Treball de Fi de Grau

Títol

Mortis Roll. Diseño y desarrollo de un videojuego roguelike con estética retro

Autoria

Iván Fernández Fresneda

Professorat tutor

Gustavo Henrique Oliveira De Almeida

Grau

Comunicació Audiovisual	
Periodisme	
Publicitat i Relacions Públiques	
Comunicació Interactiva	X
Comunicació de les Organitzacions	

Tipus de TFG

Projecte	X
Recerca	

Data

Del 26 al 29 de maig de 2026	X
24 de juliol de 2026	

Full resum del TFG

Títol del Treball Fi de Grau:

Català:	Mortis Roll. Disseny i desenvolupament d'un videojoc roguelike amb estètica retro
Castellà:	Mortis Roll. Diseño y desarrollo de un videojuego roguelike con estética retro
Anglès:	Mortis Roll. Design and development of a roguelike video game with 32-bit retro aesthetics

Autoria:		Iván Fernández Fresneda		
Professorat tutor:		Gustavo Henrique Oliveira De Almeida		
Curs:	2025/26	Grau:	Comunicació Audiovisual	
			Periodisme	
			Publicitat i Relacions Públiques	
			Comunicació Interactiva	X
			Comunicació de les Organitzacions	

Paraules clau (mínim 3)

Català:	Comunicació interactiva, Disseny de videojocs, Estètica digital, Experiència d'usuari (UX)
Castellà:	Comunicación interactiva, Diseño de videojuegos, Estética digital, Experiencia de usuario (UX)
Anglès:	Interactive communication, Video game design, Digital aesthetics, User experience (UX)

Resum del Treball Fi de Grau (extensió màxima 100 paraules)

Català:	Aquest treball documenta el disseny i desenvolupament en solitari d'un prototip de videojoc roguelike. El projecte explora la recreació tècnica de l'estètica de la primera PlayStation, utilitzant les seves limitacions gràfiques característiques (vertex jittering, dithering) com a eina expressiva per a generar una atmosfera. Mitjançant un enfocament pràctic utilitzant Godot Engine i Blender, s'han implementat sistemes de combat, mecàniques de progressió i un flux de treball integral d'art tridimensional low-poly.
Castellà:	Este trabajo documenta el diseño y desarrollo en solitario de un prototipo de videojuego roguelike. El proyecto explora la recreación técnica de la estética de la primera PlayStation, utilizando sus limitaciones gráficas características (vertex jittering, dithering) como herramienta expresiva para generar una atmósfera. Mediante un enfoque práctico utilizando Godot Engine y Blender, se han implementado sistemas de combate, mecánicas de progresión y un flujo de trabajo integral de arte tridimensional low-poly.
Anglès:	This paper documents the solo design and development of a roguelike video game prototype. The project explores the technical recreation of the original PlayStation's aesthetics, using its characteristic graphical limitations (vertex jittering, dithering) as an expressive tool to generate an atmosphere. Through a practical approach using Godot Engine and Blender, combat systems, progression mechanics, and a comprehensive low-poly 3D art workflow have been implemented.

Sumario

Lista de figuras **2**
1. Introducción **3**
1.1. Motivación 4
1.2. Objetivos generales 5
1.3. Objetivos específicos 5
2. Género y referentes **6**
2.1. Definición de Roguelike 6
2.1.1. La filosofía del Roguelike 7
2.2. Estética PS1 8
2.3. Referentes del proyecto 10
2.3.1. Mecánicas y progresión 10
2.3.2. Estética retro y low-poly 12
2.3.3. Ambientación 13
3. Diseño de juego (GDD) **15**
3.1. Resumen del proyecto (Elevator Pitch) 15
3.2. Género y Ambientación 15
3.3. El Jugador y la Estructura del Mundo 15
3.4. Ciclo de Juego (Core Loop) 16
3.5. Objetivos, Habilidades y Progresión 16
3.6. Apartado Visual e Interfaz (UX) 17
3.7. Sistema de Audio 18
4. Presupuesto **18**
4.1. Justificación de las partidas y fuentes de financiación 20
5. Metodología y Desarrollo **21**
5.1. Diseño iterativo 21
5.2. Motor de juego 22
5.3. Creación de assets 3D 22
5.4. Flujo de trabajo 23
5.5. Programación de sistemas 23
5.6. Arte 29
5.7. Postprocesado 32
6. Mejoras futuras **34**
6.1. Animaciones y Game Feel 35
6.2. Profundidad en el comportamiento enemigo 35
6.3. Generación procedimental avanzada 35
6.4. Metaprogresión y narrativa fragmentada 36
7. Conclusiones **36**
Referencias **37**

Videojuegos.....**38**
Apéndice A. El videojuego Mortis Roll..... **38**
Apéndice B. Glosario.....**39**

Lista de figuras

Figura 1 Estructura de habitaciones procedimentales e interfaz de objetos en The Binding of Isaac.....10

Figura 2 Lectura espacial de amenazas y mecánicas de esquiva en tiempo real en Enter the Gungeon.....11

Figura 3 Estética retro, baja poligonalidad y texturizado en Vagrant Story..... 13

Figura 4 Estilo surrealista y diseño de personajes oníricos en Hylics..... 14

Figura 5 Cápsula geométrica básica como "placeholder" del jugador.....24

Figura 6 Representación provisional de enemigos mediante cápsulas.....25

Figura 7 Configuración de áreas de detección (Hitboxes) en el combate..... 26

Figura 8 Disposición espacial de la sala del tesoro y el cofre interactivo.....28

Figura 9 Modelado de geometría "low-poly" para personajes y armamento..... 30

Figura 10 Atlas de texturas y archivos de imagen base para los modelos 3D..... 31

Figura 11 Modelos tridimensionales con el texturizado final aplicado..... 32

Figura 12 Aplicación de shaders de postprocesado para la emulación visual de 32 bits.....33

1. Introducción

Este Trabajo de Fin de Grado, desarrollado en el contexto del grado en Comunicación Interactiva de la Universitat Autònoma de Barcelona (UAB), consiste en el diseño, justificación y producción de un prototipo jugable de un videojuego titulado *Mortis Roll*. El proyecto se enmarca dentro del género *roguelike* (Harris, 2021) de acción en tiempo real, presentando un estilo artístico de baja poligonalidad que se inspira directamente en las características técnicas de la quinta generación de consolas (PlayStation 1). Mi propósito con esta memoria es exponer el proceso completo de desarrollo independiente y analizar cómo los sistemas y la interfaz visual se comunican de forma transparente con el usuario.

A nivel teórico, el proyecto se centra en cómo las limitaciones técnicas de la era de los 32 bits como la baja resolución, el tramado de color (*dithering*) o el temblor de vértices (*vertex jittering*), se pueden reaprovechar hoy en día a través de código y *shaders* personalizados en el motor de juego. Mi intención es demostrar que estos antiguos fallos de hardware no están obsoletos, sino que funcionan como herramientas capaces de construir una atmósfera de misterio. Pese a que hoy en día la industria suele buscar el máximo realismo gráfico, mi objetivo es probar que una estética retro bien cuidada, sumada a unas mecánicas dinámicas, es perfectamente capaz de atrapar al jugador.

Con tal de afrontar el desafío que supone desarrollar un proyecto en solitario, me he encargado de gestionar todo el flujo de trabajo de forma multidisciplinar. Este proceso abarca desde darle forma a los modelos en Blender (2024), hasta programar toda la lógica en Godot 4.6 (2024), incluyendo el movimiento, la inteligencia artificial de persecución de los enemigos y el sistema de combate mediante colisiones por capas (*hitboxes*). Entiendo este trabajo como el cierre de un largo camino de aprendizaje universitario. Es una oportunidad ideal para poner en práctica las teorías clásicas de diseño de juego (Schell, 2014).

Para plasmar todo este proceso, la memoria se ha estructurado de manera progresiva. En primer lugar, se explicará el marco teórico, definiendo el género y los referentes mecánicos y estéticos del proyecto. Posteriormente, se detallará la metodología de diseño iterativo que se ha seguido para crear el Producto Mínimo

Viable (MVP) y se expondrá el Documento de Diseño de Juego (GDD). Para concluir, se presentará un presupuesto realista adaptado al mercado laboral actual y se comentarán las futuras mejoras pensadas para convertir este prototipo en un producto comercial completo.

A lo largo de este proyecto, he concebido el desarrollo técnico del videojuego y la confección de esta memoria académica como dos procesos inseparables que se retroalimentan constantemente. Por este motivo, he decidido entrelazar estructuralmente ambos aspectos a lo largo del documento, evitando separar de forma rígida la teoría del diseño de juego de su aplicación práctica. Al utilizar la primera persona en la redacción, mi intención es reflejar la naturaleza de este reto en solitario y mostrar, de manera transparente, cómo cada decisión estética, cada obstáculo técnico superado en el motor de juego y cada solución adoptada forman parte de un proceso integrado de aprendizaje y práctica.

1.1. Motivación

La motivación que impulsa el desarrollo de este trabajo nace de una fascinación personal por la estética de la quinta generación de consolas y el desafío técnico que supone el desarrollo en solitario así como de mi gusto por los videojuegos del género roguelike. Tras años de contacto con el mundo de los videojuegos, he desarrollado un interés especial por cómo las limitaciones técnicas de la era de los 32 bits, que se traduce en gráficos de baja poligonalidad y texturas pixeladas, pueden ser utilizadas hoy en día no como una herramienta expresiva para generar atmósferas de horror y extrañeza.

Por otra parte, mi pasión por los juegos roguelike ha sido un pilar fundamental para decidirme por este proyecto. Siempre me ha gustado cómo el uso de niveles aleatorios y el hecho de tener que empezar de cero al morir hacen que cada partida sea una experiencia distinta, la cual te obliga a estar siempre aprendiendo y adaptándote. Pese a que soy consciente de lo difícil que es equilibrar el azar para que no sea injusto, creo que es el marco ideal para un juego de estas características, ya que permite que la suerte no sea un muro insalvable, sino un obstáculo que el jugador debe aprender a sortear con su estrategia. Mi intención es

que el usuario no vea el fracaso como algo malo, sino como una parte necesaria para mejorar y entender mejor el juego cada vez que lo intenta.

Por último, entiendo este trabajo como el cierre de un largo recorrido de aprendizaje en el grado de Comunicación Interactiva, en el que he procurado explorar todas las áreas del desarrollo de videojuegos. Mi intención es enfrentarme al reto de gestionar yo solo todo el flujo de trabajo, el cual abarca desde dar forma a los modelos en Blender hasta programar el corazón del juego en Godot 4.6. Con ello, quiero demostrar que, pese a las limitaciones de trabajar en solitario, se puede alcanzar un resultado profesional si se tiene una idea visual clara y una planificación constante.

1.2. Objetivos generales

Diseñar y desarrollar un prototipo jugable de un videojuego de mazmorras en 3D, el cual mezcle la profundidad de los roguelikes con una ambientación de misterio y ritualismo.

Pese a que hoy en día la industria suele buscar el máximo realismo gráfico, mi intención es demostrar que se puede generar inmersión a través de una estética de 32 bits bien cuidada y unas mecánicas de juego que resulten divertidas por sí mismas. Con tal de asegurar que todo el proyecto tenga una identidad coherente, asumiré personalmente todo el proceso de creación, utilizando Blender para el apartado artístico y Godot 4.6 para la programación, buscando siempre que el resultado final sea un producto sólido y funcional.

1.3. Objetivos específicos

1.3.1 Lograr que el juego tenga el aspecto visual de la PlayStation 1, el cual se basa en usar modelos con pocos polígonos y texturas pixeladas, aplicando efectos que imiten los "fallos" gráficos de esa época.

1.3.2 Diseñar una lógica de juego donde los objetos sean los que dicten qué puedes hacer en cada momento, pese a que el jugador tenga siempre la última palabra al decidir cómo usar esos resultados.

1.3.3 Desarrollar un sistema de juego donde morir sea parte del aprendizaje, con tal de que cada partida sea un reto nuevo y el jugador sienta que va mejorando su estrategia poco a poco.

1.3.4 Establecer un flujo de trabajo propio entre Blender y Godot 4.6 que me permita crear mis propios personajes y escenarios y meterlos en el juego sin complicaciones técnicas innecesarias.

1.3.5 Crear un catálogo de objetos que el jugador pueda encontrar, los cuales modifiquen las reglas del juego y le den ventajas para que el azar no sea lo único que importe.

2. Género y referentes

2.1. Definición de Roguelike

El término roguelike define a un subgénero de los videojuegos que se distingue por una serie de reglas que rompen con la estructura lineal de muchos otros géneros (Harris, 2021). Aunque su nombre proviene de un juego de los años ochenta, hoy en día se utiliza para describir un tipo de juego basado en la repetición y el aprendizaje constante a través del error.

El esfuerzo más reconocido por estandarizar estas reglas se produjo en 2008 con la conocida como Interpretación de Berlín (Berlin Interpretation, s.f.). En esta convención internacional de desarrollo, se establecieron los 'factores de alto valor' que definen la pureza del género, sentando unas bases académicas que, aunque han evolucionado, siguen siendo el núcleo del diseño actual.

Para que un juego sea considerado dentro de este género, debe cumplir con varias características (Harris, 2021):

- Muerte permanente (*permadeath*): Es la característica más representativa, donde el fallo del jugador supone el fin de la partida y la pérdida del progreso

actual. Esta mecánica es la que otorga un peso real a cada elección y mantiene la tensión durante toda la sesión de juego.

- Entornos procedimentales: Los niveles, la colocación de los enemigos y los objetos que aparecen se generan de forma aleatoria en cada intento. Esto se hace con tal de asegurar que el jugador no pueda avanzar simplemente memorizando un camino, sino que tenga que entender las reglas del mundo para saber adaptarse a lo que venga.
- Complejidad de sistemas: El juego suele presentar una gran variedad de objetos y mecánicas que interactúan entre sí. El aprendizaje de estas sinergias es lo que permite progresar, convirtiendo la información en el recurso más valioso que el usuario conserva tras cada muerte.

El roguelike no busca ofrecer un camino fácil, sino un sistema justo donde la victoria depende de la experiencia que se va acumulando intento tras intento.

2.1.1. La filosofía del Roguelike

Lo que hace que un roguelike enganche es cómo maneja esa sensación de no saber qué va a pasar, la cual acaba siendo el motor de todo el juego. Como señala Harris (2021), la esencia y el atractivo de estas obras radican precisamente en su "tremenda dificultad, imprevisibilidad y la muerte permanente del personaje". El progreso real del jugador no se mide tanto en el incremento de estadísticas de su avatar, sino en su propio entendimiento de los sistemas de juego y progresión. Pese a que al morir te quiten los objetos o te devuelvan al inicio, el conocimiento sobre los peligros y las mecánicas se queda contigo; esto hace que cada derrota se sienta como una forma de entrenamiento con tal de que, en el siguiente intento, seas capaz de llegar un poco más lejos. Es la aceptación de este ciclo lo que genera una experiencia de juego que, en palabras del propio Harris, resulta "difícil, salvajemente sorprendente, extrañamente caótica y muy divertida" (Harris, 2021).

Según las teorías de Schell (2014) sobre el diseño de juegos, una experiencia es verdaderamente satisfactoria cuando logra mantener al jugador en un estado de reto constante que no llega a ser frustrante. Este delicado equilibrio, académicamente conocido como el estado de flujo o flow (Chen, 2007), se consigue en el roguelike mediante un riguroso y constante ciclo de "prueba y error". Para que este bucle

funcione y el usuario acepta el castigo de volver al inicio sin abandonar el juego, el diseño debe ser sumamente transparente; es decir, lo suficientemente claro como para que, al morir, el jugador entienda qué ha hecho mal y sienta que la culpa es de sus propias decisiones tácticas, no de una aleatoriedad injusta.

Con tal de mantener este interés a largo plazo y mitigar la fricción de la derrota, es vital que el juego ofrezca recompensas indirectas (lo que en diseño se conoce como metaprogresión (Chen, 2007). Aunque el personaje pierda su poder inmediato, el sistema debe proporcionar el desbloqueo de nuevos elementos, reliquias o fragmentos narrativos que enriquezcan las futuras partidas. Todo ello sumado a la recompensa intrínseca más importante: la sensación de que el jugador es genuinamente más hábil e inteligente que hace diez minutos.

2.2. Estética PS1

El estilo visual inspirado en la quinta generación de consolas no es solamente reducción de la fidelidad gráfica, sino que es un conjunto de fenómenos técnicos que definen su identidad visual. Estas particularidades nacen de las limitaciones del hardware original de la década de 1990, las cuales hoy se replican de forma voluntaria mediante código y shaders para lograr una atmósfera de juego concreta .

Los pilares que definen este lenguaje visual son los siguientes (Ilett, 2020) :

- Vertex Jittering (Temblor de vértices): Debido a que las consolas antiguas no tenían una precisión matemática elevada para los gráficos 3D, la posición de los vértices se redondeaba constantemente a números enteros. Esto genera un movimiento nervioso en los objetos al moverse la cámara, el cual aporta una sensación de inestabilidad y fragilidad a todo el entorno tridimensional.
- Affine Texture Mapping (Mapeado afín): Pese a que los motores modernos corrigen la perspectiva de las imágenes sobre los objetos, el hardware de 32 bits proyectaba las texturas de forma lineal. Este fallo técnico provoca que las texturas parezcan deformarse o "nadar" sobre las superficies inclinadas, rompiendo la rigidez visual propia de los motores actuales.

- Dithering y profundidad de color: Para compensar la poca variedad de colores que permitía la tecnología de la época, se utilizaba el *dithering* o tramado de puntos. Esta técnica crea un patrón de ruido visual en las sombras, con tal de disimular los saltos bruscos de color y generar una textura granulada que refuerza el ambiente de misterio.
- Filtrado de texturas "Nearest Neighbor": A diferencia del suavizado que vemos en los juegos actuales, este estilo muestra los píxeles de forma directa y definida. Al trabajar con resoluciones muy bajas, el píxel se convierte en el protagonista del detalle visual, obligando a que el diseño de los escenarios sea claro y directo.

Para comprender los motivos exactos de estos fenómenos visuales, tenemos que analizar la arquitectura técnica del hardware original que los provocaba (Pezzi, 2024). A nivel estructural, la consola contaba con unas limitaciones de memoria muy severas, disponiendo únicamente de 2 MB de RAM para la lógica del juego y 1 MB de VRAM para almacenar las texturas (Pezzi, 2024).

El origen del *Affine Texture Mapping* y las deformación de las texturas se deben a que el chip gráfico de la PlayStation original estaba diseñado, principalmente, para trabajar con gráficos en 2D (Pezzi, 2024). Al intentar mostrar un objeto tridimensional en la pantalla, la consola "olvidaba" la información de profundidad; es decir, no tenía forma de calcular qué partes del modelo estaban más cerca o más lejos de la cámara. Debido a la falta de esta referencia espacial, el sistema se veía obligado a pegar las imágenes sobre los objetos como si fueran recortes planos. Esto provocaba que, al moverse por el escenario o al cambiar el ángulo de visión, las texturas no lograran adaptarse a la perspectiva real de los volúmenes, generando esa característica sensación de que las imágenes se estiran sobre las superficies (Pezzi, 2024).

Por otro lado, el característico temblor de vértices derivaba directamente de la ausencia de una Unidad de Coma Flotante (FPU) en su procesador. Al no disponer de soporte nativo para calcular números fraccionarios con precisión ni contar con precisión de subpíxeles, el sistema forzaba las coordenadas de los vértices a encajar drásticamente en la cuadrícula de números enteros más cercana de la pantalla (Pezzi, 2024).

Asimismo, esta falta de precisión matemática ocasionaba las llamadas "fisuras en T" (*T-Junction gaps*), un defecto geométrico donde los cálculos de rasterización fallaban al procesar polígonos adyacentes, generando pequeños huecos y destellos visibles entre los bordes de los modelos tridimensionales. Finalmente, cabe destacar que el *dithering* aplicado a los colores de 15 y 24 bits no era programado manualmente por los desarrolladores, sino que consistía en una función integrada directamente en el hardware de la consola para camuflar las carencias visuales de la paleta (Pezzi, 2024).

2.3. Referentes del proyecto

Para el desarrollo de Mortis Roll, se han tomado como guía diferentes obras que destacan en los pilares fundamentales del juego: la jugabilidad, el apartado visual y la atmósfera.

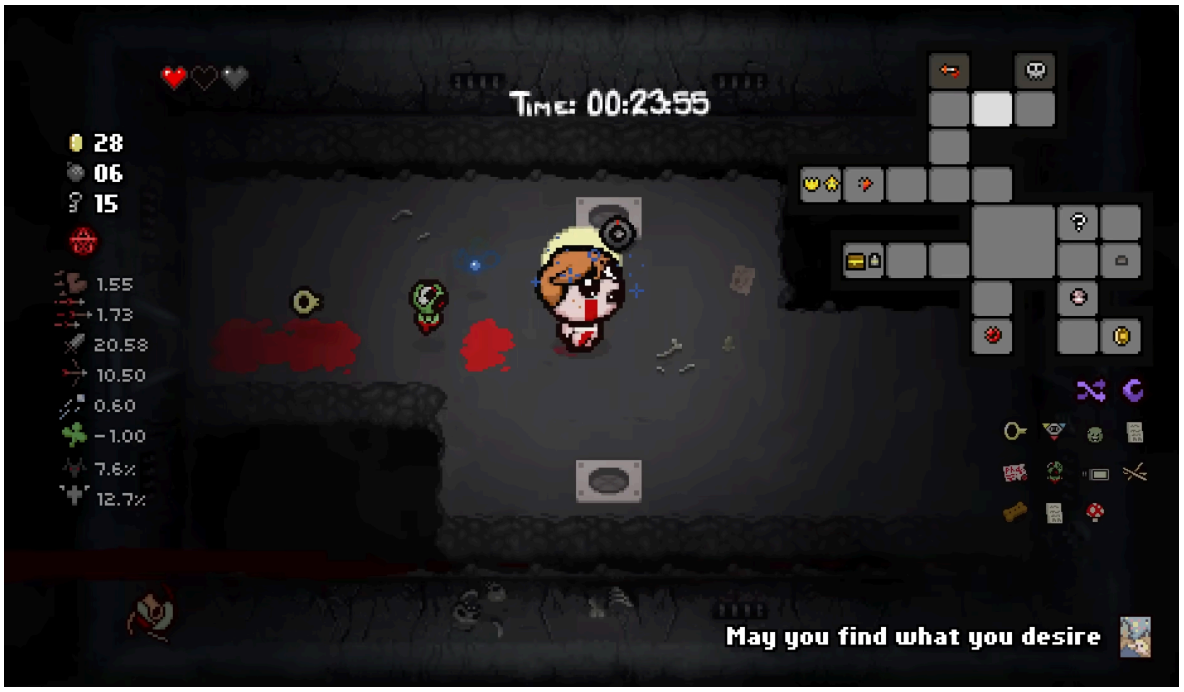
2.3.1. Mecánicas y progresión

En el plano de la estructura del mundo y la gestión del inventario, el referente más directo es *The Binding of Isaac* (2011). De esta obra se ha tomado la idea de la exploración libre a través de una mazmorra con una disposición aleatoria de estancias interconectadas por pasillos, donde cada sala funciona como un espacio interactivo cerrado que desafía al usuario de manera individual (Ver figura 1).

Pese a que *Mortis Roll* introduce una perspectiva tridimensional e isométrica, la lógica de progresión macro copia la esencia de este referente: el azar dicta qué objetos y reliquias místicas aparecen en los cofres de cada partida. El éxito de la incursión no depende de la acumulación lineal de estadísticas, sino de la habilidad del jugador para entender y adaptar su estrategia en tiempo real a las sinergias que se generan entre los modificadores recolectados, convirtiendo el conocimiento cognitivo del entorno en el recurso más valioso que se conserva tras la muerte permanente.

Figura 1

Estructura de habitaciones procedimentales e interfaz de objetos en The Binding of Isaac



Nota. Captura de pantalla ilustrativa de la navegación espacial sala por sala y el sistema de acumulación de modificadores pasivos. Tomada del videojuego *The Binding of Isaac* (Edmund McMillen, 2011).

En lo que respecta al flujo micro y las dinámicas del combate, el proyecto toma como pilar fundamental a *Enter the Gungeon* (2016). Al abandonar de forma estricta las mecánicas pausadas, el título aborda la tensión de los enfrentamientos basados en reflejos, el posicionamiento físico y la lectura de las amenazas. De este referente se ha extraído la mecánica de esquiva, la cual otorga al protagonista fotogramas (*frames*) de invulnerabilidad temporal frente a las cajas de colisión hostiles (*hitboxes*) (Ver figura 2).

Figura 2

Lectura espacial de amenazas y mecánicas de esquiva en tiempo real en Enter the Gungeon



Nota. Captura de pantalla que ilustra el combate de acción directa, el posicionamiento táctico del avatar y el uso de mecánicas de movilidad con marcos de invulnerabilidad. Tomada del videojuego *Enter the Gungeon* (Dodge Roll, 2016).

2.3.2. Estética retro y low-poly

El pilar visual del proyecto se apoya totalmente en la era de los 32 bits, tomando como referencia técnica el juego Vagrant Story (2000) de la primera PlayStation. Este título es un ejemplo excelente de cómo aprovechar una geometría sencilla y texturas de baja resolución para crear escenarios que se sientan solemnes y detallados (Ver figura 3). Al igual que en ese referente, en Mortis Roll se utiliza un modelado *low-poly* agresivo, el cual permite centrar la atención en el estilo artístico y en la composición de la escena en lugar de en la carga técnica.

Figura 3

Estética retro, baja poligonalidad y texturizado en Vagrant Story



Nota. Captura de pantalla ilustrativa del modelado *low-poly* y la atmósfera de 32 bits de la primera PlayStation. Tomada del videojuego *Vagrant Story* (Square, 2000).

Pese a las limitaciones de usar pocos polígonos, se busca que los enemigos y las mazmorras sean icónicos. Se ha optado por un texturizado que se aleja del realismo para acercarse a un estilo más marcado, similar a lo que se puede ver en proyectos independientes actuales que rescatan la estética de PS1. El objetivo es que los "fallos" gráficos, como el baile de los vértices, se conviertan en una ventaja visual que dé personalidad propia al juego.

2.3.3. Ambientación

Finalmente, la ambientación de Mortis Roll busca transmitir una sensación de pavor y misticismo. Para lograr este tono tan específico, se ha tomado como inspiración el

juego Hylics (2015), el cual destaca por su estilo surrealista y sus diseños de personajes extraños y oníricos (Ver figura 4). Pese a que el género es distinto, la forma en que Hylics (2015) presenta su mundo de forma críptica y hostil es lo que se pretende replicar en el diseño de los enemigos y guardianes del panteón.

Figura 4

Estilo surrealista y diseño de personajes oníricos en Hylics



Nota. Captura de pantalla ilustrando la estética críptica y surrealista de los enemigos. Tomada del videojuego Hylics (Mason Lindroth, 2015).

La atmósfera se apoya también en el uso de la iluminación y las sombras para esconder detalles, con tal de que el jugador sienta que siempre hay algo acechando en la oscuridad. Esta decisión artística ayuda a que el panteón no se perciba como un simple escenario de juego, sino como un lugar vivo y corrupto donde el protagonista es un intruso. En definitiva, la ambientación une la estética retro con la

narrativa fragmentada para que la experiencia de juego sea tan inmersiva como inquietante.

3. Diseño de juego (GDD)

3.1. Resumen del proyecto (Elevator Pitch)

Mortis Roll es un videojuego perteneciente al género *roguelike dungeon crawler* en tercera persona con una estética retro 3D inspirada en la quinta generación de consolas (PlayStation 1). El jugador encarna a un guerrero atrapado en un panteón subterráneo maldito e infestado por antiguos cultos, el cual debe explorar estancias generadas de manera procedimental, recolectar reliquias que alteran sus estadísticas y combatir contra entidades oscuras en tiempo real. La experiencia se define por la tensión constante, la toma de decisiones tácticas rápidas en el espacio tridimensional y un ciclo de juego basado en el aprendizaje a través de la muerte permanente (*permadeath*).

3.2. Género y Ambientación

El videojuego se define de forma estricta como un *roguelike dungeon crawler* convencional de acción. Abandonando los sistemas de turnos.El juego se ambienta en un antiguo panteón sombrío que honraba a deidades olvidadas, lo que genera un entorno de fantasía oscura y pavor constante. Siguiendo las directrices estéticas de Schell (2014) sobre el diseño de mundos interactivos, la coherencia visual granulada de los *shaders* de PS1 y el diseño opresivo del escenario actúan como un canal de comunicación indirecta que sumerge al usuario en una atmósfera de extrañeza y peligro inminente.

3.3. El Jugador y la Estructura del Mundo

El usuario controla de forma directa a un guerrero caído en una perspectiva isométrica/tercera persona fija durante los encuentros. La estructura del mundo es orgánica y fluida. El jugador navega físicamente por una mazmorra con una

disposición aleatoria de habitaciones conectadas entre ellas, generada algorítmicamente al inicio de cada partida.

Esta progresión espacial se fundamenta en la Lente de los Espacios de Jesse Schell (2014), la cual establece que la arquitectura del juego debe definir claramente los flujos de movimiento, las zonas de peligro seguro y los puntos de interés ocultos para recompensar la exploración intrínseca del jugador. El objetivo final del usuario es descubrir la entrada a la cámara central para derrotar a las deidades primordiales.

3.4. Ciclo de Juego (Core Loop)

El núcleo de la experiencia interactiva se reestructura en un bucle continuo de tres fases principales que operan en tiempo real:

[Exploración de la Mazmorra Aleatoria] ➔ [Combate en Tiempo Real y Posicionamiento] ➔ [Recolección de Objetos y Modificación de Atributos]

- Flujo Macro (Estructura de la Partida): El jugador avanza habitación por habitación a través de la arquitectura generada por el algoritmo. Cada estancia puede albergar un encuentro hostil, cofres de tesoro o zonas de descanso. La muerte finaliza la partida de forma irrevocable, devolviendo al usuario al inicio de la mazmorra con un nuevo mapa aleatorio, transformando el fracaso en conocimiento cognitivo para el siguiente intento.
- Flujo Micro (Ciclo de Combate): El combate ocurre de forma directa sobre el escenario tridimensional. El jugador debe calcular las distancias de los enemigos, ejecutar ataques mecánicos (comando *attack*) y esquivar los proyectiles o golpes por contacto de los adversarios utilizando las físicas y las cajas de colisión (*Hitboxes*) configuradas en las capas del motor de juego.

3.5. Objetivos, Habilidades y Progresión

El objetivo principal es limpiar la mazmorra de guardianes surrealistas hasta alcanzar la deidad del nexo. La progresión se sustenta de manera directa en la Lente de las Habilidades del Jugador y la Lente del Equilibrio (Game Balance) de Schell (2014):

- Habilidades físicas y mentales: El juego exige destreza motriz (coordinación espacial, cálculo de tiempos de ataque) combinada con habilidades cognitivas de toma de decisiones rápidas (gestión de prioridades de eliminación de enemigos y racionamiento de objetos curativos).
- Sistema de Objetos y Sinergias: El azar dicta qué armas y reliquias se recolectan en los cofres de la mazmorra. Las reliquias modifican variables directas del personaje en tiempo real (por ejemplo, expandir el radio del área de daño de la espada, incrementar los *frames* de invulnerabilidad o añadir efectos de estado a los ataques). El éxito de una partida depende de cómo el jugador adapta su estilo de combate en tiempo real a las sinergias de los objetos encontrados.

3.6. Apartado Visual e Interfaz (UX)

La comunicación visual es un pilar crucial en este medio. Schell (2014) señala en la Lente de la Interfaz que el estado del juego debe transmitirse de forma transparente y sin fricciones al jugador, especialmente en entornos interactivos veloces. Para ello, la interfaz de combate en pantalla (*HUD*) se simplifica en tres canales informativos directos:

- Indicador de Vitalidad Dinámico: Una barra de salud que parpadea y aplica un efecto cromático granulado en la pantalla al recibir daño físico, comunicando el estado de invulnerabilidad temporal del *character*.
- Inventario de Reliquias Activas: Ubicado en la esquina superior de la pantalla, muestra iconos pixelados claros de los modificadores mecánicos recolectados para que el usuario recuerde sus ventajas pasivas durante el combate.
- Feedback del Escenario: Los enemigos emplean siluetas contrastadas basadas en los referentes de *Hylics* y señales visuales previas a sus ataques (como un breve destello o cambio de postura en sus modelos de pocos polígonos), permitiendo al jugador leer las amenazas espaciales de forma inmediata sin necesidad de sobrecargar la pantalla con texto.

3.7. Sistema de Audio

El paisaje sonoro se transforma en una herramienta de usabilidad y navegación fundamental. El audio interactivo se adapta dinámicamente: la música mantiene una cadencia ambiental y opresiva de baja fidelidad durante la exploración, pero se acelera de forma abrupta mediante disparadores (*triggers*) en cuanto un enemigo detecta al jugador. Además, los efectos de sonido (*SFX*) aportan un *feedback* directo e indispensable: cada golpe conectado contra la geometría de una hitbox emite un sonido sordo y metálico, confirmando el éxito de la acción e interactividad de manera inmediata.

4. Presupuesto

Para evaluar la viabilidad económica del desarrollo del Producto Mínimo Viable (MVP) de *Mortis Roll*, se ha diseñado un presupuesto adaptado al marco legal, fiscal y laboral vigente en España para el año 2026. Dado que el proyecto ha sido ejecutado de forma independiente por un único desarrollador en solitario (*indie developer*), el cálculo de los costes de personal no se estructura bajo un esquema de contratación por cuenta ajena, sino mediante la estimación de honorarios profesionales como trabajador autónomo (freelance).

El cronograma estimado para alcanzar la versión de MVP funcional acumula un total de 320 horas de producción que engloban las áreas de diseño de juego, programación de sistemas y creación de *assets* tridimensionales.

A continuación, en la Tabla 1, se detallan las partidas presupuestarias requeridas para la ejecución material y la posterior distribución comercial del videojuego:

Tabla 1

Desglose de costes estimados para el desarrollo y publicación del MVP

Concepto	Detalle / Descripción	Coste Unitario	Total
Honorarios de Desarrollo	320 horas de producción (diseño, código y arte 3D) computadas a tarifa freelance de perfil Junior (Manfred, 2026).	25,00 € / hora	8.000,00 €
Seguridad Social (RETA)	Cuota mensual reducida por contingencias comunes bajo la Tarifa Plana para Nuevos Autónomos durante el periodo de desarrollo (Wolters Kluwer, 2026).	80,00 € / mes	400,00 €
Distribución (Publishing)	Tasa de registro obligatoria e irrevocable para la publicación de la aplicación en la plataforma comercial de PC (Valve Corporation, s.f.).	100,00 $ (pago único)	100,00 €
Amortización de Hardware	Depreciación lineal y desgaste de los equipos informáticos y periféricos utilizados durante el desarrollo.	60,00 € / mes	300,00 €

Gastos Operativos (Overhead)	Consumo de suministros básicos vinculados directamente a la actividad (electricidad, conectividad a internet de banda ancha).	100,00 € / mes	500,00 €
Software de Desarrollo	Licencias de uso del motor de juego Godot Engine 4.6 y la suite de modelado Blender 3D (herramientas de código abierto y uso gratuito).	0,00 €	0,00 €
TOTAL			**9.300,00 €**

Nota. Elaboración propia. Los importes de distribución digital en dólares estadounidenses han sido convertidos a euros aplicando el tipo de cambio oficial diario publicado por el Banco de España (2024) para el periodo de cálculo.

4.1. Justificación de las partidas y fuentes de financiación

La sostenibilidad del presupuesto se fundamenta en los siguientes indicadores y normativas de mercado:

- Costes de Personal y Honorarios: La tarifa asignada de 25,00 € por hora se ajusta de manera estricta al límite inferior recomendado por los estudios de mercado para perfiles tecnológicos independientes sin experiencia previa en el sector en España (Manfred, 2026). Este importe cubre de forma unificada el rol multidisciplinar asumido por el autor durante las 320 horas de desarrollo dedicadas.

- Marco Fiscal de Autónomos: Con el fin de mitigar la barrera económica inicial del emprendimiento digital, el coste relativo a la Seguridad Social se acoge al sistema de cotización de los trabajadores por cuenta propia, aplicando la cuota bonificada fija (Tarifa Plana) de 80,00 € mensuales (Wolters Kluwer, 2026).
- Comercialización y Distribución: El coste de publicación responde al depósito de seguridad *Steam Direct Fee* exigido por la distribuidora Valve Corporation (s.f.). Cabe destacar que este importe posee naturaleza de depósito recuperable, siendo reintegrado al desarrollador de forma íntegra por la plataforma una vez que el videojuego alcance un volumen mínimo de facturación bruta equivalente a 1.000,00 $ en la tienda.
- Optimización mediante Software Libre: La viabilidad financiera del videojuego se ve altamente optimizada por la decisión estratégica de utilizar un *pipeline* técnico compuesto en exclusiva por herramientas *open-source* (Godot Engine y Blender), eliminando por completo los costes fijos de licencias de software que suelen comprometer las producciones de estudios emergentes.

5. Metodología y Desarrollo

5.1. Diseño iterativo

Para el desarrollo de Mortis Roll, se ha decidido aplicar una metodología de diseño iterativo, la cual se basa en la repetición de ciclos de creación y prueba. En lugar de intentar terminar el juego de una sola vez, el trabajo se divide en pequeñas metas que permiten probar si una idea funciona antes de seguir avanzando. Pese a que trabajar en solitario requiere gestionar muchas áreas a la vez, este sistema ayuda a no perder el foco en lo más importante: que el juego sea divertido.

Siguiendo las recomendaciones de Schell (2014), cada iteración sigue cuatro pasos: diseñar, prototipar, probar y evaluar. Este proceso es vital en un roguelike, ya que permite ajustar la dificultad y el comportamiento de los enemigos poco a poco. Con tal de no malgastar tiempo, primero se crean versiones muy básicas de las mecánicas en Godot (2024) para comprobar el "game feel"; si la sensación de juego es satisfactoria, se procede entonces a crear el arte final en Blender.

5.2. Motor de juego

He elegido Godot (2024) como motor de juego debido principalmente a su arquitectura basada en nodos y escenas, la cual encaja a la perfección con la lógica de un juego modular. En este motor, cada elemento del juego se trata como una entidad independiente que puede testearse por separado. Esto facilita enormemente el trabajo de un desarrollador en solitario, ya que permite organizar el proyecto de forma limpia y ordenada.

Pese a que existen otros motores con más cuota de mercado, Godot destaca por ser extremadamente ligero y por la agilidad que ofrece su lenguaje de programación, GDScript. Esta rapidez a la hora de escribir código es clave para un juego con un tiempo de desarrollo ajustado, permitiendo prototipar la lógica de combate y los encuentros de forma ágil. De esta manera, se puede dedicar más tiempo a pulir la experiencia de juego que a pelear con tiempos de compilación excesivamente largos.

Además, Godot es un motor que ha ganado una gran popularidad durante los últimos años, y esto, sumado a que es de código abierto, ha contribuido a la formación de una gran comunidad que ha creado todo tipo de documentación.

5.3. Creación de assets 3D

Para la creación de todos los elementos visuales, Blender (2024) es la herramienta central del proyecto. La principal ventaja es la integración nativa que tiene con Godot, la cual permite importar archivos de forma dinámica y ver los cambios reflejados en el motor casi al momento. Pese a que Blender es capaz de manejar millones de polígonos, en Mortis Roll se utiliza para un modelado "low-poly", buscando siempre la sencillez de las formas, emulando el estilo 32 bits.

El uso de Blender se centra especialmente en el mapeado UV manual, el cual es imprescindible para asegurar que las texturas pixeladas encajen perfectamente en los bordes de los modelos. Esta técnica es lo que permite que el juego mantenga esa estética de 32 bits tan marcada. Con tal de asegurar que el altar y las criaturas se sientan parte de un mismo universo, se realiza todo el proceso de modelado y

texturizado bajo un mismo criterio artístico, evitando depender de librerías externas que puedan romper la cohesión visual del conjunto.

5.4. Flujo de trabajo

Para que la producción sea eficiente y no se produzcan errores técnicos, se ha establecido un flujo de trabajo claro que se repite con cada nuevo objeto que entra en el juego:

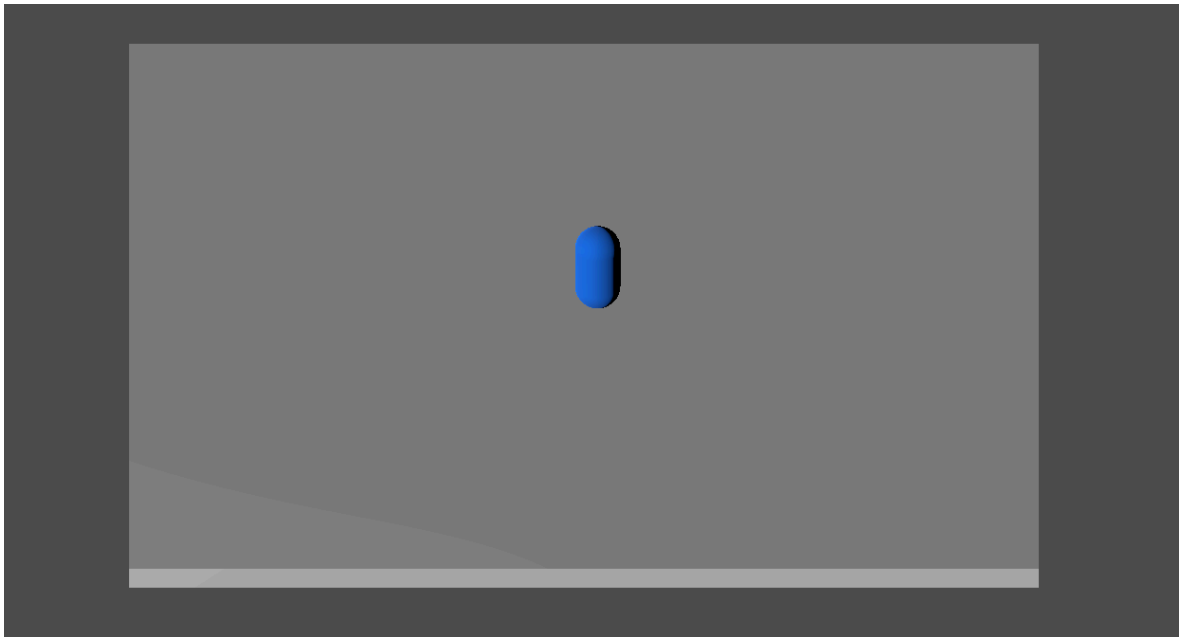
- Conceptualización: Se definen las formas básicas y la función que tendrá el objeto o enemigo dentro del juego.
- Modelado en Blender: Se construye la geometría usando la menor cantidad de polígonos posible, con tal de respetar la estética retro.
- Mapeado y Texturizado: Se preparan las mallas para recibir texturas de baja resolución donde el píxel es el protagonista.
- Exportación (.gltf): Se traslada el modelo a Godot, aprovechando la capacidad del motor para actualizar los archivos de forma dinámica.
- Implementación y Shaders: Una vez en el motor, se le asigna su lógica de comportamiento y se le aplican los efectos de post-procesado (como el *vertex jittering*), el cual termina de darle el aspecto final de juego de PS1.

5.5. Programación de sistemas

El desarrollo de este apartado comenzó con la implementación del sistema de locomoción del protagonista, denominado técnicamente en el juego como “character”. Debido a la naturaleza isométrica de la cámara elegida, el primer reto técnico fue corregir la diferencia de velocidades en los distintos ejes, un problema que se da comúnmente en perspectivas inclinadas. Siguiendo la filosofía de diseño iterativo del juego , se ha utilizado una cápsula geométrica básica como "placeholder" para representar el volumen del personaje, como se puede ver en la figura 5, permitiendo validar el funcionamiento de las colisiones antes de proceder a la integración de los modelos definitivos.

Figura 5

Cápsula geométrica básica como "placeholder" del jugador

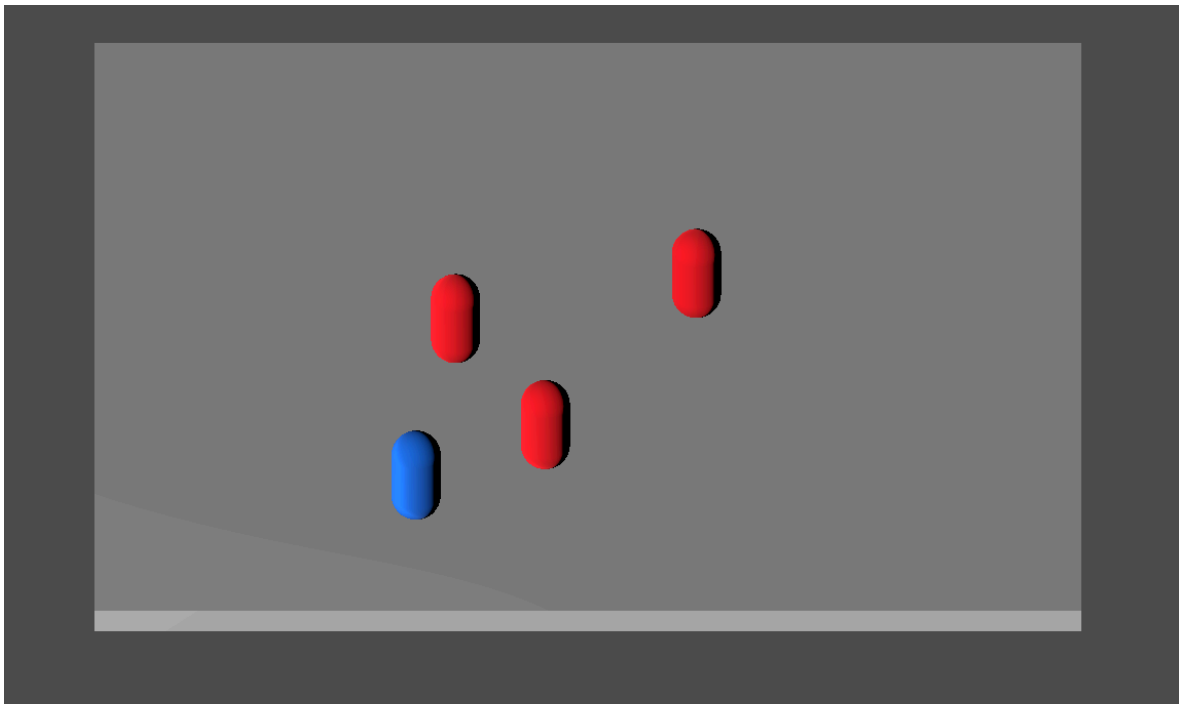


Nota. Captura de pantalla del visor de Godot mostrando la representación provisional del protagonista para probar el sistema de locomoción y colisiones. Elaboración propia.

Una vez consolidado el movimiento, creé mediante código un comportamiento básico para los enemigos. El comportamiento se diseñó mediante un sistema de persecución simple donde el enemigo calcula constantemente la distancia y el vector hacia el jugador, orientándose siempre hacia su objetivo. Para suavizar el comportamiento físico y evitar que los modelos se solapen de forma antiestética se implementó una distancia de frenado (*stopping distance*) ajustable desde el inspector de Godot. Al igual que con el protagonista, estas entidades enemigas se han representado inicialmente mediante cápsulas básicas provisionales (ver figura 6), actuando como guías visuales temporales para ajustar la velocidad de persecución y el radio de detección sin depender de la carga técnica de los *assets* finales.

Figura 6

Representación provisional de enemigos mediante cápsulas

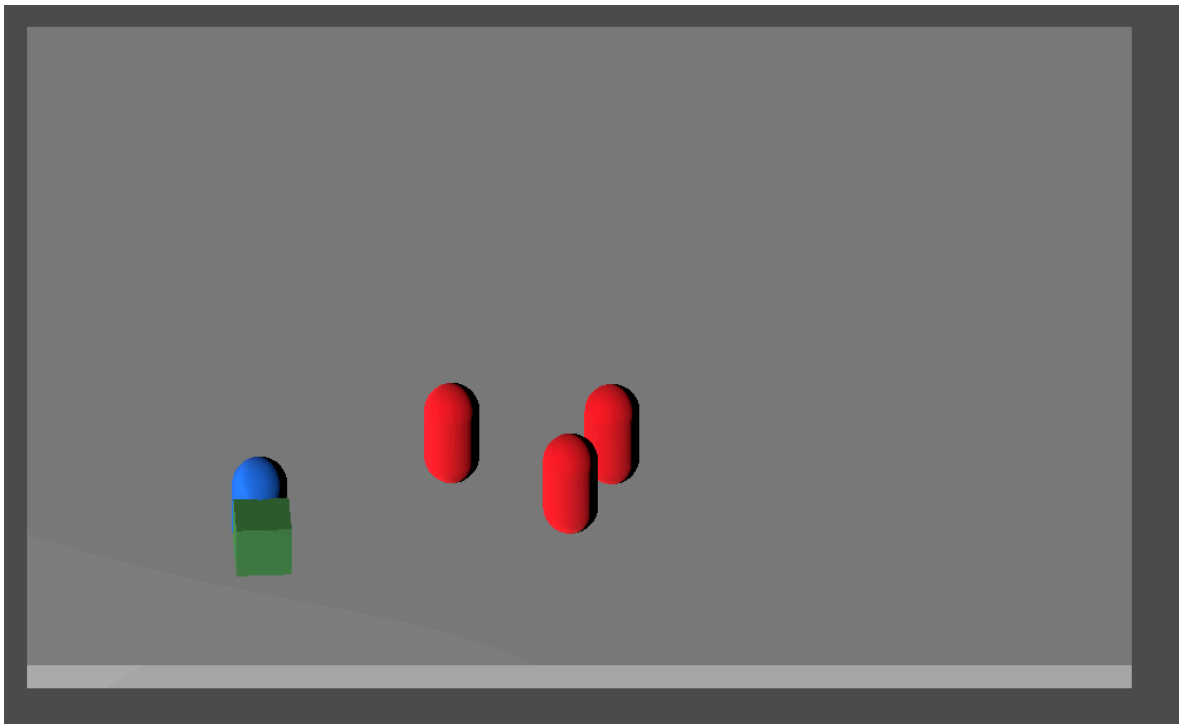


Nota. Captura de pantalla en Godot que ilustra el sistema de persecución y la distancia de frenado (*stopping distance*) con modelos temporales. Elaboración propia.

El sistema de combate se resolvió mediante el uso de nodos Area3D, configurados como Hitboxes de detección. El jugador cuenta con un área denominada zonaataque que se activa mediante el comando de entrada "attack" (ver figura 7), mientras que los enemigos disponen de un sensor de proximidad para aplicar daño por contacto. Para que ambos sistemas coexistan sin generar conflictos de autodaño o detecciones erróneas, se diseñó una matriz de colisiones por capas (*Physics Layers*). En esta configuración, se asignó al cuerpo del jugador la Capa 1 y a los enemigos la Capa 3; de este modo, el área de ataque del protagonista solo escanea la Capa 3, ignorando su propia geometría, mientras que el ataque del enemigo se filtra para buscar exclusivamente la Capa 1.

Figura 7

Configuración de áreas de detección (Hitboxes) en el combate



Nota. Captura de pantalla del entorno de Godot mostrando el área de ataque del jugador (caja verde) configurada mediante capas de colisión frente a los enemigos. Elaboración propia.

Finalmente, se integró la lógica de salud. Cada impacto detectado activa una función de daño que resta puntos de una variable de salud máxima, desencadenando la eliminación del nodo mediante queue_free() cuando esta llega a cero. Para mejorar la experiencia de usuario y evitar que el jugador pierda toda su vida de forma injusta en un solo fotograma, se programó una rutina de Frames de Invulnerabilidad. Al recibir un golpe, el jugador entra en un estado de protección durante un segundo, apoyado por un efecto visual de parpadeo que utiliza la propiedad de visibilidad del nodo. Más adelante se implementaran animaciones de muerte y un menú de derrota para sumarle al *game feeling.*

Tras asentar la lógica de movimiento y el sistema de combate básico, el desarrollo técnico se centró en la creación y la navegación por el juego. En primer lugar, se implementó un sistema de puertas condicionales coordinado con el estado de la mazmorra. Este mecanismo escanea de forma constante la habitación y permanece bloqueado por defecto, activándose de forma automática cuando el contador de enemigos llega a cero, lo que asegura que el usuario limpie la habitación antes de poder avanzar. Al cruzar la puerta, el sistema ejecuta una transición que da paso a la siguiente habitación del mapa.

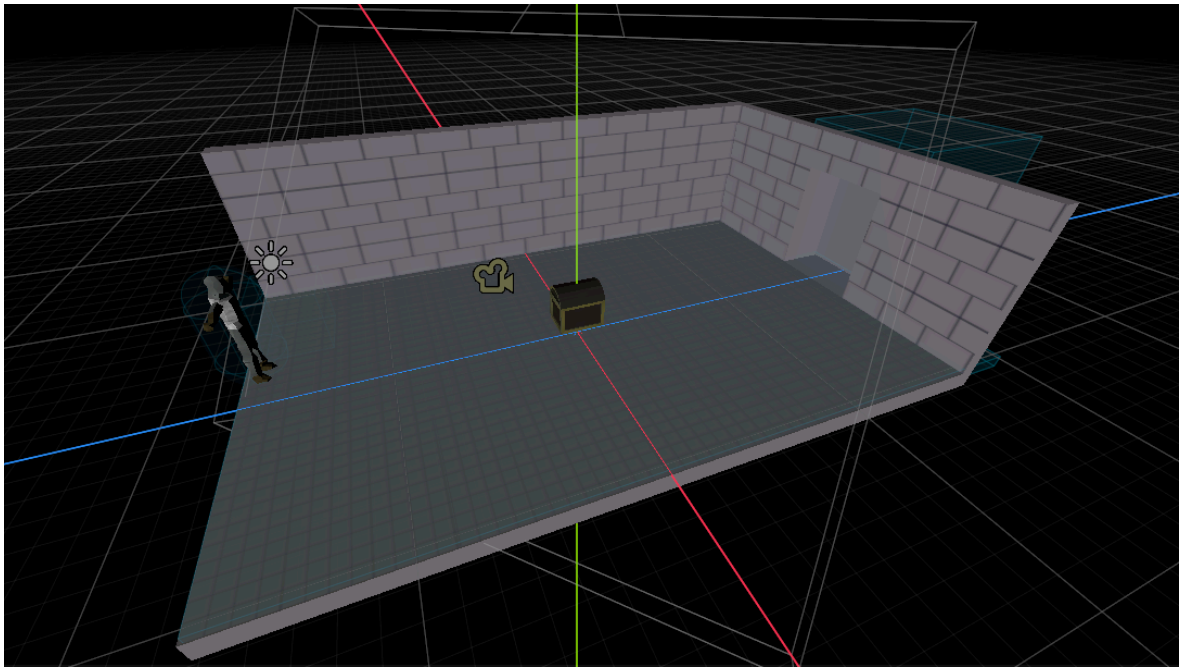
En segundo lugar, se ha desarrollado un administrador global denominado *RoomManager*. Ya que el motor de juego libera la memoria de los escenarios al cambiar de estancia, este componente actúa como un almacén de datos que no se destruye durante la partida. Gracias a este añadido, el videojuego conserva de forma transparente entre las diferentes salas las variables cruciales de la partida actual, como la salud actual del jugador, su límite de vida máxima, el catálogo de objetos recolectados y el tipo de arma que lleva equipada, logrando un flujo de juego macro orgánico y cohesionado.

En relación con el avance por las habitaciones, se ha implementado el sistema de recompensas, estructurado a través de tres componentes jerárquicos: la sala del tesoro, el cofre y los objetos recogibles.

Para darle más interés a la aleatoriedad de la mazmorra, el algoritmo de generación permite la aparición de una habitación libre de enemigos como se puede ver en la figura 8. En el centro de esta habitación se ubica un elemento cofre interactivo. Al colisionar con el personaje, este activa una animación visual en la que la tapa del modelo se desplaza de forma automática hacia atrás, simulando su apertura, y realiza una llamada al marcador de posición local configurado en su interior. Este punto exacto funciona como una guía que le indica al motor de juego dónde debe aparecer físicamente la recompensa asignada a la partida.

Figura 8

Disposición espacial de la sala del tesoro y el cofre interactivo



Nota. Captura de pantalla del entorno de desarrollo mostrando la habitación de recompensa libre de peligros y el elemento interactivo central. Elaboración propia.

Por último, se ha desarrollado la lógica del objeto recogible que sale del cofre. Con tal de evitar que el artículo se camufle con el resto de la estancia o que quede oculto a la vista del usuario, se ha programado una función visual que actúa sobre la malla del objeto. El sistema forzará a la pieza a realizar un movimiento continuo de rotación sobre su propio eje, combinado con una levitación vertical suave que flota de arriba hacia abajo de forma constante.

Para garantizar la usabilidad, este objeto cuenta con un temporizador invisible de medio segundo que impide su recolección inmediata tras la apertura del cofre, evitando capturarlo por accidente. Una vez transcurrido ese lapso de seguridad, si el jugador toca el área de detección del objeto, el sistema lee la ficha técnica del mismo y ejecuta la acción correspondiente, ya sea hacer aparecer el armamento en

la mano del protagonista o aplicar las ventajas pasivas de la reliquia directamente en su barra de vitalidad.

Con el objetivo de aportar dinamismo e imprevisibilidad a los encuentros dentro de las mazmorras sin necesidad de sobrecargar el desarrollo con el modelado de decenas de enemigos desde cero, se diseñó un sistema aleatorio de variantes.

Al instanciarse en la habitación, cada criatura ejecuta de forma interna un cálculo de probabilidad aleatoria. En caso de cumplir con la tasa establecida, la entidad se transforma automáticamente en una variante más poderosa. Este estado modifica por código las propiedades del enemigo, multiplicando sus valores de daño base y aumentando proporcionalmente su tamaño.

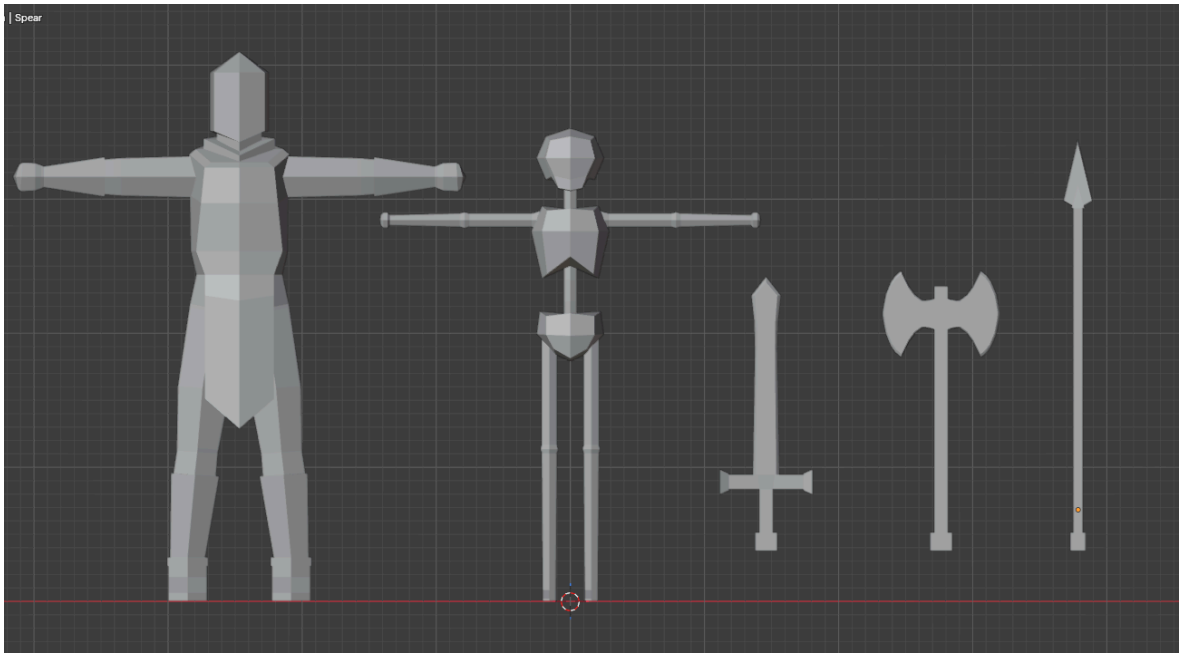
5.6. Arte

En el apartado artístico, como ya se ha aclarado anteriormente Mortis Roll intenta emular las peculiaridades técnicas de los videojuegos de la primera Playstation.

La primera limitación técnica y la más icónica de esta generación es la baja capacidad de carga poligonal. Por lo tanto, a la hora de crear los modelos, he tenido esto en cuenta y los he hecho siguiendo una estética *low-poly* (Ver figura 9).

Figura 9

Modelado de geometría "low-poly" para personajes y armamento

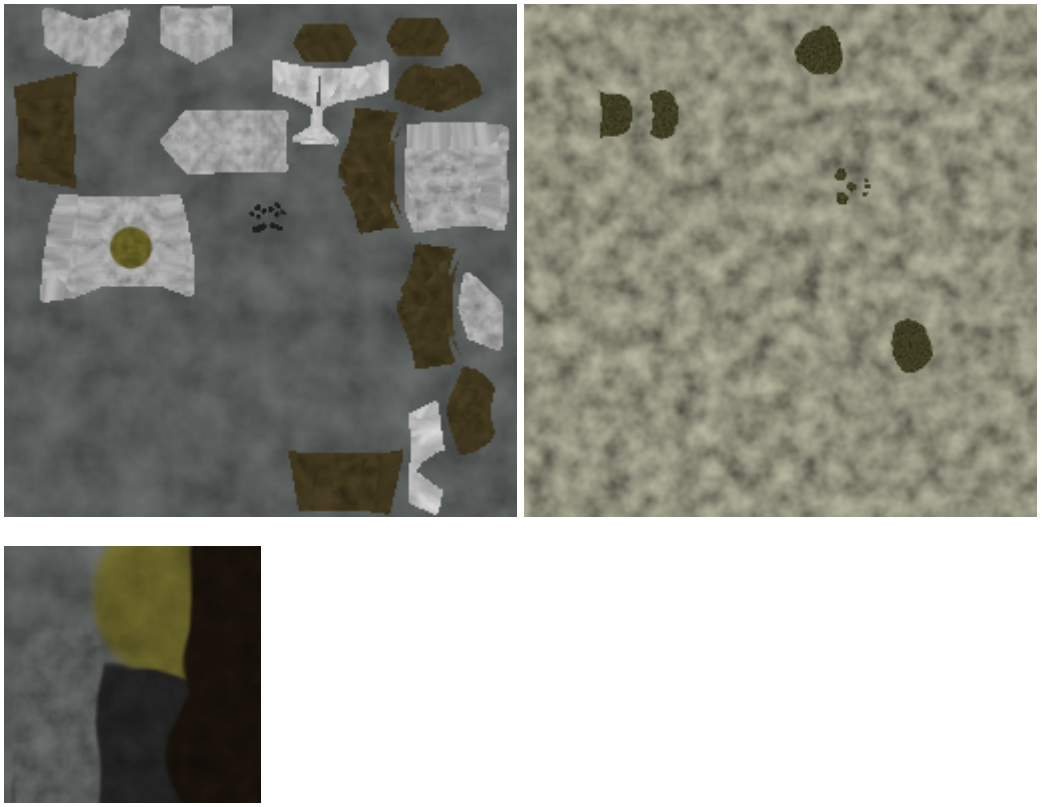


Nota. Captura de pantalla del entorno de desarrollo Blender mostrando la topología básica y sencilla de los modelos antes del texturizado. Elaboración propia.

Una vez creados los modelos llega el momento de texturizarlos. En la Playstation 1 se utilizaban texturas de baja resolución para no sobrecargar la memoria de la consola. Estas texturas eran principalmente de 128x128 píxeles y un máximo de 256x256. En el caso de este juego se han utilizado texturas de 256x256 para los personajes, ya que necesitan más nivel de detalle, y de 128x128 para los objetos (ver figura 10).

Figura 10

Atlas de texturas y archivos de imagen base para los modelos 3D



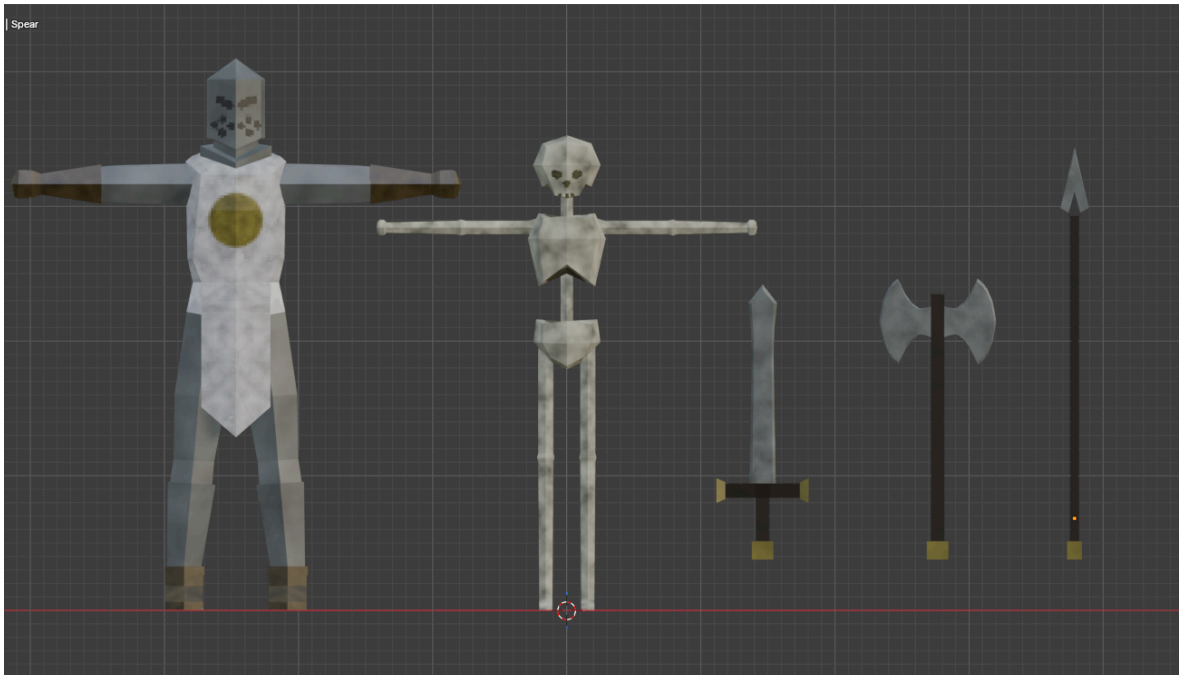
Nota. Muestra de los archivos de imagen en formato PNG importados al proyecto. Se ilustra el atlas de texturas creado para agrupar múltiples materiales de los objetos en una única imagen, optimizando así el rendimiento y la memoria. Elaboración propia.

Para la textura de las armas he creado lo que se conoce como un atlas de texturas. Una imagen en la que se incluyen las diferentes texturas que estarán presentes en los objetos. Con esto consigo tener una sola textura que puedo reutilizar para todas las armas solo jugando con el mapeado de UVs, evitando así tener que crear una textura única para cada objeto y consiguiendo una mayor optimización.

En los atributos de las texturas se ha cambiado el método de filtrado de *linear* a *nearest* consiguiendo así evitar el suavizado de píxeles que no estaba presente en la Playstation 1, como se puede ver en la figura 11.

Figura 11

Modelos tridimensionales con el texturizado final aplicado



Nota. Personajes y armas en Blender haciendo uso del atlas de texturas mediante la asignación del mapeado UV. Elaboración propia.

5.7. Postprocesado

Si bien el apartado visual de la PlayStation original suele asociarse casi exclusivamente con la baja carga poligonal (low-poly), los artefactos visuales derivados de sus limitaciones de hardware resultan igualmente fundamentales para definir la identidad estética de sus videojuegos.

En primer lugar la cuantización de color producida por la baja profundidad de bits de la consola. Este fenómeno reducía la cantidad de colores de la consola, generando saltos bruscos entre colores en lugar de degradados suaves.

Para mitigar los efectos de la cuantización la consola utilizaba una técnica conocida como dithering. Esta técnica consiste en intercalar puntos para crear la ilusión óptica de que existe un color intermedio, pero dándole a la imagen una textura granulada.

Otro efecto característico de la consola es el temblor de vértices. Este efecto se daba debido a que la PS1 no podía calcular la posición exacta de los vértices en pantalla, por lo que la redondeaba, dándole un efecto de temblor a los modelos.

Para conseguir replicar estos efectos en el juego he implementado varios *shaders* en Godot. Para la cuantización de color he programado un filtor que reduce la amplitud de color de los 8 bits por canal que son comunes hoy en día, a los 5 bits por canal que cargaba la Playstation original.

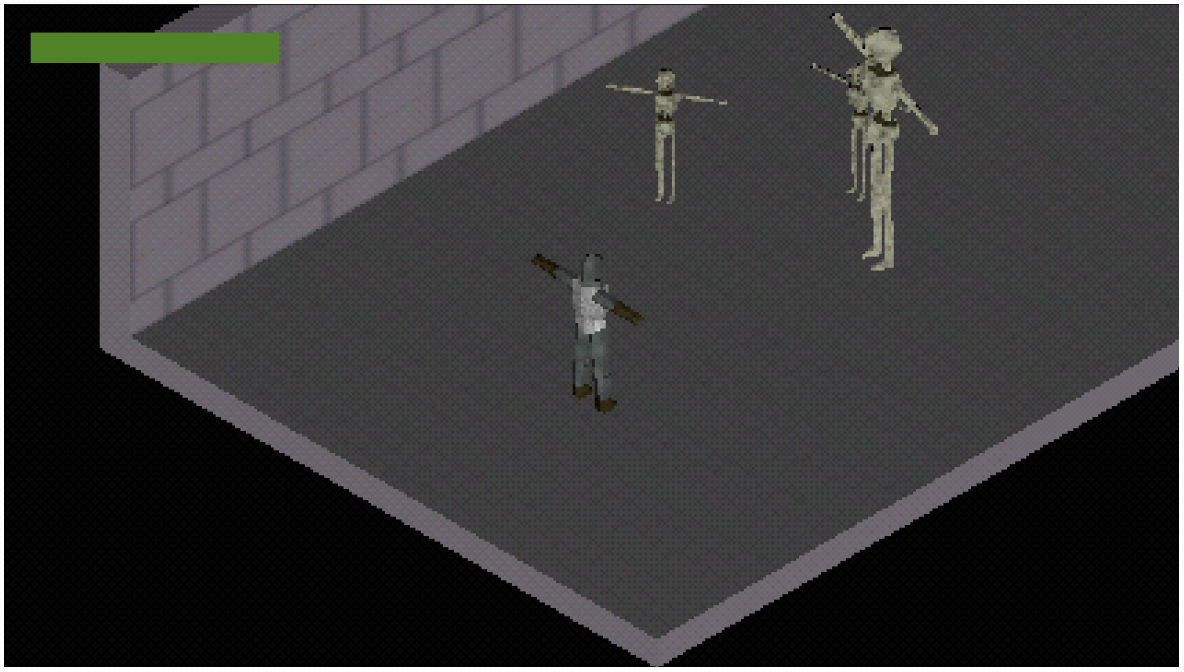
Para solventar los cortes bruscos entre colores he integrado el sistema de dithering basándose en la matriz 4x4 que utilizaba la consola, consiguiendo también la icónica textura granulada de los videojuegos de la época.

El temblor de vértices lo he logrado mediante un shader de geometría que fuerza a los vértices en pantalla a posicionarse sobre puntos concretos de una rejilla.

Todos estos efectos se pueden observar en la figura 12.

Figura 12

Aplicación de shaders de postprocesado para la emulación visual de 32 bits



Nota. Captura de pantalla del prototipo que ilustra la cuantización de color y el entramado (*dithering*) aplicado sobre los modelos tridimensionales. Elaboración propia.

6. Mejoras futuras

Pese a que el prototipo actual de *Mortis Roll* cumple con los objetivos específicos establecidos y funciona como un Producto Mínimo Viable (MVP) sólido, entiendo este trabajo como el primer paso de un proyecto a mayor escala. Al haber utilizado un motor basado en nodos y escenas como Godot 4.6, el juego cuenta con una estructura modular que facilita enormemente la adición de nuevo contenido. Con tal de acercar este prototipo a lo que sería un producto comercial y completo, he planteado una serie de mejoras futuras centradas en el diseño y la experiencia de usuario:

6.1. Animaciones y *Game Feel*

Pese a que actualmente el jugador recibe información visual básica mediante parpadeos al sufrir daño, mi intención es pulir mucho más la sensación de juego (*game feel*). Para ello, el siguiente paso lógico es crear animaciones completas en Blender para todas las entidades, las cuales incluyan ciclos de ataque fluido, reacciones al impacto y animaciones de derrota. También se planea implementar a la pantalla de *Game Over* que muestre las reliquias obtenidas y las habitaciones superadas, con tal de que el usuario vea reflejado su esfuerzo y no perciba la muerte como un castigo frustrante.

6.2. Profundidad en el comportamiento enemigo

En el prototipo, los enemigos cuentan con un comportamiento de persecución constante y directo hacia el jugador. De cara a futuras iteraciones, quiero diseñar una lógica más compleja mediante máquinas de estados. Esto permitirá que los guardianes no se limiten a perseguir, sino que puedan patrullar zonas concretas o buscar posiciones de ventaja táctica. Por otra parte, mi intención es añadir nuevos arquetipos de enemigos, como aquellos capaces de realizar ataques a distancia, los cuales obligarán al jugador a gestionar mucho mejor los tiempos y el espacio de la mazmorra.

6.3. Generación procedimental avanzada

Aunque el juego ya genera la disposición de las estancias de forma aleatoria, el sistema debe expandirse para que la navegación sea más orgánica y sorprendente. El objetivo a futuro es mejorar el algoritmo para que garantice la aparición de caminos principales, rutas sin salida y cámaras ocultas. A esto se le sumará la creación de un mayor catálogo de habitaciones prefabricadas en Blender, introduciendo estancias con trampas de entorno o altares de mercaderes, con tal de romper la monotonía visual y jugable en cada partida.

6.4. Metaprogresión y narrativa fragmentada

Como se ha explicado en la definición del *roguelike*, es vital que la experiencia recompense el aprendizaje del usuario. Por ello, una mejora futura indispensable es la integración de un sistema de metaprogresión estructurado (Chen, 2007). Mi intención es crear un recurso que se conserve tras la muerte, el cual se podrá invertir en la zona segura inicial para desbloquear nuevas reliquias pasivas en la reserva del juego. Este sistema servirá también para desplegar la ambientación del proyecto; al repetir ciclos, el jugador irá obteniendo descripciones crípticas y fragmentos visuales, los cuales le permitirán reconstruir por sí mismo el misterio que envuelve a este panteón oscuro.

7. Conclusiones

Como conclusión de este Trabajo de Fin de Grado, puedo afirmar que *Mortis Roll* ha cumplido con los objetivos generales y específicos que me propuse al inicio del proyecto. He logrado construir un Producto Mínimo Viable (MVP) completamente funcional, el cual demuestra que es posible juntar las mecánicas de un *roguelike* de acción moderno con un estilo artístico retro inspirado en la primera PlayStation.

Desde el punto de vista de la Comunicación Interactiva, considero que el mayor logro de este trabajo ha sido demostrar que los antiguos fallos técnicos de los 32 bits no son simples defectos del pasado. Pese a que la industria actual busca siempre el fotorrealismo gráfico, he comprobado que efectos como el *vertex jittering*, el *dithering* y las texturas sin suavizado funcionan como herramientas visuales muy potentes. El uso de estos elementos ha sido clave con tal de sumergir al jugador en la atmósfera opresiva que buscaba para el panteón, demostrando que este estilo visual puede contar una historia y transmitir sensaciones por sí mismo sin necesidad de gráficos avanzados.

En cuanto al diseño de juego (GDD), enfocar el sistema hacia el combate en tiempo real y la exploración aleatoria ha sido un acierto para la experiencia del usuario. He conseguido que el núcleo del juego transmita las reglas de manera clara sin tener

que llenar la pantalla de elementos visuales. Pese a que la muerte permanente (*permadeath*) puede ser frustrante si no se plantea bien, gracias a la precisión de las capas de colisión (*hitboxes*) y al *feedback* visual, el jugador entiende perfectamente por qué ha perdido. Esto hace que vea la derrota como un aprendizaje táctico y quiera volver a intentarlo, manteniéndose enganchado en ese estado de reto constante o *flow* del que habla Schell (2014).

Por último, a nivel técnico y metodológico, enfrentarme a este desarrollo en solitario ha sido un desafío muy grande. Sin embargo, organizar un flujo de trabajo claro y directo entre Blender y Godot 4.6 ha sido la clave para que todo saliera bien.

Referencias

Banco de España. (2026). *Tipos de cambio oficiales de las divisas cotizadas por el Banco Central Europeo*. Recuperado el 28 de mayo de 2026, de https://www.bde.es/webbde/es/estadis/infoest/tipos/tipos.html

Berlin Interpretation. (s.f.). En *RogueBasin*. Recuperado el 20 de mayo de 2026, de https://www.roguebasin.com/index.php/Berlin_Interpretation

Blender Online Community. (2024). *Blender - a 3D modelling and rendering package* [Software de ordenador]. Blender Foundation. https://www.blender.org/

Chen, J. (2007). Flow in games (and everything else). *Communications of the ACM*, *50*(4), 31-34.

Godot Engine contributors. (2024). *Godot Engine* (Versión 4.6) [Software de ordenador]. https://godotengine.org/

Harris, J. (2021). *Exploring roguelike games*. CRC Press.

Manfred. (2026). *Guía para Freelance Novatos en tech*. Manfred Blog. https://getmanfred.com/blog/guia-para-freelance-novatos-en-tech-manfred/

Pezzi, G. (2024, 8 de agosto). *How PlayStation Graphics & Visual Artefacts Work*. Pikuma. https://pikuma.com/blog/how-to-make-ps1-graphics

Schell, J. (2014). *The Art of Game Design: A Book of Lenses* (2.ª ed.). CRC Press.

Valve Corporation. (s.f.). *Steamworks Partner Program: Steam Direct Fee*. Steamworks Documentation. https://partner.steamgames.com/steamdirect

Wolters Kluwer. (2026). *Cuota de autónomos: novedades, tramos y cambios de cotización*. Especiales Wolters Kluwer España. https://www.wolterskluwer.com/es-es/expert-insights/cuotas-autonomos-2026

Videojuegos

Dodge Roll. (2016). *Enter the Gungeon* [Videojuego]. Devolver Digital.

Lindroth, M. (2015). *Hylics* [Videojuego]. Mason Lindroth.

McMillen, E. (2011). *The Binding of Isaac* [Videojuego]. Edmund McMillen.

Square. (2000). *Vagrant Story* [Videojuego]. Square.

Apéndice A. El videojuego Mortis Roll

A continuación, se facilita el enlace de descarga del prototipo base de *Mortis Roll*. Este ejecutable de Windows permite probar el resultado práctico del proyecto, experimentando de forma interactiva la atmósfera retro de 32 bits, la navegación procedimental y el sistema de combate en tiempo real detallados a lo largo de esta memoria:

https://drive.google.com/file/d/1KIhpkdrLJ4s8AI4QduMrfTJuw3bqLL-r/view?usp=sharing

Apéndice B. Glosario

Affine Texture Mapping **(Mapeo afín de texturas / Distorsión de texturas):** Método de cálculo para aplicar texturas sobre polígonos sin tener en cuenta la perspectiva 3D ni la profundidad (Z). En la PlayStation original, esto provocaba que las texturas parecieran deformarse, temblar o "nadar" por la pantalla cuando la cámara cambiaba de ángulo (Pezzi, 2024).

Dithering **(Entramado):** Técnica gráfica que consiste en intercalar píxeles de colores disponibles en el sistema para simular tonalidades o transiciones suaves que el hardware no puede procesar de forma nativa. Se utilizaba para enmascarar la reducción drástica de la paleta de colores en sistemas de 32 bits (Pezzi, 2024).

Vertex Jittering **(Temblor de vértices):** Artefacto visual provocado por la carencia de una unidad de coma flotante (FPU) en la consola. Al forzar a los vértices de los modelos 3D a encajar en coordenadas de números enteros en la pantalla, los polígonos sufren saltos bruscos o vibraciones durante el movimiento (Pezzi, 2024).

Ausencia de *Z-Buffer*: Limitación de hardware por la cual el sistema no calcula automáticamente qué objetos están más cerca o lejos de la cámara. Obligaba a usar el algoritmo del pintor (ordenar polígonos de atrás hacia adelante), causando ocasionalmente que texturas lejanas se superpusieran erróneamente sobre las cercanas (Pezzi, 2024).

Generación procedimental: Sistema de creación algorítmica de contenido mediante el cual el software genera dinámicamente entornos, niveles o encuentros durante la ejecución de la partida. En lugar de estar diseñados a mano, los elementos se combinan de forma aleatoria pero sujeta a reglas predefinidas, garantizando experiencias distintas en cada intento (Harris, 2021).

Permadeath **(Muerte permanente):** Regla de diseño fundamental en la que el fracaso del jugador implica la eliminación irrevocable de su estado actual, atributos e inventario. Obliga a reiniciar el juego desde el principio, trasladando el progreso del avatar virtual al conocimiento adquirido por el propio jugador (Harris, 2021).

***Roguelike*:** Subgénero de los videojuegos de rol cuyos orígenes se remontan al juego *Rogue* (1980). Tradicionalmente, se define por una serie de pilares estrictos como la generación procedimental de niveles, los combates tácticos basados en turnos y la inclusión de la muerte permanente como eje de la experiencia de juego (Harris, 2021).

Curva de interés: Representación gráfica y conceptual de la atención y el nivel de disfrute del jugador a lo largo del tiempo. Un diseño efectivo alterna picos de alta tensión y desafío (como el combate en un altar) con momentos de relajación para evitar la fatiga mental del usuario (Schell, 2014).

***Game Balance* (Equilibrio de juego):** Proceso de diseño iterativo cuyo objetivo es ajustar las reglas, las probabilidades y las recompensas para asegurar que el sistema sea justo. Un juego equilibrado ofrece retos proporcionales a las habilidades del jugador, evitando tanto la frustración extrema como el aburrimiento (Schell, 2014).

Mecánica de juego: Los procedimientos, reglas y sistemas fundamentales que dictan cómo funciona el universo del juego y cómo el jugador puede interactuar con él (por ejemplo, el lanzamiento físico de un dado para generar un valor aleatorio o aplicar daño) (Schell, 2014).