Escuela Politécnica Superior

Probador virtual: Realidad Aumentada sin marcadores

Grado en Ingeniería Multimedia

Trabajo fin de Grado

Autor:
Carlos Meca López

Tutores:
Rafael Molina Carmona
Patricia Compañ Rosique

Septiembre de 2014

Universitat d'Alacant
Universidad de Alicante

Índice

1. Introducción

Cuando hablamos de realidad aumentada nos referimos a la visión desde un dispositivo tecnológico (ordenador, Smartphone, Tablet, videoconsola…) del mundo físico mezclado con imágenes virtuales. La realidad aumentada lo que hace es añadir contenidos virtuales a la visualización del mundo real, como podemos ver en la figura 1, donde se representa una aplicación móvil de GPS.

Figura 1. Ejemplo de aplicación con realidad aumentada

Los usos de la realidad aumentada son muy diversos, desde el ocio digital hasta aplicaciones médicas. Pero las aplicaciones de la realidad aumentada no se quedan ahí, ya que está siendo utilizada el desarrollo de videojuegos (como Invizimals, de la compañía catalana Novarama, que hace uso de tarjetas RA para interactuar con el juego), en marketing (promocionando todo tipo de productos, desde coches como la campaña del Audi R8 como en moda), en medicina (facilitando campos como la cirugía, tomando datos del interior del paciente de forma no invasiva y mostrándolos en un dispositivo, proporcionando así más seguridad en los pacientes), la educación (complementando a los métodos tradicionales para hacer un aprendizaje más visual

e interactivo, como podemos ver en la figura 2 en una aplicación para aprender sobre la anatomía de los dinosaurios), la arquitectura y el interiorismo (haciendo más reales y atractivos los proyectos ante el presupuesto inicial) o el turismo (proporcionando información en tiempo real al usuario sobre distintos monumentos y actividades). Estos son sólo unos pocos ejemplos de las capacidades de esta tecnología en la vida cotidiana, y explican por qué se ha vuelto tan popular en los últimos años. Aunque todavía quedan muchas aplicaciones para la realidad aumentada por inventar, como aplicaciones para facilitar las tareas del hogar, acciones frecuentes como ir a la compra y poder ver qué alimentos son consumibles por diabéticos o celíacos o la creación de piezas industriales de manera más segura y fácil. [José Urbina, 2012]

Figura 2. La realidad aumentada en la educación

El proyecto a realizar consiste en un probador de ropa virtual. Si bien este problema ya ha sido abordado con anterioridad por otros estudios, la mayoría del software creado utiliza imágenes bidimensionales superpuestas al usuario, lo que limita su uso al no poder este girarse o ver las físicas del tejido. Este proyecto utilizará prendas diseñadas en 3D para tal uso, aumentando así el realismo para el usuario.

La realización de este proyecto nos presenta una serie de retos. El primero y principal al que nos enfrentaremos en todos los proyectos que involucren realidad aumentada es cómo recogemos los datos del mundo real. Se ha elegido la cámara

Kinect de Microsoft, que proporciona además información precisa de la situación del usuario gracias a sus sensores, si bien se usará el SDK de OpenNI en lugar del nativo, que hará el programa también compatible con la cámara de Asus Xtion Pro-Live, además de proporcionar la posibilidad de hacerlo compatible con los sistemas OS X, para los que también está disponible esta librería.

Otro problema es cómo mostraremos las imágenes en pantalla y cómo calcularemos los datos necesarios para ello. Para solucionarlo nos valdremos de Unity3D, un entorno de desarrollo especializado para videojuegos, pero que dada su potencia para mostrar gráficos 3D y sus scripts específicos para Kinect (de Zigfu, basados en OpenNI), harán más sencillo su desarrollo. Otra ventaja de Unity es que podemos compilar el programa para cualquier sistema operativo compatible con las librerías que usaremos y para entornos web. También nos permite programar tanto en Javascript como en C#, y cuenta con una gran comunidad de apoyo y un buen número de tutoriales.

Por último se presenta el problema de la animación, tanto mediante huesos como de las físicas propias de los tejidos. Unity cuenta con un tipo de GameObject preparado para mover tejidos a partir de mallas diseñadas para ello. Estos modelos los haremos mediante Blender, un programa de modelado y animación 3D, que cuenta con la ventaja de ser de código abierto y libre, además de poder exportar estos modelos a una amplia cantidad de formatos en caso de que sea necesario. Para la animación de los huesos usaremos también modelos de Blender usando la opción de armadura, y estos huesos que crearemos los enlazaremos a los puntos que detecta Kinect para moverlos.

La estructura del documento es la siguiente: en el primer apartado se indican los objetivos del desarrollo, separados en motivación y objetivos del proyecto; acto seguido se encuentra el estado del arte, donde se explicará cada una de las tecnologías utilizadas con más detalle; a continuación se encuentra el cuerpo del documento, donde se explicará con detalle todo el trabajo realizado; otro apartado de pruebas; y por último vemos el apartado de conclusiones y desarrollos posteriores.

2. Objetivos

2.1. Motivación

La razón principal de este proyecto es hacer uso de la realidad aumentada, una tecnología que se ha desarrollado muy rápido y ha sido cada vez más apta para el gran público desde hace poco tiempo, en un campo en el que aún puede explotarse bastante como es el textil. Este trabajo se ha diseñado como una manera creativa y útil de hacer uso de la realidad aumentada, además de ser un producto usable por los usuarios que no tengan un conocimiento avanzado en esta tecnología.

Otra meta de este proyecto es crear un comienzo para que los usuarios puedan conocer la realidad aumentada y expandir esta tecnología en el mercado textil, proporcionando un sistema capaz de ser ampliado para usos comerciales. También es un software destinado a atraer público a los comercios y dar la oportunidad de ofrecer al cliente una compra rápida y sencilla a la vez que atraerlo mediante el uso de la realidad aumentada como método publicitario.

2.2. Objetivos del proyecto

Los objetivos a alcanzar son los siguientes:

- Realizar un sistema de realidad aumentada, estudiando su aplicación en el sector textil y diseñar y construir un sistema para la prueba virtual de prendas de vestir.
- Habilitar al sistema para colocar modelos tridimensionales de prendas en el usuario.
- Capacitar al sistema para detectar el libre movimiento del usuario, aplicando físicas al modelo de la prenda.
- Posibilitar al sistema para cambiar entre varios modelos.
- Habilitar al sistema para hacer una captura de pantalla del usuario con la prenda.

- Desarrollar un software multiplataforma. El programa será ejecutable al menos en Windows y OS X.

3. Estado del Arte

El sistema a desarrollar es complejo y en él participan varias disciplinas. Entre ellas encontramos realidad aumentada, el uso de Kinect y Unity, físicas de tejidos y animación mediante estructuras de huesos.

Para conocer el estado del arte se van a presentar una serie de proyectos que, si bien no son exactamente iguales al software a desarrollar, si pueden ilustrar de manera contundente la evolución de este tipo de sistemas y su aceptación de cara al público. Acto seguido se presentará el estado de las tecnologías a utilizar, así como una breve explicación de por qué se van a usar y de cómo funcionan.

3.1. Desarrollos similares

Existen algunas propuestas parecidas al proyecto. Una de ellas es un sistema diseñado por un equipo francés, que se exhibe ahora en un comercio en Moscú. Consta de una pantalla vertical que hace las veces de espejo y un Kinect que capta las imágenes del usuario. El software permitía que el usuario que se pusiese enfrente apareciese en la pantalla con un modelo en realidad aumentada de una prenda de la tienda. Si el usuario movía la mano hacia un punto concreto podía cambiar la prenda y levantando la mano derecha podía fotografiarse con el modelo. La principal diferencia entre el proyecto propuesto y este producto es que los modelos en realidad aumentada que muestra son bidimensionales, con lo que no permite al usuario ver los laterales de la prenda ni aplicársele físicas para darle realismo. Además el proyecto propuesto está pensado para ser multiplataforma y este está pensado para funcionar en un hardware concreto. En la figura 3 podemos ver este sistema en funcionamiento. [Bruce Sterling, 2011]

Figura 3. Probador virtual TopShop

Otro sistema similar, aunque enfocado al calzado, es el desarrollado por Goertz, una tienda virtual de zapatos, que utiliza tres Kinects para escanear al usuario y detectar sus pies y manos para colocar el zapato con precisión, además de poder conocer su precio y comprarlos vía códigos QR. El software se mostraba en una pantalla grande para captar la atención del público, siendo instalado en un gran número de centros comerciales de toda Alemania. La figura 4 ilustra este sistema. [Aden Hepburn, 2012]

Figura 4 Tienda virtual de zapatos de Goertz

3.2. Fundamentos de la realidad aumentada

Cuando hablamos de realidad aumentada nos referimos a la combinación de elementos del mundo real con información generada por un ordenador y añadida y mostrada mediante un dispositivo en tiempo real a esta, o sea, la integración del mundo real con un mundo virtual que ofrece al usuario información generada por un computador. Esta definición es una aproximación de la del profesor Juan de Urraza. [Prof. Juan de Urraza, 2014]

Los primeros esbozos de realidad aumentada datan del año 1968, cuando Ivan Sutherland crea el primer HMD (*head-mounted display*, visualizador montado sobre la cabeza), un dispositivo que se componía de unos sensores con potenciómetros que medían los cambios en la orientación de la cabeza del usuario, un sistema de visualización mediante dos pantallas y un brazo articulado que lo sujetaba en el techo. Cuando los sensores detectaban algún movimiento se enviaban al ordenador y este generaba los pares estereoscópicos de imágenes en 3D que eran representadas en el dispositivo de visualización en formato alambre debido a las restricciones de la época. La figura 5 es una fotografía del HMD.

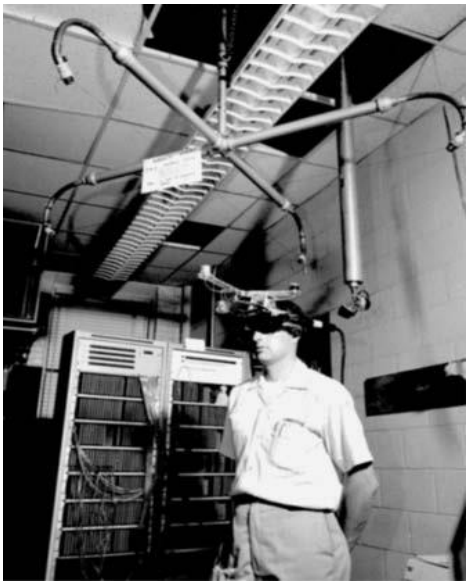

Figura 5 HMD de Ivan Sutherland

No es hasta la década de los noventa cuando se empieza a utilizar el término Realidad Aumentada, acuñado por el investigador Tom Claudell. Durante esa

década comienzan a realizarse otros proyectos como el uso de esta tecnología mediante ultrasonidos, pantallas de imagen estabilizada o rastreo de vídeo mediante realidad aumentada. [Julián Dorado & comp., 2011]

Para el desarrollo de un software que haga uso de realidad aumentada es necesario un dispositivo que capte imágenes del mundo en tiempo real, como una cámara web. La idea básica es superponer gráficos a los datos recibidos y mostrarlos, ajustando estos gráficos al movimiento de la cámara y posibilitando al usuario interactuar con ellos. Para conseguir este efecto puede hacerse uso tanto de marcadores de realidad aumentada, que son referencias reales a las posiciones que tomaran los objetos virtualizados, como el de la figura 6. Dentro de los que utilizand marcadores cabe destacar el videojuego Invizimals, desarrollado por la compañía catalana Novarama para PSP en 2009, siendo uno de los primeros videojuegos comerciales de realidad aumentada utilizando tarjetas RA y una cámara conectada al puerto principal de la consola. Por otra parte existe la realidad aumentada sin ningún tipo de marcador, cuyo software es capaz de reconocer formas (tanto orgánicas como inorgánicas) y añadir en ellas estos objetos. Esta última tecnología es más reciente pero evoluciona rápidamente y da lugar a una mejor inmersión del usuario en nuestro sistema, obteniendo así una aceptación y una usabilidad mejores que los sistemas que utilizan los marcadores a cambio de un mayor coste computacional y complejidad del software. [Facultad de Ingeniería Uruguay, 2014]

Figura 6 Ejemplo de realidad aumentada

mediante las tarjetas de Nintendo

Los usos de la realidad aumentada son prácticamente infinitos, desde la simulación en la industria del ocio digital (videojuegos, películas y otros entretenimientos) hasta su aplicación en la medicina o la aviación para mejorar la seguridad de las personas. Las aplicaciones de Realidad Aumentada se pueden desarrollar para todo tipo de dispositivos, desde ordenadores convencionales hasta tablets y smatphones, como en la figura 7, donde podemos ver un smartphone haciendo uso de una aplicación que utiliza realidad aumentada.

Figura 7 Ejemplo de aplicación con realidad aumentada,

tarjetas de visita de la empresa Logistical Outsorcing

Recientemente se han desarrollado otros dispositivos específicos para el uso de realidad aumentada. El Google Glass es un claro ejemplo de este tipo de dispositivos. Este aparato es capaz de mostrar al usuario información disponible para los sistemas Android. El Google Glass cuenta con una cámara de alta definición, capaz de tomar fotografías de hasta 5 MP y grabar vídeo en 720p; un panel táctil situado en una de las patillas para poder desplazarse por los menús con facilidad; acceso wi-fi y bluetooth; un giroscopio y un acelerómetro de tres ejes cada uno: unos sensores fotoeléctricos y de proximidad y un sistema de transmisión de sonido de entrada y salida. El Google Glass cuenta con un servicio específico de realidad aumentada llamado Google Goggles, que permite reconocer objetos mediante fotografías y devolver resultados de búsquedas e información procedente de la base de datos de Google. Este software funciona también con códigos de barras o QR, y es capaz de leer textos hasta en ocho idiomas y traducirlos. Google Goggles está también

disponible para dispositivos móviles. La Figura 8 es una captura de esta aplicación. [Google, 2014]

Figura 8 Ejemplo de uso de la aplicación Google Goggles

analizando la imagen de la Torre Eiffel

3.3. Animación mediante huesos

La animación mediante huesos es una técnica utilizada en computación para animar modelos 3D de manera similar a los movimientos musculares en la vida real, siendo así más intuitiva que la animación tradicional que se realizaba moviendo partes de la malla del modelo por separado o estableciendo relaciones de parentesco entre las distintas partes de la malla y rotándolas para conseguir movimiento. Estas últimas técnicas eran muy tediosas cuando se usaban modelos complejos. Como dice Alan Watt en su libro 'Advanced animation and rendering techniques', la animación mediante articulaciones o huesos se ha popularizado en los últimos años porque permite usar a los seres humanos a modo de actores sintéticos en la animación tridimensional. Si bien los modelos suelen estar caricaturizados, quien realiza las acciones es un ser humano, haciendo más realistas todos los movimientos. [Kristopher Babic, 1999]

Este tipo de animación divide los modelos en dos partes: los huesos (también llamados esqueleto o armadura) y la malla. Los huesos permiten modificar la malla gracias al uso de una armadura formada por polígonos más simples relacionados entre sí que establecen un peso a cada uno de los puntos del modelo. Así al trasladar, rotar o escalar el hueso, los puntos cuya influencia es mayor se transformarán más que el resto, obteniendo así una mayor fluidez de movimiento. En la figura 9 podemos ver un ejemplo de malla a la que se le ha aplicado un esqueleto. Al mover, por ejemplo, uno de los brazos, toda la nube de puntos que lo forma se desplazará con la articulación, cuanto más cerca esté el punto de esta mayor será su movimiento. Así podemos asegurar que la malla no se fragmenta y que el movimiento es suave. [Blender, 2014] [Alan H. Watt, Mark Watt, 1992]

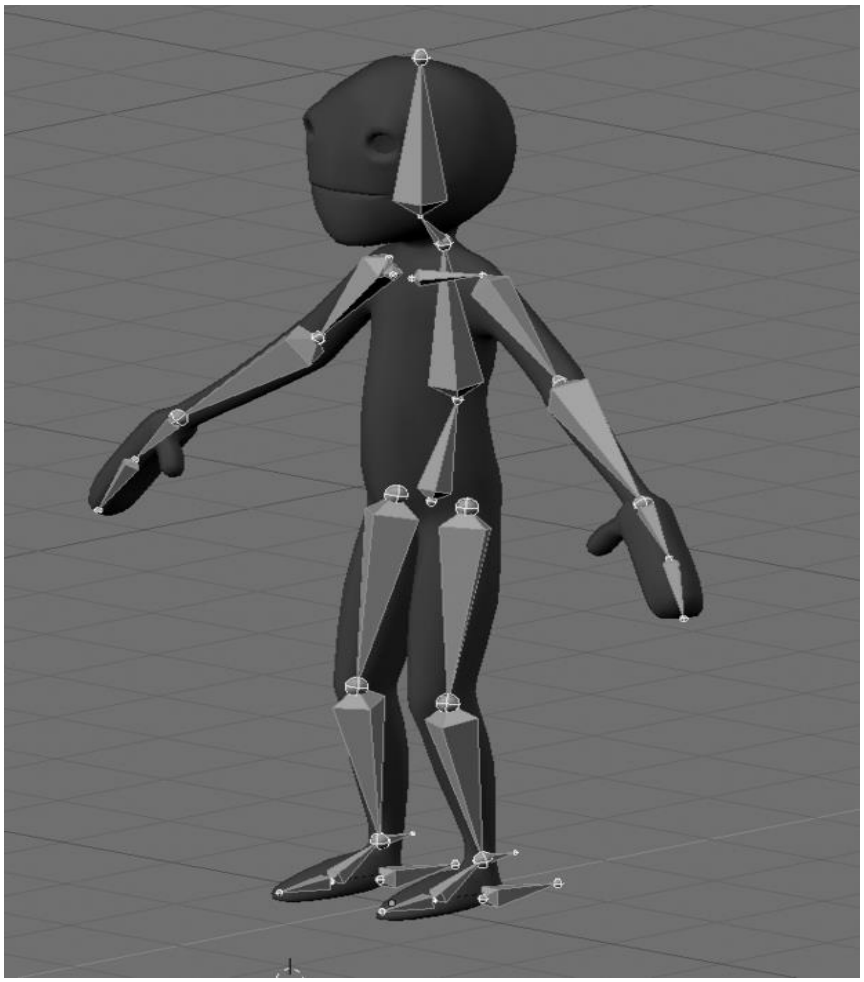

Figura 9 Ejemplo de modelo animado por huesos de la wiki de Blender

Los esqueletos se pueden crear dentro del programa de modelado, ya que casi todos cuentan con un sistema para integrar armaduras. El número máximo de huesos que pueden integrarse por objeto en OpenNI para Unity son 24, desde la cabeza hasta los pies, si bien con menos huesos podemos obtener también buenos resultados siempre y cuando utilicemos un mínimo para dar precisión a las zonas que queremos detectar. La figura 10 muestra todos los puntos que el script de OpenNI de Unity es capaz de detectar. [Nightmarekitty, 2011]

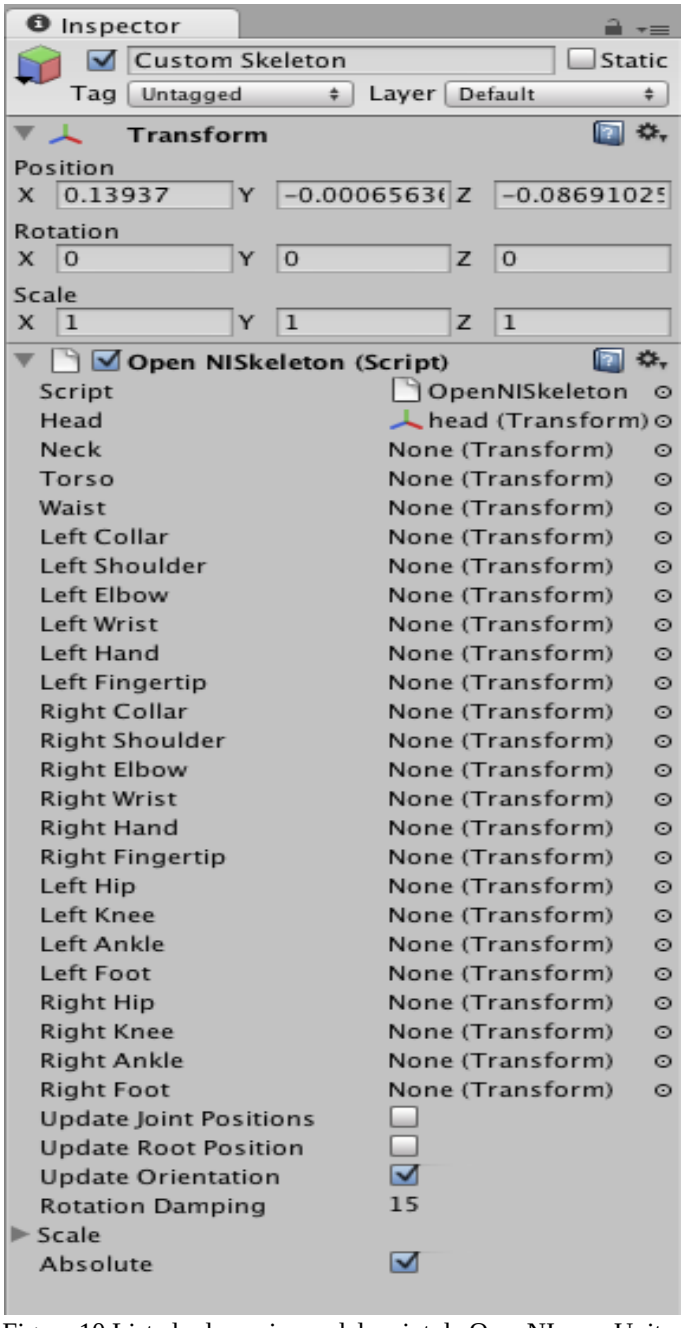

Figura 10 Listado de opciones del script de OpenNI para Unity

La animación por huesos se utiliza comúnmente en videojuegos o en la industria cinematográfica, acompañados de simulaciones mediante ordenador para escenas que podrían resultar demasiado peligrosas para los actores, haciendo uso de software capaz de representar las distintas características del cuerpo humano y la física de manera realista. [Isaac V.Kerlow, 2004]

3.4. Modelado de tejidos

Los tejidos son unos materiales completamente distintos a los demás materiales a nivel físico, puesto que no tienen una forma estática que pueda modelarse con formas geométricas simples, sino que al modelarse habitualmente mediante un sistema de partículas, dependen de la situación en la que se encuentre cada una de ellas, por lo que no podremos crearlos con las técnicas que se usan para el resto de los objetos. Según los autores Elaine Kieran, Gavin Harrison y Luke Openshaw, cuando enfocamos el modelado de tejidos en gráficos por computador debemos considerarlos fundamentalmente sistemas caóticos, debido a que cada vez que son dibujados muchos de sus detalles son distintos. Por eso cuando modelamos estos sistemas lo más importante es que de la sensación de un movimiento correcto en vez de centrarnos en los principios físicos del tejido para crearlos. [Carlos Gonzalez y David Vallejo, 2009]

La mayoría de los tejidos están basados en sistemas de partículas conectadas en forma de malla. Si bien existen otros métodos, como el basado en la geometría y el basado en otras físicas, el uso de partículas, una subcategoría de los modelos físicos, aporta al comportamiento de estos objetos un realismo mayor, si bien es el más complejo de los tres.

Los métodos geométricos fueron la primera técnica utilizada para el modelado de tejidos. Fue creado por Weil en 1986, y se desarrolló hasta la década de los 90. En este método los tejidos son modelados como una cuadrícula bidimensional de puntos tridimendionales conectados de forma recursiva mediante curvas catenarias, un tipo de ecuación matemática que describe la curva formada por una cadena suspendida por los extremos, y los cálculos se realizaban siguiendo dos pasos: primero se eliminan los puntos que cuelguen del interior de la curva y después se asegura que la distancia de las limitaciones

entre partículas se cumple para cada conexión, simulando suavidad. Estos métodos han sido combinados con otros más modernos, pero no son suficientemente precisos para el modelado de tejidos actual.

Los sistemas basados en aproximaciones físicas son aquellos que tratan de simular las reacciones del modelo en un entorno físico. Se dividen en métodos elásticos, métodos de partículas y métodos de masa.

Los métodos físicos elásticos se crean también en 1986 junto con los geométricos, y usan también las cuadrículas bidimensionales de puntos, añadiéndoles la propiedad de la energía, que incluía su tensión, curvatura y gravedad. La distancia entre cada uno de los puntos se calculaba mediante las fuerzas elásticas de los mismos. [Kristopher Babic, 1999]

Los sistemas basados en partículas tienen la peculiaridad de utilizar energía para simular su comportamiento. Así podemos evitar que las partículas intersecten, dando lugar a unos objetos más realistas, podemos darles elasticidad o gravedad o incluso sensación de suavidad a la hora de doblarse. Además se le pueden añadir con facilidad efectos como viento simplemente sumando energía a la ecuación. Los métodos físicos basados en la masa se basan en estos sistemas de partículas, solo que estas están conectadas entre sí mediante amortiguadores de muelle. En la Figura 11 podemos ver un ejemplo de modelado de una prenda utilizando el programa Blender. [Elaine Kieran & comp, 2014] [Wikipedia, 2014]

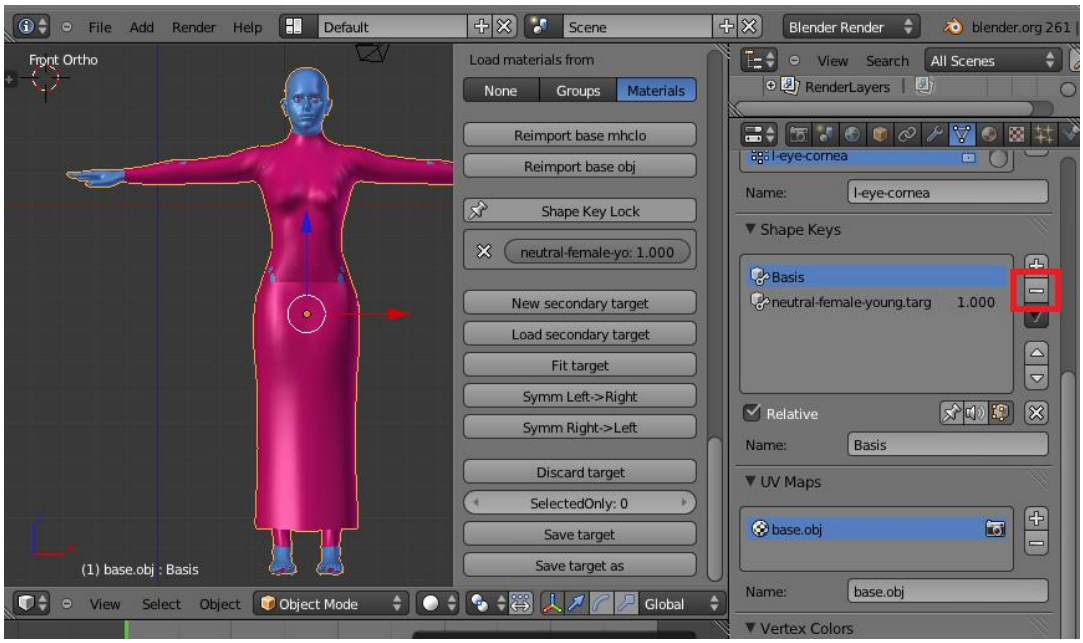

Figura 11 Ejemplo de modelado de tejidos en Blender

4. Tecnologías utilizadas

4.1. Kinect

Kinect (marca registrada por Microsoft) es un dispositivo ideado principalmente como un sistema de juego sin mando para la videoconsola de Microsoft Xbox 360. Está compuesto por unos sensores de profundidad 3D, una cámara RGB, un micrófono multi matriz y un motor que le permite inclinarse, según los datos reportados por Microsoft. [Xbox, 2014]

Microsoft anuncia Kinect en 2009 por primera vez bajo una tecnología creada por PrimeSense. Tuvo un éxito notable entre los jugadores gracias a su control fácil, intuitivo e innovador. Pero el verdadero éxito surgió cuando Kinect, se convierte en la plataforma más buscada para la creación de nuevos inventos de software. Así Microsoft liberó el SDK para uso no comercial. Esto permite a los desarrolladores crear todo tipo de aplicaciones y juegos. En la figura 12 podemos ver un esquema del hardware que compone Kinect. [Bob Widenhofer, 2010]

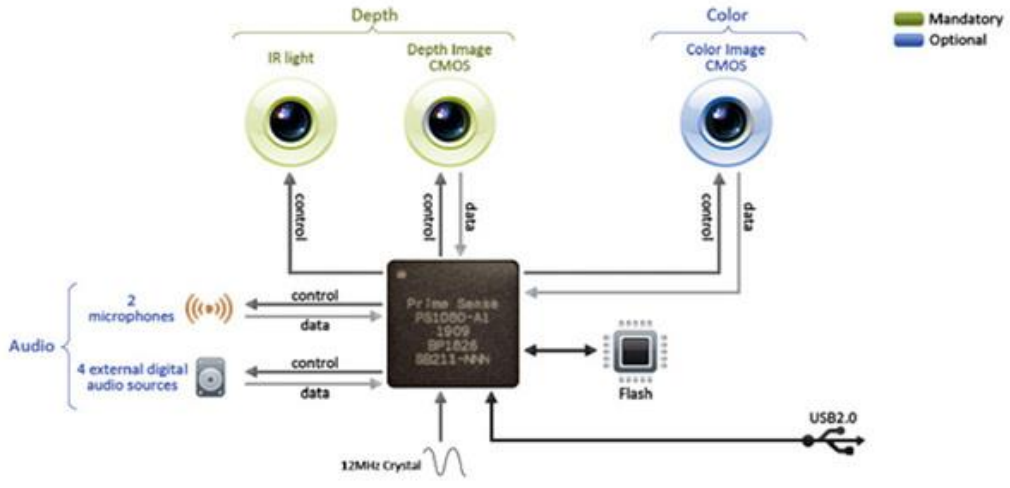

Figura 12 Esquema del hardware de Kinect por iFixit.com

Kinect cuenta también con varios SDKs de terceros, como OpenNI o Kinect4Fun, desarrollados para facilitar su uso en tareas o lenguajes concretos y permitiendo además utilizar Kinect en sistemas operativos no desarrollados por Microsoft, como OS X o Linux.

La realidad aumentada también puede hacer uso de Kinect, valiéndose de sus sensores y sus cámaras para mejorar la captación de profundidad y simplificar así el uso de aplicaciones sin marcadores. Dentro de los productos creados podemos encontrar desde técnicas de marketing como la que usó Bully! para la promoción de un automóvil en el que el usuario podía interactuar con el vehículo antes del lanzamiento, hasta el uso de Kinect para técnicas de reconocimiento facial, como la que presentaba en Tenerife Lan Party un grupo de desarrolladores. Esta permitía aumentar la eficacia y el automatismo de la gestión personal de las empresas.

4.2. Unity y Zigfu

Unity es una plataforma de desarrollo de videojuegos multiplataforma creada por la compañía Unity Technologies. El software creado con Unity es compilable para Windows, OS X, Linux, Xbox360, Play Station 3, Play Station Vita, Wii, WiiU, iPad, iPhone, Android, Windows Phone e incluso navegadores web compatibles con el Plug-In Web de Unity, convirtiendo esta plataforma en una manera de desarrollar productos para un amplio número de usuarios. Es un software propietario, si bien existe una versión gratuita con menos funcionalidades.

Unity cuenta con una gran comunidad de usuarios y un amplio número de foros y tutoriales accesibles desde su página web con los que su curva de aprendizaje se vuelve relativamente sencilla. Unity soporta además modelos de Blender, un software de modelado 3D gratuito, entre otras muchas compatibilidades. También cuenta con una tienda online de plug-ins y addons integrada en el mismo programa además de soporte de muchos grupos independientes que crean sus propios scripts, como Zigfu, que permite a Unity interactuar con Kinect. Por último, este programa es compatible con scripts en lenguaje Javascript y C#, flexibilizando su uso y aprendizaje. [Unity3D, 2014] [Wikipedia, 2014]

Zigfu está compuesto por una colección de scripts que permiten desarrollar apps utilizando Kinect en Unity. Está basado en OpenNI, una librería multiplataforma y libre de Kinect, la cual viene integrada con el instalador. Si bien la comunidad de Zigfu es muy pequeña y no existen casi tutoriales para la librería, esta cuenta con un moderado número de ejemplos integrados que pueden ser útiles para aprender utilizando ingeniería inversa. A pesar de esto, una vez aprendido aporta gran sencillez al desarrollo de software con Kinect en Unity, como podemos ver en el ejemplo de la figura 13. [Zigfu, 2014] [Nightmarekitty, 2011]

Figura 13 Ejemplo de realidad aumentada utilizando Unity 3D

5. Solución propuesta

5.1. Diseño del sistema

El sistema cuenta con varios módulos que trabajan generalmente en paralelo, tanto en el cálculo de físicas y actualización de los datos recibidos por Kinect como las operaciones internas de cambio y posicionamiento de modelo y huesos. Unity permite el cálculo automático de todos sus componentes internos mediante sus propios scripts, con lo cual a la hora de realizar el diseño del sistema podemos centrar nuestra atención en el desarrollo de las funcionalidades específicas. El diseño del sistema, por tanto, seguiría el siguiente patrón:

- Primero Kinect, mediante su sistema de sensores y cámaras, detectará al usuario si lo hay y cada uno de los puntos dentro de su esqueleto que se encuentren visibles.
- Estos datos captados por Kinect son recogidos por el script de OpenNI de Unity en nuestro programa.
- Una vez obtenemos todos los datos, el script actualizará la posición de cada una de las partes del esqueleto asignadas a cada punto. Estos huesos, al desplazarse, tirarán del bounding box que tienen asignado.
- Al mover el colisonador al que el modelo de la prenda va unido esta será arrastrada a la nueva posición del bounding box. A su vez se calculan las colisiones con el resto de boundings de los brazos y las piernas.
- Una vez colocado el modelo de la ropa en la posición y calculadas las colisiones, pasan a calcularse las físicas equivalentes a las mismas. Una vez hecho esto el modelo se deforma según las fuerzas aplicadas y las propiedades de este, como su grosor o respuesta al choque con los boundings.

- Se calcula el ciclo de update de la GUI (es decir, la interfaz del programa). En este momento se actualiza la fotografía mostrada y se calcula la iteración del usuario con los distintos botones de la aplicación, llamando a los métodos de estos.
- Se hace el ciclo de update de cada uno de los scripts. Si detecta un usuario y no había ninguno se crea una nueva prenda.
- Por último se actualiza la imagen de la textura que muestra la visualización obtenida por la cámara del Kinect.

En la figura 14 podemos ver un diagrama del funcionamiento del sistema con lo que hemos explicado.

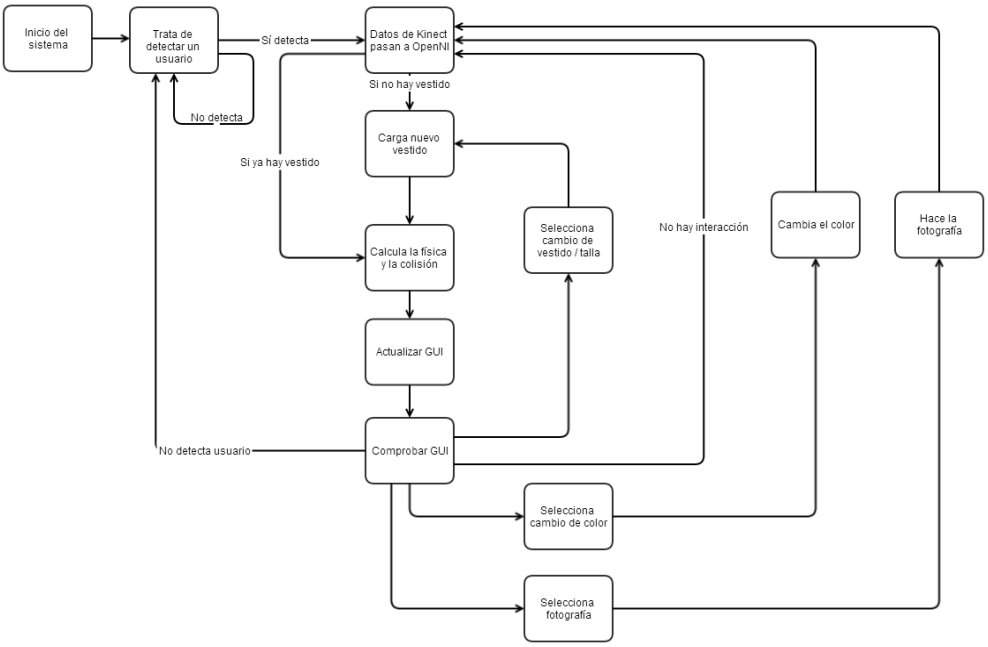

Figura 14 Diagrama del funcionamiento del sistema

5.2. Funcionamiento del sistema

El sistema que vamos a desarrollar debe contar, al menos, con todos los puntos propuestos en el apartado de objetivos. Además, dado que es un proyecto que está enfocado al público, será necesario crear un sistema perfectamente usable.

Para entender cómo funciona el programa es necesario introducir un poco de terminología básica de Unity. Para empezar, Unity trabaja con una serie de

objetos llamados GameObjects, que son los más importantes en el desarrollo con esta plataforma. Todo objeto en Unity es un GameObject, si bien los GameObjects no pueden realizar acciones por su cuenta, sino que se valen de distintos componentes asociados para realizar las tareas, como scripts, colisionadores, módulos de físicas, etc. Por otro lado tenemos los prefabs. Los prefabs son un tipo de objeto especial de Unity que permiten guardar un GameObject con una serie de propiedades y componentes para su fácil instanciación.

5.2.1. Componentes del sistema

Los elementos geométricos principales del sistema son un vestido y un modelo invisible compuesto por una serie de huesos y bounding boxes o colisionadores. Los huesos están ligados a las articulaciones del usuario real que Kinect detecta y se actualizan en cada iteración. Los huesos al moverse tiran de los colisionadores y estos a su vez tiran del modelo de la prenda, que gracias a la simulación de físicas de ropa simulan el comportamiento de los tejidos, lo que le da sensación de realismo.

Los boundings son generalmente figuras simples como ovoides, que mejoran el rendimiento del sistema porque la colisión se realiza con menos polígonos. Tan solo uno de los boundings es una figura compleja, un busto compuesto por una cabeza y un torso. A este bounding complejo va ligado el modelo de la prenda, lo cual hace que su posición sea mucho más realista.

El sistema cuenta también con una GUI, es decir, una interfaz donde se encuentran los botones y las opciones, con la que puede interactuar el usuario para utilizar las distintas funcionalidades del sistema.

5.2.2. Enfoque del desarrollo

Los problemas a los que nos enfrentamos son desarrollar un software capaz de detectar cada una de las articulaciones del cuerpo de un usuario y colocar mediante realidad aumentada una prenda en este. Dicha prenda interactúa con el usuario a nivel de físicas, es decir, al moverse el usuario la prenda es empujada por una serie de fuerzas físicas simulando el movimiento de la ropa real. El usuario puede usar una serie de herramientas implementadas en la aplicación que le permiten modificar algunas características de la prenda, siendo estas el cambio de talla entre mediana, grande y extra grande (dado que el programa está planteado para usuarios adultos); el cambio de color de la prenda de entre una serie de posibilidades disponibles; el cambio del modelo de la misma prenda; o la posibilidad de sacar una fotografía del usuario con el modelo puesto, que se exhibe en la esquina inferior izquierda de la pantalla. Todas estas funcionalidades son accesibles desde unos menús situados en la aplicación principal, siendo posible aplicarlas mediante un solo click, lo que hace el programa más usable.

Este programa está enfocado a ser un comienzo para un sistema comercial, por tanto el que sea fácil de manipular a la hora de agregar nuevas prendas es una característica muy importante para el sistema. El cliente no debe poder acceder al código para introducir nuevas prendas, pues esto sería inviable. Por eso se ha ideado un sistema que permite agregar prendas desde archivos externos y poder introducirlas en el programa automáticamente. Para ello existe un documento de texto que leerá los nombres de las prendas que se exhibirán, que puede ser modificado por el usuario. Además encontramos una carpeta dentro de los recursos del programa en la que metemos una serie de prefabs de Unity3D. En este caso los prefabs que se construyen son las prendas que se mostrarán, las cuales referenciaremos en el documento de texto. Así el programa, automáticamente elegirá de entre las opciones referenciadas en el documento en el orden del mismo para cargar ese modelo. Los prefabs deben estar disponibles en las tallas que desean mostrarse, es decir, si queremos la talla M, L y XL para una prenda debemos tener un prefab para

cada una. Para nombrar cada talla es necesario poner al final del nombre del atuendo “_m”, “_l” o “_x”. Así, el nombre del archivo para el vestido corto mediano será “Vestido_corto_m.prefab”.

5.2.3. Interacción con el sistema

El sistema consta de un plano que hace las veces de espejo. Esto se consigue aplicándole como textura la imagen captada por el Kinect. El dispositivo captará imágenes en cada iteración y cuando detecte la presencia de un usuario, cargará una de las prendas, usando siempre por defecto la talla L. Si bien Kinect permite detectar hasta dos personas al mismo tiempo, el sistema sólo utilizará para sus cálculos a un usuario cada vez, en concreto el primero que se sitúe delante del aparato. Al detectar un nuevo usuario, el modelo cambiará desde el último mostrado para dar variedad, si bien podemos acceder a todos y cada uno de ellos desde el botón de cambio de prenda. Cuando el sistema no detecte a ningún usuario, la prenda que esté en ese momento se destruirá y pasaremos a ver solamente la imagen captada por el Kinect. Esto permite realizar menos cálculos y así poder detectar con más rapidez al siguiente usuario, además también aporta un efecto de sorpresa ante él, y es estéticamente más vistoso que dejar el modelo suspendido en la pantalla.

Al cambiar o aparecer un vestido se reiniciará su posición completa junto con la del esqueleto. Esto pasa durante un solo frame, pero volverá a actualizarse acto seguido de nuevo con la posición del usuario. Esto es apreciable pero necesario para que el modelo nuevo pueda moverse sin problema y estar perfectamente ligado a los huesos.

Para realizar una captura del usuario, éste tiene que pulsar el botón de fotografiar. El sistema esperará tres segundos y acto seguido hará la captura de la pantalla y la guardará como una imagen en formato JPG, bajo el nombre de “screenshot.jpg”, solapando la última fotografía tomada. En el GUI (el menú de control) se encuentra, en la esquina inferior izquierda, una textura bidimensional que se recarga con la nueva imagen.

Para la funcionalidad del cambio de color debemos pulsar en una imagen situada en la esquina superior derecha de la pantalla. En función del píxel donde clickemos la prenda actual se teñirá del color del punto pulsado. Esta imagen cuenta con ocho colores: rojo, magenta, azul, cyan, amarillo, verde, blanco y negro. Al cambiar de modelo este volverá a mostrarse con el color por defecto, el blanco.

5.3. Modelado

El trabajo de modelado realizado para este sistema se podría dividir en dos partes: primero se ha modelado un maniquí antropomórfico que será el esqueleto de nuestra aplicación, y después se han modelado cada una de las prendas que se mostrarán en el programa.

Para el modelado de cada una de las prendas creadas para el sistema se ha utilizado el programa Blender, principalmente por su sencillez a la hora de diseñar en 3D y de importar estos modelos a Unity.

5.3.1. Modelado del esqueleto

Los modelos de las prendas se han desarrollado a partir del modelo antropomórfico utilizado para las colisiones, por tanto será imprescindible explicar cómo se realizó este maniquí para la comprensión del modelado de la ropa, que explicaremos después.

Para modelar la figura humana haremos uso de una función de Blender que nos permite colocar una imagen como fondo mientras trabajamos. Dibujamos un modelo con alzado y perfil y lo colocamos en el fondo del entorno de trabajo. Colocamos un cubo en la posición donde debería estar el punto central del modelo. Extruímos el cubo hacia arriba hasta que tenga la misma altura que el boceto y lo subdividimos. Ahora volvemos a extruir unas extremidades a la altura de los brazos, las piernas y la cabeza. Dividimos un brazo tres veces para obtener la posición del codo y la mano

y hacemos lo mismo con la pierna. Es importante que estos brazos estén en forma de T para poner la armadura correctamente.

Ahora cortamos la mitad del modelo resultante y aplicamos el modificador de espejo en el eje Z dentro de Blender, que representa el eje Y en condenadas cartesianas. Así cada vez que modificamos una cosa en una parte del modelo conseguiremos un resultado simétrico.

Poco a poco vamos subdividiendo cada una de las partes del modelo para que coincidan con los puntos clave del modelo. Estos puntos clave los marcamos nosotros, el objetivo principal es que nuestro modelo se parezca lo máximo posible al boceto del fondo. Para conseguir este resultado debemos utilizar extrusiones, subdivisiones y movimiento, pero no rotaciones, dado que modificar la orientación de ciertos puntos puede ser problemático a la hora de colocar el esqueleto, que es el paso siguiente una vez consigamos un modelo anatómicamente humano. No tiene por que ser demasiado realista dado que no se va a ver, pero si es importante definir bien el torso y los brazos puesto que a partir de esta figura crearemos las prendas.

Ahora pasamos a colocar el esqueleto. Blender nos ofrece un esqueleto antropomórfico por defecto que podemos utilizar. Otra opción es crear huesos separados y unirlos manualmente, pero dado que el programa aporta una armadura similar a lo que necesitamos, optaremos por la primera opción. Para poder agregar el esqueleto humano primero debemos habilitar el add-on que lo contiene accediendo a preferencias. Una vez creada la armadura tenemos que unirla al modelo creado mediante rigging o aparejamiento.

Debemos modificar el tamaño de los huesos antes para que se adapten mejor al tamaño del modelo. Esto es importante porque al mover la armadura, si no está bien escalado el esqueleto los huesos harán movimientos extraños y se deformarán, restando realismo al programa y dejándolo inutilizable. También es importante colocar el esqueleto de tal manera que el punto central del mismo coincida con el del modelo por la misma razón.

Cuando tengamos el esqueleto colocado dentro del modelo, debemos realizar algunas pruebas para verificar su funcionamiento y sus movimientos, y como hemos dicho antes, para evitar que aparezcan deformaciones extrañas. Una vez comprobemos que todo esté en orden podemos proceder con el modelado de los vestidos. En la figura 15 podemos ver el resultado del modelo anatómico.

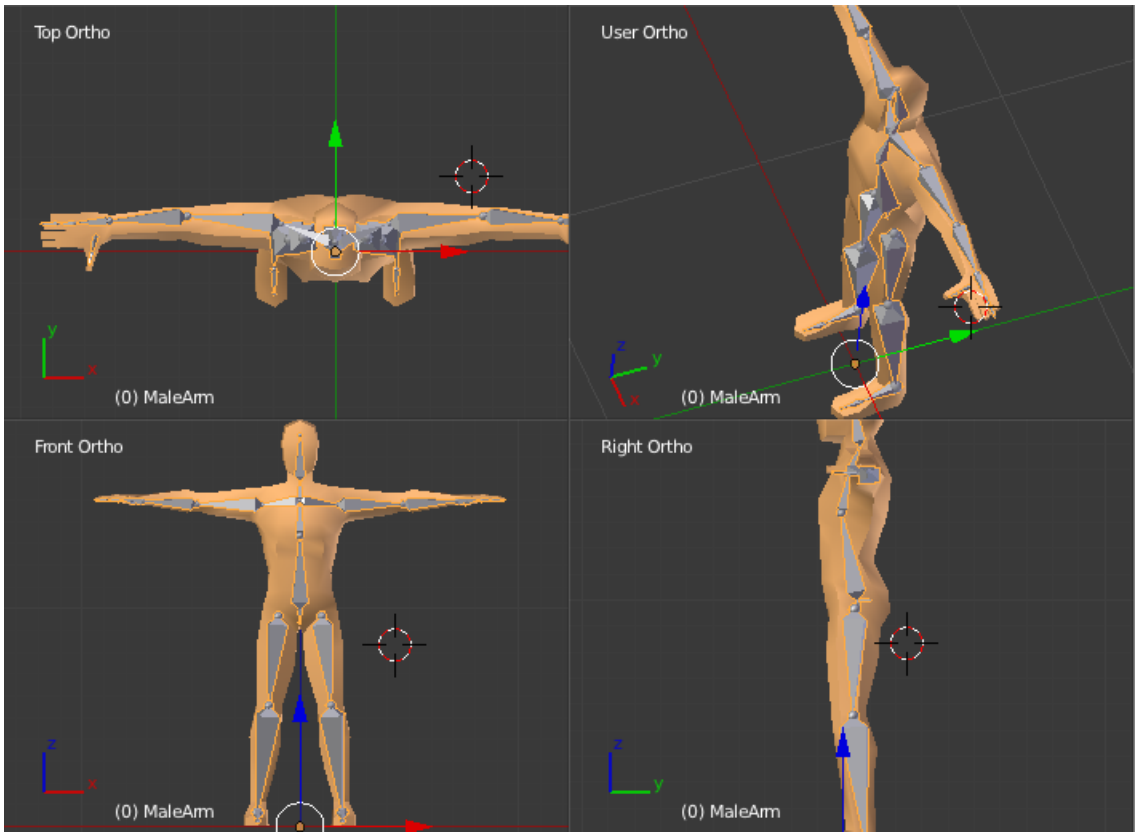

Figura 15 Modelo del esqueleto completo

5.3.2. Modelado de los vestidos

Para nuestro sistema hemos desarrollado tres vestidos, uno corto, uno largo y otro palabra de honor. La razón de haber modelado vestidos en lugar de otra clase de prendas es que visualmente son más vistosos a la hora de mostrar las físicas porque la falda se moverá a la par que el maniquí, otorgando realismo a la escena.

Para modelar los vestidos partiremos del modelo creado anteriormente. Primero debemos eliminar la parte de las piernas del humaniode

seleccionando todos los polígonos que las forman hasta el torso y eliminándolos. Es importante eliminar los polígonos, no los vértices o las aristas, porque si no se borrarán más partes de las necesarias. Después seleccionamos los vértices de la parte inferior y los extruímos hacia abajo hasta el punto 0 del eje Z para formar la falda. Ensanchamos la parte de debajo de la falda recién extruída para que tenga forma cónica. Después subdividimos esta parte varias veces con la herramienta Loop Cut and Slide para formar los pliegues.

Ahora cortamos parte de los brazos hasta que formemos unas pequeñas hombreras para el vestido. Para ellos eliminamos los polígonos de los brazos del modelo dejando solamente la parte del hombro, que triangularemos para dar un efecto más realista y estético. Por último eliminamos los polígonos de la cabeza y formamos un cuello en forma ovalada. Así terminamos el vestido largo. En la figura 16 vemos el resultado.

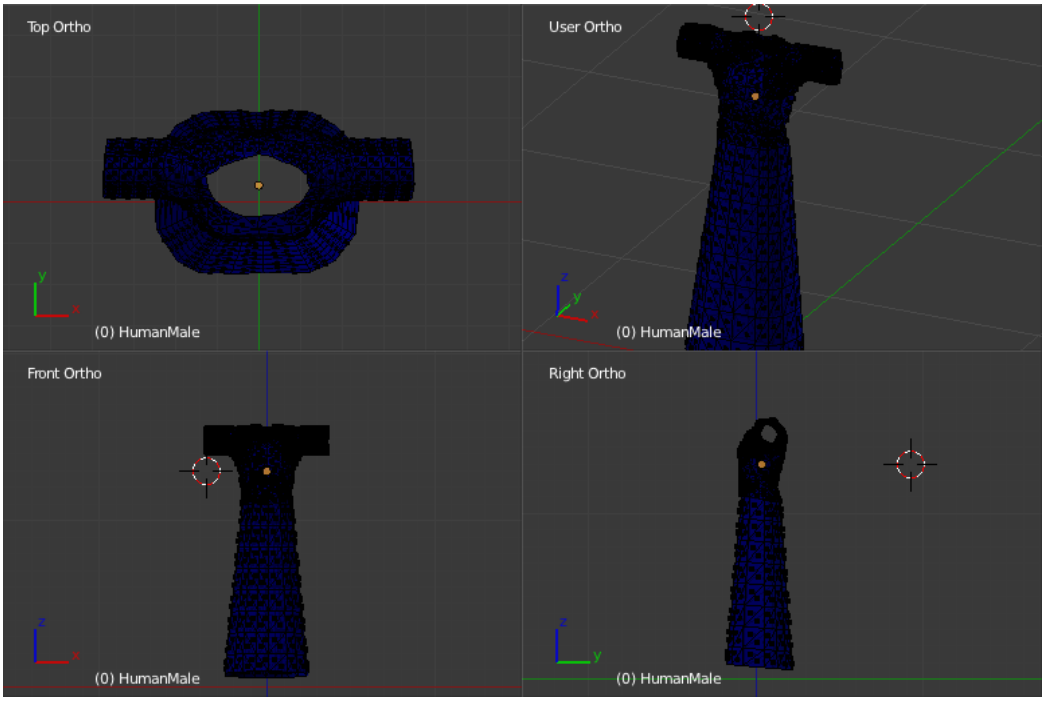

Figura 16 Modelo del vestido largo en el modo alámbrico de Blender

El vestido palabra de honor es sencillo de desarrollar una vez tenemos el vestido largo. Solamente tenemos que seleccionar todos los polígonos

desde el cuello hasta la parte alta del pecho, incluyendo los brazos, y eliminarlos para crear un escote palabra de honor. El vestido debe quedar como el de la figura 17.

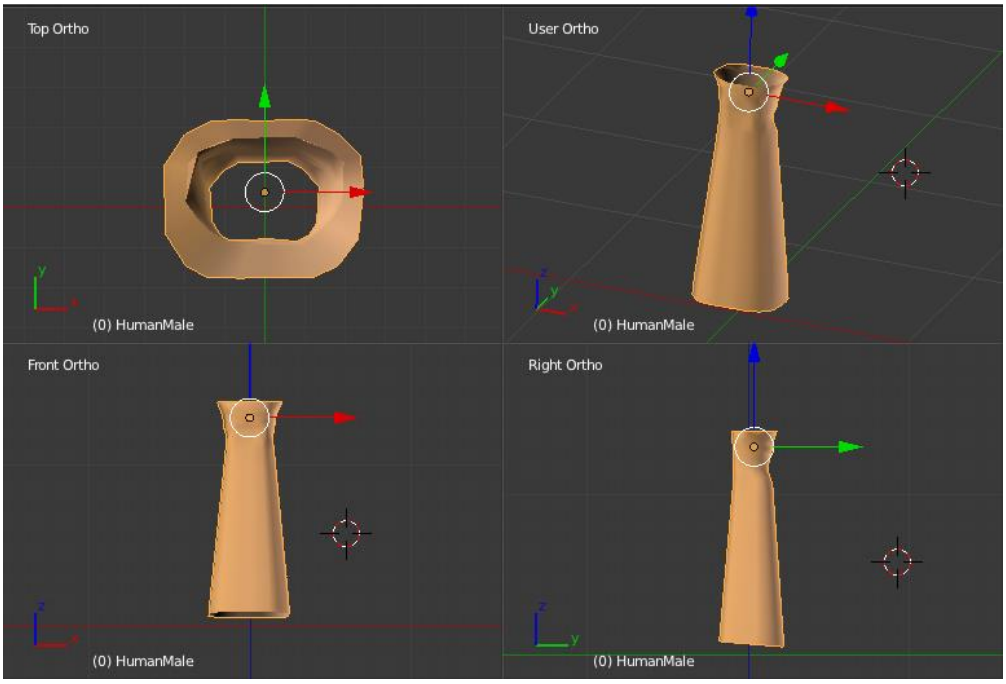

Figura 17 Modelo del vestido palabra de honor en el modo objeto de Blender

Para desarrollar el último vestido, el vestido corto, vamos a partir otra vez del modelo anatómico. Su desarrollo es bastante similar al del vestido largo. Tras eliminar otra vez las piernas, seleccionamos los vértices de la parte inferior del torso y los extruímos hacia abajo. Ensanchamos mediante escalado el resultado, esta vez un poco más amplio para dar una forma de falda corta en vez de la de un vestido. Seleccionamos los vértices del brazo incluyendo el hombro y los eliminamos para formar unos tirantes. Por último eliminamos la cabeza y desplazamos los vértices de la parte superior del pecho para formar un cuello con forma picuda tal y como lo vemos en la figura 18.

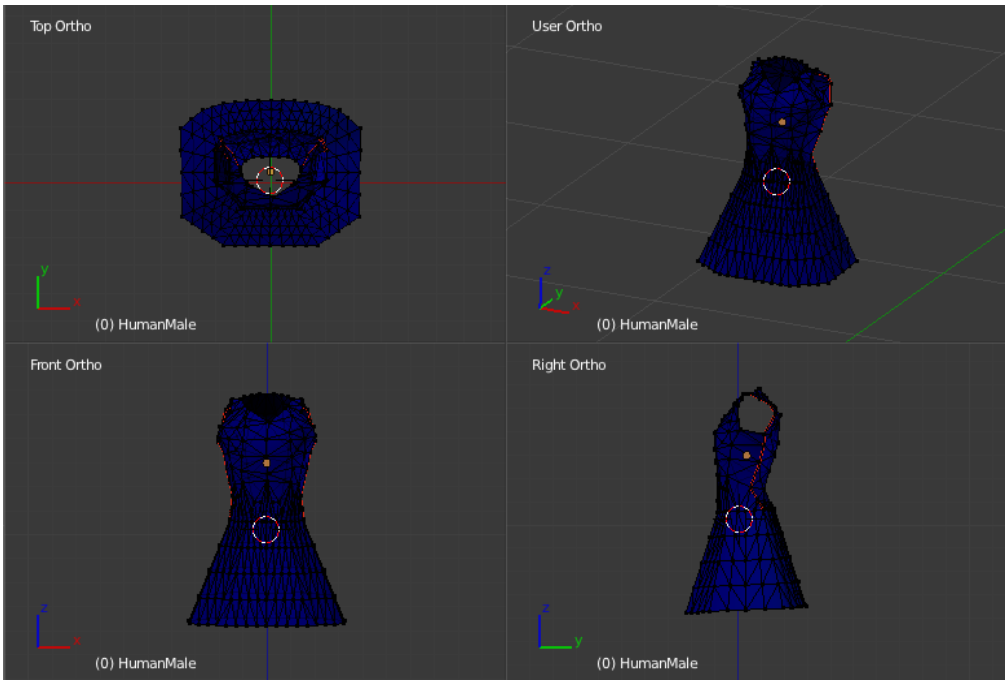

Figura 18 Modelo del vestido corto en el modo alámbrico de Blender

Ninguno de los modelos de los vestidos debe tener huesos, así que es necesario eliminar la armadura del modelo anatómico antes de crearlos. Existe la posibilidad de crear los vestidos con esqueleto y guardarlos como un skinned mesh, creando en Unity un skinned cloth a partir de esto. El principal problema es que este tipo de objetos aún no están del todo depurados, y además Blender no permite crear skinned meshes propiamente (si bien es capaz de emularlas de alguna manera, pero no siempre son capaces de funcionar en Unity). De querer crear esta clase de objetos deberíamos utilizar un programa de pago como Maya 3D.

Es importante guardar todos los archivos por separado aun estando basados en un mismo modelo. También es importante guardarlos en modo objeto a pesar de trabajar en modo alámbrico, porque Unity no es capaz de convertir los archivos .blend a .bfx, el formato de trabajo del programa. Exportarlos a este formato desde Blender tampoco es una buena solución, ya que la escala cambia y es necesario realizar ajustes manuales, con lo que restamos automatismo al sistema.

5.4. Desarrollo con Zigfu

Para desarrollar en Zigfu lo primero es realizar la instalación del paquete, que consta de las librerías de OpenNI y los scripts necesarios para el desarrollo con Kinect en la plataforma Unity. Para comenzar la instalación de Zigfu es necesario tener instalados previamente los controladores de Kinect, que podemos encontrar en la página web de Microsoft, aunque hemos de tener cuidado porque si instalamos el kit de desarrollo (sdk) original de Kinect, OpenNI puede dar errores de instalación y dejar de funcionar, ya que estas librerías entran en conflicto con las originales. De tenerlas instaladas, por tanto, procedemos a eliminarlas del equipo antes de comenzar.

Encontramos el pack de desarrollo de Zigfu en su página web oficial, dentro del apartado de descargas. La última versión cuenta con un instalador compacto que aporta sencillez al problema. Descargamos la versión para Windows, que es un archivo .unitypackage. Una vez tengamos el archivo en nuestro disco duro, abrimos Unity, creamos un nuevo proyecto 3D y vamos a Assets->Import Package, seleccionamos la ruta del archivo y procedemos a instalarlo. Ya contamos con todo lo necesario para empezar a desarrollar el proyecto. Antes de comenzar con nuestro sistema, podemos probar algunos de los ejemplos creados por Zigfu. Esto puede ser de gran ayuda debido a que, dada la falta de documentación de la librería, muchas veces será necesario utilizar ingeniería inversa para el desarrollo de ciertos apartados de nuestro programa.

Creamos una nueva escena en Unity. Necesitamos crear para empezar dos objetos muy importantes: la cámara, que será lo que nos muestre el contenido de nuestra escena en la pantalla; y un game object vacío llamado Zigfu, el cual contendrá todos los scripts necesarios para trabajar con la librería. Podemos crear ambos accediendo a la opción de la barra de menú GameObject->Create Camera/Create Empty. Renombramos ambos objetos como Main Camera y Zigfu, respectivamente. La posición que le otorguemos al objeto vacío es irrelevante, si bien de la de la cámara hablaremos más tarde.

Una vez tengamos el objeto Zigfu creado debemos ir a la carpeta del proyecto donde se encuentren los scripts de OpenNI. Arrastramos al objeto Zigfu los siguientes scripts: Zig.cs, ZigEngageSingleUser.cs, ZigDepthViewer.cs y

ZigUserRadar.cs. El primero aporta las funcionalidades básicas de OpenNI, como las librerías de Kinect y su reconocimiento y el segundo habilita la posibilidad de obtener los datos del esqueleto un usuario. El tercero y el cuarto los utilizaremos para testear, dado que nos aportan la imagen de profundidad del Kinect y el radar de usuarios que está detectando automáticamente. Estos dos los eliminaremos al acabar el programa, por tanto a pesar de que su situación en la pantalla es fija (ambos se sitúan en las esquinas inferior y superior izquierda respectivamente) y pueden molestar en el uso de otras funcionalidades que desarrollaremos más tarde, no será un problema.

Dentro de los scripts del objeto Zigfu debemos habilitar una serie de opciones. En el script Zig deben estar marcadas Update Depth, Update Image y Update Label Map, Mirror y Track Skeleton. Las opciones de Update se dedicarán a recargar la entrada del Kinect, la imagen de profundidad (que nos permite conocer la situación de las figuras en un entorno tridimensional), la imagen real captada por la cámara y el mapa de etiquetas, que se encarga de reconocer cada uno de los huesos, que será necesaria para que funcione la opción Track Skeleton. La opción Mirror es necesaria porque al marcarla la imagen se nos mostrará volteada en el eje Y, y dado que lo que queremos dibujar debe tener forma de espejo, será necesario aplicar este modificador. El tipo de entrada debemos señalarlo como automático, lo que permite usar también la cámara Asus Xtion Pro Live además de Kinect. En el script Zig Engage Single User debemos marcar la opción Skeleton Tracked y aumentar el número de usuarios asociados a uno. Ahora se nos pedirá añadir un elemento, un esqueleto. Dado que no tenemos ninguno, esto lo completaremos más tarde. Los dos últimos scripts no deben ser modificados, sino que utilizaremos los parámetros que traen por defecto.

Para hacer uso de la cámara, primero vamos a posicionarla. Para que esté centrada se ha colocado en [0, 1, -1]. La proyección de la cámara debe ser perspectiva en vez de ortogonal, porque el resultado mostrado será más realista que ortogonal, evitará que necesitemos escalar el modelo cuando nos alejemos. La vista la colocaremos entre 0.3 hasta 1000, es decir, lo más cercano a la cámara que se mostrará estará a 0.3 unidades de medida y lo más lejano a 1000. La lejanía no es un problema porque en ningún momento tendremos el modelo

tan lejano. La cercanía si es importante porque de ser menor podría dar errores al tomar los datos el Kinect. Por último vamos a añadir un script vacío en el que escribiremos la parte del programa relativa al GUI en otro apartado.

Ahora vamos a crear el espejo. Insertamos un nuevo game object, un plano. Lo colocamos en [0, 1, 10] y lo escalamos para que el tamaño de los lados tengan una relación de 4/3. También debemos rotarlo 90º en X y 180º en Y para que al mostrar la textura se muestre correctamente. A este plano le añadiremos también el script ZigImageViewer.cs, que colocará automáticamente la textura de la cámara de Kinect. El tamaño de la textura lo tendremos en VGA 640x480, la mejor resolución disponible.

Vamos a añadir también una luz direccional blanca en la parte superior de la cámara y apuntando al plano. Esto iluminará la escena y mostrará algunas sombras en los modelos de la ropa para darles realismo.

Para crear el modelo anatómico incluimos el archivo de Blender en el proyecto y lo arrastramos a la escena. Este modelo cuenta con una animación y varios hijos que sólo cuentan con una posición. Estos son los huesos. Seleccionamos el modelo de Blender y comprobamos que el rigging, es decir, la unión del esqueleto con el modelo, está bien hecho, seleccionando la animación de tipo humanoide. A este objeto le añadimos el script de Zig Skeleton, que es el encargado de obtener los puntos del esqueleto que hemos creado y asociarlo con los puntos del usuario real captados por Kinect, y colocamos cada una de las partes del modelo, de los huesos, en cada uno de los puntos del script. También debemos seleccionar la opción de Mirror por la misma razón que lo hemos hecho en el script de Zig. También seleccionamos la opción de actualización de la raíz del modelo y su orientación para que coincida con el usuario.

Debemos colocar unos colisionadores de cápsula en ciertos puntos del modelo: en ambos brazos y antebrazos. Utilizamos cápsulas porque mejoran mucho el rendimiento en comparación con las que se ajustan a las mallas debido a que cuentan con muchos menos polígonos y una ecuación mucho más simple que la de las mallas complejas, con lo cual calcular la posición requiere muchas menos operaciones, haciendo el sistema más ligero en el cálculo de físicas. También

debemos añadir un colisionador más complejo en el torso. Podríamos utilizar también bounding boxes con formas simples pero dado que a este van a ir sujetos los modelos de la ropa es mejor utilizar uno que se adapte mejor al cuerpo, porque dará una sensación más realista del contorno del usuario. Para crear este bounding debemos abrir Blender y crear, a partir del modelo anatómico, otro sin brazos ni piernas, es decir, sólo el torso. Lo importamos al proyecto y colocamos en el torso del modelo un bounding box con forma de esta malla, un mesh collider.

Por último deshabilitamos el mesh renderer para no mostrar el modelo aunque si se actualicen las posiciones. Este esqueleto se lo colocamos al script Zig Engage Single User que asignamos anteriormente al GameObject llamado Zigfu. Una vez acabado esto, y antes de deshabilitar el modelo, debemos compilar el programa y comprobar que el modelo se mueve y capta los movimientos del usuario.

Para desarrollar cada uno de los prefabs de la ropa tenemos que crear un objeto vacío para cada uno. Lo colocamos en [0, 1.3, 10]. La posición de la X y la Y es por la situación del punto centro del modelo. Que la posición Z coincida con la del plano no es casualidad. Es debido a que, como la cámara está en perspectiva, cuando la posición del modelo se actualiza se encontrará en el mismo punto en la imagen que el usuario, y al coincidir dará la sensación de realismo que buscamos. Le añadimos un componente de Cloth Renderer e Interactive Cloth. Al cloth renderer no hay que hacerle ninuna modificación. Este sirve para mostrar por pantalla la malla de la prenda.

En el componente de interactive cloth hay que dejar seleccionadas las opciones de Gravity y Self Collision. Es importante dejar el valor Tear Factor a 0 porque si no la prenda se romperá al contacto con los colisionadores. También colocar el valor Attached Collider a 1, aunque el valor del elemento al que irá fijado lo seleccionaremos más tarde mediante código. También debemos seleccionar una malla para cada tipo de ropa, y guardar un prefab para cada uno. Por último cambiamos la escala de cada prenda para tener una talla para cada una.

Para desarrollar la GUI debemos utilizar el script que creamos en la cámara. Al iniciarse la aplicación debemos declarar todas las variables que vayamos a

utilizar y leer el fichero que utilizaremos para ver qué prendas existen, guardando los nombres que encontremos en un vector de strings, así como el número de prendas que tenemos. En la GUI vamos a crear una serie de botones que harán varias funciones: una pareja en la esquina superior izquierda que servirán para cambiar el modelo de la prenda y para sacar una captura de pantalla. Esta captura la mostraremos en la esquina inferior izquierda, una GUISkin que se actualizará en cada iteración con la fotografía más actual. También pondremos otro GUISkin en la esquina superior derecha con una textura con colores, en la cual hay un módulo que detecta la posición del ratón y el color del píxel en el que está situado, y al seleccionar cambiará el color del modelo de la prenda esté actualmente en pantalla. Por último colocamos una serie de botones en la parte izquierda para poder cambiar la talla de los modelos. Esto se consigue destruyendo el modelo, reiniciando la posición del usuario y cargando el prefab de la prenda actual con la talla seleccionada. Este método también lo utilizamos para cambiar el modelo, destruyendo la prenda y cambiando el prefab actual a la prenda siguiente. Hay que guardar tanto la talla como el modelo que se está utilizando para poder utilizar estos módulos correctamente. Este último módulo lo utilizaremos también cuando un nuevo usuario sea detectado, siempre y cuando no hubiese otro ya. El programa detecta que la variable del script de Zig Engage Single User que determina si existe un usuario cambia y actualiza una variable que, al actualizarse el sistema, destruirá o cargará un modelo. Por último, para hacer la fotografía al pulsar el botón, un contador comenzará a contar. Cuando este llegue a tres segundos tomará una captura de pantalla, que se guardará en la carpeta del proyecto. Esta será la misma que actualice la pantalla. Podemos ver el sistema terminado en la figura 19.

Figura 19 Nuestro sistema en funcionamiento.

6. Pruebas

Para el correcto funcionamiento del sistema es necesario desarrollar una serie de pruebas que nos permitirán conocer si existen fallos en el programa.

Lo primero que hay que probar es el si funciona el espejo. Esto es muy sencillo, ya que conectando el Kinect y compilando el programa la textura del plano que proyecta la imagen debería cambiar. En caso contrario es posible que no esté habilitado que el Kinect no esté bien conectado al ordenador o a la corriente, o que el objeto Zigfu o alguno de los componentes de textura esté deshabilitado. Si por el contrario la imagen se ve y se actualiza pero se ve volteada es porque es necesario activar el modo Mirror.

Lo siguiente que es necesario probar es si Kinect capta bien al usuario y si estos datos recogidos se trasladan bien al modelo. Para esto debemos habilitar renderizado de la malla del personaje y compilar el programa. Si el modelo se mueve imitando al usuario significa que funciona correctamente. En caso contrario es posible que alguna de las opciones del script del esqueleto no esté habilitada. Si alguno de los huesos no se mueve correctamente puede ser tanto que Kinect no esté detectando esa parte del usuario, lo que podría solucionarse modificando la situación del aparato; o que alguno de los puntos del modelo no esté bien asignado a uno de los huesos. Si al aparecer el usuario no se muestra ninguna prenda es posible que el Kinect no esté detectando al usuario o que exista algún problema al cargar los prefabs, así que tendremos que buscar en la carpeta de prefabs o en el documento de texto donde están referenciados.

A la hora de comprobar las físicas de los modelos de las prendas es necesario hacerlo en todos ellos. Si uno de ellos tiene un fallo, es posible que sea necesario triangularizar el modelo. Aumentar el grosor de la prenda puede mejorar la sensación de realismo de las físicas. Si el modelo se desajusta y cae al vacío se debe a que el colisionador al que va adherido no está habilitado, que es demasiado pequeño en comparación con el modelo y no se ajusta bien o que la prenda no está unida a ningún colisionador.

Por último, para comprobar que las funciones de la GUI hacen su trabajo correctamente debemos pulsar los botones en cada una de ellas y comprobar el

resultado. Si la fotografía no se muestra o no se guarda bien es posible que hayamos errado al poner una de las rutas de acceso o guardado de la misma. Si el cambio de talla o de modelo hace que el modelo aparezca detrás de donde debería estar, en su posición original, y se cae antes de actualizarse, es porque en necesario resetear la posición del modelo anatómico con ella para que la prenda esté bien adherido al colisionador. Si no se carga es posible que el prefab no se encuentre en la carpeta donde debe estar, la de prefabs; que no esté referenciado correctamente en el documento de texto donde están los nombres o que no existan todas las tallas para dicho modelo. Si el modelo no cambia de color, por último, es posible que no esté buscando el prefab que debe, por lo que la solución es posible que pase por cambiar el tag o el layer de dicho modelo. En las figuras 20, 21, 22 y 23 podemos ver las pruebas de las físicas, del cambio de vestido y de color.

Figura 20 Ejemplo del vestido largo. Las físicas del sistema le aportan pliegues simulando tela

Figura 21 Ejemplo de cambio de vestido con respecto a la figura 20

Figura 22 Ejemplo de cambio de color del vestido palabra de honor a verde

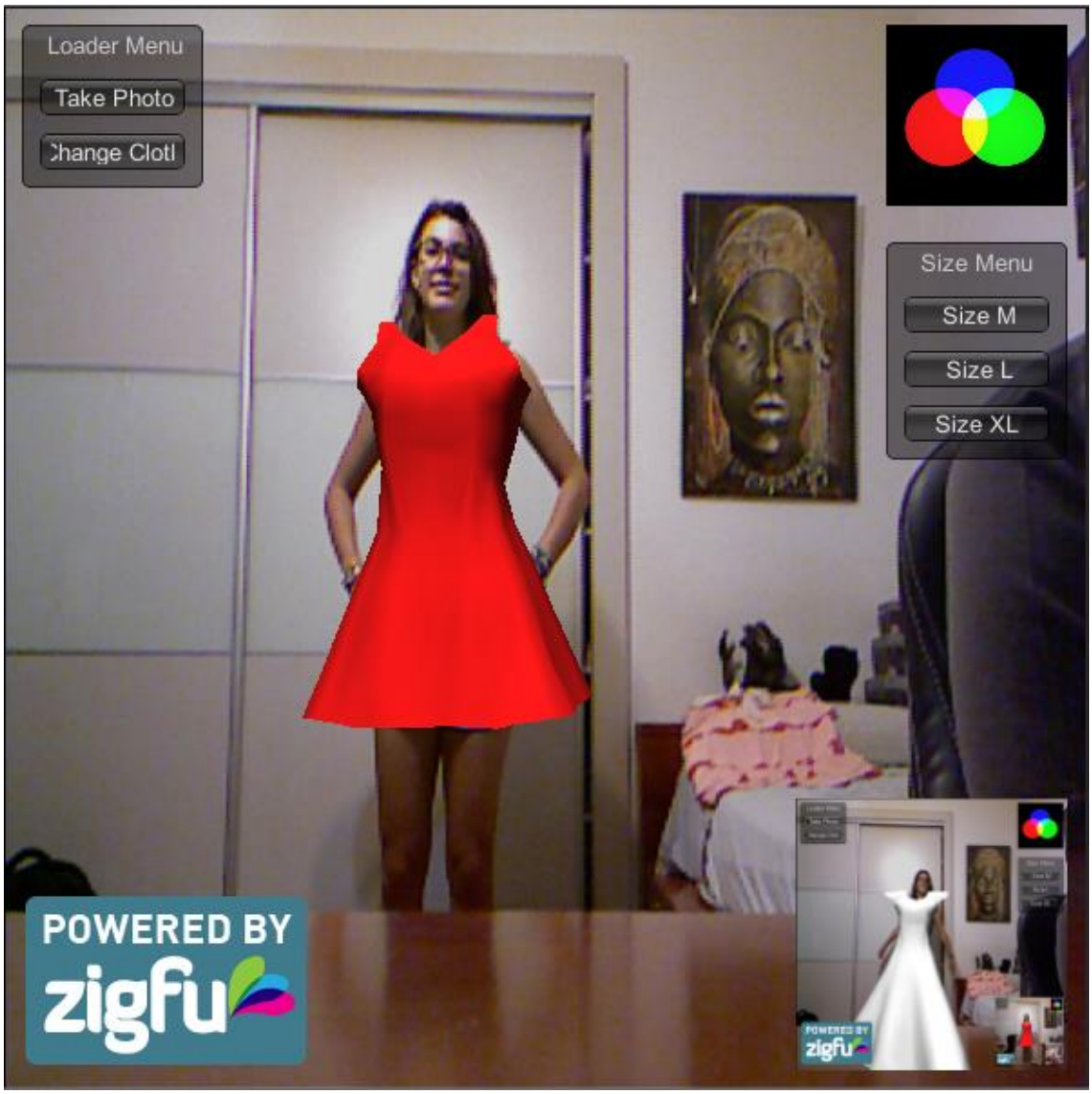

Figura 23 Ejemplo de cambio de color del vestido corto a rojo

7. Conclusiones y desarrollos posteriores

7.1. Conclusiones

El proyecto que se ha desarrollado consta de varios objetivos. El principal sería el desarrollo de un sistema capaz de hacer uso de la realidad aumentada para que un usuario pueda probarse prendas tridimensionales, objetivo que es la base del proyecto y que se puede ver que ha sido completado con éxito puesto que nuestro sistema es perfectamente capaz de mostrar imagen sintética superpuesta a una imagen real que se actualiza durante la ejecución y puede interactuar con el usuario en tiempo real.

El sistema debe ser capaz de ser compatible con varios sistemas operativos, tanto OS X como Windows. Unity nos ha permitido que esto sea posible, ampliando su compatibilidad a cualquier dispositivo que pueda contar con las librerías de OpenNI y ser compilado con Unity, como Linux, Android o en un entorno web, además de poder ser utilizado tanto con Kinect como con la cámara Asus Xtion Pro Life.

El usuario debe poder alterar la prenda para que se adecue su gusto, ya sea cambiando el color, la talla o el mismo modelo de prenda. También debe poder tomar fotografías de ejemplo con la prenda puesta. El sistema debe ser realista, utilizando para ello físicas de tejidos. Todos estos objetivos se han cumplido, según las expectativas, obteniendo unas físicas bastante realistas, un tiempo de respuesta del sistema aceptabley la posibilidad de cambiar entre tres modelos distintos con tres tallas para cada uno.

Las implicaciones del proyecto en un sentido práctico serían directas con el cliente. La aplicación podría permitir a los comercios contar con un reclamo comercial para crear ventas dado su exotismo ante el público general. Este tipo de tecnologías dan al usuario una sensación distinta al realizar acciones rutinarias como ir de compras. Según el creativo Shigueru Miyamoto*, "Si queremos llegar a alguna parte, debemos pensar en la mente del consumidor medio".* El sistema podría integrarse en hardware específico para dar una experiencia más satisfactoria, como pantallas de alta definición, y adaptarse para utilizarse con cámaras normales para la aplicación web. Podría así cambiar por

completo el sistema de compra de los usuarios, revolucionando la calidad del comercio textil online. Una vez mejorado el realismo, este sistema podría representar un paso importante para las transacciones de este tipo en un futuro próximo.

Existen una serie de limitaciones del sistema. Actualmente no es capaz de reconocer bien las prendas con mangas. Esto es debido que no se ha podido ajustar la colisión perfectamente con los boundigs boxes de los brazos puesto que requeriría unas técnicas más avanzadas que las tratadas para la resolución del problema. Otra limitación la encontramos a la hora de girar el modelo, que si bien realiza la rotación correctamente, no es capaz de mostrar los brazos del usuario si se quedan tras el modelo, puesto que sería necesario trabajar en el bucle de dibujado de Unity, lo cual podría solucionarse utilizando shaders avanzados que busque en el Z-buffer de la imagen de profundidad captada con Kinect y el del sistema y dibuje el píxel más cercano a la cámara.

Por último, como valoración personal, ha sido reconfortante poder trabajar con realidad aumentada, un tipo de tecnología que siempre me ha fascinado. Ha sido realmente instructivo aprender cómo funciona la realidad virtual, Unity y Kinect. También ha sido una importante motivación el hecho de poder crear un proyecto destinado al usuario, que pueda ser útil para estos y no solamente tenga destino didáctico, sino que tenga potencial para mejorarse y solucionar problemas reales.

7.2. Desarrollos posteriores

El sistema desarrollado es capaz de ser mejorado en posteriores desarrollos en función de las necesidades comerciales del cliente y del realismo que pretendamos otorgarle. Estas mejoras podemos dividirlas en dos clases, las de funcionalidad y las tecnológicas.

Cuando hablamos de mejoras de funcionalidad nos referimos a aquellas específicas para el público, es decir, dependiendo de dónde se muestre el sistema y a quién esté enfocado. Un ejemplo de esto es utilizar una cámara web corriente en lugar de Kinect para poder llegar a más público sin necesidad de crear una instalación para ello, haciendo el sistema web para que el usuario pueda

utilizarlo desde su ordenador personal. Si bien el programa puede ser compilado como una aplicación web gracias a Unity, el hándicap actual es que no todos los usuarios disponen de un Kinect para utilizarlo, con lo cual sería inviable. Por tanto, habilitar el sistema para hacerlo compatible a las cámaras web comunes en los ordenadores personales, aún a costa de bajar la calidad del seguimiento debido a la falta de sensores de estas, sería una opción interesante si fuese a ser este su uso comercial.

Otra mejora de funcionalidad sería desarrollar un sistema para automatizar la creación de prendas a un usuario de forma semi automática. Esto podría conseguirse creando un programa que permita modificar una serie de parámetros sencillos (tipo de prenda, tamaño de las mangas, tallas y colores disponibles, etc.) y este generase automáticamente un prefab en Unity que se pudiese incluir en el sistema a partir de un modelo base.

También podría modificarse el programa para conseguir que el sistema fuese compatible con otros dispositivos como smartphones o smart tvs. El uso de estos aparatos está en auge. Es de sobra conocida la expansión de los smartphones. El hecho de que exista una gran cantidad de usuarios que dispongan de uno los hace buenos candidatos para esta aplicación porque le permitirá llegar a una mayor cantidad de clientes en forma de app. Las smart tv son también un dispositivo que está siendo cada vez más utilizado y dada su amplitud y resolución de pantalla podría dar a este sistema una experiencia de usuario más completa.

Las mejoras tecnológicas están en cambio relacionadas con cómo funciona el sistema. Dentro de este tipo de mejoras podemos contemplar, por ejemplo, optimizaciones de rendimiento que aumenten la fluidez del movimiento de los modelos y sus físicas y conseguir una respuesta más rápida ante las acciones de los usuarios.

Perfeccionar las físicas y las colisiones es otra mejora importante. Actualmente las mangas en las prendas pueden dar algunos errores debido a los colisionadores. Mejorar estos game objects nos permitirá habilitar el sistema para otro tipo de prendas como camisas o chaquetas. También podría mejorarse

el modo de visualización, permitiendo mostrar los brazos encima de las prendas al girar el usuario.

Otra mejora tecnológica podría ser habilitar la compra de las prendas del sistema mediante una conexión a una base de datos de la que recogemos el precio y disponibilidad de la misma y habilitar el pago mediante tarjeta o DNI. Gracias al DNI electrónico es posible realizar pagos online, lo cual podría ser una funcionalidad interesante para el sistema. La posibilidad de pagar mediante Paypal o tarjeta bancaria podría habilitarse utilizando APIs como Amazon Payments, Google Wallet, E-junkie o el mismo de Paypal. Esto puede ser especialmente útil si habilitamos el sistema para utilizarse en máquinas expendedoras o dentro de las mismas tiendas, descongestionando así las colas y haciendo más fluidas las compras físicas, o si se utiliza el sistema en web.

Para la creación de mejoras es importante la creatividad. Dado que este es un sistema muy visual e interactivo, cualquier apartado destinado a mejorar la calidad del mismo y la experiencia del usuario puede ser importante. Poder mandar la fotografía tomada por e-mail, Facebook y otras redes sociales, generar un código QR con las mismas o imprimirlas es un ejemplo. Habilitar la capacidad de probarse complementos como bolsos o bisutería puede ser otra mejora interesante a contemplar. Por último, crear modelos más realistas utilizando software de modelado de pago y generar skinned cloths con él podría ser interesante para que la respuesta del consumidor sea más favorable.

8. Bibliografía

[Aden Hepburn, 2012] Goertz, the Augmented Reality Shoe Store. www.digitalbuzzblog.com/goertz-augmented-reality-virtual-shoe-fitting-store-installation/. Aden Hepbun. (21/07/2012)

[Alan H. Watt, Mark Watt, 1992] Advanced Animation and Rendering Techniques: Theory and Practice. Alan H.Watt, Mark Watt. Ed. ACM Press

[Blender, 2014] Documentación de Blender. www.blender.org/. Varios autores, última visita: (28/07/2014)

[Bob Widenhofer, 2010] Inside Xbox 360's Kinect controller. www.ee-mes.com/document.asp?doc_id=1281322. Bob Widenhofer, última visita: (28/07/2014)

[Bruce Sterling, 2011] Augmented Reality: Kinect fitting-room for TopShop, Moscow, www.wired.com/2011/05/augmented-reality-fitting-room-for-topshop-moscow. Bruce Sterling. (10/05/2011)

[Carlos Gonzalez y David Vallejo, 2009] Fundamentos de Síntesis de Imagen 3D. Un Enfoque práctico a Blender. Carlos Gonzalez y David Vallejo. Universidad de Castilla la Mancha. (13/07/2009)

[Elaine Kieran & comp, 2014] Cloth Simulation. www.nccastaff.bournemouth.ac.uk/jmacey/MastersProjects/Msc05/cloth_simulation.pdf. Elaine Kieran, Gavin Harrison, Luke Openshaw, última visita: (30/06/2014)

[Facultad de Ingeniería Uruguay, 2014] Realidad Aumentada, www.fing.edu.uy/inco/cursos/sig/clases/AugmentedReality111010.pdf. Varios Autores. Facultad de Ingeniería de la Universidad de la República, Urugay, última visita: (10/06/2014)

[Google, 2014] Google Glass, www.google.com/glass/start/, última visita: (28/07/2014)

[Isaac V.Kerlow, 2004] The Art of 3D: Computer Animation and Effects. Isaac V.Kerlow. Ed. John Wiley & Sons

[James D. Foley & comp, 2013] Computer Graphics. Principle and Practice. James D. Foley, Andries van Dam, John F. Hughes and Richard L. Phillips. Addison-Wesley. Distintas ediciones

[José Urbina, 2012] Kinect y Realidad aumentada, una combinación perfecta para marketing. José Urbina. (05/06/2012)

[Julián Dorado & comp., 2011] HMD, Head Mounted Display, www.sabia.tic.udc.es/gc/Contenidos%20adicionales/trabajos/3D/Realidad%20Virtual/web/dispositivos/hmd.html. Julián Dorado, Daniel Rivero, Enrique Fernández. 2011

[Kristopher Babic, 1999] Cloth Modeling, Kristopher Babic. (28/04/1999)

[Nightmarekitty, 2011] Unity and Kinect Tutorial, www.nightmarekitty.com/2011/10/28/unity-and-kinect-tutorial, Nightmarekitty (usuario), última visita: (10/05/2014)

[Prof. Juan de Urraza, 2014] Teoría y Aplicaciones de la Informática 2. Prof. Juan de Urraza. Universidad Católica "Nuestra Señora de la Asunción"

[Unity3D, 2014] Documentación de Unity3D. www.unity3d.com/es/unity, última visita: (28/07/2014)

[Wikipedia, 2014] Cloth Modeling. www.en.wikipedia.org/wiki/Cloth_modeling, varios autores, última visita: (28/07/2014)

[Wikipedia, 2014] Unity (software). www.en.wikipedia.org/wiki/Unity_(game_engine) , varios autores, última visita: (28/07/2014)

[Xbox, 2014] Componentes del Sensor de Kinect, www.support.xbox.com/es-ES/xbox-360/kinect/kinect-sensor-components, última visita: (28/07/2014)

[Zigfu, 2014] Zigfu Motion Control. www.zigfu.com, última visita: (25/07/2014)