

python™

Laboratório de Programação Competitiva

Profa. Silvia Brandão

2024.1

Aula de Hoje

Tópicos Avançados em Algoritmos: Geometria Computacional

- Geometria Computacional em Python: Pontos, linhas, polígonos, interseções.
- Manipulação de pontos, linhas e polígonos em Python.
- Resolução de problemas envolvendo interseções em geometria computacional.

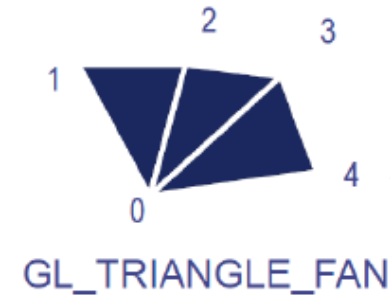
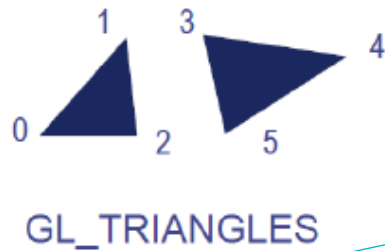
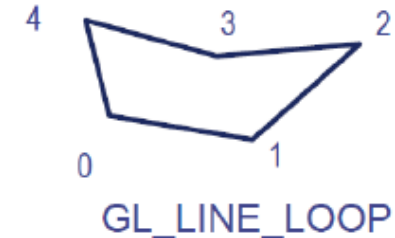
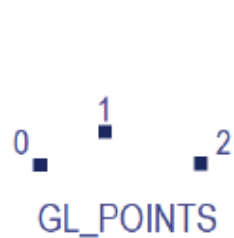
Geometria Computacional

A Geometria Computacional é um ramo da matemática e da ciência da computação que estuda algoritmos e estruturas de dados para resolver problemas geométricos. Seus conceitos fundamentais incluem:

- 1. Objetos Geométricos:** Trabalha com pontos, retas, segmentos, polígonos, círculos e sólidos, analisando suas propriedades e relações.
- 2. Algoritmos Geométricos:** Desenvolve métodos eficientes para realizar operações como interseção, união, triangulação e determinação de distância entre objetos.
- 3. Aplicações:** É amplamente aplicada em áreas como gráficos por computador, robótica, visão computacional, CAD (design assistido por computador) e modelagem 3D.
- 4. Complexidade Computacional:** Estuda a eficiência dos algoritmos, buscando minimizar o tempo e o espaço necessários para resolver problemas.
- 5. Problemas Clássicos:** Inclui questões como o problema do fecho convexo, triangulação de polígonos, e a busca de caminhos mais curtos em grafos.

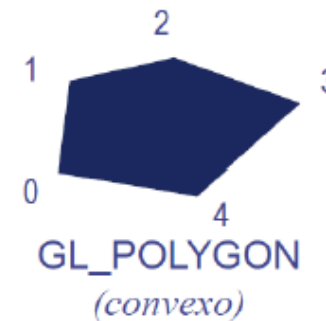
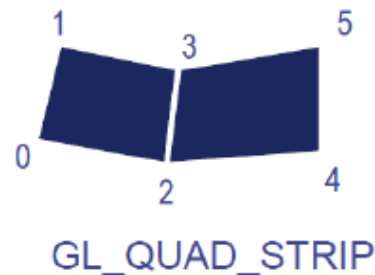
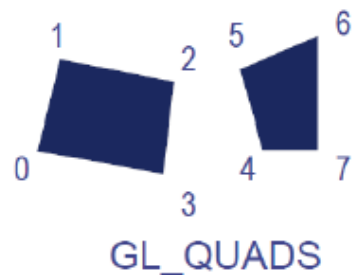
Em resumo, a Geometria Computacional combina matemática e computação para resolver problemas práticos e teóricos relacionados à forma e à estrutura no espaço.

Primitivas gráficas



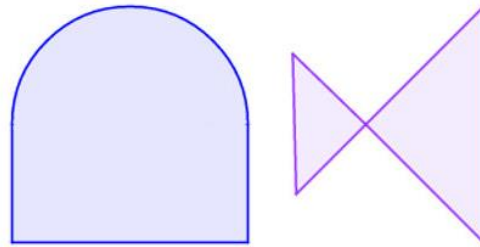
- Uma série de triângulos com um único vértice em comum
- O vértice comum é o primeiro vértice fornecido

- Uma série de triângulos conectados.
- Após o desenho do primeiro triângulo, cada vértice adicional forma um novo triângulo com dois últimos pontos fornecidos

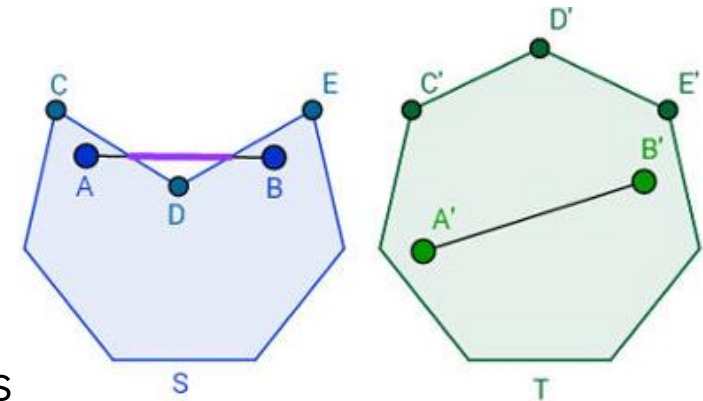


Primitivas Gráficas – Polígonos

- Áreas formadas por várias linhas conectadas
 - Arestas do polígono não podem se cruzar
 - Não pode haver no lugar de um segmento de reta, uma curva qualquer

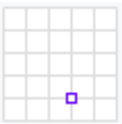
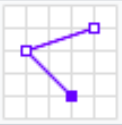
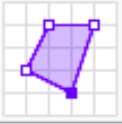
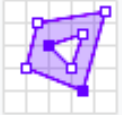


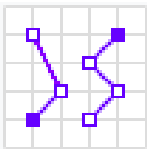
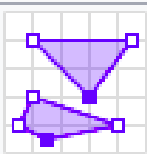
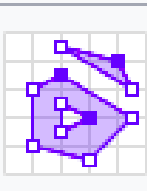
- Devem ser polígonos convexos
- **Polígonos convexos:**
 - Tomando dois pontos A e B dentro de um polígono
 - Se o segmento AB sempre estiver inteiramente no interior do polígono, independentemente da localização dos pontos A e B



Primitivas Gráficas

- Python

Point		<code>POINT (30 10)</code>
LineString		<code>LINESTRING (30 10, 10 30, 40 40)</code>
Polygon		<code>POLYGON ((30 10, 40 40, 20 40, 10 20, 30 10))</code>
		<code>POLYGON ((35 10, 45 45, 15 40, 10 20, 35 10), (20 30, 35 35, 30 20, 20 30))</code>

MultiLineString		<code>MULTILINESTRING ((10 10, 20 20, 10 40), (40 40, 30 30, 40 20, 30 10))</code>
MultiPolygon		<code>MULTIPOLYGON (((30 20, 45 40, 10 40, 30 20)), ((15 5, 40 10, 10 20, 5 10, 15 5)))</code>
		<code>MULTIPOLYGON (((40 40, 20 45, 45 30, 40 40)), ((20 35, 10 30, 10 10, 30 5, 45 20, 20 35), (30 20, 20 15, 20 25, 30 20)))</code>

Primitivas Gráficas – Python

pip install shapely

```
[10] from shapely import Point, LineString, Polygon  
     Point(5.2, 52.1)
```

```
▶ LineString([(0, 0), (1, 2)])
```

```
[14] Polygon([(0, 0), (0, 1), (1, 1), (1, 0)])
```

```
▶ coords = ((0., 0.), (0., 1.), (1., 1.), (1., 0.), (0., 0.))  
  polygon = Polygon(coords)  
  print(f'Área = {polygon.area}')  
  print(f'Perímetro = {polygon.length}')
```

- Biblioteca do Python mais comumente usada para manipulação de pontos e operações geométricas básicas - **shapely**
- Criação de Linhas entre dois pontos - **LineString([(0, 0), (1, 1)])**
- Interseção de Polígonos poly1 e poly2 - **poly1.intersection(poly2)**
- Coordenadas de um ponto p - **p = Point(2, 3); p.coords**
- Cálculo da Área de um Polígono - **poly.area**
- Verificação de Interseção de duas linhas - **line1.intersects(line2)**
- União de dois polígonos poly1 e poly2 - **poly1.union(poly2)**
- Cálculo da distância entre dois pontos point1 e point2 - **point1.distance(point2)**
- Verificação de Contenção (se um ponto p está dentro de um polígono poly) - **poly.contains(p)**
- Simplificação da geometria de um polígono - **poly.simplify(tolerance)**

Clique no link:

https://colab.research.google.com/drive/1kVYyU95eS1y_THBCEtlig_aUUFtyr3tM?usp=sharing

Geometria Computacional

Diagrama de Voronoi:

- ferramenta matemática utilizada para dividir um espaço em regiões, de forma que cada região contenha um ponto central específico;
- técnica desenvolvida pelo matemático russo Georgy Voronoi, no final do século XIX;
- amplamente aplicada em diversas áreas, incluindo a engenharia.

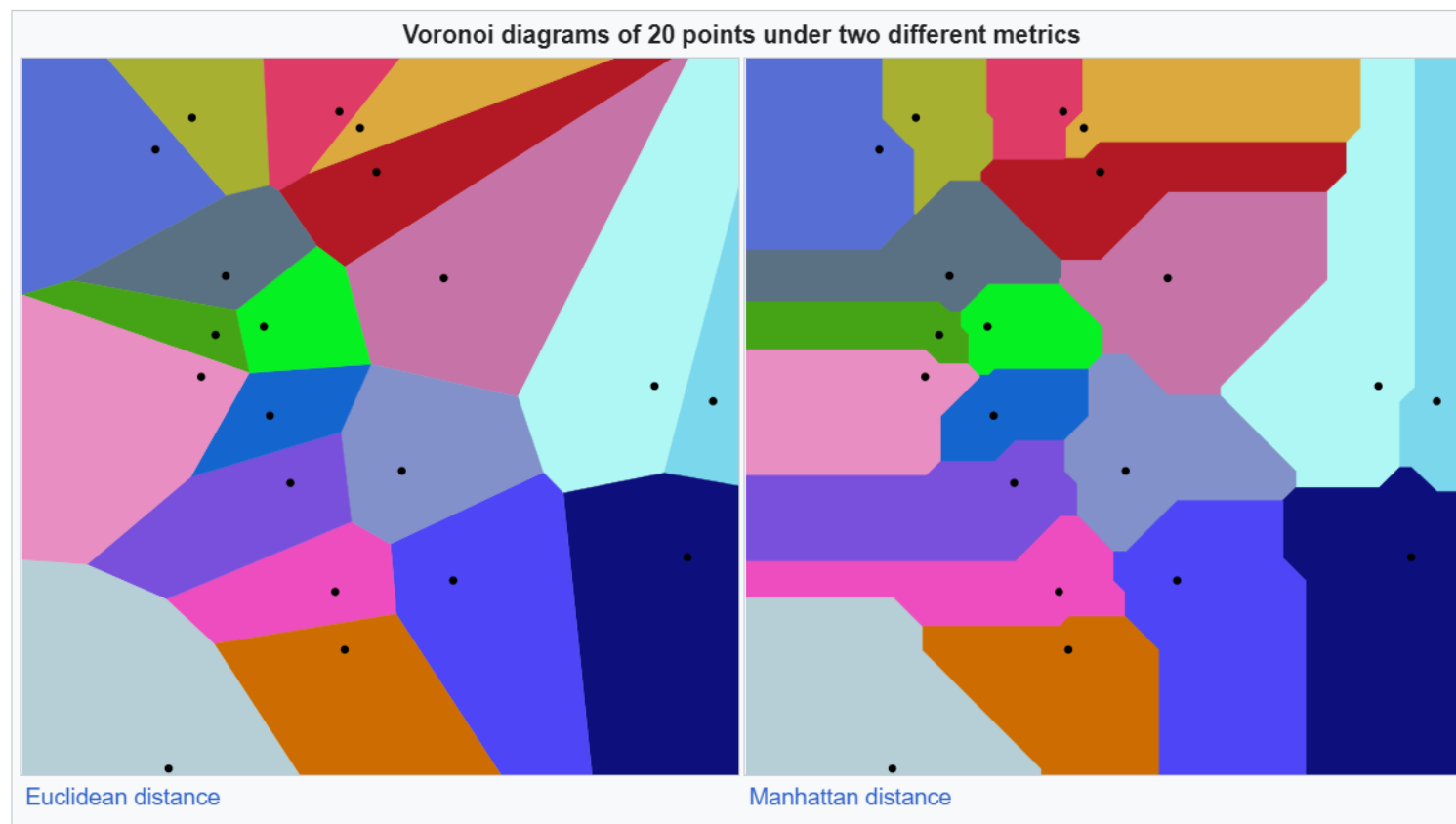


Diagrama de Voronoi

Exemplos de aplicação :

- 1. Análise de tráfego:** o diagrama de Voronoi pode ser utilizado para dividir uma área em regiões de influência de cada ponto de tráfego, auxiliando no planejamento de rotas e na análise de congestionamentos;
- 2. Planejamento urbano:** o diagrama de Voronoi pode ser utilizado para determinar a área de influência de cada equipamento urbano, como escolas, hospitais e postos de saúde, facilitando o planejamento e a distribuição desses recursos.
- 3. Mecânica de materiais granulares**

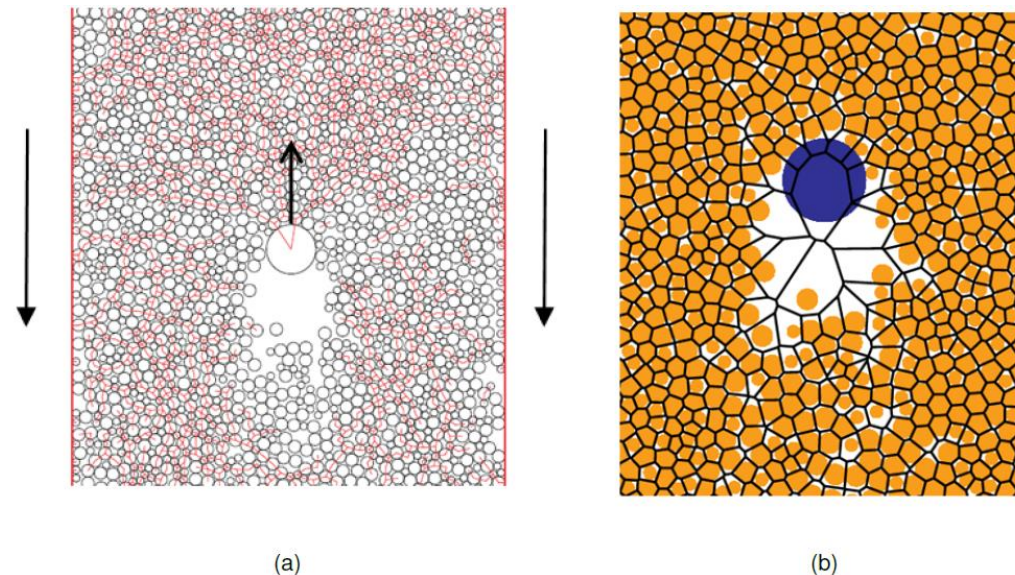
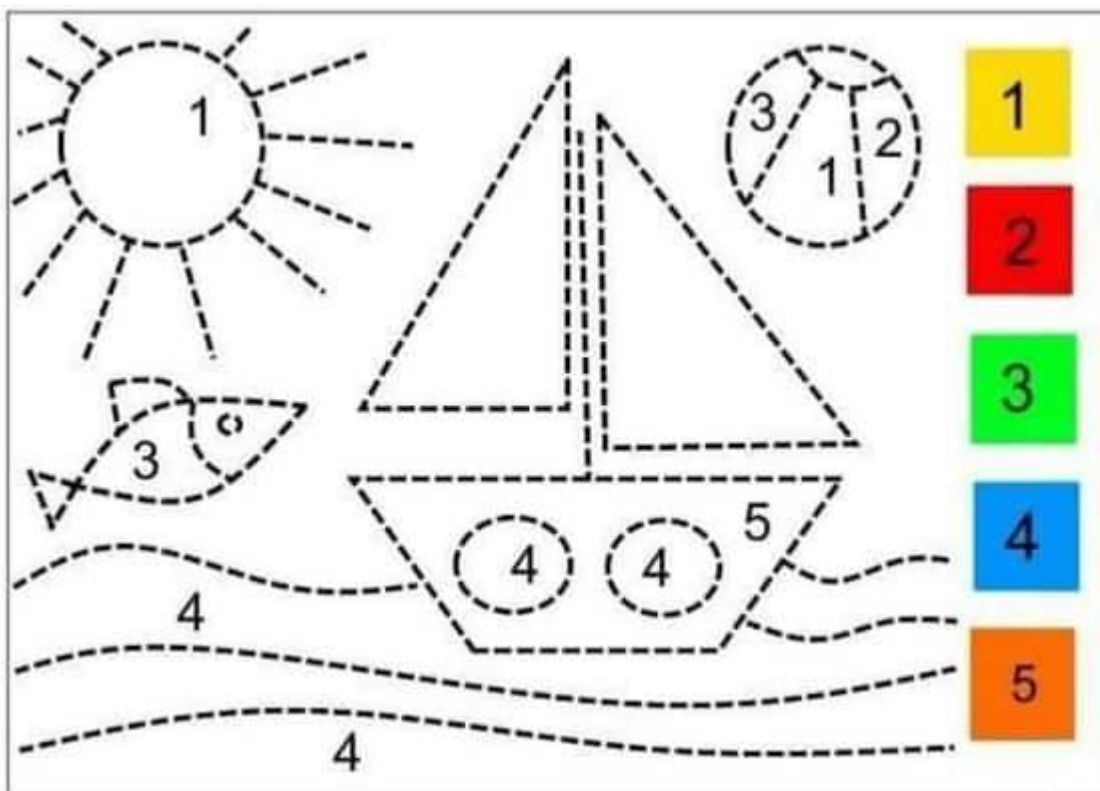


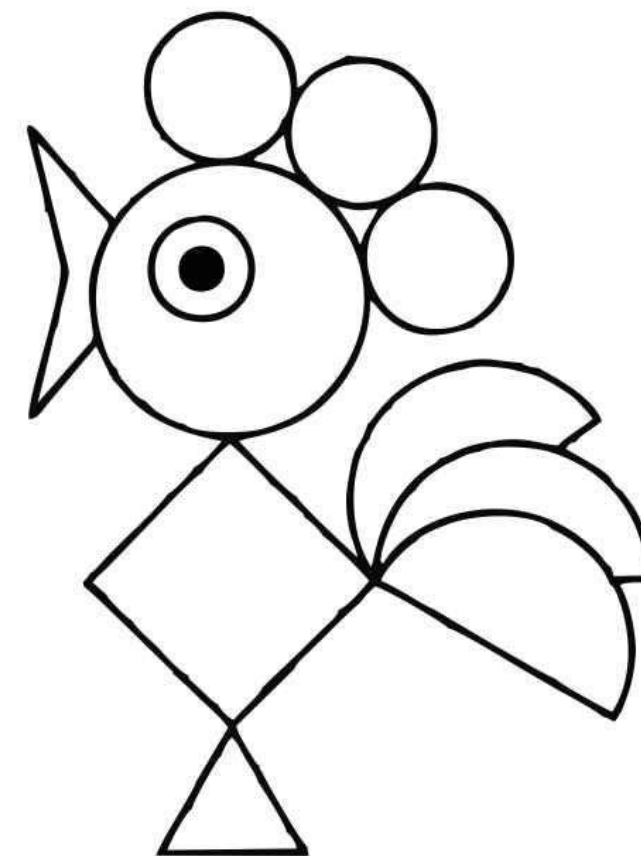
Figura – Sistema granular bidimensional e bidisperso, perturbado por um intruso. Em (a) A circunferência maior representa o intruso, o espaço atrás do intruso é a cavidade, as linhas vermelhas representam as cadeias de forças do sistema, as setas externas indicam o movimento da bandeja e a seta sobre o intruso, apesar de este ser fixo, indica o seu “deslocamento relativo” com referencial na caixa . Em (b) temos, em laranja os grãos, em azul o intruso e a tesselação de Voronoi de uma pequena região próxima ao intruso. Obs: as duas figuras retratam o sistema em momentos diferentes.

Exercícios:módulo *shapely*

9. Implemente o desenho do galo usando apenas as primitivas gráficas.
10. Implemente o desenho do quadro do barquinho e faça a coloração da paisagem



Galo



Entrega pelo portefólio (notebook),
no Google Colab.

Vale nota

Referências

- CORMEN, Thomas H. et al. **Introduction to algorithms**. MIT press, 2009.
- **CLUBE DO QGIS**. <https://clubedogis.com.br/blog/qgis-o-que-e-o-que-faz-e-para-que-serve/>
 - O QGIS fornece ferramentas diferentes para manipular dados espaciais como **visualização, edição e análise**. Além disso, é possível fazer análises espaciais e temporais, acessar banco de dados, utilizar funções conectadas com a internet, visualização 3D dos mapas e, ainda, realizar análises multicritério.
- **Transforme seu VScode em uma ferramenta poderosa para Dados Geográficos**. <https://www.dio.me/articles/transforme-seu-vscode-em-uma-ferramenta-poderosa-para-dados-geograficos>
- **Geometria do Projeto Computacional**. https://primer.dynamobim.org/pt-br/05_Geometry-for-Computational-Design/5_geometry-for-computational-design.html. Não se preocupe com a teoria do Dynamo Primer. É um guia abrangente para a programação visual no Autodesk Dynamo. E, que não será usado por nós.
- IMPA. **Introdução a Geometria Computacional**. https://impa.br/wp-content/uploads/2017/04/18_CBM_91_06.pdf
- **Geometry**. <https://shapely.readthedocs.io/en/stable/geometry.html>