Universidad de las Ciencias Informáticas

Facultad 1

“Dispositivo de Hardware Libre para Interacción en Entornos 3D”

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Jorge Enrique Rodríguez Jiménez

Tutor: M.Sc. Ivan Pérez Mallea

La Habana, 2020

Declaración de autoría

Declaro por este medio que yo **Jorge Enrique Rodríguez Jiménez**, con carné de identidad **95083033063** soy el autor principal del trabajo titulado **"Dispositivo de Hardware Libre para Interacción en Entornos 3D"** y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los _____ días del mes de _____ de _____

Autor: _____

Tutor: _____

Dedicatoria

A mis padres por tener paciencia conmigo.

A mi hermana, quien me ha apoyado en todo y “para quien espero ser un ejemplo a seguir”.

A mi abuelo… algún día nos volveremos a ver.

Agradecimientos

Especialmente a mis padres y mi abuela por encaminarme lo mejor que pudieron para que fuera una buena persona.

A mi hermana, quien me quitó el 5 en una prueba de AC, pero me consintió en todas mis malcriadeces.

A mi tutor, quien ha estado al tanto de mi superación profesional incluso antes de ser mi tutor.

A mi tío Ale, quien siempre está dispuesto a conversar sobre perros y pesca.

A mi tío Melquiades, quien me ayudó con los equipos necesarios para el desarrollo de la tesis.

A mi tía Oneida, quien me apoyó durante uno de los procesos más difíciles que he pasado en la vida.

A todos los miembros de mi familia que me han ayudado y apoyado durante todos estos años.

A mis amigos y compañeros de apartamento por acompañarme en la buenas y malas.

Resumen

Los dispositivos de interacción hombre-máquina han sufrido un aumento en su velocidad de evolución. Desde la aparición de los primeros teléfonos inteligentes, hasta la aplicación de la realidad aumentada, y más recientemente, la consolidación de la realidad virtual, el modo en que el ser humano percibe el entorno ha cambiado debido a la influencia de estas nuevas tecnologías. Si bien existe una tendencia hacia el software y hardware libre, promoviendo el desarrollo sin limitaciones y la construcción de dispositivos con un bajo costo económico, Cuba no posee los recursos necesarios para garantizar la adquisición y empleo de dispositivos de realidad virtual, por lo que su desarrollo en cuanto a tecnologías 3D puede verse retrasado con respecto al del resto del mundo. El objetivo de la presente investigación es la construcción de un dispositivo de interacción para entornos 3D de tipo guante, mediante la utilización de tecnologías libres. Para ello se realizó un estudio fisiológico articular del miembro superior lo que facilitó el análisis de los sensores y la plataforma de hardware a utilizar. Se decidió utilizar la plataforma de hardware Arduino debido a su bajo costo y la facilidad para interactuar con sensores análogo- digitales.

Palabras claves: realidad virtual, hardware libre, Arduino, entornos 3D.

Índice

Índice de Figuras

Índice de tablas

Introducción

La sociedad, en crecimiento exponencial y constante cambio, ha variado su forma de pensar a través de los años, apartando los instintos básicos para la supervivencia y tendiendo a un mayor interés en la comodidad y el bienestar físico-espiritual. Con el fin de satisfacer las nuevas necesidades del hombre, la tecnología se ha visto obligada a una acelerada evolución y desarrollo, trayendo consigo el surgimiento de la era de las máquinas.

La metáfora de interacción hombre-máquina provista por las ventanas, el ratón, los íconos y el puntero ha sido ampliamente aceptada desde la llegada de los entornos de escritorios gráficos. Esto trajo consigo la estandarización de un conjunto de técnicas y tareas básicas para el uso de este tipo de entornos. Como resultado se desarrollaron interfaces gráficas de usuario maduras y estables, se formalizaron reglas para su manejo y se crearon nuevas funcionalidades como parte de entornos integrados de desarrollo, para potenciar este tipo de vinculación entre el usuario y el ordenador.

Estos entornos evolucionaron más allá de las dos dimensiones presentando programas que sentaron las bases para el desarrollo de videojuegos, simuladores, entrenadores y otros sistemas de Realidad Virtual (VR). A la par de la evolución de los sistemas han tenido que evolucionar los dispositivos de interacción humana, que aún distan mucho de haber llegado a la madurez. Las últimas décadas han marcado un gran avance en este sentido, sin embargo, aún no se ha llegado a un consenso general en la estandarización o forma de realizar las interacciones en las interfaces que conforman estos tipos de software.

Aunque la variedad de dispositivos para interactuar con estos entornos ha crecido exponencialmente, siguen siendo caros y difícil de adquirir, razón por la cual se siguen utilizando dispositivos 2D en programas que manejan 3D limitando así la experiencia de usuario en estas interfaces gráficas (UI).

El Centro de Entornos Interactivos 3D (VERTEX) de la Universidad de las Ciencias Informáticas posee una amplia experiencia en el desarrollo de videojuegos, simuladores y laboratorios virtuales. Sin embargo, su experiencia con los Dispositivos de Interacción Humana (HID) para estos entornos se encuentra limitada, puesto que la vanguardia en esta línea de desarrollo está representada en su mayoría por compañías estadounidenses, lo que aumenta en gran medida la dificultad de adquisición de este tipo de productos.

Lo anteriormente planteado impide a VERTEX dotar a sus entornos 3D de las bondades de estos dispositivos, lo que ocasiona que se dificulte elevar la sensación de inmersión en

ambientes 3D, no se puedan manipular de forma directa objetos en el espacio virtual, se dificulte elevar la actividad cognitiva mediante la interacción directa con entornos especializados, no se aprovechen las características espaciales para la manipulación y modelado de objetos en sistemas de realización 3D, no se pueda medir el rendimiento de los individuos en sistemas de simulación que requieren acciones físicas, no se tenga en cuenta la interacción directa en sistemas de simulación para inducir el comportamiento de individuos, se dificulte elevar la precisión en el control de dispositivos robóticos a larga distancia u otros objetos que forman parte de simulaciones. Estas características permitirían generar nuevas líneas de productos y mejorar los productos ya existentes.

Teniendo en cuenta la problemática antes descrita, se identifica como **problema de la investigación:** ¿Cómo elevar los niveles de inmersión e interacción en los entornos virtuales 3D que realiza el centro VERTEX de la Universidad de las Ciencias Informáticas?

Como **objeto de estudio** se definen los Dispositivos de Interacción Humana, precisando como **campo de acción** los Dispositivos de Interacción Humana para interacción en entornos 3D.

Para dar solución al problema de investigación planteado, se define como **objetivo general** Desarrollar un dispositivo de interacción para entornos 3D basado en hardware libre que pueda ser utilizado en los entornos desarrollados en el centro VERTEX.

Se plantea como **hipótesis** que La introducción de un Dispositivo de Interacción Humana para entornos 3D en los productos de VERTEX permitirá elevar los niveles de interacción e inmersión de los usuarios de los mismos.

Para dar cumplimiento al objetivo planteado se definen las siguientes **tareas de investigación:**

1. Estudio de los HID más utilizados
2. Estudio de las características de los HID
3. Estudio de las principales funcionalidades de los HID
4. Estudio de dispositivos creados a partir de Hardware Libre
5. Estudio de otras posibles soluciones a la problemática planteada
6. Definición de la plataforma de Hardware Libre a utilizar
7. Definición de los sensores y actuadores a utilizar
8. Diseño del circuito
9. Construcción del dispositivo

10. Programación del dispositivo
11. Estudio del entorno de trabajo de Unity 3D
12. Definición de las posibles restricciones en el uso del dispositivo para la interacción en entornos 3D
13. Definición de las posibles funcionalidades del dispositivo para la interacción en entornos 3D
14. Diseño del módulo para Unity 3D
15. Implementación del módulo para Unity 3D
16. Definición de las funcionalidades a validar
17. Diseño del entorno de prueba
18. Implementación del entorno de prueba
19. Ejecución de las pruebas de validación

Como parte de la investigación se determinan las siguientes variables

Variable independiente:

Dispositivo de Interacción Humana para entornos 3D

Variables dependientes:

- Nivel de inmersión
- Nivel de interacción

Para dar cumplimiento a las tareas propuestas anteriormente, se utilizaron los siguientes **métodos de investigación** teóricos y empíricos.

Métodos teóricos

Histórico-Lógico: Permitió estudiar la trayectoria histórica real del fenómeno y las tendencias del uso actual de los dispositivos de hardware para interacción en entornos 3D, con el fin de seleccionar las más apropiadas para darle cumplimiento al objetivo general de la investigación.

Analítico-Sintético: Permitió realizar el estudio teórico de la investigación facilitando el análisis de documentos y la extracción de los elementos más importantes relacionados con el proceso de desarrollo de dispositivos para la interacción en entornos 3D, que hacen posible la elaboración de conclusiones relacionadas con el objeto de estudio.

Inductivo-Deductivo: Permitió llegar al planteamiento del objetivo, además de la extracción de las ideas fundamentales para la elaboración y fundamentación teórica del trabajo de diploma, fue utilizado para el razonamiento de la información consultada, llegando a un grupo de conocimientos particulares y generales sobre los dispositivos para la interacción en entornos 3D.

Métodos empíricos

Observación: Permitió obtener las características de los sistemas que se desarrollan en el centro VERTEX y la información necesaria para el desarrollo del dispositivo.

Entrevista: Permitió definir las herramientas y metodologías usadas por el centro VERTEX para la elaboración de sus productos.

Estructuración capitular

Capítulo 1. Fundamentación Teórica: Se expone el marco teórico de la investigación y se precisa lo referente al estudio de homólogos así como las definiciones y aspectos relacionados con la temática planteada, la metodología de desarrollo de software a utilizar y las heramientas y tecnologías necesarias para la implementación del sistema.

Capítulo 2. Análisis y diseño de la solución: Se presenta la propusta de solución, tanto de hardware como de software, así como los requisitos y funcionalidades del sistema, se define la arquitectura de software y los patrones de diseño. Se describen las historias de usuario y los procesos necesarios para la implementación de la aplicación.

Capítulo 3. Implementación y Pruebas: Se describe la etapa de implementación mediante la cual será posible la obtención del sistema. Se describen y documentan las pruebas realizadas a las funcionalidades desarrolladas.

Capítulo 1: Fundamentación Teórica

En el presente capítulo se expone el marco teórico en el que se desarrolla la investigación, para ello, se realiza un análisis sobre los principales conceptos relacionados con el objeto de estudio. Se identifica la metodología de desarrollo, así como las tecnologías y herramientas necesarias para la construcción del prototipo y la implementación de la propuesta de solución.

1.1 Realidad Virtual

A. Rowell expresaba que “La realidad virtual es una simulación interactiva por computador desde el punto de vista del participante, en la cual se sustituye o se aumenta la información sensorial que recibe”. Teniendo en cuenta el planteamiento anterior, se puede definir a la realidad virtual como un sistema informático que genera un conjunto de representaciones de la realidad, las cuales se expresan como simulaciones interactivas mediante el uso de componentes especializados. Tomando como punto de partida lo antes expuesto, es necesario señalar que existen tres elementos básicos que se encuentran presentes en todos los sistemas de realidad virtual:[1]

- Simulación interactiva
- Interacción implícita
- Inmersión sensorial

Simulación interactiva

El hecho de que una aplicación de realidad virtual sea interactiva es lo que la distingue del resto de aplicaciones tradicionales o de las animaciones. Tanto en una animación como en una aplicación tradicional, el usuario es un observador pasivo que reacciona al entorno mediante reglas previamente definidas, por lo que no podrán cambiar el contenido más allá de lo previsto por los desarrolladores. En un sistema de realidad virtual, cada acción y movimiento del usuario es improvisado por el mismo durante la marcha, sin necesidad de establecer un guion, el usuario decide qué hacer y que no, afectando de forma directa el entorno y posibilitando experiencias únicas en cada ejecución de la aplicación.

Interacción implícita

En los sistemas de realidad virtual el equipo de cómputo captura la voluntad del usuario implícita en sus movimientos naturales, por ejemplo, si el usuario desea ver la parte del mundo que le queda detrás, no mueve el ratón como en la forma clásica (interacción

explícita) ni ejecuta ningún comando, sino que simplemente gira la cabeza. Tanto en la interacción explícita como en la implícita se utilizan periféricos para la entrada y salida de datos, pero la diferencia fundamental está en la percepción que tiene el usuario sobre estos dispositivos, pues en esta última, el usuario deja de percibir al propio ordenador e interactúa de forma directa con los objetos del espacio virtual.

Inmersión sensorial

Se puede definir la inmersión sensorial como la desconexión de los sentidos del mundo real y la conexión al mundo digital. El usuario deja de percibir el entorno que lo rodea y pasa a estar completamente sumergido en el espacio virtual que se ejecuta desde el ordenador.[2] La vista es el sentido que más información le aporta a la mayoría de los seres humanos, por lo que todo sistema de realidad virtual debe garantizar que se proporcionen como mínimo, estímulos visuales.

1.2 Dispositivos de realidad virtual

Los dispositivos de realidad virtual no son más que equipos diseñados especialmente para servir como periféricos de entrada y salida[3]. Para un mejor entendimiento se dividirán dichos equipos en tres categorías, los encargados de la visión, la audición y la interacción.

Los dispositivos encargados de la visión, son aquellos que tienen la función de mostrarle al usuario los estímulos visuales del espacio virtual. En este apartado se encuentran las gafas, cascos y pantallas que son utilizados con ese propósito. Por lo general, están compuestos por una pantalla que trasmite la visión estereoscópica hacia el participante, aunque también pueden existir equipos con dos pantallas simples, una por cada ojo, por las cuales se reproducen las imágenes de forma tal que produzcan el mismo efecto que las estereoscópicas.[3] Otros equipos solo consisten en unos soportes con lentes especiales, en los cuales se acoplan los teléfonos celulares para ser utilizados como pantalla.

Los dispositivos encargados de la audición, son aquellos que tienen la función de trasmitirle al usuario los estímulos sonoros del espacio virtual. En este apartado se encuentran los auriculares, altavoces y sistemas de sonido que cumplen con los requisitos técnicos necesarios para ser utilizados con estos fines. La mayoría de los sistemas de realidad virtual poseen su propio dispositivo de audio, pero prácticamente en todos se puede utilizar cualquier tipo de auricular o altavoz, por lo que el usuario es libre de elegir el de su preferencia.[3]

Los dispositivos encargados de la interacción, son aquellos que tienen la función de trasmitir las órdenes del participante al ordenador. En este apartado se encuentran los mandos, guantes y controles (palancas, timones, etc.) que poseen los componentes necesarios para lograr que las acciones del usuario se vean reflejadas dentro del mundo virtual. Es necesario mencionar que no todos los sistemas poseen este tipo de dispositivos, algunos solo se centran en lograr la inmersión visual y auditiva. Hasta el momento, los dispositivos más utilizados para la interacción son los llamados mandos VR, los cuales poseen una serie de botones y palancas que permiten al participante ejercer su voluntad mediante ellos. Existen también guantes especializados capaces de controlar el movimiento de cada dedo, lo que mejora en gran medida la experiencia virtual, pues da la sensación al usuario de estar interactuando con los objetos de forma directa y no a través de un equipo.[4]

1.2.1 Dispositivos de realidad virtual más utilizados

Existen varias compañías que han desarrollado dispositivos para realidad virtual, cada uno tiene características que los diferencian de los otros, no solo en la estructura, sino también en los requerimientos para su uso y el precio de venta. A continuación, se realiza una breve reseña de los dispositivos más utilizados por los usuarios[5]–[7]:

Oculus

- Oculus Rift S: Incluye dos mandos Oculus Touch y un sistema posicional de audio integrado. Posee una pantalla LCD con una resolución de 1280 x 1440 y una tasa de refresco de 80 Hz. Necesita estar conectado a la computadora.
- Oculus Quest: Es completamente autónomo, dispone de una memoria de hasta 128 GB y dos mandos Oculus Touch. Posee un sistema de audio integrado y una pantalla OLED de resolución 1440 x 1600 por cada ojo con una tasa de refresco de 72 Hz.
- Oculus Go: Es el set más sencillo de la marca, posee un controlador único y se usa con la app de teléfono móvil de Oculus, por lo que es completamente autónomo. Dispone de una memoria de hasta 64 GB y tiene un sistema de audio integrado. Posee una pantalla LCD con una resolución de 1280 x 1440 y una tasa de refresco de 60 Hz.

HTC Vive

- HTC Vive: Es el modelo básico de la marca. Consiste en un sistema de periféricos que se conectan al ordenador vía USB, HDMI o por DisplayPort. El kit está compuesto por las Gafas HTC Vive, las cuales poseen una pantalla AMOLED de

3.6, pulgadas con una resolución de 1080 x 1200 por cada ojo y una tasa de refresco de 90 HZ. Link Box o caja de conexiones, es la encargada de conectar las gafas y el resto de los periféricos al ordenador. Dos sensores de posición, los cuales deben tener visión directa entre ellos o estar conectados. Auriculares de cable corto con un pack de siliconas de distintos tamaños. Par de mandos inalámbricos HTC Vive Controller.

- HTC Vive Pro: Como su nombre lo indica, es un kit superior al modelo básico. Posee todos los accesorios de su antecesor, pero con una pantalla de 3.5 pulgadas y una resolución de 1440 x 1600 por cada ojo y una tasa de refresco de 90 Hz. Auriculares con soporte de alta resistencia a impedancia y micrófono integrado. Es necesario mencionar que existe una versión llamada HTC Vive Pro Eye, la cual presenta un sistema de seguimiento ocular.
- HTC Vive Focus y Focus Plus: Estos kits son completamente inalámbricos. Poseen los mismos componentes que los anteriores, pero sus controladores inmersivos poseen 6 grados de libertad (GDL) de movimiento. Cabe destacar que si se quiere utilizar el HTC Vive estándar y el HTC Vive Pro de forma completamente inalámbrica, es necesario utilizar el HTC Vive Tracker, un dispositivo de seguimiento de realidad virtual.

Sony

- Sony PlayStation VR: El sistema está conformado por una consola PlayStation 4 estándar, una PlayStation Camera y por las gafas PlayStation VR. La pantalla OLED de 5.7 pulgadas, posee una resolución de 1920 x 1080 con una tasa de refresco entre 90 Hz y 120 Hz. Contiene un micrófono integrado y un sistema de sonido 3D.

Gafas de realidad virtual para teléfonos

Como ya se había mencionado con anterioridad, existen dispositivos que utilizan teléfonos móviles como pantalla o motor de la aplicación de realidad virtual. Estos equipos gozan de gran popularidad entre los usuarios, debido a que son alternativas económicamente viables, en comparación a los sistemas que traen todos los componentes incluidos. A continuación, se hace mención de los principales exponentes de este género[5], [8]:

- Samsung Gear VR: Estas gafas utilizan teléfonos de la misma marca como dispositivo controlador. Posee una pantalla AMOLED con una resolución de 2560 x 1440 y un mando inalámbrico para la interacción con el mundo virtual.

- Merge VR: Estos lentes están creados para ser utilizados por cualquier teléfono móvil que mida entre 123 mm y 158 mm. Aunque no posee un control propio, al utilizar la potencia del smartphone como pantalla y motor de la aplicación, da la posibilidad de conectar cualquier controlador siempre y cuando sea compatible con el sistema operativo del mismo.
- Google Daydream: Estos lentes son utilizables con la mayoría de teléfonos que tengan como sistema operativo Android 7.0 o superior. No poseen un controlador propio, por lo que al igual que las Merge VR, permiten utilizar casi cualquier mando compatible al teléfono. La calidad de visión depende en gran medida de la potencia del dispositivo controlador.

1.2.2 Análisis comparativo

Teniendo en cuenta las marcadas diferencias en los dispositivos utilizados para esta tecnología, no solo en cuanto a estructura, sino también en las características de uso y precio de adquisición en el mercado, se hace extremadamente complicado definir un estándar base a seguir para determinar cuál de estos equipos es mejor que los otros. A continuación, se expondrá un breve análisis comparativo sobre las características de los mismos.[4], [8]–[10]

Tipo de procesador

Una de las grandes diferencias existentes en estos dispositivos es el tipo de procesador que utilizan, puesto que es un indicador del grado de dependencia hacia otros equipos. Tanto las Samsung Gear VR como las Google Daydream, dependen de un teléfono celular para su funcionamiento, lo que trae algunas limitaciones para su uso. Las primeras tienen como particularidad que solo son compatibles con los teléfonos icónicos de la marca, como son el Samsung S6, S7 o Note 5. Las segundas solo pueden ser utilizadas por aquellos dispositivos preparados para el Daydream, como son el caso del Google Pixel y el Daydream Viewer.

Las Oculus Rift S y las HTC Vive en sus modelos básico y Pro, funcionan utilizando un ordenador como procesador del mundo virtual. El mismo, permite la conexión de varios accesorios y dispositivos que mejoran en gran medida la calidad de inmersión del usuario. Las limitantes en estos equipos de realidad virtual, vienen dadas por los requerimientos mínimos que deben tener las computadoras para su utilización, que por lo general tendrán que contar con avanzados microprocesadores, varios GB de memoria RAM y potentes

tarjetas gráficas. También está el caso de las Sony PlayStation VR, las cuales se ven restringidas por su dependencia a la consola PlayStation 4.

Algunos dispositivos son completamente independientes de un procesador externo, convirtiéndose en autónomos, como son el caso de Las Oculus Go y Oculus Quest. No necesitan estar conectados al ordenador o al teléfono, ni hacer uso de sensores externos. Todo lo necesario para su funcionamiento se encuentra dentro de los propios dispositivos, los cuales cuentan con memorias internas donde se almacenan los softwares utilizados en la experiencia inmersiva.

Tipo de conexiones

Que un dispositivo de realidad virtual se conecte al ordenador mediante cables o de forma inalámbrica, es un factor a tener en cuenta a la hora de analizar esta tecnología. Los equipos que ofrecen una mayor calidad son aquellos que se conectan al computador mediante cables, como son el caso de Oculus Rift S, Sony PlayStation VR y HTC Vive en sus versiones básica y Pro. Sin embargo, este tipo de conexión puede causar incomodidades y restricciones de movimiento a los usuarios. Las Samsung Gear VR y las Google Daydream se conectan a los teléfonos mediante pequeños cable USB, por lo que son considerados como dispositivos de realidad virtual móviles. Tanto las Oculus Go como las Oculus Quest en sus condiciones de equipos autónomos, son completamente inalámbricas, dejando de lado las posibles molestias que pudiera causar la conexión mediante cables.

Mandos de control

Con las gafas se obtienen los estímulos visuales y auditivos necesarios para lograr la inmersión hasta ese punto, pero solo con ellas no es suficiente para interactuar de forma efectiva con los objetos del entorno tridimensional, para esto es necesario hacer uso de los mandos de control. Dependiendo de las características estructurales del mando, la experiencia de usuario puede variar, pues cada uno presenta un espectro de funcionalidades únicas.

El HTC Vive Controller es el mando de control utilizado por los dispositivos de la misma marca. Presenta un diseño ergonómico con forma de varita, el cual posee un touchpad donde se apoya el dedo pulgar y un gatillo en el área del dedo índice, incluye dos botones laterales y su batería tiene una duración de 6 a 9 horas. Presenta un sistema de rastreo y tiene precisión en 360 grados.

Los mandos Oculus Touch son los empleados por los dispositivos desarrollados por Oculus. Poseen un diseño ergonómico. Incluyen tres botones, dos gatillos y un joystick. Al tomar el mando, los dedos se posicionan naturalmente sobre los botones. Los Oculus Touch no poseen batería recargable, por lo que utilizan pilas. No tiene sistema de rastreo y tampoco permiten la orientación en 360 grados.

El Samsung Gear Controller presenta un diseño mucho más simple en comparación con los anteriores. Contiene dos botones, un gatillo, un touchpad y un controlador de volumen. Se conecta de forma inalámbrica con las gafas. No posee batería recargable, por lo que necesita pilas. Este dispositivo es exclusivo de Samsung y viene a juego con el equipo de realidad virtual desarrollado por esta compañía.

Libertad de movimiento

No solo es necesario considerar el rastreo de posición, sino la libertad de movimiento que ofrece. Cada sistema de realidad virtual permite distintos niveles de libertad de desplazamiento sobre el terreno, lo que resulta en una variación de la experiencia inmersiva en dependencia del equipo utilizado.

En los casos de HTC Vive y Sony PlayStation VR, el movimiento está determinado por los sensores y cámaras externas. En una primera instancia se delimita el área de juego, la cual debe tener una longitud aproximada de 3 a 4 metros cuadrados. Dentro de esa área, el usuario es capaz de moverse con libertad y sus acciones se verán reflejadas en el mundo.

Otros dispositivos como Oculus Quest y Oculus Rift S no necesitan de sistemas externos de monitoreo y seguimiento. Sus propios visores contienen los componentes necesarios para permitir el desplazamiento del usuario en todas las direcciones, por lo que permiten sentir la realidad virtual sin necesidad de posicionamiento de cámaras ni delimitación de terrenos.

Los sistemas anteriormente mencionados, ya sea por el uso de sensores de posicionamiento externos o por el empleo de componentes internos de seguimiento, ofrecen la posibilidad al usuario de desplazarse por el espacio, esto se debe a que poseen 6 GDL de movimiento. Pero la mayoría de los dispositivos de realidad virtual móviles y algunos dispositivos autónomos, solo poseen 3 GDL de movimiento, como es el caso de las Samsung Gear VR, las Google Daydream y las Oculus Go. Estas permiten girar a ambos lados y mirar en todas las direcciones, pero siempre desde el mismo lugar, pues no admiten el desplazamiento por el mundo virtual.

Calidad visual

La calidad de la imagen que será observada por el participante no solo depende de las propiedades que presenten las pantallas de los dispositivos, sino también de la potencia del equipo encargado del procesamiento. HTC Vive y Oculus Rift S, al estar conectados a un ordenador con altas prestaciones, son capaces de reproducir imágenes de gran calidad. Los equipos autónomos Oculus Quest y Oculus Go, al no depender de un equipo de cómputo, presentan una pérdida en la calidad de los efectos de visión. La calidad de las imágenes de los sistemas móviles, Samsung Gear VR y las Google Daydream, depende en gran medida de las características que presenten los teléfonos que sirven como procesadores.

Precios en el mercado

A continuación, se presenta un gráfico con los precios aproximados en dólares que ostentan estos dispositivos en el mercado internacional.

Figura 1 Precio aproximado en dólares de los dispositivos de realidad virtual más utilizados

Los costos varían según las características y funcionalidades que ofrecen, aunque también dependen de las compañías que los fabrican. Debido a esto, los sistemas de realidad virtual móviles tienen bajos precios en comparación con el resto de los dispositivos, pues son los que menos niveles de inmersión e interacción otorgan, pero si se tiene en cuenta que los mismos consisten en lentes con un soporte estructural para insertar un teléfono, fácilmente se llega a la conclusión de que los dispositivos de realidad virtual tienen precios en extremo elevados.

1.3 Estudio fisiológico articular del miembro superior

Los mandos y controladores más avanzados, poseen sistemas de seguimiento que les permiten rastrear la posición del miembro superior para luego proyectarlo dentro del mundo virtual y permitir la interacción con el mismo por parte de los usuarios.[3], [4]

Este seguimiento generalmente se realiza mediante el uso de un sistema de localización local, donde solo se tiene en cuenta el movimiento de la parte que se desea proyectar. También es posible calcular la posición del miembro superior realizando un reconocimiento a los movimientos individuales que realizan cada una de sus partes.[11] Este último método es mucho más exacto y otorga un mayor control en la adquisición de datos por parte del dispositivo, permitiendo la posible adición de más funcionalidades, aunque para implementar el mismo es necesario conocer la estructura y funcionamiento del miembro superior en su totalidad.

El miembro superior es una estructura funcional compleja que se prolonga de forma lateral con respecto al tronco. Se encuentra fijado en la parte superior del tórax y su principal función es la prensión. Está compuesto por cuatro partes: la cintura escapular: formada por la clavícula y la escápula, el brazo: el cual posee un solo hueso llamado húmero, el antebrazo: constituido por el radio y el cúbito y la mano: estructurada por tres grupos de huesos, el carpo, los metacarpianos y las falanges.[12], [13]

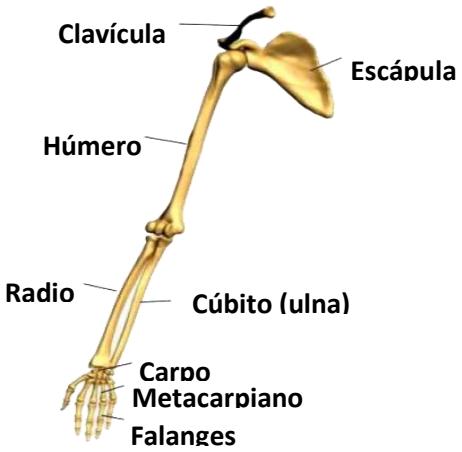

Figura 2 Esqueleto del miembro superior

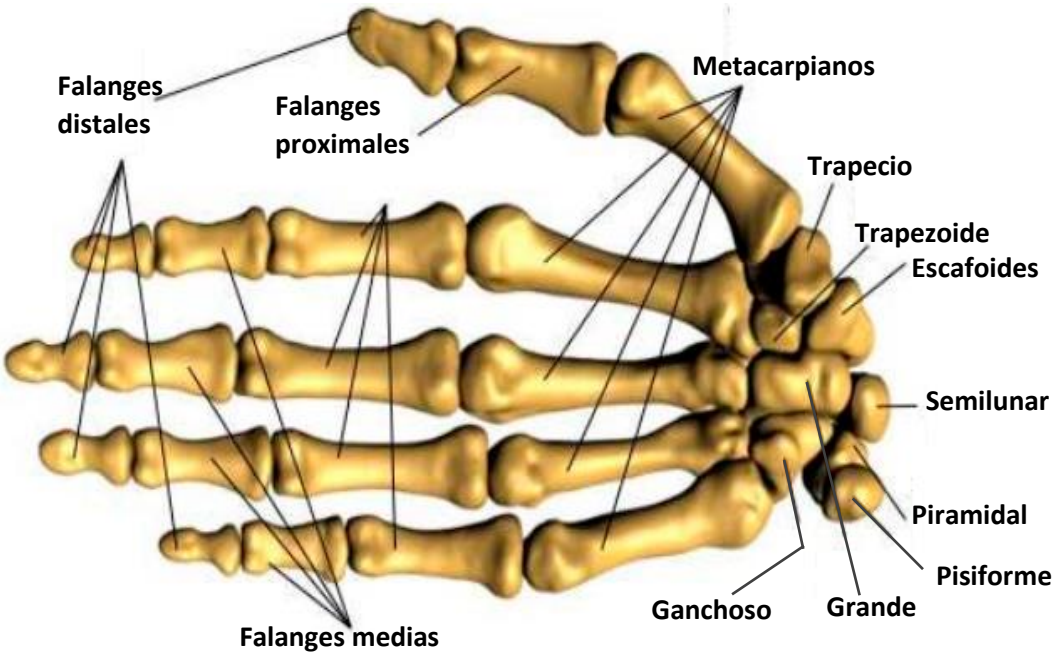

Figura 3 Esqueleto de la mano

1.3.1 Articulaciones y movimientos del miembro superior

Las articulaciones son las uniones entre los huesos, las mismas se catalogan en móviles, semimóviles e inmóviles. Cada segmento articular tiene límites físicos que le permiten describir arcos de movimiento.[12] Sería erróneo expresar que el miembro superior es capaz de realizar una determinada acción debido al desplazamiento de una de sus partes, pues estas se encuentran conectadas entre sí. Por lo antes expuesto, se puede afirmar que toda actividad realizada por este miembro, está dada por la combinación del movimiento angular de cada una de sus partes. Para un mejor entendimiento se presenta el siguiente resumen:[14], [15]

Articulación gleno-humeral (hombro)

- Flexión y extensión: Plano sagital. Eje transversal. Permite el movimiento de húmero con dirección anterior y posterior. Flexión (0º / 150º - 170º). Extensión (0º / 40º).

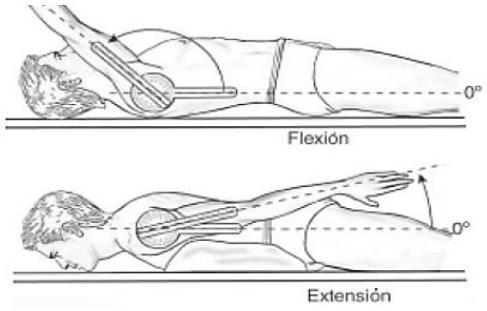

Figura 4 Flexión y extensión del hombro

- Abducción y aducción: Plano coronal o frontal. Eje anteroposterior. Permite el movimiento del húmero con dirección lateral y hacia la línea media del cuerpo. Abducción (0º / 160º - 180º). Aducción (0º / 30º).

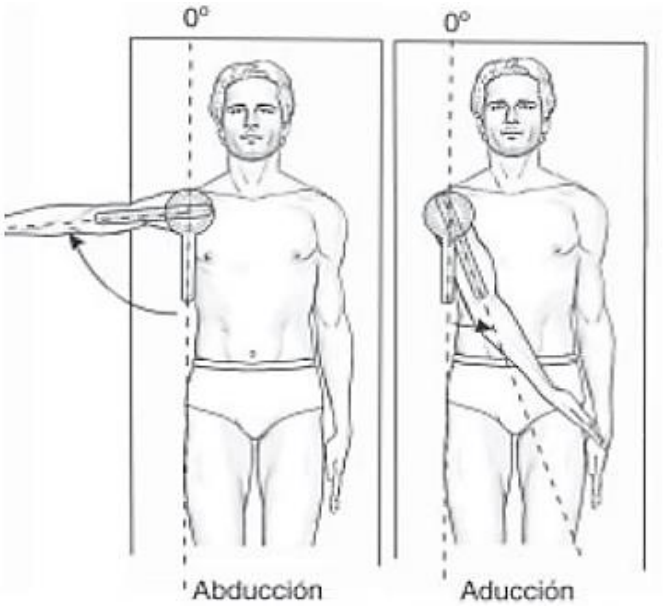

Figura 5 Abducción y aducción del hombro

- Rotación externa y rotación interna: Plano transverso. Eje longitudinal. Permite el movimiento del húmero con dirección medial y posterior, y lateral y posterior respectivamente (movimientos de rotación). Rotación externa (0º / 70º). Rotación interna (0º / 70º).

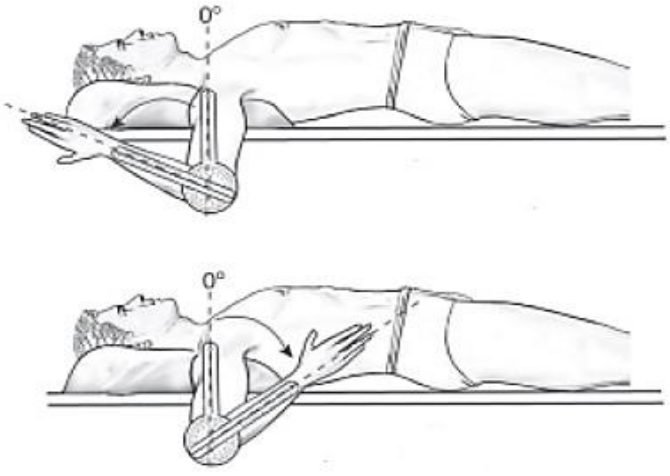

Figura 6 Rotación interna y rotación externa del hombro

Articulaciones del codo:

- Flexión y extensión: Plano sagital. Eje transversal. Permite el movimiento del radio en dirección anterior y posterior. Flexión (0º / 150º). Extensión (0º / 10º).

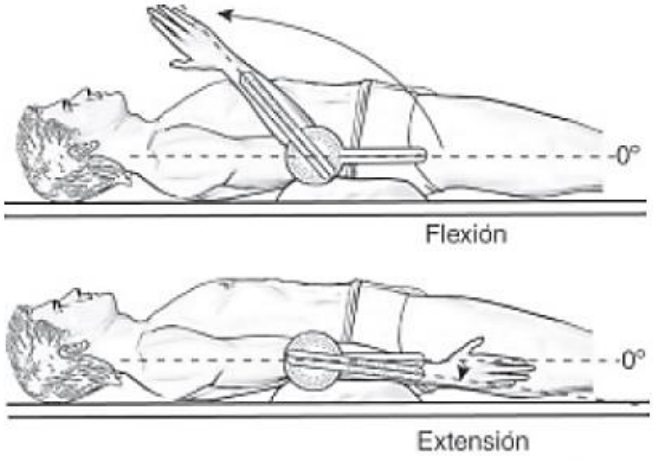

Figura 7 Flexión y extensión del codo

- Pronación y supinación: El radio se desliza en sentido anterior y medial durante la pronación (palma de la mano hacia abajo) y en sentido posterior y lateral durante la supinación (palma de la mano hacia arriba). Pronación (0º / 90º). Supinación (0º / 90º).

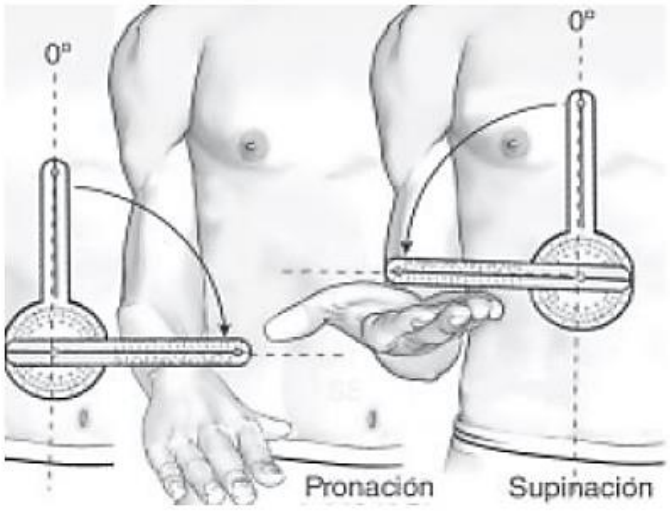

Figura 8 Pronación y supinación del codo

Articulación radio-ulno-menisco-carpiana (muñeca):

- Flexión y extensión: Plano sagital. Eje transversal. Permite el movimiento dorsal o posterior de la mano durante la extensión y el movimiento ventral o anterior de la mano durante la flexión. Flexión (0º / 50º - 60º). Extensión (0º / 35º - 60º).

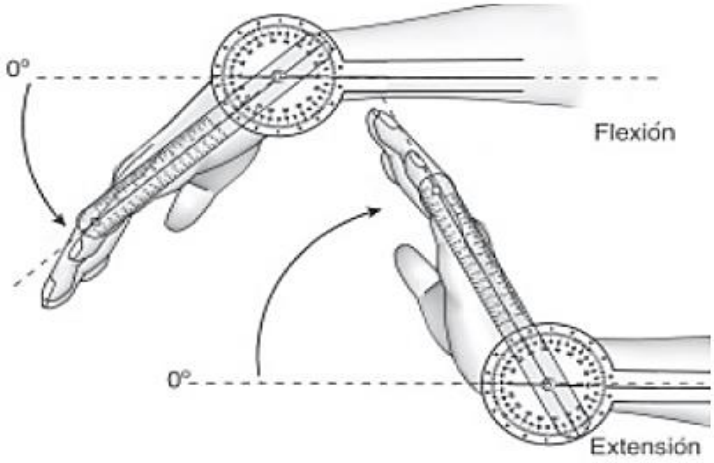

Figura 9 Flexión y extensión de la muñeca

- Desviación radial y cubital (ulnar): Plano coronal o frontal. Eje anteroposterior. Durante la desviación cubital permite el movimiento con dirección lateral de la mano, mientras que en la desviación radial se da el movimiento en dirección medial. Desviación radial (0º / 25º - 30º). Desviación cubital (0º / 30º - 40º).

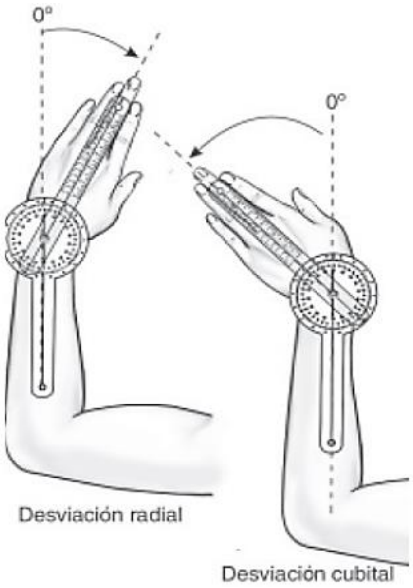

Figura 10 Desviación radial y cubital de la muñeca

1.3.2 Fundamentación físico- anatómica de los movimientos del miembro superior

Se entiende por desplazamiento de un cuerpo en un intervalo de tiempo, al cambio de su posición en ese intervalo. Dado que la posición de un cuerpo es una magnitud vectorial, el desplazamiento también lo es, y se define como la diferencia de los vectores de posición final e inicial siguiendo la ecuación $\overrightarrow{\Delta r} = \overrightarrow{r_f} - \overrightarrow{r_i}$. La velocidad es la cantidad de espacio recorrido por unidad de tiempo con la que un cuerpo se desplaza, la misma sigue la ecuación $\overrightarrow{v} = \overrightarrow{\Delta r}/\Delta t$. Teniendo en cuenta lo antes expuesto, se puede llegar a la conclusión de que dada una velocidad v, es posible calcular un desplazamiento r resolviendo la expresión $\vec{r} = \int \vec{v} * dt$. Es necesario mencionar, que desplazamiento y movimiento son dos conceptos distintos, puesto que este último se refiere a la variación de una posición en un instante de tiempo, teniendo en cuenta que dicha variación representa una transformación en el espacio tridimensional (desplazamiento y/o rotación).

Suponiendo que una de las partes del miembro superior, la mano, es considerada como un cuerpo en el espacio ¿Se podrá determinar el desplazamiento realizado por la misma? Teniendo la velocidad (con su valor vectorial para los ejes X, Y, Z) y el tiempo transcurrido, es físicamente posible realizar ese cálculo, tomando su posición actual como punto inicial en los ejes coordenados y aplicando la expresión planteada con anterioridad.

¿Se podrá determinar un movimiento realizado por la mano partiendo del punto de la mano como un ente independiente (un cuerpo en el espacio)? Es físicamente posible calcular un movimiento realizado por esta pero extremadamente complicado, pues para este caso hay que tener en cuenta el desplazamiento lineal, angular y rotación realizados por la misma, por lo que sería necesario combinar las ecuaciones de desplazamiento y posición de los movimientos rectilíneo, circular y parabólico, para crear un sistema capaz de resolver la interrogante planteada. Pero la solución puede ser mucho más sencilla si se toma la mano como lo que es, una parte de un todo.

El miembro superior funciona mediante una relación padre-hijo en forma de cadena, tomando como nodo raíz la cintura escapular[13], donde cada acción realizada por los nodos padre, tiene una repercusión directa en los nodos hijo. Como ya se había expresado con anterioridad, estos movimientos son ejecutados por las articulaciones, las cuales permiten ciertos grados de libertad, en correspondencia con sus límites físicos.

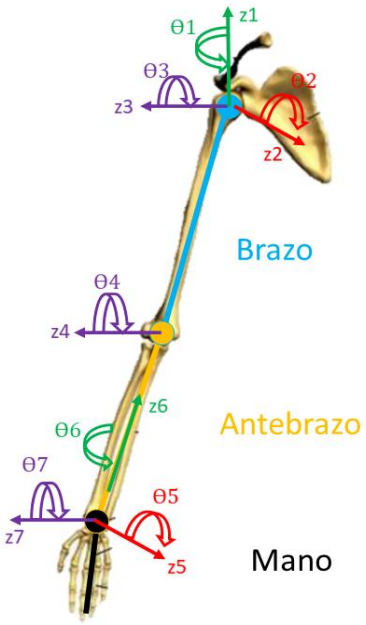

Figura 11 Movimiento articular del miembro superior

Tabla 1 Movimientos del miembro superior

Θi	Movimientos del miembro superior	**Articulaciones**
Θ1	Rotación Interna / Externa	Hombro
Θ2	Abducción / Aducción	
Θ3	Flexión / Extensión	
Θ4	Flexión / Extensión	Codo
Θ5	Desviación Radial / Cubital	Muñeca
Θ6	Pronación / Supinación	Codo
Θ7	Flexión / Extensión	Muñeca

Teniendo en cuenta lo antes expuesto, se puede afirmar que el movimiento de cada una de las partes del miembro superior, está determinada por la combinación del valor angular tomado por las articulaciones que actúan sobre la misma y sobre su cadena de nodos padre, las cuales se denotarán como articulaciones activas, siendo las que no actúan denotadas como articulaciones pasivas. Expresando el vector de la forma: *Xi (Θ1, Θ2, Θ3, Θ4, Θ5, Θ6, Θ7),* donde *i* es la parte en cuestión, *Xi* es el movimiento de la misma, y los Θ son los valores angulares correspondientes a las articulaciones. Es necesario destacar que los Θ de las articulaciones pasivas tomarán valor 0 para el cálculo de la expresión.

1.4 Plataformas de hardware

Cada equipo de realidad virtual tiene su propio dispositivo encargado del procesamiento de los datos provenientes de los sensores y actuadores, estos varían en dependencia de la compañía que los fabrica en cuanto a características y propiedades.[16] Aunque los mismos no presentan grandes diferencias en lo relacionado a su funcionamiento, pues todos siguen los mismos preceptos funcionales que los sistemas empotrados, los cuales se basan en el empleo de microcontroladores para el cumplimiento de las tareas asignadas.

Con el avance de las tecnologías ha aumentado exponencialmente el uso de sistemas embebidos. En estos sistemas de propósito especial el equipo está encapsulado por el dispositivo que controla; a diferencia de un ordenador de propósito general estos realizan tareas predefinidas con requisitos muy específicos. Los sistemas embebidos son identificados por ser un sistema de hardware que ejecuta el software directamente sobre sí mismos. Permiten la adición y sustracción de componentes, posibilitando su optimización enfocada al cumplimiento de una determinada función. Las principales características de estos sistemas son:[17], [18]

- Económicamente factibles
- Funcionan teniendo en cuenta restricciones en tiempo real
- Poseen un bajo consumo de energía
- Almacenan su código en la memoria ROM
- Pueden operar bajo condiciones ambientales extremas

Los sistemas embebidos incluyen la utilización de microcontroladores en sus estructuras de hardware. Un microcontrolador es un circuito integrado que tiene la característica de ser programable. Esto significa que el mismo es capaz de procesar y ejecutar programas previamente implementados por los desarrolladores. Dentro de su estructura, se encuentran tres componentes básicos:

- **Unidad Central de Proceso (CPU):** Es la parte encargada de la ejecución y control de las instrucciones. Hace uso de datos disponibles (datos de entrada), para generar otros diferentes (datos de salida), los cuales pueden ser utilizados o no, en la ejecución de la próxima instrucción.
- **Memorias:** Son las encargadas de alojar tanto las instrucciones, como los datos de entrada y salida. Posibilitando el acceso a esta información por parte del CPU, para su posterior uso. Generalmente se encuentran dos tipos de memoria, las persistente

y las volátiles. En las persistentes, la información se almacena de forma permanente, incluso ante la presencia de cortes en la alimentación eléctrica. En las volátiles, la información se guarda de forma temporal, la misma se pierde en caso de un corte en la alimentación eléctrica. Según las características de la información, esta se grabará de forma automática en el tipo de memoria que corresponda.

- **Entradas y salidas:** Se encargan de la comunicación con el mundo exterior. En los pines de entrada, se pueden conectar distintos tipos de sensores, para permitir que el microcontrolador perciba los datos provenientes de su entorno. En los pines de salida, se pueden conectar actuadores, para posibilitar que el mismo envíe órdenes y pueda interactuar con el mundo físico. Es necesario mencionar que la mayoría de estos dispositivos no poseen pines estrictamente de entrada o salida, sino que estos permiten ambas funciones, en dependencia del componente que se conecte a ellos.

Como principales plataformas empleadas para el desarrollo en sistemas embebidos, se identifican las Raspberry Pi y las placas Arduino. Aunque están orientados a la resolución de problemas similares, ambos dispositivos son muy diferentes, la Raspberry Pi se considera como una computadora completamente funcional, la cual actúa sobre un sistema operativo, permitiendo el cumplimiento de varias tareas al mismo tiempo. Mientras que Arduino es identificado como un microcontrolador, el cual solo es un componente de una computadora.

1.4.1 Raspberry Pi

La Raspberry Pi es la placa de un ordenador simple, compuesto por CPU, memoria RAM, puertos de entrada y salida de audio y vídeo, conectividad de red, ranura SD para almacenamiento, reloj, una toma para la alimentación, conexiones para periféricos de bajo nivel, en fin, todos los componentes necesarios para el funcionamiento de un ordenador normal.[19], [20]

Está destinada principalmente, al desarrollo de pequeños prototipos y a estimular la enseñanza de las ciencias de la computación en los centros educativos. Desarrollada en hardware libre cuenta con sistemas operativos GNU/Linux como Raspbian, aunque podemos encontrar otros sistemas operativos optimizados para el hardware, como son el caso de OSMC, PINET, RISC OS, SNAPPY UBUNTU Core y Windows 10 IOT Core.[21]

Raspberry Pi utiliza una arquitectura de tipo RISC (Reduced Instruction Set Computer), es decir, utiliza un sistema de instrucciones realmente simple, lo que le permite ejecutar tareas con un mínimo consumo de procesamiento.

Raspberry Pi posee un conjunto de características que le da ciertas ventajas en comparación con otros sistemas similares:[19]

- **Hardware de uso libre:** Raspberry Pi mantiene control sobre la plataforma, pero permiten su uso libre, tanto a nivel educativo como particular.
- **Entorno de programación flexible:** Al ejecutarse sobre un sistema operativo, la misma permite la programación desde cualquier IDE soportado por dicho sistema.
- **Comunidad activa:** Muchas personas utilizan Raspberry Pi, promoviendo la comunicación, para facilitar el aprendizaje y el acceso a información útil.
- **Reutilizables y versátiles:** Como se trata de un computador, es posible utilizar el mismo dispositivo para ejecutar varios proyectos. Debido a que dispone distintos puertos de entrada / salida, es posible conectar varios componentes (ratón, teclado, pantalla, etc.) en dependencia de la función que vaya a desempeñar.

1.4.2 Arduino

Arduino es una plataforma de prototipos electrónica de código abierto (open-source) basada en hardware y software flexible. El hardware consiste en una placa con un microcontrolador programable y varios puertos de entrada / salida, tanto digitales, como analógicos, así como salidas PWM y de comunicaciones. Arduino puede tomar información del entorno e interactuar sobre el mismo, a partir de la conexión de sensores y actuadores en sus puertos de entrada / salida.[22], [23]

A diferencia de una computadora, Arduino no posee pantalla ni teclado, se necesita de un programa externo, ejecutado en otra máquina, para poder crear las instrucciones para la placa. Este software es llamado Arduino IDE y sobre el mismo se escribe el lenguaje de programación Arduino, el cual está basado en Wiring.[24]

Arduino, además de simplificar el proceso de trabajar con microcontroladores, posee algunas características que lo ponen en una posición ventajosa respecto a otros sistemas similares:[22], [23]

- **Multiplataforma:** El software de Arduino se puede instalar en Windows, Linux y Mac OS.

- **Entorno de programación simple:** El entorno de programación de Arduino es fácil de usar para principiantes, pero lo suficientemente flexible para que lo utilicen los usuarios avanzados también.
- **Software extensible y de código abierto:** El software de Arduino está publicado bajo una licencia libre, como herramienta de código abierto.
- **Hardware ampliable y de código abierto:** Arduino está basado en los microcontroladores ATMEGA168, ATMEGA328 y ATMEGA1280. Los planos de los módulos están publicados bajo licencia CC (Creative Commons), por lo que diseñadores de circuitos con experiencia pueden hacer su propia versión del módulo, ampliándolo u optimizándolo.
- **Reutilizables y versátiles:** Reutilizables porque se puede aprovechar la misma placa para varios proyectos, pues es muy fácil desconectarla, reconectarla y reprogramarla. Versátiles porque proveen varios tipos de entradas y salidas de datos, que permiten capturar información de sensores y enviar señales a actuadores de múltiples formas.

Tecnologías y herramientas

Es evidente que cualquier solución que se vaya a realizar en este campo posee un componente de hardware para la recolección de los datos del mundo real y otro de software que permita procesar los datos y mantener la comunicación con el ordenador donde el ambiente virtual se desarrolla.[16], [25] De las tecnologías detrás de estas soluciones, generalmente propietarias y sin documentación, depende el éxito de cualquier nuevo desarrollo. Por lo que su selección es de vital importancia.

1.5.1 Selección de la plataforma de hardware

Raspberry Pi es muy superior a Arduino en cuanto a propiedades técnicas para su uso en aplicaciones de software. Pero la simplicidad de Arduino hace de este una mejor opción en cuanto a proyectos de hardware. Esto se debe a que el mismo ofrece una capacidad analógica en tiempo real, aportando la flexibilidad necesaria para trabajar con casi cualquier tipo de sensor o actuador, mientras que la Raspberry Pi requiere la asistencia de un hardware adicional para la lectura de los sensores analógicos. El entorno de desarrollo de Arduino es muy fácil de usar, pues se escribe el código en el IDE y luego se carga en la placa. Pero la Pi necesita primeramente de la instalación de un sistema operativo, luego la selección de un IDE compatible con este, por último, la instalación de las bibliotecas necesarias para comenzar la programación del hardware utilizando el lenguaje

seleccionado. Además, la Raspberry Pi al ser una computadora es mucho más compleja que Arduino y al pretender realizar un dispositivo que solo lee de sensores, sus capacidades de procesamiento serían subutilizadas y por lo tanto económicamente sería poco rentable. Por lo antes expuesto, se selecciona Arduino como plataforma de hardware a utilizar en el desarrollo de la presente investigación.

Existen varios tipos de Arduino, cada uno con características y funcionalidades únicas, los más utilizados son el Arduino UNO, DUE, Leonardo, FIO, Micro, Nano, YUN, LilyPad y Mega. Específicamente se estará haciendo uso de este último, en su versión Arduino Mega 2560.[26]

La Mega 2560 es una placa electrónica basada en el Atmega2560. Cuenta con 54 pines digitales de entrada / salida y 16 entradas analógicas, muchos más pines que las otras versiones de Arduino. Posee 4 puertos UART para la conexión serie con un ordenador, otra placa u otros microcontroladores. La placa puede ser alimentada mediante una conexión a un computador o con una fuente de alimentación externa, con una intensidad recomendada entre 7 y 12 voltios y de 0.75 a 1.5 amperes. La misma opera con corrientes de salida de 5v y 3.3v en sus pines respectivos. Tiene 256 KB de memoria flash para almacenar el código, de la cual se utilizan 8 KB para el cargador de arranque, 8 KB de SRAM y 4 KB de EEPROM. Pero la característica decisiva para su elección en esta investigación, es que la Mega 2560, a diferencia de otras placas, posee dos pares de puertos TWI (2 SDA y 2 SCL), los cuales permiten la comunicación mediante el bus I2C ideal para la conexión de sensores de posición que de manera dinámica envían datos constantemente a la placa.[27] Teniendo en cuenta las propiedades anteriormente expuestas, el Arduino Mega 2560 es la mejor opción entre todas las plataformas de hardware para ser utilizada como sistema de control y obtención de datos de los sensores.

1.5.2 Sensores

En la presente investigación se hará uso de Arduino para procesar la información proveniente del entorno. Aunque el mismo es incapaz de obtener los valores por sí solo, es necesario la utilización de sensores que permitan capturar los datos de movimiento del usuario y la envíen hacia la plataforma de hardware.

Un sensor no es más que un dispositivo capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación, y transformarlas en variables eléctricas. Las variables de instrumentación pueden ser: temperatura, intensidad lumínica, distancia,

aceleración, inclinación, desplazamiento, presión, fuerza, torsión, humedad, movimiento, etc.[28]

MPU-6050: El sensor MPU-6050 es un sensor inteligente de 6 grados de libertad (GDL), contiene el acelerómetro MEMS y un giroscopio MEMS en un chip. Es muy preciso ya que contiene hardware de conversión analógico-digital de 16 bits para cada canal. El sensor utiliza el bus I2C para interactuar con el Arduino. El sensor también contiene 1024 bytes de buffer FIFO. Si el MPU 6050 coloca datos en el búfer, señala al Arduino con una señal de interrupción para que sepa que hay datos en el búfer en espera de ser leídos.[29], [30] Eso hace posible tener dos de estos sensores en un proyecto. Es necesario destacar que el sensor es capaz de capturar los datos físicos en dos ejes coordenados, calculando los valores del tercer eje de forma virtual. Este sensor será utilizado para medir los valores de rotación de los movimientos de flexión y extensión, y de abducción y aducción de la articulación del hombro.

IMU-BNO055: El sensor BNO055 es un sensor inteligente de orientación absoluta de 9 GDL. Combina un acelerómetro, un giroscopio y un magnetómetro en el mismo chip. Es capaz de dar los valores de orientación tanto en ángulos de Euler como en Quaternion, con un alto grado de precisión. Utiliza el bus I2C para comunicarse con el Arduino y al igual que otros sensores de este tipo, posee 1024 bytes de buffer FIFO en el cual coloca los datos y luego realiza una señal de interrupción para avisar a Arduino que hay datos en el buffer en espera de leer.[31] Esto hace posible conectar dos de estos sensores a una misma placa. Esta IMU es capaz de capturar los datos físicos en los tres ejes coordenados. Será utilizado para medir los valores de rotación de los movimientos de la articulación del codo (flexión y extensión, pronación y supinación) y del movimiento de rotación de la articulación del hombro.

1.5.3 Herramientas

Entornos de desarrollo

Arduino IDE: Como ya se había explicado con anterioridad, este software es utilizado para escribir y posteriormente cargar programas en las placas Arduino.[24] En la presente investigación, será empleado para desarrollar el código necesario tanto para el control y obtención de datos de los sensores por parte del Arduino, como para el envío de información hacia el ordenador.

Unity 3D: Es un motor gráfico 3D multiplataforma. Se utiliza para crear videojuegos, aplicaciones interactivas, visualizaciones y animaciones 3D. Posee varias herramientas de edición, y la capacidad de importar modelos y materiales provenientes de otros motores.[32] En la presente investigación, Unity será empleado para desarrollar el sistema encargado de recibir los datos provenientes del Arduino y controlar el modo en que afectan a los objetos en la escena, así como de sostener todo el entorno gráfico y las reglas de interacción que se aplicarán dentro del mismo.

Fritzing: Es el programa por excelencia para la realización de esquemas eléctricos en proyectos con Arduino. Dispone bibliotecas con la mayoría de componentes incluido los propios Arduinos, placas de conexiones, led, motores, displays, etc.[33] En la presente investigación será empleado para realizar el diseño de conexión del hardware y la obtención del esquema eléctrico del mismo.

Lenguajes de programación

Lenguaje de programación de Arduino: Este lenguaje está basado en Wiring. Podría decirse que es una versión simplificada de C o C++, pues, aunque tiene un gran parecido con estos, no posee todas sus características. Está diseñado para la programación de hardware, específicamente de las placas compatibles con Arduino, aunque con la ayuda de bibliotecas, puede ser empleado para el desarrollo de programas para otras plataformas.[24]

C#: Se usa para la creación de Scripts en el motor Unity 3D. Este lenguaje es una evolución de los lenguajes C y C++. Utiliza muchas de las características de C++ en las áreas de instrucciones, expresiones y operadores. Está basado en la plataforma .NET y es utilizado para la creación de disímiles aplicaciones, pues constituye uno de los lenguajes empleados en la Programación Orientada a Objetos. [34], [35]

Metodología a utilizar

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos de software. En ellas se va indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, demostrando además qué personas deben participar en el desarrollo de las actividades y qué rol deben de tener, entre otros elementos.[36] Existe una gran cantidad de metodologías, teniendo en cuenta la filosofía de desarrollo aquellas que ponen un mayor énfasis en la planificación y control del proyecto, en la especificación precisa de requisitos

y modelado reciben el apelativo de metodologías tradicionales mientras que las que permiten la incorporación del cliente como un miembro más del equipo de desarrollo, permitiendo además dar respuestas rápidas a los continuos cambios que puedan producirse, son llamadas metodologías de desarrollo ágil.[37]

El equipo de desarrollo es pequeño, el mismo consta de un miembro, además, el tiempo destinado para el desarrollo del proyecto también es pequeño, por lo que se prefiere el empleo de una metodología de enfoque ágil. Para guiar el proceso de desarrollo de la propuesta de solución se decide emplear la metodología Extreme Programming (XP según sus siglas en ingles). Se ha seleccionado la misma teniendo en cuenta las siguientes características:[38]

- El equipo de desarrollo tiene experiencia previa en el uso de esta metodología.
- Presenta un desarrollo incremental, el cual se apoya en pequeñas y continuas liberaciones del sistema que se fundamentan a partir de las historias de usuario definidas por el cliente. Lo que permite la visualización temprana de prototipos funcionales.
- Frecuente interacción con el cliente mediante la inclusión de este en el equipo de desarrollo, facilitando la retroalimentación durante el proceso de creación. El mismo se encarga de definir las pruebas de aceptación para el sistema.
- Al realizar una liberación se realizan las pruebas correspondientes y si es posible se aplica una refactorización al código. Esto ayuda a conservar el código simple.
- En cada iteración de entrega se corrigen los errores detectados, garantizando que se puedan añadir nuevas funcionalidades de forma continua y segura.

Conclusiones parciales

En el presente capítulo se detallaron los principales conceptos relacionados con la realidad virtual, así como los dispositivos empleados para interactuar en estos entornos, lo que permitió llegar a la conclusión de que aún no existen estándares en cuanto a la estructura y funcionalidades que ofrecen los diferentes sistemas. El mercado relacionado con esta tecnología se encuentra en etapa de desarrollo. Los dispositivos de realidad virtual son caros y existen diferencias marcadas en cuanto al precio de los mismos en dependencia de la compañía que los desarrolla. El estudio fisiológico articular del miembro superior arrojó que los movimientos del mismo están dados por la combinación de los movimientos de rotación de cada una de las articulaciones, por lo que pueden ser simulados en una computadora mediante la utilización de sensores capaces de capturar estos ángulos. En la

construcción de prototipos de hardware en los que no se requiera un excesivo procesamiento de datos externos, Arduino constituye una opción más acertada que otras plataformas debido a que el mismo es barato y permite la conexión y el control en tiempo real de varios tipos de sensores.

Capítulo 2: Análisis y diseño de la solución

En el presente capítulo se describe la solución propuesta al problema de investigación. Se exponen los requisitos funcionales y no funcionales que debe cumplir la aplicación, además se realiza una descripción acerca de la estructura y funcionamiento del prototipo. Se presentan las historias de usuario y las tareas de desarrollo para guiar el proceso de construcción del sistema y se definen los estándares de codificación a utilizar durante la implementación del software.

2.1 Descripción de la propuesta de solución

Para el cumplimiento del objetivo de trabajo, se propone la creación de un dispositivo de interacción humana de tipo guante o mando de realidad virtual. El mismo debe permitir la interacción por parte del usuario con objetos del espacio virtual.

El desarrollo del sistema se ha separado en dos partes: diseño y construcción del dispositivo de hardware con su firmware, y diseño e implementación del software. La siguiente figura muestra de manera esquemática el desarrollo de la solución.

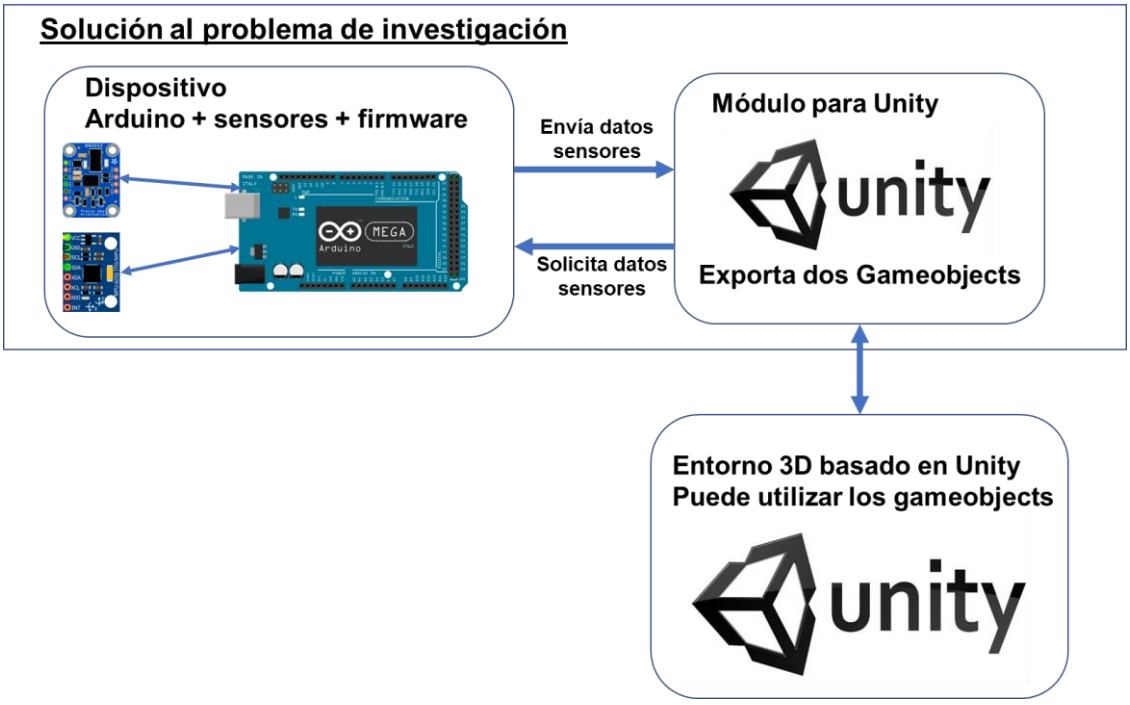

Figura 12 Presentación esquemática de la solución

El dispositivo cuenta con dos sensores que recogen la posición del brazo y antebrazo. La placa Arduino posee un programa empotrado (firmware) que recolecta estos datos, los procesa y los envía a la PC. En la PC existe un módulo para Unity que conecta con el

dispositivo y recibe los datos a partir de los cuales actualiza 2 objetos virtuales (gameobjects) que son exportados para ser utilizados en cualquier entorno 3D que utilice el motor de Unity. El módulo realiza peticiones de datos al dispositivo para actualizar la posición de los gameobjects en el mundo virtual. En los próximos epígrafes se detalla la solución tanto desde el punto de vista de hardware como de los componentes de software.

2.1.1 Descripción de la solución de hardware

El dispositivo debe permitir los principales movimientos del miembro superior sin restricciones de ningún tipo, para capturar los valores de rotación de cada parte y enviar al ordenador los datos necesarios para garantizar la sensación de inmersión con el mínimo error posible.

Teniendo en cuenta que el miembro superior se divide en tres partes (brazo, antebrazo y mano) y sus articulaciones permiten un total de 7 grados de libertad de movimiento, se ha decidido enfocar el equipo en controlar los movimientos del brazo y el antebrazo. Para ello se hará uso de los sensores IMU-BNO055 y MPU-6050, el primero se ubicará en el antebrazo, carca de la articulación de la muñeca, pues es en esta zona donde mejor se ven reflejados los movimientos; el segundo, será ubicado en el brazo, cerca de la articulación del codo.

Para capturar las acciones del usuario (tomar objetos, disparar un arma, etc.), es necesario la adición de al menos un pulsador o botón. El mismo será ubicado en la zona interior de la mano, facilitando su acceso por parte de los dedos. Debido a que se colocarán sensores y actuadores en las tres partes del miembro superior, se fijará la placa Arduino en el punto medio del antebrazo, garantizando la conexión de todos los componentes hacia el mismo.

La comunicación entre el Arduino y el ordenador será de tipo serial USB, mediante una conexión vía cable. El mismo no hará uso de fuentes de alimentación externa, toda la energía eléctrica necesaria la obtendrá a través del propio cable USB desde el computador.

El programa cargado en la memoria EEPROM de la placa Arduino, utilizará la comunicación serial con el ordenador, permitiendo el flujo de información entre ambos dispositivos. El mismo se encargará de obtener los datos de los actuadores y sensores, así como su preparación para su posterior envío a la computadora. Es necesario destacar que la calibración de sensores se realizará cargando un programa a la placa, el cual se ejecutará solo una vez, pero no influirá de forma directa en la ejecución de los sistemas necesarios para el funcionamiento del dispositivo desarrollado.

A continuación, se presenta el esquema de conexión de los componentes de hardware utilizados para la construcción del dispositivo:

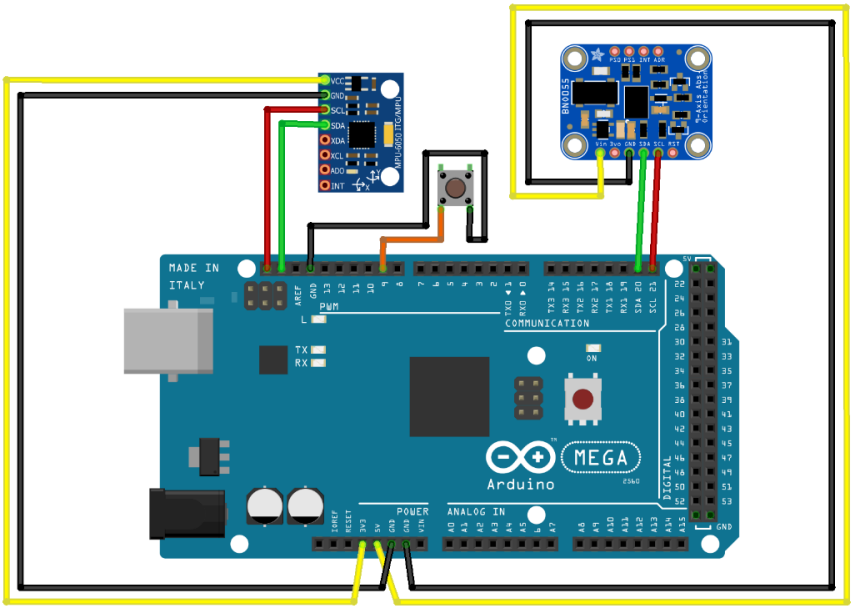

Figura 13 Diagrama de componentes de la solución

2.1.2 Descripción de la solución de software

La solución de software está compuesta por dos partes, la primera es el código empotrado en la placa Arduino que funciona como firmware del dispositivo y permite tanto la calibración como la recolección de los datos de posición de las partes del brazo en tiempo real desde los sensores.

La segunda parte tendría la función de un driver específicamente diseñado para Unity, con el fin de mantener una independencia del mundo virtual que se desee crear por cualquier desarrollador. Este módulo permitirá la conexión y la obtención de los datos enviados por Arduino, procesándolos y guardándolos en las variables correspondientes para su utilización.

Para su interacción con el mundo virtual donde se desee utilizar el dispositivo el módulo exporta dos objetos del tipo gameobject de Unity que representan el brazo y el antebrazo. Estos objetos son constantemente actualizados a partir de los datos que se reciben desde el dispositivo. Los autores de los ambientes virtuales que utilizarán el módulo pueden decidir si utilizan los dos gameobject o solo uno de ellos en función de lo que deseen implementar. También es responsabilidad de los desarrolladores que utilizan el módulo y dispositivo el

definir las posibles acciones a ejecutar por parte del usuario, así como los objetos que serán afectados por las mismas y cómo se verán reflejadas en el espacio tridimensional.

2.2 Requerimientos del sistema

Los requerimientos de software son las capacidades que debe alcanzar o poseer un sistema o componente de un sistema para satisfacer un contrato, estándar, especificación u otro documento formal[37]. Estos se pueden clasificar en funcionales y no funcionales. Los requerimientos o requisitos funcionales son capacidades o condiciones que el sistema debe cumplir, mientras que los no funcionales describen las características y aspectos de carácter técnicos que el sistema debe poseer[36].

2.2.1 Requisitos funcionales

RF1 Calibrar el dispositivo

RF2 Conectar Arduino con la PC

RF3 Enviar a la PC los datos de la posición del brazo y el estado del actuador

RF4 Solicitar a Arduino el estado de los sensores y actuadores

RF5 Gestionar los gameobjects para Unity que representan el brazo y el antebrazo.

2.2.2 Requisitos no funcionales

Tabla 2 Requisitos no funcionales de Rendimiento

Atributo de Calidad	Rendimiento
Sub-atributos/Sub-características	Comportamiento en el tiempo
Objetivo	El sistema debe realizar la transferencia de datos entre Arduino y la PC en un tiempo inferior a 0,5 segundos. (Rendimiento)
Origen	
Artefacto	
Entorno	
Estímulo	**Respuesta: Flujo de eventos**
1.a Interacción con el sistema	
Medida de respuesta	

Tabla 3 Requisitos no funcionales de Portabilidad

Atributo de Calidad	Portabilidad
Sub-atributos/Sub-características	
Objetivo	El sistema debe permitir su migración a GNU/Linux y MacOS
Origen	
Artefacto	
Entorno	
Estímulo	**Respuesta: Flujo de eventos**
1.a Interacción con el sistema	
Medida de respuesta	

Tabla 4 Requisitos no funcionales de Fiabilidad

Atributo de Calidad	Fiabilidad
Sub-atributos/Sub-características	Capacidad de recuperación
Objetivo	El sistema debe ser susceptible al reconocimiento de errores producidos por el hardware
Origen	
Artefacto	
Entorno	
Estímulo	**Respuesta: Flujo de eventos**
1.a Interacción con el sistema	
Medida de respuesta	

Tabla 5 Requisitos no funcionales de Compatibilidad

Atributo de Calidad	Compatibilidad
Sub-atributos/Sub-características	
Objetivo	El sistema debe ser compatible con Windows 10
Origen	
Artefacto	
Entorno	
Estímulo	**Respuesta: Flujo de eventos**
1.a Interacción con el sistema	
Medida de respuesta	
Objetivo	La aplicación Arduino debe poder ejecutarse en Arduino Mega 2560 o superior
Origen	
Artefacto	
Entorno	
Estímulo	**Respuesta: Flujo de eventos**
1.a Interacción con el sistema	
Medida de respuesta	

Tabla 6 Requisitos no funcionales de hardware

Atributo de Calidad	Hardware
Sub-atributos/Sub-características	
Objetivo	Para el correcto funcionamiento del sistema es necesario como mínimo, un microprocesador Intel Celeron a 1.60 GHz, 4 GB RAM, 64 bits de profundidad de color y 2128 MB de memoria de video o 128 MB

	de video dedicado, compatible con Open GL o DirectX 3D
Origen	
Artefacto	
Entorno	
Estímulo	**Respuesta: Flujo de eventos**
1.a Interacción con el sistema	
Medida de respuesta	

2.3 Fase de Planeación

En esta fase se crean las historias de usuario que describen la salida necesaria, características y funcionalidades del software que se va a elaborar. Cada historia es escrita por el cliente, el cual le asigna una prioridad en base al valor general de la característica o función para el negocio. Los programadores analizan cada una y le asignan un costo medido en semanas de desarrollo. Junto al cliente, los programadores determinan el orden en que se van a desarrollar las historias de usuario y después de la primera entrega, calculan la velocidad del proyecto para estimar la fecha de las entregas posteriores, hasta la finalización del proyecto.[38]

2.3.1 Historias de Usuario

A continuación, se muestran algunas de las historias de usuario que se elaboraron para el desarrollo del sistema. Las restantes se encuentran descritas en el Anexo 1.

Tabla 7 Historia de usuario 3

Historia de Usuario	
Número: 3	**Nombre:** Obtener los datos del sensor BNO055 y MPU 6050
Usuario: Programador	**RF3:** Enviar a la PC los datos de la posición del brazo y el estado del actuador
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Puntos Estimados:	**Iteración Asignada:**
Programador Responsable: Jorge Enrique Rodríguez Jiménez	

Descripción: Arduino obtiene los datos de rotación a partir de los sensores BNO055 y MPU 6050
Observaciones: Los datos provenientes del sensor BNO055 deben adquirirse en el sistema angular de Quaternion

Tabla 8 Historia de usuario 4

Historia de Usuario	
Número: 4	**Nombre:** Procesar datos del sensor MPU 6050
Usuario: Programador	**RF3:** Enviar a la PC los datos de la posición del brazo y el estado del actuador
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Puntos Estimados:	**Iteración Asignada:**
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Arduino procesa los datos provenientes del sensor MPU 6050 y los convierte al sistema angular de Euler	
Observaciones: Para el desarrollo del sistema solo serán necesarios los valores de rotación correspondientes a las coordenadas X y Y	

Tabla 9 Historia de usuario 6

Historia de Usuario	
Número: 6	**Nombre:** Enviar a través de la comunicación serial los datos obtenidos
Usuario: Programador	**RF3:** Enviar a la PC los datos de la posición del brazo y el estado del actuador
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Puntos Estimados:	**Iteración Asignada:**
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Arduino envía a la aplicación desarrollada en Unity los datos provenientes de los sensores a través del puerto serie	
Observaciones:	

Tabla 10 Historia de usuario 9

Historia de Usuario	
Número: 9	**Nombre:** Actualizar los gameobjects
Usuario: Programador	**RF5:** Gestionar los gameobjects para Unity que representan el brazo, el antebrazo y la mano
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Puntos Estimados:	**Iteración Asignada:**
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Se actualizan las transformaciones de rotación asignadas a los GameObject relacionados con el brazo y el antebrazo.	
Observaciones:	

2.3.2 Estimación de esfuerzo por Historias de Usuario

La estimación de esfuerzo asociado a la implementación de las historias de usuario la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos.

Tabla 11 Estimación del esfuerzo por Historia de Usuario

Iteración	**Historia de Usuario**		**Puntos Estimados (Semanas)**
1	**1**	Calibrar el dispositivo	**1.5**
	2	Conectar Arduino con la PC	**0.5**
2	**3**	Obtener los datos del sensor BNO055 y MPU 6050	**1.0**
	4	Procesar datos del sensor MPU 6050	**1.5**
	5	Determinar el estado del actuador	**0.5**
3	**6**	Enviar a través de la comunicación serial los datos obtenidos	**1.5**
	7	Solicitar a Arduino el estado de los sensores y actuadores	**1.5**
4	**8**	Procesar los datos provenientes de arduino	**1.5**
	9	Actualizar los gameobjects	**1.5**
Total			**11.0**

2.3.3 Duración de las iteraciones y Plan de entrega

Las iteraciones deben tener un tiempo de duración de no más de tres semanas. En la primera iteración se puede establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura, sin embargo, esto no es siempre posible ya que es el cliente quien decide que historias se implementaran en cada iteración.

Tabla 12 Duración de las iteraciones

Iteración	Historia de Usuario		Puntos Estimados (Semanas)
1	**1**	Calibrar el dispositivo	**2.0**
	2	Conectar Arduino con la PC	
2	**3**	Obtener los datos del sensor BNO055 y MPU 6050	**3.0**
	4	Procesar datos del sensor MPU 6050	
	5	Determinar el estado del actuador	
3	**6**	Enviar a través de la comunicación serial los datos obtenidos	**3.0**
	7	Solicitar a Arduino el estado de los sensores y actuadores	
4	**8**	Procesar los datos provenientes de arduino	**3.0**
	9	Actualizar los gameobjects	
Total			**11.0**

Tabla 13 Plan de entrega

Entrega	Iteración 1 (4ta semana de febrero)	Iteración 2 (2da semana de marzo)	Iteración 3 (1era semana de abril)	Iteración 4 (4ta semana de abril)
Entrega 1	Versión 1	Versión 2	Versión 3	Versión 4
Entrega 2	-	Versión 1	Versión 2	Versión 3
Entrega 3	-	-	Versión 1	Versión 2
Entrega 4	-	-	-	Versión 1

Desarrollo de las iteraciones

- Iteración 1: Esta iteración tiene como objetivo la implementación de las historias de usuario 1 y 2. Las cuales corresponden a la calibración del dispositivo y la conexión

entre el Arduino y la computadora mediante la implementación de una comunicación serial.

- Iteración 2: Esta iteración tiene como objetivo la implementación de las historias de usuario 3, 4 y 5. Las cuales corresponden a la obtención de datos físicos a partir de las lecturas de los sensores y el procesado de los datos provenientes del MPU 6050, así como la adquisición del estado del actuador.
- Iteración 3: Esta iteración tiene como objetivo la implementación de las historias de usuario 6 y 7. Las cuales corresponden a la comunicación del sistema Unity con Arduino, así como el envío y recepción de datos por parte de ambas plataformas mediante la comunicación serial establecida.
- Iteración 4: Esta iteración tiene como objetivo la implementación de las historias de usuario 8 y 9. Las cuales corresponden al procesamiento y almacenamiento de los datos provenientes de Arduino y la posterior actualización del estado de los gameobjects.

2.4 Fase de Diseño

En esta fase se plantean los artefactos necesarios para guiar la implementación de las historias de usuario conforme se van desarrollando, se identifica la arquitectura y los patrones arquitectónicos del sistema y se elaboran las tarjetas Clase, Responsabilidad y Colaboración (CRC) para la presentación del sistema.[38]

2.4.1 Arquitectura

La arquitectura de software puede ser vista como la estructura del sistema en función de la definición de los componentes y sus interacciones. El sistema se desarrolla sobre la plataforma Arduino, así como sobre el motor gráfico Unity 3D. Posee una arquitectura maestro-esclavo, la cual se describe a continuación:

La arquitectura maestro-esclavo consta de varios bucles paralelos, cada uno puede ejecutar tareas a distintas velocidades. Un bucle actúa como maestro y los otros como esclavos. Esta arquitectura es comúnmente usada en sistemas de tiempo real donde puede haber procesadores separados asociados con la recolección de datos del entorno del sistema, procesamiento de datos y computación. El proceso maestro, es el responsable de la computación, la coordinación, las comunicaciones y el control de los procesos esclavos, los cuales se encargan de acciones específicas, por lo general, se asocian al proceso de adquisición de datos.[36], [37] Para el desarrollo del sistema se plantea el siguiente modelo:

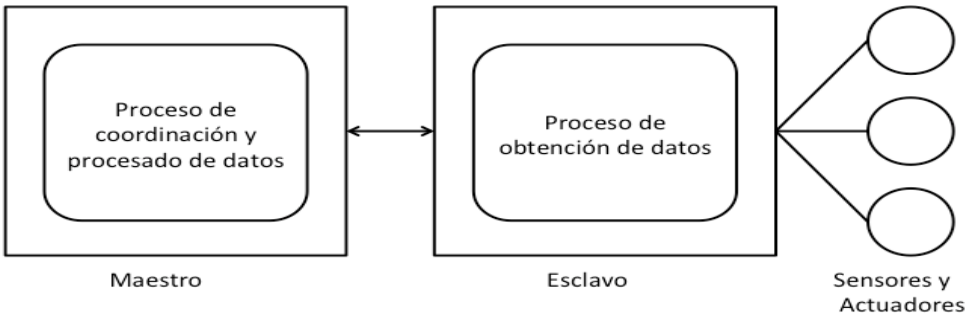

Figura 14 Modelo de arquitectura

Para garantizar el correcto funcionamiento y retroalimentación del sistema, se establece el siguiente modelo de secuencia:

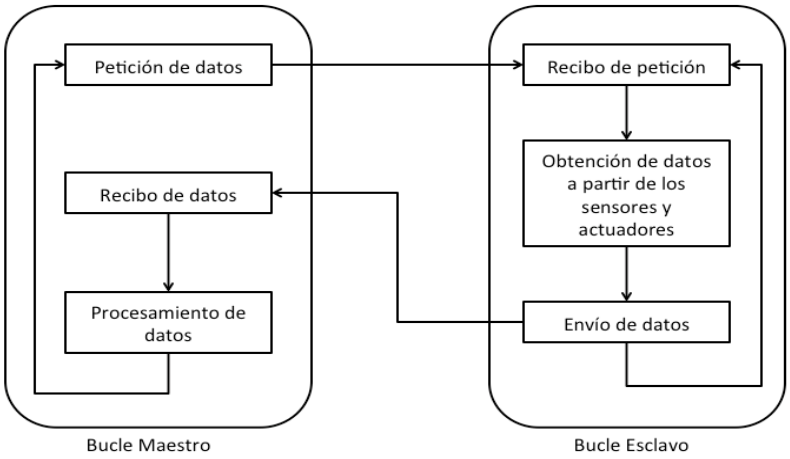

Figura 15 Modelo de secuencia

2.4.2 Patrones de diseño

Los patrones son soluciones comunes a problemas de diseño de software orientado a objetos y que además poseen ciertas características de efectividad para resolver ese problema. Son reusables ya que pueden ser aplicados en otros diseños o problemas.[37]

Los patrones de software para la asignación general de responsabilidades (GRASP, por las siglas del inglés General Responsibility Assignment Software Patterns) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones.[39], [40] A continuación, se describen aquellos patrones utilizados de acuerdo a las soluciones que brindan dentro del contexto del sistema:

- Experto: Se asigna la responsabilidad a la clase que posee la información necesaria para llevar la tarea a cabo. Esto se evidencia en la clase ConectarConUnity.ino.
- Controlador: Una clase actúa como intermediaria para el manejo de eventos. Esto

se evidencia en la clase Conexión.cs.

Los patrones de diseño Banda de los cuatro, (GOF por las siglas del inglés Gang of Four) son una serie de posibles soluciones a problemas que suelen ser comunes en el desarrollo del software, describiendo las formas en que diferentes tipos de objetos pueden ser organizados para trabajar unos con otros[40], [41]. A continuación, se describen aquellos patrones utilizados de acuerdo a las soluciones que brindan dentro del contexto del sistema:

Comportamiento:

- Command: Encapsula una operación a un objeto, permitiendo que esta se ejecute sin necesidad de conocer el contenido de la misma. Esto se evidencia en la clase ObjetoCargable.cs y en la clase Movimientos.cs.
- Mediator: Define un objeto que coordine la comunicación entre objetos de distintas clases, pero que funcionan como un conjunto. Esto se evidencia en el objeto arduino de tipo SerialPort.
- State: Permite que un objeto modifique su comportamiento cada vez que cambie su estado interno. Esto se evidencia en los objetos con la propiedad escargable de la clase ObjetoCargable.cs.

Estructurales:

- Composite: Permite tratar objetos compuestos como si se tratara de uno simple. Esto se evidencia en la clase Movimientos.cs donde se hace uso de objetos tipo GameObject.
- Flyweight: Reduce la redundancia cuando gran cantidad de objetos poseen idéntica información. Esto se evidencia en los objetos que hacen uso de la clase ObjetoCargable.cs pues adquieren la propiedad escargable.

Creación:

- Singleton: Garantiza la existencia de una única instancia para una clase y la creación de un punto de acceso global para la misma. Esto se evidencia en la clase Conexión.cs.

2.4.3 Tarjetas CRC

El modelado CRC proporciona una manera sencilla de identificación y organización de las clases que son relevantes para los requerimientos de un sistema o producto Las tarjetas CRC son fichas en las que se describen brevemente las responsabilidades de las clases y una lista de los objetos con los que colabora para llevar a cabo esas responsabilidades. A continuación, se describes algunas de estas tarjetas, las restantes se encuentran en el Anexo 2.

Tabla 14 Tarjeta CRC de la clase ConectarConUnity

Tarjeta CRC	
Clase: ConectarConUnity.ino	
Responsabilidad	**Colaboración**
• Establecer la conexión con la computadora • Obtener mensajes a través del puerto serial • Enviar datos a través del puerto serial • Obtener datos de los sensores • Obtener estado de los actuadores	Adafruit_BNO055 MPU6050 Wire

Tabla 15 Tarjeta CRC de la clase Conexion

Tarjeta CRC	
Clase: Conexion.cs	
Responsabilidad	**Colaboración**
• Establecer la conexión con Arduino • Enviar mensajes a través del puerto serial • Obtener datos a través del puerto serial • Procesar y almacenar los datos	SerialPort

en las variables designadas	

Tabla 16 Tarjeta CRC de la clase Movimientos

Tarjeta CRC	
Clase: Movimientos.cs	
Responsabilidad	**Colaboración**
• Obtiene los valores de rotación para los GameObject • Actualiza los valores de rotación para los GameObject	Conexión.cs Transform

Tabla 17 Tarjeta CRC de la clase EjecutarAccion

Tarjeta CRC	
Clase: EjecutarAccion	
Responsabilidad	**Colaboración**
• Obtiene los datos referentes al estado del botón de acción	Conexión.cs

2.4.4 Estándares de codificación

Para el desarrollo del sistema se utiliza un estándar para la escritura del código fuente a partir de referencias a buenas prácticas de programación en los lenguajes que se emplearon. Ejemplo: nombre de las variables, los métodos y las clases. Todo esto respetando los estilos de codificación de los lenguajes Wiring[24] y C#[34] respectivamente.

Los estándares de codificación propuestos se enfocan en la legibilidad del código ya que esto repercute de forma directa en la comprensión del sistema por parte del programador. Además, el mantenimiento del sistema es más fácil y puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores o mejorar el rendimiento.

Lenguaje Arduino (Wiring):

Clases: Los nombres comienzan con letra mayúscula y en caso de ser un nombre compuesto, la definición debe ser continua y cada palabra debe iniciar con letra mayúscula. Ejemplo:

ConectarConUnity

Declaración de variables: Los nombres de las variables comienzan con letra minúscula, en caso de ser un nombre compuesto, la definición debe estar separada por guión bajo y cada palabra debe iniciar con letra minúscula. Ejemplo:

float acción;

float ang_x;

Métodos: En caso de los métodos, el nombre debe comenzar con letra minúscula, en caso de ser un nombre compuesto, la representación debe ser continua y cada palabra debe comenzar con letra minúscula. Ejemplo:

void enviardatos()

Estilo de indentación: El estilo de indentación utilizado es propio para lenguajes de programación que utilizan llaves, teniendo en cuenta las preferencias del programador. Ejemplo:

// método

void loop(){

instrucción 1;

instrucción 2;

}

//condición

if(condición) {

instrucción 1;

}else{

instrucción 2;

}

Lenguaje C#:

Clases: Los nombres comienzan con mayúscula y en caso de ser compuestos, se deben representar de forma continua y cada palabra comienza con letra mayúscula. Ejemplo:

public class EjecutarAccion { }

Declaración de variables: Los nombres de las variables comienzan con letra minúscula, en caso de ser un nombre compuesto, la definición debe ser continua y cada palabra debe comenzar con letra minúscula. Ejemplo:

private float mrotx;

public bool escargable;

Métodos: Los métodos comienzan con mayúscula y en caso de ser nombres compuestos, se representan de forma continua y cada palabra comienza con letra mayúscula. Ejemplo:

public void MensajeArduino()

Estilo de indentación: El estilo de indentación utilizado, es propio para lenguajes de programación que utilizan llaves para separar e identificar bloques de código, teniendo en cuenta las preferencias del programador. Ejemplo:

```
// método

public void MensajeArduino(){

instrucción 1;

instrucción 2;

}

//condición

if(condición) {

instrucción 1;

}

else{

instrucción 2;

}
```

Conclusiones parciales

Al culminar el presente capítulo, se definió la arquitectura sobre la cual se construye el prototipo, además, se obtuvieron los artefactos generados por la metodología empleada, los cuales permitieron una mejor comprensión de las funcionalidades, elementos y guía de trabajo durante el desarrollo del sistema. Las historias de usuario permitieron la implementación ordenada del software en correspondencia con los requisitos planteados por el cliente. La arquitectura maestro-esclavo constituyó una forma de representación acertada en relación al funcionamiento del sistema. El esclarecimiento de los estándares de codificación a seguir, facilitó el proceso de implementación por parte del equipo de desarrollo.

Capítulo 3: Implementación y Pruebas

En este capítulo se describe la fase implementación y prueba según XP. Se identifican y organizan las clases relevantes para las funcionalidades del sistema, así como los resultados obtenidos en el proceso de pruebas que se llevaron a cabo. Estas se realizaron con el objetivo de que los desarrolladores pudieran probar las funcionalidades descritas mediante las historias de usuario y validar la calidad del sistema, el rendimiento y las características esperadas por el cliente.

3.1 Fase de implementación

Para describir las tareas llevadas a cabo en la fase de implementación, se emplea un lenguaje técnico, el cual no necesariamente debe ser entendible por el cliente. Dichas tareas son asignadas al equipo o programador responsable, normalmente la codificación se lleva a cabo por una pareja de programadores. Esta labor se lleva a cabo con el objetivo de detallar mejor las historias de usuario, lo cual facilita el entendimiento en el proceso de implementación. Cada historia de usuario puede contener una o más tareas de ingeniería, explicando de forma general las acciones que se realizan en la misma.[36], [38]

3.1.1 Tareas de ingeniería

A continuación, se describen las tareas asociadas a las Historias de Usuario 3, 4, 6 y 9. Las restantes tareas se encuentran descritas en el Anexo 3.

Tabla 18 Tarea de Desarrollo 3

Tarea	
Número de tarea: 3	**Número de historia de usuario:** 3
Nombre de la tarea: Obtener los datos de los sensores	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.0
Fecha de inicio: 16 de marzo 2020	**Fecha fin:** 23 de marzo 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite al sistema obtener los datos de rotación provenientes de los sensores BNO055 y MPU 6050	

Tabla 19 Tarea de desarrollo 4

Tarea	
Número de tarea: 4	**Número de historia de usuario:** 4
Nombre de la tarea: Procesar los datos del MPU 6050	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.5
Fecha de inicio: 24 de marzo 2020	**Fecha fin:** 2 de abril 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite procesar los datos recibidos del sensor MPU 6050 en ángulos de Euler	

Tabla 20 Tarea de desarrollo 6

Tarea	
Número de tarea: 6	**Número de historia de usuario:** 6
Nombre de la tarea: Enviar los datos mediante la comunicación serial	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.5
Fecha de inicio: 6 de abril 2020	**Fecha fin:** 25 de abril 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite a Arduino enviar los datos obtenidos a Unity, a través de una comunicación mediante el puerto serie	

Tabla 21 Tarea de desarrollo 9

Tarea	
Número de tarea: 9	**Número de historia de usuario:** 9
Nombre de la tarea: Actualizar los valores de rotación de los gameobjects	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.0
Fecha de inicio: 7 de mayo 2020	**Fecha fin:** 14 de mayo 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite actualizar los gameobjects correspondientes al Brazo y Antebrazo con los valores de rotación recibidos.	

Tabla 22 Tarea de desarrollo 10

Tarea	
Número de tarea: 10	**Número de historia de usuario:** 9
Nombre de la tarea: Actualizar estado de los actuadores en los gameobjects	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 15 de mayo 2020	**Fecha fin:** 18 de mayo 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite actualizar los gameobjects con los valores correspondientes a los estados de los actuadores	

3.1.2 Diagrama de despliegue

El diagrama de despliegue muestra las relaciones físicas entre los componentes: hardware y software, en el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software (procesos y objetos que se ejecutan en ellos).[36], [37]

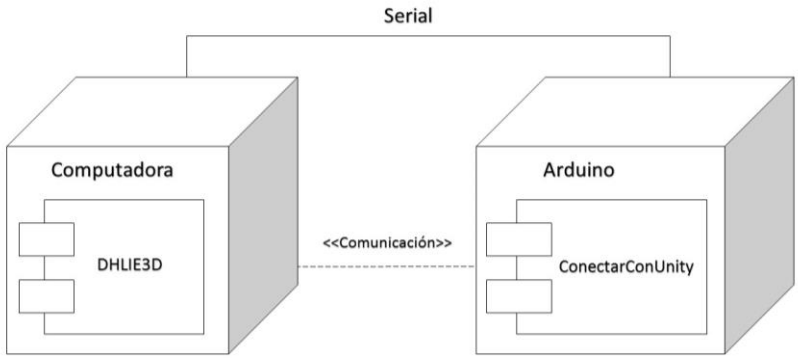

Figura 16 Diagrama de despliegue

3.1.3 Implementación de la solución

El dispositivo diseñado capta a través de los sensores conectados a la placa Arduino el movimiento del miembro superior (brazo) y lo envía a la computadora para que este sea representado en el entorno 3D.

Para el correcto funcionamiento del dispositivo es necesario calibrar el módulo MPU_6050 puesto que en su configuración original trae valores de desplazamientos predefinidos que no se corresponden con los necesarios para el funcionamiento esperado del mismo. Este

proceso de calibración solo se realiza una vez, no es necesario repetirlo cada vez que se utilice el dispositivo.

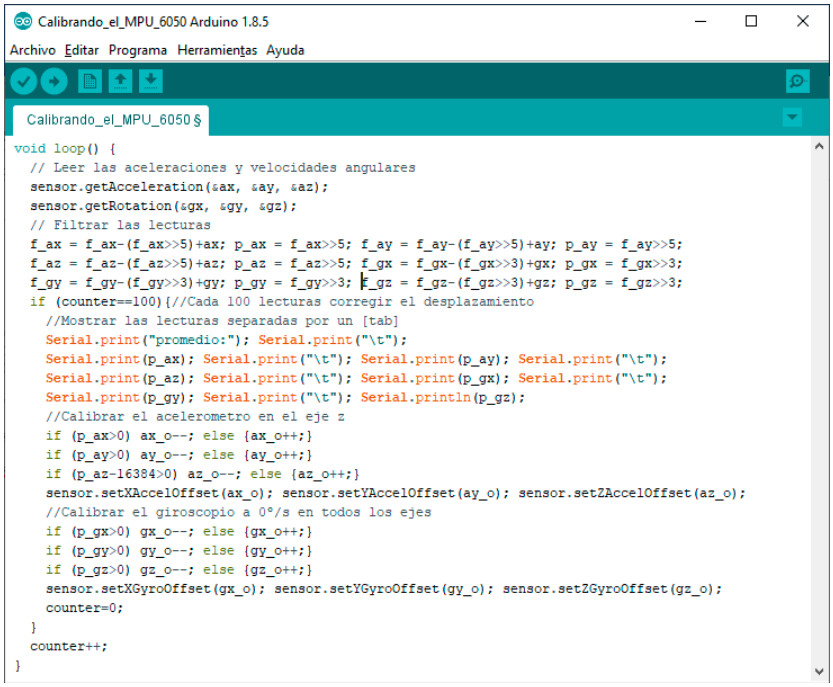

Figura 17 Calibración del MPU_6050

Otro proceso vital es el envío a través de comunicación serial de los datos a la PC. Función que se ilustra en la siguiente figura:

Figura 18 Comunicación a través del puerto serial

3.2 Fase de Pruebas

Una de las pautas fundamentales de XP es el proceso de pruebas. Esta metodología contempla la realización de las pruebas unitarias y las pruebas de aceptación fundamentalmente, e insiste en realizar pruebas a cada nueva funcionalidad que se implementa antes de pasar al desarrollo de la siguiente.[38] Sin embargo, dada las características de la solución es necesario realizar también pruebas de integración para evaluar el funcionamiento del sistema.

Un correcto proceso de pruebas permite aumentar la calidad de los sistemas disminuyendo el número de posibles errores futuros y reduciendo el tiempo de detección de los mismos. También permite mantener la estabilidad del software, aumentando la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones.

Las pruebas deben reunir un conjunto de características para que se obtengan buenos resultados:[36], [42]

- Crear la prueba abstrayéndose del futuro código, de esta forma se logra la independencia de esta respecto al código que evalúa.
- Observar la refactorización. La prueba permite verificar que un cambio en la estructura del código no tiene por qué cambiar su funcionamiento.
- Realizar pruebas a las funcionalidades generales que debe cumplir el programa especificado en las historias de usuario.

Las pruebas unitarias son creadas por los desarrolladores, son una técnica de caja blanca y consisten en comprobar la lógica interna del software[42]. Estas se fueron aplicando luego de concluir cada tarea de implementación para asegurar que no se arrastraran errores de codificación a la próxima etapa.

3.2.1 Pruebas de integración

Las pruebas de integración se encargan de probar el correcto funcionamiento e integración entre diferentes componentes o módulos.[43] Estas pruebas son creadas con el objetivo de comprobar el funcionamiento del sistema en general.

Teniendo en cuenta de las características de la solución, compuesta por un componente de hardware y un componente de software (firmware), resulta necesario comprobar que ambos son capaces de funcionar en conjunto y de la forma esperada.

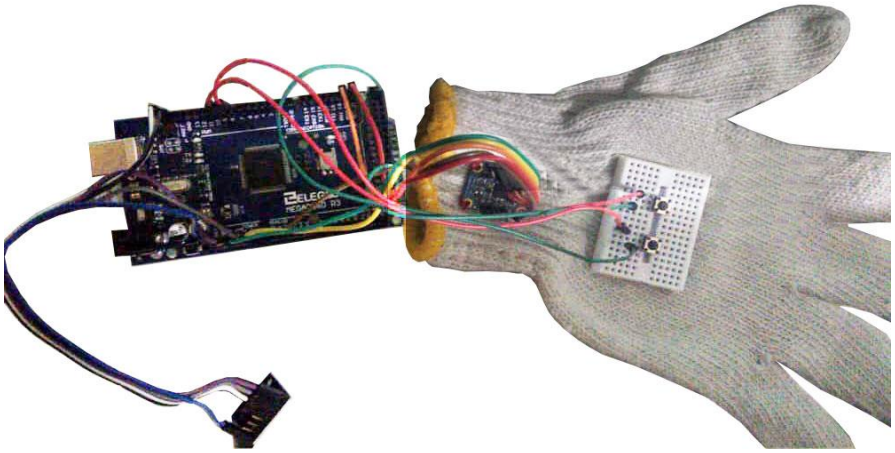

Figura 19 Prototipo del dispositivo

Por tal motivo para la validación del sistema se desarrolló un entorno de prueba constituido por los objetos y reglas que se exponen a continuación:

- El usuario contará con una representación virtual del miembro superior derecho (brazo, antebrazo y mano), del cual podrá mover libremente el brazo y el antebrazo.
- En la escena se ubicarán tres objetos en forma de cubo al alcance del usuario. Estos objetos se verán afectados por las físicas del entorno (colisiones y gravedad).
- En la escena se ubicará un objeto con forma de arma o pistola, al alcance del usuario. La misma se verá afectada por las físicas del terreno.
- Se ubicarán varios objetos tipo cubos a distancia, para ser utilizados como blancos. Los mismos se verán afectados por las físicas del terreno.

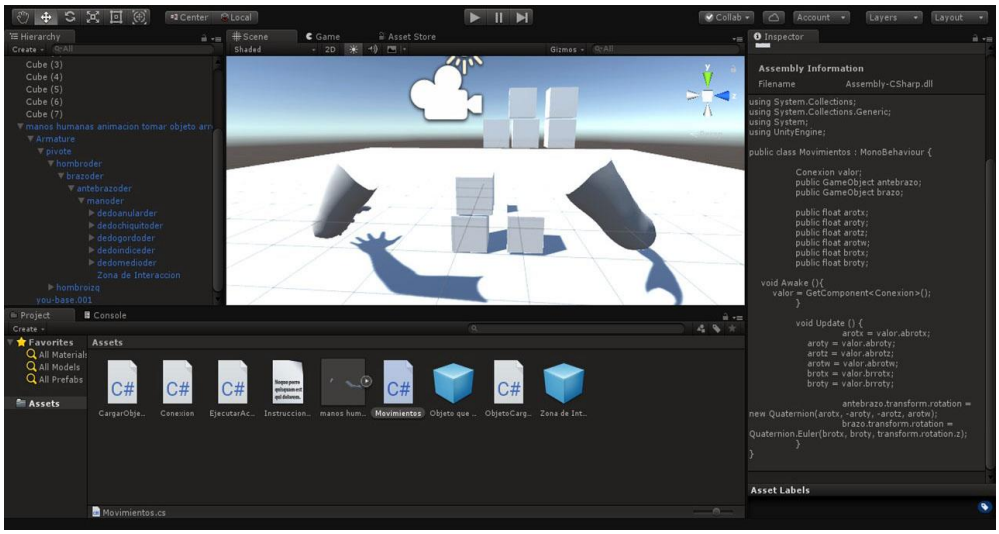

Figura 20 Entorno creado para las pruebas de integración

Para la correcta realización de la prueba, en una primera instancia el usuario debe cambiar la posición de los objetos tipo cubo, con esto se verifica el funcionamiento de los sensores de rotación y el botón destinado a la acción de cargar objetos. Una vez ejecutado lo anterior, el usuario debe tomar el arma y realizar disparos a los objetos tipo blanco, con esto se comprueba el funcionamiento del botón destinado a las acciones secundarias y se valida la posibilidad de utilizar ambos botones al mismo tiempo.

A continuación, se muestra una imagen del entorno creado para la ejecución de las pruebas de integración, específicamente la acción de cargar un objeto para moverlo dentro de la escena.

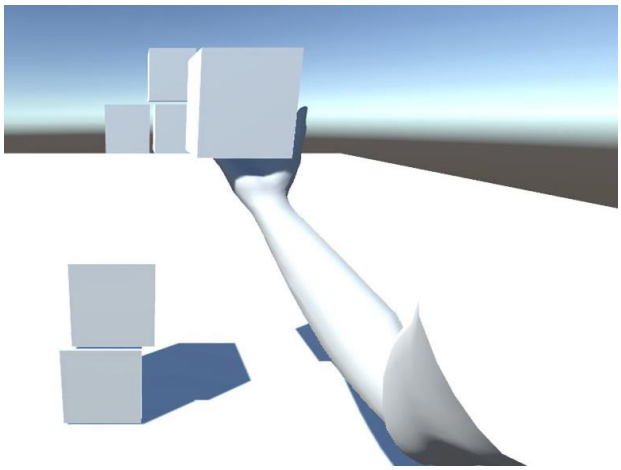

Figura 21 Ejecución de las pruebas de integración

Durante la ejecución de las pruebas tanto el sistema como el hardware se comportó de la manera esperada y se evidenció la necesidad de la calibración para el correcto funcionamiento del MPU 6050.

3.2.2 Pruebas de aceptación

Las pruebas de aceptación son creadas sobre la base de las historias de usuarios en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una HU ha sido correctamente implementada. Las pruebas de aceptación son consideradas como pruebas de caja negra. Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos.[36], [38]

Este tipo de pruebas representan la satisfacción del cliente, es por este motivo que deben ser diseñadas por el mismo. Siguiendo los preceptos de la metodología seleccionada, se tomó como criterio de aceptación el 100% de los casos de pruebas satisfactorios para pasar

a la siguiente iteración de la etapa de desarrollo. Para el diseño de las pruebas de aceptación se definieron los siguientes elementos:

- Código de la prueba: Representa el caso de prueba, incluye el número de la historia de usuario y de la prueba.
- Nombre de la historia de usuario: Incluye el nombre de la historia de usuario.
- Descripción de la prueba: Acción que debe realizar el sistema.
- Condiciones de ejecución: Características y elementos que debe contener el sistema para ejecutar el caso de prueba.
- Pasos de ejecución: Incluye los pasos necesarios para realizar la prueba.
- Resultados esperados: Respuesta que el sistema debe dar ante la ejecución de la prueba.
- Clasificación de la prueba: Clasificación de la prueba en satisfactoria o no satisfactoria.

A continuación, se presentan los casos pruebas de aceptación diseñados y aplicados a las historias de usuarios 3, 4, 6 y 9. Los restantes casos de prueba se encuentran en el Anexo 4:

Tabla 23 Caso de Prueba de Aceptación HU3-P1

Caso de prueba de aceptación	
Código de la prueba: HU3-P1	**Nombre de la historia de usuario:** Obtener los datos del sensor BNO055 y MPU 6050
Descripción de la prueba: La prueba consiste en verificar que Arduino recibe correctamente los datos de rotación correspondientes al sensor BNO055	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • El BNO055 debe estar conectado a Arduino	
Pasos de ejecución: • Abrir el monitor serie	
Resultados esperados: El monitor serie se abre y muestra los valores de rotación correspondientes	
Clasificación de la prueba: Satisfactoria	

Tabla 24 Caso de Prueba de Aceptación HU3-P2

Caso de prueba de aceptación	
Código de la prueba: HU3-P2	**Nombre de la historia de usuario:** Obtener los datos del sensor BNO055 y MPU 6050
Descripción de la prueba: La prueba consiste en verificar que Arduino recibe correctamente los datos de rotación correspondientes al sensor MPU 6050	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • El MPU 6050 debe estar conectado a Arduino	
Pasos de ejecución: • Abrir el monitor serie	
Resultados esperados: El monitor serie se abre y muestra los valores de rotación correspondientes	
Clasificación de la prueba: Satisfactoria	

Tabla 25 Caso de Prueba de Aceptación HU4-P2

Caso de prueba de aceptación	
Código de la prueba: HU4-P2	**Nombre de la historia de usuario:** Procesar los datos del sensor MPU 6050
Descripción de la prueba: La prueba consiste en verificar que se han procesado correctamente los datos provenientes del sensor MPU 6050	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • El MPU 6050 debe estar conectado a Arduino	
Pasos de ejecución: • Abrir el monitor serie	
Resultados esperados: El monitor serie se abre y muestra los valores de rotación en formato de Euler (360º)	

Clasificación de la prueba:
Satisfactoria

Tabla 26 Caso de Prueba de Aceptación HU6-P1

Caso de prueba de aceptación	
Código de la prueba: HU6-P1	**Nombre de la historia de usuario:** Enviar a través de la comunicación serial los datos obtenidos
Descripción de la prueba: La prueba consiste en verificar que Arduino envía correctamente información a través de la comunicación serial establecida con Unity	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • Los sensores BNO055 y MPU 6050 deben estar conectados a Arduino	
Pasos de ejecución: • Ejecutar programa en Unity • Abrir terminal de Unity	
Resultados esperados: Se abre la terminal y aparece una cadena con los datos provenientes de los sensores	
Clasificación de la prueba: Satisfactoria	

Tabla 27 Caso de Prueba de Aceptación HU9-P2

Caso de prueba de aceptación	
Código de la prueba: HU9-P2	**Nombre de la historia de usuario:** Actualizar los gameobjects
Descripción de la prueba: La prueba consiste en comprobar que los gameobjects reaccionan correctamente a los cambios de estado de los sensores y actuadores	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • Los sensores BNO055 y MPU6050 deben estar conectados a Arduino	

• Al menos debe haber un botón conectado a Arduino
Pasos de ejecución: • Se ejecuta el programa en Unity
Resultados esperados: Los objetos asociados a los gameobjects Brazo y Antebrazo rotan en correspondencia a la orientación de los sensores. Al pulsar el botón el jugador carga un objeto
Clasificación de la prueba: Satisfactoria

3.2.3 Análisis de los resultados de las pruebas

Al concluir las pruebas en cada una de las iteraciones se identificaron algunas no conformidades, las cuales fueron resueltas antes de avanzar a la siguiente etapa de desarrollo. En la primera iteración se desarrollaron 2 casos de pruebas, en los que se obtuvieron resultados satisfactorios, por lo que se decidió pasar a la siguiente iteración.

Para la segunda iteración se realizaron 4 casos de pruebas, en los que se detectó 1 no conformidad relacionada con la ejecución de la historia de usuario número 4. Después de la corrección de la misma se procedió a desarrollar 2 casos de pruebas sobre la historia de usuario en cuestión, en los que se obtuvieron resultados satisfactorios dando por cumplido los criterios de aceptación, por lo que se decidió pasar a la siguiente iteración.

Para la tercera iteración se ejecutaron 2 casos de pruebas, en los que se detectó 1 no conformidad relacionada con la ejecución de la historia de usuario número 7. Después de la corrección se procedió a desarrollar 2 casos de pruebas sobre la historia de usuario en cuestión, en los que se obtuvieron resultados satisfactorios, por lo que se decidió pasar a la siguiente iteración.

Para la cuarta iteración se desarrollaron 2 casos de prueba, en los que se obtuvieron 2 no conformidades relacionadas a la ejecución de las historias de usuario número 8 y 9 respectivamente. Después de realizar el proceso de corrección se procedió a realizar 4 casos de prueba, dos a cada historia de usuario, en los que se obtuvieron resultados satisfactorios, cumpliendo con los criterios de aceptación y dando por concluido el proceso de desarrollo del sistema con un 100% de resultados satisfactorios. A continuación, se presenta un gráfico con la relación de los resultados obtenidos en la ejecución de los casos de pruebas para cada iteración:

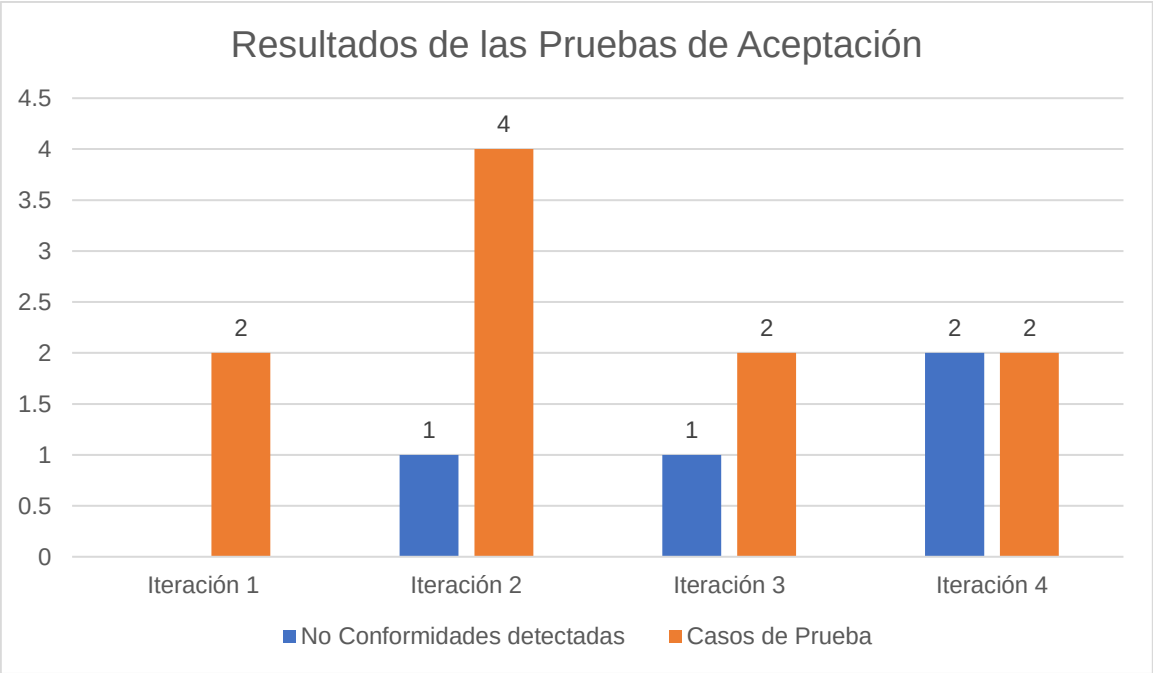

Figura 22 Resultado de las pruebas de aceptación

3.3 Cumplimiento de los resultados de la investigación

Desde un principio se determinó como hipótesis de trabajo en esta investigación que la introducción de un Dispositivo de Interacción Humana para entornos 3D en los productos de VERTEX permitirá elevar los niveles de interacción e inmersión de los usuarios de los mismos. Partiendo de la premisa de poder manipular directamente los objetos de la escena con las propias manos o en este caso con un dispositivo soportado sobre las manos, se pueden extender los criterios de percepción e inmersión, al tener el usuario que hacer los mismos movimientos y gestos que si lo hiciera en un escenario real.

El escenario de prueba ideal hubiera sido la adaptación del módulo a alguno de los sistemas ya probados en el centro y realizar pruebas con grupos de usuarios en las variantes con y sin dispositivo. Lamentablemente las condiciones de la pandemia de la COVID 19 no ha permitido la realización de este tipo de pruebas, por lo que resulta necesario utilizar otro método para validar el cumplimiento de la hipótesis de trabajo. El autor del trabajo se decanta por seguir el método de criterios de expertos sometiendo el trabajo al juicio de personas con experiencias en la temática para eliminar aspectos irrelevantes y/o modificar aquellos que lo requieran.

3.3.1 Selección de los expertos

Para la selección de los expertos se elaboró un listado preliminar de 30 especialistas o personas de la UCI y otros entornos relacionados con el campo; partiendo de que a mayor cantidad de expertos se posea, menor es el error que se incurre en la calidad o resultado de la evaluación. A continuación, se pasó a determinar el coeficiente de competencia de los candidatos a expertos (K) y para ello se tuvo en cuenta:

- Campo de trabajo actual en las temáticas de la Realidad Virtual, Sistemas empotrados, mecatrónica o Internet de las cosas.
- Tiempo que lleva en el desarrollo de estos temas.
- Investigaciones desarrolladas en el tema
- Publicaciones en los últimos 5 años

De los 30 especialistas 21 quedaron con un coeficiente entre 1 y 0.8 (Alto y medio) y se decidió dejar solo a estos como candidatos finales. Luego de comunicarse con ellos explicando el proceso solo 16 dieron su consentimiento a participar y respondieron el cuestionario (ver anexo 5). La siguiente tabla muestra la procedencia de los expertos.

Tabla 28 Procedencia de los expertos para la validación de la propuesta de solución

Procedencia	Cantidad
UCI	**9**
CUJAE	**3**
UNICA	**2**
UCLV	**1**
UHo	**1**
Total	**16**

3.3.2 Procesamiento de los resultados

Al tabular los resultados se obtiene una tabla con la puntuación dada por cada experto a cada uno de los indicadores. El siguiente gráfico muestra la media de punto otorgado a cada indicador. La tabla completa se encuentra en el anexo 6.

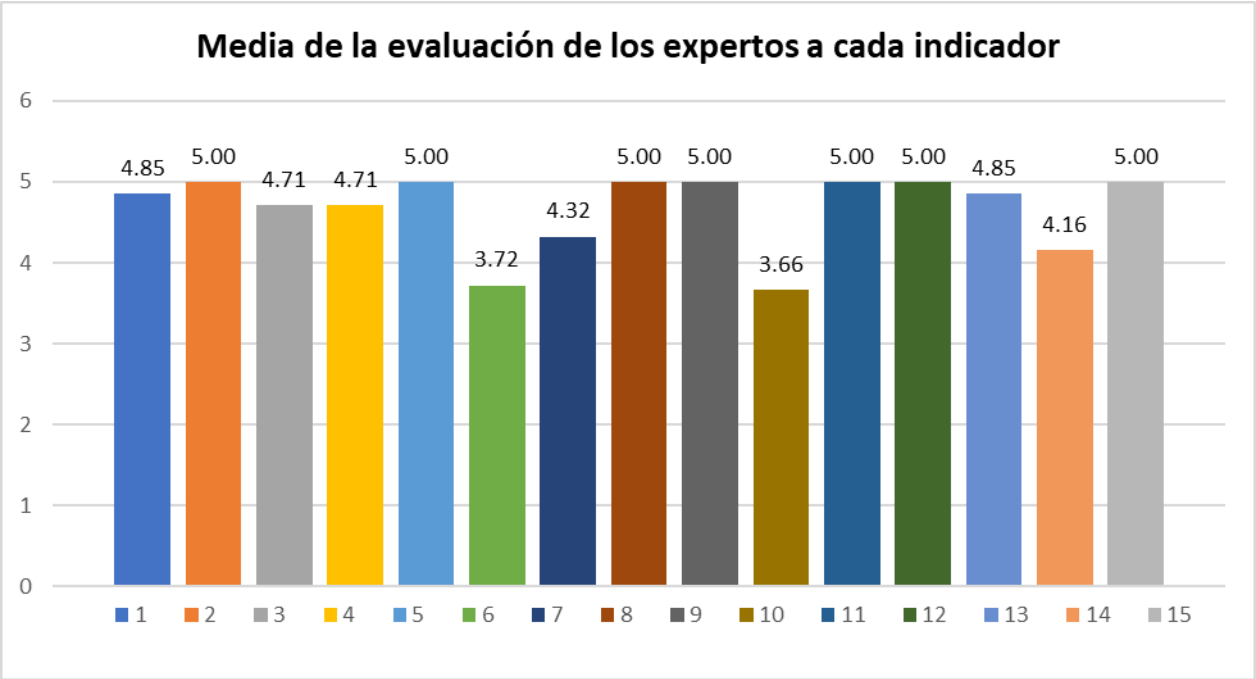

Figura 23 Media de la evaluación de los expertos a los indicadores para validar la propuesta de solución

Como se puede apreciar la mayoría de los aspectos fueron evaluados de más de 4 puntos destacándose que existe una relación directa entre los dispositivos de interacción y los niveles de inmersión. También todos valoran de muy positiva la investigación realizada del movimiento del miembro superior y la selección de tecnologías para la construcción del dispositivo.

Uno de los aspectos con peor puntuación está referido al uso de botones para los movimientos de cerrar la mano y agarrar objetos en los entornos 3D ya que no es la forma natural de hacerlo y lo mejor es utilizar sensores del tipo Flex que no se consiguieron durante el tiempo de la investigación. No obstante, todos coinciden en que es una solución acertada. El otro aspecto valorado por debajo de cuatro puntos es que la solución cableada puede afectar la libertad de movimiento al estar la persona conectada a la PC a través de un cable que se mueve junto con la mano. No obstante, todo coinciden en que se obtienen niveles de latencia más bajos que en soluciones inalámbricas.

Por último, hay una coincidencia absoluta en que la propuesta ofrece una solución completa al problema de investigación planteado y en que los objetivos de esta investigación se pueden considerar totalmente cumplidos.

Conclusiones parciales

La etapa de pruebas es fundamental durante el desarrollo de un sistema. La aplicación de los distintos casos de prueba propició la detección de las no conformidades, las cuales fueron corregidas en cada una de las iteraciones. Los procesos donde se detectaron las no conformidades están relacionados principalmente con la comunicación entre el sistema Unity y Arduino, específicamente con las funcionalidades encargadas de enviar un mensaje desde Unity a Arduino, procesar los datos provenientes de Arduino, asignar correctamente los valores de rotación a los gameobjects y procesar los datos provenientes del sensor MPU 6050. Las no conformidades se corrigieron en su totalidad, garantizando un 100% de resultados satisfactorios y cumpliendo con los requerimientos establecidos por el cliente.

La encuesta realizada a un grupo de expertos permitió validar la pertinencia y el cumplimiento de los objetivos de esta investigación, así como algunas recomendaciones para mejorar el trabajo en versiones futuras, algunas de ellas por su valor se colocan en las recomendaciones de este trabajo.

Conclusiones Generales

La investigación desarrollada permitió arribar a las siguientes conclusiones:

- El estudio a los dispositivos de realidad virtual más utilizados a nivel mundial, demostró que el mercado en relación a esta tecnología es joven y aún se encuentra en etapa de desarrollo.
- Los sistemas de realidad virtual tienen precios muy elevados.
- El estudio fisiológico-articular del miembro superior, demostró que es posible simular en una computadora los movimientos y acciones desarrolladas por el mismo, mediante la utilización de sensores especializados en la obtención de ángulos de rotación.
- En un proyecto de construcción de un dispositivo de hardware en el cual no se requiera una gran capacidad de procesamiento de datos, la utilización de Arduino constituye una opción más viable que el empleo de otras plataformas similares.
- La construcción del prototipo validó la posibilidad de construir este tipo de dispositivos mediante el uso de tecnologías libres.
- La realización del proceso de pruebas constituye una etapa fundamental del desarrollo de software, pues permita la detección temprana de no conformidades y permite garantizar la calidad requerida para el cumplimiento de los requisitos solicitados por el cliente.
- Para la validación de la hipótesis planteada se utilizó el método de expertos, el cuál arrojó resultados favorables acerca de la investigación y el cumplimiento de los objetivos trazados.

Recomendaciones

Durante el desarrollo de la investigación surgieron un conjunto de ideas que tienen como objetivo garantizar el futuro perfeccionamiento del sistema:

- Completar el control del miembro superior. Para ello se recomienda la adición de un sensor MPU 6050 u otro que sea capaz de capturar los ángulos de rotación con un mínimo de 6 GDL para controlar los movimientos de la mano y la reprogramación del microcontrolador para garantizar la obtención y procesamiento de los datos generados por el mismo.
- Construir otro dispositivo para garantizar el control de ambos brazos. Hay que tener en cuenta que al agregar otro dispositivo el total de sensores asciende a 6 (suponiendo que se cumplió con la recomendación anterior), por lo que Arduino no es capaz de manejar esa cantidad, siendo necesario la adición de un multiplexor para controlarlos.
- Agregar un visor al sistema. Se recomienda la utilización de un sensor BNO055 u otro que sea capaz de capturar ángulos de rotación con 9 GDL para controlar la rotación de la cabeza. Téngase en cuenta la posibilidad de utilizar un teléfono móvil a modo de lente, mediante la implementación de una aplicación que realice una proyección del escritorio de la computadora en la pantalla del mismo, para ello debe garantizar que la cámara a utilizar en el desarrollo del software debe tener la propiedad de trasmitir una imagen de visión estereoscópica.
- Convertir la comunicación a inalámbrica mediante la utilización de un módulo de WIFI, Bluetooth o radio frecuencia. Hay que tener en cuenta que será necesario adaptar el código Arduino para acoplar de forma correcta el módulo seleccionado, no siendo necesaria la modificación del programa desarrollado en Unity si se utiliza la comunicación vía Bluetooth puesto que esta se realiza de forma serial.

Bibliografía

[1] A. L. García García, *Realidad virtual*. Universidad Complutense de Madrid, Servicio de Publicaciones, 2004.

[2] M. Kim, C. Jeon, y J. Kim, «A study on immersion and presence of a portable hand haptic system for immersive virtual reality», *Sensors*, vol. 17, n.º 5, p. 1141, 2017.

[3] O. Bamodu y X. M. Ye, «Virtual reality and virtual reality system components», en *Advanced materials research*, 2013, vol. 765, pp. 1169–1172.

[4] M. Edwards, *Virtual reality system including smart objects*. Google Patents, 2013.

[5] V. Oculus, *LLC," Oculus Gear VR*. .

[6] M. Borges, A. Symington, B. Coltin, T. Smith, y R. Ventura, «Htc vive: Analysis and accuracy improvement», en *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 2610–2615.

[7] M. G. Solid, «Sony Playstation», *Tokyo, Japan: Konami Computer Entertainment, Inc*, 1998.

[8] C. Hillmann, «Comparing the Gear VR, Oculus Go, and Oculus Quest», en *Unreal for Mobile and Standalone VR*, Springer, 2019, pp. 141–167.

[9] P. Lamkin, «The best VR headsets: Oculust Rift, PlayStation VR, Gear VR, HTC Vive... virtual reality is back baby», *Sep*, vol. 10, p. 16, 2015.

[10] A. Borrego, J. Latorre, M. Alcaniz, y R. Llorens, «Comparison of Oculus Rift and HTC Vive: feasibility for virtual reality-based exploration, navigation, exergaming, and rehabilitation», *Games for health journal*, vol. 7, n.º 3, pp. 151–156, 2018.

[11] V. Bharath y R. Patil, «Solid modelling interaction with sensors for virtual reality welding», en *MATEC Web of Conferences*, 2018, vol. 144, p. 01008.

[12] M. Latarjet y L. Testut, *Compendio de anatomía descriptiva*. Masson, 1997.

[13] J. Sobotta, *Atlas de anatomía humana: Cabeza, cuello, miembro superior. Vol. 1*, vol. 1. Ed. Médica Panamericana, 2006.

[14] N. A. DFreitas, «Cinematica Articular», *Revista de la Sociedad Venezolana de Ciencias Morfológicas*, vol. 18, n.º 1, pp. 15–20, 2012.

[15] J. Martín, «Cinética articular del hombro. Revisión de una década de investigaciones», *Fisioterapia*, vol. 23, pp. 9–14, 2001.

[16] K. Mridul y R. Muthuganapathy, «Design and development of a portable virtual reality headset», en *Proceedings of the 2016 Virtual Reality International Conference*, 2016, pp. 1–4.

[17] S. Pedre, «Sistemas embebidos», *Laboratorio de Robótica y Sistemas Embebidos, Departamento de computación FCEN UBA*, 2017.

[18] B. G. Contreras Bermeo y V. H. Flores Morales, «Diseño, construcción e implementación de un sistema embebido de adquisición de parámetros cinemáticos de la marcha humana en tobillo, rodilla y cadera», B.S. thesis, 2015.

[19] S. Monk, *Raspberry Pi cookbook: Software and hardware problems and solutions*. O'Reilly Media, Inc., 2016.

[20] R. Pi, «Raspberry pi 3 model b», *online].(https://www. raspberrypi. org*, 2015.

[21] C. W. Zhao, J. Jegatheesan, y S. C. Loon, «Exploring iot application using raspberry pi», *International Journal of Computer Networks and Applications*, vol. 2, n.º 1, pp. 27–34, 2015.

[22] S. A. Arduino, «Arduino», *Arduino LLC*, 2015.

[23] A. Kurniawan, «Introduction to Arduino Boards and Development», en *Arduino Programming with. NET and Sketch*, Springer, 2017, pp. 1–19.

[24] C. Arduino, ««Arduino Software», *línea]. Available: https://www. arduino. cc/en/Main/Software.[Último acceso: 24 marzo 2019]*.

[25] L. Silva, R. Dantas, A. Pantoja, y A. Pereira, «Development of a low cost dataglove based on arduino for virtual reality applications», en *2013 IEEE International conference on computational intelligence and virtual environments for measurement systems and applications (CIVEMSA)*, 2013, pp. 55–59.
[26] A. Navarro, *Análisis comparativo de las placas Arduino (oficiales y compatibles)*. 2018.
[27] D. Lastra Lamarca y others, «Modelo analógico y digital en SystemC-AMS de la placa Arduino Mega 2560», 2015.
[28] L. G. Corona Ramírez, G. S. Abarca Jiménez, y J. Mares Carreño, *Sensores y actuadores*. Grupo Editorial Patria, 2014.
[29] D. Fedorov, A. Y. Ivoilov, V. Zhmud, y V. Trubin, «Using of measuring system MPU6050 for the determination of the angular velocities and linear accelerations», *Automatics & Software Enginery*, vol. 11, n.º 1, pp. 75–80, 2015.
[30] A. Yudhana, J. Rahmawan, y C. Negara, «Flex sensors and MPU6050 sensors responses on smart glove for sign language translation», en *IOP Conference Series: Materials Science and Engineering*, 2018, vol. 403, p. 012032.
[31] B. Sensortec, «Intelligent 9-axis absolute orientation sensor», *BNO055 datasheet, November*, 2014.
[32] U. G. Engine, «Unity game engine-official site», *Online][Cited: October 9, 2008.] http://unity3d. com*, pp. 1534–4320, 2008.
[33] A. Knörig, R. Wettach, y J. Cohen, «Fritzing: a tool for advancing electronic prototyping for designers», en *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, 2009, pp. 351–358.
[34] J. W. Murray, *C# game programming cookbook for Unity 3D*. CRC Press, 2014.
[35] T.-I. Kim, «Visual C# .NET Programing», *Devpia, Oct*, 2001.
[36] I. Sommerville, «Software engineering 9th Edition», *ISBN-10*, vol. 137035152, p. 18, 2011.
[37] R. S. Pressman y J. M. Troya, «Ingeniería del software», 1988.
[38] P. Letelier y M. C. Penadés, «Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP)», 2006.
[39] R. B. Tabares, «Patrones grasp y anti-patrones: un enfoque orientado a objetos desde logica de programacion», *Entre Ciencia e Ingeniería*, vol. 4, n.º 8, pp. 161–173, 2010.
[40] C. Larman, *UML y Patrones*. Pearson Educación ^ eMadrid Madrid, 2003.
[41] A. Cechich y R. Moore, «Una especificación precisa para patrones GoF», en *III Workshop de Investigadores en Ciencias de la Computación*, 2001.
[42] J. Gutiérrez, M. Escalona, M. Mejías, y J. Torres, «Pruebas del sistema en Programación Extrema», *Departamento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla*, 2009.
[43] J. A. Mera Paz y others, «Análisis del proceso de pruebas de calidad de software», 2016.

Anexos

Anexo 1. Historias de Usuario

Historia de Usuario	
Número: 1	**Nombre:** Calibrar el dispositivo
Usuario: Programador	**RF1:** Calibrar el dispositivo
Prioridad en Negocio: Baja	**Riesgo en Desarrollo:** Bajo
Puntos Estimados: 1.0	**Iteración Asignada:** 4
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Se calibran los sensores necesarios, para lograr el mínimo error en los datos de rotación con respecto al movimiento ejercido por el usuario en el uso del sistema.	
Observaciones:	

Historia de Usuario	
Número: 2	**Nombre:** Conectar Arduino con la PC
Usuario: Programador	**RF2:** Conectar Arduino con la PC
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Puntos Estimados:	**Iteración Asignada:**
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Se realiza la conexión entre Arduino y la computadora mediante la configuración de una comunicación a través del puerto serie.	
Observaciones:	

Historia de Usuario	
Número: 5	**Nombre:** Determinar el estado de los actuadores
Usuario: Programador	**RF3:** Enviar a la PC los datos de la posición del brazo y el estado del actuador
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Puntos Estimados:	**Iteración Asignada:**
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Arduino obtiene los datos provenientes de los botones.	

Observaciones: El botón es un actuador digital, por lo que solo envía estados ALTO y BAJO, se debe realizar una pequeña función para adaptar los datos y convertirlos en valores numéricos de tipo flotante.

Historia de Usuario	
Número: 7	**Nombre:** Solicitar a Arduino el estado de los sensores y actuadores
Usuario: Programador	**RF4:** Solicitar a Arduino el estado de los sensores y actuadores
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Puntos Estimados:	**Iteración Asignada:**
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Unity envía a Arduino una petición solicitando los datos provenientes de los sensores y actuadores.	
Observaciones:	

Historia de Usuario	
Número: 8	**Nombre:** Procesar los datos provenientes de arduino
Usuario: Programador	**RF5:** Gestionar los gameobjects para Unity que representan el brazo, el antebrazo y la mano
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Puntos Estimados:	**Iteración Asignada:**
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Unity procesa los datos provenientes de Arduino, los transforma en valores numéricos de tipo flotante y los almacena en las variables determinadas para su posterior uso.	
Observaciones:	

Anexo 2. Tarjetas CRC

Tarjeta CRC	
Clase: Calibrando_el_MPU_6050.ino	
Responsabilidad	**Colaboración**
• Calibrar ángulos y valores de rotación y aceleración del sensor MPU 6050	Wire MPU6050

Tarjeta CRC	
Clase: CargarObjeto.cs	
Responsabilidad	**Colaboración**
• Permite al usuario cargar un objeto determinado	Conexion.cs ObjetoCargable.cs Rigidbody Transform

Tarjeta CRC	
Clase: ObjetoCargable.cs	
Responsabilidad	**Colaboración**
• Determina si el objeto puede ser cargado por el usuario	CargarObjeto.cs Collider

Anexo 3. Tareas de desarrollo

Tarea	
Número de tarea: 1	**Número de historia de usuario:** 1
Nombre de la tarea: Calibración	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.5
Fecha de inicio: 2 de marzo 2020	**Fecha fin:** 11 de marzo 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite calibrar el sensor MPU 6050 para reducir el mínimo error posible en los valores de rotación	

Tarea	
Número de tarea: 2	**Número de historia de usuario:** 2
Nombre de la tarea: Conexión entre Arduino y la computadora	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 12 de marzo 2020	**Fecha fin:** 15 de marzo 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite realizar una comunicación serial entre Arduino y la computadora	

Tarea	
Número de tarea: 5	**Número de historia de usuario:** 5
Nombre de la tarea: Determinar estado de actuadores	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 3 de abril 2020	**Fecha fin:** 5 de abril 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite obtener los datos relacionados al estado de los botones	

Tarea	
Número de tarea: 7	**Número de historia de usuario:** 7
Nombre de la tarea: Solicitar datos a Arduino	

Tipo de tarea: Desarrollo	**Puntos estimados:** 1.5
Fecha de inicio: 16 de abril 2020	**Fecha fin:** 26 de abril 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite a Unity enviar un mensaje a Arduino solicitando los datos obtenidos a partir de los sensores y actuadores	

Tarea	
Número de tarea: 8	**Número de historia de usuario:** 8
Nombre de la tarea: Procesar datos de Arduino	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1.5
Fecha de inicio: 27 de abril 2020	**Fecha fin:** 6 de mayo 2020
Programador Responsable: Jorge Enrique Rodríguez Jiménez	
Descripción: Con el desarrollo de la presente tarea, se realiza la funcionalidad que permite a Unity separar la cadena de datos proveniente de Arduino y guardar cada valor en las variables correspondientes para su posterior uso	

Anexo 4. Casos de pruebas de aceptación

Caso de prueba de aceptación	
Código de la prueba: HU1-P1	**Nombre de la historia de usuario:** Calibrar los sensores
Descripción de la prueba: Se realizan varias iteraciones del código de calibración para verificar que se ha realizado con éxito	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • El sensor MPU 6050 debe estar conectado a la computadora	
Pasos de ejecución: • Se ejecuta el código de calibración • Se toman los valores de rotación • Se ejecuta el código calibración una vez más • Se vuelven a tomar los valores de rotación • Se comparan los valores de rotación	
Resultados esperados: No existen grandes diferencias entre los valores de rotación tomados en cada iteración	
Clasificación de la prueba: Satisfactoria	

Caso de prueba de aceptación	
Código de la prueba: HU2-P1	**Nombre de la historia de usuario:** Conectar Arduino con la PC
Descripción de la prueba: La prueba consiste en conectar al Arduino a la computadora mediante el cable de comunicación serial. El objetivo de la prueba es verificar que la computadora reconoce el puerto COM por el cual se conecta Arduino	
Condiciones de ejecución:	
Pasos de ejecución: • Conectar Arduino a la computadora mediante el cable de comunicación serial • Abrir el monitor serie	
Resultados esperados:	

El monitor serie se abre y muestra el puerto COM por el cual Arduino se comunica con la computadora
Clasificación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código de la prueba: HU4-P1	**Nombre de la historia de usuario:** Procesar los datos del sensor MPU 6050
Descripción de la prueba: La prueba consiste en verificar que se han procesado correctamente los datos provenientes del sensor MPU 6050	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • El MPU 6050 debe estar conectado a Arduino	
Pasos de ejecución: • Abrir el monitor serie	
Resultados esperados: El monitor serie se abre y muestra los valores de rotación en formato de Euler (360º)	
Clasificación de la prueba: No satisfactoria. Los datos no se muestran en formato de Euler	

Caso de prueba de aceptación	
Código de la prueba: HU4-P3	**Nombre de la historia de usuario:** Procesar los datos del sensor MPU 6050
Descripción de la prueba: La prueba consiste en verificar que se han procesado correctamente los datos provenientes del sensor MPU 6050	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • El MPU 6050 debe estar conectado a Arduino	
Pasos de ejecución: • Abrir el monitor serie	
Resultados esperados:	

El monitor serie se abre y muestra los valores de rotación en formato de Euler (360º)
Clasificación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código de la prueba: HU5-P1	**Nombre de la historia de usuario:** Determinar el estado de los actuadores
Descripción de la prueba: La prueba consiste en verificar que Arduino obtiene de forma correcta los datos correspondientes al estado de los actuadores	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • Al menos un botón debe estar conectado a Arduino	
Pasos de ejecución: • Pulsar el botón	
Resultados esperados: El led interno del Arduino, correspondiente al pin digital número 13, se encienda cada vez que se presiona el botón	
Clasificación de la prueba: Satisfactoria	

Caso de prueba de aceptación	
Código de la prueba: HU7-P1	**Nombre de la historia de usuario:** Solicitar a Arduino el estado de los sensores y actuadores
Descripción de la prueba: La prueba consiste en verificar que Unity es capaz de enviar un mensaje a Arduino solicitando los datos de los sensores y actuadores	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora	
Pasos de ejecución: • Ejecutar el programa en Unity • Abrir el monitor serie	

Resultados esperados: En el monitor serie aparece en mensaje “datos” proveniente de Unity
Clasificación de la prueba: No Satisfactoria. Al abrir el monitor serie no se muestra ningún mensaje

Caso de prueba de aceptación	
Código de la prueba: HU7-P2	**Nombre de la historia de usuario:** Solicitar a Arduino el estado de los sensores y actuadores
Descripción de la prueba: La prueba consiste en verificar que Unity es capaz de enviar un mensaje a Arduino solicitando los datos de los sensores y actuadores	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora	
Pasos de ejecución: • Ejecutar el programa en Unity • Abrir el monitor serie	
Resultados esperados: En el monitor serie aparece en mensaje “datos” proveniente de Unity	
Clasificación de la prueba: Satisfactoria	

Caso de prueba de aceptación	
Código de la prueba: HU7-P3	**Nombre de la historia de usuario:** Solicitar a Arduino el estado de los sensores y actuadores
Descripción de la prueba: La prueba consiste en verificar que Unity es capaz de enviar un mensaje a Arduino solicitando los datos de los sensores y actuadores	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora	
Pasos de ejecución: • Ejecutar el programa en Unity • Abrir el monitor serie	

Resultados esperados: En el monitor serie aparece en mensaje “datos” proveniente de Unity
Clasificación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código de la prueba: HU8-P1	**Nombre de la historia de usuario:** Procesar los datos provenientes de Arduino
Descripción de la prueba: La prueba consiste en verificar que Unity es capaz de procesar los datos provenientes de Arduino	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • Los sensores BNO055 y MPU 6050 deben estar conectados a Arduino • Al menos debe haber un botón conectado a Arduino	
Pasos de ejecución: • Ejecutar el programa en Unity • Abrir la terminal de Unity	
Resultados esperados: En la terminal se muestran los datos correspondientes a cada variable	
Clasificación de la prueba: No Satisfactoria. Al abrir la terminal se muestran los datos recibidos en forma de cadena sin procesar	

Caso de prueba de aceptación	
Código de la prueba: HU8-P2	**Nombre de la historia de usuario:** Procesar los datos provenientes de Arduino
Descripción de la prueba: La prueba consiste en verificar que Unity es capaz de procesar los datos provenientes de Arduino	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • Los sensores BNO055 y MPU 6050 deben estar conectados a Arduino • Al menos debe haber un botón conectado a Arduino	

Pasos de ejecución: • Ejecutar el programa en Unity • Abrir la terminal de Unity
Resultados esperados: En la terminal se muestran los datos correspondientes a cada variable
Clasificación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código de la prueba: HU8-P3	**Nombre de la historia de usuario:** Procesar los datos provenientes de Arduino
Descripción de la prueba: La prueba consiste en verificar que Unity es capaz de procesar los datos provenientes de Arduino	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • Los sensores BNO055 y MPU 6050 deben estar conectados a Arduino • Al menos debe haber un botón conectado a Arduino	
Pasos de ejecución: • Ejecutar el programa en Unity • Abrir la terminal de Unity	
Resultados esperados: En la terminal se muestran los datos correspondientes a cada variable	
Clasificación de la prueba: Satisfactoria	

Caso de prueba de aceptación	
Código de la prueba: HU9-P1	**Nombre de la historia de usuario:** Actualizar los gameobjects
Descripción de la prueba: La prueba consiste en comprobar que los gameobjects reaccionan correctamente a los cambios de estado de los sensores y actuadores	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora	

• Los sensores BNO055 y MPU6050 deben estar conectados a Arduino • Al menos debe haber un botón conectado a Arduino
Pasos de ejecución: • Se ejecuta el programa en Unity
Resultados esperados: Los objetos asociados a los gameobjects Brazo y Antebrazo rotan en correspondencia a la orientación de los sensores. Al pulsar el botón el jugador carga un objeto
Clasificación de la prueba: No Satisfactoria. Los ejes de rotación de los gameobjects están invertidos con respecto a los ejes de los sensores

Caso de prueba de aceptación	
Código de la prueba: HU9-P3	**Nombre de la historia de usuario:** Actualizar los gameobjects
Descripción de la prueba: La prueba consiste en comprobar que los gameobjects reaccionan correctamente a los cambios de estado de los sensores y actuadores	
Condiciones de ejecución: • Arduino debe estar conectado a la computadora • Los sensores BNO055 y MPU6050 deben estar conectados a Arduino • Al menos debe haber un botón conectado a Arduino	
Pasos de ejecución: • Se ejecuta el programa en Unity	
Resultados esperados: Los objetos asociados a los gameobjects Brazo y Antebrazo rotan en correspondencia a la orientación de los sensores. Al pulsar el botón el jugador carga un objeto	
Clasificación de la prueba: No Satisfactoria. Los ejes de rotación de los gameobjects están invertidos con respecto a los ejes de los sensores	

Anexo 5. Modelo encuesta a expertos

Estimado especialista:

Se solicita su valiosa cooperación para la valoración de los resultados del trabajo "*Dispositivo de Hardware Libre para Interacción en Entornos 3D*", el cual constituye el resultado de una investigación dirigida a enriquecer las experiencias de usuario y elevar los niveles de inmersión e interacción en los entornos virtuales 3D que realiza el centro VERTEX de la Universidad de las Ciencias Informáticas. Por lo que le pedimos nos ayude a responder el siguiente cuestionario en vistas a evaluar los resultados desde el punto de vista práctico.

Por favor lea detenidamente el informe adjunto y valore los siguientes aspectos en una escala del uno al cinco (donde 1 -Ninguna, 2 - Baja, 3 - Media, 4 - Alta, 5 - Muy alta).

No	Aspecto a valorar	Evaluación
Importancia y pertinencia		
1	Actualidad del tema	
2	Medida en que los dispositivos de interacción influyen en la inmersión e interacción en entorno 3D	
3	Validez de la problemática para nuestra realidad social	
Validez de la solución		
4	La propuesta tributa a la solución de la problemática	
5	Nivel de acierto en la selección de la plataforma para el desarrollo del dispositivo	
6	¿En qué medida la sustitución de lo sensores Flex por botones deteriora los niveles de interacción?	
7	Suficiencia de la información brindada por los dos giroscopios para un movimiento fluido y real del miembro superior	
8	Profundidad del estudio del estudio fisiológico articular del miembro superior para la investigación	
9	¿Considera la comunicación serial a través de USB adecuada para obtener una buena actualización en tiempo real de los modelos?	
10	Medida en que la solución cableada afecta la libertad de movimientos	
Resultados alcanzados		
11	La propuesta ofrece una solución completa al problema de investigación	
12	Considera el código del firmware adecuado para la solución	
13	Considera los tiempos de respuesta adecuados para aplicaciones de este tipo	
14	¿Considera la propuesta económicamente viable?	
15	¿En qué medida considera los objetivos de esta investigación como cumplidos?	

Anexo 6. Tabla con los datos resultantes de la encuesta a los expertos

	Indicadores / Expertos	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Media	Moda
1	Actualidad del tema	5	5	5	5	4	5	5	5	4	5	5	5	5	5	5	5	4.85	5
2	Medida en que los dispositivos de interacción influyen en la inmersión e interacción en entorno 3D	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5.00	5
3	Validez de la problemática para nuestra realidad social	5	5	4	4	4	5	5	4	5	5	5	5	5	5	5	5	4.71	5
4	La propuesta tributa a la solución de la problemática	4	4	5	5	5	5	5	5	5	5	5	4	5	5	4	5	4.71	5
5	Nivel de acierto en la selección de la plataforma para el desarrollo del dispositivo	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5.00	5
6	¿En qué medida la sustitución de lo sensores Flex por botones deteriora los niveles de interacción?	4	4	4	3	4	5	5	5	3	3	3	3	4	4	3	5	3.72	4
7	Suficiencia de la información brindada por los dos giroscopios para un movimiento fluido y real del miembro superior	4	4	4	4	5	5	4	4	4	5	5	5	5	4	4	4	4.32	4
8	Profundidad del estudio del estudio fisiológico articular del miembro superior para la investigación	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5.00	5
9	¿Considera la comunicación serial a través de USB adecuada para obtener una buena actualización en tiempo real de los modelos?	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5.00	5
10	Medida en que la solución cableada afecta la libertad de movimientos	4	5	3	4	3	4	3	3	3	4	4	4	4	4	4	4	3.66	4
11	La propuesta ofrece una solución completa al problema de investigación	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5.00	5
12	Considera el código del firmware adecuado para la solución	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5.00	5
13	Considera los tiempos de respuesta adecuados para aplicaciones de este tipo	5	5	5	5	4	5	5	5	5	5	5	5	4	5	5	5	4.85	5
14	¿Considera la propuesta económicamente viable?	4	4	4	4	5	4	4	4	4	5	5	4	4	4	4	4	4.16	4
15	¿En qué medida considera los objetivos de esta investigación como cumplidos?	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5.00	5