Transformações Geométricas em Python com Matplotlib e NumPy

[ICP114] Computação I

Enzo Monaco
Vitor Pedroso
João Victor Duarte

Instituto de Computação
Universidade Federal do Rio de Janeiro

Sumário

Sumário

Objetivo

Nesta apresentação, destaca-se a aplicação prática de algumas transformações lineares geométricas em quadriláteros: rotação, cisalhamento e compressão. Utilizamos aqui as bibliotecas Math, NumPy e Matplotlib, sendo essa última essencial para o funcionamento da função de plot.

Sumário

Introdução - Transformações Lineares

Definição

Sejam U e V espaços vetoriais sobre o corpo R. Uma aplicação $T : U \to V$ é denominada **Transformação Linear** de U em V se, e somente se, satisfaz:

1. $T(u_1 + u_2) = T(u_1) + T(u_2), \quad \forall u_1, u_2 \in U$
2. $T(\alpha u) = \alpha T(u), \quad \forall u \in U$

- Podemos também escrever uma transformação linear em função de uma matriz de coeficientes e um vetor coluna. Essa matriz é chamada de matriz de transformação.
- $T(\vec{v}) = A\vec{v}$, onde T é uma transformação linear e A a matriz de transformação.

Introdução - Transformações Lineares

- A grosso modo, na Álgebra Linear, uma transformação linear nada mais é do que uma função aplicada a matrizes e vetores. Até mesmo os espaços vetoriais definidos na lei da transformação são chamados de Domínio e Contradomínio.
- Assim como nas funções, também é permitido aplicar transformações geométricas em seu gráfico através de matrizes de transformação notáveis. As solicitadas para o presente trabalho são: rotação, cisalhamento e compressão.
- A ideia aqui é aplicar essas transformações usando o Python.

Sumário

O código no geral

O código realizado possui uma estrutura com base na *modularização* de funções – técnica que consiste em separar o código em várias funções com tarefas bem definidas –, a fim de se obter um caráter mais semântico e legível. Sua estrutura contém 2 funções auxiliares, 6 funções principais e uma função main que aplica todas funções numa lógica principal.

Funções Auxiliares

As funções auxiliares são funções que realizam tarefas repetitivas e que tem mais a ver com a formatação do programa no Terminal. Essas são:

- formatar(): printa uma linha para criar uma GUI (Interface Gráfica do Usuário) bonita.
- invalida(): printa uma mensagem de erro usada para quando avisar ao usuário de alguma operação inválida.

Funções Principais

As funções principais são funções que realizam tarefas essenciais da lógica do programa. Essas são:

- pontos(): Obtém os pontos do quadrilátero do usuário.
- tipo_transformacao(): Obtém o tipo de transformação desejada pelo usuário.
- rotacao(): Aplica a operação de rotação no quadrilátero dado.
- cisalhamento(): Aplica a operação de cisalhamento no quadrilátero dado.
- compressao(): Aplica a operação de compressão no quadrilátero dado.
- show_polygon(original_points, transformed_points, title): Função principal de plot.

Função Main

- A função main() contém toda a lógica do programa aplicada à GUI. Isso facilita o processo, pois o código fica com o controle de fluxo bem definido e bem legível.
- Ela começa com o loop do programa para executar a GUI. Posteriormente, ela executa a função tipo_transformacao() para receber a escolha do usuário.

Função Main

- Se o número for igual a 0, finaliza o programa.
- Se o número for igual a 1, executa a função de rotação, pede os pontos, o ângulo de rotação em graus, aplica a rotação e plota os gráficos da figura original e da figura rotacionada.
- Se o número for igual a 2, executa a função de cisalhamento, pede os pontos, os fatores de cisalhamento em x e y, aplica o cisalhamento e plota os gráficos da figura original e da figura cisalhada.
- Se o número for igual a 3, executa a função de compressão, pede os pontos, os fatores de compressão em x e y, aplica a compressão e plota os gráficos da figura original e da figura comprimida.

Função Main - Restrições

O programa não permite:

1. $tipo \notin \{0, 1, 2, 3\}$.
2. $\theta = 0$, sendo θ o ângulo de rotação.
3. os fatores de cisalhamento em x e em y serem iguais a 0 **simultaneamente**.
4. os fatores de compressão serem iguais a 1, **simultaneamente**.

Sumário

Exemplos de Aplicações - Rotação

- Rotação em 76° nos pontos $(0,0),(1,0),(1,2),(0,2)$.
- Rotação em 198° nos pontos $(2,6),(5,6),(7,3)e(0,3)$.

Exemplos de Aplicações - Cisalhamento

- Cisalhamento em x com fator 0.4 e em y com fator 0.2 nos pontos $(5, 5), (3, 8), (1, 5), (3, 2)$.
- Cisalhamento em x com fator 1.4 e em y com fator 2.2 nos pontos $(1, 3), (3, 1), (8, 1), (6, 3)$.

Exemplos de Aplicações - Compressão

- Compressão em x com fator -0.3 e em y com fator 0.2 nos pontos $(4, 2), (2, -1), (1, -4), (4, -4)$.
- Compressão em x com fator 1 e em y com fator 1.4 nos pontos $(10, 2), (10, 4), (14, 2), (14, 4)$.

Sumário

Conclusão

Em resumo, este trabalho apresentou uma implementação eficaz de transformações geométricas em quadriláteros usando Python, Matplotlib e NumPy. A modularização do código, a interface amigável e os exemplos práticos demonstraram a aplicabilidade e versatilidade das transformações lineares. Este projeto destaca a importância da álgebra linear na computação gráfica, proporcionando uma base sólida para futuros desenvolvimentos. A combinação de simplicidade e eficiência ilustra a potência das bibliotecas Python no contexto gráfico.

Conclusão

Obrigado a todos por assistir a apresentação!