



MATEMÁTICA NA ANIMAÇÃO GRÁFICA E NA INTERAÇÃO

A incorporação de elementos visuais, estáticos ou dinâmicos, que sugerem ao espetador de um filme ou a um videojogador estar na presença (virtual) de elementos a 3D é uma realidade omnipresente. Igualmente omnipresente neste contexto está um conjunto de modelos matemáticos. Não obstante estes modelos poderem ajudar a representar entidades de uma certa complexidade, acabam por ser baseados em conceitos matemáticos relativamente simples, que iremos apresentar neste texto.

1. INTRODUÇÃO

Os computadores têm vindo a revelar-se potentes ferramentas para a produção de gráficos de forma rápida e económica. Não há nenhuma área que não tenha tirado partido da visualização gráfica computacional, não sendo portanto de espantar que a computação gráfica tenha tido umas tão vastas disseminação e aceitação. Embora no seu início tenha sido necessário recorrer a equipamento caro e pouco prático, os avanços tecnológicos forneceram à computação gráfica meios práticos e bastante económicos, sendo possível encontrá-la em áreas como a medicina, engenharia, ciência, indústria, entretenimento, arte, educação e formação, entre outras.

Em muitas destas aplicações, os objetos são inicialmente representados através de estruturas em linhas, conhecidas por *wireframes*, que desvendam a forma genérica e as características internas do objeto em causa. Uma das virtudes desta abordagem consiste em permitir ao desenhador aperceber-se rapidamente dos efeitos dos ajustamentos iterativos efetuados aos objetos. O *software* típico de CAD é composto por um ambiente multijanela, em que as várias janelas apresentadas permitem ter visões de partes e em escalas diferentes de um mesmo objeto.

Um outro contexto em que é suficiente o recurso a estruturas gráficas muito simples é o dos simuladores de

túnel de vento, uma aplicação muito comum na indústria automóvel e na indústria aeroespacial.

Quando a fase de desenho do objeto se aproxima do seu final, são aplicados modelos de iluminação e de renderização de superfícies às estruturas anteriormente criadas, sendo assim mostrada a aparência do produto final.

Os métodos de computação gráfica são bastante usados na criação artística. Há várias formas de o fazer, mas uma das mais interessantes resulta do uso de uma combinação de programas de CAD e de construção e manipulação de imagens, que conjuntamente permitem desenhar objetos e o ambiente que os rodeia, recorrendo a todo o tipo de efeitos.

Um outro exemplo muito interessante consiste no recurso à visualização de relações matemáticas. Na figura 1, apresentamos uma criação artística que parte da fórmula $x^n + y^n = z^n$, com $n = 5$, um caso particular que deriva do Último Teorema de Fermat.

Os métodos da computação gráfica têm vindo a ser muito bem-sucedidos no campo do entretenimento, como em cinema, vídeos musicais e televisão. Dois bons exemplos combinam imagem real com produções gráficas virtuais: o filme *TRON: Legacy*¹ e a saga *Matrix*².

Modelos gerados em computador de sistemas físicos, económicos e financeiros têm sido usados como comple-

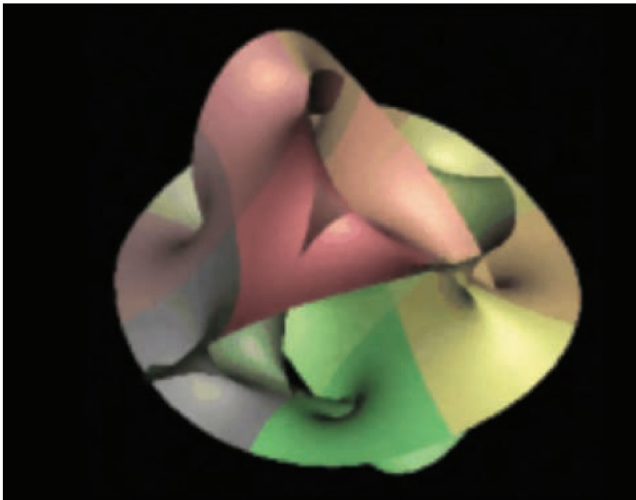


Figura 1. Criação a partir da superfície em \mathbb{R}^3 dada por $x^n + y^n = z^n$, com $n = 5$, decorrente do Último Teorema de Fermat.

mentos à aprendizagem nestas áreas. Uma outra área onde a computação gráfica tem vindo a ser protagonista são os chamados simuladores, sistemas que permitem simular, por exemplo aviões, comboios ou automóveis, usados para formação de pilotos ou condutores (ver figura 2).

Cientistas, engenheiros e médicos, entre outros, são muitas vezes confrontados com a necessidade de analisar grandes quantidades de informação ou de estudar o comportamento de determinados processos. A visualização dessa informação pode ajudar a perceber quais as tendências e características iminentes a estes fenómenos.

Embora haja intersecções entre a computação gráfica e o processamento de imagem, as duas áreas atuam sobre domínios distintos. Enquanto na computação gráfica o computador é usado para gerar a imagem, em processamento de imagem são usadas técnicas matemáticas e computacionais para analisar imagens previamente existentes, tais como fotografias ou imagens de televisão. No primeiro caso, temos as aplicações à tomografia computadorizada, em que a partir da informação numérica se reconstruem secções tridimensionais da parte do corpo a ser observada. O interesse no segundo caso é sobretudo suscitado pela robótica.

Neste trabalho começamos por mostrar como as transformações geométricas mais simples podem ser formuladas matematicamente, revelando o papel preponderante que as matrizes e as suas operações têm neste contexto. Seguidamente apresentamos resumidamente alguns conceitos matemáticos necessários à descrição de trajetórias no espaço, desde contextos mais simples, como a aplica-



Figura 2. Exemplos de simuladores de aviões, de automóveis e de comboios.

ção na representação de órbitas de planetas de um sistema solar, aos mais complexos, como a representação da trajetória de um automóvel de competição numa pista. Terminaremos discutindo brevemente a questão das deformações de modelos tridimensionais, mostrando que os valores próprios – outra noção vinda da álgebra linear – são uma peça fundamental neste domínio.

¹ <https://www.imdb.com/title/tt1104001>

² <https://www.imdb.com/title/tt0133093>

2. TRANSFORMAÇÕES GEOMÉTRICAS

As transformações geométricas são usadas em computação gráfica para cumprir dois objetivos:

1. Modelar e construir cenários;
2. Navegar no interior de um espaço bi ou tridimensional.

Por exemplo, se quisermos representar um edifício com n janelas, podemos proceder segundo os passos seguintes:

1. Construir uma janela recorrendo a funções e variáveis gráficas básicas;
2. Replicar a janela n vezes;
3. Colocar cada janela na localização pretendida recorrendo a translações e rotações.

Assim se conclui que as transformações como as translações e as rotações podem ser consideradas como operações de modelação de cenários. Estas operações podem igualmente ser usadas para mover um *bot* ou um *avatar* num ambiente virtual como um jogo FPS (*First-Person Shooter*).

Podemos considerar a translação como o movimento de um ponto ou de um objeto que o desloca de uma certa distância ao longo de uma certa direção.

Por exemplo, se o ponto $A = (x, y)$ é deslocado Δx unidades ao longo do eixo dos XX' e Δy unidades ao longo do eixo dos YY' , transforma-se no ponto

$$A' = (x + \Delta x, y' + \Delta y)$$

cujas coordenadas são dadas por

$$\begin{cases} x' = x + \Delta x \\ y' = y + \Delta y \end{cases}$$

Na figura 3 apresentamos uma representação esquemática desta transformação.

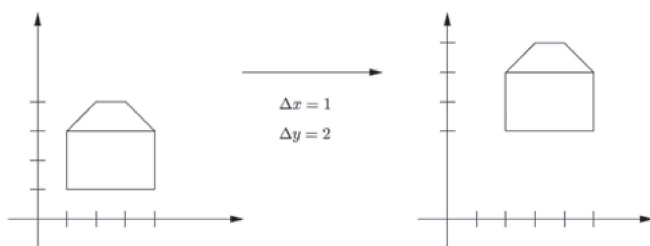


Figura 3. Representação de uma translação 2D.

Se representarmos matricialmente quer os pontos quer a translação propriamente dita, obtemos

$$A = \begin{bmatrix} x \\ y \end{bmatrix}, A' = \begin{bmatrix} x' \\ y' \end{bmatrix}, T = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

e portanto a translação pode ser expressa como

$$A' = A + T.$$

De uma forma genérica, aplicar uma translação a um objeto poligonal corresponde a aplicar a mesma translação aos seus vértices (por exemplo, os cantos ou os pontos extremos) de tal forma que as linhas ou os polígonos que constituem o objeto podem ser desenhados sobre os vértices transformados.

No que diz respeito às rotações em torno da origem, rodar um objeto segundo um ângulo θ corresponde a rodá-lo em torno da origem segundo esse mesmo ângulo θ . Usando coordenadas polares (r, φ) , as coordenadas cartesianas de um ponto no plano são dadas pelas seguintes equações:

$$\begin{cases} x = r \cos \varphi \\ y = r \sin \varphi \end{cases}$$

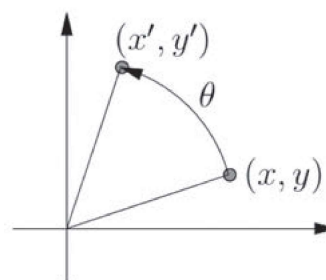


Figura 4. Representação de uma rotação 2D.

Após rodar o ponto de um ângulo θ em torno da origem, obtemos o seguinte ponto transformado:

$$\begin{cases} x' = r \cos(\varphi + \theta) \\ y' = r \sin(\varphi + \theta) \end{cases}$$

ou

$$\begin{cases} x' = r \cos \varphi \cos \theta - r \sin \varphi \sin \theta \\ y' = r \cos \varphi \sin \theta + r \sin \varphi \cos \theta \end{cases}$$

ou seja

$$\begin{cases} x' = x \cos \theta - y \sin \theta \\ y' = x \sin \theta + y \cos \theta \end{cases}$$

e em notação matricial

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Se

$$R(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

denotar a matriz de rotação 2×2 , então podemos escrever

$$A' = R(\theta)A.$$

Em resumo, a translação exprime-se através da adição de matrizes, enquanto a rotação em torno da origem recorre ao produto de matrizes. Isto quer dizer que podemos combinar um número arbitrário de translações recorrendo à soma de matrizes, bem como um número arbitrário de rotações recorrendo ao produto. Contudo, não é possível calcular da mesma forma combinações de translações e rotações. Nesse sentido, seria desejável recorrer a uma única operação, por exemplo, o produto, para representar quer as translações quer as rotações.

As coordenadas homogêneas são uma das formas possíveis de resolver este problema. Com este tipo de coordenadas será possível combinar um número arbitrário destas operações recorrendo ao produto de matrizes. Deste modo, translações e rotações definidas em coordenadas homogêneas são dadas por

Translação: $A' = TA$;

Rotação: $A' = RA$.

Em coordenadas homogêneas um ponto $P = (x, y)$ é representado pelo ponto homogêneo $P = (X, Y, W)$, $W \neq 0$, com

$$X = \frac{x}{W} \quad \text{e} \quad Y = \frac{y}{W}.$$

Em computação gráfica escolhe-se habitualmente $W = 1$ por questões de simplicidade. Usando coordenadas homogêneas, as transformações euclidianas são expressas recorrendo a matrizes 3×3 da forma seguinte:

Translação:

$$T(\Delta x, \Delta y) = \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix};$$

Rotação:

$$R(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

A matriz de rotação que apresentámos é eficaz se pretendermos rodar um ponto em torno da origem. Vejamos agora qual o procedimento a seguir se pretendermos rodar um ponto em torno de um ponto arbitrário (x_0, y_0) :

1. Transladar o ponto (x_0, y_0) até à origem, ou seja, aplicar $T(-x_0, -y_0)$;
2. Aplicar a rotação $R(\theta)$;
3. Transladar de novo o ponto para a localização original, ou seja, aplicar $T(x_0, y_0)$.

De notar, por outro lado, que a ordem das transformações geométricas representadas por estas matrizes é relevante, uma vez que o produto de matrizes, sendo associativo, ou seja, $ABC = (AB)C = A(BC)$ não é em geral comutativo, ou seja, nem sempre $AB = BA$. Usando este tipo de matrizes, a rotação em torno de um ponto arbitrário é representada pela seguinte matriz C :

$$C = T(x_0, y_0)R(\theta)T(-x_0, -y_0).$$

Multiplicando por um ponto P , obtemos

$$CP = T(x_0, y_0)R(\theta)T(-x_0, -y_0)P.$$

As transformações euclidianas são as que preservam a distância entre os pontos, sendo por isso também conhecidas por transformações rígidas. As transformações afins, que agora iremos abordar, caracterizam-se, por seu turno, por preservar o paralelismo. Isto quer dizer que duas linhas paralelas se mantêm paralelas após aplicação de uma transformação afim. Como consequência deste invariante, outras propriedades são preservadas. Por exemplo, uma transformação afim preserva a colinearidade (todos os pontos sobre uma linha mantêm-se nessa linha após a transformação), bem como as razões entre distâncias (o ponto médio de um segmento mantêm-se como ponto médio após a transformação).

Há muitos exemplos de transformações afins. Podemos, contudo, afirmar que são todas elas composições de quatro transformações afins básicas: rotação, translação, variação de tamanho e cisalhamento. Analisemos agora estas duas últimas.

A variação de tamanho consiste na alteração da dimensão do objeto em causa. Escalar um ponto (x, y) segundo um fator λ_x segundo o eixo dos XX' e λ_y segundo o eixo dos YY' corresponde a multiplicar as suas coordenadas pelo respetivo fator de escalonamento:

$$\begin{cases} x' = \lambda_x x \\ y' = \lambda_y y \end{cases}$$

ou, usando notação matricial sobre coordenadas homogêneas,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \lambda_x & 0 & 0 \\ 0 & \lambda_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

O cisalhamento (*shearing*) é a transformação que goza da propriedade segundo a qual todos os pontos ao longo de uma dada linha I se mantêm fixos, enquanto que os restantes são deslocados ao longo de uma direção paralela a I numa distância que é proporcional à sua distância perpendicular a I . De notar que o cisalhamento sobre um objeto não altera a sua área. O cisalhamento é generalizável a três dimensões, considerando-se planos em vez de linhas.

Cisalhar um ponto (x, y) por um fator h_x ao longo do eixo dos XX' e por um fator h_y ao longo do eixo dos YY' corresponde a efetuar as seguintes operações:

$$\begin{cases} x' = x + h_x y \\ y' = y + h_y x \end{cases}$$

ou, usando notação matricial e coordenadas homogêneas,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & h_x & 0 \\ h_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

3. REPRESENTAÇÃO DE CURVAS

Representar curvas no plano ou no espaço reveste-se de grande importância neste contexto, por dois motivos: por um lado pela representação do objeto em si, mas também porque poderá corresponder à construção de uma trajetória de uma entidade numa cena a duas ou três dimensões. As parametrizações de curvas têm sido por este motivo bastante usadas no âmbito de sistemas com realidade virtual e aumentada, correspondendo, por exemplo, à trajetória ao longo das órbitas de planetas em torno da estrela do seu sistema planetário.

Assim, dá-se o nome de curva no espaço ao conjunto geométrico γ de pontos (x, y, z) de \mathbb{R}^3 que são traçados no espaço à medida que um parâmetro real t (que habitualmente corresponde ao tempo) percorre um intervalo real $[a, b]$. Denomina-se parametrização de uma curva γ toda a função $s: [a, b] \rightarrow \mathbb{R}^3$ cujo contradomínio é o lugar geométrico da curva γ .

Iremos considerar algumas premissas no que à cons-

trução de parametrizações diz respeito. Deste modo, o traçado da curva tem de ser contínuo, ou seja, a função s tem de ser contínua no seu domínio $[a, b]$. A própria velocidade a que o traçado é levado a cabo terá também de ser contínua, ou seja, a função s terá de ser continuamente diferenciável em $[a, b]$.

Dentro do universo das curvas no espaço, estaremos interessados nas denominadas curvas regulares. Uma curva γ diz-se regular se admitir uma parametrização $s(t)$ com $t \in [a, b]$ tal que

- ▶ $s(t)$ percorre continuamente e a uma velocidade contínua a curva γ ;
- ▶ não há paragens no traçado do gráfico definido por $s(t)$;
- ▶ nenhum ponto do espaço é percorrido mais do que uma vez.

Se, além disso, $s(a) = s(b)$, ou seja, se no final regressarmos ao ponto de partida, então a curva diz-se fechada.

Vejamos agora alguns casos particulares de parametrizações, começando pelo caso mais simples de um segmento de reta, que iremos considerar, sem perda de generalidade, contido no plano XOY , e que está representado na figura 5.

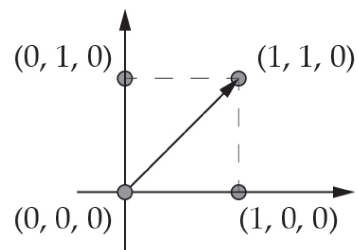


Figura 5. Um segmento de reta no plano XOY .

Aquela que provavelmente será a forma mais simples de parametrizar o segmento de reta da figura 5, admitindo um movimento a velocidade constante do ponto $P_0 = (0, 0, 0)$ ao ponto $P_1 = (1, 1, 0)$ será a seguinte:

$$s(t) = (1 - t)\vec{OP}_0 + t\vec{OP}_1, \quad t \in [0, 1].$$

Consoante o tipo de movimento pretendido, nomeadamente no que toca à velocidade a que o segmento de reta é percorrido, podemos considerar parametrizações alternativas, como a seguinte:

$$s_2(t) = (2t, 2t, 0), t \in \left[0, \frac{1}{2}\right],$$

em que o segmento de reta é percorrido no mesmo sentido, mas ao dobro da velocidade, ou então

$$s_3(t) = (1 - 2t, 1 - 2t, 0), t \in \left[0, \frac{1}{2}\right].$$

em que o segmento de reta é agora percorrido de P_1 para P_0 , igualmente ao dobro da velocidade.

Uma outra trajetória muito comum em computação gráfica é a da circunferência. Uma parametrização para uma circunferência de raio 1, centrada na origem $(0,0,0)$, percorrida uma vez no sentido anti-horário com início no ponto $(1,0,0)$ e contida no plano XOY , tal como a representada na figura 6, poderá ser dada por

$$s(t) = (\cos t, \sin t, 0), t \in [0, 2\pi].$$

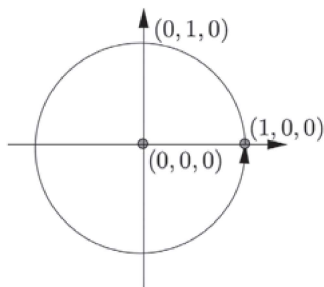


Figura 6. Uma circunferência no plano XOY .

Como caso particular, podemos estar interessados em construir uma trajetória correspondente a apenas um arco da circunferência, percorrido num dado sentido, tal como mostrado na figura 7, onde, sem perda de generalidade, é representado um arco de circunferência percorrido no sentido horário.

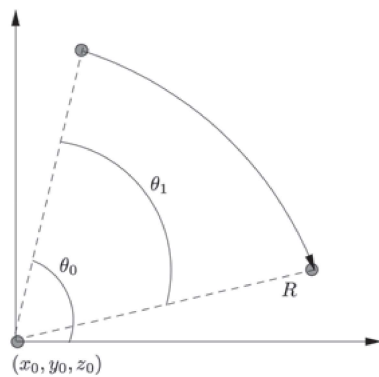


Figura 7. Um arco de circunferência no plano XOY .

De uma forma genérica, uma parametrização de um arco de circunferência com centro em (x_0, y_0, z_0) , raio $R > 0$, com amplitude $\theta_1 \in [0, 2\pi]$ e cujo ângulo inicial com o eixo dos XX' é $\theta_0 \in [0, 2\pi]$, pode ser dada por

$$s(t) = (x_0, y_0, z_0) + R(\cos(\theta_0 \pm t), \sin(\theta_0 \pm t), 0),$$

com $t \in [0, \theta_1]$, em que o sinal “+” é escolhido para o sentido anti-horário e o sinal “-” para o sentido horário. Estamos a admitir, sem perda de generalidade, que o arco de circunferência está mergulhado num plano paralelo ao plano XOY e que contém o ponto $(0,0,z_0)$.

Finalizamos esta secção com um caso particularmente interessante para a construção de uma trajetória de objetos planetários em sistemas solares, que é tipicamente uma elipse, que iremos admitir percorrida no sentido anti-horário. Este tipo de parametrização tem sido usada para construção de órbitas virtuais de planetas no âmbito de sistemas interativos com realidade aumentada, como o projeto PlanetarySystemGO, apresentado em [4].

Consideremos uma elipse centrada em (x_0, y_0) e eixos paralelos aos eixos coordenados, com focos nos pontos F_1 e F_2 , como a representada na figura 8.

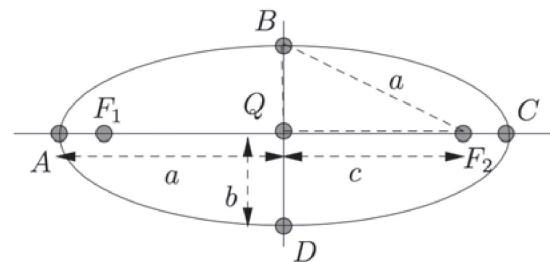


Figura 8. Uma elipse no plano XOY .

Se pretendermos escrever uma parametrização desta elipse, mas definida em relação ao foco $F_1 = (x_1, y_1)$, correspondente à localização da estrela em torno da qual orbitam os planetas do sistema solar, é possível demonstrar que se pode considerar

$$s(t) = (x_1 + c + a \cos t, y_1 + b \sin t), t \in [0, 2\pi],$$

com $c = +\sqrt{a^2 - b^2}$, expressão esta que pode ser usada em sistemas que simulem a órbita de planetas, como detalhado em [9]. Na figura 9 apresentamos um exemplo do uso deste tipo de trajetórias numa aplicação *mobile*, por um utilizador do já mencionado PlanetarySystemGO, um jogo em que um dos objetivos consiste na perceção, por parte dos estudantes do ensino básico, das escalas correspondentes aos objetos astronómicos, como planetas e estrelas.

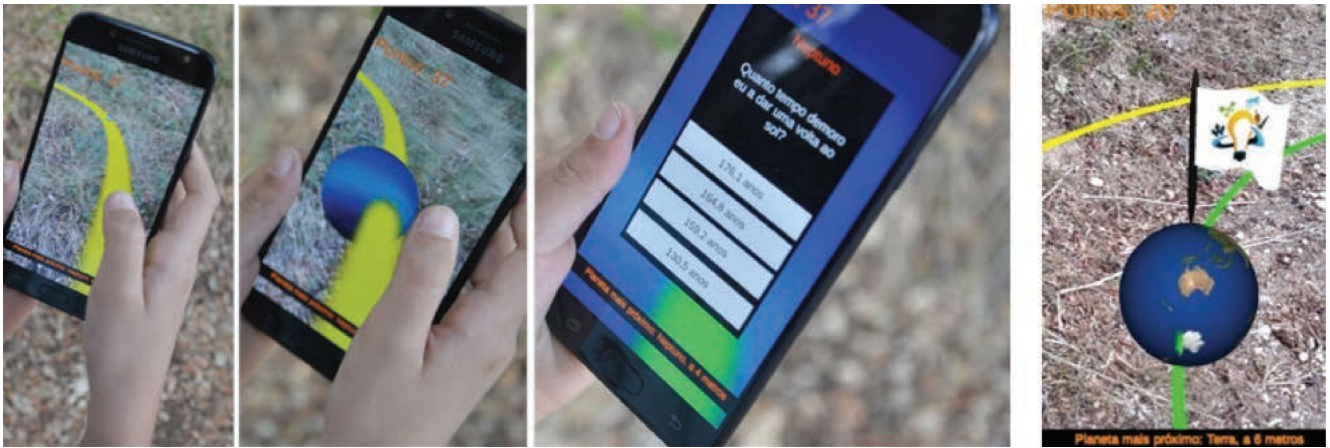


Figura 9. Algumas seqüências do PlanetarySystemGO [3].

Na figura 10 apresentamos trajetórias dos planetas do nosso sistema solar, parametrizadas como elipses, tal como são apresentadas no ecrã do jogador do PlanetarySystemGO.

Uma aplicação diferente do conceito de trajetória foi utilizada no projeto BREUCA³. Este projeto consistiu numa parceria entre as empresas tecnológicas Sketchpixel e Bubblecode, o Instituto Politécnico de Tomar e a Universidade do Minho. O seu objetivo foi o desenvolvimento de um simulador de realidade virtual projetado para ser usado num ambiente de jogo, dando possibilidade aos seus utilizadores de competir em tempo real contra pilotos na pista. Procurou-se que os utilizadores estivessem imersos num ambiente tão real quanto possível, permitindo ajustes na condução e nas definições do automóvel por forma a obter o melhor tempo por volta.

Uma das componentes principais do sistema de simulação desenvolvido foi a descrição da pista. Tendo-se recorrido a pistas reais, como os autódromos do Estoril ou de Braga, a pista é descrita à custa da curvatura κ como função da parametrização s da trajetória correspondente ao percurso da pista feito na sua zona central. A posição do carro é então dada por s , pelo vetor \vec{n} de translação do centro da pista e pelo rumo α relativo à direção da linha central da pista, tal como está esquematizado na figura 11.

Uma pista de automóveis, considerada como um objeto 3D, pode ser modelada à custa de sete variáveis:

- ▶ A posição central da pista $C(s) = (x(s), y(s), z(s))$;
- ▶ A largura da pista $r_w(s)$;
- ▶ A matriz de orientação $R(s)$ que é o produto de três matrizes de rotação em torno de cada um dos eixos:

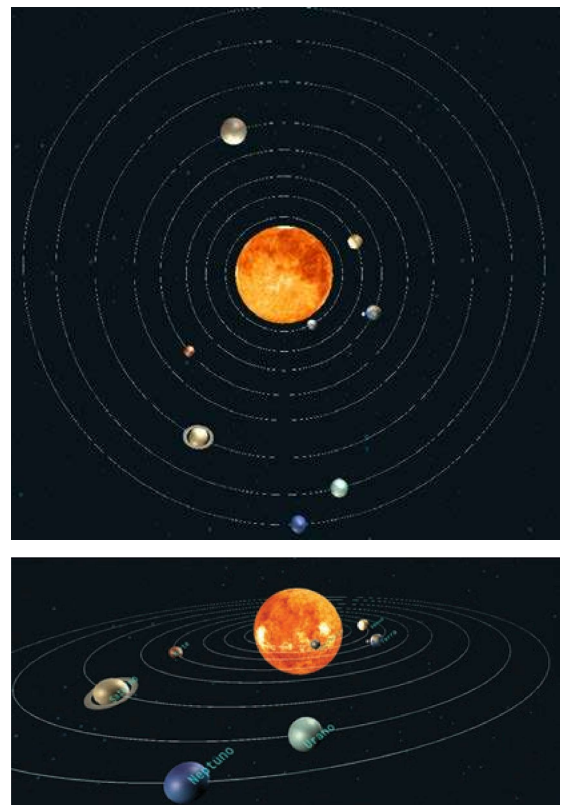


Figura 10. Perspetivas distintas do sistema solar.

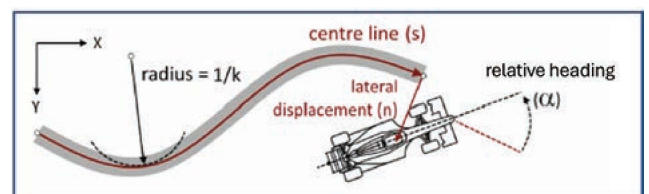


Figura 11. Modelo de representação da pista.

$$R_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \gamma & -\sin \gamma \\ 0 & \sin \gamma & \cos \gamma \end{bmatrix};$$

$$R_y(\gamma) = \begin{bmatrix} \cos \gamma & 0 & -\sin \gamma \\ 0 & 1 & 0 \\ \sin \gamma & 0 & \cos \gamma \end{bmatrix};$$

$$R_z(\gamma) = \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Esta informação é depois usada para construir trajetórias ótimas, no sentido de proporcionar os melhores tempos por volta, recorrendo a algoritmos de controle ótimo [6] e de otimização [12].

4. MODELAÇÃO DE CURVAS E SUPERFÍCIES

Seja f uma função real de variável real definida no intervalo real $[a, b]$, e suponhamos que os valores desta função são conhecidos nos pontos $a = x_0 < x_1 < \dots < x_n = b$. O nosso problema consiste em tentar obter uma aproximação do valor da função f num ponto \bar{x} distinto dos pontos x_i , $i = 0, \dots, n$.

A aproximação de funções arbitrárias em intervalos fechados recorrendo a polinómios tem várias vantagens, tais como a facilidade de criação de estruturas de dados para representar polinómios interpoladores. No entanto, a natureza oscilatória dos polinómios de grau elevado, bem como o facto de uma flutuação dos dados, por muito pequena que seja, poder induzir uma grande alteração nos valores do polinómio, restringe de alguma forma o seu uso. Uma mitigação desta dificuldade reside no uso de aproximações lineares nos intervalos definidos pelos pontos x_i , com $i = 0, \dots, n$, o que, do ponto de vista geométrico, quer dizer que esta função não é “suave”. Esta “suavidade” é muitas vezes um requisito do contexto físico do problema. Nesta secção iremos abordar uma técnica de interpolação polinomial segmentada “suave” que não requer qualquer derivada. A forma mais comum de obter estas aproximações consiste em considerar polinómios de terceiro grau entre cada par de nós, obtendo-se os denominados splines cúbicos segmentados [7]. Um polinómio cúbico tem quatro coeficientes, pelo que há bastante margem de flexibilidade para garantir o cumprimento de um dado número de condições, como, por exemplo, a existência de primeiras e segundas derivadas contínuas, de modo a assegurar a pretendida “suavidade”.

Apresentamos seguidamente a definição de spline interpolador cúbico.

Dada a função f nas condições referidas no início desta secção, o spline interpolador cúbico S de f no intervalo $[a, b]$ é uma função que satisfaz as seguintes condições:

1. A restrição de S ao intervalo $[x_j, x_{j+1}]$ é definida, para cada $j = 0, 1, \dots, n-1$, pelo polinómio cúbico $S_j(x)$, no intervalo $[x_j, x_{j+1}]$, para todo $j = 0, 1, \dots, n-1$;
2. $S(x_j) = f(x_j)$ para todo $j = 0, 1, \dots, n$;
3. $S_{j+1}(x_{j+1}) = S_j(x_{j+1})$ para todo $j = 0, 1, \dots, n-2$;
4. $S'_{j+1}(x_{j+1}) = S'_j(x_{j+1})$ para todo $j = 0, 1, \dots, n-2$;
5. $S''_{j+1}(x_{j+1}) = S''_j(x_{j+1})$ para todo $j = 0, 1, \dots, n-2$;
6. Uma das seguintes condições de fronteira é satisfeita:
 - (a) $S''(x_0) = S''(x_n) = 0$ (fronteira livre ou natural);
 - (b) $S'(x_0) = f'(x_0)$ e $S'(x_n) = f'(x_n)$ (fronteira ligada).

Os polinómios B-spline correspondem geometricamente a curvas no espaço, sendo dos tipos de spline mais usados em várias aplicações. Tipicamente são construídos aproximando um conjunto de pontos, tendo, no entanto, duas importantes vantagens:

1. O grau de um polinómio B-spline é independente (até certo ponto) do número de pontos de controle considerados;
2. Os B-splines permitem algum controlo sobre a forma da curva construída.

Como contrapartida a estas duas vantagens, os B-splines são de alguma forma mais complexos do que outros tipos de splines bastante usados, como os de Bézier [10].

O cálculo das coordenadas das posições ao longo de um B-spline pode ser feito a partir da expressão

$$P(u) = \sum_{k=0}^n p_k B_{k,d}(u), \quad (4.1)$$

com $u_{\min} \leq u \leq u_{\max}$, $2 \leq d \leq n+1$, em que p_k são os pontos de controle, com $k = 1, \dots, n$. As funções $B_{k,d}$ são definidas através das denominadas fórmulas recursivas de

³ <https://breuca.com>

Cox-de Boor [2].

Assim como os B-splines que acabamos de apresentar nos permitem representar curvas, é possível estender este conceito para a representação de superfícies. Em vez de termos um conjunto de pontos de controle p_k , com $k = 0, \dots, n$, passamos a ter um retângulo formado por pontos p_{kj} , com $k = 1, \dots, n$ e $j = 1, \dots, m$, obtendo-se

$$P(u, v) = \sum_{k=0}^n \sum_{j=0}^m p_{kj} B_{k,d}(u) B_{j,d}(v) \quad (4.2)$$

Na figura 12 apresentamos um exemplo de uma superfície B-spline. Esta representação foi feita com recurso à biblioteca OpenGL [8], introduzindo efeitos de iluminação em tempo real para uma melhor percepção da tridimensionalidade do objeto representado.

A representação de uma superfície como a da figura 12



Figura 12. Exemplo de uma superfície B-spline.

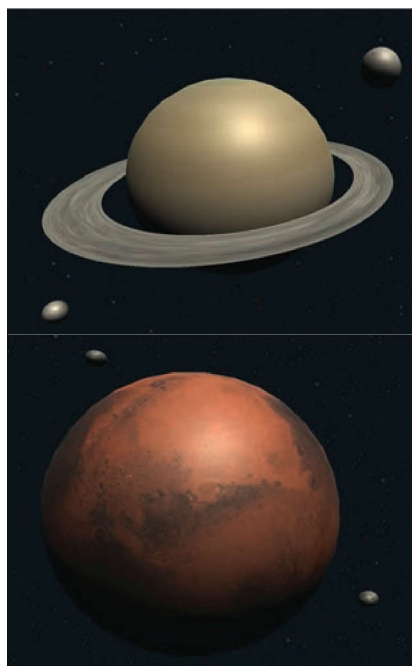


Figura 13. Representação 3D de planetas.

é pouco cativante do ponto de vista do utilizador final. Este aspeto pode ser melhorado recorrendo a modelos de iluminação mais sofisticados ou à aplicação de texturas sobre a superfície [11], que corresponde ao mapeamento de uma imagem bidimensional à superfície 3D, dando-lhe assim maior realismo e atratividade. As representações 3D dos planetas no PlanetarySystemGO são um caso de aplicação, e na figura 13 são apresentados exemplos.

5. ANIMAÇÃO, DEFORMAÇÃO E MOVIMENTO

Uma das aplicações fundamentais da computação gráfica tem a ver, como já foi demonstrado, com a criação de modelos tridimensionais e sua animação. Associado a estes conceitos, há necessariamente a considerar a noção de deformação, que consiste em alterar de forma dinâmica as dimensões e posições relativas de objetos presentes na cena interativa, como, por exemplo, em jogos de vídeo de combate corpo a corpo. Há um vasto conjunto de conceitos matemáticos necessários para a criação de modelos de deformação e de animação, além da breve análise que fizemos sobre modelos de movimentação [1]. Entre estes conceitos contam-se os de valor e de vetor próprio.

Recordemos que, se A é uma matriz quadrada (real ou complexa) de ordem n , então diz-se que o escalar λ (real ou complexo – mesmo para matrizes reais) é um valor próprio de A se existir um vetor x de ordem n diferente do vetor nulo tal que

$$Ax = \lambda x.$$

Ao vetor x que satisfaça esta condição dá-se o nome de vetor próprio de A associado ao valor próprio λ .

No contexto da animação e deformação em computação gráfica, os valores próprios são frequentemente utilizados para analisar e controlar a deformação de modelos 3D, particularmente no campo de animação de personagens [5]. Seguidamente apresentamos alguns campos concretos de aplicação.

Os modelos de personagens em animação são frequentemente representados como malhas. Os valores próprios de matrizes associadas a essas malhas podem ser usados para analisar como a malha se deforma quando submetida a várias transformações, como, por exemplo, na animação esquelética ou em processos de metamorfose (*morph targets*). Por outro lado, os valores próprios permitem compreender a influência das intersecções da malha – as articulações esqueléticas do modelo de uma personagem – durante o processo de animação. Trata-se de uma questão crucial para obter deformações realistas e de aparência natural.

Por outro lado, os vetores próprios estão associados a formas de representação da deformação de uma estrutura ou de um modelo. Neste sentido, podem ser usados para criar *eigenskeletons*, que são formas de recuperar, pelo menos parcialmente, o realismo da personagem original, que poderá ter-se perdido ao representá-la por uma malha, tal como é exemplificado na figura 14, onde mostramos deformações sobre o modelo original, mais à esquerda.



Figura 14: Movendo o braço de Neptuno usando deformações [5].

6. CONCLUSÕES

Nestas breves páginas pretendemos mostrar ao leitor como muito do que se vê, no que à Realidade Virtual diz respeito, depende da qualidade dos modelos tridimensionais que se criam, que por sua vez recorrem fortemente a modelos e conceitos matemáticos. Neste trabalho não tivemos a pretensão de elaborar sobre os conceitos mais avançados, mas sim mostrar como até os conceitos mais conhecidos do cálculo e da álgebra linear estão presentes e ajudam a descrever o mundo virtual. Por outro lado, tivemos a preocupação de mostrar estes conceitos em ação em dois projetos em que estivemos envolvidos, o projeto PlanetarySystemGO, que visou a produção de um sistema de realidade virtual e aumentada, e o projeto BREUCA, que conduziu ao desenvolvimento de um simulador de corridas de automóveis, destinado ao treino fora de pista de pilotos profissionais, mas também a análise do seu desempenho, considerando a trajetória e as demais características da pista.

REFERÊNCIAS

[1] K. Anjyo, H. Ochiai, and B.A. Barsky. *Mathematical Basics of Motion and Deformation in Computer Graphics: Second Edition*. Synthesis Lectures on Visual Computing: Computer Graphics, Animation, Computational Photography and Imaging. Morgan & Claypool Publishers, 2017.

[2] R. L. Burden. *Numerical analysis*. Brooks/Cole Cengage Learning, 2011.

[3] M. C. Costa, A. Manso, and J. M. Patrício. "Design of a mobile augmented reality platform with game-based learning purposes". *Information*, 11(3), 2020.

[4] M. C. Costa, P. Santos, J. M. Patrício, and A. Manso. An interactive information system that supports an augmented reality game in the context of game-based learning. *Multimodal Technologies and Interaction*, 5(12), 2021.

[5] T. K. Dey, P. Ranjan, and Y. Wang. "Eigen deformation of 3D models". *Vis Comput*, 28:585–595, 2012.

[6] Donald E Kirk. *Optimal control theory: an introduction*. Courier Corporation, 2004.

[7] G. D. Knott. *Interpolating Cubic Splines*. Birkhauser, 2000.

[8] S. Kosarevsky and V. Latypov. *3D Graphics Rendering Cookbook: A comprehensive guide to exploring rendering algorithms in modern OpenGL and Vulkan*. Packt Publishing Ltd, 2021.

[9] V. Marçal and H. Simões. *Planetary system frontoffice*. Technical report, Instituto Politécnico de Tomar, 2020.

[10] H. Prautzsch, W. Boehm, and M. Paluszny. *Bézier and B-Spline Techniques*. Mathematics and Visualization. Springer Berlin Heidelberg, 2013.

[11] G. Sellers, R. Wright, and N. Haemel. *OpenGL superBible: comprehensive tutorial and reference*. Addison-Wesley, 2013.

[12] Jan A Snyman, Daniel N Wilke, et al. *Practical mathematical optimization*. Springer, 2005.

SOBRE O AUTOR

João Patrício é Professor Adjunto do Instituto Politécnico de Tomar (Unidade Departamental de Matemática e Física) e membro integrado do Centro de Investigação em Cidades Inteligentes do IPT. Os seus interesses de investigação dividem-se entre os métodos numéricos para problemas de grande dimensão, a otimização e análise de dados e os modelos matemáticos e computacionais para computação gráfica e realidade virtual. Foi diretor da Escola Superior de Tecnologia de Tomar do IPT e presentemente é pró-presidente do Instituto Politécnico de Tomar.

Secção coordenada pela PT-MATHS-IN, Rede Portuguesa de Matemática para a Indústria e Inovação pt-maths-in@spm.pt