

JUEGO SERIO PARA FÍSICA COMO ESTRATEGIA DE APRENDIZAJE ACTIVA Y LÚDICA

Pacheco-González, Alberto
TecNM campus Chihuahua
División de Posgrado e Investigación
Av. Tecnológico 2909
alberto.pg@chihuahua.tecnm.mx

RESUMEN.

Se presenta el desarrollo de un juego serio como estrategia de aprendizaje activo supervisado para un curso de Física, para aplicar la ecuación de caída libre de forma inductiva y lúdica. Para la creación del juego serio se integraron diversos métodos, como la matematización y la programación en vivo, logrando una simulación eficiente y precisa del fenómeno físico incorporado. Se describe el motor de juegos, el diseño del juego serio, la planeación y estrategia didáctica del proyecto para lograr que estudiantes sin experiencia de programación concluyeran con éxito y a tiempo el juego de una forma autónoma, entusiasta y creativa.

Palabras Clave: Simulación, matematización, juegos serios, física, motores de juego, programación en vivo, desarrollo ágil.

ABSTRACT.

The development of a serious game is presented as a supervised active learning strategy for a Physics course, to discover the free fall equation in an inductive and playful way. For the creation of the serious game, various methods were integrated, such as mathematization and live coding, achieving an efficient and precise simulation of the incorporated physical phenomenon. The game engine, the design of the serious game, and the didactic strategy of the project are described to ensure that students lacking prior programming experience, could autonomously, enthusiastically, and creatively complete their game project successfully and on time.

Keywords: Simulation, mathematization, serious games, physics education, game engines, live coding, software agile methodologies.

1. PROBLEMÁTICA

Un serio problema en la educación actual de las Ciencias y específicamente, de la Física, es el creciente desinterés, el bajo rendimiento académico y la alta tasa de deserción, donde al parecer, el método pasivo y tradicional de enseñanza esta siendo inefectivo [1]. Para atender esta problemática, autores como [2, 3], recomiendan el uso de estrategias de aprendizaje activo y aprendizaje lúdico, i.e. juegos serios y gamificación.

El presente trabajo describe el diseño y desarrollo de un juego serio capaz de modelar y estimular el aprendizaje de las leyes de la Física involucradas en la caída libre de objetos, partiendo de un estado de reposo. El juego serio tiene los siguientes objetivos educativos: 1) para los estudiantes universitarios, matematizar

los principios cinemáticos vistos de manera teórica en un curso de Física siguiendo un conjunto de estrategias enfocadas en promover el aprendizaje activo supervisado, destacando además, que la mayoría de los estudiantes no tienen experiencia en programación; 2) utilizar dicho juego serio con estudiantes de Física de nivel medio superior para ilustrar las nociones de cinemática y dinámica de cuerpos rígidos.

2. MARCO CONCEPTUAL

2.1. Matematización.

A lo largo de la historia, la Ciencia ha progresado de forma inequívoca y gradual gracias a la verificación experimental de modelos teóricos, producto de la matematización, misma que permite sentar las bases para resolver problemas mediante el modelado matemático. Para desarrollar dicha competencia entre los estudiantes se recomienda que aprendan a transformar problemas reales en modelos matemáticos [4].

La Física originada en el siglo XVII con Galileo Galilei y Simon Stevin, alcanzó su logro supremo en la obra de Isaac Newton, siendo ejemplo contundente de la matematización como descripción matemática, cuantificando aquellos procesos que se estén analizando. Es decir, definiendo modelos donde “entran números y salen números” [5]. Para reducir la brecha entre el dominio matemático y el modelado de fenómenos físicos, en [6-8] se sugiere la incorporación de herramientas digitales, el pensamiento lógico, analítico, algebraico, sistémico, crítico y computacional, más recientemente englobados dentro de las habilidades STEM.

Las herramientas digitales permiten mejorar la matematización no sólo dentro de la Ciencia, sino también, en este caso, dentro de la enseñanza de la Física, al cerrar la brecha existente entre el dominio matemático y la realidad física [8]. Sin embargo, a pesar de los beneficios potenciales de las herramientas digitales, existe una curva de aprendizaje por vencer para que los educadores implementen dichas herramientas en sus clases.

2.2. Entornos, herramientas y motores de juego.

Para crear videojuegos, a diferencia del uso de librerías para graficación y componentes de juego, como OpenGL, DirectX, WebGL, Metal, Box2D, Cocos2D, PyQt, PyGame, un entorno integrado de desarrollo (*IDE*) ofrece una plataforma abstracta y

un arsenal de herramientas, editores, motores, gestores, librerías y componentes reusables para crear videojuegos. La arquitectura de un entorno integrado para juegos ofrece una parte visible y otra oculta. La parte externa interactúa con el desarrollador y ofrece diversas herramientas, como editores de escenas, niveles, scripts, propiedades, eventos, recursos, animaciones, sonidos y efectos (Fig. 1). La parte interna oculta es el motor de juegos (*game engine*) compuesto por un controlador de fuerzas, objetos y colisiones sujetas a las leyes de la Física (*physics engine*), un gestor de las eventos y controles de interfaz de usuario, un manejador de tarjetas gráficas (GPUs), manejador de audio, gestores de escenas, niveles, recursos, errores y un interprete de scripts.

A su vez, existen diversos géneros de videojuegos: acción, estrategia, aventura, deportes, casuales, de plataforma, Arcade, masivos y de roles (RPG, MMO, MOBA), simulación y educativo. De acuerdo con [9], en México los géneros más populares son los juegos de acción, aventura y casuales. Entre los motores de juegos más completos se encuentran Unity, Unreal Engine, Game Maker Studio, Scratch y Godot Engine, entre otros.

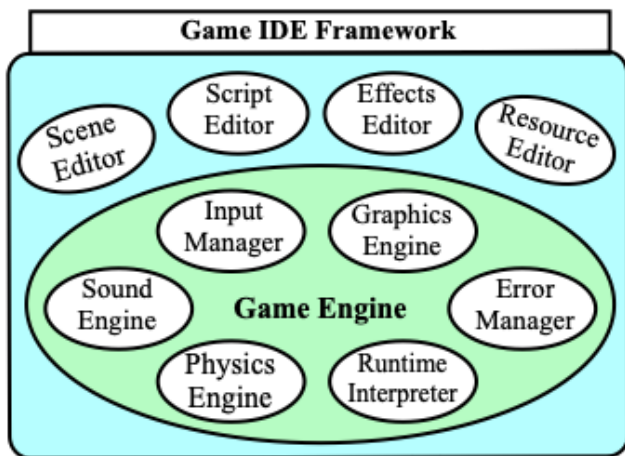


Figura 1. Principales subsistemas de un motor de juego.

El entorno multiplataforma Godot es un proyecto de software de código abierto bien soportado y documentado [10]. Este entorno es ligero y ofrece todos los componentes necesarios en un solo entorno (Fig. 1). Godot permite crear juegos 2D y 3D, soporta tarjetas gráficas basadas en Open GL y Vulkan, y posee un lenguaje de programación nativo similar a Python, llamado GDScript. Un juego se representa en Godot como un árbol de escenas y a su vez, cada escena es un árbol de nodos (Fig. 2). Existen además, muchos tipos de nodos: objetos cinemáticos, estáticos y rígidos, textos, botones, listas, ventanas, *timers*, *shaders*, imágenes, audios, animaciones, efectos, etc.

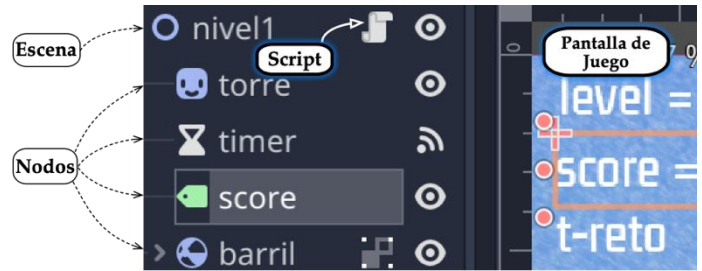


Figura 2. Entorno Godot: una escena del juego serio (nivel 1).

2.3. Juegos serios.

Un juego serio es un videojuego que además de ofrecer esparcimiento y diversión, integra intenciones pedagógicas, informativas e interactivas orientadas a educar a un individuo [11]. Los juegos serios permiten fortalecer la creatividad, la resolución de problemas, la colaboración y participación de los estudiantes [2, 11-14]. Diversos estudios en torno al diseño de juegos serios enfatizan, sobre todo, la importancia de alinear la dinámica del juego con los objetivos de aprendizaje para soportar así, la adquisición del conocimiento por parte del estudiante [13, 14].

3. DESARROLLO

3.1. Metodología.

Para atender la problemática relacionada con el bajo interés y rendimiento de los estudiantes de Física, se propone el desarrollo de un juego serio como estrategia de aprendizaje activo y aprendizaje inductivo [15], cuyo tratamiento pedagógico ha sido expuesto de forma más detallada en [16]. En concreto, el juego serio atiende dos objetivos educativos distintos (Fig. 3): 1) para el jugador, promueve un aprendizaje inductivo y lúdico de la Física al plantear diversos retos con la intención de descubrir principios físicos para la caída libre de objetos; 2) el juego sirvió como de evidencia de aprendizaje para el estudiante de Física, al ser el desarrollador del juego, contando con: la asistencia de una metodología y un mentor, una planificación de sesiones de programación en vivo (Tabla 1), buscando aplicar y matematizar la teoría relacionada con el tema de caída libre del curso de Física. En este trabajo se consideraron varias limitaciones académicas: el tiempo limitado para realizar el proyecto, estudiantes sin experiencia de programación y un grupo con un alto índice de reprobación [16]. Por tanto, como se resume la Fig. 3, la metodología fusiona tres flujos de trabajo concurrentes: D₁) para el estudiante-desarrollador, se aplican diversas estrategias didácticas, incluyendo un aprendizaje supervisado por un mentor [16]; D₂) para el jugador, como se mencionó, el juego promueve un aprendizaje inductivo y lúdico; D₃) para soportar un desarrollo rápido del juego se usó un motor de juegos y un método ágil de prototipado en espiral, obteniendo de forma gradual los módulos y versiones del juego [17, 18].

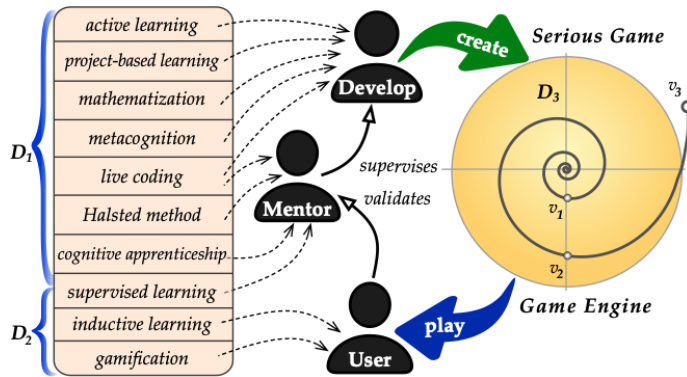


Figura 3. Métodos, roles, competencias y versiones del juego.

3.2. Diseño y desarrollo del juego serio.

El juego consta de seis escenas principales (Fig. 4): la pantalla inicial, cuatro niveles de juego y una pantalla final donde se reporta la puntuación global. Se elaboró una planeación didáctica y un programa de trabajo para las sesiones semanales de programación en vivo para desarrollar los distintos módulos y versiones del juego como se detalla en la Tabla 1, con las siguientes columnas: 1) número de sesión de programación en vivo supervisada; 2) actividad y tema de Física considerado; 3) herramientas y componentes creados siguiendo la metodología de prototipado en espiral.

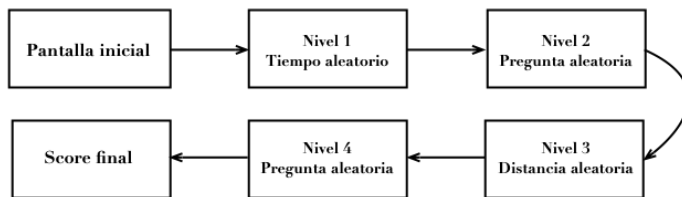


Figura 4. Pantallas y niveles del juego serio para Física.

El diseño de las pantallas de juego incluye dos plantillas superpuestas (*layouts*): un tablero de juego (*heads-up display*, HUD), y una escena para el nivel de juego. El tablero HUD (fig. 5) indica: el nivel de juego, el reto actual, el tiempo restante de juego y la puntuación (*score*). El tablero se creó en las sesiones 8-10 descritas en Tabla 1, usando una escena de tipo *singleton* (*autoload*) y además diversos nodos de control UI. Por otro lado, cada nivel de juego contiene (Fig. 6): una imagen de fondo, una regleta (escala de referencia), un personaje (objeto en caída libre), un peldaño móvil, una flecha, un barril y un piso. Cada estudiante desarrolló de forma libre y creativa sus propios recursos del juego (*assets*) asignando además su propia temática y estilo (color, tamaño, *font*, ubicación, etc.).

Tabla 1. Plan de trabajo: sesiones, temas y herramientas según [16].

Sesión	Temas de Física incorporados	Herramientas de Godot aplicada en cada sesión
1	Investigar juegos para Física	Encuesta inicial diagnóstica
2	Historia de la Física (Galileo)	Simul. y matematización de la Física
3	Cuerpos estáticos y rígidos	Manejo del entorno, editores y ventanas
4	Caída libre y cte. gravedad	Manejo de recursos (<i>assets</i> , imágenes)
5	Propiedades: masa, posición, veloc. inicial, elasticidad	Manejo de propiedades (peso, giro, damping)
6	Torque angular, fricción, amortig.	Scripts, variables, funciones, <i>random</i> , eventos
7	Matematizar (desplazar, propiedades)	Detección de colisiones y control teclado (cursor)
8	Cálculo de tiempo	Tablero: HUD y layouts
9	Actualizar tablero	Sonidos, timers y efectos
10	Cálculo de distancia	Manejo de niveles y var. globales
11	Mov. circular uniforme	UI: botón, lista, contenedor
12	Torque angular	Nivel 2: generador preguntas y opción múltiple al azar
13	Deducción de segunda parte de modelo de caída libre	Nivel 3: Reusar nivel 1 con distancia al azar
14	Preguntas de Repaso para nivel 2 y nivel 4	Pantalla de inicio. Nivel 4 (pregs). Pantalla final
15	Prueba del juego	Depuración de errores
16	Avances del juego de Física	Dudas y ajustes (código)
17	Reporte proyecto (parte 1)	Desc. pantallas y videos del juego
18	Jugadores invitados (nivel medio superior)	Encuestas finales
19	Reporte proyecto (parte 2)	Desarrollo y conclusiones
20	Presentación de juegos	Comentarios finales

El nivel 1 y nivel 3 del juego se consideran las siguientes dinámicas (Fig. 6): a) el personaje (objeto rígido) inicia en reposo y al pulsar la tecla *enter*, dicho objeto cae libremente hasta impactarse contra el piso o caer dentro de un barril (objetos estáticos); b) el peldaño se desplaza verticalmente usando las teclas de cursor hacia arriba y abajo; c) al inicio de cada jugada, el barril se desplaza a una posición fija de forma aleatoria; d) el usuario puede girar el ángulo del peldaño usando las teclas izquierda y derecha, con la intención de lograr que cuando rebote el objeto, el ángulo del peldaño contribuya a que dicho objeto, siguiendo una trayectoria parabólica, caiga dentro del barril para lograr así, una mayor puntuación de juego.

Nivel: 1	Reto: $t = 1.2s$
Tiempo restante: 60s	Score: 75

Nivel: 3	Reto: $d = 8.6m$
Tiempo restante: 30s	Score: 135

Figura 5. Tablero de juego para: a) nivel 1; b) nivel 3.

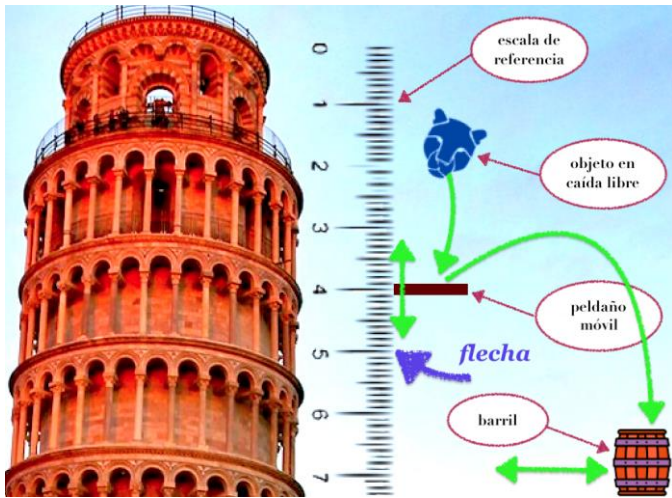


Figura 6. Principales elementos y controles del juego.

Las pantallas y escenas para el nivel 2 y nivel 4 son idénticas y ofrecen un cuestionario usando los controles UI de Godot: contenedores, textos, iconos, listas y botones. A partir de un banco de preguntas correspondiente al nivel de juego anterior, se selecciona al azar y se presenta una pregunta con cuatro posibles respuestas (opciones) sobre los temas de cinemática, movimiento uniformemente acelerado y caída libre de cuerpos físicos ilustrados y aplicados en la dinámica del juego. El jugador elige una de las opciones y se marca si la respuesta es correcta o no (iconos ☒ y luego pulsar el botón continuar (Fig. 7).

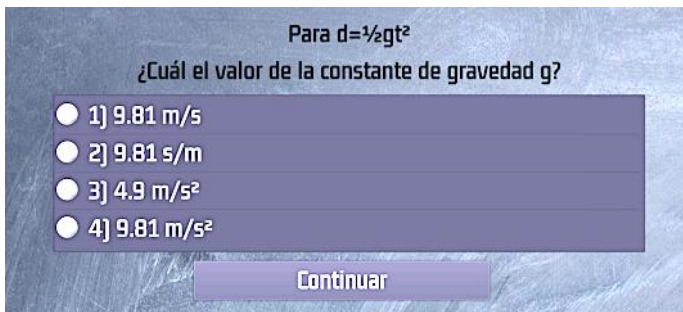


Figura 7. Pregunta temática aleatoria sobre caída libre y movimiento uniformemente acelerado.

De forma sintética, el algoritmo de una jugada es el siguiente: 1) se activa música de fondo e inicia tiempo de juego; 2) muestra personaje en posición inicial, se elige al azar tiempo de caída t_{reto} y posición del barril; 3) muestra reto y coloca barril; 4) usuario mueve peldaño con teclas cursor; 5) se espera evento de entrada de tecla *enter* para iniciar *timer*, sonido y caída del objeto; 6) se espera evento de colisión objeto-peldaño para detener *timer*, reproduce sonido choque y calcula tiempo de caída; 7) se estima error para obtener puntuación de jugada; 8) se espera evento de colisión objeto-barril para activar sonido/efecto, indicar con

flecha distancia correcta del reto asignado y se actualiza puntuación; 9) se espera evento de colisión objeto-piso para activar sonido/efecto, borrar personaje y regresar al paso 2; 10) se espera evento temporizado hasta agotar tiempo de juego para detener la jugada y llamar al siguiente nivel de juego o terminar.

3.3. Matemización y pruebas del fenómeno físico.

Un proceso central para el desarrollo del juego fue, desde el punto de vista didáctico, la matemización del fenómeno físico de caída libre de un objeto, partiendo del reposo y despreciando la resistencia del aire. Para ello fue esencial establecer una simulación del fenómeno lo más eficiente y fidedigna posible para aproximar de forma correcta y precisa las magnitudes de las distancias y tiempos correspondientes al modelo matemático de caída libre, corroborando además si el motor de física de Godot se ajustaba al modelo. En las sesiones 7-10 (Tabla 1), se idearon los cálculos tiempo-distancia, se realizaron pruebas de medición de tiempos y se calibró el modelo para indicar las distancias unitarias en la regla de referencia (Fig. 6) mapeando distancias de unidades métricas a píxeles. Replicando la experimentación del plano inclinado de Galileo [5], considerando las mismas mediciones históricas, es decir, para $t=1s$ la regla debía marcar $d=1u$, para $t=2s$ es $d=4u$ y para $t=3s$ es $d=9u$ (ver escala en Fig. 6). Un hallazgo para el motor de juegos, fué una ligera degradación cuando había varias aplicaciones corriendo en la PC, notoria el tener $t=3s$, por unos cuantos píxeles, no se alcanzaba $d=9u$, por ello se recomendó a los estudiantes al correr el juego cerrar primero todas las aplicaciones activas. Salvo ese detalle, gracias a la adecuada matemización del problema fue posible estimar correctamente (*pixel perfect*) la distancia correspondiente para cada tiempo-reto y estimar el error obtenido al compararse contra la distancia de impacto objeto-peldaño, que fue elegida por el jugador.

Matemizando formalmente el problema de caída libre de objetos, al despreciar la resistencia del aire y partiendo del reposo, tenemos que la distancia recorrida por el objeto será:

$$d = \frac{1}{2}gt^2$$

Dado que la intención del juego es descubrir dicha fórmula (aprendizaje inductivo), es posible reescribir dicha fórmula en dos partes, una parte constante y la otra en función del tiempo:

$$d = k \cdot f(t)$$

En el primer nivel del juego se pretende deducir solamente la relación de proporcionalidad dada por $f(t)$, proporcionando para ello un reto lúdico consistente en encontrar la distancia recorrida para un tiempo-reto aleatorio dado (t_{reto}). El jugador deberá estimar, en fracciones de segundo, cómo desplazar el peldaño, mediante las teclas cursor, hasta una distancia apropiada donde deberá impactar objeto-peldaño en t_{reto} . A la vez, debe girar el peldaño para lograr que el objeto rebote y caiga dentro del barril (B_{hit}), logrando así una mayor puntuación. Al finalizar la jugada aparecerá una flecha en la regla indicando la

distancia correcta. De esta forma es posible partir sin conocimientos previos del modelo y mediante esta dinámica de juego, ir reforzando el aprendizaje hasta lograr aproximar un modelo mental más cercano al modelo físico correcto. El tiempo de impacto objeto-peldaño (t_{hit}) y el tiempo del reto (t_{reto}) son utilizados para estimar el error y con ello calcular la puntuación de cada jugada (máx. 100 pts), según:

$$score = 50 \cdot \left(1 - \frac{|t_{reto} - t_{hit}|}{t_{max}}\right) + 50 \cdot B_{hit}$$

Con esta dinámica, luego de varias jugadas se espera que el jugador deduzca $f(t)$, por ejemplo para $t_{reto}=[1,2,3]$, si un jugador sólo recuerda $d=[2,4]$, deducirá de forma incorrecta $f(t) = 2t$. Sin embargo, si observa que $d=[2,4,9]$, podrá deducir que dicha parte de la fórmula corresponde a $f(t) = t^2$.

De acuerdo con el algoritmo de juego presentado arriba, al agotarse el tiempo del primer nivel, aparecerá una pregunta al azar para corroborar el conocimiento adquirido (Fig. 7). En caso de que el jugador falle, lo cual indica que aún no puede inferir correctamente la relación de proporcionalidad tiempo-distancia, regresará al primer nivel para reforzar una mejor comprensión del fenómeno. Si contesta correctamente, avanza al siguiente nivel donde debe deducir la magnitud de la constante k , y por consecuencia, la constante de aceleración gravitacional g .

De forma análoga, en el nivel 3 se genera una distancia aleatoria (d_{reto}) y el jugador debe mover el peldaño para aproximarla a la distancia dada ahora en unidades métricas (Fig. 6). Al agotarse el tiempo, una pregunta al azar verifica el conocimiento logrado (Fig. 7). Por ejemplo, para $d_{reto}=4.9m$, deberá ubicar la repisa en $d_u=1u$ (regleta), correspondiendo a $t_{hit}=1s$, es decir:

$$d = \frac{g}{2} t^2 = k \cdot f(1s) = k \cdot 1s^2 = 4.9m,$$

$$\text{o sea } k = \frac{d}{t^2} = \frac{4.9m}{1s^2} = 4.9m/s^2,$$

$$\text{y por lo tanto, } g \approx 9.8m/s^2.$$

4. RESULTADOS

La metodología propuesta, el soporte del mentor y del motor de juegos permitieron que estudiantes sin experiencia previa de programación, lograran ser capacitados (curva de aprendizaje) en alrededor de 9 semanas (sesiones 3-11 en Tabla 1). La integración de métodos ágiles y estrategias didácticas (Fig 3) permitió crear el juego serio en alrededor de 20 semanas. Al finalizar el curso de Física, se evaluó en base a los criterios y la rúbrica descrita en [16], obteniendo como resultado un 100% de eficiencia terminal de un proyecto con altas calificaciones (93 ± 7) (Fig. 8). El juego terminado ocupó alrededor de 2-4 Mb y 300 líneas de código. La experiencia didáctica contó con un buen nivel de aceptación, una percepción más favorable de la Física y una nutrida participación como lo constatan las encuestas, evaluaciones y evidencias reportadas de forma detallada en [16].



Figura 8. Ejemplos de escenas de juego creadas por estudiantes.

5. LIMITANTES Y TRABAJO A FUTURO

El método propuesto solo se aplicó en dos grupos pequeños, por tanto, una posible limitante, debido a que requiere una supervisión personalizada, tal vez no sea viable para aplicarlo en grupos más numerosos. La implementación de gran parte de la lógica de los niveles de juego son muy similares, por lo que es posible reducir hasta un 25% de código duplicado. Sin embargo, introducir técnicas avanzadas de programación para realizar dicha mejora puede complicar e incrementar la curva de aprendizaje para estudiantes sin experiencia previa de programación. Por otro lado, es posible usar: un control de versiones (GitHub); incorporar de forma colegiada un banco de preguntas más extenso y por nivel escolar; agregar más temas curriculares de un curso de Física como manejo de vectores, inercia, fricción y tiro parabólico. Algo interesante que soporta Godot es la posibilidad de migrar el juego a una plataforma móvil para habilitar la descargar y ejecución del videojuego en teléfonos inteligentes (*smartphones*).

6. CONCLUSIONES

A diferencia de un método tradicional, deductivo, descriptivo pasivo y basado en retención de conceptos y fórmulas típico de la enseñanza de la Física, se exploró un enfoque matemático-computacional, mezclando ciencia e ingeniería, con estrategias de aprendizaje activo, inductivo, basado en problemas, metacognitivo, analítico (matematización) y algorítmico (programación en vivo) para aplicar conceptos teóricos vistos en la clase de Física a problemas simulados en un juego serio, de una forma supervisada, gradual, metódica, creativa, activa y lúdica (Fig. 3), siguiendo una cuidadosa planeación didáctica (Tabla 1), bajo la mentoría y supervisión de un experto usando un motor de juegos abierto, multiplataforma, amigable, para obtener una simulación fidedigna y eficiente de los modelos cinemáticos involucrados en la dinámica de los objetos de un juego serio para lograr los objetivos de aprendizaje planteados.

Agradecimientos.

Al Tecnológico Nacional de México, la academia de Ciencias Básicas, grupo de Física del Tecnológico de Chihuahua y grupo de videojuegos de la Universidad Autónoma de Chihuahua.

Referencias.

- [1] M. Marusic & J. Slisko, Increasing the attractiveness of school physics: the effects of two different designs of physics learning, *Revista Mexicana de Física*, E 58(1): 75–83, 2012.
- [2] A. Aondofa, & V. Atindi, Effectiveness of the use of Gamification by teachers of Physics in secondary schools in Buruku local government area of Benue State, *IJAMR*, 6(8):6-11, 2022.
- [3] A. Aprilia et al, Attitudes and beliefs toward physics: A mini review, *Rev. Mex. Física*, E 20(1):1–8, 2023.
- [4] S. Altun, & Z. Morkoyunlu, The use of error based activities to improve the mathematization competency, *Int. J. Prog. Education*, 19(3):134-148, 2023.
- [5] J. Van Hemmen, Mathematization of nature: how it is done, *Biol Cybern*, Vol. 115, pp. 655–664, 2021.
- [6] A. Azzouzi, et al, Physics mathematization: Teachers observations on the application of ICT, *Int. Conf. Intel. Sys. Comp. Vision (ISCV)*, pp. 1-5, 2022.
- [7] J. Bilbao, et al, Introducing computational thinking and algebraic thinking in the European educational systems, *Int. J. Education and Info. Tech.*, 18:11-19, 2024.
- [8] L. Ivanjek, et al, Enhancing mathematization in Physics education by digital tools, *Physics Education Today*, pp. 35-53, Springer Nature, 2024.
- [9] Statista, “Géneros de videojuegos más jugados en México”, [en línea], disponible: <https://shorturl.at/zpGpq> (2024).
- [10] Godot, Motor de juegos de código abierto Godot [en línea], disponible: <https://godotengine.org> (2024).
- [11] H. Ben & R. Ben, The educational effectiveness of serious games, *Mediations and Mediatizations*, vol. 2, pp. 131–155, 2019.
- [12] J. Tomalá, et al, Serious Games: Review of methodologies and games engines for their development, *Iberian Conf. Info. Sys. and Tech. (CISTI)*, pp. 1-6, 2020.
- [13] S. Gurbuz & M. Celik, Serious games in future skills development: A systematic review of the design approaches, *Comp. App. Engineering Education*, 30(5): 1591-1612, 2022.
- [14] A. Van der Linden, et al, Learning Newtonian mechanics with an intrinsically integrated educational game, *J. Comp. Assisted Learning*, pp. 1–11, 2024.
- [15] M. Prince & R. Felder, Inductive teaching and learning methods: Definitions, comparisons, and research bases, *J. Engr. Education*, 95(2): 123–138, 2006.
- [16] A. Pacheco, Desarrollo del pensamiento computacional aplicando programación en vivo con retos desvanecidos en la creación de un juego serio para Física, *Revista EAI*, 2024 [por publicar].
- [17] M. Carr & J. Verner, Prototyping and software development approaches, University of Hong Kong, pp. 319-38, 1997.
- [18] C. Stefanou, *Encyclopedia of Information Systems*, Academic Press, 2003.