

Trabajo de Fin de Grado  
Grado en Ingeniería Informática  
Computación

---

**Simulador educativo de soldadura de baterías de litio  
en realidad virtual**

---

*Iraida Abad López*

**Dirección**  
Iñigo López Gazpio  
Igor Rodríguez Rodríguez

8 de octubre de 2025



---

## Agradecimientos

---

Quiero expresar mis agradecimientos a todas las personas que me han apoyado durante la realización de este trabajo fin de grado. En primer lugar, quiero agradecer a mis tutores de TFG Iñigo López e Igor Rodríguez por darme la oportunidad de realizar este proyecto y confiar en mi para llevarlo a cabo.

También quiero agradecer a mi familia y amigos, especialmente a mi madre y a mi pareja, por su apoyo incondicional y toda su ayuda durante el desarrollo de este proyecto.



---

# Resumen

---

La realidad virtual es una tecnología que ha ganado popularidad en los últimos años. Aunque su uso se ha popularizado principalmente en el ámbito del entretenimiento, también ha ganado relevancia en la educación y formación profesional.

La incorporación de dispositivos de realidad virtual en los entornos de aprendizaje ofrece una vía pedagógica innovadora con un gran potencial. Aunque esta tecnología todavía presenta limitaciones, diversos estudios respaldan los beneficios de su implementación. Resulta especialmente útil para la formación en prácticas que conllevan riesgos, ya que en un entorno virtual dichos peligros se eliminan por completo. Por ello, y con el objetivo de explorar la realidad virtual como recurso educativo, este trabajo fin de grado se ha centrado en el desarrollo de una aplicación de realidad virtual orientada al ensamblaje y soldadura de un paquete de baterías de litio.

Para el desarrollo de la aplicación ha sido necesario modelar en 3D los materiales necesarios utilizando Blender, mientras que la implementación del resto de funcionalidades se ha llevado a cabo en la plataforma de desarrollo Unity. El resultado es una aplicación en versión experimental que simula el proceso de montaje de un paquete de baterías en un entorno seguro y controlado. La práctica se centra específicamente en el ensamblaje de baterías con configuración 10s4p, lo que implica organizar 40 celdas de litio individuales en 10 columnas conectadas en serie y 4 filas en paralelo. La aplicación recrea un aula virtual inmersiva en la que el alumnado puede interactuar con los materiales y herramientas necesarias para completar la práctica. Aunque se trata de una versión inicial, constituye una base sólida para trabajos futuros.

## Siglas

- **RV:** Realidad virtual
- **UI:** Interfaz de usuario (User interface en inglés)



---

# **Objetivos de Desarrollo Sostenible**

---

Dentro de la Agenda 2030 la educación de calidad se establece como el cuarto Objetivo de Desarrollo Sostenible (ODS). Este objetivo busca garantizar el derecho a una educación universal, inclusiva y equitativa. Entre sus metas se encuentra asegurar el acceso igualitario a todos los niveles de educación, incluida la formación profesional, para las personas pertenecientes a grupos vulnerables como aquellas con discapacidad.

En este contexto, la implementación de dispositivos de realidad virtual (RV) en la educación supone un avance hacia la consecución de dicho objetivo. La recreación de actividades prácticas en entornos de realidad virtual no requiere de ningún esfuerzo físico, lo que permite adaptar estas experiencias a las necesidades de personas con discapacidades motoras, permitiendo que estas realicen la práctica de manera autónoma. Además, estos entornos también pueden configurarse para atender las necesidades de personas con dificultades auditivas mediante la incorporación de subtítulos u otros indicadores visuales. Asimismo, la formación práctica en entornos inmersivos elimina cualquier riesgo al que el alumnado estaría expuesto al realizar la práctica real.

Por otra parte, esta tecnología ofrece ventajas para el alumnado que cursa estudios de forma remota, facilitando experiencias inmersivas que, de otra forma, solo serían posibles en entornos presenciales.

De este modo, la realidad virtual no solo representa un avance en la digitalización del sistema educativo, sino que también abre nuevas oportunidades para la formación en condiciones seguras y controladas y para una educación más inclusiva e interactiva.



---

# Índice de contenidos

---

|  |            |
|--|------------|
| <b>Índice de contenidos</b>                                  | <b>vii</b> |
| <b>Índice de figuras</b>                                     | <b>ix</b>  |
| <b>Índice de tablas</b>                                      | <b>x</b>   |
| <b>Índice de algoritmos</b>                                  | <b>xi</b>  |
| <b>1 Introducción</b>  | <b>1</b>   |
| 1.1. Objetivos . . . . .                                     | 2          |
| 1.2. Tecnologías . . . . .                                   | 3          |
| 1.2.1. Blender . . . . .                                     | 3          |
| 1.2.2. Unity . . . . .                                       | 3          |
| 1.2.3. Unity Hub . . . . .                                   | 3          |
| 1.2.4. Unity Version Control . . . . .                       | 4          |
| 1.2.5. XR Interaction Toolkit . . . . .                      | 4          |
| 1.2.6. Gafas de realidad virtual Oculus Meta Quest . . . . . | 4          |
| <b>2 Planificación</b>                                       | <b>7</b>   |
| 2.1. Paquetes de trabajo . . . . .                           | 7          |
| 2.2. Cronograma y estimaciones de tiempo . . . . .           | 8          |
| 2.3. Análisis de riesgos . . . . .                           | 9          |
| 2.4. Seguimiento y control . . . . .                         | 9          |
| <b>3 Realidad virtual en la educación</b>                    | <b>13</b>  |
| 3.1. Fabricación de baterías en realidad virtual . . . . .   | 15         |
| <b>4 Desarrollo de la aplicación</b>                         | <b>17</b>  |
| 4.1. Proceso de montaje de una batería . . . . .             | 17         |
| 4.1.1. Materiales y herramientas . . . . .                   | 17         |
| 4.1.2. Pasos para el montaje de una batería 10s4p . . . . .  | 18         |
| 4.2. Modelos digitales . . . . .                             | 19         |
| 4.2.1. Soporte 10s4p . . . . .                               | 19         |
| 4.2.2. Celda de litio . . . . .                              | 21         |
| 4.2.3. Cinta de níquel . . . . .                             | 21         |
| 4.2.4. Soldador . . . . .                                    | 22         |
| 4.2.5. Multímetro . . . . .                                  | 22         |

|   |           |
|---|-----------|
| 4.2.6. Cajas . . . . .                                  | 23        |
| 4.2.7. Otros objetos de la escena . . . . .             | 23        |
| 4.3. Implementación del montaje . . . . .               | 24        |
| 4.3.1. Fijar las pilas en el soporte inferior . . . . . | 25        |
| 4.3.2. Fijar el soporte superior . . . . .              | 28        |
| 4.3.3. Colocar las cintas de níquel y soldar . . . . .  | 28        |
| 4.3.4. Validar práctica . . . . .                       | 30        |
| 4.3.5. Voltímetro . . . . .                             | 30        |
| 4.4. Otras interacciones . . . . .                      | 31        |
| <b>5 Conclusiones</b>                                   | <b>33</b> |
| 5.1. Trabajo futuro . . . . .                           | 33        |
| <b>Apéndice</b>   | <b>41</b> |
| Interacciones entre objetos y layers mask . . . . .     | 41        |
| Link carpeta Google Drive . . . . .                     | 41        |
| Socket dinámico . . . . .                               | 41        |
| Mensaje retroalimentación . . . . .                     | 41        |
| <b>Bibliografía</b>                                     | <b>45</b> |

---

# Índice de figuras

---

|       |  |    |
|-------|--|----|
| 1.1.  | Dispositivos de realidad virtual . . . . .         | 5  |
| 2.1.  | Diagrama paquetes de trabajo . . . . .             | 8  |
| 2.2.  | Diagrama de Gantt . . . . .                        | 8  |
| 2.3.  | Tiempo dedicado a cada tarea . . . . .             | 10 |
| 2.4.  | Tiempo estimado y tiempo real . . . . .            | 11 |
| 4.1.  | Soporte individual real y modelo digital . . . . . | 20 |
| 4.2.  | Soportes 10s4p . . . . .                           | 21 |
| 4.3.  | Pila real y modelo digital . . . . .               | 21 |
| 4.4.  | Cinta de níquel real y modelo digital . . . . .    | 22 |
| 4.5.  | Soldador de puntos real y modelo digital . . . . . | 22 |
| 4.6.  | Multímetro real y modelo digital . . . . .         | 23 |
| 4.7.  | Modelo digital cajas . . . . .                     | 23 |
| 4.8.  | Mesa principal . . . . .                           | 24 |
| 4.9.  | Escena completa . . . . .                          | 24 |
| 4.10. | Ejes pila . . . . .                                | 25 |
| 4.11. | Sockets del soporte inferior . . . . .             | 26 |
| 4.12. | Colocar pilas en el soporte inferior . . . . .     | 28 |
| 4.13. | Colocar soporte superior . . . . .                 | 29 |
| 4.14. | Puntos a soldar . . . . .                          | 29 |
| 4.15. | Sockets para las cintas de níquel . . . . .        | 30 |
| 4.16. | Soldar cintas . . . . .                            | 30 |
| 4.17. | Trigger collider de los polos . . . . .            | 31 |
| 4.18. | Medir voltaje de una pila . . . . .                | 31 |
| 5.1.  | Ejemplo funcionamiento voltímetro caso 1 . . . . . | 36 |
| 5.2.  | Ejemplo cortocircuito I . . . . .                  | 36 |
| 5.3.  | Ejemplo cortocircuito II . . . . .                 | 36 |
| 5.4.  | Soldadura en paralelo . . . . .                    | 37 |
| 5.5.  | Tensión nula . . . . .                             | 37 |
| 1.    | Retroalimentación de la práctica . . . . .         | 43 |

---

# Índice de tablas

---

|      |   |    |
|------|---|----|
| 2.1. | Distribución estimada de horas de trabajo por paquete . . . . . | 9  |
| 2.2. | Duración estimada y real . . . . .                              | 10 |
| 1.   | Interacciones entre objetos y layer mask . . . . .              | 41 |

---

# **Lista de Algoritmos**

---

|      |  |    |
|------|--|----|
| 4.1. | Método inicial BatteriesInMount.cs . . . . .   | 27 |
| 4.2. | FillUserAns con XR Socket Interactor . . . . . | 27 |
| 4.3. | FillUserAns con Dynamic Socket . . . . .       | 28 |
| 5.1. | Encender voltímetro . . . . .                  | 38 |
| 5.2. | Medir voltaje . . . . .                        | 39 |



---

## Capítulo 1

# Introducción

---

El término realidad virtual (RV), fue empleado oficialmente por primera vez en 1960, aunque su uso comenzó hacia el siglo XIX, con la creación del primer mural 360° [1]. Desde entonces, el significado del concepto “realidad virtual” ha evolucionando notablemente.

Hoy en día, la Real Academia Española (RAE) define la realidad virtual como “la representación de escenas o imágenes de objetos producida por un sistema informático, que da la sensación de su existencia real”. Sin embargo, esta definición no representa con precisión el uso actual del término.

Una definición más acorde con la que hoy nos referimos por ese término es la de Roehl en 1996: “La Realidad Virtual es una simulación de un ambiente tridimensional generada por computadoras, en el que el usuario es capaz tanto de ver como de manipular los contenidos de ese ambiente” [2]. Otra definición que la literatura admite es la definición de  $I^3$ , que reúne los términos interacción, inmersión e imaginación [3].

Actualmente, la realidad virtual es una tecnología en auge que está ganando cada vez más popularidad. Su uso más extendido y más conocido por el público general es en el mundo del entretenimiento; más concretamente los videojuegos. En ellos, los y las jugadoras pueden sumergirse totalmente en escenas virtuales e interactuar en ellas a través de distintos sentidos, como la vista o el tacto. Sin embargo, su uso no se limita solo al ocio; en los últimos años la RV ha comenzado a incorporarse en el ámbito educativo. Cada vez más empresas e instituciones implementan la realidad virtual con fines educativos y formativos, aprovechando su potencial para ofrecer experiencias de aprendizaje activo.

La educación siempre ha sido un sector en constante evolución, en el que las herramientas digitales han transformado la dinámica del aula. Si bien los recursos digitales, como tablets y ordenadores, ya forman parte de la mayoría de centros educativos, ahora se abre un nuevo debate: la implementación de tecnologías de realidad virtual como herramientas complementarias. Varios estudios afirman los beneficios de la inclusión de estos dispositivos, señalando que su uso no solo mejora la retención de la información y el desarrollo de habilidades prácticas, sino que también enriquece la experiencia de aprendizaje al aportar escenarios inmersivos.

Además de estos beneficios, esta tecnología permite simular situaciones complejas y peligrosas, haciendo posible la formación en campos técnicos de una manera segura y controlada, eliminando completamente los costes y riesgos asociados a los equipos y materiales reales. En este sentido, la realidad virtual se presenta como una herramienta con un gran potencial para transformar la educación.

En los últimos años, varias disciplinas han optado por instaurar dispositivos de realidad virtual o aumentada, siendo la ingeniería una de ellas [4, 5]. Entre las distintas disciplinas ingenieriles, la eléctrica es todavía un terreno poco explorado.

El auge de la movilidad eléctrica y los sistemas de almacenamiento de energía ha posicionado a las baterías de litio como un componente esencial en la tecnología moderna. No obstante, su manipulación y montaje, especialmente en la configuración de paquetes de baterías, conlleva riesgos inherentes como cortocircuitos, sobrecalentamientos y, en casos extremos, incendios o explosiones.

Tradicionalmente, la formación en el proceso de ensamblaje de paquetes de baterías requiere de materiales físicos, lo que no solo supone un alto consumo de estos componentes, sino que también expone al alumnado a estos posibles peligros.

Para abordar este desafío, este trabajo fin de grado propone el desarrollo de una aplicación educativa de RV en la que se pueda simular el proceso de montaje de un paquete de baterías de litio. La aplicación se centra en el ensamblaje de una batería con configuración 10s4p. Esta para montar esta configuración es necesario emplear 40 celdas de ion de litio individuales, formando 10 columnas de pilas conectadas en serie (10s) y 4 filas en paralelo (4p). Existen varios tipos de celdas de litio, en este caso se utilizan las 18650. El nombre viene de sus dimensiones, 18mm de diámetro y 65mm de longitud, el 0 final indica su forma cilíndrica. Este tipo de baterías recargables tienen un voltaje nominal de 3,7 voltios, a diferencia de las baterías AA que tienen un voltaje de 1,5 voltios. Las ventajas de utilizar este tipo de baterías son: no tienen efecto memoria, es decir, no se reduce su capacidad, tienen un bajo nivel de autodescarga, son más ligeras y tienen una larga vida útil.

Esta aplicación permitirá al estudiantado interactuar con los materiales y herramientas necesarias de manera virtual, ofreciendo la oportunidad de ensamblar y soldar una batería de forma segura y repetible.

Al trasladar la práctica a un entorno digital, se eliminan todos los peligros físicos y se optimiza el uso de recursos materiales, ya que el coste de los ensayos se reduce a cero. De esta forma, la aplicación no solo facilita un aprendizaje más seguro y eficiente, sino que también sirve como complemento para la labor del docente.

### 1.1. Objetivos

El objetivo principal de este proyecto es desarrollar una práctica experimental de realidad virtual en la que el usuario pueda practicar la soldadura de baterías de litio sin exhibirse a los riesgos que conlleva.

La soldadura de baterías de litio (en este caso la batería 18650) implica numerosos riesgos: desde cortocircuitos y sobrecalentamientos hasta explosiones e incendios. Por este motivo, es esencial ofrecer un entorno controlado y seguro en el que los y las estudiantes puedan practicar y perfeccionar sus habilidades sin exponerse a estos peligros. Esta aplicación ofrece la posibilidad de interactuar en un laboratorio virtual completamente libre de peligros reales.

Para lograr este objetivo general, se han definido los siguientes subobjetivos de desarrollo, que guiarán el proceso de creación de la aplicación:

1. Diseño y modelado 3D del entorno y componentes: Se deben modelar todos los materiales y herramientas necesarias para realizar el montaje de una batería, incluyendo las celdas de litio, las cintas de níquel, el soporte para las celdas y la soldadora por puntos. El escenario virtual debe recrear un entorno de trabajo realista y sin distracciones adicionales. Es importante mencionar que, al tratarse de la versión en desarrollo de la aplicación, se ha decidido simplificar el diseño de los modelos de las herramientas y los componentes.

2. Implementación de interacciones: La persona usuaria debe poder interactuar con los objetos del entorno. Esto incluye el uso de las herramientas necesarias, la manipulación de los materiales para el montaje de la batería y el desplazamiento dentro del entorno virtual. Resulta crucial que los objetos virtuales reproduzcan con la mayor fidelidad posible los comportamientos de sus equivalentes reales, de modo que la experiencia sea realista.
3. Integración de un sistema de retroalimentación: Con el objetivo de facilitar a la persona usuaria identificar y corregir sus errores y favorecer una formación autodirigida, la versión final de la aplicación debe incorporar un sistema que proporcione un informe detallado de la práctica una vez haya sido completada, indicando en qué pasos se ha cometido algún error.
4. Desarrollo de una interfaz de usuario (UI): Se implementará una interfaz de usuario que permita a la persona usuaria gestionar la práctica, ofreciendo opciones como reiniciarla o corregirla en cualquier momento. En la versión experimental de la aplicación, esta interfaz también incluye controles adicionales, como el uso del voltímetro o el boqueo de las pilas. Sin embargo, en la versión final de la aplicación, estas funciones no se controlarán desde la UI.
5. Configurar hardware y software: Se deberá asegurar la compatibilidad de la aplicación con los dispositivos de realidad virtual disponibles, como las gafas de realidad virtual, y los motores de juego.
6. Incorporación del multímetro: Con el fin de crear una experiencia formativa más completa, la versión final de la aplicación deberá incluir un multímetro funcional que permita al alumnado realizar las mediciones necesarias durante el proceso de montaje y comprobar el estado de la batería.

## 1.2. Tecnologías

En este apartado se presentan las distintas herramientas utilizadas tanto para el diseño de los modelos digitales como para la creación de la escena de realidad virtual.

### 1.2.1. Blender

Blender es un software de código abierto que se utiliza principalmente para la creación de contenido 3D y 2D. Este software reúne varias herramientas para la creación de gráficos, entre ellas, las más conocidas son las funciones de modelado, animación, simulación y edición de video. Esta herramienta se ha empleado para la creación de todos los modelos que aparecen en la escena.

### 1.2.2. Unity

Unity es un motor de desarrollo en tiempo real utilizado para videojuegos y aplicaciones interactivas en 2D, 3D, realidad aumentada y realidad virtual. Unity permite crear aplicaciones para que pueden ser exportadas a múltiples plataformas, incluyendo PC, consolas, dispositivos móviles y dispositivos de realidad extendida.

### 1.2.3. Unity Hub

Unity Hub es una aplicación que permite sincronizar los proyectos de Unity con la nube. A través de ella es posible crear, abrir, eliminar y gestionar proyectos, así como importar proyectos de otras ubicaciones. Por otra parte, Unity Hub también permite instalar y utilizar múltiples versiones de Unity de forma simultánea.

#### 1.2.4. Unity Version Control

Unity Version Control, anteriormente conocida como Plastic Scm, es una herramienta que permite controlar las versiones de los proyectos desarrollados con Unity. Mediante esta aplicación es posible crear copias de seguridad de los proyectos, facilitando la revisión y restauración de versiones anteriores. Además, ofrece la opción crear distintas ramas a partir del proyecto principal, lo que favorece el trabajo en paralelo y reduce el riesgo de pérdida de datos o modificaciones no deseadas en el trabajo ya desarrollado.

#### 1.2.5. XR Interaction Toolkit

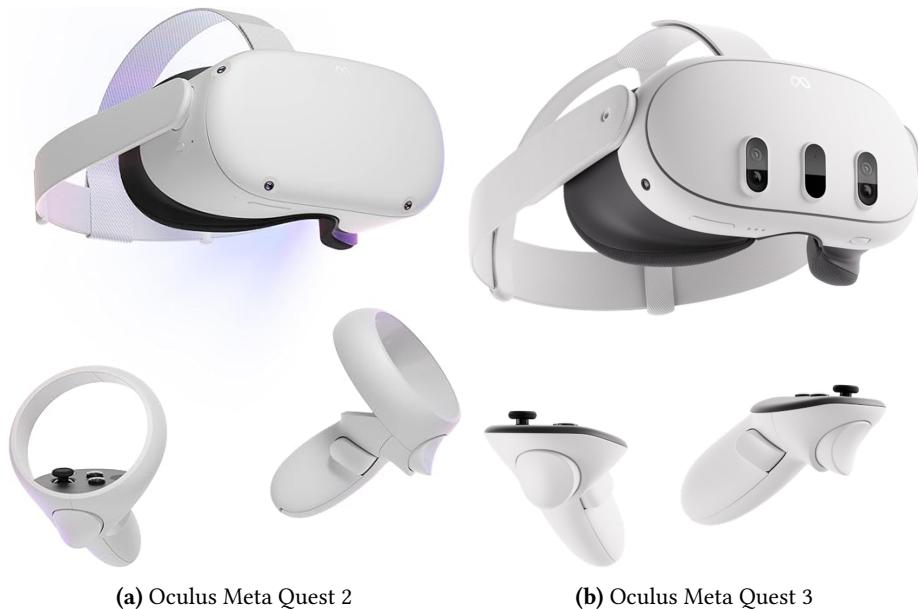
XR Interaction Toolkit es el paquete que se ha utilizado principalmente para gestionar las interacciones dentro de la escena.

Este paquete incluye distintos componentes para la creación de experiencias de realidad virtual y aumentada. El paquete se centra en un sistema de “*interactors*” (componentes que realizan la acción) e “*interactables*” (componentes sobre los que se aplica la acción). Además también incluye componentes para la locomoción, elementos visuales e interacción con UI.

#### 1.2.6. Gafas de realidad virtual Oculus Meta Quest

Para el desarrollo correcto de esta aplicación ha sido imprescindible testear cada función implementada, para ello se han utilizado los modelos “Meta Quest 3” y “Meta Quest 2”. Estos aparatos permiten el uso de aplicaciones de realidad virtual. Ambos dispositivos presentan diferencias a nivel de hardware, sin embargo, dichas diferencias no han tenido impacto a la hora de ejecutar la aplicación. Por ello, en este apartado se describen los elementos comunes de ambos dispositivos:

- Gafas inalámbricas. Estas gafas incorporan distintos sensores en la parte frontal. Una de las mayores ventajas de estos dispositivos es la posibilidad utilizarlos de forma inalámbrica; no obstante, disponen también de un puerto USB-C que permite conectarlas a un ordenador, y que, a su vez, funciona como puerto de carga. Por otra parte, estos dispositivos también cuentan con un sistema de audio integrado.
- Controladores inalámbricos. Cada controlador dispone de dos botones y un joystick en la parte superior, además de un botón tipo gatillo en la parte frontal y otro botón adicional en la parte interior.



(a) Oculus Meta Quest 2

(b) Oculus Meta Quest 3

**Figura 1.1:** Dispositivos de realidad virtual



---

## Capítulo 2

# Planificación

---

En este capítulo se explica la planificación del proyecto, describiendo los paquetes de trabajo, la duración estimada y la real de cada uno de estos y el análisis de riesgos. Además de una breve descripción de los métodos utilizados para el seguimiento y control del trabajo.

### 2.1. Paquetes de trabajo

Con el fin de facilitar el desarrollo del proyecto, este se ha dividido en varios paquetes de trabajo (ver imagen 2.1). A excepción de la planificación inicial, todos los paquetes se han desarrollado en paralelo, aunque la dedicación a cada paquete ha variado en función de la etapa del proyecto. A continuación se detallan los paquetes de trabajo:

- **Planificación**

Este paquete incluye tanto la planificación inicial del proyecto, es decir, identificar los paquetes de trabajo, estimar la duración de cada tarea y crear el diagrama de Gantt, como el seguimiento del desarrollo del proyecto.

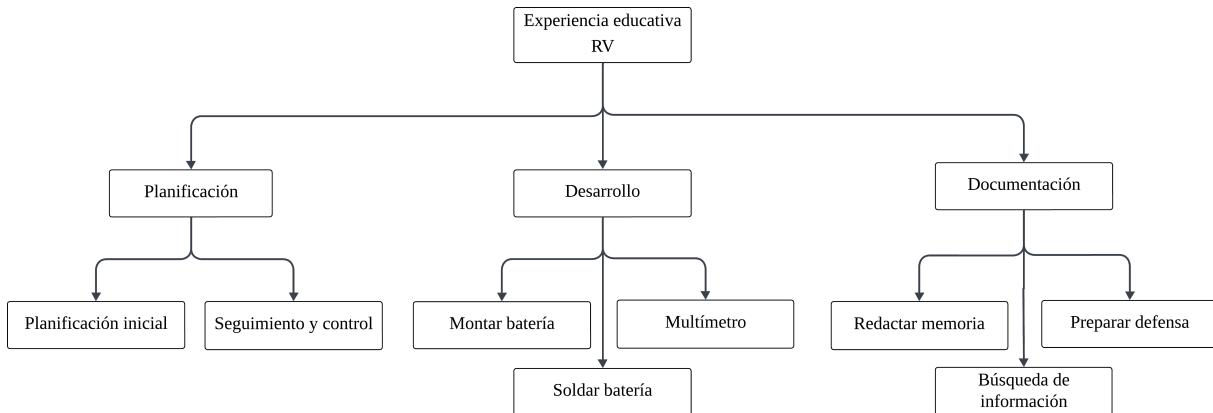
- **Desarrollo de la aplicación**

El desarrollo de la aplicación se divide en tres partes principales:

- Montaje de la batería. Esta tarea corresponde a la parte visual e interactiva del proyecto. Incluye el modelado 3D de los objetos, así como la programación de las interacciones necesarias para ensamblar la batería.
- Soldadura de la batería. Esta parte se centra en las funciones asociadas con el proceso de soldadura. Lo que incluye la vinculación de los objetos de la escena con estructuras de datos programables y la creación de métodos para verificar la soldadura.
- Multímetro. Se centra únicamente en programar el funcionamiento del multímetro dentro del entorno virtual.

■ Documentación

Este paquete abarca la elaboración de la memoria así como la búsqueda y análisis de la información necesaria para su redacción. Además, incluye la preparación de la defensa del proyecto.



**Figura 2.1:** Diagrama paquetes de trabajo

## 2.2. Cronograma y estimaciones de tiempo

En la figura 2.2 se observa el diagrama de Gantt. En este diagrama se indica la fecha de inicio y de fin de cada tarea.

| Paquete de trabajo | Tarea   | MAYO | JUNIO | JULIO | AGOSTO | SEPTIEMBRE |
|--------------------|---|------|-------|-------|--------|------------|
| Planificación      | Planificación inicial                         |      |       |       |        |            |
|                    | Seguimiento y control                         |      |       |       |        |            |
| Desarrollo         | Montar batería                                |      |       |       |        |            |
|                    | Soldar batería                                |      |       |       |        |            |
|                    | Multímetro                                    |      |       |       |        |            |
| Documentación      | Redactar memoria +<br>búsqueda de información |      |       |       |        |            |
|                    | Preparar defensa                              |      |       |       |        |            |

**Figura 2.2:** Diagrama de Gantt

En el inicio del proyecto, los paquetes que más dedicación han requerido han sido la planificación, la búsqueda de información y las fases iniciales del desarrollo del espacio virtual. A medida que el desarrollo de la aplicación ha avanzado, la redacción de la memoria ha cogido más peso. En cambio, el seguimiento y control del proyecto se ha llevado a cabo durante todo su desarrollo.

En la tabla 2.1 se muestra el tiempo estimado para cada tarea.

| Conjunto      | Tarea                        | Duración Estimada |
|---------------|------------------------------|-------------------|
| Planificación | Planificación inicial        | 10h               |
|               | Seguimiento y control        | 15h               |
| Desarrollo    | Montar batería               | 50h               |
|               | Soldar batería               | 65h               |
|               | Multímetro                   | 40h               |
| Documentación | Redacción de la memoria      | 80h               |
|               | Análisis del estado del arte | 20h               |
|               | Preparación de la defensa    | 20h               |
| <b>TOTAL</b>  |                              | <b>300h</b>       |

Tabla 2.1: Distribución estimada de horas de trabajo por paquete

## 2.3. Análisis de riesgos

En este apartado se describen los riesgos que pueden afectar a este proyecto y los planes de contingencia de cada uno.

- **Perder el progreso por fallo de la aplicación:**

No es inusual que algunos programas informáticos se cierren inesperadamente cuando reciben demasiada carga de trabajo. Para prevenir la perdida de progreso, se ha usado las aplicaciones *Unity Version Control* y *Unity Cloud* para guardar el progreso del programa en la nube.

- **Trabajo demasiado extenso:**

Dado que se busca cierto grado de realismo en este simulador, el trabajo puede resultar demasiado extenso para completarse dentro del tiempo disponible. En caso de no contar con suficiente tiempo para finalizar el proyecto cumpliendo todas las expectativas iniciales, se simplificarán las partes del trabajo que no afecten significativamente al objetivo principal.

- **Problemas al conectar distintos aparatos:**

Para asegurar el correcto desarrollo de esta aplicación es necesario testear cada cambio en las gafas de realidad virtual. La idea principal es usar *Meta Link*, que permite conectar el dispositivo de realidad virtual con el ordenador a tiempo real. Sin embargo, *Meta Link* requiere que el ordenador cumpla con unos requisitos mínimos para poder establecer la conexión. En caso de que el ordenador no cumpla esos requisitos, se usará la aplicación *Side Quest*, una alternativa que permite transferir archivos entre el ordenador y el dispositivo de realidad virtual.

## 2.4. Seguimiento y control

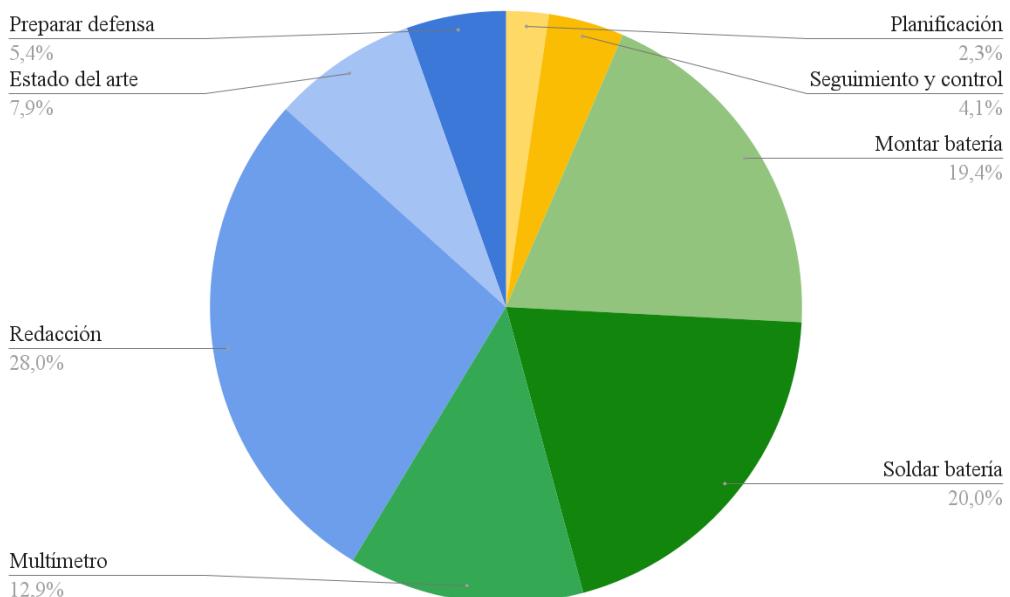
El seguimiento y control de este proyecto se ha realizado mediante reuniones y el uso de herramientas colaborativas como Google Drive. Durante las primeras semanas del proyecto se organizaron reuniones para orientar el desarrollo del trabajo y evaluar el progreso. Una vez la aplicación alcanzó un nivel de avance suficiente, se comenzó a distribuir el material de desarrollo, como archivos APK y videos, a través de una carpeta compartida. Además, se utilizó el correo electrónico como el canal para el intercambio de observaciones y retroalimentación. En la sección [5.1](#) del apéndice se proporciona el link a la carpeta de Google Drive que contiene videos de la aplicación, así como el archivo apk de esta.

## 2. PLANIFICACIÓN

---

| Tarea                        | Duración Estimada | Duración Real   |
|------------------------------|-------------------|-----------------|
| Planificación inicial        | 10h               | 7h 25'          |
| Seguimiento y control        | 15h               | 13h 10'         |
| Montar batería               | 50h               | 64h 15'         |
| Soldar batería               | 65h               | 63h 40'         |
| Multímetro                   | 40h               | 42h             |
| Redacción de la memoria      | 80h               | 89h 15'         |
| Análisis del estado del arte | 20h               | 25h 15'         |
| Preparación de la defensa    | 20h               | 17h 20'         |
| <b>TOTAL REAL</b>            |                   | <b>320h 50'</b> |

**Tabla 2.2:** Duración estimada y real



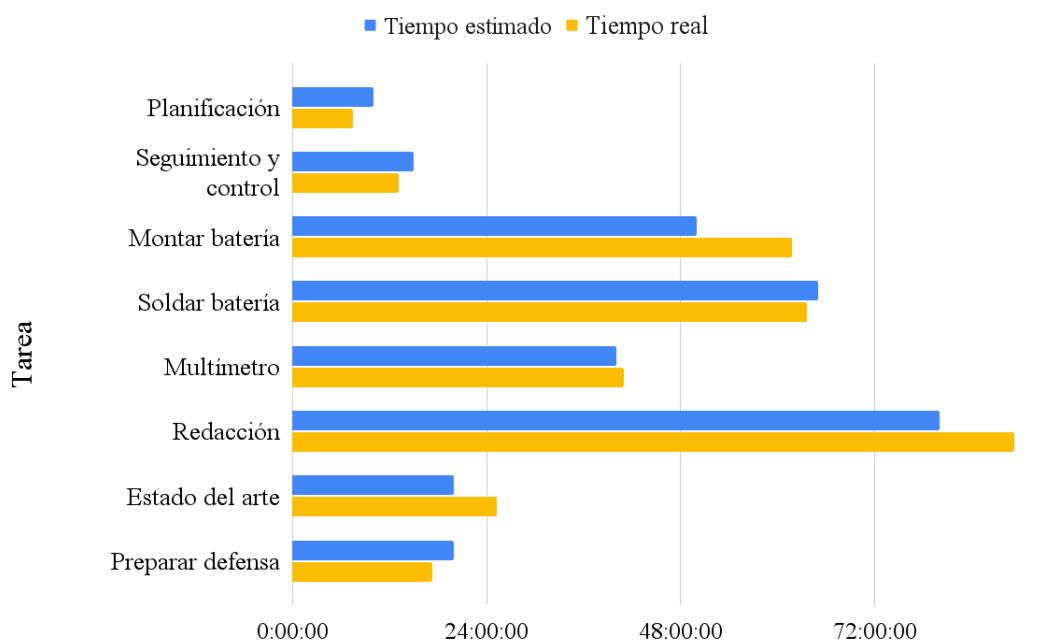
**Figura 2.3:** Tiempo dedicado a cada tarea

En la tabla 2.2 se muestra la comparación del tiempo estimado para cada tarea y el tiempo real empleado. Si bien algunas tareas (como la planificación o el seguimiento) han requerido menos tiempo del previsto, otras actividades relacionadas con el desarrollo de la aplicación han requerido de más atención. Este aumento de tiempo se debe principalmente a la aparición de errores imprevistos durante el proceso de desarrollo.

La diferencia entre las horas estimadas y las reales se puede observar de forma más clara en el gráfico 2.4. Aunque algunas tareas hayan requerido más tiempo que el estimado, otras han precisado de menos tiempo. En conjunto ha supuesto un desajuste de 20 horas por encima de lo planificado.

Observando el gráfico 2.3 se puede ver que en proporción con el resto de paquetes de trabajo, el desarrollo de la aplicación es al que se le ha dedicado más tiempo, abarcando el 52,3 % del tiempo total. El 47,7 % restante se distribuye de la siguiente manera:

- Documentación: 41,3 %
- Planificación: 6,4 %



**Figura 2.4:** Tiempo estimado y tiempo real



---

## Capítulo 3

# Realidad virtual en la educación

---

La implementación de la realidad virtual (RV) en el ámbito de la educación ha sido objeto de estudio desde la década de 1990. Un análisis bibliométrico [6], reveló que, si bien la publicación de artículos en este campo creció lentamente hasta el 2014, A partir de ese año se produjo un notable aumento, coincidiendo con el lanzamiento de dispositivos más asequibles como Google Cardboard. Hoy en día, con la tecnología más avanzada y accesible, los dispositivos de realidad virtual se han convertido en una herramienta de formación más atractiva para las empresas e instituciones educativas.

No obstante, la idea de implementar estos dispositivos en la educación ha generado un nuevo debate sobre sus limitaciones. Por un lado, algunos expertos señalan los posibles efectos secundarios derivados de su uso prolongado, como mareos, fatiga visual y cansancio general [7, 8], además de la falta de estudios sobre sus posibles efectos adversos a largo plazo. También se menciona que, pese a la reducción de los precios de estos equipos en los últimos años, su coste sigue siendo superior al de otras herramientas educativas como tablets u ordenadores, y aún existe una falta de hardware y software específico para ciertos contextos.

Por otro lado, diversos estudios respaldan los beneficios de la RV. Las simulaciones de RV son especialmente útiles para la formación en contextos de riesgo como la aviación, la medicina o la ingeniería, ya que crean un entorno seguro y controlado. Además, la incorporación de entornos virtuales se ha demostrado beneficiosa para el desarrollo de habilidades prácticas y comprensión de conceptos complejos, incrementando la motivación y satisfacción del alumnado. Un ejemplo es el trabajo de Ahmad A. Mazhar y Mustafa M. Al Rifae [7], en el que se comparó el desempeño de dos grupos de estudiantes en un curso que requería habilidades avanzadas de modelado 3D. El grupo que se formó con RV obtuvo mejores resultados en todos los indicadores evaluados, incluyendo la retención de conocimiento, la involucración y la capacidad de aplicar los aprendidos en contextos reales.

Sin embargo, para que la experiencia educativa sea efectiva y cumpla su objetivo, el espacio virtual debe cumplir unos requisitos, entre ellos, el no requerir de procesos de computación excesivos y ser lo suficientemente realista como para crear una experiencia inmersiva.

Según el estudio [5] la investigación sobre el uso de esta tecnología se ha extendido a áreas como la geografía y las matemáticas y la geometría. Sin embargo, el campo que más atención ha recibido ha sido la medicina, donde numerosos estudios han analizado la aplicación de la RV y han concluido que su incorporación puede ser una herramienta complementaria a la educación

convencional.

Por contra, en el ámbito de la ingeniería, la implementación de la RV era un terreno que apenas había comenzado a explorarse. El objetivo de este estudio fue ofrecer una visión general sobre cómo se ha aplicado la realidad virtual en el área de la ingeniería entre los años 2015 y 2019. De los estudios analizados, once de ellos se centraban específicamente en alguna disciplina de la ingeniería, mientras que uno abordaba su uso tanto en la ingeniería eléctrica como mecánica, y otro en ingeniería civil, industrial y mecánica.

En cuanto a la distribución por especialidad, se encontró el mismo número de investigaciones ( $n = 4$ ) enfocadas en la ingeniería mecánica e ingeniería civil. Un número menor de estudios ( $n = 2$ ) se centraba en la ingeniería eléctrica e ingeniería industrial, mientras que en áreas como la ingeniería neumática y la ingeniería del software se halló solamente un estudio en cada caso. Además, seis estudios no especificaban explícitamente en qué rama de la ingeniería se centraba la investigación [5].

En 2014 se publicó la aplicación educativa VEMA (Virtual Electrical Manual), con el fin de ayudar a los y las estudiantes a entender los conceptos de los circuitos eléctricos sin la necesidad de estar en un laboratorio real o exponerse a sus riesgos [9]. La aplicación cuenta con distintos apartados con conceptos fundamentales para el alumnado, como el equipo, capacitores (condensadores eléctricos), construcción de circuitos de corriente directa, corriente alterna y resonancia. Cada apartado cuenta con información específica del circuito eléctrico.

Más recientemente, en 2023, se publicó un estudio con el propósito de analizar las aplicaciones de la realidad aumentada y realidad virtual en ingeniería eléctrica desde un punto de vista académico e industrial [10].

Estas son algunas de las aplicaciones de realidad virtual estudiadas:

Se creó una aplicación de RV para formar a los trabajadores de una empresa en como armar un transformador con el fin de ahorrar costes durante la pandemia de Covid-19 [11].

Otra compañía creó una aplicación de RV con el fin de desarrollar un entorno de entrenamiento seguro para la inspección y verificación de interruptores de línea 10kV [12]. En su artículo explican que entrenar con materiales reales supone demasiados riesgos y costes, esta aplicación aborda estos dos problemas además de ofrecer una formación más completa.

Otra aplicación creada con el fin de ayudar a los alumnos de ingeniería eléctrica fue la de Zoltán Kvasznica [13]. En su artículo afirma que explicar la estructura y el funcionamiento de las máquinas eléctricas mediante imágenes estáticas es complicado y que las animaciones en 3D ayudan a entender mejor los conceptos.

Por otro lado, en el artículo [14], exponen la aplicación V-LAB. Esta aplicación simula los laboratorios de maquinas eléctricas de la Universidad Americana de Kuwait y permite al alumnado llevar acabo distintos experimentos eléctricos, además esta aplicación cuenta con un sistema de calificaciones.

La aplicación de recursos virtuales ha crecido durante los últimos años. Según una revisión del 2023 las disciplinas en las que predomina la RV son la física, ciencia computacional, ingeniería y medicina [4].

En definitiva, la realidad virtual se presenta como una herramienta con un gran potencial para transformar la educación, aunque todavía presente limitaciones técnicas o económicas que limitan su implementación. Los estudios existentes demuestran que las aplicaciones de RV pueden mejorar la motivación y la comprensión de la práctica, además de eliminar totalmente los riesgos que algunas pueden conllevar.

### 3.1. Fabricación de baterías en realidad virtual

Tal y como se ha mencionado anteriormente, ya existen algunas aplicaciones de realidad virtual orientadas a la formación profesional, así como a la educación superior, en el área de la ingeniería electrónica. Pero, en lo que respecta al sector de las baterías, la literatura existente sobre aplicaciones de realidad virtual se centra principalmente en la fabricación de celdas de litio individuales.

En 2023 un grupo dirigido por el profesor Alejandro A. Franco publicó dos aplicaciones relacionadas con la ciencia de las baterías. La primera, una aplicación cooperativa de realidad mixta, en la cual una persona conducía un vehículo eléctrico por un entorno virtual mientras que el resto del grupo optimizaba el circuito eléctrico mediante objetos impresos con el fin de que el vehículo completara una misión. La segunda, llamada SIMUBAT 4.0, es una aplicación que simula un gemelo digital de una línea de producción de celdas de batería. Esta aplicación se puede utilizar mediante un navegador de Internet o mediante un dispositivo de realidad virtual [15]. Posteriormente crearon la aplicación Simubat 4.0 Gen-2, combinando la realidad virtual y la mixta.

Finalmente, en [16] se introduce una plataforma educativa de realidad virtual llamada “Battery manufacturing metaverse” (BMM), sucesora de la aplicación Simubat 4.0. Esta aplicación recrea una linea de fabricación de baterías de ion de litio, permitiendo a los y las alumnas de distintas localizaciones colaborar entre sí. Los y las autoras explican que, motivados por el éxito de sus anteriores herramientas educativas de realidad virtual y realidad mixta, crearon esta nueva herramienta que promueve la accesibilidad, la inclusión y el aprendizaje colaborativo del proceso de manufacturación de baterías. Algunas de las funcionalidades que incorpora esta nueva aplicación incluye la posibilidad de que las y los usuarios seleccionen el tipo de químico de la batería, así como modificar los valores de los parámetros de cualquier máquina. Además, la aplicación permite utilizar configuraciones “incorrectas” con el fin crear escenarios más flexibles. La aplicación también ofrece retroalimentación detallada basada en los parámetros elegidos.

A diferencia de las aplicaciones revisadas anteriormente, la aplicación propuesta en este trabajo fin de grado se centra en el montaje de baterías de litio. En su versión experimental, la aplicación se dedica específicamente a baterías con configuración 10s4p, destinadas a un vehículo eléctrico ligero, como podría ser un patinete o una bicicleta. La aplicación de realidad virtual recrea un aula en la que la persona usuaria dispone de modelos digitales de todos los materiales y herramientas necesarias para montar y soldar la batería. El alumnado debe aplicar sus conocimientos sobre el proceso de montaje para completar correctamente la tarea. Para que la práctica se considere correcta, es necesario colocar todas las celdas en la posición adecuada y soldar las cintas de níquel evitando tanto los cortocircuitos como los circuitos abiertos. Además, la persona usuaria dispone de un multímetro con el que puede verificar parámetros básicos de la batería, como el voltaje, una vez finalizado el proceso de montaje.



---

## Capítulo 4

# Desarrollo de la aplicación

---

En este capítulo se describe el proceso de ensamblaje de una batería además de los componentes y herramientas necesarias para llevarlo a cabo y cómo se han digitalizado. También se describe cómo se han programado las interacciones entre los objetos en la escena virtual.

### 4.1. Proceso de montaje de una batería

La fabricación de baterías es un proceso complejo que conlleva numerosos riesgos y variables, los cuales pueden dificultar el trabajo, como el voltaje de cada celda, los ajustes del soldador o la forma de las cintas de níquel.

Antes de comenzar, con el fin de garantizar el correcto funcionamiento de la batería, es crucial verificar el voltaje de todas las celdas y asegurarse de que las cintas de níquel no presenten defectos ni holguras que puedan comprometer la calidad de la soldadura.

Hoy en día existen varios métodos para soldar baterías de litio. Entre los más comunes se encuentra la soldadura por puntos, que es la técnica implementada en esta aplicación.

Las soldadura por puntos es una técnica de resistencia que se basa en la presión y temperatura. Una corriente eléctrica pasa a través de un electrodo, generando el calor necesario para que la cinta de níquel se funda con la batería.

Los principales problemas asociados a este proceso suelen derivar de una configuración incorrecta de la máquina de soldadura, ya que la corriente, el tiempo y la presión deben ajustarse en función de las características del material de soldadura empleado. Un exceso de corriente puede quemar la batería, mientras que una presión insuficiente puede resultar en una soldadura débil o defectuosa.

Otro aspecto fundamental en el montaje de baterías es la distribución de las celdas, que debe adaptarse al espacio disponible. En esta aplicación experimental se ha implementado exclusivamente la configuración 10s4p en forma rectangular, una matriz de 4x10 compuesta por 10 columnas de celdas conectadas en serie y 4 filas en paralelo.

#### 4.1.1. Materiales y herramientas

En esta sección se enumeran los materiales y herramientas necesarias para llevar a cabo la práctica de soldadura:

- **Celdas de litio:** Se requieren 40 pilas modelo 18650.

- **Cinta de níquel:** Material utilizado para conectar las celdas entre sí.
- **Materiales de aislamiento:** Estos materiales son cruciales para prevenir cortocircuitos. Uno de los más usados son los tubos termorretráctiles.
- **Soporte:** Estructura plástica que mantiene las celdas en su posición durante el montaje.
- **Multímetro:** Instrumento que permite a la persona usuaria comprobar el estado de la batería una vez completado el montaje.
- **Soldador por puntos:** Herramienta principal para fijar las cintas de níquel a las celdas.
- **Alicates de corte:** Herramienta que permite cortar las cintas de níquel a la longitud requerida.
- **Pistola de calor:** Herramienta utilizada para contraer los tubos termorretráctiles alrededor de las conexiones.
- **Equipo de seguridad:** Los guantes aislantes y las gafas de seguridad son elementos indispensables en los procesos de soldadura. También es recomendable disponer de extintores de incendios.

#### **4.1.2. Pasos para el montaje de una batería 10s4p**

A continuación se detallan los pasos a seguir para el montaje de un paquete de baterías de configuración 10s4p.

##### **4.1.2.1. Preparación**

Antes de comenzar el proceso de montaje resulta fundamental evaluar el voltaje, la capacidad y la resistencia de cada celda. Aunque este paso resulta esencial en el proceso real, no es necesario realizarlo en el proceso virtual, ya que todos los componentes y herramientas han sido programados para operar en condiciones ideales.

##### **4.1.2.2. Montaje del paquete de baterías**

Para mantener las celdas individuales en su posición se emplean soportes plásticos, los cuales pueden ensamblarse entre sí para conformar la estructura requerida. Para garantizar una fijación adecuada, cada celda debe colocarse entre dos soportes, uno en la parte superior y otro en la inferior. En este caso, se deben ensamblar de manera que formen un rectángulo de 4x10 celdas. Es fundamental colocar las 40 celdas en la orientación correcta, de manera que las columnas puedan conectarse en paralelo y las filas en serie.

##### **4.1.2.3. Soldadura por puntos**

Las cintas de níquel deben colocarse asegurándose de que queden correctamente alineadas con las celdas y de que su colocación no derive en un cortocircuito. Es imprescindible soldar todos los puntos de unión entre las cintas de níquel y las celdas de litio empleando la corriente y la presión adecuadas, evitando así dañar los materiales o el rendimiento de la batería.

##### **4.1.2.4. Aislamiento**

Para evitar cortocircuitos es esencial utilizar los tubos termorretráctiles sobre las conexiones expuestas, además de aplicar cinta resistente al calor a las zonas más sensibles.

#### 4.1.2.5. Sistema de gestión de baterías

El BMS, o sistema de gestión de batería, es un componente encargado del control y gestión avanzada del sistema de almacenamiento. Su función principal es controlar la carga y descarga de la batería, aunque también puede realizar otras funciones que permiten recopilar, procesar y almacenar información importante de la batería.

Si bien se trata de una herramienta útil, no resulta esencial para el montaje de baterías, por lo que se ha decidido no implementarlo en la práctica virtual.

## 4.2. Modelos digitales

En este apartado se describen los modelos que se han diseñado específicamente para la escena virtual. Empezando por los materiales esenciales para montar la batería y siguiendo con los demás objetos que completan la escena.

Cabe señalar que de los materiales y herramientas mencionados en el punto anterior los que se utilizan en la práctica son: las celdas de litio, el soporte, las cintas de níquel, el soldador y el multímetro. Esto se debe a que la aplicación simula una práctica bajo condiciones ideales, de modo que todos los materiales y herramientas están configurados previamente para su uso inmediato, evitando la necesidad de modificaciones.

Antes de describir los componentes de la escena, es necesario aclarar algunos términos específicos de Unity que se utilizan en este capítulo:

- **Socket:** Este componente permite fijar objetos de la escena a otros objetos o zonas designadas.
- **Collider:** Se utilizan para designar el área física de un objeto, gestionando las colisiones con otros objetos. Pueden adoptar distintas formas (cubo, esfera o cápsula) para adaptarse al objeto que lo usa.
  - **Trigger Collider:** Es una variante del Collider que, en lugar de detener las colisiones, detecta cuando otro objeto entra en el área definida.
- **Layer mask:** Estas máscaras sirven para gestionar las interacciones definidas por el paquete XR Interaction Toolkit entre objetos en Unity. Un objeto únicamente puede interactuar con aquellos con la misma máscara. Es posible asignar varias máscaras a un mismo objeto. Es importante no confundir las layer mask (del paquete XR Interaction Toolkit) con las *layer* de Unity. Las layer mask solamente delimitan las acciones realizadas por los componentes *interactor* e *interactable* del mismo paquete. En cambio, las layer de Unity delimitan todas las interacciones con las físicas de la escena virtual.
- **Script:** Es un conjunto de instrucciones escritas en un lenguaje de programación (en este caso C#) que define el comportamiento de los distintos objetos de la escena.
- **Frame o fotograma:** Es una imagen concreta dentro de una sucesión de imágenes en movimiento. En aplicaciones de realidad virtual se recomienda una tasa de fotogramas de 72 fotogramas por segundo (fps), 90fps o 120fps para garantizar una experiencia fluida y evitar mareos.

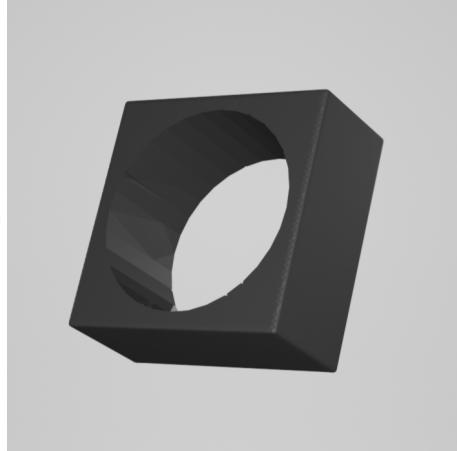
#### 4.2.1. Soporte 10s4p

En la práctica real, es posible ensamblar baterías de distintos tamaños y, además, construir una misma configuración de diversas formas. Para lograr el tamaño deseado, se utilizan soportes

individuales de plástico que se acoplan entre sí (ver imagen 4.1), formando una estructura con forma de malla.



(a) Soporte de plástico para pilas 18650



(b) Modelo digital soporte

**Figura 4.1:** Soporte individual real y modelo digital

En esta primera versión de la experiencia educativa, únicamente es posible crear una batería de configuración 10s4p. Por esa razón, para simplificar y agilizar el desarrollo de la práctica, la estructura del soporte ya se presenta completamente ensamblada. Además, con el fin de facilitar la evaluación de la actividad, se ha incorporado una pequeña diferencia entre el soporte superior e inferior para poder distinguirlos (ver imagen 4.2).

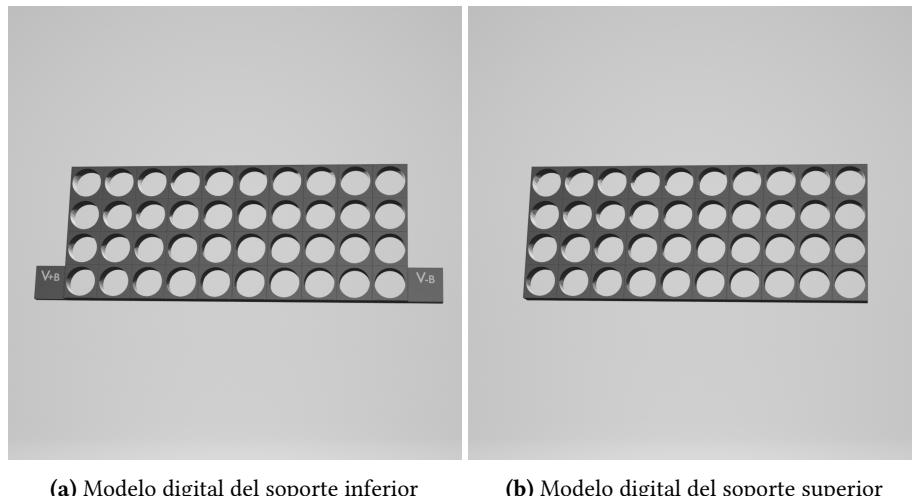
El soporte inferior incluye a los laterales dos marcadores que señalan los puntos de medición del voltaje total: uno indica el polo positivo ( $V+_{Total}$ ) y el otro indica el polo negativo ( $V-_{Total}$ ). De esta manera, se establece una única disposición correcta de las pilas, garantizando que la práctica solo se considere válida cuando estas se coloquen respetando la orientación. A diferencia, el soporte superior no cuenta con ningún indicador.

El soporte inferior es la pieza principal de la práctica, ya que la batería se monta alrededor de esta.

Se han implementado dos formas para poder fijar las pilas al soporte. La primera versión, y la que más se ha probado durante el desarrollo de la aplicación, cuenta con dos sockets para cada celda, uno por cada orientación posible de la pila, como se puede ver en la imagen 4.11a. Estos sockets solamente interactúan con la layer mask “battery”. El problema de esta distribución reside en que, para colocar las pilas en el socket inferior, es necesario evitar el contacto con el socket superior, lo que complica especialmente la inserción de las pilas centrales.

Para solucionar este problema, se ha implementado la segunda forma, aunque menos probada. Se ha creado una clase hija a partir de la clase “XR Socket Interactables” del paquete XR Interaction Toolkit. Los XR Socket Interactables base, solamente permiten fijar los objetos en una posición predeterminada, pero en la práctica es necesario colocar la pila de dos formas distintas. Por esa razón, se ha incluido un método que detecta cual es la orientación aproximada de la pila antes de engancharla al socket y la fija de esa manera.

Finalmente, cada soporte cuenta con ocho sockets en los que es posible colocar cintas de níquel (ver imagen 4.15). Para evitar que otros objetos se enganchen es estos sockets se ha limitado su interacción a la layer mask “nickel”, que corresponde a la máscara de las cintas.



(a) Modelo digital del soporte inferior

(b) Modelo digital del soporte superior

**Figura 4.2:** Soportes 10s4p

#### 4.2.2. Celda de litio

A diferencia de las pilas 18650 reales donde la diferencia entre los polos es más sutil, los polos de los modelos digitales están señalizados de distintos colores para facilitar su identificación; además de un signo positivo (+) o negativo (-) en ambas caras. Asimismo, todas las baterías tienen el mismo voltaje nominal, 3.7 voltios, por lo que no es necesario verificarlo antes de utilizar cada pila.

Todas las pilas integran un trigger collider en cada uno de sus polos, lo que permite detectar los bornes del multímetro.



(a) Pila 18650 real

(b) Modelo digital pila 18650

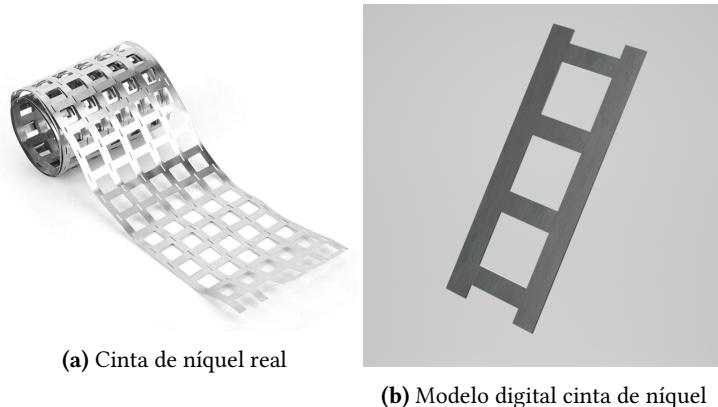
**Figura 4.3:** Pila real y modelo digital

#### 4.2.3. Cinta de níquel

Con el fin de facilitar el desarrollo de la actividad, se ha decidido proporcionar cintas de níquel con un tamaño ya definido que encaja en la batería sin necesidad de modificarla (ver imagen 4.5). Utilizando este tipo de cintas es posible soldar 8 pilas utilizando una sola tira de níquel.

Para soldar correctamente la cinta a las pilas, es necesario soldar donde ambas entran en contacto. Por esa razón, en esas ocho zonas, la cinta cuenta con un trigger collider que detecta las

interacciones con el soldador. Cuando se presiona el soldador sobre uno de esos puntos, aparece una marca en la cinta indicando que la soldadura se ha realizado correctamente.

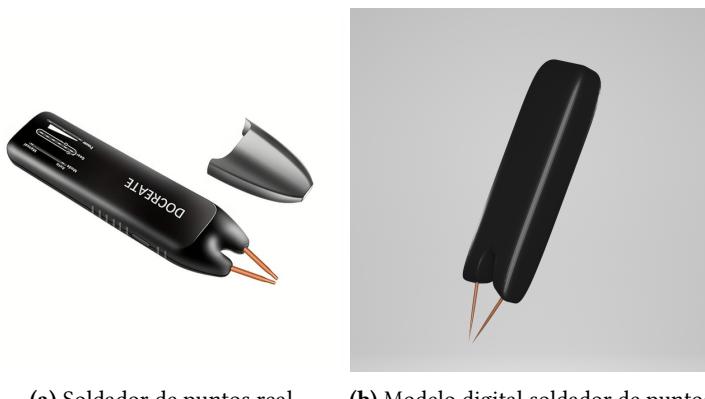


**Figura 4.4:** Cinta de níquel real y modelo digital

#### 4.2.4. Soldador

Existen varios modelos de soldadores de puntos, con el fin de facilitar el desarrollo de la aplicación se ha elegido un modelo inalámbrico. Mientras que en la práctica real el alumnado debe elegir la potencia adecuada del soldador, con el fin de simplificar el proceso, la potencia y el tiempo que se debe usar en la práctica virtual son parámetros fijos y no es posible modificarlos. Para facilitar el proceso de soldadura se ha incorporado una luz verde que se enciende cuando la soldadura se ha realizado correctamente.

Para poder detectar cuando el soldador entra en contacto con las cintas de níquel, la punta de este tiene un pequeño collider con la etiqueta “soldador”.



**Figura 4.5:** Soldador de puntos real y modelo digital

#### 4.2.5. Multímetro

Para la versión experimental de la aplicación, el multímetro solamente tiene función de voltímetro y se controla a través de un botón en la UI, por lo que la rueda y los botones integrados son decorativos.

El multímetro permanece fijo en la escena como un objeto estático, por lo que no puede ser desplazado ni manipulado directamente. La interacción se limita exclusivamente a los bornes de medida, que pueden moverse libremente y colocarse sobre los polos de las celdas para realizar las comprobaciones correspondientes. Cabe mencionar que los terminales del multímetro no están

conectados mediante cables, ya que esta funcionalidad no se ha implementado en esta fase de la aplicación.



(a) Multímetro real

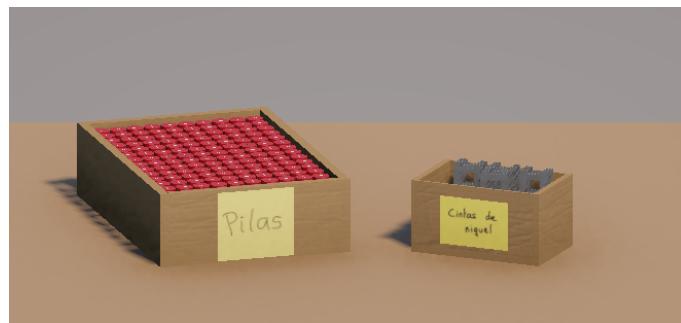
(b) Modelo digital del multímetro

**Figura 4.6:** Multímetro real y modelo digital

#### 4.2.6. Cajas

La función principal de las cajas es evitar la sobrecarga de la aplicación causada por la creación de múltiples objetos no estáticos en la escena, además de no limitar el número de pilas y cintas de níquel disponibles y así prevenir problemas en caso de que algún elemento desaparezca debido a un fallo de la aplicación. Cada caja genera una instancia del objeto asignado cuando se acerca cualquiera de los controladores a ella.

Por otra parte, para evitar interacciones no deseadas, estos objetos no se ven afectados por las físicas de la escena ni otras interacciones de la persona usuaria.

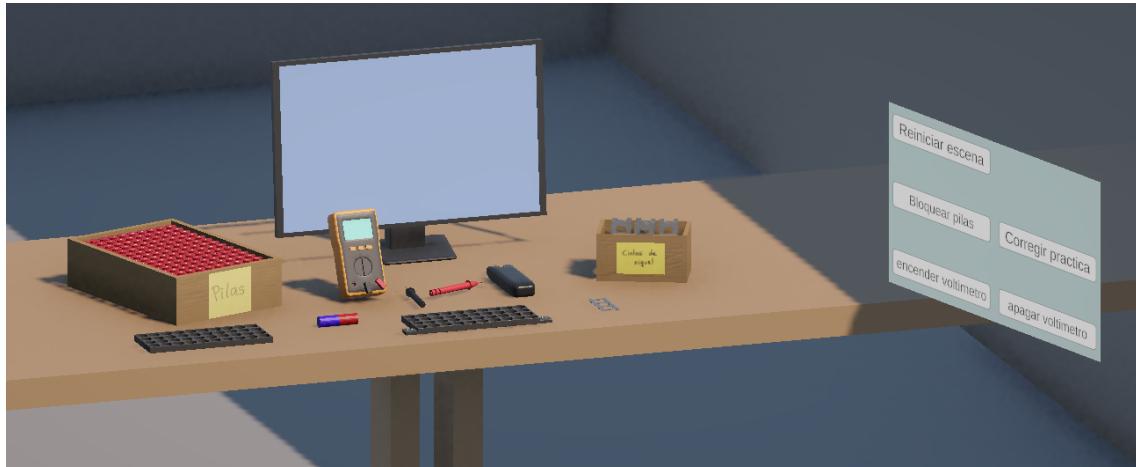
**Figura 4.7:** Modelo digital cajas

#### 4.2.7. Otros objetos de la escena

En la escena existen algunos objetos secundarios cuya función es principalmente estética y no constituyen elementos esenciales para la práctica. Entre ellos se encuentran, por ejemplo, la pantalla que muestra el resultado de la actividad así como las mesas y los ordenadores del aula, que pretenden aumentar el grado de inmersión en la escena y recrear de una manera más realista un entorno educativo.

Con el fin de recrear de manera más realista un aula universitaria, también se han añadido mesas adicionales con ordenadores. Estos objetos son completamente estáticos y decorativos, y su propósito exclusivo es reforzar la ambientación del escenario virtual (ver imagen 4.9).

Para esta versión experimental, todos los objetos se han modelado utilizando la menor cantidad de polígonos posible, con el fin de optimizar el rendimiento y evitar sobrecargar la escena. Asimismo se han empleado materiales con colores básicos para simplificar su representación visual.



**Figura 4.8:** Mesa principal



**Figura 4.9:** Escena completa

### 4.3. Implementación del montaje

En este apartado se describe cómo se han programado las interacciones entre los objetos. Esta aplicación simula un proceso simplificado del montaje de baterías de litio explicado anteriormente, por lo que se ha decidido no simular algunos pasos, como el aislamiento y el sistema de gestión de baterías.

Para conseguir completar la práctica correctamente es necesario seguir los siguientes pasos:

1. Fijar las 40 pilas correctamente en el soporte inferior
2. Colocar el soporte superior
3. Colocar las cintas de níquel y soldarlas
4. Validar práctica

## 5. Uso del voltímetro

A continuación se explican detalladamente como funcionan las interacciones en cada paso.

### 4.3.1. Fijar las pilas en el soporte inferior

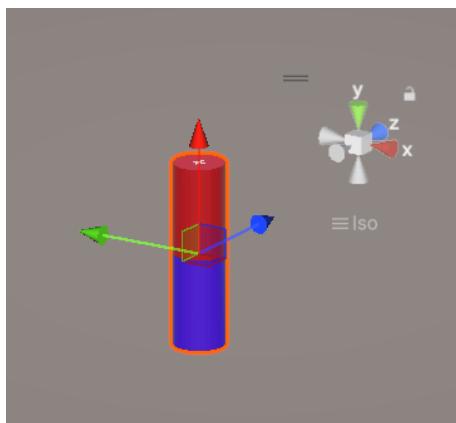
El paso principal para montar la batería es colocar las pilas en el soporte inferior. Para montar correctamente la batería hay que colocar las pilas de forma que sea posible conectar las 10 columnas en paralelo y las 4 filas en serie, por lo que es imprescindible saber en qué orientación son colocadas. Para ello, cada pila tiene integrado el script `ObjectOrientation.cs`.

La función principal de este script es `CheckOrientation()`, la cual devuelve un valor numérico que representa la orientación del eje local X (`transform.rigth`) del objeto que tiene integrado el script respecto al eje Y de la referencia. En este caso, el soporte inferior, cuyo eje Y está alineado con el eje Y de la escena.

Este script se utiliza únicamente para obtener la orientación de las pilas. El eje local X de la pila representa su altura y está alineado de forma que apunta desde el polo negativo al positivo (ver imagen 4.10). El método `CheckOrientation()`, compara la orientación del eje X de la pila (con valores entre -1 y 1) respecto al eje Y de la referencia. El valor 1 indica que la pila está completamente alineada con la referencia, es decir el polo positivo está hacia arriba; mientras que el valor -1 indica que la pila está colocada con el polo positivo hacia abajo.

Dado que los valores entre -1 y 1 son prácticamente infinitos y que resulta muy difícil alinear una pila perfectamente con el eje Y de la referencia, se ha definido un margen de tolerancia: los valores superiores a 0,1 se aproximan a 1, mientras que los valores inferiores a -0,1 se aproximan a -1.

El método devuelve la orientación de la pila. Como solo hay dos posiciones posibles (lado positivo arriba o abajo), en caso de que haya algún fallo y se coloque la pila de otra manera, devolverá 2.



**Figura 4.10:** Ejes pila

Es importante que la orientación de la pila se evalúe desde el sistema de referencia del soporte y no desde el de la escena. Ya que, si el soporte se gira después de colocar la pila, la orientación de la pila habrá cambiado respecto al sistema global, aunque en realidad su orientación respecto al soporte sea la misma.

Como se ha explicado en el apartado anterior, en la versión más probada de la aplicación, cada soporte individual cuenta con dos sockets que permiten fijar las pilas en dos orientaciones posibles (ver imagen 4.11a). Para poder gestionar cada par de sockets, cada pareja usa el script `EnableOneAtATime.cs`, el cual desactiva el socket vacío para que no interfiera con la pila ya colocada, mientras que ambos sockets incorpora el script `HasBatteryOnSelect()`.

La función de `HasBatteryOnSelect.cs` es indicar si hay una pila fijada al socket. Para que estos puedan funcionar correctamente, todos los objetos que integran un socket deben incorporar también un trigger collider. Los trigger collider delimitan la zona en la que puede actuar el socket, por lo que en su ausencia el socket no funcionaría.

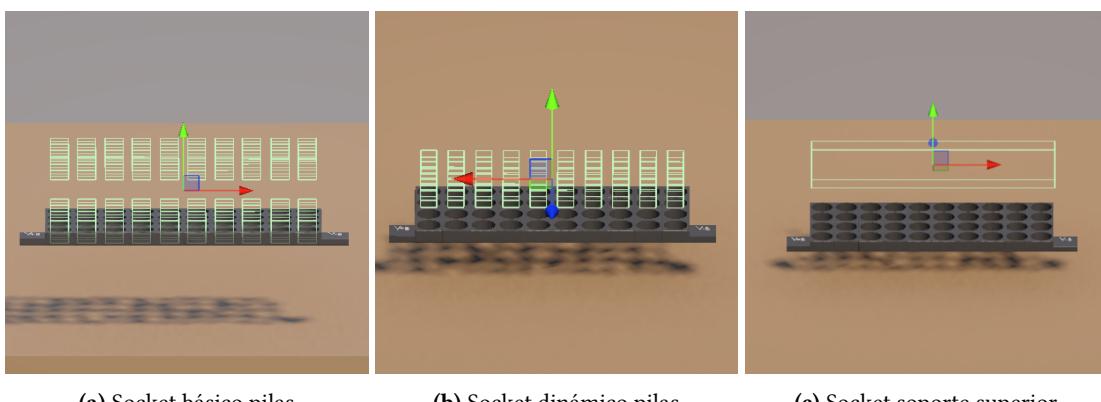
Los componentes de tipo *interactable* (por ejemplo los sockets) integran las funciones `OnSelectEntered(SelectEnterEventArgs args)` y `OnSelectExited(SelectExitEventArgs args)`. Los parámetros `SelectEnterEventArgs args` y `SelectExitEventArgs args` representan los objetos de tipo *interactable*, en este caso los objetos que se acoplan al socket. Estas funciones permiten saber cuando un socket tiene un objeto fijado.

Este script recibe de forma continua la información de orientación de la pila en caso de que haya una acoplada. Este seguimiento constante es necesario ya que, si la orientación se obtiene únicamente en el momento en el que se fija la pila, el valor registrado correspondería al frame anterior y podría no coincidir con la orientación real de la pila una vez asegurada en el socket.

En la otra versión de la aplicación, cada celda del soporte cuenta con un único socket modificado (ver imagen 4.11b). Estos sockets integran el script `DynamicSocket.cs`, el cual redefine la función encargada de determinar la posición en la que se fija el objeto. En el apartado 5.1 se explica cómo se ha implementado esta modificación. Además, este script gestiona la detección de las pilas y su orientación.

En la imagen 4.12 se muestran las 4 celdas de litio colocadas correctamente.

En la subsección 4.3.1.1 explican las diferencias entre la implementación de las dos versiones.



**Figura 4.11:** Sockets del soporte inferior

El script `BatteriesInMount.cs`, implementado en el soporte inferior, es el script principal de la escena y se ocupa de guardar la orientación de las pilas colocadas en una matriz.

Para explicar mejor la ejecución de este script a continuación se muestra el pseudocódigo de las funciones principales.

Primero, al cargar la escena se guardan en una lista todos los sockets vacíos que componen el soporte inferior, se inicializa una matriz de 4x10 con la respuesta correcta `FillCorrectAns()` y se inicia otra matriz de 4x10 a cero (`InitializeUserAns()`) donde se guardará la respuesta del usuario. Luego, se crea una lista con todos los sockets que componen el soporte inferior (ver algoritmo 4.1).

Para evitar la sobrecarga de la aplicación, se ha decidido actualizar la matriz con la respuesta del usuario solamente cuando se pulse el botón “Corregir práctica”, en lugar de actualizarla cada vez que se inserta una pila. Cuando la persona usuaria presiona el botón, se activa la función `FillUserAns()`. Esta función recorre la lista de sockets y guarda la orientación de la pila en

## Start()

---

```

1  i ← 0
2  j ← 0
3  FillCorrectAns()
4  InitializeUserAns();
5  while i < 4
6    while j < 10
7      socket ← find(socket + i + j)
8      lista.add(socket.GetComponent<SocketInteractor>())
9    done
10   done

```

---

**Algoritmo 4.1:** Método inicial BatteriesInMount.cs

su posición respectiva en la matriz. Si el socket no tiene pila, la respuesta será 0.

#### 4.3.1.1. Comparación entre versiones

## FillUserAns()

---

```

1  current ← 0
2  for k = 0; k < lista.size()
3    parejaSockets←lista[k]
4    foreach HasBatteryOnSelect aux in parejaSocket
5      if aux tiene pila then
6        current ← aux.GetOrientation()
7      else
8        continue
9      fi
10     rof
11     i ← k/10
12     j ← k mód 10
13     userAns[i, j]←current
14   rof

```

---

**Algoritmo 4.2:** FillUserAns con XR Socket Interactor

Los elementos de la lista creada en la función Start () (4.1) dependen del tipo de socket utilizado.

En la versión que emplea los sockets originales, la lista almacena 40 objetos de tipo *GameObject*, y a su vez cada uno de ellos contiene una sublistas con los dos sockets asociados.

En cambio, en la versión que utiliza los sockets dinámicos, la lista guarda directamente los 40 componentes de tipo *DynamicSocket*, sin necesidad de recurrir a los objetos completos.

Al usar los sockets básicos es necesario crear scripts que sirvan de intermediarios para poder recoger información sobre las pilas fijadas y poder agrupar los sockets por pares, haciendo el proceso más lento y complicado. En cambio, al utilizar los sockets dinámicos la orientación de las pilas se obtiene directamente desde la clase modificada.

En lo que respecta al desarrollo general de la aplicación, en esta versión experimental las interacciones entre los objetos presentan aspectos susceptibles de mejora. La presencia de va-

## FillUserAns()

---

```

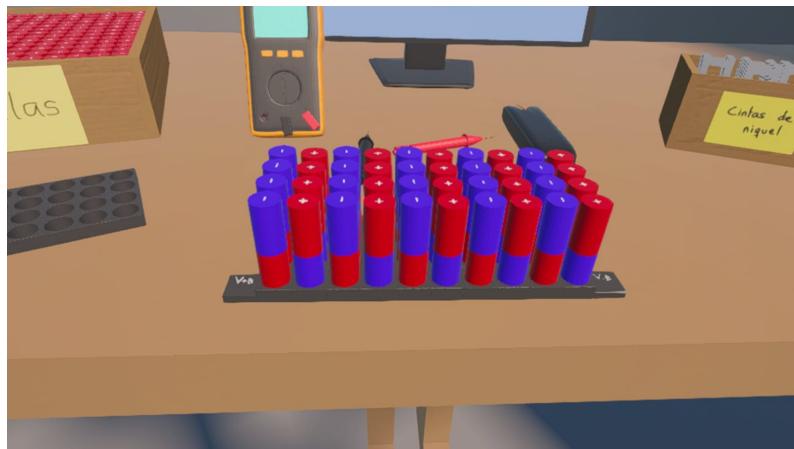
1 current ← 0
2 for k = 0; k < lista.size()
3     current←lista[k].GetOrientation()
4     i ← k/10
5     j ← k mód 10
6     userAns[i, j]←current
7 rof

```

---

**Algoritmo 4.3:** FillUserAns con Dynamic Socket

rios elementos no estáticos en un espacio reducido (como ocurre con las pilas colocadas en el soporte) puede generar interacciones indeseadas. Por ello, para evitar estos comportamientos, se recomienda utilizar el botón “Bloquear pilas” de la interfaz de usuario tras la colocación de cada conjunto de pilas. El script `BatteriesInMount.cs` también se encarga de esta función. Al pulsar el botón, se desactivan los colliders de las pilas colocadas, por lo que después de pulsarlo es imposible coger o cambiar de posición esas pilas.

**Figura 4.12:** Colocar pilas en el soporte inferior

#### 4.3.2. Fijar el soporte superior

Una vez colocadas las 40 celdas, se habilita un socket adicional que permite fijar el soporte superior. Este socket únicamente interactúa con la layer mask “mount” (ver imagen [4.11c](#)).

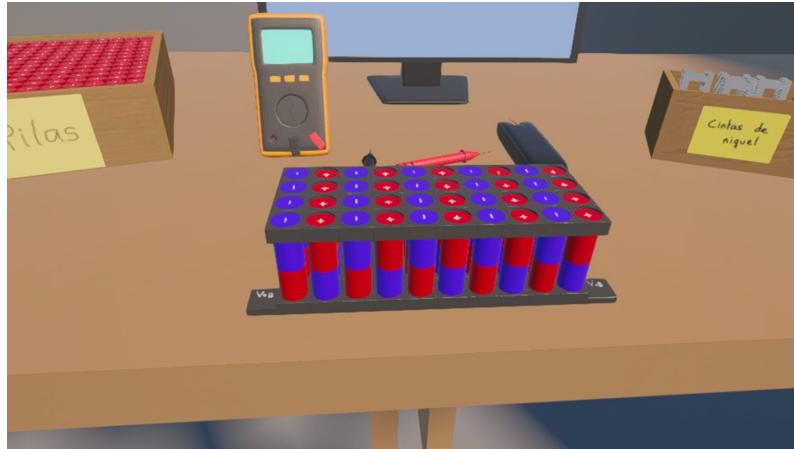
Por otra parte, para evitar que la batería atravesase la mesa al voltearla, es necesario pulsar el botón “Collider superior” de la interfaz.

Se puede ver este paso completado en la imagen [4.13](#).

#### 4.3.3. Colocar las cintas de níquel y soldar

Para conseguir realizar la práctica correctamente es obligatorio colocar el soporte superior antes de fijar las cintas de níquel. Si el soporte superior se retira antes de soldar las cintas, las cintas de níquel que estén colocadas caerán.

Cada soporte tiene integrado el script `CompleteNickelTapes.cs`, el cual comprueba en qué sockets están fijadas las cintas. Con el fin de que la práctica no sea completamente guiada y permitir a la persona usuaria cometer errores, es posible colocar las cintas de níquel en cualquiera



**Figura 4.13:** Colocar soporte superior

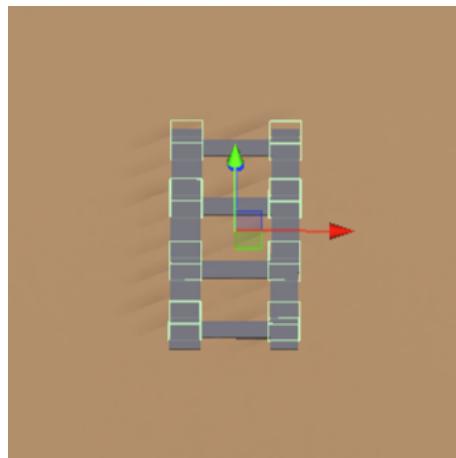
de los ocho sockets de cada soporte, aunque solamente una combinación sea la correcta (ver imagen 4.15).

En la parte superior de la batería es obligatorio colocar las cintas en las posiciones 0, 2, 4, 6 y 8 para que la práctica sea válida, mientras que en el soporte inferior es necesario colocarlas en las posiciones 1, 3, 5 y 7. En caso de que alguna cinta no esté colocada específicamente en estos sockets, la práctica no podrá considerarse correcta.

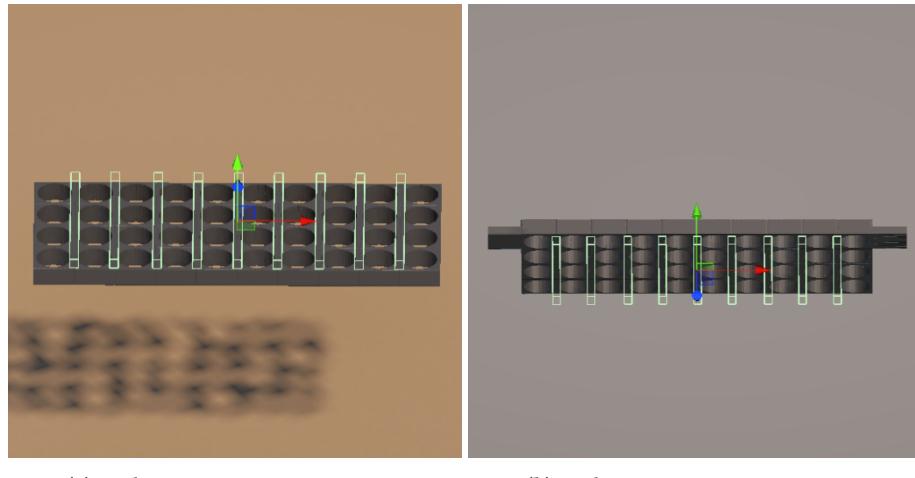
Cada socket implementa el script `HasNiquelTapeOnSelect.cs`, el cual indica cada vez que una cinta queda fijada al socket.

La soldadura solo puede realizarse una vez las cintas de níquel han sido colocadas en el soporte. Para soldar una cinta correctamente, es necesario soldarla en los ocho puntos que están en contacto con las pilas (ver imagen 4.14). Cada uno de estos puntos utiliza el script `Weld.cs`. Este script indica cuándo la punta del soldador ha ejercido presión sobre el punto correspondiente, a su vez, activa la luz del soldador y activa una marca visual en la cinta para indicar que la soldadura es correcta. Una vez que los ocho puntos de la cinta han sido soldados, el script `CheckWeld.cs` valida el proceso e indica que la cinta está soldada correctamente. Una vez soldado al menos un punto de cualquiera de las cintas colocadas en el soporte superior, es imposible retirar tanto la cinta soldada como el soporte superior.

En la imagen 4.16 se puede ver el proceso de soldadura por ambos lados de la batería.



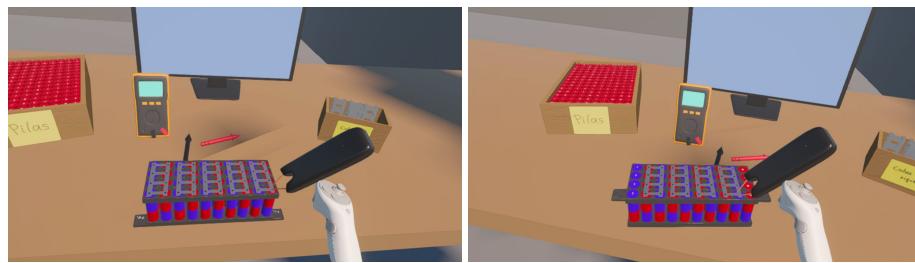
**Figura 4.14:** Puntos a soldar



(a) Sockets cintas soporte superior

(b) Sockets cintas soporte superior

**Figura 4.15:** Sockets para las cintas de níquel



(a) Soldar cintas parte superior

(b) Soldar cintas parte inferior

**Figura 4.16:** Soldar cintas

#### 4.3.4. Validar práctica

El alcance actual de la aplicación no incluye un sistema de retroalimentación detallado, sino que se centra en la simulación del proceso de montaje. Al pulsar el botón “Corregir práctica”, se muestra en la pantalla la matriz con la respuesta del usuario junto con la respuesta correcta y un texto que indica si las cintas de níquel de la parte superior e inferior se han colocado y soldado correctamente. En la imagen 1 se puede apreciar el mensaje en caso de que la práctica sea correcta o incorrecta.

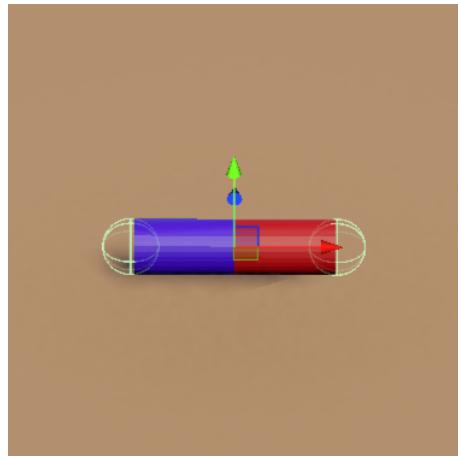
Al presionar dicho botón se activa el script `Check.cs`. Primero se verifica la posición de las pilas mediante el script `BatteriesInMount.cs`, para después comprobar las cintas de níquel de ambos soportes. Si la orientación de las pilas es correcta, el soporte superior se ha colocado, las cintas de níquel se han fijado en los sockets correctos y se han soldado todos los puntos, en la pantalla también aparecerá el mensaje “Práctica correcta”. Si alguno de estos pasos no se ha realizado o no se ha completado correctamente, el mensaje será “Práctica incorrecta”. No hay un número límite de correcciones, por lo que el alumnado podrá realizar todos los intentos que desee.

#### 4.3.5. Voltímetro

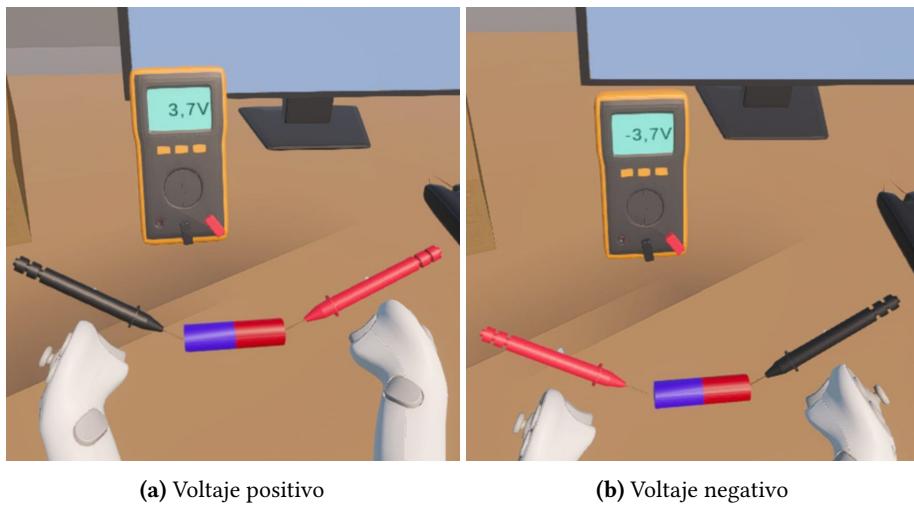
En esta versión de la aplicación, el voltímetro aun es una función en desarrollo, por lo que su utilidad es limitada.

No obstante, ya está implementada la capacidad de medir el voltaje de celdas individuales. Cada pila cuenta con un trigger collider en cada uno de sus polos (ver imagen 4.17), los cuales implementan el script `PolesBattery.cs`. Mediante este script, se detecta cuando los bornes del voltímetro

(con la etiqueta “voltímetro”) entran en contacto con los polos y según la disposición (qué borne toca cada polo), el script `BatteryVolts.cs`, colocado en la pila, muestra el voltaje positivo o negativo en la pantalla del multímetro como se puede ver en la imagen 4.18.



**Figura 4.17:** Trigger collider de los polos



**Figura 4.18:** Medir voltaje de una pila

## 4.4. Otras interacciones

En este apartado se describen otras interacciones que hacen de la aplicación una experiencia más cómoda.

Una de las ellas es la de las cajas. En la escena existen dos cajas que generan una instancia del objeto asignado (una pila o una cinta de níquel) cada vez que se aproxima uno de los controladores. La detección de los mandos se lleva a cabo mediante un trigger collider incorporado en cada caja, el cual envía una señal cuando un objeto con la etiqueta “player” (es decir, los mandos) entra en contacto con ella.

Además, para evitar la generación simultánea de múltiples objetos, se ha incorporado un retardo de un segundo entre cada creación.

Por otra parte, en la versión que se emplean dos sockets por cada pila, fue necesario implementar un script que desactiva uno de los sockets cuando el otro detecta una pila fijada en él, con el fin de evitar interacciones indeseadas.

#### 4. DESARROLLO DE LA APLICACIÓN

---

Para programar el desplazamiento dentro del aula virtual se ha utilizado el componente “XR Origin (XR Rig)” del paquete XR Interaction Toolkit. Este componente ha sido modificado para conservar únicamente las funcionalidades necesarias para la práctica. Por esa razón, se han eliminado características como el teletransporte por el espacio digital, la interacción con objetos a larga distancia y la interacción mediante la vista.

En la tabla 1 se puede ver claramente qué objetos con componentes de tipo “*interactable*” pueden interactuar con cada layer mask.

---

# **Capítulo 5**

## **Conclusiones**

---

Este proyecto ha logrado el objetivo de desarrollar una aplicación de realidad virtual funcional para formación en el ensamblaje y soldadura de paquetes de baterías de litio. A través de un entorno inmersivo y seguro, se ha replicado con éxito el proceso de montaje de una batería con configuración 10s4p, eliminando los riesgos y costes asociados al uso de materiales reales.

A pesar de los logros alcanzados, es importante reconocer las limitaciones de esta primera versión del proyecto. En su estado actual, la aplicación no incorpora un sistema de retroalimentación detallado, lo que dificulta identificar con precisión los fallos cometidos por la persona usuaria. Además se ha optado por una configuración de soldadura predefinida y no ajustable, lo cual simplifica el proceso de desarrollo de la aplicación pero limita la experiencia de aprendizaje a un escenario específico.

Por el momento, esta aplicación no ha sido evaluada fuera del entorno de desarrollo. Las pruebas realizadas se han limitado a verificar el funcionamiento técnico de la aplicación. En consecuencia, aún no es posible extraer conclusiones sobre la experiencia del usuario, su nivel de satisfacción o el grado de retención del conocimiento adquirido, aspectos fundamentales para validar la utilidad de la aplicación en un contexto real.

### **5.1. Trabajo futuro**

En este proyecto se ha desarrollado la versión experimental de una aplicación de realidad virtual, por lo que aún quedan varios aspectos por mejorar. En este apartado se proponen varias mejoras para continuar con el proyecto.

Con el objetivo de crear un espacio más realista e inmersivo, es necesario mejorar tanto la calidad de los objetos estáticos del aula como la de los materiales con los que se interactúa. Mejorar la fidelidad visual del aula es fundamental para que la experiencia educativa sea motivadora. Del mismo modo, en las aplicaciones formativas de realidad virtual, la precisión y el realismo de los materiales son esenciales para que las habilidades adquiridas en el entorno virtual puedan transferirse de manera efectiva a la práctica real.

Por otra parte, es necesario optimizar las interacciones entre los objetos. Dado que el tiempo de desarrollo de este proyecto es limitado, se ha decidido aplicar soluciones temporales a estos fallos. Por ejemplo, en esta versión de desarrollo, es necesario bloquear las baterías puestas en el soporte para evitar interacciones indeseadas, lo que conlleva a no poder volver a interactuar con

## 5. CONCLUSIONES

---

ellas una vez puestas. Por lo que si una pila se coloca de manera incorrecta por error y se bloquea, será necesario reiniciar la práctica para poder completarla con éxito.

Otra forma de aumentar el realismo sería incorporar más variables a la aplicación. En la versión actual, todas las pilas tienen el mismo voltaje, una representación poco acertada de la realidad. Añadir pequeñas variaciones al voltaje de las pilas, o incluso, añadir alguna pila descargada permitiría crear una experiencia más completa y cercana a la realidad.

A su vez, con el fin de otorgar mayor libertad en la práctica virtual, será necesario permitir al usuario ajustar la potencia del soldador. En esta versión la potencia y el tiempo necesario para soldar son parámetros fijos. Al implementar diferentes niveles de potencia, cada uno contaría con un tiempo óptimo para realizar la soldadura correctamente. Si el tiempo de aplicación es insuficiente, la cinta no quedará soldada y el circuito estará incompleto. En cambio, si el tiempo se excede, la pila y la cinta de níquel se quemarían, siendo necesario reemplazarlas.

Por otro lado, sería interesante añadir distintas configuraciones de batería. La o el usuario dispondría de una selección limitada de configuraciones para realizar. Además convendría permitir distintas formas de ensamblar la misma configuración en vez de limitarlo a una sola.

Otra posible mejora sería incorporar niveles de dificultad a la práctica. El nivel más bajo sería un tutorial completamente guiado; en el, será obligatorio seguir las indicaciones de la aplicación para completar la práctica correctamente. El nivel intermedio sería una práctica más libre, en la que se aconsejará seguir unos pasos aunque permitiendo cometer fallos. El último nivel sería una práctica completamente libre.

Sería de gran utilidad mejorar el sistema de retroalimentación de la aplicación. Por ahora, la aplicación solamente muestra información sobre la forma en la que se han colocado las pilas y el resultado general del ejercicio. Añadir información más detallada sobre los fallos en la colocación de las cintas de níquel o en la soldadura realizada ayudaría a detectar y corregir mejor los fallos y evitar futuros errores.

Para mejorar tanto la experiencia de uso de la aplicación como la experiencia práctica de montaje de la batería, sería conveniente que la aplicación fuera evaluada por profesionales con experiencia en la materia.

Como se ha comentado anteriormente, la función del multímetro es todavía muy limitada. A continuación se explica cómo está programado actualmente la funcionalidad para medir el voltaje de la batería ya montada:

Para evitar la sobrecarga de la aplicación el voltímetro funciona mediante un botón; por lo que cada vez que se quiere realizar una nueva medición es necesario activarlo manualmente mediante el botón “Encender voltímetro” de la interfaz de usuario. Aunque esta solución no resulta la más cómoda ni dinámica, se ha optado por programarlo de esta manera ya que mantener activa la función del voltímetro de forma constante supone un coste computacional demasiado elevado.

Además, cuando se pulsa el botón se deshabilita la funcionalidad de medir el voltaje de las celdas individuales para evitar interferencias. Para volver a activar esa funcionalidad basta con pulsar el botón “Apagar voltímetro”.

En condiciones ideales, el voltímetro funcionaría de la siguiente manera:

Al presionar el botón se activa la corriputina `EncenderVoltímetro()` (ver pseudocódigo 5.1) del script `Voltímetro.cs`, la cual busca la posición de los bornes del voltímetro en la batería. Si pensamos en la batería como una matriz de cuatro filas y diez columnas, en cada columna las pilas están soldadas en paralelo, por lo que en una columna el voltaje es constante, independientemente del punto de medición; por ello solamente es necesario saber en qué columna está conectado cada borne. En el pseudocódigo de la corriputina `EncenderVoltímetro()` se muestra con más claridad cómo se busca la posición de los bornes.

En Unity una función debe ejecutarse completamente antes de devolver el control al motor de juego, lo que implica que todas las acciones dentro de la función deben resolverse en un solo frame. En la función 5.1 el bucle de la línea 13 se repite hasta que los dos bornes del voltímetro se conectan a la batería. El problema es que resulta imposible ejecutar esta acción en un solo frame; si fuera una función convencional la aplicación se bloquearía al quedarse atrapada en ese bucle.

Para evitar este problema se utilizan las corrutinas, que permiten dividir la ejecución de la función en varios frames. De esta forma, la aplicación puede seguir ejecutándose mientras la coroutines espera a que se cumpla una condición. En este caso, la aplicación se pausa al ejecutar el comando **yield** en la línea 47 hasta que se conectan los dos bornes a la batería.

Cuando se han encontrado los dos bornes, se activa el método `MedirVoltaje(int posBorneNegro, int posBorneRojo, bool borneNegroPositivo, bool borneRojoPositivo)` (ver pseudocódigo 5.2), el cual analiza el estado de la batería. Este método tiene como parámetros las posiciones de los bornes (`posBorneNegro` y `posBorneRojo`) y en qué polo se encuentra cada borne (`borneNegroPositivo` y `borneRojoPositivo`). Para comenzar, es necesario saber en qué posiciones están colocadas las cintas de níquel en ambos soportes, así como la configuración de las pilas.

Esta información se obtiene directamente de los otros objetos de la escena. Las posiciones de las cintas de níquel se obtienen de los scripts `CompleteNickelTapes.cs` de ambos soportes. La información sobre las celdas individuales, se obtiene mediante el componente `BatteriesIn-Mount.cs` del soporte inferior. Para asegurarse de que la matriz con la respuesta del usuario no está vacía, se llama a los métodos `FillUserAns()` y `GetUserAns()`. Una vez obtenida la matriz con la posición de las pilas, se analizan los siguientes casos posibles:

#### ■ Batería correctamente montada

Si la batería está correctamente ensamblada, es decir, todas las celdas están en la orientación adecuada, las cintas de níquel están correctamente colocadas y todos los puntos están soldados, se calcula la distancia entre las columnas en las que se encuentran los bornes. Dependiendo de esa distancia se calcula el voltaje mediante la siguiente fórmula:

$$volt = 3,7 \times (|borneNegro - borneRojo| + aux)$$

donde:

- *borneNegro* representa la columna en la que se encuentra conectado el borne negativo del voltímetro
- *borneRojo* representa la columna en la que se encuentra conectado el borne positivo del voltímetro
- *aux* tiene un valor de 0 si ambos bornes se colocan en el mismo polo.  
Tiene valor de 1 si ambos bornes están en polos opuestos.  
Tiene el valor de -1 si uno de los bornes está conectado al polo negativo de la primera columna y el otro está conectado a un polo positivo.

Por ejemplo, en la imagen 5.1 se puede observar que los bornes del voltímetro están situados en las columnas 2 y 5 y que los bornes están conectados a polos opuestos (cabe mencionar que las columnas se empiezan a contar desde el número 1, no desde el 0). Aplicando la fórmula conseguimos el voltaje:

$$volt = 3,7 * (|2 - 5| + 1) \Rightarrow volt = 3,7 * 4 \Rightarrow volt = 14,80$$

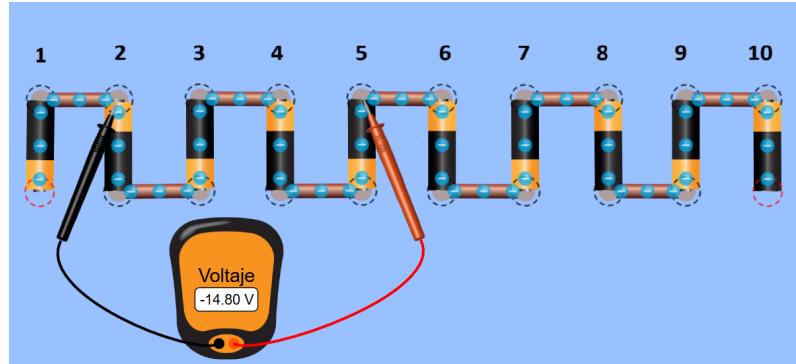


Figura 5.1: Ejemplo funcionamiento voltímetro caso 1

■ **De una a tres pilas incorrectas**

En caso de que en una columna entre una y tres pilas se encuentren colocadas en sentido contrario y las cintas de níquel hayan sido colocadas y soldadas correctamente, se produciría un cortocircuito, por lo que la pantalla del multímetro mostrará el mensaje “corto”. En la imagen 5.2 se puede ver el ejemplo.

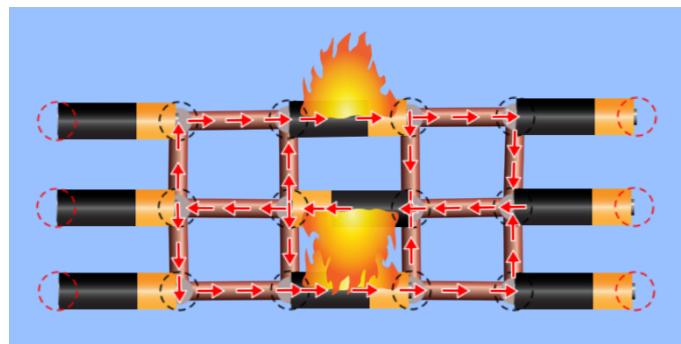


Figura 5.2: Ejemplo cortocircuito I

■ **Coneectar dos celdas por ambos polos**

Si se sueldan al menos dos cintas de níquel en la misma posición por la parte superior e inferior de la batería (como se muestra en la imagen 5.3), se produce un cortocircuito y en la pantalla del multímetro aparece el mensaje “corto”.

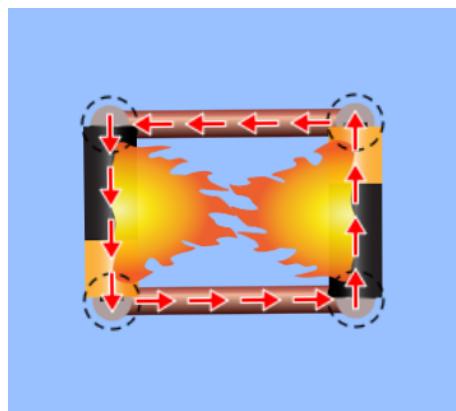
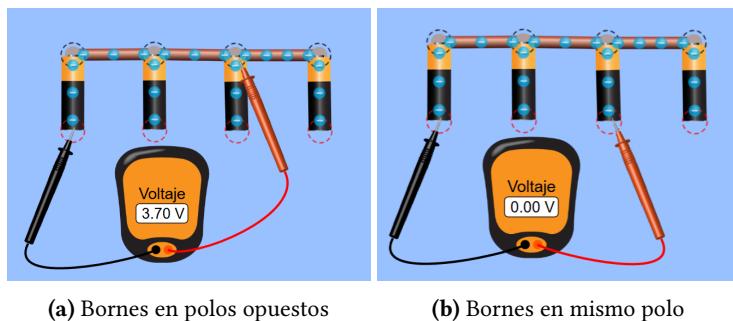


Figura 5.3: Ejemplo cortocircuito II

■ **Soldadura en paralelo**

Si todas las pilas se encuentran colocadas en la misma dirección y las cintas están colocadas

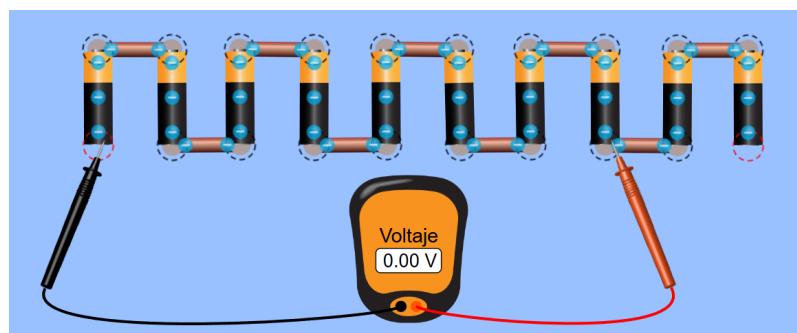
únicamente en uno de los lados de la batería, es decir, solo quedarían unidos los polos positivos o los polos negativos, todas las pilas estarían soldadas en paralelo. En tal caso, el voltaje en cualquier parte de la batería sería de “3.7V” siempre y cuando lo bornes se sitúen en polos distintos (ver imagen 5.4a). Por el contrario, si ambos bornes se conectan al mismo polo, la lectura será de “0.0V” (ver imagen 5.4b).



**Figura 5.4:** Soldadura en paralelo

#### ■ Mismo polo conectado

En los casos en los que la tensión sea nula, por ejemplo cuando las cintas de níquel estén soldadas correctamente pero todas las pilas se encuentren colocadas en la misma dirección y ambos bornes se conecten al mismo polo (ver imagen 5.5), en el voltímetro aparecerá “0.0V”.



**Figura 5.5:** Tensión nula

Cabe señalar que, al tratarse de una funcionalidad aún en desarrollo, la medición del voltaje se limita únicamente a su valor absoluto. Además, en esta fase solo se tratan los casos explicados anteriormente, quedando por incorporar múltiples situaciones posibles.

Resulta conveniente ampliar las capacidades del voltímetro haciendo que la medición se muestre de forma instantánea en la pantalla del dispositivo cada vez que se cambia la posición de los bornes. Además, sería recomendable habilitar los botones integrados en el aparato para sustituir el de la interfaz de usuario. Finalmente, también sería necesario incorporar la función de amperímetro.

EncenderVoltmetro()

```
1 borneNegroEncontrado ← false
2 borneRojoEncontrado ← false
3 borneNegroPositivo ← false
4 borneRojoPositivo ← false
5 posBorneNegro ← 0
6 posBorneRojo ← 0
7 posicion ← 0
8 userAns ← GetUserAns()
9 listaSockets ← GetListaSockets()
10 if userAns vacia then
11     return
12 else
13     while borneNegroEncontrado o borneRojoEncontrado == false
14         for i = 0; i < listaSockets.size()
15             socket←listaSockets[i]
16             posicion++
17             if socket null then
18                 continue
19             else
20                 if socket no tiene pila then
21                     continue
22                 else
23                     pila←socket.GetPila()
24                     poloPositivo←pila.GetPositivo()
25                     poloNegativo←pila.GetNegativo()
26                     if poloPositivo en contacto con borne negro then
27                         borneNegroEncontrado← true
28                         borneNegroPositivo← true
29                         posBorneNegro←posicion mod 10
30                     else if poloPositivo en contacto con borne rojo then
31                         borneRojoEncontrado← true
32                         borneRojoPositivo← true
33                         posBorneRojo←posicion mod 10
34                     fi
35                     if poloNegativo en contacto con borne negro then
36                         borneNegroEncontrado← true
37                         borneNegroPositivo← false
38                         posBorneNegro←posicion mod 10
39                     else if poloNegativo en contacto con borne rojo then
40                         borneRojoEncontrado← true
41                         borneRojoPositivo← false
42                         posBorneRojo←posicion mod 10
43                     fi
44                     fi
45                     if borneNegroEncontrado y borneRojoEncontrado == true then
46                         break
47                     rof
48                     yield return
49                     done
50                     MedirVoltaje(posBorneNegro, posBorneRojo, borneNegroPositivo, borneRojoPositivo)
51 fi
```

---

MedirVoltate(int posBorneNegro, int posBorneRojo, bool borneNegroPositivo, bool borneRojoPositivo)

---

```

1 aux ← 0
2 diferencia ← |posBorneNegro – posBorneRojo|
3 posCintasSuperiores ← soporteSuperior.GetPosicionCintas()
4 posCintasInferiores ← soporteInferior.GetPosicionCintas()
5 userAns ← soporteInferior.GetUserAns()
6 if practica correcta then
7   if (borneNegroPositivo y borneRojoPositivo = true) o (borneNegroPositivo y borne-
     RojoPositivo = false) y diferencia ≠ 0
8     aux ← 1
9   else if (borneNegroPositivo = false y posBorneNegro = 1 y borneRojoPositivo =
     = true) o (borneRojoPositivo = false y posBorneRojo = 1 y borneNegroPositivo =
     = true)
10    aux ← -1
11  else if (borneNegroPositivo y borneRojoPositivo = true) o (borneNegroPositivo y
     borneRojoPositivo = false) y diferencia = 0
12    voltímetro ← 0,0
13    return
14  fi
15  voltímetro ← 3,7 * |diferencia + aux|
16  return
17 fi
18 if posCintasSuperiores = posCintasInferiores then
19   voltímetro ← “corto”
20 fi
21 if cintas de níquel correctas then
22   if de una a tres pilas incorrectas en la misma columna then
23     voltímetro ← “corto”
24   fi
25   if todas las pilas en la misma orientación then
26     voltímetro ← 0,0
27   fi
28   return
29 fi
30 if todas las pilas en la misma orientación y todas las cintas superiores e inferiores then
31   if (borneNegroPositivo y borneRojoPositivo = true) o (borneNegroPositivo y borne-
     RojoPositivo = false) then
32     voltímetro ← 0,0
33   else
34     voltímetro ← 3,7
35   fi
36 fi
```

---

**Algoritmo 5.2:** Medir voltaje



---

# Apéndice

---

## Interacciones entre objetos y layers mask

En esta tabla se puede observar con que layer mask pueden interactuar los objetos que contienen algún componente “*interactor*”, ya sea de tipo grab o socket.

| Objeto \ Layer mask   | Default | Battery | Mount | Nickel |
|-----------------------|---------|---------|-------|--------|
| Controladores         | X       | X       | X     | X      |
| Sockets pilas         |         | X       |       |        |
| Socket soporte        |         |         | X     |        |
| Sockets cintas níquel |         |         |       | X      |

Tabla 1: Interacciones entre objetos y layer mask

## Link carpeta Google Drive

[carpeta Google Drive](#)

## Socket dinámico

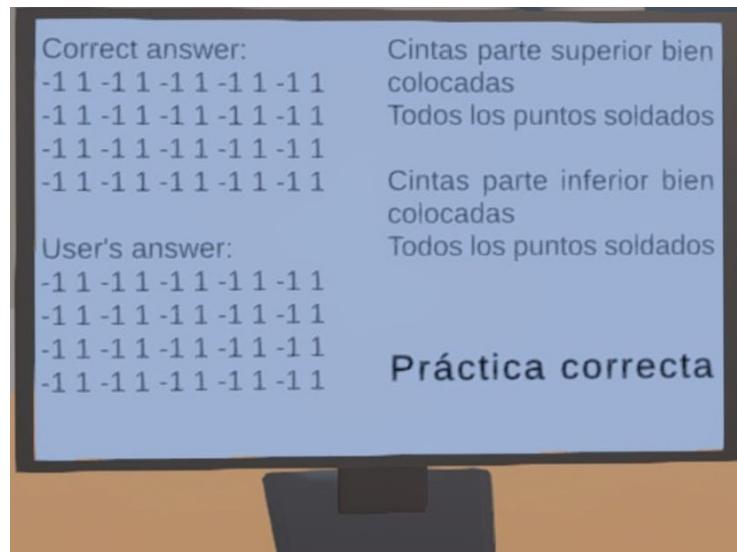
Los componentes *XRSocketInteractor* permiten fijar objetos de la escena a otros. Por defecto, un objeto solamente puede anclarse en una posición, la cual puede ser modificada. Para habilitar varias posiciones predefinidas se ha modificado la función encargada de colocar el objeto. La función *GetAttachTransform()* modificada obtiene la orientación del objeto que se va a fijar al socket, en este caso la pila (`int ori = battery.GetOrientation();`), y ajusta la posición a una de las posiciones permitidas. Dado que la orientación puede adoptar el valor 1 o -1, si se aproxima a 1 el objeto se alinea con el eje *posUp*, de modo que la pila queda fijada con el polo positivo hacia arriba; en caso contrario, si la orientación se aproxima a -1, se alinea con el eje *negUp*, quedando el polo negativo hacia arriba (ver código 1).

## Mensaje retroalimentación

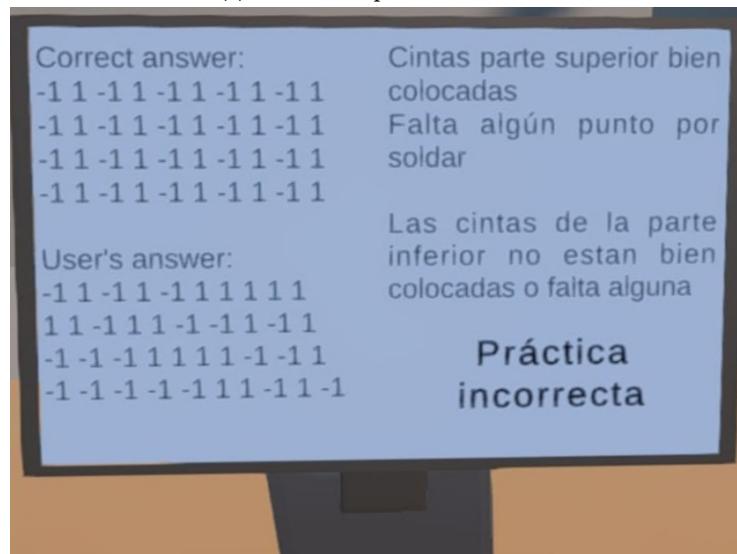
Ver imágenes [1a](#) y [1b](#).

**Listing 1:** Implementación socket dinámico

```
1  public class DynamicSocket : XRSocketInteractor
2  {
3      public Transform posUp;
4      public Transform negUp;
5
6      public override Transform GetAttachTransform(IXRInteractable
7          interactable)
8      {
9          GameObject obj = interactable.transform.gameObject;
10         var battery = obj.GetComponent<ObjectOrientation>();
11         if (battery != null)
12         {
13             int ori = battery.GetOrientation();
14             return (ori == 1) ? posUp : negUp;
15         }
16         return base.GetAttachTransform(interactable);
17     }
18 }
```



(a) Comentario práctica correcta



(b) Comentario práctica incorrecta

**Figura 1:** Retroalimentación de la práctica



---

# Bibliografía

---

- [1] Michael A Gigante. Virtual reality: definitions, history and applications. In *Virtual reality systems*, pages 3–14. Elsevier, 1993. Ver página [1](#).
- [2] Emilio R Escartín. La realidad virtual, una tecnología educativa a nuestro alcance. *Pixel-Bit. Revista de Medios y Educación*, (15):5–21, 2000. Ver página [1](#).
- [3] G.C. Burdea and P. Coiffet. *Virtual Reality Technology*. IEEE Press. Wiley, 2003. Ver página [1](#).
- [4] Javier Alfredo Caballero-Garriazo, Jaime Rolando Rojas-Huacanca, Angélica Sánchez-Castro, and Alberto Frank Lázaro-Aguirre. Revisión sistemática sobre la aplicación de la realidad virtual en la educación universitaria. *Revista Electrónica Educare*, 27(3):463–480, 2023. Ver páginas [2](#), [14](#).
- [5] Jaiden A di Lanzo, Andrew Valentine, Ferdous Sohel, Angie YT Yapp, Kudakwashe C Muparadzi, and Merkorios Abdelmalek. A review of the uses of virtual reality in engineering education. *Computer Applications in Engineering Education*, 28(3):748–763, 2020. Ver páginas [2](#), [13](#), and [14](#).
- [6] Jacobo Roda-Segarra, Santiago Mengual Andrés, Rosabel Martínez-Roig, et al. Using virtual reality in education: a bibliometric analysis. 2022. Ver página [13](#).
- [7] Ahmad A Mazhar and Mustafa M Al Rifaee. A systematic review of the use of virtual reality in education. In *2023 International Conference on information technology (ICIT)*, pages 422–427. IEEE, 2023. Ver página [13](#).
- [8] Belén Guerrero Cuevas and Luis Valero Aguayo. Efectos secundarios tras el uso de realidad virtual inmersiva en un videojuego. *International journal of psychology and psychological therapy*, 13(2):163–178, 2013. Ver página [13](#).
- [9] S AlAwadhi, N AlHabib, D Murad, F AlDeei, M AlHouti, T Beyrouthy, and S Al-Kork. Virtual reality application for interactive and informative learning. In *2017 2nd international conference on Bio-engineering for Smart Technologies (BioSMART)*, pages 1–4. IEEE, 2017. Ver página [14](#).
- [10] Youssef Asham, Mohamed H Bakr, and Ali Emadi. Applications of augmented and virtual reality in electrical engineering education: A review. *IEEE Access*, 11:134717–134738, 2023. Ver página [14](#).
- [11] Zhenjun Jiang, Yang Yang, Qingshu Yuan, Pengfei Leng, Yanyan Liu, and Zhigeng Pan. Virtual reality training environment for electric systems. In *2021 IEEE 7th International Conference on Virtual Reality (ICVR)*, pages 314–318. IEEE, 2021. Ver página [14](#).
- [12] Xun Xu, Lei Shen, Shundai Li, and Siqi Wang. Research on 10kv line breaker check training system based on virtual reality. In *2019 International Conference on Smart Grid and Electrical Automation (ICSGEA)*, pages 18–21. IEEE, 2019. Ver página [14](#).
- [13] Zoltán Kvaszniczka. Teaching electrical machines in a 3d virtual space. In *2017 8th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, pages 000385–000388, 2017. Ver página [14](#).
- [14] Batool Hasan, Yara Al-Quorashy, Shahad Al-Mousa, Yousef Al-Sahhaf, and Mohammed El-Abd. V-lab—the virtual electric machines laboratory. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pages 72–77. IEEE, 2020. Ver página [14](#).

- [15] Alejandro A Franco, Emilie Loup-Escande, Goran Loiseaux, Jean-Noël Chotard, Diana Zapata-Dominguez, Jan Ciger, Aubin Leclere, Lucie Denisart, and Romain Lelong. From battery manufacturing to smart grids: Towards a metaverse for the energy sciences. *Batteries & Supercaps*, 6(1):e202200369, 2023. Ver página [15](#).
- [16] Soorya Saravanan, Utkarsh Vijay, Sophie Tran, Maris Minna Mathew, Desislava Yordanova Apostolova, Inaki Gandarias, Aubin Leclere, Romain Lelong, and Alejandro A Franco. Metaverse for battery manufacturing: Connecting students from different geographical locations to solve battery manufacturing problems in the virtual reality space. *Batteries & Supercaps*, page 2500098, 2025. Ver página [15](#).