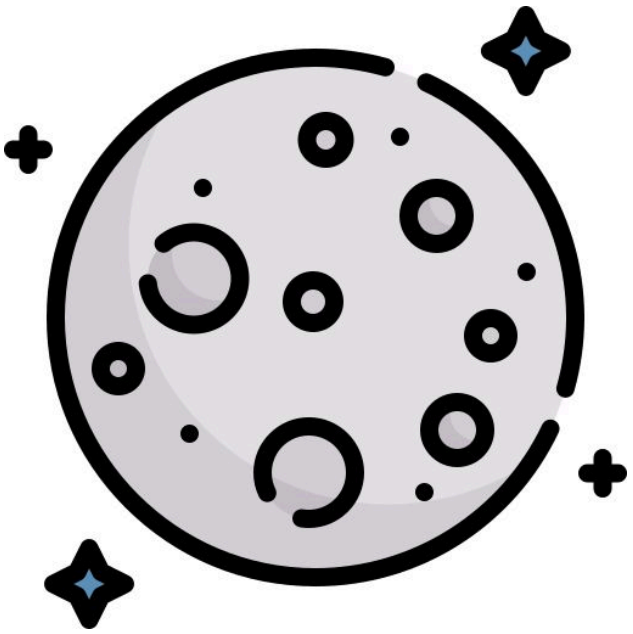
Proyecto 2D: Lunar Lander

Videojuegos 2024-2025



818665 - Fabián Murgado
818665@unizar.es

812019 - Alex Petrov
812019@unizar.es

Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

Índice

Proyecto 2D: Lunar Lander **0**
Índice **0**
Introducción **2**
Desarrollo **2**
- Dibujado 2
- Interfaz 4
  - Menús Principal y de Opciones 4
  - HUD 5
  - Menús Pausa y Fin de Juego 5
- Físicas 6
  - Movimiento 6
  - Colisiones 6
- Terreno 7
- Plataformas 7
- Sonido 8
- Asteroides 8
- Piloto Automático 9

Principales Dificultades **10**
- Generación Aleatoria del Terreno 10
- Inteligencia Artificial 10

Introducción

Durante este proyecto de la asignatura de Videojuegos hemos desarrollado un clon, lo más parecido posible, del clásico videojuego de arcade Lunar Lander para la plataforma Windows. Además se han incluido algunas opciones extra como la opción de activar una IA que juega en lugar del jugador y la opción de incluir un par de cinturones de asteroides para aumentar la dificultad.

Desarrollo

Este clon del videojuego Lunar Lander ha sido desarrollado para la plataforma Windows, utilizando el lenguaje de programación de programación C y la API de Windows para gestionar cosas como la ventana del juego, el pulsado de teclas y el dibujado de puntos y líneas en la ventana. También se ha usado en mucha menor medida algunas partes de la librería estándar de C.

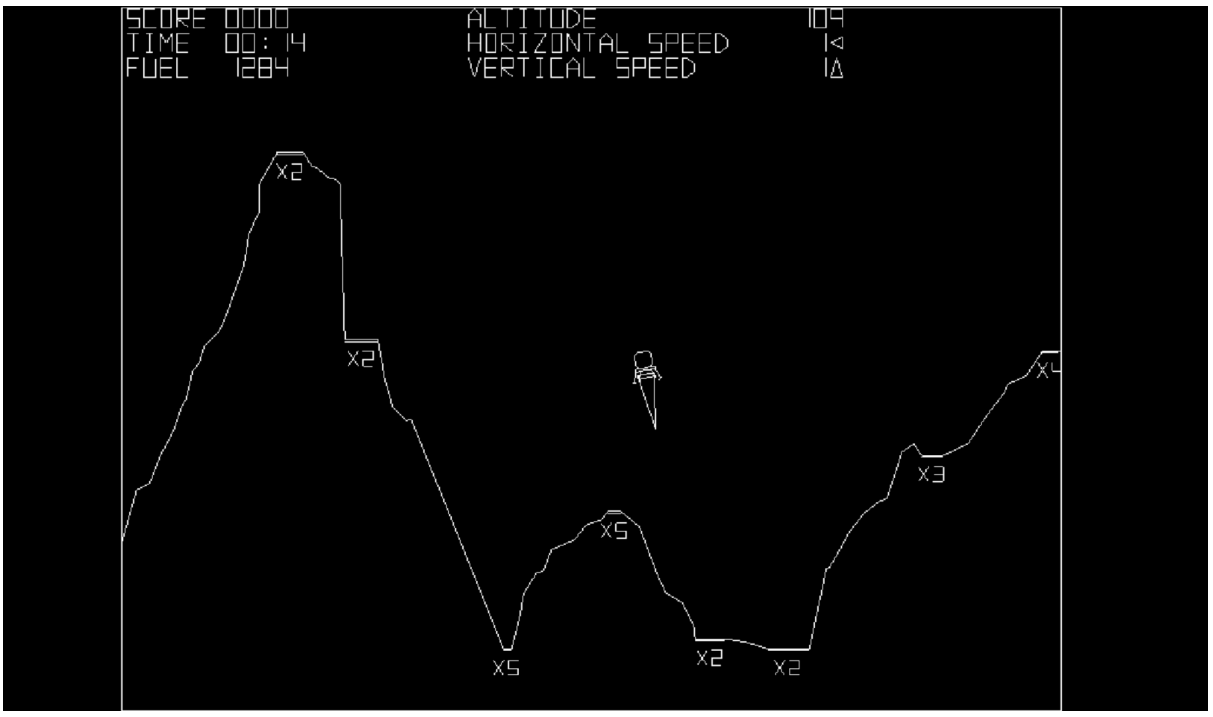
Dibujado

La representación gráfica de elementos del juego en pantalla se ha hecho a través de un objeto genérico Dibujable, el cual está formado por una lista de puntos, otra de aristas y un centro. Este objeto también implementa una serie de funciones, las cuales hacen uso de la API de Windows, para dibujar en la ventana del juego. Sobre el objeto Dibujable, también se implementan una serie de transformaciones geométricas útiles. Concretamente las de traslación, rotación y escalado.

Para eliminar parpadeos se ha utilizado un sistema de doble buffer, el cual dibuja primero sobre un buffer interno (de resolución fija). Después el buffer se escala y centra en el tamaño real de la ventana, utilizando unos márgenes calculados de forma dinámica (letterbox).

Cada vez que el programa recibe el evento WM_PAINT ejecuta la función DibujarFrame, la cual implementa el pipeline de renderizado completo. Este consta de las siguientes fases:

1. **Preparar buffers:** se crean dos buffers, uno con el tamaño de la ventana y otro de resolución fija.
2. **Limpiar** Rellenar todo el buffer interno de negro.
3. **Dibujar escena:** según el estado del juego, se llama a las funciones adecuadas: menú, juego, opciones…
4. **Transformar:** calcular zoom y posición de cámara…
5. **Trazar líneas:** recorrer cada objeto Dibujable y usar DrawLine para dibujar sus aristas.
6. **Copiar la pantalla:** escalar y centrar el buffer interno en el buffer completo. Volcar el resultado a la ventana.



Interfaz

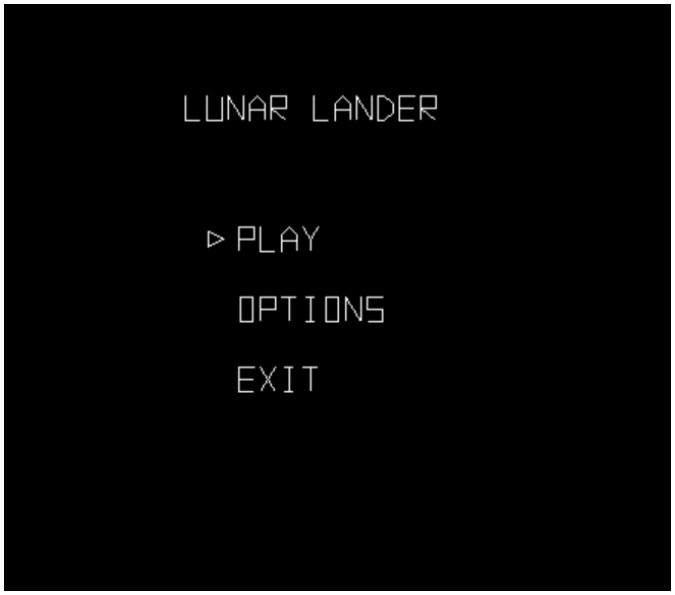
La interfaz del juego agrupa los elementos visuales y de interacción con el jugador. Incluye las pantallas estáticas (menú principal, menú de opciones, menú de pausa y menú fin), además de los indicadores dinámicos durante la partida (HUD).

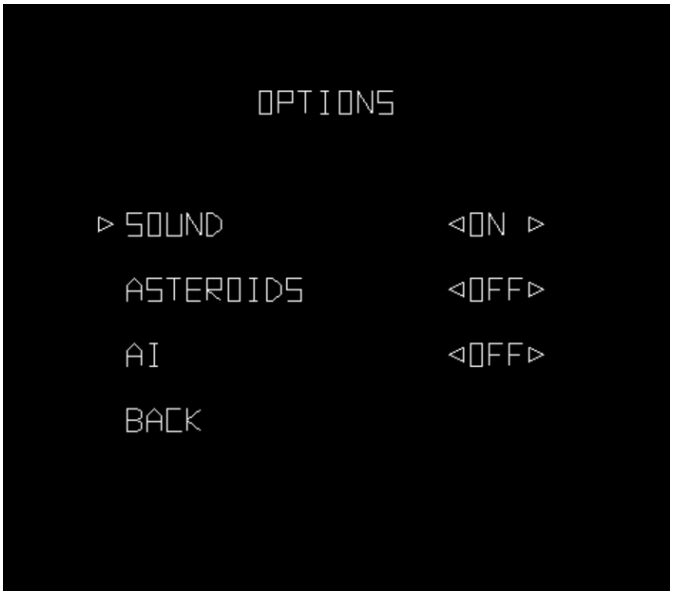
Para representar el texto se han definido objetos constantes de tipo Dibujable de cada una de las letras y símbolos necesarios para representar las distintas interfaces. También se han implementado funciones auxiliares para formar objetos dibujables a partir de cadenas de texto y su posición. Permitiendo así representar cualquier cadena posible con los caracteres y símbolos definidos, y permitiendo ajustar cosas como el tamaño o la posición con las transformaciones geométricas definidas anteriormente.

Menús Principal y de Opciones

El menú principal presenta las tres opciones básicas “PLAY”, “OPTIONS”, “EXIT” centradas dinámicamente en pantalla. Al iniciar se crea un objeto de texto para cada etiqueta y, en cada fotograma, se recalculan sus posiciones para mantener el menú siempre centrado tanto horizontal como verticalmente. El usuario navega con las flechas arriba/abajo y confirma con Enter; se dibuja un indicador “>” junto a la opción seleccionada y se reproducen efectos de sonido al mover o seleccionar.

El menú de opciones permite activar o desactivar tres flags “SOUND”, “ASTEROIDS”, “AI” y volver al menú principal “BACK”. Cada línea combina el texto de la opción con su valor actual (“ON” y “OFF”), espaciados uniformemente y centrados en el área de juego. La navegación es idéntica al menú principal, con flechas para alternar opciones o salir, y el mismo sistema de sonidos para feedback.





HUD

Durante toda la partida se muestr un HUD en la parte superior de la pantalla, mostrando 6 datos importantes para la partida igual que en el videojuego original:

- **Puntuación (SCORE):** contador de puntos acumulados.
- **Tiempo (TIME):** tiempo de juego en minutos y segundos.
- **Combustible (FUEL):** nivel actual de combustible.
- **Altitud (ALTITUD):** distancia vertical entre la nave y el terreno, calculada en tiempo real.
- **Velocidades (HORIZONTAL SPEED / VERTICAL SPEED):** componentes de la velocidad con una flecha direccional para indicar el sentido.

Menús Pausa y Fin de Juego

Además de los menús principales y de opciones, también se ha desarrollado otra clase de menú tipo *overlay*. Los cuales se dibujan sobre el juego en momentos concretos de la partida. Concretamente tenemos los menús de pausa y de fin de juego.Ambos muestran el tiempo de la partida actual y la puntuación, pero las opciones que ofrecen son distintas.

Por un lado, el menú de pausa, que aparece cuando el usuario presiona la tecla “Esc” durante la partida, muestra opciones para:

- Cerrar el menú y continuar la partida.
- Insertar una moneda
- Volver al menú principal.

Por otro lado, el menú de fin de juego es el que aparece cuando la nave se estrelle, y solo ofrece dos opciones:

- Insertar moneda, la cual en caso de seleccionarse reinicia la partida.
- Volver al menú principal

Físicas

Movimiento

El motor de físicas desarrollado para el juego utiliza un sistema de integración en intervalos fijos (16ms). Para actualizar la posición, velocidad y aceleración de cada objeto:

1. Se aplica la gravedad en el eje vertical.
2. Si el propulsor está activado y hay combustible, se añade aceleración en la dirección actual de la nave.
3. Se integra la velocidad: v += a·dt.
4. Se aplica fricción horizontal para simular la resistencia: ralentiza gradualmente la velocidad en X.
5. Se integra la posición: p += v·dt.
6. Se resetea la aceleración para el siguiente ciclo.

La nave es controlada a través de dos funciones de girar izquierda y derecha, las cuales modifican el ángulo de orientación en incrementos fijos y luego rota el dibujable de forma gráfica. El propulsor, activado por el jugador, modifica la aceleración de la nave y añade su dibujable, además está limitado por el combustible.

Colisiones

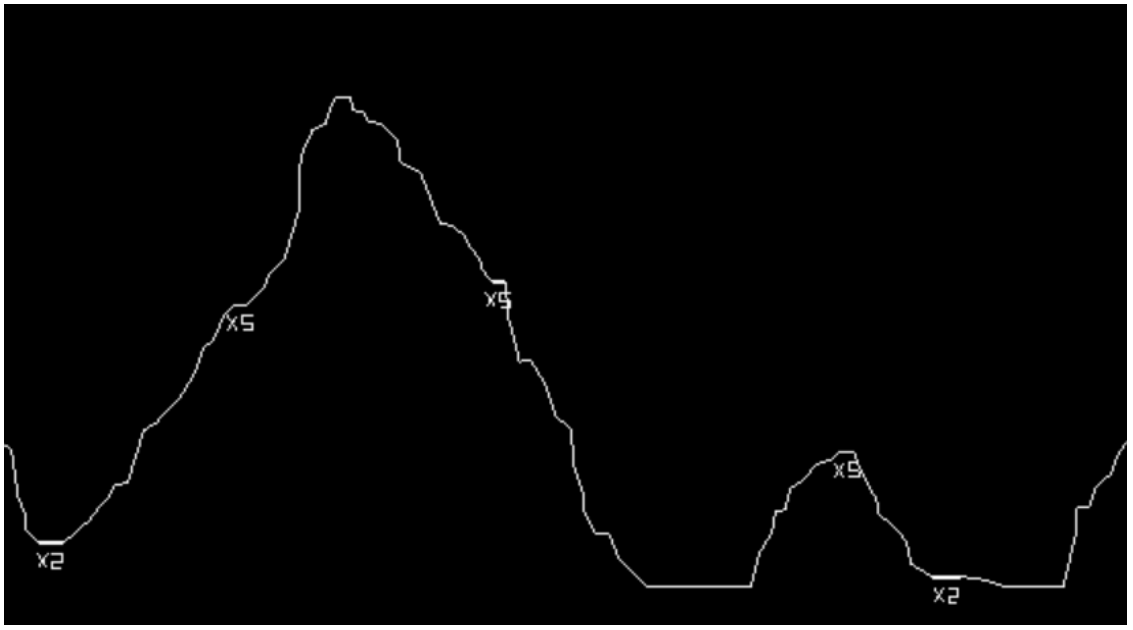
Tenemos ya definido cada objeto con sus puntos y aristas gracias a la clase Dibujable. Por lo que calcular colisiones es bastante simple, solamente es necesario buscar intersecciones entre todas las parejas de aristas de dos dibujables. Cuando

se detecta una colisión se registran también las aristas concretas que la produjo, para luego gestionar de una forma u otra la colisión.

Terreno

El mapa del videojuego (la superficie lunar) se almacena como un objeto de tipo Dibujable, con todos los puntos y aristas que definen la geografía de la superficie lunar. Cada uno de estos puntos y aristas se han definido a mano con el objetivo de seguir un estilo parecido en la medida de lo posible al mapa del videojuego original.

Este objeto se instancia al comienzo de la partida y se dibuja repetidamente desplazado horizontalmente para crear un efecto de “wrap around” continuo: cuando la nave avanza más allá de un borde, el terreno y los asteroides se trasladan en bloque en sentido contrario, manteniendo la ilusión de un terreno infinito.



Plataformas

Las plataformas son los tramos horizontales de terreno que otorgan un bonus al aterrizar sobre ellas. Aunque el terreno es fijo, la generación de plataformas a lo largo de este es dinámica.
Al iniciarse la partida se recorre la lista de puntos que define el terreno y se buscan segmentos consecutivos con la misma altura. Cada uno de estos tramos se almacena como candidato para ubicar una plataforma. Se selecciona un número definido de segmentos para ubicar plataformas y se le asigna un multiplicador en función de su tamaño.

Cuando la nave colisiona con el terreno, se detecta si el impacto ha ocurrido con una arista seleccionada como plataforma o con terreno normal para hacer el cálculo de puntos.

Sonido

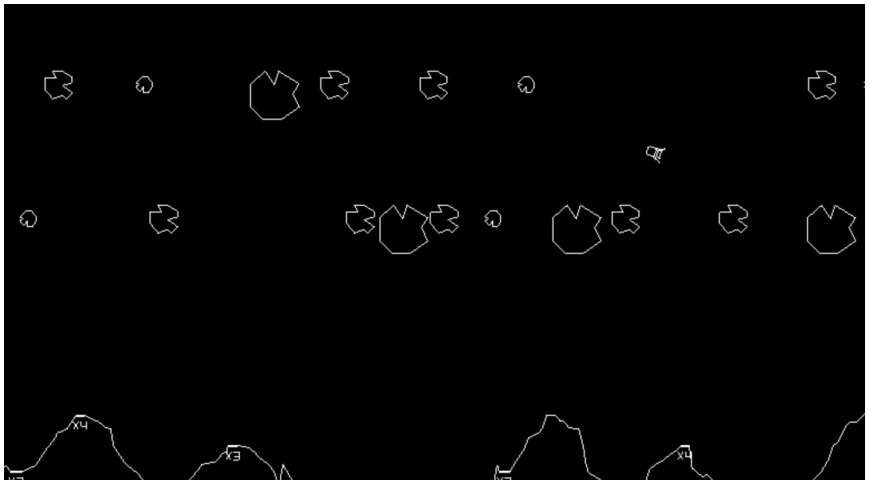
El juego cuenta con la opción de activar o desactivar el sonido, accesible desde la opción “SOUND” del menú de opciones. Cuando se activa, el videojuego hace uso de una serie de efectos de sonido sencillos para:

- Activar el propulsor
- Introducir monedas
- Colisiones
- Moverse por un menú
- Seleccionar una opción en un menú

Los sonidos se almacenan como archivos WAV y se gestionan a través de llamadas a la API de Windows, encargada de cargar los sonidos en memoria e iniciar y detener su reproducción.

Asteroides

Una de las funcionalidades nuevas incluidas en esta versión de Lunar Lander es la poder incluir 2 cinturones de asteroides que aumentan la dificultad del aterrizaje. Esta función es activada a través del menú de opciones y genera dos cinturones de asteroides a lo largo de los cuales se distribuye de forma aleatoria un número fijo de asteroides de distintos tamaños. Una colisión con un asteroide se considera equivalente a una colisión con la superficie lunar. Además, a los asteroides se les aplica un wrap-around, de forma que cuando salen por un extremo del terreno reaparecen en el otro, dando así la impresión de continuidad en el mapa.



Piloto Automático

Una de las funcionalidades nuevas incluidas en esta versión de Lunar Lander es la de un piloto automático. Esta opción se activa desde el menú de opciones, y cuando está activa le quita el control de la nave al jugador y esta pasa a pilotar de forma automática. La IA seleccionará una plataforma objetivo de forma aleatoria e intentará aterrizar sobre ella.
La versión final de la IA de piloto automático se ha desarrollado utilizando únicamente una máquina de estados presente en *ai.c*. La cual divide el proceso de aterrizaje en una serie de fases, en su mayoría secuenciales. Por ejemplo, mantenerse a cierta altura, desplazarse horizontalmente hasta una plataforma, estabilizar la nave, y descender suavemente.

No se ha llegado a conseguir implementar una IA de aterrizaje automático que aterrice sobre una plataforma aleatoria de forma exitosa un porcentaje de veces. Este tema se desarrolla en el apartado de principales dificultades.

Principales Dificultades

Generación Aleatoria del Terreno

Durante el desarrollo del videojuego se creó a mano un pequeño terreno para hacer pruebas de distintas funcionalidades. El terreno final del juego se pretendía que fuera generado de forma aleatoria, pero tras probar con distintos métodos, principalmente con variaciones de perlin noise, no se consiguió un resultado lo suficientemente satisfactorio. Finalmente se optó por expandir a mano el terreno de pruebas, lo cual produce unos resultados mejores, pero requiere tiempo y ajustes. También se estableció un punto horizontal de spawn aleatorio para la nave para darle una sensación de aleatoriedad a cada partida. Sin embargo, la generación de plataformas se que se hace de forma dinámica, soportando una generación aleatoria del terreno.

Inteligencia Artificial

La mayor dificultad encantada durante este proyecto ha sido la implementación de una IA de aterrizaje automático. Rápidamente después de considerar varias opciones se llegó a la conclusión de que la opción óptima para implementar este sistema sería a través de una máquina de estados y múltiples controladores PID. Sin embargo, este enfoque trae una serie de dificultades que no habíamos previsto.

- En caso de utilizar controladores PID sería necesario ajustar sus valores iniciales y parámetros de forma muy precisa, ya que cualquier pequeño error haría fracasar el aterrizaje.
- También habría que ajustar de forma muy fina los criterios de transición entre estados y los comportamientos de estos.
- Sería necesario configurar y ajustar múltiples controladores para los distintos estados.
- Este ajuste y configuración sistema. Se optó por una IA que se colocará encima de la plataforma y se dejase caer verticalmente, en línea recta y con la menor velocidad horizontal posible.

Reparto de Tareas y Dedicación

Ambos miembros del equipo hemos colaborado en prácticamente todas las partes del proyecto. Para cada nueva funcionalidad hemos creado una rama de Git, lo que nos ha permitido desarrollar en paralelo sin pisarnos el trabajo. Cada uno ha ido alternando la implementación de las distintas funcionalidades, compartiendo los avances a través de GitHub. De este modo hemos podido trabajar de forma eficiente, sin llegar a trabajar sobre la misma característica al mismo tiempo.

	Fabián	Alex
Dibujado	9h	4h
Menús e Interfaz	11h	2h
Físicas	2h	5h
Terreno y Plataformas	9h	7h
Sonido	1h	6h
IA	9h	1h
Asteroides	2h	10h
Documentación	4h	4h
Total	**54**	**39**