ANDRES MATA BRETÓN 2019

INTRODUCCIÓN

CPCRogue es un juego tipo “Rogue-like” como se conoce en inglés. Este tipo de juegos tiene sus orígenes a finales de los 70 principios de los 80 cuando los terminales estaban orientados a texto. Este tipo de juegos viene a ser el precursor de los juegos de rol actuales, pero con unas características particulares:

- La representación gráfica suele estar realizada con caracteres ASCII
- Los niveles se generan aleatoriamente
- El “mundo” no es conocido en su totalidad, sino que va apareciendo a medida que se explora
- El jugador va progresando de nivel a medida que elimina los enemigos
- Suele haber diferentes tipos de elementos que el jugador puede utilizar para ayudarle en su tarea: pociones, comida, armas, armadura, etc.

CPCRogue comenzó siendo mi versión personalizada de la *Amstrad GameDev Challenge* (AGC), organizado por Fran Gallego y Hector Linares. Siguiendo la filosofía de la AGC, en principio la idea era realizar un juego sencillo en lenguaje Basic y C, pero enseguida me decanté por realizar un juego “Rogue-like” más completo y centrarme en una versión única en C.

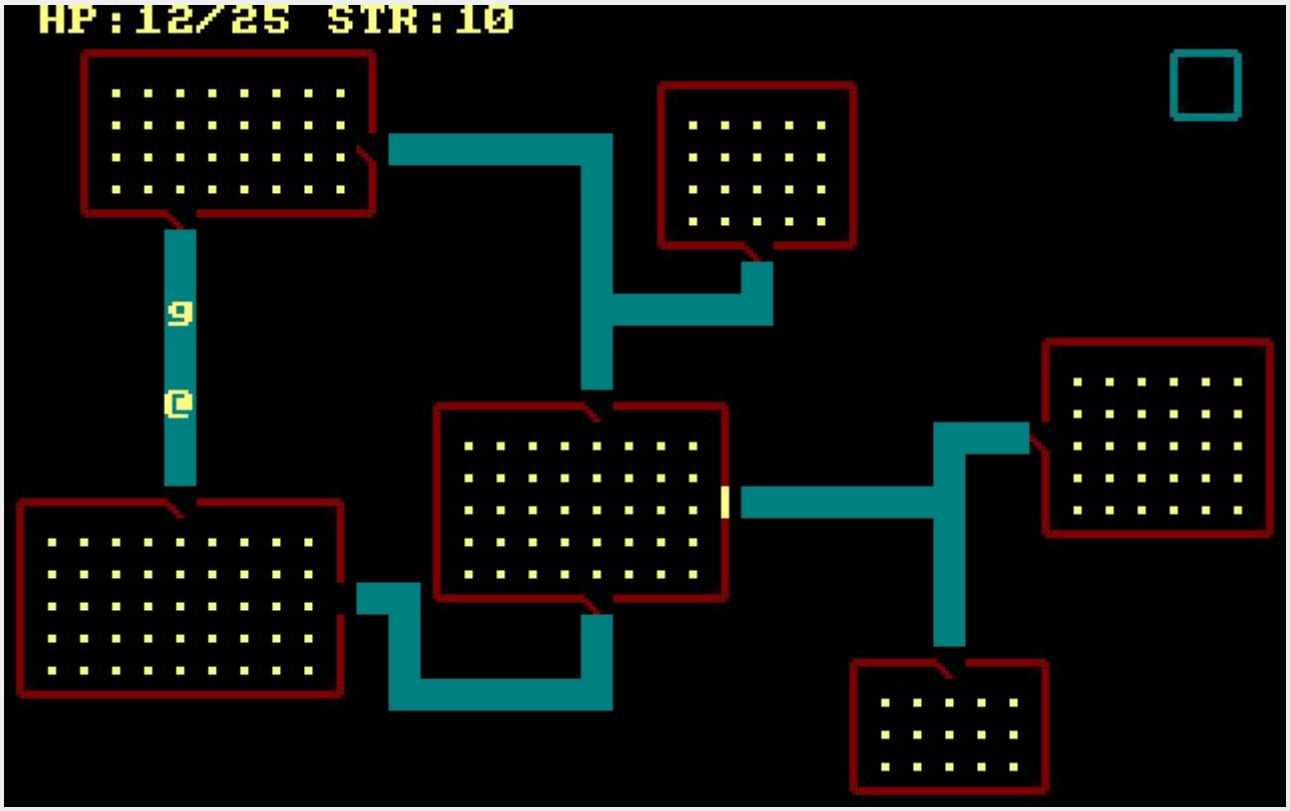
Mi inspiración para realizar el juego ha sido el juego **Rogue - The Adventure Game** considerado quizá como el clásico *rogue-like* por excelencia. De ahí que la estética sea muy parecida.

Debido a su naturaleza tipo rol, el juego contiene muchos elementos no predeterminados y varios de azar. Entre ellos el personaje principal (el jugador) que no tiene un nombre predeterminado, sino que lo elegimos al comenzar la partida. Otra característica implementada es la generación aleatoria de niveles.

GRAFICOS

Como se ha comentado, los gráficos del juego están realizados con caracteres de texto, en concreto en **Modo 1**. De hecho se han utilizado las propias rutinas que *CPCtelera* dispone para escribir textos y caracteres en pantalla. Viéndolo ahora en perspectiva y conociendo mejor las limitaciones de velocidad -y la gran cantidad de espacio que pueden llegar a ocupar estas rutinas- no las habría utilizado y habría optado por usar *sprites* reales. Eso lo dejo para *CPCRogue 2* en el que ya estoy trabajando…

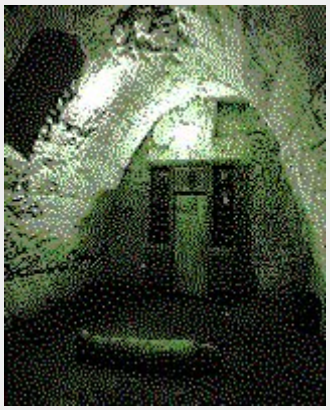
Antes de llegar a la versión definitiva, realicé varios "MOCAPS" con el programa *REXPaint* para ver diferentes alternativas de visualización. Como se comentará más adelante, en un principio mi objetivo era implementar iluminación dinámica limitada por la línea de visión del personaje. Pero también exploré una alternativa visual más al estilo *Rogue The Adventure Game*, que es la que he acabado usando para la versión final. En las siguientes imágenes puede verse algunos "MOCAPS" explorados:



Para la pantalla de carga, he de reconocer que recurrí al método rápido de convertir una imagen real a formato CPC. Para esto recurrí al programa *MTPaint*, un programa quizá no muy conocido, pero que permite trabajar con paletas y reducir colores de una forma ágil y rápida.

Sí que dediqué un tiempo en optimizar a mano las imágenes y observar los resultados según los diferentes métodos de *cuantización* del programa hasta llegar a un resultado satisfactorio. También en este caso he optado por usar el **Modo 1,** que a pesar de sus limitados 4

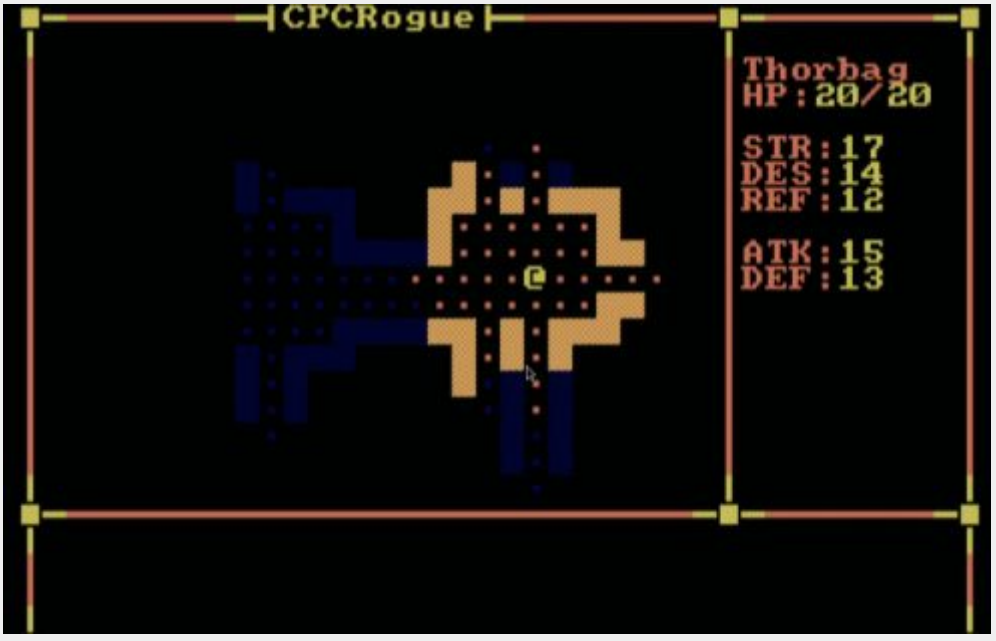
colores, me gusta más el resultado gráfico debido a su mayor resolución. En las siguientes imágenes puede verse el antes y después de la conversión de la pantalla de carga usada:

ASPECTOS TÉCNICOS

Al tratarse de mi primer juego, *CPCRogue* ha sufrido múltiples cambios y re-escrituras desde sus primeros bocetos. También me ha servido para darme cuenta que no tenía ni idea de programar en C. La mayor parte de los *crashes* que tuve durante el desarrollo se debía a que no entendía realmente cómo funcionaban los punteros en C (yo pensaba que sí).

En un principio quería implementar iluminación progresiva al recorrer el mapa -lo que se conoce como *level of sight*- es decir, dibujar con colores más claros las zonas en nuestra línea de visión y colores más oscuros las zonas fuera de ésta. En las siguientes pantallas puede verse el efecto:



Aunque visualmente queda muy “resultón”, tuve que descartarlo por ser prohibitivamente costoso y lento. Tuve que plantearme de nuevo toda la creación y visualización del mapa desde cero. En la versión final, simplemente se redibujan las “celdas” alrededor del jugador hasta una cierta distancia. El efecto no es tan atractivo, pero es mucho más rápido de dibujar:

A nivel de mecánicas, se ha intentado implementar un sistema basado en componentes no confundir con ECS (Entidad/Component/System). Así, las entidades tienen un puntero a diferentes componentes (componente AI, componente Item, etc.) y cada componente tiene su comportamiento específico. Combinando diferentes componentes pueden realizarse entidades más o menos complejas.

Por ejemplo, los componentes *Item* tienen una acción asociada que es la de *coger* ese item. En cambio, un componente *AI* tiene una acción asociada que es la de perseguir al personaje principal. De este modo, una entidad “Oro” tiene un componente *Item* -con lo que se puede coger- pero no tiene un componente *AI*. Una entidad “Goblin” tiene un componente *AI* que persigue al jugador y no tiene componente *Item*. No obstante, podría tenerlo, con lo que tendríamos un “Goblin” que una vez matado se podría coger (qué hacer con el cadaver de un goblin es algo que aún estaría por decidir…)

Este sistema tiene ventajas e inconvenientes. Por un lado es bastante flexible, pero me ha resultado bastante complicado implementarla de forma correcta en C. De hecho, estoy seguro que quedan bastantes *bugs* todavía. Pero sí que es cierto, que una vez implementado, añadir diferentes tipos de *items* y enemigos ha resultado relativamente sencillo y rápido.