DESARROLLO DE UN ENTORNO VIRTUAL EN PRIMERA PERSONA DIRIGIDO POR UN CONTROL BLUETOOTH PARA DISPOSITIVOS MÓVILES BASADOS EN ANDROID.

LUIS FERNANDO BLUM MENDOZA

UNIVERSIDAD DE PAMPLONA

FACULTAD DE INGENIERÍAS Y ARQUITECTURA

INGENIERÍA MECATRÓNICA

PAMPLONA

2020

DESARROLLO DE UN ENTORNO VIRTUAL EN PRIMERA PERSONA DIRIGIDO POR UN CONTROL BLUETOOTH PARA DISPOSITIVOS MÓVILES BASADOS EN ANDROID.

LUIS FERNANDO BLUM MENDOZA

Trabajo de grado para optar por el título de ingeniero en mecatrónica.

Directora del proyecto:

M.Sc. YARA ANGELINE OVIEDO DURANGO

Magister en controles industriales.

UNIVERSIDAD DE PAMPLONA

FACULTAD DE INGENIERÍAS Y ARQUITECTURA

INGENIERÍA MECATRÓNICA

PAMPLONA

2020

Nota de aceptación:

_____

_____

_____

_____

_____

_____

_____

Firma de la directora.

_____

Firma del jurado.

_____

Firma del jurado.

Pamplona, 18 de junio, 2020.

Este trabajo de grado se lo dedico principalmente a mi familia, a mis padres Luis Eduardo Blum y Carmen Mendoza, a mi hermana Stephanie Blum Mendoza que sin su apoyo y confianza no fuera sido posible este título profesional.

AGRADECIMIENTOS.

Le agradezco a mis padres Luis Eduardo Blum y Carmen Mendoza por su apoyo incondicional no solamente durante toda la carrera universitaria si no durante toda mi, vida que forjaron en mí una personalidad fuerte y responsable.

A mi hermana Stephanie Blum Mendoza que igualmente por su apoyo en momentos difíciles que sin sus consejos y ejemplos me ayudaron continuar con fortaleza y éxito en el proceso de la universidad, a Víctor Alfonso Jaimes que le agradezco indirectamente por apoyar a mi hermana y brindarle felicidad siendo su cónyuge.

A la ingeniera Yara Oviedo Durango por su apoyo académico durante la realización de este proyecto y la redacción del informe escrito del trabajo de grado.

A cada uno de los profesores que hicieron parte del proceso formativo profesional en especial al ingeniero Diego José Barrera por haberme colaborado incondicionalmente en asuntos problemáticos en la universidad, al ingeniero Luis Neira Ropero por sus ayudas y asesorías académicas durante el transcurso de la carrera profesional.

A mis compañeros y amigos cercanos de alguna u otra forma hicieron parte de este proceso formativo, especialmente a Anderson Sánchez, Alejandro Celeita, Adrián Larrota, Mateo Perales, Emerson Cabanzo y Sergio Cardona por brindarme su valiosa amistad y en muchas ocasiones por su apoyo al proceso profesional y personal.

TABLA DE CONTENIDO

LISTA DE FIGURAS.

GLOSARIO.

ASSETS: Es una representación de cualquier ítem que puede ser utilizado en su juego o proyecto, un asset podría venir de un archivo creado afuera de Unity, tal como un modelo 3D, un archivo de audio, una imagen, o cualquiera de los otros tipos de archivos que Unity soporta.

APK: Por sus siglas traducidas en ingles paquete de aplicación Android es un paquete para el sistema operativo Android.

AUDIO SOURCE: Es el componente de los gameobjects de Unity que permiten agregar archivos de audio dentro del proyecto.

AUDIO LISTENER: Es un componente para los gameobjects que permiten que los archivos de audio agregados puedan ser escuchados por el usuario no reproducidos por el dispositivo que ese esté utilizando para el videojuego.

BOX COLLIDER: Es una herramienta que poseen los objetos dentro del escenario que contienen las diferentes librerías de colisiones para garantizar la interacción del usuario con estos objetos.

COMPONENTE: Son el conjunto de scripts basados en visual studio que comprenden una gran cantidad de características que le agregan diferentes capacidades a los gameobjects del escenario ya sean detección de colisiones, simulación de gravedad, captura de movimientos, entre otros.

FBX: Es un formato de archivo patentado desarrollado por Kaydara y propiedad de Autodesk desde 2006. Se utiliza para proporcionar interoperabilidad entre aplicaciones de creación de contenido digital.

GAMEOBJECT: Son objetos fundamentales de Unity que representan personajes y escenarios. Estos no logran nada por si mismos, pero funcionan como contenedores para componentes, que implementan la verdadera funcionalidad.

HUESOS: También llamados armadura son los elementos que permiten que determinada pieza en Blender tenga la capacidad de moverse o realizar animaciones.

JOYSTICK: Es un periférico de entrada que consiste en una palanca que gira sobre una base e informa su ángulo o dirección al dispositivo que está controlando.

LINEAR ROLLOF: Es una propiedad del componente *audio source* que define el comportamiento con el que se reproduce determinado sonido desde ciertos puntos de distancia, es decir, si el jugador entra o está cerca de un objeto con este componente este escuchara el audio si se aleja deja de escucharlo, si se acerca lo escuchara más fuerte, como su nombre en inglés lo indica la forma de cambio de volumen antes mencionada es de forma lineal.

PREFABS: Es un tipo de *asset* que le permite almacenar un objeto de tipo *gameobject* completamente con componentes y propiedades. También actúa como una pantalla a partir de la cual se pueden crear nuevas instancias del objeto en la escena.

SCRIPT: Es un documento de texto donde se colocan instrucciones u órdenes que luego serán ejecutadas por un dispositivo inteligente.

STL: Es un formato de archivo informático de diseño asistido por computadora que define geometría de objetos 3D, que excluyen información como color, texturas o propiedades físicas que si incluyen otros formatos CAD.

RESUMEN GENERAL DE TRABAJO DE GRADO

TITULO:	**DESARROLLO DE UN ENTORNO VIRTUAL EN PRIMERA PERSONA DIRIGIDO POR UN CONTROL BLUETOOTH PARA DISPOSITIVOS MÓVILES BASADOS EN ANDROID.**
AUTOR:	**LUIS FERNANDO BLUM MENDOZA**
FACULTAD:	**INGENIERÍAS Y ARQUITECTURA.**
DIRECTORA:	**YARA ANGELINE OVIEDO DURANGO**

RESUMEN

Los dispositivos móviles permiten que el usuario tenga contacto con múltiples tipos de videojuegos en los cuales este trabajo de grado se destaca la realidad virtual la cual está constituida por diferentes objetos de apariencia real en donde el jugador puede interactuar con estos y realizar diferentes tareas u objetivos.

Este proyecto presenta el desarrollo de una aplicación de realidad virtual de un laberinto sencillo por medio de compilador para la generación de archivos de tipo apk que son destinados para dispositivos que están basados en el sistema operativo Android que a su vez será manejado por un control bluetooth común de teléfonos celulares para generar movimientos básicos, haciendo uso de los paquetes que contiene el software de creación de videojuegos y los códigos de programación que capturan los comandos del mando inalámbrico, este entorno virtual fue creado por medio del software unity con las diferentes herramientas que aporta el programa para diseñar escenarios ya sea con esferas, cubos y demás objetos presentes, dentro del escenario virtual habrá 3 tipos de robots los cuales estarán situados en diferentes posiciones en el laberinto y serán dibujados por medio del programa blender así como su respectiva animación.

El objetivo del usuario es buscar cada uno de los robots para poder salir del laberinto, en el momento en el que el usuario encuentre uno este le aportara información acerca de él y realizara una animación sencilla de su funcionamiento.

PALABRAS CLAVE: Realidad virtual, Robótica, Primera Persona, edujuegos.

ABSTRACT

Mobile devices allow the user to have contact with multiple types of video games in which this degree work highlights virtual reality, which is made up of different objects of real appearance where the player can interact with them and perform different tasks or objectives.

This project presents the development of a virtual reality application of a simple maze by means of a compiler to generate apk-type files that are intended for devices that are based on the Android operating system, which in turn will be managed by a bluetooth control. common cell phones to generate basic movements, making use of the packages that the video game creation software contains and the programming codes that capture the commands of the wireless controller, this virtual environment was created by means of the unity software with the different tools that The program contributes to design scenarios with spheres, cubes and other objects present, within the virtual stage there will be 3 types of robots which will be located in different positions in the maze and will be drawn using the blender program as well as their respective animation.

The objective of the user is to search for each of the robots in order to leave the maze, at the moment the user finds one, it will provide information about it and make a simple animation of its operation.

KEY WORDS: Virtual reality, Robotics, First Person, learn-games.

INTRODUCCIÓN

Desde hace muchos años más exactamente en los años 60 la realidad virtual ha formado parte de nuestra vida, con los primeros prototipos en donde se crearon cascos con pantallas incluidas que mostraban al usuario una serie de imágenes determinadas, y a pesar de que eran los primeros inicios de esta tecnología no fue sino hasta el año 1989 que se inventó el termino realidad virtual, al pasar de los años y con muchos otros prototipos y al llegar a la actualidad este término se ha expandido y ha conllevado a crear muchos sistemas diferentes en donde no solamente se intenta que el usuario interactúe con un entorno virtual de manera inmersiva sino que también se logra que este pueda sentir los objetos con los que en ese momento este interactuando, este tipo de sistemas se implementan mediante pequeñas vibraciones que dan al jugador la sensación de tacto en sus manos esto se logra mediante guantes con pequeños motores vibratorios como el prototipo llamado avatar vr que fue presentado en la Madrid *Gaming Experience*, en donde mostraba su guante háptico junto con el sistema *HTC vive* en donde logran que el usuario interactúe con un entorno virtual recibiendo una retroalimentación por medio del tacto en sus manos.

Actualmente en dispositivos móviles que están basados en sistemas operativos IOS y Android también se han desarrollado aplicaciones de realidad virtual, pero que a diferencia del anterior prototipo solamente se genera la visualización e interacción con el escenario, para estos dispositivos móviles se implementan unas gafas de realidad virtual pero que son más sencillas que los modelos conocidos comercialmente como el *Oculus Rift* o el *HTC Vive*, ya que simplemente son dos lentes biconvexos que están situados dentro de una caja de plástico o cartón con unas correas licradas para usar como casco, el dispositivo móvil se coloca delante de estos lentes dentro de las gafas, el de estas aplicaciones de realidad virtual junto con los lentes es mostrar en la pantalla una imagen doble del escenario idénticas entre si y una la lado de la otra de tal forma para que al posicionarlas con los lentes de las gafas estos actúen como una lupa magnificando la imagen para que llene todo el campo de visión, logrando que los ojos compensen la imagen y así se cree una sensación de espacialidad o tridimensional para que se genere el sistema de realidad virtual.

Hay dos tipos de escenarios en estos dispositivos uno de ellos la visualización e interacción dentro del mismo en un entorno que se mueve mientras que el jugador

permanece estático, el otro tipo se emplea de manera inversa ya que el escenario permanece estático mientras que el jugador es el que se mueve.

En el caso de este proyecto se realizará basándose en la segunda forma y, además de tener un aprendizaje sobre los robots tradicionales y animaciones sobre estos.

1. OBJETIVO GENERAL

Desarrollar un entorno virtual en primera persona dirigido por medio de un control bluetooth para dispositivos móviles basados en Android.

1.2. OBJETIVOS ESPECIFICOS.

- Diseñar el entorno virtual en el software unity.
- Modelar tres tipos tradicionales de robots para animarlos en el entorno virtual.
- Elaborar la aplicación de realidad virtual con conexión inalámbrica para los movimientos dentro del escenario virtual.
- Generar la comunicación entre el ambiente virtual y el dispositivo de control.
- Validar el proyecto con diferentes usuarios.

2. MARCO TEÓRICO.

Antes de iniciar con el desarrollo de este proyecto se debe indagar acerca de los conceptos básicos que componen el área en la cual se está trabajando ya sea temas relacionados con la realidad virtual o diseños de videojuegos entre otros.

2.2. Realidad virtual.

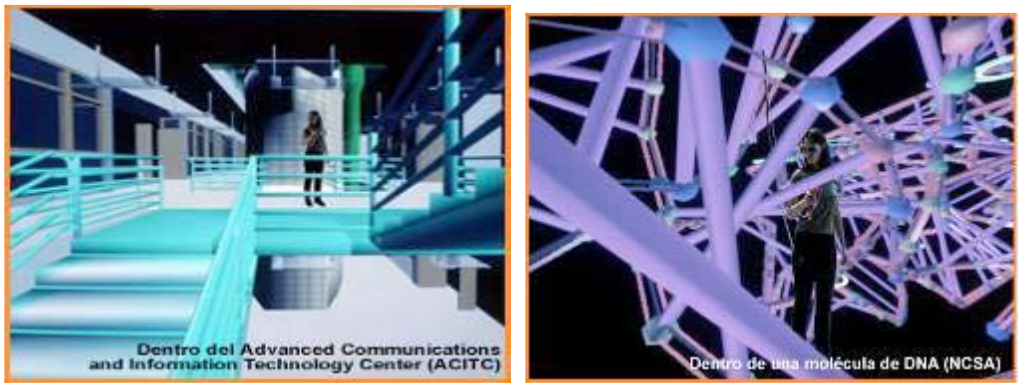
La realidad virtual como definición básica comprende la interfaz hombre-máquina, que permite al usuario sumergirse en una simulación de tipo grafica con efectos 3D generada por ordenador, y navegar e interactuar en ella en tiempo real, desde una perspectiva centrada en el usuario.[5]

La realidad virtual es una experiencia mediante la cual se busca que el usuario logre sustituir la realidad física por un entorno ficticio que se genera o se crea por ordenador, la realidad es muy parecido a una máquina del tiempo ya que esta permite recrear virtualmente cualquier tipo de espacio en tres dimensiones y situarlo en cualquier época que sea lo más creíble que se pueda.

Un usuario que entra en contacto con un espacio virtual tiene al menos dos formas de hacerlo que son activos o pasivos, la manera activa es que este interactúa con los objetos presente en el entorno, la forma pasiva corresponde a que el usuario actúa como un simple observador y no puede interactuar con nada de los objetos presentes solamente puede moverse dentro del espacio virtual, pero la interacción es necesaria ya que sin la inmersión “física” no se tendrá una verdadera realidad virtual.

[5] F. J. P. Martínez, «Presente y Futuro de la Tecnología de la Realidad Virtual,» *Creatividad y sociedad,* 2011.

Figura 1. Entornos virtuales creados por NCSA.



Fuente: F. J. P. Martínez, «Presente y Futuro de la Tecnología de la Realidad Virtual,» *Creatividad y sociedad,* 2011.

Una de las principales ventajas de la realidad virtual es que permite encontrarse cara a cara con la información, tal y como es posible observar en la fig. 1 en donde los investigadores pueden recorrer el interior de una molécula de ADN con un control o un guante virtual y una gafa de visualización estereoscópica, que permiten al usuario acercarse tanto como desee al punto concreto de su interés.

2.3. Triangulo de la realidad virtual.

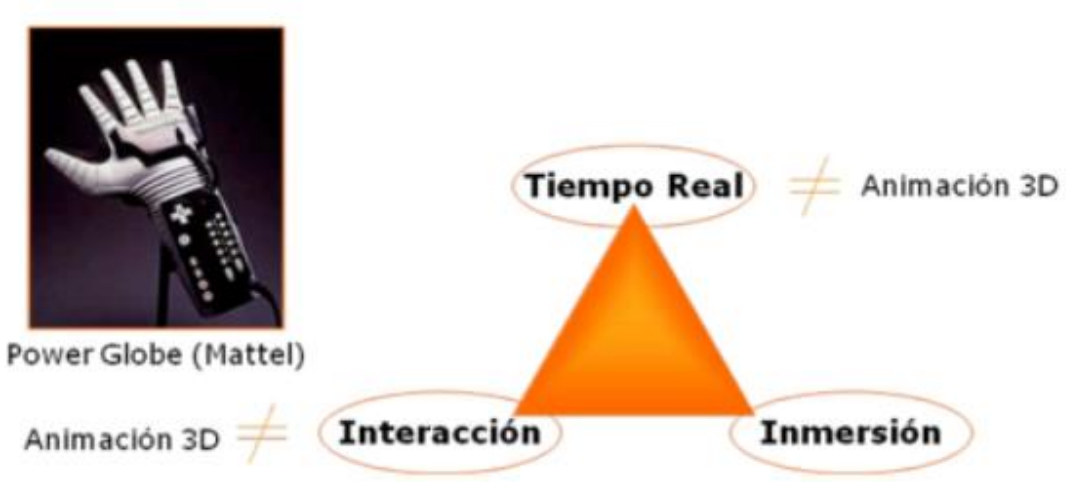
La realidad virtual se puede considerar como un medio de comunicación en sí mismo, que puede ser capaz de permitir en un futuro la participación corporal total en un mismo espacio compartido de telecomunicaciones generado por ordenador que podría estar dotado de retroalimentación multisensorial.[5]

Hay tres características fundamentales que definen la realidad virtual frente a las simples animación en tres dimensiones tradicionales las cuales son posibilidad de tiempo real, que permite elegir la dirección con la cual se genera el movimiento en el interior del escenario o hacia dónde dirigir la mirada, otra característica es la inmersión completa por el interior del mismo, en donde se pierde el contacto con la realidad exterior al percibir únicamente los estímulos del mundo virtual, por último la interacción con los elementos que lo conforman, lo que permite interactuar con el mundo virtual a través de diversos dispositivos de entrada.

_____

[5] F. J. P. Martínez, «Presente y Futuro de la Tecnología de la Realidad Virtual,» *Creatividad y sociedad,* 2011.

Figura 2.Triangulo de la realidad virtual.



Fuente: F. J. P. Martínez, «Presente y Futuro de la Tecnología de la Realidad Virtual,» *Creatividad y sociedad,* 2011.

Dos características están relacionadas a pesar de que no significan lo mismo las cuales son tiempo real e interactividad. Estas son necesarias para una definición de realidad virtual, entre otros motivos.

2.4. Vistas empleadas en videojuegos.

Hay dos tipos de vistas típicas que se utilizan en los videojuegos las cuales son:

2.4.1. Cámara en primera persona.

Esta vista se caracteriza porque el jugador o usuario ve el mundo virtual desde la perspectiva del personaje protagonista, es decir en los ojos del protagonista está situada la cámara del juego donde el usuario tiene la perspectiva del juego. Este tipo de vista es comúnmente usado en los videojuegos de disparos, dando lugar al conocido genero llamado disparos en primera persona o por sus siglas en inglés *FPS* o *first person shooter*, sin embargo, no solamente se utiliza esta vista para este género ya que hay otros como el de tipo aventura en donde también se utilizan.

Una de las principales ventajas es que da un mayor nivel de realismo y además

aporta la sensación de presencia en el videojuego, pero una desventaja es que, al ver únicamente hacia el frente, el jugador debe rotar para saber lo que lo rodea.

Dos de los primeros videojuegos en donde se usó este tipo de vista fue en *Maze War* de 1974 y *Spasim* del mismo año no obstante el videojuego que logro consagrar esta vista primero fue *Wolfenstein* 3D del año 1992 y después con *Doom* del año 1993 a partir de ese momento la vista en primera persona empezó a ser ampliamente usada en el género de disparos en primera persona.

2.4.2. Cámara en tercera persona.

Esta vista es una de las más frecuentes en cuanto a los videojuegos de géneros como aventuras gráficas, juegos de rol entre otras, esta vista tiene la particularidad que el personaje que se controla es visto de cuerpo entero y generalmente de espaldas.

En juegos como *Silent Hill*, *Resident Evil*, *Gears of War*, la saga de GTA o en la mayoría de juegos que pertenezcan al género *Survival Horror* se utilizan con frecuencia esta vista debido a que da una sensación de inmersión en el juego. Es ideal para videojuegos en donde la acción se centra principalmente en la búsqueda de objetos o indicios para descubrir la historia o trama del videojuego, una ventaja que presenta es que el jugador tiene la capacidad de ver directamente a los alrededores del personaje que controla si la necesidad de tener que voltear la vista.

2.5. Historia de la realidad virtual.

Iván Sutherland [46], público en el año 1965 un artículo llamado *The Ultimate Display* en el que describía el HMD, casco o *head mounted display*, hasta que Jaron Lanier el gerente del laboratorio planetario virtual quien invento el termino realidad virtual en 1989, la NASA y el ejército de los estados unidos fueron algunos de los primeros que usaron este tipo de tecnología.

El cortador de césped de 1992 es una de las primeras películas que se producen cuya temática es la realidad virtual, igualmente juegos de ordenador. Hacia 1994 aparece un software de archivo estándar que tenía la capacidad de visualizar modelos en tres dimensiones llamado en la web como virtual *reality modeling language*, este programa si se considera 3D interactiva, a pesar de que no es inmersiva. En el laboratorio de visión electrónica de Chicago inventaron el *computer*

[46] I. E. Sutherland, «The ultimate display,» *Multimedia: From Wagner to virtual reality,* vol. 1, 1965.

automatic virtual enviroment en el año 1992 que se basa en la proyección de imágenes sobre unas paredes translucidas, que son pantallas de retroproyección, normalmente opera mediante un sistema de visión llamado estereoscópico de esta manera múltiples usuarios pueden interactuar entre ellos y dentro del entorno virtual compartido.

2.6. Sistemas de realidad virtual.

Los sistemas de realidad virtual son aquellos que definen el tipo de escenario que va a interactuar el usuario y se dividen en los siguientes tipos:

2.6.1. Sistemas desktops de realidad virtual o *window on world*.

Este sistema muestra una imagen en dos o tres dimensiones en un monitor, es el simple concepto de los videojuegos para ordenadores o consolas comunes, el usuario ve la imagen en primera persona.

2.6.2. Realidad virtual en segunda persona.

El usuario sabe que está dentro del entorno virtual porque se ve a sí mismo dentro de la escena, en este caso se aplica la idea de inducir la sensación de presencia.

2.6.3. Sistemas de tele presencia.

Con cámaras, dispositivos táctiles y de retroalimentación que están ligados a elementos de control remoto que permiten manipular robots ubicados a distancia mientras se experimentan en forma virtual.

2.6.4. Sistemas de inmersión de realidad virtual.

Este sistema sumerge al usuario en el mundo virtual, usando sistemas virtuales tipo CAVE, con sensores de posición y movimiento, quedando el usuario, sumergirlo realmente en la atmosfera virtual y formando parte de ese mundo.

2.7. Realidad aumentada.

Aunque este tipo de sistemas de videojuegos no son parte del trabajo de grado es necesario tener los conceptos claros acerca de esta área para así lograr diferenciar entre ellos.

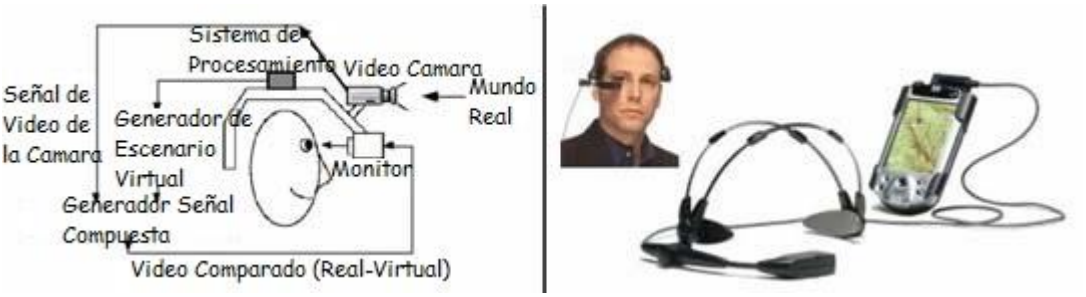
La realidad aumentada o también conocida por sus siglas RA adquiere presencia en el mundo científico a principios de los años 1990 cuando la tecnología basada en ordenadores de procesamiento rápido, técnicas de renderizado de gráficos en tiempo real y sistemas de seguimiento de precisión portable.[2]

Estos permiten implementar la combinación de imágenes generadas por el ordenador sobre la visión del mundo real que tiene el usuario. En muchas aplicaciones industriales y domésticas se disponen de un gran número de información que están generalmente asociados a objetos del mundo real, y la realidad aumenta se presenta como el medio que tiene la capacidad de unir ambos mundos como son la información específica ya sea muy descriptiva o básica y los objetos del mundo real a los que pertenecen esta información. Con este sistema se pueden visualizar en un espacio vacío de una habitación los diferentes objetos que se han diseñado anteriormente.

La realidad aumentada y la realidad virtual están ligadas entre sí que la última está más extendida en la sociedad que esta presenta algunas características en común los cuales son modelos virtuales en dos dimensiones o tres dimensiones en el campo de visión del usuario. La gran diferencia de la realidad aumentada con la realidad virtual es que esta no reemplaza el mundo real por uno virtual, sino que mantiene el mundo real que ve el usuario añadiéndole información virtual superpuesta al real. Este sistema le da la ventaja al usuario de no perder el contacto con el mundo real y además puede interactuar con la información de manera virtual que tiene a su alcance.

[2] X. Basogain, M. Olabe, K. Espinosa, C. Rouèche y J. C. Olabe, «Realidad Aumentada en la Educación: una tecnología emergente,» *Escuela Superior de Ingeniería de Bilbao, EHU. Recuperado de http://bit. ly/2hpZokY,* 2007.

Figura 3.Diagrama conceptual de un sistema de realidad aumentada.



Fuente: X. Basogain, M. Olabe, K. Espinosa, C. Rouèche y J. C. Olabe, «Realidad Aumentada en la Educación: una tecnología emergente,» *Escuela Superior de Ingeniería de Bilbao, EHU. Recuperado de http://bit. ly/2hpZokY,* 2007.

En la Figura 3 se observa un sistema básico de un sistema de realidad aumentada el cual la cámara captura la información del mundo real y el sistema de posicionamiento determina la posición y orientación del usuario en cada momento, con esta información se genera el escenario virtual que después se va a mezclar con la señal de video capturada para así generar el sistema. Esta escena compuesta se presenta al usuario a través del dispositivo de visualización, en la derecha de la Figura 3 se presenta un ejemplo basado en dispositivos móviles, PDA y además un visor de imagen portátil.

La tecnología de realidad aumentada actualmente está siendo desarrollada por diferentes grupos de investigación en las diversas tecnologías involucradas como, seguimiento de la posición del usuario, procesado de la señal, visualización de la información, visión por ordenador, generación de imágenes virtuales, renderizado de gráficos, estructuración de la información y computación distribuida.[2]

Existe una plataforma llamada el portal de la realidad aumentada en la dirección web aumenteg.reality.org en donde hay formación relevante sobre las tecnologías, grupos de investigación, proyectos, productos y recursos relativos a la realidad aumentada. Un aspecto importante de esta tecnología en cuanto a las aplicaciones es la tendencia a crear entornos en donde se requiere que el usuario se mueva, estas nuevas aplicaciones que se basan en la computación móvil se requiere acceder a servicios independientes del lugar o del tiempo.

Una de las aplicaciones más conocidas de la realidad aumentada en el área educacional es el proyecto llamado *magic book* del grupo de investigación HIT de

[2] X. Basogain, M. Olabe, K. Espinosa, C. Rouèche y J. C. Olabe, «Realidad Aumentada en la Educación: una tecnología emergente,» *Escuela Superior de Ingeniería de Bilbao, EHU. Recuperado de http://bit. ly/2hpZokY,* 2007.

nueva Zelanda, en donde el usuario lee un libro real a través de un visualizador en donde se muestran a medida que va pasando las paginas reales contenidos virtuales de todo tipo dependiendo de la información reflejada en el libro real, de esta manera el usuario puede introducirse dentro de la escena y experimentar en un entorno virtual.[3]

2.8. El tacto como una respuesta de interfaz de retroalimentación.

Otro tipo de inmersión dentro del mundo de la realidad virtual son las interfaces hápticas o de retroalimentación por medio del tacto.

Es el tacto lo que nos proporciona el sentido de realidad toda nuestra concepción de lo que existe fuera está basada en el sentido del tacto, en el campo de la realidad virtual, en el más amplio de la interacción persona-ordenador, la tecnología de retorno de fuerzas o en general toda aquella que proporciona una información a través del sentido del tacto no es nueva sin embargo su uso no esta tan extendido y desarrollado como otras más tradicionales que estimulan los sentidos como la vista o el oído.[8]

2.9. Guantes de datos y de retorno táctil.

Uno de los dispositivos de control y retroalimentación o háptica para escenarios virtuales son los guantes de datos y retorno táctil los cuales son dispositivos de entrada y salida, que a su vez puede ser también de salida si en este caso ofrece un retorno háptico, en algunos casos cuando se utilizan como dispositivos de entrada, se suele incorporar una serie de sensores que permiten determinar el grado de flexión de cada uno de los dedos, e igualmente se suelen combinar con dispositivos de localización para poder complementar la información sobre las posiciones relativas de cada dedo con respecto a la posición absoluta de la mano. Hay dos formas básicas de medir los gestos de la mano, una de ellas es utilizar sensores flexibles, o una estructura de segmentos articulados, la cual se fija a la mano de forma que se parezca aun exoesqueleto.

Algunos de los guantes que se encuentran comercialmente son:

---

[3] M. Billinghurst, R. Grasset, R. Green y M. Haller, «Inventing the future down under: the human interface technology laboratory new zealand [hit lab nz],» *ACM SIGGRAPH Computer Graphics,* vol. 39, pp. 18-23, 2005.

[8] J. Martínez, J. P. Molina, A. S. García, D. Martínez y P. González, «Desarrollo de un guante de datos con retorno háptico vibro-táctil basado en Arduino,» de *Interacción 2009-Jornadas de Realidad Virtual*, 2009, pp. 1-10.

2.9.1. *Fakespace pinch gloves*.

Estos guantes no tienen la capacidad de medir la flexión o abducción de los dedos, tampoco la posición de la mano, en cambio detectan el contacto entre dos o más dedos como si tuvieran pulsadores normales, esto se realiza por medio de una tela conductora que se encuentra en las yemas del guante una ventaja de estos que no es necesaria una calibración antes de que un usuario deferente lo use.

2.9.2. *Fifth dimensión data glove*.

Este guante creado por la empresa 5DT permite medir la flexión de cada dedo y también la orientación de la mano, para esta tarea emplea bucles de fibra óptica en cada articulación o dedo, y una unidad de control que incluye el circuito que realiza la medición de la orientación.

2.9.3. *Inmersion cyberglove*.

Este guante es inventado por Jim Kramer y comercializado por la empresa *Virtex* que ahora es conocida como inmersión, este dispositivo está constituido de 18 0 22 delgados medidores de esfuerzo montados sobre un fino tejido elástico de nylon.

2.9.4. *Inmersion cybertouch*.

Este guante consiste básicamente en 6 pequeños estimuladores de tipo vibro táctil que se le añaden al guante, uno en cada dedo y uno en la palma de la mano, por medio de todo el conjunto de estimuladores se logra generar desde sensaciones simples hasta complejos patrones de realimentación táctil.

2.9.5. *Inmersion cybergrasp*.

Este producto consiste en un exoesqueleto externo que se une al guante para poder proporcionar una retroalimentación de fuerzas de tipo resistivas a cada uno de los dedos, para esto aplica fuerzas individuales mediante una red de tendones guiados por la estructura del guante y se programa para que en el momento en el que el

usuario interactúe con el objeto este actúe y no permita que se atraviesen dichos objetos.

Figura 4. Guantes de datos. a) FakeSpace Pinch Gloves b) Fifth Dimension Data Glove c) Immersion CyberGlove d) Immersion CyberTouch e) Immersion CyberGrasp.

a) b) c)

d) e)

Fuente: J. Martínez, J. P. Molina, A. S. García, D. Martínez y P. González, «Desarrollo de un guante de datos con retorno háptico vibro-táctil basado en Arduino,» de *Interacción 2009-Jornadas de Realidad Virtual*, 2009, pp. 1-10.

2.10. Detección de colisiones.

Los sistemas de tele operación permiten extender las capacidades sensoriales y de destreza humana con el fin de controlar la realización de tareas de un robot que se pude llamar esclavo debido a su condición de recibimiento de ordenes en entornos remotos reales o virtuales.[10]

[10] M. L. Pinto-Salamanca, J. I. Sofrony-Esmeral y D. F. Jiménez, «Detección de colisiones con librerías V-Collide y PhysX para interacción virtual con interfaces hápticas,» *Revista de Investigación, Desarrollo e Innovación,* vol. 5, pp. 119-128, 2015.

En un sistema básico de tele operación, el operador encargado maneja un tipo de manipulador maestro para indicar las acciones que se deben ejecutar el manipulador de tipo esclavo a través de cualquier tipo de interfaz.

Un aspecto importante de la realidad virtual es el contacto con objetos dentro del escenario para esto se tiene que generar la sensación al usuario que realmente los objetos que ve están ahí, para esto se realiza un proceso llamado detección de colisiones que se maneja mediante librerías que son herramientas de software que se encargan de comprobar mediante modelos matemáticos computacionales, si los objetos que componen una escena grafica entran o no en contacto con el usuario se determina en función de las características físicas propias de cada modelo y las consecuencias que generen tal contacto.

Estos sistemas de colisiones son ampliamente utilizados en las simulaciones quirúrgicas al igual que en el campo de los videojuegos, algunas librerías para detección de colisiones que están disponibles y presentadas como paquetes de libre distribución implementadas en ANSIC O C++ son:

2.10.1. *ICollide*.

Esta fue desarrollada en la universidad de carolina del norte, tiene la capacidad de determinar con gran rapidez y exactitud las colisiones presentes en un entorno virtual que tenga muchos elementos con movimiento rígido, si en el juego existe coherencia temporal que básicamente se atribuye a que el estado de simulación no cambia significativamente entre dos pasos de simulación consecutivos, por lo que esta librería actuara de forma eficiente.

2.10.2. *V-Collide*.

Esta es una librería que como se mencionó anteriormente se encarga de detectar colisiones que fue desarrollada por el grupo GAMMMA o “Geometric algorithm for modeling, motion and animation” por sus siglas en inglés pertenecientes a la universidad de carolina del norte, esta librería está especialmente diseñada para trabajar de manera eficiente en ambientes que contienen un gran número de objetos de diferentes formas. Si se habla de su forma de trabajo esta lo hace mediante dos pasos el primero de ellos es construir “OBB´s” para cada uno de los objetos que básicamente son cuatros que delimitan el objeto y que están diseñados para verificar la interacción entre ellos lo que corresponde a la segunda etapa del proceso

mediante el cual esta librería detecta las colisiones.

2.10.3. *Rapid*.

Por otro lado, rapid aunque es una librería independiente esta se encuentra dentro v-collide pero a diferencia de la anterior es que esta no está capacitada para detectar colisiones de muchos objetos simultáneamente ya que solamente puede hacerlo con dos a la vez, igualmente por esta desventaja de solo dos objetos esta librería solo reporta si dos parejas de triángulos colisionaron exactamente, pero esto no quiere decir que sea una desventaja ya que por este motivo es utilizada para aplicaciones donde se requiera una enorme precisión en cuanto a la animación y el resultado correspondiente que el usuario este visualizando, se utiliza mucho en simuladores. Esta fue presentada por *Gottschalk, Manocha y Lin.*

2.10.4. Swift.

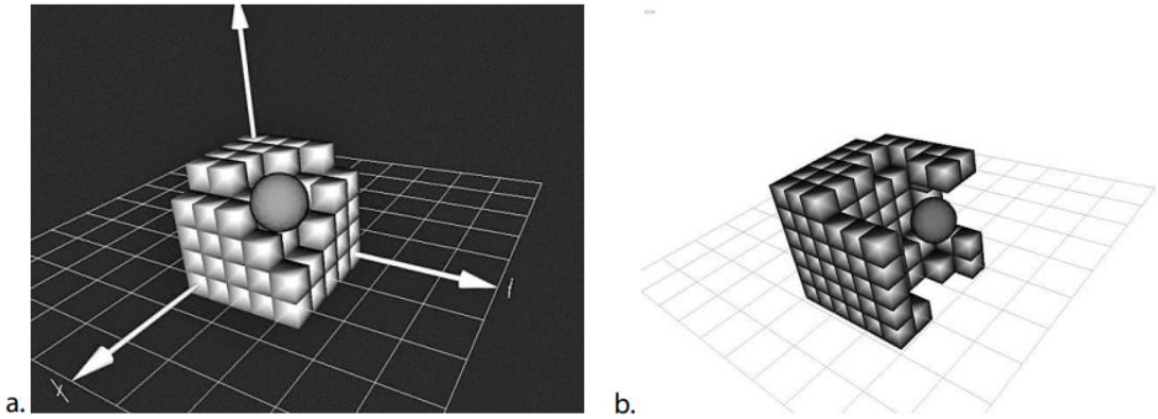
Esta librería determina contactos entre elementos poligonales en tres dimensiones que posean un movimiento rígido y también puede medir distancias, soporta geometrías cerradas y convexas y para el cálculo de distancias utiliza una jerarquía de modelos multiresolución que junto con un algoritmo de optimización hace que esta librería tenga una gran velocidad de computo, también utiliza cajas de inclusión para una primera aproximación en el cálculo de las colisiones.

Es más rápida que las otras y funciona de manera eficiente incluso cuando no hay coherencia temporal.

El objetivo principal de los algoritmos para detectar colisiones es calcular las interacciones geométricas entre los objetos, sin importar el numero ni la complejidad física que los objetos puedan tener. Normalmente estas librerías requieren una gran cantidad de pruebas de intersección geométrica verificando así todos los polígonos que modelan la superficie de un objeto, intersectando la superficie de un objeto con otro, determinando de esta manera cuando dos objetos colisionan.

La detección de colisiones en un ambiente virtual con realimentación de fuerzas, mediante el desarrollo de aplicaciones de software como herramienta de apoyo que propone una integración de un modelo mixto de tipo superficial-volumétrico, para simular el maquinado de ciertos objetos.

Figura 5. Escena virtual. A) matriz de cubos. B) interacción virtual.



Fuente: M. L. Pinto-Salamanca, J. I. Sofrony-Esmeral y D. F. Jiménez, «Detección de colisiones con librerías V-Collide y PhysX para interacción virtual con interfaces hápticas,» *Revista de Investigación, Desarrollo e Innovación,* vol. 5, pp. 119-128, 2015.

El modelo mixto para el caso de la figura 5, consiste en una matriz de cubos de tamaño variable cuyos componentes pueden aumentarse en cada eje coordenado con la posibilidad de considerar además transformaciones geométricas y eliminaciones de escena por la interacción con una herramienta virtual que representa un cursor virtual.

hay dos tipos de identificación de colisiones las cuales están definidas por:

2.10.5. Colisión delimitada por esferas.

Esta técnica consiste en encerrar los objetos en esferas, dependiendo de la máxima coordenada del objeto.[11]

La colisión se detecta con la comparación de las distancias desde el centro de las esferas, y determinando si la suma es mayor o menor.

[11] P. A. Loaiza Gómez, M. A. Vega Uribe y others, «Simulador tridimensional para el desarrollo de la habilidad de transferencia de aros para cirugía laparoscopia usando una interfaz háptica:" Silaph 3D",» 2015.

Figura 6. Modelo de colisiones por esferas.

Fuente: P. A. Loaiza Gómez, M. A. Vega Uribe y others, «Simulador tridimensional para el desarrollo de la habilidad de transferencia de aros para cirugía laparoscopia usando una interfaz háptica:" Silaph 3D",» 2015.

2.10.6. Colisiones con cajas envolventes alineadas con los ejes.

Este tipo de colisión que en inglés es llamado axis *aligned bounding bosex*, consiste en delimitar los objetos con cuadros que los encierran de forma que sean estas las que produzcan la colisión.

La colisión es presentada cuando las proyecciones de las cajas sobre los ejes de coordenadas se solapan.

2.11. Realidad virtual en el entretenimiento médico.

En la formación de un médico cirujano existen diferentes etapas, una de ellas es la etapa de entrenamiento la cual el aspirante es sometido a un proceso de adiestramiento para desarrollar habilidades mínimas y así poder proceder a aplicarlas en un paciente real.

A nivel comercial existen muchos softwares con la capacidad de generar simulaciones en el área de la medicina, pero con la desventaja de que son demasiado costosos, por esta razón se incentiva más la investigación local para desarrollar elementos que pueden contribuir a la formación médica.

La implementación de simuladores virtuales permite hacer un análisis de la actividad realizada, y sea por medio de análisis del comportamiento o mediante puntajes que la misma simulación posea, haciendo más sencillo crear niveles de dificultad, dependiendo del nivel del aprendiz.

2.12. Videojuegos como posibilidad de aprendizaje.

Todo lo que permita al ser humano entretenerse en las tareas que realiza agregan un valor que facilita la evolución cognitiva, y despierta el interés del mismo por lo que aprende y hace.

Freud afirmo que la imposibilidad de realizar los deseos crea una tensión, que se incrementa con el tiempo. Al mismo tiempo, se refiere al juego como un factor importante para reducir esta tensión.[12]

Los juegos propician la creatividad y potencian la lógica y la racionalidad, además de que estimulan la recreación y la colección utilizando contenidos entretenidos como estrategia de motivación y demandan orden absoluto y supremo. Posee espacio y tiempo propio, por lo que conlleva una dinámica particular de relacionarse con el entorno del jugador. El concepto de juego se puede traer a la actualidad, en donde predominan los aparatos electrónicos y las computadoras, los dispositivos Android y las videoconsolas, que se han convertido en un aspecto clave para el desarrollo del ser humano desde su etapa más temprana de desarrollo intelectual.

Los modelos de educación actual intentan que el alumno en cuestión desempeñe un papel activo en el proceso que compone enseñanza y aprendizaje, con el fin de desarrollar sus competencias permitiendo estar altamente calificado y competente para asumir los retos que las nuevas sociedades plantean.

El componente lúdico en los últimos años se le ha prestado especial atención ya que al presentarse esta manera de aprendizaje se enriquece creando un espacio dinámico y virtual que transforma el entorno e incluso la forma de interactuar con este.

Este tipo de aprendizaje está basado principalmente por el juego mediante este el estudiante comienza a pensar y actuar en medio de una situación determinada que fue construida con semejanza en la realidad, con un propósito pedagógico. [11]

Un concepto que se viene manejando en los videojuegos en donde se enfoca como recurso didáctico es el “edutainment” o “edujuegos” que tienen la intención de aprender, entretener y divertir al usuario.

---

[11] P. A. Loaiza Gómez, M. A. Vega Uribe y others, «Simulador tridimensional para el desarrollo de la habilidad de transferencia de aros para cirugía laparoscopia usando una interfaz háptica:" Silaph 3D",» 2015.

[12] G. Méndez, E. Obviedo, G. Fallas, C. Vega y A. Méndez, «Análisis de las herramientas Unity y Blender para el desarrollo de videojuegos con un enfoque educativo,» Escuela de Computación, Tecnológico de Costa Rica, p. 13, 2014.

2.13. Unity y Blender para el desarrollo de videojuegos.

Estos softwares son utilizados para crear videojuegos que pueden ser ejecutados en varias plataformas como computadores dispositivos móviles entre otros. Por una parte, la herramienta Blender permite modelar, texturizar, y hacer *rigging* que es un proceso en el que se le da una estructura similar a los huesos a un modelo, de manera que se pueda controlar la cinemática y darle movimiento de forma gradual, funcionas las cuales son importantes para la creación de personajes y estructuras en tres dimensiones de diferentes naturalezas.

2.13.1. Blender.

En este programa se pueden crear personajes, darles textura y movimiento, e incluso programarlos para que tengan un determinado comportamiento lógico. Además, la interfaz gráfica permite personalizar diferentes formas y manejar todas esas formas al mismo tiempo. Un ejemplo de esta ventaja es que se pueden agrupar todas las ventanas y menús que sirven para la textura de los personajes u objetos diseñados y guardarlas con un nombre determinado por el usuario.

Una desventaja de Blender es que por ser un software libre no permite exportar objetos a consolas conocidas comercialmente ya que estas solo aceptas formato con licencia de tipo GPL ya que este solo tiene la capacidad de exportar a formatos de tipo SDK.

2.13.2. Unity.

El software Unity permite conectar a los objetos 3D y de otro tipo como cámaras, luces, fuerzas de viento, agua y demás componentes para crear un escenario virtual optimo, en cuanto a la programación este se maneja por medio del lenguaje *unityscript*, además permite agrupar los objetos tridimensionales para crear estructuras cuyos elementos que conforman el espacio virtual puedan interactuar de forma colectiva. También permite crear escenarios en tres dimensiones, mezclando elementos gráficos con programación, una ventaja de este software es que trae scripts ya prediseñados que dentro de la interfaz se llaman *assets* los cuales traen diferentes componentes que ayudan a la elaboración de determinadas escenas dentro de espacio virtual.

2.13.3. Aspectos matemáticos y de computación en Unity y Blender.

Antes de desarrollar un videojuego se requiere de un cierto nivel de conocimiento de computación, en primer lugar, las representaciones de los objetos que componen un videojuego dentro del editor 3D se traducen en realidad como transformaciones en las coordenadas del espacio cartesiano de tres dimensiones a un plano sin profundidad o de dos dimensiones. En palabras de algebra lineal para el caso anterior se conoce como proyección el cual es una de las transformaciones lineales junto con el escalamiento y la translación.

Para una programación adecuada en donde se realizan modificaciones de objetos tridimensionales se pueden utilizar composiciones de matrices de transformación, básicamente se realiza una transformación y a esta aplicarle otra y así sucesivamente se aplican transformaciones lineales logrando así movimientos de los objetos sin que estos lleguen a deformarse, igualmente se trabaja con transformaciones graficas de los vértices aristas o caras de las figuras tridimensionales.

Por otra parte, existen otros aspectos en el videojuego que requieren programación, tales como algoritmos de inteligencia artificial que conllevan a que ocurran eventos inesperados que logren una respuesta por parte del jugador.

Uno de los algoritmos más utilizados en los videojuegos es el llamado *minimax* el cual tiene una función básica que es la de minimizar el daño y maximizar la oportunidad de ganar la partida que dependerá del tipo de situación e implementación.

2.14. Breve repaso por la robótica industrial y los tipos tradicionales que existen actualmente.

Una definición del robot industrial que pertenece a la asociación de industria de robótica lo define como un manipulador multifuncional reprogramable, capaz de mover materias, piezas, herramientas o dispositivos especiales, según trayectorias variables, programadas para realizar tareas diversas.[23]

---

[23] D. Blanco, S. Al Ansari, C. Castejón Sisamón, B. López Boada y L. E. Moreno, «Manfred: robot antropomórfico de servicio fiable y seguro para operar en entornos humanos,» 2005.

Se puede decir que una máquina o un mecanismo articulado entre sí, el cual tiene 3 distintivos esenciales:

- Es multifuncional.
- Puede ser controlado por un operador humano o un dispositivo lógico.
- Es reprogramable.

Un robot industrial está conformado por cuatro características principales las cuales se componen por tener un brazo mecánico con capacidad de manipulación, el cual puede ser controlado, también se componen de elementos estructurales rígidos, llamados eslabones o enlaces que son conectados por articulaciones, las cuales pueden ser lineales o rotatorias, que a su vez terminan en puntos terminales o también llamados manipuladores los cuales pueden ser pinzas o herramientas.

2.14.1. Tipos de robots industriales.

Existen pautas las cuales conllevan diferenciar los tipos de robots industriales los cuales se dividen en 6 parámetros que se definen de la siguiente manera:

- Grados de libertad o la suma de las articulaciones que lo componen.
- Espacio de accesibilidad que corresponde al número de puntos accesibles al punto terminal, dependiendo de la configuración geométrica.
- Capacidad de posicionamiento el cual mide el grado de exactitud de los movimientos en una tarea programada.
- Capacidad de carga o el peso que el robot puede transportar.
- Velocidad que corresponde a la velocidad máxima que ´puede alcanzar.

Ahora se verá un repaso a los tipos de robots industriales que hay definiendo cada uno de ellos teniendo en cuenta que en el proyecto se va a utilizar los modelos de robots llamados scara, antropomórfico y delta.

2.14.2. Robot industrial cartesiano.

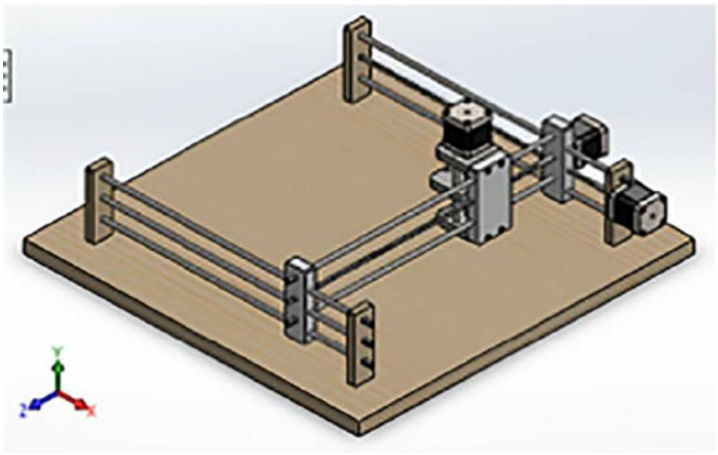
Esencialmente los robots industriales cartesianos se distinguen por posicionarse mediante 3 articulaciones lineales, generando movimientos perpendiculares de acuerdo con los 3 ejes cartesianos X, Y, Z.[25]

Este tipo de robots trabajan mediante el sistema de coordenadas propuestas que trazan los puntos donde debe realizar el movimiento. El ordenador por medio de un programa es el que controla los movimientos y a su vez los optimiza de tal manera que el robot trabaje de una manera eficiente.

En la mayoría de casos los movimientos lineales se realizan en los ejes por servomotores y dependiendo del tipo de movimiento serán operados por el eje portal o por el eje de extensión o por el eje telescópico.

Entre las ventajas que poseen estos robots están el buen nivel de precisión y repetitividad, facilidad de programación y costo económico relativamente bajo ya que puede ser considerado como la solución de menor costo para la industria de la soldadura, pues puede realizar operaciones simples como soldar, colocar o escoger de forma eficiente y barata.

Figura 7. Robot cartesiano.



Fuente: J. Berrio, E. Arcos, J. Zuluaga y S. Corredor, «Diseño y construcción de un robot cartesiano de 3 grados de libertad,» *Memorias,* 2015.

---

[25] J. Berrio, E. Arcos, J. Zuluaga y S. Corredor, «Diseño y construcción de un robot cartesiano de 3 grados de libertad,» *Memorias,* 2015.

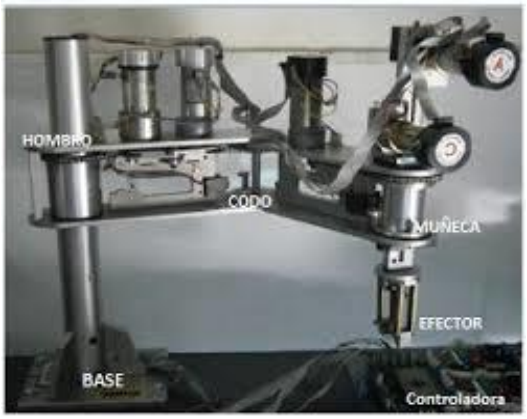
2.14.3. Robot industrial scara.

Aunque este tipo de robot se desplaza bajo las mismas condiciones que el robot cartesiano ya que sus movimientos obedecen al plano cartesiano, este se diferencia por incorporar un eje final del plano Z para hacer girar la herramienta o pinza al final del brazo robótico.

Comúnmente estos robots se usan principalmente para procesos de ensamblaje, aunque no son tan universales, puesto que la terminación del brazo limita su alcance.[26]

El nombre de estos robots significa por la traducción de sus siglas en ingles brazo de robot de montaje selectivamente adaptable. Este tipo de robots, imitan la forma humana, lo que les permite realizar diversos movimientos en todas direcciones para realizar una amplia variedad de tareas en los procesos productivos de numerosas industrias. Está compuesto por cuatro grados de libertad con posicionamiento horizontal y son conocidos por sus ciclos rápidos de trabajo, excelente repetitividad, gran capacidad de carga y su amplio campo de aplicación, estos son generalmente más rápidos y sencillos que los sistemas comparables de robots cartesianos, comúnmente se montan en un pedestal simple que requiere de una pequeña superficie ocupada y proporciona una fácil forma de montaje, además tienen la desventaja de que están limitados en gran medida en sus desplazamientos en el eje z.

Figura 8. Robot scara.



Fuente: C. Pillajo y J. E. Sierra, «Human Machine Interface HMI using Kinect sensor to control a SCARA Robot,» de *2013 IEEE Colombian Conference on Communications and Computing (COLCOM)*, 2013.

[26] C. Pillajo y J. E. Sierra, «Human Machine Interface HMI using Kinect sensor to control a SCARA Robot,» de *2013 IEEE Colombian Conference on Communications and Computing (COLCOM)*, 2013.

Estos robots son ideales para procesos en los que se manejan pequeños productos y al mismo tiempo se demanda velocidad y precisión en los movimientos.

2.14.4. Robot industrial antropomórfico.

Es una maquina concebida para el manipulado de todo tipo de elementos, destacada por su gran polivalencia.[28]

Este tipo de robot antropomórfico puede programarse para manipular cualquier tipo de elemento, para ellos se diseña y fabrica garras que permiten el manipulado de cajas, garrafas, botellas, sacos o cualquier elemento. Además de la programación del propio robot, en función del tipo de elemento a manipular se integran en la celda toda una serie de maquina auxiliar, como pueden ser movimientos, formadoras de botellas, embudos entre otros.

Este robot también es llamado manipulador de codo que generalmente consiste en dos articulaciones tipo hombre una de ellas para girar sobre un eje vertical y otra para elevarse a la articulación de elevación del hombro y dos o tres articulaciones de muñeca en el extremo del manipulador. Tienen la característica que están conformados por cuerpo, brazo, muñeca y efector final, a este último se le conoce comúnmente como pinza.

Entre las aplicaciones estos se utilizan principalmente donde es necesario el uso de máquinas para no poner en peligro al personal, se encargan de tareas pesadas y repetitivas.

[28] A. Rodríguez Bermejo, «Desarrollo y programación de un brazo antropomórfico para su uso en centros de tratamiento de residuos,» 2013.

Figura 9. Robot antropomórfico.

Fuente: A. Rodríguez Bermejo, «Desarrollo y programación de un brazo antropomórfico para su uso en centros de tratamiento de residuos,» 2013.

2.14.5. Robot tipo delta.

Es un tipo de robot paralelo que posee tres grados de libertad conformado por dos bases unidas por tres cadenas cinemáticas basadas en el uso de paralelogramos, la base superior se encuentra fija mientras que la base inferior, donde están ubicados el efector final es móvil y siempre esta paralela a la base fija. Para su construcción se pueden utilizar actuadores rotacionales o lineales según la aplicación para la cual quiera usarse.

Estos son muy utilizados en los procesos industriales de selección y clasificación de productos, dada su rapidez, precisión y manejo de carga.[29]

Entre sus características principales es que están compuestos por una base fija y otra móvil, unidas entre sí por un conjunto de tres brazos articulados y actuadas cada una por un motor, dando como resultado que el efector final compuesto por la base móvil, pueda posicionarse y orientarse a discreción del control que se realice sobre el robot.

[29] J. Serracín, I. Moreno, T. Vásquez y I. Bonilla, «Prototipo de Robot Paralelo Delta para fortalecer el proceso educativo a nivel superior,» de *Memorias de Congresos UTP*, 2017.

Figura 10. Robot delta.

Fuente: J. Serracín, I. Moreno, T. Vásquez y I. Bonilla, «Prototipo de Robot Paralelo Delta para fortalecer el proceso educativo a nivel superior,» de *Memorias de Congresos UTP*, 2017.

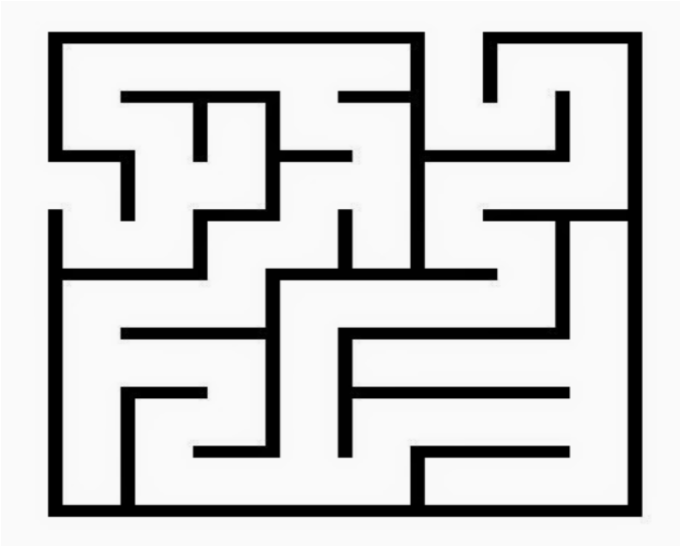
.

3. Desarrollo del proyecto.

Como primer paso a seguir para desarrollar el proyecto y teniendo en cuenta cada uno de los objetivos planteados se procede a cumplir con la primera tarea que consiste en realizar el diseño del escenario virtual, que en este caso se elabora a partir de un dibujo en dos dimensiones de un laberinto sencillo para luego ser transformado en el entorno virtual requerido en tres dimensiones.

3.1. Diseño del entorno virtual.

Figura 11. Plantilla del laberinto a diseñar.



Fuente: https://rosafernandezsalamancaprimaria.blogspot.com/2014/02/laberintos-fichas-para-trabajar-la.html

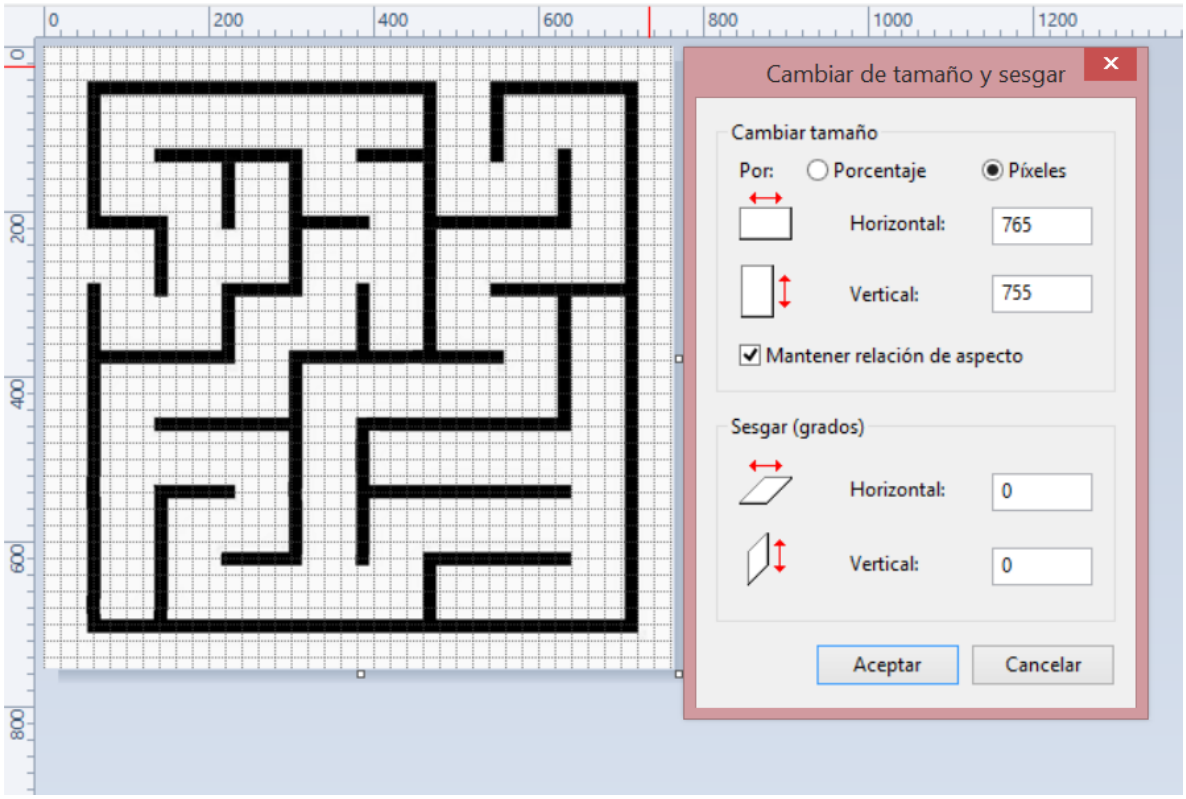
Con el laberinto de la figura 11 se plantea el diseño en el software para esto se debe tener en cuenta que el espacio total de juego será de 100 unidades por 100 unidades, estas dimensiones corresponden a las medidas que utiliza el programa para poder declarar un espacio estándar de juego, es decir, que aproximadamente

.

cada unidad en el espacio virtual creado equivale a un metro en la vida real, con este dato se procede a escalar el laberinto de la figura 11.

Antes de empezar a diseñar directamente en unity se tiene que tener una guía en cuanto a las mediciones de cada una de las líneas que componen el laberinto, para esto se debe realizar un escalamiento de la imagen, en ese caso se optó por utilizar el número de pixeles que posee la figura tanto en ancho como en el largo de esta, así como se observa en la figura 12.

Figura 12. Tamaño en pixeles de la imagen.



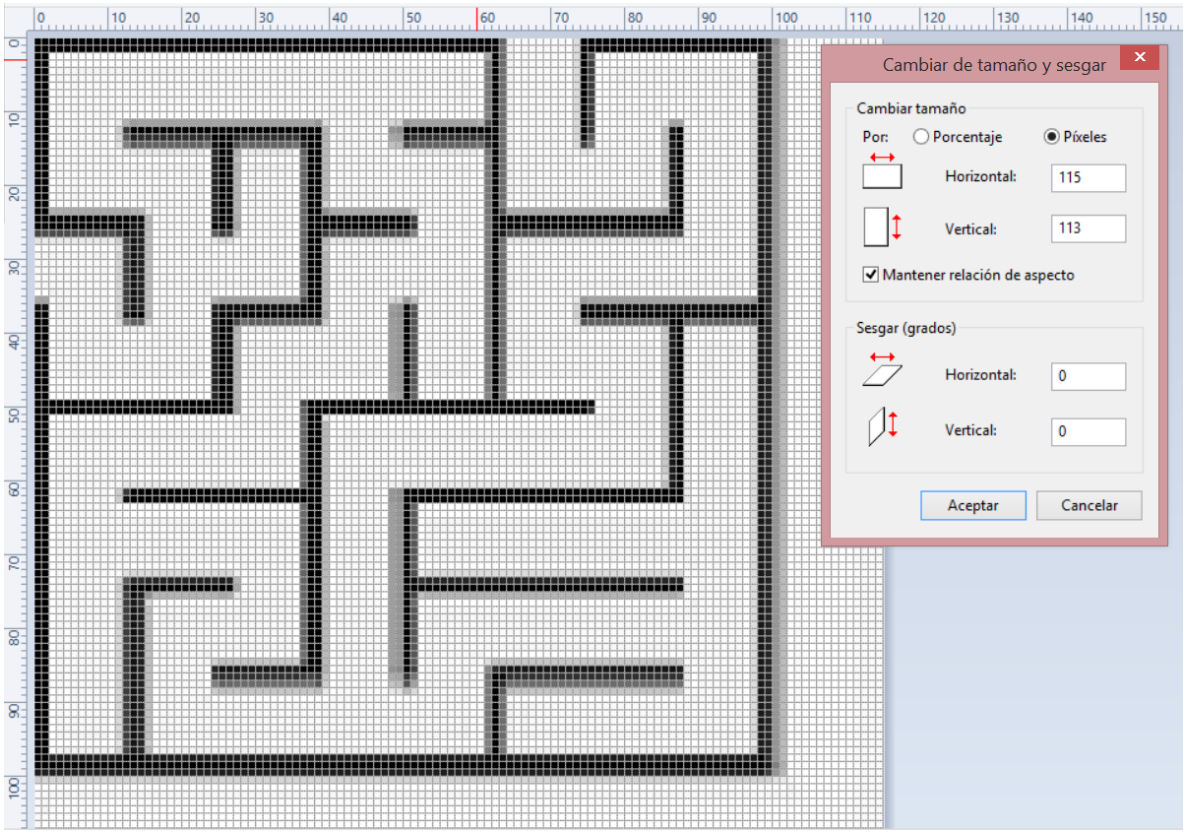
Fuente: [Autor]

Según los datos que aporta la herramienta para cambiar el tamaño de las imágenes se obtiene el tamaño correspondiente en pixeles, pero el problema de basarse en

.

esta medida para realizar el diseño del laberinto en Unity radica en el momento de escalar cada unidad de pixeles a metros según el número correspondiente, en este caso no se tendría un cálculo exacto del tamaño de los pasillos, por lo que se optó por definir el tamaño de los pasillos de extremo a extremo 100 pixeles o 10 metros en ambos lados, de esta manera se obtiene la medición de los pasillos y grosor de las paredes para luego llevarlos a las unidades que requiere el software de diseño.

Figura 13.Tamaño escalado en pixeles.



Fuente: [Autor]

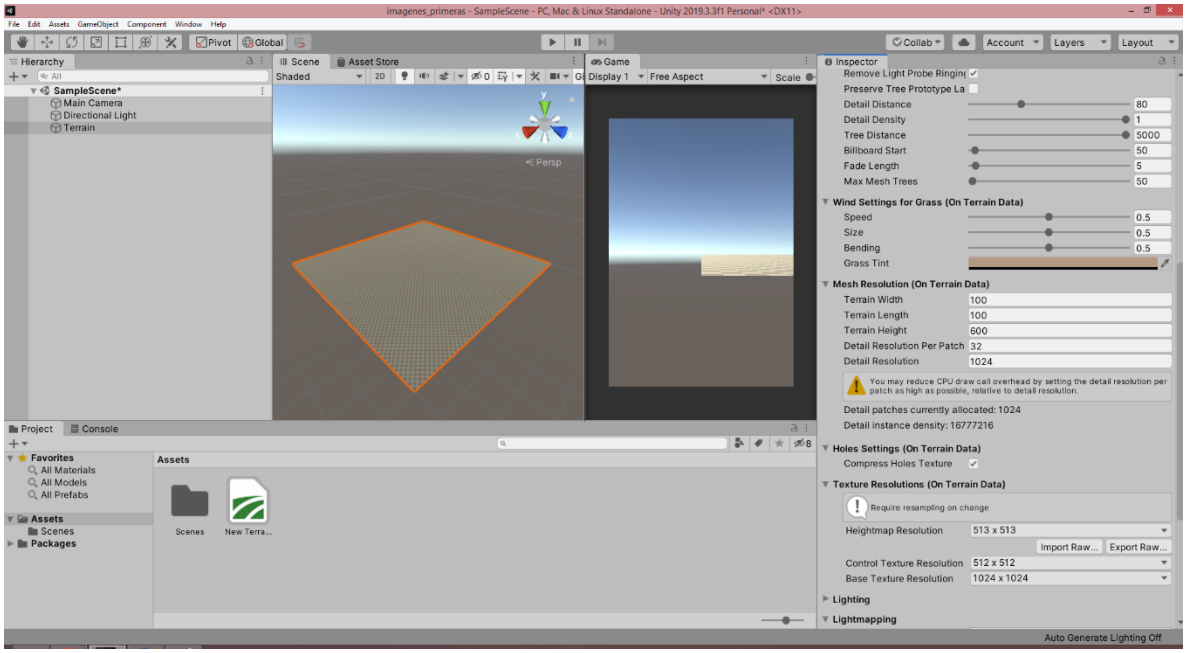
En la figura 13 se observa el tamaño escalado de la imagen según las mediciones de los pixeles se obtiene que para cada pasillo se tendrían aproximadamente 10 pixeles lo que se traduce en 10 metros de la vida real y 10 unidades en Unity, así mismo para las paredes ya que tendrían dos unidades de espesor.

.

3.1.1. Diseño en Unity.

A partir de estos valores se inicia con el diseño del escenario para esto se ingresa a unity creando un nuevo proyecto y se añade un terreno el cual se le modifica el tamaño a 100 unidades en cada lado.

Figura 14. Tamaño del terreno.

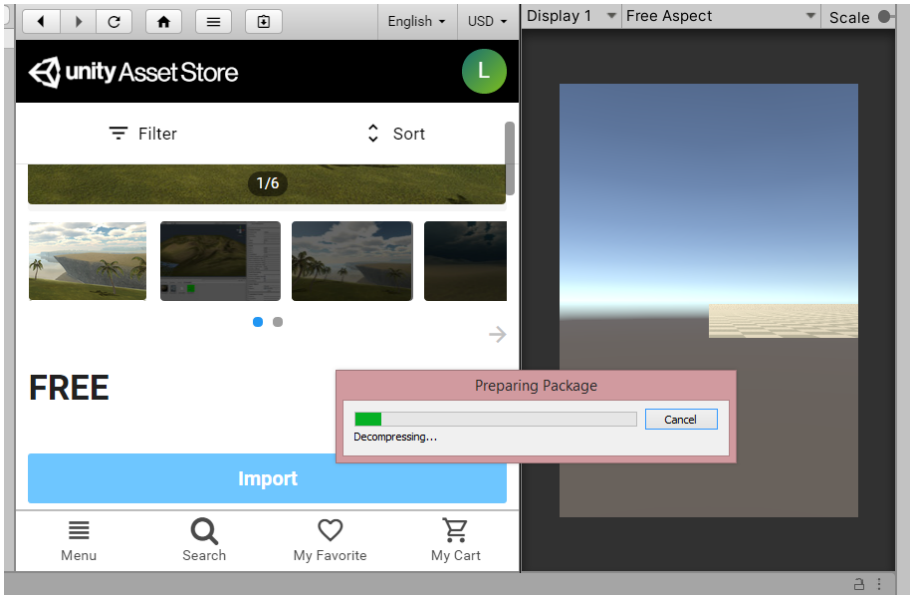


Fuente: [Autor]

Con el terreno establecido en el espacio de trabajo se importa una apariencia la cual le dará un mejor aspecto visual, así que el paquete llamado *tiny materials* se descarga de la tienda directamente de Unity la cual ofrece una serie de apariencias gratuitas para utilizar en el proyecto. Igualmente, dentro de la configuración del terreno se debe activar el componente *collider* que básicamente es el encargado de simular las colisiones de los objetos dependiendo de la forma que estos tienen como se observa en las figuras 15 y 16.

.

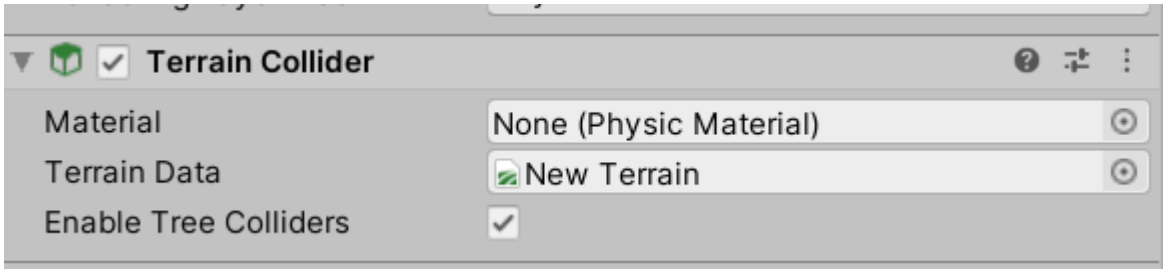
Figura 15. Importando los paquetes de terreno.



Fuente: [Autor]

En la figura 15 se observa el proceso de importación del paquete que corresponde a las apariencias que se agregan al terreno anteriormente configurado.

Figura 16. Librería de colisiones.



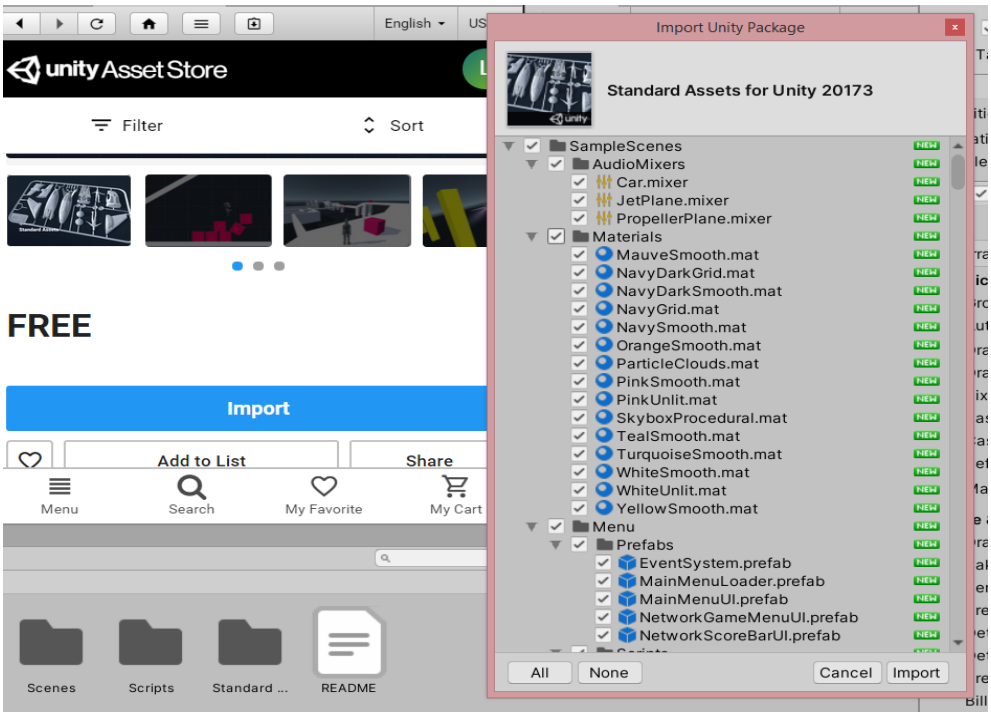
Fuente: [Autor]

A continuación, se procede a configurar el método con el que el usuario se va a mover y mirar dentro del espacio virtual para esto se tomó la decisión de realizarlo con la vista en primera persona para lograrlo se debe importar el paquete necesario el cual es llamado *characters* que pertenecen a un grupo de paquetes más grande

.

llamado *standard assets*, en versiones anteriores del programa no hay necesidad de agregarlos ya que viene implícito en la instalación, ya que es el caso de una versión reciente tratándose de la 2019.3.3f1 hay que importarlos desde la tienda.

Figura 17. Importando los paquetes estándar.

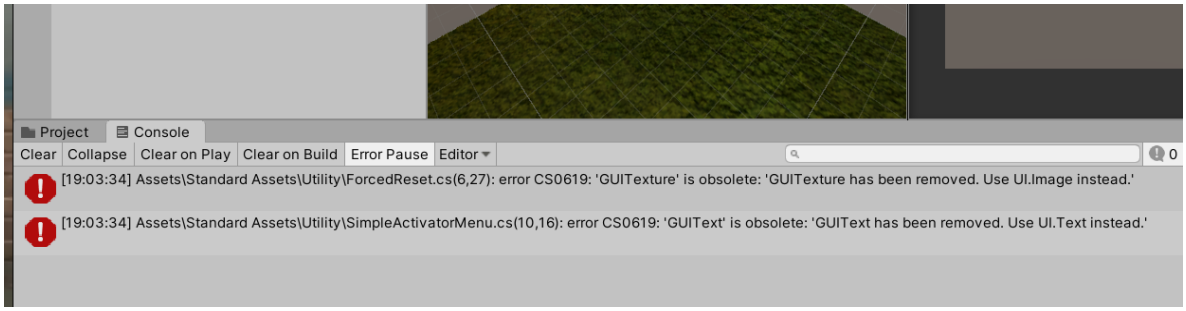


Fuente: [Autor]

En la figura 17 se contemplan todas las herramientas que se van a agregar al escenario como controladores en primera y tercera persona, objetos entre otros.

.

Figura 18. Error dado después de importar el paquete.



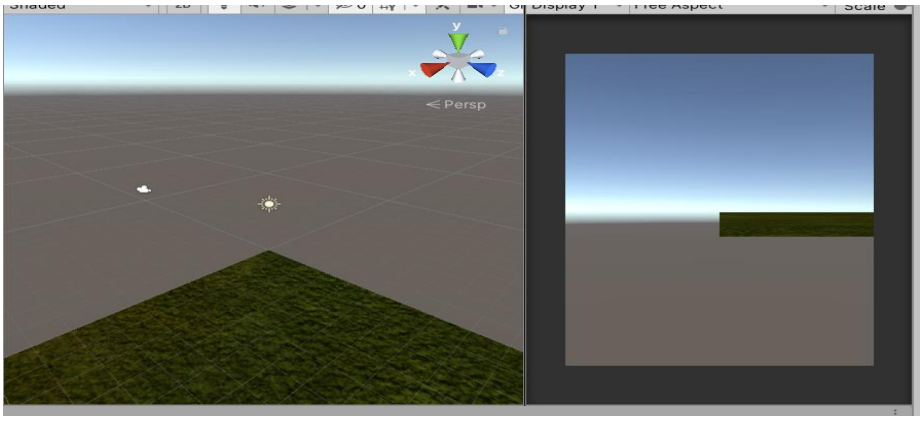
Fuente: [Autor]

En la figura 18 se observa un error que aparece después de intentar utilizar el paquete, debido a que este fue creado principalmente para la versión 2017 y las librerías ya se encuentran obsoletas se hace necesario cambiarlas por las nuevas, en el caso de *GUITexture* y *GUIText* se deben cambiar en los scripts que las manejan para evitar que este error impida utilizar la herramienta.

Como se mencionó anteriormente se busca la carpeta llamada *characters*

en la cual se busca el controlador de primera persona en donde están incluidos tanto el control de la mirada del jugador como el control de movimiento dentro del entorno, pero antes de realizar esto se debe eliminar la cámara principal que aparece en el momento en el que se crea el proyecto para así no crear conflictos con o errores con controlador en primera persona.

Figura 19. Posición errónea de la cámara principal.

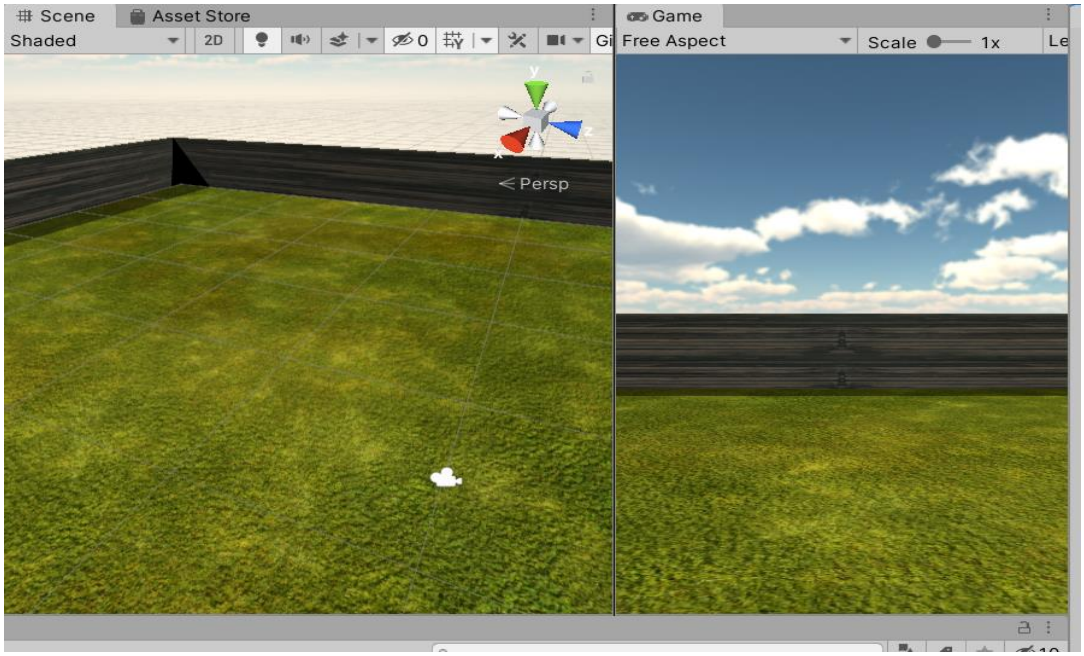


Fuente: [Autor]

.

En la figura 19 se observa la posición de la cámara principal que aparece a partir de agregar el controlador para solucionar esto simplemente se cambia de posición el terreno logrando que la cámara quede dentro del espacio establecido que conformara el laberinto. Además, se le añadirán los muros exteriores para evitar que el usuario pueda caerse al vacío, hay que activar la librería de colisiones en estos muros ya que sin ella el jugador los atravesaría, igualmente se le importaran apariencias, estos cambios se observan en la figura 20.

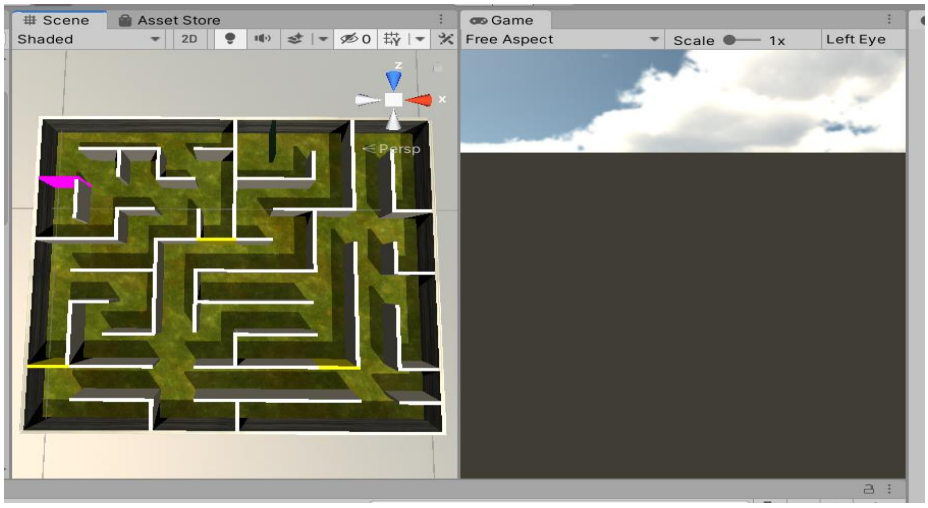
Figura 20. Muros exteriores y posición nueva de la cámara.



Fuente: [Autor]

.

Figura 21. Diseño final del laberinto.



Fuente: [Autor]

En la figura 21 se observa el resultado final del diseño de laberinto, finalmente se dejaron los pasillos de menor tamaño para así lograr que el laberinto fuera un poco más grande ya que el dibujo inicial era sencillo con lo cual se optó por aumentar la dificultad. El tamaño máximo que se pudo lograr fue de 20 unidades a cada lado.

El punto de color rosa a la derecha de la figura 21 es la salida del jugador, en este lugar está posicionada la cámara principal, donde los muros están pintados de amarillo son los tres lugares donde estarán los robots con sus respectivas animaciones y el bloque que se observa de color negro es el final del laberinto o la salida de este, a este conjunto de muros también se le agregara apariencia.

3.1.2. Controlador en primera persona.

Hay dos aspectos fundamentales que deben estar implícitos en este proyecto los cuales están ligados con el concepto de inmersión en un espacio virtual, estos implican que el usuario tenga la capacidad de ver libremente todo el escenario realizando movimientos de la cabeza, para así intentar simular la sensación de espacio que experimenta cualquier persona en la vida real ya que una persona puede ver el lugar que desee y así identificar a detalle los objetos que se encuentran

.

en su entorno además de generar la acción de avance que igualmente es una característica importante debido a que permite la correcta exploración del mundo virtual en el que el jugador se encuentra.

El encargado de generar este avance y el movimiento de la vista del jugador dentro de Unity es el controlador en primera persona que posee varias herramientas como scripts que básicamente son códigos de programación que almacenan los datos provenientes del mando inalámbrico y los transforman en los movimientos que posteriormente realiza la cámara principal, dentro de estos scripts también vienen incluidos varios parámetros que componen la raíz de funcionamiento del videojuego ya sea para las colisiones entre los objetos presentes o para generar la apariencia de estos.

También están los *Prefabs* que son *Assets* típicos pero que se pueden modificar de muchas maneras ya sea agregando componentes y ajustando completamente sus propiedades, para el caso del proyecto se tiene un *Prefab* el cual posee *Scripts* y componentes que realizan el enlace entre el mando inalámbrico, el dispositivo móvil y la aplicación además también organiza los controles para poder moverse correctamente dentro del escenario por medio del control que se va a utilizar.

```
using System;
using UnityEngine;
using UnityStandardAssets.CrossPlatformInput.PlatformSpecific;

namespace UnityStandardAssets.CrossPlatformInput
{
	public static class CrossPlatformInputManager
	{
		public enum ActiveInputMethod
		{
			Hardware,
			Touch
		}


		private static VirtualInput activeInput;

		private static VirtualInput s_TouchInput;
		private static VirtualInput s_HardwareInput;


		static CrossPlatformInputManager()
		{
			s_TouchInput = new MobileInput();
			s_HardwareInput = new StandaloneInput();
#if MOBILE_INPUT
		    activeInput = s_TouchInput;
#else



.
```

```
                activeInput = s_HardwareInput;
#endif
        }

public static void SwitchActiveInputMethod(ActiveInputMethod activeInputMethod)
        {
                switch (activeInputMethod)
                {
                        case ActiveInputMethod.Hardware:
                                activeInput = s_HardwareInput;
                                break;

                        case ActiveInputMethod.Touch:
                                activeInput = s_TouchInput;
                                break;
                }
        }
```

Teniendo en cuenta el texto del código anterior en donde se referencia la primera parte del código que realiza la comunicación entre el dispositivo móvil y el mando que controla los movimientos dentro del videojuego, este script además contiene dos formas de obtener la entrada de datos por parte del jugador ya que al declarar las variables *hardware* y *touch*, se está definiendo los métodos de entrada disponibles que también están predeterminados dentro del sistema operativo Android, la variable hardware define que el control será por medio dispositivos externos como controles inalámbricos o por cable USB, por otra parte la declaración *touch* establece que la pantalla táctil del celular será la que va a controlar la interacción con la aplicación.

```
        public static bool AxisExists(string name)
        {
                return activeInput.AxisExists(name);
        }

        public static bool ButtonExists(string name)
        {
                return activeInput.ButtonExists(name);
        }

        public static void RegisterVirtualAxis(VirtualAxis axis)
        {
                activeInput.RegisterVirtualAxis(axis);
        }


        public static void RegisterVirtualButton(VirtualButton button)
        {
                activeInput.RegisterVirtualButton(button);
```

.

```
        }


        public static void UnRegisterVirtualAxis(string name)
        {
            if (name == null)
            {
                throw new ArgumentNullException("name");
            }
            activeInput.UnRegisterVirtualAxis(name);
        }


        public static void UnRegisterVirtualButton(string name)
        {
            activeInput.UnRegisterVirtualButton(name);
        }


// returns a reference to a named virtual axis if it exists otherwise null
        public static VirtualAxis VirtualAxisReference(string name)
        {
            return activeInput.VirtualAxisReference(name);
        }


        // returns the platform appropriate axis for the given name
        public static float GetAxis(string name)
        {
            return GetAxis(name, false);
        }
```

Continuando el código.

```
        public static Vector3 mousePosition
        {
            get { return activeInput.MousePosition(); }
        }


        public static void SetVirtualMousePositionX(float f)
        {
            activeInput.SetVirtualMousePositionX(f);
        }


        public static void SetVirtualMousePositionY(float f)
        {
            activeInput.SetVirtualMousePositionY(f);
        }
```

.

```
        public static void SetVirtualMousePositionZ(float f)
        {
            activeInput.SetVirtualMousePositionZ(f);
        }


// virtual axis and button classes - applies to mobile input
// Can be mapped to touch joysticks, tilt, gyro, etc, depending on desired
implementation.
// Could also be implemented by other input devices - kinect, electronic sensors,
etc
        public class VirtualAxis
        {
            public string name { get; private set; }
            private float m_Value;
            public bool matchWithInputManager { get; private set; }


            public VirtualAxis(string name)
                : this(name, true)
            {
            }


            public VirtualAxis(string name, bool matchToInputSettings)
            {
                this.name = name;
                matchWithInputManager = matchToInputSettings;
            }
```

Las anteriores líneas de código básicamente realizan un mapeo de datos que corresponde con el tipo de entrada que tiene el mando inalámbrico o la pantalla táctil, es decir, que al mover un joystick hacia determinada posición este genera un vector con una serie de datos que son normalizados y a su vez traducidos a los parámetros que maneja el software Unity como velocidad de avance, fuerza de salto que es un parámetro que aumenta o disminuye la altura del salto que puede realizar el jugador entre otros.

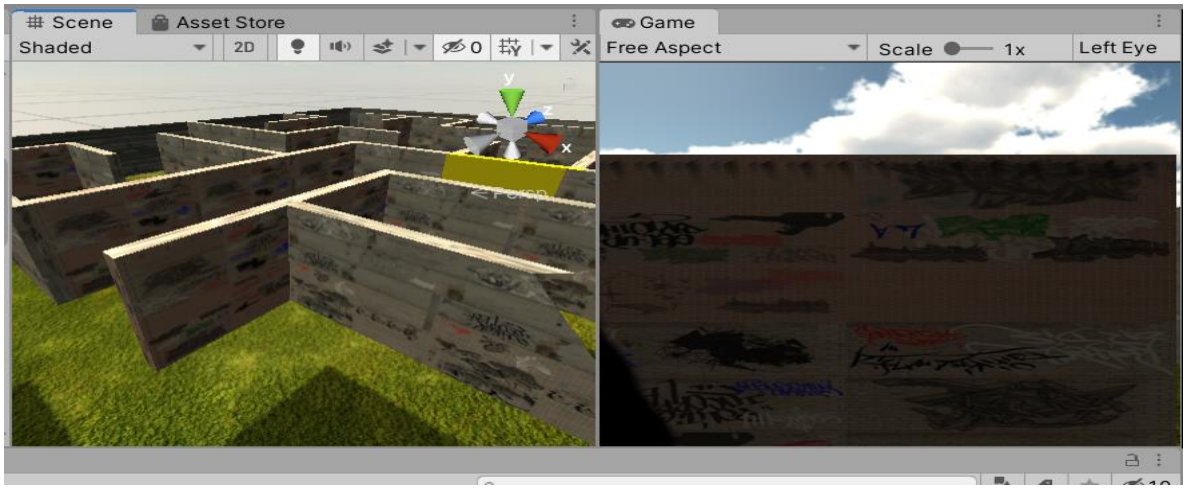
El control de movimiento y mirada dentro de Unity para verificar su correcto funcionamiento se realiza por medio del mouse y el teclado asignando las diferentes teclas mediante el input manager que es la herramienta que utiliza el software para realizar el cambio de controles y demás, pero en el momento que se exporta esta aplicación al dispositivo es necesario implementar un método para lograr el enlace de controles con el videojuego, para habilitar el manejo por medio de dispositivos externos se hace mediante la creación de ejes y botones virtuales los cuales específicamente imitan los movimientos que realiza el mouse dependiendo de los

.

datos generados por el mando utilizado, con la herramienta antes mencionada input manager esta habilita la entrada de datos proveniente del control y la asigna al mouse y los botones virtuales, además se le asignan los diferentes tipos de movimientos como adelante, atrás, derecha e izquierda.

El resultado final del laberinto con las apariencias incluidas se observa en la figura 22.

Figura 22. Resultado final del laberinto.



Fuente: [Autor]

3.1.3. Aplicabilidad del entorno virtual en usuarios.

Después de terminado el diseño inicial del escenario virtual se procede a realizar pruebas con diferentes usuarios para así tener una retroalimentación de este, recibiendo sugerencias, recomendaciones y comentarios acerca de las experiencias que tuvieron al jugarlo, las pruebas se realizaron con dos usuarios adultos.

.

3.1.3.1. Usuario 1.

Datos personales:

Edad: 32 años.

Profesión: Militar.

Experiencia: El usuario refiere que: “El videojuego es muy entretenido ya que anteriormente no había probado la experiencia de la realidad virtual solamente lo había hecho viendo películas con el dispositivo móvil que tiene, afirma que el diseño visual del laberinto es agradable y que los controles son intuitivos y se acomodan a la necesidad de las mecánicas del juego, después de algunos minutos y ciertas indicaciones para que lograra correctamente llegar al final del laberinto este comenta que la dificultad del juego en si es alta debido a las condiciones actuales de diseño el cual dificulta en gran medida la orientación dentro del escenario vista de que no se llega diferenciar la ubicación actual ni tampoco saber si ya se pasado por el mismo lugar, este usuario recomienda que al inicio del juego se pueda ver un mapa del laberinto para así tener mejor referencia y orientación además de tener pistas en forma de incentivo para que el jugador sepa que va en la dirección correcta”.

Resultado: Realmente es buena la idea del mapa al inicio del laberinto ya que anteriormente se había pensado en implementarlo para una mayor facilidad de resolver el laberinto, en cuanto a las pistas incluidas dentro del laberinto esta sugerencia se puede aplicar con los robots que se van a animar, pero la diferencia es que en el momento que la animación termine se dará una dirección a la cual seguir que le dé al jugador una idea acertada para continuar con la resolución del laberinto.

.

Figura 23. Prueba con el usuario 1.

Fuente: [Autor]

Figura 24. Explicación del funcionamiento del juego.

Fuente: [Autor]

.

3.1.3.2. Usuario 2.

Datos personales:

Edad: 28 años.

Profesión: Psicóloga.

Experiencia: El usuario refiere que: “este usuario ya había experimentado antes videojuegos de realidad virtual por medio de su teléfono celular, pero dice que hasta ahora no había probado ninguno que tuviera movimiento por medio de un control inalámbrico solamente había jugado aquel tipo de juegos en los que el modo solamente se limita a visualizar un escenario que se mueve alrededor de él, este cambio de perspectiva es interesante por su idea de poder moverse en cualquier dirección que se desee él dice que la inmersión dentro del espacio virtual es mucho más profunda debido a este detalle de movimiento. También afirma que sería más fácil que dentro del laberinto deberían existir pistas las cuales ayudaran a la resolución de este y que dieran una mayor perspectiva de orientación y ubicación.”

Resultado: Dos usuarios que probaron el videojuego tienen la misma sugerencia a seguir, el diseño visual del laberinto requiere que existan pistas para así brindar mejores referencias mientras que se va resolviendo el laberinto además de mostrar al usuario un mapa al inicio del juego en el momento en el que están explicando las instrucciones. Además de los robots que se van a animar para dar las pistas dentro de este, se realizaran imágenes alusivas al laberinto los cuales proporcionen la parte del mapa que el jugador lleve en ese momento descubierta para así evitar que el usuario este dando vueltas por el mismo lugar sin que este sepa.

.

Figura 25. Prueba con el usuario 2.

Fuente: [Autor]

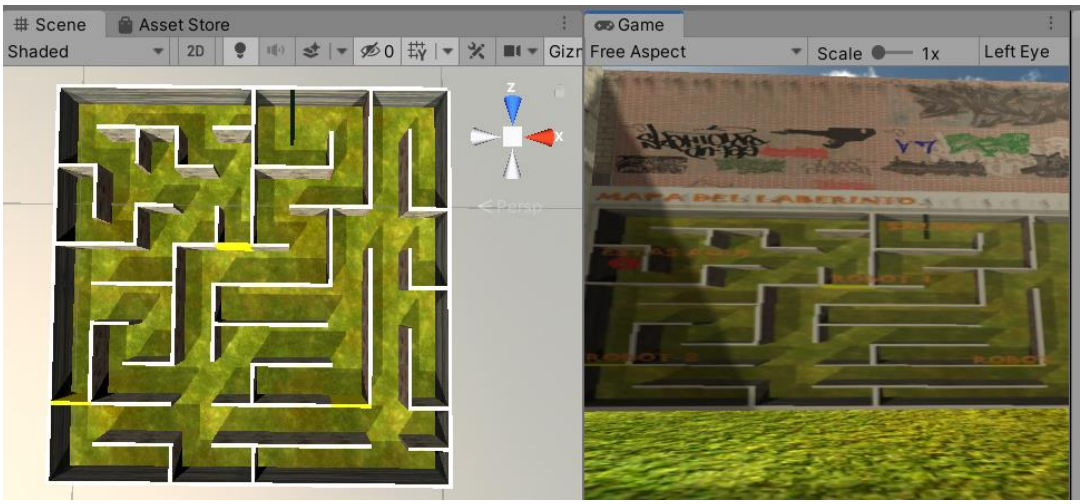
3.1.4. Mejoras al diseño del laberinto.

Acatando las sugerencias hechas por los dos usuarios que probaron el primer diseño realizado que principalmente se basaron en la orientación y ubicación dentro del escenario virtual se procede a mejorar la condición actual realizando el mapa del laberinto e implementándolo dentro del entorno virtual, en este caso se posicionara en la primera pantalla que el jugador ve o la dirección que la cámara está mirando por primera antes de empezar a moverse, simplemente se tomó una captura de pantalla desde la vista superior en la pestaña de escena y se importó

.

dentro de las herramientas de trabajo de Unity como un material común luego se creó un nuevo objeto para ingresarle la apariencia e incluirlo dentro del laberinto.

Figura 26. Mapa incluido.



Fuente: [Autor]

En la figura 26 se observa el mapa ingresado directamente en la vista en primera persona del jugador, no solamente se dejó el mapa, sino que se indicaron dentro del el, ciertas ayudas como ubicación actual y la ubicación de cada uno de los robots animados además de la ubicación de la salida que en este caso será el fin del juego.

Para las pistas que se incluirán en las posiciones donde estarán los robots o cerca de estos, se realizara el mismo proceso que anteriormente se hiso por medio de capturas de pantalla, básicamente se recorta la parte del mapa que al jugador le falta en ese momento y solamente se mostrara la zona que este ya ha explorado, para este caso no se tomara en cuenta si el jugador ya ha explorado todas las zonas que aparecen antes del robot encontrado o si encontró una forma de llegar directamente, ya que no se tendrá ningún método para identificar por cuales zonas ya ha pasado el jugador, esto para facilitar el diseño y la programación del videojuego.

.

Figura 27. Pistas implementadas.

Fuente: [Autor]

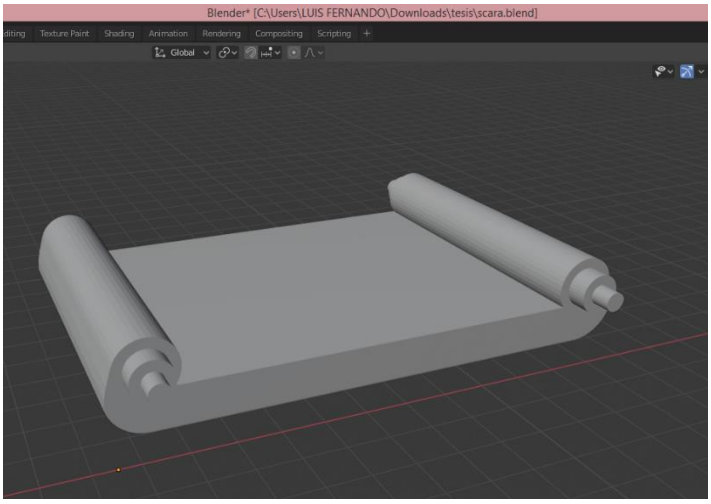
En la figura 27 se observa el método que se utiliza para las pistas dentro del juego, finalmente se optó por ubicarlas cerca de los robots que el usuario tiene que encontrar para poder salir y terminar el juego, se determinó por mostrar una pequeña parte del mapa que se incluyó al principio al encontrar el primer robot, cuando sea encontrado el segundo robot será igualmente encontrado la segunda pista que también se agregara una parte más del mapa y así sucesivamente cada vez que se encuentren pistas nuevas, cabe resaltar que una de las pistas no está cerca de ninguno de los robots sino que está ubicada en una de las esquinas del escenario ya que así se espera que se facilite más la exploración y resolución del laberinto al jugador.

.

Los componentes de texto dentro del escenario estarán dispuestos delante de piezas con forma de pergamino para resaltar de mejor manera cada uno de estos ya que una piza solida dentro del escenario llamara más la atención y se evitara que el jugador llegue a pasarlos por alto.

Se diseñó una pieza en el software *Solidworks* la cual se exporto como archivo stl a Blender para luego ser exportado e incluido en el entorno virtual como un archivo de tipo fbx.

Figura 28. Pergamino exportado en Blender.



Fuente: [Autor]

Figura 29. Pista cerca de la salida.

Fuente: [Autor]

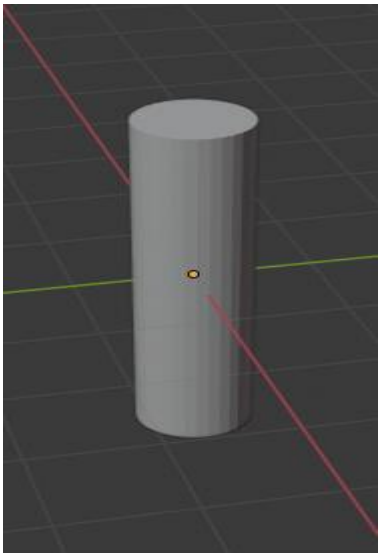
.

Además, la quinta pista se dejó cerca de la salida para avisarle al jugador que tiene que continuar por el mismo camino para terminar el laberinto.

Estos textos flotantes que aparecen cerca de las pistas junto con las instrucciones en el inicio del laberinto se realizaron mediante la herramienta de *unity text-textmeshpro* el cual permite crear un texto en 2D pero que se puede mover como un objeto común 3D ya que se puede mover por todo el espacio de trabajo y realizarle muchas configuraciones.

Para darle un tono más realista al videojuego y por sugerencia de dos estudiantes de diseño industrial se optó por modificar la posición de cada uno de los letreros (las posiciones anteriores se observan en la figura 31 en la derecha.), dentro del escenario además de añadirle un sólido el cual tiene apariencia de madera para que le dé al usuario la ilusión que esta pieza está sosteniendo el letrero y que además se eviten que estos atraviesen los muros principales del laberinto para así también darle más realismo a la escena.

Figura 30. Pieza creada en Blender para el soporte de los letreros.

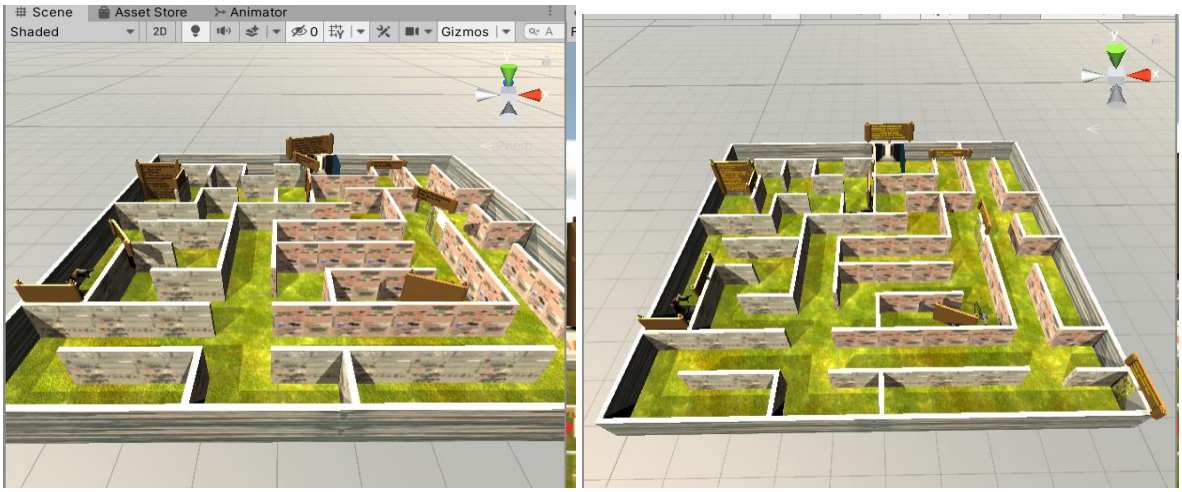


Fuente: [Autor]

Como se observa en la figura 32 el sólido anteriormente nombrado que dará al jugador la ilusión de que está sosteniendo el letrero fue creado en blender y así mismo exportado hacia unity.

.

Figura 31. Zonas separadas por apariencias (derecha posición anterior de los letreros izquierda posición actual de los letreros.).



Fuente: [Autor]

Como último cambio para acatar a las sugerencias de los usuarios que probaron por primera vez el videojuego y para mejorar la orientación dentro del escenario se cambiaron las apariencias de los bloques que conforman las paredes de tal forma que el usuario identifique en que parte del mapa esta y tenga una mejor referencia y evitar estar constantemente dando vueltas en el mismo lugar sin darse cuenta de ello, para realizar este proceso se dividieron en dos zonas las cuales se identifican por dos apariencias diferentes, para la primera zona se optó por el material que aparece en la parte izquierda de la figura 29 y para segunda zona se aplicó el material que se observa en la parte derecha de la figura 29, estas apariencias se obtienen por el mismo paquete importado anteriormente.

Para disminuir la dificultad en cuanto a la búsqueda de pistas se decidió por incluir un aviso sonoro en el momento que el jugador se encuentre cerca de la misma y así darle una pequeña referencia del lugar en el que está.

.

Figura 32. Sonidos cerca de pistas en el laberinto.

Fuente: [Autor]

El método utilizado para la inclusión de sonidos fue por medio de la herramienta llamada audio *source* que permite añadir pistas de audio dentro del juego e implementarlas como música ambiente efectos de sonido al llegar a ciertas áreas efectos de sonido que producen los objetos presentes entre otros.

Para este escenario virtual se agregó una canción que se reproducirá durante todo el tiempo que se esté jugando como un tipo de sonido ambiente y para las pistas se reproduce una pista de poca duración solamente cuando el jugador este cerca.

La música añadida se importó directamente dentro de la carpeta de archivos que se crea a partir de la raíz del proyecto de unity, simplemente se copia y se pega dentro de la carpeta *assets*.

.

Figura 33. Límites de reproducción de sonidos.

Fuente: [Autor]

Como se observa en la figura 33 se agrega el componente de *audio source* que se mencionó anteriormente, posteriormente se desplaza el audio de la misma manera que se agrega un nuevo material o apariencia a los objetos, hay que tener en cuenta que este sonido solamente será escuchado por el usuario cuando este se encuentre cerca.

Por lo tanto, se tiene que configurar la pestaña llamada 3d *sound settings* que está dentro el componente audio *source* que se añadió a los objetos que muestran al jugador las partes del mapa que lleva descubiertas, esta pestaña permite entre otras cosas configurar una zona en específico para oír el sonido correspondiente, esta configuración se realiza definiendo un límite máximo y mínimo en unidades de espacio en Unity en este caso se dejó en 1 y 20 respectivamente.

El software deja visualizar esto por medio de la esfera de color azul alrededor del objeto seleccionado, además se escoge la opción *linear rollof* que como se ve en la gráfica de la figura 33 que representa en color rojo con un comportamiento lineal el

.

volumen del audio con respecto a la distancia en la que se encuentra el usuario del objeto con el audio en cuestión, es decir, entra más cerca se encuentre de este más fuerte se oirá la pista musical, si el jugador esta fuera del límite máximo será inaudible completamente.

Para la música ambiente se añadió nuevamente el audio *source*, pero en este caso se le agrega en la cámara principal para se escuche en todo momento y en todo el escenario virtual, como ultima configuración se importó también a la cámara principal el componente llamado *audio listener* que básicamente por su significado en ingles permite que los audios se puedan escuchar y reproducirse cuando se requiera.

3.2. Diseño de los robots en Blender.

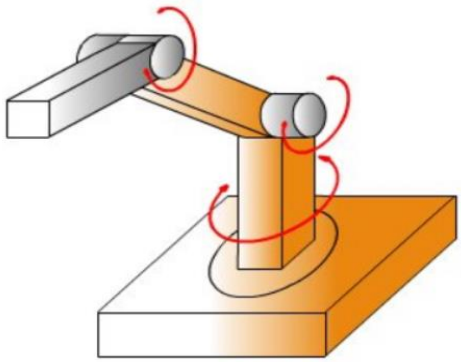
En este apartado del desarrollo del proyecto se tendrán en cuenta tres tipos de robots, los cuales serán robot antropomórfico, robot scara y robot delta, estos serán diseñados y animados en el software blender para luego ser exportados en unity.

3.2.1. Robot antropomórfico.

Como se mencionó anteriormente la característica principal de este tipo de máquinas es que su estructura posea un cuerpo, un brazo, una muñeca y un efector final o pinza los cuales se deben tener en cuenta para realizar el diseño, para lograr que se asemeje todo lo posible a un robot real.

.

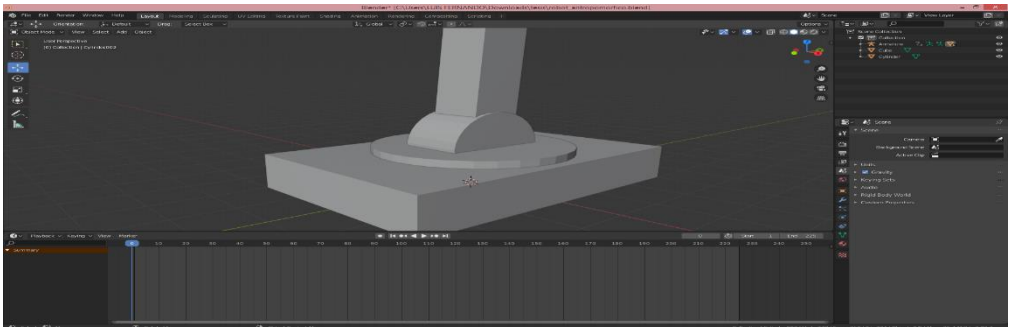
Figura 34. Boceto a seguir.



Fuente: http://www.portaleso.com/web_robot_3/robot_indice.html

Para una mayor facilidad de diseño se toma en cuenta el boceto de la figura 34 ya que este posee las características requeridas, el robot tendrá una base giratoria y 5 articulaciones incluyendo el efector final que en este caso será una pinza con el fin de simular el agarre de objetos.

Figura 35. Primer paso del diseño.

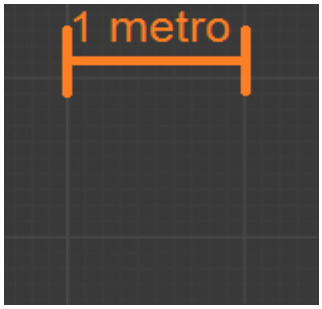


Fuente: [Autor]

En la figura 35 se observa la base del robot que se dejó de 4 por 6 metros basándose en la escala que posee el programa con respecto a cuadricula de este, según la configuración prestablecida cada cuadro principal equivale a un metro en la vida real por lo que para la realización del dibujo se tiene que encienta esta escala e igualmente teniendo en cuenta las medidas del escenario diseñado en Unity, ya que cada pasillo tiene 10 metros de longitud.

.

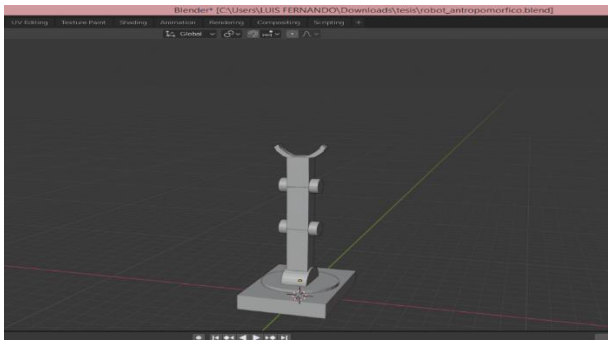
Figura 36. Escala en Blender.



Fuente: [Autor]

Los dos primeros eslabones partiendo desde la base tienen una longitud de 3 metros, el eslabón del efector final tiene una longitud de dos metros, todos estos tienen el mismo grosor de 1m por 1m, cada articulación es representada por un cilindro que también representa los motores y la pinza que consta de dos cubos que cuando está cerrada mide 1m de largo al sumar todas estas longitudes se tendría un área de trabajo de 10 metros cuando el robot está completamente estirado.

Figura 37. Diseño final.



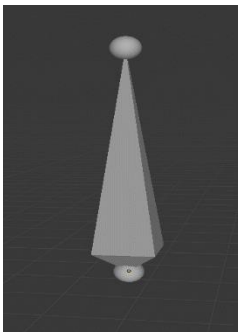
Fuente: [Autor]

En la figura 37 se observa el diseño final del robot, este inicialmente tiene esa posición debido a que en este punto solamente se comporta como un sólido común y corriente, es decir, que es una pieza que no posee ningún tipo de animación o movimiento al ser un conjunto de figuras en tres dimensiones nada más, para generar una animación con el objeto creado se debe hacer uso de la herramienta que posee el programa llamada armadura, que permite añadirle una serie de elementos a la pieza diseñada para que comporten como un esqueleto y de esta

.

manera darle movimiento al sólido, dentro de Blender estos elementos se conocen como huesos.

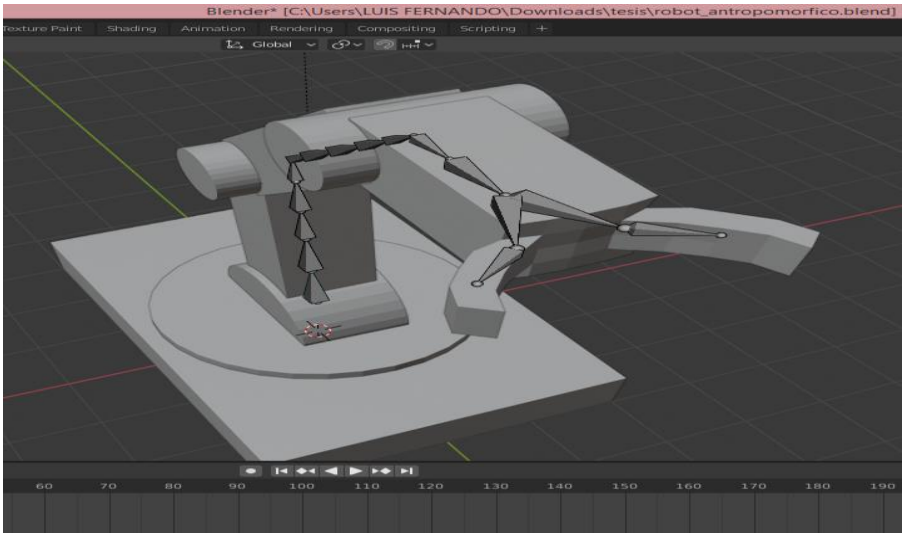
Figura 38. Hueso en Blender.



Fuente: [Autor]

En el software se representa los huesos de la forma como se observa en la figura 38 que es un octaedro y dos puntos de conexión llamados cabeza que está ubicado en la parte estrecha y la cola que está ubicada en la parte ancha del cuerpo principal de este que así es como es llamado el octaedro de la pieza, en los puntos de conexión van situados los otros huesos para así generar el esqueleto que necesitan los personajes para poder moverse.

Figura 39. Creación del esqueleto en el robot

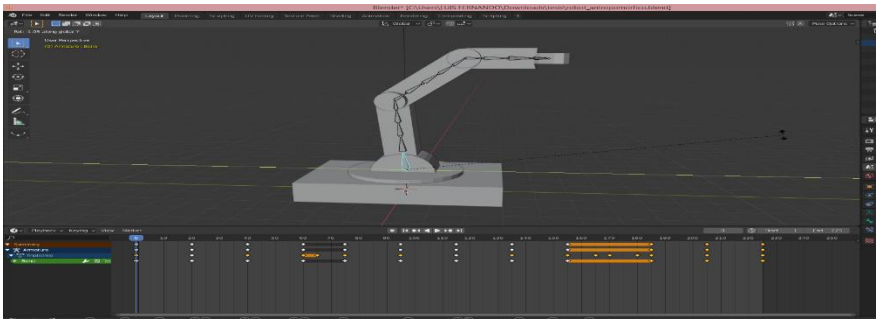


Fuente: [Autor]

.

El esqueleto formado dentro del robot se observa en la figura 39 el cual está posicionado en el centro del mismo para lograr un buen resultado en cuanto a la animación, luego se procede a relacionar el esqueleto con el robot para que al realizar movimientos con respecto al plano cartesiano este pueda seguir los movimientos que realiza la armadura, antes de exportar los archivos a Unity se debe crear el clip de animación, esto se realiza por medio de la herramienta *timeline* dentro de Blender la cual proporciona una línea de tiempo y un *recorder* que guarda cada uno de los movimiento que se realicen.

Figura 40. Creación de la animación.



Fuente: [Autor]

En la figura 40 se observa el proceso llevado a cabo para la creación de la animación, al seleccionar uno de los huesos que componen la armadura y oprimir la tecla correspondiente para rotar con respecto a uno los tres ejes ya sea X, Y o Z el software graba este movimiento y lo añade a la línea de tiempo que se observa en la parte inferior de la figura 40, esto es posible gracias a la opción *keyframe* de Blender que se activa y desactiva solo cuando es presionada una tecla de movimiento en este y según el requerimiento, se utiliza la tecla R que permite hacer una rotación siguiendo los ejes del plano cartesiano, la serie de rotaciones que se crean es para simular que el robot está tomando objetos con la pinza y los está llevando y soltando a otra zona, luego en el apartado de *dope sheet* se le da un nombre a esta animación y finalmente se exporta como archivo .fbx que este guarda las gráficas de los sólidos diseñados junto con los ficheros de animación.

El clip creado simula un proceso donde el robot toma objetos de un sitio determinado, los lleva hacia otro y los suelta, así en un ciclo infinito.

En Unity se importa como un paquete *asset* y se añade al escenario.

.

Figura 41. Robot dentro del escenario virtual.

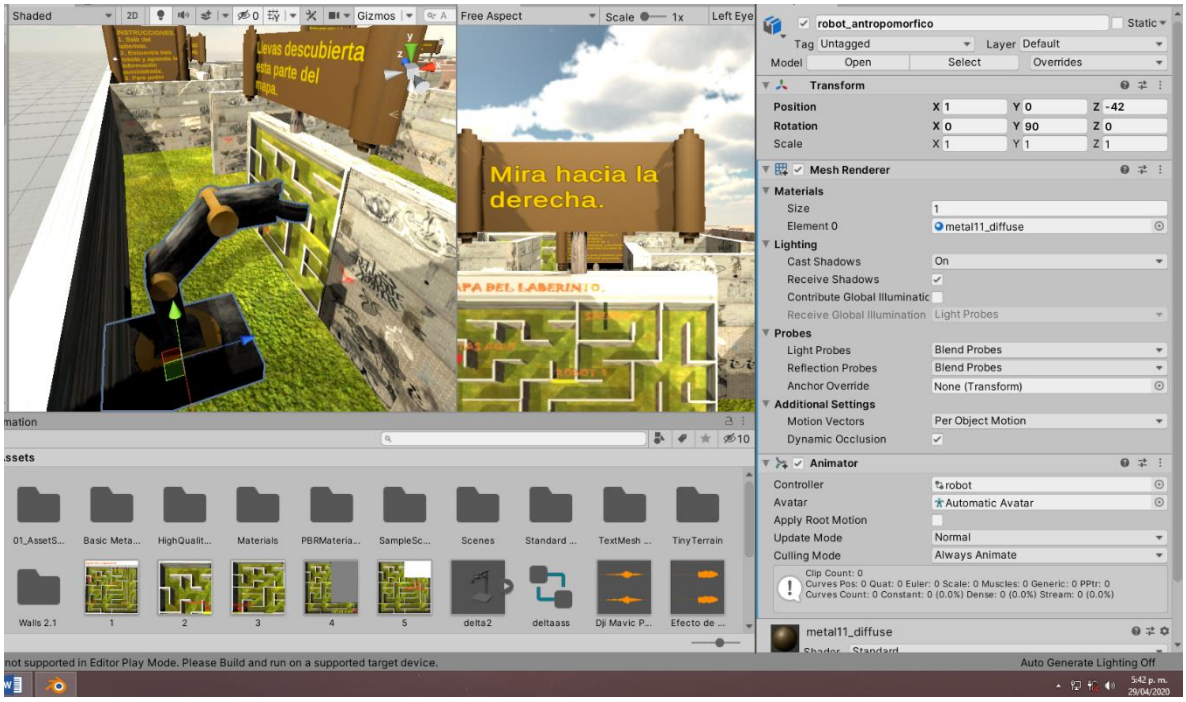
Fuente: [Autor]

En la figura 41 se observa el robot dentro del entorno virtual, también se le añadieron texturas, para la animación es necesario importar el paquete llamado *animation controller* para poder extraer el clip animado que se creó en Blender e ingresarlo al videojuego además de añadir el componente *animator* al objeto.

También se agregó el componente *mesh renderer* que permite importarle materiales a la figura.

.

Figura 42. Configuración del objeto robot.



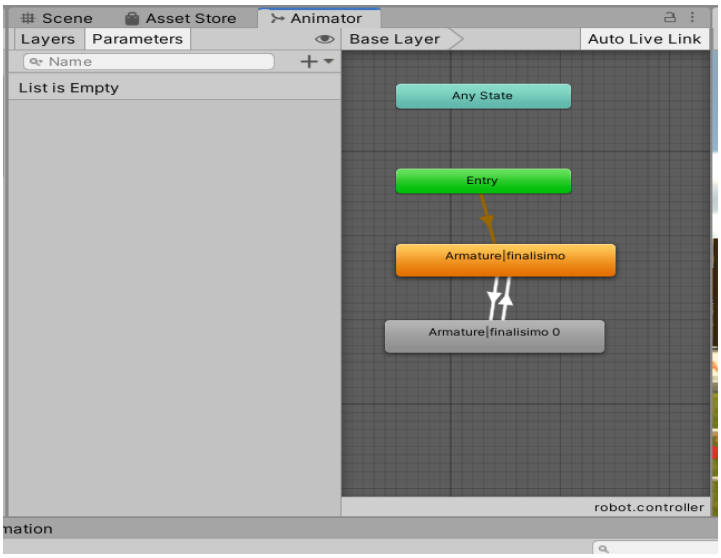
Fuente: [Autor]

El paquete de *animator controller* se nombró como robot, este se desplaza hasta el componente *animator* que está dentro del objeto para su posterior utilización dentro de la pestaña *animator*, después se desplaza el fichero que contiene el clip de animación en la pestaña que controla estos archivos, debido a que es este tiene corta duración se necesita que quede reproduciéndose en un bucle infinito por lo que se tiene que agregar nuevamente y luego entrelazarlos por medio de transiciones que lo que hacen es que en el momento en el que termine una animación se reproduzca la otra y así sucesivamente durante toda la ejecución del juego, además de conectarse con el bloque *entry*

para que pueda empezar desde el principio del juego.

.

Figura 43. Animación dentro de Unity.



Fuente: [Autor]

Finalmente se agrega la información que el usuario debe aprender de este robot.

Figura 44. Información del robot

Fuente: [Autor]

.

3.2.2. Robot scara.

El proceso a seguir para incluir tanto el sólido del robot como su animación es exactamente el mismo, para el diseño en Blender se tomó como referencia la maquina construida por la empresa Toshiba que está destinado para trabajos en la industria el cual se observa en la figura 45.

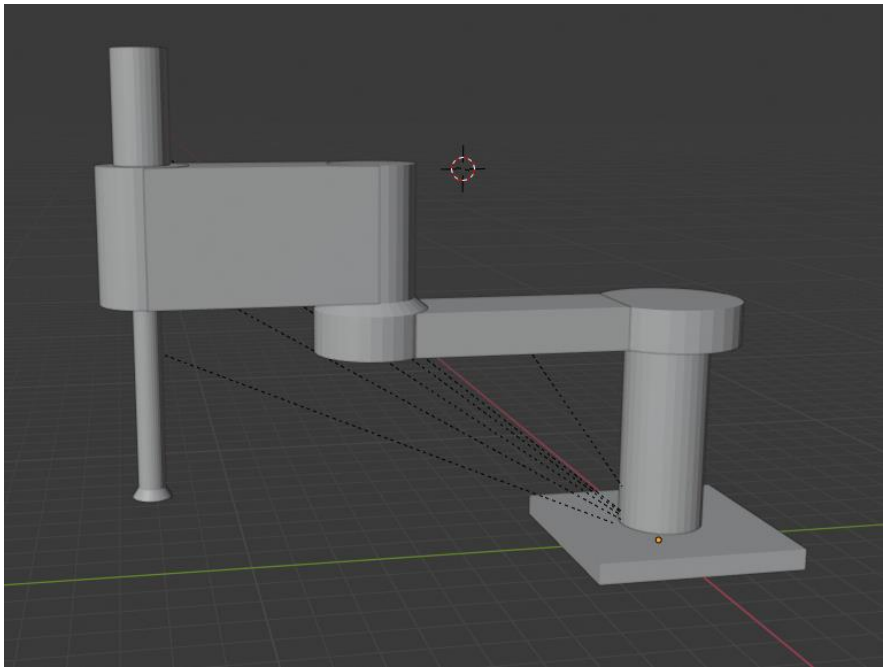
Figura 45. Robot scara Toshiba TH650A.

Fuente: https://www.fegaut.com/es/marcas/toshiba-machine-industrial-robots/22/

Una de las principales características de este tipo de robots es el efector final que en la mayoría de casos posee un pistón que se desplaza solamente en el eje z si se tiene en cuenta que la referencia es un sistema cartesiano, además de que los movimientos de las otras articulaciones solamente pueden realizarse dentro de las coordenadas X y Y.

.

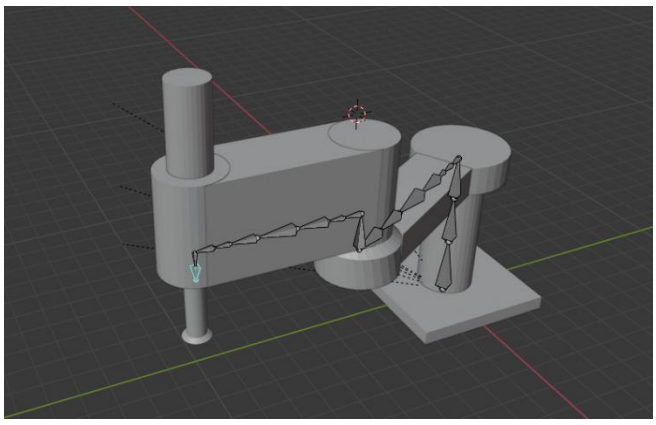
Figura 46. Diseño final del robot scara.



Fuente: [Autor]

En la figura 46 se observa el resultado final del robot hecho en blender, cada una de las articulaciones tienen 3 metros de longitud según la escala que proporciona blender, el efector final tiene 3 metros de longitud completamente estirado y se tendría un área de trabajo máxima de 6 metros.

Figura 47. Armadura añadida.

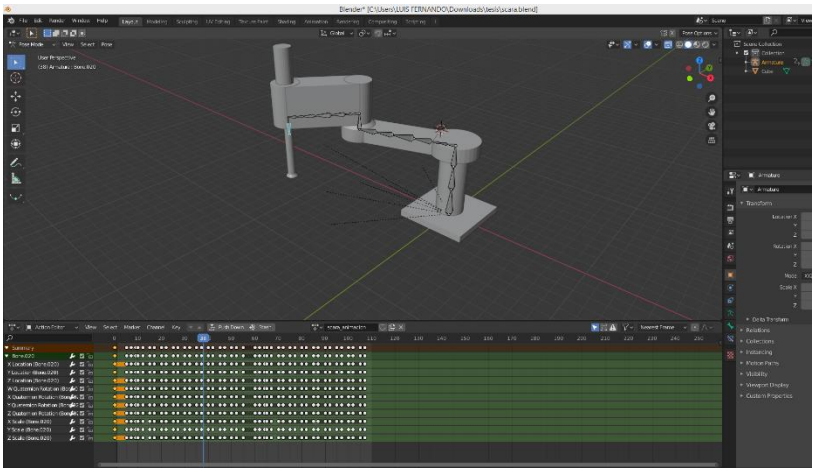


Fuente: [Autor]

.

Ya con la armadura incluida y relacionada a la pieza se procede a crear el clip de animación, en este caso para las articulaciones se moverá con respecto al eje z, en cuanto al efector final se va a realizar un escalado también en el eje z para crear el efecto de salida y entrada del pistón.

Figura 48. Clip de animación.

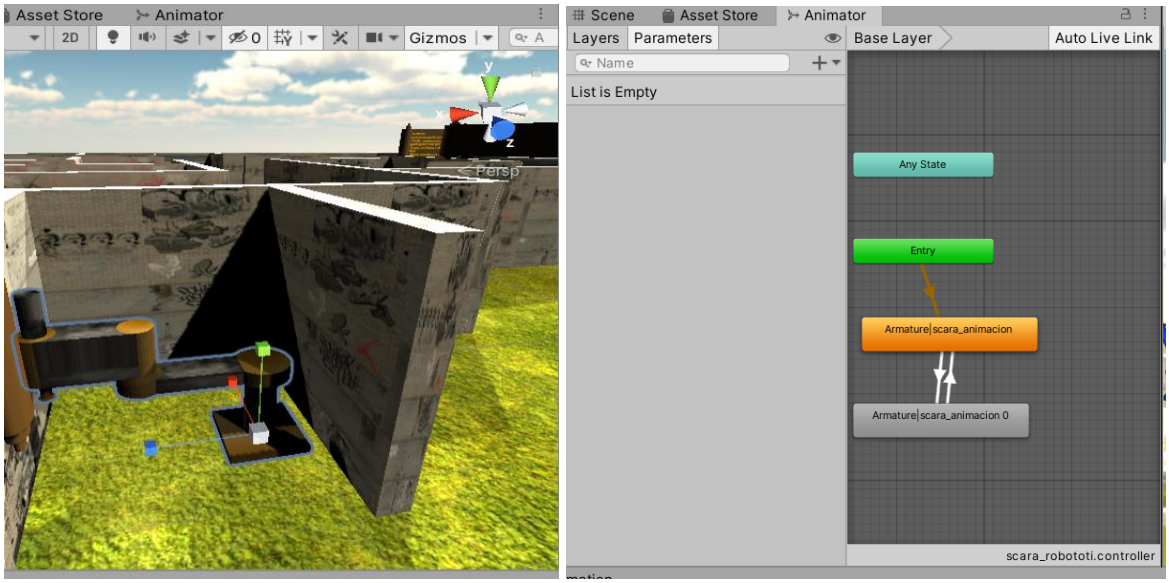


Fuente: [Autor]

En la figura 48 se contempla el clip de animación ya creado, los movimientos que se presentan son muy rápidos debido a que esta es una de las principales características de este tipo de máquinas ya que su configuración mecánica le permite realizar estos desplazamientos tan ágiles.

.

Figura 49. Objeto del robot scara importado en Unity.



Fuente: [Autor]

En la figura 49 se observa el robot junto con la animación y su respectivo controlador de animaciones, igualmente se realizaron transiciones para hacer el bucle infinito.

Figura 50. Información del robot scara.

Fuente: [Autor]

.

3.2.3. Robot delta.

Un aspecto importante de este robot es que consta de dos placas unidas entre sí por los brazos que conforman los tres grados de libertad, la placa superior siempre deberá permanecer inmóvil y la placa inferior será la que se ira moviendo, pero es importante que durante todos los movimientos esta permanezca de forma paralela a la placa superior.

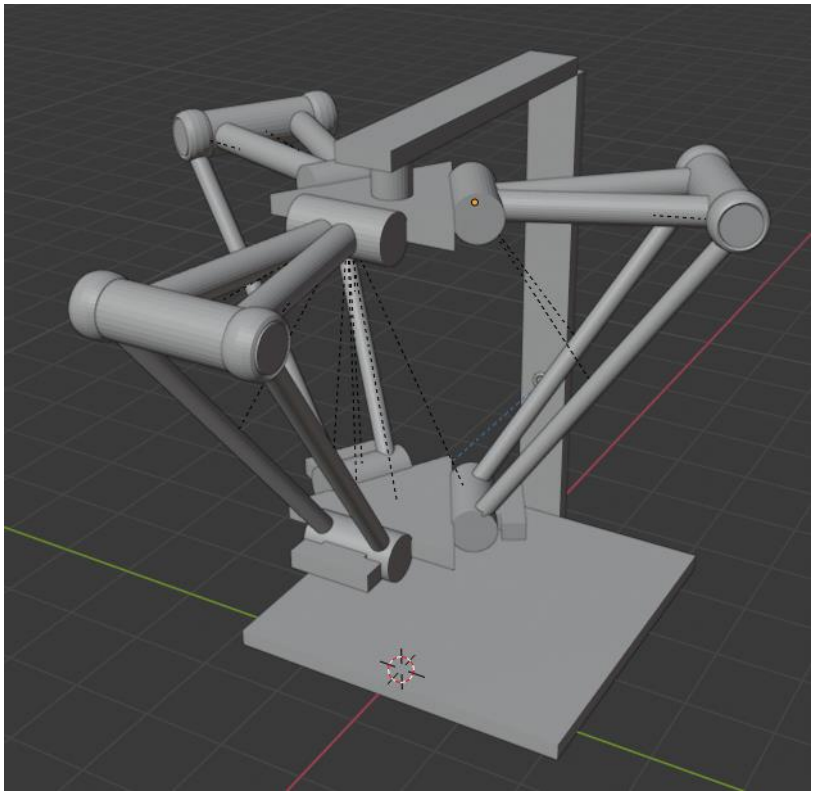
Figura 51. Robot delta.

Fuente: https://eduardojvblog.wordpress.com/2016/01/15/cinematica-directa-y-cinematica-inversa-de-un-robot-delta-angular-y-de-un-robot-delta-lineal/

Las dimensiones del robot diseñado en Blender son de 6 metros de alto partiendo desde la base, la longitud de los brazos es de 4 metros y dos metros tienen de longitud los brazos superiores que están conectados directamente con la base superior que esta fija.

.

Figura 52. Robot delta en blender.



Fuente: [Autor]

En la figura 52 se observa el diseño final del robot tipo delta, la posición de los brazos posee entre cada una de ellas 120 grados respectivamente, así como aparece en la figura 51 para respetar la configuración de este tipo de máquinas, para este caso no se optó por usar armaduras, sino que se buscó por generar una serie de relaciones entre los mismos objetos para poder obtener una animación realista y de cierto modo una relación mecánica entre cada uno de ellos y también para disminuir la complejidad a la hora de realizar los movimientos.

.

Figura 53. Robot dentro del escenario.

Fuente: [Autor]

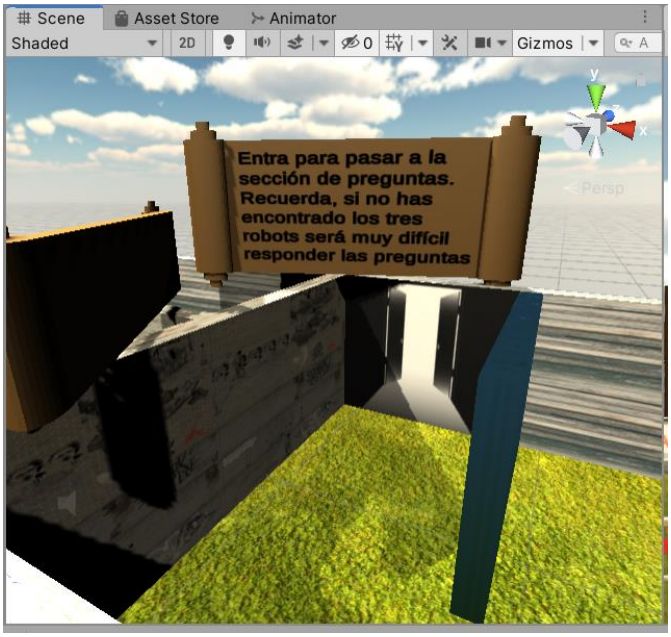
En la figura 53 se observa el robot delta diseñado en Blender dentro del escenario virtual junto con el clip de animación exportado del software, así como la información del mismo.

3.3. Segunda parte del juego.

Después de agregar cada uno de los robots al escenario se procede a diseñar la segunda parte del juego que constara de una serie de preguntas que serán contestadas por el usuario dependiendo de la información suministrada dentro del laberinto. Habrá cuatro preguntas en total, las cuales se añadirán con la herramienta anteriormente usada para incluir textos, además se incluyen escenas que solamente estarán destinadas para estas preguntas.

.

Figura 54. Salida del laberinto.

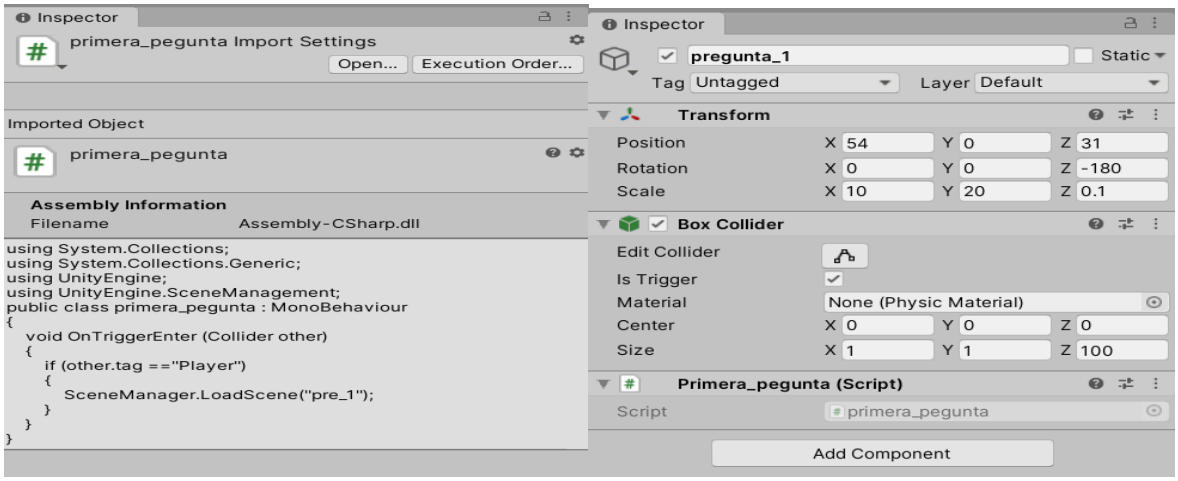


Fuente: [Autor]

En la figura 54 se observa el nuevo aspecto que se le dio a la salida del escenario el cual posee una advertencia por si el usuario llega antes de encontrar los tres robots, ya que las preguntas están basadas en la información proporcionada, el jugador tendrá que acercarse a la zona azul para transportarse a la escena que contiene la primera pregunta, esto es posible gracias a el *gameObject* que tiene resaltados los vértices en color verde que es llamado pregunta_1.

.

Figura 55. Configuración del objeto encargado de cambiar de escenas.

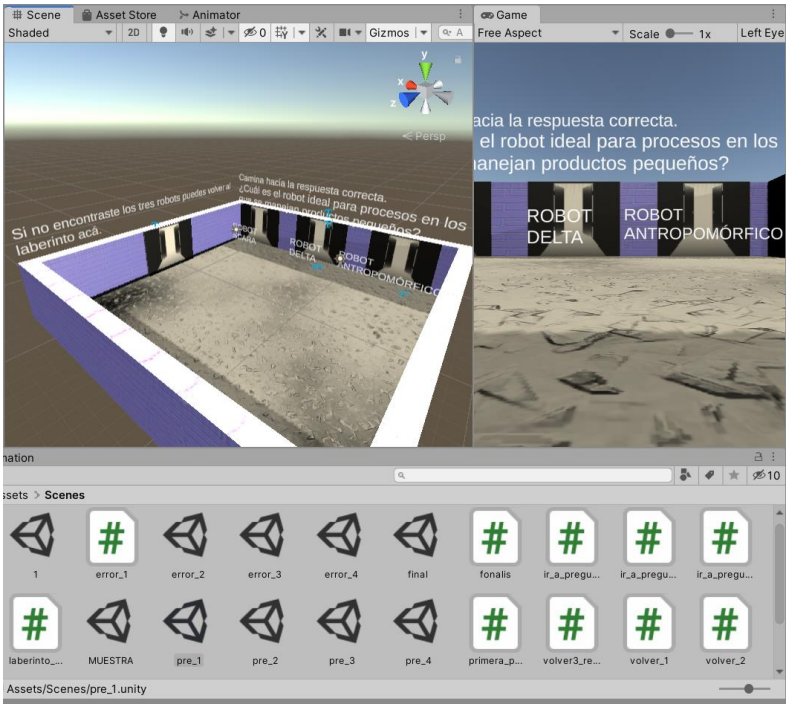


Fuente: [Autor]

En la figura 55 en la parte derecha se observa la configuración realizada en este objeto en el cual se agregaron dos componentes, uno de estos es el *box collider* que le añade facultades de un objeto común dentro de Unity, pero con la diferencia que se activa la opción llamada *inTrigger* que hace posible que el jugador pueda a travesar el objeto y también detectar esa acción por medio del script que aparece en la parte izquierda de la figura 55 que también se añade como complemento, en el momento en el que el jugador entra en el cubo se cambia de escena que en este caso es a la primera pregunta, para que el código funcione se debe incluir cada una de las escenas en el *build settings* y configurar el *tag* del controlador en primera persona como Player ya que esta variable está relacionada con la instrucción que verifica si el usuario entra o no en el objeto de la figura 55.

.

Figura 56. Escena pregunta 1.

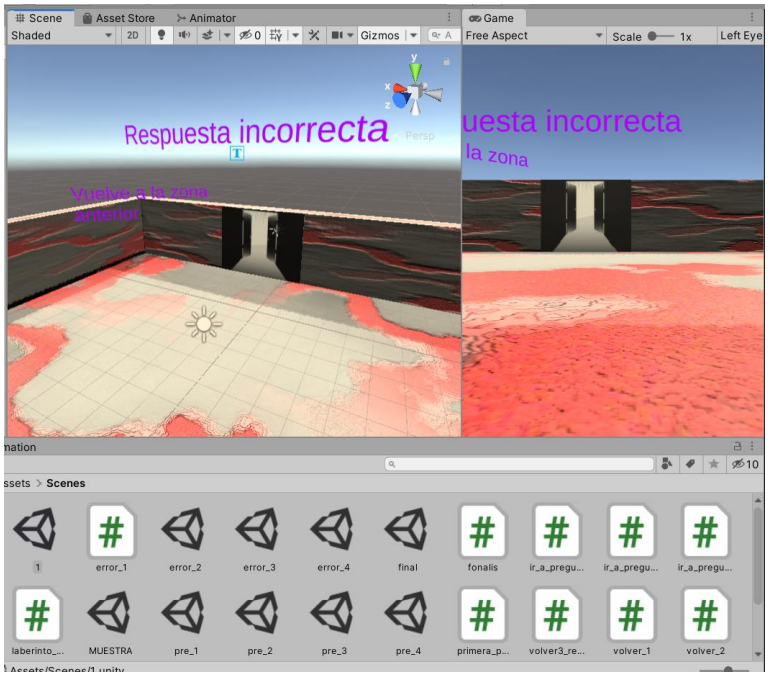


Fuente: [Autor]

En la figura 56 se observa el modelo de escena que se implementa para las 4 preguntas del juego, para el caso de la primera pregunta tendrá una zona en donde se podrá regresar al laberinto o la escena principal por si el jugador no pudo encontrar los tres robots y no conoce la respuesta, si el jugador acierta será llevado a la siguiente pregunta o la escena de la misma, si no responde correctamente será llevado a un escenario que le dirá que la respuesta es errónea y también tendrá una zona que lo llevara nuevamente a la pregunta en la cual estaba.

.

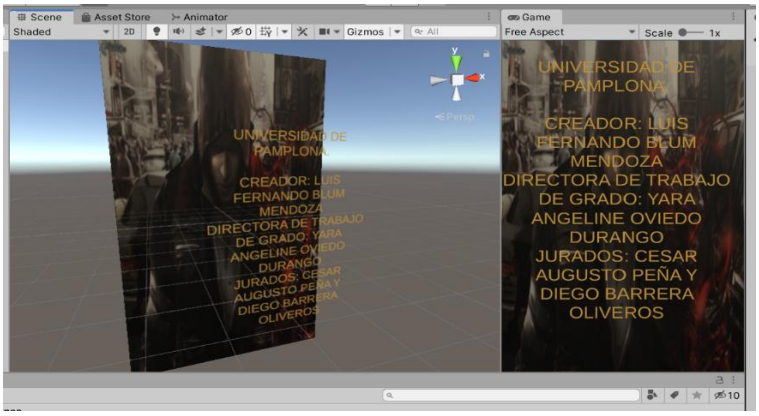
Figura 57. Zona de respuestas incorrectas.



Fuente: [Autor]

Además, ambas zonas tendrán efectos de sonidos para amenizar la experiencia de juego. Después de responder todas preguntas correctamente al jugador se le llevara a una escena simple de créditos indicándole que el juego termino.

Figura 58. Escena final.



Fuente: [Autor]

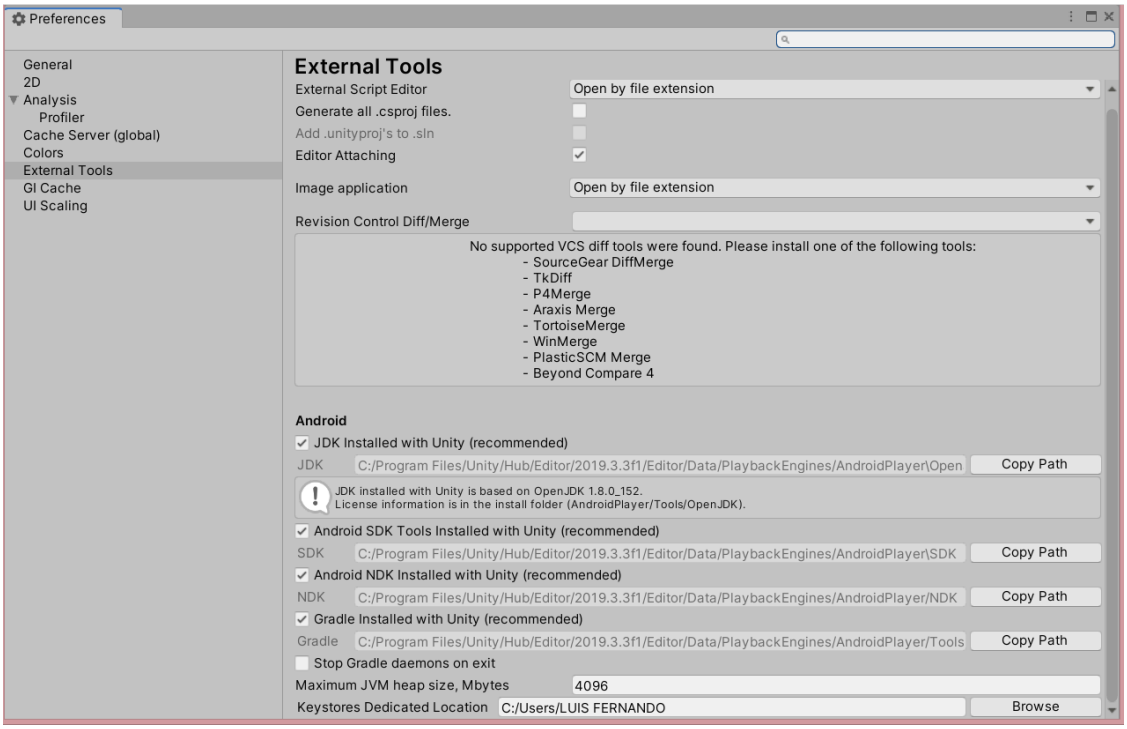
.

3.4. Exportar el videojuego a la plataforma Android.

Antes de poder utilizar la aplicación creada en unity en dispositivos móviles que posean sistemas operativos basados en Android se requiere exportar todos los datos del entorno virtual, para esto se deben instalar ciertos complementos como el *Android studio* que se encarga de descargar el conjunto de herramientas de desarrollo, depurador de código entre otros que permite realizar el proceso para generar la aplicación de Android este complemento es llama *Android SDK tools.*

También se debe descargar el kit para desarrolladores de java o JDK, ambos complementos se agregan en la pestaña preferencias definiendo la ruta donde quedaron instalados, así como se observa en la figura 59.

Figura 59. Herramientas añadidas en el software.

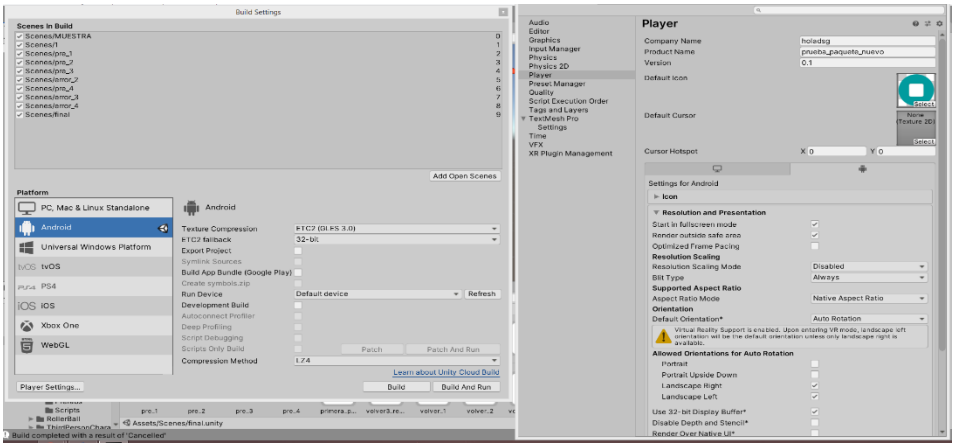


Fuente: [Autor]

Luego se procede a cambiar a la plataforma Android en el apartado de *build settings* para luego configurar los demás parámetros y exportar correctamente la aplicación.

.

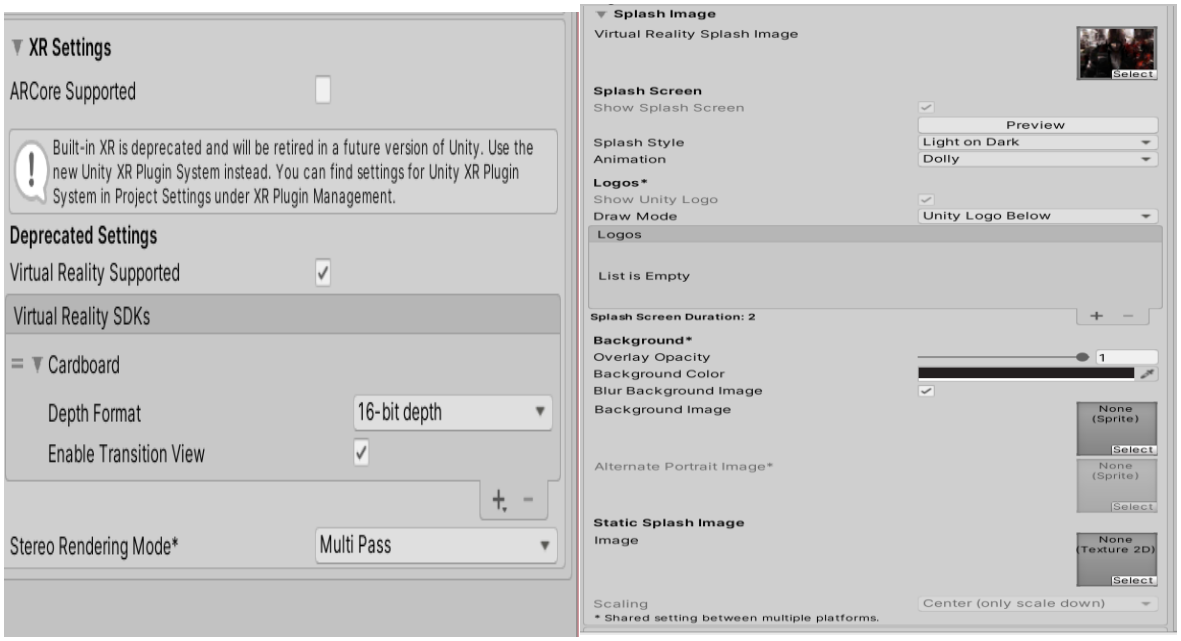
Figura 60. Configuración build settings



Fuente: [Autor]

En la figura 60 se observan las configuraciones que se deben realizar entre otras están el nombre de la compañía, nombre del juego, versión actual entre otras, además se puede añadir el icono que tendrá la aplicación en el dispositivo móvil, también se configura la rotación que tendrá la pantalla y que el modo de visualización sea en pantalla completa todo esto se configura en la pestaña llamada resolución y presentación.

Figura 61. Configuraciones en player settings.

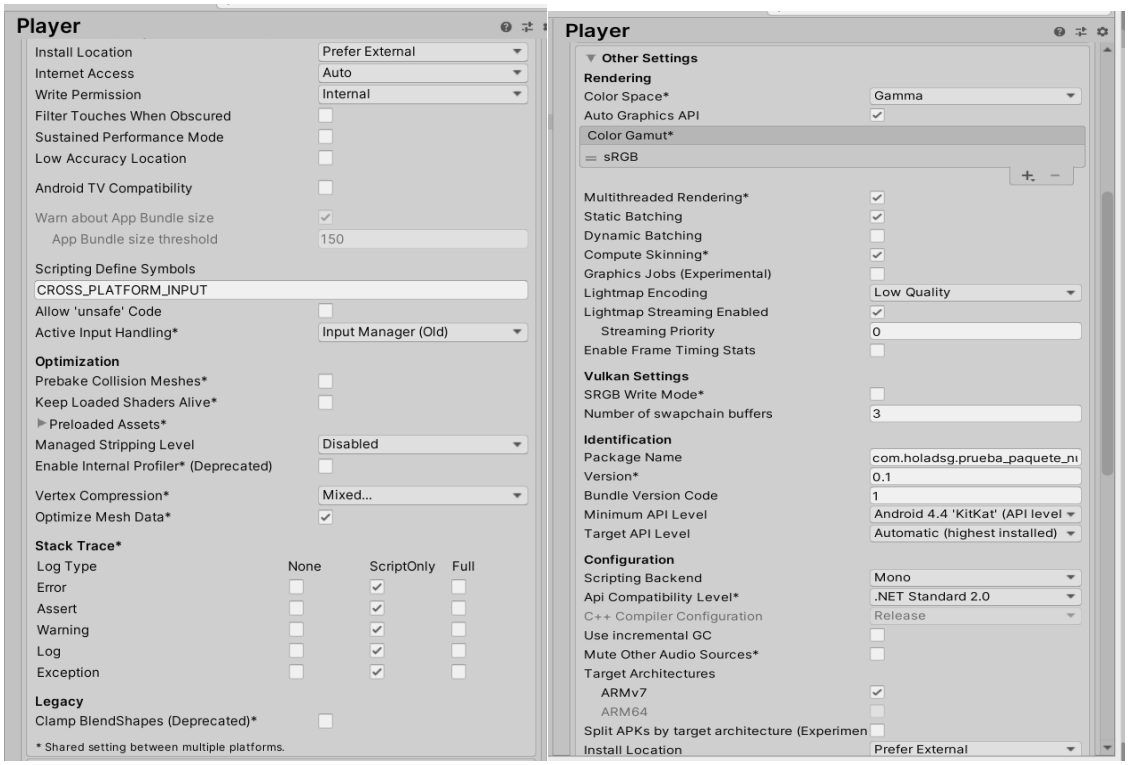


Fuente: [Autor]

.

Para personalizar aún más el diseño de la aplicación se optó por mostrar una imagen en la pantalla de carga, la cual se puede añadir en *splash image*, como se observa en la parte derecha de la figura 61, en XR *settings* se genera la visualización en modo de realidad virtual activando el *virtual reality supported* en el cual viene incluido el soporte para *cardboard*.

Figura 62. Otras configuraciones.



Fuente: [Autor]

En la pestaña *other settings* se debe definir la versión de Android mínima que debe tener el dispositivo móvil para que pueda correr la aplicación correctamente.

Finalmente se presiona el botón *build* que exportara la aplicación en un archivo de tipo apk el cual se instalara directamente en el móvil.

.

3.5. Segunda prueba realizada a usuarios.

Atendiendo las solicitudes y sugerencias dadas por los usuarios que anteriormente probaron el primer prototipo del videojuego y rediseñándolo nuevamente se decidió por realizar esta prueba por segunda vez ya con el producto final para verificar si realmente ambos jugadores tienen la capacidad de resolver el laberinto y terminar completamente el videojuego.

3.5.1. Usuario 1.

Figura 63. Usuario 1 probando el juego.

Fuente: [Autor]

.

La imagen del celular se transmitió a un televisor por medio de conexión wifi para poder apreciar en todo momento lo que el jugador está viendo, en la figura 63 se observa que el usuario 1 esta interactuando con el entorno virtual cabe resaltar que las indicaciones sonoras y visuales implementadas ayudaron en gran medida a que el jugador encontrara rápidamente el segundo robot.

Figura 64. Cerca de la salida del laberinto.

Fuente: [Autor]

En la figura 64 se observa que el jugador 1 logro resolver el laberinto de manera correcta encontrando cada uno de los robots por lo que el nuevo diseño dentro del entorno virtual conllevo a que el usuario 1 se ubicara de mejor manera.

En la sección de preguntas estas fueron resultas correctamente gracias a la información suministrada en la zona principal y llegando así al final del juego.

.

Figura 65. Usuario 1 en el final del videojuego.

Fuente: [Autor]

3.5.2. Usuario 2.

Los resultados obtenidos del usuario 2 fueron igualmente satisfactorios que los del usuario 1 ya que este logro llegar al final del laberinto y resolver las preguntas correctamente terminando el videojuego.

.

Figura 66. Interacción del usuario 2 con el videojuego.

Fuente: [Autor]

.

4. Análisis de resultados.

Después de crear la primera versión del videojuego en donde solamente estaba presente la parte física del mismo, es decir, solamente muros y terreno se decidió por generar el movimiento del personaje realizando diferentes métodos para esta tarea, el primero de ellos que se utilizo fue el paquete que suministra de manera libre google cardboard el cual se ve reflejado su manejo por los autores Samartha, Kotwal y Vidyashri [30], que fue un artículo tomado como referencia para realizar y capturar cada uno de los movimientos que el usuario desea emplear dentro del escenario, los autores del artículo antes mencionado llevan a cabo su diseño a partir de una versión antigua de unity por lo que a la hora de intentar implementar este proceso, fallo debido a incoherencias entre archivos nuevos de la plataforma utilizada que en este caso se trata de la 2019.3.3.f1.

El error dado a partir de este paquete fue específicamente cuando se exportaba el entorno a la plataforma de Android y la única manera de resolver este problema era reinstalar a una versión más antigua que la actual, por lo que se optó por buscar en otros proyectos nuevas formas de elaborar este proceso, después de buscar una gran cantidad de diferentes autores y proyectos finalmente se encontró una forma en la que el autor Linowes [31], afirma que dentro del software existen una serie de componentes que contribuyen a generar diferentes opciones dentro del videojuego y como ya se ha mencionado en el apartado de desarrollo del proyecto existe una que permite realizar el requerimiento para moverse en el entorno virtual.

Pero al importar estos archivos nuevamente se obtienen errores debido a que la versión para la que están hechos solamente funcionan hasta la 2017 pero en este caso en particular y acatando las recomendaciones que da el software se procedió a resolver esto ya que simplemente existía un conflicto con actualizaciones de librerías ya con esto solucionado funciona correctamente permitiendo que no solamente fuera permitido moverse dentro del espacio virtual sino que también su posterior exportación hacia una aplicación con capacidad de ser instalada en cualquier dispositivo con sistema operativo Android.

Con la aplicación ya lista en su primera versión se procede a realizar las primeras pruebas con dos usuarios a los cuales se les explico de manera breve los controles y las instrucciones del juego, después unos minutos de haber probado el producto

[30] S. M. Samartha, R. S. Kotwal y V. M. Vidyashri, «Exploring the world of virtual reality gaming with Google Cardboard and Unity,» International Journal of Modern Trends in Engineering and Research, 2016.

[31] J. Linowes, Unity virtual reality projects, Packt Publishing Ltd, 2015.

se obtiene ambos usuarios fracasaron en resolver el laberinto y ambos refieren que es demasiado difícil ubicarse dentro del entorno virtual debido a que no lograron diferenciar ni ubicarse en ningún momento, estos afirman que sería mejor si existieran pistas o mapas para una mejor percepción del entorno, estas recomendaciones son muy similares a las encontradas por los autores Sánchez, Marín y Villagomez [32], los cuales después de realizado un estudio con muchos voluntarios tanto hombres como mujeres que se sometieron a resolver un laberinto dentro de un entorno virtual, descubrieron que a pesar de que cada género se enfrenta de manera diferente ya sea por orientación para los hombres y navegación para las mujeres, estos requieren de las mismas condiciones de pistas y métodos para orientarse en donde lograron un resultado bastante parecido para ambos géneros por lo cual se procede a elaborar la segunda versión del juego acatando las recomendaciones tanto de los usuarios como de los autores que realizaron el estudio mencionado anteriormente y sin llegar a que los jugadores lleguen a la zona de preguntas ya que la idea principal es verificar que el laberinto pueda ser resuelto de manera correcta y efectiva, los usuarios mencionados probaron nuevamente el juego con resultados satisfactorios ya que lograron pasados unos minutos llegar a la salida del laberinto y además encontrando los tres robots, por lo que realmente el usuario con ayuda de pistas colocadas en determinadas posiciones dentro del laberinto logran que los jugadores se orienten correctamente.

Después de que se terminara la zona del laberinto y empezara la zona de preguntas los usuarios respondieron correctamente a todas estas a pesar de algunos errores que cometieron, pero ambos jugadores llegaron al final de juego. Realmente las preguntas mal respondidas pueden deberse a que el usuario al interactuar con varias cosas en la pantalla al encontrar un robot ya que en el espacio donde está posicionado existen diferentes objetos como pistas y letreros por lo que el jugador puede distraerse mirando el mapa y olvidando que tiene que aprender acerca del robot que ha encontrado, aunque también puede suceder de que algún usuario se disponga solamente a salir del laberinto sin haber hecho lo anterior así como sucedió cuando uno de los voluntarios llego rápidamente a la salida y empezó directamente con las preguntas, sin saber la respuesta de ninguna, este usuario decidió devolverse nuevamente a la escena principal utilizando la puerta que conduce desde la zona de preguntas hacia la zona del laberinto, esta puerta está pensada específicamente para evitar que el videojuego quede estancado y por esa razón el usuario quede atrapado de cierta forma y no pueda terminarlo.

Para el diseño de los robots se tomó en cuenta entre los diferentes tipos que existen los más sencillos en cuento a criterios de diseño y facilidad para animarlos, por lo

[32] I. Sánchez, F. J. S. Marín, R. S. Villagomez y others, «Diferencias de género en la orientación espacial en un ambiente virtual,» *Revista Mexicana de Psicología,* vol. 28, pp. 211-216, 2011.

cual se optó basándose en la experiencia personal por el robot antropomórfico, delta y el scara; cuando se realizó la primera animación en el software blender se decidió por unir la menor cantidad de huesos con la pieza que en este caso fue el antropomórfico, pero al generar los movimientos los sólidos que componen el robot se deformaban demasiado a tal punto de que los desplazamientos no eran para nada realistas, así que se cambió el número de huesos en una disposición en la cual evite y disminuya considerablemente la deformación producida por los huesos y determinar la posición de estos para garantizar que ambos componentes se solapen teniendo en cuenta los huesos adyacentes al desplazamiento deseado según la articulación que se está animando, para esto se tuvo en cuenta las recomendaciones que proporciona Pérez [33], en su libro en donde afirma que para evitar esto es necesario disponer de un numero de huesos en la malla los cuales permitan que se solapen con el hueso adyacente y así obtener una animación más natural y realista, el mismo proceso se realizó para los otros robots.

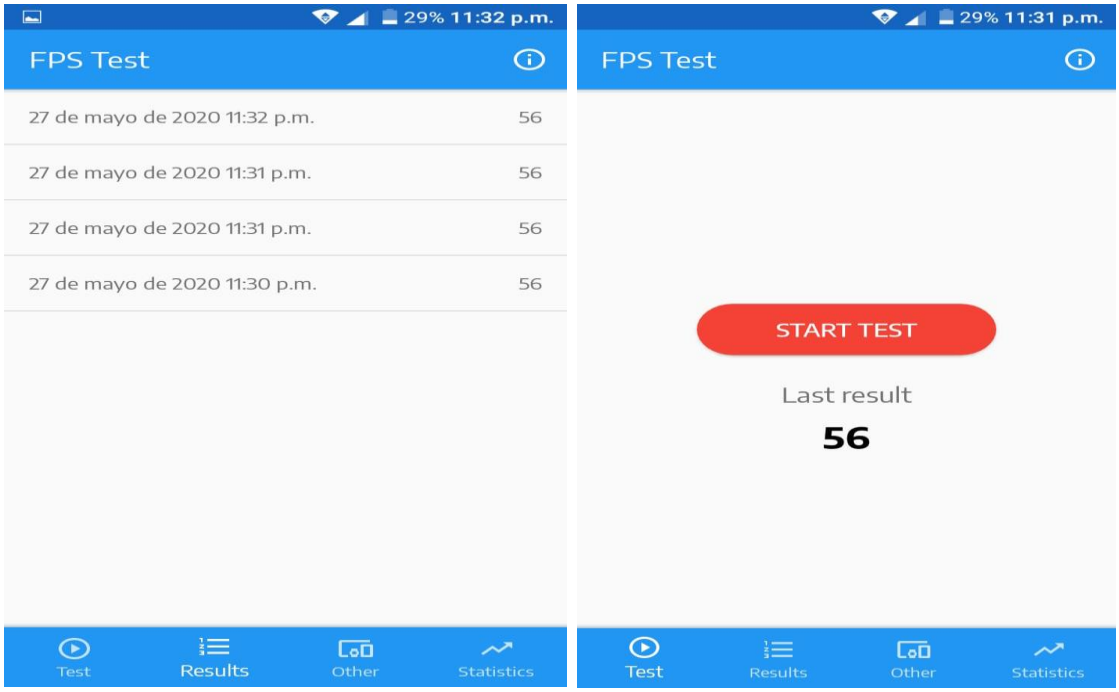
Es necesario resaltar que tanto la animación como el diseño del robot delta fueron muy complejos debido a la composición misma del tipo de máquina. Sus características como la posición de las articulaciones, el número de estas, la distribución de los motores entre otras cosas hace que se dificulte en gran medida esta tarea, el primer diseño creo como el modelo común, pero a la hora de efectuar la animación ya que para rotar los huesos los programas tienen en cuenta el plano cartesiano por lo que no se obtiene de manera correcta los movimientos y también aparecen deformaciones en las articulaciones, para solucionar esto y tener una animación correcta se ubicaron los brazos del robot paralelamente a cada uno de los ejes así como se construyen los robots delta de cuatro ejes, con esta disposición fue sencillo generar la animación.

Después de exportar los archivos creados en blender hacia unity hubo problemas con las animaciones ya que los tres robots generaban conflictos a la hora de quedar importados dentro de la carpeta de componentes del juego, lo que ocasionaba que no fuera posible acceder a todas las opciones que aporta el software, entre ellas está la opción de repetir esta animación infinitamente ya que el clip que la contiene solamente dura unos segundos, pero por este error la opción para poder realizar esto esta deshabilitada, la única solución que funciono para este caso fue dentro del controlador de animaciones entrelazar dos componentes del clip entre sí para así generar el bucle infinito necesario, cabe resaltar que este problema no interfirió a la hora de exportar el juego a una aplicación basada en Android

[33] P. S. Pérez, Manual de modelado y animación con Blender, Universidad de Alicante, 2011.

Desde el inicio del diseño y la programación de la aplicación se tuvo en cuenta un aspecto importante a la hora de la creación de un videojuego y es la optimización del mismo en cuanto al procesamiento grafico el cual debe tener en cuenta la calidad gráfica de los objetos presentes en el escenario y el realismo que estos tengan, así como la rapidez o fluidez con que las imágenes son presentadas en la pantalla del dispositivo cada segundo, tal como afirma Damián [47] que para una perfecta visualización e inmersión en el entorno se necesitan un mínimo de 30 fotogramas por segundo, por lo que es una prioridad que este videojuego tenga esas características para cualquier dispositivo ya sea de gama alta o baja.

Figura 67. Test de FPS con el juego ejecutándose.



Fuente: [Autor]

Para garantizar el buen funcionamiento del videojuego en un celular de gama baja siendo este un dispositivo de la marca Alcatel y de modelo A7, basándose en el procesamiento grafico se realizaron una serie de pruebas en donde se midieron el numero fotogramas por segundo mientras el videojuego se ejecutaba, haciendo uso de la aplicación llamada *FPStest* la cual dio como resultado un numero de FPS estables de 56 tal como se observa en la figura 67 por lo que se cumple

[47] D. I. F. Montes, «TARJETAS GRÁFICAS: SU IMPACTO EN LOS USUARIOS DE VIDEOJUEGOS,» *Revista Digital de Tecnologías Informáticas y Sistemas,* vol. 3, 2019.

satisfactoriamente el requerimiento para una correcta visualización de este tipo de plataformas.

El producto final realmente supera las expectativas iniciales y se logra que este a la altura de muchos otros productos cuyo contenido está basado en la realidad virtual, ya que no solamente paso por las diferentes opiniones de usuarios normales, sino que también recibió sugerencias de personas cuyo trabajo o área de estudio es el diseño industrial por lo que el proyecto tubo un proceso de mejoras y rediseño constante hasta la versión final el cual se acomoda a todas las sugerencias recibidas a lo largo de la creación del mismo y teniendo en cuenta cada una de las características que debería tener todos los videojuegos de este tipo, ya sea por realismo, inmersión, tiempo real, entre otros.

5. Conclusiones.

El videojuego desarrollado para dispositivos Android recrea al usuario la sensación de estar en medio de un laberinto en la vida real gracias a sus gráficos tridimensionales que realmente logran una inmersión lo suficientemente aceptable al igual que otros productos de esta misma categoría.

El punto en el cual este producto estaba destinado era la realidad virtual y que igualmente lograra una interacción con el entorno a tal nivel que el jugador realmente se sintiera dentro de una situación en la cual estuviera perdido, al encontrarse dentro de un espacio desconocido en este caso un laberinto, y aunque por sugerencias de los mismos jugadores para evitar desorientación se logró satisfactoriamente una representación de lo que sería en la vida real resolver un laberinto gigante sin tener ningún tipo de conocimiento acerca de este.

Las medidas que se tomaron como referencia para la creación de los muros en la escena principal no fueron completamente exactas ya que se amplió del escenario para así crear un laberinto más grande que el que se tomó como ejemplo, esto ocurrió debido a los errores de escala que existen dentro del protocolo de diseño de unity, aunque ninguno de los jugadores realmente llego a apreciar esto siendo estos tan pequeños por lo que la experiencia de juego no quedo afectada en lo absoluto, si se desea realizar un producto de mayor profesionalidad se debería usar un software de diseño especializado en donde se tuvieran en cuenta cada una de las medidas de manera más exacta y así solamente configurar la escala deseada dentro de unity.

La herramienta de animación que posee blender está destinada mayoritariamente para objetos relacionados con movimientos no rígidos sino de naturaleza flexible, ya sea por seres humanos o animales, por lo que las deformaciones que este software añade a su interfaz física de movimiento afectan visualmente a los robots ya que estos no pueden deformarse de ninguna forma siendo estos de articulaciones mecánicas simples, por esto y para añadirle mayor realismo se optó por delimitar todos los movimiento para así evitar estas deformaciones.

La simulación de movimiento de cada uno de los robots cumple de manera satisfactoria con el comportamiento real de estas máquinas dándole al videojuego un toque de realismo importante e interés mayor por conocerlos e informarse acerca de su funcionamiento.

La aplicación creada a partir del videojuego diseñado en unity funciono de manera correcta en todos los dispositivos con sistema operativo Android que se probaron, ya que no se produjo ningún tipo de error ni en el proceso de instalación ni en el manejo tanto del control como de la vista del jugador, lo único que se tiene que tener en cuenta a la hora de instalar esta aplicación es que el celular tenga giroscopio dentro de su hardware ya que para la mayoría del contenido de realidad virtual actualmente necesita de este sensor para su correcta visualización, aunque en la mayoría de dispositivos de gama media en adelante poseen este tipo de sensores por lo que no es un problema demasiado importante.

Este videojuego no requiere de un muy grande procesamiento grafico debido a su diseño simple y poco texturizado por lo que es una ventaja a considerar ya que no es necesario que el usuario tenga un celular de gama alta, al contrario, solamente tendrá que cumplir con el requisito de tener el giroscopio.

El método para moverse dentro del entorno virtual que en este caso en un control bluetooth funciono de manera correcta con todos los dispositivos en los que se probó el juego, además este prototipo está destinado para funcionar con la mayoría de mandos de esta naturaleza inalámbrica que existan en el mercado.

Debido a que muchos de los mandos inalámbricos tienen diferentes modos para su uso en el dispositivo móvil ya que no todos estos modos están destinados para jugar es recomendable que el usuario tenga en cuenta las instrucciones del propio control para evitar problemas dentro de la aplicación.

Este prototipo paso por muchas etapas de rediseño y correcciones gracias a que diferentes usuarios lo jugaron y dieron sus opiniones al respecto las cuales se tuvieron en cuenta en todas y cada una de las modificaciones que se realizaron ya que no solamente se deben obtener un buen funcionamiento del mismo sino una buena calidad gráfica que sea visualmente aceptable para todos los jugadores.

Bibliografía

[1] R. A. Española., *Diccionario de la lengua española.,* Madrid, España, 2019.

[2] X. Basogain, M. Olabe, K. Espinosa, C. Rouèche y J. C. Olabe, «Realidad Aumentada en la Educación: una tecnología emergente,» *Escuela Superior de Ingeniería de Bilbao, EHU. Recuperado de http://bit. ly/2hpZokY,* 2007.

[3] M. Billinghurst, R. Grasset, R. Green y M. Haller, «Inventing the future down under: the human interface technology laboratory new zealand [hit lab nz],» *ACM SIGGRAPH Computer Graphics,* vol. 39, pp. 18-23, 2005.

[4] R. Gubern, Del bisote a la realidad Virtual, Barcelona, 1996.

[5] F. J. P. Martínez, «Presente y Futuro de la Tecnología de la Realidad Virtual,» *Creatividad y sociedad,* 2011.

[6] F. R. A. N. C. I. S. C. O. J. A. V. I. E. R. PÉREZ MARTÍNEZ, «La Tecnología de la Realidad Virtual: Hoy y Mañana,» *Ciclos Complutenses Ciencia y Tecnología,* 2006.

[7] J. A. Incera, «Nuevas Interfaces y sus Aplicaciones en las Tecnologías de Información y Comunicaciones,» *Reporte técnico,* pp. 1-1007, 2007.

[8] J. Martínez, J. P. Molina, A. S. García, D. Martínez y P. González, «Desarrollo de un guante de datos con retorno háptico vibro-táctil basado en Arduino,» de *Interacción 2009-Jornadas de Realidad Virtual*, 2009, pp. 1-10.

[9] A. Vélez Escorial, «Diseño mecánico de un interfaz háptico para realidad virtual,» 2011.

[10] M. L. Pinto-Salamanca, J. I. Sofrony-Esmeral y D. F. Jiménez, «Detección de colisiones con librerías V-Collide y PhysX para interacción virtual con interfaces hápticas,» *Revista de Investigación, Desarrollo e Innovación,* vol. 5, pp. 119-128, 2015.

[11] P. A. Loaiza Gómez, M. A. Vega Uribe y others, «Simulador tridimensional para el desarrollo de la habilidad de transferencia de aros para cirugía laparoscopia usando una interfaz háptica:" Silaph 3D",» 2015.

[12] G. Méndez, E. Obviedo, G. Fallas, C. Vega y A. Méndez, «Análisis de las herramientas Unity y Blender para el desarrollo de videojuegos con un enfoque educativo,» *Escuela de Computación, Tecnológico de Costa Rica,* p. 13, 2014.

[13] D. A. A. Daniel, Á. C. J. Antonio y S. Bernal, «Visualización 3D en Blender para Manejar un Brazo Robótico».

[14] O. A. V. Alban y D. E. G. Villamarín, «Herramienta software para la práctica de la robótica quirúrgica,» *Ingeniería y Universidad,* vol. 19, pp. 7-25, 2015.

[15] A. Alhamidi, «Simulation and Evaluation of Rescue Robots in Blender,» de *人工知能学会全国大会論文集 第25 回全国大会 (2011)*, 2011.

[16] I. Ouazzani, «Manual de creación de videojuego con Unity 3D,» 2012.

[17] A. Moncada y E. David, «Creación de un videojuego multiplataforma 2.5 d en unity 3d y blender,» 2017.

[18] T. Alemañ Baeza, «Desarrollo de un videojuego para móviles con Unity,» 2015.

[19] O. F. Gómez y U. E. Gómez, «Simulación Cinemática de un Robot Seguidor de Línea para el Desarrollo del Videojuego de Programación Rusty Roads en el Framework Unity,» *Información tecnológica,* vol. 28, pp. 55-64, 2017.

[20] S. Moreno Cano y others, «Desarrollo de videojuegos en Unity para educación,» 2019.

[21] G. A. Pescador Barreto y others, «Deep learning en videojuegos: Aprendizaje por refuerzo en el entorno Unity,» 2019.

[22] S. Belli y C. L. Raventós, «Breve historia de los videojuegos,» *Athenea Digital. Revista de pensamiento e investigación social,* pp. 159-179, 2008.

[23] D. Blanco, S. Al Ansari, C. Castejón Sisamón, B. López Boada y L. E. Moreno, «Manfred: robot antropomórfico de servicio fiable y seguro para operar en entornos humanos,» 2005.

[24] A. O. Baturone, Robótica: manipuladores y robots móviles, Marcombo, 2005.

[25] J. Berrio, E. Arcos, J. Zuluaga y S. Corredor, «Diseño y construcción de un robot cartesiano de 3 grados de libertad,» *Memorias,* 2015.

[26] C. Pillajo y J. E. Sierra, «Human Machine Interface HMI using Kinect sensor to control a SCARA Robot,» de *2013 IEEE Colombian Conference on Communications and Computing (COLCOM)*, 2013.

[27] M. F. Pedraza, P. F. Cárdenas, F. J. Rodríguez y E. Yime, «Aproximación al diseño de robots paralelos, análisis de caso del robot delta,» *Memorias,* 2015.

[28] A. Rodríguez Bermejo, «Desarrollo y programación de un brazo antropomórfico para su uso en centros de tratamiento de residuos,» 2013.

[29] J. Serracín, I. Moreno, T. Vásquez y I. Bonilla, «Prototipo de Robot Paralelo Delta para fortalecer el proceso educativo a nivel superior,» de *Memorias de Congresos UTP*, 2017.

[30] S. M. Samartha, R. S. Kotwal y V. M. Vidyashri, «Exploring the world of virtual reality gaming with Google Cardboard and Unity,» *International Journal of Modern Trends in Engineering and Research,* 2016.

[31] J. Linowes, Unity virtual reality projects, Packt Publishing Ltd, 2015.

[32] I. Sánchez, F. J. S. Marín, R. S. Villagomez y others, «Diferencias de género en la orientación espacial en un ambiente virtual,» *Revista Mexicana de Psicología,* vol. 28, pp. 211-216, 2011.

[33] P. S. Pérez, Manual de modelado y animación con Blender, Universidad de Alicante, 2011.

[34] J. Cerdá Boluda, «Blender: Interpolación y extrapolación en animaciones.,» 2019.

[35] A. G. Montufar Gallardo, «Desarrollo de una aplicación web para el aprendizaje de la Cinemática con animaciones en 3D utilizando la metodología DESED.,» 2016.

[36] R. D. Morelli, H. A. P. Ctenas y L. S. Nieva, «Modelado Paramétrico 3D, Render y Animación con Software Libre: Interacción Freecad+ Blender,» de *Geometrías & Graphica 2015 Proceedings. III Conferencia Internacional de Aproged. XI International Conference on Graphics Engineering for Arts and Design. I*, 2015.

[37] R. Rosales, D. Murillo y R. Miguelena, «Modelado y Animación 3D,» *El Tecnológico,* vol. 27, pp. 11-12, 2017.

[38] M. E. S. Lapo, «Análisis del equilibrio gráfico-narrativo presente en la experiencia interactiva de los videojuegos,» *Con A de animación,* pp. 94-111, 2020.

[39] J. L. G. Sánchez, N. P. Zea, F. L. Gutiérrez y M. J. Cabrera, «De la Usabilidad a la Jugabilidad: Diseño de Videojuegos Centrado en el Jugador,» *Proceedings of INTERACCION,* pp. 99-109, 2008.

[40] A. González Aller, «Celestial Road-Videojuego de laberintos en Realidad Virtual con Unity,» 2019.

[41] F. Etxeberría y others, «Videojuegos y educación,» 2000.

[42] J. Díaz, C. Queiruga, C. B. Tzancoff, L. Fava y V. Harari, «Robótica Educativa y Videojuegos en el Aula de la Escuela Educational Robotics and Videogames in the Classroom».

[43] O. Miglino, L. S. Sica y M. L. Nigrelli, «Videojuegos de rol, simulaciones por ordenador, robots y realidad aumentada como nuevas tecnologías para el aprendizaje: guía para profesores, educadores y formadores,» *Videojuegos de rol, simulaciones por ordenador, robots y realidad aumentada como nuevas tecnologías para el aprendizaje,* pp. 1-138, 2013.

[44] C. Morales, M. C. Rivara y others, «Animación de robot y figuras humanas,» *Revista Facultad de Ingeniería-Universidad de Tarapacá,* vol. 13, pp. 31-38, 2005.

[45] R. Montero Sacristán y others, «Modulo de creación de laberintos en Unity,» 2016.

[46] I. E. Sutherland, «The ultimate display,» *Multimedia: From Wagner to virtual reality,* vol. 1, 1965.

[47] D. I. F. Montes, «TARJETAS GRÁFICAS: SU IMPACTO EN LOS USUARIOS DE VIDEOJUEGOS,» *Revista Digital de Tecnologías Informáticas y Sistemas,* vol. 3, 2019.