Você sabia que seu material didático é interativo e multimídia?
Ele possibilita diversas formas de interação com o conteúdo, a qualquer hora e de qualquer lugar. Mas na versão impressa, alguns conteúdos interativos são perdidos, por isso, fique atento! Sempre que possível, opte pela versão digital. Bons estudos!

Imprimir

Computação Gráfica e Processamento de Imagens

CGPI: representações da imagem

Unidade 1 - Seção 3

Abordaremos nesta webaula o processamento de imagens 2D (vetoriais e matriciais), a criação de retas em imagens vetoriais e matriciais, bem como algoritmos para o desenho de retas e para o desenho de círculos.

A linguagem Python

A linguagem Python é uma linguagem gratuita provida de um vasto conjunto de bibliotecas gratuitas para computação gráfica e processamento de imagens. Possui sintaxe simplificada, nos moldes de linguagens de script, mas com suporte a orientação a objetos e integração com bibliotecas de alta performance implementadas em C.

Possui, também, bibliotecas gratuitas de manipulação de imagens vetoriais e matriciais bastante intuitivas, o que permite concentrarmo-nos nos conceitos de computação gráfica e processamento de imagens e não em dificuldades da linguagem ou das bibliotecas.

Processamento de imagens vetoriais

Para o processamento de imagens vetoriais, utilizaremos a biblioteca Pycairo, uma interface Python para a biblioteca gráfica cairo, implementada em C. Para criar uma imagem vetorial no cairo, é preciso criar um objeto Surface.
Há vários tipos de Surfaces, de acordo com o tipo de descritor de imagem desejado. Para se criar uma imagem v em memória, a ser posteriormente escrita em um arquivo do tipo *Scalable Vector Graphics* (SVG) e nome *exemplo.svg*, com resolução de 200x200 pixels, basta uma linha além de importar a biblioteca.

```
1 import cairo
2 v = cairo.SVGSurface("cgpiU1S2.svg", 201, 201)
```

Para o processamento de imagens matriciais

Para o processamento de imagens matriciais, utilizaremos a biblioteca NumPy, também implementada em C e com interface para o Python. NumPy é uma biblioteca que facilita a manipulação de matrizes em Python. Para se criar uma imagem matricial *f* em níveis de cinza, toda preta, de resolução 200x200 pixels, basta importar a biblioteca e criar uma matriz 200x200 de zeros no tipo inteiro de 8 bits.

```
1 import numpy
2 f = numpy.zeros((200,200),dtype=numpy.uint8)
```

Criação de retas em imagens vetoriais e matriciais

Tanto em imagens vetoriais quanto em imagens matriciais, a criação de retas de fato não é feita, mas sim a criação de segmentos de retas, visto que as imagens são funções bidimensionais dentro dos limites de um quadro. A criação de um segmento de reta em uma imagem vetorial é muito simples.
Basta acrescentar a descrição de que deve ser adicionado um segmento de reta. A reta deverá ser desenhada posteriormente sobre uma imagem matricial pelo aplicativo que irá exibir ou imprimir a imagem vetorial. Com o cairo é preciso criar um Context para desenhar sobre uma Surface. Considerando a imagem vetorial v criada anteriormente, isto é feito como se segue.

```
context = cairo.Context(v)
context.scale(10, 10)
context.set_line_width(0.02)
context.move_to(0.2, 0.2)
context.line_to(0.8,0.6)
context.stroke()
```

Importante!

A criação de segmentos de retas em imagens matriciais exige o desenho de cada ponto pertencente ao segmento. Isto nem sempre é evidente, visto que uma imagem matricial possui pontos apenas em coordenadas inteiras.

Algoritmo para o desenho de retas

O algoritmo mais conhecido para o desenho de retas é o algoritmo de Bresenham. Para iniciar, pode-se criar a função *desenhaReta(f,p,q,g)*, que desenha sobre a imagem matricial f um segmento de reta que parte do ponto $p = (xp, yp)$para o ponto $q = (xq, yq)$na cor g.
Vale mencionar que para otimizar a função *desenhaRetaCaso1*, Bresenham eliminou o uso de multiplicações de números reais e arredondamentos, trabalhando apenas com números inteiros.

Algoritmos de computação gráfica e processamento de imagens exigem muitas iterações. O desenho de uma forma, por exemplo, com um milhão de pontos, exigirá um milhão de iterações, uma para cada ponto. No nível do hardware, operações com números inteiros são executadas muito mais rápido que operações com números reais. Por este motivo, o uso de variáveis inteiras deve ser sempre priorizado. Operações com inteiros são executadas mais rápido no processador.

Algoritmo para o desenho de círculo

O algoritmo para o desenho de círculos se comporta de maneira bastante similar ao de desenho de retas, mas para compreende-lo é preciso apresentar algumas propriedades do círculo discreto. A equação do círculo contínuo de centro em *(xc,yc)* e raio *r* é $(xc + x)^2 + (yc + y)^2 = r^2$, onde *(x,y)* é um deslocamento a partir do centro *(xc,yc)*.

Tome nota!

Por fim, espera-se que com essa webaula o futuro profissional seja capaz de compreender a prática de implementação de algoritmos de computação gráfica e processamento de imagens. Não se esqueça de recorrer ao livro didático para aprofundar os seus estudos.

Vídeo de encerramento

Para visualizar o vídeo, acesse seu material digital.