OPQA Vs. QAOP

The Final Battle

Cómo se hizo

Idea original

A principios de este año, tenía pensado hacer un juego muy distinto para la CPCRetroDev. Era sobre un pequeño robot que debía escapar de una fábrica superando pantallas con distintos obstáculos. Quería hacerlo en modo 1 con movimiento al píxel y después de una semana me di cuenta de que era bastante ambicioso para los conocimientos que tenía sobre el CPC, por lo que terminé dejándolo aparcado.

A finales de verano intenté retomar el proyecto, pero decidí que era un buen momento para pensar otra idea más sencilla y realizable en el tiempo que me quedaba. Como siempre me ha gustado el debate (absurdo) entre OPQA y QAOP, pensé que era un momento perfecto para usarlo en el juego, y como me gustan los juegos de puzle, hacer algo similar a Puyo Puyo lo veía interesante y viable.

Prototipo

Teniendo clara la nueva idea de juego, fui elaborando un prototipo sencillo. Primero vi la generación aleatoria de los bloques, luego hice que fuesen cayendo en el tablero, y quizás lo más complicado fue comprobar que iban haciendo combinaciones del mismo color. Como no es necesario que estuviesen en línea, tuve que hacer una función recursiva optimizada (para no abusar de la pila) en la que iba viendo las cuatro direcciones hasta que ya no encontraba fichas del mismo color.

Implementando esta funcionalidad el juego empezaba a funcionar, aunque si me hubiese quedado aquí, el juego para un jugador habría sido algo aburrido...

IA

Otro pilar fundamental del juego era crear una inteligencia artificial lo suficientemente "lista" para que cada partida fuese un reto, pero no demasiado como para que el jugador sienta que ha sufrido una derrota injusta.

Al final opté por una sencilla máquina de estados en la que se buscaba la mejor posición para dejar el bloque actual (teniendo como objetivo hacer columnas del mismo color), luego se movía y se giraba el bloque y finalmente se dejaba caer. En cada fase se modifican los parámetros para que cada vez el jugador controlado por la máquina haga caer más rápido la ficha.

Después de diseñar la IA, me di cuenta de que podía asignarla a los dos jugadores y así pude implementar de forma sencilla un modo demo que salta automáticamente cuando termina la música del menú.

Músicas y modos de juego

Para dar más alegría al juego, creé algunas músicas para el menú, el cambio de fase, etc. En el modo de juego principal no puse ninguna porque reproducía los efectos de sonido en estéreo (usando los canales A y C), y además no quería que un tema "machacón" hiciese pesadas las partidas.

Nunca había compuesto nada, pero echando un ojo a los temas de ejemplo y a algunos MODs de Amiga pude ver algunos patrones interesantes y componer los temas del juego me llevó menos de una semana.

En cuanto a los modos de juego, desde el principio tenía claros los modos básicos: un jugador contra la máquina y dos jugadores enfrentados. Como hice que el juego se acelerara a medida que íbamos encadenando combinaciones de color, un modo "supervivencia" sería interesante de cara a ver puntuaciones máximas.

Más tarde, investigando las versiones originales de Puyo Puyo (para Famicom Disk System y MSX2) vi que incluían un interesante modo misiones. Me basé en algunas de ellas, aunque posteriormente fui pensando en otras basadas en puntuación para que se tuviesen que realizar en orden y de forma que se usasen técnicas para conseguir más puntos.

Optimizaciones y retoques finales

Después de meter los gráficos de los avatares, tipos de piezas, el fondo de pantalla, la música, las mecánicas de juego, etc... me fui quedando sin memoria.

Las primeras optimizaciones consistieron en limpiar código y refactorizar algunas funciones. Posteriormente, comprimí el mapa de tiles del fondo y pasó de ocupar mil bytes a unos ochenta.

Para hacer esto último fácilmente, usé la versión 1.5 de CPCtelera, que en el momento de escribir estas líneas requiere hacer un *checkout* de la rama development del repositorio. Como ventajas adicionales, fue más sencillo crear un CDT personalizado con pantalla de carga, y la creación de ficheros SNA en cada compilación hizo que probar cada versión fuese más rápido.

Unos par de consejos para aspirantes a programadores del CPC: no subestiméis la potencia de este ordenador y no pretendáis hacer un juego complejo como vuestro primer proyecto.