Little Rogue Project Report

Francisco [fc55555@alunos.fc.ul.pt](mailto:fc55555@alunos.fc.ul.pt)

Vasco Cruz [fc48760@alunos.fc.ul.pt](mailto:fc48760@alunos.fc.ul.pt)

Inteligência Artificial em Jogos

O trabalho final da unidade curricular Inteligência Artificial em Jogos passa por desenvolver componentes de inteligência artificial para um jogo à escolha.

No nosso caso, esse jogo é o Little Rogue, um “top-down role-playing dungeon crawler rogue-like” repleto de items, enimigos e diferentes níveis com mapas gerados procedimentalmente.

O jogo em causa é relativamente simples em termos de mecânicas e componentes de design do mesmo, portanto, para atingir simplicidade de implementação e modularidade de alteração e iteração da componente de inteligência artificial, decidimos implementar a mesma com a utilização de “Behaviour Trees”.

Para a definir, começámos por determinar as ações essenciais para ser possível jogar e avançar neste jogo. Apanhar items, usar tais items, atacar, movimentar, progredir de nível.

Após definir as ações essenciais, definimos condições para executar as mesmas e, através das mesmas, definimos as verificações a fazer antes de executar ações.

Verificar os pontos de vida e a quantidade de poções disponíveis antes de decidir usar uma poção para restaurar pontos de vida à personagem, ou verificar se a personagem sem encontra em cima do objecto “escadas” antes de executar a ação de progredir para o próximo nível do jogo são exemplos de verificações feitas a cada turno para decidir que ação executar.

Ao otimizar tanto as ações como as verificações, chegámos à árvore de decisão seguinte:

Em termos de falhas de comportamento, não encontrámos nada que podesse ser classificado como tal. Contudo, fomos capazes de identificar situações onde o comportamento da personagem poderia ser melhorado e tornado mais complexo e adaptável a variadas situações no jogo.

A identificação da superioridade de um inimigo, fora a sua classificação, em termos de dano causado pelo mesmo poderia ser utilizada para poder decidir sobre a utilização de certos items como pergaminhos para aumentar o dano causado ao inimigo, de modo a terminar a batalha mais depressa para evitar sofrer demasiado dano e preparar confrontos futuros, é um exemplo de aumento de complexidade e adaptabilidade ao jogo, mas não considerámos uma falha comportamental.

Foi-nos possível incluir componentes extra ao projeto, de modo a podermos demonstrar o funcionamento do código implementado. A visualização do funcionamento do algoritmo de pesquisa A\* demonstra como o “pathmaking” foi implementado e como este executa, também como a performance do mesmo:

De modo a melhorar o caminho criado, este processo é aplicado a cada turno, tornando o “pathmaking” mais capaz de se adaptar a alterações no terreno, como por exemplo, a eliminação de um inimigo que se encontrava entre a personagem do jogador e as escadas que dão acesso ao próximo nível. Antes, o algoritmo faria a personagem do jogador contornar este espaço mesmo após eliminar o inimigo, pois, o caminho teria sido criado antes do sucedido e não era capaz de responder ao mesmo. Com a atualização do caminho a cada turno, é possível alterar o mesmo, tornando-o capaz de responder e adaptar-se a tais alterações.