Programação Orientada a Objetos

Gonzalo Travieso

2021

Quaternions

Quaternions são números com álgebra não-comutativa (isto é, $ab \neq ba$ em geral) que generalizam números complexos. Um quaternion pode ser representado como:

$$q_1 + q_2\mathbf{i} + q_3\mathbf{j} + q_4\mathbf{k},$$

onde q_1, q_2, q_3 e q_4 são números reais, enquanto $\mathbf{i}$, $\mathbf{j}$ e $\mathbf{k}$ são números distintos (imaginários) que satisfazem as relações:

$$\begin{aligned}
\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} &= -1, \\
\mathbf{ij} = -\mathbf{ji} &= \mathbf{k}, \\
\mathbf{jk} = -\mathbf{kj} &= \mathbf{i}, \\
\mathbf{ki} = -\mathbf{ik} &= \mathbf{j}.
\end{aligned}$$

As três últimas linhas são redundantes com a primeira. O número real x corresponde a $x + 0\mathbf{i} + 0\mathbf{j} + 0\mathbf{k}$ e o número complexo $z = a + b\mathbf{i}$ corresponde a $a + b\mathbf{i} + 0\mathbf{j} + 0\mathbf{k}$.

O *conjugado* de $q = q_1 + q_2\mathbf{i} + q_3\mathbf{j} + q_4\mathbf{k}$ é definido como

$$\bar{q} = q_1 - q_2\mathbf{i} - q_3\mathbf{j} - q_4\mathbf{k}.$$

A soma de $a = a_1 + a_2\mathbf{i} + a_3\mathbf{j} + a_4\mathbf{k}$ com $b = b_1 + b_2\mathbf{i} + b_3\mathbf{j} + b_4\mathbf{k}$ é

$$a + b = (a_1 + b_1) + (a_2 + b_2)\mathbf{i} + (a_3 + b_3)\mathbf{j} + (a_4 + b_4)\mathbf{k}.$$

O produto é dado por

$$\begin{aligned}
ab &= (a_1b_1 - a_2b_2 - a_3b_3 - a_4b_4) \\
&+ (a_1b_2 + a_2b_1 + a_3b_4 - a_4b_3)\mathbf{i} \\
&+ (a_1b_3 - a_2b_4 + a_3b_1 + a_4b_2)\mathbf{j} \\
&+ (a_1b_4 + a_2b_3 - a_3b_2 + a_4b_1)\mathbf{k}.
\end{aligned}$$

Note que, se a é real, então $a_2 = a_3 = a_4 = 0$ e $a_1 = a$, e portanto a expressão acima simplifica para $ab = ab_1 + ab_2\mathbf{i} + ab_3\mathbf{j} + ab_4\mathbf{k}$. Também dá para verificar que $q\bar{q} = \bar{q}q = q_1^2 + q_2^2 + q_3^2 + q_4^2$ é sempre real.[1]

[1] Esta expressão, apesar de uma consequência da fórmula de multiplicação, **deve ser usada em calculos em computador sempre que for adequada**: Os valores dos coeficientes de $\mathbf{i}, \mathbf{j}, \mathbf{k}$ se anulam, mas devido a problemas de arredondamento, se $q\bar{q}$ for calculado pela fórmula de multiplicação podem restar coeficientes imaginários não-nulos (o que ocasiona problema, por exemplo, no cálculo da norma).

O inverso de um quaternion é definido como:

$$q^{-1} = \frac{1}{q\bar{q}}\bar{q}$$

e sua norma como:

$$|q| = \sqrt{q\bar{q}}.$$

A partir do inverso podemos definir a divisão:

$$\frac{a}{b} = ab^{-1} = \frac{1}{b\bar{b}}a\bar{b}.$$

Rotações

Quaternions são úteis para representar rotações em espaços tridimensionais. Uma forma de realizar isso é a seguinte:

1. Considere um ponto P com coordenadas cartesianas (x, y, z). Esse ponto pode ser representado pelo quaternion $p = 0 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ (repare na parte real 0).

2. Uma rotação é especificada através de um vetor $\mathbf{v}$, que indica o eixo da rotação, e por um ângulo α, que indica o ângulo de rotação em torno desse eixo. Representamos uma rotação definida dessa forma através do quaternion:

$$r = \cos\frac{\alpha}{2} + \sin\frac{\alpha}{2}\frac{1}{\|\mathbf{v}\|}(v_x\mathbf{i} + v_y\mathbf{j} + v_z\mathbf{k}),$$

   onde (v_x, v_y, v_z) são as componentes de $\mathbf{v}$ e $\|\mathbf{v}\| = \sqrt{v_x^2 + v_y^2 + v_z^2}$.

3. Para realizar uma rotação no ponto representado pelo quaternion p (gerando o novo ponto p'), basta fazer:[2]

$$p' = rpr^{-1}.$$

   (Lembre-se que o produto não é comutativo!)

Trabalho

1. Escreva uma classe para lidar com quaternions, cuidando para que objetos dessa classe possam trabalhar conjuntamente com valores inteiros, reais e complexos, quando adequado (i.e. deve-se definir operações aritméticas tanto de quaternions entre si como com esses outros tipos).

   A implementação deve ser feita num módulo denominado `quaternion`, implementado em um arquivo `quaternion.py`, e a classe deve se chamar `Quaternion`.

2. Escreva um programa que realize o seguinte:

   (a) Cria um cubo com vértices nos pontos $(1, 1, 1)$, $(1, 1, -1)$, $(1, -1, 1)$, $(-1, 1, 1)$, $(1, -1, -1)$, $(-1, 1, -1)$, $(-1, -1, 1)$ e $(-1, -1, -1)$.

[2] Você consegue mostrar que p' terá parte real 0, como adequado para a representação de um ponto? Não é necessário entregar esta demonstração.

(b) Repetidamente pede ao usuário um eixo e um ângulo de rotação. Realiza a rotação especificada no cubo (acumuladamente, isto é, sobre a posição do cubo após a última rotação) e mostra os novos valores dos vértices (para cada rotação). Devem ser mostradas as coordenadas x, y, z, e não os quaternions correspondentes.

Algumas observações:

- Note que foi pedida uma classe para quaternions sem se referir ao programa de rotação. *Você deve implementar uma classe que se baseie nos conceitos básicos de quaternions expostos, e não apenas o que será necessário para o programa de rotação.*
- Use sobrecarga de operadores.
- No programa de rotação, use uma classe para representar o cubo, com um método para realizar a rotação especificada por um quaternion dado e métodos para acesso aos vértices. A classe para cubo não precisa ser geral, pode ter apenas o que é necessário ao programa.
- Você pode usar para a implementação qualquer característica da linguagem que desejar, bem como módulos pré-definidos no Python, mas não outros módulos da comunidade. Veja a lista de módulos do Python.
- Para testar, aplique rotações de múltiplos de 90º, que devem resultar no mesmo cubo, apenas com os vértices posicionados em outra ordem.
- Você pode fazer o programa mais flexível ou geral do que o pedido.
- Entregue os arquivos com os códigos do módulo de quaternions e o programa de rotação do cubo, empacotados em um único `.zip` ou `.tar.gz` .