ISSN-e: 2444-2887

Herramienta para la resolución y simulación de problemas físicos sobre choques en una y dos dimensiones con fines de uso didáctico e interactivo

María del Pilar Cuenca Marcano (autora de contacto)

Profesora de la Universidad Metropolitana (Caracas, Venezuela)
pcuenca@unimet.edu.ve | https://orcid.org/0000-0002-1029-1919

Salomón Mizrachi Cohén

Profesor de la Universidad Metropolitana (Caracas, Venezuela)
smizrachi@unimet.edu.ve | https://orcid.org/0009-0004-8895-8025

Gabriela Banezca Luces

Graduada en Ingeniería de Sistemas por la Universidad Metropolitana (Caracas, Venezuela)
gbanezca.dev@gmail.com | https://orcid.org/0009-0001-2232-7261

José Villegas Vera

Graduado en Ingeniería de Sistemas por la Universidad Metropolitana (Caracas, Venezuela)
joseivillegasvera@gmail.com | https://orcid.org/0009-0009-8868-5537

Extracto

El estudio de la física presenta desafíos para estudiantes y docentes (hombres y mujeres). Implica comprender y predecir el comportamiento natural a través de leyes asociadas a la experimentación. Las simulaciones ofrecen oportunidades para el aprendizaje práctico y la exploración de conceptos abstractos al permitir crear situaciones ideales, manipular variables y observar su impacto en fenómenos concretos. El propósito de este trabajo es crear un laboratorio de física virtual sobre colisiones entre partículas con la finalidad de ayudar a los estudiantes de los primeros cursos de Física universitaria a comprender estos conceptos a través de representaciones visuales y el proceso matemático asociado. Se evalúa la viabilidad y eficacia de utilizar un motor de videojuegos para el desarrollo rápido de un simulador que represente con precisión distintos tipos de colisiones, aprovechando que el uso de colisiones es parte fundamental del motor.

El proyecto consiste en una investigación aplicada, orientada a resolver problemas prácticos mediante la creación de un *software* específico. Se inició con una revisión bibliográfica, considerando aspectos pedagógicos y técnicos, y continuó con el desarrollo del *software*, empleando Scrum para crear un producto mínimo viable. Se utilizó el motor de juegos Unity. El simulador permite generar problemas personalizados y consta de cuatro módulos que muestran el comportamiento de las partículas antes y después de la colisión, tanto gráfica como matemática y vectorialmente.

Palabras clave: colisiones; simulador; física; enseñanza; Unity; aprendizaje; Scrum.

Recibido: 15-03-2024 | Aceptado: 19-12-2024 | Publicado (anticipado): 24-02-2025

Cómo citar: Cuenca Marcano, M.ª P., Mizrachi Cohén, S., Banezca Luces, G. y Villegas Vera, J. (2025). Herramienta para la resolución y simulación de problemas físicos sobre choques en una y dos dimensiones con fines de uso didáctico e interactivo. *Tecnología, Ciencia y Educación*, *31*, 169-193. https://doi.org/10.51302/tce.2025.21499

ISSN-e: 2444-2887

Software for solving and simulating physical problems about collisions in one and two dimensions for educational and interactive use purposes

María del Pilar Cuenca Marcano (corresponding author)

Professor at the Universidad Metropolitana (Caracas, Venezuela)
pcuenca@unimet.edu.ve | https://orcid.org/0000-0002-1029-1919

Salomón Mizrachi Cohén

Professor at the Universidad Metropolitana (Caracas, Venezuela)
smizrachi@unimet.edu.ve | https://orcid.org/0009-0004-8895-8025

Gabriela Banezca Luces

Graduated in Systems Engineering from the Universidad Metropolitana (Caracas, Venezuela)
gbanezca.dev@gmail.com | https://orcid.org/0009-0001-2232-7261

José Villegas Vera

Graduated in Systems Engineering from the Universidad Metropolitana (Caracas, Venezuela)
joseivillegasvera@gmail.com | https://orcid.org/0009-0009-8868-5537

Abstract

The study of physics presents challenges for students and teachers (men and women). It involves understanding and predicting natural behavior through laws associated with experimentation. Simulations provide opportunities for hands-on learning and exploration of abstract concepts by allowing the creation of ideal situations, manipulating variables, and observing their impact on concrete phenomena. The purpose of this work is to create a virtual physics laboratory on particle collisions to help students in the first courses of university Physics to understand these concepts through visual representations and the associated mathematical process. The feasibility and effectiveness of using a video game engine for the rapid development of a simulator that accurately represents different types of collisions is evaluated, taking advantage of the fact that the use of collisions is a fundamental part of these.

The project is applied research, oriented to solve practical problems through the creation of specific software. It started with a bibliographic review, considering pedagogical and technical aspects, followed by the development of the software using Scrum to create a minimum viable product. The Unity game engine was used. The simulator allows the generation of customized problems and consists of four modules that show the behavior of the particles before, during and after the collision, graphically, mathematically and vectorially.

Keywords: collisions; simulator; physics; teaching; Unity; learning; Scrum.

Received: 15-03-2024 | Accepted: 19-12-2024 | Published (preview): 24-02-2025

Citation: Cuenca Marcano, M.ª P., Mizrachi Cohén, S., Banezca Luces, G. and Villegas Vera, J. (2025). Software for solving and simulating physical problems about collisions in one and two dimensions for educational and interactive use purposes. *Tecnología, Ciencia y Educación*, *31*, 169-193. https://doi.org/10.51302/tce.2025.21499

Sumario

1. Introducción
2. Objetivo
    2.1. Preguntas de investigación
    2.2. Alcance de la investigación
3. Método
4. Resultados
    4.1. Requerimientos
    4.2. Selección de las herramientas de desarrollo
    4.3. Desarrollo del simulador
        4.3.1. Modelador
        4.3.2. Simulador
        4.3.3. Solucionador
        4.3.4. Diagrama
    4.4. Validación del simulador
    4.5. Distribución web
5. Discusión
6. Conclusiones

Nota: con la finalidad de distribuir la aplicación de este proyecto académico entre estudiantes y profesores de forma pública, se tomó la decisión de subir dicha aplicación a Itch.io, una plataforma web especializada en la distribución de contenido interactivo. Repositorio: joseivv.itch.io/simulador-de-colisione

1. Introducción

Los primeros cursos de Física constituyen un reto, tanto para el que los sigue como para los docentes encargados de impartir esas asignaturas. El quid del asunto está en que para aprender física no basta con la simple memorización de hechos ni con la aplicación de fórmulas. De manera abreviada, la física trata de explicar y predecir el comportamiento de la naturaleza mediante leyes. Las leyes están definitivamente asociadas a la experimentación. Para enseñar física es entonces necesario explicar sus leyes y mostrar su aplicación mediante diversos experimentos, pero esto no es siempre posible o conveniente, incluso para el caso de escenarios cotidianos. Empleando los problemas como «experimentos», el estudiante puede, a su vez, desarrollar el razonamiento lógico y la habilidad indispensable para entender la física (Elizondo Treviño, 2013).

Los simuladores educativos son útiles para investigar sistemas físicos de forma controlada y para el aprendizaje de conceptos abstractos, no solo porque permiten visualizar fenómenos que de otra forma podrían ser inaccesibles, sino también porque facilitan un aprendizaje de los conceptos y principios basado en la investigación de los estudiantes y apoyado en el uso de procedimientos propios del trabajo científico. Los simuladores educativos son especialmente útiles para investigar sistemas físicos, ya que permiten a los alumnos visualizar los fenómenos e interactuar con ellos, lo que conduce a mejorar la interpretación de los conceptos físicos, que pueden resultar difíciles para los estudiantes (Barragán Suescún, 2020; Kortemeyer, 2023; Makamu y Ramnarain, 2022; Pérez-Higuera *et al.*, 2020). Estos proporcionan herramientas para que los alumnos pongan en práctica las teorías aprendidas por medio de simulaciones aplicadas a problemas físicos complejos del mundo real (Barragán Suescún, 2020; Carangui Cárdenas *et al.*, 2017). La simulación es una herramienta eficaz para la enseñanza de la física, ya que mejora la interacción y comprensión de sistemas complejos. Sin embargo, es crucial utilizar una metodología apropiada para obtener resultados satisfactorios, en particular debido a que los modelos de simulación representan realidades simplificadas, por lo que es importante ajustar adecuadamente los parámetros de simulación para lograr un análisis adecuado (Boettcher y Behr, 2021; Elizondo Treviño, 2013; Prado Martínez, 2008).

La integración de herramientas informáticas en la educación puede tener múltiples beneficios. Estas herramientas pueden aumentar la participación activa de los estudiantes, proporcionarles una retroalimentación personalizada y ayudarles a identificar áreas de mejora. Además, permiten medir no solo los conocimientos adquiridos, sino también el proceso de aprendizaje, lo que contribuye a una evaluación más integral. Los entornos virtuales se utilizan en educación para mejorar el proceso de enseñanza-aprendizaje y para desarrollar

habilidades interpersonales, facilitando una comunicación e interacción más eficiente. Sin embargo, su implementación puede generar resistencia al cambio (Isela Aguilar Vargas y Otuyemi Rondero, 2020; López-Mera *et al.*, 2021; Otto *et al.*, 2023). Por su parte, las metodologías ágiles son útiles en la educación en entornos VUCA (volátiles, inciertos, complejos y ambiguos), ya que fomentan la autonomía y autorregulación de los estudiantes, la colaboración y el aprendizaje basado en proyectos; en particular, si estos proyectos están enfocados en desafíos reales y complejos (Monfort Salvador y E-Martín, 2022; Salimzyanova, 2022; Torres-Blasco y Pérez-Garcias, 2023).

En los videojuegos, las colisiones entre objetos tienen una relevancia primordial. Gracias a sus motores, se logran representaciones visuales precisas de diferentes tipos de colisiones. Utilizar esos motores para desarrollar simulaciones educativas de colisiones en una y dos dimensiones es una opción viable y eficaz que permite satisfacer necesidades específicas de la enseñanza de la física. Las colisiones entre objetos se rigen por el principio de conservación del momento lineal, un concepto fundamental en física que, por su naturaleza vectorial, suele ser complejo para los estudiantes de Física elemental. Los motores de física de los videojuegos implementan este principio para representar los choques de manera realista. Aprovechando las capacidades de los videojuegos, se diseñó una simulación para visualizar y comprender el principio de conservación del momento lineal. La simulación permite concebir de forma atractiva el fenómeno de las colisiones; manipular variables como la masa, la velocidad, el momento y el tipo de colisión; experimentar con diferentes escenarios y condiciones, y ofrecer soluciones numéricas que permiten comparar los resultados de la simulación con las predicciones teóricas. Una ventaja importante de esta herramienta educativa es que permite a los estudiantes avanzar a su propio ritmo; además, promueve la exploración de diferentes condiciones y estimula el interés por la física.

2. Objetivo

El objetivo general (OG) de este estudio de investigación es el siguiente:

OG. Implementar una herramienta interactiva que permita a profesores y alumnos resolver y simular problemas de colisión unidimensionales o bidimensionales.

El *software* debe permitir la personalización de los problemas. La personalización de los problemas es especialmente importante en el *software* educativo, donde los alumnos tienen distintos niveles de competencia y estilos de aprendizaje. Al permitir a los profesores crear problemas adaptados a las necesidades del alumno, el *software* educativo puede proporcionar un enfoque más individualizado del aprendizaje.

2.1. Preguntas de investigación

Se plantean las siguientes preguntas de investigación (P):

P1. ¿Es viable y eficaz utilizar un motor de videojuegos para el desarrollo rápido de un simulador de colisiones en una y dos dimensiones que sea capaz de apoyar tanto a estudiantes como a profesores de Física de la Universidad Metropolitana en el proceso de aprendizaje y que permita la visualización, resolución y personalización de los problemas?

P2. ¿Cómo se pueden adaptar y personalizar las funcionalidades del motor de videojuegos para satisfacer las necesidades específicas de la enseñanza de la física de colisiones en una y dos dimensiones?

2.2. Alcance de la investigación

El *software* permitirá visualizar en 3D simulaciones de choques unidimensionales y bidimensionales, lo que facilitará una mayor interacción con el usuario. Los choques que se van a simular seguirán la metodología propuesta por el Departamento de Física de la Universidad Metropolitana (Caracas, Venezuela). El público objetivo de este *software* son estudiantes y profesores de la asignatura de Física.

La estrategia más eficiente para el desarrollo rápido de un simulador de colisiones que integre funcionalidades específicas y valiosas para mejorar la enseñanza de la física en los primeros cursos universitarios implica aprovechar las ventajas gráficas y físicas de los motores de videojuegos. Al hacerlo, se logra crear una herramienta interactiva y visualmente atractiva que resulta útil para estudiantes y profesores, sin necesariamente validar su impacto directo en el proceso de aprendizaje.

3. Método

El trabajo consiste en una investigación aplicada que busca la resolución de problemas prácticos a través de la aplicación de conocimientos y tecnologías disponibles, destinada a crear un *software* capaz de resolver un problema específico (Hernández Sampieri *et al.*, 2014).

Se inició con una fase de revisión bibliográfica de conceptos relevantes con el objetivo de generar un marco referencial para el abordaje del simulador, considerando los aspectos pedagógicos y técnicos para el desarrollo del *software*. Se examinaron simuladores existentes, las arquitecturas de aplicaciones similares, patrones de diseño y la teoría fundamental de la física sobre colisiones y sus implicaciones prácticas. A continuación, se inició la fase de desarrollo del *software* con el levantamiento de los requerimientos del Departamento de Física. Seguidamente, se seleccionó la plataforma para la elaboración del simulador y se

diseñó la arquitectura que permitiera satisfacer las funciones principales de mapeo, análisis, resolución y visualización de problemas de choque.

Para el desarrollo del *software* se utilizó el *framework*[1] ágil Scrum, ampliamente utilizado debido a sus beneficios en productividad, calidad y satisfacción del cliente (Scott y Campo, 2023; Torres-Blasco y Pérez-Garcias, 2023). Scrum es un marco de gestión de proyectos que utiliza un enfoque empírico y un proceso iterativo e incremental (Schwaber y Sutherland, 2020). Tiene ciclos iterativos llamados *sprints*, en los que equipos autogestionados completan tareas previamente acordadas en periodos predeterminados. Se realizan reuniones diarias de planificación, revisión y retrospección que fomentan la transparencia, la responsabilidad y la mejora continua. Esto permite a los equipos adaptarse rápidamente a cambios en los requisitos del proyecto (Chandra *et al.*, 2021; Rodrigues de Oliveira *et al.*, 2023). El proceso de desarrollo del simulador siguió una metodología incremental con el objetivo de satisfacer los requisitos de los profesores para generar un producto mínimo viable (*minimum viable product* [MVP]). Un MVP es un producto con un número mínimo pero suficiente de funciones para que los usuarios finales lo utilicen y puedan evaluar el producto en términos de su funcionalidad y facilidad de uso (Alonso *et al.*, 2023; Umbreen *et al.*, 2022). En este proyecto se realizaron *sprints* de dos semanas y se contó con un profesor del Departamento de Física en el rol de propietario del producto. Los métodos ágiles como Scrum mejoran la comunicación, colaboración y productividad en el desarrollo del *software*. Permiten reducir errores, riesgos y costos al facilitar la resolución de problemas complejos y adaptarse rápidamente a los cambios. Además, mejoran la eficiencia y los procesos de toma de decisiones. La planificación de *sprints* también mejora la calidad general del desarrollo, incluyendo entornos virtuales (Chandra *et al.*, 2021; González-González *et al.*, 2015; Monfort Salvador y E-Martín, 2022; Rodrigues de Oliveira *et al.*, 2023; Torres-Blasco y Pérez-Garcias, 2023).

El Departamento de Física se encargó de establecer y supervisar continuamente los requisitos de desarrollo del *software*, utilizando historias de usuario para crear y priorizar el *product backlog*. El enfoque MVP permitió la detección y corrección temprana de errores, mejoras incrementales y una estrecha participación de las partes interesadas, garantizando la generación de valor y la adaptación a las demandas académicas. El trabajo en equipo aportó diversidad de perspectivas, enriqueciendo la definición del producto (Monfort Salvador y E-Martín, 2022). Con el fin de satisfacer las necesidades gráficas y de representación de las leyes de la física en el proyecto, se decidió emplear un motor de videojuegos. Los motores de videojuegos son plataformas de *software* que proporcionan un conjunto de herramientas de programación para el diseño y la creación de juegos. Estos motores integran sistemas de cálculo de física en dos y tres dimensiones, que permiten simular leyes físicas y gestionar colisiones de manera realista. El entorno de programación de videojuegos povee un ambiente versátil e integral lo suficientemente robusto como para permitir el desarrollo de las funcionalidades requeridas por el Departamento de Física.

[1] Conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar.

Para este proyecto en particular, se evaluaron varias herramientas de desarrollo, considerando cinco criterios: capacidad gráfica; capacidad de emular o integrar un motor de física en 3D; compatibilidad con distintas plataformas; facilidad de uso, evaluada como contar con una curva de aprendizaje corta; y disponibilidad de documentación completa y actualizada. Adicionalmente, se estableció la necesidad de acceso gratuito a la tecnología. Los patrones de diseño de *software* se utilizaron para definir la arquitectura del proyecto, asegurando la implementación de buenas prácticas en su desarrollo (Hernández-Paez *et al.*, 2018; Jiménez-Torres *et al.*, 2014). El uso de patrones de diseño para videojuegos mejora la arquitectura y funcionalidad del *game engine*. Su implementación optimiza la interacción y el funcionamiento de los elementos del juego, lo que contribuye al rendimiento del *software*. Además, propicia un proceso de desarrollo eficiente, facilita el mantenimiento y la escalabilidad y mejora la comprensión y el mantenimiento del *software* (Chover *et al.*, 2020).

Al utilizar la metodología Scrum en la fase de desarrollo del simulador, se logró, para cada iteración, planificar y priorizar los requisitos especificados en historias de usuario. Durante cada *sprint* se siguió un ciclo de planificación, desarrollo, revisión y retrospectiva, lo que fomenta una comunicación frecuente entre el equipo de desarrollo, el dueño del producto y las partes interesadas, conformadas por los profesores del Departamento de Física. Esto permitió adaptar el simulador de acuerdo con las expectativas de los profesores y realizar ajustes en cada iteración, logrando una entrega de valor continua. Al final de cada *sprint*, se obtuvo un incremento funcional del simulador, lo que facilitó la retroalimentación temprana y la validación del simulador.

Las pruebas de *software* se integran en el proceso de desarrollo de manera iterativa e incremental, lo que permite identificar y corregir problemas y refinar los requerimientos a medida que evoluciona el proyecto. Cada incremento de *software* debe ser probado para garantizar su correcto funcionamiento y cumplimiento de los criterios de aceptación. Se realizaron pruebas funcionales, de integración y aceptación. Las pruebas de aceptación se llevaron a cabo para validar que el simulador de colisiones cumplía con los requisitos definidos por el propietario del producto. Estas pruebas se basaron en escenarios de uso reales. Al final de cada *sprint*, se realizó una retrospectiva para evaluar el proceso de desarrollo y las pruebas ejecutadas. Se identificaron oportunidades de mejora y se implementaron acciones correctivas para refinar la especificación de las historias de usuario en el *backlog* y el proceso de desarrollo en los siguientes *sprints*.

El MVP del simulador de colisiones desarrollado satisfizo los requerimientos funcionales de los profesores del Departamento de Física en términos de funcionalidad para el proceso educativo y de las características técnicas del simulador. El desarrollo incremental permitió obtener retroalimentación temprana de los usuarios, por lo que se adaptaron las características funcionales a medida que se desarrollaba y evaluaba el simulador. Esta validación de los requerimientos funcionales fue esencial para garantizar que el simulador cumplía con los objetivos educativos y las necesidades específicas de los profesores del Departamento de Física de la Universidad Metropolitana.

4. Resultados

La adaptación y personalización de motores de videojuegos para el desarrollo de simuladores para la enseñanza de la física de colisiones implica un entendimiento profundo de las herramientas de simulación física que estos ofrecen y cómo estas pueden ser ajustadas para cumplir con objetivos educativos específicos. Con la correcta implementación, los motores de videojuegos tienen el potencial de hacer el aprendizaje más interactivo, atractivo y efectivo.

4.1. Requerimientos

Se estableció como MVP un producto con las siguientes características:

- **Flexibilidad.** El simulador debe ser capaz de ajustarse a diferentes niveles de enseñanza y contenidos curriculares, permitiendo a los profesores personalizar y ajustar los escenarios de colisiones según las necesidades específicas de sus estudiantes.
- **Interactividad.** El simulador debe ofrecer una experiencia interactiva que permita a los estudiantes explorar y experimentar con diferentes parámetros y variables relacionadas con las colisiones. Debe proporcionar retroalimentación inmediata y visual para ayudar a los estudiantes a comprender los conceptos de manera efectiva.
- **Variedad de escenarios.** El simulador debe ofrecer una amplia gama de escenarios de colisiones que cubran diferentes situaciones y conceptos de la física de colisiones.
- **Análisis de datos.** El simulador debe permitir a los estudiantes recopilar y analizar datos relacionados con las colisiones, como velocidad, masa y energía. Debe proporcionar herramientas y gráficos que faciliten la interpretación de los resultados y la comprensión de los conceptos subyacentes.
- **Accesibilidad.** El simulador debe cumplir con los estándares y pautas de accesibilidad web, asegurando que sea compatible con diferentes navegadores y dispositivos. Esto permitirá a los estudiantes acceder al simulador desde cualquier lugar y en cualquier momento, facilitando su uso como herramienta de enseñanza tanto en el aula como fuera de ella.

Para satisfacer estos requerimientos, el *software* debía implementar funciones tanto de mapeado como de cálculo para poder trabajar eficazmente. El mapeado se refiere al dibujo de cuerpos geométricos en 3D que representan los cuerpos físicos implicados en la simulación de colisiones. Mientras, la identificación de los datos proporcionados y la selección del modelo correspondiente para resolver paso a paso el problema físico planteado por el

usuario se denomina «cálculo». El simulador debe permitir cargar cualquier problema de colisión, lo que significa que los profesores deben poder introducir y personalizar distintos escenarios para que los alumnos los analicen y resuelvan. Esto incluye la posibilidad de ajustar masas, velocidades iniciales, coeficientes de restitución y otros parámetros relevantes para cada problema específico. Al permitir la carga de cualquier problema de colisión, el simulador proporciona a los profesores la flexibilidad necesaria para adaptar el contenido de la física de colisiones a sus necesidades y objetivos docentes.

4.2. Selección de las herramientas de desarrollo

Se evaluaron tres herramientas de desarrollo del tipo *game engine*, en función de los cinco criterios presentados en el apartado 3. Las herramientas consideradas fueron Unity, Unreal Engine y O3DE.

Las evaluaciones se llevaron a cabo considerando criterios fundamentales para un simulador de choque de partículas. La calidad gráfica desempeñó un papel crucial al facilitar la visualización realista de las partículas y sus colisiones. De manera similar, la eficacia del motor de física resultó esencial para simular con precisión el movimiento y la interacción de las partículas.

La compatibilidad multiplataforma fue importante para garantizar la ejecución exitosa del simulador en diversos sistemas operativos. Además, la documentación detallada y una curva de aprendizaje accesible y realista a los tiempos de realización de este proyecto contribuyeron significativamente al aprendizaje efectivo y a la utilización eficiente de la herramienta por parte del equipo de desarrollo (Epic Games, 2024; O3DE, 2024; Unity Technologies, 2025a). Cabe destacar que, en el ámbito de la documentación, la existencia de una comunidad activa de usuarios añadió un valor significativo. La posibilidad de plantear y resolver dudas en este entorno colaborativo mejoró la experiencia de desarrollo. Por último, la consideración del bajo costo o gratuidad emergió como un factor determinante para mantener los costos del proyecto en niveles gestionables.

En la matriz de valoración empleada, se asignaron calificaciones en una escala del 0 al 1, donde el valor más alto, es decir, 1, se designó como «excelente», reflejando un rendimiento óptimo. En contraste, el valor más bajo, 0, se asignó para indicar el rendimiento «menos favorable».

Es crucial destacar que esta escala no solo cuantifica el desempeño, sino que también se adhiere a una ponderación específica. Este enfoque busca otorgar mayor relevancia a los aspectos considerados de mayor importancia dentro del contexto de la matriz de valoración. Como se muestra en el cuadro 1, se eligió una ponderación porcentual, distribuyendo el 100 % de la importancia entre los criterios elegidos, a manera de obtener una ponderación porcentual.

Cuadro 1. **Matriz de valoración**

Criterios	Peso
Buena capacidad gráfica	15 %
Motor de físicas	40 %
Multiplataforma	5 %
Documentación y curva de aprendizaje	30 %
Costo bajo o gratuito	10 %

Fuente: elaboración propia.

Cuadro 2. **Matriz de valoración**

	Buena capacidad gráfica	Motor de físicas	Multiplata-forma	Documenta-ción/Curva aprendizaje	Costo bajo o gratuito
Unity	0,8	1	1	0,8	1
Unreal Engine	1	1	1	0,6	0,6
O3DE	0,6	0,8	0,8	0,6	0,8

Fuente: elaboración propia.

La valoración de cada una de las herramientas viene dada por el valor alcanzado en cada criterio (véase cuadro 2) y la ponderación relativa del criterio, como se presenta a continuación:

- **Unity:** (15 × 0,8) + (40 × 1) + (5 × 1) + (30 × 0,8) + (10 × 1) = 91 %.
- **Unreal Engine:** (15 × 1) + (40 × 1) + (5 × 1) + (30 × 0,6) + (10 × 0,6) = 84 %.
- **O3DE:** (15 × 0,6) + (40 × 0,8) + (5 × 0,8) + (30 × 0,6) + (10 × 0,8) = 71 %.

Al tener el valor según la matriz de valoración más cercano al 100 %, se seleccionó el motor Game Unity Engine para el desarrollo de este proyecto.

En los *framework* de desarrollo evaluados, se pueden evidenciar distintas características y bondades de cada uno. Unreal Engine ofrece un realismo excepcional gracias a sus avanzadas capacidades gráficas, lo que facilita significativamente el aprendizaje. Además,

permite la integración de tecnologías como la inteligencia artificial y el análisis de datos en tiempo real. Por su parte, el marco de desarrollo Unity aumenta la reutilización y modularidad del código, por lo que disminuye el tiempo de aprendizaje, lo que se traduce en un desarrollo más rápido y eficiente.

La arquitectura del *software* organiza las funcionalidades básicas de los videojuegos, integrando el diseño por capas y el diseño basado en componentes (Boettcher y Behr, 2021; Chandra *et al.*, 2021; Chen y Price, 2022; Ciekanowska *et al.*, 2021; El-Wajeh *et al.*, 2022; Wiesing *et al.*, 2020). El marco de trabajo de Unity permite la compatibilidad con una gran variedad de dispositivos y consolas (Aguas *et al.*, 2023). El motor de videojuegos Unity y el motor Unreal Engine ofrecen funcionalidades similares, tales como sistemas de renderizado, físicas y animación. Cada motor o *game engine* se enfoca en diferentes públicos. Unity goza de mayor popularidad entre desarrolladores independientes y estudios más pequeños, mientras que Unreal Engine suele ser utilizado por empresas de mayor envergadura. Según un estudio de comparación de rendimiento, ambos motores abordan tareas similares de manera distinta. Unity se distingue por su interfaz de fácil utilización, mientras que Unreal Engine es reconocido por su alta calidad gráfica. En lo concerniente a la comunidad y los recursos requeridos, Unity cuenta con una mayor aceptación entre desarrolladores principiantes. Si bien tanto Unity como Unreal Engine comparten las características fundamentales necesarias para el desarrollo de juegos, difieren considerablemente en cuanto al público objetivo, las características de rendimiento y las capacidades específicas (Ciekanowska *et al.*, 2021).

4.3. Desarrollo del simulador

El *software* fue estructurado en cuatro módulos para satisfacer los requerimientos funcionales de la aplicación. Estos se presentan a continuación:

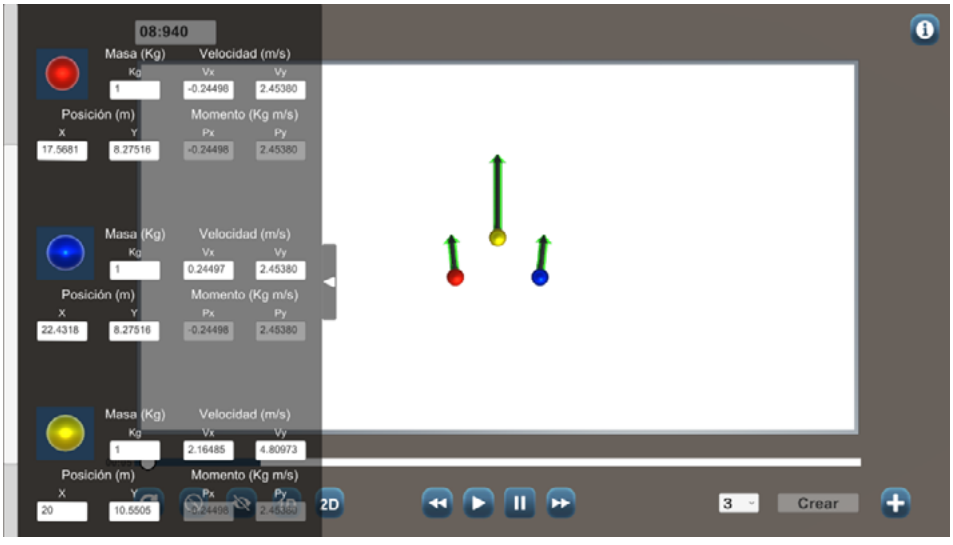
- **Modelador.** Componentes encargados de la modelación de los choques.
- **Simulador.** Componentes encargados de la simulación de los choques.
- **Solucionador.** Subsistema encargado de proporcionar la resolución matemática del choque simulado.
- **Diagrama.** Conjunto de herramientas usadas para la ilustración de la suma vectorial y el momento total.

4.3.1. Modelador

El requerimiento principal que se implementa en este módulo es que el usuario sea capaz de crear los elementos del problema de colisión, es decir, crear las partículas que partici-

pan en la colisión. Este proceso debe permitir ingresar de forma sencilla datos como masa, posición, velocidad, momento y elasticidad de las partículas. En el presente trabajo, se refiere a las partículas como esferas, ya que con dicha forma se presentan en el *software*. La generación de esferas (objetos 3D) se realiza en una interfaz simple e intuitiva que presenta, en el plano, esferas en una posición inicial, con un vector velocidad y un vector momento en forma representativa de flecha negra y flecha verde, respectivamente. Se presentan al usuario dos modalidades de ingreso de los datos, la gráfica y la textual. El método gráfico se basa en que el usuario puede interactuar tanto con la esfera como con la flecha de velocidad, de modo que, al mantener el clic sobre cualquiera, esta se moverá según la dirección del ratón. Al soltar el clic, la esfera se posicionará en una nueva coordenada o la flecha de velocidad se extenderá o acortará dependiendo del nuevo vector velocidad indicado por el usuario de forma gráfica. El método textual permite ingresar los datos relacionados a las esferas de forma más precisa por medio de un menú deslizable. El menú presenta opciones generales de visibilidad para elementos como el vector velocidad, el vector momento, el resultado del cálculo de energía cinética, la elasticidad de las esferas, entre otros, y presenta una opción que regula la elasticidad de las esferas. Cabe mencionar que, al modificar de forma gráfica los elementos mencionados anteriormente, es decir, la posición de la esfera o su velocidad por medio del *mouse*, los formularios del menú actúan como visualizadores que reflejan a tiempo real la representación vectorial y numérica de los cambios gráficos. En la figura 1 se presenta una vista de la interfaz gráfica.

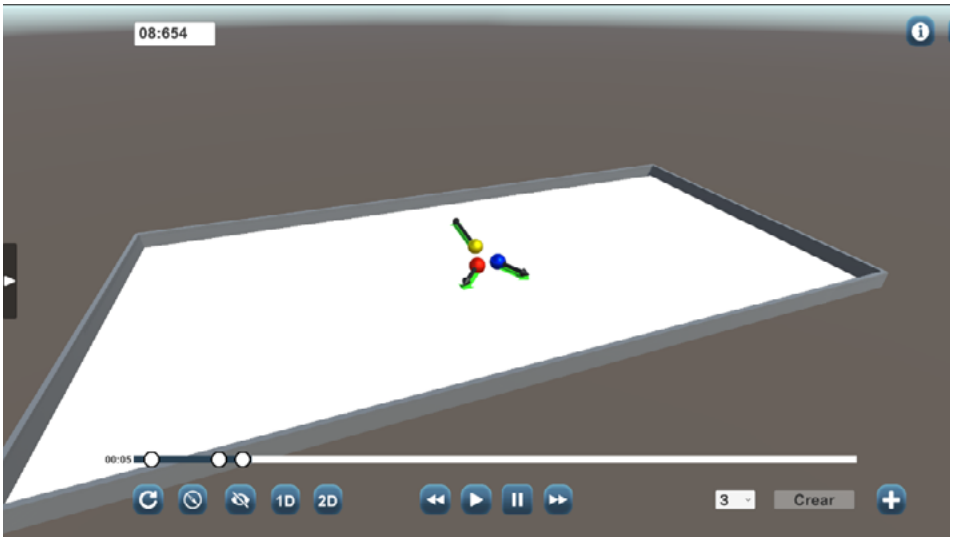
Figura 1. **Interfaz con menú deslizable izquierdo y datos a tiempo real**



Fuente: elaboración propia.

En general, todos los cuerpos con nombre pueden moverse y son completamente interactivos. Se pueden ajustar sus parámetros para modificar su comportamiento para diferentes escenarios y condiciones. Esto permite a los estudiantes explorar y comprender el comportamiento de las colisiones en diversos contextos.

Figura 2. **Interfaz implementando la vista 3D**



Fuente: elaboración propia.

En cuanto al comportamiento que puede realizar cada cuerpo u objeto presente en el ambiente de la simulación, se describe de la siguiente manera:

- **Plano.** Puede ser rotado con el clic derecho desde todos los ángulos, dándole al usuario una amplia perspectiva. En la figura 2 se presenta una vista de la interfaz en 3D.
- **Paredes.** Se encuentran delimitando el plano. No son arrastrables, pues mantienen su posición según el plano, incluso si este es rotado. Poseen una característica reflectante que puede ser habilitada o deshabilitada a requerimiento del usuario para que interfieran o no en la dinámica de la simulación. Es decir, si son habilitadas, las paredes se mostrarán en la escena y tendrán un efecto reflectante en las esferas, haciéndolas rebotar de manera elástica dentro del área delimitada por el plano. Si las paredes son deshabilitadas, las esferas seguirán su curso, eventualmente desapareciendo de la escena en pantalla. Una notificación avisa

al usuario de la situación y le recomienda utilizar el botón de «Retorno» para llevar las esferas a su posición inicial. La característica reflectante de las paredes puede configurarse desde el menú.

- **Esferas.** Son la representación gráfica de las partículas de choque. Son arrastrables dentro del área del plano. Cada esfera posee dos flechas: una negra, que muestra el vector de velocidad, y una verde, como vector de momento.
- **Vector velocidad.** Se representa por una flecha negra asociada a la esfera. Esta es arrastrable desde su segmento triangular o punta. Es arrastrable de forma libre dentro de los parámetros [-10, 10] tanto en el eje *x* como en el eje *y*. Al llegar a dicho límite, la flecha solo permite rotación. Dichos límites fueron fijados para mantener una representación gráfica adecuada, ya que, si la velocidad alcanza una magnitud mayor, esta no puede ser apreciada de manera correcta en la pantalla.
- **Vector momento.** Se refleja por una flecha verde asociada a una esfera. No es arrastrable, pues depende del vector velocidad y la masa de la esfera. Esta flecha solo es de carácter visual.

4.3.2. Simulador

Este módulo implementa la simulación del comportamiento físico de las colisiones, conformado por el movimiento de las esferas, la detección de colisiones entre esferas y el sistema de reproducción de la simulación.

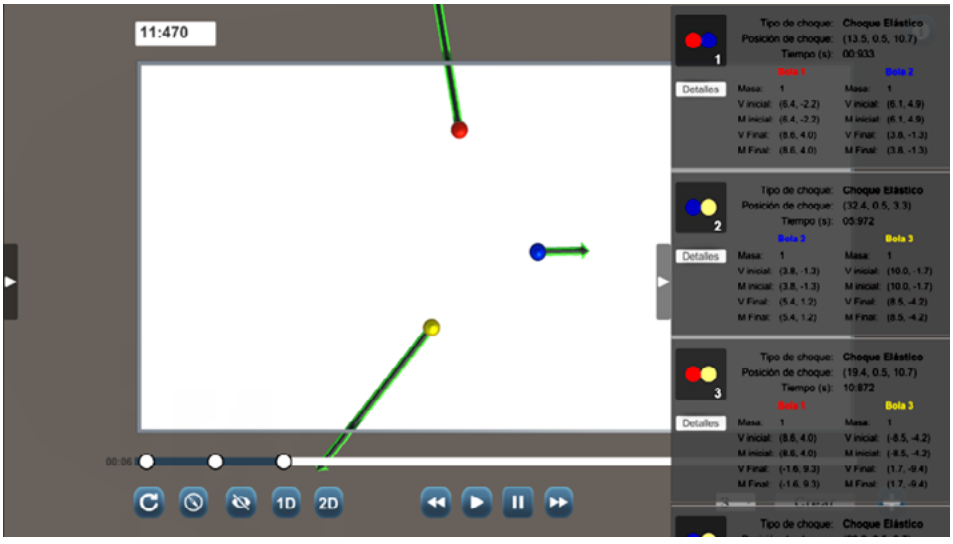
Estas funcionalidades del *software* de simulación son posibles gracias al uso de los componentes del motor de física integrado en el motor de desarrollo de videojuegos Unity. Los motores de física de Unity proporcionan componentes que manejan la simulación de la física, permitiendo un comportamiento realista de los objetos mediante colisiones, gravedad y otras fuerzas. Unity dispone de motores de física independientes para 2D y 3D que pueden controlarse mediante *scripts* para dotar a los objetos de movimiento dinámico (Unity Technologies, 2019). Estos componentes proporcionan realismo y precisión en las simulaciones y comportamientos físicos de los objetos en el entorno virtual. Estas funcionalidades del *software* se implementan gracias a los componentes Rigidbody y Collider. Al agregar un componente Rigidbody a un objeto en el *software* de simulación, su movimiento queda bajo el control del motor de físicas de Unity. Al activarlo, incluso sin agregar ningún código específico, un objeto Rigidbody será empujado hacia abajo por la gravedad y reaccionará a las colisiones con los objetos entrantes si en estos también está presente el componente Collider que lo configura como un cuerpo físico; es decir, se comporta como los objetos físicos en el mundo real. El componente Collider tiene tres características de interés para el *software*. Estas son el cuerpo con forma geométrica o volumen que ocupa el Collider, el material físico que lo compone y el tipo de colisión (Collider). El material físico del componente permite manejar propiedades como la fricción y el rebote (elasticidad) del objeto al

chocar. Para el simulador, se utilizó el modo ForceMode.Impulse con la finalidad de medir las fuerzas suministradas en Newtons por segundo (United Technologies, 2025b). Este modo emplea la fuerza para alterar la velocidad de forma proporcional a la masa del objeto, por lo que el efecto resultante depende de la masa del cuerpo.

Para la visualización del problema de choque se implementó un reloj o temporizador que marca el tiempo de la simulación y una barra de reproducción que, además de señalar el tiempo transcurrido gráficamente, también indica el momento exacto en el que ocurre un choque. El usuario puede pausar la simulación, acceder y poder repetir todos los choques que ocurrieron en la línea de tiempo disponible, cambiar de choque unidimensional a bidimensional y cambiar de modo 2D a 3D. Estas funciones le aportan al simulador el dinamismo para poder adaptarse a las diferentes situaciones problemáticas que requiera experimentar el usuario.

Los choques ocurridos se muestran en un menú informativo deslizable, ubicado en el lado derecho de la interfaz (véase figura 3), que se desplaza al centro al hacer clic sobre su pestaña. En él se presenta un listado de los choques ocurridos y los datos de las esferas involucradas en dos fases claramente definidas: el momento previo y el momento posterior al choque. Esto se representa para poder comparar ambas situaciones. Los datos relacionados a las esferas partícipes en una colisión alimentan al módulo «Solucionador».

Figura 3. **Interfaz con menú deslizable derecho**



Fuente: elaboración propia.

4.3.3. Solucionador

El solucionador representa algebraicamente el comportamiento físico mostrado visualmente en el simulador e indica la teoría física aplicada y su justificación. Esta representación aporta un gran valor pedagógico al simulador. Gracias a los datos proporcionados por el simulador, tales como masa, velocidad, momento y posición de las esferas partícipes de un choque, se procede con la aplicación de la teoría de la física. La formulación matemática permite presentar información sobre la energía cinética, el vector final de momento y las velocidades finales de las esferas después del choque.

4.3.4. Diagrama

El diagrama de momento es un módulo que permite la visualización de la suma vectorial para el vector momento total. Este presenta, en menor escala, las flechas que modelan el momento. Está conformado por un recuadro en el que se presenta gráficamente la sumatoria de los vectores momento de las esferas activas y su vector momento resultante, utilizando el método del polígono. Este método elige una escala apropiada para trazar los vectores que, posteriormente, se dibujan uno seguido de otro; es decir, se traza el primer vector y, al final de este, se comienza a trazar el segundo vector, y así sucesivamente con todos los vectores que hay que sumar, manteniendo siempre su magnitud y dirección. Por último, se dibuja el vector resultante (suma de los vectores), que va desde el origen hasta el final del último vector.

Los módulos «solucionador» y «diagrama» están integrados al simulador, permitiendo a los usuarios, profesores o estudiantes, utilizar el diagrama vectorial de los momentos y ver en tiempo real cómo cambian con cada choque ocurrido en el sistema y cómo el momento total se conserva.

4.4. Validación del simulador

El simulador presentado es una gran ayuda para cumplir con el objetivo general del tema de choques que se cubre en la física básica: poder aplicar la ley de conservación del momento lineal para analizar las colisiones entre objetos. La representación visual del modelo de choques es indispensable para entender el evento donde dos objetos macroscópicos se aproximan entre sí e interactúan durante un breve tiempo. El resultado de esa interacción es el cambio casi instantáneo de momento de cada uno de los objetos sin modificar sus posiciones; esto se evidencia cuando se ralentiza la simulación. Así, las únicas fuerzas experimentadas por los objetos aparecen cuando están en contacto con los otros objetos o con las paredes del contenedor. Cuando no están chocando, viajan en línea recta a rapidez constante: se comportan como partículas libres. Este hecho se puede usar para refor-

zar que las únicas fuerzas que actúan durante este choque son las debidas a la interacción entre los objetos: las fuerzas impulsivas internas. Esto implica que en este modelo se debe conservar el momento lineal.

La opción del simulador que muestra el vector momento lineal total en cualquier instante de tiempo del sistema de dos objetos que se mueven en el plano es adecuada para entender que, aunque los objetos interactúen, ese vector nunca cambia ni en magnitud ni en dirección: se conserva. Desde el punto de vista educativo, es una representación muy conveniente, ya que los estudiantes, a menudo, suman solo las magnitudes de los momentos individuales.

Los choques, en una dimensión, ofrecen una vía para tratar en forma simple la conservación del momento. Adicionalmente, cuando se supone la conservación de la energía cinética, nos da la facilidad de obtener las velocidades finales de los objetos que intervienen en la colisión a partir de las velocidades iniciales, conocidas las masas de los objetos. Esta situación puede visualizarse en el laboratorio de manera sencilla y comparar los resultados reales a los de la simulación.

La inclusión de varios objetos chocando en un recinto cerrado, considerando que los choques son elásticos, es una opción que ofrece el simulador. Es de gran ayuda para visualizar el modelo elemental de los gases ideales.

Una posible ampliación del simulador se obtendrá al incorporar una pequeña fuerza externa que haga que los objetos no se muevan en línea recta cuando no estén interactuando. Esto enfatiza la idea de que la conservación del momento lineal en los choques solo se cumple para el momento lineal total, justo antes y justo después del choque.

El simulador de colisiones se considera una herramienta valiosa para la enseñanza de la ley de conservación del momento lineal en física básica. La representación visual y la posibilidad de explorar diferentes escenarios de colisión permiten a los estudiantes comprender mejor los conceptos físicos involucrados, facilitando un aprendizaje más profundo y significativo.

4.5. Distribución web

Con la finalidad de distribuir la aplicación y que su acceso fuera público para estudiantes y profesores, se tomó la decisión de subirla a una plataforma de distribución web, accesible a través de dispositivos que soportaran OpenGL y tuvieran acceso a internet. Se eligió Itch.io como canal de distribución, ya que es una plataforma de distribución digital destinada principalmente a la publicación de videojuegos independientes y que permite la ejecución en línea sin coste alguno.

5. Discusión

La metodología Scrum implica la entrega de una versión parcialmente completa y funcional del producto al final de cada *sprint*, denominada «incremento». Cada incremento debe ser potencialmente utilizable por los usuarios finales e incluir nuevas funcionalidades, mejoras o correcciones según criterios de aceptación predefinidos. El producto se desarrolla de manera iterativa e incremental, basándose en los incrementos anteriores, para obtener una retroalimentación temprana y continua de las partes interesadas y garantizar la transparencia en el progreso del proyecto. La revisión del incremento funcional al finalizar cada *sprint* permite al propietario del producto y a las partes interesadas (*stakeholders*), en este caso los profesores del Departamento de Física, evaluar su grado de satisfacción con el resultado obtenido y, en función de esto, redefinir la forma en que desean que el simulador opere. De esta manera, se fueron mejorando los aspectos académicos de acuerdo con la experiencia de los profesores del Departamento de Física para determinar cómo debía comportarse el simulador para la experimentación de los estudiantes. Este enfoque empírico permite flexibilidad, adaptabilidad y colaboración en el proceso de desarrollo, garantizando la entrega continua de valor.

Mediante la utilización de motores de videojuegos, se ha logrado crear un *software* capaz de simular colisiones de sistemas de partículas. Se ilustró de manera gráfica el principio de conservación del momento lineal en estos procesos y se puso de manifiesto la clasificación de los choques atendiendo al balance de energía cinética justo antes y después de que ocurriera. Los motores de videojuegos no solo ofrecen capacidades gráficas, sino que también incorporan un sistema de cálculo de físicas para representar el comportamiento de objetos reales en el entorno de los videojuegos.

Los motores de videojuegos, con sus capacidades de simulación física, detección de colisiones y modelado de comportamientos realistas, no solo son esenciales para crear experiencias inmersivas en los juegos, sino que también se presentan como herramientas valiosas para la enseñanza de la física.

Son diversos los aportes del motor de videojuegos al simulador de colisiones:

- **Representación visual.** Al permitir representar de manera visual y dinámica las colisiones y otros fenómenos físicos, lo que facilita la comprensión de los conceptos.
- **Interactividad.** Al permitir a los estudiantes interactuar con los objetos y sistemas simulados, lo que les brinda la oportunidad de experimentar y explorar diferentes escenarios y configuraciones.
- **Retroalimentación inmediata sobre el resultado de las colisiones y sus representaciones matemáticas (algebraica y vectorial).** Permite a los estudiantes comprender e interpretar la situación problemática desde diversas perspectivas.

- **Motivación y compromiso.** El uso de motores de videojuegos en el aprendizaje de la física puede aumentar la motivación y el compromiso de los estudiantes, ya que brinda la oportunidad de aprender de manera lúdica y divertida.

El dinamismo y la adaptabilidad del *software* a las situaciones de estudio se logra por medio de las diversas funciones que tiene el simulador para que el usuario pueda definir las características de las partículas; continuar o pausar la simulación; acceder y poder repetir todos los choques que ocurrieron en la línea de tiempo disponible; cambiar de choque unidimensional a bidimensional; cambiar de modo 2D a 3D; utilizar el diagrama vectorial de los momentos; y ver, en tiempo real, de qué modo se transforman con cada choque que se produce en el sistema y cómo el momento total se conserva. Estas funciones le aportan al simulador un elemento dinámico necesario para que los usuarios puedan responder a distintas situaciones problemáticas.

El simulador permite que los estudiantes y docentes puedan modelar problemas particulares. Los profesores pueden crear los problemas según lo que deseen mostrar a sus estudiantes por medio de la simulación. Los estudiantes, por su parte, pueden cargar los problemas que tienen que resolver y ver la simulación y la resolución matemática de los mismos. Se desea motivar la exploración libre de los alumnos al permitir que los estudiantes puedan «jugar» libremente con el simulador a fin de comprender la dinámica de las colisiones al crear situaciones particulares según su curiosidad e interés.

Se responde a los requerimientos por medio de:

- **Funcionalidades específicas.** El simulador cuenta con funciones específicas para la visualización de escenarios de colisión en una y dos dimensiones, la configuración de propiedades de los objetos involucrados en la colisión y la personalización de parámetros físicos.
- **Interfaz intuitiva.** El simulador tiene una interfaz intuitiva y fácil de usar que permite a los usuarios interactuar de manera efectiva con sus funcionalidades. Esto incluye controles claros y una navegación sencilla para acceder a diferentes opciones y configuraciones.
- **Visualización clara.** El simulador proporciona una visualización clara y precisa de las colisiones, utilizando las representaciones gráfica, algebraica y vectorial para ayudar a los estudiantes a comprender mejor los fenómenos físicos involucrados. Esto incluye representar las trayectorias de los objetos, las velocidades y los momentos involucrados en las colisiones.
- **Personalización.** El simulador permite a los usuarios personalizar diferentes aspectos de las colisiones, como las propiedades de los objetos, las condiciones iniciales y los parámetros físicos. Esto brinda a los estudiantes la oportunidad de

explorar diferentes escenarios y experimentar con diferentes variables, enriqueciendo su comprensión de los conceptos de colisión.

- **Accesibilidad.** El simulador es accesible desde diferentes dispositivos y plataformas, como ordenadores, tabletas y dispositivos móviles. La elección de una plataforma de distribución, como Itch.io, representa una ventaja en cuanto a la difusión del simulador entre los estudiantes, además de que facilita su uso en las aulas al requerir solo un ordenador con acceso a internet. Esto permitirá a los estudiantes acceder al simulador desde cualquier lugar y en cualquier momento, facilitando su uso como herramienta de enseñanza tanto en el aula como fuera de ella.

6. Conclusiones

La utilización de motores de videojuegos en el estudio de la física y las colisiones ha permitido desarrollar un *software* que complementa el aprendizaje de estos conceptos mediante representaciones visuales, interactivas y en movimiento. Estas herramientas ofrecen beneficios, como la representación visual, la interactividad, la retroalimentación inmediata y la motivación, así como el compromiso de los estudiantes.

Se ha desarrollado un laboratorio de física virtual que se enfoca en las colisiones entre partículas. El objetivo principal de este laboratorio es ayudar a los estudiantes de los primeros cursos de Física universitaria a comprender, a través de representaciones visuales, la conservación del momento lineal y, en general, la no conservación de la energía cinética en estos eventos. Lo interesante de este trabajo es que se utiliza la ventaja de los motores de los videojuegos para crear nuevas simulaciones adaptadas a las especificaciones de los docentes. Además, permite una mayor interactividad y participación de los estudiantes en el aprendizaje de la física a partir de las propias simulaciones creadas. El desarrollo de simulaciones en sí mismo tiene la virtud de ayudar a los creadores a profundizar sus conocimientos de física. Al crear estas simulaciones, los docentes y estudiantes pueden explorar y facilitar la comprensión de los conceptos y principios físicos involucrados en las colisiones entre partículas.

La metodología Scrum, al implicar la entrega de incrementos parcialmente completos y funcionales del producto al final de cada *sprint*, permite un desarrollo iterativo e incremental, con énfasis en la retroalimentación temprana y continua de las partes interesadas. La revisión del incremento funcional al finalizar cada *sprint* facilita la evaluación del grado de satisfacción de los usuarios finales, en este caso los profesores del Departamento de Física, y la redefinición de las funcionalidades del simulador para satisfacer las necesidades académicas específicas. De esta manera, se logra una mejora continua en función de la experiencia y retroalimentación de los profesores, lo que garantiza la entrega continua de valor, flexibilidad y adaptabilidad en el proceso de desarrollo.

El motor de física presente en el marco de desarrollo Unity es una ventaja fundamental para el desarrollo de simuladores. Las capacidades avanzadas del motor de física permiten a los desarrolladores crear simulaciones realistas que reflejen con precisión y exactitud la física del mundo real, lo que lo convierte en una herramienta ideal para fines formativos y educativos. Esta característica permite a los desarrolladores recrear el comportamiento de los objetos en el mundo real sin la necesidad de ocuparse de los cálculos detallados que modelan su comportamiento, lo que convierte a Unity en una opción ideal para crear simulaciones realistas.

El *software* desarrollado no se limita a un grupo de situaciones preestablecidas, sino que permite la generación de problemas particulares, permitiendo reproducir los problemas que el estudiante aborda en sus sesiones de estudio o generar nuevas situaciones de acuerdo con su interés. El *software* se ha estructurado en módulos –simulador, solucionador y diagrama– que permiten mostrar el comportamiento de las partículas de forma gráfica, algebraica y vectorial en al menos tres momentos clave de la simulación: antes, durante y después del choque.

Al proporcionar una experiencia personalizada, el *software* puede mejorar el aprendizaje, aumentar el compromiso y mejorar el rendimiento académico. Se cuenta con estudios previos que muestran los beneficios del uso de simuladores en la educación, particularmente en el área de la ingeniería. El simulador desarrollado busca aprovechar la familiaridad de los estudiantes con los videojuegos y la tecnología para mejorar su comprensión de conceptos físicos complejos.

Como herramienta educativa, el simulador de colisiones cumplió las expectativas de los profesores del Departamento de Física de la Universidad Metropolitana. No solo funcionalmente, sino que también proporciona a los alumnos una experiencia de aprendizaje enriquecida al centrarse en áreas que, según las experiencias de los profesores, requieren más apoyo y experimentación.

Los resultados son aplicables a la ciencia, la tecnología, la ingeniería y la matemática (*science, technology, engineering and mathematics* [STEM]). En este caso, la física es un vehículo para mostrar las virtudes de los motores de los videojuegos para la creación de simulaciones. La simulación de colisiones es de gran utilidad como recurso educativo para los cursos de Física elemental universitaria.

Referencias bibliográficas

Aguas, L., Recalde, H., Toasa, R. y Salazar, E. (2023). Design of a video game applying a layered architecture based on the unity framework. En Á. Rocha, C. Ferrás y W. Ibarra (Eds.), *Information Technology and Systems* (pp. 535-550). Springer International Publishing. https://doi.org/10.1007/978-3-031-33261-6_46

Alonso, S., Kalinowski, M., Ferreira, B., Barbosa, S. D. J. y Lopes, H. (2023). A systematic mapping study and practitioner insights on the use of software engineering practices to develop MVPs. *Information and Software Technology*, *156*. https://doi.org/10.1016/j.infsof.2022.107144

Barragán Suescún, F. (2020). Simulaciones interactivas: nuevas herramientas en el aprendizaje contextualizado de la Física universitaria. *Revista Ciencias de la Educación*, *30*(56), 541-568. http://servicio.bc.uc.edu.ve/educacion/revista/56/art02.pdf

Boettcher, K. y Behr, A. (2021). Using virtual reality for teaching the derivation of conservation laws in fluid mechanics. *International Journal of Engineering Pedagogy*, *11*(4), 42-57. https://doi.org/10.3991/ijep.v11i4.20155

Carangui Cárdenas, L. R., Cajamarca Criollo, O. A. y Mantilla Crespo, X. A. (2017). Impacto del uso de simuladores en la enseñanza de la administración financiera. *Innovación Educativa (México, DF)*, *17*(75), 103-122. https://www.redalyc.org/pdf/1794/179454112006.pdf

Chandra, A. Y., Prasetyaningrum, P. T., Suria, O., Santosa, P. I. y Nugroho, L. E. (2021). Virtual reality mobile application development with Scrum framework as a new media in learning english. *International Journal of Interactive Mobile Technologies*, *15*(8), 31-49. https://doi.org/10.3991/ijim.v15i08.19923

Chen, J. y Price, E. (2022). *Game Development with Unity for .NET Developers: The Ultimate Guide to Creating Games with Unity and Microsoft Game Stack*. Packt Publishing Ltd.

Chover, M., Marín, C., Rebollo, C. y Remolar, I. (2020). A game engine designed to simplify 2D video game development. *Multimedia Tools & Applications*, *79*(17-18), 12.307-12.328. https://doi.org/10.1007/s11042-019-08433-z

Ciekanowska, A., Kiszczak-Gliński, A. y Dziedzic, K. (2021). Comparative analysis of Unity and unreal engine efficiency in creating virtual exhibitions of 3D scanned models. *Journal of Computer Sciences Institute*, *20*, 247-253. https://doi.org/10.35784/jcsi.2698

Elizondo Treviño, M.ª S. (2013). Dificultades en el proceso enseñanza aprendizaje de la Física. *Presencia Universitaria*, *Año 3*(5), 70-77. http://eprints.uanl.mx/3368/1/Dificultades_en_el_proceso_ense%C3%B1anza_aprendizaje_de_la_F%C3%ADsica.pdf

El-Wajeh, Y. A. M., Hatton, P. V. y Lee, N. J. (2022). Unreal Engine 5 and immersive surgical training: translating advances in gaming technology into extended-reality surgical simulation training programmes. *British Journal of Surgery*, *109*(5), 470-471. https://doi.org/10.1093/bjs/znac015

Epic Games. (2024). *Unreal Engine 5.3 Documentation*. https://docs.unrealengine.com/5.3/en-US/

González-González, C. S., Toledo-Delgado, P. y Muñoz-Cruz, V. (2015). Agile human centered methodologies to develop educational software. *DYNA*, *82*(193), 187-194. https://doi.org/10.15446/dyna.v82n193.53495

Hernández-Paez, A., Domínguez Falcón, J. A. y Pi Cruz, A. A. (2018). Arquitectura de software para el desarrollo de videojuegos sobre el motor de juego Unity 3D. *Revista de I+D Tecnológico*, *14*(1), 54-65. https://doi.org/10.33412/idt.v14.1.1803

Hernández Sampieri, R., Fernández Collado, C. y Baptista Lucio, P. (2014). *Metodología de la investigación* (6.ª ed.). McGraw-Hill.

Isela Aguilar Vargas, L. R. y Otuyemi Rondero, E. O. (2020). Análisis documental: importancia de los entornos virtuales en los procesos educativos en el nivel superior. *Tecnología, Ciencia y Educación*, *17*, 57-77. https://doi.org/10.51302/tce.2020.485

Jiménez-Torres, V. H., Tello-Borja, W. y Rios-Patiño, J. I. (2014). Lenguajes de patrones de arquitectura de software: una aproximación al estado del arte. *Revista Scientia et Technica*, *19*, 371-376. https://revistas.utp.edu.co/index.php/revistaciencia/article/view/8595/5869

Kortemeyer, G. (2023). Writing virtual reality teaching resources. *The Physics Teacher*, *61*(2), 107-109. https://doi.org/10.1119/5.0067963

López-Mera, D. D., Archila-Gutiérrez, A. C., Hernández-Montoya, B. C., Suárez-Chávez, S. E. y Pérez-Rojas, E. H. (2021). Modelo para la creación del juego de realidad alternativa «El Plan de Gauss»: matemáticas, relatos y juegos en instituciones de educación superior. *Tecnología, Ciencia y Educación*, 133-154. https://doi.org/10.51302/tce.2021.529

Makamu, G. y Ramnarain, U. (2022). Physical sciences teachers' enactment of simulations in 5E inquiry-based science teaching. *Education Sciences*, *12*(12), Article 12. https://doi.org/10.3390/educsci12120864

Monfort Salvador, V. y E-Martín, Y. (2022). CoopDev project: learning developments through different methodologies and strategies. *Conference Proceedings International Scientific & Professional Conference Contemporary Issues in Economy & Technology, CIET* (pp. 848-859).

O3DE. (2024). https://o3de.org/

Otto, A. I., Bakieva Karimova, M. y García Laborda, J. (2023). Evaluación formativa a través de herramientas informáticas: nuevos enfoques y perspectivas. *Tecnología, Ciencia y Educación*, *26*, 7-8. https://www.tecnologia-ciencia-educacion.com/index.php/TCE/article/view/19285

Pérez-Higuera, G. D., Niño-Vega, J. A. y Fernández-Morales, F. H. (2020). Estrategia pedagógica basada en simuladores para potenciar las competencias de solución de problemas de física. *Aibi. Revista de Investigación, Administración e Ingeniería*, *8*(3) 17-23. https://doi.org/10.15649/2346030X.863

Prado Martínez, W. (2008). Simulación computacional para la enseñanza de la física. *Entre Ciencia e Ingeniería*, *3*, 111-124. https://revistas.ucp.edu.co/index.php/entrecienciaeingenieria/article/view/826

Rodrigues de Oliveira, E., Cabral Ribeiro, P. C., Picinini Méxas, M. y Barbará de Oliviera, S. (2023). Scrum method assessment in Federal Universities in Brazil: multiple case studies. *Brazilian Journal of Operations & Production Management*, *20*(1), 1-14. https://doi.org/10.14488/BJOPM.1496.2023

Salimzyanova, E. Sh. (2022). Agile in digital didactics in the era of the VUCA world in education. *ARPHA Proceedings*, *5*, 1.417-1.432. https://zenodo.org/records/6132816

Schwaber, K. y Sutherland, J. (2020). *La guía Scrum: la guía definitiva de Scrum: las reglas del juego.* https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish-European.pdf

Scott, E. y Campo, M. (2023). An adaptive 3D virtual learning environment for training *software* developers in Scrum. *Interactive Learning Environments*, *31*(8), 5.200-5.218. https://doi.org/10.1080/10494820.2021.1999985

Torres-Blasco, C. y Pérez-Garcias, A. (2023). Indicadores de agencia en experiencias educativas Agile: una revisión panorámica. *Píxel-Bit. Revista de Medios y Educacion*, *68*, 183-215. https://doi.org/10.12795/pixelbit.98457

Umbreen, J., Mirza, M. Z., Ahmad, Y. y Naseem, A. (2022). Assessing the role of minimum viable products in digital startups. *2022 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)* (pp. 1.073-1.077). https://doi.org/10.1109/IEEM55944.2022.9989653

Unity Technologies. (2019). *Unity Documentation. Unity User Manual*. *Física.* https://docs.unity3d.com/es/2018.4/Manual/PhysicsSection.html

United Technologies. (2025). *Unity Documentation-Scripting API: ForceMode.Impulse*. https://docs.unity3d.com/ScriptReference/ForceMode.Impulse.html

Unity Technologies. (2025). *Unity Documentation. Docs and Guides to Work with the Unity Ecosystem*. https://docs.unity.com/

Wiesing, M., Fink, G. R. y Weidner, R. (2020). Accuracy and precision of stimulus timing and reaction times with Unreal Engine and SteamVR. *PLOS ONE*, *15*(4), 1-24. https://doi.org/10.1371/journal.pone.0231152

María del Pilar Cuenca Marcano. Ingeniera de Sistemas con Maestría de Administración (Mención Gerencia de Finanzas) por la Universidad Metropolitana (Caracas, Venezuela). Actualmente, trabaja como profesora del Departamento de Gestión de Proyectos y Sistemas en la Universidad Metropolitana en las áreas de Bases de Datos, Sistemas de Información e Ingeniería de Software. Ha incorporado el marco de trabajo ágil en la educación a nivel de pregrado en el área de Desarrollo de Software.

Salomón Mizrachi Cohén. Licenciado en Física e ingeniero civil. Actualmente, trabaja como profesor de Física del Departamento de Física de la Universidad Metropolitana (Caracas, Venezuela). Es responsable de la creación y difusión de simulaciones de física y de las actividades asíncronas de dicho departamento.

Gabriela Banezca Luces. Ingeniera de sistemas. Profesional con habilidades destacadas para el trabajo en equipo y orientada a la mejora continua como desarrolladora. Posee competencias en C#, Python, JS, Java, HTML, CSS y Unity. Cuenta con experiencia en el desarrollo de bases de datos en SQL Server para proyectos .NET y roles de asistente de investigación. Se graduó con mención honorífica por su tesis sobre un simulador de colisiones físicas en Unity. Ha recibido reconocimientos académicos y ha completado formación complementaria en áreas como SQL Server, React JS, MongoDB, JavaScript y redes neuronales.

José Villegas Vera. Ingeniero de sistemas. Profesional con amplia trayectoria en trabajo en equipo y en métodos de desarrollo con metodologías ágiles. Cuenta con experiencia en distintos lenguajes, como HTML, Java, JS, CSS, SQL, C# y C++; en diferentes *frameworks*, como Angular y React; en motores de videojuegos, como Unity; y en otras tecnologías, como MongoDB, PostgreSQL, PyTorch y TensorFlow. Ha desempeñado roles como asistente de profesorado en la Universidad Metropolitana (Caracas, Venezuela) dentro del área de Administración de Base de Datos. Actualmente, trabaja como auditor de sistemas en Mercantil Banco.

Contribución de autores. Idea: S. M. C., G. B. L. y J. V. V.; Revisión de literatura (estado del arte): M.ª P. C. M. y S. M. C.; Metodología: M.ª P. C. M., G. B. L. y J. V. V.; Análisis de datos: M.ª P. C. M., S. M. C., G. B. L. y J. V. V.; Resultados: M.ª P. C. M., S. M. C., G. B. L. y J. V. V.; Discusión y conclusiones: M.ª P. C. M. y S. M. C.; Redacción (borrador original): G. B. L., J. V. V. y M.ª P. C. M.; Revisiones finales: M.ª P. C. M. y S. M. C.