Relatório Projeto CV

Descrição

O presente trabalho foi inspirado em duas versões do GeometryDash em que uma tem uma perspetiva lateral do jogo e do jogador e outra tem uma perspetiva traseira. O jogador ganha pontos à medida que avança pelo nível, perdendo quando colide com um obstáculo, exceto no caso em que o obstáculo é do mesmo material do jogador, onde pode atravessá-lo sem problemas. O jogo possui zonas pré-definidas ao longo do nível que alteram as propriedades do gameplay de alguma forma.

O jogo toma partido ainda da biblioteca mediapipe que torna possível controlar o cubo a partir de gestos da mão direita, um tutorial de como utilizar pode ser visto nos slides da apresentação enviada em anexo.

Existem 4 zonas pré-definidas que ao entrar nelas alteram e dinamizam o gameplay, ilustrando as diferentes abordagens de demonstrar os conceitos chave abordados na cadeira ao longo do semestre.

- Zona 1- Muda a perspetiva da câmara ao jogador.
- Zona 2- Aumenta a velocidade do jogador.
- Zona 3- Diminui a luminosidade geral da cena.
- Zona 4- Inverte os controlos.

A música de fundo também muda na entrada de certas zonas dando uma audible cue do que se passa, por exemplo, na Zona 2 a velocidade da música também aumenta com a velocidade do jogador, supostamente toca em reverse na zona que inverte os controlos, mas não conseguimos perceber se de facto isso estava a acontecer.

Conceitos chave de computação visual

Aqui fazemos uma breve descrição dos elementos implementados no que toca aos conceitos de computação visual, para melhor compreensão, as imagens presentes na apresentação servem como ilustrações dos elementos descritos.

Iluminação

O jogo possui duas luzes, uma pointlight que se posiciona acima do jogador e que o vai acompanhando até ao fim da zona “lights out”, permanecendo fixa após isso para visualizar melhor a sua influência na cena, e uma ambientlight que ilumina o jogo de forma geral. Na Zona 3 brincamos com os valores destas luzes.

Materiais

Tentámos alterar as propriedades dos materiais, nomeadamente dos obstáculos, mas devido a problemas provavelmente relacionados com as texturas ou talvez as propriedades usadas na modelização do Blender, não tiveram efeito nenhum aparente.

Câmara

Adicionamos 2 câmaras, uma à retaguarda do jogador (por defeito) e outra ao seu lado, ambas o acompanham ao longo do nível, as perspetivas são trocadas à medida que o jogador entra e sai de certas zonas.

Texturas

Aplicámos texturas diferentes para os vários modelos, 3 texturas diferentes para o cubo e obstáculos, uma para o chão, e uma textura animada aplicada aos planos que se situam no fundo da cena.

Deteção de Gestos com as mãos

Na apresentação inicial da nossa demo não tínhamos implementado ainda deteção de gestos, sob a sugestão dos professores adicionamos nesta iteração esta componente, um exemplo de como funciona pode ser visto no vídeo associado ao trabalho. É de notar que esta modalidade deteriora bastante a performance do jogo, e por vezes a deteção não é bem eficiente, mas isto foi apenas um add-on, o jogo foi desenvolvido com o intuito de ser jogado com os botões do teclado.

Funcionamento geral

Tudo começa na importação dos vários modelos, o jogador e o chão são postos numa posição fixa e os obstáculos são gerados logo ao início com offsets fixos quanto ao espaço entre si, mas com posicionamento aleatório quanto à altura e disposição lateral, simplesmente fixamos um seed para garantir que conseguimos gerar o mesmo nível novamente. Ao começar o jogo, várias tasks correm para garantir o funcionamento de várias componentes básicas de jogabilidade, como o update do score, o movimento do cubo e das câmaras, deteção de colisões, tratamento dos inputs, entre outras.

Em intervalos de tempo fixos, o aspeto tanto dos obstáculos como do cubo do jogador são alterados, ao longo do nível é sempre calculado e guardado o obstáculo mais próximo do jogador (numa task), caso se verifique que o modelo do cubo se situa dentro do modelo desse tal obstáculo, é verificada a textura de ambos e caso não corresponda o jogador perde.

A pontuação é calculada conforme o tempo passado, visto que a velocidade do jogador é fixa, dependendo apenas da zona em que se encontra. Em várias replays, passado x tempo, o jogador encontra-se sempre na mesma localização.

Pontos fortes/fracos

Achamos que o jogo está interessante, em termos de jogabilidade, tendo em conta as várias dificuldades que encontrámos devido à biblioteca Panda3D. As várias zonas adicionam um nível substancial de dificuldade, e não é completamente trivial chegar ao fim do nível, mas também não é impossível, o que apenas exige uma certa aprendizagem por parte do jogador.

Achamos que a sua apresentação podia estar bastante melhor, como já foi referido, dificuldades relacionadas com o Panda3D e falta de experiência com o Blender fizeram com que perdêssemos bastante tempo a tentar resolver coisas longe do foco do trabalho.

Instruções

Correr o jogo é bastante simples, basta executar o ficheiro main.py com 0 caso pretenda jogar com os controlos do teclado, e 1 caso queira jogar usando gestos como método de input:

- python main.py 0
- python main.py 1

Trabalho futuro

Se tivéssemos mais tempo disponível para desenvolver melhor o jogo, gostaríamos de adicionar mais zonas diferentes e de tornar possível uma distribuição aleatória das mesmas (não ter zonas/ordem de poderes predefinida). Gostávamos também de melhorar vários aspetos na geração do nível, nomeadamente de permitir ao jogador ter acesso a níveis à medida que os vai completando, tendo em conta que cada nível vai progressivamente gerando obstáculos mais próximos (e adicionalmente a velocidade do jogador também é aumentada para aumentar a dificuldade).

No que toca à apresentação do jogo, gostávamos de tentar novamente avançar com a ideia original de usar vários materiais (ao invés de várias texturas com cores diferentes) no cubo e obstáculos, para haver uma melhor diferenciação entre eles e uma melhor incorporação dos conceitos de CV abordados.