

# Animação e Visualização Tridimensional 2022

## Grupo P07

ist196894 Marta Vicente  
ist193694 Carolina Ramos  
ist192457 Duarte Boto

O projeto de AVT de 2022 constituía numa simulação da missão do Rover em Marte. Através de diversas técnicas ensinadas durante as aulas, modelámos o rover e terreno envolvente de modo a ser interativo entre objetos e o utilizador.

Durante este relatório serão explicadas todas as técnicas utilizadas com os respectivos exemplos.

### Ambiente

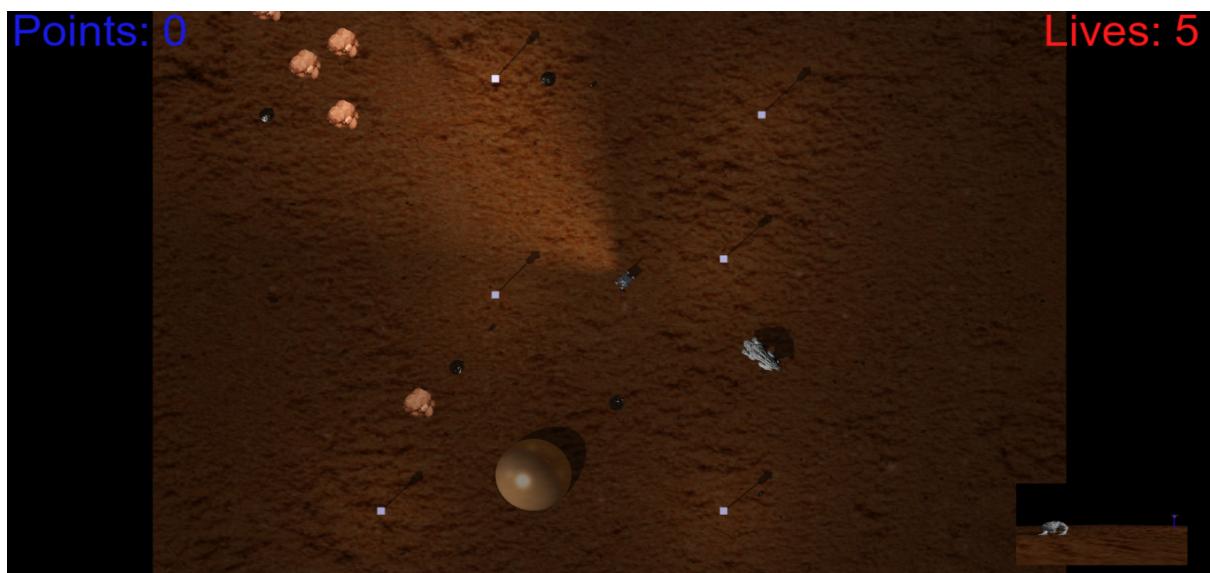


Fig 1. Visão total do ambiente

Começámos por criar o ambiente de Marte, utilizando objetos geométricos tridimensionais para vários propósitos:

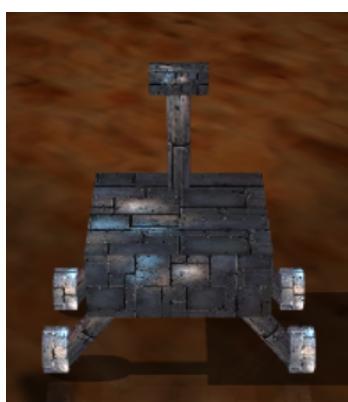


Fig 2. Rover

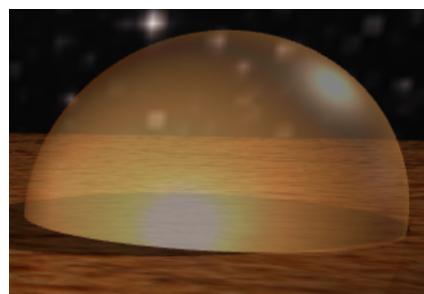


Fig 3. Cúpula

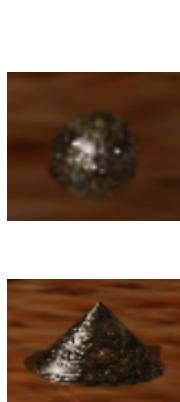


Fig 4. Rochas

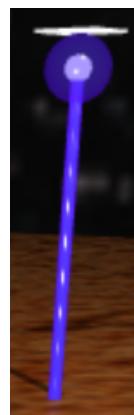


Fig 5.Candeeiro

Para a construção do Rover (Fig 2) foram utilizados vários cubos e cilindros manipulados para formar o aspetto mais próximo do robot. A cúpula (Fig 3) foi construída com uma semi-esfera. Para diversificar e distinguir os diferentes tipos de rochas (Fig 4) criámos esferas para as que se movimentam constantemente e para as rochas estáticas cones. Por fim, para arranjar uma solução para as spotlights que vamos referir posteriormente, desenvolvemos candeeiros (Fig 5) com um cilindro como base, duas esferas (com a interior a simular a lâmpada) e por fim um paralelepípedo fino foi usado para imitar um painel solar.

## Câmera

Foram definidas quatro câmeras:

- câmera 1 (Fig 6): câmera fixa (vista orbital) com projeção ortogonal;
- câmera 2 (Fig 7): câmera fixa (vista orbital) com projeção perspetiva;
- câmera 3 (Fig 8): câmera móvel (segue o rover) com projeção perspetiva;
- câmera 4 (Fig 9): câmera móvel (segue o rover, apontando para trás) com projeção perspetiva;

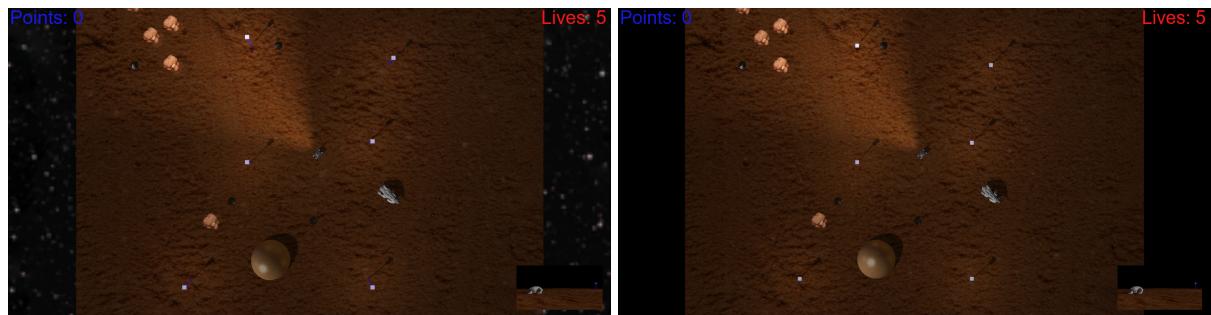


Fig 6. Câmera 1

Fig 7. Câmera 2



Fig 8. Câmera 3

Fig 9. Câmera 4 (com contorno vermelho)

É possível o utilizador alternar entre as 3 primeiras câmeras através das teclas 1, 2 e 3 respetivamente. Já a quarta câmera (Fig 9) é sempre desenhada no ecrã, no canto inferior direito, de modo ao utilizador poder ter conhecimento do que o envolve a qualquer altura. É ainda possível rodar a câmera 3 (Fig 8) com o rato, de modo ao utilizador poder olhar em qualquer direção, sem necessidade de rodar o próprio rover.

Para a câmera 3 e 4, obtemos a posição atual do rover (*pos*), através da função *Rover::getFollowPos* e a mesma posição (*target*), mas com um y mais elevado (através da função *Rover::getFollowTarget*). Assim, a câmera 3 (Fig 8) posiciona-se em *pos* e aponta para *target* e a câmera 4 (Fig 9) posiciona-se em *target* e aponta para *pos*.

## Movimento do Rover

É possível mover o rover com as teclas Q (frente) e A (trás) e rodar o mesmo com as teclas O (esquerda) e P (direita). O movimento é uniformemente acelerado. Assim, o rover terá velocidade crescente quando inicia o movimento e velocidade decrescente quando a tecla Q/A deixa de ser premida. Para realizar este efeito, utilizamos variáveis (`_moveState` e `_rotateState`) que indicam o movimento desejado para o rover, uma variável `deltaTime` que indica quanto tempo passou desde a última vez que a posição do rover foi atualizada e constantes `acceleration`, `slowDown` e `speedLimit`. Todas estas variáveis e constantes foram depois aplicadas às leis do movimento de Newton. Para as rotações, foram aplicados princípios trigonométricos.

## Movimento das Rochas

O movimento das rochas é aleatório, elas começam no limite de um círculo de 40 de raio, têm uma direção aleatória para o interior do círculo. No início o programa começa com 2 rochas, mas sempre que a rocha chega ao fim do círculo desaparece, criando uma nova rocha com características de movimento novas. A velocidade da rocha vai aumentando com o tempo do programa, aumentando assim também a dificuldade do programa.

## Luzes e Nevoeiro

Foram aplicadas vários tipos de luzes e técnicas associadas durante o projeto. Temos uma luz global que utiliza uma fonte de luz direcional que simula dia e noite (Fig 10 e 11), para ligar ou desligar usa se a tecla N. No sítio de cada candeeiro, que são no total 6, temos point lights que iluminam melhor certas zonas do chão para facilitar a movimentação do Rover, podem ser ativados ou desativados com a tecla C. Por fim, o rover tem 2 spot lights à frente para simular faróis que apontam para o chão (Fig 12), com a tecla H é possível ligar ou desligar.

O jogo começa com a função de nevoeiro ligada (Fig 13), que pode ser desativado ou ativado com a tecla F. Tanto as luzes como o nevoeiro são calculados no pointlight.frag.



Fig 10. Luz direcional desligada com spotlight ligado

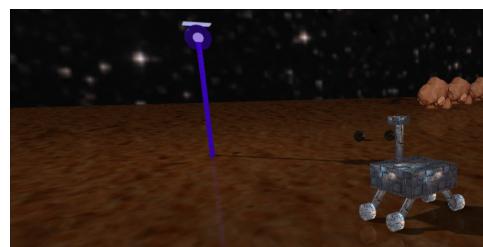


Fig 11. Luz direcional ligada com spotlight ligado



Fig 12. Faróis do Rover desligado e ligado

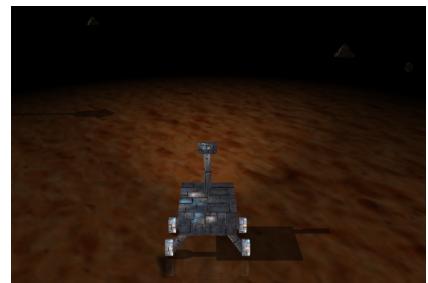


Fig 13. Nevoeiro ligado

## Colisões

Para as colisões, foi criada uma classe *Box*, que representa os oito vértices de um cubo envolvendo os objetos, em coordenadas locais. Foi também criada a classe *MinMaxBox*, que armazena dois pontos (*min* e *max*) em coordenadas do mundo. Para todos os objetos (com a exceção do rover) estes atributos são estáticos, visto as suas posições ou ângulos não se alterarem. Já o rover tem de atualizar a sua *MinMaxBox* cada vez que se move (função *Rover::updateMinMax*). Assim, as coordenadas da *Box* em coordenadas locais são transformadas em coordenadas do mundo, determina-se quais os dois vértices da *Box* que correspondem aos extremos *min* e *max*. Com os extremos do rover e de todos os outros objetos em coordenadas do mundo, já é possível comparar a sua localização e determinar se existem colisões.

## Partículas

Introduzimos as partículas em interação com a colisão, isto é, se o rover colidir com umas das rochas que se encontram em movimento, esta torna-se numa nuvem de partículas (Fig 14). Para desenvolver as partículas foi utilizado o exemplo disponibilizado na aula, no entanto alterámos a variável que simula a aceleração da gravidade para positiva para as partículas ficarem mais realistas com a situação e afastando-se do modelo de fogo de artifício que era o objetivo do exemplo dado.

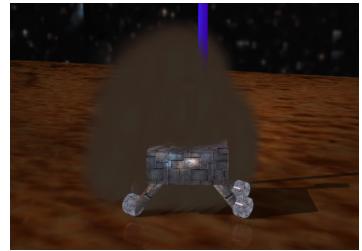


Fig 14. Partículas

## Texturas e Transparência

Para tornar o ambiente mais realistas utilizamos várias texturas em diferentes objectos.

- Rover (Fig 15): Textura metalizada;
- Rochas (Fig 16): Textura de uma rocha;
- Chão (Fig 17): Multi-Texturing com duas texturas de areias;
- Cúpula e candeeiros (Fig 18): Transparência;
- Partículas (Fig 19): Textura simples dada pelo professor.



Fig 15. Rover



Fig 16. Rochas



Fig 17. Multi-texturing



Fig 18. Transparência: Cúpula e Parte Superior do candeeiro

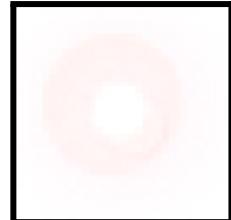
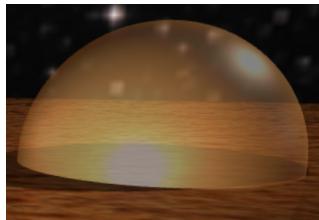


Fig 19. Partícula

O cálculo das diferentes texturas foi efectuado no pointlight.frag. Para as transparências desenvolvemos como representado na aula manipulando o alfa.

## HUD (Pausa, gameOver e Reset)

Durante o jogo é possível fazer pausa ou dar reset. Temos um sistema de pontos e vidas, estas variáveis aparecem na parte superior da janela. Quando existe colisão com uma das rochas em movimento perde-se uma vida e para ganhar pontos é sobreviver o máximo tempo, a cada 30 segundos recebemos um ponto.

Se utilizar a tecla S o jogo pára e aparece uma mensagem “Game Paused”(Fig 20). O Rover não se consegue movimentar e quaisquer objectos que se encontrassem em movimento param. Para retornar é só voltar a clicar na tecla S.

Se durante o jogo as 5 vidas, com que começaram no início, chegarem ao fim, o jogo acaba e aparece uma mensagem “Game Over”(Fig 21). A única maneira de reiniciar o jogo é usando a tecla R que dá reset total do jogo, esta função pode também ser usada durante o jogo para começar do início.



Fig 20. Menu de pausa



Fig 21. Menu de fim de jogo

## Assimp

Para os objetos assimp foi aplicado o código disponibilizado nas aulas laboratoriais a um objeto que representa um alien (Fig 22) como demonstração.



Fig 22. Objeto assimp

## Billboard

O billboard foi desenvolvido baseado no exemplo dado pelo professor. Temos 5 billboard que utilizam a textura de uma rocha (Fig 23). Os Billboard seguem a câmera móvel para estarem sempre visíveis. Para conseguir este efeito o quad é sempre alinhado com a normal da câmara.



Fig 23. Billboards

## Câmera traseira

A câmera traseira foi definida como explicado na seção **Câmera**. Para apresentar a cena traseira no canto do ecrã ao mesmo tempo que outra câmera apresenta a sua cena no resto do ecrã, recorreu-se à manipulação de viewports e ao stencil test. Definimos um quad no qual será desenhada a cena traseira, enquanto que a cena alternativa é desenhada fora deste quad, aplicando assim o stencil test. O viewport com a cena traseira é definido com um 1/7 do tamanho do viewport original e posicionado no local do quad anteriormente referido.

## Lens Flare

Foi definido um lens flare para representar a fonte de luz direcional, simulando a reflexão dos componentes da câmara ao olhar diretamente para uma luz forte. Os componentes do lens flare ficam maiores e mais opacos quando a luz está perto do centro da câmara e mais pequenos e transparentes quando está longe.

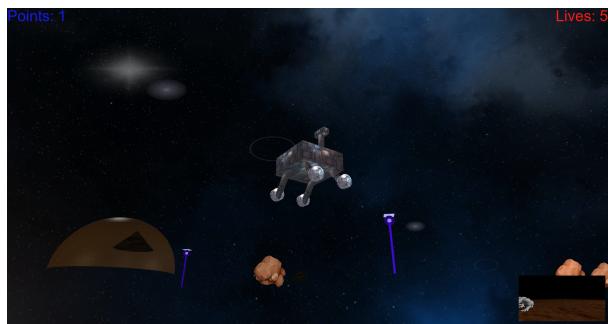


Fig 24. Lens Flare longe



Fig 25. Lens Flare perto

## Planar Shadows and Planar Reflections

As Planar Shadows e as Planar reflections foram ambas aplicadas no plano do chão para representar as sombras e reflexos dos objetos da cena. As sombras são aplicadas através da matriz de projeção de sombras, ligeiramente acima do plano do chão. Os reflexos são desenhados invertidos do outro lado do plano do chão, sendo o chão ligeiramente transparente para ser visível. Ambos usam o stencil test para desenhar apenas no chão, sendo que as sombras também o usam para apenas escurecer o chão uma vez por fragmento, tornando as sombras mais uniformes.



Fig 26. Shadows + reflections rover e assimp



Fig 27. Shadows + reflections stencil

## Bump-mapping

O bump-mapping foi aplicado às rochas, tanto estáticas (Fig 28) como móveis (Fig 29).

Para este efeito, foi utilizado o ficheiro *bump.jpg*. No vertex shader, as coordenadas são convertidas para o espaço tangente. Já no fragment shader, as normais são retiradas do já referido *bump.jpg* e é aplicada a textura do ficheiro *rocks.tga*.



Fig 28. Rocha estática

Fig 29. Rocha móvel

## Skybox

Para a skybox foi utilizada uma textura de nébula espacial (Fig 26). Como ensinado na aula, foi criado um grande cubo com centro na origem, no qual foram aplicadas 6 texturas.

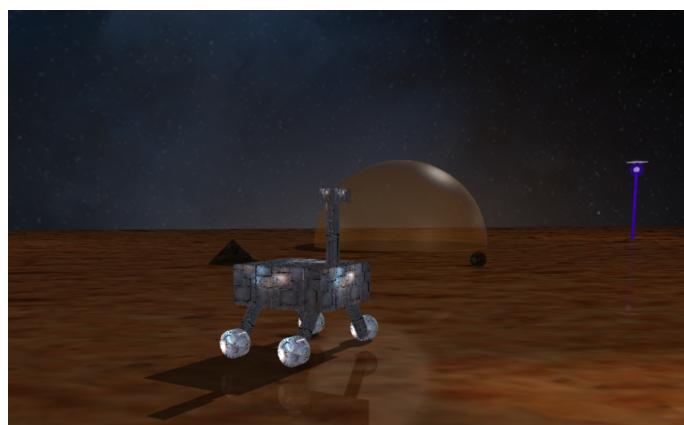


Fig 30. Skybox

## Environment mapping

Usámos depois a skybox como um environment map, usando o vetor da reflexão para a textura cúbica. Criámos uma esfera que reflete a nébula espacial e fica assim com uma textura cósmica.

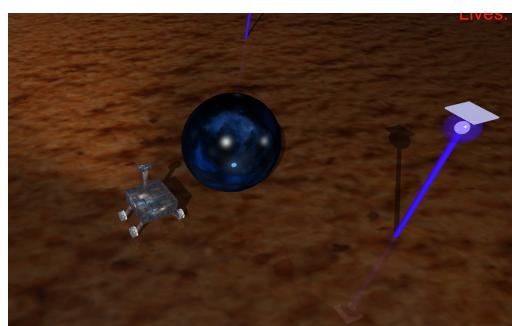


Fig 31. Esfera de cima

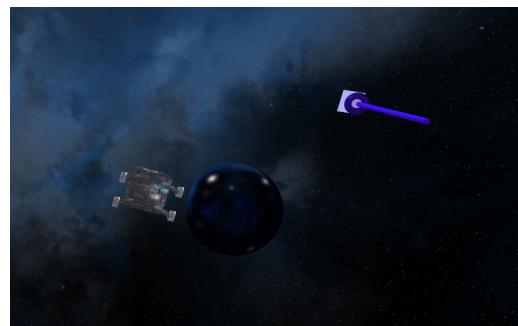


Fig 32. Esfera de baixo