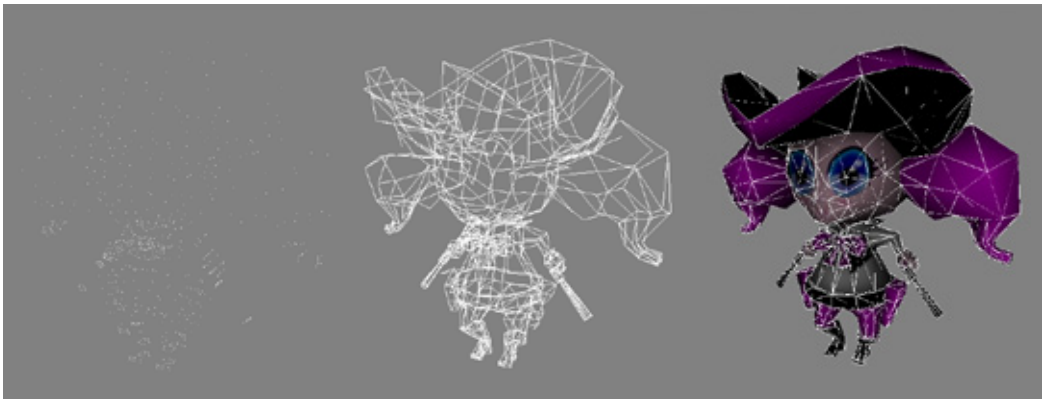
Modos de dibujo OBJ: Esta biblioteca implementa distintos modos de dibujar el modelo. A partir de la función keyPressed(), se pueden ilustrar los distintos modos en tiempo real. Así, al ejecutar el boceto, pulsando la tecla "t" se puede seleccionar entre activar o no la textura. Del mismo modo, pulsando la tecla "s" se puede visualizar o ocultar la malla del modelo. Por último, a través de las teclas 1, 2 y 3 se puede seleccionar el modo de dibujo entre POINTS (sólo dibuja los vértices como puntos de colores), LINES (sólo dibuja los bordes) y TRIANGLES (dibuja caras triangulares con propiedades y texturas).



La imagen más a la izquierda del modelo muestra el efecto que se produce al pulsar la tecla "t" una vez ejecutado el boceto (desactiva la textura). La imagen de la derecha muestra el efecto de la tecla "s", en el que se ve el modelo con la malla. En ambos casos, al volvera a pulsar la tecla del modelo vuelve a sus condiciones originales.

3.8 Trabajar con archivos OBJ.

Modos de dibujo OBJ: Esta biblioteca implementa distintos modos de dibujar el modelo. A partir de la función keyPressed(), se pueden ilustrar los distintos modos en tiempo real. Así, al ejecutar el boceto, pulsando la tecla "t" se puede seleccionar entre activar o no la textura. Del mismo modo, pulsando la tecla "s" se puede visualizar o ocultar la malla del modelo. Por último, a través de las teclas 1, 2 y 3 se puede seleccionar el modo de dibujo entre POINTS (sólo dibuja los vértices como puntos de colores), LINES (sólo dibuja los bordes) y TRIANGLES (dibuja caras triangulares con propiedades y texturas).

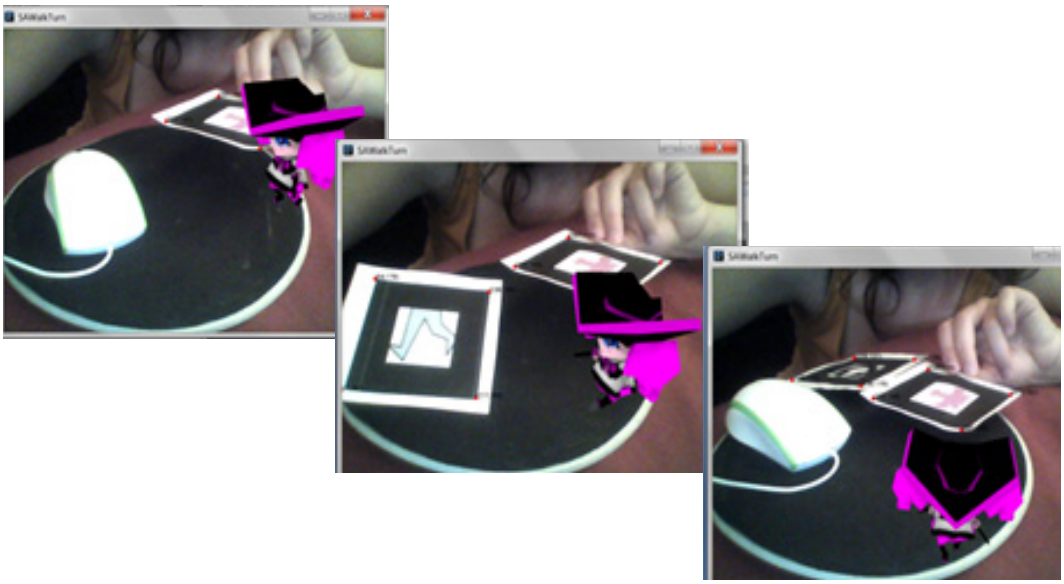


De izquierda a derecha las imágenes del modelo muestran el efecto que se produce al pulsar las teclas 1, 2 y 3: dibujo únicamente de los vértices como puntos de color en la ventana del boceto, dibujo de los bordes del modelo y dibujo de las caras triangulares con propiedades y textura.

3.9 Animaciones sencillas con matrices OBJ, biblioteca NyAR4psg y uso de marcadores.

Para obtener la animación en el boceto se han utilizado archivos OBJ independientes que se cargan como fotogramas. Con estos archivos y utilizando la biblioteca NyAR4pgs, que se he descargado a parte del programa, se ha obtenido la siguiente aplicación de Realidad Aumentada.

En ella, a través de tres marcadores distintos, se puede ver el modelo, controlar el que se anime y controlar la rotación. Así, con el primer marcador se genera el modelo y cuando aparece el segundo, el modelo comienza a andar. Si éste desaparece el modelo vuelve a ser estático. Por último, al introducir el tercer marcador y girarlo, el modelo rota con él. Pueden colocarse los tres marcadores a la vez, es decir, que se anime el modelo y rote al mismo tiempo.



Annie estática (un marcador), annie animada (dos marcadores) y annie girando con el marcador (dos o tres marcadores).

3.10 Problemas durante la realización de los ejercicios.

En la realización del último boceto se tuvieron muchos problemas para lograr que funcionara. La cámara no se activaba, y cuando lo hacía no reconocía los marcadores.

La solución para que funcionar fue descargar otra versión de Processing (Processing 1.5.1) que fuera compatible con la biblioteca. Además, se tuvo que cambiar la ubicación de la carpeta NyAR2 (la biblioteca necesaria para la aplicación), ya que la ubicación que indicaba el libro de estudio no daba buen resultado. Por último fue necesario descargar complementos para la cámara, como "cleyemulticam".

APÉNDICE 4

Aplicación a realizar y plataforma para su desarrollo.

Índice del Apéndice 4.

4.0 Introducción.

En este apéndice se resume el estudio que se ha realizado sobre el tipo de aplicaciones de Realidad Aumentada que existen en el mercado. A continuación se proponen distintas alternativas para la aplicación a desarrollar para este Trabajo de Fin de Grado y se selecciona una de ellas.

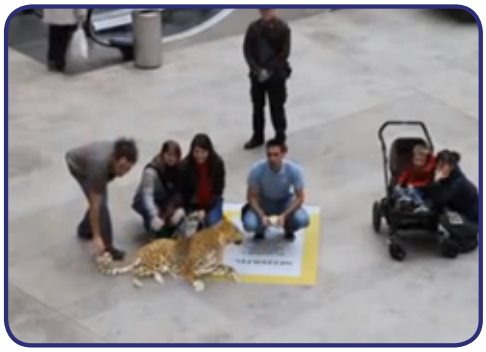
También se muestra el estudio sobre las distintas plataformas con las que podría desarrollarse la aplicación y se selecciona la más adecuada. Se finaliza ampliando la información sobre la plataforma seleccionada para este proyecto.

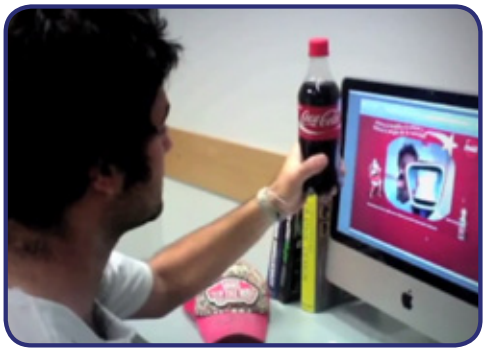
4.1 Aplicaciones realidad aumentada.

La Realidad Aumentada es una tecnología versátil, que puede utilizarse en ámbitos muy diferentes. Algunas de las aplicaciones más características encontradas son las siguientes.

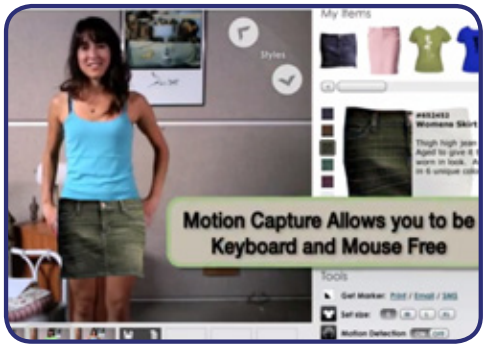
-Muestra de de maquetas: Arquitectura, productos...

-Publicidad: En centros comerciales, en productos...



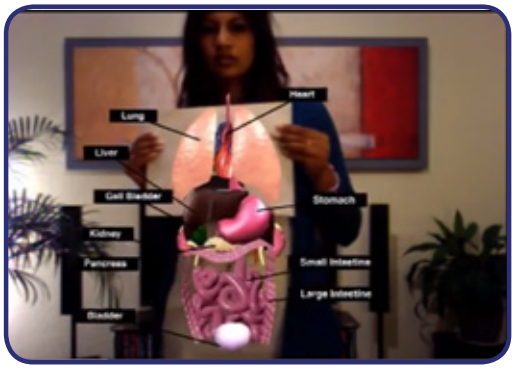


-Visualización de productos: Ropa, gafas, relojes, mobiliario...



4.1 Aplicaciones realidad aumentada.

-Aprendizaje: tanto a nivel de usuario como de trabajador.



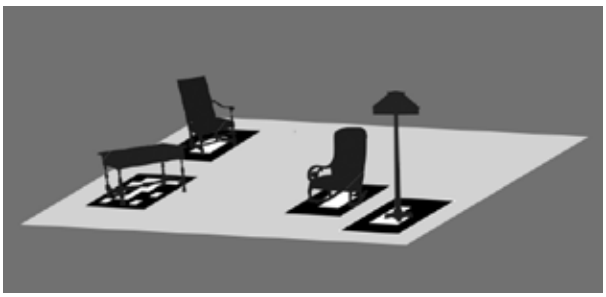
-Ampliación de información: En libros, revistas, periódicos...

-Otras: entretenimiento (juegos), localización de lugares.

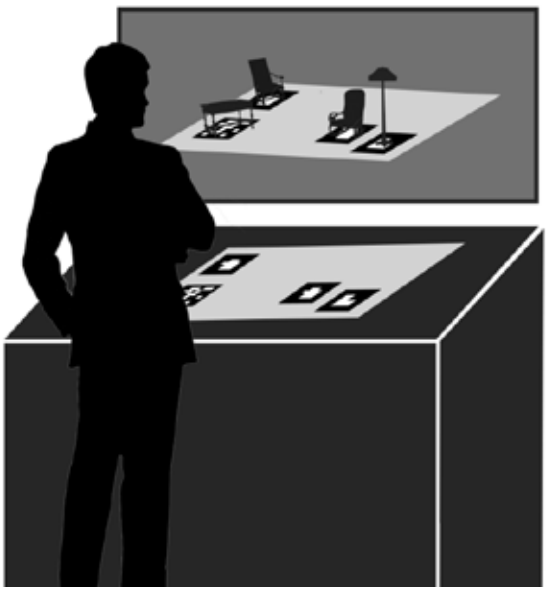
4.2 Propuestas de aplicaciones.

1. Distribución de muebles.

Se trata de una aplicación destinada a tiendas de muebles, que ofrece la posiblidad a los compradores de generar sus propias habitaciones virtualmente, combinando los distintos muebles del catálogo y visualizando su aspecto a través de la pantalla de un ordenador. La aplicación utilizaría marcadores asociados a distintos muebles. Los usuarios serían personas adultas, de ambos sexos, clientes de la tienda. Esta aplicación podría enseñarse en la tienda, y dar luego la posibilidad de utilizarla desde casa, permitiendo la descarga e impresión de los marcadores desde la web de la empresa. La aplicación utiliza marcadores y se muestra a través de la pantalla de un ordenador.



Ejemplo de aplicación. Cada mueble está asociado a un marcador.

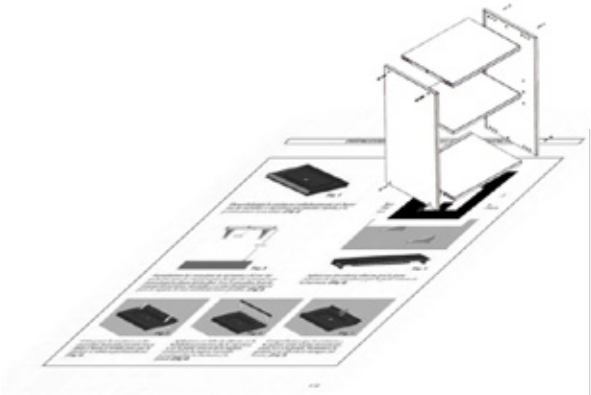


Ejemplo de uso de la aplicación.

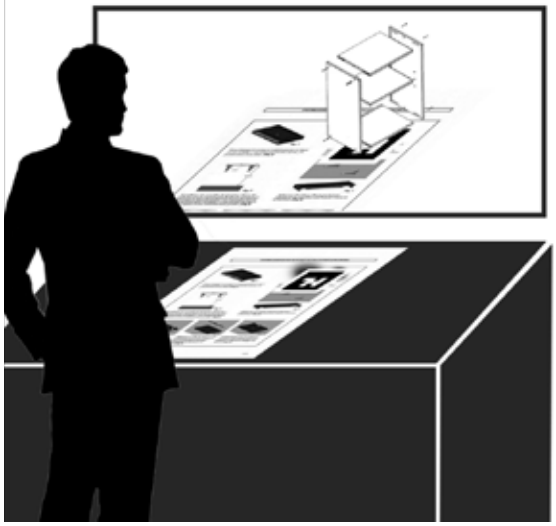
4.2 Propuestas de aplicaciones.

2. Ayuda en el montaje de un producto.

Esta aplicación está destinada a ayudar al usuario en el montaje de un producto, ya sea un mueble, juguete, maqueta, etc. En el propio manual de instrucciones hay impresos marcadores asociados a animaciones con el montaje del producto. De esta manera, cuando el usuario quiere ayuda mientras trabaja en el producto, coloca el manual delante del ordenador y aparece en la pantalla el montaje del mismo. Esta aplicación está destinada a personas adultas, de ambos sexos, y su entorno de uso es la casa del usuario. Se basa en el uso de marcadores y se utiliza con ordenadores. A continuación se muestra el ejemplo con un manual de instrucciones sobre el montaje de una estantería.



Ejemplo de la aplicación.



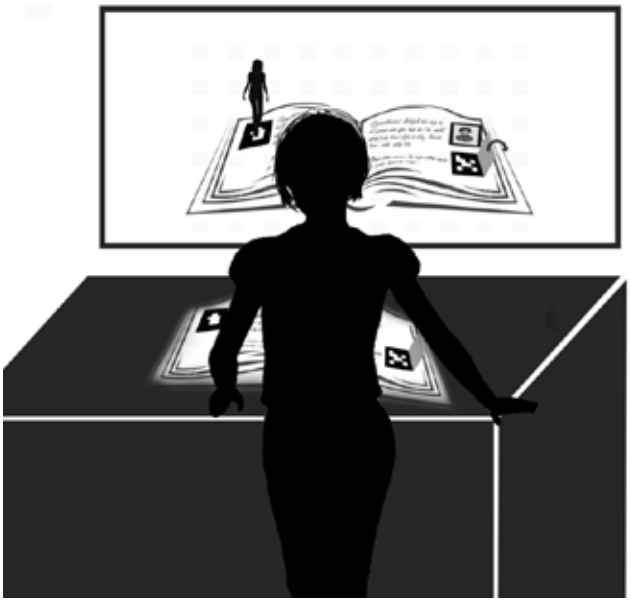
Ejemplo de uso de la aplicación.

4.2 Propuestas de aplicaciones.

3. Libro para niños de entre 7 y 9 años interactivo.

Se trata de un libro para niños enriquecido con contenidos de Realidad Aumentada. A través de los marcadores impresos en las páginas, se completa la información del libro con modelos 3D, animaciones, sonido, etc. proporcionando un nuevo nivel de interactividad. El usuario al que está destinada esta aplicación son niños entre 7 y 9 años, de ambos sexos y su entorno de uso es doméstico. La aplicación utiliza marcadores impresos y se visualiza a través de la pantalla de un ordenador.

Ejemplo de la aplicación.



Ejemplo de uso de la aplicación.

4.3 Aplicación seleccionada.

Libro para niños de entre 7 y 9 años interactivo.

La aplicación que se ha seleccionado es la tercera, el libro para niños interactivo, ya que permite trabajar con una mayor variedad de aplicaciones de Realidad Aumentada (incluir sonido, imágenes, modelos, etc.) y proporciona mayor flexibilidad en cuanto al modelado y las animaciones (objetos, personajes, efectos, etc).

Se busca que el niño juegue a la vez que lee una historia, por lo que se quiere realizar un libro que permita la interacción directa con las páginas: abriendo sobres, levantando solapas, etc. que desubran los modelos.

A continuación se muestran algunas imágenes de libros de este estilo, que servirán de referencia para diseñar el libro propio de la aplicación.

4.4 Unity.

Unity es un motor de videojuego multiplataforma creado por Unity Technologies. Sus características más destacables son las siguientes:

-Muy buena calidad de las animaciones. Es fácil de hacer gráficos de gama alta.

-Sencillo de utilizar.

-3D y RA en un mismo programa.

-Intuitivo.

-Permite vista en tiempo real.

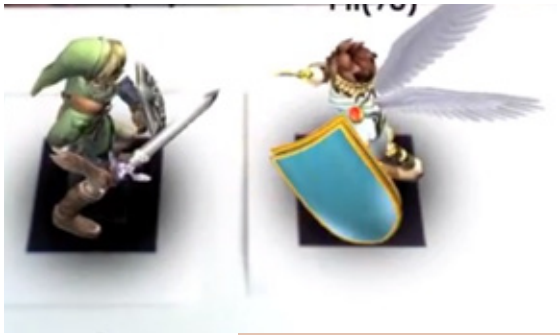
-Para aplicaciones para escritorio y móviles.

-Tiene librerías con marcadores incorporados.

-Gratis la versión que no es pro.

-Tutoriales.

-Unity soporta la integración con 3ds Max, Maya, Softimage, Blender, Modo, ZBrush, Cinema 4D, Cheetah3D, Adobe Photoshop. Los cambios realizados a los objetos creados con estos productos se actualizan automáticamente en todas las instancias de ese objeto durante todo el proyecto sin necesidad de volver a importar manualmente.



Ejemplo de dos modelos en realidad aumentada realizados con Unity. Los modelos están animados y reaccionan ante el ataque de su oponente. La calidad gráfica de los modelos es muy buena.

4.5 Vuforia.

SDK gratuito para desarrollar aplicaciones de realidad aumentada que funcionen en dispositivos Android y iOS (iPhone/iPad).

-Está más orientado a aplicaciones para móviles.

-Permite la inclusión en sus aplicaciones de botones virtuales.

-Puede trabajar tanto con imágenes planas como con elementos en 3D.

-Trabaja a tiempo real.

-Herramienta fácil de entender y rápida compilando.

-Es compatible con tipos de target 2D y 3D, incluidos aquellos que funcionan sin marcador y configuración de Multi-Target 3D.

-Facilidad para trabajar con Unity, ya que posee un plug-in para este programa.

Realidad aumentada con Vuforia. Aplicación para tablet.

4.6 Metaio.

SDK para el aplicaciones de realidad aumentada generado por una empresa privada de realidad aumentada.

-Provee software para PC, web, móviles e instalaciones offline.

-Capacidad para simular la gravedad, el reconocimiento de objetos 3D y las máscaras para ocultar objetos virtuales.

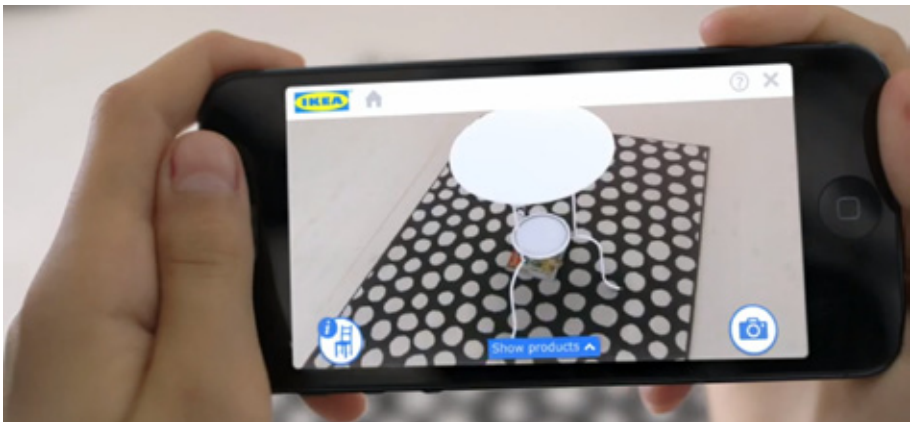
-El SDK para aplicaciones de realidad aumentada es gratuito, pero lleva marca de agua.

-Motor de renderizado en 3D poderoso.

-Capaz de trabajar con imágenes 2D y con modelos 3D.

-Capacidad de seguimiento sin necesidad de marcador.

-Facilidad para trabajar con Unity, ya que posee un plug-in para este programa.



Realidad aumentada con Metaio. Aplicación para móvil (iPhone).

4.7 FLARManager.

FLARManager es una plataforma ligera para crear aplicaciones de realidad aumentada fácilmente con Flash.

-Pensado especialmente para navegadores (no suele utilizarse con móvil).

-Es compatible con una variedad de librerías y espacios de trabajo en 3D.

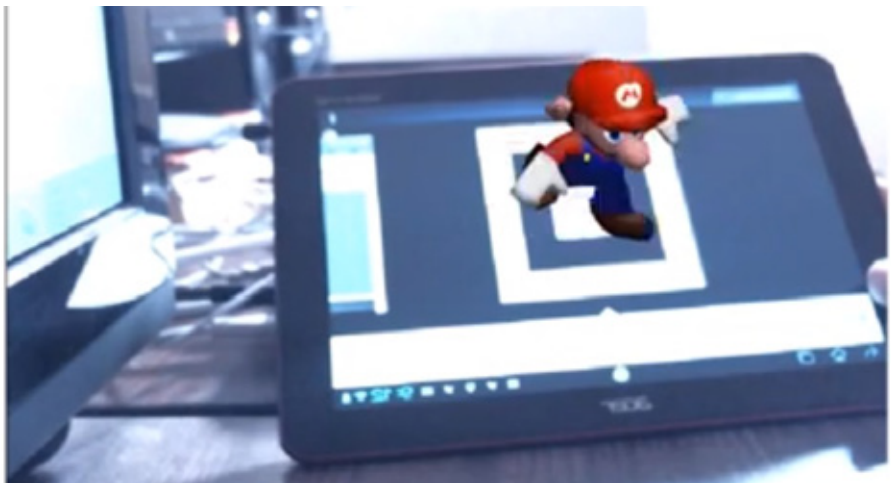
-Provee un sistema más robusto de bases de eventos para manejar la adición, movimiento y eliminación de marcadores.

-Soporta la detección y manejo de múltiples patrones, así como múltiples marcadores de un mismo patrón.

-Las librerías que soporta son: flare*tracker, flare*NFT y FLARToolkit.

-Las plataformas de 3D que soporta son: Alternativa3D, Away3D, Away3D Lite, Papervision3D, Sandy3D.

-Flash Builder y Flash Develop son los entornos en los que suele utilizarse.



Realidad aumentada con FLARManager y Flash Builder. Aplicación para navegador.

4.8 Nyartoolkit.

NyARToolKites una clase de librería visual de realidad aumentada de ARToolKit. Esta librería provee el API para la visualización de la realidad aumentada. NyARToolKitcorre sobre diversas plataformas virtuales: Java, C#, Actionscript3; y además algunos proyectos derivados para flash, Silverlight, Processing y Android.

-Basado en seguimiento de marcadores.

-Soporta un gran número de formatos de imagen.

-Tanto para plataformas de escritorio como para móviles.

-Plug-ins para FLARManager y para Unity.

-Función de etiquetado más rápida.

-Puede operar en diferentes plataformas y sistemas operativos.

Realidad aumentada a través de Nyartoolkit.

4.9 Plataforma seleccionada: FLARManager+Nyartoolkit.

Finalmente la plataforma seleccionada ha sido FLARManager, dado que la aplicación que se ha pensado realizar (libro enriquecido con contenidos de realidad aumentada) está pensada para ser visualizada a través de ordenador. FLARManager junto al plugin de Nyartoolkit (FLARToolkit) forma una plataforma potente para el desarrollo de la aplicación, además de que se posee mucha información sobre la misma y numerosos ejemplos que pueden facilitar el desarrollo del trabajo. Al no buscarse una aplicación para móvil, el que esta plataforma no esté bien preparada para este dispositivo no es un problema.

4.10 FLARManager: Información de la plataforma elegida.

Como se ha dicho anteriormente, FLARManager es una plataforma ligera que facilita el trabajo con Flash para la realización de aplicaciones de Realidad Aumentada. FLARToolKit es una de las familias de puertos ARToolkit basadas en el puerto NyARToolKit, y esta es la librería que se utiliza en FLARManger.

FLARManager incorpora una selección de bibliotecas de seguimiento, incluida FLARToolKit y otras. También puede trabajar con distintas estructuras 3D para Flash, como:

-Alternativa3D.
-Away3D.
-Away3D Lite.
-Papervision3D.
-Sandy3D.

De este modo cuenta con diversas opciones para elegir la combinación correcta para la aplicación que se desee crear y las herramientas disponibles. Esta flexibilidad es una gran ventaja. Además, las herramientas se han agrupado en un paquete sencillo de usar.

Para usar la estructura de FLARManager se necesita un entorno de programación adecado para Flash. La opción que se ha escogido es el programa Flash Builder

4.10 FLARManager: Información de la plataforma elegida.

Las librerías que soporta son: flare*tracker, flare*NFT y FLARToolkit. Dado que se quiere trabajar junto a NyartoolKit, la librería que va a utilizarse para la aplicación será FLARToolkit.

FLARManager es una plataforma destinada a realizar aplicaciones para ordenador. No suele utilizarse para aplicaciones con móviles.

Provee un sistema muy robusto de bases de eventos para mejorar la adición, movimiento y eliminación de marcadores y soporta la detección y el manejo de múltiples patrones. Permite trabajar con varios marcadores de forma simultánea fácilmente, por lo que es muy bueno para aplicacione multimarcador.

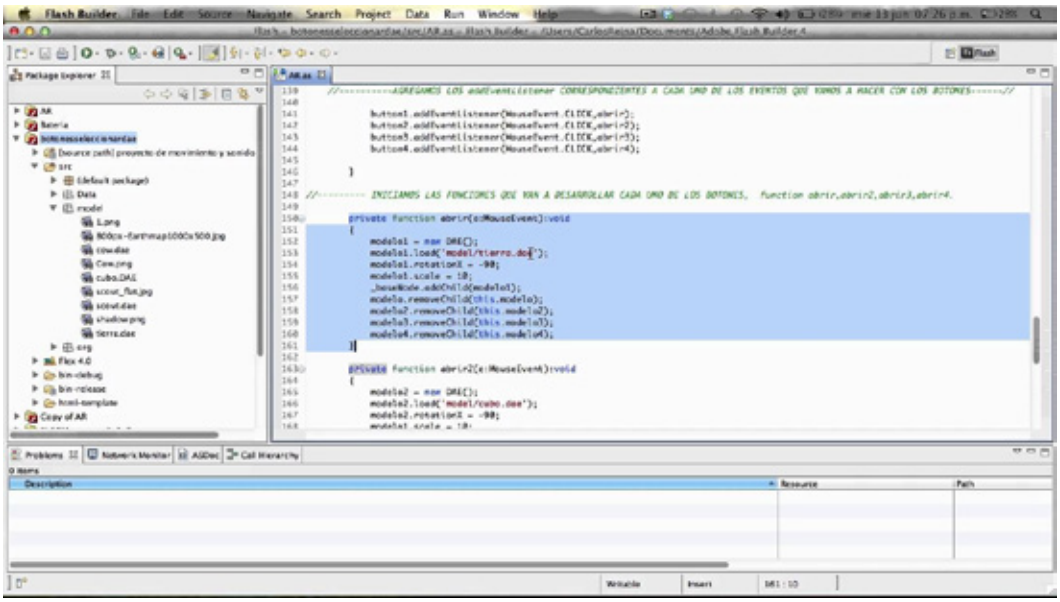
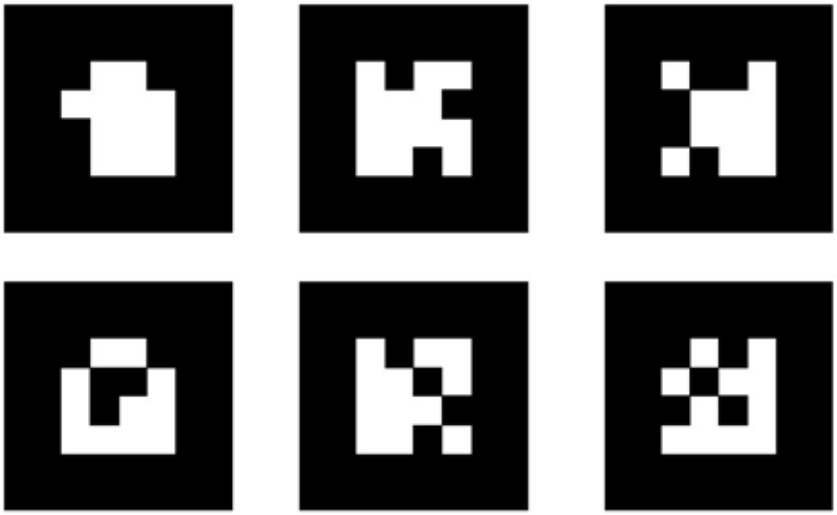


Imagen del espacio de trabajo de Flash Builder.

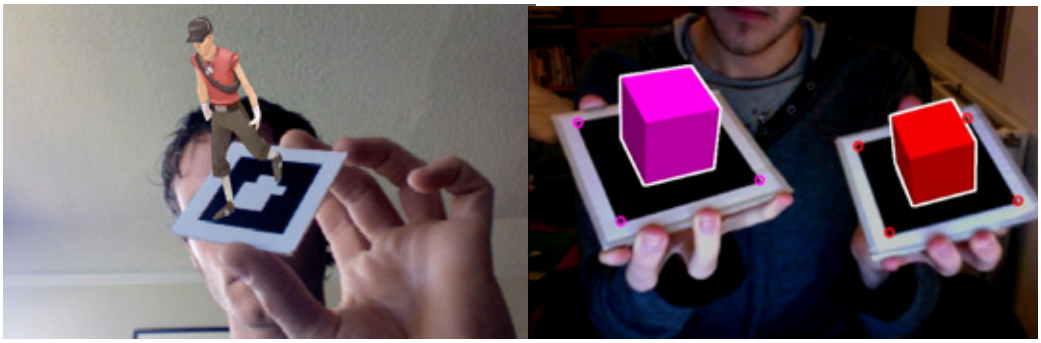
4.10 FLARManager: Información de la plataforma elegida.

Los marcadores utilizados en FLARManager tienen este aspecto:



Marcadores utilizados por FLARManager para las aplicaciones de RA.

A continuaión se muestran algunos ejemplos de aplicaciones realizadas con FLARManager. La primera permite que se muestre un modelo 3D complejo y animado sobre un marcador. En el segundo se muestran dos modelos simples sobre distintos marcadores, de forma simultánea.



Ejemplos de aplicaciones realizadas con FLARManager.

APÉNDICE 5

Estudio de las capacidades de FLARManager.

Índice del Apéndice 5:

5.0 Introducción.

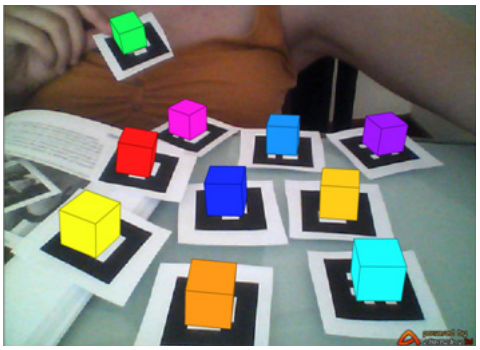
En este apéndice se muestra el estudio de las posibilidades que ofrece la plataforma FLARManager.

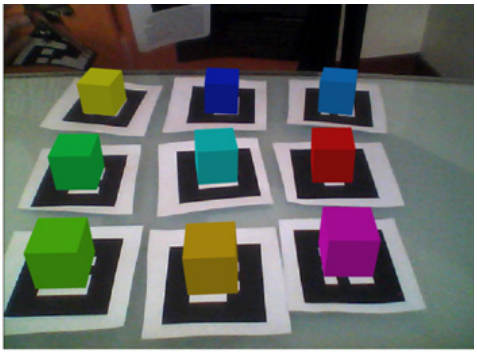
Se ha realizado estudiando los ejemplos que ofrece la propia plataforma, a través de búsquedas en la web, y modificando los ejemplos originales.

Las aplicaciones que se han considerado de mayor interés se han probado en el propio ordenador (algunas de ellas han sido generadas a partir de información encontrada en la web, tomando como base el código original de la aplicación), y las que no se han visto muy interesantes para la aplicación (y que no ofrecía la plataforma) sólo se han documentado.

5.1 Número de marcadores con bajo número de polígonos.

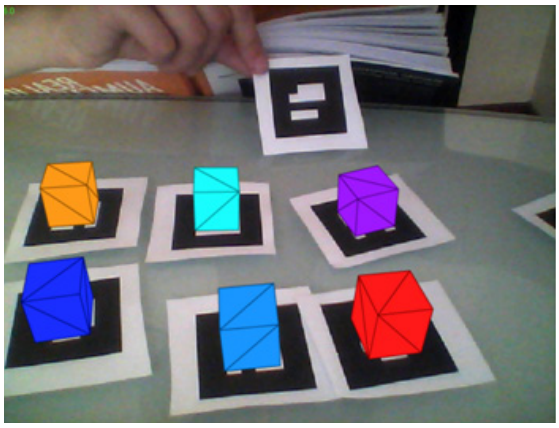
Admite un gran número de marcadores. Se han podido colocar hasta diez simultáneamente sin que dé problemas, a partir de ahí, siguen mostrando marcadores nuevos, pero los que había empiezan a parpadear o desaparecen (en el caso de que se muestren todos a la vez, si se quitan marcadores antiguos y se meten nuevos no da problemas).





5.2 Número de marcadores con número de polígonos medio.

En otra prueba realizada con cubos de más polígonos, se ha comprobado que acepta hasta seis marcadores simultáneos. Al introducir un séptimo (primera imagen), este no muestra la imagen del cubo.

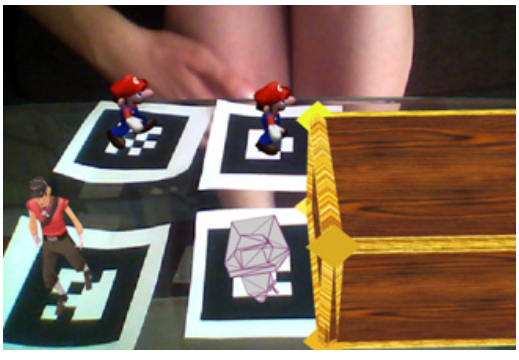


5.3 Modelo de un gran número de polígonos.

Admite modelados de un mayor número de polígonos.

5.4 Prueba con múltiples marcadores asociados a distintos modelos complejos.

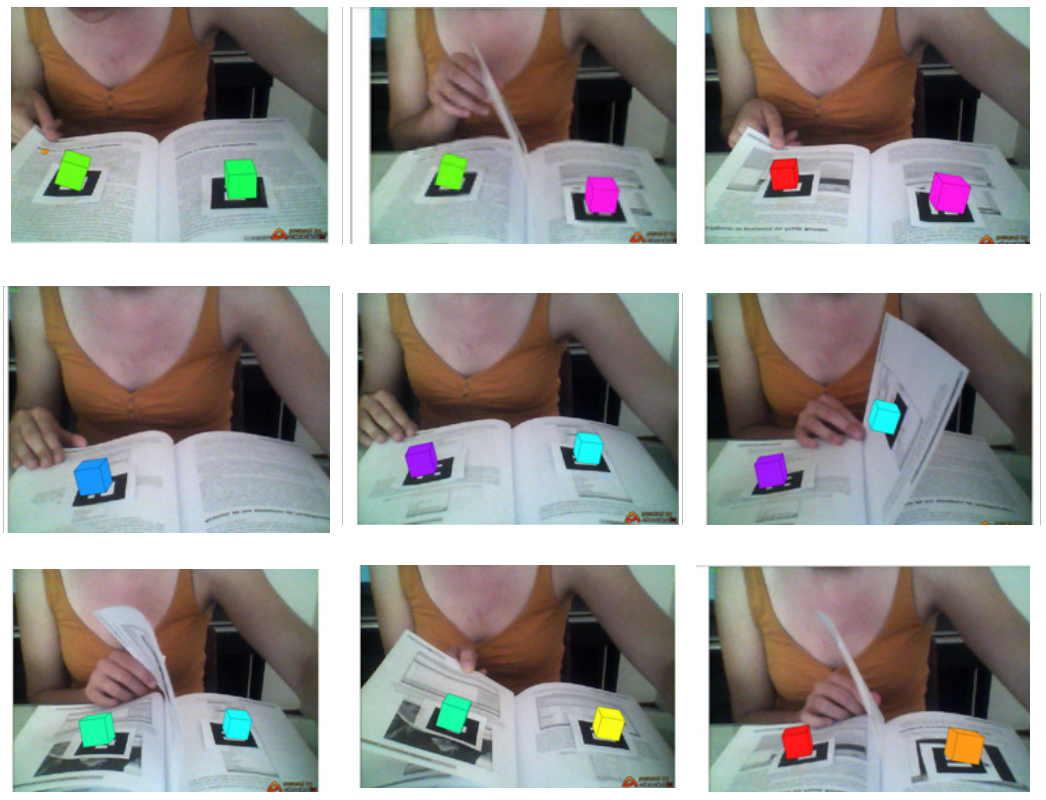
Transformando el código de uno de los ejemplos del libro de RA se ha conseguido obtener varios modelos simultáneamente. Se ha conseguido visualizar correctamente hasta seis modelos simultáneamente asociados a distintos marcadores. A partir de este número, se producen problemas en la imagen de los modelos (parpadeos) o algunos desaparecen al mover su marcador. Si los modelos pasan de uno en uno delante de la cámara, no da problemas en ningún momento.



5.5 Reacción de los marcadores ante la situación de pasar página.

Los marcadores funcionan bien ante la situación de pasar una página. Se ha realizado la prueba distribuyendo los marcadores a lo largo de varias páginas en un libro (11 páginas, 10 marcadores).

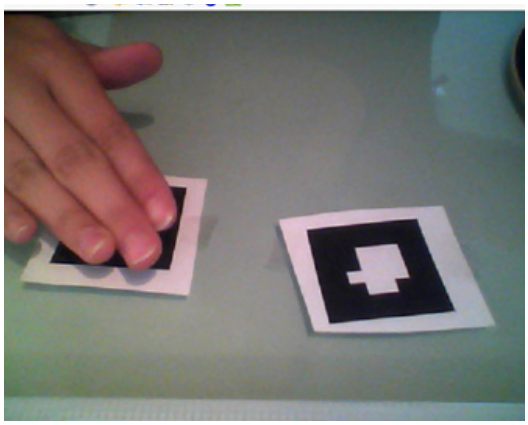
Los marcadores generan los cubos según van pasando las páginas, aunque dependiendo de la posición del marcador hay problemas con el color del cubo (al haber formas parecidas una posición muy en perspectiva puede generar errores respecto al modelo correspondiente). Esto no suele ocurrir una vez la página se ha pasado completamente y los marcadores están quietos, aunque alguna vez sí que ha continuado el problema.

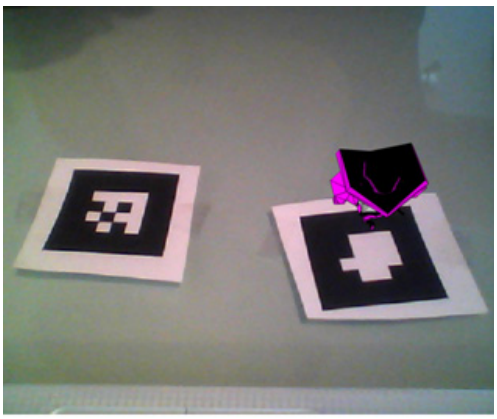


5.6 Reacciones entre marcadores.

El modelo con el que se trabajó en processing se vinculó a un marcador que reaccionaba ante la aparición de otros marcadores: se animaba y rotaba. Modificando un documento de FLARManager, se ha obtenido una forma de interacción parecida, en la que uno de los marcadores contiene al modelo, y el segundo lo hace visible. Además, girando el segundo marcador rota también el modelo del primero.

Con este ejemplo se ha comprobado que es posible la interacción entre marcadores.

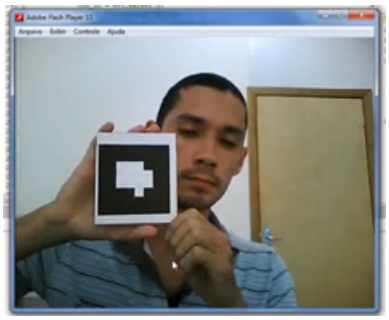




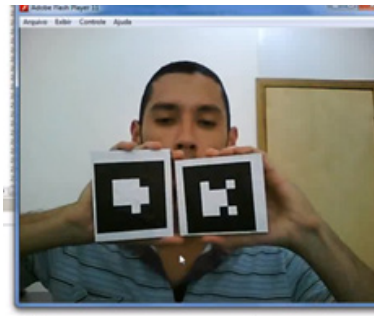
5.7 Reacciones entre marcadores (II).

Se utiliza un marcador para determinar la posición y escala del modelo y los otros dos para determinar la visibilidad del mismo. Sólo cuando ambos marcadores son visibles se ve el modelo. Si alguno de los dos se baja o es tapado parcialmente, la imagen 3D desaparece. De esta forma, se puede ver que la herramienta es capaz de reaccionar ante la presencia de un nuevo marcador cuando hay dos en escena y de producir un efecto en ellos.

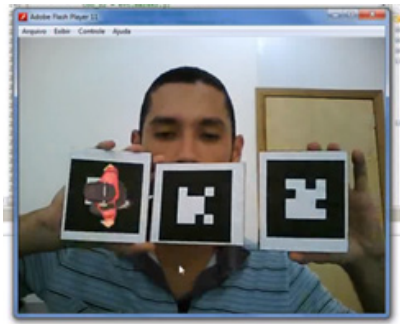
Este ejemplo se encontró por la web, no se ha probado en el ordenador de trabajo.



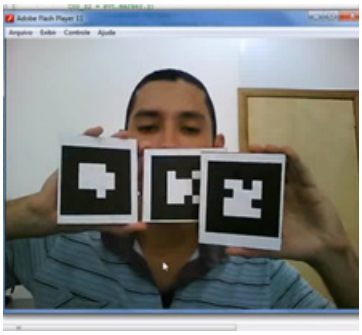
Primer marcador: Contenedor del modelo.



Segundo marcador: Necesario para ver el modelo.



Tercer marcador: Necesario para ver el modelo.



Si alguno de los marcadores no se ve correctamente el modelo no se muestra.

5.8 Varios modelos en un sólo marcador.

Se ha encontrado un ejemplo en el que a través de un mismo marcador se visualizan varios modelos complejos simultáneamente. Esto puede resultar interesante en alguna aplicación. Este ejemplo se encontró por la web, no se ha probado en el ordenador de trabajo.

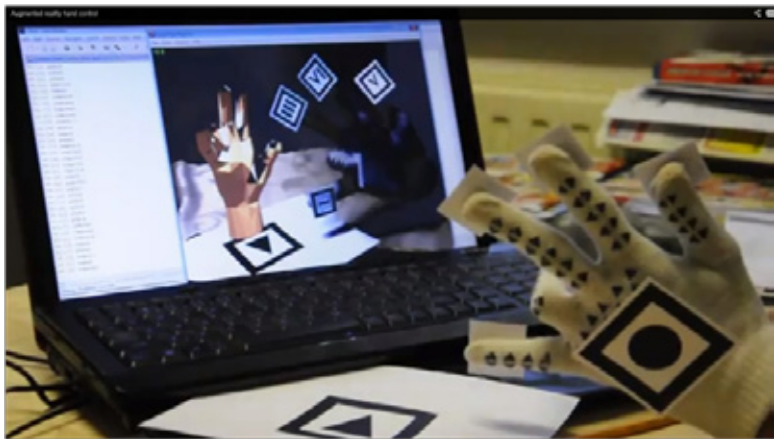
5.9 Otras capacidades complejas: Letras animadas en marcador.

En este ejmplo se puede ver una aplicación que juega con letras en el que un marcador está asociado a un conjunto de letras dispuestas en 3 dimensiones. Al pasar la mano se reordenan de una determinada manera. Este ejemplo se encontró por la web, no se ha probado en el ordenador de trabajo.

5.10 Otras capacidades complejas: Efecto galería, control de animación.

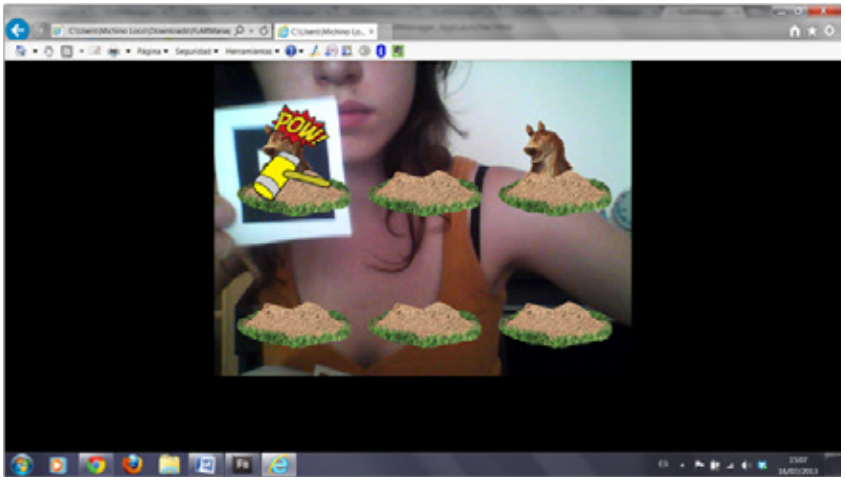
Esta apliación se basa en la rotación de varias imágenes dispuestas en un espacio tridimensional: Moviendo ligeramente el marcador, las imágenes rotan y se pueden visualizar como una galería. Este ejemplo se encontró por la web, no se ha probado en el ordenador de trabajo.

A través de la visibilidad o no de los marcadores el modelo actúa de una forma u otra. Cada marcador se relaciona con un dedo, cuando éste es visible, el dedo permanece erguido, si el marcador desaparece, el dedo se dobla, simulando que imita el movimiento de la mano real. Este ejemplo se encontró por la web, no se ha probado en el ordenador de trabajo.

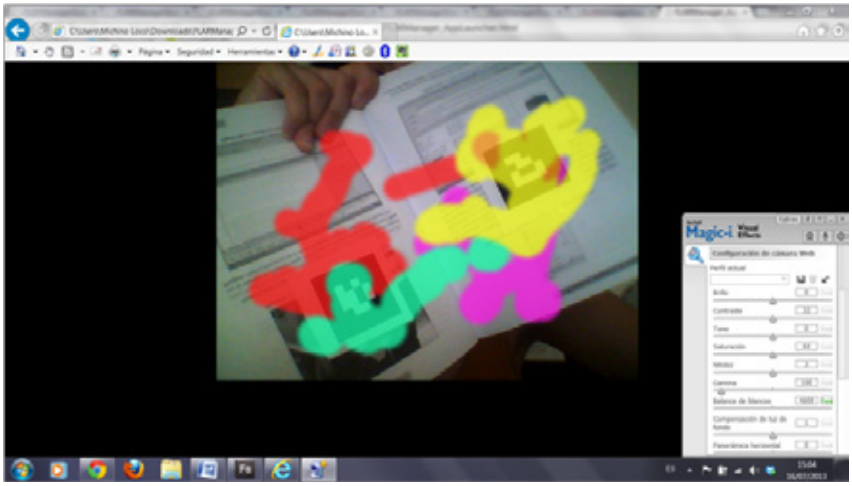


5.11 Otras capacidades complejas: Aplicación mini-juego, colorear.

Calcula la posición del marcador, crea una imagen en él y cuando ésta coincide con una imagen del animal la animación cambia. Permite la interactividad. Incluye sonido. Para un solo marcador.

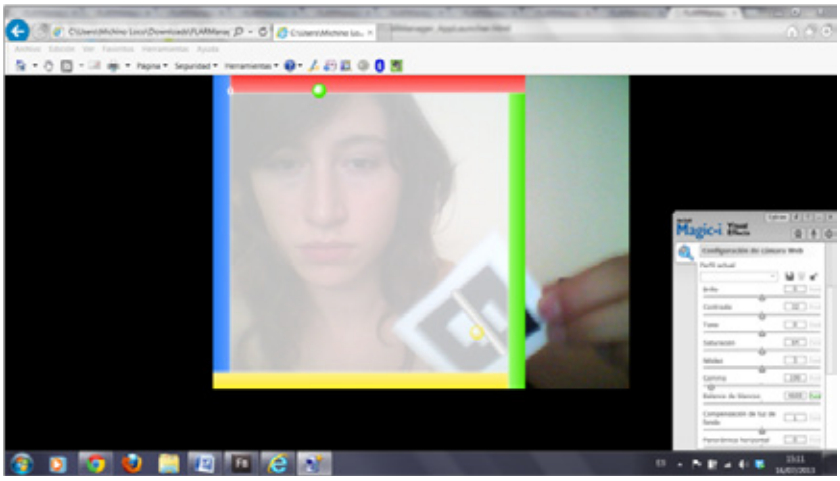


Generar líneas de colores siguiendo la trayectoria del marcador. Es capaz de realizarlo también con varios marcadores y con varios marcadores visibles simultáneamente. Calcula la posición del marcador y crea un punto de color donde se encuentra.

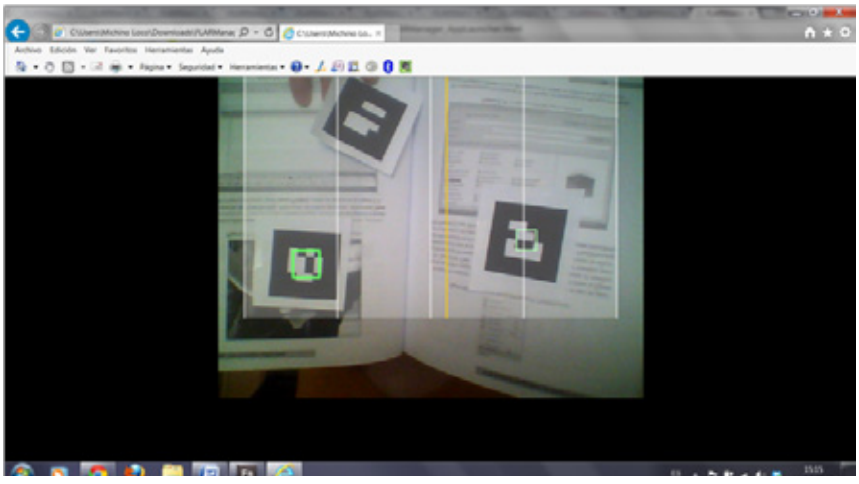


5.12 Otras capacidades complejas: Aplicación mini-juego, escala de sonidos.

Juego bolas: calcula la posición del marcador, interpreta cuándo coincide con una de las animaciones y crea las trayectorias en función de la posición del marcador. Genera información de acierto o error dependiendo de si el color de la bola coincide con el marco de del mismo color o no, a través de sonido.



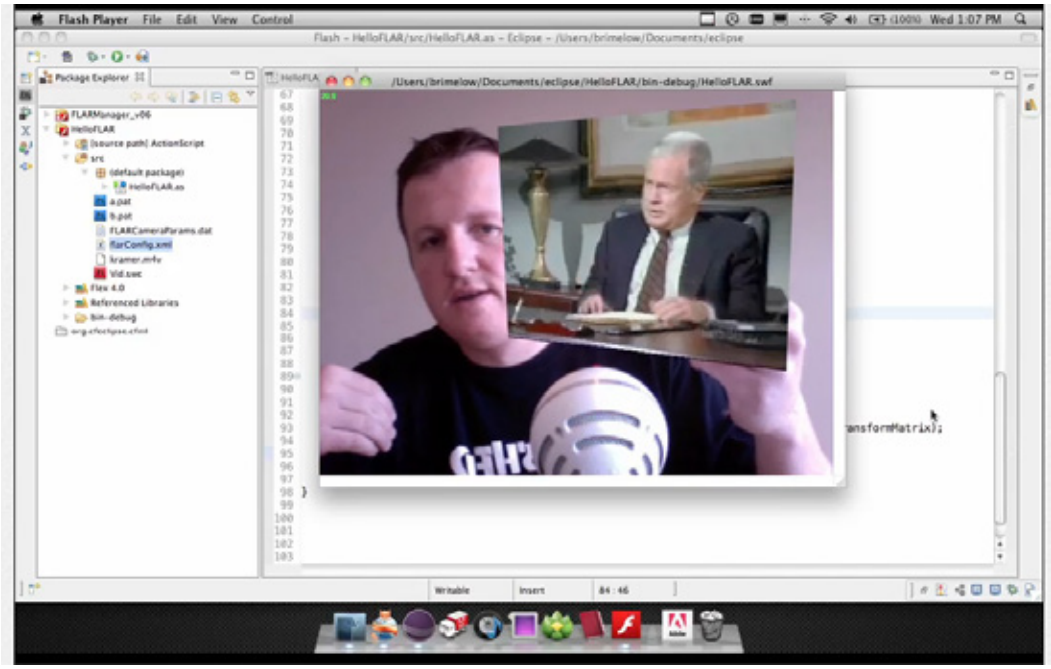
Aplicación sonidos: se asocia a cada marcador un sonido distinto, cuando el marcador coincide con la barra vertical que se mueve de izquierda a derecha de la pantalla se reproduce el sonido. Pensada para el uso de varios marcadores simultáneamente.



5.13 Otras capacidades complejas: Marcador asociado avídeo.

Aplicación en la que se incluye un vídeo asociado a un marcador. Se han encontrado dos formas en las que se hacen: asociando el vídeo directamente en FlarManager o realizando un modelo de un rectángulo con textura de vídeo.

Este ejemplo se encontró por la web, no se ha probado en el ordenador de trabajo.



APÉNDICE 6

Desarrollo de la aplicación.

Índice del Apéndice 6:

6.0 Introducción.

En este apéndice se explica el desarrollo de la aplicación elegida. Se muestra en primer lugar el Storyboard de la aplicación, donde se define cómo se distribuyen las páginas del libro y qué contiene cada una de ellas.

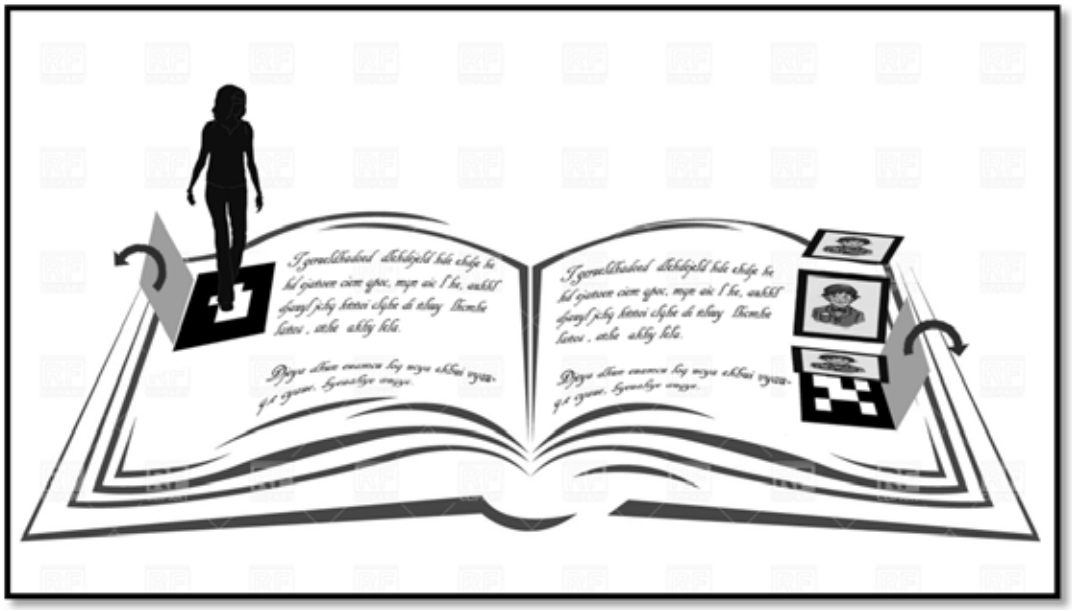
Se explica el modo en el que se ha trabajado la Realidad Aumentada y se copia el código generado para realizar las distintas aplicaciones necesarias para implementar la Realidad Aumentada que aparece en el libro.

A continuación se muestran los modelos creados para la aplicación y el diseño gráfico de cada una de las páginas del libro. Se enseña también el modo de uso de la aplicación a tavés de imágenes tomadas durante el uso de la misma.

Se termina este apéndice citando los programas y asignaturas que han sido importantes para el desarrollo del proyecto, y explicando para qué área se han utilizado; y con las conclusiones finales sobre este Trabajo de Fin de Grado.

6.1 Storyboard.

La historia del libro cuenta la resolución de un misterio. A lo largo de la historia, el niño irá encontrando pistas y deberá realizar acciones sencillas para ver los modelos 3D, como levantar solapas del libro, mover marcadores, etc. Las escenas son las siguientes.



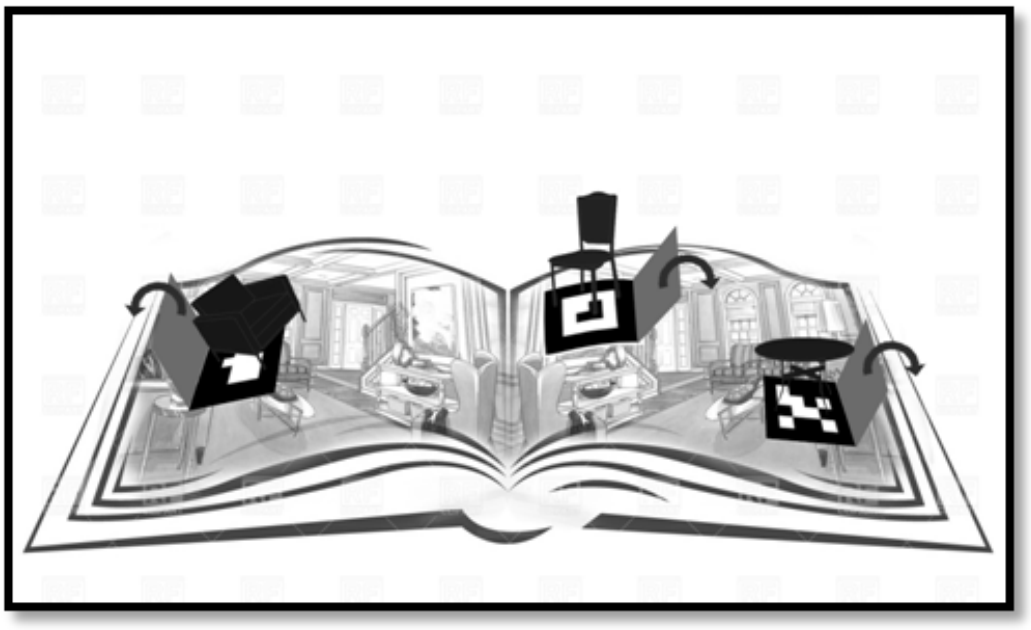
Introducción de la historia. En esta escena hay dos solapas que ocultan dos marcadores. Al abrir la primera se muestra el modelo 3D de la protagonista de la historia sobre el marcador. Cuando se abre la segunda solapa se muestran tres planos 2D con textura. El conjunto de planos 2D rota verticalmente según el movimiento horizontal del ratón, creándose el efecto de una galería de imágenes. No hay interacción entre los marcadores de la escena.

6.1 Storyboard.

En la primera página se presenta a la protagonista y en la segunda comienza el caso a resolver: la protagonista, Sara, va a casa de un vecino y éste le cuenta que le ha desaparecido algo de dinero que guardaba en el salón. Ella le pregunta quiénes fueron los últimos que estuvieron en la casa y su vecino le habla de tres personas, que serán los sospechosos (fotos planos 2D).

Diseño gráfico de la escena:

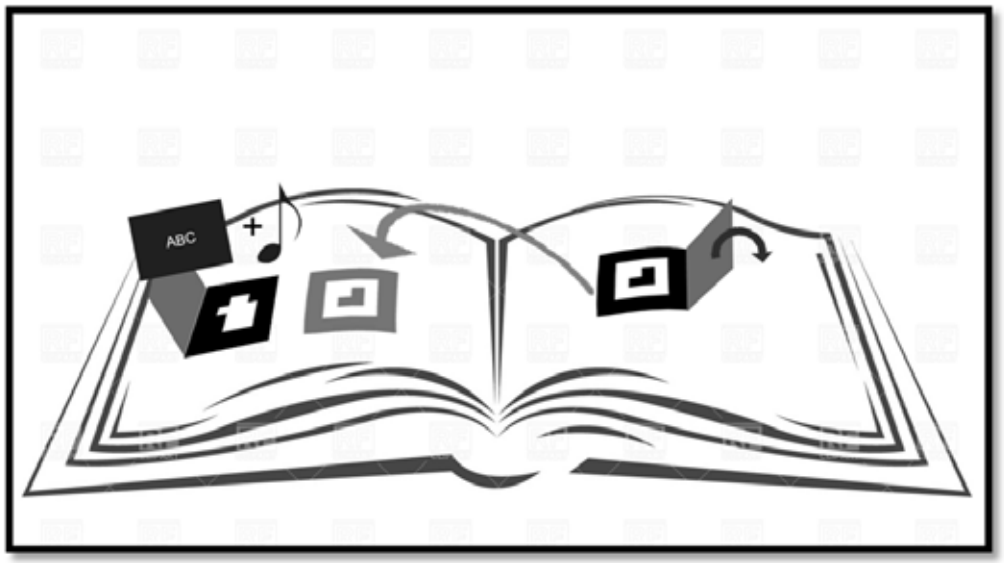
6.1 Storyboard.



Búsqueda de pistas. En esta segunda escena hay una ilustración que ocupa ambas páginas. Hay también tres solapas que se integran en la imagen, completando la ilustración. Al levantar las solapas se muestran modelos 3D de muebles. No hay interacción entre los marcadores de la escena.

Sara investiga el salón en busca de pistas, y encuentra tres (modelos 3D): 1. La estantería donde guardaba el dinero su vecino está muy desordenada, quien robó no tuvo mucho cuidado de dejarlo bien. 2. Hay una pulsera de colores olvidada junto a una silla. 3. Al lado del sofá hay una huella de barro de una zapatilla.

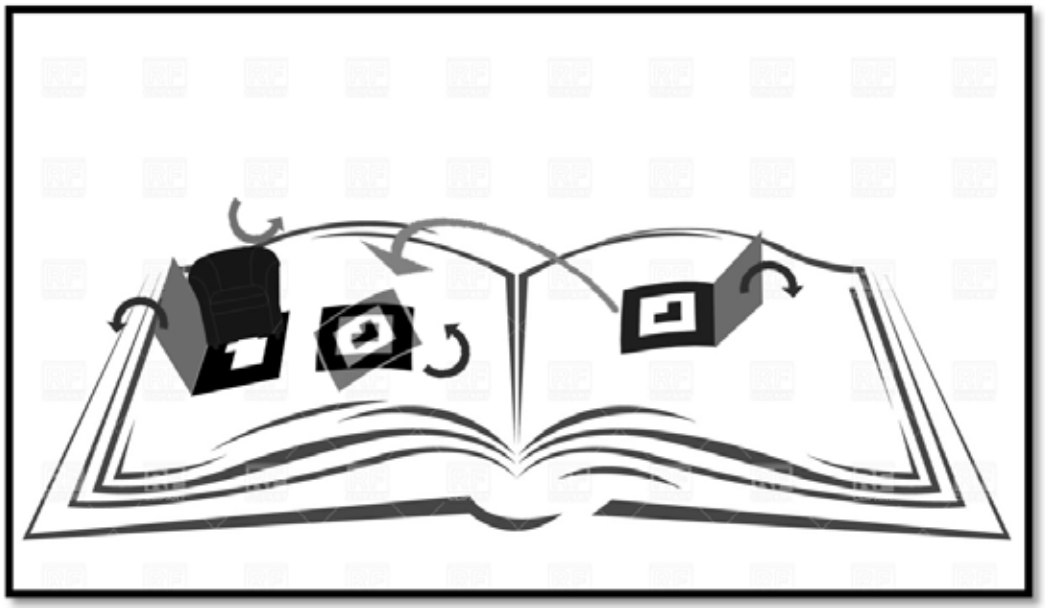
6.1 Storyboard.



Sospechoso 1: Asistente. En esta escena hay una solapa bajo la que se encuentra un marcador y un sobre que guarda otro. Los dos marcadores interactúan entre sí. Cuando ambos son visibles (una vez que se ha levantado la solapa del primero y se ha sacado el segundo del sobre) se muestra un plano con textura y se reproduce un clip de sonido.

El asistente llega esa misma tarde a la casa para realizar su trabajo y Sara aprovecha para investigar sus cosas. Encuentra su móvil y una vez da con el código para desbloquearlo (segundo marcador, sonido), encuentra un mensaje que da a entender que él no ha podido ser.

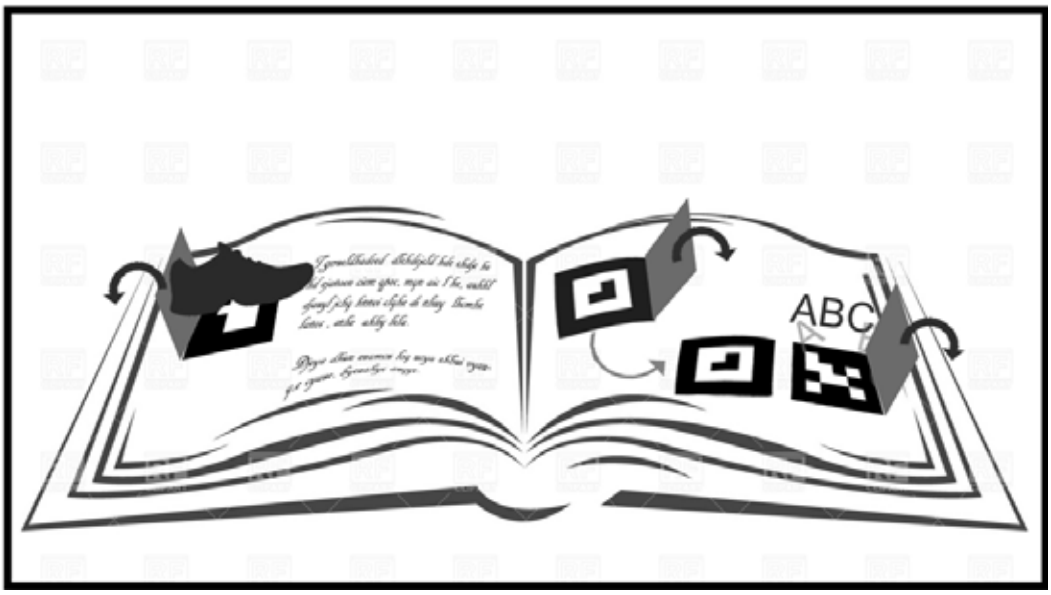
6.1 Storyboard.



Sospechoso 2: Vecina. En esta escena hay dos solapas. Al levantar la primera aparece un modelo 3D de un estante con zapatos. Al levantar la segunda aparece un segundo marcador. Este marcador controla la rotación del modelo y permite ver lo que hay en su otra cara. Por lo tanto, los dos marcadores de esta escena interactúan entre sí.

Sara visita a la vecina en su casa e investiga en su habitación. Ve unas pulseras parecidas a la que encontró, lo que le hace sospechar, así que busca las zapatillas. La mujer tiene un armario que puede rotarse, con zapatos, pero cuando mira a ambos lados no ve nada, así que se queda con la duda.

6.1 Storyboard.



Sospechoso 3: Vecina adolescente. En esta escena hay dos solapas. Al abrir la primera aparece un modelo 3D de las zapatillas. La segunda solapa se abre dos veces: en la primera se muestra un marcador que contiene el la animación de un modelo 3D (letras que se ordenan) y en la segunda otro marcador que activa este modelo. Por lo tanto, en esta escena los dos marcadores bajo la segunda solapa interactúan entre sí, mientras que el primero es independiente.

Sara consigue entrar también en el dormitorio de la tercera sospechosa, y encuentra las zapatillas cubiertas de barro (modelo 3D). Mirando un poco más, encuentra un diario con una carta para una amiga, cuyo final está escrito en clave (marcador 2, animación 3D). Cuando encuentra el alfabeto para descifrarlo (marcador 3), puede terminar de leerlo, donde queda patente que ella ha sido la causante del robo.

6.2 Realidad Aumentada: Modo de trabajo.

Se ha trabajado con un total de 24 aplicaciones, siete de ellas obtenidas a partir de las iniciales y de información encontrada en la web y el resto dadas por el propio programa. A continuación se describirá brevemente lo que había en cada una de ellas y los cambios realizados:

Originales.

1. Recuadro verde sobre el marcador, indicando su identificación.
2. Un único modelo simple (cubo).
3. Modelo 3D complejo y animado (Away 3D).
4. Modelo 3D complejo y animado (Papervision3D).
5. Brújula (una línea que se orienta siempre hacia el norte).
6. Imagen en 2D.
7. Varios modelos simples (cubos) con Alterna3D.
8. Varios modelos simples (cubos) con Away 3D.
9. Varios modelos simples pero de más polígonos con Away 3D Lite.
10. Varios modelos simples (cubos) con PV3D.
11. Varios modelos simples (cubos) con Sandy3D.
12. Vista de varios marcadores moviéndose en pantalla.
13. Cambio del formato de pantalla.
14. Aplicación para dibujar con el marcador sobre la pantalla. Posibilidad de utilizar varios marcadores simultáneamente.

6.2 Realidad Aumentada: Modo de trabajo.

15. Aplicación para un juego, en el que se muestran animaciones de animales apareciendo a través de un agujero, al que se debe pegar en este momento con el marcador (sobre el que aparece un mazo). Incluye sonido. Es interactivo.

16. Aplicación para un juego, en el que el marcador es una barra con la que se golpea una bola para dirigirla a su color. Incluye sonido. Es interactivo.

17. Aplicación en el que cada marcador se asocia a un sonido. Una barra pasa de izquierda a derecha de la pantalla, y al pasar sobre los marcadores se genera el sonido, permitiendo crear melodías.

Modificados.

18. Con ayuda del libro de estudio, se cambió el modelo inicial (Mario Bross animado) por un modelo animado generado en Blender (Annie). Además, se incluyó un marcador más, que controlaba la rotación del personaje. Se convirtió, por lo tanto, en una aplicación interactiva.

19. Se cambió también en este caso el modelo inicial (Scout animado) por un modelo propio realizado en 3dsMax (baúl estático). Se mantuvo un único marcador.

6.2 Realidad Aumentada: Modo de trabajo.

20. Se cambió el modelo inicial (Scout animado) por una animación propia creada en 3dsMax (cilindro que se dobla). Se mantuvo un único marcador.

21. Se consiguió cargar varios modelos asociados a múltiples marcadores. En concreto, se cargaron cuatro modelos, uno de ellos de creación propia. Esto se consiguió a través de un código de la web, integrado en una de las aplicaciones originales, dado que el código por sí sólo no funcionaba.

22. Se consiguió otra aplicación interactiva con dos marcadores, uno asociado al modelo, y otro controlando su visibilidad. Cuando el segundo se muestra, aparece el modelo en el primero. Si se quita, el modelo desaparece. Esto se obtuvo nuevamente a través de información encontrada en la web y a partir de uno de los ejemplos originales.

23. Se obtuvo una aplicación en la que se asociaba música a un marcador. Esto se obtuvo gracias a información encontrada en la web y a partir de uno de los ejemplos originales.

24. Por último, se consiguión una aplicación en la que se controla la rotación de un modelo a través del movimiento de ratón. Cuando el ratón se mueve horizontalmente, el modelo gira verticalmente. Esto se obtuvo a partir de uno de los ejemplos originales y con conocimientos obtenidos durante el estudio del programa Processing.

6.2 Realidad Aumentada: Modo de trabajo.

Finalmente, combinando algunas de las aplicaciones que se han generado e introduciendo en todas ellas contenidos propios, se ha obtenido una aplicación para este Trabajo de Fin de grado dividida en cinco escenas, que permiten lo siguiente:

-Mostrar tanto modelos 3D como animaciones asociados a marcadores.

-Mostrar varios modelos simultáneamente, asociados a distintos marcadores.

-Asociar sonido al marcador.

-Interacción entre marcadores: Un marcador contiene un modelo 3D que se hace visible cuando se muestra un segundo marcador.

-Interacción entre marcadores: Un marcador contiene un modelo 3D cuya rotación depende de un segundo marcador.

-Efecto de galería: El modelo rota mostrando distintos elementos del mismo cuando el ratón se mueve de lado a lado de la pantalla, obteniendo el efecto de una galería de imágenes.

6.2 Realidad Aumentada: Modo de trabajo.

A continuación se explica cómo funciona el código general de la aplicación.

- Se empieza importando elementos propios de FLARManager y de Flash necesarios para la aplicación, como marcadores, cámaras, motores de renderización, etc.

- A continuación, se crea la clase pública de la aplicación, donde se declaran las variables a utilizar (marcadores, contenedores, luces, cámaras, etc.) y se incluyen las funciones para la aplicación.

- Se llama a la página de configuración de FLARManager para el uso de cámara y del seguimiento de marcadores, y se añaden los eventos que se van a utilizar al controlador de eventos.

- Después se define la función para iniciar el programa, en la que se crean los elementos necesarios para iniciar la escena, como el motor de renderizado, la luz, cámara, tipo de escena, etc. Además, se cargan los modelos 3D, determinando su escala y posición, y los contenedores para los modelos.

6.2 Realidad Aumentada: Modo de trabajo.

- Dependiendo de la acción que se realice con los marcadores los modelos actúan de distintas formas, a través de distintas funciones:

- Cuando se introduce un marcador a la escena, se activa el marcador y el modelo es visible (función onMarkerAdded).

- Cuando se actualiza el marcador, se realizan las mismas acciones (función onMarkerUpdated).

- Cuando el marcador se retira de la escena, el marcador se desactiva y el modelo ya no es visible (función onMarkerRemoved).

- Por último, mediante otra función (onEnterFrame), se controlan las transformaciones geométricas de los marcadores.

En cualquiera de estas acciones, el programa distingue el marcador por un número asociado a éste, de modo que se puede definir en cada una de estas funciones, qué marcador contiene y hace visible de cada modelo.

- Por último, se renderiza la escena.

6.2 Realidad Aumentada: Modo de trabajo.

En la aplicación en la que se ha introducido sonido, se declaran las variables asociadas al sonido al comienzo de la clase pública, junto al resto de variables (sonido, url del archivo, canal de sonido, etc) y se activa el sonido en las funciones onMarkerAdded y onMarkerUpdated, para que se reproduzca cuando aparezca el marcador que se haya elegido.

En las aplicaciones donde se controla el movimiento de un modelo (tanto la rotación a través de un marcador como a través del movimiento del ratón), se describe el modo y condiciones en las que se mueven en la función onEnterFrame.

Por último, en las aplicaciones en las que son necesarios varios marcadores para ver un modelo, se determina qué marcador contiene el modelo y qué marcador lo hace visible en las funciones asociadas al comportamiento de los marcadores (onMarkerAdded, onMarkerUpdated y onMarkerRemoved).

6.3 Realidad Aumentada: Código.

Primera escena:

```
package examples {
        import com.transmote.flar.FLARManager;
        import com.transmote.flar.camera.FLARCamera_PV3D;
        import com.transmote.flar.marker.FLARMarker;
        import com.transmote.flar.marker.FLARMarkerEvent;
        import com.transmote.flar.tracker.FLARToolkitManager;
        import com.transmote.flar.utils.geom.PVGeomUtils;
        import flash.display.Sprite;
        import flash.events.Event;
        import flash.geom.Rectangle;
        import org.papervision3d.cameras.Camera3D;
        import org.papervision3d.lights.PointLight3D;
        import org.papervision3d.objects.DisplayObject3D;
        import org.papervision3d.objects.parsers.DAE;
        import org.papervision3d.render.LazyRenderEngine;
        import org.papervision3d.scenes.Scene3D;
        import org.papervision3d.view.Viewport3D;

                public class FLARManagerTutorial_Collada_PV3D_pag1 extends Sprite {
                private var flarManager:FLARManager;
                private var scene3D:Scene3D;
                private var camera3D:Camera3D;
                private var viewport3D:Viewport3D;
                private var renderEngine:LazyRenderEngine;
                private var pointLight3D:PointLight3D;
                private var activeMarker:FLARMarker;
                private var activeMarker1:FLARMarker;

                private var modelContainer:DisplayObject3D;
                private var modelContainer1:DisplayObject3D;

                        public function FLARManagerTutorial_Collada_PV3D_pag1 () {
                        this.addEventListener(Event.ADDED_TO_STAGE, this.onAdded);
                }
                                private function onAdded (evt:Event) :void {
                        this.removeEventListener(Event.ADDED_TO_STAGE, this.onAdded);
                        this.flarManager = new FLARManager("../resources/flar/flarConfig.xml", new FLARToolkitManager(), this.stage);
                        this.flarManager.addEventListener(FLARMarkerEvent.MARKER_ADDED, this.onMarkerAdded);
                        this.flarManager.addEventListener(FLARMarkerEvent.MARKER_UPDATED, this.onMarkerUpdated);
                        this.flarManager.addEventListener(FLARMarkerEvent.MARKER_REMOVED, this.onMarkerRemoved);
```

6.3 Realidad Aumentada: Código.

```
this.flarManager.addEventListener(Event.INIT, this.onFlarManagerInited); }

private function onFlarManagerInited (evt:Event) :void {
this.flarManager.removeEventListener(Event.INIT, this.onFlarManagerInited);

this.scene3D = new Scene3D();
this.viewport3D = new Viewport3D(this.stage.stageWidth, this.stage.stageHeight);
this.addChild(this.viewport3D);
this.camera3D = new FLARCamera_PV3D(this.flarManager, new Rectangle(0, 0, this.stage.stageWidth, this.stage.stageHeight));
this.renderEngine = new LazyRenderEngine(this.scene3D, this.camera3D, this.viewport3D);
this.pointLight3D = new PointLight3D();
this.pointLight3D.x = 1000;
this.pointLight3D.y = 1000;
this.pointLight3D.z = -1000;

var model1:DAE = new DAE(true, "model1", true);
model1.load("../resources/assets/protaanimada.dae");
model1.rotationX = 90;
model1.rotationZ = -90;
model1.scale = 0.5;

var model2:DAE = new DAE(true, "model2", true);
model2.load("../resources/assets/sospechososrot.DAE");
model2.rotationX = 0;
model2.rotationZ = -90;
model2.scale = 1.6;

this.modelContainer = new DisplayObject3D();
this.modelContainer.addChild(model1);
this.modelContainer.visible = false;
this.scene3D.addChild(this.modelContainer);

this.modelContainer1 = new DisplayObject3D();
this.modelContainer1.addChild(model2);
this.modelContainer1.visible = false;
this.scene3D.addChild(this.modelContainer1);

this.addEventListener(Event.ENTER_FRAME, this.onEnterFrame);
}
```

6.3 Realidad Aumentada: Código.

```
private function onMarkerAdded (evt:FLARMarkerEvent) :void {
            if(evt.marker.patternId == 0)
            {this.modelContainer.visible = true;
             this.activeMarker = evt.marker; }

              if(evt.marker.patternId == 1)
              {this.modelContainer1.visible = true;
               this.activeMarker1 = evt.marker; }
}
private function onMarkerUpdated (evt:FLARMarkerEvent) :void {
            if(evt.marker.patternId == 0)
            {this.modelContainer.visible = true;
             this.activeMarker = evt.marker; }

              if(evt.marker.patternId == 1)
              {this.modelContainer1.visible = true;
               this.activeMarker1 = evt.marker; }
}
private function onMarkerRemoved (evt:FLARMarkerEvent) :void {
            if(evt.marker.patternId == 0)
            {this.modelContainer.visible = false;
             this.activeMarker = null; }

              if(evt.marker.patternId == 1)
              {this.modelContainer1.visible = false;
               this.activeMarker1 = null; }
}
}
private function onEnterFrame (evt:Event) :void {
              if (this.activeMarker) {
               this.modelContainer.transform = PVGeomUtils.convertMatrixToPVMatrix(this.activeMarker.transformMatrix);}

                if (this.activeMarker1) {
                this.modelContainer1.transform = PVGeomUtils.convertMatrixToPVMatrix(this.activeMarker1.transformMatrix);
                this.modelContainer1.rotationX = mouseX;}

                this.renderEngine.render(); }
            }

}
```

6.3 Realidad Aumentada: Código.

Segunda escena:

```
package examples {
        import com.transmote.flar.FLARManager;
        import com.transmote.flar.camera.FLARCamera_PV3D;
        import com.transmote.flar.marker.FLARMarker;
        import com.transmote.flar.marker.FLARMarkerEvent;
        import com.transmote.flar.tracker.FLARToolkitManager;
        import com.transmote.flar.utils.geom.PVGeomUtils;
        import flash.display.Sprite;
        import flash.events.Event;
        import flash.geom.Rectangle;
        import org.papervision3d.cameras.Camera3D;
        import org.papervision3d.lights.PointLight3D;
        import org.papervision3d.objects.DisplayObject3D;
        import org.papervision3d.objects.parsers.DAE;
        import org.papervision3d.render.LazyRenderEngine;
        import org.papervision3d.scenes.Scene3D;
        import org.papervision3d.view.Viewport3D;

                public class FLARManagerTutorial_Collada_PV3D_pag2 extends Sprite {
                private var flarManager:FLARManager;
                private var scene3D:Scene3D;
                private var camera3D:Camera3D;
                private var viewport3D:Viewport3D;
                private var renderEngine:LazyRenderEngine;
                private var pointLight3D:PointLight3D;
                private var activeMarker:FLARMarker;
                private var activeMarker1:FLARMarker;
                private var activeMarker2:FLARMarker;

                private var modelContainer:DisplayObject3D;
                private var modelContainer1:DisplayObject3D;
                private var modelContainer2:DisplayObject3D;

                private var markerId:int;

                        public function FLARManagerTutorial_Collada_PV3D_pag2 () {
                         this.addEventListener(Event.ADDED_TO_STAGE, this.onAdded);
                }
                                private function onAdded (evt:Event) :void {
                         this.removeEventListener(Event.ADDED_TO_STAGE, this.onAdded);
```

6.3 Realidad Aumentada: Código.

```
this.flarManager = new FLARManager("../resources/flar/flarConfig.xml", new FLARToolkitManager(), this.stage);
this.flarManager.addEventListener(FLARMarkerEvent.MARKER_ADDED, this.onMarkerAdded);
this.flarManager.addEventListener(FLARMarkerEvent.MARKER_UPDATED, this.onMarkerUpdated);
this.flarManager.addEventListener(FLARMarkerEvent.MARKER_REMOVED, this.onMarkerRemoved);
this.flarManager.addEventListener(Event.INIT, this.onFlarManagerInited);}

private function onFlarManagerInited (evt:Event) :void {
this.flarManager.removeEventListener(Event.INIT, this.onFlarManagerInited);
this.scene3D = new Scene3D();
this.viewport3D = new Viewport3D(this.stage.stageWidth, this.stage.stageHeight);
this.addChild(this.viewport3D);
this.camera3D = new FLARCamera_PV3D(this.flarManager, new Rectangle(0, 0, this.stage.stageWidth, this.stage.stageHeight));
this.renderEngine = new LazyRenderEngine(this.scene3D, this.camera3D, this.viewport3D);
this.pointLight3D = new PointLight3D();
this.pointLight3D.x = 1000;
this.pointLight3D.y = 1000;
this.pointLight3D.z = -1000;
var model1:DAE = new DAE(true, "model1", true);
model1.load("../resources/assets/sofahuella.dae");
model1.rotationX = 90;
model1.rotationZ = 90;
model1.scale = 1;

var model2:DAE = new DAE(true, "model2", true);
model2.load("../resources/assets/silla.dae");
model2.rotationX = 90;
model2.rotationZ = 90;
model2.scale = 1.3;

var model3:DAE = new DAE(true, "model3", true);
model3.load("../resources/assets/estanteria2.dae");
model3.rotationX = 90;
model3.rotationZ = 90;
model3.scale = 1.5;

this.modelContainer = new DisplayObject3D();
this.modelContainer.addChild(model1);
this.modelContainer.visible = false;
this.scene3D.addChild(this.modelContainer);
```

6.3 Realidad Aumentada: Código.

```
            this.modelContainer1 = new DisplayObject3D();
            this.modelContainer1.addChild(model2);
            this.modelContainer1.visible = false;
            this.scene3D.addChild(this.modelContainer1);

            this.modelContainer2 = new DisplayObject3D();
            this.modelContainer2.addChild(model3);
            this.modelContainer2.visible = false;
            this.scene3D.addChild(this.modelContainer2);

             this.addEventListener(Event.ENTER_FRAME, this.onEnterFrame);
}            private function onMarkerAdded (evt:FLARMarkerEvent) :void {
            trace("["+evt.marker.patternId+"] added");
            if(evt.marker.patternId == 2)
            {this.activeMarker = evt.marker;
             modelContainer.visible=true;}

            if(evt.marker.patternId == 3)
            {this.activeMarker1 = evt.marker;
              modelContainer1.visible=true;   }

            if(evt.marker.patternId == 4)
            { this.activeMarker2 = evt.marker;
             modelContainer2.visible=true;}
}
           private function onMarkerUpdated (evt:FLARMarkerEvent) :void {
            if(evt.marker.patternId == 2)
            {this.activeMarker = evt.marker;
              modelContainer.visible=true;}

            if(evt.marker.patternId == 3)
            { this.activeMarker1 = evt.marker;
                modelContainer1.visible=true;  }

            if(evt.marker.patternId == 4)
            {this.activeMarker2 = evt.marker;
             modelContainer2.visible=true    }

            }
```

6.3 Realidad Aumentada: Código.

```
private function onMarkerRemoved (evt:FLARMarkerEvent) :void {
            trace("["+evt.marker.patternId+"] removed");
            if(evt.marker.patternId == 2)
            {this.modelContainer.visible = false;
             this.activeMarker = null; }

            if(evt.marker.patternId == 3)
            {this.modelContainer1.visible = false;
             this.activeMarker1 = null; }

            if(evt.marker.patternId == 4)
            {this.modelContainer2.visible = false;
             this.activeMarker2 = null;}
}
        private function onEnterFrame (evt:Event) :void {
            if (this.activeMarker) {
            this.modelContainer.transform = PVGeomUtils.convertMatrixToPVMatrix(this.activeMarker.transformMatrix); }

            if (this.activeMarker1) {
            this.modelContainer1.transform = PVGeomUtils.convertMatrixToPVMatrix(this.activeMarker1.transformMatrix); }

            if (this.activeMarker2) {
            this.modelContainer2.transform = PVGeomUtils.convertMatrixToPVMatrix(this.activeMarker2.transformMatrix); }

            this.renderEngine.render();

}
```

6.3 Realidad Aumentada: Código.

Tercera escena:

```
package examples {
        import com.transmote.flar.FLARManager;
        import com.transmote.flar.camera.FLARCamera_PV3D;
        import com.transmote.flar.marker.FLARMarker;
        import com.transmote.flar.marker.FLARMarkerEvent;
        import com.transmote.flar.tracker.FLARToolkitManager;
        import com.transmote.flar.utils.geom.PVGeomUtils;
        import flash.display.Sprite;
        import flash.events.Event;
        import flash.geom.Rectangle;
        import org.papervision3d.cameras.Camera3D;
        import org.papervision3d.lights.PointLight3D;
        import org.papervision3d.objects.DisplayObject3D;
        import org.papervision3d.objects.parsers.DAE;
        import org.papervision3d.render.LazyRenderEngine;
        import org.papervision3d.scenes.Scene3D;
        import org.papervision3d.view.Viewport3D;

        import flash.media.Sound;
        import flash.media.SoundChannel;
        import flash.net.URLRequest;
        import flash.media.SoundLoaderContext;

                public class FLARManagerTutorial_Collada_PV3D_pag3 extends Sprite {
                private var flarManager:FLARManager;
                private var scene3D:Scene3D;
                private var camera3D:Camera3D;
                private var viewport3D:Viewport3D;
                private var renderEngine:LazyRenderEngine;
                private var pointLight3D:PointLight3D;
                private var activeMarker:FLARMarker;
                private var modelContainer:DisplayObject3D;

                private var req:URLRequest;
                private var context:SoundLoaderContext = new SoundLoaderContext(8000);
                private var s:Sound;
                private var channel:SoundChannel = new SoundChannel();

                public function FLARManagerTutorial_Collada_PV3D_pag3 () {
                        this.addEventListener(Event.ADDED_TO_STAGE, this.onAdded);
                }
```

6.3 Realidad Aumentada: Código.


```
private function onAdded (evt:Event) :void {
this.removeEventListener(Event.ADDED_TO_STAGE, this.onAdded);
this.flarManager = new FLARManager("../resources/flar/flarConfig.xml", new FLARToolkitManager(), this.stage);
this.flarManager.addEventListener(FLARMarkerEvent.MARKER_ADDED, this.onMarkerAdded);
this.flarManager.addEventListener(FLARMarkerEvent.MARKER_UPDATED, this.onMarkerUpdated);
this.flarManager.addEventListener(FLARMarkerEvent.MARKER_REMOVED, this.onMarkerRemoved);
this.flarManager.addEventListener(Event.INIT, this.onFlarManagerInited);}

 private function onFlarManagerInited (evt:Event) :void {
this.flarManager.removeEventListener(Event.INIT, this.onFlarManagerInited);
this.scene3D = new Scene3D();
this.viewport3D = new Viewport3D(this.stage.stageWidth, this.stage.stageHeight);
this.addChild(this.viewport3D);
this.camera3D = new FLARCamera_PV3D(this.flarManager, new Rectangle(0, 0, this.stage.stageWidth, this.stage.stageHeight));
this.renderEngine = new LazyRenderEngine(this.scene3D, this.camera3D, this.viewport3D);
this.pointLight3D = new PointLight3D();
this.pointLight3D.x = 1000;
this.pointLight3D.y = 1000;
this.pointLight3D.z = -1000;

var model:DAE = new DAE(true, "model", true);
model.load("../resources/assets/mensajemovil.DAE");
model.rotationX = 90;
model.rotationZ = 90;
model.scale = 1.5;
this.modelContainer = new DisplayObject3D();
this.modelContainer.addChild(model);
this.modelContainer.visible = false;
this.scene3D.addChild(this.modelContainer);

this.addEventListener(Event.ENTER_FRAME, this.onEnterFrame);
}        private function onMarkerAdded (evt:FLARMarkerEvent) :void {
if (evt.marker.patternId == 5)
 {this.modelContainer.visible = false;
this.activeMarker = evt.marker;}
if (evt.marker.patternId == 6)
{this.modelContainer.visible = true;
req = new URLRequest("../resources/assets/teclado.mp3");
s = new Sound (req, context);
s.play(0,0,null); }
```

6.3 Realidad Aumentada: Código.


```
        private function onMarkerUpdated (evt:FLARMarkerEvent) :void {
                trace("["+evt.marker.patternId+"] updated");
                if (evt.marker.patternId == 5)
                {this.modelContainer.visible = false;
                this.activeMarker = evt.marker;}

                if (evt.marker.patternId == 6)
                {this.modelContainer.visible = true;
                req = new URLRequest("../resources/assets/Sonido.mp3");
                s = new Sound (req, context);
                s.play(0,0,null);   }
        }
        private function onMarkerRemoved (evt:FLARMarkerEvent) :void {
                trace("["+evt.marker.patternId+"] removed");
                if (evt.marker.patternId == 5)
                {this.modelContainer.visible = false;
                this.activeMarker = null; }

                if (evt.marker.patternId == 6)
                {this.modelContainer.visible = false;
                this.activeMarker = null;
                req = new URLRequest("../resources/assets/Sonido.mp3");
                s = new Sound (req, context);
                s.play(0,0,null); }
        }
        private function onEnterFrame (evt:Event) :void {
                if (this.activeMarker) {
                        this.modelContainer.transform = PVGeomUtils.convertMatrixToPVMatrix(this.activeMarker.transformMatrix);}

                this.renderEngine.render();
           }
      }
}
```

6.3 Realidad Aumentada: Código.

Cuarta escena:

```
package examples {
        import com.transmote.flar.FLARManager;
        import com.transmote.flar.camera.FLARCamera_PV3D;
        import com.transmote.flar.marker.FLARMarker;
        import com.transmote.flar.marker.FLARMarkerEvent;
        import com.transmote.flar.tracker.FLARToolkitManager;
        import com.transmote.flar.utils.geom.PVGeomUtils;
        import flash.display.Sprite;
        import flash.events.Event;
        import flash.geom.Rectangle;
        import org.papervision3d.cameras.Camera3D;
        import org.papervision3d.lights.PointLight3D;
        import org.papervision3d.objects.DisplayObject3D;
        import org.papervision3d.objects.parsers.DAE;
        import org.papervision3d.render.LazyRenderEngine;
        import org.papervision3d.scenes.Scene3D;
        import org.papervision3d.view.Viewport3D;
        import flash.utils.getTimer;

                public class FLARManagerTutorial_Collada_PV3D_pag4 extends Sprite {
                private var flarManager:FLARManager;
                private var scene3D:Scene3D;
                private var camera3D:Camera3D;
                private var viewport3D:Viewport3D;
                private var renderEngine:LazyRenderEngine;
                private var pointLight3D:PointLight3D;
                private var activeMarker:FLARMarker;
                private var modelContainer:DisplayObject3D;

                private var rotationMarker:FLARMarker;

                public function FLARManagerTutorial_Collada_PV3D_pag4 () {
                        this.addEventListener(Event.ADDED_TO_STAGE, this.onAdded);
                }
                                private function onAdded (evt:Event) :void {
                        this.removeEventListener(Event.ADDED_TO_STAGE, this.onAdded);
                        this.flarManager = new FLARManager("../resources/flar/flarConfig.xml", new FLARToolkitManager(), this.stage);
                        this.flarManager.addEventListener(FLARMarkerEvent.MARKER_ADDED, this.onMarkerAdded);
                        this.flarManager.addEventListener(FLARMarkerEvent.MARKER_UPDATED, this.onMarkerUpdated);
                        this.flarManager.addEventListener(FLARMarkerEvent.MARKER_REMOVED, this.onMarkerRemoved);
                        this.flarManager.addEventListener(Event.INIT, this.onFlarManagerInited);}
```

6.3 Realidad Aumentada: Código.

```
        private function onFlarManagerInited (evt:Event) :void {
        this.flarManager.removeEventListener(Event.INIT, this.onFlarManagerInited);
        this.scene3D = new Scene3D();
        this.viewport3D = new Viewport3D(this.stage.stageWidth, this.stage.stageHeight);
        this.addChild(this.viewport3D);
        this.camera3D = new FLARCamera_PV3D(this.flarManager, new Rectangle(0, 0, this.stage.stageWidth, this.stage.stageHeight));
        this.renderEngine = new LazyRenderEngine(this.scene3D, this.camera3D, this.viewport3D);
        this.pointLight3D = new PointLight3D();
        this.pointLight3D.x = 1000;
        this.pointLight3D.y = 1000;
        this.pointLight3D.z = -1000;

        var model:DAE = new DAE(true, "model", true);
        model.load("../resources/assets/armario.DAE");
        model.rotationX = 90;
        model.rotationZ = 150;
        model.scale = 3.5;

        this.modelContainer = new DisplayObject3D();
        this.modelContainer.addChild(model);
        this.modelContainer.visible = false;
        this.scene3D.addChild(this.modelContainer);
        this.addEventListener(Event.ENTER_FRAME, this.onEnterFrame);

}       private function onMarkerAdded (evt:FLARMarkerEvent) :void {
                if (evt.marker.patternId == 7)
                {this.modelContainer.visible = true;
                this.activeMarker = evt.marker; }

                if (evt.marker.patternId == 8)
                {this.rotationMarker = evt.marker;}

}         private function onMarkerUpdated (evt:FLARMarkerEvent) :void {
                if (evt.marker.patternId == 7)
                {this.modelContainer.visible = true;
                this.activeMarker = evt.marker;}

                if (evt.marker.patternId == 8)
                { trace ("rot: " + this.rotationMarker.rotationZ + ".");}
}
```

6.3 Realidad Aumentada: Código.

```
        private function onMarkerRemoved (evt:FLARMarkerEvent) :void {
                trace("["+evt.marker.patternId+"] removed");
                if (evt.marker.patternId == 7)
                {this.modelContainer.visible = false;
                 this.activeMarker = null; }
                 if (evt.marker.patternId == 8)
                 { this.rotationMarker = null; }
        }
          private function onEnterFrame (evt:Event) :void {
                if (this.activeMarker) {
                this.modelContainer.transform = PVGeomUtils.convertMatrixToPVMatrix(this.activeMarker.transformMatrix);
                this.modelContainer.moveBackward(50);
                if(this.rotationMarker) {
                this.modelContainer.roll(-this.rotationMarker.rotationZ);}
                }

                this.renderEngine.render();
        }
}
```

6.3 Realidad Aumentada: Código.

Quinta escena:

```
package examples {
        import com.transmote.flar.FLARManager;
        import com.transmote.flar.camera.FLARCamera_PV3D;
        import com.transmote.flar.marker.FLARMarker;
        import com.transmote.flar.marker.FLARMarkerEvent;
        import com.transmote.flar.tracker.FLARToolkitManager;
        import com.transmote.flar.utils.geom.PVGeomUtils;
        import flash.display.Sprite;
        import flash.events.Event;
        import flash.geom.Rectangle;
        import org.papervision3d.cameras.Camera3D;
        import org.papervision3d.lights.PointLight3D;
        import org.papervision3d.objects.DisplayObject3D;
        import org.papervision3d.objects.parsers.DAE;
        import org.papervision3d.render.LazyRenderEngine;
        import org.papervision3d.scenes.Scene3D;
        import org.papervision3d.view.Viewport3D;

                public class FLARManagerTutorial_Collada_PV3D_pag5 extends Sprite {
                private var flarManager:FLARManager;
                private var scene3D:Scene3D;
                private var camera3D:Camera3D;
                private var viewport3D:Viewport3D;
                private var renderEngine:LazyRenderEngine;
                private var pointLight3D:PointLight3D;
                private var activeMarker:FLARMarker;
                private var modelContainer:DisplayObject3D;
                private var activeMarker1:FLARMarker;

                private var modelContainer1:DisplayObject3D;

                public function FLARManagerTutorial_Collada_PV3D_pag5 () {
                        this.addEventListener(Event.ADDED_TO_STAGE, this.onAdded);
                }
                                private function onAdded (evt:Event) :void {
                        this.removeEventListener(Event.ADDED_TO_STAGE, this.onAdded);
                        this.flarManager = new FLARManager("../resources/flar/flarConfig.xml", new FLARToolkitManager(), this.stage);
                        this.flarManager.addEventListener(FLARMarkerEvent.MARKER_ADDED, this.onMarkerAdded);
                        this.flarManager.addEventListener(FLARMarkerEvent.MARKER_UPDATED, this.onMarkerUpdated);
                        this.flarManager.addEventListener(FLARMarkerEvent.MARKER_REMOVED, this.onMarkerRemoved);
                        this.flarManager.addEventListener(Event.INIT, this.onFlarManagerInited);}
```

6.3 Realidad Aumentada: Código.


```
private function onFlarManagerInited (evt:Event) :void {
this.flarManager.removeEventListener(Event.INIT, this.onFlarManagerInited);
this.scene3D = new Scene3D();
this.viewport3D = new Viewport3D(this.stage.stageWidth, this.stage.stageHeight);
this.addChild(this.viewport3D);
this.camera3D = new FLARCamera_PV3D(this.flarManager, new Rectangle(0, 0, this.stage.stageWidth, this.stage.stageHeight));
this.renderEngine = new LazyRenderEngine(this.scene3D, this.camera3D, this.viewport3D);
this.pointLight3D = new PointLight3D();
this.pointLight3D.x = 1000;
this.pointLight3D.y = 1000;
this.pointLight3D.z = -1000;

var model:DAE = new DAE(true, "model", true);
model.load("../resources/assets/letras.DAE");
model.rotationX = 90;
model.rotationZ = 90;
model.scale = 1.3;

var model1:DAE = new DAE(true, "model", true);
model1.load("../resources/assets/zapatillas.DAE");
model1.rotationX = 90;
model1.rotationZ = 100;
model1.scale = 0.3;

this.modelContainer = new DisplayObject3D();
this.modelContainer.addChild(model);
this.modelContainer.visible = false;
this.scene3D.addChild(this.modelContainer);
this.modelContainer1 = new DisplayObject3D();
this.modelContainer1.addChild(model1);
this.modelContainer1.visible = false;
this.scene3D.addChild(this.modelContainer1);

this.addEventListener(Event.ENTER_FRAME, this.onEnterFrame);
}       private function onMarkerAdded (evt:FLARMarkerEvent) :void {
trace("["+evt.marker.patternId+"] added");
if (evt.marker.patternId == 10)
{this.modelContainer.visible = false;
this.activeMarker = evt.marker; }
```

6.3 Realidad Aumentada: Código.


```
                    if (evt.marker.patternId == 11)
                    {this.modelContainer.visible = true; }

                    if (evt.marker.patternId == 9)
                    {this.modelContainer1.visible = true;
                    this.activeMarker1 = evt.marker; }
        }
            private function onMarkerUpdated (evt:FLARMarkerEvent) :void {
                    trace("["+evt.marker.patternId+"] updated");
                    if (evt.marker.patternId == 10)
                    {this.modelContainer.visible = false;

                     this.activeMarker = evt.marker;}
                    if (evt.marker.patternId == 11)
                    {this.modelContainer.visible = true;}

                    if (evt.marker.patternId == 9)
                    {this.modelContainer1.visible = true;
                     this.activeMarker1 = evt.marker;}
        }
            private function onMarkerRemoved (evt:FLARMarkerEvent) :void {
                    trace("["+evt.marker.patternId+"] removed");
                    if (evt.marker.patternId == 10)
                    {this.modelContainer.visible = false;
                    this.activeMarker = null;}

                     if (evt.marker.patternId == 11)
                    {this.modelContainer.visible = false;
                     this.activeMarker = null;}

                    if (evt.marker.patternId == 9)
                    {this.modelContainer1.visible = false;
                    this.activeMarker1 = null;}
         }
            private function onEnterFrame (evt:Event) :void {
                    if (this.activeMarker) {
                    this.modelContainer.transform = PVGeomUtils.convertMatrixToPVMatrix(this.activeMarker.transformMatrix); }
                    if (this.activeMarker1) {
                    this.modelContainer1.transform = PVGeomUtils.convertMatrixToPVMatrix(this.activeMarker1.transformMatrix);}
                    this.renderEngine.render(); }
        }
}
```

6.4 Modelos 3D.

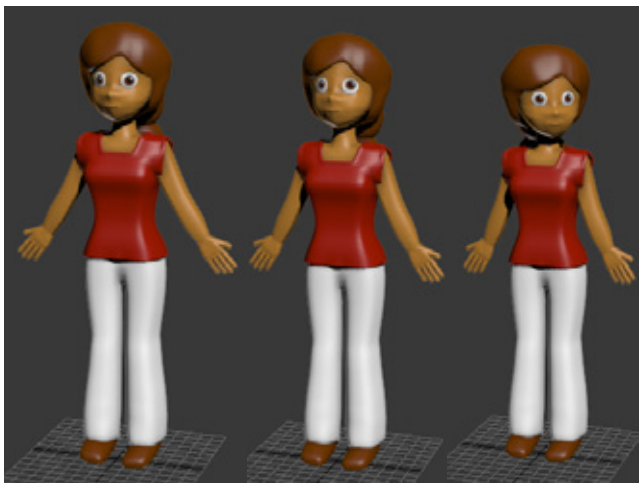
Modelo 3D personaje principal.

-Número de polígonos: 23064 polígonos.

-Texturas: una textura aplicada, 4 colores planos.

-Animación: Sí.

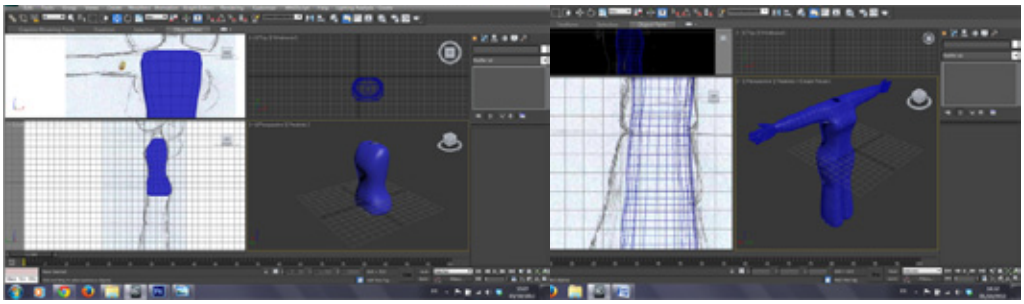
Secuencia de la animación.

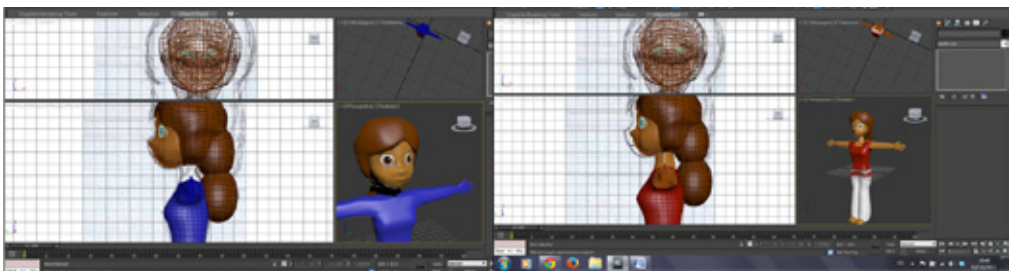


6.4 Modelos 3D.

Imágenes del proceso:

Para el modelado del personaje se hicieron dos bocetos, el primero de frente y el otro de perfil y se colocaron de fondo en el programa para que sirvieran de referencia. Se comenzó modelando el cuerpo a través de cajas convertidas en polígonos editables, moviendo sus vértices para conseguir la forma deseada. Luego se continuó por la cabeza, ojos, boca y pelo, para finalmente unirlos y aplicarles los materiales. Los modificadores más utilizados fueron Symetry (genera simetría) y TurboSmooth (suaviza las formas).





6.4 Modelos 3D.

Modelo 3D sofá.

-Número de polígonos: 17572 polígonos.

-Texturas: tres texturas aplicadas.

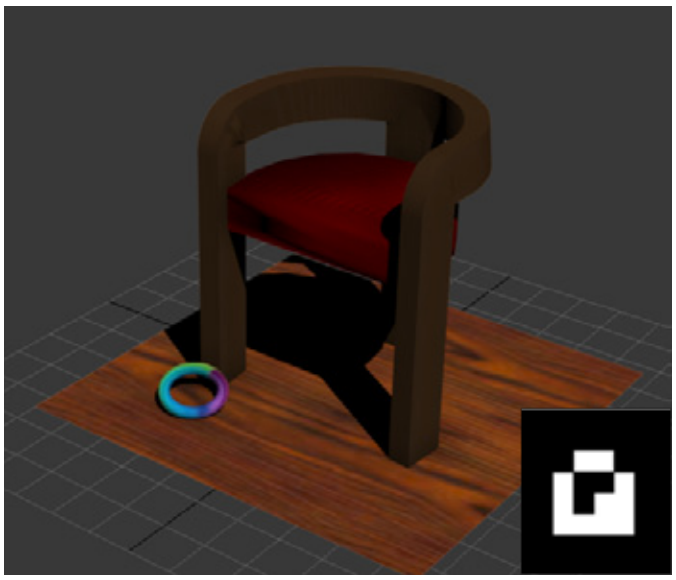
-Animación: No.

Modelo 3D silla y pulsera.

-Número de polígonos: 1276 polígonos.

-Texturas: cuatro texturas aplicadas.

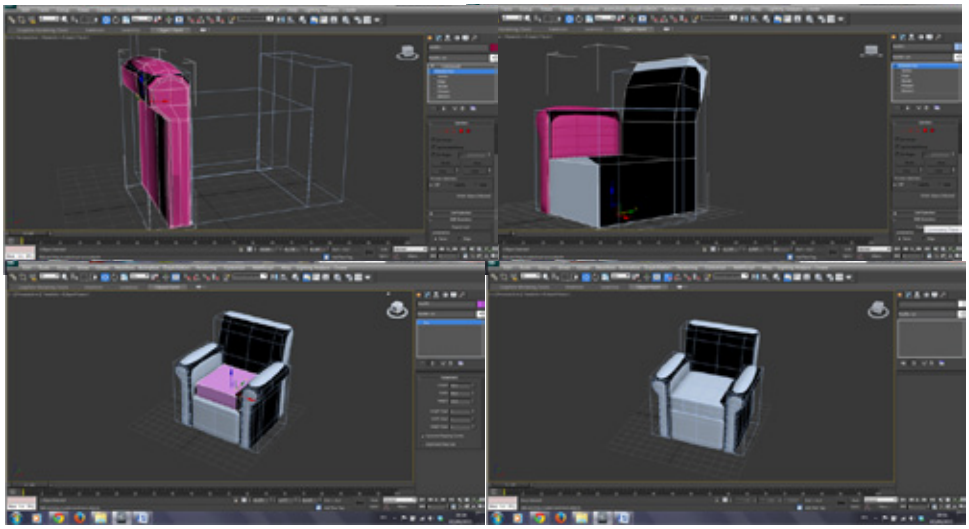
-Animación: No.

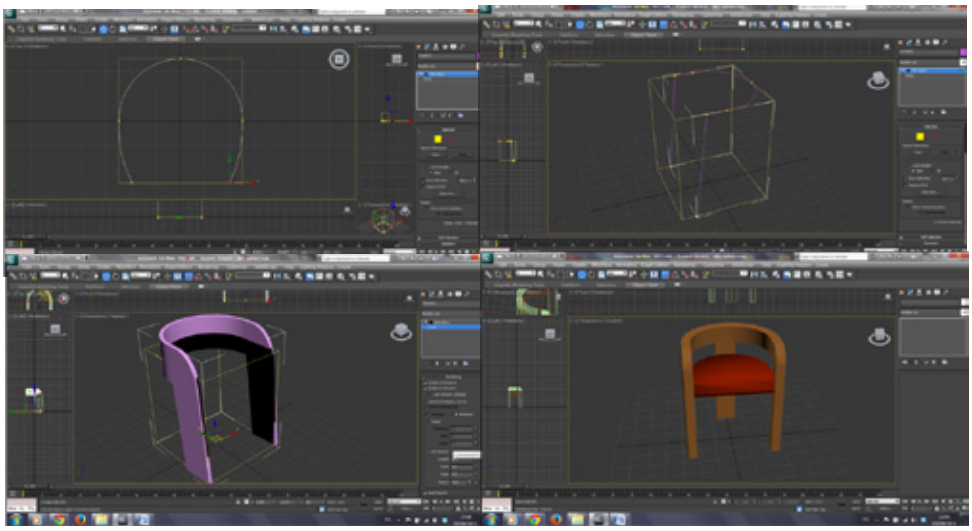


6.4 Modelos 3D.

Imágenes del proceso:

Tanto el modelado 3D del sofá como el de la silla se hicieron siguiendo unos tutoriales encontrados en la web. El sofá se hizo de la misma forma qu el modelo del personaje, a través de cajas convertidas en polígonos editables. La silla se hizo a través de splines 2D a los que se aplicó distintas operaciones para darle tres dimensiones.





6.4 Modelos 3D.

Modelo 3D estantería.

-Número de polígonos: 3792 polígonos.

-Texturas: once texturas aplicadas.

-Animación: No.

Modelo 3D estante.

-Número de polígonos: 37934 polígonos.

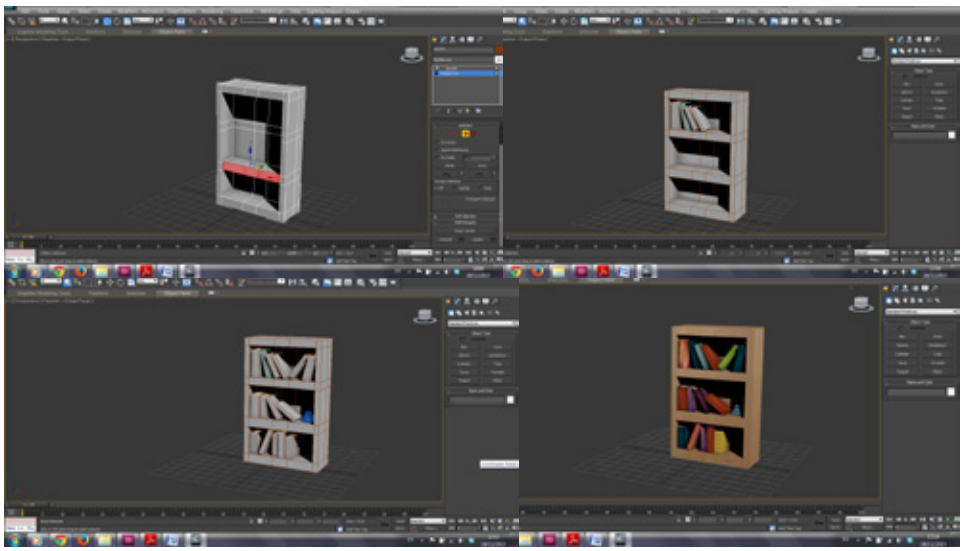
-Texturas: ninguna textura aplicada, 12 colores planos.

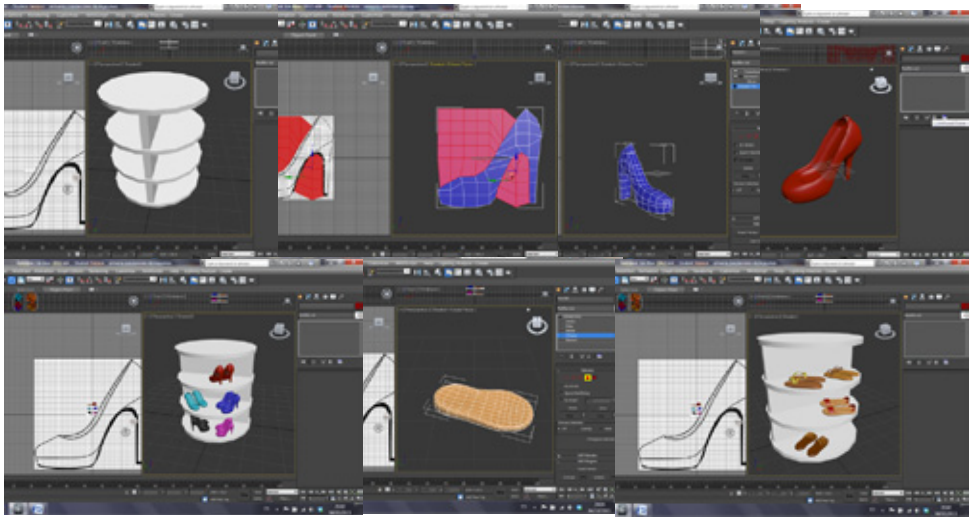
-Animación: No.

6.4 Modelos 3D.

Imágenes del proceso:

La estantería de libros se ha realizado a través del modelado poligonal, a partir de una caja convertida en polígono editable. Los libros son también cajas. En el caso del estante de zapatos, se han realizado los tacones a partir de un plano extruido, en el que se ha imitado la forma de un tacón de perfil a partir de una imagen. El resto se ha realizado con polígonos editables.





6.4 Modelos 3D.

Modelo 3D pantalla teléfono.

-Número de polígonos: 2 polígonos.

-Texturas: una textura aplicada.

-Animación: Sí.

Secuencia de la animación.

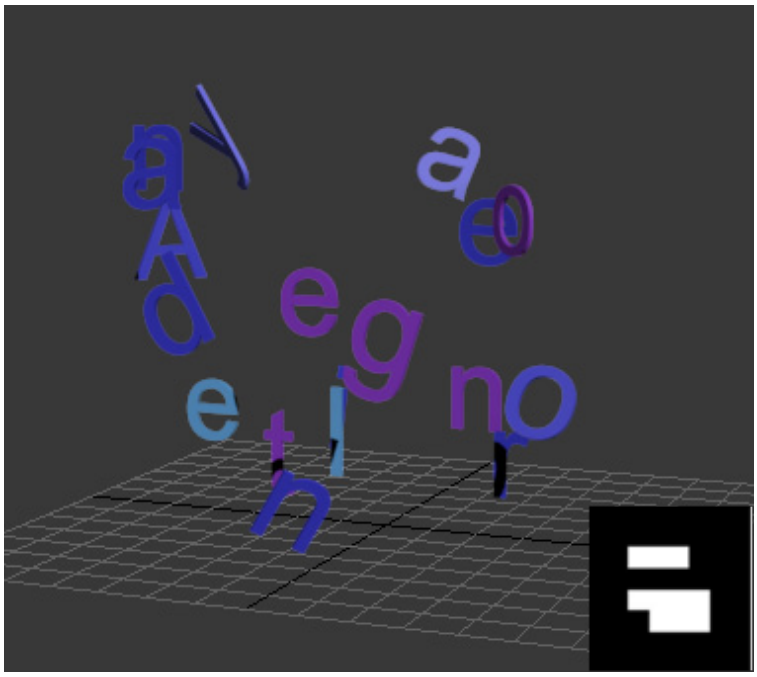
6.4 Modelos 3D.

Modelo 3D letras animadas.

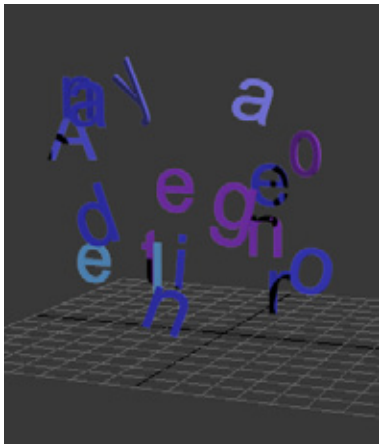
-Número de polígonos: 7240 polígonos.

-Texturas: ninguna textura aplicada, 4 colores planos.

-Animación: Sí.



Secuencia de la animación.



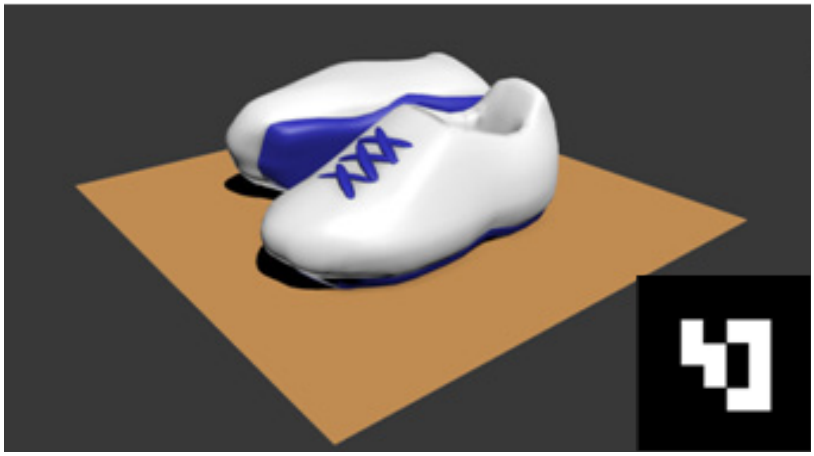
6.4 Modelos 3D.

Modelo 3D zapatillas.

-Número de polígonos: 7002 polígonos.

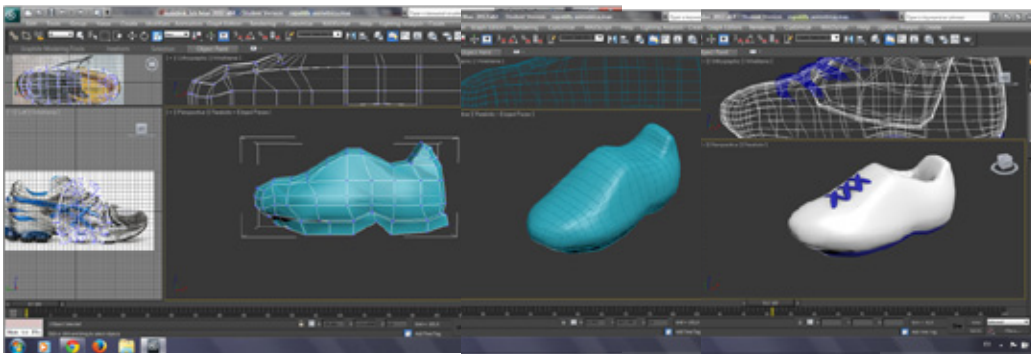
-Texturas: ninguna textura aplicada, dos colores planos.

-Animación: No.



Imágenes del proceso:

Para el modelado de las zapatillas también se ha partido de imágenes de referencia, con las que se ha trabajado el perfil y la vista superior de la zapatilla. Se ha partido también de una caja convertida en polígono editable y se ha trabajado con sus vértices y caras.



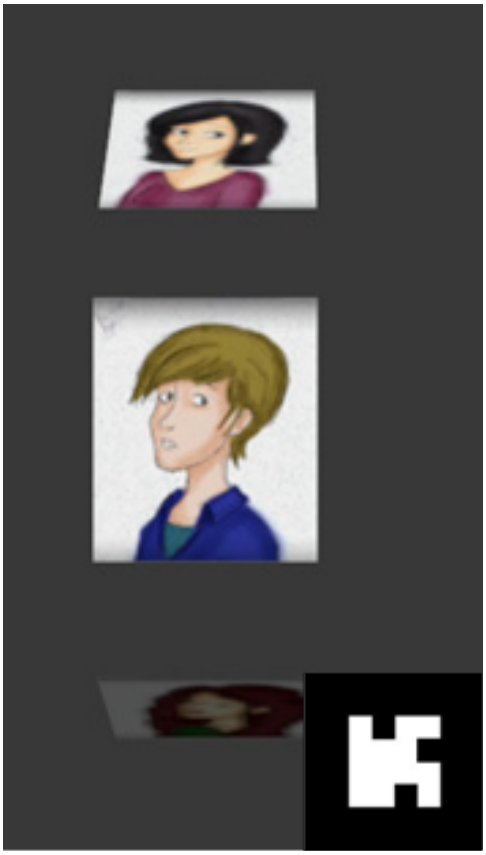
6.4 Modelos 3D.

Planos 2D con imagen.

-Número de polígonos: 6 polígonos.

-Texturas: tres texturas aplicadas.

-Animación: No.



Texturas aplicadas.

6.5 Diseño gráfico.

El diseño gráfico del libro está compuesto por distintos elementos integrados en las páginas:

-Texto: el tipo de fuente utilizado ha sido Papyrus, a tamaño 20 para el texto y 30 para los títulos.
-Dibujos a modo de boceto.
-Imágenes de texturas para fondos, objetos, etc.
-Elementos vectoriales como marcos, símbolos, etc.

Texto.

La fuente que se ha utilizado para el texto del libro es Papyrus, con un tamaño de 20 pt para los párrafos y de 30 pt para los títulos.

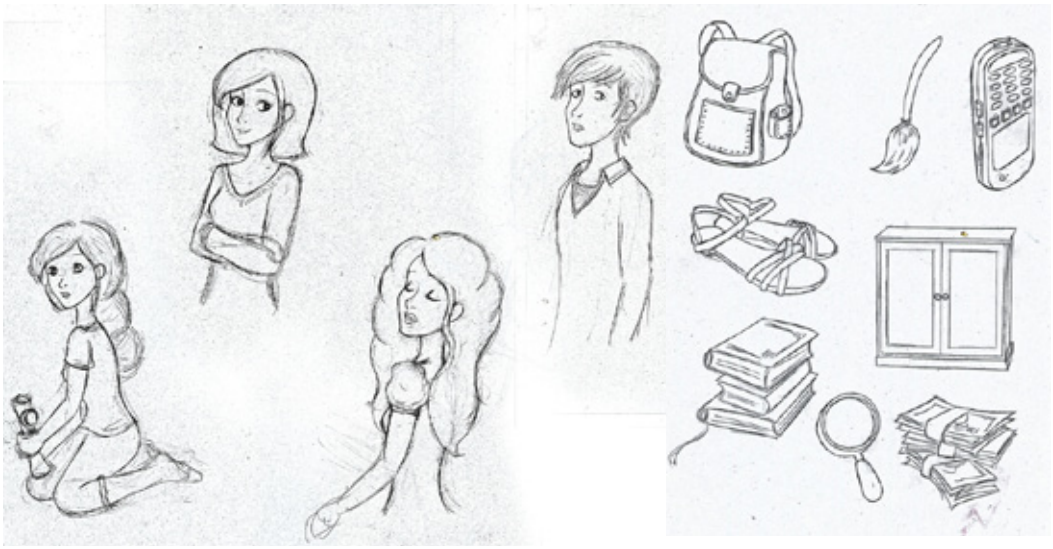
Sospechoso 3: Alumna...

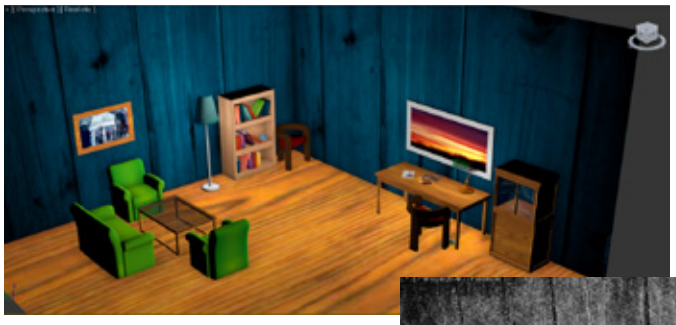
No he encontrado pruebas contra Adela, así que visito a la última sospechosa: Elena.

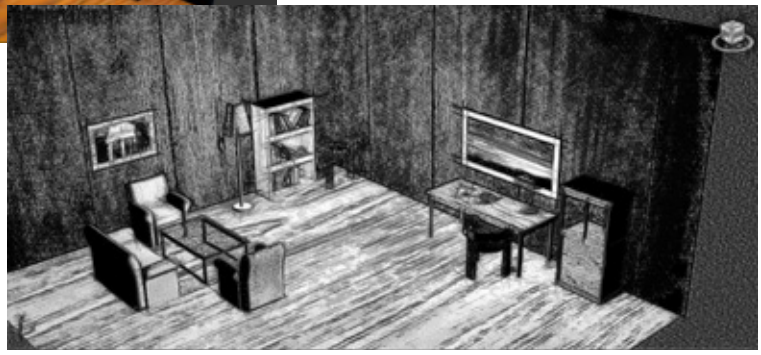
6.5 Diseño gráfico.

Dibujos a modo de boceto.

Se ha trabajado en dibujos de los personajes de la historia, así como de distintos objetos a los que se hace referencia en las páginas del libro. Para una de las escenas, se ha partido de una escena 3D creada con 3ds MAX, cuya imagen se ha editado con la herramienta Photoshop para darle aspecto de boceto.



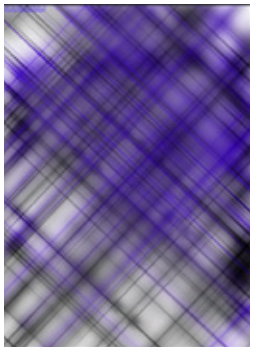




6.5 Diseño gráfico.

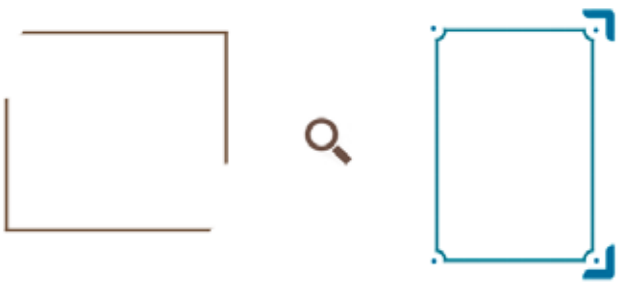
Imágenes como texturas.

Se han utilizado distintas imágenes como texturas, tanto para el fondo de las páginas, como para otros objetos. A continuación se muestran algunas de las texturas finales, con las combinaciones y efectos ya hechos:

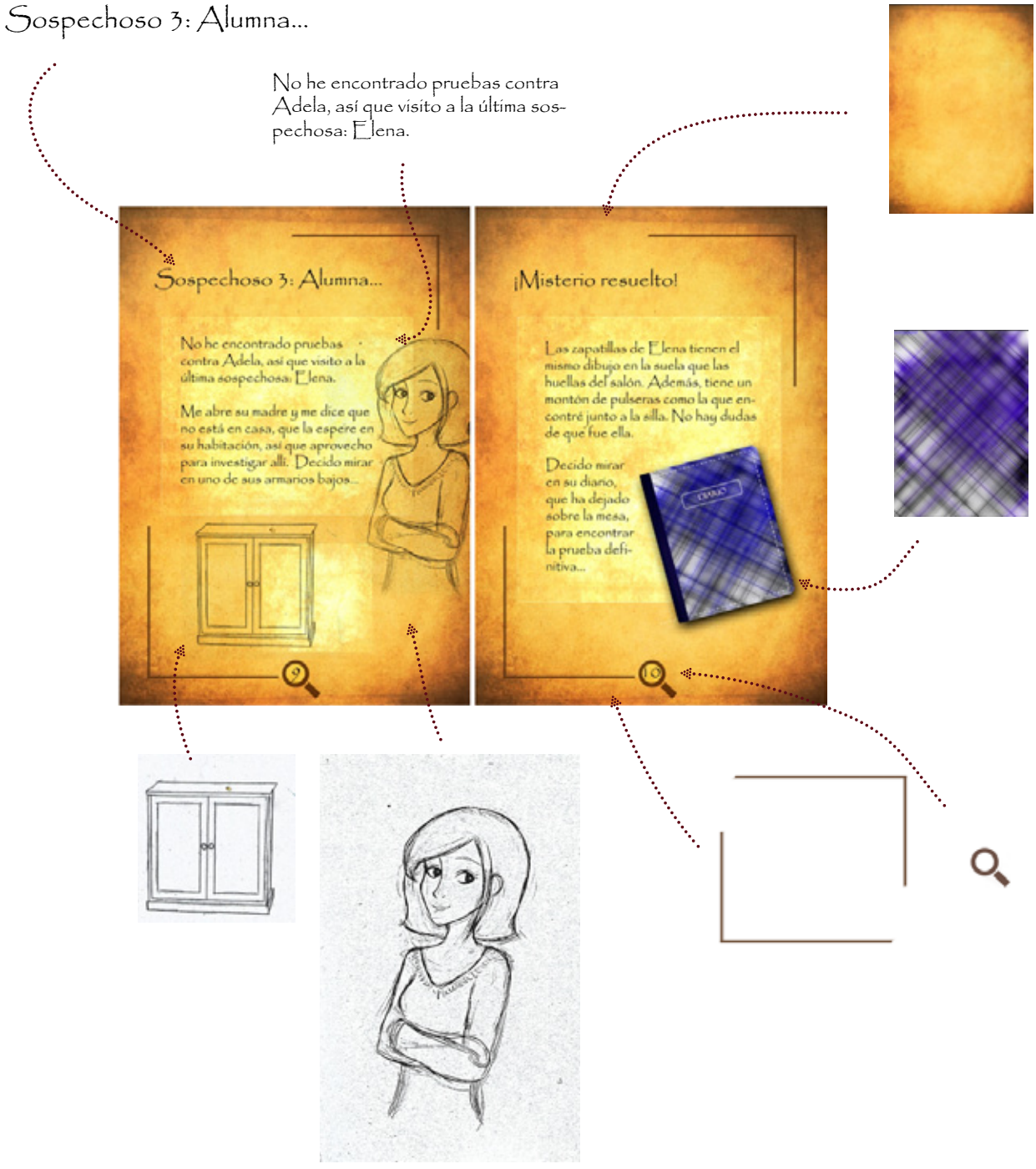

Elementos vectoriales.

Por último, se han incluido distintos elementos vectoriales creado con el programa Illustrator, tanto para las páginas de libro como para otros objetos sobre ellas.



6.5 Diseño gráfico.

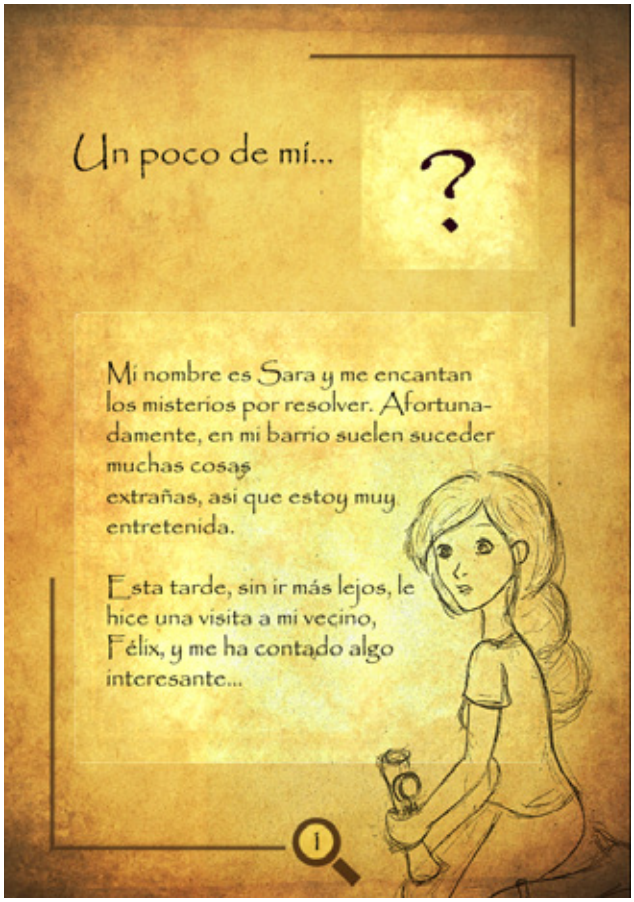
Ejemplo de la intergración de los elementos en la escena:



6.5 Diseño gráfico.

Primera página.

El marcador asociado al modelo 3D de la protagonista de la historia se encuentra bajo la solapa con un interrogante impreso. Cuando la solapa se levanta, el modelo aparece. A continuación se muestra la primera página con la solapa cerrada y abierta.



6.5 Diseño gráfico.

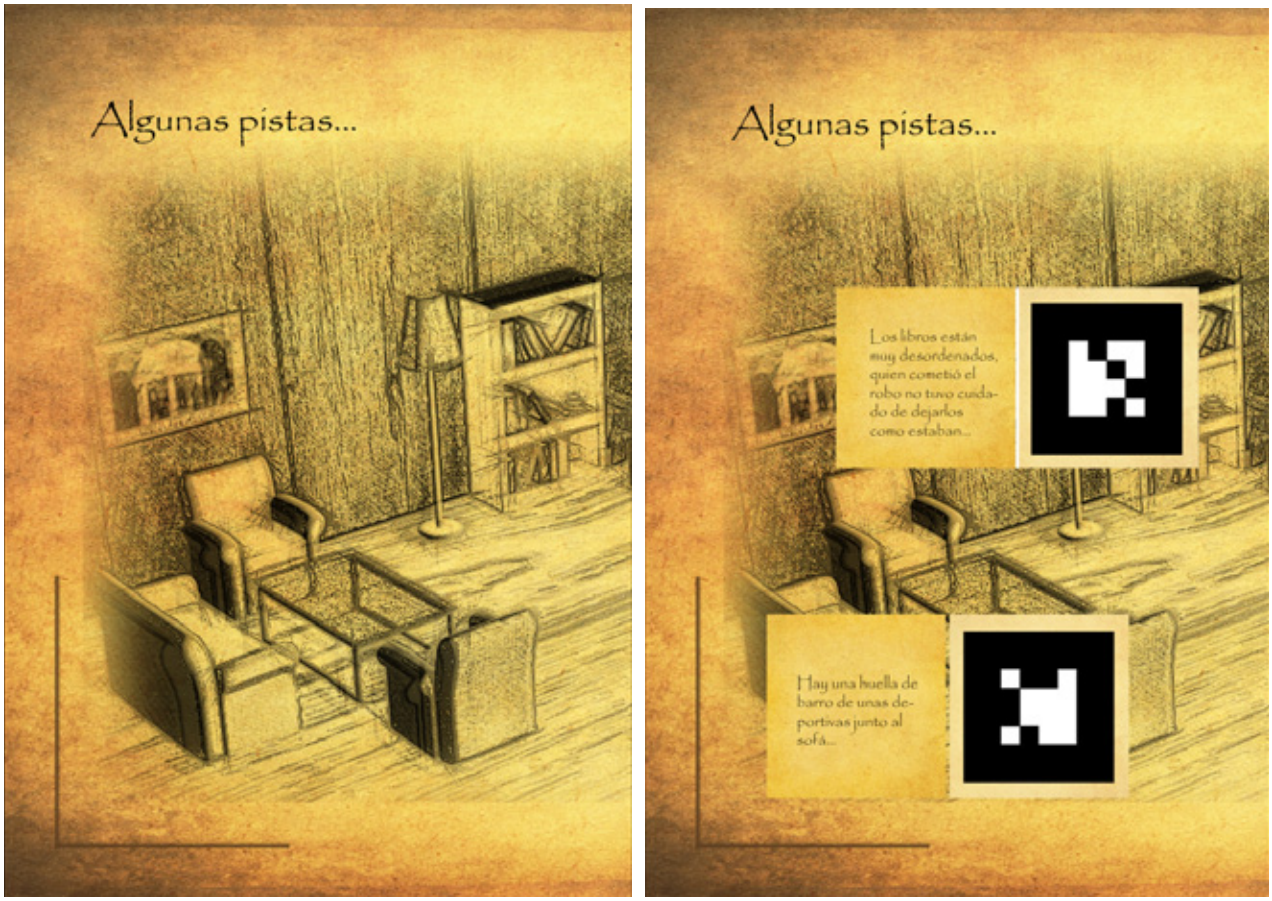
Segunda página.

El marcador asociado al conjunto de planos 2D con imagen se encuentra bajo la solapa que representa un álbum de fotos. Cuando la solapa se levanta, el modelo aparece. A continuación se muestra la segunda página con la solapa cerrada y abierta.

6.5 Diseño gráfico.

Tercera página.

Las solapas que esconden los marcadores de los muebles se integran en la escena, completando la imagen. Cuando las solapas se levantan, el modelo aparece. A continuación se muestra la tercera página con sus dos solapas cerradas y abiertas.



6.5 Diseño gráfico.

Cuarta página.

La solapa que esconde el marcador del tercer mueble se integra en la escena, completando la imagen. Cuando la solapa se levanta, el modelo aparece. A continuación se muestra la cuarta página con la solapa abierta y cerrada.

6.5 Diseño gráfico.

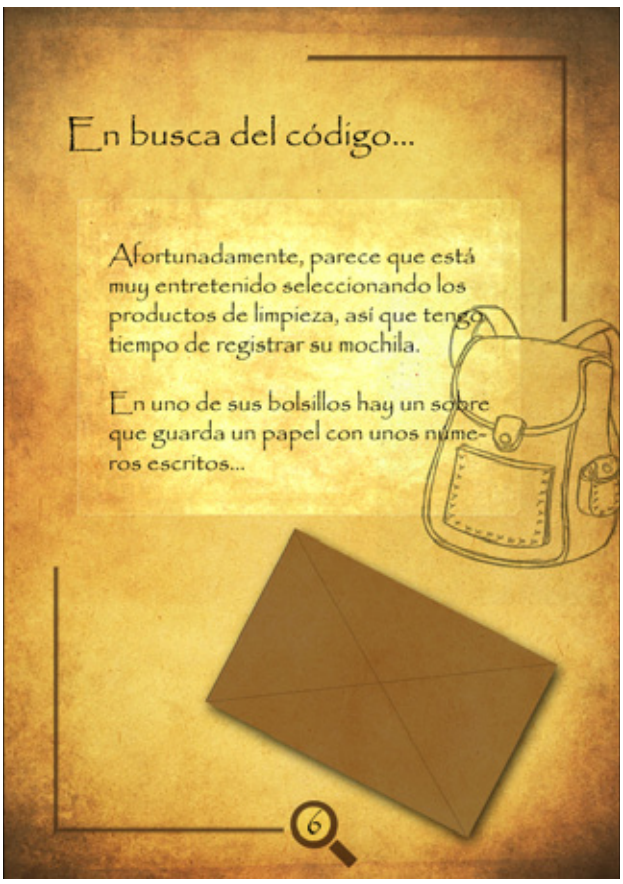
Quinta página.

El marcador que contiene el modelo de la pantalla de teléfono está bajo la solapa en la que hay dibujada un teléfono. Cuando la solapa se levanta, el modelo aún no es visible. Sin embargo, cuando el segundo marcador de la escena (en la sexta página) se coloque a su lado, el modelo será visible y se reproducirá un clip de música. A continuación se muestra la quinta página con la solapa abierta y cerrada.

6.5 Diseño gráfico.

Sexta página.

El marcador que activa el modelo de la pantalla de teléfono y el sonido se encuentra guardado en un sobre. A continuación se muestra la sexta página con la solapa abierta y cerrada.



6.5 Diseño gráfico.

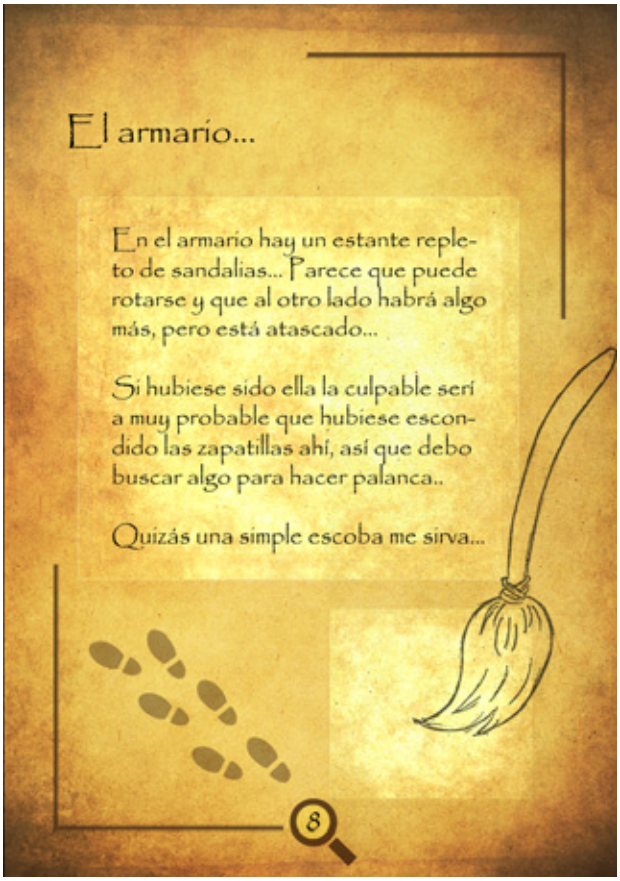
Séptima página.

El marcador que contiene el modelo del estante de zapatos se encuentra bajo la solapa en la que están dibujadas unas sandalias. Cuando la solapa se levanta, el modelo es visible. Al introducir un segundo marcador (en la octaba página) se podrá controlar la rotación del modelo, rotando este segundo marcador. A continuación se muestra la séptima página con la solapa abierta y cerrada.

6.5 Diseño gráfico.

Octaba página.

El marcador que controla la rotación del modelo de la estantería con zapatos se encuentra bajo la solapa que continúa el dibujo de la escoba. Este marcador se puede sacar y llevar junto al otro para iniciar la interacción. A continuación se muestra la octaba página con la solapa abierta y cerrada.



6.5 Diseño gráfico.

Novena página.

El marcador que contiene el modelo 3D de las zapatillas se encuentra bajo la solapa que continúa el dibujo del armario. Cuando la solapa se levanta y se ve el marcador el modelo se hace visible. A continuación se muestra la novena página con la solapa abierta y cerrada.

6.5 Diseño gráfico.

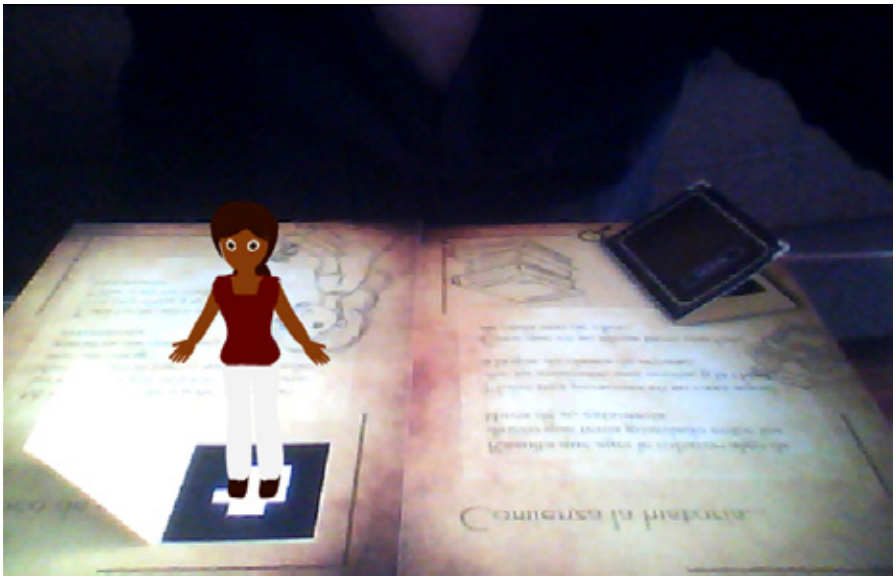
Décima página.

La solapa que representa un diario se despliega dos veces. La primera vez muestra el marcador que contiene el modelo 3D de las letras que se ordenan (animación). La segunda muestra el marcador que activa el modelo. El modelo no es visible hasta que no se muestran los dos marcadores. A continuación se muestra la décima página con la solapa abierta y cerrada.

6.6 Funcionamiento de la aplicación.

Primera escena.

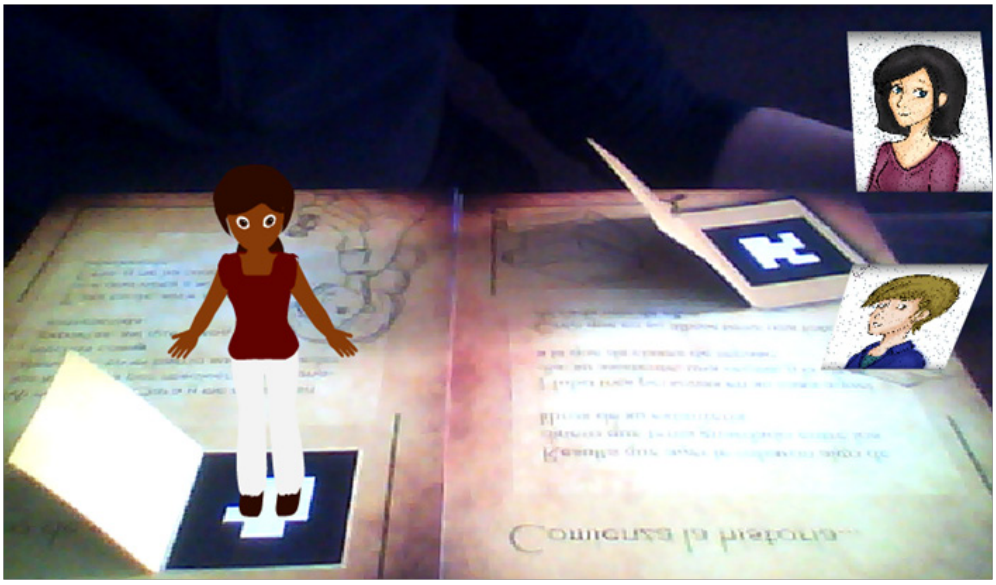
En esta escena hay dos marcadores. Uno de ellos contiene el modelo 3D animado del personaje principal de la historia y el otro contiene tres planos con textura, que rotan verticalmente según el movimiento horizontal del ratón, actuando a modo de galería.



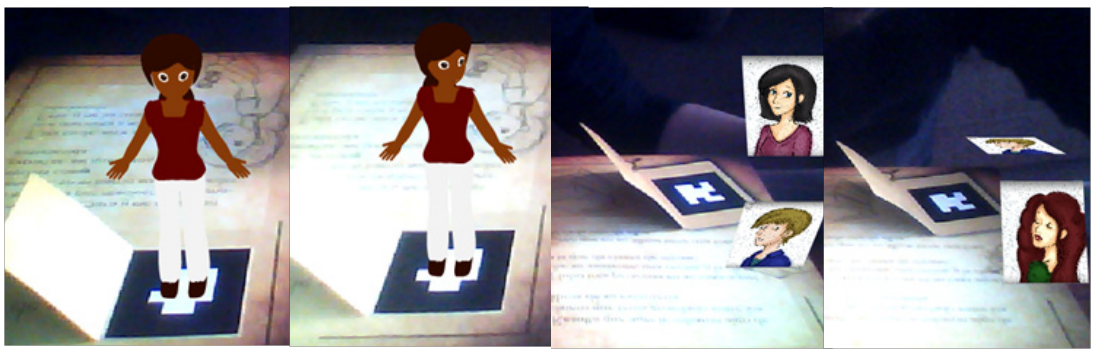
6.6 Funcionamiento de la aplicación.

Primera escena.

Los dos modelos pueden mostrarse simultáneamente.



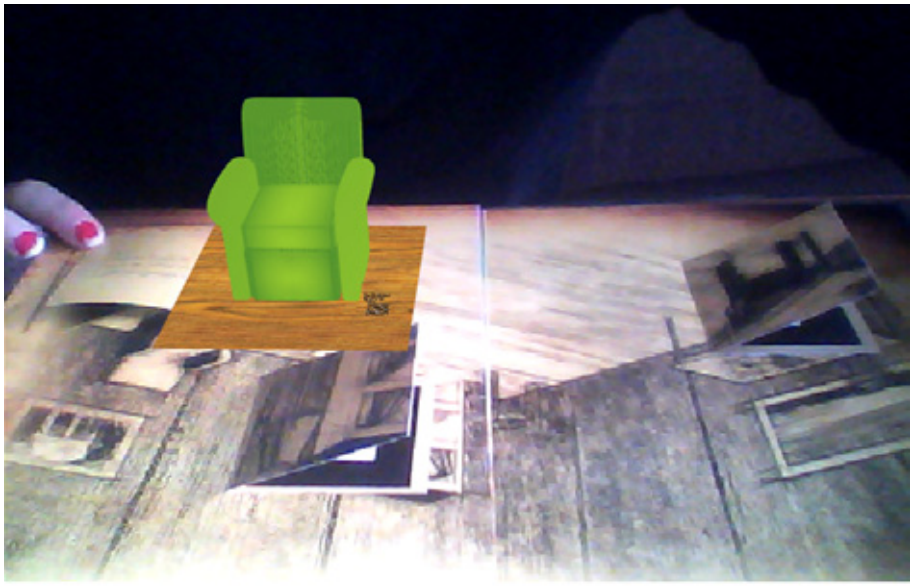
En las siguientes imágenes se ve el movimiento del personaje animado, así como el efecto de galería de las imágenes asociadas al segundo marcador.



6.6 Funcionamiento de la aplicación.

Segunda escena.

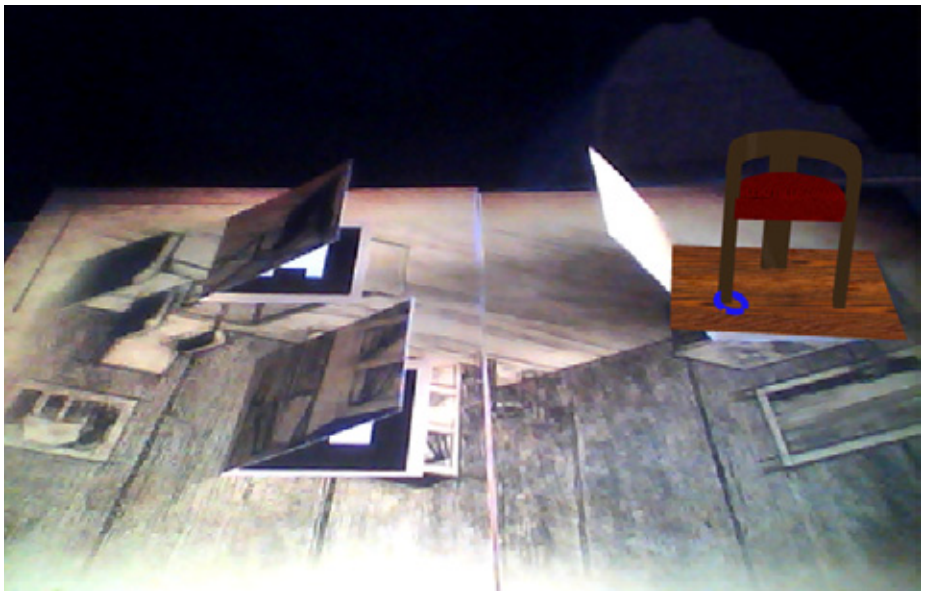
En esta escena hay tres marcadores independientes. Cada uno de ellos contiene un modelo 3D estático. Los tres modelos pueden mostrarse simultáneamente.

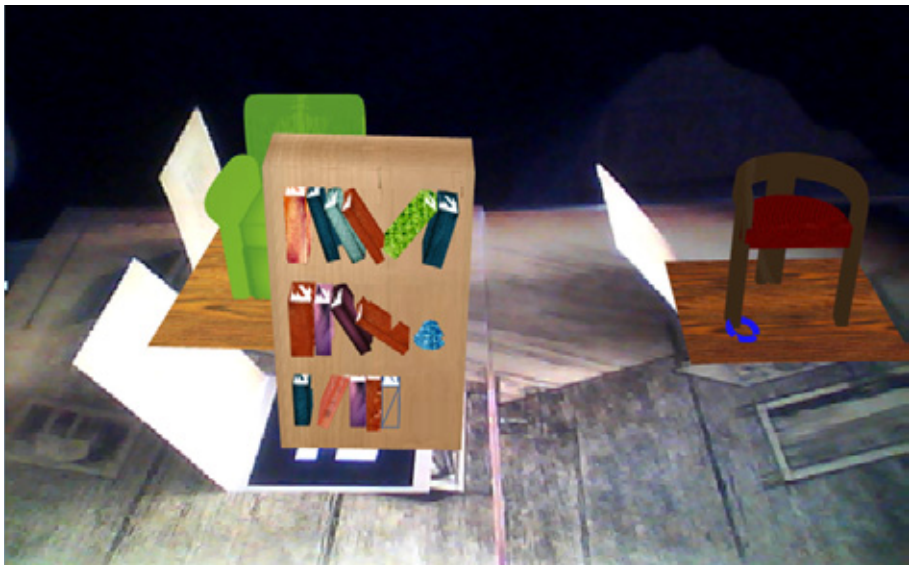


6.6 Funcionamiento de la aplicación.

Segunda escena.

En esta escena hay tres marcadores independientes. Cada uno de ellos contiene un modelo 3D estático. Los tres modelos pueden mostrarse simultáneamente.

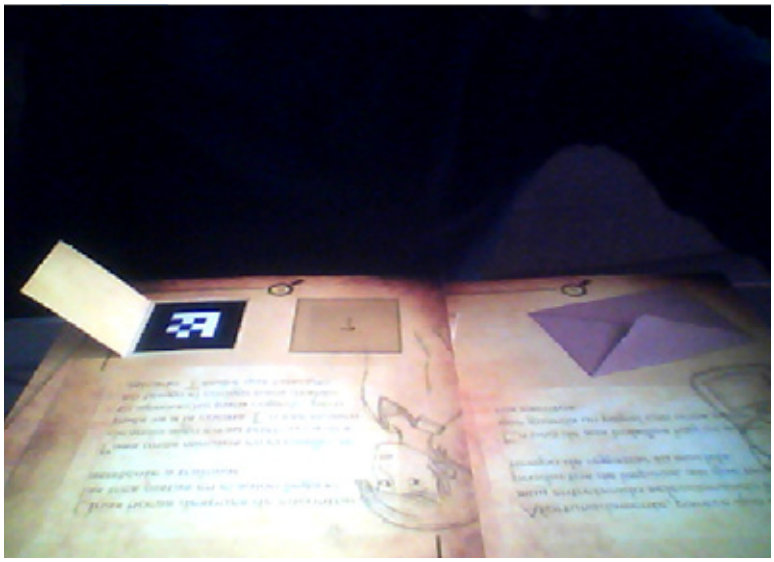


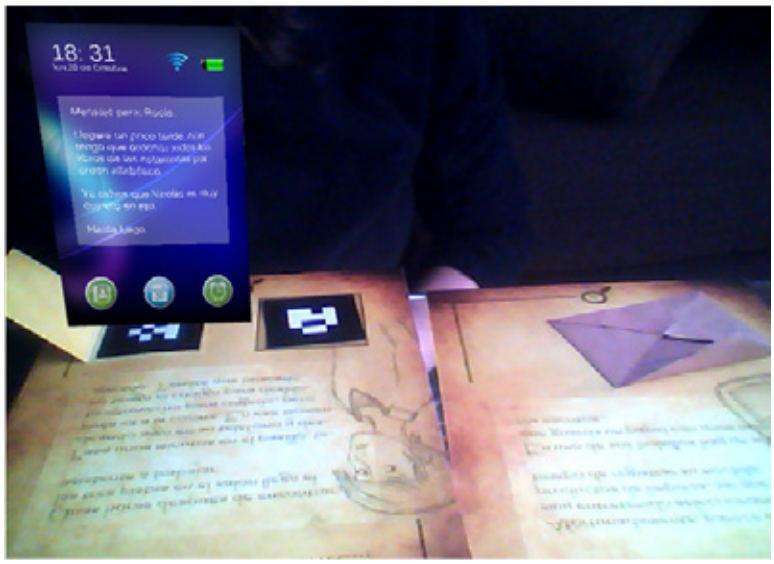


6.6 Funcionamiento de la aplicación.

Tercera escena.

En esta escena dos marcadores que interactúan entres sí. El primero contiene un modelo, pero cuando se muestra solo el modelo no aparece. Al colocar el segundo marcador a su lado aparece el modelo (plano con textura) y se reproduce un clip con sonido.

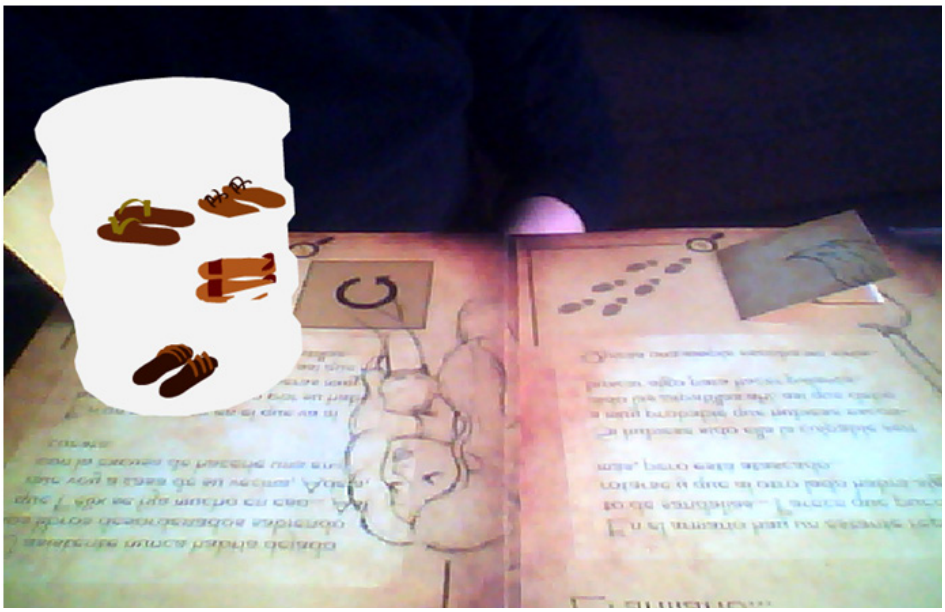




6.6 Funcionamiento de la aplicación.

Cuarta escena.

En esta escena hay dos marcadores que interactúan entre sí. El primero contiene un modelo 3D y el segundo controla la rotación de este modelo.



En la siguiente secuencia de imágenes se muestra cómo al rotar el segundo marcador, el modelo rota con él.

6.6 Funcionamiento de la aplicación.

Quinta escena.

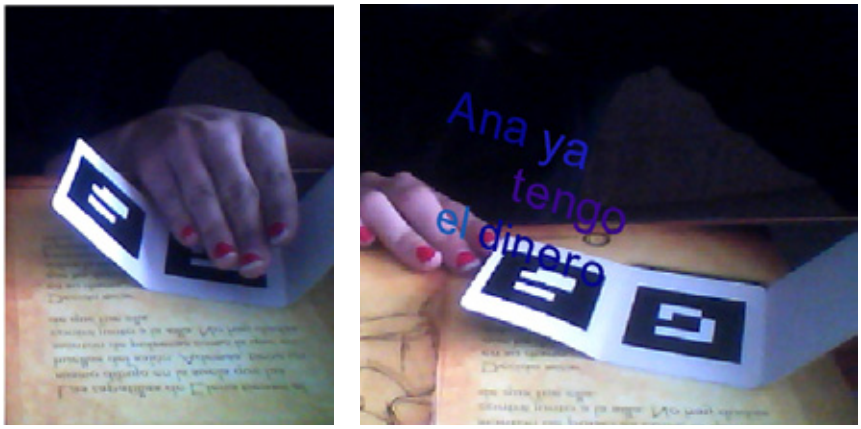
En la última escena hay tres marcadores: uno independiente y dos que interactúan entre sí. El primero muestra un modelo 3D. Los otros dos se combinan para mostrar una animación: uno de ellos contiene el modelo y el otro controla su visibilidad.

6.6 Funcionamiento de la aplicación.

Quinta escena.

En la siguiente secuencia de imágenes se muestra la interacción entre marcadores y el progreso de la animación. Sólo cuando el segundo marcador es visible, se activa la animación.

Al taparse el segundo marcador el modelo desaparece del primero. Si se vuelve a mostrar, el modelo reaparece.



6.7 Programas utilizados para el Trabajo de Fin de Grado.

A continuación se citan los programas más importantes que se han utilizado para realizar el proyecto y se indica en qué área se ha trabajado con cada uno de ellos.

Programas utilizados para el estudio y desarrollo de la Realidad Aumentada:

-FLARManager + Flash Builder: FLARManager ha proporcionado la estructura base y las bibliotecas necesarias para realizar la aplicación, mientras que Flash Builder se ha utilizado para modificar y generar el código.

-Processing: Este programa se ha utilizado durante la fase de estudio de la Realidad Aumentada, realizando las primeras aplicaciones sencillas y facilitando la comprensión del código de estas aplicaciones. Algunos de los ejemplos que se han estudiado con este programa han ayudado a realizar cambios en las aplicaciones creadas para el proyecto.

6.7 Programas utilizados para el Trabajo de Fin de Grado.

Programas utilizados para la creación de modelos 3D:

-3ds MAX: Este programa se ha utilizado durante la fase de estudio sobre la generación de modelos 3D, animación, aplicación de materiales, luces, etc. También ha sido el programa con el que se ha trabajado para realizar los modelos y animaciones que se han utilizado en la aplicación de este Trabajo de Fin de Grado.

-Adobe Photoshop: Este programa ha sido utilizado para crear y modificar las texturas que han sido aplicadas a los modelos 3D como materiales, así como para pintar y modificar los dibujos de las texturas que se han incluido en los tres planos de la galería.

6.7 Programas utilizados para el Trabajo de Fin de Grado.

Programas utilizados para la realización del Diseño gráfico del prototipo del libro:

-Adobe Photoshop: Ha sido el programa que más se ha utilizado para generar el diseño gráfico de las páginas del libro. Se ha trabajado el retoque de imágenes incluidas en él, la aplicación de texturas, texto y distintos efectos. Por último se ha realizado el montaje final de las páginas.

-Adobe Illustrator: Gracias a este programa se han generado algunos de los elementos que se incluyen en las páginas (marcos, solapas, etc.).

Programas utilizados para la maquetación del proyecto:

- Adobe InDesign: Con este programa se han maquetado los apéndices.

6.8 Asignaturas utilizadas durante el desarrollo de este Trabajo de Fin de Grado.

Durante la realización del proyecto se han integrado los conocimientos de muchas de las asignaturas de la carrera, como Entornos 3D, Informática, Comunicación Multimedia, Composición y Edición de imágenes, Diseño gráfico y comunicación, Taller de diseño y Oficina Técnica.

Entornos 3D: Esta asignatura ha proporcionado mucha información sobre la Realidad Aumentada, y sobre las aplicaciones que existen y las posibilidades que ofrece esta tecnología. Además, en esta asignatura se realizaron las primeras pruebas con herramientas de modelado poligonal, lo que ha ayudado al aprendizaje sobre el modelado 3D.

Informática y Comunicación Multimedia: Estas asignaturas han facilitado la comprensió y el manejo de código para generar las aplicaciones de Realidad Aumentada. No obstante, al no haberse profundizado mucho en estas materias, no se han realizado aplicaciones de un alto nivel de complejidad.

-Composición y Edición de imágenes: Esta asignatura ha sido muy útil a la hora de trabajar con el diseño gráfico, ya que ha proporcionado las nociones básicas para editar imágenes y trabajar con composiciones.

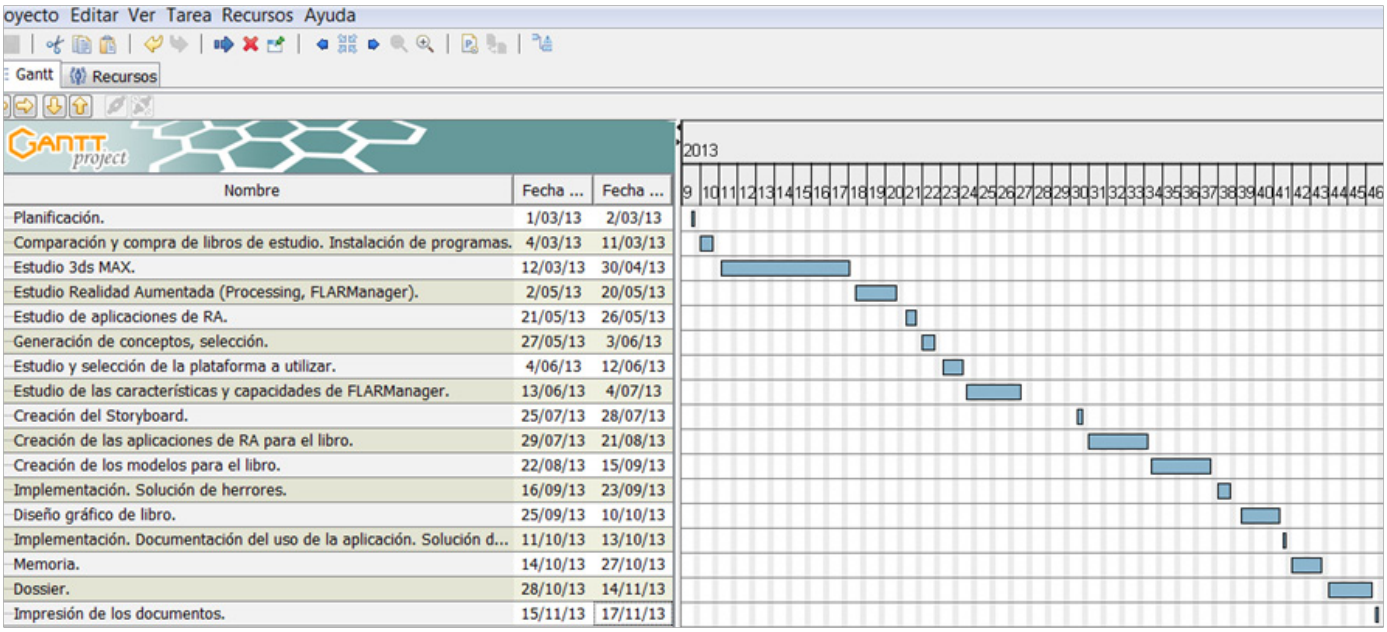
6.8 Asignaturas utilizadas durante el desarrollo de este Trabajo de Fin de Grado.

Taller de diseño: Esta asignatura ha proporcionado los conocimientos necesarios para llevar una buena metodología de trabajo, tanto en el proceso de información, como en el creativo y en el de desarollo del producto.

Oficina técnica: Esta asignatura ha facilitado la planificación del proyecto, ya que ha proporcionado los conocimientos necesarios para poder definir las tareas, tiempos y recursos necesarios durante la realización del proyecto.

6.9 Diagrama de Gantt.

En el siguiente diagrama se muestra el conjunto de tareas que se han realizado para lograr el objetivo de este Trabajo de Fin de Grado, junto a la duración de cada una de ellas y su el período de su realización. Las tareas han llevado a cabo entre los meses de Marzo y Noviembre de 2013. El dossier se ha ido realizando junto a cada una de las tareas, por lo que la tarea con este nombre se refiere a la unificación de contenidos, mejoras e introducción de nuevos contenidos, y maquetación de la misma.



Nombre	Fecha ...	Fecha ...
Planificación.	1/03/13	2/03/13
Comparación y compra de libros de estudio. Instalación de programas.	4/03/13	11/03/13
Estudio 3ds MAX.	12/03/13	30/04/13
Estudio Realidad Aumentada (Processing, FLARManager).	2/05/13	20/05/13
Estudio de aplicaciones de RA.	21/05/13	26/05/13
Generación de conceptos, selección.	27/05/13	3/06/13
Estudio y selección de la plataforma a utilizar.	4/06/13	12/06/13
Estudio de las características y capacidades de FLARManager.	13/06/13	4/07/13
Creación del Storyboard.	25/07/13	28/07/13
Creación de las aplicaciones de RA para el libro.	29/07/13	21/08/13
Creación de los modelos para el libro.	22/08/13	15/09/13
Implementación. Solución de herrores.	16/09/13	23/09/13
Diseño gráfico de libro.	25/09/13	10/10/13
Implementación. Documentación del uso de la aplicación. Solución d...	11/10/13	13/10/13
Memoria.	14/10/13	27/10/13
Dossier.	28/10/13	14/11/13
Impresión de los documentos.	15/11/13	17/11/13

La realización del Trabajo de Fin de Grado ha conllevado un total de 870 horas aproximadas de trabajo.

6.10 Conclusiones.

Recordemos que el objetivo del PFG consiste en especificar un proyecto de un libro basado en Realidad Aumentada para niños entre 7 y 9 años que, gracias a esta tecnología, se vuelve interactivo y utiliza la imagen de un modo más novedoso. Se quiere con ello demostrar el uso de las potencialidades de la RA, resolviendo los problemas tecnológicos que surgirían durante la implementación del libro.

Durante el desarrollo del proyecto se busca integrar en el libro diversas aplicaciones basadas en el uso de marcadores, y mostrar algunos ejemplos de modelos y animaciones que se podrían utilizar en un libro como el propuesto.

A través del aprendizaje de varios programas (FLARManager junto a Flash Builder, 3ds Max, Processing,) y del estudio de las distintas opciones que ofrece la Realidad Aumentada, tanto respecto a filosofías para trabajar con ella, como de plataformas con las que implementarla, se ha podido escoger el modo y el entorno de trabajo, así como realizar la aplicación planteada.

Durante la realización del proyecto se han integrado los conocimientos de muchas de las asignaturas de la carrera, como Entornos 3D, Informática, Comunicación Multimedia, Composición y Edición de imágenes, Diseño gráfico y comunicación, Taller de diseño y Oficina Técnica.

6.10 Conclusiones.

Se ha conseguido el objetivo deseado para el proyecto, ya que se ha logrado mezclar la tecnología de la Realidad Aumentada con un libro infantil, de modo que su contenido se enriquezca. Se han realizado distintas aplicaciones con RA en las que se ha conseguido interacción entre marcadores, muestra de varios modelos, inclusión de sonido, etc. Además, se han diseñado e introducido modelos y animaciones propios para la aplicación con éxito y se ha realizado el diseño gráfico que muestra cómo sería la aplicación real.

La parte más complicada durante el desarrollo de este Trabajo de Fin de Grado ha sido la generación del código para realizar las distintas aplicaciones de Realidad Aumentada ya que, debido a que no se tienen conocimientos muy amplios sobre programación, el proceso de introducción de nuevo código y depuración de errores ha sido muy costoso, y no se han podido realizar aplicaciones de un alto nivel de complejidad.

La parte más interesante ha sido la realización de modelos 3D y animaciones, ya que era un tipo de modelado con el que apenas se había trabajado antes y con el que se han conseguido buenos resultados.

Como trabajo futuro sería mejorable la puesta en marcha de la aplicación que, debido a la capacidad del ordenador de trabajo, tiene problemas (pocos modelos complejos a la vez, problemas al cargar varios

6.10 Conclusiones.

elementos distintos, etc.). Esto obliga a que la aplicación se haya tenido que dividir en aplicaciones más pequeñas, y que no pueda funcionar todo el libro con una misma aplicación. Además, podrían incluirse otro tipo de aplicaciones que incluyesen botones, vídeo, etc.