UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE CUENCA

CARRERA DE INGENIERÍA ELECTRÓNICA

DESARROLLO DE UN SISTEMA PROTOTIPO DE AGENTE VIRTUAL PARA COMUNICACIÓN HUMANO-MÁQUINA UTILIZANDO BERT

Trabajo de titulación previo a la obtención del título de Ingeniero Electrónico

AUTORES: DAVID ALEJANDRO OCHOA MUÑOZ
JONNATHAN ALEXANDER PINTADO YUNGA
TUTOR: ING. CHRISTIAN RAÚL SALAMEA PALACIOS, Ph.D.

Cuenca - Ecuador
2022

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, David Alejandro Ochoa Muñoz con documento de identificación N° 0705133098 y Jonnathan Alexander Pintado Yunga con documento de identificación N° 0104895438; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Cuenca, 11 de marzo del 2022

Atentamente,

David Alejandro Ochoa Muñoz

0705133098

Jonnathan Alexander Pintado Yunga

0104895438

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, David Alejandro Ochoa Muñoz con documento de identificación N° 0705133098 y Jonnathan Alexander Pintado Yunga con documento de identificación N° 0104895438, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto técnico: "Desarrollo de un sistema prototipo de agente virtual para comunicación humano-máquina utilizando Bert", el cual ha sido desarrollado para optar por el título de: Ingeniero 9lectrónico, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, 11 de marzo del 2022

Atentamente,

David Alejandro Ochoa Muñoz

0705133098

Jonnathan Alexander Pintado Yunga

0104895438

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Christian Raúl Salamea Palacios con documento de identificación N° 0102537180, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DESARROLLO DE UN SISTEMA PROTOTIPO DE AGENTE VIRTUAL PARA COMUNICACIÓN HUMANO-MÁQUINA UTILIZANDO BERT, realizado por David Alejandro Ochoa Muñoz con documento de identificación N° 0705133098 y Jonnathan Alexander Pintado Yunga con documento de identificación N° 0104895438, obteniendo como resultado final el trabajo de titulación bajo la opción Proyecto técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Cuenca, 11 de marzo del 2022

Atentamente,

Ing. Christian Raúl Salamea Palacios, Ph.D.

0102537180

ÍNDICE

AGRADECIMIENTOS

Agradezco a mis padres Patricio Ochoa y Teresa Muñoz por siempre estar alentándome a seguir adelante y cumplir mis sueños. A mi hermana Patricia Ochoa por ser mi apoyo y motivación para culminar este proceso de educación universitaria. A mi hijo Gabriel Ochoa por ser mi pilar fundamental y mayor motivación. A mi persona especial Angélica Peralta por apoyarme y acompañarme en los días y noches de estudios. A mis amigos que estuvieron dándome ánimos para no rendirme y culminar esta etapa.

Agradezco a nuestro tutor Dr. Christian Raúl Salamea Palacios PH. D por orientarnos y brindándonos su tiempo para el desarrollo y culminación de este proyecto de titulación.

David Ochoa Muñoz.

Al concluir una de las etapas maravillosas de mi vida, quiero expresar mi más sincero agradecimiento a la Universidad Politécnica Salesiana por darme acogida para la realización de mis estudios, abrirme las puertas hacia nuevos horizontes y guiarme por un buen camino para cumplir mis metas y sueños anhelados.

A mis queridos padres José y Gladys, por su apoyo incondicional, por estar siempre a mi lado en los momentos más duros en el transcurso de mi vida profesional, por sus consejos, por el apoyo por el amor que me tienen, por la vibra y energía que siempre me han estado motivando para no caer en el abismo y seguir adelante.

Agradezco Dios por todo lo que tengo, por todo lo que he vivido, por permitirme cumplir una meta más en vida, por todas las bendiciones recibidas.

Jonnathan Pintado Yunga.

.

DEDICATORIA

El presente proyecto de titulación lo dedico a mi familia por el amor y sacrificio en todos estos años, por el apoyo moral que me han brindado en cada etapa de mi vida, sin ustedes no lo hubiera logrado. A todas las personas que me han apoyado y han hecho que el trabajo se realice con éxito en especial a aquellos que me abrieron las puertas y compartieron sus conocimientos.

David Ochoa Muñoz.

Dedico este proyecto de titulación a mis queridos Padres José y Gladys, por todo el apoyo incondicional, por el esfuerzo y sacrificio que me han brindado en mi vida para seguir adelante. A mi hermano Bryan por su cariño y apoyo que me ha generado en la vida. A mi esposa Daniela Chapa y a mi hijo Thiago Pintado por la paciencia, confianza y el amor que han demostrado, por ser mis pilares fundamentales, mis motores de arranque, por ser el motivo de mi vida, doy gracias a Dios por tenerles en mi vida, quienes en el camino que me acompañaron en mis estudios, a mis compañeros con los cuales la universidad se volvió tiempo de alegrías y de esfuerzo mutuo donde se aprendió nuevas formas de cómo es la vida, docentes que nos han aportado sus conocimientos diarios para poder superarnos día a día.

Jonnathan Pintado Yunga.

GLOSARIO

HMI: Interacción humano máquina
VA: Agente Virtual
TIC: Tecnologías de la información y de la comunicación.
IA: Inteligencia artificial.
BERT: Bidirectional Encoder Representations from Transformers (Representaciones de Codificador Bidireccional de Transformadores).
API: Interfaz de programación de aplicaciones.

RESUMEN

Este proyecto técnico consiste en un sistema de diálogo que permita la comunicación humano-máquina, para la comunicación se diseñó un agente virtual que mediante una Inteligencia Artificial responde a las preguntas que se le genera de un texto previamente ingresado. Para desarrollar este proyecto se realizó una investigación de los diferentes sistemas de diálogo existentes y como se los están aplicando. Luego se crea un agente virtual en 3D mediante el software Blender y se lo exporta a Unity para realizar los diferentes movimientos que se realiza al hablar, al mismo tiempo se trabaja en el software Python con inteligencia artificial que sea capaz de recibir mediante micrófono una pregunta de un texto dado y nos dé una respuesta valida, todo esto se lo realiza con las diferentes librerías existentes en Python. Seguidamente se realiza una comunicación entre Python y Unity para que funcionen de manera adecuada, esto sería que al momento que se escuche el audio de respuesta el agente virtual realice los movimientos correspondientes. Se explica cómo se realizó el agente virtual en Blender y posterior mente exportado a Unity, también como se desarrolló el sistema de dialogo y como se llevó a cabo correctamente la comunicación entre estos dos programas. Finalmente se realiza un análisis de resultados mediante dos pruebas, la primera consistió en la velocidad de respuesta, esta se la realizó con diferentes tamaños de texto efectuando 9 preguntas para cada caso, y se controló el tiempo que demora en responder en las diferentes extensiones de texto, y la segunda prueba fue de vocalización que se la realizo haciéndole diferentes preguntas que generen diferentes respuestas para luego verificar los movimientos de la boca que realiza el avatar, mediante estas dos pruebas se presenta la evidencia que el proyecto técnico cumple con los resultados esperados.

INTRODUCCIÓN

En la actualidad el uso de los avatares o agentes virtuales animados han ido creciendo notoriamente ya que con el desarrollo de la computadora y con la llegada del internet los desarrolladores de tecnologías fueron teniendo nuevas ideas, obteniendo avatares.

Los avatares permiten interactuar de ciertas maneras tales como guías, ayudantes, también brindan ayuda en búsquedas de información sobre un tema, hasta pueden llegar a mantener ciertas conversaciones de algún tema en general, es por ello que estos avatares generan respuestas de manera natural.

El presente proyecto "Desarrollo de un Sistema Prototipo de Agente Virtual para comunicación Humano-Máquina utilizando BERT", propone un sistema basado en la interacción persona-maquina creando un avatar 3D realista mediante la integración de programa de creación, animación y lenguaje de programación asignándole al avatar de movimiento, el cual tiene como finalidad generar una comunicación en tiempo real aplicando inteligencia artificial para aprender cierta información.

La animación generada del agente virtual se desarrolló a partir de Blender que es el diseño del personaje su texturizado y renderizado, Unity para los movimientos como es el caso del cuerpo, las manos, la cabeza, músculos faciales y Python para lo que es el sistema de dialogo con el objetivo de desarrollar la interacción en tiempo real.

Python en conjunto con Unity, fue posible generar el dialogo al avatar mediante scripts que se desarrolló en función de la interacción que se busca en el sistema humano-maquina, permitiendo que el avatar genere expresiones.

El sistema utiliza Inteligencia Artificial (IA) con arquitectura BERT, mediante la implementación de librerías de Python ayudando en la conversión de audio a texto dando respuesta en tiempo real.

ANTECEDENTES DEL PROBLEMA DE ESTUDIO

A lo largo de los años, el humano ha ido dejando la evolución biológica por la evolución tecnológica, donde fue creando rápidamente avances tecnológicos para lograr satisfacer un estilo de vida cómoda, dando un inicio de interacción humano-máquina durante horas. Hoy en día el contacto del usuario con la máquina están muy avanzados ya sea por medio de controladores o Interfaces Humano-Máquina (HMI), donde el desarrollo de estas tecnologías hace que las máquinas sean capaces de responder al diálogo de una manera amigable y confiable [1].

La Interacción Humano-Máquina (HMI), es una disciplina enfocada en diseños, evaluaciones e implementaciones de sistemas computacionales interactivos para el uso humano, donde ha generado un crecimiento acelerado cambiando fundamentalmente la computación, donde se tienen las siguientes áreas: Realidad Virtual, 3D, Realidad Aumentada, Lenguaje natural, reconocimiento e interpretación de voz, reconocimiento de gestos [2].

Para generar una Interacción Humano-Máquina (HMI), se da por medio de agentes virtuales (VA), que son herramientas informáticas con la aparición de personajes virtuales representando mediante gráficas 3D con movimientos corporales, donde son utilizados como imágenes de ALTER EGO en redes sociales, sitios web, permitiendo responder de una forma automática, las necesidades de los usuarios [3]. Gracias a estas tecnologías es posible interactuar con agentes virtuales, donde nos pueden dar información de climas, tráficos, lugares turísticos, hasta poder ser parte de videojuegos en 3D realidad virtual, incluso se puede interactuar en tiempo real.

En el desarrollo de videojuegos se detectan herramientas de realidad virtual teniendo las siguientes que son el Unreal Engine, Cry Engine y el Unity 3D siendo esta opción la más utilizada por su fácil forma de uso teniendo un gran apoyo de grandes Industrias como es el Google, Nintendo, ya que nos permiten realizar multiplataformas [4].

Unity 3D además de ser una multiplataforma también es considerada un ambiente de desarrollo, debido a que permiten interacciones con multiprogramas externos y uso de diferentes lenguajes de programación, ya que este software es de

uso libre, también nos permite esta herramienta a crear una gran compatibilidad entre Blender [4].

Blender es un software que contiene su propio motor de videojuegos donde tiene varias herramientas que son muy útiles al momento de crear personajes, objetos y animaciones para un video juego [4].

Se han desarrollado diversas tecnologías de la información y de la comunicación (TIC), permitiendo el manejo de herramientas para la comunicación con el apoyo del Sistema Inteligente por medio de la Inteligencia Artificial (IA), que es uno de los sistemas capaces de realizar interacciones y tareas a partir de la comunicación comandados mediante la voz [5], dando un paso importante y a la vez dando buenos resultados permitiendo así la creación de agentes virtuales realistas, es por ello que impulsan retos hacia las tecnologías donde se puedan realizar una interacción fluida capaz de llevar a un dialogo coherente, permitiendo procesos de enseñanza y aprendizaje [6][7].

Las nuevas actualizaciones tecnológicas tales como avatares 3D conversacionales, agentes conversacionales, y asistentes interactivos virtuales son herramientas que han permitido innovar y transformar la forma de comunicación del individuo con respecto a la interacción Hombre-Máquina [8], donde resulta uno de los más exponentes medios de dialogo, es por ello que existen investigaciones y estudios realizados respecto a este tema, proponiendo técnicas para que se pueda llevar a cabo esta interacción [9].

En base a la presencia del mundo virtual la Interacción pretende representarse no solo en el ámbito computacional sino en un ámbito enfocado a la realidad, es por ello por lo que se propone diseñar un agente virtual Inteligente que tenga una Interacción Humano-Máquina aplicando técnicas de procesamiento de lenguaje natural y utilizando la arquitectura de BERT para la interacción. Este Agente Virtual será diseñado con rasgos y expresiones humanas, voz, animación e interacción. Para el diseño de este Agente Virtual se usará herramientas de Unity, el modelo Blender que nos permitirán el desarrollo y la creación del personaje, la herramienta de Python el cual nos permitirá desarrollar el sistema de dialogo entre el usuario y el agente virtual utilizando la arquitectura BERT.

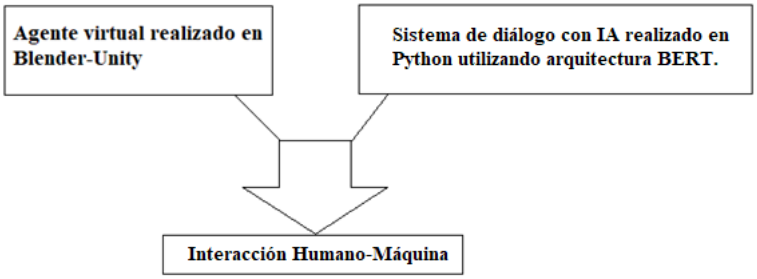


Ilustración 1. Diagrama del proyecto técnico (Autor)

JUSTIFICACIÓN (IMPORTANCIA Y ALCANCES)

El Grupo De Investigación En Interacción Robótica Y Automática (GIIRA) de la Universidad Politécnica Salesiana desarrollan continuamente proyectos de ciencia y tecnología en la Interacción Humano-Máquina (HMI), actualmente se encuentran desarrollando el proyecto "**AGENTE VIRTUAL INTELIGENTE",** el mismo servirá para brindar información mediante la Interacción Humano-Máquina (HMI) por medio de la creación de un Agente Virtual (VA) con el fin de brindar información precisa, oportuna y permanente.

Este Agente Virtual (VA) utiliza métodos de Inteligencia Artificial, dicho agente está dotado de rasgos y expresiones humanas, voz y animación. En la Actualidad la mayoría de los agentes virtuales han sido desarrollados con fines comerciales para atención de ventas.

El proyecto consta de varias etapas, siendo una de ellas el desarrollo de un Agente Virtual (VA) y su Interacción Humano-Máquina (HMI), para un sistema de diálogo con el usuario, esta etapa es el objetivo de este proyecto técnico a realizarse, donde es una herramienta tecnológica que nos brindara un mejor servicio permanente y a la vez sea capaz de interactuar mediante la voz como un ser humano real compartiendo información por medio del diálogo, a modo de conversación con los usuarios, este agente virtual se lo realizará en software libre, cuya personalización será con movimientos en 3D, teniendo un aspecto real al momento de conversar generando movimientos básicos como es el movimiento bucal a la hora de hablar con dicho usuario, y también permita responder preguntas que son realizadas por los usuarios. Los resultados de este proyecto nos van a permitir establecer un diseño adecuado donde nos brindara información y orientación.

OBJETIVOS

OBJETIVO GENERAL

Desarrollar un sistema prototipo Agente Virtual para comunicación Humano-Máquina utilizando BERT.

OBJETIVOS ESPECÍFICOS

- Aplicar la comunicación existente entre Blender y Unity para posibilitar la creación Interactiva del Agente Virtual.
- Diseñar un agente virtual humanoide usando Unity y Blender que servirá como recurso visual de la Interacción Humano-Máquina.
- Diseñar la estructura del diálogo Humano-Máquina utilizando la arquitectura BERT.
- Adaptar el sistema de diálogo Humano-Máquina mediante librerías existente en Python para integrarlo con el Agente Virtual de tal manera que se posibilite la Interacción.

CAPÍTULO 1

1. FUNDAMENTO TEÓRICO

Para este proyecto técnico se realiza una descripción de los métodos empleados y se describirán conceptos más relevantes que sustentan el trabajo desarrollado.

1.1. UNITY 3D

Unity es un motor de creación de videojuegos lanzado el 1 de Junio del año 2005, este motor nos permite crear videojuegos interactivos, contenidos, diseños o animaciones en 3D [10].

Unity nos ofrece tres lenguajes a la hora de programar, que son:

- UnityScript
- Boot
- C#

Una de las ventajas que tiene Unity 3D es multiplataforma que se puede desarrollar para PCs tanto en sistema operativos Windows, Linux, Mac, también para teléfonos iOS, Android, Windows phone e incluso para consolas como Wii, Xbox360, XboxOne, PS3 y PS4, entre otras [11].

Unity ganó varios premios, en 2006 fue finalista por el mejor uso de gráficos en Mac Os X de Apple, también en 2010 ganó el premio de innovación tecnológica de Wall Street Journal [10].

1.2. MICROSOFT VISUAL STUDIO

Este software contiene un entorno de desarrollo integrado, debido a que se encuentra en compatibilidad con Unity y a través del lenguaje de programación C#, el cual permitió la creación del código script donde se pueda controlar las funcionalidades y las integraciones de los agentes virtuales [8].

1.3. MODELADO EN 3D

El modelado 3D utiliza software que crea una representación matemática de un objeto de una manera tridimensional. El objeto creado es un modelo 3D, donde

se puede utilizar para varias aplicaciones como es en la industria del cine, televisión, desarrollo de productos, arquitectura, videojuegos, ciencia o medicina, estos usan el modelado 3D para realizar visualización, simulaciones o renderizar deseños gráficos [12].

Para la creación del modelado 3D tenemos diferentes softwares libres [13], donde se mencionaran a continuación:

Nombre	**Nivel**	**Sistema Operativo**
SketchUp	Principiante	Windows y Mac
Blender	Intermedio	Windows, Mac y Linux
MakeHuman	Intermedio	Windows, Mac, Linux
Meshmixer	Intermedio	Windows, Mac
NanoCAD	Intermedio	Windows
OpenSCAD	Intermedio	Windows, Mac y Linux

Tabla 1. Softwares libres para modelado 3D

1.4.BLENDER

Blender es un software multiplataforma y gratuito que nos permite crear, renderizar y animar objetos, personajes y escenas en 3D, generando al final una imagen 2D, esto lo realiza mediante motores gráficos. [14].

Blender también incorpora un editor de video, permitiendo crear un proyecto sin necesidad de usar otro tipo de software, también lee varios formatos y se puede importar o exportar el contenido de o hacia otros programas. Finalmente, Blender dispone de 2 motores de renderizado, el Blender Internal y el Cycles, logrando así un renderizado profesional, esto lo hace un muy buen software para el modelado 3D [4].

1.5. INTELIGENCIA ARTIFICIAL

Desde las últimas décadas el hombre junto con la tecnología ha ido creciendo poco a poco en la Inteligencia Artificial teniendo en cuenta algunas reseñas históricas que vale señalar lo siguiente:

Alan Turing en el año de 1950, realiza una prueba donde se puede reconocer los comportamientos inteligentes, este tipo de comportamiento es recibido por una terminal que es conectada hacia un interlocutor que es desconocido por parte del humano [3].

Alled Newell, Herbet Simon de la CMU, J.C.Shaw, estos investigadores informáticos crean el primer lenguaje de Inteligencia Artificial llamado IPL II (Information Processing Languaje) [3].

1.6. PYTHON

Python es un lenguaje de programación que utiliza un script, archivo que contiene un código escrito que permite identificar la extensión de Python. El lenguaje de script ejecuta instrucciones escritos en un lenguaje de alto nivel, lo que significa que hace lo que el programa dice, esto da una ventaja respecto a los lenguajes compilados ya que el lenguaje script es portable y más flexible. Python tiene algunas características de los lenguajes compilados así que se podría decir que es un lenguaje de programación semi-interpretado. En Python, el código fuente se pasa a pseudo código máquina intermedio llamado bytecode, generando archivos .py o. ipynb los cuales se ejecutarán en varias ocasiones [15]

1.7. BERT

Bert es un sistema basado en IA previamente entrenado del lenguaje natural, fue desarrollado por Google, este sistema permite comprender de mejor manera el lenguaje natural que usan los usuarios para poder ser usado en amplias tareas como lo son respuestas a preguntas o resultados de búsquedas [16].

1.8. SISTEMAS DE DIÁLOGO

A los sistemas de diálogo se les conoce también como sistemas convencionales, estos sistemas son tecnologías que en la actualidad son muy generadas en el mundo virtual, ya que facilita la interacción natural humano máquina mediante el habla, son herramientas esenciales para desarrollar servicios de atención tales como: cliente, telefonía, periodismo, tutores, etc [3].

Este tipo de sistemas de diálogo son aplicados en diferentes tareas como son: compras de diferentes productos, reservaciones de viajes, call centers, comunicación entre personajes que hablan en diferentes idiomas [3].

Estos sistemas conllevan ciertas características que facilitan una mejor conexión al usuario:

- Reconocimiento del habla
- Extracción de información semántica de la frase
- Gestor de dialogo
- Generación de respuestas
- Sinterización de voz [3].

En los últimos años se ha generado sistemas multimodales donde los usuarios expresan peticiones al sistema dado de una manera intencionada apoyándose en la vinculación de voz, texto, avatares [17].

En la Universidad de los Andes, Duque desarrolla una interfaz de interacción multimodal para ayudar en la exploración de mapas geográficas, mediante gestos y comando de voz, que son captados por medio de un sensor llamado Kinect, dicha interacción va a permitir generar comandos verbales. Se realizaron pruebas a varios usuarios entre las edades de 20 y 40 años, que dieron un desempeño del 70% de reconocimiento de voz y un 74% de reconocimiento de gestos por medio de las manos [18].

En el año 2015 en Polonia, Pisz, desarrolló un proyecto llamado FaceController, que es un sistema de interfaz multimodal que permite controlar la máquina mediante movimientos de gestos faciales y comandos de voz utilizando el sensor Kinect [18].

La comunicación de estos sistemas se basa en la aplicación y uso de interfaces mediante lenguaje natural [17].

Dr. Richard Wallace junto con la comunidad del software de Alicebot desarrolla un lenguaje de programación que sirve para generar el lenguaje natural como una expresión verbal como es el caso del AIML (Artificial Intelligence Markup Languaje Lenguaje), que brindar y crea diversas frases de diálogos escritos por el diálogo de texto, cuyo lenguaje es compatible con el XML [19].

1.9.AVATARES VIRTUALES

La tecnología hoy en día ha permitido ha permitido el uso de la computadora para diversos tipos de herramientas en la sociedad ayudando a recrear un ambiente tecnológico, es por ello que surge la necesidad de realizar una interface de interacción Humano-Máquina [8].

Una de las formas en la que el ser humano se comunica es por medio del habla, donde las nuevas tecnologías e interacción de usuarios generan reconocimientos y procesamiento de voz [8].

Para que la interacción sea más realista se usa un agente virtual en 3D, cuyo nombre es llamado AVATAR, personaje creado por un computador enlazado a un mundo virtual, dicho personaje puede simular aspectos y características que se da en la comunicación humana [8].

Los avatares virtuales son de gran utilidad donde realizan diferentes áreas de aplicación como es el caso de la Industria, Educación, Medicina [3].

Alex y Jenn son agentes virtuales que se dedican en dar información sobre vuelos de diferentes aerolíneas [3].

Ilustración 2. Alex y Jenn agentes virtuales [3].

SGT STAR es un agente que brinda información sobre el puesto en marcha en el ejército de EE.UU [3].

Ilustración 3. SGT STAR agente virtual [3].

SPIKE un agente interactivo que permite dar información a estudiantes y profesores sobre la universidad de Gonzaga [3].

Ilustración 4. SPIKE agente virtual [3].

En junio del año 2003,Linden Lab, empresa estadounidense, creaSecond life en el que los habitantes diseñan su propio avatar para poder explorar el mundo virtual y lograr interactuar con otros habitantes, participando en varias actividades [20].

Ilustración 5. Secón life mundo virtual [20].

CAPÍTULO 2

2. MARCO METODOLÓGICO

En este capítulo se describe todo el proceso de desarrollo del Sistema de Interacción Humano-Máquina en tiempo real, su metodología utilizada para la creación del avatar, técnicas de Inteligencia Artificial (IA) y el lenguaje de programación para la animación del personaje.

2.1. INTERFAZ Y HERRAMIENTAS DE BLENDER UTILIZADAS EN EL DISEÑO PARA LA CREACIÓN DEL PERSONAJE

La interfaz por defecto de BLENDER está constituida por una serie de ventanas con cabeceras que permiten comprender el área en donde se va a trabajar:

- Ventana de Visualización 3D
- Modificadores
- Elementos de escena
- Panel de propiedades
- Línea de Tiempo
- Barra de estado

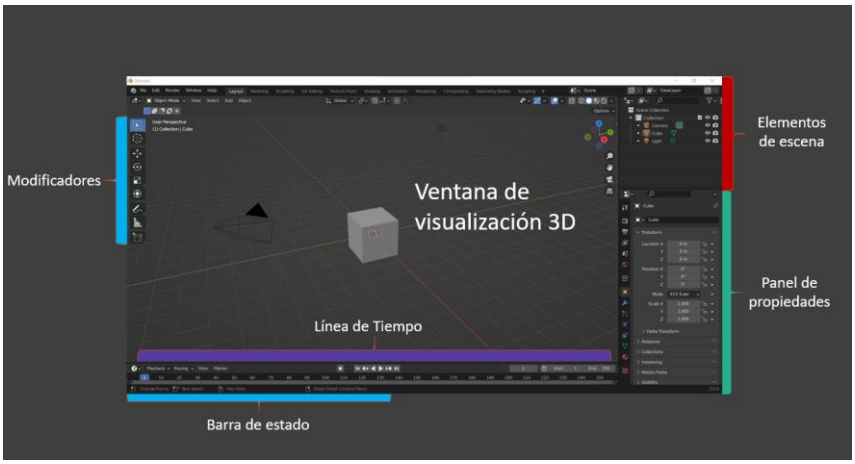


Ilustración 6. Interfaz de blender. (Autor)

Dado la visualización de la interfaz de Blender y conocido todas las vistas que se requiere para el diseño del avatar o agente virtual, se tomó en cuenta un aspecto visual, así como la personalidad y características del personaje.

CAPÍTULO 2. MARCO METODOLÓGICO

Para el proyecto se ha realizado una foto de distintas vistas tanto frontal como lateral tales que serán usadas para el diseño del personaje en 3D.

Ilustración 7. Vistas frontales y laterales. (Autor)

Una vez tomada en diferentes vistas se implementa las imágenes en el software Blender para crear el modelo en 3D.

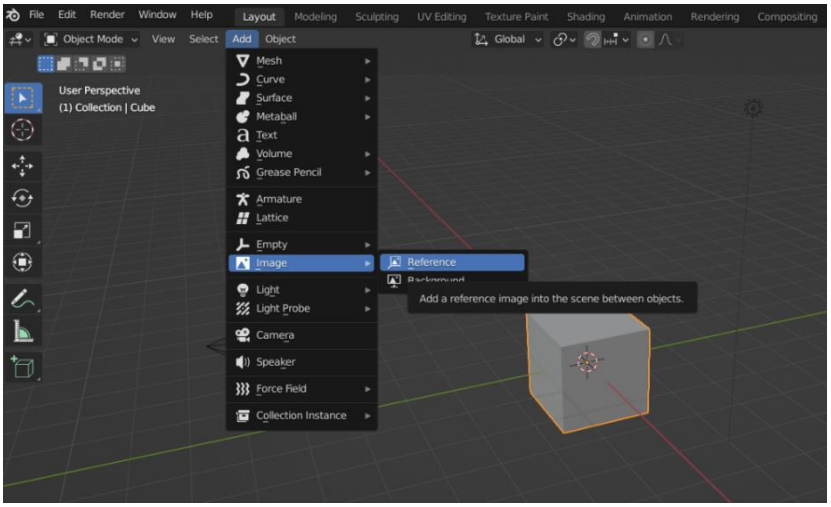


Ilustración 8. Implementación de las fotos tomadas para la creación del personaje. (Autor)

Como se puede apreciar en la ilustración 9, se tiene las dos referencias frontal y lateral que nos servirán como guía de perfiles para el diseño del personaje.

CAPÍTULO 2. MARCO METODOLÓGICO

Ilustración 9. Implementación de las vistas frontales y laterales. (Autor)

Para el modelamiento del personaje en 3D se ha utilizado el método de la malla donde tiene una relación eficaz con el diseño 3D, este método de la malla cabe recalcar que son estructuras basadas en caras, en 3D se los llama FACETAS, se edita con respecto al tamaño del volumen que se desea moldear. La malla crea nuevos vértices y duplica caras.

Luego de elegir dicho método, se agregó un cubo para el modelamiento del personaje, y se aplicó un sombreado de vista, método que nos sirve para mostrar los objetos en la vista 3D, en modo estructura.

Una vez que se muestra el modo estructura seleccionamos el cubo y eliminamos un lado de la cara, en la ventana de propiedades le aplicaremos el siguiente modificar llamado SIMETRIA, permitiendo realizar el mismo reflejo de la otra cara.

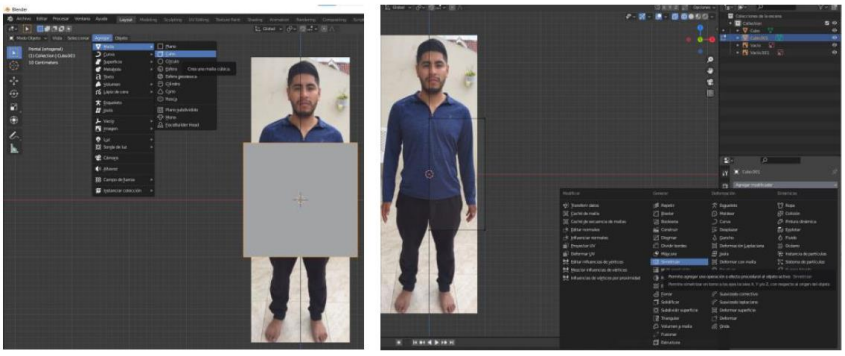


Ilustración 10. Modelamiento del personaje utilizando el método malla. (Autor)

Con la simetría aplicada en el cubo damos forma al personaje en sus vistas frontal y lateral quedándonos de la siguiente manera:

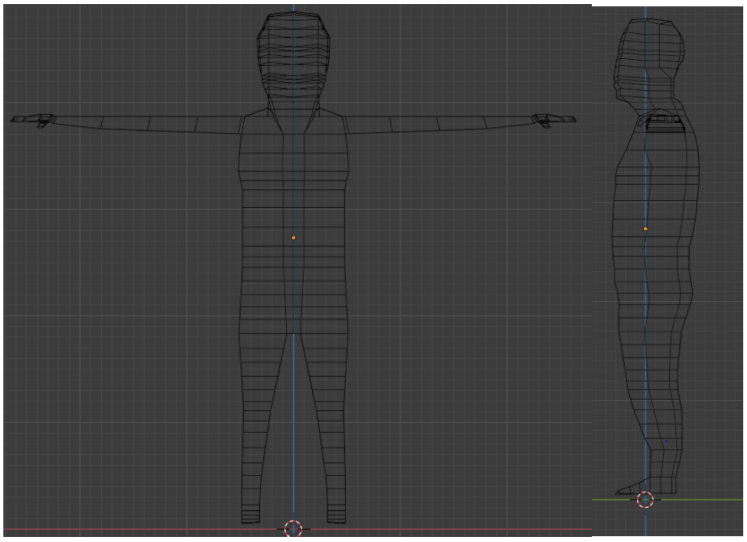


Ilustración 11. Modelamiento del personaje vista frontal y vista lateral. (Autor)

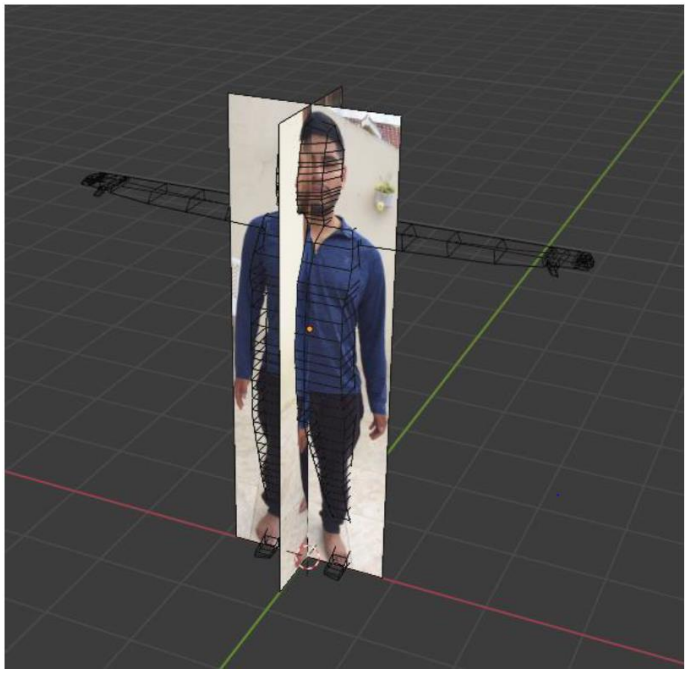


Ilustración 12. Diseño del personaje utilizando el método de extrusión. (Autor)

Al momento de que se termina de diseñar el personaje, se aplica la herramienta modo sólido, que permite tener una estructura para un moldeamiento real del personaje quedándonos de la siguiente manera, como se puede observar en la siguiente ilustración 13.

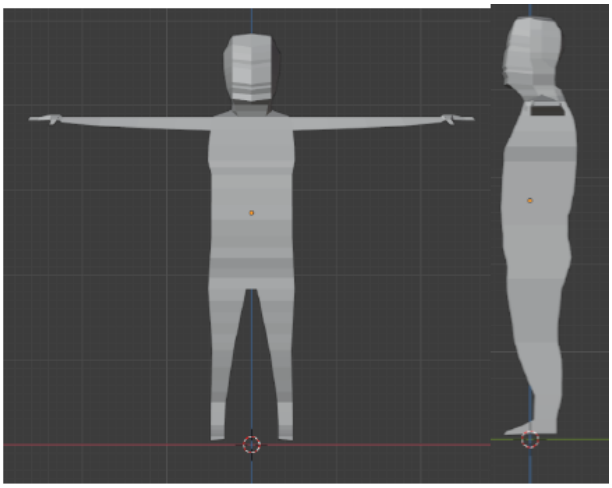


Ilustración 13. Personaje aplicado modo sólido. (Autor)

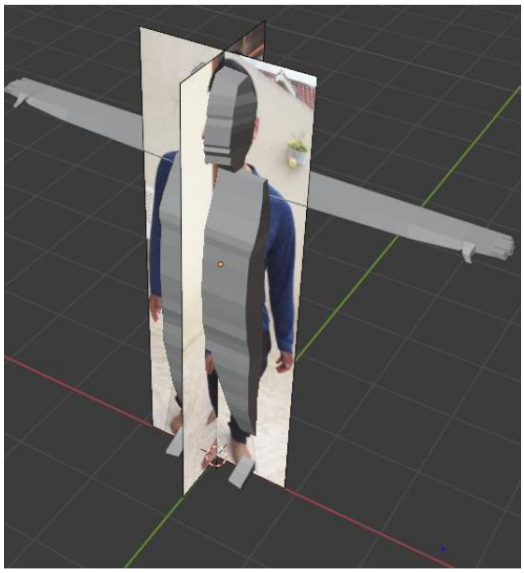


Ilustración 14. Vista en lateral y frontal aplicado modo sólido. (Autor)

Para darle un estilo más realista al personaje teniendo un cuerpo en 3D más adecuada, se aplica un sombreado suavizado subdividiendo varios vértices a todo el modelo.

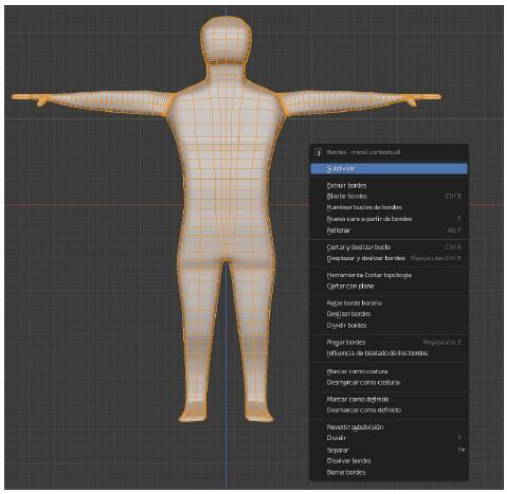


Ilustración 15. Suavizado del personaje. (Autor)

Para el diseño del cuerpo y rostro se aplica el modo escultura que trabaja un modelamiento real, donde se va definiendo por medio de un pincel modificando todos los vértices aplicados al personaje.

En este apartado se ha comenzado a dar vida a nuestro personaje, mediante modo escultura, damos forma al rostro, a los ojos, labios, orejas, nariz, cabeza, parte superior e inferior.

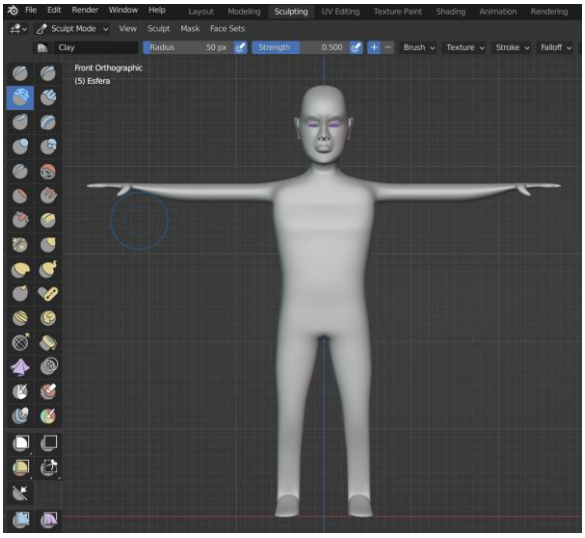


Ilustración 16. Esculpido del personaje. (Autor)

Para la vestimenta del personaje se ha creado la gorra, buso, pantalón, utilizando la propia geometría del modelo como base de las prendas quedándonos de la siguiente manera:

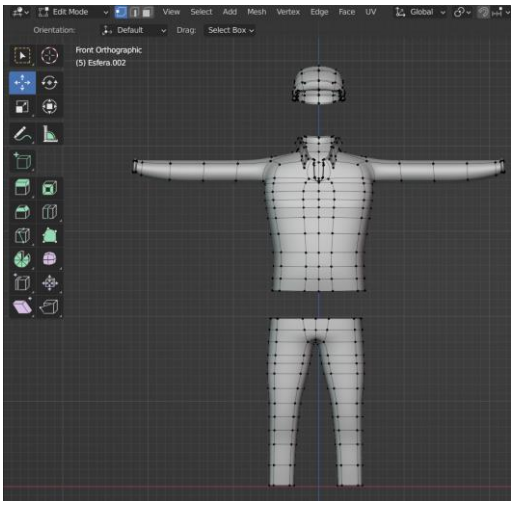


Ilustración 17. Accesorios del personaje. (Autor)

El diseño de las prendas realizadas le acoplamos a nuestro modelo, y le esculpimos dando pequeñas formas, quedándonos así un personaje real.

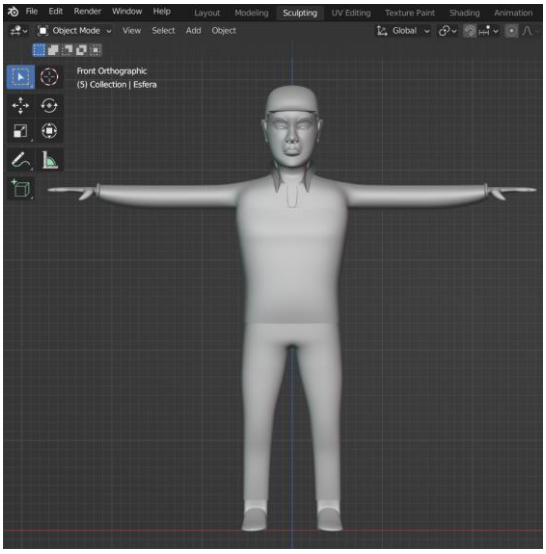


Ilustración 18. Implementación de accesorios al personaje. (Autor)

Para que el personaje no se quede con un aspecto gris le damos color al cuerpo y a las prendas de vestir mediante herramientas shading, editor el cual nos ha permitido crear materiales de color usando nodos, donde se crea TEXTURAS básicas y mapas de textura UV de cada material, cada vez que se le pinta las partes del modelo se va visualizando en la ventana del editor.

Ilustración 19. Texturas aplicadas en diferentes materiales del modelo. (Autor)

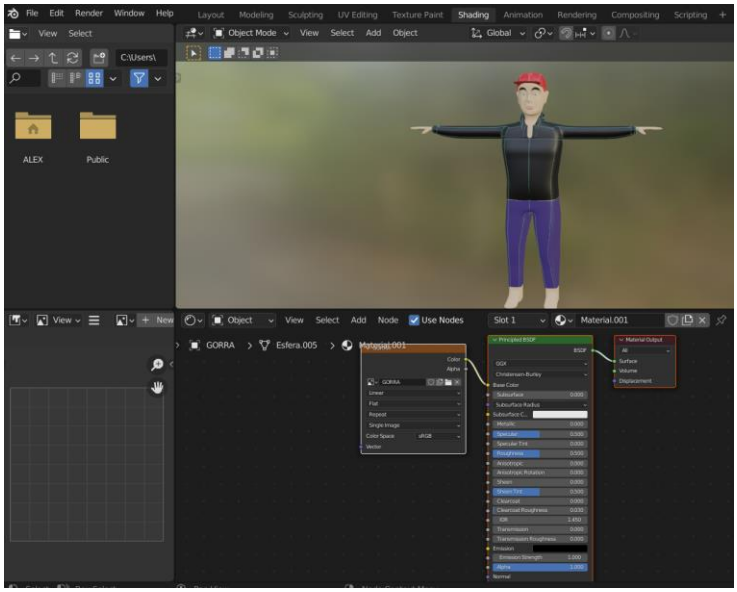


Ilustración 20. Interfaz de blender utilizando la herramienta shading, mediante nodos de texturas. (Autor)

En la siguiente Ilustración 21, se puede observar a nuestro avatar implementado junto con las texturas generadas mediante los nodos y mapas de textura UV, teniendo un personaje real.

Ilustración 21. Texturizado del personaje. (Autor)

.

Para que nuestro personaje sea animado se necesita de una estructura interna de elementos como es el caso de huesos, controles de malla para que pueda moverse.

En la opción agregar seleccionamos Armature y le damos un RIGGING, especie de huesos o esqueleto que permite el giro y movimiento de los ojos, cejas, labios, cabeza, brazos, piernas una vez aplicado estaremos dando vida a nuestro avatar teniendo un aspecto humanoide.

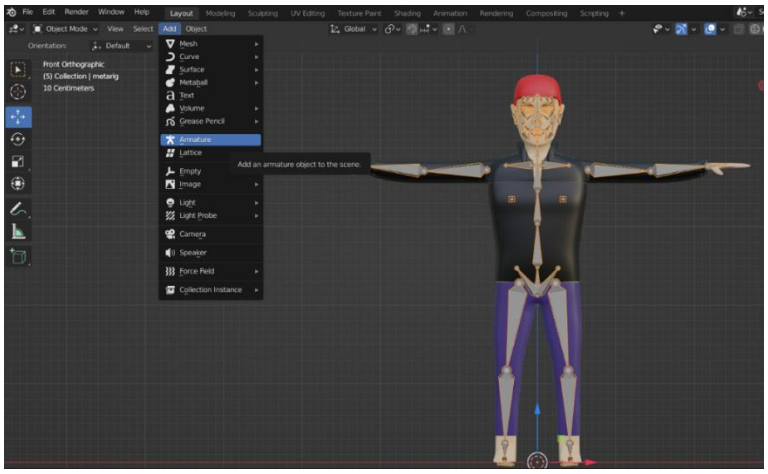


Ilustración 22. Implementación de la armadura al personaje (Huesos). (Autor)

2.1.1. MOVIMIENTO DEL PERSONAJE

Para que nuestro avatar funcione se ha realizado un rigify que ayuda a construir, crear controladores y huesos para el movimiento de nuestro personaje.

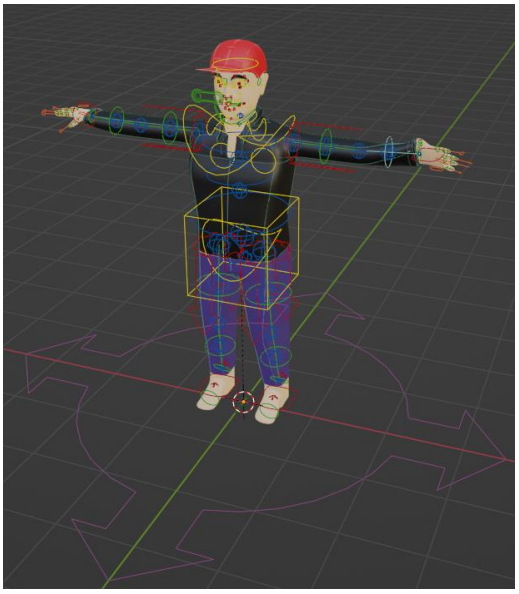


Ilustración 23. Método rig en el personaje. (Autor)

Dado el rig en el avatar se ha realizado 2movimientos:

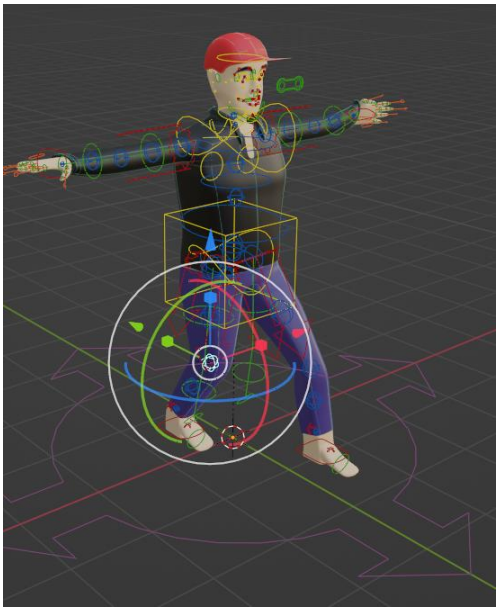


Ilustración 24. Movimientos corporales del personaje. (Autor)

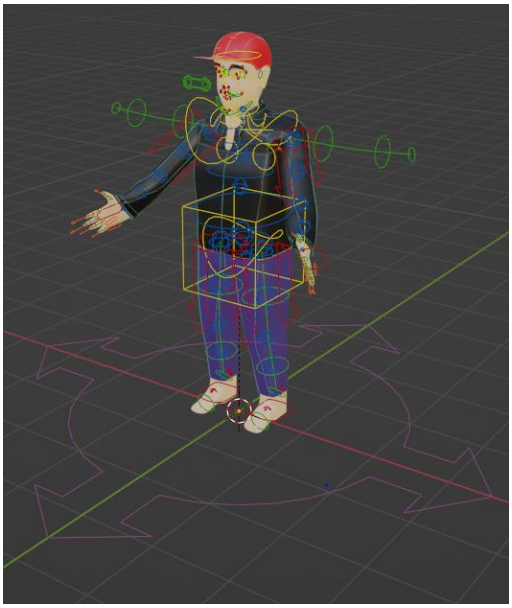


Ilustración 25. Movimientos de brazos. (Autor)

Ilustración 26. Personaje terminado. (Autor)

2.1.2. ANIMACIÓN UNITY

Unity es una herramienta de interacción que desarrolla artes, escenas de manera rápida, y consta de componentes que permite animar personajes.

- Ventana del proyecto
- Vistas de escenas
- Ventana de jerarquía
- Ventana de inspector
- Barra de herramientas

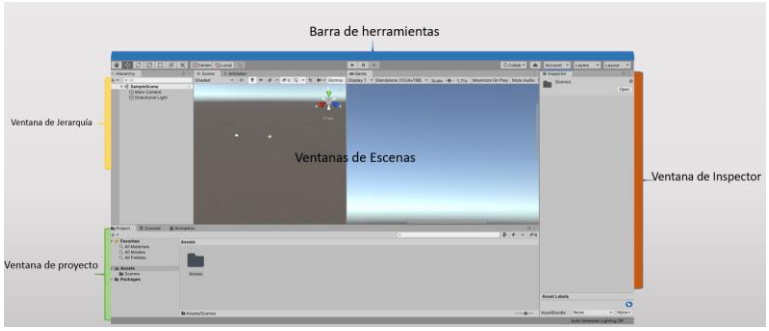


Ilustración 27. Interfaz de Unity. (Autor)

Luego de haber creado el avatar en la plataforma de BLENDER con los movimientos, se le exporta al personaje por medio de un archivo tipo FBX, formato el cual es utilizado principalmente para intercambiar animaciones de personajes, donde es soportado por la aplicación de Unity.

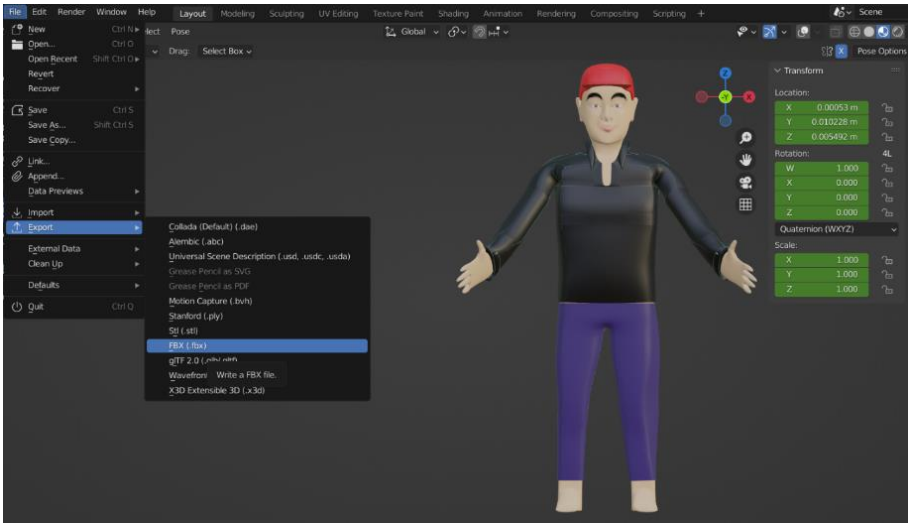


Ilustración 28. Exportación del archivo FBX de blender. (Autor)

En la siguiente Ilustración 29, se observa el avatar importado en el software UNITY.

Ilustración 29. Importación del archivo FBX a unity juntos con todos los componentes generados en blender. (Autor)

La exportación que realiza es de todos los componentes que lleva el personaje, las texturas, los movimientos, la armadura que son los huesos que se implementó en el avatar.

Para las animaciones tomamos en cuenta la herramienta inspectora que nos permite modificar los tiempos de animaciones que permite el avatar.

Para el caso de las animaciones de las vocales se tiene en la ilustración 30, los diferentes tiempos tanto de inicio como el final generado.

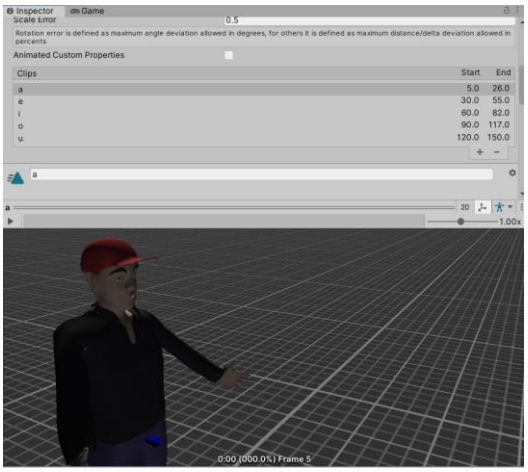


Ilustración 30. Gesticulación de las vocales del habla al personaje. (Autor)

2.2. ESTRUCTURA DEL SISTEMA DE DIÁLOGO HUMANO-MÁQUINA UTILIZANDO ARQUITECTURA BERT

Para el diseño de la estructura del diálogo humano-máquina, se ha utilizado una IA con arquitectura Bert en español para responder preguntas y respuestas, se usó un micrófono para capturar la voz del usuario para luego proceder a convertirla a texto, esta conversión se la realiza con las diferentes librerías existentes en Python, este texto entra al sistema de diálogo con IA y arquitectura BERT, el cual se encargará de reconocer y comprender el diálogo para generar una respuesta adecuada, echo esto se realiza la conversión de la respuesta de texto a audio, para estos procesos se usaron varias librerías de Python. Las librerías usadas para este fin fueron:

2.2.1. SPEECH RECOGNITION

El reconocimiento de voz necesita una entrada de audio, en vez de crear scripts para tener acceso al micrófono y procesarlo desde cero, la librería Speech_Recognition tiene acceso al micrófono y la procesa de una manera muy sencilla. Speech_Recognition tiene acceso a varias API populares por lo que es flexible. Una de estas API es la de Google que permite convertir esta entrada de voz a texto [21].

Para ello se ha usado la librería Speech_Recognition con la API de Google para reconocer, procesar y pasar a texto la entrada de audio que se transmite mediante un micrófono conectado a la PC.

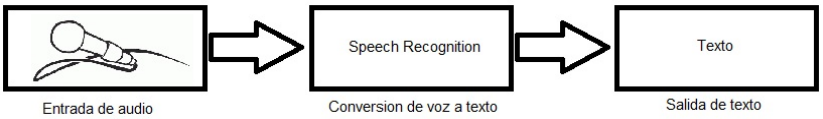


Ilustración 31. Uso de Speech Recognition para convertir de voz a texto. (Autor)

2.2.2. GTTS

gTTS es una librería de Python creada por Google, que nos ha servido para conectarnos con la API de conversión texto a voz de Google Translate, se puede escoger el idioma donde se realizará la conversión para así tener una correcta pronunciación [22]. Se ha usado esta librería para convertir la respuesta que nos da la IA de texto a audio.

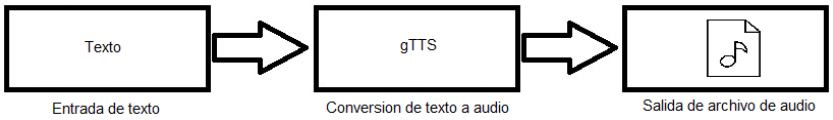


Ilustración 32. Uso de gTTS para convertir de texto a audio. (Autor)

2.2.3. PYWORLD

Pyworld es una librería que se ha usado para la manipulación y análisis de voz en alta calidad, puede generar voz con parámetros estimados [23]. Al momento de usar gTTS nos genera una voz femenina, para no tener esta voz, se ha usado esta librería para modificarla y hacerla más grave para usarla en nuestro Sistema.

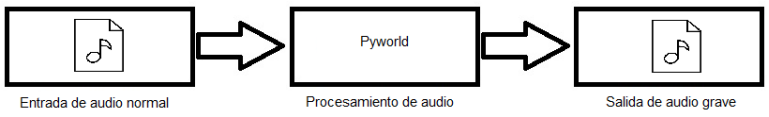


Ilustración 33. Uso de Pyworld para pasar el audio normal a audio grave. (Autor)

2.2.4. PYGAME

Pygame es una biblioteca de Python gratuita que sirve para desarrollo de videojuegos y funciona con bibliotecas de sonido, está diseñado para poder manipular diferentes formatos de video, audio, etc. [24]. Esta librería se ha usado para la reproducción de audio modificado que nos dio la librería gTTS.

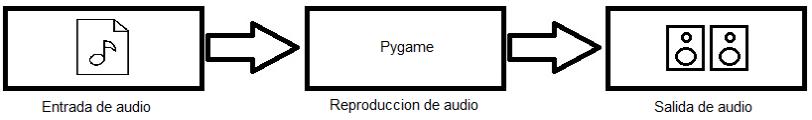


Ilustración 34. Uso de Pygame para reproducción de audio. (Autor)

La implementación del diálogo se muestra en el diagrama de la siguiente Ilustración:

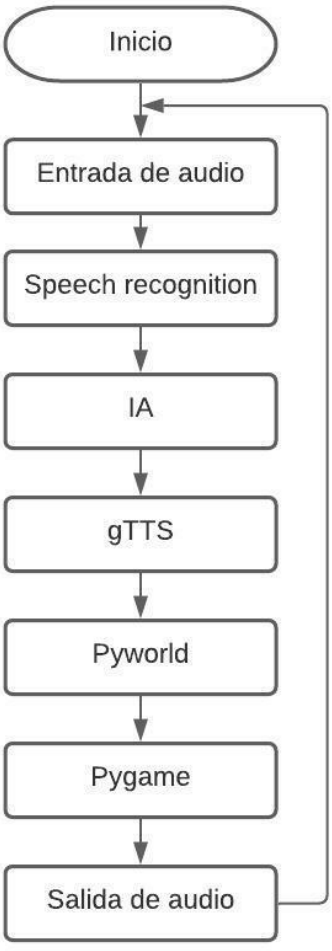


Ilustración 35. Diagrama del sistema de diálogo. (Autor)

2.3. INTEGRACIÓN DEL AVATAR CON EL SISTEMA DE DIÁLOGO

Para integrar el avatar y el sistema de diálogo se ha utilizado el siguiente sistema:

2.3.1. ZEROMQ

ZeroMQ es un sistema que nos ayuda con la comunicación entre dos o más aplicaciones diferentes. Funciona como mensajería entre múltiples aplicaciones, originalmente se crea como sistema de mensajería rápido, más tarde se cambió el enfoque en mejorar su usabilidad [25].

En nuestro proyecto de titulación se ha usado ZeroMQ por su fácil uso para enviar un mensaje de una aplicación a otra y así se puede realizar la comunicación entre Python y Unity, se envía un mensaje desde Python que será la respuesta, luego Unity lo recibe y lo descompone de letra en letra, se verifica cada letra si es una vocal, si es vocal realiza la animación correspondiente a esa vocal y si es consonante no realiza ninguna acción.

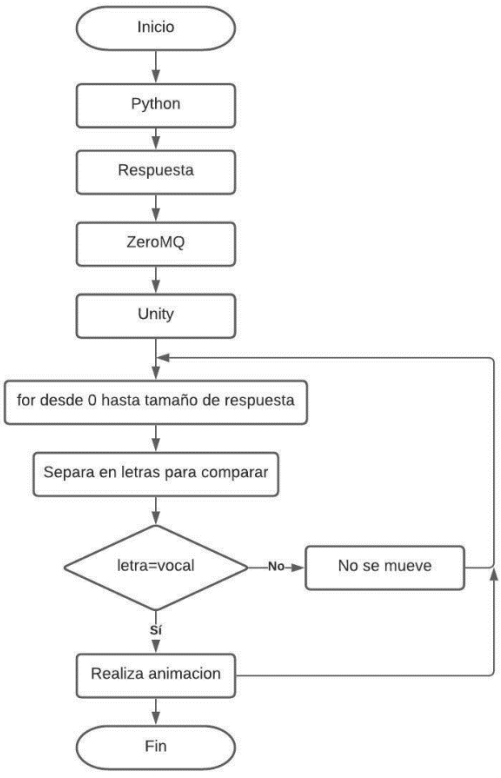


Ilustración 36. Diagrama de integración Python-Unity. (Autor)

CAPÍTULO 3

3. IMPLEMENTACIÓN Y ANÁLISIS DE RESULTADOS

Al finalizar todos los procesos y diseño del avatar explicados en el capítulo anterior, se dará a conocer el análisis y las diferentes pruebas para poder evaluar, verificar la eficiencia de respuesta del sistema de diálogo, con el fin de tener un correcto funcionamiento del sistema. Se los dividen en 2 categorías, estas son tiempo de respuesta y vocalización.

3.1. TIEMPO DE RESPUESTA

Se ha realizado varias pruebas ingresando un texto con diferentes números de palabras para medir el tiempo de respuesta del sistema de diálogo humano-máquina.

Se ha realizado 9 preguntas, las cuales fueron las mismas en todos los casos para medir el tiempo de respuesta, como resultado de aquello tenemos las siguientes tablas y gráficas.

Preguntas realizadas:

1. ¿Cómo se llama oficialmente Quito?
2. ¿Cuántos habitantes tiene Quito en el área urbana?
3. ¿Cuántos habitantes tiene Quito en el área metropolitana?
4. ¿Cuál es la capital de la provincia de Pichincha?
5. ¿A cuántos metros sobre el nivel del mar está ubicado Quito?
6. ¿En cuántas parroquias urbanas está dividido Quito?
7. ¿En cuántas parroquias rurales está dividido Quito?
8. ¿Cuál es la capital del Ecuador?
9. ¿Dónde tienen la mayoría de las empresas transnacionales su matriz?

NÚMERO DE PALABRAS: 163	
Número de pregunta	**Tiempo de respuesta (segundos)**
1	4,32
2	4,52
3	4,08
4	4,64
5	4,2
6	4,82
7	4,29
8	4,42
9	4,72
Promedio	**4,445555556**

Tabla 2. Tiempo de respuesta en 163 palabras. (Autor)

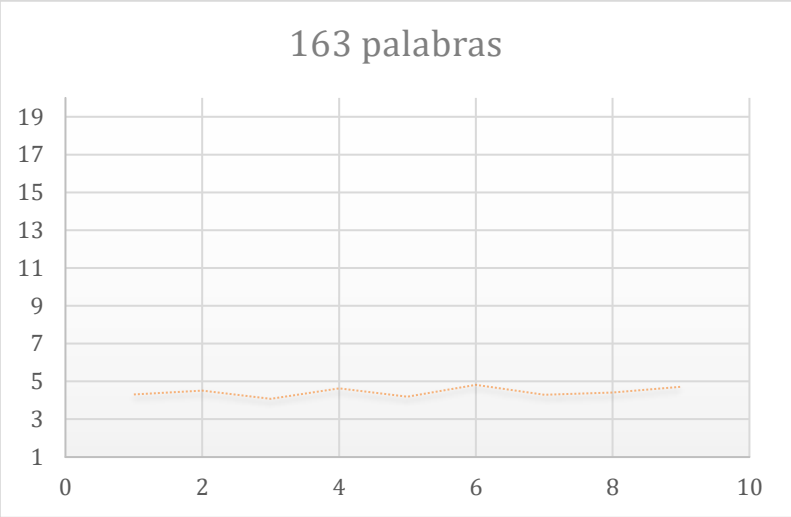


Ilustración 37. Gráfica de los tiempos en las diferentes preguntas realizadas con 163 palabras. (Autor)

Como se puede observar en la Tabla 2. Tiempo de respuesta en 163 palabras. los diferentes tiempos de respuesta son aproximados y da un promedio de 4,44 segundos, este tiempo es corto debido a que el número de palabras son pocas.

NÚMERO DE PALABRAS: 403	
Número de pregunta	**Tiempo de respuesta (segundos)**
1	7,9
2	7,43
3	7,56
4	7,32
5	7,83
6	7,79
7	7,83
8	6,93
9	7,44
Promedio	**7,558888889**

Tabla 3. Tiempo de respuesta en 403 palabras. (Autor)

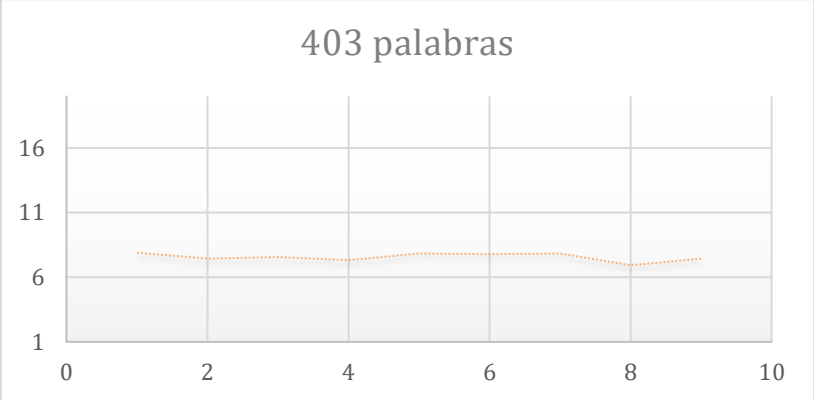


Ilustración 38. Gráfica de los tiempos en las diferentes preguntas realizadas con 403 palabras. (Autor)

Como se puede observar en la Tabla 3. Tiempo de respuesta en 403 palabras. los diferentes tiempos de respuesta son aproximados y da un promedio de 7,55 segundos, este tiempo va incrementando respecto a la Tabla 2. Tiempo de respuesta en 163 palabras. esto se debe a que el número de palabras aumento.

NUMERO DE PALABRAS: 548	
Numero de pregunta	**Tiempo de respuesta (segundos)**
1	9,32
2	9,56
3	9,35
4	9,22
5	9,36
6	9,42
7	9,54
8	9,19
9	9,51
Promedio	**9,385555556**

Tabla 4. Tiempo de respuesta en 548 palabras. (Autor)

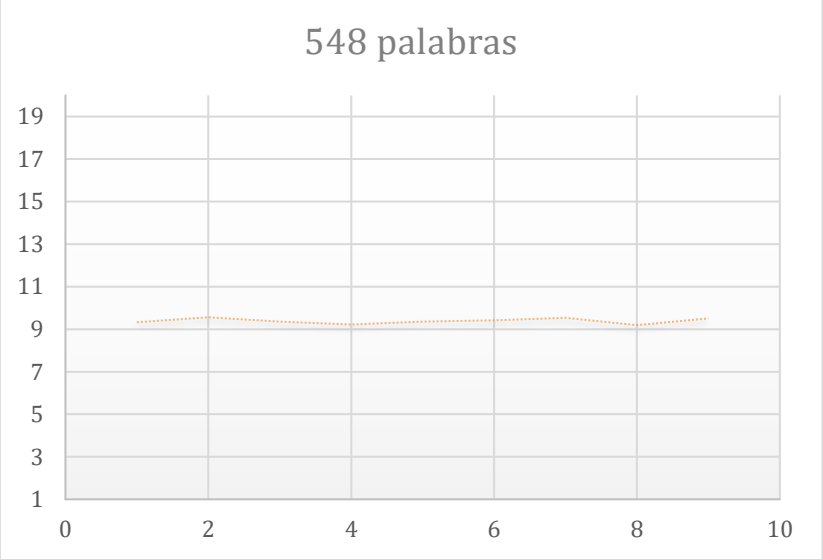


Ilustración 39. Gráfica de los tiempos en las diferentes preguntas realizadas con 548 palabras. (Autor)

Como se puede observar en la Tabla 4. Tiempo de respuesta en 548 palabras. los diferentes tiempos de respuesta son aproximados y da un promedio de 9,38 segundos, este tiempo va incrementando respecto a las tablas anteriores esto se debe a que el número de palabras va aumentando.

NÚMERO DE PALABRAS: 787	
Número de pregunta	**Tiempo de respuesta (segundos)**
1	11,41
2	12,11
3	11,87
4	11,65
5	11,54
6	11,53
7	11,32
8	12,06
9	11,97
Promedio	**11,71777778**

Tabla 5. Tiempo de respuesta en 787 palabras. (Autor)

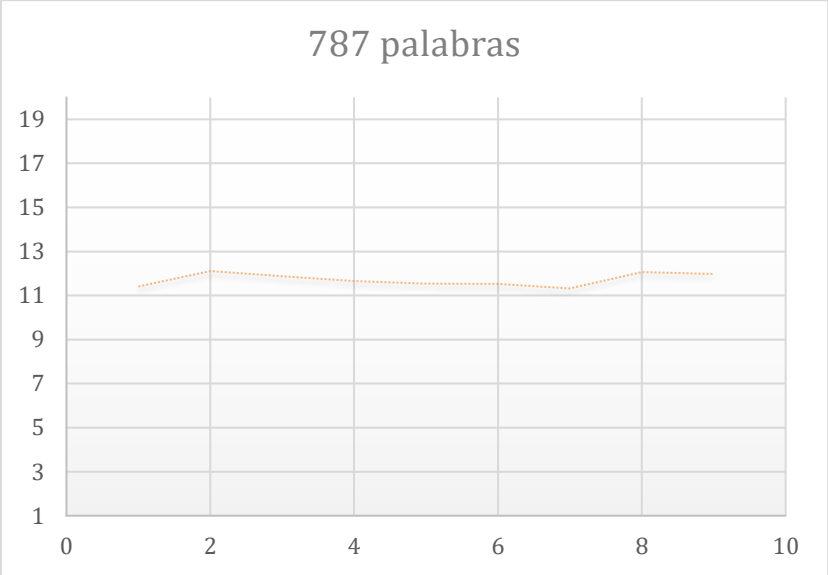


Ilustración 400. Gráfica de los tiempos en las diferentes preguntas realizadas con 787 palabras. (Autor)

Como se puede observar en la Tabla 5. Tiempo de respuesta en 787 palabras. los diferentes tiempos de respuesta son aproximados y da un promedio de 11,71 segundos, como se sigue observando el tiempo de respuesta se incrementa respecto a las tablas anteriores.

NÚMERO DE PALABRAS: 1075	
Número de pregunta	**Tiempo de respuesta (segundos)**
1	15,1
2	15,63
3	15,92
4	15,21
5	15,49
6	15,32
7	15,42
8	15,67
9	15,23
Promedio	**15,44333333**

Tabla 6. Tiempo de respuesta en 1075 palabras. (Autor)

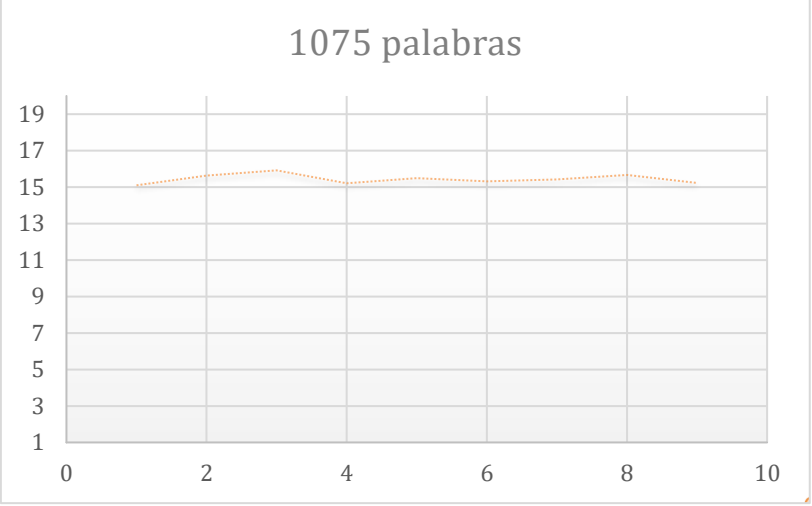


Ilustración 41. Gráfica de los tiempos en las diferentes preguntas realizadas con 1075 palabras. (Autor)

Como se puede observar en la Tabla 6. Tiempo de respuesta en 1075 palabras. los diferentes tiempos de respuesta son aproximados y nos da un promedio

de 15,43 segundos, como se sigue observando el tiempo de respuesta se incrementa respecto a las tablas anteriores.

Número de palabras	Tiempo de respuesta (s)
163	4,44
403	7,55
548	9,38
787	11,71
1075	15,44

Tabla 7. Comparación entre tiempo de respuesta y numero de palabras. (Autor)

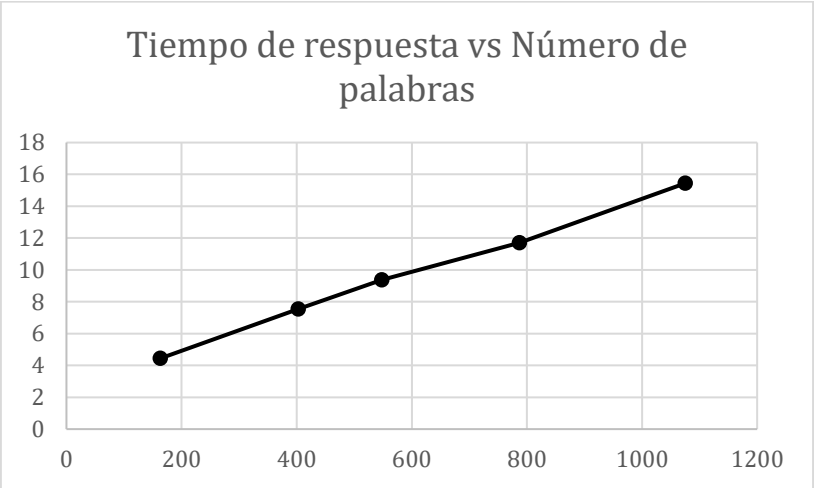


Ilustración 42. Gráfica de comparación. (Autor)

En la tabla 7 y en la ilustración 42, se puede observar que mientras más amplio sea el texto que se introduzca en el sistema humano-máquina, más tiempo se demorar el sistema de diálogo en generarnos la respuesta más adecuada, siendo así que con 163 palabras se demora 4,44 segundos en responder y con 1075 palabras se demora 15,43 segundos.

3.2. VOCALIZACIÓN

Se ha llevado a cabo una prueba que consiste en realizar varias preguntas con respuestas diferentes para medir el nivel de vocalización que tiene el avatar, esta vocalización la realiza pronunciando cada vocal que existe en la respuesta a pronunciar.

CAPÍTULO 3. IMPLEMENTACIÓN Y ANÁLISIS DE RESULTADOS

Se han realizado 8 preguntas que da como resultado respuestas diferentes para así poder medir el nivel de vocalización, como resultado de aquello se tiene la siguiente tabla.

Preguntas realizadas:

1. ¿A qué provincia pertenece Quito?
2. ¿Cuántos habitantes tiene Quito en el área urbana?
3. ¿Cuántos habitantes tiene Quito en el área metropolitana?
4. ¿A qué altitud promedio se encuentra Quito?
5. ¿En cuántas parroquias urbanas está dividido Quito?
6. ¿En cuántas parroquias rurales está dividido Quito?
7. ¿A qué país pertenece Quito?
8. ¿Sobre qué hoya está ubicada Quito?

Una vez realizada la pregunta el sistema de diálogo genera una respuesta, esta respuesta se descompone y solo se toma en cuenta las vocales que contiene la respuesta generada para así proceder a comparar con cada una de las vocalizaciones animadas haciendo lo más real posible.

Para verificar que el sistema esté vocalizando correctamente, se ha realizado una captura de video y se lo analizó en un tiempo de reproducción con velocidad baja.

Ilustración 43. Vocalización de la vocal a.

Ilustración 44. Vocalización de la vocal e.

 *Ilustración 45. Vocalización de la vocal i.*	 *Ilustración 46. Vocalización de la vocal o.*
 *Ilustración 47. Vocalización de la vocal u.*	

Tabla 8. Captura de vocalización de las vocales.

Por ejemplo, si la respuesta que nos genera es la palabra ECUADOR, el sistema descompone la palabra letra por letra y procede a comparar: primero ingresa la letra E y esta letra se comparará entre las opciones y como es vocal procede a realizar la animación correspondiente, luego ingresa la letra C y al ser consonante no se la vocaliza y así sucesivamente con cada letra.

Número	Respuesta	Vocalización	Porcentaje de vocalización correcta
1	Pichincha	i-i-a	100,00%
2	Dos millones	o-i-o-e	100,00%
3	Tres millones	e-i-o-e	100,00%
4	Dos mil ochocientos cincuenta	o-i-o-o-i-e-o-i-u-e-a	72,00%
5	Treinta y dos	e-i-a-o	100,00%
6	Treinta y tres	e-i-a-e	100,00%
7	Ecuador	e-u-a-o	100,00%
8	Guayllabamba	u-a-a-a-a	80,00%
		Promedio:	94,00%

Tabla 9. Vocalización de las respuestas obtenidas. (Autor)

Como se puede observar en la Tabla 9. Vocalización de las respuestas de las 8 preguntas vocalizó correctamente 6 respuestas mientras que 2 de las respuestas las vocalizó incompletas teniendo un porcentaje promedio de acierto de 94%, mientras más vocales tenga la respuesta baja el porcentaje de vocalización.

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

En la actualidad en el que vivimos existen una variedad de métodos de Interacción Humano-Maquina que han avanzado drásticamente con el tiempo, esto es debido gracias a las nuevas Tecnologías de la Información y Comunicación (TIC), que han permitido desarrollar nuevos sistemas para que la comunicación Humano-Máquina sea más fluida y eficiente, aproximándose a un mundo real, interactuando de forma social con avatares, que permite a los usuarios conectarse a un avatar ocasionando espacios complejos creando relaciones de información sobre política, historia, educación y culturales, es por ello que se investigan nuevas interacciones, algoritmos, Inteligencia Artificial, Interfaces que faciliten la comunicación para el ser humano.

En el presente trabajo de Tesis se ha propuesto un Sistema Prototipo de Interacción Humano-Máquina utilizando la arquitectura BERT, que permita Interactuar con el usuario mediante Voz, usando un avatar virtual animado. Diseñar un avatar virtual utilizando los softwares de Blender y Unity 3D, básicamente como modeladores 3D como principiantes en el manejo de estas herramientas se tuvo un proceso desafiante. Durante el desarrollo de la Tesis se aprendieron nuevas habilidades para superar los retos que surgieron. Este proyecto requirió de mucha planificación tales como realizar un plano de diseño, hacer una lista de modelados a crear, organizar los modelos dentro Blender y Unity.

Para el primer y segundo objetivo se tuvo que realizar un estudio y la forma en el que se maneja los softwares, que comandos se utilizan para el diseño y animaciones del avatar, y ver la compatibilidad entre BLENDER y UNITY, ya que son programas gratuitos con código abierto.

Para el segundo objetivo se diseñó el avatar mediante el software de BLENDER quien nos facilitó varias herramientas al momento de crear, de tal forma que se fue creando el personaje viéndose un poco realista como al de un humano. Blender es uno de los programas que proporciona huesos, como es el caso de nuestro avatar se crea estos huesos para los movimientos del avatar. Para que el avatar sea animado Blender contiene archivos tipo FBX, lo que permite que Unity sea accesible a Blender ya que este tipo de archivo permite que Unity importe Archivos de Blender.

CAPÍTULO 4. CONCLUSIONES Y RECOMENDACIONES

Para el tercer objetivo en el cual se diseñó una estructura del diálogo Humano-Máquina utilizando la arquitectura BERT, para realizarlo se utilizó el programa Python y varias librerías como se explicó en la sección del capítulo 2. En Python se procedió a utilizar una arquitectura BERT en español y se procedió a usar las diversas librerías para proceder a capturar la voz y generar respuesta auditiva. Con esto se obtuvo nuestro sistema de dialogo listo para integrarlo a nuestro avatar.

Finalmente, para adaptar el sistema de diálogo Humano-Máquina al agente virtual se utilizó la librería ZeroMQ tanto para Python como para Unity que nos sirve para realizar la comunicación entre estos dos programas, desde Python se envía la respuesta a Unity en forma de texto y mediante programación en C# se descompone en consonantes y vocales, las consonantes se las descartan y las vocales entran a las diferentes decisiones dentro de la programación para verificar de que vocal se trata y así el avatar empieza a vocalizar cada una de las vocales encontradas cumpliendo así nuestro cuarto objetivo.

Respecto a los resultados obtenidos se realizaron dos partes que fueron de tiempo de respuesta y vocalización. La primera parte se refiere al tiempo de respuesta que demora en responder el avatar esta medición de tiempo se la realizo desde que se realiza la pregunta hasta el momento de empezar a vocalizar el avatar. La segunda parte se refiere respecto a la vocalización que realiza el avatar como que tan efectiva es, esto se llevó a cabo realizando varias preguntas que tenían diferentes respuestas y se tomó en cuenta de cómo el avatar iba vocalizando, al principio se tuvo el inconveniente de que el avatar seguía vocalizando después de que el audio de respuesta deje de sonar y se lo corrigió enviando una señal que ya deje de realizar la vocalización lo que ocasiono que no vocalice todas las vocales cuando la respuesta es demasiada larga.

Se puede concluir que el proyecto es muy factible al momento de ejecutarse ya que funciona de manera adecuada sin tener mayores inconvenientes, además contiene una receta de ejecución accesible para cualquier usuario que necesite ejecutarlo. Así culmina este proyecto técnico de titulación, otorgando un proyecto funcional al Grupo de Investigación GIIRA para trabajos futuros.

4.2 RECOMENDACIONES

Se debe tener en cuenta que para los programas de Blender y Unity, los renderizados y procesado 3D consumen una infinidad de recursos, sobre todo CPU, RAM y gráficos.

Como estudiantes de la Universidad Politécnica Salesiana consideramos que la recomendación más importante es que el Grupo de Investigación brinde más apoyo a este tipo de proyecto ya es una de las tecnologías futuras hacia los estudiantes y sobre todo brindar una mayor atención a la sociedad.

Para que el sistema de diálogo Humano-Máquina funcione en su estado optimo se recomienda que el texto introducido para generar la respuesta no sea muy extenso para así evitar que se demore en responder, esto se lo puede realizar optimizando en lo posible el texto y que incluya respuestas cortas así el sistema se demorara lo menos posible en responder, así mismo se recomienda tener una computadora de gama alta que permita ejecutar el proyecto de manera adecuada.

Para la ejecución del sistema se recomienda que se agreguen cada una de las librerías necesarias para que funcione correctamente y evitar que ocasione errores, así mismo seguir la receta de ejecución del apéndice A paso a paso para poder así replicar el proyecto en la computadora que se desee, en esta receta de ejecución consta todas las librerías y pasos necesarios para la ejecución del proyecto.

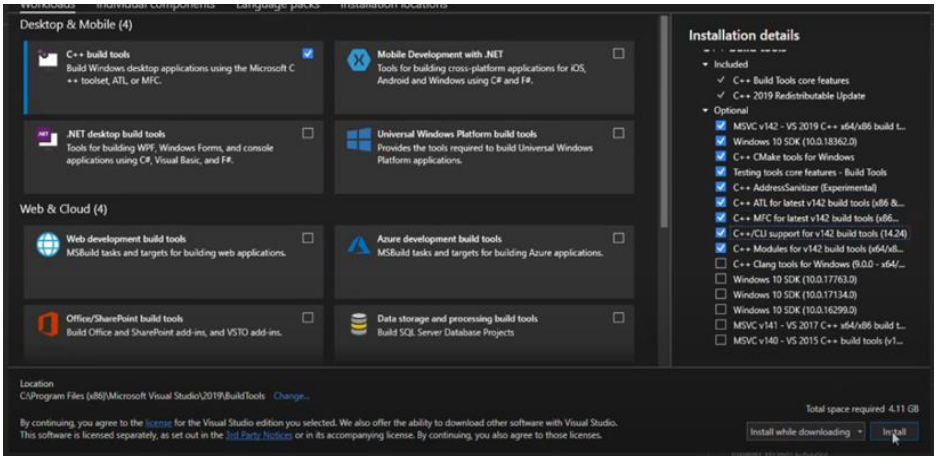
APÉNDICES

APÉNDICE A

RECETA DE EJECUCIÓN

1. Descargar Anaconda de https://www.anaconda.com/products/individual
2. Descargar Unity 2019.3.13f1
3. Instalar el archivo que se encuentra dentro de la carpeta llamado vs_BuildTools, de doble clic al archivo.

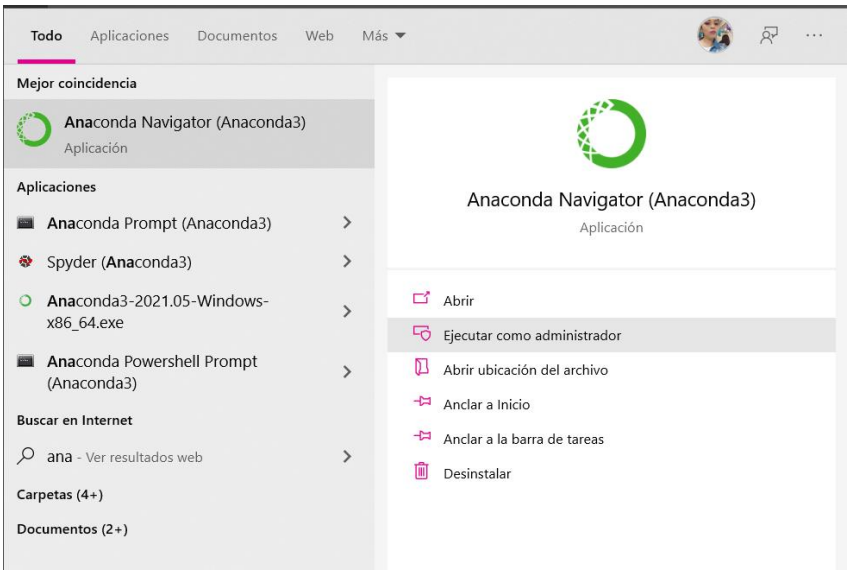
Luego le da clic a continuar, una vez realizado eso saldrá la siguiente ventana donde seleccionaremos C++ build tools y en la parte derecha en installation details seleccionaremos todas estas opciones y le daremos a instalar.



Esperamos termine la instalación y procedemos a realizar el siguiente paso.

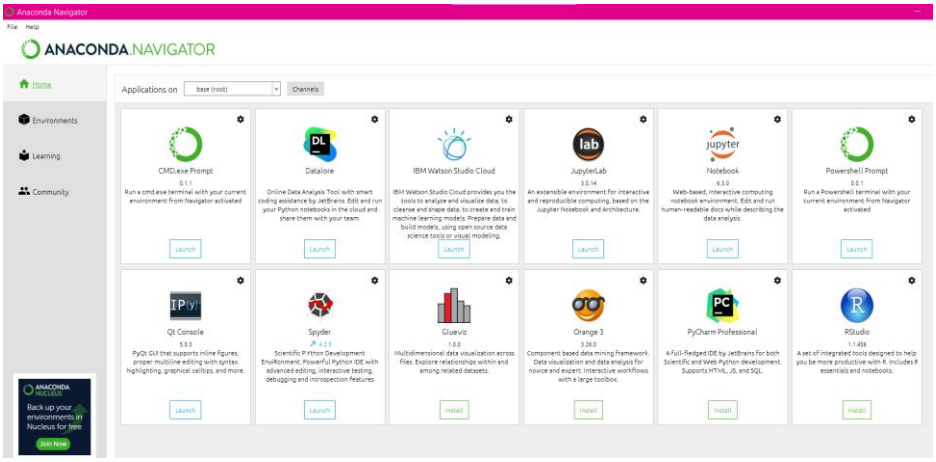
4. Dentro del proyecto descargado existe un archivo llamado berto.yml este contiene la configuración de anaconda para funcionar correctamente, para cargar esta configuración haga lo siguiente:

-Abra anaconda ejecutando como administrador

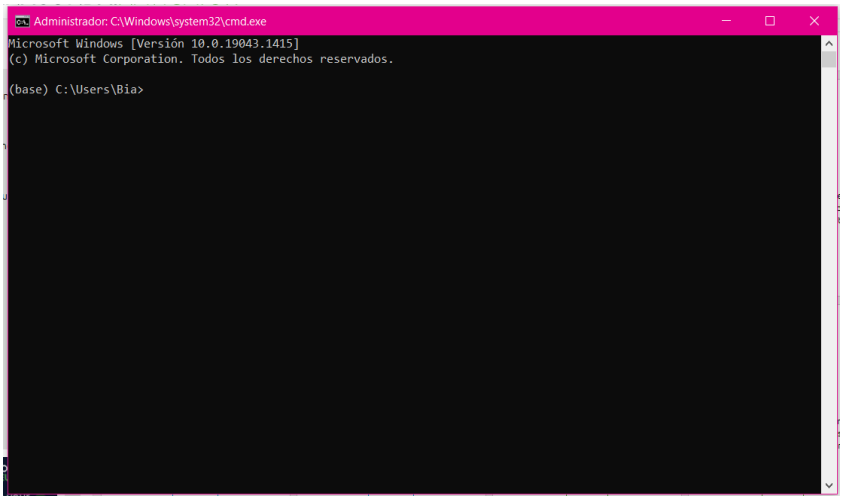


-Una vez dentro de anaconda ejecute el cmd

APÉNDICES



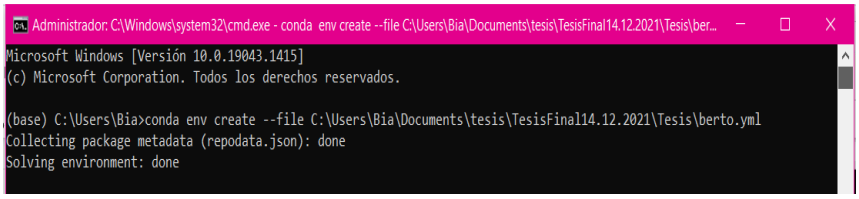
-Cuando este abierto el cmd ponga la siguiente línea de código



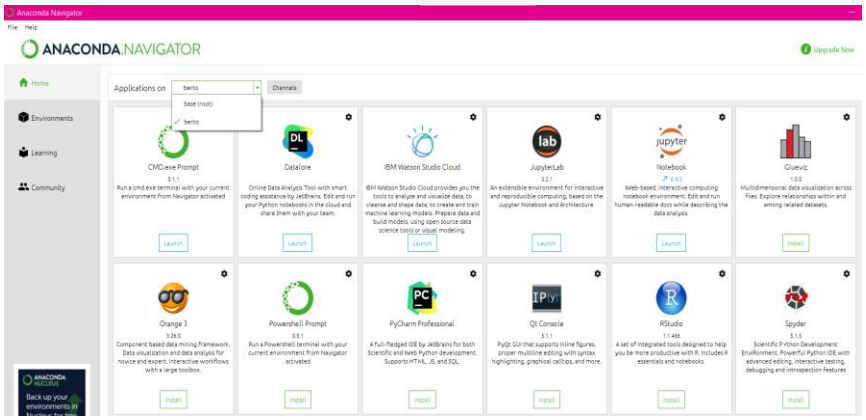
conda env create --file C:\Users\Bia\Documents\tesis\TesisFinal14.12.2021\Tesis\berto.yml

La dirección debe de ser la que está actualmente el archivo berto.yml, una vez que ejecute se cargara un nuevo entorno a anaconda espere a que termine el proceso.

APÉNDICES



-una vez finalizada la instalación inicie el entorno nuevo creado e inicie el cmd para instalar las librerías que se usaran.



Dentro del cmd ejecute las siguientes líneas para instalar las diferentes librerías a usar.

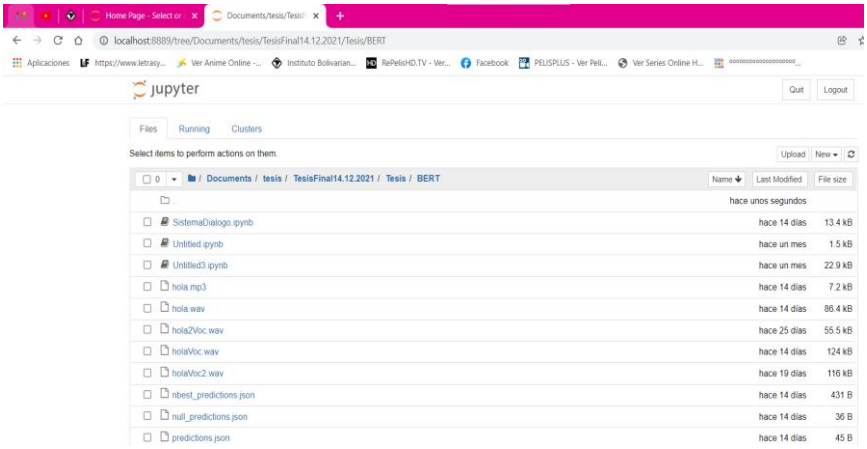
- pip install soundfile
- pip install pygame
- pip install pyworld
- pip install gtts
- pip install pydub
- pip install matplotlib
- pip install speechrecognition
- pip install pyaudio
- pip install ipywidgets
- jupyter nbextension enable --py widgetsnbextension
- pip install transformers==3.3.1
- pip3 install torch torchvision torchaudio

Luego reinicie el pc para que se reinicien algunos de los procesos necesarios para que se ejecuten los cambios.

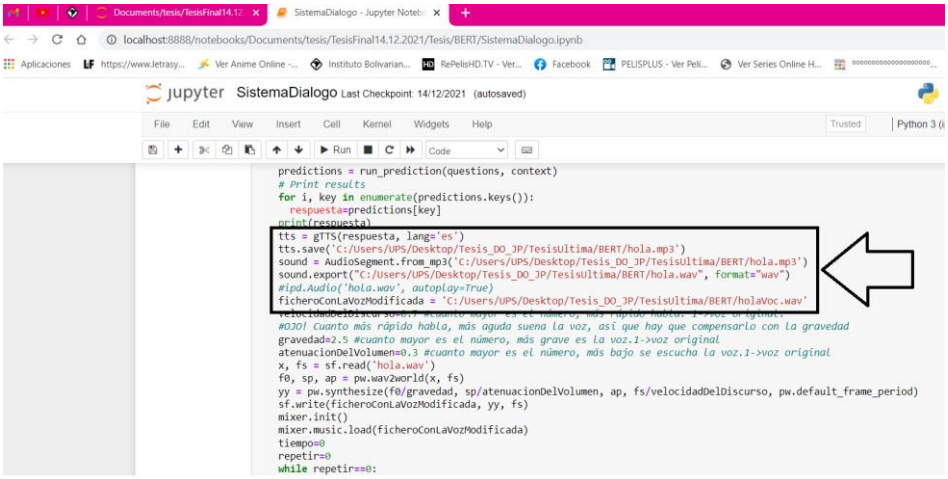
APÉNDICES

5. Abra anaconda, luego inicie el entorno bert y a continuación inicie jupyter notebook.

6. Una vez iniciado vaya a la dirección del proyecto descargado luego abra la carpeta bert y de clic sobre el archivo SistemaDialogo.ipynb

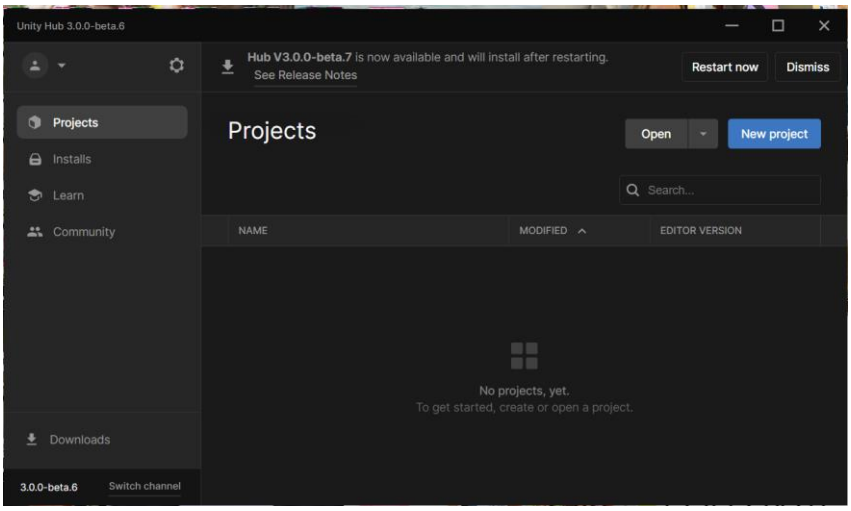


7. En las siguientes líneas cambie la dirección con la ubicación del proyecto descargado con los respectivos archivos necesarios como lo es en la primera dirección hola.mp3, en la segunda la misma dirección del hola.mp3 en la tercera la dirección de hola.wav y por último la dirección de holaVoc.wav.

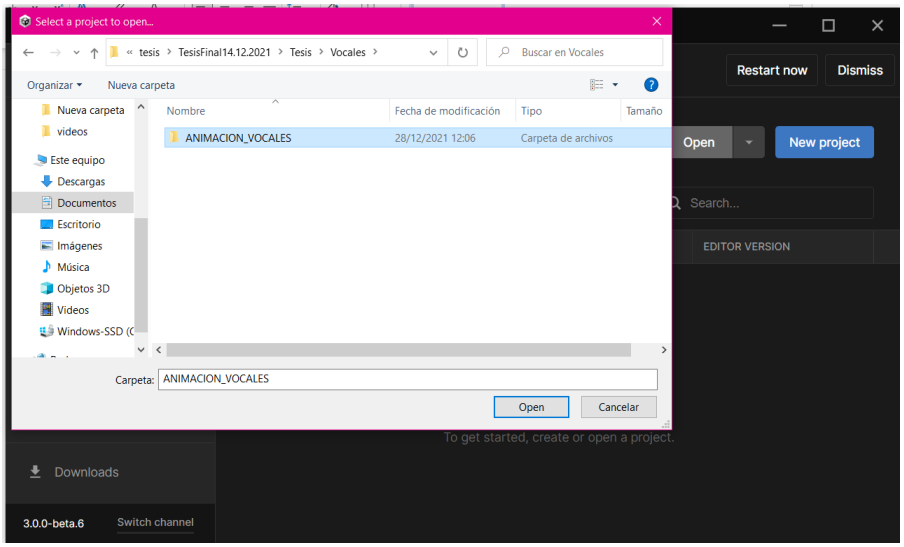


8. Dele a ejecutar en este punto estará funcionando la IA de diálogo con arquitectura BERT y a su vez enviará la respuesta que va a ser recibida por unity.

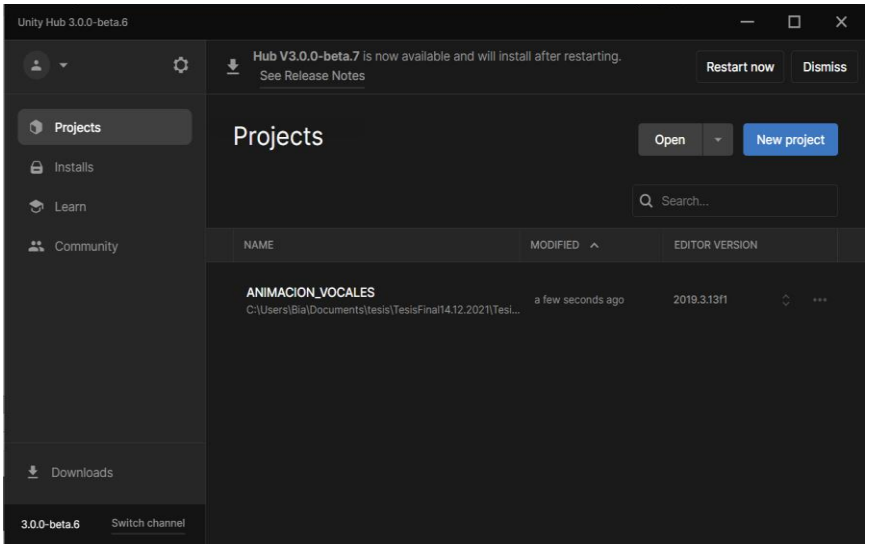
9. Abra unity hub para agregar el proyecto de unity.



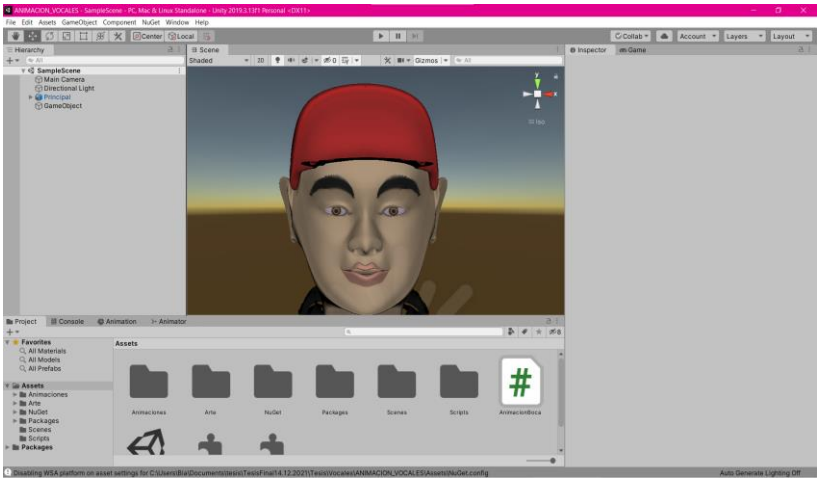
10. Haga clic en open y diríjase a la dirección donde se descargó el proyecto, entre a vocales y cargue la carpeta animación vocales.



11. De clic al proyecto que se cargó.



12. A continuación, se abre el avatar al momento de darle clic en play recibirá la respuesta generada en Python y realizará los respectivos movimientos faciales. Hay que asegurarse que se esté ejecutando Python y enviando respuesta ya que si no es así el avatar se quedará esperando y tocará reiniciar UNITY.



APÉNDICE B

CÓDIGO DE PYTHON

```
#Importar librerías
import soundfile as sf
import pygame
import pyworld as pw
from gtts import gTTS
from tempfile import TemporaryFile
from pygame import mixer
from pydub import AudioSegment
from os import path
import os
import matplotlib.pyplot as plt
import IPython.display as ipd
import wave
import speech_recognition as sr
import contextlib
import zmq
import time
context = zmq.Context()
socket = context.socket(zmq.REP)
socket.bind('tcp://*:5556')
import os
import torch
```

```
from torch.utils.data import DataLoader, RandomSampler, SequentialSampler
from transformers import (
    AutoTokenizer,
    AutoModelForQuestionAnswering,
    squad_convert_examples_to_features
)
from transformers.data.processors.squad import SquadResult, SquadV2Processor, SquadExample
from transformers.data.metrics.squad_metrics import compute_predictions_logits
def run_prediction(question_texts, context_text):
    """Setup function to compute predictions"""
    examples = []
    for i, question_text in enumerate(question_texts):
        example = SquadExample(
            qas_id=str(i),
            question_text=question_text,
            context_text=context_text,
            answer_text=None,
            start_position_character=None,
            title="Predict",
            is_impossible=False,
            answers=None,
        )
        examples.append(example)
    features, dataset = squad_convert_examples_to_features(
        examples=examples,
        tokenizer=tokenizer,
        max_seq_length=384,
        doc_stride=128,
```

```
        max_query_length=64,
        is_training=False,
        return_dataset="pt",
        threads=1,
    )
    eval_sampler = SequentialSampler(dataset)
    eval_dataloader = DataLoader(dataset, sampler=eval_sampler, batch_size=10)
    all_results = []
    for batch in eval_dataloader:
        model.eval()
        batch = tuple(t.to(device) for t in batch)
        with torch.no_grad():
            inputs = {
                "input_ids": batch[0],
                "attention_mask": batch[1],
                "token_type_ids": batch[2],
            }
            example_indices = batch[3]
            outputs = model(**inputs)
            for i, example_index in enumerate(example_indices):
                eval_feature = features[example_index.item()]
                unique_id = int(eval_feature.unique_id)
                output = [to_list(output[i]) for output in outputs]
                start_logits, end_logits = output
                result = SquadResult(unique_id, start_logits, end_logits)
                all_results.append(result)
    output_prediction_file = "predictions.json"
    output_nbest_file = "nbest_predictions.json"
    output_null_log_odds_file = "null_predictions.json"
```

```
    predictions = compute_predictions_logits(
        examples,
        features,
        all_results,
        n_best_size,
        max_answer_length,
        do_lower_case,
        output_prediction_file,
        output_nbest_file,
        output_null_log_odds_file,
        False,  # verbose_logging
        True,  # version_2_with_negative
        null_score_diff_threshold,
        tokenizer,
    )
    return predictions
model_name_or_path = "mrm8488/distill-bert-base-spanish-wwm-cased-finetuned-spa-squad2-es"
output_dir = ""
# Config
n_best_size = 1
max_answer_length = 30
do_lower_case = True
null_score_diff_threshold = 0.0
def to_list(tensor):
    return tensor.detach().cpu().tolist()
# Setup model
model_class, tokenizer_class = (AutoModelForQuestionAnswering, AutoTokenizer)
```

```
tokenizer = tokenizer_class.from_pretrained(
   model_name_or_path, do_lower_case=True)
model = model_class.from_pretrained(model_name_or_path)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)
processor = SquadV2Processor()
context = "Quito, oficialmente San Francisco de Quito, es la capital de la República del Ecuador, de la Provincia de Píchínchá y la capital más antigua de Sudamérica. Es la ciudad más poblada del Ecuador, con dos millones de habitantes en el área urbana, y aproximadamente tres millones en todo el Área metropolitana."
Context
while True:
   bandera=0
   while bandera==0:
      r=sr.Recognizer()
      #desde el microfono
      with sr.Microphone() as recurso:
         audio=r.listen(recurso)
         try:
            texto=r.recognize_google(audio,language="es-ES")
            bandera=1
         except:
            print('Lo siento no te entendi intentalo de nuevo')
   texto='¿'+texto+'?'
   questions = [texto]
   print(texto)

   # Run method
   predictions = run_prediction(questions, context)
   # Print results
```

```
    for i, key in enumerate(predictions.keys()):
      respuesta=predictions[key]
    print(respuesta)
    tts = gTTS(respuesta, lang='es')

tts.save('C:/Users/ALEX/Documents/TESIS_FINALIZADO/Tesis/BERT/hola.mp
3')
    sound = AudioSegment.from_mp3('C:/Users/ALEX/Documents/TESIS_FINALIZADO/Tes
is/BERT/hola.mp3')
sound.export("C:/Users/ALEX/Documents/TESIS_FINALIZADO/Tesis/BERT/hol
a.wav", format="wav")
    #ipd.Audio('hola.wav', autoplay=True)
    ficheroConLaVozModificada = 'C:/Users/ALEX/Documents/TESIS_FINALIZADO/Tesis/BERT/holaVoc2.wav'
    velocidadDelDiscurso=0.7 #cuanto mayor es el número, más rápido habla. 1->voz original.
    #OJO! Cuanto más rápido habla, más aguda suena la voz, así que hay que compensarlo con la gravedad
    gravedad=2.5 #cuanto mayor es el número, más grave es la voz.1->voz original
    atenuacionDelVolumen=0.3 #cuanto mayor es el número, más bajo se escucha la voz.1->voz original
    x, fs = sf.read('hola.wav')
    f0, sp, ap = pw.wav2world(x, fs)
    yy = pw.synthesize(f0/gravedad, sp/atenuacionDelVolumen, ap, fs/velocidadDelDiscurso, pw.default_frame_period)
    sf.write(ficheroConLaVozModificada, yy, fs)
    mixer.init()
    mixer.music.load(ficheroConLaVozModificada)
    tiempo=0
    repetir=0
    while repetir==0:
```

```
if tiempo==0:
    data = {
        'str': respuesta
    }
if tiempo==1:
    repetir=repetir+1;
    data = {
        'str': 'Nnn'
    }
socket.recv()
socket.send_json(data)
if tiempo==0:
    time.sleep(0.7)
    mixer.music.play()
    while mixer.music.get_busy():
        pygame.time.Clock().tick(100)
    mixer.quit()
tiempo=tiempo+1;
```

APÉNDICE C

CÓDIGO SCRIPT C#

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class AnimacionBoca : MonoBehaviour
{
    public Animator anim;
    private float timer = 0;
    private float timerMax = 0;
    private float timer2 = 0;
    private float timerMax2 = 0;
    private float timer3 = 0;
    private float timerMax3 = 0;
    private float timer4 = 0;
    private float timerMax4 = 0;
    private float timer5 = 0;
    private float timerMax5 = 0;
    public Client client;
    public string indicador2;
    public string comparador="dd";
    private char[] comparacion;
    private int i;
    private int bandera=0;
    private int bandera2 = 0;
    private char letraActual;
    public string indicador3="Mu";
    // Start is called before the first frame update
    void Start()
    {
        client = FindObjectOfType<Client>();
    }
    // Update is called once per frame
    void Update()
    {
        if (client.indicador.Length > 1)
        {
            if (bandera==1)
            {
```

```
            indicador3 = client.indicador;
            timer = 0;
            timer2 = 0;
            timer3 = 0;
            timer4 = 0;
            timer5 = 0;
            print("Actualiza");
//            return;
            bandera = 0;
            bandera2 = 0;
            if (indicador3==comparador)
            {
               indicador3 = "N";
               bandera2 = 1;
            }
         }
      }
      if (bandera2==0)
      {
         comparador = indicador3;
      }
      indicador2 = indicador3.ToLower();
      comparacion = indicador2.ToCharArray();
      for (i = 0; i < indicador2.Length; i++)
      {
         letraActual = comparacion[i];
         if (letraActual == 'a' || letraActual == 'á')
         {
            anim.SetBool("a", true);
            if (!Waited(0.4f)) return; //0.8
            anim.SetBool("a", false);
         }
         else
         {
            if (letraActual == 'e' || letraActual == 'é')
            {
               anim.SetBool("e", true);
               if (!Waited2(0.5f)) return;//1
               anim.SetBool("e", false);
            }
            else
            {
```

```
            if (letraActual == 'i' || letraActual == 'í')
            {
               anim.SetBool("i", true);
               if (!Waited3(0.45f)) return;//0.9
               anim.SetBool("i", false);
            }
            else
            {
               if (letraActual == 'o' || letraActual == 'ó')
               {
                  anim.SetBool("o", true);
                  if (!Waited4(0.55f)) return;//1.1
                  anim.SetBool("o", false);
               }
               else
               {
                  if (letraActual == 'u' || letraActual == 'ú')
                  {
                     anim.SetBool("u", true);
                     if (!Waited5(0.6f)) return;//1.2
                     anim.SetBool("u", false);
                  }
                  else
                  {
                     anim.SetBool("a", false);
                     anim.SetBool("e", false);
                     anim.SetBool("i", false);
                     anim.SetBool("o", false);
                     anim.SetBool("u", false);
                  }
               }
            }
         }
      }
      if (indicador3.Length > 1)
      {
         indicador3 = indicador3.Remove(0, 1);
         timer = 0;
         timer2 = 0;
         timer3 = 0;
         timer4 = 0;
         timer5 = 0;
```

```
            bandera2 = 1;
        }
        if (indicador2.Length==1)
        {
            bandera = 1;
        }
         if (client.indicador=="Nnn")
        {
            indicador3="Nnn";
        }
    }
}
private bool Waited(float seconds)
{
    timerMax = seconds;
    timer += Time.deltaTime;
    if (timer >= timerMax)
    {
        return true; //max reached - waited x - seconds
    }
    return false;
}
private bool Waited2(float seconds)
{
    timerMax2 = seconds;
    timer2 += Time.deltaTime;
    if (timer2 >= timerMax2)
    {
        return true; //max reached - waited x - seconds
    }
    return false;
}
private bool Waited3(float seconds)
{
    timerMax3 = seconds;
    timer3 += Time.deltaTime;
    if (timer3 >= timerMax3)
    {
        return true; //max reached - waited x - seconds
    }
    return false;
}
```

```
    private bool Waited4(float seconds)
    {
        timerMax4 = seconds;
        timer4 += Time.deltaTime;
        if (timer4 >= timerMax4)
        {
            return true; //max reached - waited x - seconds
        }
        return false;
    }
    private bool Waited5(float seconds)
    {
        timerMax5 = seconds;
        timer5 += Time.deltaTime;
        if (timer5 >= timerMax5)
        {
            return true; //max reached - waited x - seconds
        }
        return false;
    }
}
```

REFERENCIAS BIBLIOGRÁFICAS

[1] M. I. Vallejos Sandoval, «Prototipo de interfaz humano-maquina basado en AIML con capacidad de realizar tareas preprogramadas», PhD Thesis, Universidad Andrés Bello, 2018.

[2] E. O. MONTOYA y J. H. M. DE LA CRUZ, «INTERFAZ HUMANO MÁQUINA AUDIOVISUAL».

[3] M. Morales-Rodríguez y J. Domínguez-Martínez, «Agentes conversacionales como un sistema de diálogo», *Memorias del V Encuentro de Investigadores del ITCM*, 2011.

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

[5] Y. Ocaña-Fernández, L. A. Valenzuela-Fernández, y L. L. Garro-Aburto, «Inteligencia artificial y sus implicaciones en la educación superior», *Propósitos y Representaciones*, vol. 7, n.º 2, ene. 2019, doi: 10.20511/pyr2019.v7n2.274.

[6] M. F. Garrido, «Formación basada en las Tecnologías de la Información y Comunicación: Análisis didáctico del proceso de enseñanza-aprendizaje», 2003.

[7] R. de Mántaras, «El futuro de la IA: hacia inteligencias artificiales realmente inteligentes», *OpenMind BBVA.?` Hacia una nueva ilustración*, 2018.

[8] J. A. LECHÓN MORALES, «DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN», UNIVERSIDAD DE LAS FUERZAS ARMADAS, 2019.

[9] L. E. Ruano, E. L. Congote, y A. E. Torres, «Comunicación e interacción por el uso de dispositivos tecnológicos y redes sociales virtuales en estudiantes universitarios/Communication and interaction by the use of technological devices and virtual social media in university students», *Revista ibérica de sistemas e tecnologias de informação*, n.º 19, p. 15, 2016.

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

[11] G. J. D\'\ias, «Tutorial Unity: El paseo del astronauta». Madrid: http://gaia. fdi. ucm. es/files/people/guille/tallerUnity2015~…, 2015.

[12] O. H. Sanabria Peña y others, «Análisis de relaciones del movimiento Maker con la educación en tecnolog{\'\i}a. Una mirada al semillero "Robótica e Impresión 3D" de la ETITC.».

[13] A. Locker, «Los mejores programas de diseño 3D de 2021», 2021.

[14] C. P. Córcoles Briongos, «Manual de introducción a Blender», 2012.

[15] R. González Duque, «Python para todos». N/A, 2014.

REFERENCIAS BIBLIOGRÁFICAS

[16] J. Devlin, M.-W. Chang, K. Lee, y K. Toutanova, «Bert: Pre-training of deep bidirectional transformers for language understanding», *arXiv preprint arXiv:1810.04805*, 2018.

[17] J. N. P. Morán, «Integración de un sistema de diálogo con un sistema inteligente de tutoría dirigido al entrenamiento procedimental», PhD Thesis, Universidad Politécnica de Madrid, 2021.

[18] J. H. M. DE LA CRUZ, «SISTEMA DE INTERACCIÓN HUMANO MÁQUINA AUDIOVISUAL», PhD Thesis, UNIVERSIDAD DEL VALLE, 2018.

[19] A. C. R. Tadeo, «Sistema Inteligente Conversacional para la Orientación Vocacional», 2009.

[20] E. I. Abade Gil y others, «Creación de avatares 3D para gaming y realidad virtual», 2017.

[21] D. Amos, H. S. R. Works, I. SpeechRecognition, y I. PyAudio, «The Ultimate Guide To Speech Recognition With Python», *Real Python*, 2018.

[22] S. Subhash, P. N. Srivatsa, S. Siddesh, A. Ullas, y B. Santhosh, «Artificial Intelligence-based Voice Assistant», en *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, 2020, pp. 593-596.

[23] S. Liu, Y. Cao, D. Wang, X. Wu, X. Liu, y H. Meng, «Any-to-Many Voice Conversion With Location-Relative Sequence-to-Sequence Modeling», *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1717-1728, 2021.

[24] M. M. K. M. L. Thakur, M. M. A. M. S. Lopes, y N. Parmar, «Music Player–Using Python».

[25] M. Sústrik y others, «ZeroMQ», *Introduction Amy Brown and Greg Wilson*, 2015.