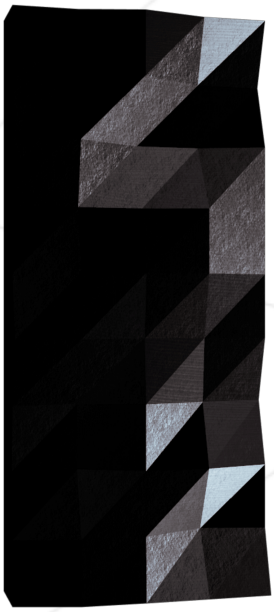


FDF

Resumen: Este proyecto consiste en crear una representación tridimensional de un paisaje con relieve.

Versión: 2

Índice general

Capítulo I

Avance

Esto es lo que dice Wikipedia sobre `Ghosts'n Goblins`:

`Ghosts'n Goblins` es un juego de plataformas donde el jugador controla a un caballero, llamado Sir Arthur, que tiene la habilidad de arrojar lanzas, dagas, antorchas, hachas y otras armas con las que debe derrotar a zombis, demonios y otras criaturas fantasmagóricas para poder rescatar a una princesa. Cada vez que un monstruo toca su armadura ésta se rompe, quedando el personaje "en calzoncillos" y expuesto a cualquier ataque. A lo largo del camino, el jugador puede coger nuevas armas, bonus y trajes extra de armadura que pueden ayudar en esta tarea. Sir Arthur aparece también en el juego "Marvel vs. Capcom" como partner.

El juego a menudo es considerado muy difícil para los estándares de los arcade y es comúnmente considerado como uno de los juegos más difíciles lanzados para Nintendo (NES). El juego tiene un modo de dos jugadores, pero éste consiste simplemente en alternar entre cada jugador. El jugador empieza con 3 vidas del personaje (la del personaje en juego y dos más), y además existe otra vida extra cuando el jugador alcanza 20,000 y 70,000 puntos después de eso. Sin embargo, esta configuración podía cambiarse usando los interruptores DIP.

Dos golpes de un ataque enemigo resultan en la pérdida de una vida. Después del primer golpe, Sir Arthur pierde su armadura. Un segundo golpe lo transforma irremediablemente en un esqueleto y pierde, de esa manera, una vida. Al principio de cada nivel, Sir Arthur está envuelto en una armadura completa independientemente de si está con ella o no al finalizar cada nivel. En ciertos puntos del juego, Sir Arthur puede sufrir una muerte instantánea ya sea que esté con su armadura o no.

Si el jugador pierde una vida, se le devuelve al inicio del nivel, o a la mitad del mismo si se ha llegado hasta ese punto del escenario. Además, hay un límite de tiempo para acabar el nivel (generalmente alrededor de tres minutos). Si éste se acaba, el jugador pierde una vida y tiene que volver a empezar. El reloj se reinicia al principio de cada nivel.

- Portabilidad

  Muchas conversiones a ordenadores domésticos fueron producidas por Elite Systems.

  - La versión de Commodore 64, publicada en 1986, es conocida por la música de Mark Cooksey, quien se inspira en el Preludio No. 20 de Frédéric Chopin. Dados los recursos limitados de la Commodore 64, fue algo distinta de la versión arcade. Cuenta con un número reducido de zonas. El jugador empieza el juego con cinco vidas. El demonio que secuestra a la princesa reemplaza a Astaroth en la pantalla del título. Adicionalmente, el cíclope (o "Unicorno") es el jefe de los niveles uno al tres, siendo el dragón el enemigo final.
  - La versión de Commodore 16/116 y Commodore Plus/4, publicadas también en 1986 por Elite Systems, fue incluso más limitada que la versión de C64. Se escribió para funcionar en Commodore 16, que tenía solo 16KB de RAM.
    Por lo tanto, esta versión cuenta con solo dos niveles y música. Adicionalmente, los dos niveles restantes y la jugabilidad están simplificados. Por ejemplo, en el nivel del cementerio, el pájaro atacante, los "monstruos planta" y el demonio alado no existen en la versión de C16, que cuenta con un solo arma. La pantalla del título no cuenta con más gráficos que las letras de Ghosts'n Goblins estilizadas.
  - Una versión para la Commodore Amiga fue publicada en 1990. Mientras que el avanzado hardware de la Amiga permitía casi conversiones perfectas de los juegos arcade, fallaba al emular el éxito de la versión de Commodore 64. El jugador empiezaba el juego con seis vidas y sin música a menos que la Amiga estuviera equipada con al menos 1MB de RAM.
    La configuración estándar de una Amiga 500 era 512KB de RAM.

  - La versión de NES fue desarrollada por Micronics. Esto también sirve como los principios de la versión de Game Boy Color, la cual utiliza códigos para permitir al jugador saltarse niveles. La versión de NES fue portada a la Game Boy Advance como parte de la serie de Clásicos NES, solo en Japón.
  - La versión de NES se publicó otra vez para descarga de la Nintendo Virtual Console en norteamérica el 10 de Diciembre del 2007 (Wii) y el 25 de Octubre de 2012 (Nintendo 3DS) y en la región PAL el 31 de Octubre del 2008 (Wii), también el 3 de Enero de 2013 (Nintendo 3DS) mientras que la versión de Wii U salió en ambas regiones el 30 de Mayo de 2013. La versión arcade salió en la Virtual Console Arcade de Wii en Japón el 16 de Noviembre de 2010, y en la región PAL el 7 de Enero de 2011. Por último, en norteamética el 10 de Enero de 2011.
  - Ghosts'n Goblins fue también portada a la ZX Spectrum, Amstrad CPC, MSC, Atari ST, compatiblce con IBM PC, Game Boy Color, Game Boy Advance. La versión arcade original del juego se incluía también en la compilación Capcom Generations Vol.2: Chronicles of Arthur para la PlayStation (en Japón y Europa) y la Sega Saturn (solo en Japón), que también contenía Ghouls'n Ghosts y Super Ghouls'n Ghosts. Los tres juegos (basados en las versiones de Capcom Generation) fueron más tarde coleccionados como parte de la serie Capcom

Classic. El juego se publicitó en la compilación Capcom Arcade Cabinet para la PlayStation 3 y la Xbox 360.

- Recepción

Computer Gaming World llamó a Ghosts'n Goblins "un ejemplo excelente de lo que la [NES] puede hacer ... mientras vagamente representa el tipo de juego que hizo a Nintendo famoso". Ghosts'n Goblins fue segundo en la categoría de estilo Arcade en los GOTY (Game of the Year) en la Golden Joystick Awards. La versión de NES de Ghosts'n Goblins quedó en el puesto 129 como mejor juego de Nintendo Systems en la Nintendo Power'sTop 200 Games List. Fue también el juego más vendido de la NES, con 1.64 millones de copias. Ghosts'n Goblins se usa frecuentemente como ejemplo de uno de los juegos más difíciles de la historia dada su extrema dificultad y el hecho de que el jugador debe pasarse el juego dos veces para completarlo.

- Legado

  Ghosts'n Goblins fue seguido de una serie de secuelas y spin-offs, haciendo que eventualmente fuera el octavo juego mejor vendido de la franquicia de Capcom con 4.4 millones de unidades. Su secuela incluye Ghouls'n Ghosts, Super Ghouls'n Ghosts y Ultimate Ghosts'n Goblins. Crearon además Gargoyle's Quest y los spin-off de la saga Maximo. Sin embargo, aunque se originó como un título arcade, la franquicia se ha publicitado en una amplia variedad de juegos de PC y consolas con la última entrega de la serie, Ghosts'n Goblins: Gold Knights, publicada en iOS. Adicionalmente, la franquicia frecuentemente hace cameos con el personaje Arthur en otros títulos de Capcom, el último de ellos Ultimate Marvel vs. Capcom 3.

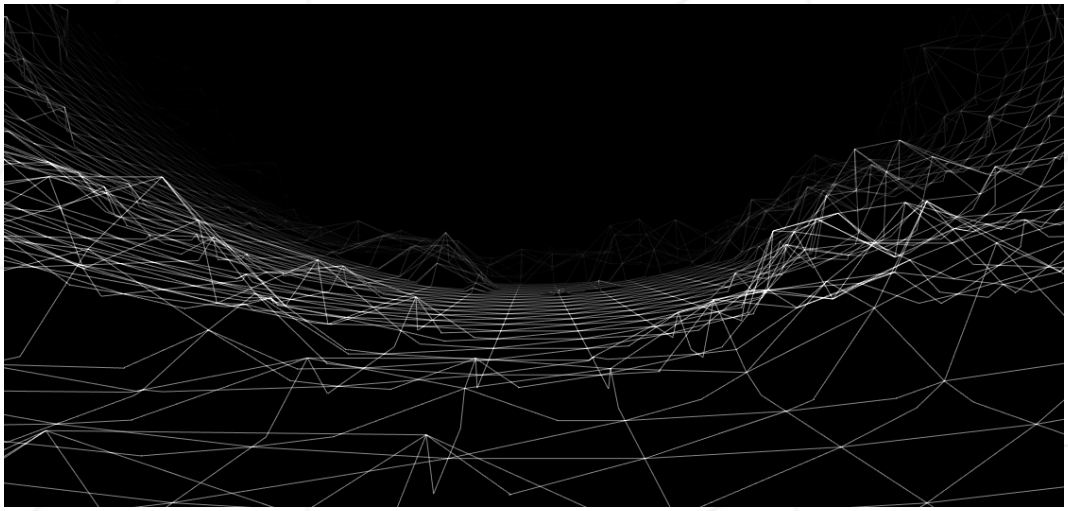
Figura I.1: Portada del juego

Capítulo II

Introducción

La representación tridimensional de un paisaje es un aspecto crítico en el el mapping moderno. Por ejemplo, en los tiempos de la exploración espacial, tener una representación tridimensional de Marte es un requisito necesario para su conquista. Como ejemplo adicional, al comparar varias representaciones tridimensionales de áreas con alta actividad tectónica podrás entender mejor estos fenómenos y su evolución, y como resultado estar mejor preparado.

Es tu turno de modelar en tres dimensiones algunas paisajes impresionantes, ya sean imaginarios... O no.



Capítulo III

Objetivos

En este proyecto descubrirás los principios de la programación gráfica, y en particular cómo colocar puntos en el espacio, cómo unirlos con segmentos y sobre todo, cómo observar la escena desde un punto de vista particular.

Descubrirás también tu primera librería gráfica: `miniLibX`. Esta librería fue desarrollada internamente e incluye los elementos más básicos: abrir una ventana, iluminar un píxel y trabajar con eventos enlazados a esta ventana a través del teclado y el ratón. Este proyecto introduce programar con "eventos".

Capítulo IV

Instrucciones generales

- Tu proyecto debe estar escrito siguiendo la Norma. Si tienes archivos o funciones adicionales, estas están incluidas en la verificación de la Norma y tendrás un `0` si hay algún error de norma dentro.
- Tus funciones no deben terminar de forma inesperada (segfault, bus error, double free, etc) ni teener comportamientos indefinidos. Si esto pasa tu proyecto será considerado no funcional y recibirás un `0` durante la evaluación.
- Toda la memoria alocada en heap deberá liberarse adecuadamente cuando sea necesario. No se permitirán leaks de memoria.
- Si el subject lo requiere, deberás entregar un `Makefile` que compilará tus archivos fuente al output requerido con las flags `-Wall`, `-Werror` y `-Wextra`, por supuesto tu `Makefile` no debe hacer relink.
- Tu `Makefile` debe contener al menos las normas `$(NAME)`, `all`, `clean`, `fclean` y `re`.
- Para entregar los bonus de tu proyecto, deberás incluir una regla `bonus` en tu `Makefile`, en la que añadirás todos los headers, librerías o funciones que estén prohibidas en la parte principal del proyecto. Los bonus deben estar en archivos distintos `_bonus.{c/h}`. La parte obligatoria y los bonus se evalúan por separado.
- Si tu proyecto permite el uso de la `libft`, deberás copiar su fuente y sus `Makefile` asociados en un directorio `libft` con su correspondiente `Makefile`. El `Makefile` de tu proyecto debe compilar primero la librería utilizando su `Makefile`, y después compilar el proyecto.
- Te recomendamos crear programas de prueba para tu proyecto, aunque este trabajo **no será entregado ni evaluado**. Te dará la oportunidad de verificar que tu programa funciona correctamente durante tu evaluación y la de otros compañeros. Y sí, tienes permitido utilizar estas pruebas durante tu evaluación o la de otros compañeros.
- Entrega tu trabajo a tu repositorio `Git` asignado. Solo el trabajo de tu repositorio `Git` será evaluado. Si Deepthought evalúa tu trabajo, lo hará después de tus compañeros. Si se encuentra un error durante la evaluación de Deepthought, la evalaución terminará.

- Este proyecto solo será corregido por humanos. Siéntete libre de organizar y nombrar a tus archivos como quieras... Respetando las normas dadas aquí.
- El nombre del ejecutable será `fdf`.
- Debes enviar un `Makefile`.
- No puedes utilizar variables globales.
- Tienes que utilizar la `miniLibX`. Ya sea la versión disponible en el sistema, o con su fuente. En caso de querer trabajar con la fuente, deberás respetar las mismas reglas que con el `libft`, escritas arriba.
- Para la parte obligatoria, puedes utilizar las siguientes funciones:
  - `open, read, write, close`
  - `malloc, free`
  - `perror, strerror`
  - `exit`
  - Todas las funciones dadas por la librería `math` (-lm and man 3 math)
  - Todas las funciones definidas en la `miniLibX`
- Puedes utilizar otras funciones para completar la parte extra siempre y cuando su uso esté justificado durante tu evaluación. Sé inteligente.
- Puedes preguntar en Slack, Google, a tus compañeros, etc.

Capítulo V

Parte obligatoria

Este proyecto consiste en crear un “marco” gráfico simple (“wireframe” en inglés o “fils de fer” en francés, de ahí el nombre del proyecto). Este representará el relieve de un paisaje enlazando varios puntos `(x, y, z)` a través de segmentos. Las coordenadas de este paisaje se guardan en un archivo pasado como parámetro a tu programa. Aquí tienes un ejemplo:

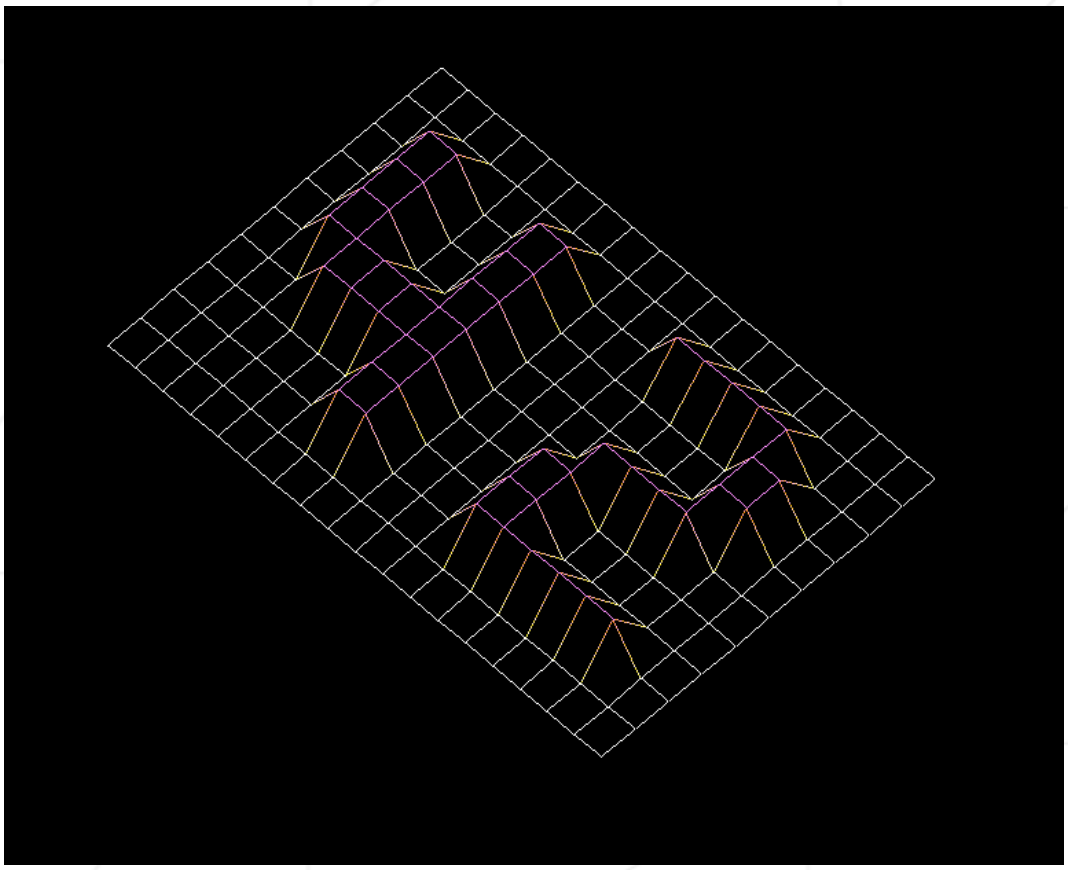
```
$>cat 42.fdf
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0 10 10  0  0 10 10  0  0  0 10 10 10 10 10 0 0  0
0  0 10 10  0  0 10 10  0  0  0  0  0  0  0 10 10 0  0
0  0 10 10  0  0 10 10  0  0  0  0  0  0  0 10 10 0  0
0  0 10 10 10 10 10 10 0 0  0  0 10 10 10 10 0 0  0
0  0  0 10 10 10 10 10 0 0  0 10 10 0  0  0  0  0  0
0  0  0  0  0  0 10 10 0  0  0 10 10 0  0  0  0  0  0
0  0  0  0  0  0 10 10 0  0  0 10 10 10 10 10 10 0 0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
$>
```

Cada número corresponde a un punto en el espacio:

- La posición horizontal corresponde a su eje.
- La posición vertical corresponde a su eje de coordenadas (Y).
- El valor corresponde a su altitud.

Si quieres ejecutar tu programa `fdf` con este archivo, deberías ver algo como esto:

```
$>./fdf 42.fdf
$>
```



Recuerda hacer uso óptimo de tu librería `libft`. El uso de `get_next_line`, `ft_split` y `ft_atoi` te permitirá fácilmente leer el archivo.

Recuerda que el objetivo de este proyecto no es analizar mapas, aunque esto no significa que tu programa pueda fallar cuando se ejecute... Esto significa que el mapa contenido en el archivo debe estar correctamente formateado.

Con respecto a la representación gráfica:

- Tu fdf debe mostrar el mapa utilizando proyección isométrica.
- Debes poder terminar el programa al presionar `ESC`.
- El uso de `imágenes` de la `miniLibX` no se requiere para validar el proyecto, aunque te recomendamos utilizarlas.
- Localiza el binario adjunto llamado `fdf` y el ejemplo `42.fdf`.

`man mlx`

Capítulo VI

Parte extra

Habitualmente te recomendaríamos crear tus bonus originales, pero hay otros proyectos gráficos mucho más interesantes esperándote. No te vuelvas loco aquí, avanza rápido.

Ganarás puntos de experiencia adicionales si:

- Incluyes una proyección adicional (por ejemplo: paralela o cónica)
- Puedes hacer zoom y trasladar tu mapa
- Puedes rotar tu mapa

Capítulo VII

Entrega y evaluación de pares

Entrega tu trabajo en tu repositorio `Git` como ya es habitual. Solamente el trabajo en tu repositorio será evaluado.

Buena suerte ⌣̈