Manejo de entornos virtuales mediante el sensor "*Kinect*" y su aplicación en fisioterapia.

Manipulation in virtual environments using the Kinect sensor to be applied to physiotherapy.

Néstor Antonio Morales Navarro (1).
I.T. de Tuxtla Gutiérrez.
nstrmorales@gmail.com

M.C. Aída Guillermina Cossío Martínez (2), I.T. de Tuxtla Gutierrez, acossio_m@yahoo.com.mx

Jorge Octavio Guzmán Sánchez (3), I. T. de Tuxtla Gutiérrez, jogs78@gmail.com

Mariana Iveth Hernández Meneses (4), I.T. de Tuxtla Gutiérrez, maihzme_91@hotmail.com

Kevin Matías Herrera (5), I.T. de Tuxtla Gutiérrez, kmatias146@hotmail.com

Artículo recibido en septiembre 30, 2013; aceptado en noviembre 29, 2013.

Resumen.

Este artículo presenta la creación de entornos virtuales para terapia física mediante "Visual Studio XNA", con la ayuda de "Blender" para ser manipulados por el sensor "Kinect". El manejo de entornos virtuales ayuda al usuario a situarse en algún contexto de la vida cotidiana para fines como ejercicios físicos, juegos, rehabilitación y enseñanza, entre otros. La finalidad de crear dicho entorno es captar la atención del paciente mostrando un sistema interactivo basado en ejercicios de terapia física en tercera dimensión, donde el paciente pueda estar inmerso, logrando así despertar y estimular su interés por interactuar en un mundo virtual.

Palabras claves. Entornos tridimensionales, mundo virtual, terapia física, videojuegos, sensor *Kinect*.

Abstract.

This paper presents the creation of virtual environments to physical therapy using Visual Studio XNA with the help of Blender and to be handled by the sensor Kinect. The manipulation in virtual environments allows the user to be placed in some context of daily life for purposes such as physical exercises, games, rehabilitation and education among others. Our goal of creating a virtual environment is to capture the attention of the patient through an interactive system based on physical therapy exercises in three dimensions, where the patient can be immersed, achieving their interest in interacting in a virtual world.

Keywords. Three-dimensional environments, virtual world, physical therapy, video games, Kinect sensor.

I. Introducción.

Desarrollar un videojuego comprende varias actividades y procesos, los más generales y que abarcan todo el proceso son diseño y producción, para terminar entregando el producto final, en este caso un videojuego. A

partir de lo que implica diseñar y desarrollar este tipo de software se puede separar en más procesos y actividades que den un flujo lógico para conseguir este fin.

Los entornos virtuales se centran generalmente en la interacción interpersonal, que a pesar de no producirse en el mismo espacio-tiempo, si es percibida como un acto colectivo.

Tiene una estrecha relación con el mundo físico dada su interrelación e influencia mutua. La experiencia en la realidad virtual viene mediada por la experiencia en el mundo real y es influida por lo que allí es experimentado.

La creación de entornos 3D para fisioterapia tiene como finalidad plantear una estrategia nueva en el ámbito de rehabilitación física mediante el cual se inmersa al paciente en un mundo virtual donde pueda experimentar acciones de la vida cotidiana como levantar un brazo o una pierna, atrapar un objeto, mediante videojuegos.

El desarrollo de los videojuegos de terapia física tendrán niveles de dificultad dependiendo de la parte física a rehabilitar estos videojuegos darán información al médico para generar estadísticas de progreso en un rehabilitación.

La evolución de los videojuegos tuvo más auge con la aparición de mundos virtuales en 3D, que de la mano del avance de la tecnología como computadoras con varios procesadores, tarjetas de video que soportan grandes cálculos matemáticos que presentados a través de dispositivos de salida como el monitor en caso de PC y televisores en el caso de consolas de videojuegos como XBOX, Nintendo, etc. atraen al jugador o usuario de la aplicación y estimulando en él la curiosidad por jugar y según como sea del agrado del usuario este tendrá varias horas de entretenimiento.

Los videojuegos además de ser un software de entretenimiento, aportan también con el avance en otras disciplinas, como la inteligencia artificial, simulación computacional, y el desarrollo de hardware más avanzado que soporte la complejidad de los gráficos generados por este tipo de software.

En este artículo se hablará de la primera forma la cual consiste en el desarrollo del sistema interactivo puro donde se trabaja el diseño de los personajes que representa al usuario al momento de interactuar con el videojuego para ello usamos *Blender* para darle seguimiento en XNA.

II. Métodos.

Calibración estéreo. Es una técnica que nos permite asegurar que los datos proporcionados por el sistema de visión sean los correctos. La calibración permite tener la correlación entre una imagen digital y el espacio real, permitiendo conocer las métricas de un objeto a partir de una imagen digital como se observa en la figura 1a. Cabe mencionar que el sistema de visión está formado por dos cámaras, lo que permite hacer la similitud al sistema de visión humano como se muestra en la figura 1b.

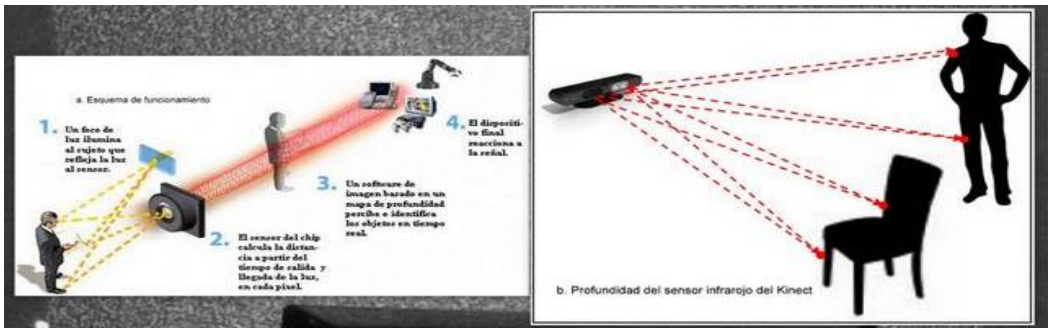


Figura. 1. Sistema de visión Estéreo. (a) Esquema de funcionamiento y (b) Profundidad del sensor infrarrojo del *Kinect.*

Retroalimentación visual basada en posición. Es el nombre de una técnica utilizada en Visión Artificial, la cual se encarga de obtener una imagen bidimensional utilizando un sistema de visión calibrada, y posteriormente procesar la imagen para obtener información tridimensional que permite hacer posible al seguimiento de algunas características medibles de objetos y/o personas como se observa en la figura 2.

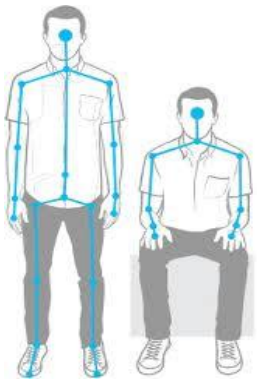


Figura. 2. Puntos obtenidos por el sistema de visión.

Crear un juego. Lo primero sería entender que un sistema interactivo es un motor. Este motor produce un cierto tipo de experiencia a partir de un determinado *input* que le introduzcamos.

Habrá que definir o clasificar los tipos de experiencias que pueden derivarse de los distintos sistemas. Podemos llamar formas a cada tipología de esta clasificación ver figura 3. Conviene aclarar que una forma no es un medio. Un medio para un pintor es su lienzo y para un diseñador de videojuegos será seguramente una computadora. Una forma tampoco es un género, un género es más bien un estilo. Una forma es la categoría que permite distinguir unos motores de otros por el tipo de experiencia que producen o aportan.

Hace falta dar nombres a cada una de estas formas por lo que usaremos el sistema de nomenclatura que propone Burgun es su teoría del diseño de juegos (Burgun, 2013).

Primera forma: El sistema interactivo puro. Puede llamarse juguete, *Sandbox* o Simulador. Carece de metas concretas y se basa en los principios de juego libre y ausencia de estructura.

Un claro ejemplo de esta forma sería *Minecraft* o el *Flight Simulator* de *Microsoft* en su modo de vuelo libre.

Segunda forma: El *puzzle*. Los *puzzles* son el sistema anterior al que se le ha añadido una meta o solución. Entendamos que en este contexto la palabra *puzzle* describe una categoría de forma, una clase de sistema y no hace referencia a un género, como ya se ha dicho.

Tercera forma: El concurso. Un concurso sería un *puzzle* al que le añadimos algún tipo de competencia. Un buen concurso debería aportar una medida clara de alguna habilidad ya sea motriz o de otro tipo. Un ejemplo sería el *Dance Revolution*.

Cuarta y última forma: El juego. Un juego es el sistema anterior al que le hemos añadido la toma de decisiones. Un buen juego le dará al jugador una batería inagotable de decisiones difíciles e interesantes. Podríamos incluir el *Starcraft* como ejemplo, pero también el *Tetris*. Esto último es interesante, ya que la mayoría de la gente lo clasificaría como *puzzle*, aunque si atendemos a las definiciones planteadas, es indudable que se clasificaría como juego.

Establecidas las formas, será mucho más sencillo plantear qué clase de sistema queremos crear. Nuestro diseño será más claro desde un primer momento, haciendo que todo el proceso sea más eficaz y resultando, seguramente, en un mejor producto final.

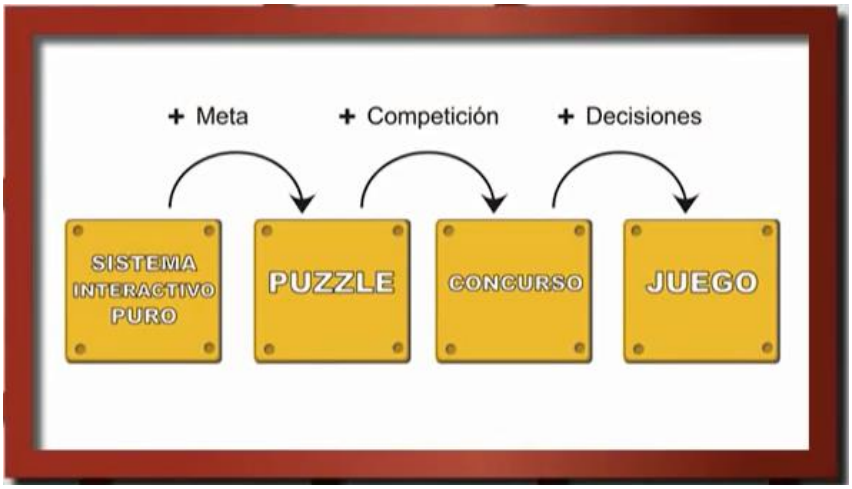


Figura 3. Clasificación para el desarrollo de un juego

Herramientas utilizadas. Visual Studio es un conjunto completo de herramientas de desarrollo para la generación de aplicaciones Web ASP.NET, servicios Web XML, aplicaciones de escritorio y aplicaciones móviles. Visual Basic, Visual C++, Visual C# y Visual J# utilizan el mismo entorno de desarrollo integrado (IDE), que les permite compartir herramientas y facilita la creación de soluciones en varios lenguajes. Así mismo, dichos lenguajes aprovechan las funciones de .NET Framework, que ofrece acceso a tecnologías clave para simplificar el desarrollo de aplicaciones Web ASP y Servicios Web XML (*Microsoft*, 2005).

XNA *Game Studio* 4.0 es un *framework* para el desarrollo de videojuegos que proporciona un amplio conjunto de librerías de clases específicas para el desarrollo de juegos por lo que muchas de las tareas comunes en el desarrollo de videojuegos ya están implementadas (*Microsoft*, 2010).

XNA proporciona un bucle principal integrado a través de la clase *Game*, por lo que el desarrollado solo debe escribir los métodos *update* y *draw* (*Microsoft*, 2010).

Blender es un paquete de modelado y animación en 3D que se actualiza constantemente. El hecho de tener un motor de juegos lo hace diferente de los demás paquetes de modelado y animación en 3D. Las fortalezas principales del motor de juegos de *Blender* son las siguientes: Ambiente de edición integrado, modelado, animación y juegos, motor de física *Engine Bullet*, definición de sensores, multiplataforma (Blender, 2011).

III. Desarrollo.

Ha llegado el momento de la creación del *avatar* pero ¿Qué es un *avatar*? Es un personaje digital que el usuario puede crear y personalizar. Para el desarrollo de estos hemos usado un motor de juego llamado *Blender*. En esta herramienta se realizan todos los entornos en 3D que son parte del videojuego para darles "vida" en XNA ya que la plataforma en el que se desarrolla es *Windows* para que funcione con el *Kinect*.

La parte fundamental del desarrollo de escenas 3D es el modelado, ya que es en esta etapa en la que se crea el contenido (modelos). La creación de modelos es entretenida y a veces puede constituir un reto.

Los personajes creados para este videojuego se realizan mediante la subdivisión partiendo desde un cubo como se muestra en la figura 4.

Antes de empezar a crear el avatar debemos de tener el diseño del personaje ya sea en papel o en alguna herramienta como *Photoshop*.

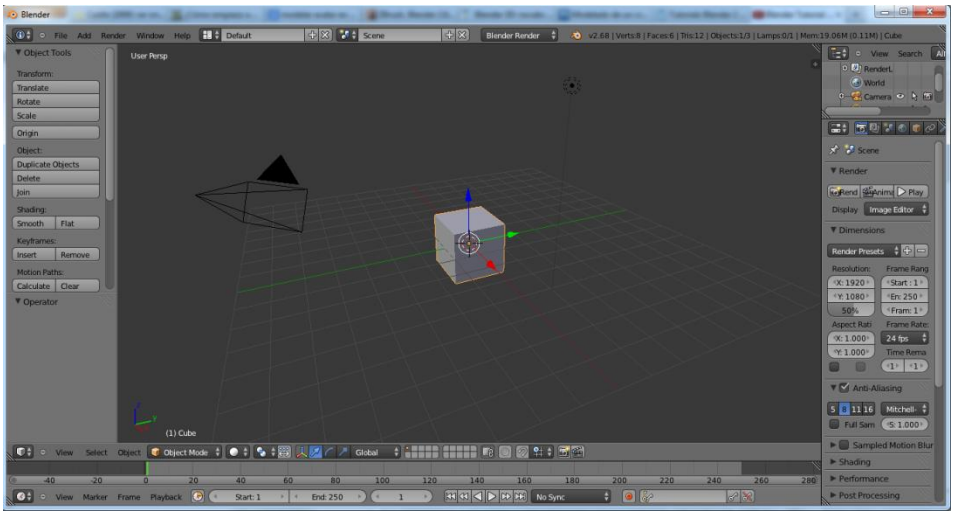


Figura 4. Pantalla principal de Blender para empezar el modelado

Después de un largo proceso de modelado de los personajes, obtenemos el resultado final como se muestra en las figuras 5 y 6.

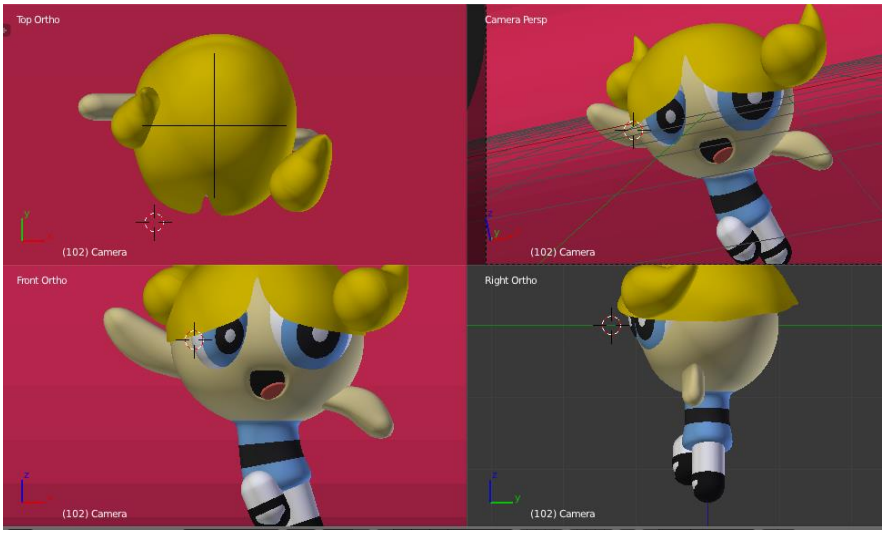


Figura 5. Personaje de Burbuja

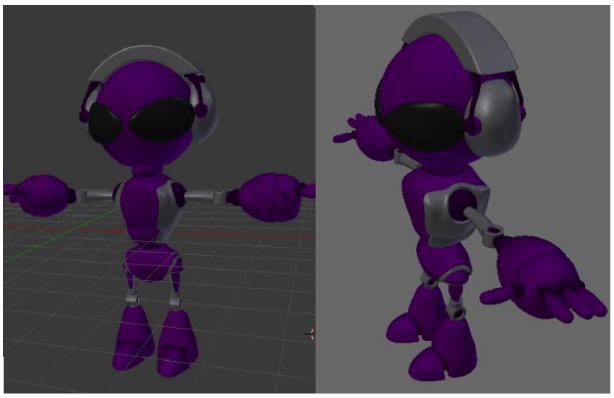


Figura 6. Personaje de Rebotín.

Teniendo los personajes en *Blender* se realiza la exportación de los modelos en archivos de formato FBX para su uso con XNA.

Tal y como se ha comentado, XNA es un *framework* para el desarrollo de videojuegos que proporciona un amplio conjunto de librerías de clases específicas para el desarrollo de juegos por lo que muchas de las tareas comunes en el desarrollo de videojuegos ya están implementadas.

XNA proporciona un bucle principal integrado a través de la clase *Game*, por lo que el desarrollado sólo debe escribir los métodos *update* y *draw*. El desarrollador puede configurar XNA para que ejecute su bucle principal de dos modos diferentes:

Tiempo fijo: El desarrollador le indica a XNA un tiempo **t** y este se encarga de que cada **t** instantes de tiempo se llame al método *update*, ejecutando además, el método *draw* inmediatamente después del método *update*. Si XNA no puede conseguir la velocidad deseada, ya sea bien porque el método *update* o *draw* están llevando demasiado tiempo o por que el sistema se ha puesto lento, por ejemplo; por un ciclo de recogida de basura, empezará a saltarse llamadas al método *draw* con el objetivo de conseguir la frecuencia deseada en el método *update*.

Tiempo variable: XNA ejecuta los métodos *update* y *draw* de forma alternativa lo más rápido posible y es tarea del programador llevar un control del tiempo y controlar que el juego se ejecute a la misma velocidad en todos los sistemas y controlar el tiempo transcurrido entre ciclos.

La segunda opción proporciona una mayor libertad y mayor potencia de cálculo, pero es más compleja de desarrollar.

Conclusiones.

Uno de los problemas a los que se enfrenta el grupo de desarrollo de videojuegos es la elección de las herramientas de software que les permita alcanzar de manera flexible y rápida los objetivos de su diseño. Para esto es importante conocer las características de los motores de juegos, ya que dicha elección repercutirá en la eficiencia de trabajo, además de que se derivan aspectos de tiempo, herramientas artísticas y de programación, monto de inversión.

En este caso usamos *Blender* y XNA *Game Studio* 4.0 ambos son motores de juegos cada uno con características particulares pero que se complementan para desarrollar el videojuego en *Visual Studio*.

Blender es conveniente para animaciones más que para videojuegos debido a que la ejecución del juego depende del entorno. XNA es buena opción cuando la plataforma elegida es *Windows* y se desea seguir en esa línea para elaborar juegos en la consola Xbox 360.

Al saber un poco de los motores de desarrollo y teniendo los bocetos de los personajes se comienza a desarrollar en Blender para poder darle un toque más real a los personajes ya que eso ayudará a que los pacientes tengan un gran interés en interactuar con el videojuego.

El desarrollo de un videojuego es algo muy complejo por eso en este artículo se habla solamente de una parte del desarrollo que ayuda a dar inicio a la forma dos llamada *Puzzle* y así consecutivamente hasta llegar al objetivo que es el videojuego y poder dar inicio a la interacción de paciente-videojuego.

Agradecimientos.

Los autores agradecen y dan créditos al Instituto Tecnológico de Tuxtla Gutiérrez por todas las facilidades y apoyo para la realización de este proyecto.

Referencias bibliográficas.

Burgun, K. (2013). *"Game Systems as Engines"*. Recuperado de: http://keithburgun.net/game-systems-as-engines/

Microsoft (2005). *Introducción a Visual Studio*. Recuperado de: http://msdn.microsoft.com/es-es/library/fx6bk1f4 %28v=vs.80 %29.aspx.

Microsoft (2010). *Introducción al desarrollo de XNA Game Studio*. Recuperado de: http://msdn.microsoft.com/esES/library/bb203894 %28v=xnagamestudio.40 %29.aspx.

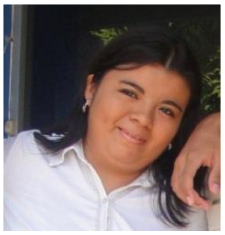
Blender (2011). *Doc:2.6/Manual/Introduction.* Recuperado de: http://wiki.blender.org/index.php/Doc:2.6/Manual/Introduction

Información de Autores.

Néstor Antonio Morales Navarro es Maestro en Ciencias en Ingeniería Mecatrónica por el Instituto Tecnológico de Tuxtla Gutiérrez en 2010. Es profesor en el área de Ingeniería en Sistemas Computacionales del Instituto Tecnológico de Tuxtla Gutiérrez desde 2012 y en el área de Licenciatura en Ingeniería Mecatrónica de la Universidad Valle de México Campus Tuxtla desde 2011. Se especializa en el área de Visión e Inteligencia Artificial.

Aída Guillermina Cossío Martínez es Maestra en Ciencias en Administración por el Instituto Tecnológico de Tuxtla Gutiérrez en 2002. Es profesora de tiempo completo del área de Ingeniería en Sistemas Computacionales desde 1994. Se especializa en la formulación y evaluación de proyectos, así como el emprendimiento y desarrollo de planes de negocio.

Jorge Octavio Guzmán Sánchez tiene la Maestría en Ciencias de la Computación, especialidad bases de datos y sistemas de información, es Ingeniero en Sistemas Computacionales, profesional certificado por Microsoft en la administración de servidores con Windows. Ejerce la docencia desde hace más de una década, actualmente docente del Instituto Tecnológico de Tuxtla Gutiérrez así como en la Universidad Descartes.



Mariana Iveth Hernández Meneses es estudiante del Instituto Tecnológico de Tuxtla Gutiérrez de la carrera de Ingeniería en Sistemas Computacionales. Tiene una publicación en el Congreso Internacional de Investigación *Academia Journals* 2013.

Kevin Matias Herrera es estudiante del Instituto Tecnológico de Tuxtla Gutiérrez de la carrera de Ingeniería en Sistemas Computacionales. Tiene una publicación en el Congreso Internacional de Investigación *Academia Journals* 2013.