UNIVERSIDAD NACIONAL DE COLOMBIA

Desarrollo de una herramienta de *software* para la integración de sensores hápticos a interfaces de realidad aumentada orientadas al entrenamiento industrial

Laura Andrea Moreno Rodríguez

Universidad Nacional de Colombia
Facultad de Ingeniería
Bogotá, Colombia
2021

Desarrollo de una herramienta de *software* para la integración de sensores hápticos a interfaces de realidad aumentada orientadas al entrenamiento industrial

Laura Andrea Moreno Rodríguez

Trabajo final presentado como requisito parcial para optar al título de:
Maestría en Ingeniería de Sistemas y Computación

Director:
Ph.D., Fredy Andrés Olarte Dussan

Línea de Investigación:
Desarrollo de recursos digitales para ambientes de realidad aumentada o virtual
Grupo de Investigación:
GITEI Tecnología para la educación y la innovación

Universidad Nacional de Colombia
Facultad de Ingeniería
Bogotá, Colombia
2021

Resumen

Los sistemas de realidad virtual y aumentada son tecnologías que han impuesto nuevas formas de interacción entre los usuarios y entornos simulados por computadora. Con diversos campos de aplicación, estos sistemas han demostrado tener potencial en el sector industrial para apoyar el entrenamiento de trabajadores en múltiples sectores. Sin embargo, se hace necesario desarrollar interfaces hombre-máquina cada vez más intuitivas y menos invasivas para lograr un alto impacto y aceptación por parte de los usuarios. Los guantes hápticos son dispositivos modernos alineados con este objetivo, pues gracias a un conjunto de sensores integrados, rastrean el movimiento de las manos y los dedos con precisión, permitiendo a los usuarios interactuar tanto con el mundo real a su alrededor, como con objetos virtuales agregados a la escena.

A pesar de encontrar diferentes productos comercializados de este tipo, aplicaciones de realidad mixta para el sector industrial y múltiples proveedores de guantes hápticos, no existe una única solución general para diversos campos de la industria. De hecho, tampoco es posible reutilizar recursos entre proyectos debido a las grandes diferencias que enfrenta cada aplicación, en cuanto a requerimientos, herramientas y dispositivos implementados. Con el fin de aportar a la solución, en este proyecto se diseñó e implementó una librería de *software* para integrar rápidamente los guantes hápticos Captoglove a cualquier aplicación de realidad mixta desarrollada en Unity. Con esta librería se simplifica el desarrollo de aplicaciones dirigidas al entrenamiento industrial de trabajadores, al sintetizar el proceso de configuración, procesamiento de datos e interacción con objetos virtuales a través de gestos específicos y comúnmente encontrados en diferentes entornos de capacitación.

Palabras clave: realidad aumentada, realidad virtual, guantes hápticos, interfaz hombre-máquina, rastreo de manos, entrenamiento industrial.

Abstract

Virtual and augmented reality systems are technologies that have imposed new forms of interaction between users and computer simulated environments. With diverse fields of application, these systems have shown potential in the industrial sector to support the training of workers. However, it is necessary to develop intuitive and less invasive human-machine interfaces to achieve high impact and acceptance by the users. Haptic gloves are modern devices aligned with this objective, because a set of integrated sensors that track the movement of the hands and fingers with precision, allowing users to interact both with the real world around them, as with virtual objects added to the scene.

Despite there are different commercialized products of this type, mixed reality applications for the industrial sector and multiple suppliers of haptic gloves, there is no an unique general solution for different fields of industry. In fact, it is also not possible to reuse resources among projects due to the great differences that each application faces, in terms of requirements, tools and implemented devices. In order to contribute to the solution, a software library was designed and implemented in this project to quickly integrate Captoglove haptic gloves to any mixed reality application developed in Unity. With this library, the development of applications for industrial worker training is simplified, by synthesizing the configuration process, data processing and interaction with virtual objects through specific gestures commonly found in different training environments.

Keywords: augmented reality, virtual reality, haptic gloves, human-machine interface, hand tracking, industrial training.

Este Trabajo Final de Maestría fue calificado en Abril de 2021 por el siguiente evaluador:

Edgar Miguel Vargas Chaparro Msc.
Profesor Departamento de Ingeniería de Sistemas e Industrial
Facultad de Ingeniería
Universidad Nacional de Colombia

Contenido

Resumen v

Lista de figuras ix

1. Introducción 3
- 1.1. Planteamiento del problema . . . 4
- 1.2. Solución propuesta . . . 5
- 1.3. Contenido del documento . . . 6

2. Conceptos asociados 7
- 2.1. Continuo de la virtualidad . . . 7
  - 2.1.1. Realidad virtual . . . 8
  - 2.1.2. Realidad aumentada . . . 9
- 2.2. Interacción tangible . . . 10
  - 2.2.1. Rastreo del cuerpo . . . 11
  - 2.2.2. Gestos . . . 11
  - 2.2.3. Contacto . . . 12
- 2.3. Sensores hápticos . . . 13
- 2.4. Herramientas de desarrollo . . . 14
  - 2.4.1. Unity . . . 14
  - 2.4.2. Vuforia . . . 17
- 2.5. Resumen del capítulo . . . 17

3. Descripción de la solución 18
- 3.1. Descripción de los guantes hápticos . . . 19
  - 3.1.1. Datos capturados . . . 22
  - 3.1.2. Fórmula de movimiento . . . 27
  - 3.1.3. Limitaciones . . . 29
- 3.2. Descripción de la librería GITEICaptoglove . . . 30
  - 3.2.1. Gestos definidos . . . 31
  - 3.2.2. Descripción de las clases . . . 33
- 3.3. Resumen del capítulo . . . 42

4. Resultados **43**
4.1. Librería DLL . . . 43
4.2. Documentación y código fuente . . . 45
4.3. Modelos 3D . . . 45
4.4. Aplicaciones de ejemplo . . . 46
4.4.1. Aplicación en realidad virtual . . . 46
4.4.2. Aplicación en realidad aumentada . . . 49
4.4.3. Simulación de DPS . . . 51
4.5. Instrucciones de configuración . . . 55
4.6. Resumen del capítulo . . . 55

5. Conclusiones y trabajo futuro **56**
5.1. Resumen . . . 56
5.2. Conclusiones . . . 57
5.3. Trabajo futuro . . . 58

A. Anexos digitales **59**
A.1. Aplicación DPS . . . 59
A.2. Aplicación en realidad aumentada . . . 59
A.3. Aplicación en realidad virtual . . . 59
A.4. Código fuente GITEICaptoglove . . . 60
A.5. Diagrama de clases GITEICaptoglove . . . 60
A.6. Documentación GITEICaptoglove . . . 60
A.7. Documentación SDK Captoglove . . . 60
A.8. Captoglove Suite . . . 61
A.9. Instructivo Captoglove . . . 61
A.10. Librería GITEICaptoglove . . . 61
A.11. Modelos 3D . . . 61
A.12. Script movimiento de la mano derecha . . . 62
A.13. Script movimiento del antebrazo derecho . . . 62
A.14. Script aplicación DPS . . . 62
A.15. SDK Captoglove . . . 62
A.16. Vídeo simulación DPS . . . 63
A.17. Vídeo aplicación en realidad aumentada . . . 63
A.18. Vídeo aplicación en realidad virtual . . . 63
A.19. Vídeo movimiento de los guantes Captoglove . . . 63

Bibliografía **64**

Lista de Figuras

2-1. Continuo de la virtualidad [30] 7
2-2. Visor HTC Vive (izquierda) [34] y Samsung Gear VR (derecha) [30] 8
2-3. Guantes de pellizco [30] 10
2-4. Gesto de rectángulo con las dos manos [30] 12
2-5. Editor de Unity con paneles de escena (1), inspector (2) y jerarquía (3) . . . 14

3-1. Guante *CaptoGlove* [3] 19
3-2. Sensor *CaptoSensor* [3] 20
3-3. Calibración de flexómetros en *CaptoGlove Suite* [3] 21
3-4. SDK de Captoglove 22
3-5. Ejes de rotación *CaptoSensor* [3] 23
3-6. Rotación en el eje *pitch* del *CaptoSensor* 23
3-7. Rotación en el eje *yaw* del *CaptoSensor* 24
3-8. Rotación en el eje *roll* del *CaptoSensor* 25
3-9. Flexión de los dedos en *CaptoGlove* 26
3-10.Sensor de presión del *CaptoGlove* 26
3-11.Máxima rotación en el eje X del dedo índice en Unity 27
3-12.Mínima rotación en el eje X del dedo índice en Unity 27
3-13.Rotación del dedo índice para cada valor del flexómetro 29
3-14.Gestos de interacción: mano cerrada (1), mano abierta (2), gesto del número uno (3), gesto del número dos (4), gesto del número tres (5) [8] 32
3-15.Diagrama de clases de la librería *GITEICaptoglove* (simplificado) 33
3-16.Diagrama de flujo para *MyHand:MyHand() y MyArm:MyArm()* 35
3-17.Diagrama de flujo para *Module:Start()* 36
3-18.Diagrama de flujo para *MyHand:MoveHand()* 37
3-19.Diagrama de flujo para *MyHand:MoveFingers()* 38
3-20.Diagrama de flujo para *MyHand:HandClosed()* 39
3-21.Diagrama de flujo para *MyHand:FingerGesture1()* 40
3-22.Diagrama de flujo para *MyHand:FingerGesture2()* 40
3-23.Diagrama de flujo para *MyHand:FingerGesture3()* 41
3-24.Diagrama de flujo para *MyHand:SensorPressed()* 41

4-1. Diagrama de flujo para un desarrollador 44
4-2. Modelo 3D de las manos en escena de Unity 46

4-3. Aplicación en realidad virtual 47
4-4. Atrapar y arrastrar la cápsula cerrando la mano 48
4-5. Cambiar color de la esfera mediante gestos 48
4-6. Aplicación en realidad aumentada 50
4-7. Oprimir botón con sensor de presión 50
4-8. Atrapar y mover palanca cerrando la mano en puño 51
4-9. Entrenamiento real para la manipulación de un DPS [6] 51
4-10.Aplicación en realidad aumentada: simulación de DPS 52
4-11.Acercar mano a DPS energizado 53
4-12.Diagrama de flujo para interacción con DPS sin librería *GITEICaptoglove* . . 54
4-13.Diagrama de flujo para interacción con DPS usando librería *GITEICaptoglove* 55

Siglas

API Application Programing Interface. 14, 15

AR Augmented Reality. 9, 10

DLL Dynamically Linked Library. 16, 21, 34, 43, 58

DPS Dispositivo de Protección de Sobrecarga. 31, 52, 53

GITEI Grupo de Investigación de Tecnología para la Educación y la Innovación. 5, 31, 46

HMD Head-Mounted Display. 8, 10, 58

HMI Human-Machine Interface. 3–5

ID Identificación. 34, 35

IDE Integrated Development Enviroment. 15

SDK Software Development Kit. 17, 21, 22, 27, 30, 34, 35, 43

VR Virtual Reality. 8–10

1. Introducción

La realidad virtual y aumentada son tecnologías en constante evolución que han incursionado en diferentes campos de aplicación, demostrado su potencial y versatilidad gracias a su componente interactivo característico [28]. La realidad aumentada, por ejemplo, ha sido aplicada en la educación, para la elaboración de libros infantiles interactivos [4]; el entretenimiento, para la creación de vídeo juegos; la publicidad, para el modelamiento de nuevos productos [9]; la medicina, para la asistencia en cirugías de alta complejidad; la construcción, para el modelamiento 3D de estructuras y controles de calidad; y en la industria operacional, para el entrenamiento de trabajadores y aprendices [30].

En fábricas con procesos de ensamble manual y líneas de producción complejas, como en la industria automotriz y aeroespacial, la realidad aumentada es utilizada para manejar el despliegue de información pertinente al trabajador durante el proceso de ensamble, y así reducir el número de errores durante la ejecución de una tarea [29, 31]. En procesos a menor escala, como en la electrónica, se encuentran aplicaciones donde por medio de visores de realidad aumentada y códigos de colores, se enseña a los nuevos trabajadores el proceso de ensamble de circuitos impresos [10].

Para ampliar los beneficios de los sistemas de realidad aumentada en el sector industrial y de mantenimiento, y mejorar las capacidades de los trabajadores, se deben combinar herramientas como sistemas de monitoreo en tiempo real, diagnóstico de averías y sistemas de prevención [21]. Por ejemplo, el rastreo de manos y cuerpo en tiempo real, es utilizado dentro del sector de la construcción para realizar capacitaciones virtuales y disminuir los riesgos de accidente a los que se exponen los trabajadores durante las capacitaciones en sitio [13]. Con soluciones de este tipo, se espera que la realidad virtual y aumentada sean lo suficientemente estables en un futuro próximo, favoreciendo el desarrollo de interfaces hombre-máquina (*Human-Machine Interface (HMI)*) cada vez más complejas que pueden tener un gran impacto en el sector industrial [28].

Las interfaces HMI se refieren a la comunicación entre una persona y un dispositivo, sistema o máquina en particular. Con la ayuda de tecnología avanzada, y la incorporación de la realidad virtual y aumentada, las soluciones HMI han migrado de los dispositivos de control convencionales, como el teclado, pantallas táctiles y el Joystick, a alternativas más diversificadas y creativas para una interacción más intuitiva. Las HMI basadas en guantes portátiles, por ejemplo, tienen la ventaja única de ofrecer alta precisión y rastreo de múltiples grados de libertad de la mano [43].

Los guantes hápticos portátiles capturan varios parámetros importantes para obtener la información completa de la mano y entregan estos datos a las aplicaciones virtuales; la flexión de los dedos es la acción que se mide con mayor frecuencia, junto con los grados de libertad de la mano. Además, los movimientos direccionales laterales, se convierten en otro punto de interés, según el requisito de registrar todas las fuerzas interactivas para el uso de herramientas externas y lograr una interacción eficaz, intuitiva y fluida. Por otra parte, también son necesarias una serie de técnicas de retroalimentación para establecer una experiencia de inmersión y mejorar la capacidad del usuario de detectar objetos virtuales. Sin embargo, las soluciones comercializadas de este tipo, siguen limitadas por los altos costos de fabricación, consumo de energía y cálculo de potencia [43].

En este sentido, actualmente se adelantan múltiples investigaciones en el rastreo del movimiento de las manos por medio de guantes hápticos. Estudiantes de la Universidad de Singapur crearon recientemente unos guantes para interfaces HMI aumentadas, con retroalimentación háptica, sensores de flexión de dedos, sensor de deslizamiento de la palma y estimuladores mecánicos piezoeléctricos. El guante además logra el reconocimiento de objetos reales mediante la técnica de aprendizaje automático, con una precisión del 96 % [43]. Otras investigaciones incluyen técnicas de retroalimentación al usuario por medio de vibración [25], aprendizaje de máquina para detectar gestos complejos [41], y desarrollos para áreas con necesidades específicas de interacción con maquinaria [7].

1.1. Planteamiento del problema

La realidad aumentada, enriquecida con *hardware* especializado de interacción, es una tecnología moderna y poderosa que puede utilizarse para mejorar la operatividad de los trabajadores en campo. Esta tecnología se integró al entorno industrial con el objetivo de proporcionar instrucciones digitales intuitivas, al mismo tiempo que los trabajadores realizan sus tareas, minimizando, por ejemplo, el tiempo dedicado a buscar manuales de instrucciones.

Las múltiples investigaciones adelantadas en esta área evidencian que no existe una única solución general para diversos campos de aplicación: cada escenario debe analizarse cuidadosamente para elegir los dispositivos y las plataformas de desarrollo más adecuadas, y diseñar su entorno a partir de necesidades específicas, generando interfaces personalizadas difíciles de adaptar a otras áreas de trabajo [21].

No obstante, dentro de un mismo equipo de desarrollo de aplicaciones, es posible establecer herramientas de *software* y dispositivos estándar para crear sistemas HMI con diferentes propósitos. Esto abre la posibilidad de generar recursos reutilizables para disminuir el tiempo de desarrollo del equipo y evitar actividades duplicadas; en general, para realizar un trabajo más eficiente.

1.2. Solución propuesta

El Grupo de Investigación de Tecnología para la Educación y la Innovación (GITEI), de la Universidad Nacional de Colombia, ha trabajado en la creación de soluciones tecnológicas para la capacitación de trabajadores en la industria, incluyendo sistemas de realidad mixta. Para avanzar en sus objetivos, y de acuerdo con los antecedentes presentados, GITEI ha identificado la necesidad de incorporar guantes hápticos a sus interfaces HMI. Por esta razón, con el fin de aportar a la solución, en este proyecto se creó una librería de *software* que simplifica el proceso de integración de los guantes hápticos Captoglove a cualquier aplicación desarrollada en Unity.

La librería se encuentra orientada a la capacitación de trabajadores, en el sentido en que se establecieron una serie de gestos y movimientos de la mano para ser detectados, y que pueden ser adaptados a entornos de diferentes industrias. En resumen, la librería puede ser utilizada por desarrolladores de *software* para crear aplicaciones de realidad mixta en las que se requiera la interacción del usuario a través de las manos con objetos virtuales, tales como botones, palancas, herramientas y/o maquinaria simulada. La librería se encarga de la conexión y configuración de los módulos Captoglove de acuerdo con el entorno establecido por el desarrollador, y realiza internamente el procesamiento de los datos de los sensores para rastrear y simular los movimientos de los brazos y las manos del usuario en tiempo real.

1.3. Contenido del documento

En los siguientes capítulos de este documento se presentan los detalles del trabajo realizado y los resultados obtenidos. En este capítulo se realizó una breve descripción de la problemática en cuanto a la reutilización de recursos de *software* en proyectos de realidad mixta, y se introdujo la librería de integración propuesta para facilitar el uso de guantes hápticos en aplicaciones para el entrenamiento de trabajadores. En el segundo capítulo, se definen los conceptos y términos más relevantes sobre la realidad virtual y aumentada, necesarios para comprender el contexto y contenido de este documento. En el tercer capítulo, se describe en detalle la solución desarrollada, iniciando con la descripción de los guantes hápticos seleccionados para este proyecto, seguido de los parámetros de diseño de la librería de *software* y finalmente su implementación. En el cuarto capítulo, se encuentran los resultados obtenidos al integrar la librería a tres aplicaciones de ejemplo en Unity, y se describen los demás entregables generados en este proyecto. Por último, en el quinto capítulo, se presentan las conclusiones obtenidas luego del desarrollo de este trabajo y algunas recomendaciones para el trabajo futuro con sensores hápticos en sistemas de realidad virtual y aumentada.

2. Conceptos asociados

En este capítulo se definen los conceptos básicos y la terminología necesaria para comprender el contexto y el contenido de este documento. Específicamente se abarcan los conceptos de realidad mixta, interacción tangible, sensores hápticos y herramientas de *software* para el desarrollo de aplicaciones de realidad virtual y aumentada.

2.1. Continuo de la virtualidad

Al espacio entre la realidad y la realidad virtual, que permite combinar elementos reales y virtuales a diferentes niveles mediante la tecnología informática, se le llama realidad mixta. El continuo de la virtualidad, propuesto por Miligram y Kishino, captura todas las posibles combinaciones entre estos dos mundos como se muestra en el diagrama de la figura **2-1** [18] [30]. La virtualidad aumentada es aquella donde hay mayoritariamente elementos virtuales pero aún se involucra el mundo real. Por ejemplo, un videojuego donde los *avatar*[1] son animaciones virtuales que utilizan el rostro del usuario con texturas en tiempo real extraídas de la cámara del jugador [30]. La realidad virtual y la realidad aumentada se explican a continuación.

Figura 2-1.: Continuo de la virtualidad [30]

[1]Identidad virtual que escoge un usuario de computadora o videojuego para que lo represente en una aplicación o sitio web [22]

2.1.1. Realidad virtual

La realidad virtual, o *Virtual Reality (VR)*, es aquella donde se sumerge al usuario en un entorno completamente generado por computadora. Esto elimina todas las restricciones sobre lo que un usuario puede hacer o experimentar durante una simulación y por eso se ha convertido en el espacio más popular para los videojuegos [30]. Hasta hace algunos años la realidad virtual estaba fuera del alcance de los usuarios promedio debido a sus altos costos. Sin embargo, los avances en esta tecnología de las últimas décadas han iniciado una revolución en el mercado masivo que podría llegar a ser tan influyente como la invención de la televisión, los teléfonos inteligentes e Internet [23].

El *hardware* para VR de consumo es todavía una área joven de la que han emergido una variedad de dispositivos como los visores *Head-Mounted Display (HMD)*. Estos permiten proyectar imágenes en una pantalla muy cerca de los ojos del usuario abarcando todo su campo de visión [23]. Los HMD más avanzados cuentan con su propio procesador computacional para rastrear el movimiento de la cabeza y permiten la interacción del usuario a través de control remoto; dentro de los visores más populares de este tipo se encuentran los Oculus Rift y los HTC Vive. Otro tipo de HMD son aquellos que usan los teléfonos inteligentes como procesador principal y dispositivo de entrada. Estos convierten la pantalla monocular del teléfono en una binocular para presentar imágenes separadas a cada ojo del usuario; ejemplos de este tipo de visores son los Google Cardboard y Samsung Gear VR [30]. Ver figura **2-2**.

Figura 2-2.: Visor HTC Vive (izquierda) [34] y Samsung Gear VR (derecha) [30]

Por otra parte, el *software* para crear y entregar experiencias VR a los consumidores ha evolucionado y continua creciendo rápidamente. Los motores de videojuego Unity 3D y Unreal, populares por permitir el desarrollo de aplicaciones de escritorio y móviles en *software* libre, se han convertido en las plataformas de desarrollo por defecto para realidad virtual y aumentada [23]. Su popularización se debe además, a que existe una gran comunidad de usuarios, foros, documentación y cursos en línea que soportan el aprendizaje autónomo del uso de estas herramientas.

2.1.2. Realidad aumentada

La realidad aumentada, o *Augmented Reality (AR)*), es la combinación del mundo real con el mundo virtual a través de métodos computacionales, en donde existen más elementos del primer mundo que del segundo [30]. En otras palabras, es la visualización a través de una pantalla del mundo físico con elementos virtuales agregados que enriquecen el entorno [9].

Para que un sistema sea considerado AR debe permitir la interacción con el usuario en tiempo real y tener un registro espacial 3D que le permita desplegar correctamente la información virtual [30], es decir, una interfaz AR debe rastrear los objetos de interés visualizados, obtener sus coordenadas locales y globales, y agregar nuevas imágenes tridimensionales en la posición y orientación correcta de tal manera que el usuario las perciba como parte del mundo real [30, 42]. Para esto, se utilizan cámaras de vídeo digital que capturan el campo visual del usuario, y se rastrean patrones en la imagen, conocidos como *targets*, que indican el lugar y la orientación de los objetos virtuales por agregar [9, 4].

El objetivo de la realidad aumentada es simplificar la vida de los usuarios brindando información que enriquece tanto la percepción como la interacción con el entorno, muchas veces destacando información que se encuentra presente, pero no puede ser detectada por los sentidos del cuerpo humano [9]. Su mayor ventaja frente a la realidad virtual es que, al no modificar drásticamente el entorno, ofrece una interacción más natural entre el usuario y la tecnología [42].

2.2. Interacción tangible

Los usuarios de realidad mixta controlan continuamente el punto de vista, la dirección y el enfoque de la escena en una aplicación, por ejemplo, al mover la cabeza mientras usan un visor HMD. Esta forma de interacción es una parte fundamental de la mayoría de experiencias VR y AR que convierte a los usuarios en observadores pasivos[2]. Para ir un poco más allá de la navegación en la escena y ofrecer nuevas experiencias a los usuarios, existen una gran variedad de técnicas que permiten crear interfaces de usuario diferentes a la tradicional aplicación de escritorio y, que en particular, involucran gestos y contacto físico [30].

Dentro de las experiencias VR y AR, las manos del usuario son utilizadas generalmente para controlar el movimiento y señalamiento tridimensional mediante dispositivos portátiles como el *Joystick*[3]. Alternativamente, las manos pueden involucrarse mucho más en la experiencia al usarlas por sí mismas, sin dispositivos extrínsecos. Este tipo de interacción se denominada interacción tangible. Por ejemplo, al agregar un *target* sobre la parte trasera de las manos es posible conocer cuando estas son visibles en una escena AR y habilitar la interacción del usuario con objetos virtuales [30]. Otra opción es integrar dispositivos externos como los guantes de pellizco que se muestran en la figura **2-3**. Estos aumentan las posibilidades de interacción en la escena al utilizar contactos eléctricos en la punta de los dedos para detectar la acción del usuario al presionarlos entre sí [26, 30].

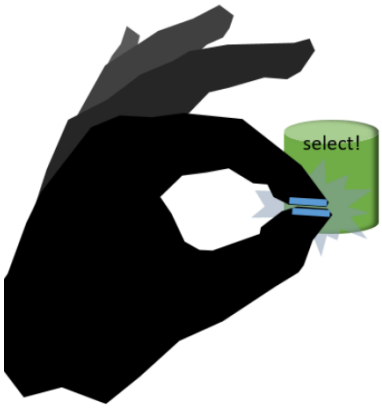

Figura 2-3.: Guantes de pellizco [30]

[2]Que permanece inactivo dejando que las cosas ocurran sin su intervención [22]
[3]Palanca de control que permite desplazar manualmente, y con gran rapidez, el cursor en una pantalla de computadora o videojuego [22]

Los objetos rastreados en una escena de realidad mixta no necesariamente deben ser agarrados con las manos en todo momento. Si hay múltiples elementos disponibles para interactuar, el usuario puede comunicar intenciones al mover las manos de una forma específica. Esto quiere decir, que la interacción tangible también se logra mediante el rastreo de la posición del cuerpo, los gestos y el contacto [30].

2.2.1. Rastreo del cuerpo

El uso de dispositivos como el cursor de un *mouse* para rastrear el movimiento del cuerpo tiene capacidades limitadas que no capturan la verdadera riqueza de la interacción humana con el mundo real. En consecuencia, múltiples investigaciones tratan de incorporar métodos de rastreo bastante complejos que se basan en el uso de una gran cantidad de sensores de entrada, como cámaras de alta resolución y sensores que pueden capturar el rango de movimiento del esqueleto humano. Conocer la posición de cada hueso del cuerpo es suficiente para detectar la mayoría de interacciones. Rastrear solo el esqueleto resulta además más sencillo que rastrear formas, pues las configuraciones del esqueleto están restringidas por limitaciones anatómicas [30].

Para algunas aplicaciones es suficiente rastrear solo las partes más relevantes del cuerpo como la cabeza, los brazos y las manos. El rastreo de las manos es particularmente importante, ya que la interacción del humano con su entorno ocurre principalmente a través de esta parte del cuerpo. En conjunto, la mano y los dedos tienen más de 20 grados de libertad que permiten cualquier manipulación precisa y fina. Por consiguiente, el problema del rastreo preciso de las manos ha recibido mucha atención en la investigación [20, 30].

2.2.2. Gestos

Un caso de uso importante para el rastreo de las manos es la interacción mediante gestos. Los gestos dinámicos tienen la ventaja de que pueden entregar simultáneamente datos cualitativos y cuantitativos a la aplicación. Por ejemplo, un usuario formando un rectángulo con los dedos, como se muestra en la figura **2-4**, puede ser interpretado como la acción de enfocar un área de la escena. Además, se puede conocer la cantidad del acercamiento que el usuario desea, interpretando la distancia entre las dos manos. Para este tipo de reconocimiento se utiliza principalmente visión computacional y cámaras de vídeo de alta definición[30].

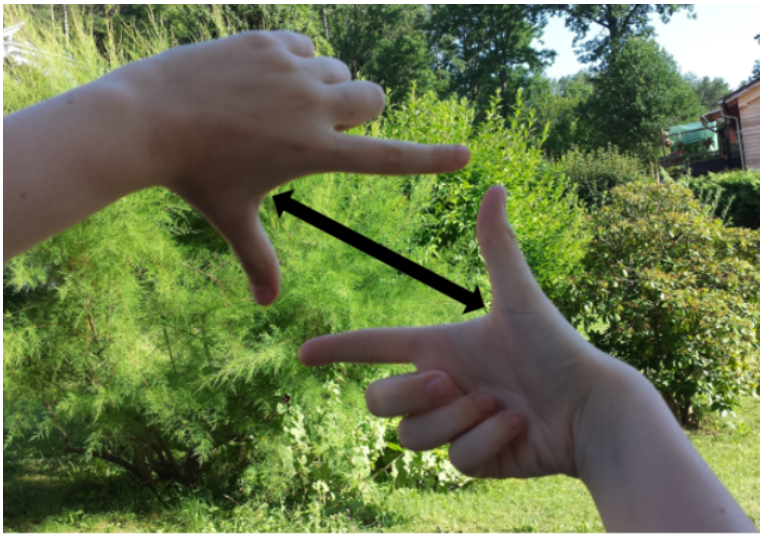

Figura 2-4.: Gesto de rectángulo con las dos manos [30]

El lenguaje de gestos es muy expresivo, pero también puede llegar a ser muy complejo en comparación con las interfaces de interacción tradicionales, pues requiere mayor aprendizaje tanto por parte de la computadora como del usuario. Adicionalmente, si se utiliza visión computacional, la auto-oclusión[4] del cuerpo puede llevar a situaciones donde el reconocimiento confiable de gestos es problemático [30]. No obstante, existen dispositivos de mano, como los guantes de pellizco y los guantes hápticos, que entregan retroalimentación a la aplicación por medio de sensores, y no a través de imágenes, reduciendo este tipo de errores.

2.2.3. Contacto

La interpretación de gestos en espacios libres sufre una falta de soporte físico que afecta la ejecución de operaciones delicadas. Dado que los humanos tienen un excelente sentido del contacto, es deseable construir interfaces táctiles, esto quiere decir, interfaces por medio de las cuales la interacción se dé a través del contacto mientras se entrega retroalimentación pasiva al usuario [30].

[4]En visión computacional, el fenómeno de la oclusión ocurre cuando no es posible visualizar un objeto en la escena debido a la obstrucción del mismo por parte de otro objeto que se antepone [30]

Para los propósitos de la realidad mixta, se pueden utilizar objetos físicos específicos para proporcionar retroalimentación, como las pantallas táctiles con vibración. Sin embargo, es mucho más complicado agregar retroalimentación a objetos virtuales, por lo que los desarrollos en esta área están enfocados en tratar de sintetizar y reproducir impresiones convenientes al tacto a través de dispositivos y entornos instrumentalizados. Para el mundo de la realidad virtual se han logrado avances importantes en la investigación de estas interacciones. Sin embargo, para la realidad aumentada siguen siendo limitadas, específicamente para las aplicaciones móviles, pues requieren tecnologías no obstructoras para sensar el tacto. Este es el dominio de investigación de la tecnología háptica [30].

2.3. Sensores hápticos

Mediante el uso de sensores hápticos es posible reproducir una variedad de fenómenos táctiles en realidad mixta. La retroalimentación háptica puede ser clasificada en dos tipos: retroalimentación cinestética y táctil. La primera es aquella que es forzada, por ejemplo, inyectando una señal eléctrica al cuerpo del usuario y es sensada por los nervios en las articulaciones y los músculos. La segunda, es el contacto directo con una superficie. Muchos dispositivos hápticos portátiles de este tipo han sido explorados como guantes, zapatos, chalecos, chaquetas y exoesqueletos [14, 32, 33, 35].

Jeon y Choi extendieron el continuo de la virtualidad de Miligram al dominio háptico. La realidad háptica incluye el uso de accesorios tangibles, rastreadores de posición y *targets*. Mientras que la virtualidad háptica corresponde a entornos con sensaciones hápticas totalmente sintetizadas, acompañadas de simulaciones visuales y auditivas. En resultado, la realidad mixta háptica usa la combinación de objetos reales y actuadores sintéticos [11, 30].

Los guantes para realidad virtual y aumentada son dispositivos hápticos diseñados para capturar el movimiento de las manos y los dedos del usuario, permitiendo identificar acciones específicas como la selección, el levantamiento y el lanzamiento de un objeto, o la aplicación de una fuerza sobre el mismo [16]. Estos dispositivos cuentan con un grupo de sensores integrados como acelerómetros, giroscopios y magnetómetros que permiten rastrear la posición y el movimiento de la mano con exactitud. También incluyen sensores de presión para detectar el contacto con otra superficie, flexómetros para detectar la flexión de los dedos y algunos de ellos entregan realimentación al usuario por medio de señales de vibración [16, 19, 40].

2.4. Herramientas de desarrollo

2.4.1. Unity

Unity es un motor de juego multi-plataforma desarrollado por Unity Technologies para la creación de aplicaciones bidimensionales y tridimensionales. Unity se utiliza principalmente para crear videojuegos móviles y de escritorio, pero su sistema en tiempo real, complementado con otras herramientas y servicios, ofrece a los desarrolladores la posibilidad de crear aplicaciones en todas las áreas. Esto lo ha convertido en la herramienta de *software* estándar para interfaces de realidad mixta [38].

Unity consiste en la fusión de varios módulos para administrar y renderizar[5] objetos 3D, aplicar leyes de la física, agregar iluminación, animaciones, audio y vídeo a las aplicaciones. Cada módulo cuenta con una interfaz de programador, o *Application Programing Interface (API)*, con un amplio conjunto de clases y funciones, de tal manera que el sistema puede ser accedido completamente a través de *scripts* escritos en el lenguaje de programación C# [15].

Editor de Unity

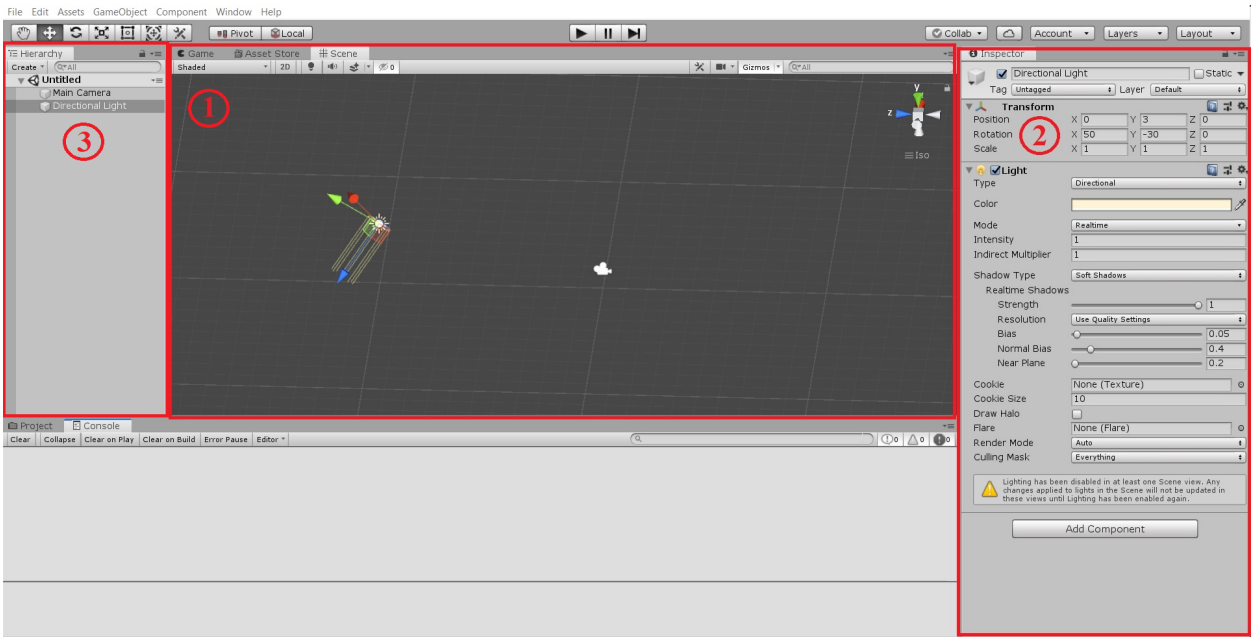

Figura 2-5.: Editor de Unity con paneles de escena (1), inspector (2) y jerarquía (3)

[5] En visión computacional, se refiere al procesamiento de una imagen usando color y sombreado para que parezca sólida y tridimensional en la escena [22]

La interfaz de Unity 3D, conocida como Editor, consiste en un conjunto de paneles y ventanas que permiten la creación de una aplicación tridimensional como se muestra en la figura **2-5**. El panel principal es la Escena donde se puede crear visualmente el espacio 3D de la aplicación y agregar objetos. La escena vacía por defecto de Unity, consiste en una cámara y una luz direccional ubicadas sobre un plano infinito de referencia que se extiende en los tres ejes x, y y z [15].

A la escena se pueden agregar cualquier tipo de objetos proporcionados por el motor, modelos extraídos de sitios web o creados en herramientas externas, como Blender[6]. Todos los objetos y sus componentes son listados en el panel de Jerarquía, al lado izquierdo del Editor. Al lado derecho se encuentra el panel de Inspector, donde es posible visualizar la transformada de cada objeto en la escena. Esta define cómo su geometría debe ser posicionada, rotada y escalada en el espacio 3D. Por ejemplo, una posición (0,3,0) corresponde a tres unidades hacia arriba del centro del plano (en la dirección y), x=0 y z=0. Una rotación de (50,330,0) significa que el objeto está rotado 50 grados alrededor del su eje x y 330 grados alrededor de su eje y. Las rotaciones en aplicaciones 3D generalmente se representan de dos maneras: Cuaterniones o ángulos de Euler. Unity usa Cuaterniones internamente, pero muestra valores de los ángulos de Euler equivalentes en el Inspector para facilitar la edición. Estos valores pueden ser modificadas libremente por el desarrollador desde el Editor de Unity o mediante *scripts* [15, 39].

Scripts

Los *scripts* son componentes configurables que permiten a los desarrolladores acceder al código interno de Unity a través del API de *scripting*. Este API es muy completo y tiene suficiente documentación disponible, lo que permite que los usuarios creen complementos para el motor usando sus mismos recursos. Esta es la forma en que la comunidad ha enriquecido la plataforma en funcionalidades a través de los años [15].

El *script* estándar en C# de Unity es un *Integrated Development Enviroment (IDE)* llamado *MonoDevelop*. Allí se heredan las características de la clase principal *MonoBehaviour*, incluyendo la capacidad de iniciar, actualizar y detener funciones. Unity controla el llamado de estos métodos de forma automática durante la ejecución de la aplicación de la siguiente manera:

[6]Programa de diseño dedicado especialmente al modelamiento, renderizado, animación y creación de gráficos tridimensionales [1]

- ***Start()*:** esta función es llamada una sola vez cuando inicia la ejecución de la aplicación y justo antes de que *Update()* sea llamado por primera vez [36]. Es donde generalmente se inicializan las variables del *script* [15].
- ***Update()*:** es un bucle que se repite una y otra vez durante el tiempo de ejecución de la aplicación para actualizar la pantalla en cada cuadro o *frame* del vídeo; aproximadamente 60 veces por segundo. Por tal motivo, es en esta función donde generalmente se encuentran los métodos principales de la aplicación [15, 36] .
- ***OnDestroy()*:** esta función es llamada únicamente cuando se detiene la ejecución de la aplicación y puede ser utilizada para liberar memoria o desconectar de forma segura dispositivos externos [36].

Plugins

En Unity es posible utilizar código o *scripts* creados por terceros en forma de complementos conocidos como *plugins*. Hay dos tipos de *plugins*: *managed plugins* y *native plugins*. Los primeros son ensambles .NET[7], creados con herramientas como Visual Studio, que solo tienen acceso a las características soportadas por estas librerías. Sin embargo, la mayoría son accesibles desde Unity gracias a las herramientas .NET estándar que utiliza el motor para compilar los *scripts*. Esto quiere decir que hay poca diferencia entre usar *managed plugins* y *scripts*, excepto por el hecho que los *plugins* son compilados fuera de Unity y es posible que el código fuente no sea visible para el desarrollador. Los *native plugins* son librerías de código nativo para plataformas específicas. Estas librerías pueden hacer llamados al sistema operativo y acceder a librerías de terceros que no están disponibles por defecto en Unity [37].

Para crear cualquier tipo de *plugins* es necesario escribir los *scripts* fuera del motor y generar una biblioteca *Dynamically Linked Library (DLL)* utilizando un compilador externo. El archivo DLL resultante puede luego ser agregado a cualquier proyecto Unity y desde allí acceder a las clases contenidas, tal como con los *scripts* tradicionales. La mayor ventaja de estos complementos, es la posibilidad de entregar funcionalidades a otros sin suministrar el código fuente [37]. Por ejemplo, como parte del *software* de un producto comercial o para contribuir a la comunidad de desarrollo de Unity.

[7] Plataforma de desarrollo creada por Microsoft e integrada a Unity para crear aplicaciones orientadas a objetos [17]

2.4.2. Vuforia

Vuforia es un *Software Development Kit (SDK)* de realidad aumentada para dispositivos móviles. Utiliza tecnología de visión artificial para reconocer y rastrear en tiempo real imágenes planas y objetos 3D simples. Su capacidad de registro permite a los desarrolladores posicionar modelos 3D en relación con objetos del mundo real cuando se ven a través de la cámara de un dispositivo móvil [27].

Vuforia es compatible con iOS y Android. Facilita principalmente el reconocimiento de diferentes tipos de formas, objetos, textos o espacios, usando una base de datos previamente establecida por el usuario localmente o en la nube. La integración de Unity y Vuforia hacen de ésta una herramienta más versátil [27].

2.5. Resumen del capítulo

En este capítulo se describió la técnica emergente de la interacción tangible que busca transformar los sistemas de realidad virtual y aumentada en interfaces hombre-máquina cada vez más intuitivas y menos invasivas, a través del uso de dispositivos hápticos para detectar gestos del usuario y simular el contacto con objetos virtuales. También se explicaron las generalidades de las plataformas Unity y Vuforia, por ser las herramientas de *software* estándar para el desarrollo de aplicaciones de realidad mixta utilizadas en este proyecto.

3. Descripción de la solución

En esta sección se describen los guantes hápticos Captoglove seleccionados para el desarrollo de este proyecto, sus características principales, funcionamiento y limitaciones. También se describe la librería diseñada e implementada para integrar rápidamente estos guantes a cualquier aplicación de realidad mixta en Unity.

La librería permite la interacción del usuario con objetos virtuales, a través del movimiento de las manos y gestos específicos. La librería se encarga de la conexión y configuración de los guantes Captoglove de acuerdo con el entorno establecido por el desarrollador, y realiza internamente el procesamiento de los datos de los sensores para capturar y simular el movimiento de las manos y antebrazos del usuario, esto quiere decir, que el desarrollador simplemente debe hacer llamados a las funciones de la librería para rastrear los movimientos del usuario, y ejecutar acciones de acuerdo a los requerimientos de su aplicación.

En consecuencia, los guantes Captoglove pueden ser utilizados en aplicaciones dirigidas al entrenamiento de trabajadores en diferentes industrias. Por ejemplo, por medio de la librería se habilita tanto la manipulación de herramientas, botones y palancas virtuales, como la interacción con maquinaria y dispositivos eléctricos simulados para actividades de mantenimiento e instalación. A este documento se adjuntan tres aplicaciones de ejemplo para demostrar su funcionalidad en los anexos A.1, A.2, y A.3, con sus respectivos vídeos de funcionamiento en los anexos A.16, A.17, y A.18. Dichas aplicaciones se describen específicamente dentro del siguiente capítulo de este documento.

A continuación se presentan más detalles del *hardware* y el *software* de esta solución.

3.1. Descripción de los guantes hápticos

Luego de realizar una comparación de costo-beneficio entre los guantes hápticos para realidad virtual disponibles en el mercado, se escogieron los guantes fabricados por la empresa estadounidense Captoglove. Esta empresa inició como un emprendimiento para fabricar herramientas de rehabilitación para la salud y evolucionó hasta desarrollar una interfaz de máquina manual y portátil para juegos y dispositivos inteligentes [2].

Captoglove ofrece dos tipos de productos: los guantes para ser utilizados directamente como dispositivos de entrada bajo la tecnología *plug-and-play* y diferentes *kits* de desarrollo con los que se pueden crear aplicaciones propias. Para este proyecto se adquirió el *kit* más completo llamado *Business Exclusive Suite* que tiene un costo de $900 USD y contiene: un par de guantes *CaptoGloves* y dos módulos adicionales *CaptoSensors* para aplicaciones más avanzadas. Los cuatro dispositivos cuentan con comunicación Bluetooth BLE, batería recargable y son diferenciados entre sí por un ID de cuatro dígitos asignado por el fabricante [2].

Figura 3-1.: Guante *CaptoGlove* [3]

Como se observa en la figura **3-1**, cada guante tiene cinco flexómetros para detectar el movimiento de los dedos en el eje de flexión con una resolución menor a un grado de curvatura, y un sensor de presión en la punta del dedo pulgar capaz de percibir presiones desde 100g a 10kg. Por otra parte, los módulos *CaptoSensors* de la figura **3-2**, que pueden ser ubicados en cualquier parte del cuerpo, recolectan datos a bajo nivel como acelerómetro, giroscopio y datos de la brújula, o en un nivel superior como información de rotación en los ejes *pitch, yaw* y *roll*[1] [3]. Cada módulo tiene diez puertos disponibles para conectar sensores externos, esto indica que dentro de cada *CaptoGlove* se encuentra un *CaptoSensor* al que se conectan los flexómetros y el sensor de presión.

El *kit* de desarrollo también incluye una interfaz llamada *CaptoGlove Suite* que permite configurar los módulos, actualizar la versión de *firmware*, probar la conexión Bluetooth, entre otras cosas. Una de las principales características de esta interfaz es que permite calibrar los módulos con respecto a su entorno y los sensores con respecto al usuario. Por ejemplo, un usuario puede calibrar los flexómetros abriendo y cerrando los dedos de la mano mientras tiene el guante puesto, y visualizar los puntos mínimos y máximos alcanzados por los sensores como se muestra en la figura **3-3**, donde las barras verdes se desplazan hacia la derecha a medida que aumenta la flexión de los dedos [3].

Figura 3-2.: Sensor *CaptoSensor* [3]

[1]Ejes definidos en aviación para describir el movimiento de un avión. Alternativamente se les llama eje transversal, vertical y longitudinal [5]

Figura 3-3.: Calibración de flexómetros en *CaptoGlove Suite* [3]

Por último, con el *kit* se adquieren los permisos para descargar el SDK de integración de los guantes con la plataforma Unity 3D. Dicho SDK contiene tres bibliotecas de enlace dinámico o archivos DLL como se muestra en la figura **3-4**: *GSdkNet.BLE.Winapi* que controla principalmente la conexión Bluetooth de los módulos a aplicaciones en Windows, *GSdkNet.BLE.Droid* que controla la conexión Bluetooth de los módulos a aplicaciones en Android, y *GSdkNet* que transmite constantemente y en tiempo real cadenas de datos con los valores sensados a bajo nivel, para que sean interpretados y procesados en la aplicación Unity [3].

En este proyecto se utilizaron los *CaptoGloves* para sensar el movimiento y la rotación de las manos y los dedos del usuario, y los *CaptoSensors* para sensar el movimiento y la rotación de los antebrazos. De esta manera, se ofrece mayor movilidad al usuario dentro de las aplicaciones de realidad virtual. Sin embargo, el número de módulos a utilizar durante la ejecución de la aplicación es configurable y no se requieren los cuatro módulos conectados a la vez.

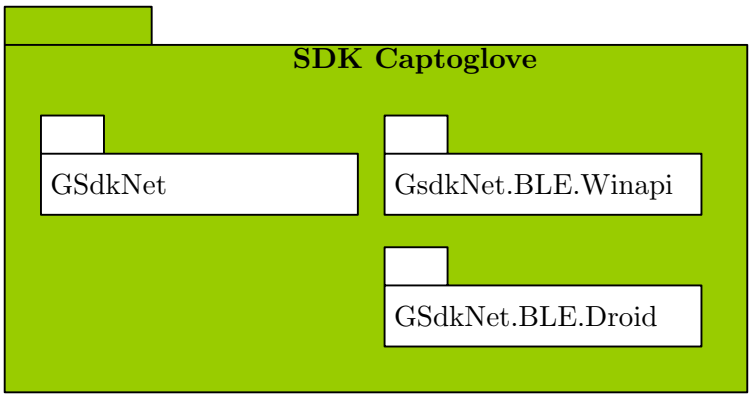

Figura 3-4.: SDK de Captoglove

3.1.1. Datos capturados

Como se mencionó anteriormente, los módulos Captoglove pueden sensar tres ejes de rotación con ayuda del giroscopio y el acelerómetro: *pitch, yaw* y *roll*. De acuerdo con la figura **3-5**, para los módulos ubicados en las manos del usuario, el movimiento *pitch* es una rotación respecto del eje transversal de la mano y se produce al inclinarla hacia arriba y hacia abajo. El movimiento *yaw* es una rotación con respecto al eje vertical de la mano, que se produce al moverla de izquierda a derecha manteniéndola paralela al suelo. El movimiento *roll* es una rotación respecto al eje longitudinal de la mano y se produce al girarla boca arriba y boca abajo. Los ejes de rotación se interpretan de la misma manera para los módulos ubicados en los antebrazos del usuario.

Utilizando el SDK para Unity se capturaron los valores sensados en los tres ejes de rotación, comprendiendo que cuando el *CaptoSensor* se encuentra en reposo sobre una superficie plana, la rotación en los tres ejes (X, Y, Z) tiene un valor de cero. En la figura **3-6** se presenta la variación en la rotación del eje X al inclinar el módulo hacia arriba (punto A) y hacia abajo (punto B) mientras que los otros dos ejes se mantienen relativamente estables. Lo anterior indica que el eje X es el eje transversal del *CaptoSensor* y su rotación toma valores en el rango de [1, -1] en unidades de Cuaterniones[2]; donde uno corresponde a un giro de 180º hacia arriba y menos uno a un giro de 180º hacia abajo. Cuando un usuario usa los *CaptoGloves* no se alcanzan estos valores máximos y mínimos debido a las limitaciones de movimiento natural de la mano.

[2]Los cuaterniones unitarios, en ingles *Unit quaternions*, proporcionan una notación matemática conveniente para representar orientaciones espaciales y rotaciones de objetos en tres dimensiones [12]. Son las unidades de rotación utilizadas internamente por Unity [39]

Figura 3-5.: Ejes de rotación *CaptoSensor* [3]

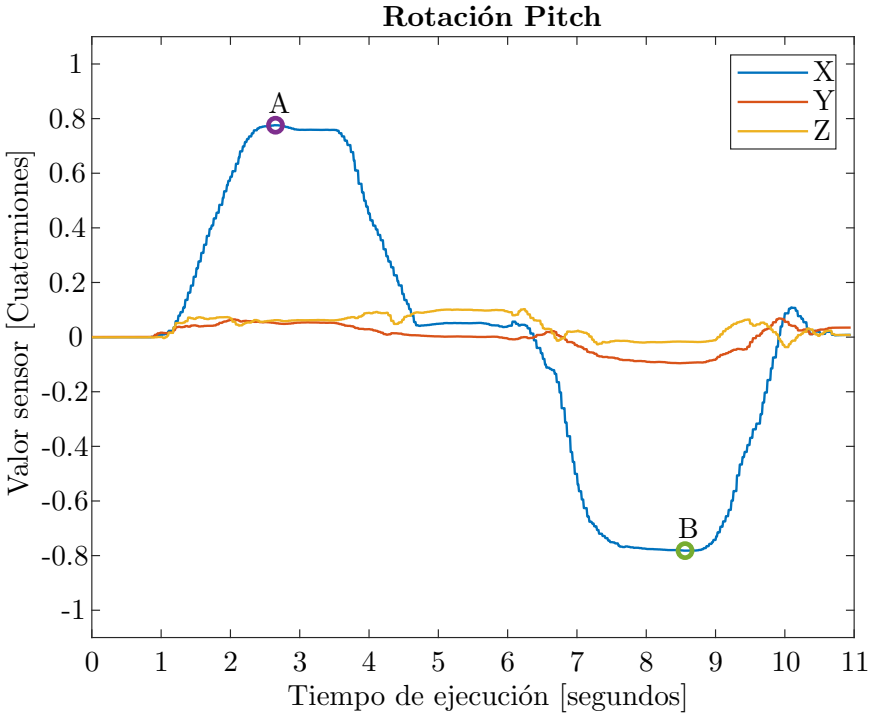

Figura 3-6.: Rotación en el eje *pitch* del *CaptoSensor*

Los datos de la figura **3-7** se obtuvieron con el movimiento del *CaptoSensor* hacia la derecha (punto A) y hacia la izquierda (punto B) manteniendo el dispositivo paralelo al suelo. Allí se evidencia que el eje Y es el eje vertical del módulo y su rotación toma valores en el rango de [1, -1] en unidades de Cuaterniones; donde uno es a un giro de 180º hacia la derecha y menos uno es un giro de 180º hacia la izquierda. Al igual que para el caso anterior, la rotación de los otros dos ejes permanece relativamente estable cerca a cero.

Con los resultados anteriores, se deduce que el eje Z es el eje longitudinal del *CaptoSensor* y se verifica al observar los datos de la figura **3-8**, donde la rotación de este eje varía entre [1, -1] en unidades de Cuaterniones; siendo menos uno un giro de 180º hacia la derecha hasta dejar el módulo boca arriba (punto A), y uno un giro de 180º hacia la izquierda hasta que el módulo esté boca arriba nuevamente (punto B). Mientras un usuario utiliza los *CaptoGloves* solamente es posible alcanzar el punto A con la mano derecha y el punto B con la mano izquierda debido a las limitaciones anatómicas de la mano.

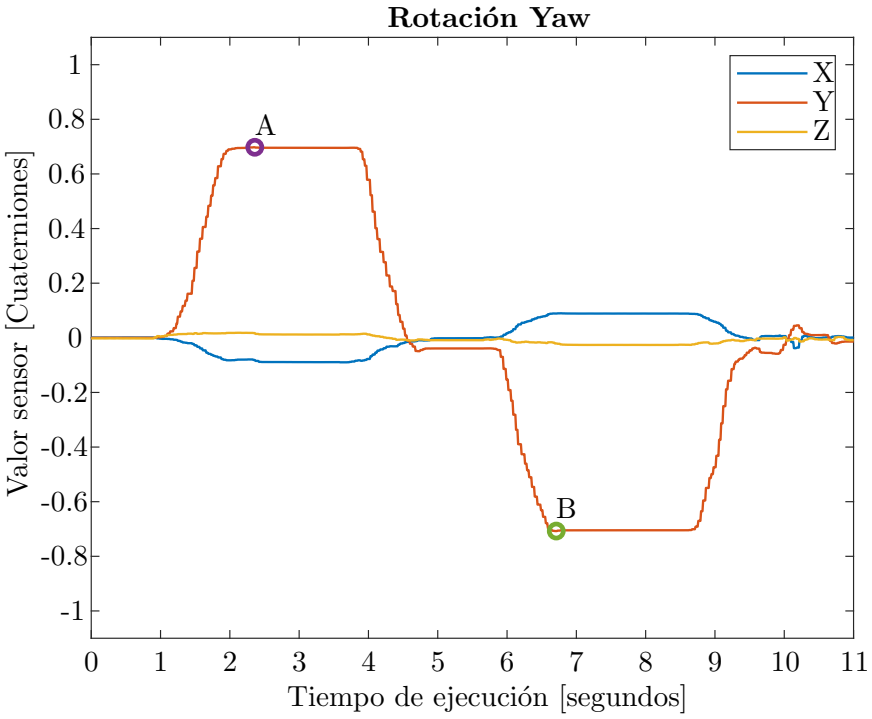

Figura 3-7.: Rotación en el eje *yaw* del *CaptoSensor*

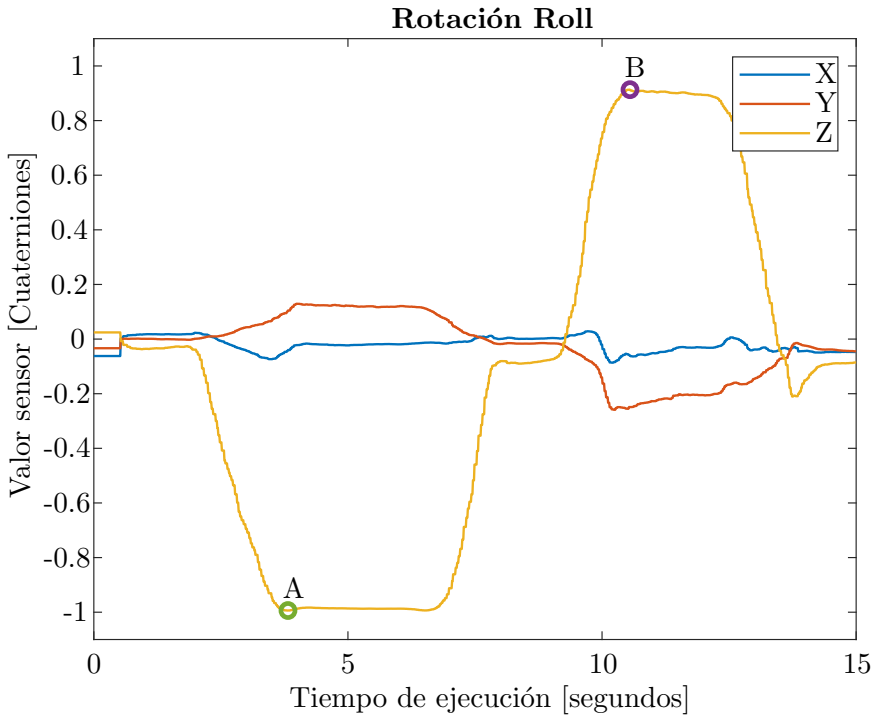

Figura 3-8.: Rotación en el eje *roll* del *CaptoSensor*

Por otra parte, al analizar los datos de los flexómetros en los *CaptoGloves*, representados en la figura **3-9**, se encontró que estos toman un valor máximo de conductividad[3] mientras los dedos se encuentran totalmente estirados (rango de tiempo [0, 5] segundos) y disminuye proporcionalmente a medida que se flexionan los dedos, hasta tomar valores muy cercanos a cero cuando la mano está totalmente cerrada en puño (rango de tiempo [5, 10] segundos). Luego de múltiples mediciones, se encontró que los sensores no son estables todo el tiempo y eventualmente alguno de ellos pierde su calibración entregando valores erróneos, como se observa en este caso para el dedo medio a partir de los 10 segundos del tiempo de ejecución. Sin embargo, esto no fue una limitación para el desarrollo de este proyecto, ya que se puede solucionar con la re-calibración de los módulos utilizando la interfaz *CaptoGlove Suite*.

Por último, en la figura **3-10** se encuentran los datos entregados por el sensor de presión ubicado en la punta del dedo pulgar de los *CaptoGloves*. Allí se observa que el sensor toma un valor de cero conductividad en su estado natural no presionado, y aumenta de acuerdo con la fuerza ejercida sobre él mientras se presiona. Para este caso en particular, el sensor estuvo presionado en el rango de tiempo [2, 5] segundos y alcanzó un valor máximo de 70548 [S/m].

[3]Las unidades de conductividad se encuentran expresadas en Siemens por metro [S/m] [3]

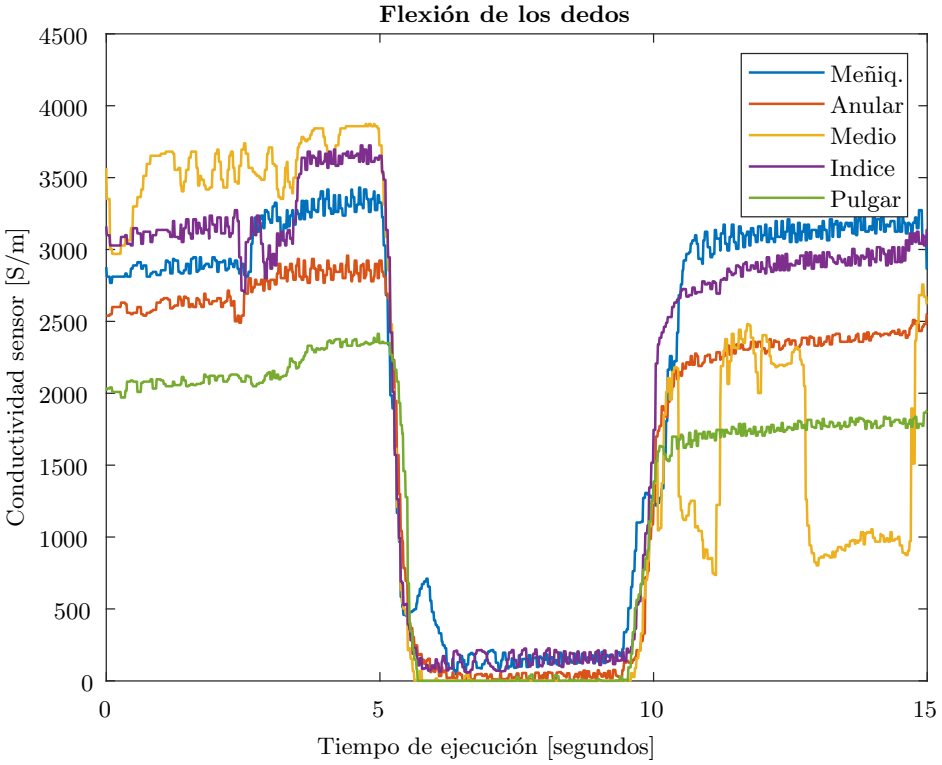

Figura 3-9.: Flexión de los dedos en *CaptoGlove*

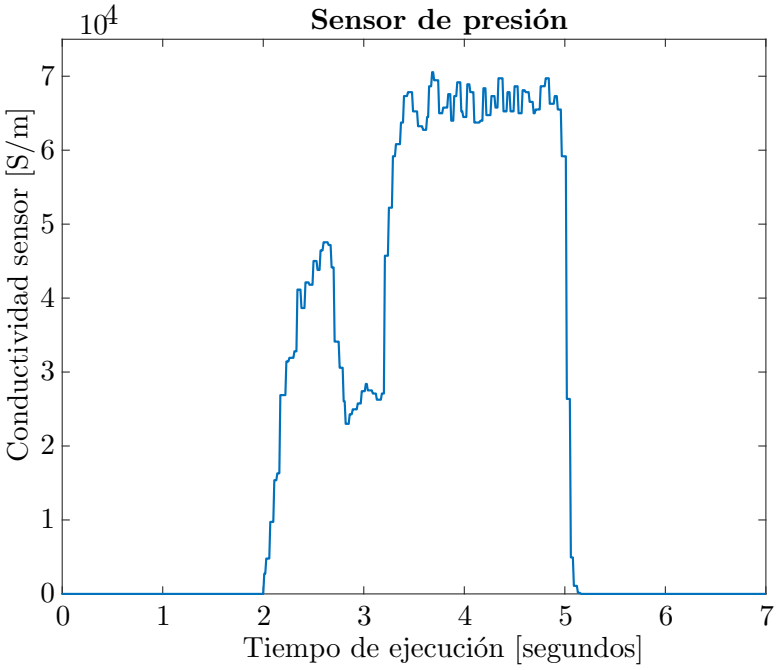

Figura 3-10.: Sensor de presión del *CaptoGlove*

3.1.2. Fórmula de movimiento

Con los datos recolectados se logró establecer una fórmula para simular el movimiento de las manos del usuario en un entorno virtual, acorde a su movimiento en el mundo real. Como se mencionó anteriormente, la calibración inicial de los módulos y sensores se realiza utilizando la interfaz *Captoglove Suite*, la cual almacena en la memoria interna de cada módulo las coordenadas que indican la posición de los módulos con respecto a su entorno, los valores máximos y mínimos de rotación y los límites de conductividad alcanzados por el usuario. Es importante tener en cuenta que los límites para los ejes de rotación son siempre [1, -1] en unidades de Cuaterniones, mientras que los límites de los flexómetros y sensor de presión varían de acuerdo al usuario que realiza la calibración.

Utilizando el SDK de Captoglove, es posible recuperar los valores máximos y mínimos almacenados en cada módulo y usarlos posteriormente en la fórmula de movimiento; estos valores son denominados *MaxC* y *MinC* en la ecuación. Adicionalmente, una vez establecido el modelo 3D de las manos que será utilizado en la aplicación, utilizando el panel de Inspector de Unity es posible determinar la rotación máxima y mínima que puede tomar tanto la mano como cada uno de los dedos en la escena; estos valores son denominados *MaxU* y *MinU* en la ecuación. Ver figuras **3-11** y **3-12**.

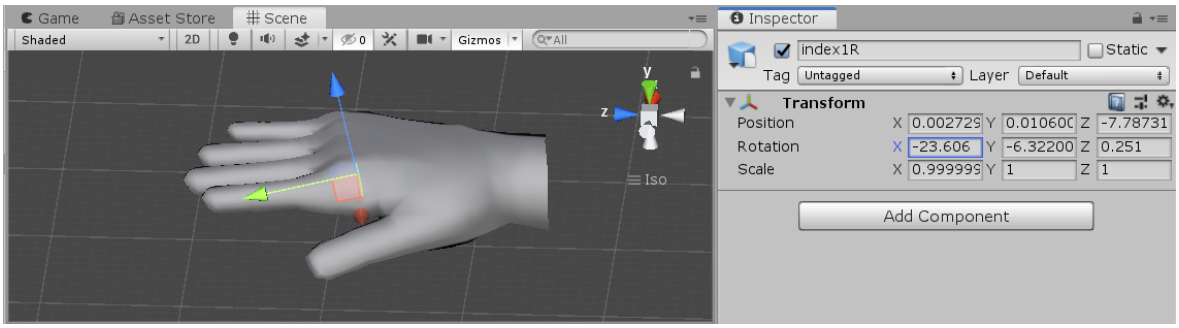

Figura 3-11.: Máxima rotación en el eje X del dedo índice en Unity

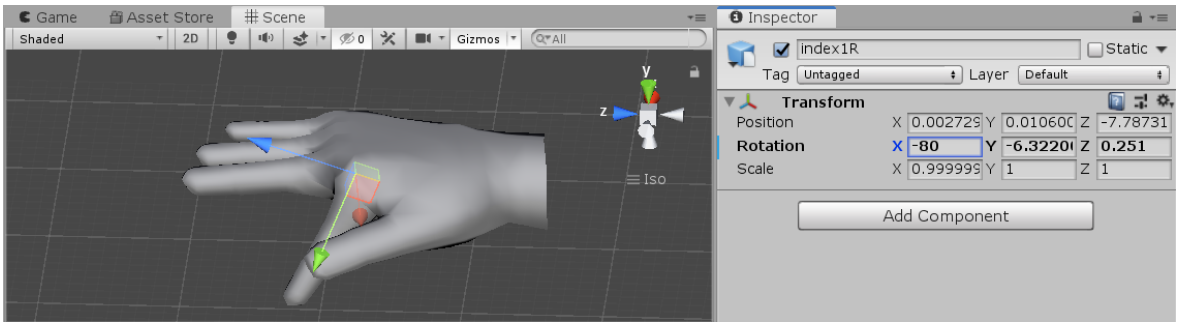

Figura 3-12.: Mínima rotación en el eje X del dedo índice en Unity

La fórmula de movimiento se establece como la ecuación de una recta donde x es el valor entregado por el sensor del módulo, y y es la rotación del objeto 3D en Unity en grados. Utilizando los datos mencionados anteriormente, se calculan las constantes m y b:

$$y = x * m + b$$

$$m = \frac{MinU - MaxU}{MinC - MaxC}$$

$$b = MaxU - m * MaxC$$

Es importante mencionar que el antebrazo, la mano y cada uno de los dedos en el modelo 3D, son tratados como objetos independientes en la escena de Unity, esto significa, que la fórmula de movimiento debe ser calculada para cada objeto y así asignar la rotación correcta de acuerdo con el valor sensado por los módulos. Por ejemplo, si para el dedo índice los valores máximos y mínimos de conductividad del flexómetro luego de la calibración son:

$$MaxC = 3231{,}921[S/m]$$

$$MinC = 2{,}457[S/m]$$

Y de acuerdo con las figuras **3-11** y **3-12**, los límites de rotación en el eje X para el modelo del dedo índice en Unity son:

$$MaxU = -23{,}606^{o}$$

$$MinU = -80^{o}$$

Entonces se obtiene la siguiente fórmula de movimiento:

$$y = x * 0,017 - 80,043$$

La gráfica de esta recta se presenta en la figura **3-13**, donde se conoce la rotación del modelo del dedo índice para cada valor que toma el flexómetro en cualquier instante de tiempo, sin sobrepasar los límites establecidos. Adicionalmente, la misma fórmula de movimiento debe ser asignada a cada una de las falanges del dedo, para así lograr una simulación más fiel al movimiento natural de las manos, donde las tres falanges rotan simultáneamente al cerrar la mano en puño.

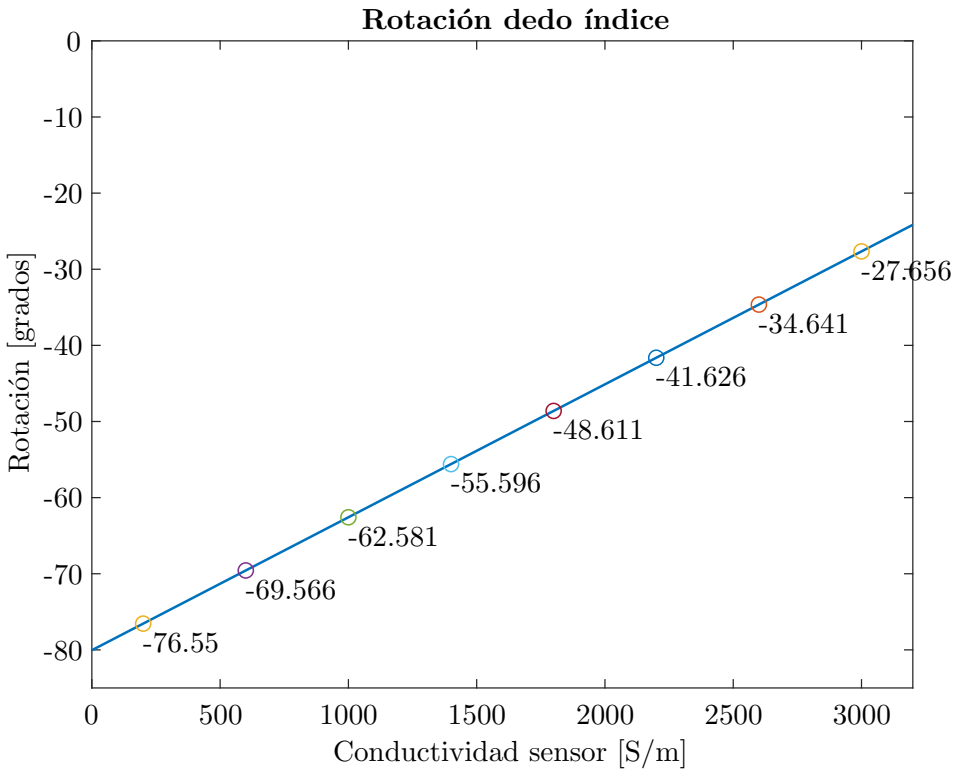

Figura 3-13.: Rotación del dedo índice para cada valor del flexómetro

3.1.3. Limitaciones

Es importante mencionar que los dispositivos Captoglove son productos que aún se encuentran en desarrollo y el proveedor entrega periódicamente actualizaciones de *firmware* con versiones más estables a las anteriores. Esto quiere decir, que aún existen fallas tanto de *hardware* como de *software* que impiden que los guantes puedan ser utilizados para crear una solución completa. Los dos principales problemas que se encontraron durante el desarrollo de este proyecto son:

- A través de su página web, Captoglove ofrece sus productos con compatibilidad para diferentes sistemas operativos: Windows, Android y iOS. Sin embargo, el SDK para Android es una versión Beta no estable y el *plugin GSdkNet.BLE.Droid.dll* genera errores durante la compilación de una aplicación en Unity, esto quiere decir, que actualmente las aplicaciones con guantes Captoglove no pueden ser exportadas a teléfonos inteligentes Android. El problema fue reportado a Captoglove pero aún no hay una solución por parte del proveedor.
- Una falla en el SDK para Unity causa que la plataforma se congele cuando la aplicación es detenida, incluso cuando se ejecuta desde la interfaz del Editor, esto quiere decir, que se debe forzar el reinicio de Unity luego de cada ejecución de la aplicación, limitando así la velocidad de desarrollo y pruebas para el programador. Este problema ha sido reportado múltiples veces a Captoglove y de acuerdo con su revisión es causado a una falla en el SDK cuando intenta detener la comunicación Bluetooth de los módulos con la aplicación. Sin embargo, aún no hay una solución por parte del proveedor.

En la siguiente sección se describe la librería desarrollada para integrar los guantes Captoglove a cualquier aplicación Unity.

3.2. Descripción de la librería GITEICaptoglove

Una vez conocidas las posibilidades con los *CaptoGloves* y los *CaptoSensors*, se compararon algunos entornos de entrenamiento de trabajadores en el sector textil, petrolero y eléctrico, con el fin de encontrar elementos e interacciones en común que fueron la base para el desarrollo de la librería *GITEICaptoglove*.

Para la caracterización del entorno en el sector textil, se contó con la colaboración de un trabajador de la empresa Protela S.A en el área de estampado. El entrenamiento en esta área consiste principalmente en observar el funcionamiento de las máquinas de estampado durante un par de semanas, e intervenir en el proceso únicamente bajo la orden y supervisión de un superior. Las intervenciones pueden ser para configurar los parámetros de la máquina o para ubicar la tela a estampar. Las máquinas se configuran mediante tableros de control digitales e interruptores. El principal riesgo para el operador cuando se encuentra muy cerca a la maquinaria, es el atrapamiento de manos y objetos colgantes.

En el sector petrolero se caracterizó un entorno de entrenamiento para los cuñeros[4] de un pozo. Su capacitación consiste principalmente en la adopción de medidas y elementos de seguridad, y la asignación de tareas de menor a mayor dificultad progresivamente. Sin embargo, la única forma de aprender la manipulación de las tuberías es mediante la práctica en campo. Para una correcta manipulación, las manos del operario deben posicionarse en un lugar específico sobre la tubería durante muy poco tiempo, en sincronización con la rotación de la misma. En la plataforma del pozo se encuentra maquinaria pesada, a altas temperaturas, en rotación y movimiento, que representan un constante peligro para los trabajadores. Esta caracterización se realizó con la ayuda de una extrabajadora de la petrolera Chevron Corporation en el área de perforación.

Por último, para caracterizar el entorno en el sector eléctrico, se tomó como referencia el manual *Maniobras en sistemas de distribución de redes de energía eléctrica*, proporcionado por la empresa Enel-CODENSA al grupo GITEI [6]. Allí se describe el proceso para reemplazar un Dispositivo de Protección de Sobrecarga (DPS) defectuoso en campo. La secuencia de pasos incluye la manipulación de herramientas específicas para energizar y desenergizar el dispositivo, así como de llaves para montar y desmontar el DPS del poste de energía. El mayor riesgo para el operador en este caso está en las descargas eléctricas y posible caída de alturas.

Luego del análisis de los tres escenarios anteriores, y de evaluar la posibilidad de transformarlos en sistemas seguros de entrenamiento mediante realidad virtual y aumentada, se encontró que para simular total o parcialmente entornos de capacitación industrial en general, es necesario rastrear la posición de las manos del aprendiz en todo momento para poder alertar de posibles peligros en la escena, permitir la interacción con tableros de control, botones e interruptores, al igual que detectar gestos de la mano que puedan ser interpretados como la acción de agarrar llaves y herramientas. En base a esto, se diseñaron las funciones de la librería *GITEICaptoglove* y los gestos de interacción descritos a continuación.

3.2.1. Gestos definidos

Los siguientes son los movimientos de la mano que se definieron para permitir a los usuarios de los guantes Captoglove interactuar adecuadamente con objetos virtuales en un sistema de realidad mixta enfocado al entrenamiento industrial. Dichos gestos pueden ser visualizados en la figura **3-14**.

[4]El cuñero, o *Floor Hand*, es el miembro de la cuadrilla de perforación que efectúa principalmente conexiones y desconexiones durante los viajes de la tubería en un pozo petrolero [24]

1. **Mano cerrada en puño:** este movimiento ocurre al flexionar todos los dedos de la mano y puede ser interpretado principalmente como la acción de agarrar y arrastrar un objeto en la escena.

2. **Mano abierta:** este movimiento se da al mantener estirados todos los dedos de la mano y puede ser interpretado como la acción de liberar o soltar un objeto en la escena.

3. **Gesto del número uno:** este movimiento consiste en mantener el dedo índice estirado mientras los demás dedos se encuentran completamente flexionados. La acción asociada a este gesto debe ser definida por el desarrollador de acuerdo a la aplicación.

4. **Gesto del número dos:** este movimiento se da cuando el dedo índice y el dedo medio se mantienen estirados mientras los demás están totalmente flexionados. La acción asociada a este gesto debe ser definida por el desarrollador de acuerdo a la aplicación.

5. **Gesto del número tres:** este movimiento consiste en mantener los dedos índice, medio y anular estirados mientras los demás se encuentran completamente flexionados. La acción asociada a este gesto debe ser definida por el desarrollador de acuerdo a la aplicación.

6. **Oprimir sensor de presión:** esta acción ocurre el oprimir el sensor de presión ubicado en la punta del dedo pulgar de los *CaptoGloves*, y puede ser interpretada principalmente como la fuerza ejercida sobre un objeto o un botón en la escena.

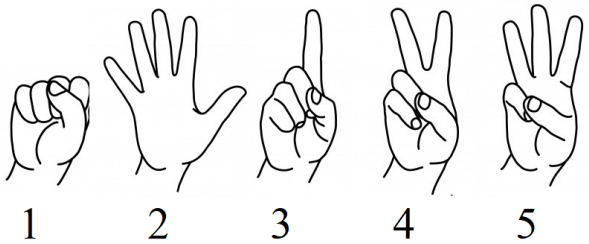

Figura 3-14.: Gestos de interacción: mano cerrada (1), mano abierta (2), gesto del número uno (3), gesto del número dos (4), gesto del número tres (5) [8]

3.2.2. Descripción de las clases

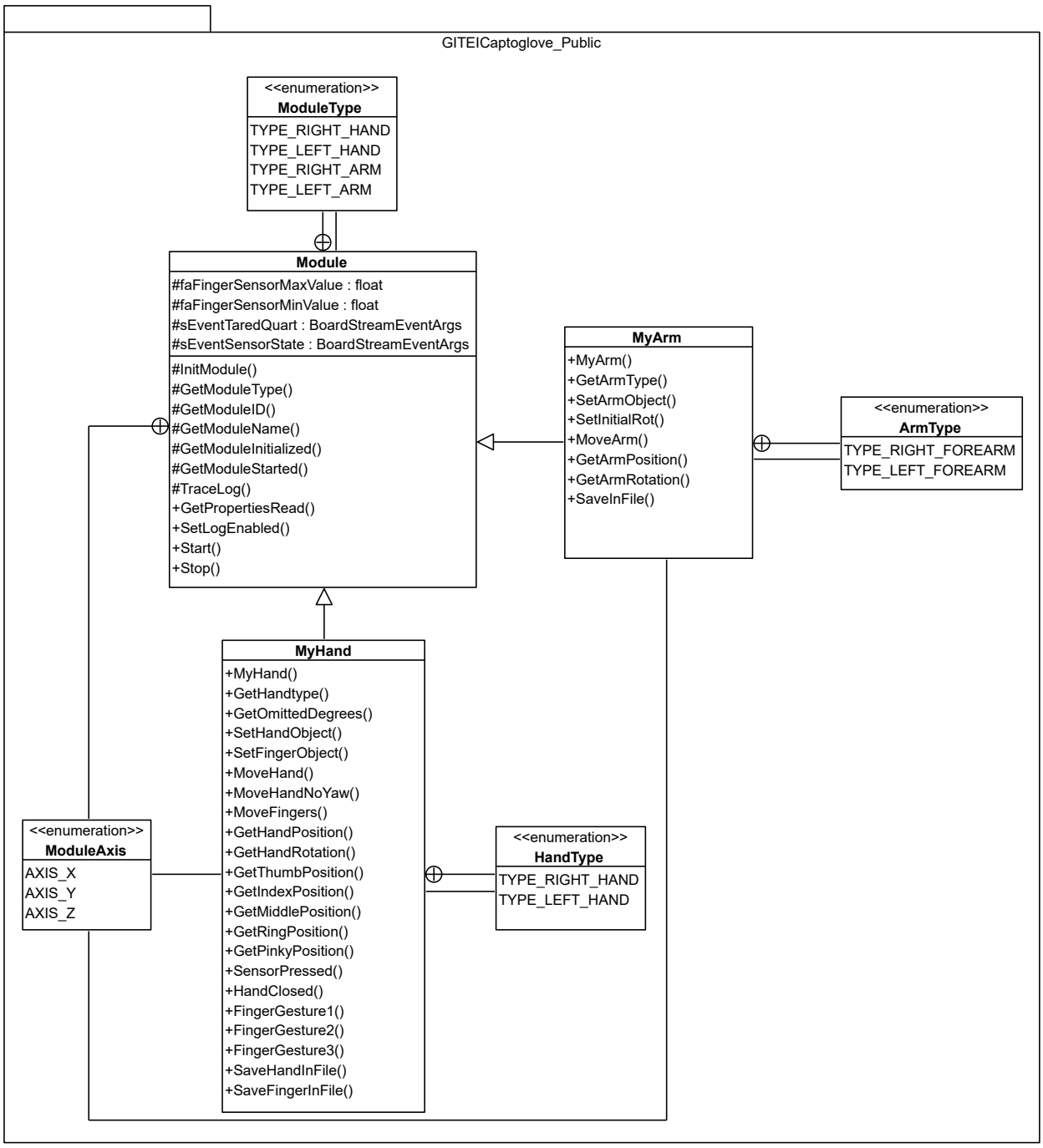

Figura 3-15.: Diagrama de clases de la librería *GITEICaptoglove* (simplificado)

La librería DLL *GITEICaptoglove* fue diseñada e implementada en este proyecto para integrar rápidamente los módulos Captoglove a cualquier aplicación Uniy; esta debe ser utilizada como *plugin* para Unity en conjunto con el SDK de Captoglove: *GSdkNet.BLE.Winapi* y *GSdkNet*. La biblioteca está compuesta por tres clases principales: *Module, MyHand* y *MyArm*, donde las dos últimas son clases que heredan atributos y métodos de la clase principal *Module*. En la figura **3-15** se presenta el diagrama de clases para esta librería incluyendo solamente los atributos y métodos públicos para una mejor visualización. El diagrama de clases completo se encuentra disponible en el anexo A.5 de este documento.

En la librería se utilizó un diseño modular que permite conectar uno o varios módulos Captoglove a la aplicación de Unity de forma independiente, pero pueden trabajar en conjunto si todos se encuentran encendidos durante la ejecución de la aplicación. A continuación se explican en detalle cada una de las clases, y algunos de sus métodos más relevantes, para ilustrar la forma en que estas simplifican la integración de los guantes al realizar internamente todo el procesamiento de los datos recibidos de los módulos Captoglove.

Module

Esta clase principal permite agregar, configurar, iniciar y detener el módulo Captoglove durante la ejecución de la aplicación. Sin embargo, el usuario de las librerías no puede hacer llamados directos a la mayoría de sus métodos, sino que debe acceder a ellos a través de las clases hijas *MyHand* y *MyArm*.

Para crear o agregar un módulo a la aplicación se debe utilizar el constructor de la clase *MyHand* o *MyArm*, el cual recibe como parámetros el ID del módulo Captoglove y el tipo de uso que se le dará en la aplicación, bien sea como sensor de mano derecha, mano izquierda, antebrazo derecho o antebrazo izquierdo. Como se muestra en el diagrama **3-16**, el constructor establece la configuración inicial del módulo de acuerdo con su modo de uso, incluyendo los límites de rotación para el modelo 3D de las manos o brazos establecido por defecto en Unity.

Para conectar el módulo con la aplicación se debe llamar la función *Start()*, de la clase principal *Module*, que ejecuta el procedimiento descrito en el diagrama **3-17**. Allí se utiliza internamente el SDK de Captoglove para buscar el ID del módulo dentro de los dispositivos Bluetooth conectados al computador, establecer la conexión con la aplicación, activar los sensores necesarios de acuerdo con el modo de uso escogido por el desarrollador, y finalmente capturar constantemente los datos de los sensores transmitidos por el módulo.

Al detener la ejecución de la aplicación Unity, es necesario también detener la transmisión de datos y desconectar los módulos Captoglove de la aplicación utilizando el método *Stop()* de la clase *Module*; esto para permitir que los módulos puedan ser conectados sin problema a otra aplicación posteriormente. En los anexos A.12 y A.13 se ilustra la forma en que ocurren los llamados a las funciones mencionadas anteriormente dentro de un *script* de Unity.

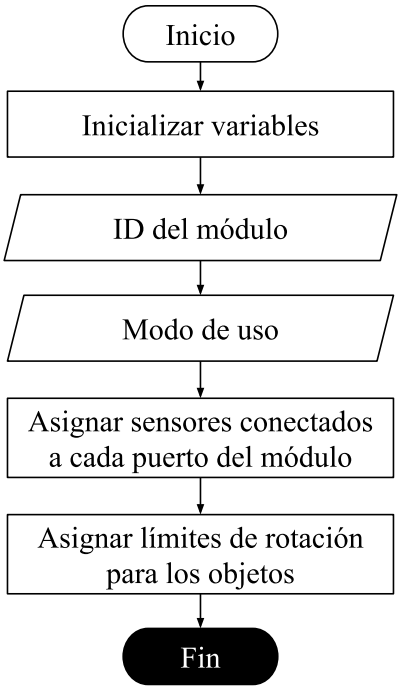

Figura 3-16.: Diagrama de flujo para *MyHand:MyHand() y MyArm:MyArm()*

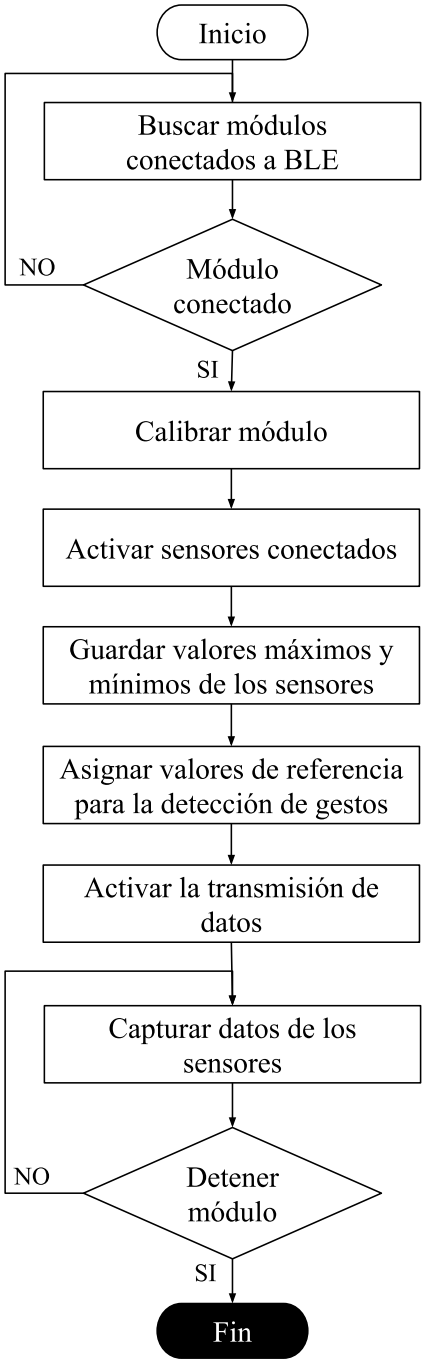

Figura 3-17.: Diagrama de flujo para *Module:Start()*

MyHand

Esta clase permite asignar y sincronizar el modelo 3D de las manos en Unity con el movimiento real de los *CaptoGloves*. Además, captura los gestos y movimientos del usuario definidos anteriormente y entrega esta información a la aplicación para que se tomen las acciones necesarias.

La detección de gestos de los dedos no requiere de un modelo 3D en la escena, lo que resulta particularmente útil en aplicaciones de realidad aumentada. Sin embargo, el *plugin GITEICaptoglove* contiene un modelo 3D predefinido de las manos, que puede ser incluido en cualquier aplicación Unity para sincronizar el movimiento real de las manos del usuario con el objeto en la escena. Para este modelo en específico, los ejes y limites de rotación son establecidos por defecto durante la configuración inicial del módulo, por lo que el usuario de las librerías simplemente debe asociar los objetos en la escena al módulo Captoglove utilizando los métodos *SetHandObject()* y *SetFingerObject()*.

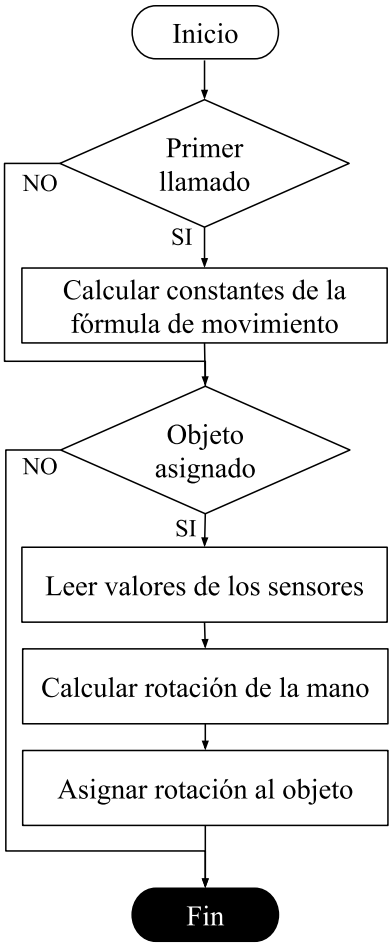

Figura 3-18.: Diagrama de flujo para *MyHand:MoveHand()*

Para capturar en tiempo real el movimiento de las manos y los dedos, se deben llamar los métodos *MoveHand()* y *MoveFingers()*, respectivamente, desde la función *Update()* del *script* de Unity para que se ejecuten en cada *frame* de la aplicación. En las figuras **3-18** y **3-19** se muestra el diagrama de flujo para estas dos funciones, en las que se procesan los datos de los sensores y se calcula la rotación de los objetos en la escena a través de la fórmula de movimiento. Las constantes de la fórmula de movimiento se establecen internamente en la librería *GITEICaptoglove* durante la configuración de los módulos, por lo que es un procedimiento transparente para el desarrollador.

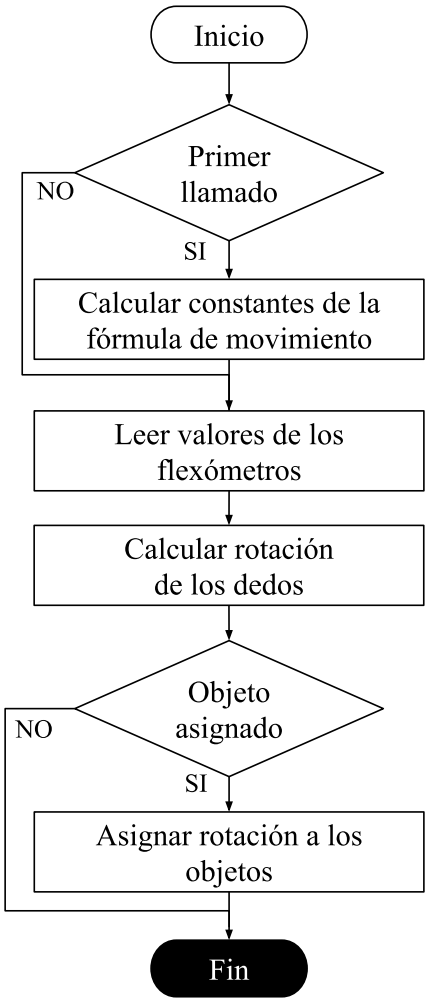

Figura 3-19.: Diagrama de flujo para *MyHand:MoveFingers()*

Por otra parte, por medio de las funciones *HandClosed()*, *FingerGesture1()*, *FingerGesure2()*, *FingerGesture3()* y *SensorPressed()*, la librería captura en cualquier momento los gestos de interacción del usuario definidos. Los diagramas **3-20** a **3-24** muestran el procedimiento interno de estas funciones, en las que constantemente se comparan los valores de los flexómetros y el sensor de presión contra el valor de referencia establecido durante la configuración de los módulos. Allí se evalúa el cumplimiento de las condiciones definidas para cada uno de los gestos, y se retorna un valor de falso o verdadero al usuario de las librerías para que pueda tomar las acciones necesarias. Si ninguno de los gestos listados anteriormente es detectado, la librería asume la mano del usuario como completamente abierta.

Para lograr la detección de gestos en tiempo real, estas funciones deben ser llamadas desde la función *Update()* del *script* de Unity. En el anexo A.12 se encuentra un ejemplo de un *script* donde se implementan las funciones mencionadas anteriormente para configurar un *CaptoGlove* como sensor de la mano derecha.

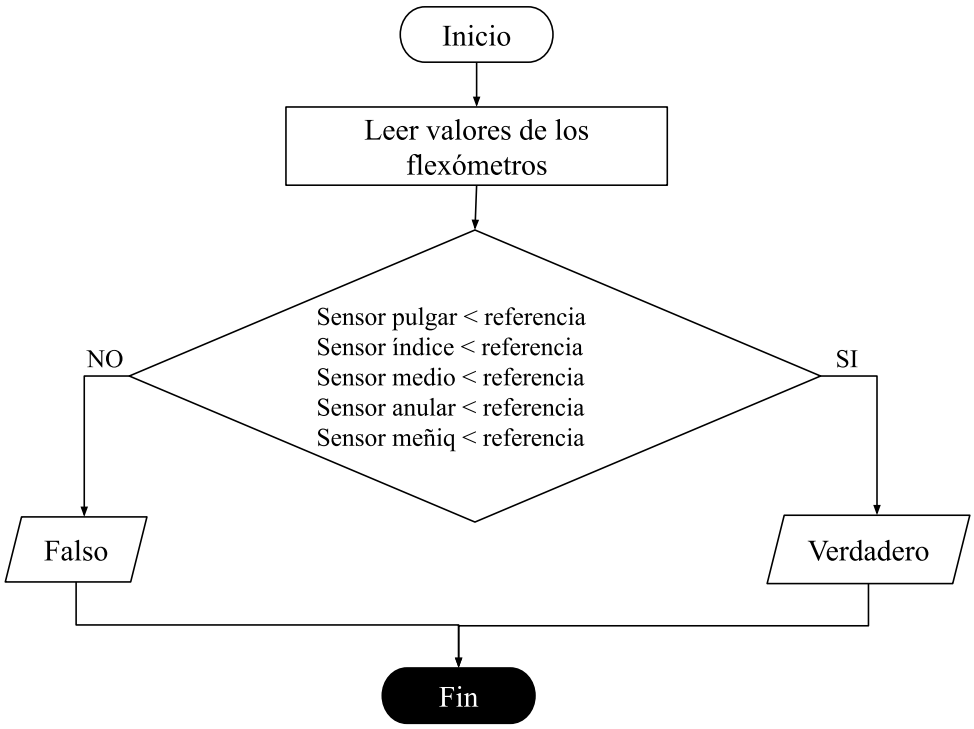

Figura 3-20.: Diagrama de flujo para *MyHand:HandClosed()*

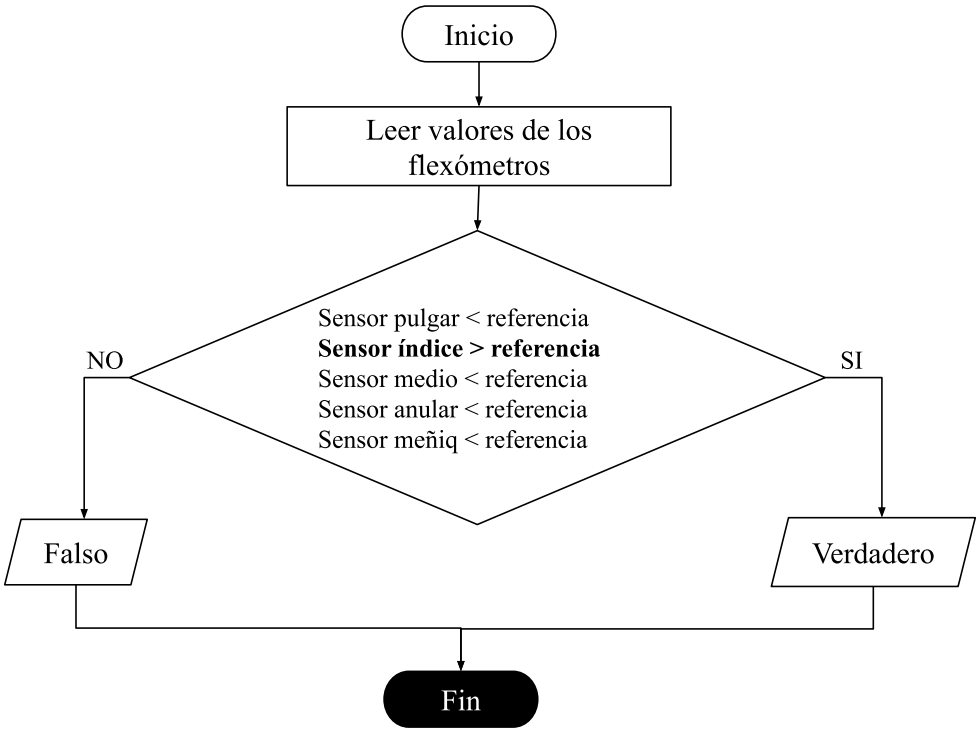

Figura 3-21.: Diagrama de flujo para *MyHand:FingerGesture1()*

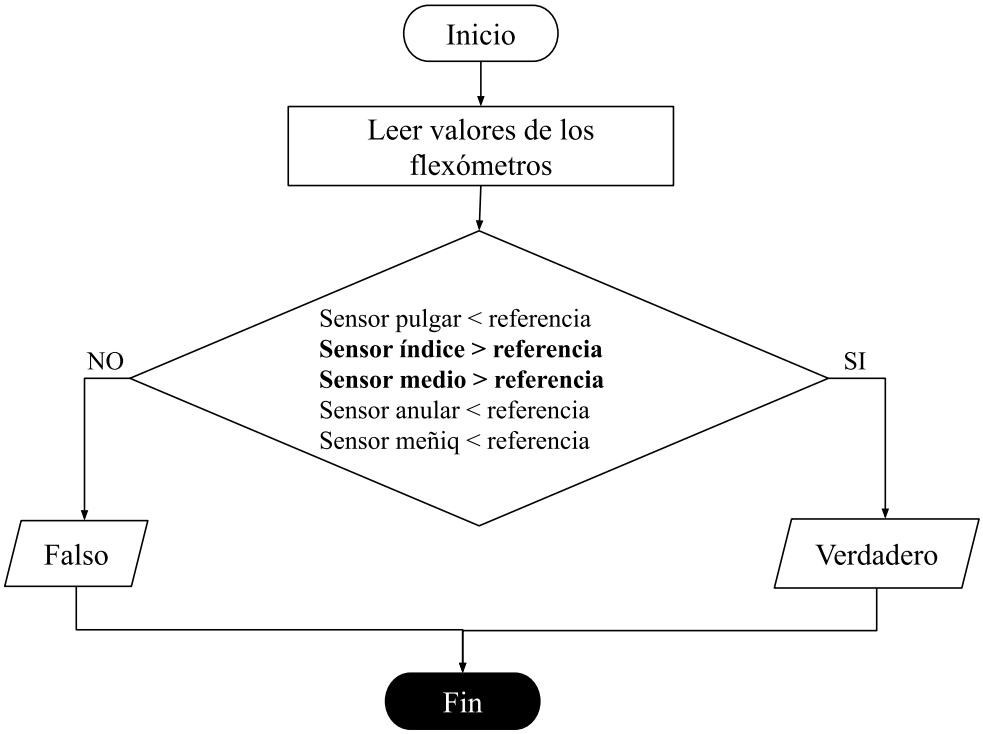

Figura 3-22.: Diagrama de flujo para *MyHand:FingerGesture2()*

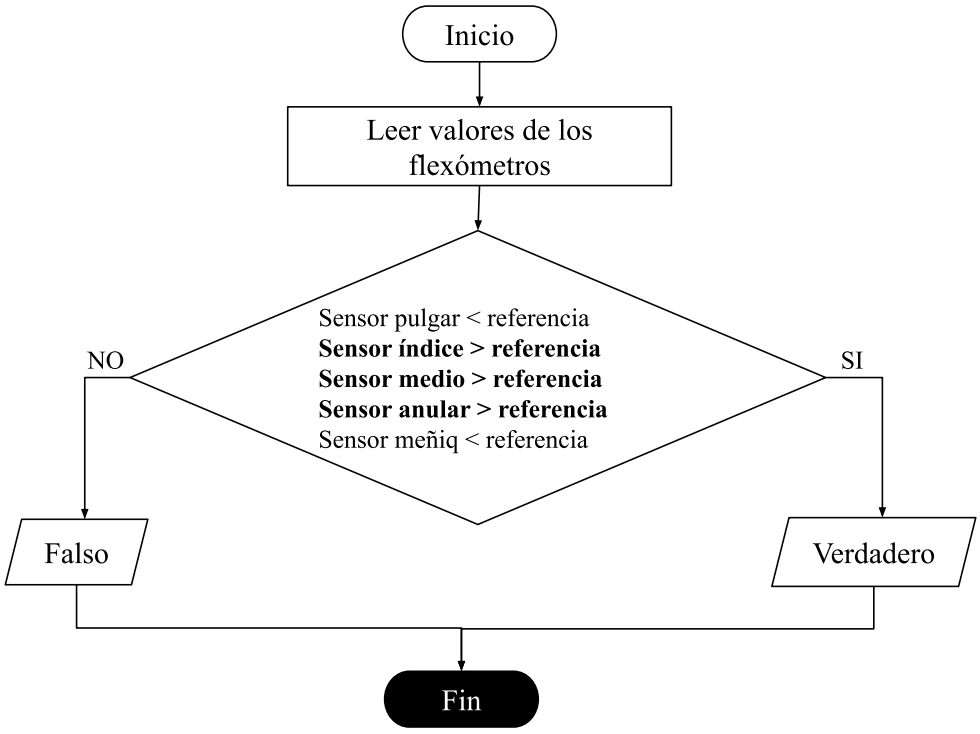

Figura 3-23.: Diagrama de flujo para *MyHand:FingerGesture3()*

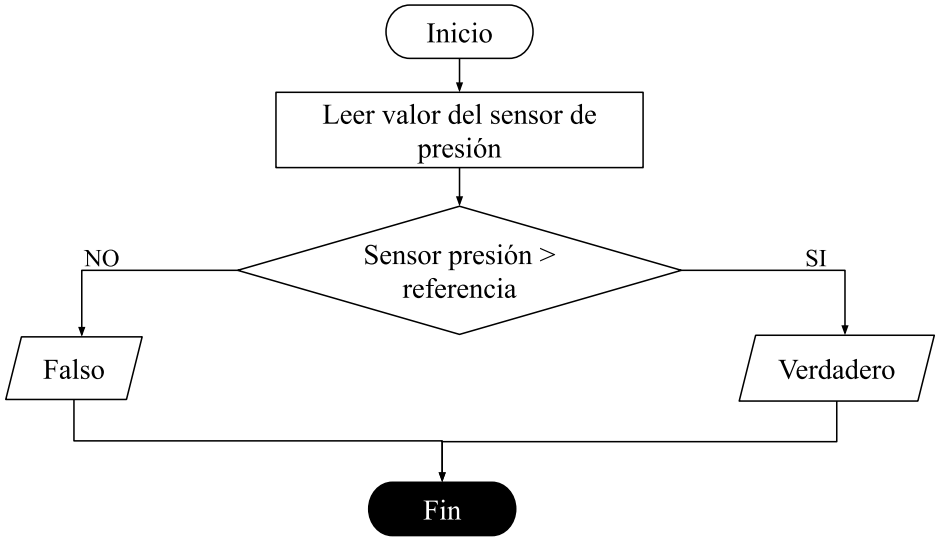

Figura 3-24.: Diagrama de flujo para *MyHand:SensorPressed()*

MyArm

Esta clase permite asignar y sincronizar el modelo 3D de los brazos en Unity al movimiento real de los *CaptoSensors*, ubicados en los antebrazos del usuario. El *plugin GITEICaptoglove* contiene un modelo 3D de los brazos que puede ser asociado al módulo utilizando el método *SetArmObject()*. Para sincronizar el movimiento de este objeto con el movimiento real del usuario, se debe llamar el método *MoveArm()* desde la función *Update()* del *script* de Unity para que se ejecute en cada *frame* de la aplicación. *MoveArm()* sigue el mismo procedimiento interno descrito en el diagrama **3-18** para la función *MoveHand()*.

Esta clase es de uso opcional si el desarrollador desea utilizar los *CaptoSensors* junto con los *CaptoGloves* en una misma aplicación; fue diseñada principalmente para ofrecer mayor movilidad a los usuarios en aplicaciones de realidad virtual, pues en las aplicaciones de realidad aumentada, el movimiento de los brazos no necesita ser simulado.

En el código del anexo A.13 se presenta un ejemplo del *script* de Unity para asignar un *CaptoSensor* como sensor del antebrazo derecho, haciendo los llamados correspondientes a las funciones de la librería *GITEICaptoglove* mencionadas. Cada módulo debe ser creado desde un *script* independiente de Unity, esto quiere decir, que a una misma aplicación se pueden agregar hasta cuatro *scripts* para utilizar todos los módulos Captoglove a la vez.

3.3. Resumen del capítulo

Los guantes hápticos Captoglove, adquiridos para este proyecto, entregan información a bajo nivel acerca de la rotación de las manos y dedos del usuario, que permite crear interacciones con objetos virtuales dentro de aplicaciones de realidad virtual y aumentada. La librería desarrollada, *GITEICaptoglove*, se encarga de configurar los módulos Captoglove, capturar los datos de lo sensores y detectar los gestos definidos de forma transparente para el desarrollador. En el siguiente capítulo se presentan los resultados de este proyecto, incluyendo tres aplicaciones de ejemplo con diferentes propósitos en las que se implementó la librería *GITEICaptoglove*; en una de las aplicaciones se incluyeron los *CaptoSensors* como sensores de los antebrazos del usuario. Con esto se busca demostrar que la configuración de los módulos, a través de la librería, funciona correctamente independiente del entorno al que se integran, por lo que es posible crear escenarios para la capacitación de trabajadores en diferentes industrias, simplemente modificando las acciones ejecutadas por los guantes de acuerdo con los requerimientos de la aplicación.

4. Resultados

En esta sección se presentan los resultados de este proyecto luego del desarrollo de la librería de integración *GITEICaptoglove* y se describen los entregables generados con el producto final:

- Librería DLL para Unity: *GITEICaptoglove*
- Documentación
- Código fuente
- Modelo 3D de manos y brazos
- Tres aplicaciones de ejemplo
- Instrucciones para configurar y calibrar módulos Captoglove

4.1. Librería DLL

La biblioteca *GITEICaptoglove* contiene las siguientes tres clases:

1. ***Module*:** permite configurar, conectar y detener los módulos Captoglove durante la ejecución de la aplicación en Unity. Allí se integra el SDK de Captoglove para controlar la conexión Bluetooth de los módulos y la transmisión de datos de los sensores.
2. ***MyHand***: permite sincronizar el modelo 3D de las manos en Unity con el movimiento real de los *CaptoGloves* y captura los gestos de interacción definidos de acuerdo con el movimiento de las manos del usuario.
3. ***MyArm***: permite sincronizar el modelo 3D de los brazos en Unity al movimiento real de los *CaptoSensors* ubicados en los antebrazos del usuario.

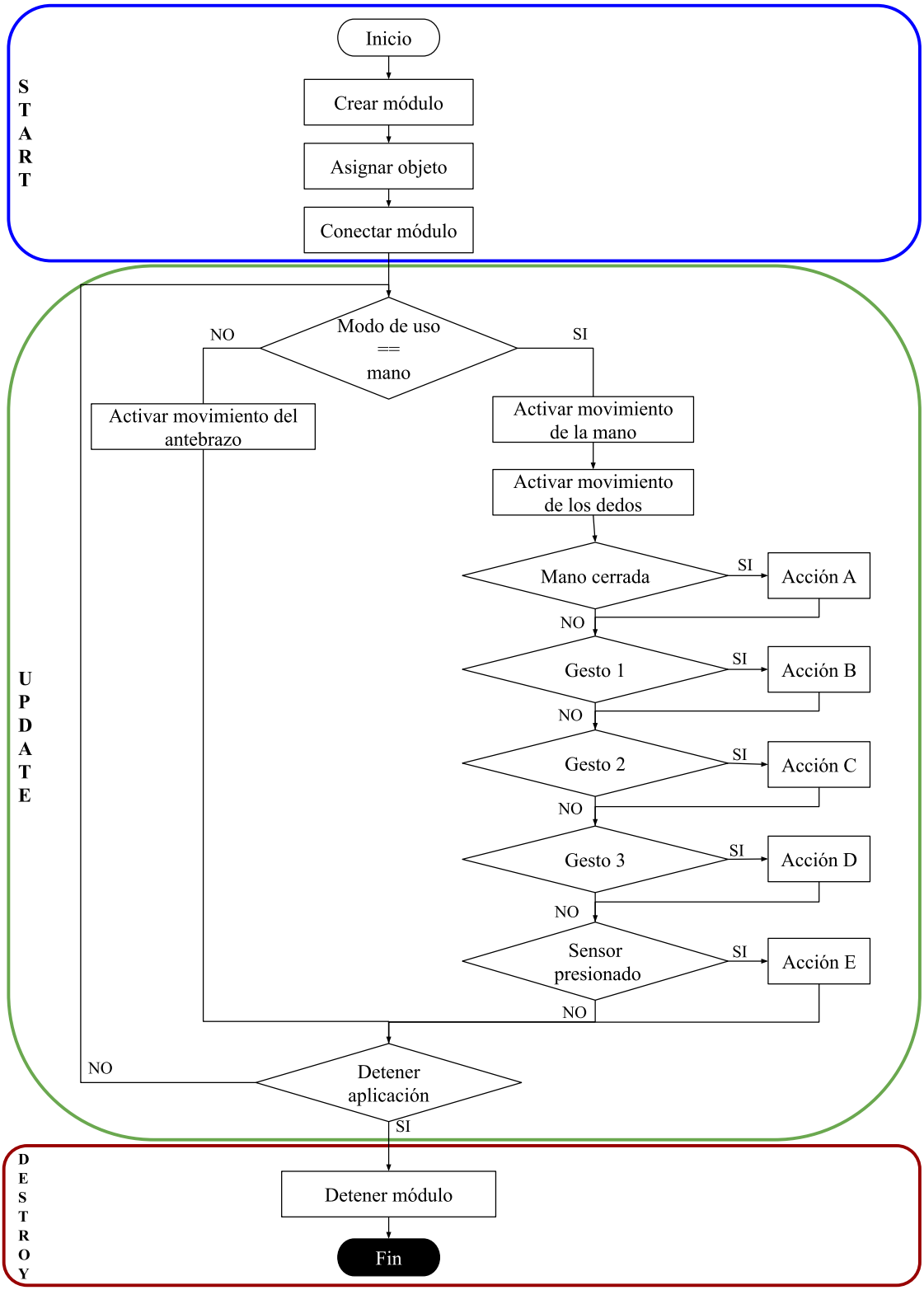

Figura 4-1.: Diagrama de flujo para un desarrollador

El diagrama de flujo de la figura **4-1** es un ejemplo de la forma en que la librería *GITEI-Captoglove* debe ser utilizada dentro de un *script* de Unity; el código de implementación se encuentra en los anexos A.12 y A.13 de este documento. Gracias a la estructura modular de la librería, el desarrollador puede definir el número de módulos Captoglove a utilizar en una misma aplicación, las sincronizaciones de movimiento activas y los gestos de la mano por capturar de acuerdo con los requerimientos del entorno.

Siguiendo las instrucciones del diagrama **4-1**, en la función *Start()* del *script* se debe crear un nuevo módulo, asociarlo con su respectivo objeto 3D en la escena (opcional) y conectarlo a la aplicación a través de comunicación Bluetooth. En la función *Update()* del *script*, que es llamada cada *frame* durante la ejecución de la aplicación, se debe activar la sincronización de brazos, manos y dedos para que los modelos 3D en la escena roten de acuerdo con el movimiento del usuario en tiempo real, y se deben capturar los gestos de interacción para ejecutar las acciones definidas en el momento correcto. Por último, en la función *OnDestroy()* del *script*, que se ejecuta solamente al detener la aplicación Unity, se debe interrumpir la transmisión de datos y la comunicación Bluetooth de los módulos con la aplicación. Sin embargo, como se mencionó dentro de las limitaciones de este proyecto, el SDK de Captoglove presenta un error durante este proceso y eventualmente se debe forzar la terminación de la aplicación y del Editor de Unity, desde el administrador de tareas de Windows.

4.2. Documentación y código fuente

La librería *GITEICaptoglove* se entrega con su respectiva documentación y con la documentación del SDK recibida por parte del proveedor. Ver anexos A.6 y A.7. Adicionalmente, se suministra el código fuente de la librería para que pueda ser modificado y utilizado en futuros proyectos relacionados. Ver anexo A.4.

4.3. Modelos 3D

Con el fin de agilizar el desarrollo de aplicaciones Unity, la librería *GITEICaptoglove* se entrega junto con un modelo 3D para Unity que incluye todas las articulaciones de antebrazos, manos y dedos con sus respectivas falanges. Al ser un modelo conocido, sus ejes y límites de rotación son asignados por defecto durante la configuración de un nuevo módulo Captoglove en la aplicación. De esta manera, el usuario de la librería simplemente debe asociar los objetos de las manos a los *CaptoGloves* y los objetos de los antebrazos a los *CaptoSensors* si desea sincronizar la rotación de los modelos con el movimiento real del usuario durante la ejecución de la aplicación.

Como se muestra en la figura **4-2**, los modelos de los antebrazos, *LowArmL* y *LowArmR*, están presentes dentro del objeto principal *Mano_Izquierda* y *Mano_Derecha* respectivamente, pero no contienen material sólido, por lo que solo las manos son visibles en la escena de Unity. Los modelos fueron creados en la herramienta de diseño Blender por integrantes del grupo GITEI y se encuentran disponibles en el anexo A.11 de este documento. Adicionalmente, en el vídeo del anexo A.19 se demuestra la sincronización del movimiento de las manos del usuario con los objetos 3D en la escena de Unity, por medio de la librería *GITEICaptoglove*.

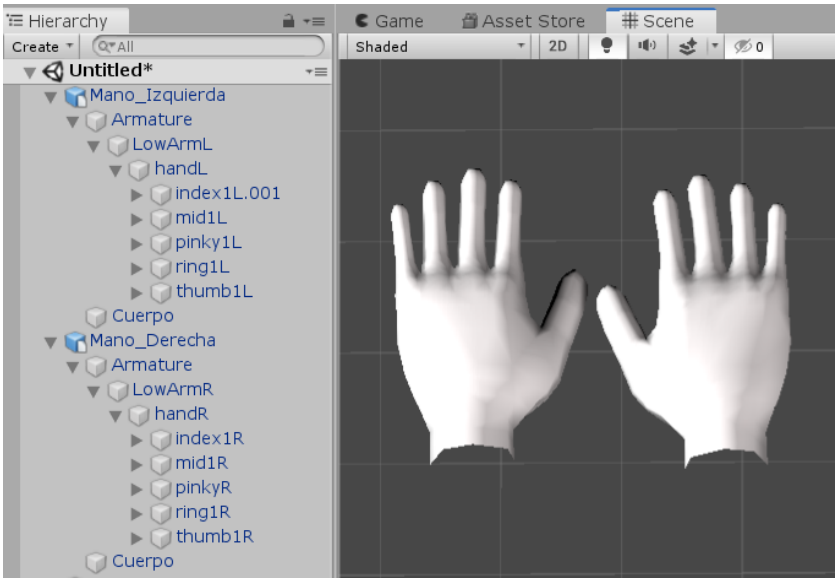

Figura 4-2.: Modelo 3D de las manos en escena de Unity

4.4. Aplicaciones de ejemplo

4.4.1. Aplicación en realidad virtual

El escenario presentado en la figura **4-3** permite al usuario probar fácilmente todos los movimientos definidos para interactuar con un entorno de realidad virtual utilizando los guantes hápticos Captoglove. Por defecto, la aplicación se conecta tanto con los *CaptoGloves* como con los *CaptoSensors*. Sin embargo, el número de módulos a utilizar es configurable y la aplicación funciona incluso cuando solo uno de ellos se encuentra encendido. Esta aplicación Unity se encuentra disponible en el anexo A.3, con su respectivo vídeo de funcionamiento en el anexo A.18.

En esta aplicación el usuario puede mover y rotar la vista de la cámara utilizando el teclado del computador en donde se ejecuta la aplicación; las manos en la escena se mueven junto con la cámara en todo momento. Una vez las manos se encuentran lo suficientemente cerca de cada objeto resaltado en rojo, se permiten las siguientes interacciones:

1. Atrapar la cápsula sobre la mesa cerrando la mano en puño y arrastrarla hasta cualquier lugar en la escena. La cápsula es liberada cuando se abre la mano nuevamente. Ver Figura **4-4**.

2. Oprimir el botón en la pared presionando el sensor de presión del dedo pulgar para cambiar la posición de la repisa que se encuentra a su lado derecho.

3. Cambiar el color de la esfera a azul, amarillo o verde realizando gestos con la mano del número uno, dos y tres respectivamente. Ver figura **4-5**.

4. Mover la palanca en la pared para cambiar la posición de la puerta a su lado izquierdo. La mano se debe cerrar en puño para atrapar la perilla y arrastrarla hacia abajo.

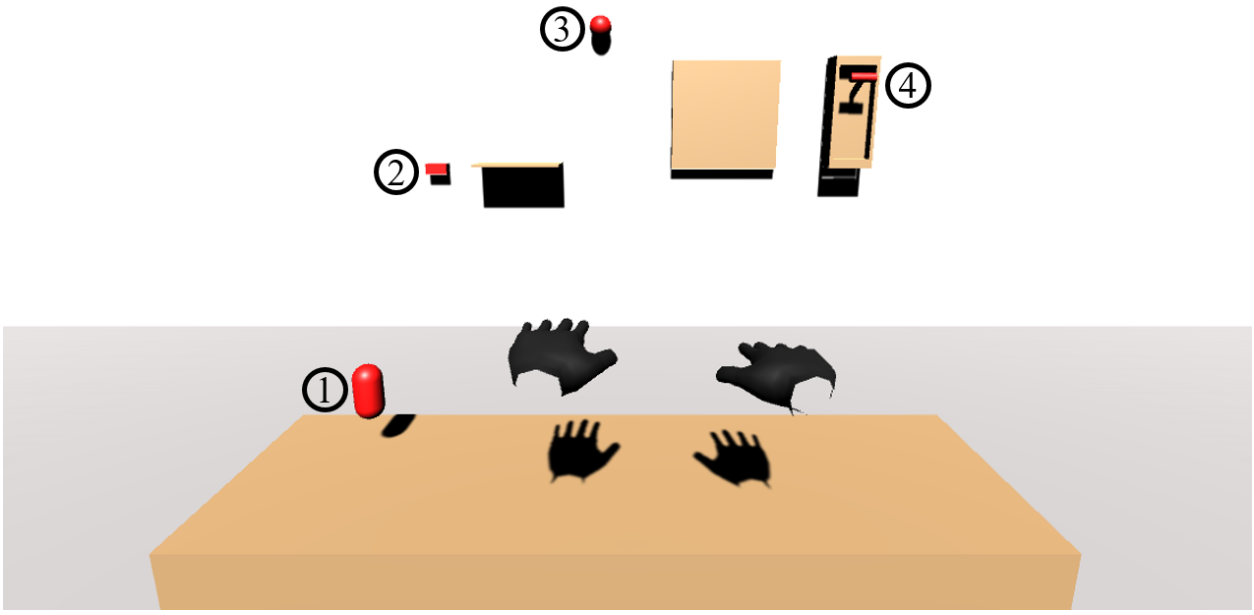

Figura 4-3.: Aplicación en realidad virtual

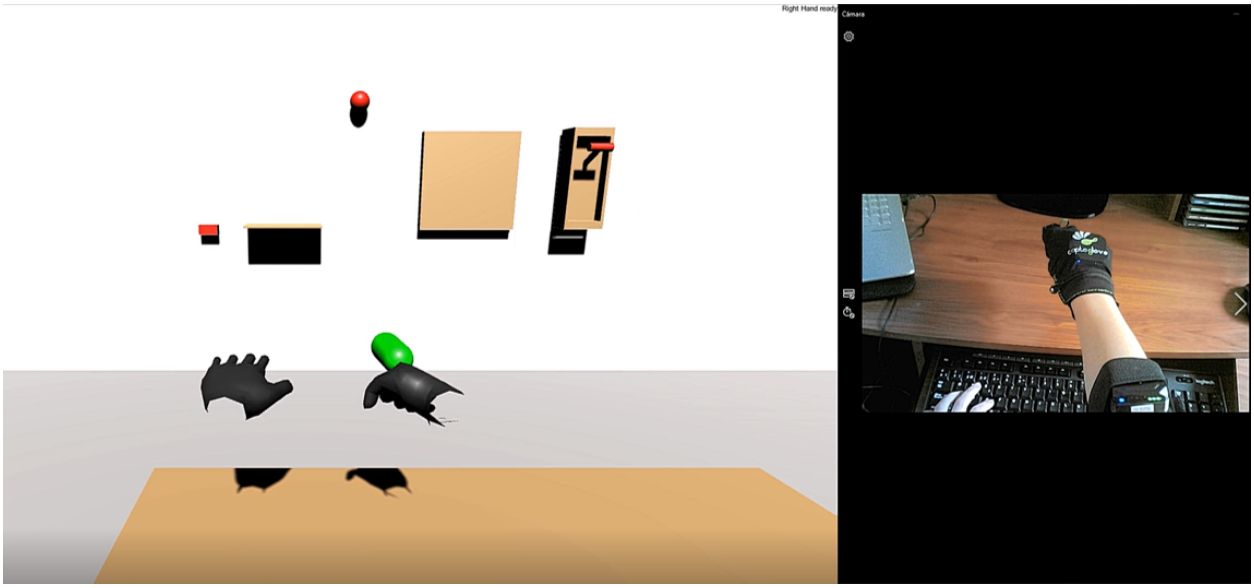

Figura 4-4.: Atrapar y arrastrar la cápsula cerrando la mano

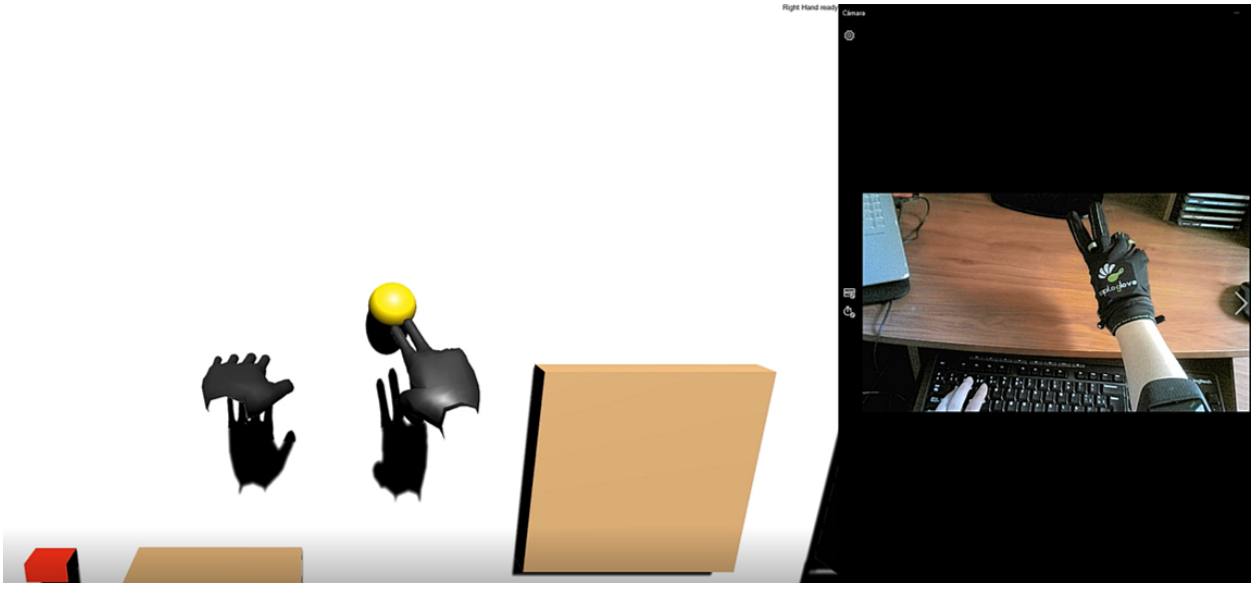

Figura 4-5.: Cambiar color de la esfera mediante gestos

4.4.2. Aplicación en realidad aumentada

Al igual que el escenario anterior, esta aplicación permite probar fácilmente todos los gestos definidos para interactuar con un entorno en realidad aumentada a través de los guantes Captoglove. Por defecto, esta aplicación se conecta únicamente al *CaptoGlove* configurado como la mano derecha del usuario. Sin embargo, puede ser reemplazado por el sensor de la mano izquierda si el usuario de la librería lo desea. Esta aplicación Unity se encuentra disponible en el anexo A.2, con su respectivo vídeo de funcionamiento en el anexo A.17

Al ser una aplicación en realidad aumentada, se requiere ubicar diferentes *targets* en el mundo real sobre los cuales se dibujan los elementos virtuales. Como se observa en la figura **4-6**, para esta aplicación se crearon cinco códigos QR: cuatro de ellos representan los objetos con los que el usuario puede interactuar y uno adicional para rastrear la posición de la mano derecha en la escena. En esta aplicación no es necesario simular el movimiento de los antebrazos ya que la posición de la mano con respecto a su entorno se conoce en todo momento rastrando el *target* correspondiente. Cuando la mano derecha está lo suficientemente cerca de cada uno de los objetos, las interacciones permitidas son:

1. Atrapar la cápsula cerrando la mano en puño y arrastrarla hasta cualquier lugar al rededor de su *target*. La cápsula es liberada al abrir la mano nuevamente.

2. Oprimir el botón presionando el sensor de presión del dedo pulgar para cambiar la posición de la puerta a su lado izquierdo. Ver figura **4-7**.

3. Cambiar el color de la esfera a azul, amarillo o verde realizando gestos con la mano del número uno, dos y tres respectivamente.

4. Mover la palanca cerrando la mano en puño para atrapar la perilla y arrastrarla hacia abajo. Ver Figura **4-8**.

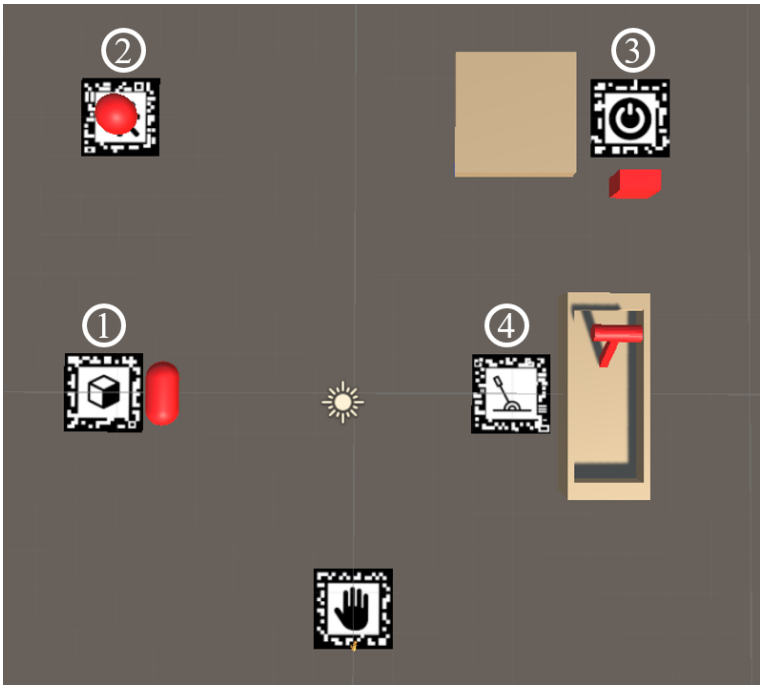

Figura 4-6.: Aplicación en realidad aumentada

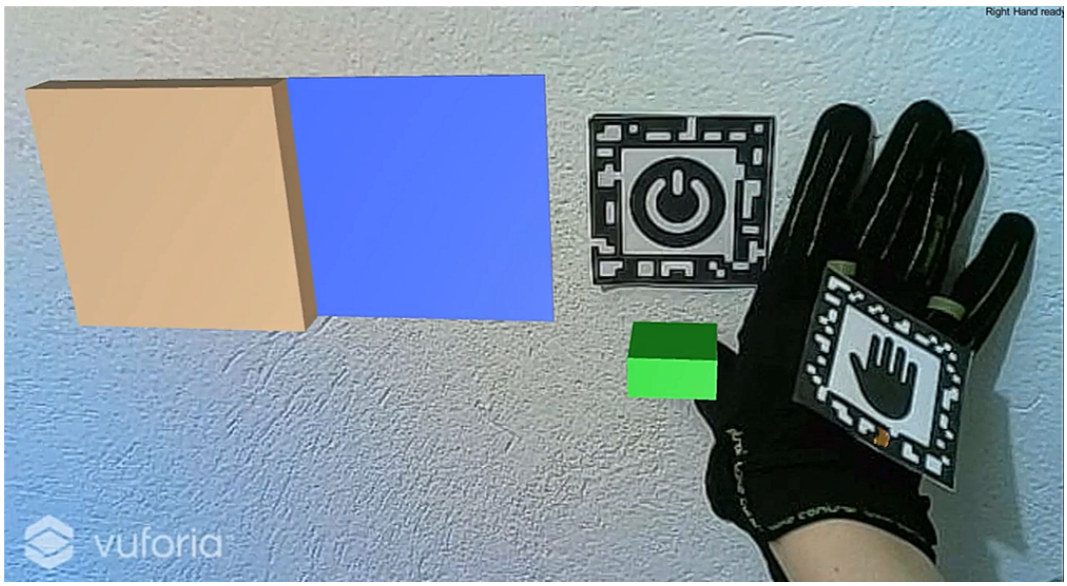

Figura 4-7.: Oprimir botón con sensor de presión

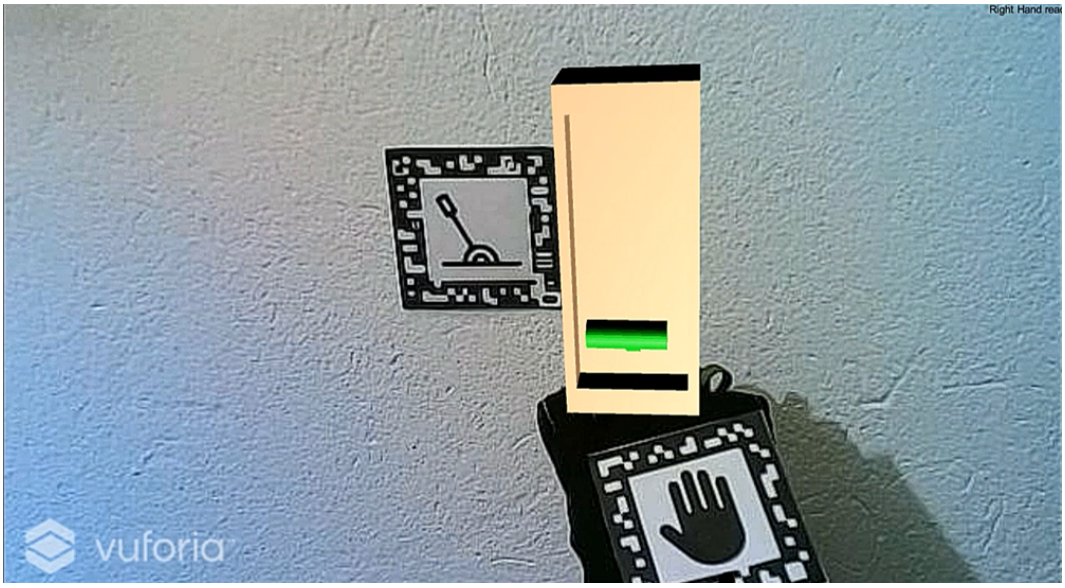

Figura 4-8.: Atrapar y mover palanca cerrando la mano en puño

4.4.3. Simulación de DPS

Figura 4-9.: Entrenamiento real para la manipulación de un DPS [6]

Como se mencionó en el capítulo anterior, uno de los escenarios de entrenamiento de trabajadores que se analizó para el desarrollo de la librería *GITEICaptoglove* fue en el sector eléctrico, con base en el manual de instrucciones proporcionado por Enel-Codensa para el cambio de un Dispositivo de Protección de Sobrecarga (DPS). Los DPS son dispositivos que se encuentran en la parte superior de los postes de la red de energía eléctrica y deben ser desenergizados por los operarios de mantenimiento antes de manipularlos. La figura **4-9** es una foto real del escenario de entrenamiento, donde los dispositivos son ubicados a nivel de piso para disminuir los riesgos de alturas a los que se exponen los trabajadores durante una situación real [6].

Tomando esta información como referencia, se creó una aplicación en realidad aumentada sencilla que ejemplifica una de las posibilidades que se tienen con los guantes Captoglove para interactuar en un entorno de entrenamiento de este tipo. Como se muestra en la figura **4-10**, en este escenario se utilizan tres *targets*: el primero de derecha a izquierda, para rastrear la posición de la mano del usuario en la escena, el siguiente para ubicar una palanca de energización y el tercero para simular un DPS; las líneas amarillas que salen del DPS representan el campo eléctrico presente a su al rededor mientras este se encuentra energizado. La aplicación Unity de este ejemplo se encuentra disponible en el anexo A.1, con su respectivo vídeo de funcionamiento en el anexo A.16.

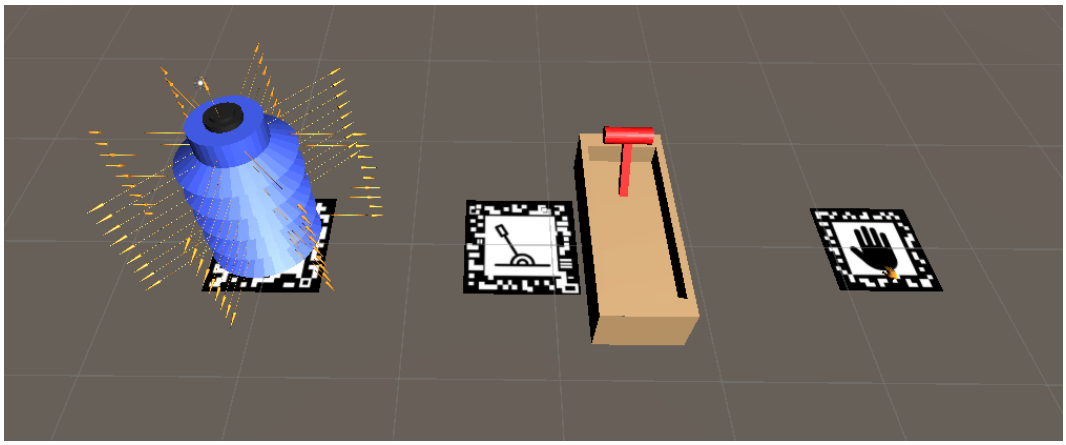

Figura 4-10.: Aplicación en realidad aumentada: simulación de DPS

Como se observa en la figura **4-11**, cuando el *target* del DPS es detectado por la cámara, la aplicación lo aumenta simulando el dispositivo junto con el campo eléctrico que se genera a su alrededor mientras se encuentra energizado. En esta condición, si la mano del usuario se acerca demasiado al DPS, la aplicación despliega un mensaje de alerta en la esquina superior izquierda de la pantalla. Para evitar esto, el usuario puede utilizar los guantes Captoglove e interactuar con la palanca virtual en la escena para desenergizar el DPS como se muestra en la figura **4-8**; bajo estas condiciones, la mano del usuario ahora puede acercarse e interactuar con el dispositivo sin restricciones.

Para lograr la interacción del guante háptico con la palanca, sin ayuda de la librería *GITEI-Captoglove*, el desarrollador de la aplicación debe implementar la serie de pasos descritos en el diagrama **4-12**. Adicionalmente, la misma secuencia se debe repetir si se desea agregar el segundo *CaptoGlove* como sensor de la mano izquierda en la misma aplicación.

Por otra parte, utilizando las funciones de la librería *GITEICaptoglove*, el diagrama de la figura **4-12** se reduce a las instrucciones del diagrama **4-13**. Allí se visualiza la forma en la que la librería simplifica el proceso de desarrollo condensando los pasos para la configuración de los módulos, la captura de datos y la detección de gestos. Además, las mismas funciones pueden ser reutilizadas rápidamente para la configuración de cada nuevo módulo Captoglove agregado a la escena. El código de implementación del diagrama **4-13** se encuentra en el anexo A.14 de este documento.

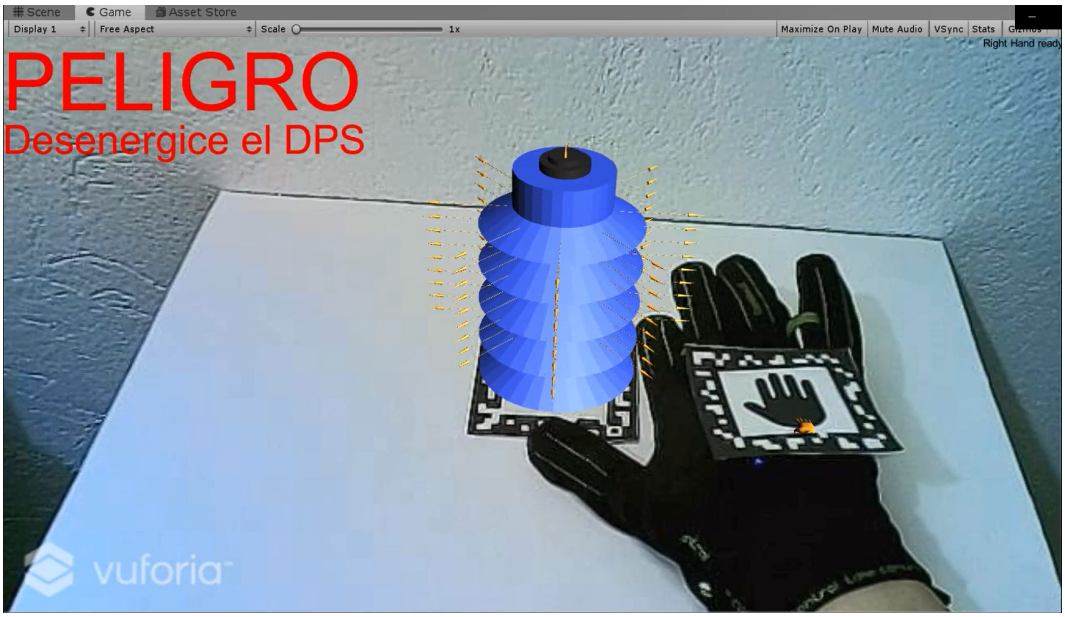

Figura 4-11.: Acercar mano a DPS energizado

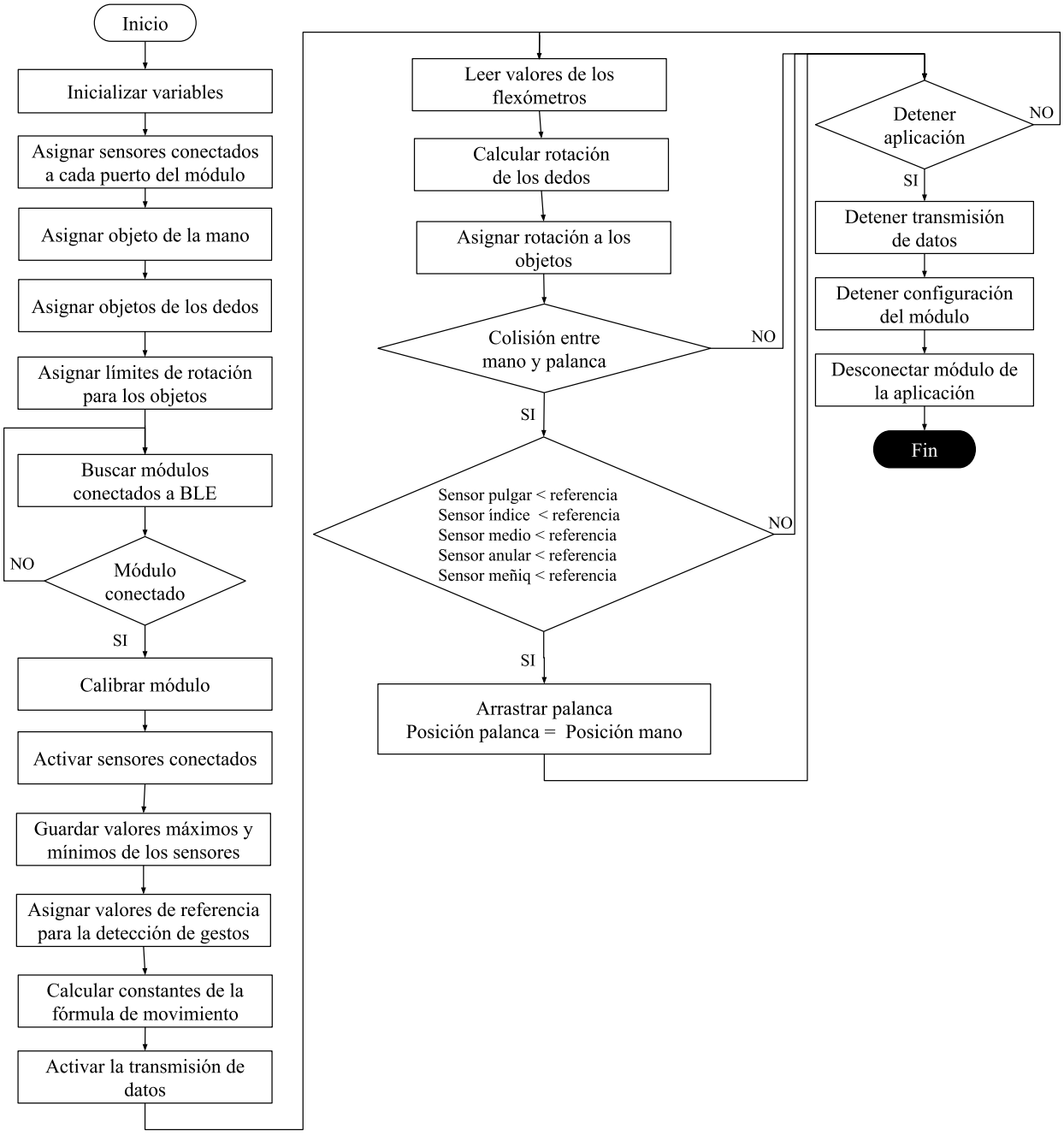

Figura 4-12.: Diagrama de flujo para interacción con DPS sin librería *GITEICaptoglove*

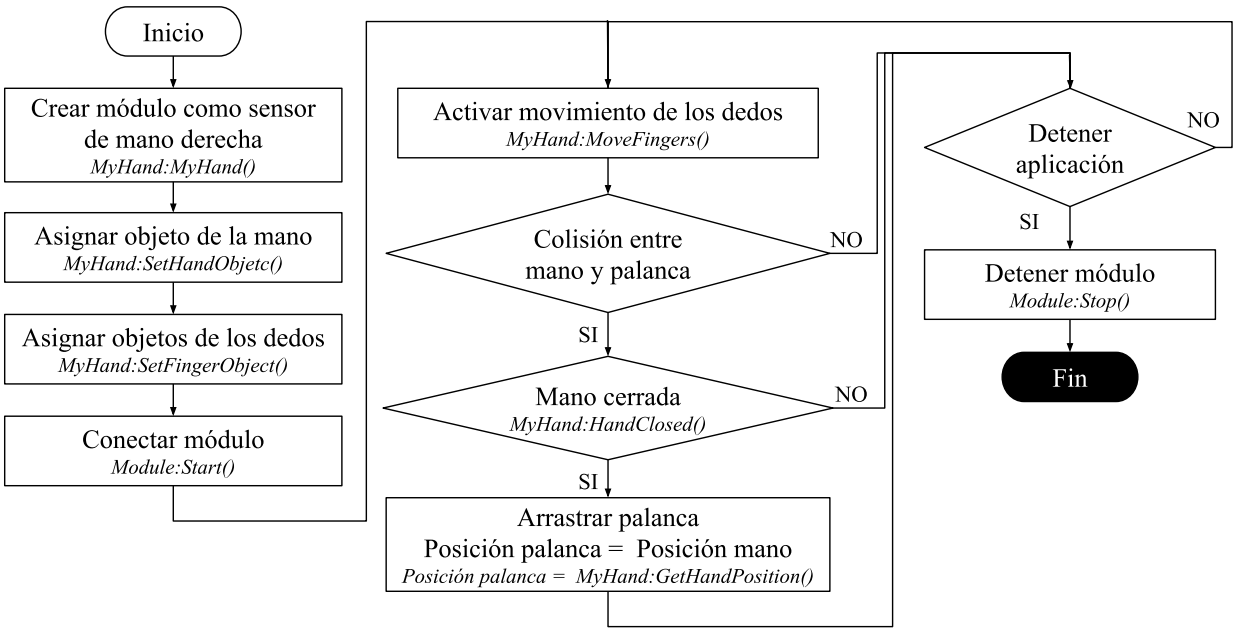

Figura 4-13.: Diagrama de flujo para interacción con DPS usando librería *GITEICaptoglove*

4.5. Instrucciones de configuración

Gracias a la experiencia adquirida durante el desarrollo de este proyecto, se recopilaron una serie de instrucciones para calibrar y configurar los módulos Captoglove correctamente antes de vincularlos a una aplicación Unity. El documento generado se encuentra en el anexo A.9.

4.6. Resumen del capítulo

Los resultados presentados en este capítulo demuestran la funcionalidad de la librería *GITEI-Captoglove* dentro de aplicaciones de realidad virtual y aumentada en Unity, evidenciando los beneficios de la solución en cuanto a la simplificación y reutilización de recursos de *software* durante el desarrollo de aplicaciones con diferentes propósitos; específicamente, para aquellas en las que se requiere la interacción del usuario con objetos virtuales a través del movimiento de las manos usando los guantes hápticos Captoglove.

5. Conclusiones y trabajo futuro

5.1. Resumen

La integración de *hardware* especializado a aplicaciones de realidad mixta, habilitan formas de interacción diferentes a las tradicionales entre los usuarios y la interfaz. Los guantes hápticos permiten a los usuarios interactuar con objetos reales y virtuales en una escena, de forma más natural al emplear las manos por sí mismas y comunicar intenciones mediante gestos y el contacto. Este tipo de interacción es requerida en la industria a la hora de emplear herramientas tecnológicas, como la realidad virtual y aumentada, para el entrenamiento de trabajadores en el sector operativo, pues a través de entornos simulados es posible enseñar tareas específicas reduciendo el riesgo al que se exponen los aprendices durante el entrenamiento en campo.

Dentro de diferentes tipos de capacitación de trabajadores en actividades operativas es posible encontrar elementos de interacción en común, como la manipulación de herramientas, maquinaria y tableros de control, y la activación de botones y palancas con diferentes propósitos de acuerdo a los objetivos del entrenamiento. Estas interacciones fueron recopiladas en la librería *GITEICaptoglove* mediante el rastreo de las manos, permitiendo su integración a múltiples entornos de capacitación como en el sector petrolero, textil, eléctrico y de manufactura.

La librería *GITEICaptoglove*, junto con los entregables generados en este proyecto, simplifican el desarrollo de aplicaciones de realidad virtual y aumentada en Unity que integran los guantes hápticos Captoglove. Esta biblioteca se encarga del procesamiento de datos en segundo plano para configurar los módulos de acuerdo a los parámetros establecidos por el desarrollador, capturar los valores sensados de rotación de las manos y antebrazos, la flexión de los dedos, y detectar gestos de interacción. Adicionalmente, la librería permite simular los movimientos del usuario en el mundo real dentro de entornos virtuales mediante la sincronización de objetos 3D en Unity con el movimiento de los módulos Captoglove. La librería es el punto de referencia para futuros proyectos con los guantes Captoglove y sus alcances se extienden a las necesidades de cada proyecto para ofrecer soluciones personalizadas a cada industria.

5.2. Conclusiones

A continuación se presentan las conclusiones obtenidas luego del desarrollo de este trabajo:

- La identificación de necesidades y procesos repetitivos dentro de un equipo de desarrollo de *software* con objetivos específicos, permite proponer y crear soluciones personalizadas como la desarrollada en este proyecto, que favorecen la reutilización de recursos y el trabajo eficiente dentro del equipo. Soluciones que no es posible encontrar en el mercado dado sus características particulares como, en este caso, la combinación de realidad virtual y aumentada, sensores hápticos y el entrenamiento industrial de trabajadores.

- La librería de *software* *GITEICaptoglove* sintetiza procesos extensos del procesamiento de datos que se requiere para integrar sensores hápticos a una aplicación de realidad mixta en Unity, reduciendo así el esfuerzo y el tiempo que los desarrolladores deben invertir para lograr una interacción adecuada entre los usuarios de los guantes Captoglove y la aplicación.

- La librería desarrollada puede ser fácilmente modificada mediante la detección de nuevos gestos y movimientos para extender su aplicación a otras áreas de la realidad virtual y aumentada, diferentes al entrenamiento de trabajadores.

- La combinación de diferentes dispositivos hápticos dentro de una misma aplicación de realidad mixta, habilita formas de interacción más intuitivas y deseables en entornos de entrenamiento de trabajadores. En este proyecto se exploró la implementación de los *CaptoGlove* como sensores de las manos y los *CaptoSensors* como sensores de los antebrazos, dando mayor movilidad al usuario dentro de una escena virtual. Utilizando la librería *GITEICaptoglove* como referencia, los módulos Captoglove pueden ser ubicados en otras articulaciones y explorar nuevas formas de interacción para entornos de capacitación específicos.

- Los guantes hápticos disponibles en el mercado son dispositivos aún en desarrollo que aunque bien permiten reconocer y evaluar los alcances de esta tecnología dentro de entornos virtuales de capacitación, aún requieren mejoras para lograr una solución completa, confiable, de alta precisión y lo suficiente estable para ser integrados a aplicaciones complejas, como las que se requieren en el área industrial.

5.3. Trabajo futuro

Las limitaciones de funcionamiento de los módulos Captoglove mencionadas en este documento, son conocidas por el fabricante y actualmente se encuentran en la búsqueda de una solución. También hay funcionalidades planeadas y pendientes por implementar en los módulos, por lo que se espera un soporte técnico a largo plazo por parte de Captoglove. En especial, una vez se logre la compatibilidad de los guantes con dispositivos Android, estos podrán ser integrados a aplicaciones móviles utilizando la misma librería creada *GITEICaptoglove* en combinación con el DLL *GSDKNet.BLE.Droid* del fabricante.

También se espera por parte de Captoglove, la inclusión de motores de vibración dentro de los módulos para enviar señales de retroalimentación al usuario y así enriquecer la interacción con su entorno. Con esto se podrá crear un sistema de alertas dentro de las aplicaciones de realidad mixta para el entrenamiento de trabajadores, y ofrecer de forma rápida y oportuna información al usuario sobre errores o precauciones que se deben tener antes de realizar una actividad en específico.

Por otra parte, actualmente los guantes Captoglove son totalmente compatibles con aplicaciones de realidad virtual en Windows. Lo anterior extiende la posibilidad de integrar los guantes con visores HMD bajo el mismo sistema operativo, como los HTC vive. Incluso es posible reemplazar los controles estándar de este visor por los guantes Captoglove para interactuar por medio de gestos en la interfaz.

Por último, es importante evaluar la respuesta de los usuarios a interfaces de realidad mixta tangibles utilizando guantes hápticos, y recibir retroalimentación que permita crear sistemas cada vez más intuitivos y menos invasivos. Este proyecto estuvo enfocado en la parte técnica e integración de los guantes Captoglove con aplicaciones en Unity, por lo que ahora se pueden poner a prueba, y realizar comparaciones entre diferentes métodos de interacción para determinar la ruta en la que se deben encaminar las siguientes investigaciones y exploraciones en esta área.

A. Anexos digitales

A continuación se listan los archivos adjuntos a este documento indicando el repositorio y el directorio donde se encuentran. Los cuatro repositorios mencionados son:

`https://github.com/lalumoreno/Captoglove.git`
`https://github.com/lalumoreno/CaptogloveAR.git`
`https://github.com/lalumoreno/CaptogloveVR.git`
`https://github.com/lalumoreno/DPS_AR.git`

A.1. Aplicación DPS

Archivo: *PackageDPS.unitypackage*
Ubicación: `https://github.com/lalumoreno/DPS_AR`
Repositorio: `https://github.com/lalumoreno/DPS_AR.git`
Descripción: Proyecto Unity de la aplicación de ejemplo en realidad aumentada para la interacción con un DPS.

A.2. Aplicación en realidad aumentada

Archivo: *PackageAR.unitypackage*
Ubicación: `https://github.com/lalumoreno/CaptogloveAR`
Repositorio: `https://github.com/lalumoreno/CaptogloveAR.git`
Descripción: Proyecto Unity de la aplicación de ejemplo en realidad aumentada.

A.3. Aplicación en realidad virtual

Archivo: *PackageVR.unitypackage*
Ubicación: `https://github.com/lalumoreno/CaptogloveVR`
Repositorio: `https://github.com/lalumoreno/CaptogloveVR.git`
Descripción: Proyecto Unity de la aplicación de ejemplo en realidad virtual.

A.4. Código fuente GITEICaptoglove

Archivos: *Module.cs, MyHand.cs, MyArm.cs*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/GITEI%20Captoglove%20Libraries/GITEICaptoglove`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Código fuente de la librería *GITEICaptoglove* desarrollada.

A.5. Diagrama de clases GITEICaptoglove

Archivo: *ClassDiagram.svg*
Ubicación: `https://github.com/lalumoreno/Captoglove/blob/master/GITEI%20Captoglove%20Libraries/ClassDiagram.svg`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Diagrama de clases de la librería *GITEICaptoglove*.

A.6. Documentación GITEICaptoglove

Archivo: *index.html*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/GITEI%20Captoglove%20Libraries/Doc/Doc`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Documentación de la librería *GITEICaptoglove*.

A.7. Documentación SDK Captoglove

Archivo: *index.html*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/SDK/GsdkNet_1.3.0/Docs/GSdkNet`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Documentación del SDK de Captoglove.

A.8. Captoglove Suite

Archivo: *GSuite.exe*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/Suite`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Interfaz para la configuración de los módulos Captoglove.

A.9. Instructivo Captoglove

Archivo: *InstruccionesCaptoglove.pdf*
Ubicación: `https://github.com/lalumoreno/Captoglove/blob/master/GITEI%20Captoglove%20Libraries/InstruccionesCaptoglove.pdf`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Instrucciones para la configuración de los módulos Captoglove antes de vincularlos a una aplicación Unity.

A.10. Librería GITEICaptoglove

Archivo: *GITEICaptoglove.dll*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/GITEI%20Captoglove%20Libraries/GITEICaptoglove/bin/Debug/netstandard2.0`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Librería DLL desarrollada para la integración de los módulos Captoglove con aplicaciones Unity.

A.11. Modelos 3D

Archivos: *Mano_Derecha.blend, Mano_Izquierda.blend*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/Modelos/Hands`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Modelos 3D de la mano derecha e izquierda para usar en la integración de los guantes *CaptoGloves* a aplicaciones Unity.

A.12. Script movimiento de la mano derecha

Archivos: *MainRightHand.cs*
Ubicación: `https://github.com/lalumoreno/CaptogloveVR/blob/master/Assets/Scripts/MainRightHand.cs`
Repositorio: `https://github.com/lalumoreno/CaptogloveVR.git`
Descripción: *Script* de ejemplo implementando la librería *GITEICaptoglove* para simular el movimiento de las manos.

A.13. Script movimiento del antebrazo derecho

Archivos: *MainRightArm.cs*
Ubicación: `https://github.com/lalumoreno/CaptogloveVR/blob/master/Assets/Scripts/MainRightArm.cs`
Repositorio: `https://github.com/lalumoreno/CaptogloveVR.git`
Descripción: *Script* de ejemplo implementando la librería *GITEICaptoglove* para simular el movimiento de los antebrazos.

A.14. Script aplicación DPS

Archivos: *MainRightHand.cs*
Ubicación: `https://github.com/lalumoreno/DPS_AR/blob/master/Assets/Scripts/MainRightHand.cs`
Repositorio: `https://github.com/lalumoreno/DPS_AR.git`
Descripción: *Script* de Unity de ejemplo implementando la librería *GITEICaptoglove*.

A.15. SDK Captoglove

Archivos: *GSdkNet.dll, GSdkNet.BLE.Winapi.dll, GSdkNet.BLE.Droid.dll*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/SDK/GsdkNet_1.3.0/Plugins`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Librerías incluidas en el SDK de Captoglove.

A.16. Vídeo simulación DPS

Archivo: *AppDPS.mp4*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/Videos`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Vídeo del funcionamiento de la aplicación de ejemplo en realidad aumentada para la interacción con un DPS.

A.17. Vídeo aplicación en realidad aumentada

Archivos: *AppARButton.mp4, AppARLever.mp4, AppARLight.mp4*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/Videos`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Vídeos del funcionamiento de la aplicación de ejemplo en realidad aumentada.

A.18. Vídeo aplicación en realidad virtual

Archivo: *AppVR.mp4*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/Videos`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Vídeo del funcionamiento de la aplicación de ejemplo en realidad virtual.

A.19. Vídeo movimiento de los guantes Captoglove

Archivo: *HandMov.mp4*
Ubicación: `https://github.com/lalumoreno/Captoglove/tree/master/Videos`
Repositorio: `https://github.com/lalumoreno/Captoglove.git`
Descripción: Vídeo de la sincronización de los objetos en la escena de Unity con el movimiento de los *CaptoGloves*.

Bibliografía

[1] Blender. The software. https://www.blender.org/about/. Consultado: 2020-11-10.

[2] Captoglove. Shop captoglove. https://www.captoglove.com/shop/, . Consultado: 2020-09-23.

[3] Captoglove. Sdk captloglove. https://www.captoglove.com/sdks/, .

[4] S. Cawood and M. Fiala. *Augmented Reality A Practical Guide*. The Pragmatic Programmers, 2008.

[5] D. Crocker. *Dictionary of aviation*. A and C Black Business Information and Development, London, 2007.

[6] Enel-CODENSA. Nodo 6: Maniobras en sistemas de distribución de redes de energía eléctrica. 2020. Documento privado.

[7] F. Fahmi, F. Nainggolan, B. Siregar, Soeharwinto, and M. Zarlis. User experience study on crane operator erection simulator using senso glove in a virtual reality environment. 2020. IOP Conference Series: Materials Science and Engineering.

[8] Freepik. Counting hands showing different number of fingers premium vector. https://www.freepik.com/premium-vector/counting-hands-showing-different-number-fingers_4310322.htm. Consultado: 2020-11-10.

[9] B. Furht and J. Carmigniani. *Handbook of Augmented Reality*. Springer Publishing Company, 2011.

[10] J. Hahn, B. Ludwig, and C. Wolff. Augmented reality-based training of the pcb assembly process. 2015. MUM '15 Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia.

[11] S. Jeon and S. Choi. Haptic augmented reality: Taxonomy and an example of stiffness modulation. 2009. Teleoperators and Virtual Enviroments.

[12] Y. Jia. Quaternions and rotations. http://graphics.stanford.edu/courses/cs348a-17-winter/Papers/quaternion.pdf, 2013.

[13] M. Kuriena, M. Kimb, M. Kopsidaa, and I. Brilakisa. Real-time simulation of construction workers using combined human body and hand tracking for robotic construction worker system. 2018. Automation in Construction.

[14] R. Lindeman, R. Page, Y. Yanagida, and J. Silbert. Towards full-body haptic feedback. 2004. Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST).

[15] J. Linowes and K. Babilinski. *Augmented Reality for Developers. Build practical augmentend reality applications with Unity, ARCore, ARKit and Vuforia.* Packt, Birmingham, 2017.

[16] Manus-VR. Manus. `https://manus-vr.com/`. Consultado: 2019-06-20.

[17] Microsoft. What is .net? `https://dotnet.microsoft.com/learn/dotnet/what-is-dotnet`. Consultado: 2020-10-31.

[18] P. Miligram and F. Kishino. A taxonomy of mixed reality visual displays. 1994. IEICE Transactions on Information and Systems. Vol. E77-D.

[19] Noitom-International-Inc. Hi5 vr glove. `https://hi5vrglove.com/#features`. Consultado: 2019-06-20.

[20] M. Oberweger, P. Wohlhart, and V. Lepetit. Hands deep in deep learning for hand pose estimation. 2015. Proceedings of the Computer Vision Winter workshop (CVWW).

[21] R. Oliveira, T. Farinha, H. Raposo, and N. Pires. Augmented reality and the future of maintenance. 2014. Proceedings of Maintenance Performance Measurement and Management (MPMM).

[22] Oxford-University-Press. Oxford languages. `https://www.lexico.com/es/`. Consultado: 2020-10-11.

[23] T. Parisi. *Learning Virtual Reality. Developing inmersive experiences and applications for desktop, web and mobile.* O'Reilly Media, Inc., Sebastopol,CA, 2016.

[24] Perforador-2.0. ¿quién es el cuñero? `https://perforador20.wordpress.com/2018/12/13/quien-es-el-cunero/`. Consultado: 2020-12-01.

[25] V. Petrenko, F. Tebueva, V. Antonov, A. Apurin, and U. Zavolokina. Development of haptic gloves with vibration feedback as a tool for manipulation in virtual reality based on bend sensors and absolute orientation sensors. 2020. IOP Conference Series: Materials Science and Engineering.

[26] W. Piekarski and B. Thomas. Tinmith-hand: Unified suer interface technology for mobile outdoor augmented reality and indoor virtual reality. 2002. Proceedings of IEEE Virtual Reality.

[27] PTC-Inc. Vuforia. `https://www.vuforia.com/`. Consultado: 2020-09-23.

[28] M. Quandt, B. Knoke, C. Gorldt, M. Freitag, and K. Thoen. General requirments for industrial augmented reality applications. 2018. 51st CIRP Conference on Manufacturing Systems.

[29] T. Robertson, J. Bischof, M. Greyman, and E. Ilse. Reducing maintenance errors with wearable technology. 2018. 2018 Annual Reliability and Maintainability Symposium (RAMS).

[30] D. Schmalstieg and T. Höllerer. *Augmented reality. Principles and practice.* Addison-Wesley, Boston, 2016.

[31] J. Serván, F. Mas, J. Menéndez, and J. Ríos. Using augmented in airbus a400m shop floor assemply work instructions. 2011. American Institute of Physics.

[32] H. Tan and A. Pentland. Tactual displays for sensory substitution and wearable computers. 2001. Fundamentls of Wearabñe Computers and Augmented Reality.

[33] J. Teh, A. Cheok, R. Peiris, Y. Choi, V. Thuong, and S. Lai. Huggy pajama: Amobile parent and child hujjing communication system. 2008. Proceedings of the International Conference on Interaction Design and Children (IDC).

[34] V. TM. Vive cosmos elite. `https://www.vive.com/mx/`. Consultado: 2020-11-21.

[35] D. Tsetserukou, K. Sato, and S. Tachi. Exointerfaces: Novel exosceleton haptic interfaces for virtual reality, aumgmented sport and rehabilitation. 2010. Proceedings of the ACM Augmented Human International Conference (A), article 1.

[36] Unity-Technologies. Unity documentation. monobehaviour. `https://docs.unity3d.com/ScriptReference/MonoBehaviour.html`. Consultado: 2020-10-12.

[37] Unity-Technologies. Unity documentation. plugins. `https://docs.unity3d.com/es/530/Manual/Plugins.html`. Consultado: 2020-10-31.

[38] Unity-Technologies. Unity. `https://unity3d.com/unity`, . Consultado: 2020-09-23.

[39] Unity-Technologies. Rotation and orientation in unity. `https://docs.unity3d.com/2019.3/Documentation/Manual/QuaternionAndEulerRotationsInUnity.html`, . Consultado: 2020-10-07.

[40] VR-Gluv. All-in-one force feedback gloves for vr training. `https://www.vrgluv.com/#technology`. Consultado: 2019-06-21.

[41] F. Wen, Z. Sun, T. He, Q. Shi, M. Zhu, Z. Zhang, L. Li, T. Zhang, and L. C. Machine learning glove using self-powered conductive superhydrophobic triboelectric textile for gesture recognition in vr/ar applications. 2020. WILEY-VCH Verlag GmbH Co.

[42] D. Williams II and J. Chianetta. *Augments Essential Guide to Augmented Reality*. Augmented, 2016.

[43] M. Zhu, Z. Sun, Z. Zhang, Q. Shi, T. He, H. Liu, T. Chen, and C. Lee. Haptic-feedback smart glove as a creative human-machine interface (hmi) for virtual/augmented reality applications. 2020. Science Advances.