CÁLCULO II

USO DE CURVAS PARAMETRIZADAS E COORDENADAS POLARES NO DESENVOLVIMENTO DE UMA APLICAÇÃO LÚDICA

Antônio Carlos Durães da Silva
Joel Will Belmiro

Resumo:

Neste documento juntamente ao material auxiliar (código-fonte), propõe-se a apresentação de aplicações que utilizem a comodidade proporcionada pelo sistema de coordenadas polares e da maleabilidade que a parametrização de equações permite. Além disso, os conhecimentos citados serão agregados à programação de computadores, com uso da linguagem de programação Python, para construir um simulador de um radar.

O principal intuito desse material é apresentar a aplicação desenvolvida, para mostrar de forma lúdica e descontraída que o aprendizado dos conteúdos citados não precisa estar fundamentado apenas em fórmulas complexas ou em aplicações puramente industriais.

Palavras-chaves:
curvas paramétricas, coordenadas polares, python, matplotlib, radar, dragon ball

1 INTRODUÇÃO

A parametrização de equações é um recurso que, em resumo, trata-se de expressar uma equação ou mais equações em função de uma variável em comum, chamada de parâmetro [1]. Com a parametrização, é possível, por exemplo, expressar a fórmula geral de uma circunferência em termos de variável única (geralmente chamada de **t**, por ser associada ao tempo), em vez das duas variáveis, x e y, veja na fórmula 1.

Fórmula 1: Equação de um círculo

(a) Equação em coordenadas cartesianas;
(b) Equação em equações paramétricas

(a)

$$r^2 = x^2 + y^2$$

(b)

$$x = r\cos(t)$$
$$y = r\,sen(t)$$

Imagine um ponto **A** marcado na origem de um plano e um ponto **B** marcado em uma posição qualquer. O sistema de coordenadas polares, trata-se de uma ferramenta que utiliza o comprimento do segmento de reta **AB** e a inclinação desse segmento em relação ao eixo X para determinar a localização do ponto em um plano bidimensional [2]. Embora o exemplo dado seja com apenas um ponto, tal sistema de coordenadas possibilita a representação gráfica de formas geométricas simples e conhecidas, como uma circunferência, às mais complexas, fascinantes e pouco conhecidas curvas.

Fórmula 2: Equação de um círculo

(a) Equação em coordenadas cartesianas; (b) Equação em coordenadas polares

(a)

$$r^2 = x^2 + y^2$$

(b)

$$r = \textbf{\textit{constante}}$$

2 EXEMPLO DE APLICAÇÕES

Tendo em vista a facilidade que o sistema de coordenadas polares proporciona para trabalhar-se com objetos ou superfícies circulares, esféricas, cilíndricas e cônicas; um de seus empregos é no desenvolvimento de Sistemas de Informação Geográfica (SIG/GIS)[3], na construção de radares, mapas ou localização geográfica de pontos de interesse. Tais coordenadas também são aplicadas na modelagem de peças para máquinas e na própria movimentação de algumas máquinas [4].

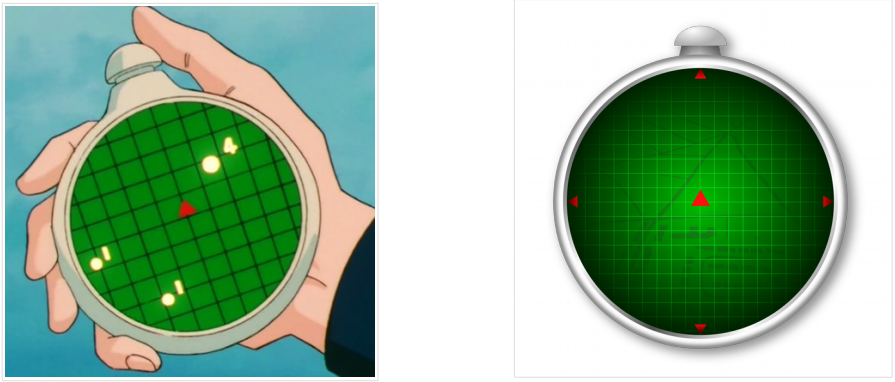
As equações parametrizadas possuem curvas: Como a de Bézier, que são utilizadas no processo de aproximação e aperfeiçoamento de superfícies nas áreas de Computação Gráfica e Design [5]; que permitem simular a temperatura de estruturas de aço em casos reais de incêndio [6]; que aproximam-se da modelagem de formas arquitetônicas contemporâneas [7], permitindo sua reprodução e customização em ferramentas Computer Aided Design (CAD).

3 PROJETO DESENVOLVIDO

A fim de criar uma aplicação atrativa aos olhos de quem está iniciando o aprendizado sobre coordenadas polares e/ou equações parametrizas, contudo, sem perder totalmente a base em aplicações reais, um radar especial foi desenvolvido. O lúdico desse objeto se dá pela sua inspiração no radar utilizado na animação japonesa Dragon Ball Z.

Figura 1: Radares usados como inspiração

(a) (b)



Fonte:

(a): https://dragonball.fandom.com/wiki/Dragon_Radar?file=DBLocationsOnNamek.png

(b): https://www.kisscc0.com/clipart/imaging-radar-computer-icons-dragon-radar-dragon-b-wimxus/

4 POR QUE UTILIZAR COORDENADAS POLAR E EQUAÇÕES PARAMÉTRICAS?

4.1 APLICAÇÃO DE COORDENADAS POLARES

Há três objetos principais que foram modelados para construção o radar, são eles: Plano polar (engloba toda a área do radar), setor (responsável por percorrer o plano de theta variando de 0 a 360 graus) e os pontos que são sorteados pseudo-aleatoriamente e desenhados no plano.

O plano, suas cores e os indicadores triangulares vermelhos (aparecem no centro do mapa e em 0°, 90°, 180°, 270°) são desenhados apenas uma vez pelo computador. Portanto, a forma como a plotagem desses objetos ocorre não traz impactos significativos durante a animação do radar.

Contudo, o setor que percorre o plano, é formado por duas posições (uma indica onde é o ponto de início e a outra indica o ponto de término) e é

atualizado constantemente. Se o setor se mover a cada 1 grau, são 360 movimentos multiplicado por 2 (Número de posições a serem atualizadas por movimento) a cada volta completa na circunferência. Somado ao custo citado, há o custo de identificar no mapa, a posição de cada ponto gerado.

Dentro dos custos envolvidos, com sistema de coordenadas polares, a única operação matemática realizada é o incremento de theta1 e theta2 para movimentar o setor, algo pouco custoso computacionalmente, pois trata-se de uma soma simples, além de que, o valor de r não é alterado. Se o sistema fosse de coordenadas retangulares com uso de distância euclidiana, a cada atualização do setor seria necessário a radiciação da soma de dois quadrados.

4.2 APLICAÇÃO DE EQUAÇÕES PARAMETRIZADAS

Para simular um efeito de gradação de cores (gradiente), visualizado ao fundo do plano, foi necessária a plotagem de dezenas de círculos de diferentes tonalidades e diâmetros. Este recurso foi o único encontrado para simular o degradê, uma vez que o efeito nativo e verdadeiro apresentou altíssimo consumo de recursos (principalmente memória) quando trabalhado junto a animação do radar.

Em resumo, a ideia aplicada baseia-se na plotagem do maior círculo (C_n) próximo a um tom bem escuro da cor verde, e o menor círculo (C_1), próximo a um verde mais vivo, mais claro. A medida que o diâmetro do círculo diminui, a cor de seu preenchimento aproxima-se mais do tom de verde vivo.

Ao optar em trabalhar com o sistema de cores RGB em hexadecimal, é possível ter a cor com brilho entre 0 (preto absoluto) até 255 (cor mais viva possível). Tendo em vista este intervalo, a simulação perfeita de um degradê, aplicada no contexto explicado, requereria 256 círculos (ou linhas), cada um com o brilho variando em um ponto. Contudo, verificou-se que, com um número de círculos igual ou superior a 100, a animação do radar apresentou perda de fluidez.

Dado que a variação de brilho entre um círculo e outro deve ser a mesma, e que o maior círculo deve ter seu brilho com valor 0 (preto), e o menor, com valor 255 (mais claro), é possível relacionar a quantidade de círculos com a variação absoluta de brilho (ver fórmula 3).

Fórmula 3: Variação de brilho por círculo plotado

B = Variação absoluta de brilho; **T** = Número de círculos

$$T \cdot B = 255$$

$$B = \frac{255}{T} \ \ ou \ \ T = \frac{255}{B}$$

O número de círculos (chamado anteriormente de “T”) é um valor essencial para determinar a diferença das dimensões entre os círculos, considerando que diferença de diâmetro entre um círculo C_i e um círculo C_{i+1} é igual, e que o diâmetro do fundo do radar é uma constante. Confira a relação na fórmula 4.

Fórmula 4: Variação de brilho por círculo plotado

R = Raio do fundo do radar

D = Diferença entre a dimensão de um círculo e seu próximo

$$D = \frac{R}{T} \ \ ou \ \ D = \frac{R}{\left(\frac{255}{B}\right)} \ \ ou \ \ D = \frac{R \times B}{255}$$

Abaixo segue um exemplo para o número de 25 círculos e valor 0.52 para o raio do fundo:

Fórmula 5: Exemplo de aplicação das fórmulas para 25 círculos

$$T \cdot B = 255$$

$$B = \frac{255}{25} = 10.2 * \qquad D = \frac{R}{T} = \frac{0.52}{50} = 0.0208$$

**Como o valor de brilho para cor deve ser um inteiro, o valor seria arredondado para 10. Ou seja, cada um dos círculos possui o brilho maior em 10 pontos absolutos (o brilho máximo, neste caso, será de 250).*

Exemplo com gradiente mais suave, contendo 85 círculos:

Fórmula 6: Exemplo de aplicação das fórmulas para 100 círculos

$$T \cdot B = 255$$

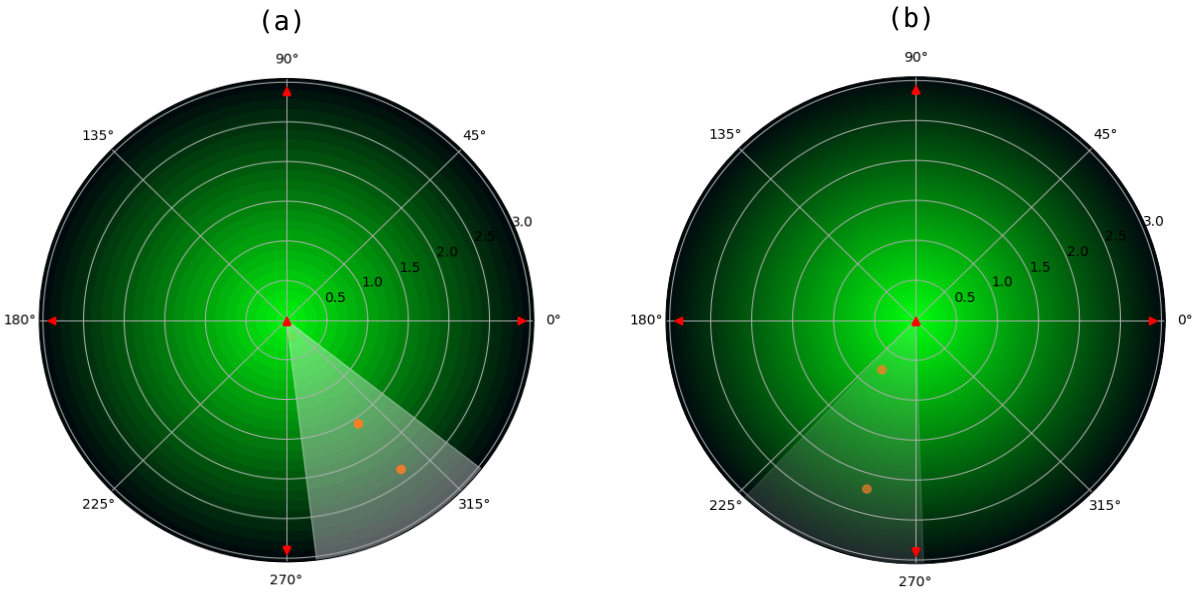
$$B = \frac{255}{85} = 3 \qquad D = \frac{R}{T} = \frac{0.52}{85} = 0.00611\ldots$$

Abaixo na figura 2 é possível visualizar o impacto na suavidade que aumento do número de círculos pode causar.

Figura 2: Diferença entre gradientes com 25 e 85 círculos

(a) Radar com gradiente composto por 25 círculos;

(b) Radar com gradiente composto por 85 círculos



Como a cor e as dimensões do radar são propriedades estáticas, ou seja, definidos pelo desenvolvedor da aplicação, o uso de equações parametrizadas foi aplicado para obter com maior facilidade os valores numéricos correspondentes ao número de círculos, a diferença entre a dimensão entre os círculos e a variação do brilho. Equacionar foi essencial para perceber e compreender a existência das relações entre o número de círculos (“T”) com a suavidade do efeito e com a espessura de cada circunferência, de forma que os valores não ultrapassassem os limites válidos para brilho e diâmetro do fundo.

5 FERRAMENTAS UTILIZADAS

A linguagem de programação utilizada foi a Python (versão 3) junto à biblioteca matplotlib 3.1, ferramenta utilizada para plotagem e animação de gráficos.

6 COMO EXECUTAR O PROJETO

Após baixar o pacote que contém o código fonte, basta entrar na pasta “src” e abrir o arquivo “main.py” com o Python 3.

O código-fonte já contém trechos adicionais para tentar instalar as bibliotecas necessárias, contudo, caso ocorra algum erro, basta executar o comando 1 como administrador no prompt de comando (no caso de sistemas Windows) ou executar o comando 2 no terminal (se o sistema operacional for Linux ou MAC).

Comando 1: Comando para Windows

```
pip install matplotlib
```

Comando 2: Comando para Linux e Mac OS

```
sudo pip3 install matplotlib
```

6.1 PARÂMETROS

A fim de tornar a execução do radar dinâmica, a execução do programa solicita os valores para o ângulo de abertura do setor, a quantidade total de pontos que aparecerá no plano e o número de graus que o setor se movimentará a cada iteração (parâmetro chamado de velocidade).

No comando 3, é possível ver um exemplo de chamada do programa com abertura de 45° para o setor, 10 pontos no plano, e cada movimento com passo de 1.5° (Quanto maior o passo, mais rápido o movimento do setor).

Comando 3: Exemplo de chamada do programa no Windows

```
python main.py -a 45 -p 10 -v 1.5
```

Comando 4: Exemplo de chamada do programa em sistemas Linux

```
python3 main.py -a 45 -p 10 -v 1.5
```

7 REFERÊNCIAS

[1] Página 2. Acesso em 03/06/2019. Disponível em: http://www.abenge.org.br/cobenge/arquivos/16/artigos/EIT740.pdf

[2] Página 3. Acesso em 03/06/2019. Disponível em: https://midia.atp.usp.br/plc/plc0002/impressos/plc0002_05.pdf

[3] Página 7. Acesso em 03/06/2019. Disponível em: http://www.dpi.inpe.br/gilberto/livro/introd/cap6-cartografia.pdf

[4] Página 5. Acesso em 03/06/2019. Disponível em: https://www.researchgate.net/publication/266069182_Aplicacao_de_Robos_nas_Industrias

[5] Página 14. Acesso em 03/06/2019. Disponível em: http://www.im.ufal.br/evento/bsbm/download/minicurso/analise.pdf

[6] Página 1. Acesso em 03/06/2019. Disponível em: https://www.abcem.org.br/construmetal/2014/downloads/contribuicao-tecnocientifica/Arthur-Ribeiro.pdf

[7] Página 25. Acesso em 03/06/2019. Disponível em: https://repositorio.ufba.br/ri/handle/ri/15339