UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE CUENCA

CARRERA DE INGENIERÍA ELECTRÓNICA

Trabajo de titulación previo
a la obtención del título de
Ingeniera Electrónica

PROYECTO TÉCNICO CON ENFOQUE INVESTIGATIVO:

"DISEÑO DE UN SISTEMA PROTOTIPO QUE PERMITA TRADUCIR COMANDOS DE TEXTO A LENGUAJE DE SEÑAS ECUATORIANO UTILIZANDO UN AVATAR VIRTUAL"

AUTORA:

KAREN ALEJANDRA ARIAS SALCEDO

TUTOR:

DR. CHRISTIAN RAÚL SALAMEA PALACIOS

CUENCA - ECUADOR

2022

CESIÓN DE DERECHOS DE AUTOR

Yo, Karen Alejandra Arias Salcedo con documento de identificación N° 0105598213, manifiesto mi voluntad y cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autora del trabajo de titulación: **"DISEÑO DE UN SISTEMA PROTOTIPO QUE PERMITA TRADUCIR COMANDOS DE TEXTO A LENGUAJE DE SEÑAS ECUATORIANO UTILIZANDO UN AVATAR VIRTUAL**", mismo que ha sido desarrollado para optar por el título de: *Ingeniera Electrónica*, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autora me reservo los derechos morales de la obra antes citada. En concordancia, suscribo este documento en el momento que hago entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, enero de 2022.

Karen Alejandra Arias Salcedo

C.I. 0105598213

CERTIFICACIÓN

Yo, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: **"DISEÑO DE UN SISTEMA PROTOTIPO QUE PERMITA TRADUCIR COMANDOS DE TEXTO A LENGUAJE DE SEÑAS ECUATORIANO UTILIZANDO UN AVATAR VIRTUAL"**, realizado por Karen Alejandra Arias Salcedo, obteniendo el *Proyecto Técnico con enfoque investigativo*, que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana.

Cuenca, enero de 2022.

Dr. Christian Raúl Salamea Palacios

C.I. 0102537180

DECLARATORIA DE RESPONSABILIDAD

Yo, Karen Alejandra Arias Salcedo con documento de identificación N° 0105598213, autora del trabajo de titulación: **"DISEÑO DE UN SISTEMA PROTOTIPO QUE PERMITA TRADUCIR COMANDOS DE TEXTO A LENGUAJE DE SEÑAS ECUATORIANO UTILIZANDO UN AVATAR VIRTUAL"**, certifico que el total contenido del *Proyecto Técnico con enfoque investigativo,* es de mi exclusiva responsabilidad y autoría.

Cuenca, enero de 2022.

Karen Alejandra Arias Salcedo

C.I. 0105598213

DEDICATORIA

Esta tesis va dedicada exclusivamente a mi tía Victoria, espero que esta herramienta pueda mejorar la calidad de vida y la inclusión de los sordos del Ecuador.

Karen Arias Salcedo

AGRADECIMIENTOS

A *mi familia* y *seres queridos* por haberme acompañado y apoyado en esta travesía. De manera especial a *mi madre* que nunca me dejó caer y a *Ricardo* por sus enseñanzas, a *Camila* que siempre supo qué decir, a *Fernando* con quien pude compartir cada momento y a *Taty* que me acompañó incondicionalmente en cada desvelo, llanto y alegría. Agradezco también a *mis profesores*, con ellos obtuve el conocimiento necesario para entender que el poder de cambiar el mundo está en mi mente y en mis manos. Finalmente, a *mí misma*, por haberme levantado después de cada caída, por saber que cada éxito y cada fracaso son solo míos, porque hoy estoy orgullosa de la persona en la que me convertí y, sobre todo, de ser una Ingeniera.

Karen Arias Salcedo

ÍNDICE

GLOSARIO

API: Interfaz de Programación de Aplicaciones - Application Programming Interface

ArSL: Lengua de Señas Árabe – Arabic Sign Language

ASL: Lengua de Señas Americano – American Sign Language

ATLASLang NMT: Traductor Automático Neuronal de Texto en Lengua Árabe a Lengua de Señas Árabe – Arabic Text Language into Arabic Sign Language Neural Machine Translation.

CONADIS: Consejo Nacional de Igualdad de Discapacidades

CNN: Redes Neuronales Convolucionales

DNN: Redes Neuronales Profundas

DSSM: Modelos Semánticos de Estructuración Profunda

FBX: Filmbox

FENASEC: Federación Nacional de Sordos del Ecuador

FPS: Cuadros por Segundo – Frames per Second

GANs: Redes Generativas Adversarias

GIF: Formato de Intercambio de Gráficos - Graphics Interchange Format

GUI: Interfaz Gráfica de Usuario – Graphic User Interface

HamNoSys: Sistema de Notación de Hamburgo – Hamburg Notation System

ISL: Lenguaje de Señas Indio – Indian Sign Language

LS: Lengua de Señas

LSE: Lengua de Señas Ecuatoriano

NER: Reconocimiento de Entidades Nombradas – Named Entities Recognition

NLP: Procesamiento Natural del Lenguaje – Natural Language Processing

OpenGL: Librería Abierta de Gráficos – Open Graphics Library

PDO: Extensión de Objetos de Datos de PHP – PHP Data Objects

PLS: Lenguaje de Señas Peruano – Peruvian Sign Language

RNN: Redes Neuronales Recurrentes

SynSets: Conjunto de Sinónimos – Synonyms Sets

UI: Interfaz de Usuario – User Interface

UTI: Universidad Tecnológica Indoamericana

VSL: Lenguaje de Señas Vietnamita – Vietnamese Sign Language

WebGL: Biblioteca de Gráficos Web – Web Graphics Library

XML: Lenguaje Extensible de Marcas - Extensible Markup Language

RESUMEN

Dentro del sistema educativo nacional, las personas sordas no han podido ser incluidas en su totalidad y han sido muy limitadas en el ámbito profesional y social. A penas en el año 2012 se creó el diccionario "Gabriel Román" con más de 4300 señas, permitiendo que la educación de los sordos deje de ser exclusivamente oralista. Para mejorar la calidad de vida de las personas con discapacidad auditiva y llevar al Ecuador al siguiente nivel de inclusión, esta tesis propone un software dedicado a la traducción de texto a lengua de señas ecuatoriana (LSE) y su aprendizaje a través de una agente virtual en 3D llamada Ema que fue diseñada en Makehuman y animada en Blender. Posteriormente, en Unity se importó el personaje con las animaciones, se programó el procesamiento de texto en C# en Visual Studio Community y se creó la interfaz gráfica de usuario (GUI). Finalmente, resultó en un ejecutable para Windows que funciona de la siguiente manera: el texto introducido en la barra de búsqueda de la GUI será procesado aplicando ciertas reglas de la gramática de la LSE, luego, será comparado (mediante los algoritmos de distancia y similitud entre *strings* Damerau-Levenshtein y Jaro-Winkler respectivamente) con las palabras del diccionario correspondientes a las 133 animaciones de señas que constituyen la base de datos. Si la palabra ingresada tiene una seña, el personaje la interpretará, caso contrario deletreará la palabra. Además, la interfaz cuenta con botones que permiten al usuario seleccionar cualquiera de las señas del diccionario para que el personaje las represente y un tutorial para manejar el programa. Entre otras características generales están la rotación del personaje con las flechas del teclado y un botón de presentación de Ema, con lo que, la persona sorda entiende de qué se trata el software. Para la evaluación del sistema se realizaron encuestas a 12 personas desde los 24 hasta los 71 años de edad sobre el uso de la interfaz, se mostró las señas interpretadas por Ema a una persona sorda que conoce LSE para verificar si son entendibles y se emplearon matrices de confusión para comparar la efectividad en el reconocimiento de texto al introducir 135 de las 500 palabras más comunes del español en los algoritmos Damerau-Levenshtein y Jaro-Winkler. Como resultado de las encuestas se encontró que en general la interfaz era fácil de utilizar, pero, requería una mejora en la distribución de los botones para que sea más familiar y parecida a otros programas conocidos. El 96,99% de las animaciones de señas fueron completamente entendibles y, en cuanto al reconocimiento de texto, el algoritmo Jaro-Winkler fue el ganador con un acierto del 95,6% frente al 60% del algoritmo Damerau-Levenshtein. En el futuro se espera poder agregar un reconocimiento de voz, mejorar el procesamiento natural de lenguaje, aumentar la base de datos de animaciones de señas y, llevar el software a más sistemas operativos, plataformas y dispositivos, haciéndolo accesible para todos.

INTRODUCCIÓN

Debido a la falta de una inclusión adecuada de los sordos en el sistema educativo, profesional y social del Ecuador, muchos de ellos se han visto limitados, no por su discapacidad, sino por los oyentes que han guardado recelosos el acceso a la información, por ende, se les ha negado a los sordos la oportunidad y los derechos de una mejor calidad de vida a pesar de tener la misma capacidad intelectual que cualquier persona de la comunidad oyente [1]. Esto se ve reflejado en la historia detrás de la lengua de señas ecuatoriana (LSE), antes de su estandarización en 2012, los sordos aprendían a comunicarse únicamente de forma oralista, es decir, leyendo los labios, y a través de señas que ellos mismos creaban para su círculo familiar [2]. Una vez publicado el diccionario digital de señas "Gabriel Román", la situación mejoró sustancialmente, sobre todo para la comunicación entre sordos, sin embargo, aún son excluidos en una reunión social, en la prestación de servicios, en la educación media y superior, y, en el campo profesional. Para disminuir esta brecha y, con el objetivo de ofrecer una alternativa para el aprendizaje y la comunicación efectiva de la LSE, se propone esta tesis, que consta de la creación de un software prototipo de traducción de texto a lengua de señas y su enseñanza, a través, de un avatar virtual en 3D. De esta manera, cualquier persona puede comunicarse y aprender LSE, y los sordos obtienen una llave a la información. Este trabajo está dividido en 4 capítulos principales:

El capítulo 1 aborda la fundamentación teórica que empieza con la comprensión de lo que es una persona sorda, continúa con parte de la historia de la LSE y sus reglas gramaticales y sintácticas, y, termina con una explicación breve sobre los tipos de software y los algoritmos de procesamiento natural de lenguaje utilizados en la producción de este proyecto.

El capítulo 2 contiene el marco metodológico que describe el desarrollo de la tesis, desde la creación del personaje en Makehuman y la creación de la base de datos de animaciones de señas en Blender, hasta la creación de la interfaz gráfica en Unity y la implementación de los algoritmos de reconocimiento de texto y traducción a LSE.

En el capítulo 3 se muestran los resultados obtenidos y la evaluación al software final con ayuda de: encuestas para identificar la operatividad de la interfaz gráfica, matrices de confusión para definir el mejor algoritmo de reconocimiento de texto y, la consulta a una persona sorda conocedora de LSE que compruebe si las señas realizadas por el personaje son comprensibles.

El capítulo 4 comprende las conclusiones, recomendaciones y el trabajo a futuro que puede surgir a partir de este proyecto.

Finalmente, se encuentran los apéndices A y B que contienen las encuestas realizadas y, los cálculos de las matrices de confusión de los algoritmos de reconocimiento de texto utilizados, respectivamente.

ANTECEDENTES DEL PROBLEMA DE ESTUDIO

Durante los últimos años se han desarrollado diferentes métodos y plataformas para traducir el lenguaje utilizado por oyentes en diferentes idiomas, a su equivalente en lenguaje de señas (LS). El objetivo de estas plataformas es ofrecer una herramienta que facilite: la enseñanza de lenguaje de señas, inclusión de alumnos sordos en clases, la interacción con dispositivos electrónicos y páginas web, traducción de libros, la posibilidad de entablar una conversación en tiempo real, entre otras aplicaciones. De esta forma, se puede lograr una mayor integración de las personas con discapacidad auditiva en la sociedad.

La literatura existente en torno a la temática de LS es amplia en el ámbito global, pero ya en torno al contexto ecuatoriano, la misma se ve reducida. Para el desarrollo del presente estudio se revisaron cuatro tesis, tres plataformas gubernamentales y dieciocho artículos científicos relacionados a diferentes tecnologías que traducen el LS. De ellos, podemos extraer [3] que es un sistema bidireccional donde una cámara 3D (Microsoft Kinect V2) escanea los gestos y los lleva a texto o voz (sign-to-text). A la vez, reconoce texto o voz como entrada, busca la seña equivalente en una base de datos y lo interpreta en señas (text-to-sign), utilizando un personaje virtual animado en Unity y 3D Studio Max. Otro sistema bidireccional es VirtualSign [4]. Realiza la traducción entre texto y LS de seis países europeos y Brasil, empleando dos componentes principales: text-to-sign y sign-to-text. Para el primero se introduce el texto y se selecciona el idioma, luego, el texto pasa por las reglas gramaticales del idioma y cambia de acuerdo al orden de palabras y los tiempos verbales, para concluir busca la oración en la base de datos y la seña equivalente la presenta en LS por medio de un avatar 3D en Unity. En el segundo, recae la traducción en un conjunto de guantes de datos y Kinect. Una característica importante que ofrece VirtualSign es la posibilidad de mostrar expresiones faciales

como neutral, feliz, triste, sorprendido, inquisitivo y enojado en sus animaciones. Además, para ampliar su base de datos poseen editores que ingresan animaciones de LS desde un diccionario sin conocimiento previo y validadores certificados que avalan a los editores.

Siguiendo el esquema (text-to-sign), un prototipo para enseñar y aprender lenguaje de señas árabe (ArSL) es implementado en [5]. Se compone de cuatro módulos (ART, MSA, LIT y SAR). Entonces, un texto árabe dado será procesado por el módulo ART, tokenizado y pasado al segundo módulo MSA para extraer las especificaciones morfológicas. El resultado lo trata el módulo LIT, que traduce a ArSL y luego se visualiza con el módulo SAR, en donde un elemento de LIT es buscado en el diccionario de señas. Si existe, su animación se transfiere a un archivo temporal para la interpretación; caso contrario, se muestra la palabra deletreada en LS. Un modelo similar es mostrado en [6], ya que, traduce inglés al modelo de traducción de archivos de lenguaje de marcado de gestos de señas (SiGML). El sistema acepta texto como entrada y lo representa con el sistema de notación de Hamburgo (HamNoSys), luego es trasladado a SiGML para que pueda ser interpretado por cualquier software de renderización 3D que permita este formato. Para la animación tiene en cuenta la orientación, localización y movimientos de las manos de cada una de las categorías o glosas del LS, tanto manuales como no manuales. Un traductor para el LS peruano a través de un asistente virtual 3D es presentado en [7], con el objetivo de llegar a traducir enciclopedias y escritos académicos. Para ello, primero obtiene el texto y lo tokeniza, luego se hace el reconocimiento de entidades nombradas (NER) con la ayuda del procesamiento natural del lenguaje (NLP) proveído por la Universidad de Standford; realiza un modelo gramatical y cada oración es analizada léxica y sintácticamente, resultando en oraciones con palabras morfosintácticas; identifica el conjunto de sinónimos (SynSets), para las oraciones que tienen nombres, adjetivos y verbos; aplica la librería NLTK de python para manejar las ambigüedades; se conecta con la base de datos WordNet, donde determina cada palabra de la traducción; y, finalmente, el

avatar genera la seña con un asistente virtual. Un traductor automático neuronal de texto en lenguaje árabe a lenguaje de señas árabe (ATLASLang NMT) es mostrado en [8]. Consiste en la traducción de una oración árabe a ArSL en tres pasos: análisis morfosintáctico (le da características morfológicas a cada palabra de la oración), transferencia (codifica cada palabra con propiedades morfológicas resultando en un vector numérico que puede ser aceptado por la red neuronal) y generación (se decodifica una oración en ArSL con la base de datos de animaciones). Mediante HamNoSys convierten los símbolos en formato SiGML que se puede mostrar en un avatar 3D con un software de renderización. En [9], se hace una compilación sobre sistemas de notación de signos y sistemas de traducción a lenguaje de señas a través de avatares 3D de Estados Unidos y Europa. También, propone un traductor de LS a partir de texto checo y ruso. Su arquitectura se compone de cuatro partes, la primera que procesa el texto en la entrada y genera transcripciones, fonemas y visemas (componente visual que representa la postura del rostro y boca al hablar [10]) con HamNoSys. La segunda se trata de un modelo facial 3D humanoide que controla la articulación de labios, mímicas y expresiones faciales. En la tercera muestra un modelo de la parte superior del cuerpo en 3D controlado por HamNoSys y codificado con una librería abierta de gráficos (OpenGL). La última parte hace referencia a la interfaz de usuario que integra los tres componentes anteriores para formar el avatar virtual. Un sistema generador de LS basado en la gramática del lenguaje de señas indio (ISL) es [11]. Es una aplicación móvil que trabaja en la traducción de inglés a ISL y posee un rico corpus de palabras en inglés y oraciones de uso común. Se compone de un analizador de la gramática de ISL, se vale de HamNoSys para la transcripción y el formato de salida de sus archivos es SiGML para la animación con un avatar 3D. Dentro de este artículo, se citaron otros modelos similares como TEAM [12], INGIT [13] que traduce texto de lenguaje hindi a ISL, un traductor del habla árabe a ArSL con el reconocimiento de voz de Microsoft [14] y un traductor de texto en español a LS en español por glosas [15]. Existen otros ejemplos de traducción text-to-sign, que no solo incorporan asistencia para personas con

discapacidad auditiva, sino también, para personas con discapacidad visual, como por ejemplo BCD API [16], que es una base de datos que contiene todos los gestos de LS, así como su contexto y configuración manual, está pensada principalmente para desarrolladores dedicados a hacer más accesible el contenido digital para la educación, dispone de un controlador de complemento y uno de conexión, que funcionan de la siguiente manera: el controlador de complemento utiliza una llave para para identificar una aplicación usada en BCD API y después se comunica con la base de datos para verificar a que funcionalidades tiene acceso, el controlador de conexión emplea la llave de la aplicación para determinar a qué base de datos tiene acceso, y luego la aplicación puede enviar mensajes de consulta a las bases de datos a través de PHP Data Objects (PDO). Finalmente, se desarrolla la animación dentro de Unity Game Development Platform, ya que, soporta tecnologías externas como reconocimiento de voz, síntesis de texto y traducción a LS. Un nuevo método para traducir texto ruso a LS ruso basado en la comparación de estructuras sintácticas se manifiesta en [17]. Primero resalta la lista de alternativas de significado de una palabra a través del análisis morfológico de palabras en una oración, análisis sintáctico y semántico, debido a que, la calidad de traducción se determina por la relación de una palabra con otra para resolver el problema de ambigüedad léxica; luego, procesa la fraseología y las preposiciones, asegura el significado léxico y busca el gesto correspondiente. En este artículo, cita a Zardoz [18] que traduce de inglés a ASL.

En el año 2014 se presentó TuniSigner [19], un sistema basado en la interpretación de señas a través de un avatar de forma automática con el objetivo de enseñar a leer y escribir, comenzando por el lenguaje de señas escrito Sutton SignWriting. Sutton SignWriting es el método más común para la escritura de cualquier lenguaje de señas, ya que, transcribe los gestos en símbolos de dos dimensiones que representan expresiones faciales, movimiento de manos y cambios de postura [19]. TuniSigner toma la notación SignWriting en formato de lenguaje

extensible de marcas (XML) y genera el equivalente LS que será realizado por el avatar virtual.

SignAR [20] es una aplicación móvil que traduce a lenguaje de señas capturando primero el texto o imagen con la cámara de un celular inteligente y lo procesa para obtener una animación de realidad aumentada mostrando la seña correspondiente con un avatar generado en Unity.

Fuera de los artículos relacionados con el uso de un avatar virtual en tres dimensiones para producir la traducción de texto a LS, está un servicio web que emplea el formato de intercambio de gráficos (GIF) integrando varias imágenes pertenecientes a cada palabra para generar la seña o secuencia [21]. Este servicio lee las palabras en idioma chino, las ordena desde la más larga y las compara con el diccionario que almacena las palabras que pueden ser traducidas a LS, luego, las guarda en un array y las compara con las del mensaje, después, busca imágenes de las palabras que pasaron la fase anterior para producir la secuencia de imágenes como GIF.

Todas las plataformas presentadas muestran traducciones de LS en diferentes idiomas como el ruso, indio, portugués, chino, español e inglés. No se han encontrado artículos científicos referentes al lenguaje de señas ecuatoriano, pero si tesis que involucran diferentes procesos o dispositivos para lograr la traducción. Uno de ellos es la interpretación bidireccional del alfabeto en LS con una mano robótica y una aplicación móvil [22]. Propone un sistema donde se envía texto desde una aplicación móvil de forma inalámbrica a la mano robótica y ésta realiza la seña correspondiente al deletreo de la palabra mediante LS. También, a través de un guante con sensores se pueden trasladar las señas realizadas por una persona que lo viste, permitiendo a la mano robótica imitar esos movimientos y a la vez convertirlos a texto que llegará a la aplicación móvil. Un guante similar, pero para niños, es presentado en [23]. Cuenta con 8 sensores resistivos que varían su valor al ser doblados, envían la información por USB a un microcontrolador que procesa los datos y luego a una computadora que los convierte en la seña concerniente a la letra

del alfabeto en LS, la cual se muestra en una interfaz realizada en Matlab R2010a con el propósito de enseñar LS y crear otro tipo de juegos didácticos. En [24] se diseña e implementa otro guante con sensores resistivos, sensibles al doblarse. En este caso, los datos son procesados a través de Arduino y al igual que los anteriores, solo puede representar una letra del alfabeto en LS. Por otra parte, [25] proporciona un método de traducción a través del reconocimiento de texto para devolver la seña correspondiente por un personaje virtual. Esto con el objetivo de mejorar la enseñanza del LS a niños y niñas con discapacidad auditiva en centros educativos utilizando una red de computadoras que permiten a los usuarios interactuar con el programa.

Entre otras soluciones integradas por el gobierno, tenemos el Diccionario Oficial de la Lengua de Señas Ecuatoriana, un diccionario en formato web [26], y el Glosario Básico de Lengua de Señas Ecuatoriano [27]. Sin embargo, ninguno es capaz de aportar la sensación de traducción en tiempo real, ya que, solo contemplan el uso de videos explicativos y/o gráficos estáticos.

JUSTIFICACIÓN

De acuerdo con el censo de 2010, en el Ecuador existen 207.541 personas con discapacidad auditiva [28], de los cuales 3.719 están en edad escolar [29]. No se ha logrado incluir a niños sordos al sistema educativo regular en su totalidad, principalmente porque los docentes desconocen el LS y por la gran cantidad de alumnos en las escuelas fiscales [1]. La educación para sordos en Ecuador era exclusivamente oralista, es decir, se les enseñaba a pronunciar palabras y leer los labios; a partir de 1980 se realiza la inclusión del LS en el país y apenas en 2012 se presenta el Diccionario Oficial de la Lengua de Señas Ecuatoriana "Gabriel Román" con más de 4.300 palabras, que se lo puede encontrar a manera de libro digital o como una página web [26] que cuenta con videos explicativos para construir las señas [30]. Sin embargo, hoy en día solo 66.111 ecuatorianos conocen el lenguaje de señas [1], lo que impide que las personas con discapacidad auditiva se desenvuelvan adecuadamente en ámbitos educativos, profesionales y sociales, por lo que, no pueden comunicarse efectivamente con la comunidad oyente. Con el objetivo de disminuir esta brecha en la sociedad, se ha desarrollado esta propuesta de tesis para que los ecuatorianos puedan aprender activamente el lenguaje de señas y, además, generar la traducción de textos en LS ecuatoriano mediante un avatar 3D. Finalmente, los no oyentes tendrán una herramienta para integrarse a un mundo que no fue diseñado para ellos.

OBJETIVOS

OBJETIVO GENERAL

Diseñar un sistema prototipo que permita traducir comandos de texto a lenguaje de señas ecuatoriano utilizando un avatar virtual.

OBJETIVOS ESPECÍFICOS

- Crear una base de datos con las señas más comunes del lenguaje de señas ecuatoriano con su equivalencia en comandos de texto.
- Desarrollar un algoritmo de búsqueda para comparar un comando en forma de texto introducido por el usuario con la base de datos.
- Diseñar un intérprete virtual en un software de animación 3D para la generación de señas.
- Programar rutinas que generen las animaciones correspondientes a cada seña según el comando de texto introducido.

CAPÍTULO 1

1. FUNDAMENTACIÓN TEÓRICA

Luego de analizar los antecedentes, se vio que existen diversas formas de llevar a cabo un "traductor de texto a lenguaje de señas" mediante diferentes tipos de software, hardware y algoritmos. Sin embargo, para este proyecto se destacan el uso de tres plataformas principales para la creación, animación del avatar 3D, y, para la composición de la interfaz gráfica: Makehuman, Blender y Unity respectivamente. Para realizar las animaciones del LSE, se empleó como base el Diccionario Gabriel Román [2].

1.1 ¿Qué es una persona sorda?

La sordera o hipoacusia es una condición fisiológica con diversos orígenes que se presenta como la pérdida de la capacidad auditiva en un individuo en diferentes niveles que van desde los casos leves hasta los más severos [31].

1.1.1 Visión clínica-terapeuta

Recoge todos los síntomas y consecuencias biológicas que la sordera ocasiona en una persona. Aquí se incluyen las cirugías y otras intervenciones logopédicas y psicológicas para recuperar el habla y mitigar las "consecuencias" de esta discapacidad. Intenta integrar a los niños sordos a través del conocimiento de la lengua hablada. En conclusión, la sordera es una limitación en las áreas: cognoscitivas, comunicativas y de lingüística [32].

1.1.2 Visión socio-antropológica

La palabra sordomudo (documentada por primera vez en 1975) suele utilizarse para referirse a las personas sordas, sin embargo, también implica una

connotación discriminatoria, ya que, supone que estas personas no tienen la capacidad de pensar y comunicar. Actualmente, se denominan a sí mismos como "sordo/a" y gracias a sus características propias han conformado la Comunidad de Sordos [32].

1.2 Lengua de Señas Ecuatoriana (LSE)

1.2.1 Inicios

En [30] se muestra que el primer escrito sobre LSE, "Lenguaje de Señas: Guía Básica Sobre una Comunicación Especial Tomo I", se desarrolló por la asociación de sordos Ponce de León en Quito en 1988. A partir de la recopilación y registro de 250 señas de Quito y Guayaquil se elaboraron ilustraciones, fotografías y dibujos explicativos. Luego de dos años de consulta con las asociaciones de sordos del país, en 2012 la Federación Nacional de Sordos del Ecuador (FENASEC) y la Universidad Tecnológica Indoamericana (UTI) coordinan la publicación del diccionario oficial de lengua de señas ecuatoriana "Gabriel Román", desarrollado por el Consejo Nacional de Igualdad de Discapacidades (CONADIS), que hace un compendio de 4.363 señas ilustradas y su respectiva glosa española dividido en 2 módulos e incluye un CD interactivo. Actualmente, el diccionario "Gabriel Román" se lo puede encontrar en formato web con más de 5.000 palabras explicadas con gráficos y videos [2].

1.2.2 Connotación gramatical y sintáctica.

Se considera a la LSE como viso-gestual, debido a que, se percibe a través de la visión y se manifiesta con el posicionamiento de las manos y, la expresión facial y corporal [32]. En Ecuador se distinguen 2 dialectos en el lenguaje de señas de acuerdo a las regiones costa y sierra. Además, se evidencia un 30% de influencia de la ASL y un 20% de la lengua de señas española. El otro 50% fue desarrollado por las comunidades de sordos del país [30]. La LSE posee sus propias estructuras

sintácticas y organizativas [32]. Para representar cada postura de la mano cuenta con 82 variantes de señas divididas en 9 grupos de configuraciones [2].

Existen ciertas reglas gramaticales y características de la lengua de señas que se aplican de forma general [16], las cuales están resumidas en la Ilustración 1. Sin embargo, dependiendo del idioma y la cultura de cada país, se producen variaciones.

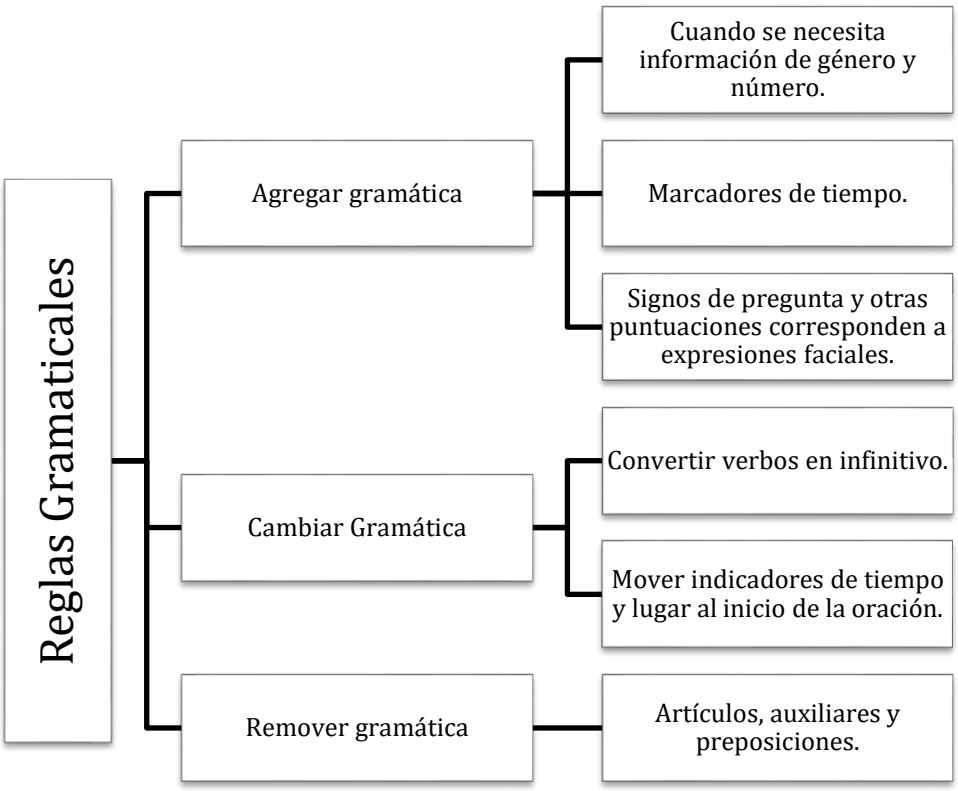


Ilustración 1: reglas gramaticales generales de la lengua de señas.

1.3 Makehuman

Makehuman es un software libre de modelado de humanoides en tres dimensiones que utiliza barras deslizables para definir la apariencia del personaje. Entre los parámetros principales, se pueden asignar: edad, peso, sexo, raza, tamaño

de nariz, forma de cara, proporción de brazos y piernas, entre otros. Su diseño se basa en Python y utiliza Tecnología Morphing 3D por lo que puede crear bebés, adolescentes, jóvenes o adultos. Inicia con un humanoide andrógino que, de acuerdo, a los valores introducidos por el usuario, se definen atributos masculinos y/o femeninos. Se puede crear prácticamente cualquier personaje debido a su gran base de datos y sus algoritmos de interpolación lineal y cálculo de factor de forma [33].

1.4 Blender

1.4.1 Historia

Ton Rosendaal desarrolló Blender a mediados de los 90's para su compañía de animación que quebró en 2002. Después, Ton fundó la organización sin ánimo de lucro "Blender Foundation" con la que, en 7 semanas, reunió 100.000 EUR para convencer a los inversionistas de convertir a Blender en un software de código abierto. El 13 de octubre de 2002 se publica bajo los términos de la Licencia Pública General (GNU por sus siglas en inglés). Actualmente, voluntarios de todo el mundo, guiados por Ton, desarrollan y mantienen Blender [34].

1.4.2 ¿Qué es Blender?

Blender es un software de modelado 3D de código abierto desarrollado y mantenido por Blender Foundation. Además, tiene compatibilidad con los sistemas operativos: Windows, Mac OS X y Linux [34].

1.4.3 Creación de un personaje 3D

Blender utiliza formas geométricas primitivas para modelar un personaje en tres dimensiones. A través de cubos, esferas, cilindros, etc; se da forma a los personajes. Primero se modifica la cantidad de vértices y caras de la figura básica, también, se juntan las formas geométricas y se las estira, encoge, etc en la pestaña

de edición. Además, se pueden agregar modificadores para crear simetría y subdivisiones para que el personaje tenga una superficie más suave. Al entrar al modo de esculpido, las figuras toman la consistencia de plastilina moldeable para darle los detalles finales y conseguir una apariencia más realista. Finalmente, se genera la malla UV para agregar la textura al personaje [34].

1.4.4 Rigging y skinning

Una vez creado el personaje, se construye un esqueleto o armadura de acuerdo a su tamaño, este proceso se llama *rigging*. Luego, se ejecuta el *skinning*, en donde el esqueleto debe ajustarse a cada parte del cuerpo del personaje a manera de huesos y articulaciones, tomando en cuenta los ejes de dirección de cada uno de los huesos. Después, se emparentan el personaje y el esqueleto para generar los controladores de cada grupo de huesos, facilitando la animación [34].

El primer hueso usualmente toma el nombre de "raíz". A partir de este hueso, se crea una cadena de jerarquía, donde los huesos nacidos de la raíz serán sus hijos. Entonces, al mover el hueso raíz, los otros huesos se moverán por inercia (cinemática directa). Sin embargo, con los controladores se pueden mover los huesos de la punta de la cadena, es decir, una mano o un pie y esto provocará el movimiento de toda la extremidad (cinemática inversa) [34].

1.4.5 Key-framing

Se le denomina *key-framing* al proceso de asignar una postura o un parámetro específico como rotación, escala y posición a un personaje u objeto en un punto de tiempo determinado. De esta manera se generan los fotogramas claves de la línea de tiempo que luego serán procesados por la computadora para obtener los fotogramas intermedios a través de interpolación. Dentro de Blender existe una

herramienta llamada “graph editor” que permite controlar como serán interpolados los fotogramas intermedios [34].

1.4.6 Principios de animación

En 1980, los animadores de Disney desarrollaron los 12 principios básicos de animación hecha a mano [34] que aún hoy, son la guía principal de cualquier animador digital:

1. Las técnicas de compresión y extensión son una buena manera de exagerar la animación y dar más sensación de movimiento.
2. La anticipación es utilizada para que la audiencia sepa que el personaje se va a mover.
3. La puesta en escena es un método que mantiene al público enfocado en el tema principal.
4. Existen dos técnicas de animación:
    - Acción directa hace referencia a la creación de cada pose de animación en forma de secuencia.
    - Pose a pose en cambio genera las posiciones más importantes para mostrar las acciones y ajustar el tiempo correctamente, antes de agregar más detalles.
5. El seguimiento del personaje separa las partes del cuerpo porque algunas pueden parar de moverse, pero otras continuar haciéndolo. Por otro lado, la acción superpuesta significa que diferentes partes del cuerpo se moverán en diferentes tiempos y velocidades.
6. Deben haber tiempos de aceleración y desaceleración del movimiento del personaje.
7. Todos los movimientos deben ser en forma de arco para que se vean más realistas y menos robóticos.

8. Se deben crear acciones secundarias para dar soporte a la animación principal.
9. Se deben asignar tiempo y espacio a los personajes para crear la ilusión de movimiento dentro de las leyes físicas. El tiempo se refiere a la cantidad de fotogramas entre dos poses y el espacio a como esos fotogramas individuales son colocados.
10. Se utiliza la exageración para dar la ilusión de que el personaje tiene un peso y colisiona con otros objetos.
11. Se deben mantener el volumen y peso de los personajes durante toda la animación.
12. El personaje debe ser carismático y encajar la animación con la personalidad que se le quiera dar.

1.4.7 Exportación del personaje

Una vez que se ha creado el personaje, se le ha asignado un esqueleto, generado los controladores, y animado; se debe hacer un *bake* de la animación, es decir, convertir la animación en acciones y generar los fotogramas de interpolación. Luego, se seleccionan en orden todos los elementos del personaje y finalmente, los controladores. En la pestaña de exportación se escoge el formato *Filmbox (fbx)* y se ajustan los parámetros de salida como la dirección del eje principal, qué huesos exportar y la configuración de las animaciones, y el archivo estará listo para ser exportado [34].

1.5 Unity

1.5.1 ¿Qué es Unity?

Unity es un motor de juegos utilizado en múltiples plataformas para crear video juegos de calidad profesional. Cuenta con simulación de físicas, normalización

de mapas, oclusión ambiental del espacio de la pantalla, sombras dinámicas, entre otros [35].

1.5.2 Historia

Unity fue lanzado en 2005 y comenzó como un software dedicado para Mac OS pero a los pocos meses fue portado a Windows. Gradualmente se fueron incorporando más plataformas web en 2006, para iPhone en 2008, Android en 2010 y finalmente, consolas como Xbox y Playstation. Recientemente, se incluyó la biblioteca de gráficos web (WebGL) [35].

1.5.3 Ventajas de Unity

- Tiene un flujo de trabajo visual productivo, ya que, posee un editor sofisticado personalizable que se utiliza para diseñar las escenas, medios artísticos y códigos en objetos interactivos. Además, hace rápidas iteraciones con lo que se realizan pruebas cíclicas con diversos prototipos [35].
- Alto grado de soporte multiplataforma: los juegos se desarrollan en Windows y Mac, pero se pueden jugar en PC, web, móviles y consolas [35].
- Unity viene con un sistema de paquetes de funciones modulares que se combinan para construir objetos en el juego en forma de colecciones de componentes en lugar de una jerarquía de clases, esto facilita un prototipado rápido [35].

1.5.4 Desventajas de Unity

- El sistema de componentes de Unity puede llegar a ser tedioso en escenas complejas porque se debe revisar objeto por objeto para determinar que funciones se le asignaron a cada uno [35].
- Las librerías de código externas deben colocarse manualmente en cada proyecto que las necesite, porque, Unity no cuenta con una ubicación central compartida [35].

1.5.5 Importación de un personaje y sus animaciones

Para importar un personaje, este debe estar en formato *.fbx*. El archivo se lo coloca en la carpeta *Models* dentro de la carpeta *Assets* del proyecto de Unity. Luego se selecciona el personaje del área de trabajo y con el inspector se ve la secuencia de animaciones guardada en una acción, para convertirlas en clips de acuerdo al número de cuadros por segundo que ocupaba originalmente una animación [34]. En la ventana de jerarquías se crea un nuevo objeto 3D vacío y se arrastra el personaje hasta esa sección, esto hará que aparezca en escena.

1.5.6 Programación

Unity soporta algunos lenguajes de programación, principalmente C# y JavaScript. Entre los beneficios que tiene C# está la rigurosidad en el tipo de dato que requiere el lenguaje, mientras que JavaScript no maneja dicha rigurosidad. Además, JavaScript en Unity es diferente del utilizado en el desarrollo de páginas web por cómo trabaja en diferentes contextos de programación y, a su vez, es muy similar en comandos y funciones [35].

Cualquier ejecución de código en Unity se hace a partir de los archivos de código o scripts, enlazados a los objetos en escena (instancias). Después, el código se ejecuta dentro del motor de juegos de Unity en lugar de tener su propio ejecutable [35].

La clase base para los componentes de scripts se denominan *Monobehaviour*. Esta clase define como los componentes se unen a los objetos del juego. Además, proporciona dos métodos de ejecución automática [35]:

1. Start(): se llama una vez cuando el objeto se vuelve activo.
2. Update(): se llama en cada fotograma.

El código general se coloca dentro de estos métodos para ser ejecutado de forma cíclica, mientras corre el juego.

1.5.7 Creación de una interfaz de usuario (UI)

La interfaz de usuario se crea por la necesidad de transmitirle información al usuario, para ello contendrá diferentes elementos visuales como cuadros de texto, barras y botones. Unity posee cuatro sistemas diferentes de UI con diversos propósitos cada una:

1. Elementos de UI: sirve para extender el editor de Unity con ventanas y herramientas personalizadas.
2. IMGUI: es una interfaz creada a partir de *scripts* con secuencias de comandos y se utiliza para modificar el editor de Unity.
3. Unity UI: es una interfaz basada en objetos que es aplicable únicamente dentro del juego.
    - Canvas es el objeto maestro que contendrá y renderizará la UI.
    - RectTransform es la característica a cargo de posicionar y adaptar cada elemento de la UI en la pantalla.
4. UI Toolkit: interfaz basada en elementos visuales que pueden ser arrastrados, modificados y animados dentro de la ventana de UI Builder.

1.6 Interacción Humano – Máquina

1.6.1 Vínculo entre un humano y el agente virtual

En 2006 se publica el artículo “Virtual Rapport” de [36], que afirma que se crea un mejor vínculo entre el usuario y el agente virtual si es que existe una interacción bidireccional durante la conversación. Es decir, si el agente virtual tiene un lenguaje corporal inconsciente que produzcan la sensación de una comunicación

efectiva, agrado, confianza e influencia en el humano. En una segunda versión denominada “Virtual Rapport 2.0” publicada en 2011 por [37], se utilizó la teoría de tres factores de Tickle-Degnen y Rosenberg (positividad, atención mutua y coordinación) y se lo comparó con la primera versión del artículo, concluyendo que con esta teoría se predice mejor el momento en que debe hacerse la retroalimentación generando comportamientos más naturales que desembocan en un sentimiento más fuerte de vinculación entre el humano y el agente.

Por otro lado, el artículo “Why Fat Interface *Char*acters are Better e-Health Advisors” de [36], muestra que los usuarios perciben a los personajes con sobrepeso más confiables que los delgados, especialmente los que tienen un tipo de cuerpo similar.

En [38] se hizo un estudio para examinar si los usuarios reportan mayor compenetración con el agente virtual si estos son extrovertidos o introvertidos, para simular esos tipos psicológicos se le asignó tres características de diálogo. Sin embargo, los usuarios no reportaron mayor vinculación con un agente de similar nivel de extroversión.

En [36], [37] y [38] se pueden encontrar más estudios y tipos de agentes virtuales creados principalmente para chatbots, aplicaciones de ayuda, entre otros, donde se utilizaron técnicas que hacían que el agente imite al usuario a través de voz y/o texto para que se adapte a la personalidad del usuario. Otros que se diseñaron para conversar con adultos mayores, para proveer de personalidades socialmente aceptables en robots, para generar impacto emocional positivo en el usuario, para predecir la calidad de servicio en el diálogo con clientes, guías de museos, reconocimiento de emociones, etc.

1.6.2 Agentes Inteligentes

Un agente inteligente es un sistema capaz de procesar la información recibida a través de sensores y de reaccionar a ella con actuadores para alcanzar una

meta específica. En términos matemáticos se puede representar con una función mapea todas las secuencias posibles de percepciones P^* para una acción apropiada A [39].

1.7 Procesamiento del Lenguaje Natural

El Procesamiento del Lenguaje Natural (NLP por sus siglas en inglés) investiga el uso de procesos computacionales para entender los lenguajes humanos con el propósito de desarrollar tareas útiles [40]. Entre sus posibles aplicaciones encontramos:

- Sistemas de diálogo.
- Análisis léxico.
- Traducción automática.
- Generación del lenguaje natural.

De esta forma podemos entender que la forma física del lenguaje natural es el texto y su contraparte es la voz, la versión ruidosa del texto. La lengua de señas es una forma codificada de los símbolos del lenguaje natural. A través del NLP, se crean sistemas de diálogo, también conocidos como agentes conversacionales, agentes virtuales o chatbots; generalmente son utilizados en servicios de soporte técnico a herramientas de aprendizaje del lenguaje y entretenimiento. Existen diversas metodologías de aprendizaje para los componentes de un diálogo:

1.7.1 Aprendizaje Discriminativo

Entre los métodos discriminativos están los modelos semánticos de estructuración profunda (DSSM) [41], éstos se emplean generalmente en aplicaciones de aprendizaje profundo, para la clasificación de clases de texto y, además, intrínsecamente aprende las similitudes entre dos textos mientras descubre características latentes. Es un tipo de técnica de modelado de redes neuronales profundas (DNN) [42] para representar cadenas de texto en un espacio semántico

continuo y modelar semánticas similares entre dos cadenas de texto, además, tiene dos tipos de arquitectura similares: redes neuronales convolucionales (CNN) [42] que hace uso de capas con filtros convolucionales que son aplicados a características puntuales y, las redes neuronales recurrentes (RNN) [42] para el modelamiento secuencial. Las redes de memoria se aplican a preguntas y respuestas, modelamiento de lenguaje, etc; y trabajan leyendo iterativamente de componentes de memoria a través de saltos que pueden almacenar diálogos históricos y contextos a corto plazo para razonar sobre la respuesta requerida. Para tener un aprendizaje profundo efectivo, se puede pre-entrenar a un modelo con bases de datos relacionadas, permitiendo escalar a entornos más complejos [40].

1.7.2 Aprendizaje Generativo

Se consideran al agrupar objetos e instancias en los datos, extraer características de texto no estructurado o reducir la dimensionalidad. Se especializan en modelos neuronales generativos. Son comunes también los modelos de diálogo neuronal basados en codificación y decodificación, en donde la red de codificación utiliza el historial entero para codificar la semántica del diálogo y el decodificador genera una expresión natural del lenguaje. Las redes generativas adversarias (GANs) son específicamente para la generación de respuesta de un diálogo [40].

1.7.3 Aprendizaje basado en la toma de decisión

También conocido como política de diálogo, escoge la acción del sistema a cada paso de la conversación para guiar el diálogo satisfactoriamente. Incluye interacción con el usuario, negociación y oferta de alternativas. La optimización de gestores de diálogo estadístico utilizando el método de aprendizaje reforzado, encaja en el diálogo porque está pensado para situaciones donde la retroalimentación esté retrasada. Cuando un agente conversacional lleva a cabo un diálogo con un usuario, puede saber si el diálogo fue exitoso y si la tarea fue completada solo después de que el diálogo terminó [40].

De acuerdo a [43], dentro del procesamiento natural del lenguaje las oraciones de un texto son analizadas en términos de su sintaxis, esto provee un orden y estructura que es más susceptible a un análisis en términos de semántica o significado literal; esto seguido por una etapa de análisis pragmático donde se determina el contexto del texto. Las etapas para el NLP son las siguientes:

1. Entrada de texto.
2. Tokenización de lenguajes delimitados por espacio o lenguajes no segmentados.
3. Análisis léxico.
4. Análisis sintáctico: extrae el significado de una oración para determinar la estructura gramatical.
5. Análisis semántico: provee de significado a la oración.
6. Análisis pragmático: confiere un contexto a la oración.
7. Significado intencionado del usuario.

Uno de los principales problemas del NLP es la ambigüedad debido a los signos de puntuación y a los factores contextuales como la distinción de casos, longitud de palabras, terminaciones léxicas, prefijos y sufijos, clases de abreviaciones, nombres propios, entre otros. Además, la segmentación y tokenización están limitados por el sistema de escritura y la ortografía de un lenguaje.

También conocido como política de diálogo, escoge la acción del sistema a cada paso de la conversación para guiar el diálogo satisfactoriamente. Incluye interacción con el usuario, negociación y oferta de alternativas.

1.7.4 Algoritmos básicos para determinar similitud, distancia y superposición entre *strings*.

1.7.4.1 Distancia Damerau-Levenshtein

La distancia Damerau-Levenshtein es una métrica utilizada para medir la distancia entre dos *strings*, la cual está definida como la cantidad

de operaciones necesarias para transformar un *string* en otro [44]. Las operaciones empleadas para la conversión son:

a) Insertar: olor → color

b) Eliminación: color → olor

c) Sustitución: taza → tasa

d) Permutación: salri → salir

$$lev_{a,b}(i,j) = \begin{cases} max(i,j) \text{ si } min(i,j) = 0 \\ min \begin{cases} lev_{a,b}(i-1,j) + 1 \\ lev_{a,b}(i,j-1) + 1 \text{ caso contrario} \\ lev_{a,b}(i-1,j-1) + \mathbb{I}_{a_i \neq b_j} \end{cases} \end{cases} \quad (1)$$

1.7.4.2 Similitud Jaro-Winkler

Este algoritmo obtiene la similitud entre dos *strings*. Los *strings* que se sean similares al inicio tendrán una puntuación más alta [45]. El cálculo se hace a través de las siguientes ecuaciones [46]:

$$sim_j = \begin{cases} 0 & \text{si} \quad m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & \text{caso contrario} \end{cases} \quad (2)$$

$$sim_w = sim_j + \ell p(1 - sim_j) \quad (3)$$

Donde:

- $|s_i|$ es la longitud del $string\ s_i$
- m es el número de caracteres emparejados
- t es la mitad del número de transposiciones
- sim_j es la similitud Jaro entre dos *strings* s_1 y s_2
- ℓ es la longitud del prefijo común al inicio del *string* hasta un máximo de 4 caracteres.
- p es una constante que permite ajustar el algoritmo. Normalmente es igual a 0.1.

1.7.4.3 Coeficiente Szymkiewicz-Simpson

En [44] se encuentra también el coeficiente Szymkiewicz-Simpson que es también conocido como coeficiente de superposición y mide cuanto se superponen dos *strings* (X, Y). Si el coeficiente es igual a 1, indica que uno de los *strings* es un *substring* del otro. Si el coeficiente es 0, quiere decir que no tienen ningún elemento en común. El coeficiente se obtiene a través de la siguiente ecuación:

$$Overlap(X, Y) = \frac{X \cap Y}{min(|X|, |Y|)} \quad (5)$$

CAPÍTULO 2

2. MARCO METODOLÓGICO

Para llevar a cabo el traductor de texto a lengua de señas, se requieren 5 etapas principales: modelar el agente virtual 3D en Makehuman, crear las animaciones a partir de LSE y exportar el personaje en Blender, importar el personaje y delimitar sus animaciones en Unity, programar los algoritmos que enlazarán cada animación con su significado en texto y, finalmente, desarrollar la UI.

2.1 Modelado 3D del agente virtual en Makehuman.

El agente virtual fue desarrollado en el software Makehuman a partir de un humanoide andrógino (Ilustración 2) al que se modeló utilizando barras de parametrización para darle características físicas únicas, entre estas se encuentran las modificaciones propias del género escogido, forma de la cabeza y edad, profundidad y dimensiones del torso y extremidades, además, proporciones corporales, vello, cabello, color de ojos, peso y estatura, y aspectos generales de acuerdo a sus orígenes regionales como asiático, caucásico y africano (Ilustración 3). También, incluyen algunos tipos de vestimenta (Ilustración 4).

CAPÍTULO 2. MARCO METODOLÓGICO

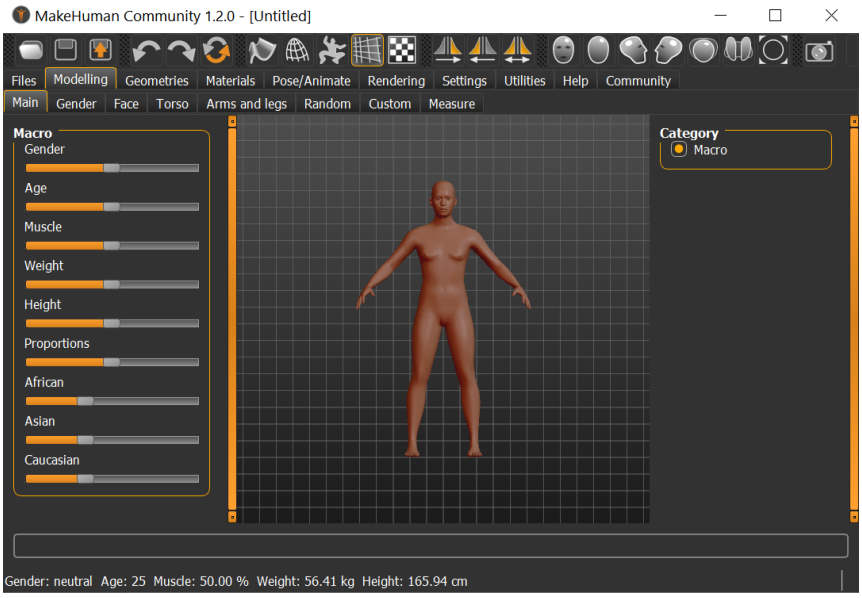


Ilustración 2: modelo inicial de Makehuman [Fuente: elaboración propia]

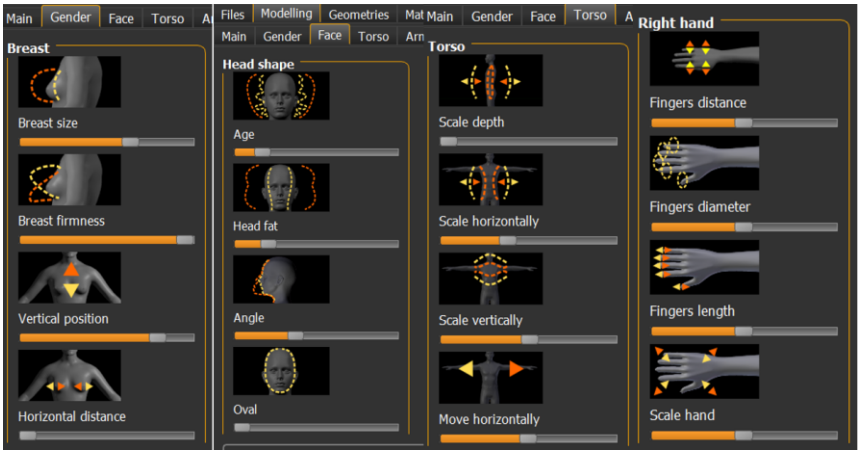


Ilustración 3: configuración física del personaje en Makehuman [Fuente: elaboración propia].

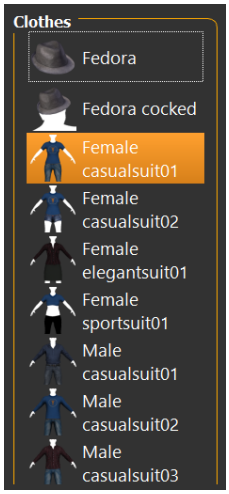


Ilustración 4: tipos de vestimenta en Makehuman [Fuente: elaboración propia].

Para el desarrollo de este proyecto se configuraron los parámetros con el objetivo de crear un personaje con apariencia femenina caucásica de alrededor de 20 años de edad y una altura de 1,60 m. Para ello se deslizó la barra de género hacia el lado izquierdo, logrando una apariencia femenina; las barras de origen asiático y africano se dejaron al mínimo, de esta forma prevalece el aspecto caucásico. La barra de edad apenas fue modificada por lo que se muestra una mujer joven. Las proporciones del cuerpo se calcularon automáticamente con la altura y peso asignados. Finalmente, se escogió una cabellera que mantenga las orejas a la vista porque son importantes en algunas señas del LSE. Se le asignó una vestimenta simple para que la atención del usuario se enfoque en la traducción del LSE.

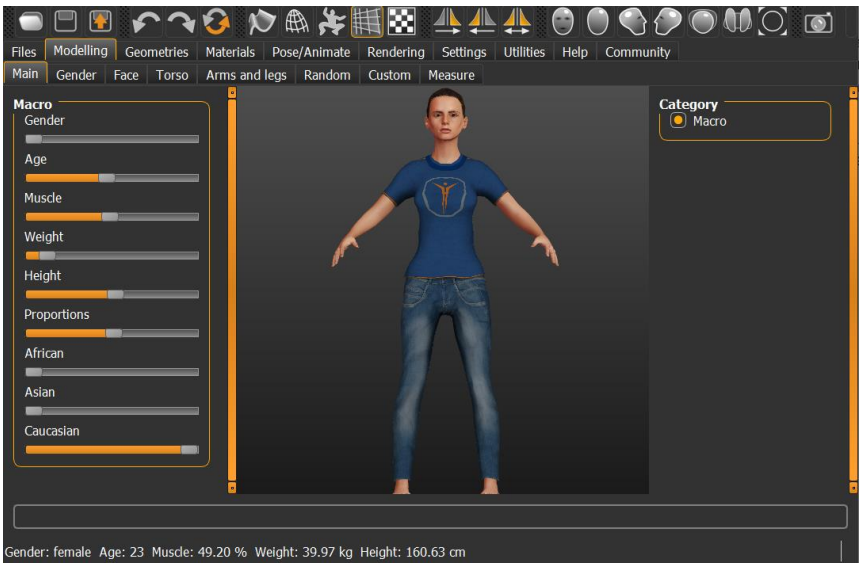


Ilustración 5: personaje terminado con sus características [Fuente: elaboración propia].

Una vez terminado el personaje se exporta en formato fbx. Este formato permite obtener las texturas utilizadas en el agente y el modelo para ser utilizado en cualquier software de animación.

2.2 Animación del personaje a partir del LSE en Blender.

En esta sección se detallará la elaboración de la base de datos de animaciones basadas en el LSE en el software Blender. Cabe recalcar que para este proyecto se utilizó Blender 2.93.1 en inglés y previamente se instaló el módulo adicional *Rigging: Rigify* para poder utilizar el esqueleto humanoide prediseñado de Blender y generar los controladores.

2.2.1 Importación del personaje a Blender.

En Blender, en la pestaña *file* > *import* > *FBX* se importó el personaje hecho en Makehuman aplicando la escala adecuada al personaje y se le corrigieron algunos detalles en el cuerpo dentro de la opción de esculpido.

2.2.2 Creación y asignación de un esqueleto al agente virtual.

Blender tiene distintos tipos de armadura, la primera es hueso a hueso y la segunda es un grupo de huesos prefabricados que tienen forma humanoide o de algunos animales. Dentro de las formas humanoides está la simple y la completa, para este proyecto se utilizó la completa, ya que, eran necesarios los huesos del rostro y manos para realizar correctamente las posiciones del LSE al momento de animar.

Se seleccionó el esqueleto en el botón *Add* > *Armature* > *Human (meta-rig)*, entonces aparece en escena un esqueleto humanoide con todos los huesos y articulaciones como se ve en la Ilustración 6.

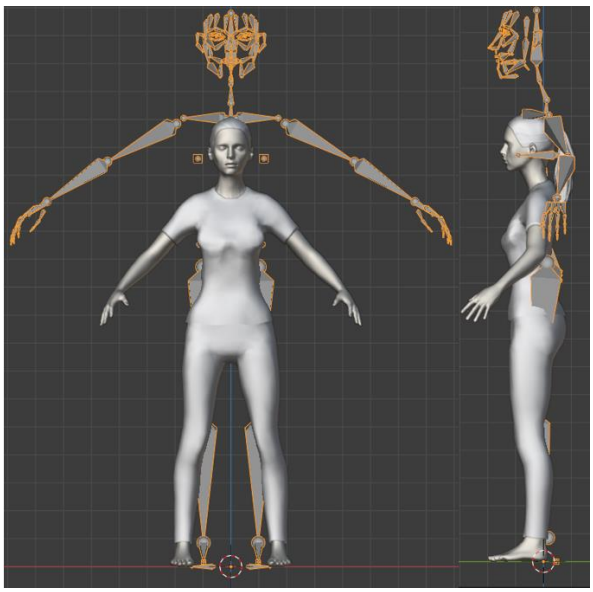


Ilustración 6: esqueleto Human (meta-rig) agregado a la escena [Fuente: elaboración propia]

Comienza el proceso de *rigging*, donde se ajustó el esqueleto para que encaje perfectamente con el cuerpo del modelo, para ello se modificó la escala de toda la armadura y se la posicionó exactamente en el centro general del personaje. Luego, en la opción *viewport display* se colocan los huesos como siempre visibles y en forma de palillos para poder encajarlos mejor (Ilustración 7). Durante el *skinning* se hicieron calzar los huesos generales del cuerpo y las extremidades (Ilustración 8) y,

finalmente, los huesos de manos y cara (Ilustración 9) con ayuda de las herramientas de simetría, rotación y traslación.

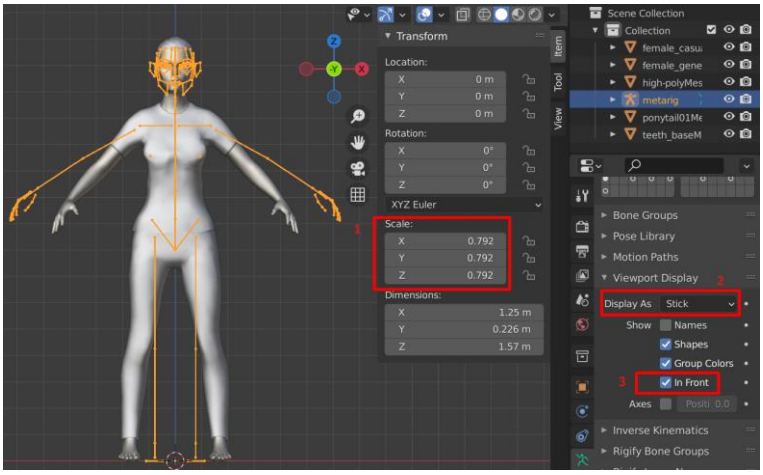


Ilustración 7: ajuste inicial de la armadura [Fuente: elaboración propia].

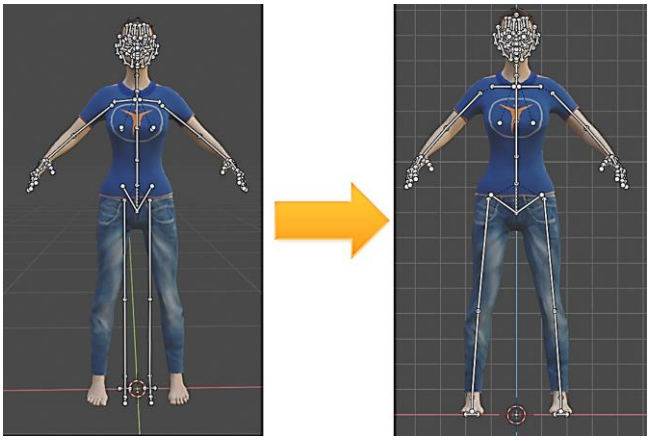


Ilustración 8: ajuste general de los huesos corporales [Fuente: elaboración propia].

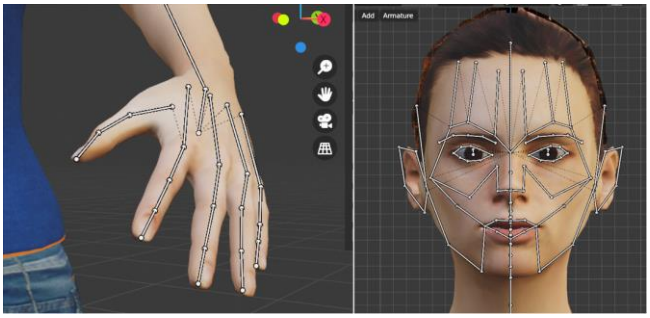


Ilustración 9: ajuste de huesos de manos y cara [Fuente: elaboración propia].

2.2.3 Generación de controladores.

Una vez estuvieron colocados adecuadamente los huesos, se seleccionaron todas las partes del personaje y por último la armadura, al presionar *ctrl+p* se muestra el menú de opciones para emparentar el esqueleto al personaje, se seleccionó la asignación de pesos automáticos dentro de la opción de deformación de la armadura para que el programa genere automáticamente la influencia que tiene cada hueso sobre cada parte del cuerpo y que a partir de eso se creen los controladores de cinemática directa e inversa. Los controladores se guardan en capas de huesos como se observa en la Ilustración 10 Luego, se hizo el pulido de la asignación de pesos automáticos, a través de la opción de pintura de pesos. En esta ventana el personaje adquiere una coloración azul si es que el hueso seleccionado no ejerce influencia en esa parte y de color rojo si ejerce la máxima influencia. Para que no haya cortes muy exagerados en el modelo, entre los extremos de la influencia se muestran una secuencia de colores que oscilan entre verde, amarillo y naranja que indican una transición suave entre una zona con influencia 0 y otra con influencia 1. Para saber exactamente qué hueso influye en cada músculo del personaje, Blender generó una lista de grupos de vértices, donde se muestran los huesos padres y sus hijos con nombres específicos de acuerdo a la parte del cuerpo en la que se encuentran (Ilustración 11).

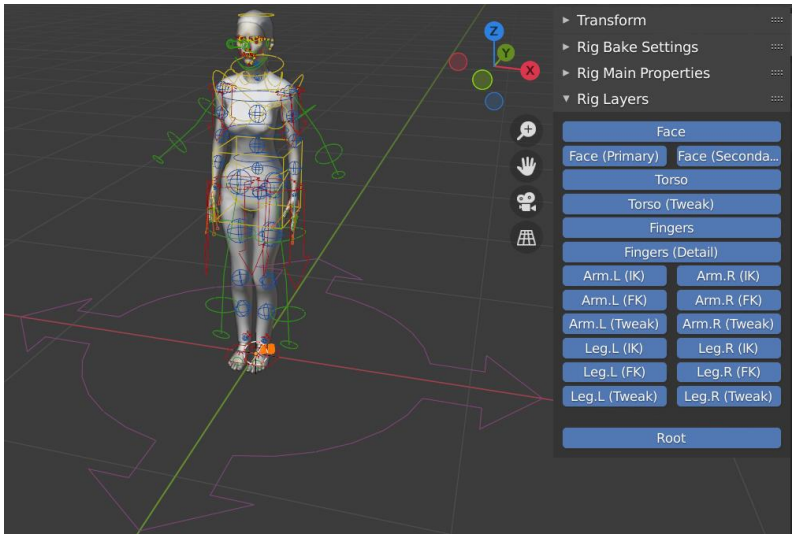


Ilustración 10: controladores y sus capas [Fuente: elaboración propia].

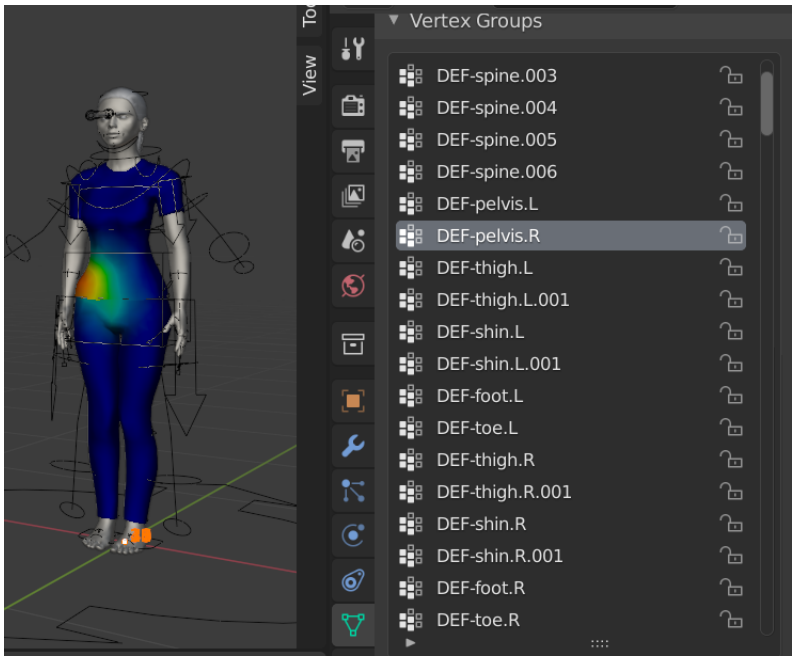


Ilustración 11: ejemplo de pintado de pesos del grupo de huesos de la pelvis derecha [Fuente: elaboración propia].

2.2.4 Animación

Con los controladores generados y los pesos corregidos se procede a animar el personaje en la ventana *pose mode*. En la parte inferior se encuentra la línea de tiempo con cada fotograma clave. Se utilizó la herramienta *Auto Keying* para que guarde automáticamente en la línea de tiempo un fotograma clave con la escala, rotación y/o movimiento cada vez que se realiza una modificación en la postura del personaje. Paralelamente, se colocaron marcadores al inicio de cada animación para saber el inicio y fin de cada una. Dentro de los parámetros de animación se configuró la animación a 30 fps para que coincida con Unity y el fin de la línea de tiempo en el fotograma 6.780 para que contenga toda la secuencia de animaciones.

Para iniciar con las animaciones, se tomaron como base los videos de LSE explicativos del Diccionario Oficial de la Lengua de Señas Ecuatoriana "Gabriel Román" [2] con las señas más comunes como: saludos y despedidas, meses y días de la semana, números, familiares, actividades, instituciones, prestación de servicios y pronombres, entre otros; y grupos de señas de preguntas frecuentes (¿Cómo?, ¿Cuándo?, ¿Dónde? y ¿Por qué?). También, se incluyó el abecedario tomado de la Glosario Básico LSE [27] para que se puedan deletrear nombres y palabras que no se hayan incluido dentro de las señas anteriores.

De los videos explicativos se identificaron las posiciones claves de la seña, movimientos importantes y cantidad de repeticiones, de manos, brazos, torso, cabeza y rostro. Por ejemplo, de la seña de la palabra "Universidad", se identifica la primera posición de la parte superior del cuerpo como se muestra en la Ilustración 12, luego se registra el movimiento realizado, en este caso, un movimiento circular de la mano desde la posición inicial hacia abajo (segunda posición clave) mientras se acerca la mano al pecho (tercera posición clave) y termina yendo hacia arriba nuevamente (Ilustración 14). Finalmente, se determina la cantidad de veces que realiza este movimiento, que son tres. Se puede animar de esta manera debido a que Blender

calcula los fotogramas intermedios a través de una interpolación lineal modificable en la ventana “graph editor”.

Ilustración 12: primera posición de "Universidad" [47].

Ilustración 13: segunda posición de "Universidad" [47].

Ilustración 14: movimiento de "Universidad" [47].

Las posiciones claves se almacenaron en la biblioteca de poses que Blender ofrece para poder aplicarlas nuevamente a un grupo de controladores si son necesarias, esto ahorra tiempo, ya que, hay varias animaciones que requieren de la misma postura específica de las manos u otra parte del cuerpo. De igual forma, se pueden copiar y pegar los fotogramas dentro de la línea de tiempo.

Una vez terminada la secuencia de 134 animaciones, se fueron corrigiendo una por una para evitar cualquier desfase ocasionado por la interpolación automática generada entre la animación de cada seña, dando la ilusión de que son independientes y no se solapen entre sí, tanto en la línea de tiempo (es decir, que una animación comience antes de que la anterior termine) como en el cuerpo del personaje (por ejemplo, si la mano del personaje atraviesa el cuerpo como en la Ilustración 15).

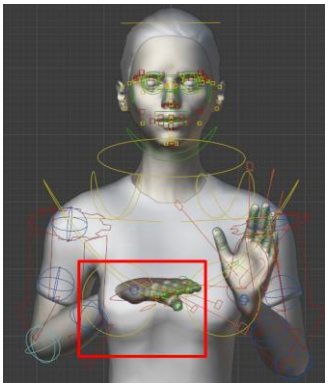


Ilustración 15: error de solapamiento del cuerpo del personaje [Fuente: elaboración propia].

En la línea de tiempo, se escalaron los fotogramas claves para que las animaciones duren 30, 60, 90 o 120 fps según su complejidad, de esta forma se logra una transición suave entre las posiciones clave de cada seña. Para monitorear en tiempo real como se visualizarán las animaciones, se trabajó con el personaje sin las texturas, porque, con las texturas activadas, al programa le tomaba más tiempo renderizar y mostrar cada animación provocando una falsa sensación de demora en la reproducción de los movimientos.

2.2.5 Exportación

Para exportar el personaje y sus animaciones desde Blender a Unity se siguieron los siguientes pasos:

1. Crear el proyecto en Unity donde se utilizará el personaje.
2. Aplicar todos los modificadores y transformaciones de escala, rotación y movimiento del personaje.
3. Seleccionar todas las capas del *rig* utilizadas y hacer un *bake* de las animaciones entrando en la pestaña *Object* > *Animation* > *Bake Action* ... Ahí se escogen el fotograma de inicio y del final de la

secuencia de animaciones, para ejecutar el *bake* se presiona el botón *Ok*.

4. Seleccionar en orden todos los objetos que componen el modelo 3D y al final seleccionar el *rig* con los controladores manteniendo la tecla *shift* presionada.
5. Ir a la pestaña *File > Export > FBX (.fbx)* para que se abra el menú de exportación. En el menú se selecciona la carpeta de destino que será la carpeta *Models* dentro de la carpeta *Assets* del proyecto de Unity. En el parámetro P*ath Mode* seleccionar la opción *Auto*, en la pestaña *Include* seleccionar la casilla *Selected Objects* y debajo seleccionar únicamente *Armature* y *Mesh.* Dentro de *Transform* en *Forward* cambiar *-Z Forward* por *Z Forward*. En *Armature* desactivar la casilla *Add Leaf Bones*. Mantener activada la casilla de *Bake Action* pero desactivar las casillas *NLA Strips* y *Force Start/End Keying.* Deben quedar los parámetros como se muestra en la Ilustración 16. Para finalizar, se le da un nombre al archivo y se presiona el botón *Export FBX*.

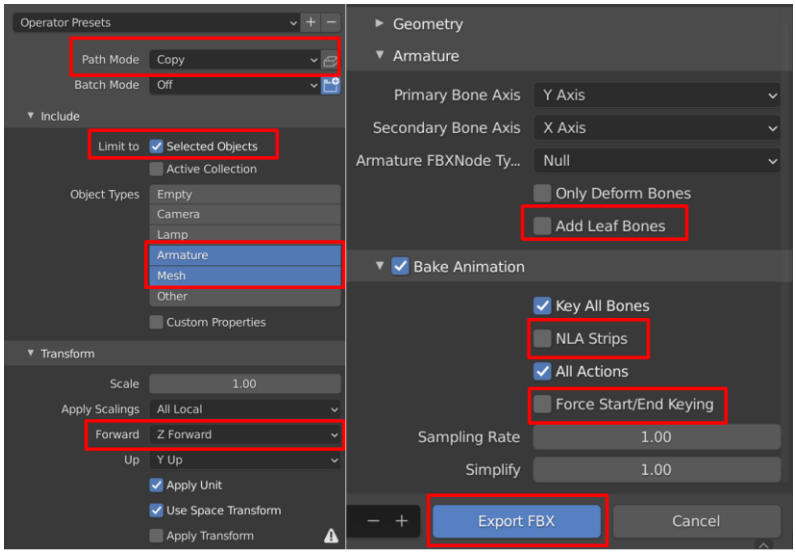


Ilustración 16: configuración de exportación de Blender a Unity [Fuente: elaboración propia].

2.3 Personaje y animaciones

En Unity, se importan el personaje con todas sus texturas, armadura y animaciones. Para poder utilizar el personaje en escena, primero se lo debe instanciar como un objeto y luego, separar en fotogramas el archivo general de animación.

2.3.1 Agregar un personaje a la escena.

En el panel de jerarquía de Unity, se crea un nuevo objeto 3D vacío instanciado como “player”, al cual, se arrastra el modelo con el personaje para que éste aparezca en escena. En este punto se deben comprobar que la dirección y tamaño del personaje sean los adecuados.

2.3.2 Separar las animaciones.

Para separar las animaciones del archivo general, dentro de la ventana del inspector, con el personaje seleccionado, se encuentra la pestaña de animación. En

esta pestaña, se selecciona el archivo de fuente de la animación y a partir de él se van delimitando los cuadros por segundo de cada animación y se les asigna un nombre. Finalmente, se aplican los cambios.

2.4 Configuración de escena en Unity.

La escena estará compuesta por una habitación en donde se encontrarán el personaje, la luz y la cámara. Después, el usuario únicamente visualizará lo que se encuentre en el rango de visión de la cámara.

2.4.1 Habitación

En Blender, se diseñó una habitación de 8 m^2 por 2 m de alto. De esta forma, se asegura que el personaje no se pierda de vista con la rotación de la cámara. La habitación se exportó a Unity, donde se le asignó la textura de ladrillo blanco a las paredes, un piso de madera al suelo y un cielo raso al techo. Además, a manera de decoración se agregó un plano con una imagen del abecedario en lengua de señas para el fondo de la habitación.

2.4.2 Luces

Se agregó una luz de tipo direccional dentro de la habitación, frente al personaje para que éste si visualice correctamente y una luz de punto detrás para que ilumine la habitación y no se generen sombras que entorpezcan la visualización de las señas.

2.4.3 Cámara y controles de movimiento.

Se colocó una cámara a 2 metros del personaje, con un rango visual de hasta 5 metros. Con el componente Cinemachine se crearon 3 anillos que hacen de órbita para la cámara, además, cada órbita se une de forma vertical con las otras para lograr el movimiento de la cámara en los tres ejes. Se establecieron la posición y las dimensiones de los anillos, logrando la visualización deseada como se puede ver en

la Ilustración 17. Finalmente, se asignó el movimiento de la cámara a las flechas del teclado.

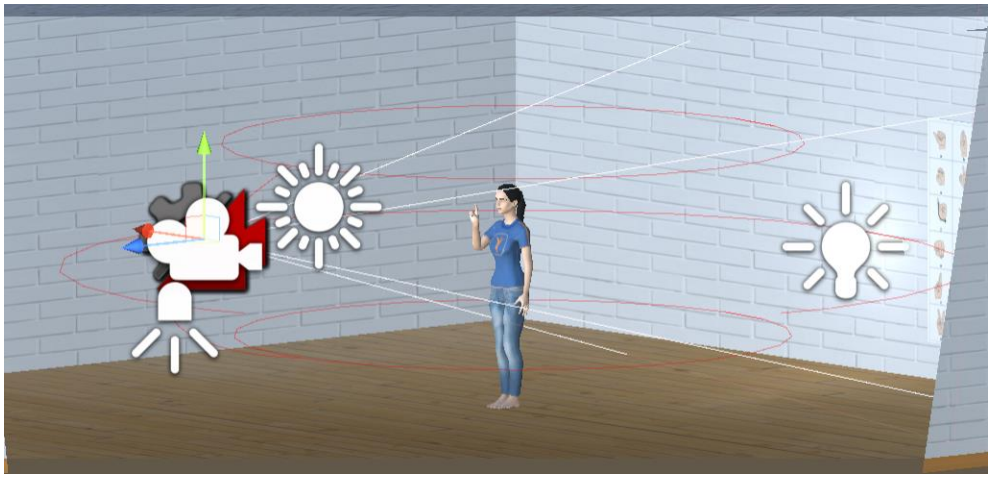


Ilustración 17: órbitas de la cámara de Cinemachine.

2.5 Creación de la interfaz gráfica con UI Toolkit en Unity.

La interfaz gráfica es lo que permitirá la interacción humano-máquina. Para la interfaz se establecieron los elementos que debe contener: un menú con botones de todas las animaciones, barra de búsqueda para escribir las frases o palabras que el personaje deba traducir, un botón de presentación del personaje y un pequeño tutorial de cómo utilizar el traductor.

2.5.1 Instalar el complemento UI Toolkit.

Recientemente, Unity lanzó el complemento UI Toolkit, el cual permite crear interfaces gráficas de forma más dinámica que la UI tradicional, ya que, se une al complemento UI Builder con el cual se genera la interfaz en una ventana a parte de la escena. Para utilizarla, en la pestaña de *edit* > *preferences* > *add* se debe colocar el enlace de Git URL `com.unity.ui` y, para UI Builder, `com.unity.ui.builder`.

2.5.2 Crear UIDocument.

En el panel de jerarquía de Unity, se crea un UIDocument, este objeto permitirá la ejecución del lienzo creado con UIBuilder y el script diseñado para su control. Dentro del proyecto en *Assets* se crea otro UIDocument que será el archivo contenedor de la interfaz. Este archivo debe ser asignado al objeto UIDocument a través del inspector y puede ser abierto para comenzar a crear la interfaz.

2.5.3 Agregar elementos en la interfaz.

UI Builder trae por defecto una librería con los controladores básicos como botones, etiquetas, barras de desplazamiento, menús, campos de texto, etc que deben ser asignados dentro de un contenedor de elementos visuales a manera de hijos. El contenedor debe ser arrastrado a la pantalla de construcción de la interfaz y posteriormente se le agregan los controladores. De todos ellos se configuró la posición, tamaño y diseño y, se les asignó un nombre para poder llamarlos desde el script de la interfaz.

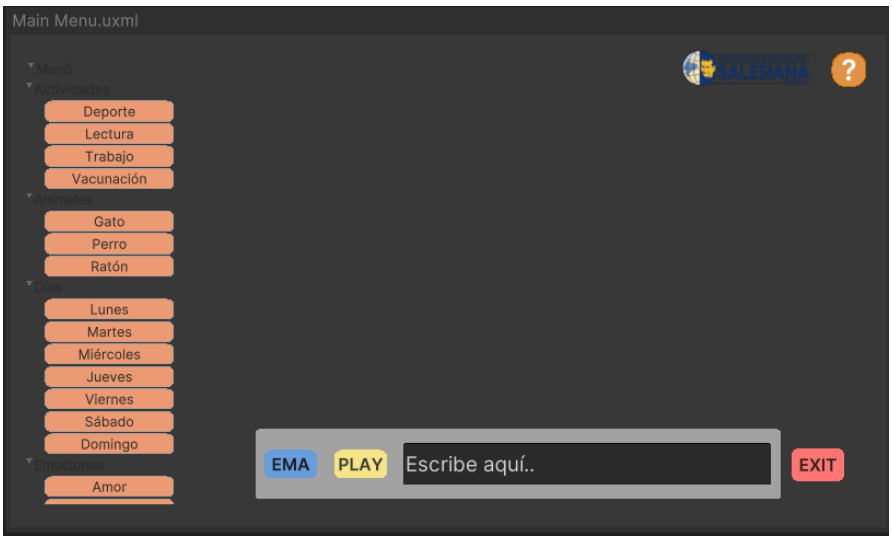


Ilustración 18: diseño de la interfaz gráfica en UI Builder.

2.5.4 Instanciar los botones en el script de C#.

En el código, se deben instanciar las variables de la interfaz asignándoles un tipo: “Button”, “Label” o “TextField”. Después, se llama al método OnEnable() que indica lo que tiene que hacer la interfaz apenas empieza el programa. Ahí se enlazan los botones, etiquetas y campos de texto creados con los nombres asignados en UI Builder.

2.5.5 Utilización de los elementos visuales en el script de C#.

Para llamar a un botón, se debe crear un método de tipo *void* que establecerá lo que hace el programa cuando ese botón sea seleccionado.

2.6 Algoritmo de reconocimiento del lenguaje.

El personaje virtual debe ser capaz de interpretar señas o frases escritas, para ello, se hizo un análisis del texto introducido creando una concordancia con las señas de la base de datos de animaciones. Se estableció que, si una palabra no se encuentra en el diccionario, es decir, que no tiene una seña asignada, esta palabra deberá ser deletreada. Sin embargo, hay que tomar en cuenta que, dentro de la lengua de señas, una misma seña es utilizada para una palabra y sus configuraciones, por ejemplo: deporte, deportista y deportivo utilizan la misma seña, simplemente cambian por el contexto de la oración.

2.6.1 Almacenamiento de la barra de búsqueda.

Las palabras escritas en el campo de texto almacenado en la UI se guardan en una variable de tipo *string* cuando el botón “play” es seleccionado. Posteriormente se convierte a todas las palabras en minúsculas y se las guarda en un vector donde cada palabra ocupa una posición. Luego, se reemplazan todas las letras con acentos a letras simples.

2.6.2 Eliminar selectivamente las palabras que no pertenecen al LSE.

Se creó un vector de tipo *string* que contiene las palabras que no tienen una traducción textual a la lengua de señas como por ejemplo el verbo ser o estar, palabras complementarias como “de”, entre otras. El vector de *strings* se convierte

en una lista llamada “values”, en la cual se busca si contiene alguna de las palabras que deben ser eliminadas, de ser así, las remueve de la lista.

2.6.3 Determinar si una palabra es similar y/o raíz de otra.

Un vector llamado diccionario contiene los nombres de todas las señas que tengan una animación en la base de datos. Se comparan una a una las palabras del diccionario con las de la lista a través de un algoritmo. Se contrastó la eficacia de 3 algoritmos: Damerau-Levenshtein, coeficiente de superposición y Jaro-Winkler; para determinar el mejor para la aplicación.

El algoritmo de distancia entre palabras Damerau-Levenshtein devuelve un valor de 0 si los *strings* son exactamente la misma palabra y va subiendo su valor dependiendo de qué tan diferentes sean. Entonces, se estableció un valor de umbral de 4 con el cual se determina si la palabra es una variación de las palabras del diccionario o si no pertenece al diccionario y debe ser deletreada. Para mejorar la precisión del algoritmo en palabras cortas, se convierten todas las palabras de la lista “values” en vectores de tipo *char*, de esta forma se mide la longitud de la palabra. Entonces, el valor del umbral disminuirá a 2 si la palabra tiene 5 letras o menos.

Al utilizar el coeficiente de superposición, se obtuvieron resultados erróneos, ya que, bastaba que uno de los *strings* contuviera una de las letras del otro para que exista una similitud y el programa no podía discernir entre todas las palabras que contenían dicha letra.

Por otra parte, el algoritmo Jaro-Winkler devuelve un valor de tipo *double* donde mientras más cercano a 1 sea, más similar es la palabra ingresada a una del diccionario. Todos los valores se guardan en un vector y luego el más alto es seleccionado, si el valor más alto es igual o mayor a un umbral de 0.4, se obtiene el índice donde se encuentra ese valor en el diccionario y se representa la seña correspondiente. Caso contrario, la palabra tiene que ser deletreada.

2.6.4 Deletreo de palabras.

Si el resultado del algoritmo Damerau-Levenshtein es superior a 4, se determina que la palabra debe ser deletreada. Para deletrear las palabras, éstas son convertidas a un vector de tipo *char* con lo que se obtiene cada una de las letras que la componen. Cada letra se convierte nuevamente a dato de tipo *string* y es enviada al comando para reproducir la animación:

```
ema.GetComponent<Animator>().SetBool(letra, true);
```

2.7 Representación de la lengua de señas con el avatar virtual.

Con la lista de las palabras relevantes y sabiendo si se tiene la seña correspondiente o si se debe deletrear la palabra, se puede llamar a las animaciones necesarias para que el avatar las interprete. Sin embargo, primero deben inicializarse y agregarse al personaje. Además, se implementó una etiqueta en la interfaz gráfica que va mostrando en texto la seña exacta que el personaje está realizando.

2.7.1 Árbol de animaciones

Unity posee un componente denominado Animator, el cual es una máquina de estados que permite hacer la transición entre animaciones utilizando un tiempo de salida o ciertas variables (de tipo *bool, float, int* y *trigger*) para accionar cada animación. Principalmente se compone de un bloque de entrada, uno de "cualquier estado" y otro de salida. Los clips de animaciones generados en un inicio deben ser arrastrados y posicionados en esta ventana. La primera animación colocada se convertirá en la animación por defecto que tendrá el objeto cuando el programa inicie, en este caso, es la de descanso del personaje. Esta se conecta con el bloque de "cualquier estado", a partir del cual, se hace la transición a todas las demás animaciones a través de una variable booleana para cada una. Cada animación también se conecta con la animación de descanso para que luego de que termine cada una, puedan volver a esa posición.

2.7.2 Botones del menú de animaciones

El menú de botones de la interfaz gráfica se dividió en categorías de acuerdo a la clasificación de las señas. Al seleccionar un botón, se da un valor verdadero a la variable booleana que lleva a la animación correspondiente y se colocan todas las demás en falso para evitar el solapamiento de animaciones.

2.7.3 Representación de las palabras de la barra de búsqueda en señas.

Con la palabra o letra encontrada en el diccionario a través del algoritmo Damerau-Levenshtein, se envía un valor verdadero a la variable booleana correspondiente a la seña, se ponen en falso las demás y se espera el tiempo que dura la animación antes de pasar a la siguiente, una vez terminada la secuencia, se muestra la animación de descanso.

2.7.4 Presentación del Avatar Virtual

El avatar cuenta con un botón de presentación, en el cual se reproduce una secuencia de señas (“Hola, me llamo Ema y soy una intérprete de lengua de señas”), con el cual se logra una mejor vinculación con el usuario.

2.8 Exportar el programa.

Con las animaciones, el código y la interfaz gráfica terminados; se procedió a crear un ejecutable para poder utilizarlo en cualquier computadora. Se corrigieron el tamaño de la letra y botones de la interfaz gráfica para adaptarla a una pantalla de resolución Full HD (1920 x 1080) y se exportó el programa de la siguiente manera:

1. En *File > Build Settings* se deben activar todas las escenas a exportarse.
2. Escoger la palataforma de destino, en este caso “PC, Mac & Linux Standalone”, y la arquitectura (x86_64).
3. Presionar el botón “Build” y escoger la carpeta donde se guardarán los archivos de ejecución. Cuando haya terminado de exportar, se podrá ejecutar el programa desde el íncono de la aplicación.

CAPÍTULO 2. MARCO METODOLÓGICO

CAPÍTULO 3

3. IMPLEMENTACIÓN Y ANÁLISIS DE RESULTADOS

Al finalizar todos los procesos descritos en el capítulo anterior, se comenzaron a hacer las pruebas pertinentes para establecer un indicador de la eficiencia del traductor. Se establecieron tres categorías: interacción del usuario con la interfaz gráfica, cuán comprensibles son las señas y precisión de la traducción.

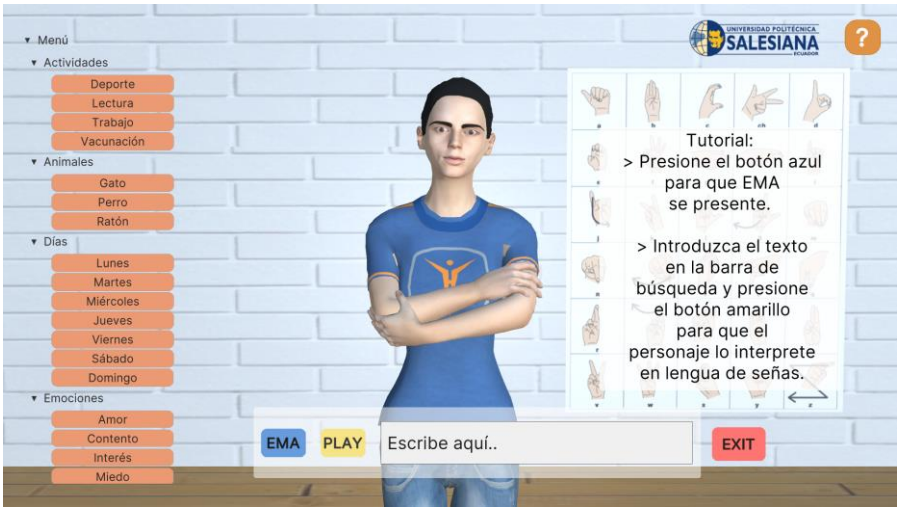


Ilustración 19: programa terminado.

3.1 Interacción del usuario con la interfaz gráfica.

Para analizar la usabilidad del programa, se efectuó una encuesta a dos grupos de usuarios, el primero grupo de 8 personas (4 hombres y 4 mujeres de entre 24 y 65 años de edad) y, el segundo grupo de 4 personas (3 hombres y 1 mujer de entre 26 y 71 años de edad). La encuesta contiene las siguientes preguntas:

1. ¿La interfaz fue fácil de utilizar? (Si/No)
2. ¿Fue interactiva la comunicación con el programa? (Si/No)
3. ¿La información fue clara? (Si/No)

4. ¿Utilizaría el programa en el futuro? (Si/No)
5. ¿Qué cambiaría, agregaría o quitaría al programa? (Respuesta corta)
6. ¿Recomendaría el programa? (Si/No)

En el primer grupo de encuestados se obtuvieron los siguientes resultados:

Tabla 1: respuestas del primer grupo de encuestados.

Preguntas	**Si**	**No**
¿La interfaz fue fácil de utilizar?	8	0
¿Fue interactiva la comunicación con el programa?	7	1
¿La información fue clara?	7	1
¿Utilizaría el programa en el futuro?	8	0
¿Recomendaría el programa?	8	0
¿Qué cambiaría, agregaría o quitaría al programa?	Más señas, agregar reconocimiento de voz, agregar un tutorial y mejorar la distribución de los botones.	

De estos resultados se puede determinar que la interfaz y la información que contiene no fue suficientemente clara para que personas mayores lo pudieran utilizar. Entonces, se cambiaron los botones para que tengan una distribución más familiar con Windows (botón para cerrar el programa en la esquina superior derecha) y otros sistemas como los de Facebook, Whatsapp y el buscador de Google que poseen el

botón de acción a la derecha de la barra de texto o búsqueda y, que, a su vez, permiten realizar la acción presionando la tecla “Enter” del teclado. Todos estos cambios se pueden apreciar en la Ilustración 22Ilustración 22. También, se agregó una ventana de inicio en donde se explica brevemente lo que hace el programa y se puede escoger comenzar con el tutorial o continuar con el programa directamente (Ilustración 20). Al elegir el tutorial, se muestra la interfaz, pero con etiquetas explicativas junto a cada botón y en cada parte importante de la interfaz (Ilustración 21).

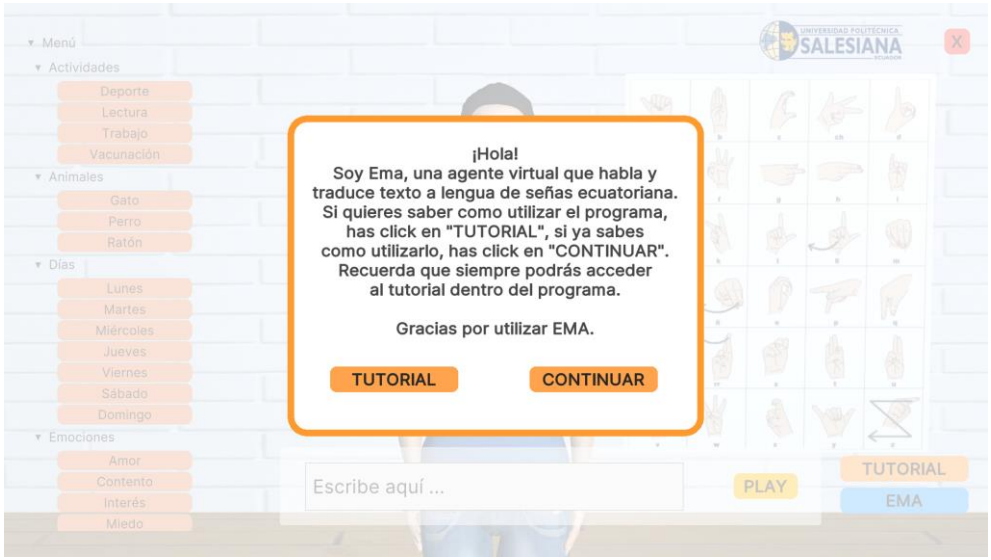


Ilustración 20: ventana de inicio de la interfaz.

CAPÍTULO 4. CONCLUSIONES Y RECOMENDACIONES

Ilustración 21: tutorial del programa.

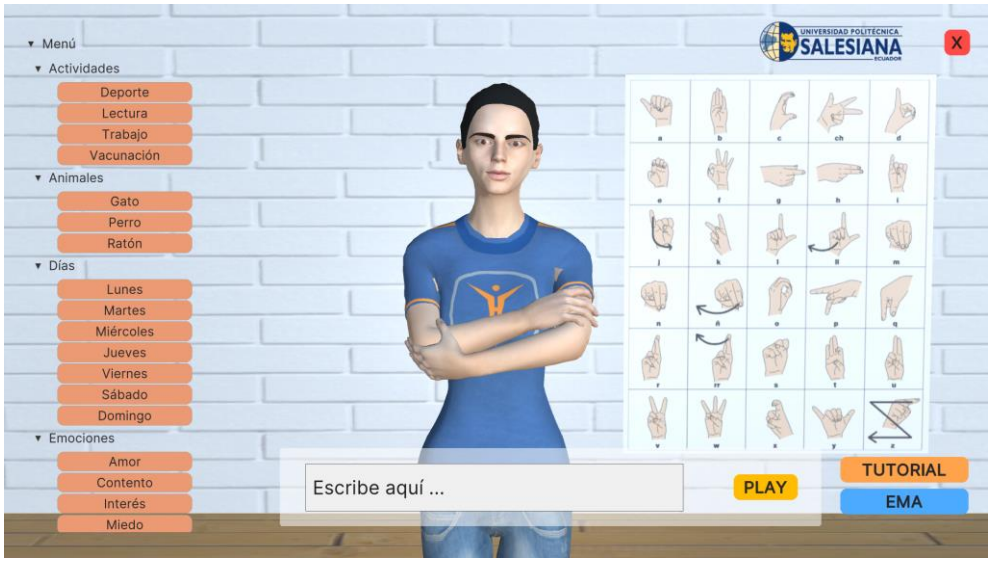


Ilustración 22: interfaz con la distribución de botones mejorada.

Con las mejoras aplicadas, se presentó el programa y se encuestó al segundo grupo de usuarios, obteniendo los siguientes resultados:

Tabla 2: resultados de las encuestas del segundo grupo.

Preguntas	**Si**	**No**
¿La interfaz fue fácil de utilizar?	4	0
¿Fue interactiva la comunicación con el programa?	4	0
¿La información fue clara?	4	0
¿Utilizaría el programa en el futuro?	4	0
¿Recomendaría el programa?	4	0
¿Qué cambiaría, agregaría o quitaría al programa?	Más señas y agregar reconocimiento de voz.	

De la Tabla 2 se puede deducir que la interfaz resultó más fácil de utilizar y el propósito del programa es más claro. Sin embargo, ambos grupos desearían que el programa contara con un diccionario más amplio y con reconocimiento de voz, dando inicio al trabajo a futuro que requiere el programa. La información exacta de las encuestas se encuentra en el APÉNDICE A.

3.2 Comprensibilidad de las señas.

Para determinar si las señas realizadas por el personaje resultan comprensibles para los sordos, se mostraron todas las señas del programa a una persona no oyente que conoce la lengua de señas ecuatoriana. De las 133 señas que componen el diccionario, 4 no resultaron lo suficientemente comprensibles. Esas señas fueron: ¿Por qué?, primo, tío y WiFi. Las tres primeras debido a un desfase entre las manos y/o el cuerpo del personaje debido a una diferencia de cálculo de la interpolación de cuadros entre Blender y Unity, sin embargo, la persona fue capaz de inferir su significado. La cuarta seña contenía posiciones de los dedos similares a la seña de “cama”, con la cual difiere por el movimiento y posición de las manos, entonces el usuario la relacionó con esa seña primero, además, la seña de WiFi es relativamente nueva.

Se puede concluir entonces que el 96,99 % de las animaciones de las señas del diccionario son completamente comprensibles.

3.3 Precisión de la traducción.

Para determinar la precisión de los algoritmos Damerau-Levenshtein y Jaro-Winkler y, comparar cual es el mejor, se introdujeron 135 de las 500 palabras más usadas en español según la Real Academia de la Lengua (RAE) [48]. Se descartaron los artículos, pronombres y variaciones de palabras que ya habían sido seleccionadas. Durante el experimento se observó el comportamiento del sistema a través del personaje, es decir, si al introducir la palabra, el personaje la deletreaba o hacía una seña. Sin embargo, hubo palabras que debían ser deletreadas pero el personaje hizo una seña que no correspondía y en otras ocasiones se deletrearon palabras que si tenían una seña en el diccionario. En base a este criterio se realizaron las matrices de confusión correspondientes a cada algoritmo. La lista de palabras y más información

sobre cómo fueron obtenidas las matrices se encontrarán en el APÉNDICE B. A continuación, se muestra un resumen de los resultados:

Tabla 3: resumen de la comparación entre los algoritmos Damerau-Levenshtein y Jaro-Winkler

	Damerau-Levenshtein	**Jaro-Winkler**
Señas representadas en señas	15	17
Señas deletreadas	4	2
Deletreos confundidos con señas	50	4
Deletreos deletreados	66	112
Total de aciertos	81 de 135 = 60%	129 de 135 = 95,6%

Los resultados de la Tabla 3 se pueden traducir en mejoras visibles como:

- Menor confusión en palabras cortas, ya que, con el algoritmo Damerau-Levenshtein, casi cualquier palabra de cuatro letras o menos como “cola”, “lola”, “ola” y “casa” era representada con la seña de “hola” que se encontraba en el diccionario.
- También, con Damerau-Levenshtein, nombres como “Karen”, “Fernando”, “Camila” y “Victoria” eran confundidos con días de la semana y otras palabras del diccionario. Pero utilizando el algoritmo Jaro-Winkler, las palabras cortas ahora se reconocen efectivamente y los nombres son deletreados en lugar de ser confundidos con una seña.

3.3.1 Algoritmo Damerau-Levenshtein

En principio se utilizó el algoritmo Damerau-Levenshtein, la siguiente matriz de confusión:

Tabla 4: matriz de confusión Damerau-Levenshtein.

MATRIZ DE CONFUSIÓN DL		
	seña	**deletreo**
seña	15	50
deletreo	4	66
Total	19	116
	135	

De la matriz de confusión en la Tabla 4 se puede extraer que 19 de las palabras introducidas tenían una seña correspondiente, pero 4 fueron deletreadas. De las 116 palabras que no tenían una seña, solo 66 fueron deletreadas y 50 de ellas fueron confundidas con una seña que no correspondía a su significado. En rasgos generales se podría decir que tuvo un 60% de aciertos.

3.3.2 Algoritmo Jaro-Winkler

Finalmente, se optó por emplear el algoritmo Jaro-Winkler, en el cual se produjo el mismo experimento que en el algoritmo Damerau-Levenshtein. Sin embargo, los resultados fueron significativamente mejorados, como se puede observar en la siguiente matriz de confusión:

Tabla 5: matriz de confusión Jaro-Winkler

MATRIZ DE CONFUSIÓN JW		
	seña	**deletreo**
seña	17	4
deletreo	2	112
Total	19	116
	135	

De la matriz de confusión de la Tabla 5 se puede extraer que de las 19 palabras que tenían seña, 17 se reconocieron como las señas correspondientes. Por

otra parte, de las 116 palabras que debían ser deletreadas, solo 4 fueron confundidas con alguna seña, las otras 112 fueron deletreadas exitosamente dando un total de 95,6% de aciertos.

CAPÍTULO 4. CONCLUSIONES Y RECOMENDACIONES

CAPÍTULO 4

4. Conclusiones, Recomendaciones y Trabajo a Futuro

4.1 Conclusiones

Esta tesis tuvo como objetivo diseñar un sistema prototipo que permita traducir comandos de texto a lengua de señas ecuatoriana utilizando un avatar virtual. Estos sistemas de traducción han tenido muy buena acogida, ya que, son empleados en diversos países e idiomas [3]–[9], [12]–[15], [17], [18], [18]–[20] con su LS correspondiente. Además, los artículos científicos revisados muestran varios métodos y algoritmos que permiten: realizar el reconocimiento de texto mediante un NLP profundo y, crear el software adecuado para que el personaje virtual en 3D represente la señas. En ese contexto, para el desarrollo de este trabajo de titulación se establecieron cuatro objetivos específicos que serán respondidos a continuación.

En el primer objetivo específico se planteó la creación de una base de datos de las animaciones de las señas más comunes de la LSE. Para lo cual, se aprovecharon los beneficios de *rigging* y *skinning* en la elaboración del esqueleto del personaje y, el proceso de *key-framing* para la animación de las señas en el programa de código abierto Blender. Además, se siguieron los principios de animación 4, 6, 7, 8, 9 y 11 detallados en la fundamentación teórica. Una vez realizadas todas las poses y animaciones secuenciales, se obtuvo una base de datos lista para su exportación con 133 animaciones de señas de letras y palabras, una seña de transición y una posición de descanso del avatar.

El segundo objetivo planteado comprende el desarrollo de un algoritmo de búsqueda y comparación entre el texto introducido por el usuario y la base de datos, tomando en cuenta que no solo debían encontrarse palabras equivalentes sino

también, palabras derivadas de otras. Por lo tanto, se emplearon, en un principio, tres algoritmos de superposición, similitud y distancia entre *strings*: Szymkiewicz-Simpson, Jaro-Winkler y Damerau-Levenshtein, respectivamente. El primero de ellos, fue descartado enseguida dado que, buscaba caracteres iguales entre la palabra introducida con las de la base de datos; entonces se consideraba "superpuesta" una palabra con otra aún si solo compartían una letra en común. Por otra parte, los algoritmos Damerau-Levenshtein y Jaro-Winkler proporcionaron un reconocimiento de texto más acertado, por lo que, luego fueron comparados para determinar el que mejor encaja con la aplicación. De esta manera, el software puede reconocer qué letra, palabra u oración introdujo el usuario y, mostrar la seña correspondiente o deletrear la palabra si no se encuentra en la base de datos.

Para lograr el tercer objetivo, el cual requería diseñar un intérprete virtual en 3D, se utilizó el programa de código abierto Makehuman. En este programa, a través de parámetros, se moldeó el personaje Ema con características femeninas caucásicas, 1,60 m de altura y apariencia juvenil. Con ello, se obtuvo a la representante del software, a quien se le asignaron las señas luego de ser exportada a Blender, para que posteriormente las ejecute en Unity.

Finalmente, se programaron rutinas que generen las animaciones correspondientes a cada seña según el texto introducido, cumpliendo así, el cuarto objetivo. Las rutinas fueron escritas en el lenguaje de programación C# en el software de código abierto Unity 3D mediante el compilador Visual Studio Community. La primera rutina permite enviar al árbol de animaciones un valor booleano que debe llamar a la animación que pertenece a la palabra reconocida previamente y, desactivar todas las demás durante el tiempo que dure la seña activa, evitando el solapamiento. La segunda genera una animación de transición que sirve como indicador de que una palabra ha terminado de deletrearse y, que viene la siguiente (si es que también se deletrea). De este modo, el personaje puede realizar gran cantidad de señas secuenciales sin que se superpongan entre sí y, es capaz de separar las palabras deletreadas cuando es necesario.

Por otro lado, en relación con los resultados obtenidos durante este proyecto se concluyen tres categorías que demuestran la validez del traductor de texto a LSE. La primera categoría hace referencia a la interacción del usuario con la GUI, para su evaluación se realizaron encuestas a dos grupos de usuarios, revelando que, en general, la interfaz es fácil de utilizar y la comunicación con ella es interactiva. Sin embargo, para ciertos adultos mayores que probaron el software, les pareció que la información proporcionada sobre su propósito y manejo no fue clara. Por ello, se agregó una ventana de inicio que explica la finalidad del programa y el acceso a un tutorial. Entre otras sugerencias, se encontró que las personas mayores preferían una interfaz con una distribución de botones más familiar, entonces se aplicó esta modificación para que se parezca a otras interfaces conocidas de Windows, Google y Whatsapp. Es importante mencionar que, a pesar de estos inconvenientes, la reacción del público es muy positiva y todos afirmaron que utilizarían el programa en el futuro y lo recomendarían.

Como segunda categoría, se valoró la comprensibilidad de las señas realizadas por Ema por medio de la asistencia y evaluación de una persona sorda conocedora de LSE. Mediante este análisis externo fue posible comprobar que el 96,99% (129 de 133 animaciones) de las señas incluidas en la base de datos del programa representadas por el personaje fueron completamente entendibles. Tres de las cuatro restantes sufrieron un desfase en la posición debido a la interpolación automática de Unity. La última, en cambio, era una seña relativamente nueva. Sin embargo, se puede concluir que es posible crear animaciones precisas a partir de los videos explicativos del diccionario “Gabriel Román” y, que estás serán comprendidas por las personas sordas que sepan LSE.

La última categoría evalúa la precisión de la traducción de texto a lengua de señas. En este apartado se compraron los dos algoritmos empleados en el reconocimiento de texto (Jaro-Winkler y Damerau-Levenshtein) mediante un experimento en el cual se introdujeron 135 de las 500 palabras más comunes del español a cada uno, para determinar cual reconocía mejor si la palabra pertenece o

se deriva de alguna palabra del diccionario o, si la palabra no existe en la base de datos y debía deletrearse. En función de los resultados se crearon matrices de confusión que revelaron que el algoritmo Damerau-Levenshtein confundió 50 de las 116 palabras que debían ser deletreadas con alguna palabra que si tiene seña en el diccionario y, 4 de las 19 palabras que correspondían a una seña las deletreó, obteniendo un acierto del 60%. Por ende, el personaje hacía una seña errónea en lugar de deletrear o, deletreaba palabras que si tenían seña. Por otra parte, el algoritmo Jaro-Winkler confundió solo 4 de las 116 palabras con señas y, 2 de las 19 señas con deletreos; generando un 95,6% de precisión, es decir, 30% más que el anterior. Esto se traduce en que ahora el programa es capaz de deletrear correctamente nombres sin confundirlos con alguna seña, que puede distinguir palabras con mínimas diferencias entre caracteres y, aun así, saber si una palabra se deriva de otra.

Se puede concluir entonces que el programa posee una interfaz gráfica utilizable por la mayoría de los usuarios y un sistema de reconocimiento de texto eficiente para una traducción simple. Además, asegura que un sordo o sorda que conozca LSE va a poder entender las señas realizadas por el personaje. De esta manera culmina este trabajo de titulación, entregando una herramienta útil para la comunidad, a favor de la inclusión de sordos e hipoacúsicos en el Ecuador.

4.2 Recomendaciones

Al momento de diseñar un personaje en tres dimensiones, se deben tomar en cuenta las medidas del personaje (altura y peso), ya que, los algoritmos de interpolación, gravedad, colisión, entre otros; de Blender y Unity utilizan estos datos. También, durante la asignación del esqueleto al personaje, verificar que los ejes cartesianos de cada uno de los huesos estén en la dirección correcta, de no estarlo pueden provocar desfases en el cálculo de la interpolación y, al realizar las animaciones en secuencia, comprobar que las poses siguientes no dañen las

anteriores durante la interpolación. Seguidamente, para animar al personaje, se deben activar la escala, rotación y movimiento en todos los ejes, caso contrario, el personaje se deforma con cada pose. Igualmente, verificar la configuración de cuadros por segundo del programa de origen de las animaciones (en este caso Blender), porque deben coincidir con los del programa objetivo (Unity).

Además, se deben realizar todas las configuraciones y adición de paquetes en Unity antes de importar el personaje, ya que, estas variaciones podrían corromper el archivo. Una vez importado el personaje con las animaciones en Unity, hay que corroborar que las animaciones se ejecuten correctamente y que se han exportado como acciones y no como poses. Del mismo modo, dentro de las preferencias del programa en Unity, se tiene que configurar por defecto el compilador utilizado para poder extraer todo el potencial. De igual importancia es agregar todas las extensiones de código necesarias dentro del *script* para que funcionen adecuadamente los comandos. Así mismo, al diseñar la interfaz gráfica en Unity, se debe colocar las dimensiones del dispositivo objetivo en la pantalla de visualización y realizar modificaciones utilizando el visor de juego en lugar de Unity Builder para mejor precisión en la ubicación y visualización de los elementos de la interfaz. Finalmente, hay que asegurar que todas las escenas sean visibles y estén seleccionadas antes de crear el ejecutable y, verificar el sistema operativo y dispositivos que lo utilizarán.

4.3 Trabajo a futuro

A través de este proyecto se brindará a la comunidad una herramienta que puede ser de interés para diversas empresas que busquen una ventaja competitiva y dar servicios a un campo más amplio de la población. También, se genera la base para distintas aplicaciones dirigidas al sector público y comercial a favor del cambio y la actualización de la forma de atender a clientes sordos.

Además, abre paso a nuevos programas de investigación dentro de la traducción e interpretación de la lengua de señas, a partir de los cuales se pueden desarrollar artículos académicos y un aporte a la comunidad científica en general.

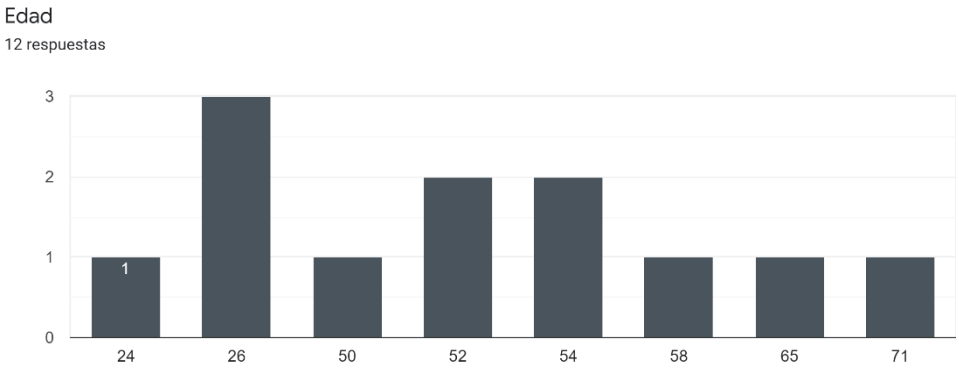
En el futuro a este programa se le agregará reconocimiento de voz y un mejor procesamiento del lenguaje que incluya todas las características de la lengua de señas ecuatoriana para la interpretación realizada por el personaje. Se añadirán gestos faciales a Ema para que la comunicación sea más acertada e interactiva. Se espera aumentar significativamente el diccionario y la base de datos de animaciones de señas ecuatorianas y llegar a implementar este programa en dispositivos móviles.

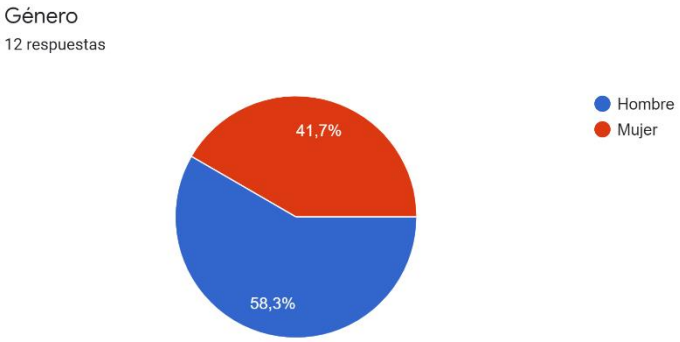
Con el tiempo, esta nueva herramienta podría llevar al Ecuador al siguiente nivel de inclusión. Cualquier persona podría aprender lengua de señas fácilmente y a la vez, comunicarse con un sordo.

APÉNDICES

APÉNDICE A

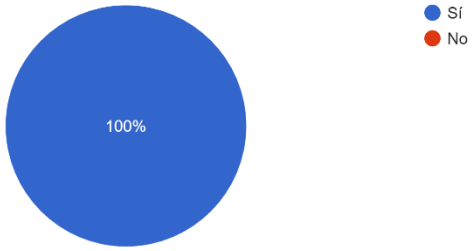
Respuestas de la encuesta “Experiencia de usuario con el traductor de texto a lengua de señas ecuatoriana EMA” realizado a 12 personas.



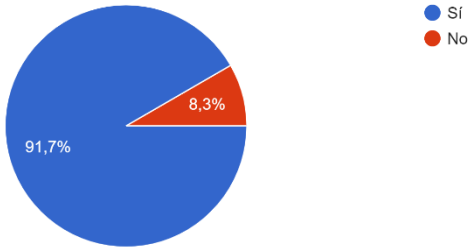


APÉNDICES

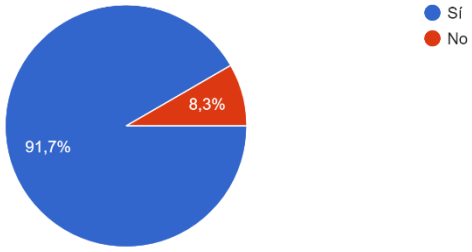
¿La interfaz fue fácil de utilizar?
12 respuestas



¿Fue interactiva la comunicación con el programa?
12 respuestas

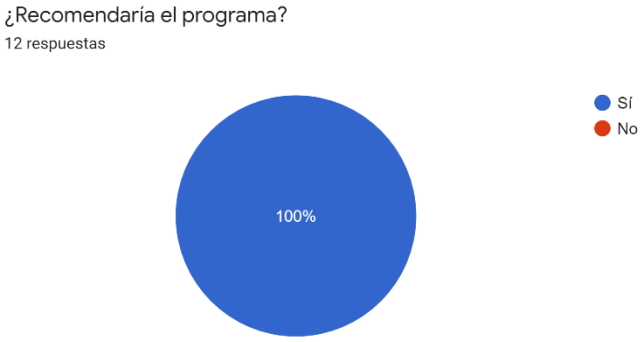


¿La información fue clara?
12 respuestas



APÉNDICES


¿Utilizaría el programa en el futuro?
12 respuestas
100%
Sí
No


¿Recomendaría el programa?
12 respuestas
100%
Sí
No

APÉNDICE B

Lista de palabras introducidas en el software para comprar la efectividad entre los algoritmos Damerau-Levenshtein y Jaro-Winkler:

Input
cola
gana
café
arco
buzo
cero
vino
pato
mote
boa
nieto
universitario
deportista
deportivo
ábaco
babel
bahía
cable
cabra
ver
veces
partido
personas
grupo
cuenta
pueden
tienen

misma
cual
fueron
mujer
frente
José
tras
cosas
fin
ciudad
social
manera
tener
sistema
historia
muchos
Juan
tipo
dentro
decir
punto
noche
agua
fuera
situación
bajo
ejemplo
países

país
horas
tarde
ley
guerra
realidad
lado
mano
número
sociedad
centro
padre
gente
cuerpo
obra
madre
Carlos
Hombres
Ojos
morir
nombre
público
mañana
derecho
verdad
María
Cabeza
tierra

director
familia
seguridad
semana
señor
luz
claro
español
palabras
internacional
empresa
libro
político
Dios
México
amor
música
hablar
hijo
servicio
niños
dinero
dirección
papel
Barcelona
diferentes
capital
Europa

libertad
espacio
población
salud
estudio
cultura
arte
paz
interés
julio
edad
actividad
joven
futuro
mes
nuevos
educación
González
aire
investigación
central
comunidad
necesidad
quien

APÉNDICES

Respuesta del algoritmo Damerau-Levenshtein

Output	Predicted	Expected	Seña	Deletreo	Señas incorrectas	Deletreo incorrecto	¿Funcionó?
hola	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
gato	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	seña	not seña	not deletreo	in seña	not deletreo	no
deletreo	deletreo	seña	not seña	not deletreo	in seña	not deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
primo	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
buenas	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
jueves	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
viernes	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
enero	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
mujer	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
cliente	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
enero	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
triste	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
hora	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
buenos	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
tio	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
enero	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si

APÉNDICES

deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
noches	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
dirección	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
tiempo	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
martes	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
hora	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
tardes	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
cuadra	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
alado	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
mayo	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
buenos	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
enero	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	seña	not seña	not deletreo	in seña	not deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
enero	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	seña	not seña	not deletreo	in seña	not deletreo	no
martes	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
hombre	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
nombre	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
julio	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
mamá	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
derecha	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
viernes	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
sabado	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
perro	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
dirección	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
familia	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
hermana	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
luz	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
ecuador	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
martes	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
poco	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
adios	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
medidor	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no

APÉNDICES

amor	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
mi	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
hola	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
servicio	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
enero	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
dirección	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
viernes	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
hospital	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
enero	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
poco	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
junio	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
cuadra	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
interés	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
julio	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
enero	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
jueves	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
vacunación	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
contrato	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
necesitar	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
quien	seña	seña	ok seña	not deletreo	not seña	not deletreo	si

Predicciones hechas por clases:

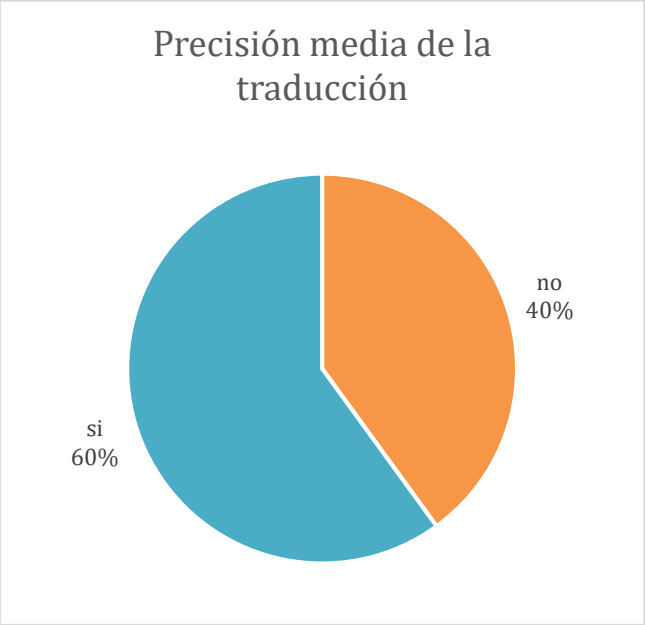
	Predicciones correctas por clase	**Predicciones incorrectas por clase**
Seña	15	4
Deletreo	66	50

Precisión media de la traducción:

No	54	40%
Si	81	60%

APÉNDICES

Total	135	100%



Matriz de confusión

MATRIZ DE CONFUSIÓN DL		
	seña	**deletreo**
seña	15	50
deletreo	4	66
Total	19	116
	135	

Respuesta del algoritmo Jaro-Winkler

Output	Predicted	Expected	Seña	Deletreo	Señas incorrectas	Deletreo incorrecto	¿Funcionó?
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
universidad	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deporte	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
depore	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
mujer	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si

deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
noches	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
hora	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
tardes	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	seña	not seña	not deletreo	in seña	not deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	seña	not seña	not deletreo	in seña	not deletreo	no

deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
hombre	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
nombre	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
derecha	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
direccion	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
luz	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
interes	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
amor	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
servicio	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
direccion	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si

deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
estudiante	seña	deletreo	not seña	not deletreo	not seña	in deletreo	no
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
interes	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
julio	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
deletreo	deletreo	deletreo	not seña	ok deletreo	not seña	not deletreo	si
necesitar	seña	seña	ok seña	not deletreo	not seña	not deletreo	si
quien	seña	seña	ok seña	not deletreo	not seña	not deletreo	si

Predicciones hechas por clase:

	Predicciones correctas por clase	**Predicciones incorrectas por clase**
Seña	17	2
Deletreo	112	4

Precisión media de la traducción:

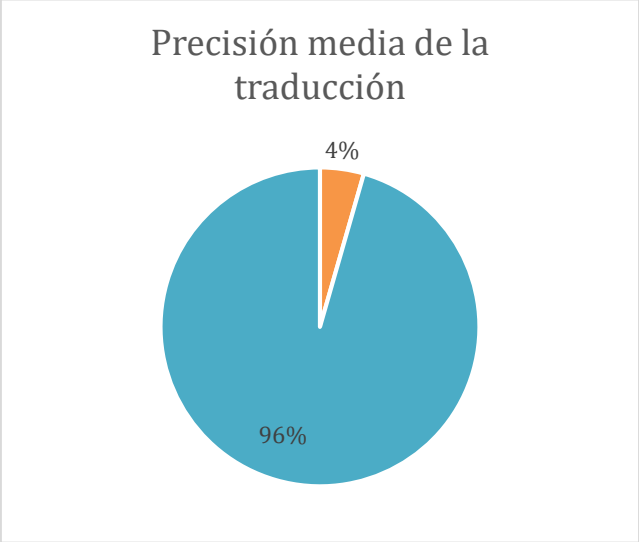
No	6	4,4%

APÉNDICES

Si	129	95,6%
Total	135	100%

Matriz de confusión:

MATRIZ DE CONFUSIÓN JW		
	seña	**deletreo**
seña	17	4
deletreo	2	112
Total	19	116
	135	



REFERENCIAS BIBLIOGRÁFICAS

[1] V. Heredia, "El lenguaje de señas, vital para la inclusión de sordos.", *El Comercio*, sep. 2019. https://www.elcomercio.com/actualidad/lenguaje-senas-inclusion-sordos-docentes.html#:~:text=Son%20utilizadas%20por%2072%20millones,en%20Ecuador%20hay%2066%20111.

[2] CONADIS, "Diccionario de Lengua de Señas Ecuatoriano 'Gabriel Román'", *Consejo Nacional para la Igualdad de Discapacidades*. https://www.consejodiscapacidades.gob.ec/diccionario-de-lengua-de-senas-ecuatoriano-gabriel-roman/

[3] M. Ahmed, M. Idrees, Z. ul Abideen, R. Mumtaz, y S. Khalique, "Deaf talk using 3D animated sign language: A sign language interpreter using Microsoft's kinect v2", en *2016 SAI Computing Conference (SAI)*, London, United Kingdom, jul. 2016, pp. 330–335. doi: 10.1109/SAI.2016.7556002.

[4] T. Oliveira, N. Escudeiro, P. Escudeiro, E. Rocha, y F. M. Barbosa, "The VirtualSign Channel for the Communication Between Deaf and Hearing Users", *IEEE Rev. Iberoam. Tecnol. Aprendiz.*, vol. 14, núm. 4, pp. 188–195, nov. 2019, doi: 10.1109/RITA.2019.2952270.

[5] K. Ayadi, Y. O. M. Elhadj, y A. Ferchichi, "Prototype for Learning and Teaching Arabic Sign Language Using 3D Animations", en *2018 International Conference on Intelligent Autonomous Systems (ICoIAS)*, Singapore, mar. 2018, pp. 51–57. doi: 10.1109/ICoIAS.2018.8493979.

[6] M. Varghese y S. K. Nambiar, "English To SiGML Conversion For Sign Language Generation", en *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*, Kottayam, India, dic. 2018, pp. 1–6. doi: 10.1109/ICCSDET.2018.8821212.

[7] G. P. G. Ramos y M. Elisabeth, "Translator System for Peruvian Sign Language Texts through 3D Virtual Assistant", *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, núm. 1, 2020, doi: 10.14569/IJACSA.2020.0110168.

REFERENCIAS BIBLIOGRÁFICAS

[8] M. Brour y A. Benabbou, “ATLASLang NMT: Arabic text language into Arabic sign language neural machine translation”, *J. King Saud Univ. - Comput. Inf. Sci.*, p. S1319157819303532, jul. 2019, doi: 10.1016/j.jksuci.2019.07.006.

[9] A. Karpov, I. Kipyatkova, y M. Zelezny, “Automatic Technologies for Processing Spoken Sign Languages”, *Procedia Comput. Sci.*, vol. 81, pp. 201–207, 2016, doi: 10.1016/j.procs.2016.04.050.

[10] “Visemas y Amazon Polly - Amazon Polly”. https://docs.aws.amazon.com/es_es/polly/latest/dg/viseme.html (consultado sep. 20, 2021).

[11] Sugandhi, P. Kumar, y S. Kaur, “Sign Language Generation System Based on Indian Sign Language Grammar”, *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 19, núm. 4, pp. 1–26, jul. 2020, doi: 10.1145/3384202.

[12] L. Zhao, K. Kipper, W. Schuler, C. Vogler, N. Badler, y M. Palmer, “A Machine Translation System from English to American Sign Language”, en *Envisioning Machine Translation in the Information Future*, Berlin, Heidelberg, 2000, pp. 54–67. doi: 10.1007/3-540-39965-8_6.

[13] M. Reddy, “INGIT: Limited Domain Formulaic Translation from Hindi Strings to Indian Sign Language”, *ICON*, Consultado: nov. 03, 2020. [En línea]. Disponible en: https://www.academia.edu/33373439/INGIT_Limited_Domain_Formulaic_Translation_from_Hindi_Strings_to_Indian_Sign_Language

[14] S. M. Halawani, D. Daman, S. Kari, y A. R. Ahmad, “An Avatar Based Translation System from Arabic Speech to Arabic Sign Language for Deaf People”, p. 10, 2013.

[15] J. Porta, F. López-Colino, J. Tejedor, y J. Colás, “A rule-based translation from written Spanish to Spanish Sign Language glosses”, *Comput. Speech Lang.*, vol. 28, núm. 3, pp. 788–811, may 2014, doi: 10.1016/j.csl.2013.10.003.

[16] J. Ulisses, T. Oliveira, E. Rocha, P. M. Escudeiro, N. Escudeiro, y F. M. Barbosa, “Blind/Deaf Comunication API for Assisted Translated Educational Digital Content”, en *2018 28th EAEEIE Annual Conference (EAEEIE)*, Hafnarfjordur, sep. 2018, pp. 1–9. doi: 10.1109/EAEEIE.2018.8534217.

[17] M. Grif y Y. Manueva, “Semantic analyses of text to translate to Russian sign language”, en *2016 11th International Forum on Strategic Technology (IFOST)*, Novosibirsk, Russia, jun. 2016, pp. 286–289. doi: 10.1109/IFOST.2016.7884107.

REFERENCIAS BIBLIOGRÁFICAS

[18] T. Veale y A. Conway, “Sign-Language Generation in ZARDOZ: An English to Sign-Language Translation System”, 1994. Consultado: nov. 03, 2020. [En línea]. Disponible en: https://www.aclweb.org/anthology/W94-0333

[19] Y. Bouzid y M. Jemni, “tuniSigner: A Virtual Interpreter to Learn Sign Writing”, en *2014 IEEE 14th International Conference on Advanced Learning Technologies*, Athens, Greece, jul. 2014, pp. 601–605. doi: 10.1109/ICALT.2014.176.

[20] N.-U.-N. Soogund y M. H. Joseph, “SignAR: A Sign Language Translator Application with Augmented Reality using Text and Image Recognition”, en *2019 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS)*, Tamilnadu, India, abr. 2019, pp. 1–5. doi: 10.1109/INCOS45849.2019.8951322.

[21] T. Vichyaloetsiri y P. Wuttidittachotti, “Web Service framework to translate text into sign language”, en *2017 International Conference on Computer, Information and Telecommunication Systems (CITS)*, Dalian, China, jul. 2017, pp. 180–184. doi: 10.1109/CITS.2017.8035336.

[22] J. Garrido y P. Uvillus, “Interpretación bidireccional del alfabeto dactilológico mediante una mano robótica y una aplicación móvil para la interacción con personas sordomudas.”, Tesis, Universidad Politécnica Salesiana, Quito, Ecuador, 2019. [En línea]. Disponible en: http://dspace.ups.edu.ec/handle/123456789/19113

[23] P. Espinosa y H. Pogo, “Diseño y construcción de un guante prototipo electrónico capaz de traducir el lenguaje de señas de una persona sordomuda al lenguaje de letras.”, Tesis, Universidad Politécnica Salesiana, Cuenca, Ecuador, 2013. [En línea]. Disponible en: http://dspace.ups.edu.ec/handle/123456789/4211

[24] D. Duque y M. Ibarra, “Diseño e implementación de un guante electrónico que permite transformar el lenguaje de señas en caracteres y reproducción sonora de voz artificial.”, Tesis, Universidad Politécnica Salesiana, Quito, Ecuador, 2014. [En línea]. Disponible en: http://dspace.ups.edu.ec/handle/123456789/6329

[25] J. León, D. Lozada, C. Lucero, y M. Paredes, “Plataforma de comunicación y sistema de enseñanza asistida por software en 3 dimensiones para niños y adultos con discapacidad auditiva.”, Tesis, Universidad Politécnica Salesiana, Cuenca, Ecuador, 2011. [En línea]. Disponible en: http://dspace.ups.edu.ec/handle/123456789/8991

[26] CONADIS, “Diccionario de Lengua de Señas Ecuatoriana Gabriel Román”, *Plataforma Conadis*. http://www.plataformaconadis.gob.ec/~platafor/diccionario/

REFERENCIAS BIBLIOGRÁFICAS

[27] FENASEC, “Glosario Básico de Lengua de Señas Ecuatoriano”. 2013. [En línea]. Disponible en: https://www.vicepresidencia.gob.ec/wp-content/uploads/downloads/2013/04/Compilacion-Final-Interactivo.pdf

[28] “Población por condición de discapacidad, según provincia, cantón, parroquia y área de empadronamiento.”, *Instituto Nacional de Estadísticas y Censos*. https://www.ecuadorencifras.gob.ec/wp-content/plugins/download-monitor/download.php?id=326&force=0

[29] “Estudiantes con discapacidad en educación básica, media y bachillerato.”, *Consejo Nacional de Igualdad de Discapacidades*. https://www.consejodiscapacidades.gob.ec/estadisticas-de-discapacidad/

[30] A. Oviedo, X. Carrera, y C. Rocío, “Ecuador, atlas sordo.”, *Cultura Sorda*. https://cultura-sorda.org/ecuador-atlas-sordo/

[31] T. C. Lorduy y T. C. Pereira, “EVALUACIÓN DEL PACIENTE CON HIPOACUSIA.”, p. 14.

[32] “Personas sordas del Ecuador”, *Federación Nacional de Personas Sordas del Ecuador*. https://www.fenasec.ec/comunity.html (consultado jun. 09, 2021).

[33] “MakeHuman - EcuRed”. https://www.ecured.cu/MakeHuman (consultado sep. 06, 2021).

[34] L. Z. Eng, *Building a game with Unity and Blender learn how to build a complete 3D game using the industry-leading Unity game development engine and Blender, the graphics software that gives life to your ideas*. Birmingham: Packt Publishing, 2015. Consultado: sep. 11, 2021. [En línea]. Disponible en: http://www.myilibrary.com?id=876508

[35] J. Hocking, *Unity in action: multiplatform game development in C#*. Shelter Island, NY: Manning Publications Co, 2015.

[36] J. M. Gratch, *Intelligent virtual agents: 6th international conference, IVA 2006, Marina Del Rey, CA, USA, August 21-23, 2006 proceedings*. Berlin New York: Springer, 2006.

[37] Hannes Högni Vilhjálmsson, *Intelligent virtual agents: 11th International Conference, IVA 2011, Reykjavik, Iceland, September 15-17, 2011 proceedings*. Heidelberg: Springer, 2011.

[38] L. Devillers, M. Eskenazi, y J. Mariani, Eds., *Advanced Social Interaction with Agents: 8th International Workshop on Spoken Dialog Systems*, 1st ed. 2019.

Cham: Springer International Publishing : Imprint: Springer, 2019. doi: 10.1007/978-3-319-92108-2.

[39] S. M. Cossu, *Beginning game AI with Unity: programming artificial intelligence with C#*. 2021. Consultado: sep. 11, 2021. [En línea]. Disponible en: https://discover.gcu.ac.uk/discovery/openurl?institution=44GLCU_INST&vid=44GLCU_INST:44GLCU_VU2&?u.ignore_date_coverage=true&rft.mms_id=992002694510203836

[40] L. Deng y Y. Liu, *Deep learning in natural language processing*. Springer.

[41] W. Yih, X. He, y J. Gao, "Deep Learning and Continuous Representations for Natural Language Processing", en *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorial Abstracts*, Denver, Colorado, may 2015, pp. 6–8. doi: 10.3115/v1/N15-4004.

[42] W. Yin, K. Kann, M. Yu, y H. Schütze, "Comparative Study of CNN and RNN for Natural Language Processing", *ArXiv170201923 Cs*, feb. 2017, Consultado: sep. 20, 2021. [En línea]. Disponible en: http://arxiv.org/abs/1702.01923

[43] N. Indurkhya y F. Damerau, *Handbook of Natural Language Processing.*, 2a ed. Chapman & Hall/CRC, 2010.

[44] V. Björk, "Mapping of open-answers using machine learning", *2018*, p. 66, 2018.

[45] P. Verschuuren, S. Palazzo, T. Powell, S. Sutton, A. Pilgrim, y M. F. Giannelli, "Supervised machine learning techniques for data matching based on similarity metrics", *ArXiv200704001 Cs Stat*, sep. 2021, Consultado: dic. 02, 2021. [En línea]. Disponible en: http://arxiv.org/abs/2007.04001

[46] S. kulkarni, "Jaro Winkler vs Levenshtein Distance", *Medium*, mar. 30, 2021. https://srinivas-kulkarni.medium.com/jaro-winkler-vs-levenshtein-distance-2eab21832fd6 (consultado dic. 02, 2021).

[47] "Universidad – Diccionario Gabriel Roman". http://www.plataformaconadis.gob.ec/~platafor/diccionario/?st_kb=universidad (consultado sep. 13, 2021).

[48] E. j Isava, "Las 500 palabras más usadas en español", *Español al Día*, jul. 08, 2016. https://espanolaldia.wordpress.com/2016/07/08/las-500-palabras-mas-usadas-en-espanol/ (consultado dic. 02, 2021).