MAKING-OF

ROBOBIT

Cristina Rivera Baydal
Miguel Sancho Peña
Fernando Verdejo Moreno
Subject: Automated Reasoning
2016 / 2017
Degree in Computer Engineering

CONTENTS

1. Description of how the game was created

The game was created by Cristina Rivera, Miguel Sancho y Fernando Verdejo.

The game was developed in 5 weeks with any previous knowledge of CPC Amstrad and very little knowledge of ASM. We firstly developed entities and moving functionality for these, and then we started working on the chase algorithm and state machine for the enemies. Due to the lack of time, we didnt spend much on unit testing.

2. Technologies used

We started learning how to develop programs for CPC Amstrad by watching some videos on YouTube. The members of our team had been developing in languages such as C/C++ and little ASM, so it was not too hard to learn how to manage pointers and that kind of C stuff. We learned how video memory was implemented in CPC and how much memory was available for us to use. The chosen video mode was mode 0 (160x200).

We used SDCC as compiler although none of us had ever used it before.

For the music, Arkos Tracker was used and music's completely original.

For license and sharing matters, and under the rules of this contest, our code is destributed under GPL license.

For sprites, graphics and maps Tiled was used and all drawings all original.

3. Problems and solutions found

We spend a lot of hard hours. The main problem that we had was the optimization. Because our game takes a lot of space memory.

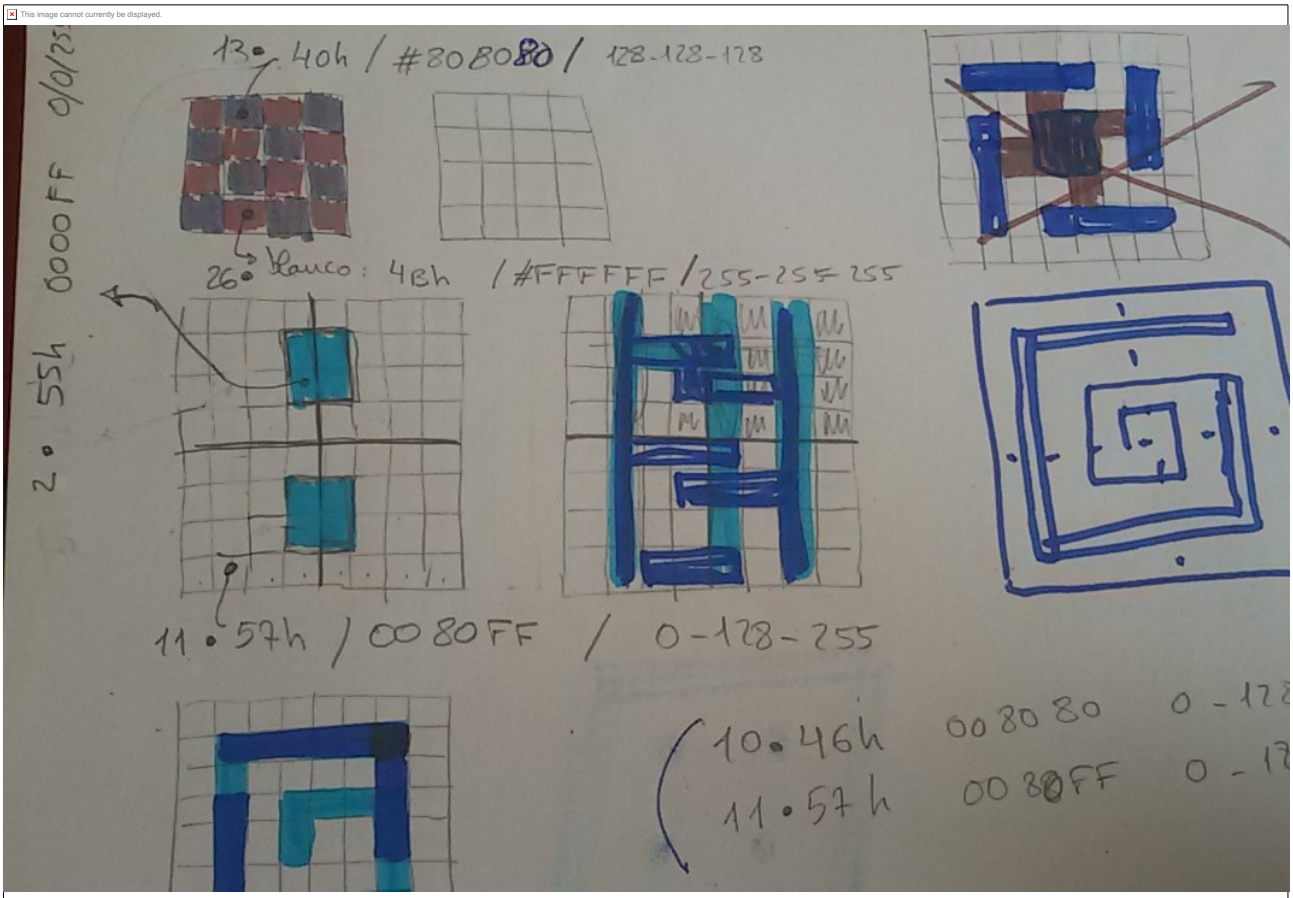
Our main problem was with the limited capabilities of the hardware and the extension of our game, we initially wanted to add many more enemies but we soon found out that it wasn't viable. The main reason for this was that we have a complex AI system which functions using a state machine. The state machine itself isn't a problem but for the AI to work naturally we had to specify different behaviours for each state and some of them, for example the pathfinding algorithm is very computationally heavy so we determined that we could not use that algorithm for every movement of the enemy. That is when we developed a chase algorithm which is an heuristic algorithm which executes very rapidly but loses some accuracy. When designing the algorithm, the main problem was calculating distances as those calculations can get very slow if you start using squared roots so we had to use faster calculations.

4. Learned lessons

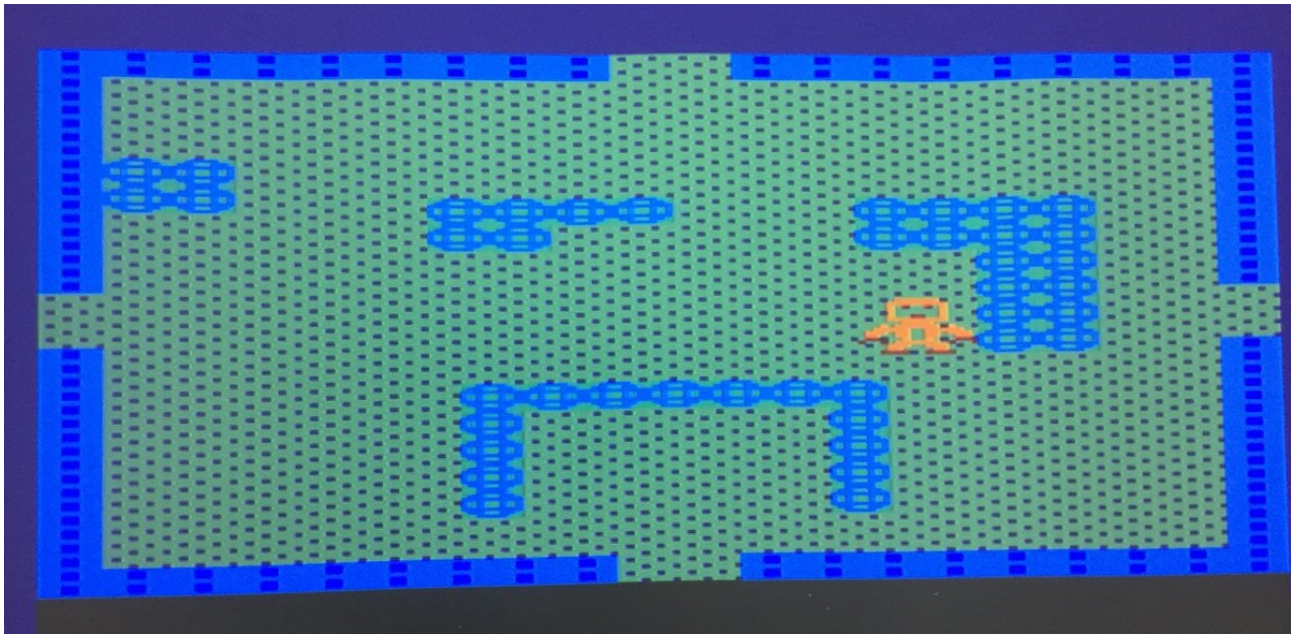
We have learned that knowing the machine is your best ally, it will vastly improve the way you program and the way your code executes.
We have also learned that each bit counts and that optimisation is everything when having such a limited amount of memory and computational power.
Lastly we have refreshed and improved our assembly skills.

5. Pictures and screen-shots of the development

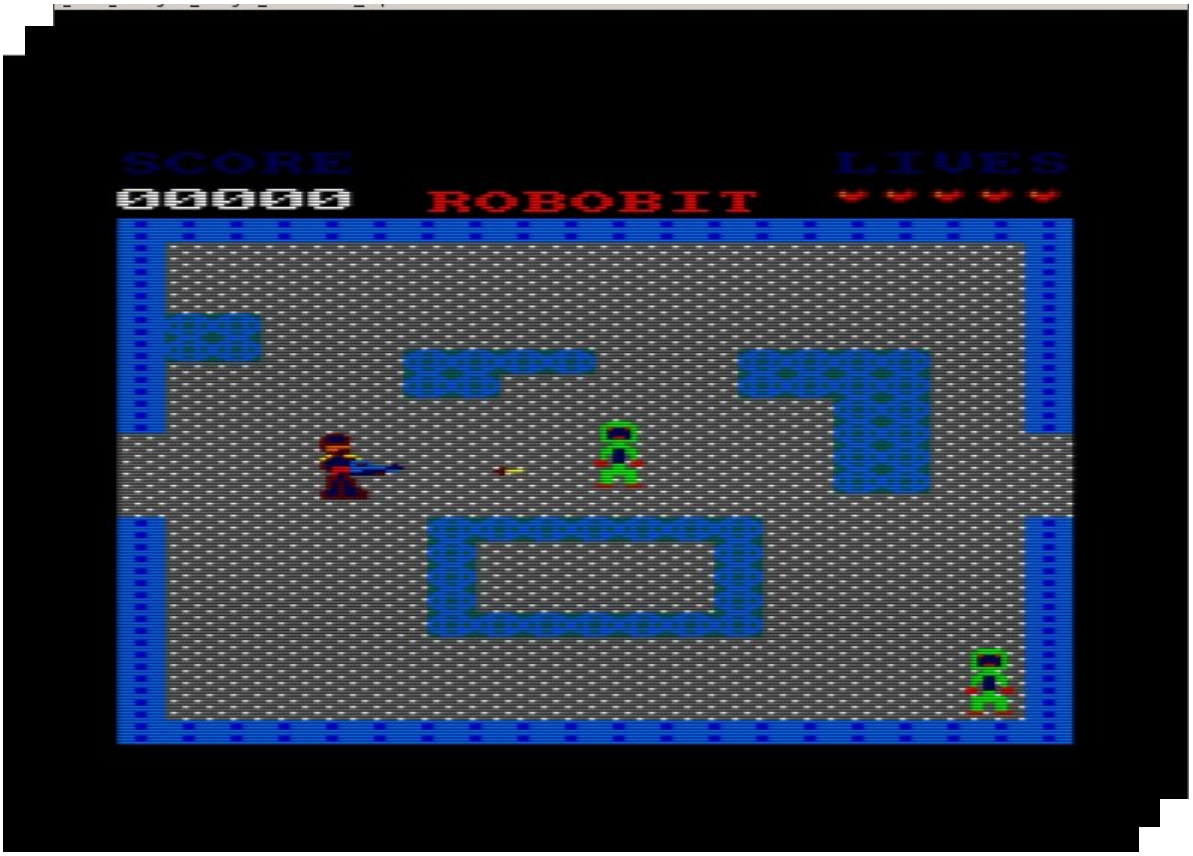
Designing maps…

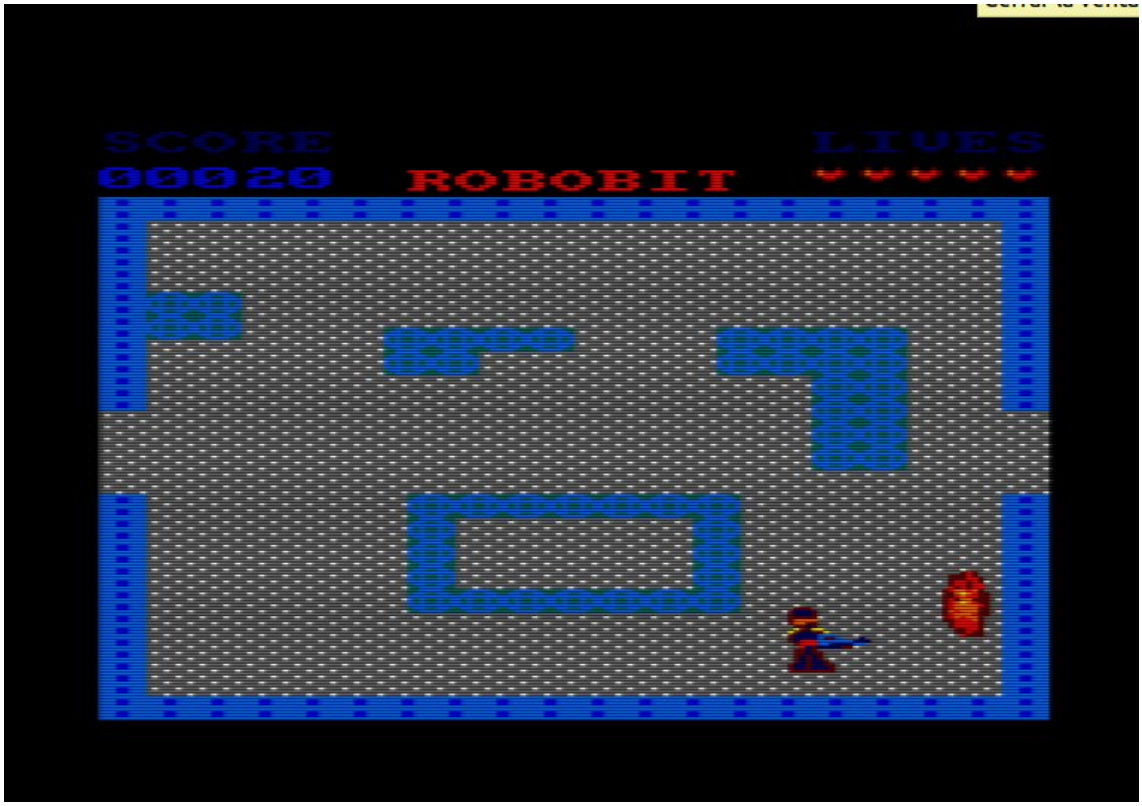


The first map and enemy...



Improving the game...


SCORE
00000
ROBOBIT
LIVES


SCORE
00020
ROBOBIT
LIVES