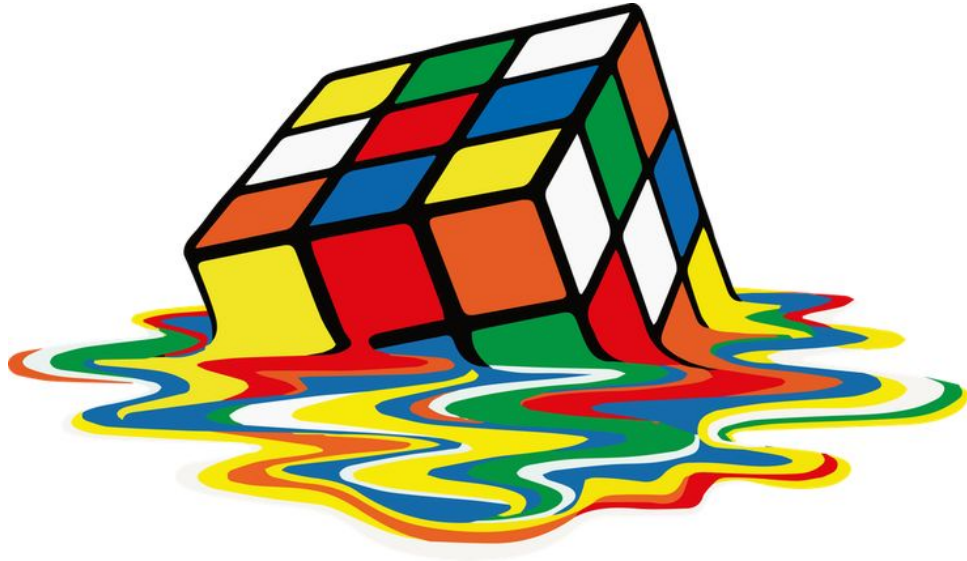


Introdução à Modelagem Matemática – SME0241

Primitivas Geométricas

Afonso Paiva
ICMC-USP

20 de setembro de 2024

Operações com Vetores

- soma vetorial: $sum(\mathrm{x}, \mathrm{y}) = \mathrm{x} + \mathrm{y}$
- multiplicação por escalar: $mult(\lambda, \mathrm{x}) = \lambda \mathrm{x}$
- produto escalar: $dot(\mathrm{x}, \mathrm{y}) = \mathrm{x} \cdot \mathrm{y} = x_1 y_1 + \cdots + x_n y_n$
- norma: $norm(\mathrm{x}) = ||\mathrm{x}|| = \sqrt{\mathrm{x} \cdot \mathrm{x}} = \sqrt{dot(\mathrm{x}, \mathrm{x})}$

Operações com Vetores

- soma vetorial: $sum(\mathrm{x}, \mathrm{y}) = \mathrm{x} + \mathrm{y}$
- multiplicação por escalar: $mult(\lambda, \mathrm{x}) = \lambda \mathrm{x}$
- produto escalar: $dot(\mathrm{x}, \mathrm{y}) = \mathrm{x} \cdot \mathrm{y} = x_1 y_1 + \cdots + x_n y_n$
- norma: $norm(\mathrm{x}) = ||\mathrm{x}|| = \sqrt{\mathrm{x} \cdot \mathrm{x}} = \sqrt{dot(\mathrm{x}, \mathrm{x})}$

Distancias e Ângulos

- distância: $dist(\mathrm{x}, \mathrm{y}) = ||\mathrm{x} - \mathrm{y}|| = norm(\mathrm{x} - \mathrm{y})$
- ângulo: $angle(\mathrm{x}, \mathrm{y}) = \arccos\left(\frac{\mathrm{x} \cdot \mathrm{y}}{||\mathrm{x}||||\mathrm{y}||}\right)$

Operações com Vetores

- soma vetorial: $sum(x, y) = x + y$
- multiplicação por escalar: $mult(\lambda, x) = \lambda x$
- produto escalar: $dot(x, y) = x \cdot y = x_1 y_1 + \cdots + x_n y_n$
- norma: $norm(x) = ||x|| = \sqrt{x \cdot x} = \sqrt{dot(x, x)}$

Distancias e Ângulos

- distância: $dist(x, y) = ||x - y|| = norm(x - y)$
- ângulo: $angle(x, y) = \arccos\left(\frac{x \cdot y}{||x||||y||}\right)$

A primitiva acima é motivada pela identidade

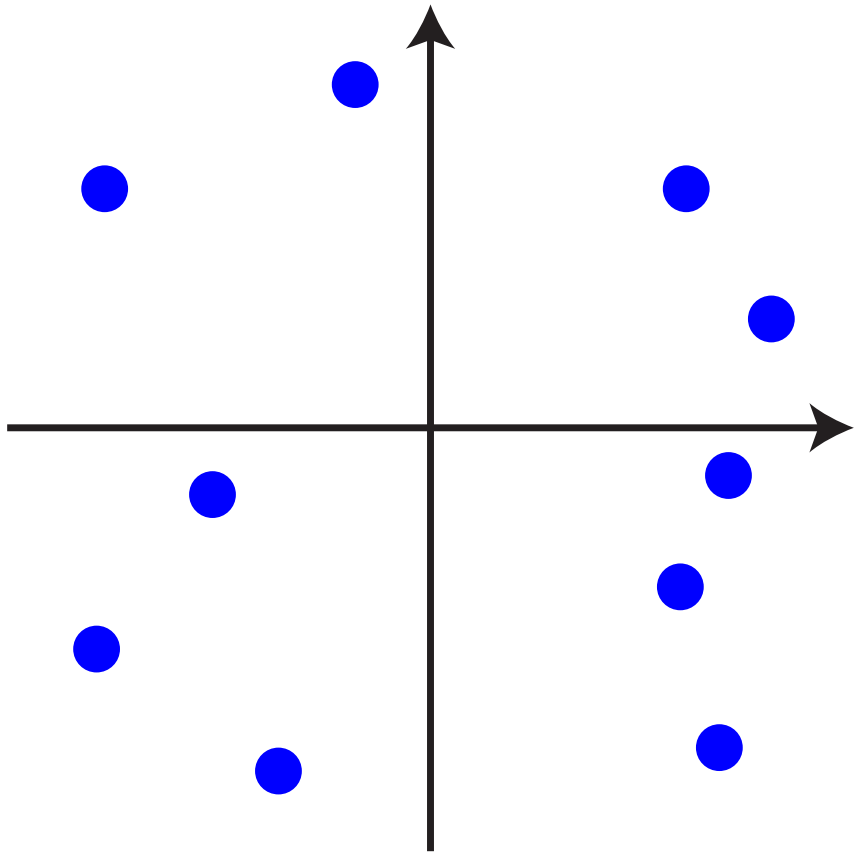
$$x \cdot y = \cos(\theta)||x||||y||\,,$$

onde $\theta \in [0, \pi]$ é o ângulo formado por x e y.

Ordenação Polar

Problema

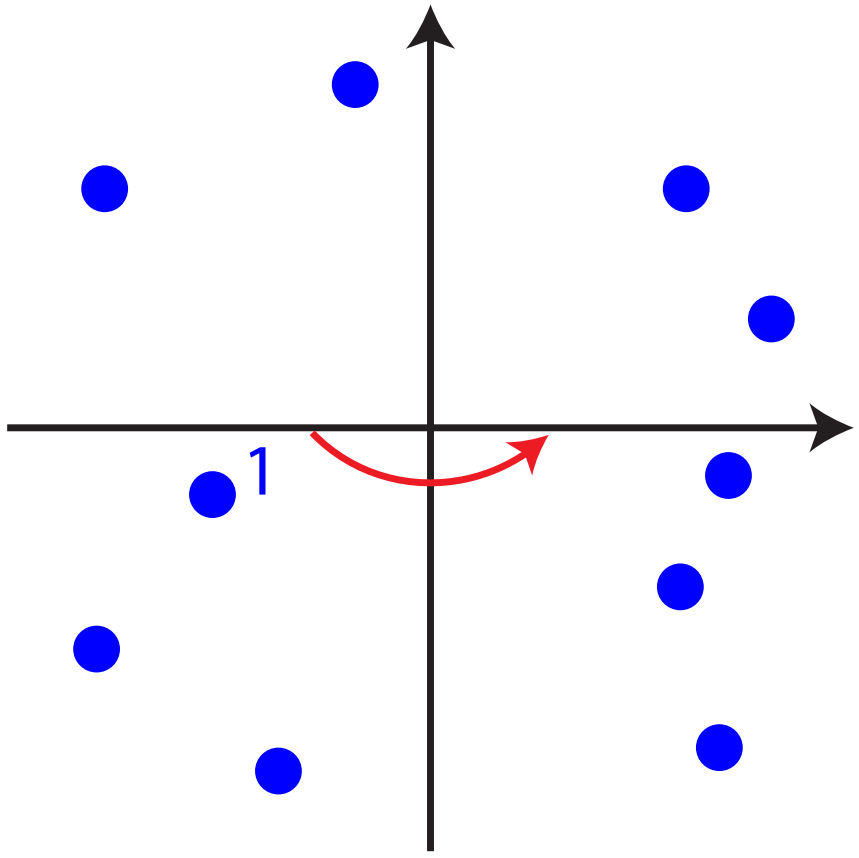
Dados os pontos (vetores) $v_1, v_2, \ldots, v_n \in \mathbb{R}^2$, ordená-los angularmente no sentido anti-horário.



Ordenação Polar

Problema

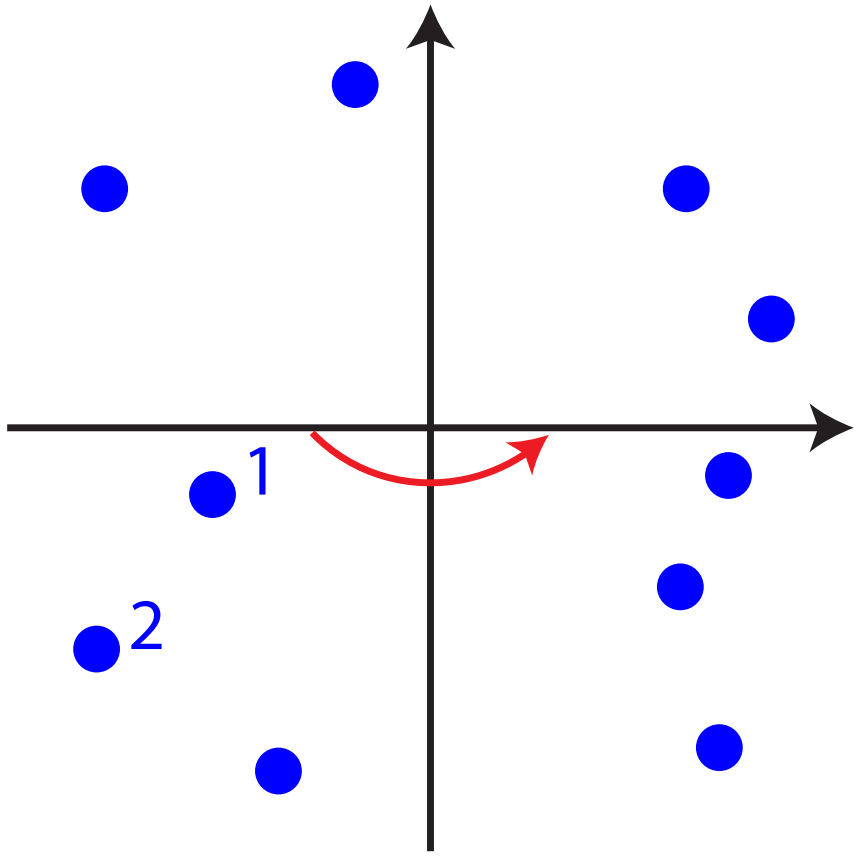
Dados os pontos (vetores) $v_1, v_2, \ldots, v_n \in \mathbb{R}^2$, ordená-los angularmente no sentido anti-horário.



Ordenação Polar

Problema

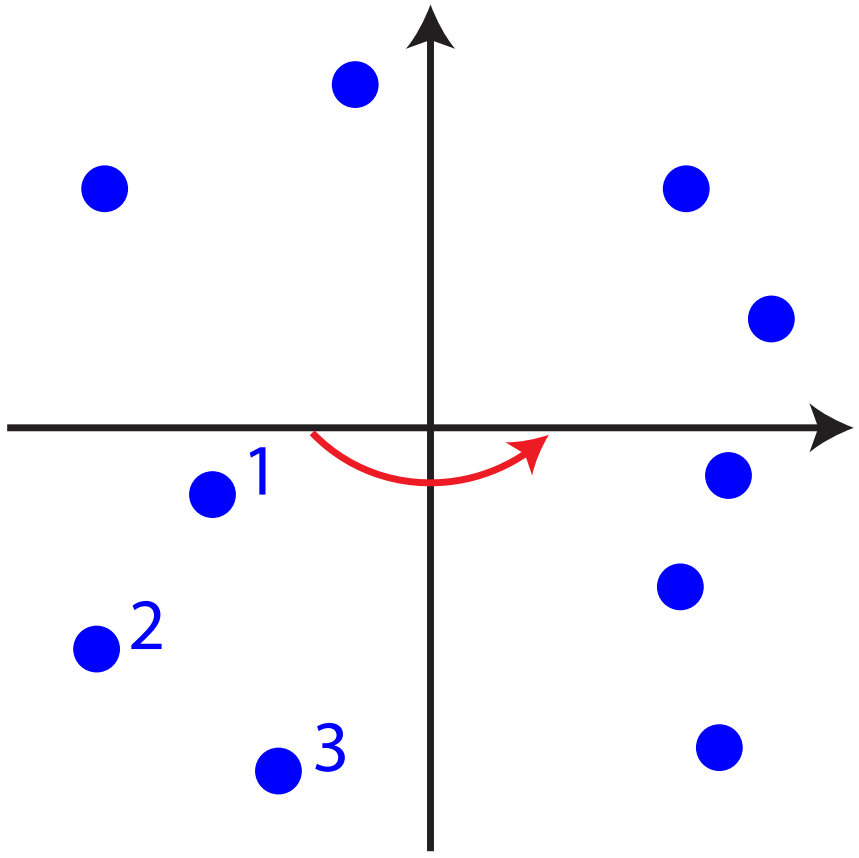
Dados os pontos (vetores) $v_1, v_2, \ldots, v_n \in \mathbb{R}^2$, ordená-los angularmente no sentido anti-horário.



Ordenação Polar

Problema

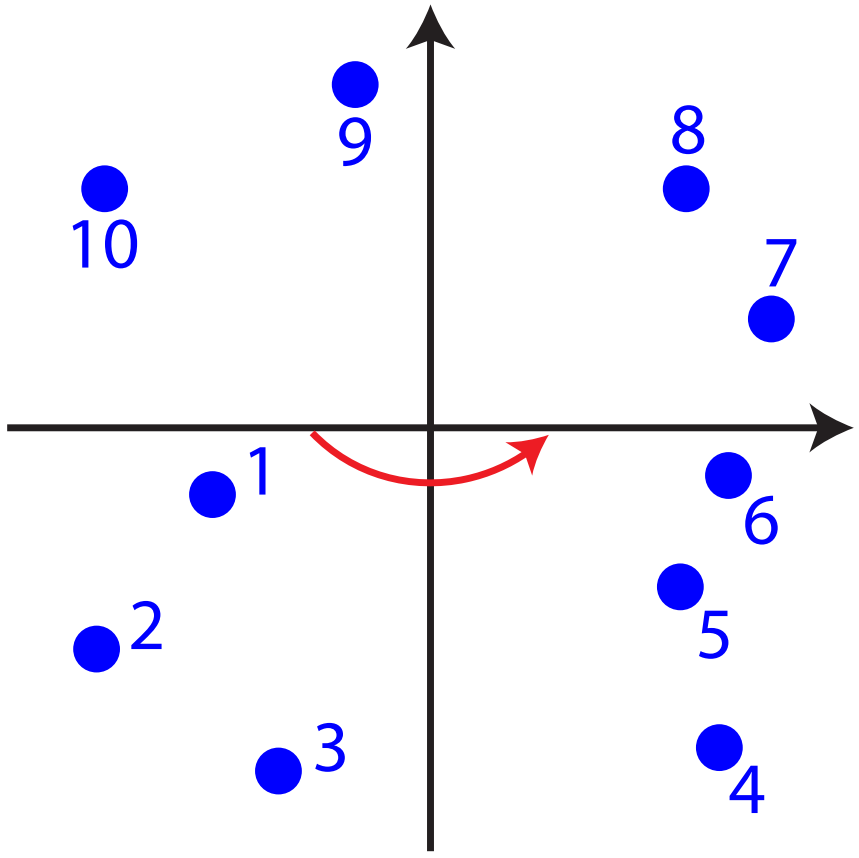
Dados os pontos (vetores) $v_1, v_2, \ldots, v_n \in \mathbb{R}^2$, ordená-los angularmente no sentido anti-horário.



Ordenação Polar

Problema

Dados os pontos (vetores) $v_1, v_2, \ldots, v_n \in \mathbb{R}^2$, ordená-los angularmente no sentido anti-horário.



Ângulo Orientado

Dado um vetor $x = (x_1, x_2) \neq \overline{0}$ no $\mathbb{R}^2$ e $u = (1, 0)$, o ângulo orientado definido por x é dado por:

$$angle(x) = \begin{cases} angle(u, x) & , \text{se } x_2 \geq 0 \\ -angle(u, x) & , \text{se } x_2 < 0 \end{cases},$$

onde $angle(u, x) = \arccos\left(\frac{x_1}{||x||}\right)$ definido em $(-\pi, \pi]$.

Ângulo Orientado

Dado um vetor $\mathrm{x} = (x_1, x_2) \neq \overline{0}$ no $\mathbb{R}^2$ e $\mathrm{u} = (1, 0)$, o ângulo orientado definido por x é dado por:

$$angle(\mathrm{x}) = \begin{cases} angle(\mathrm{u}, \mathrm{x}) & , \text{se } x_2 \geq 0 \\ -angle(\mathrm{u}, \mathrm{x}) & , \text{se } x_2 < 0 \end{cases},$$

onde $angle(\mathrm{u}, \mathrm{x}) = \arccos\left(\frac{x_1}{||\mathrm{x}||}\right)$ definido em $(-\pi, \pi]$.

O problema dessa função é que ela é transcendental (computacionalmente cara), o que foge das nossas operações básicas.

Ângulo Orientado

Dado um vetor $\mathrm{x} = (x_1, x_2) \neq \overline{0}$ no $\mathbb{R}^2$ e $\mathrm{u} = (1, 0)$, o ângulo orientado definido por x é dado por:

$$angle(\mathrm{x}) = \begin{cases} angle(\mathrm{u}, \mathrm{x}) & , \text{se } x_2 \geq 0 \\ -angle(\mathrm{u}, \mathrm{x}) & , \text{se } x_2 < 0 \end{cases},$$

onde $angle(\mathrm{u}, \mathrm{x}) = \arccos\left(\frac{x_1}{||\mathrm{x}||}\right)$ definido em $(-\pi, \pi]$.

O problema dessa função é que ela é transcendental (computacionalmente cara), o que foge das nossas operações básicas.

Note, precisamos apenas comparar ângulos!

Pseudo-ângulos

Afim de comparar ângulos, precisamos criar uma função que seja monótona em relação ao ângulo. Por exemplo,

$$f(\theta) = 1 - \cos(\theta)\,, \quad \theta \in [0, \pi]$$

Pseudo-ângulos

Afim de comparar ângulos, precisamos criar uma função que seja monótona em relação ao ângulo. Por exemplo,

$$f(\theta) = 1 - \cos(\theta), \quad \theta \in [0, \pi]$$

Essa função define uma primitiva chamada pseudo-ângulo definida como:

$$f(x, y) = 1 - \frac{x \cdot y}{||x||||y||}$$

A função acima é monótona crescente em relação a θ e toma valores no intervalo $[0, 2]$.

Pseudo-ângulos

Afim de comparar ângulos, precisamos criar uma função que seja monótona em relação ao ângulo. Por exemplo,

$$f(\theta) = 1 - \cos(\theta), \quad \theta \in [0, \pi]$$

Essa função define uma primitiva chamada pseudo-ângulo definida como:

$$f(x, y) = 1 - \frac{x \cdot y}{||x||||y||}$$

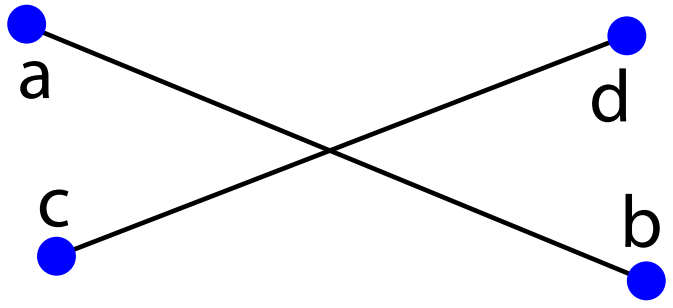
A função acima é monótona crescente em relação a θ e toma valores no intervalo $[0, 2]$.

Exercício 1

Dados n pontos em $[-1, 1] \times [0, 2]$, faça um programa em Python que ordene angularmente esses pontos. Dica: use o comando `sort`.

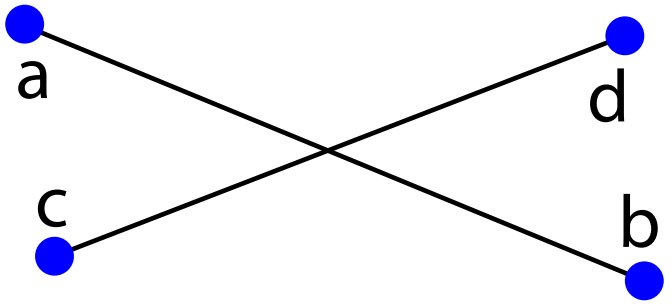
Interseção de Segmentos

Problema: detectar quando dois segmentos no plano tem interseção.



Interseção de Segmentos

Problema: detectar quando dois segmentos no plano tem interseção.

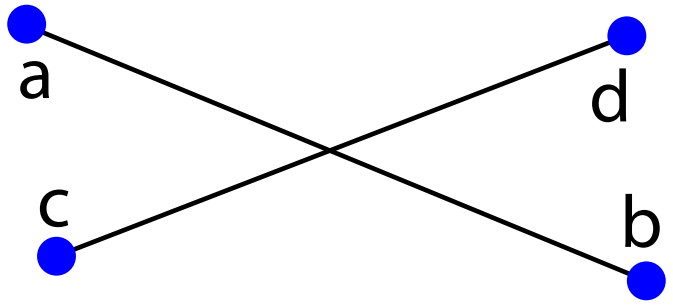


Produto Vetorial

$$(x_1, x_2, x_3) \times (y_1, y_2, y_3) = \begin{bmatrix} i & j & k \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix}$$

Interseção de Segmentos

Problema: detectar quando dois segmentos no plano tem interseção.

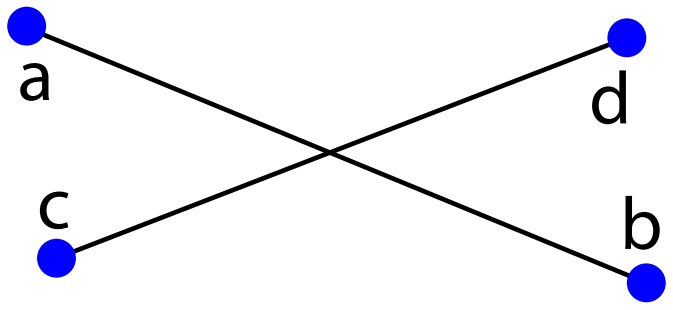


Produto Vetorial

$$
\begin{aligned}
(x_1, x_2, x_3) \times (y_1, y_2, y_3) &= \begin{bmatrix} i & j & k \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix} \\
&= (x_2 y_3 - x_3 y_2, x_3 y_1 - x_1 y_3, x_1 y_2 - x_2 y_1)
\end{aligned}
$$

Interseção de Segmentos

Problema: detectar quando dois segmentos no plano tem interseção.



Produto Vetorial

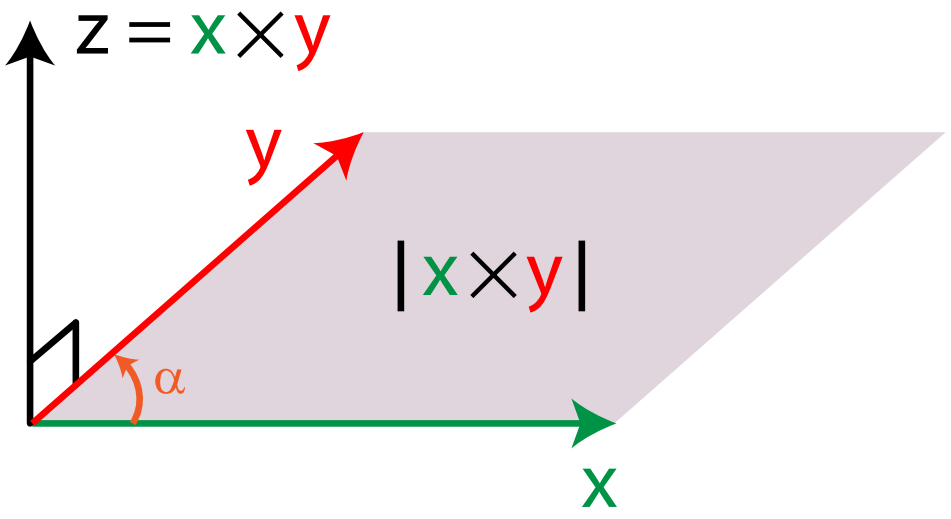
$$
\begin{aligned}
(x_1, x_2, x_3) \times (y_1, y_2, y_3) &= \begin{bmatrix} i & j & k \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix} \\
&= (x_2 y_3 - x_3 y_2, x_3 y_1 - x_1 y_3, x_1 y_2 - x_2 y_1)
\end{aligned}
$$

Tomando a terceira componente nula, podemos definir o produto vetorial em $\mathbb{R}^2$ como uma função escalar $\times : \mathbb{R}^2 \to \mathbb{R}$ dada por:

$$(x_1, x_2) \times (y_1, y_2) = x_1 y_2 - x_2 y_1$$

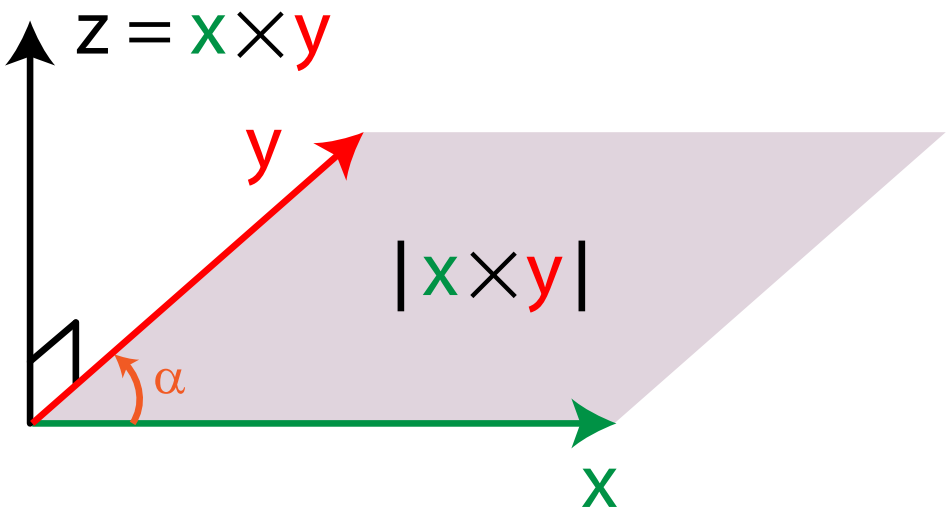
Interseção de Segmentos

Espaço 3D: regra da mão direita

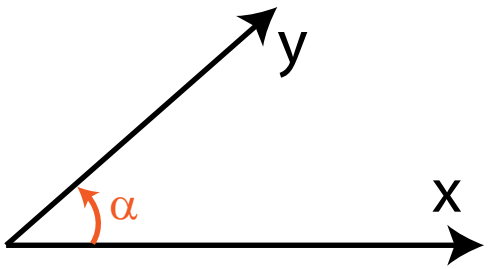


Interseção de Segmentos

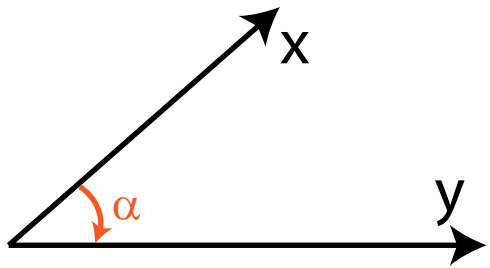
Espaço 3D: regra da mão direita



Espaço 2D: ângulo orientado



$x \times y > 0$: y à esquerda de x



$x \times y < 0$: y à direita de x

Interseção de Segmentos

Solução:

Basta verificar se $[(b-a)\times(c-a)][(b-a)\times(d-a)] < 0$ e $[(d-c)\times(a-c)][(d-c)\times(b-c)] < 0$.

Interseção de Segmentos

Solução:

Basta verificar se $[(b-a)\times(c-a)][(b-a)\times(d-a)] < 0$ e $[(d-c)\times(a-c)][(d-c)\times(b-c)] < 0$.

Exercício 2

Faça um programa em Python que dado dois segmentos retorne `True` se os segmentos possuem interseção ou `False` caso contrário. Desenhe os segmentos de entrada para testar os seus resultados.

Exercício 3

Faça um programa em Python que dado um polígono convexo e um ponto, retorne `True` se o ponto esta no interior do polígono ou `False` caso contrário. Desenhe o polígono e o ponto de entrada para testar os seus resultados.

Coordenadas Baricéntricas

Definição

O ponto v é o baricentro dos pontos $v_1, \ldots, v_n$ com pesos $w_1, \ldots, w_n$ se somente se

$$v = \frac{w_1 v_1 + \cdots + w_n v_n}{w_1 + \cdots + w_n}$$

Os valores w_i são as coordendas baricéntricas de v.

Coordenadas Baricéntricas

Definição

O ponto v é o baricentro dos pontos $v_1, \ldots, v_n$ com pesos $w_1, \ldots, w_n$ se somente se

$$v = \frac{w_1 v_1 + \cdots + w_n v_n}{w_1 + \cdots + w_n}$$

Os valores w_i são as coordendas baricéntricas de v.

Coordenadas Baricéntricas Normalizadas

$$\lambda_i(v) = \frac{w_i(v)}{w_1(v) + \cdots + w_n(v)}$$

Coordenadas Baricéntricas

Definição

O ponto v é o baricentro dos pontos $v_1, \ldots, v_n$ com pesos $w_1, \ldots, w_n$ se somente se

$$v = \frac{w_1 v_1 + \cdots + w_n v_n}{w_1 + \cdots + w_n}$$

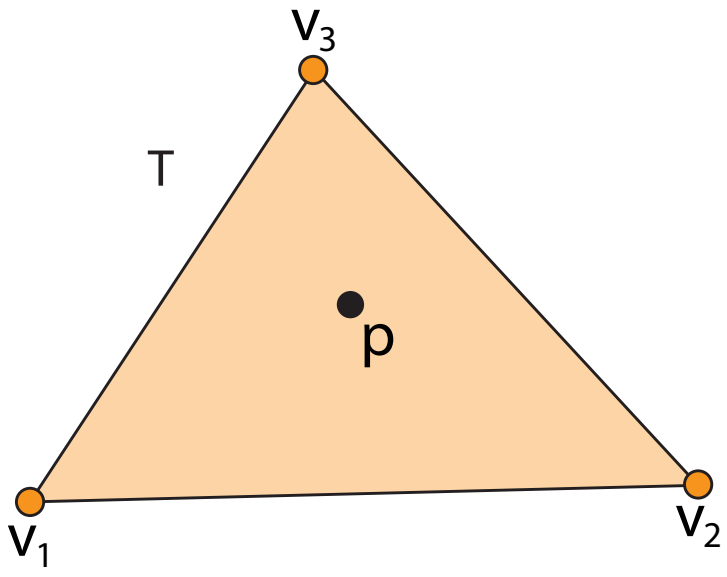
Os valores w_i são as coordendas baricéntricas de v.

Coordenadas Baricéntricas Normalizadas

$$\lambda_i(v) = \frac{w_i(v)}{w_1(v) + \cdots + w_n(v)}$$

Logo, $v = \sum_i \lambda_i v_i$ com $\sum_i \lambda_i = 1$, isto é, uma combinação convexa dos pontos $v_1, \cdots, v_n$.

Coordenadas Baricéntricas no Triângulo



Objetivo: dado x $\in T$, queremos $\lambda_1, \lambda_2, \lambda_3 \geq 0$ tal que:

$$\lambda_1 + \lambda_2 + \lambda_3 = 1\,,$$

e

$$p = \lambda_1 v_1 + \lambda_2 v_2 + \lambda_3 v_3$$

Coordenadas Baricéntricas no Triângulo

Precisamos resolver o sistema linear de ordem 3:

$$\begin{bmatrix} 1 & 1 & 1 \\ v_1^x & v_2^x & v_3^x \\ v_1^y & v_2^y & v_3^y \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} = \begin{bmatrix} 1 \\ p^x \\ p^y \end{bmatrix}$$

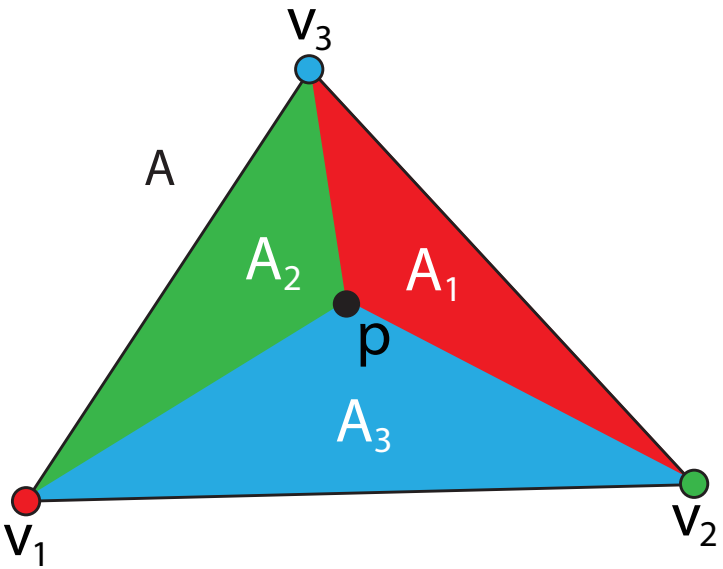
Coordenadas Baricéntricas no Triângulo

Precisamos resolver o sistema linear de ordem 3:

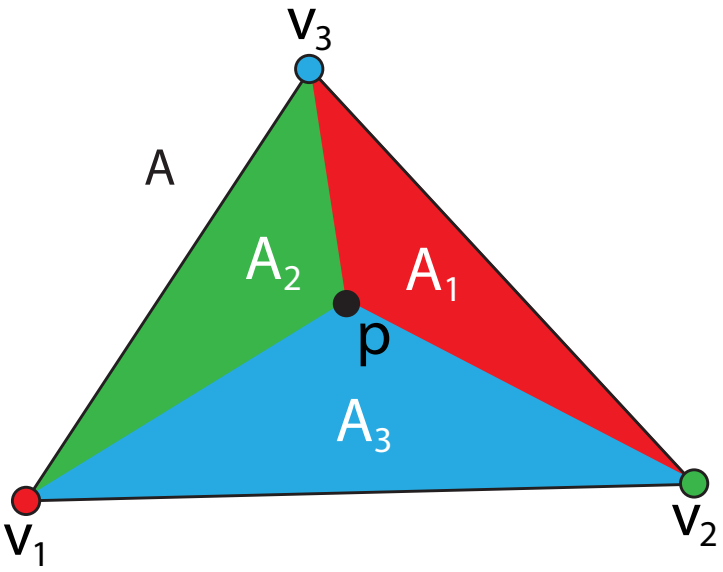
$$\begin{bmatrix} 1 & 1 & 1 \\ v_1^x & v_2^x & v_3^x \\ v_1^y & v_2^y & v_3^y \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} = \begin{bmatrix} 1 \\ p^x \\ p^y \end{bmatrix}$$

Pela Regra de Cramer a solução (única) é

$$\lambda_1 = \frac{A_1}{A}\,, \quad \lambda_2 = \frac{A_2}{A}\,, \quad \lambda_3 = \frac{A_3}{A}\,.$$

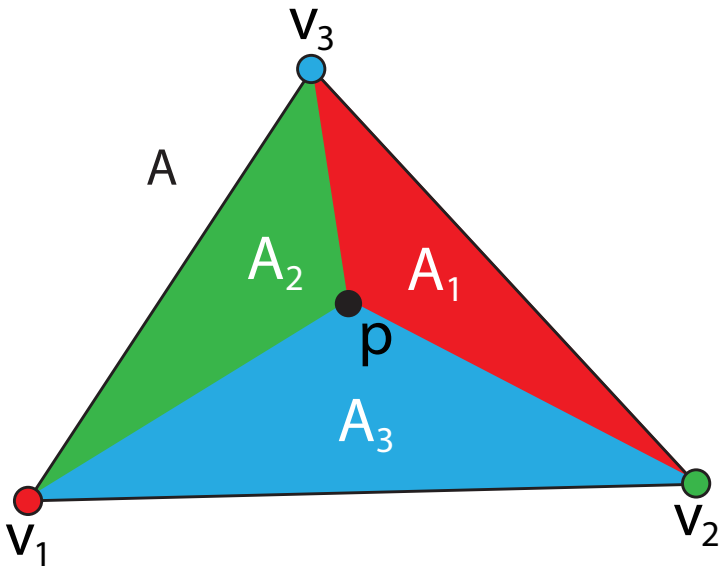


Coordenadas Baricéntricas no Triângulo



Propriedades

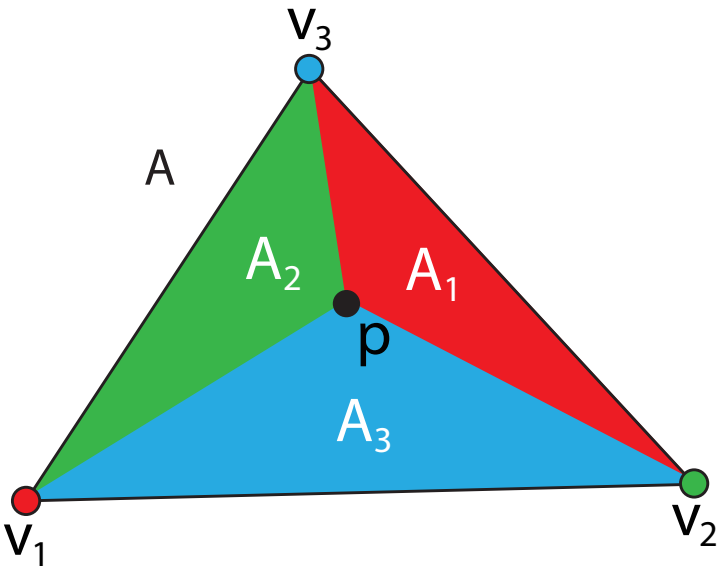
Coordenadas Baricéntricas no Triângulo



Propriedades

- teste de in-out: um ponto está fora do $\triangle$ se $\exists \lambda_i(p) < 0$;

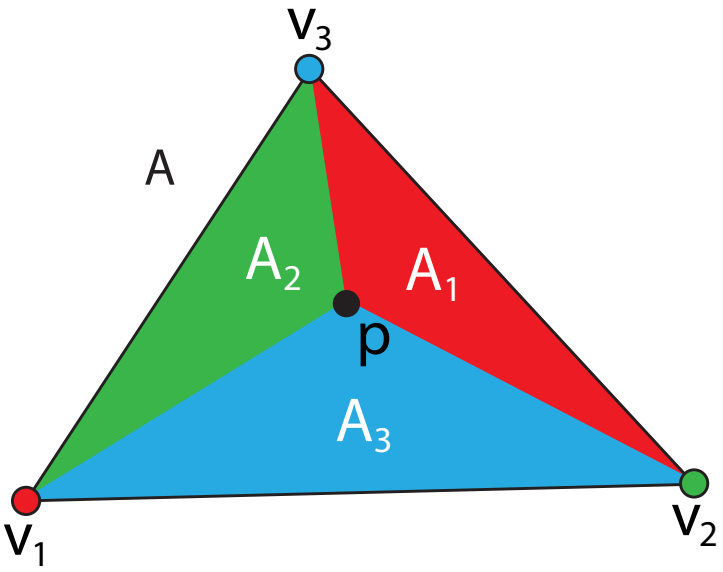
Coordenadas Baricéntricas no Triângulo



Propriedades

- teste de in-out: um ponto está fora do $\triangle$ se $\exists \lambda_i(\mathrm{p}) < 0$;
- propriedade de Lagrange: $\lambda_i(\mathrm{v}_j) = \delta_{ij}$;

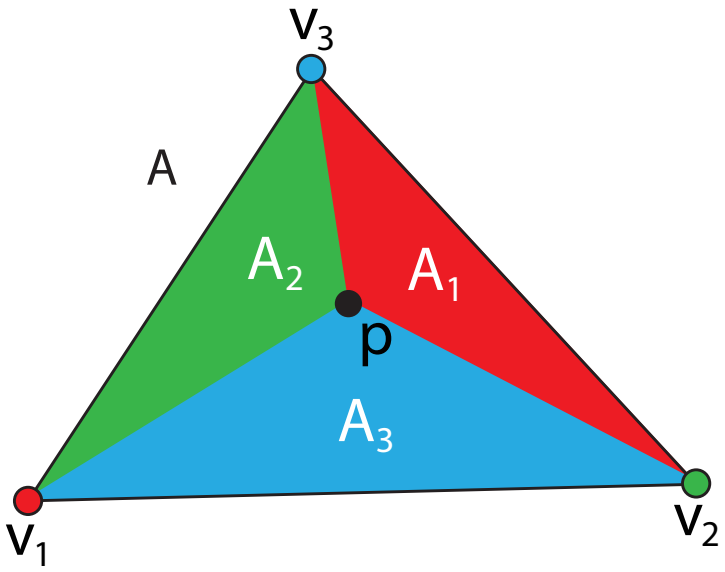
Coordenadas Baricéntricas no Triângulo



Propriedades

- teste de in-out: um ponto está fora do $\triangle$ se $\exists \lambda_i(\mathrm{p}) < 0$;
- propriedade de Lagrange: $\lambda_i(\mathrm{v}_j) = \delta_{ij}$;
- inteporlação linear: $g(\mathrm{p}) = \sum_{i=1}^{3} \lambda_i(\mathrm{p}) f(\mathrm{v}_i)$;

Coordenadas Baricéntricas no Triângulo



Propriedades

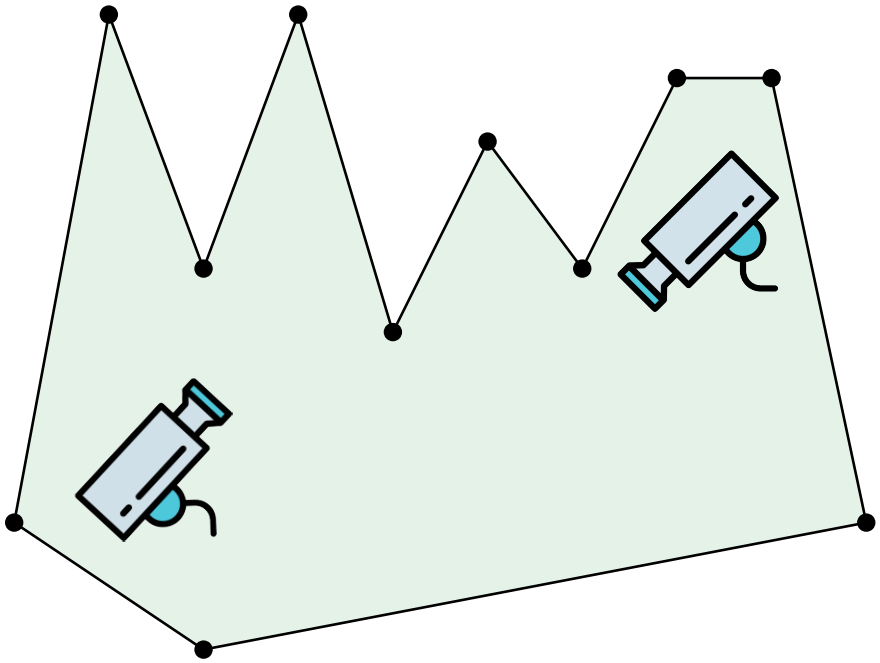
- teste de in-out: um ponto está fora do $\triangle$ se $\exists \lambda_i(\mathrm{p}) < 0$;
- propriedade de Lagrange: $\lambda_i(\mathrm{v}_j) = \delta_{ij}$;
- inteporlação linear: $g(\mathrm{p}) = \sum_{i=1}^{3} \lambda_i(\mathrm{p}) f(\mathrm{v}_i)$;
- precisão linear: se f é linear $\Rightarrow g = f$.

Coordenadas Baricéntricas no Triângulo

Exercício 4

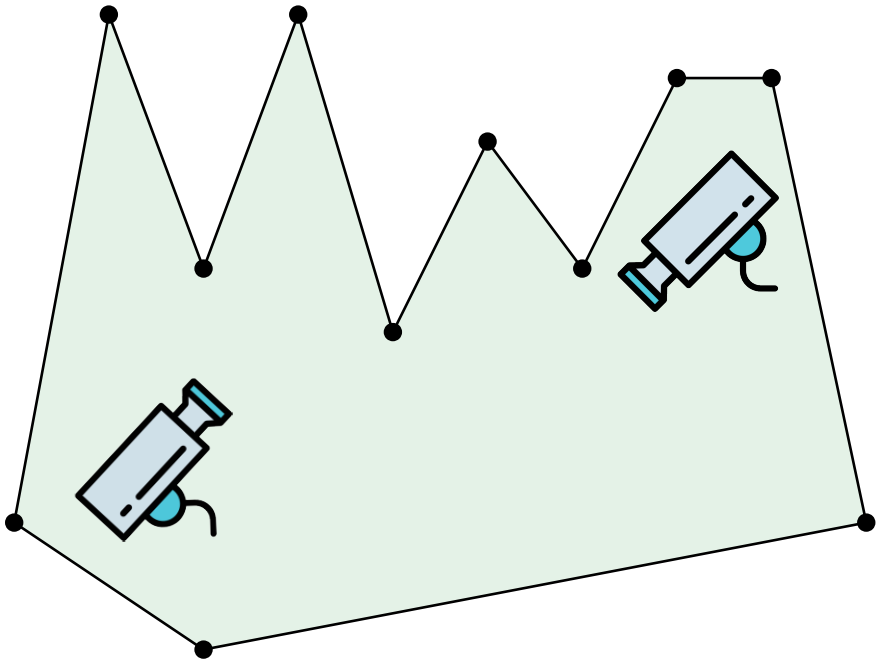
Faça um programa em Python que dado os vértices de um triângulo, calcule as coordenadas baricéntricas de um ponto dado.

Problema da Galeria de Arte



Problema: Dada uma galeria de arte, instale câmeras de tal forma possam ver toda galeria usando o menor número possível de câmeras. As câmeras não podem se mover, mas podem rotacionar 360°.

Problema da Galeria de Arte



Problema: Dada uma galeria de arte, instale câmeras de tal forma possam ver toda galeria usando o menor número possível de câmeras. As câmeras não podem se mover, mas podem rotacionar 360°.

Objetivo: Quantas câmeras vamos precisar e aonde instalaremos elas?

Polígono Simples

Vamos modelar a planta da galeria como um polígono simples.

Polígono Simples

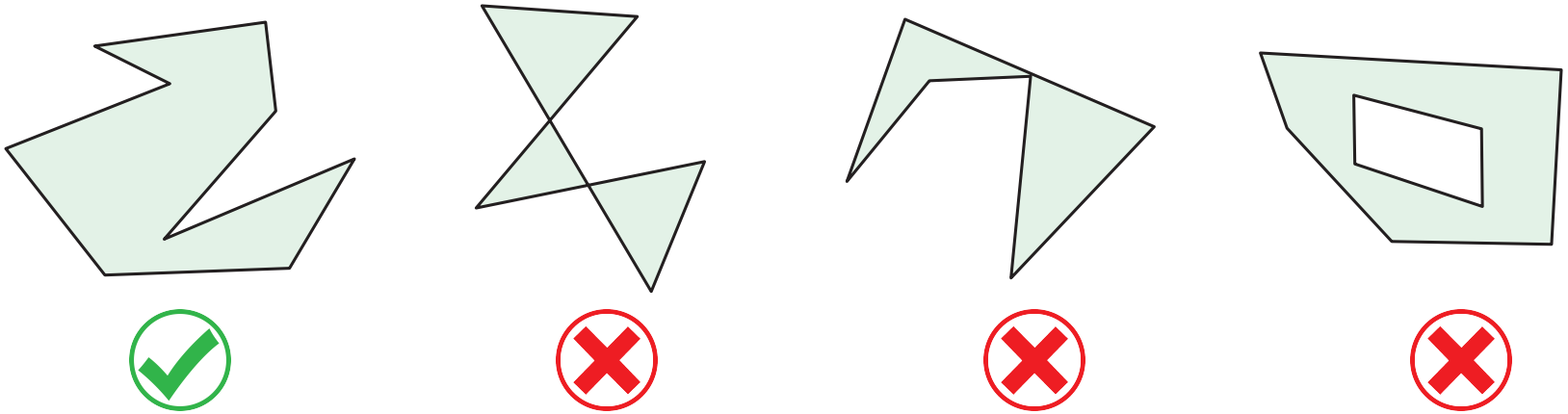
Vamos modelar a planta da galeria como um polígono simples.

- região do plano, limitada por uma coleção de segmentos formando uma curva fechada sem auto-interseção;
- regiões com buracos não serão permitidas;
- apenas dois segmentos incidem em cada vértice.

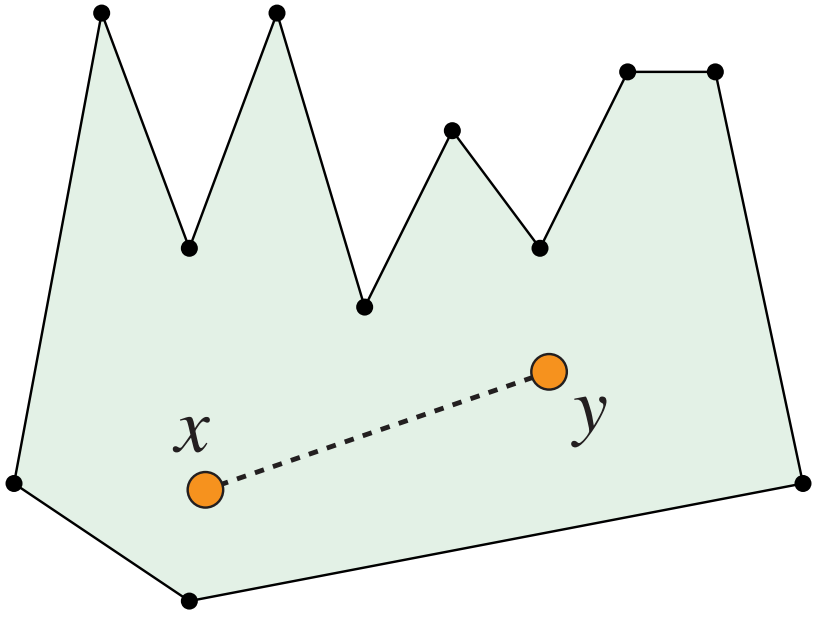
Polígono Simples

Vamos modelar a planta da galeria como um polígono simples.

- região do plano, limitada por uma coleção de segmentos formando uma curva fechada sem auto-interseção;
- regiões com buracos não serão permitidas;
- apenas dois segmentos incidem em cada vértice.

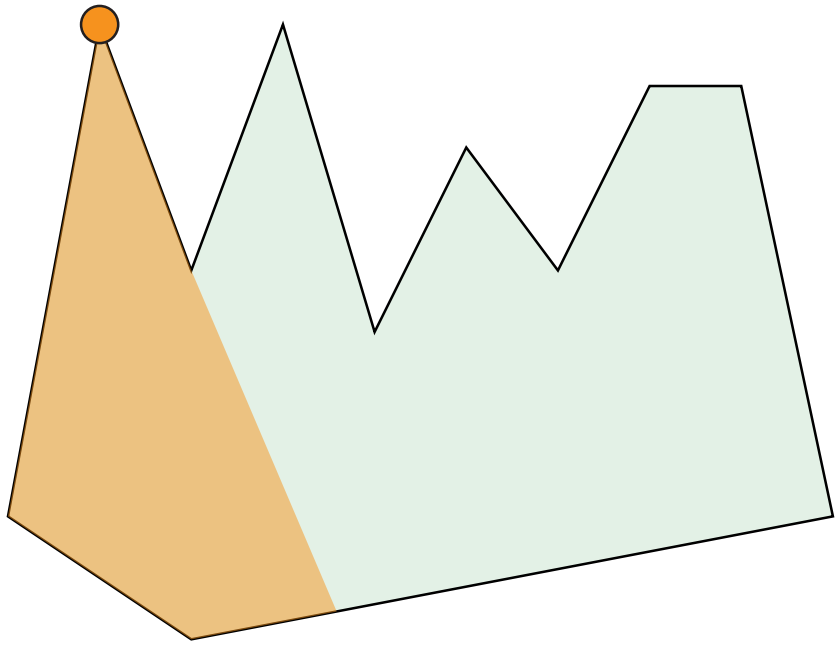


Visibilidade



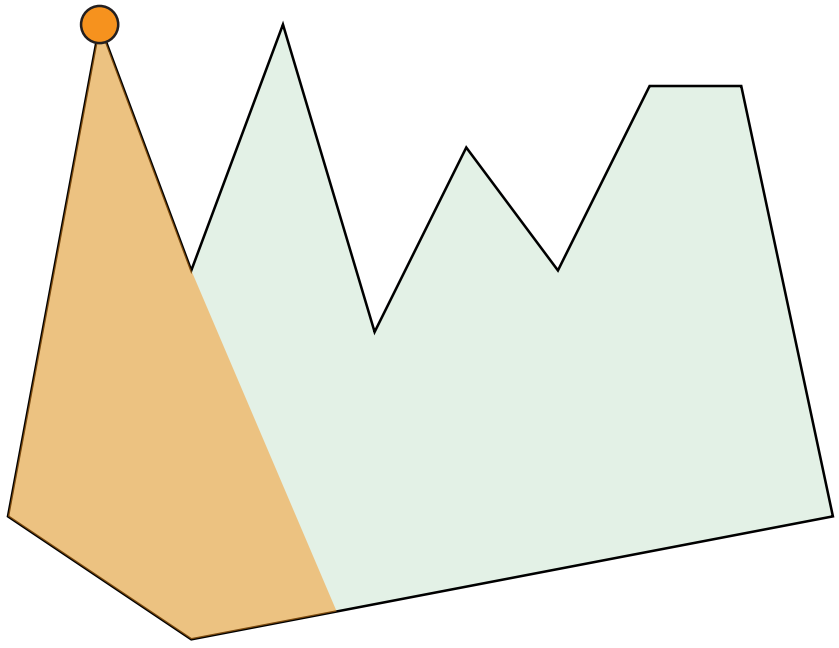
Seja P um polígono simples, dois pontos $x, y \in P$ são mutualmente visíveis, se o segmento $\overline{xy} \subset P$.

Problema da Galeria de Arte – Formalização



Problema: Dado um polígono simples P, encontre o número mínimo de pontos em P capaz de ver todo polígono P.

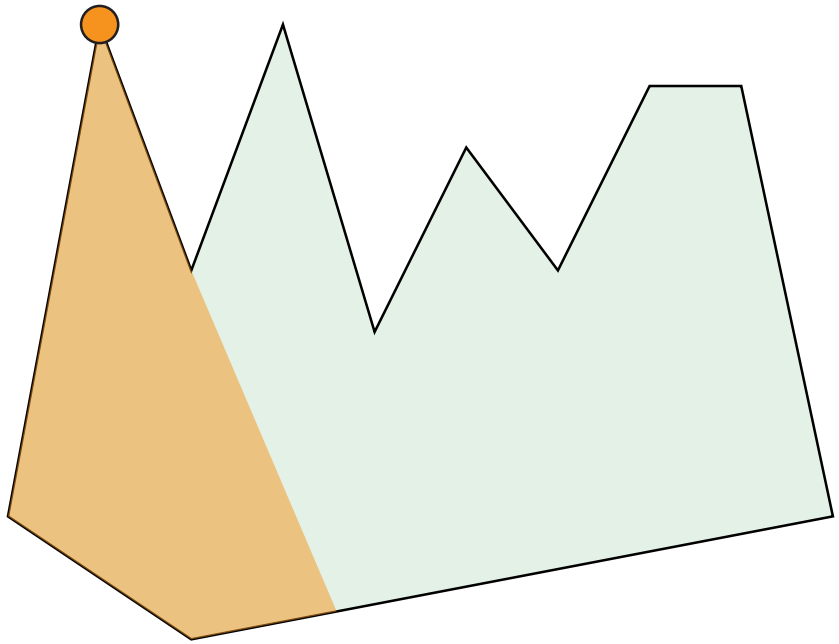
Problema da Galeria de Arte – Formalização



Problema: Dado um polígono simples P, encontre o número mínimo de pontos em P capaz de ver todo polígono P.

Má notícia: esse problema é NP-difícil (tempo exponencial).

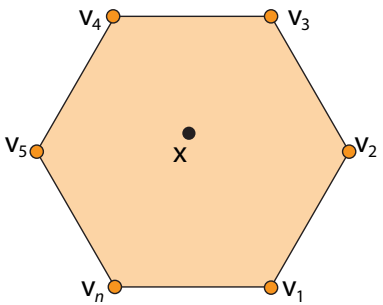
Problema da Galeria de Arte – Formalização



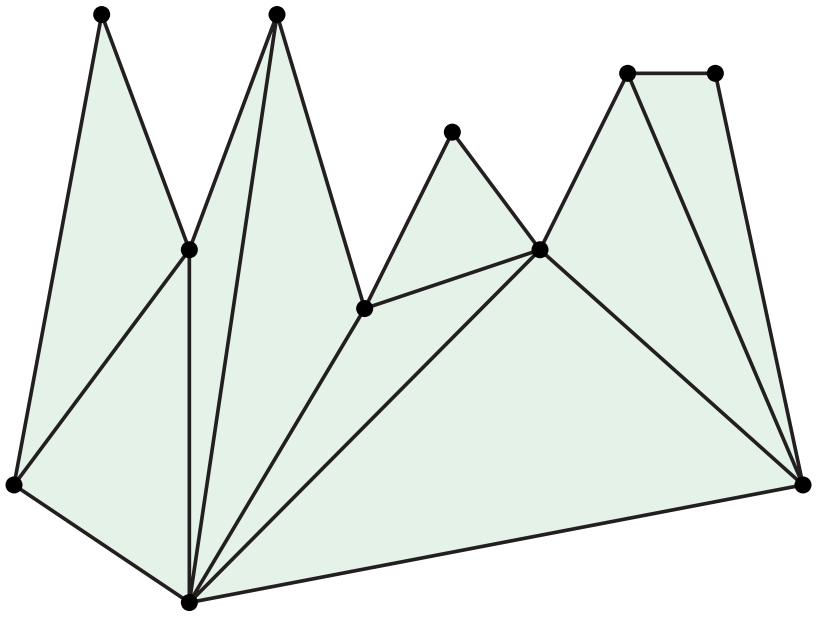
Problema: Dado um polígono simples P, encontre o número mínimo de pontos em P capaz de ver todo polígono P.

Má notícia: esse problema é NP-difícil (tempo exponencial).

Obs.: uma região convexa pode ser vigiada por uma único ponto (câmera).

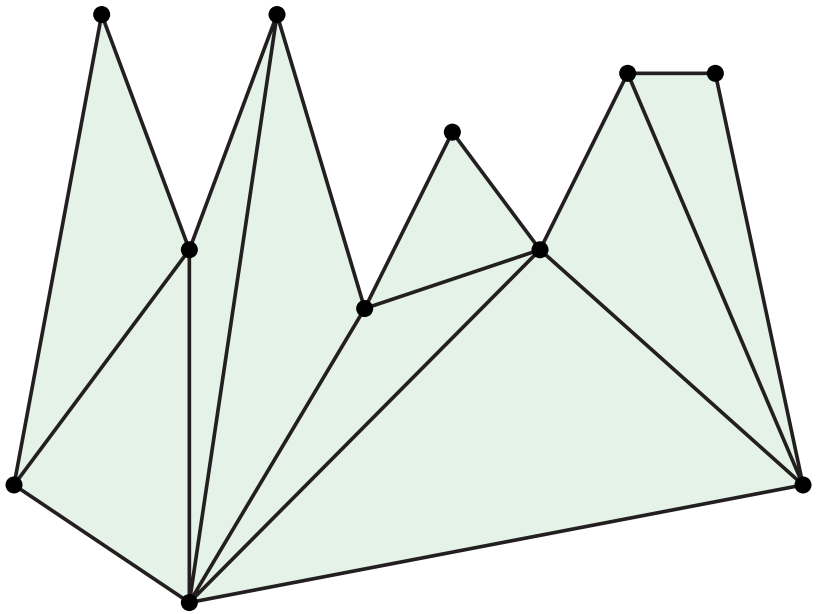


Triangulação de um Polígono



Objetivo: quebrar um polígono P em triângulos.

Triangulação de um Polígono



Objetivo: quebrar um polígono P em triângulos.

Adicione diagonais entre os vértices de P

- uma diagonal de P é um segmento aberto que conecta dois vértices e está contido no interior de P;
- uma triangulação de P é o conjunto maximal de diagonais de P;
- podemos vigiar P colocando uma câmera em cada triângulo.

Triangulação de um Polígono

Teorema 1

Todo polígono simples P admite uma triangulação. Qualquer triangulação de P com n vértices consiste em $(n-2)$ triângulos.

Triangulação de um Polígono

Teorema 1

Todo polígono simples P admite uma triangulação. Qualquer triangulação de P com n vértices consiste em $(n-2)$ triângulos.

Demo.: vamos provar a **existência** por indução.

– **caso trivial:** é verdade para triângulos ($n = 3$).

Triangulação de um Polígono

Teorema 1

Todo polígono simples P admite uma triangulação. Qualquer triangulação de P com n vértices consiste em $(n-2)$ triângulos.

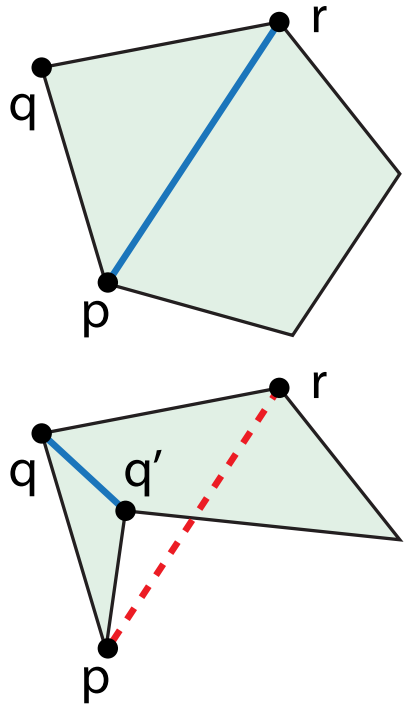
Demo.: vamos provar a **existência** por indução.

– **caso trivial:** é verdade para triângulos ($n = 3$).

– **caso indutivo:** é verdade para $m < n$.

Seja q o vértice mais a esquerda e p e r seus dois vizinhos.

- $\overline{pr} \subset P \Rightarrow \overline{pr}$ é uma diagonal de P;
- caso contrário, $\triangle pqr$ contem pelo menos um vértice. Seja q' o vértice mais próximo de $q \Rightarrow \overline{qq'}$ é uma diagonal.



Triangulação de um Polígono

Teorema 1

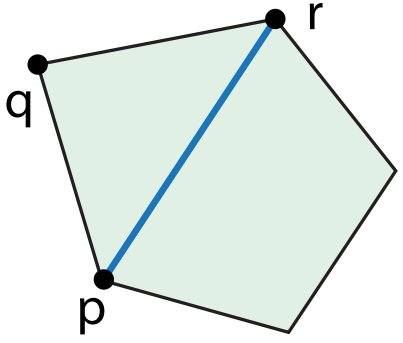
Todo polígono simples P admite uma triangulação. Qualquer triangulação de P com n vértices consiste em $(n-2)$ triângulos.

Demo.: vamos provar a **existência** por indução.

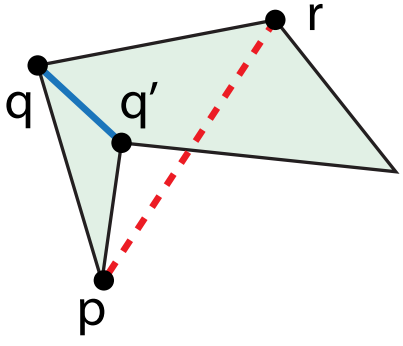
– **caso trivial:** é verdade para triângulos ($n = 3$).

– **caso indutivo:** é verdade para $m < n$.

Seja q o vértice mais a esquerda e p e r seus dois vizinhos.

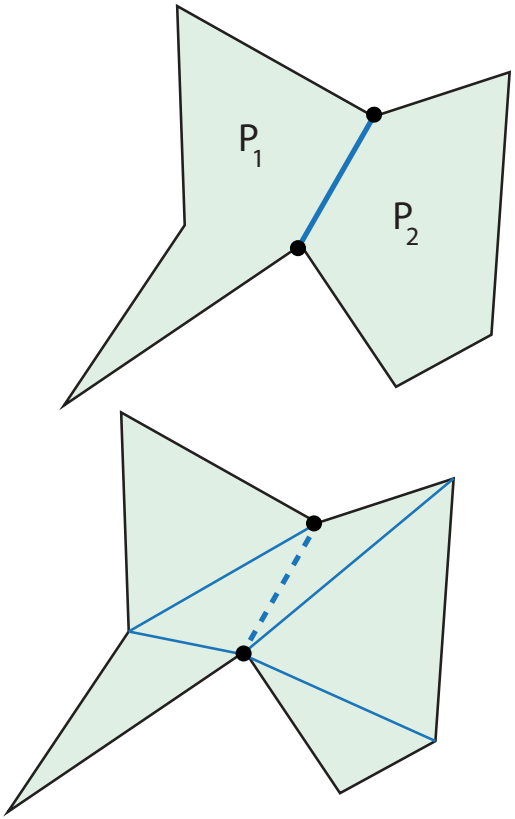


- $\overline{pr} \subset P \Rightarrow \overline{pr}$ é uma diagonal de P;
- caso contrário, $\triangle pqr$ contem pelo menos um vértice. Seja q' o vértice mais próximo de $q \Rightarrow \overline{qq'}$ é uma diagonal.



A diagonal divide P em dois, que por indução podem ser triangulados.

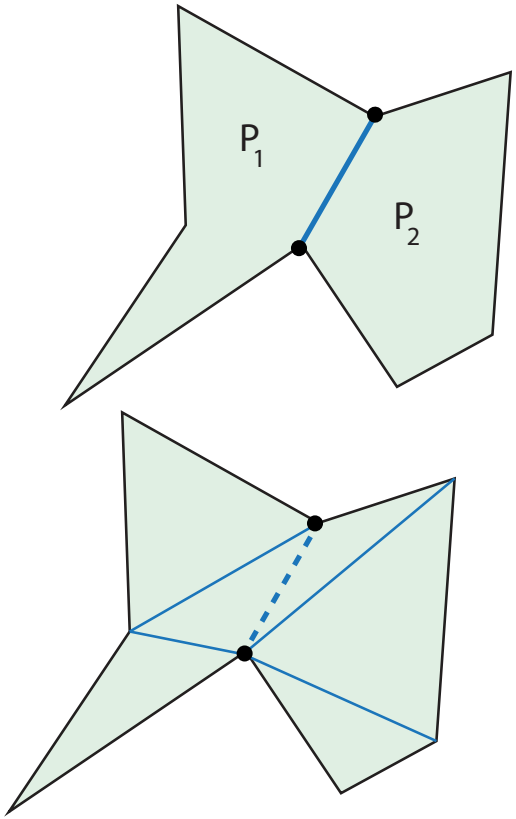
Triangulação de um Polígono



Demo.: vamos provar $\#\triangle = n - 2$.

Qualquer diagonal divide P em dois polígonos simples P_1 e P_2 com m_1 e m_2 vértices, respectivamente.

Triangulação de um Polígono



Demo.: vamos provar $\#\triangle = n - 2$.

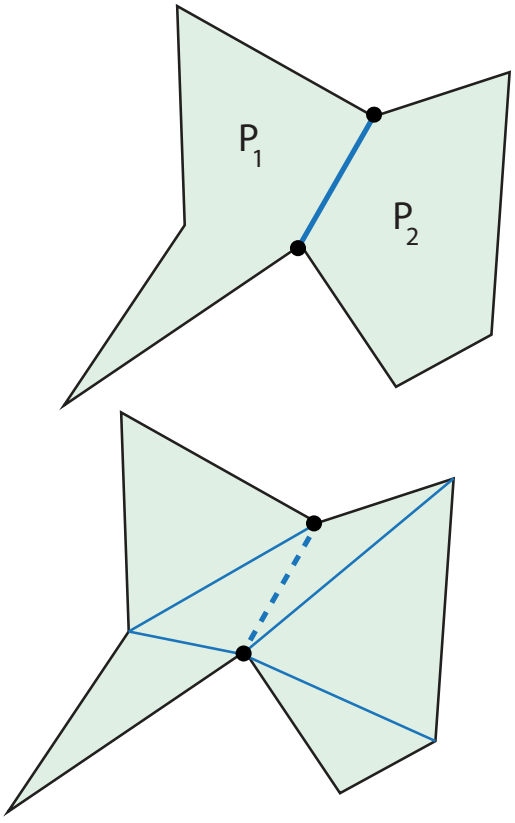
Qualquer diagonal divide P em dois polígonos simples P_1 e P_2 com m_1 e m_2 vértices, respectivamente.

Por indução, P_1 e P_2 podem ser triangulados em $(m_1 - 2)$ e $(m_2 - 2)$ triângulos, respectivamente.

- os vértices da diagonal aparecem em P_1 e P_2;
- os demais vértices aparecem apenas uma vez em cada polígono.

Portanto, $m_1 + m_2 = n + 2$.

Triangulação de um Polígono



Demo.: vamos provar $\#\triangle = n - 2$.

Qualquer diagonal divide P em dois polígonos simples P_1 e P_2 com m_1 e m_2 vértices, respectivamente.

Por indução, P_1 e P_2 podem ser triangulados em $(m_1 - 2)$ e $(m_2 - 2)$ triângulos, respectivamente.

- os vértices da diagonal aparecem em P_1 e P_2;
- os demais vértices aparecem apenas uma vez em cada polígono.

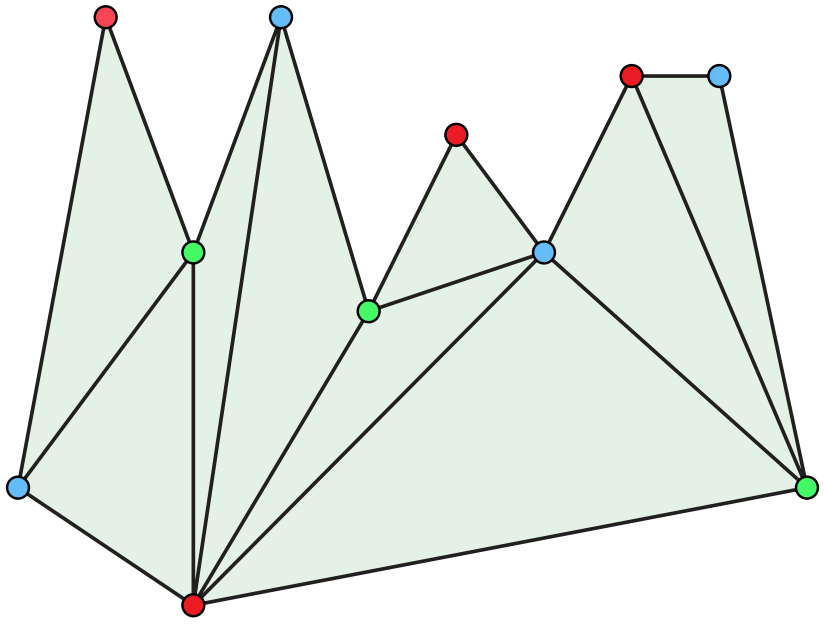
Portanto, $m_1 + m_2 = n + 2$.

Por indução, a triangulação de P tem $(m_1 - 2) + (m_2 - 2) = n - 2$ triângulos.

Problema da Galeria de Arte

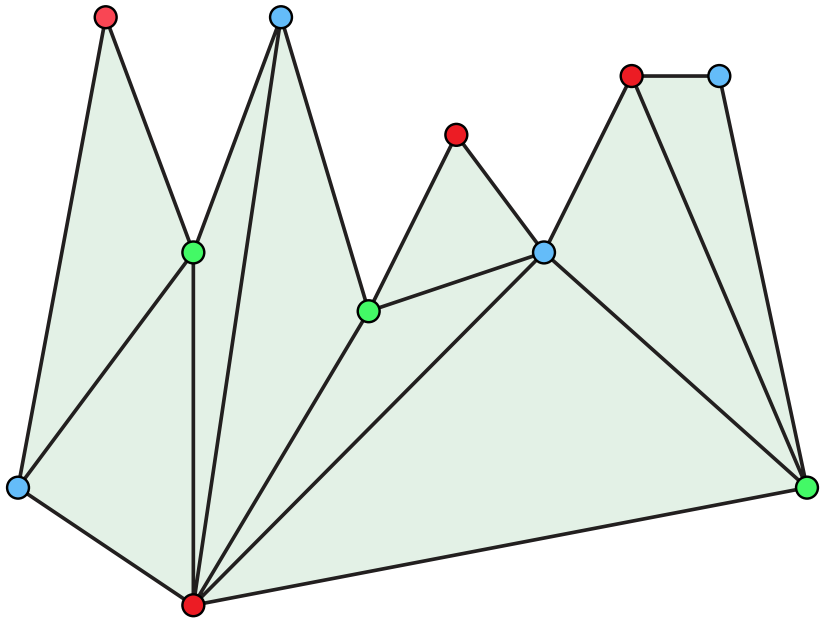
- Teorema 1 $\Rightarrow$ $(n-2)$ câmeras podem vigiar um polígono simples;
- uma câmera na diagonal pode vigiar dois triângulos $\Rightarrow$ número de câmeras pode cair pela metade ($\lfloor \frac{n-2}{2} \rfloor$);
- um vértice é adjacente a vários triângulos $\Rightarrow$ colocar câmeras nos **vértices** pode fornecer uma solução melhor.

Triangulações 3-coloridas



Ideia: colorir os vértices de uma triangulação de P usando 3 cores tal que os vértices qualquer triangulo tenham cores distintas.

Triangulações 3-coloridas



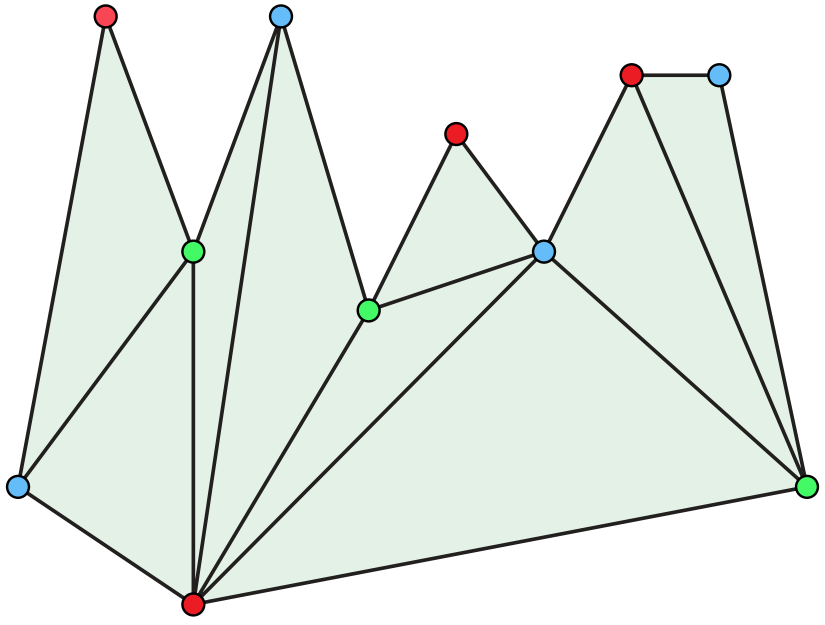
Ideia: colorir os vértices de uma triangulação de P usando 3 cores tal que os vértices qualquer triangulo tenham cores distintas.

Solução: Coloque as câmeras em todos os vértices da mesma cor.

- escolha a cor que aparece menos $\Rightarrow$ $\lfloor \frac{n}{3} \rfloor$ câmeras;
- No exemplo acima, as câmeras seriam colocadas nos vértices verdes.

Algoritmo de Triangulação de um Polígono

Assumiremos que os vértices de P estão ordenados no sentido **anti-horário**.

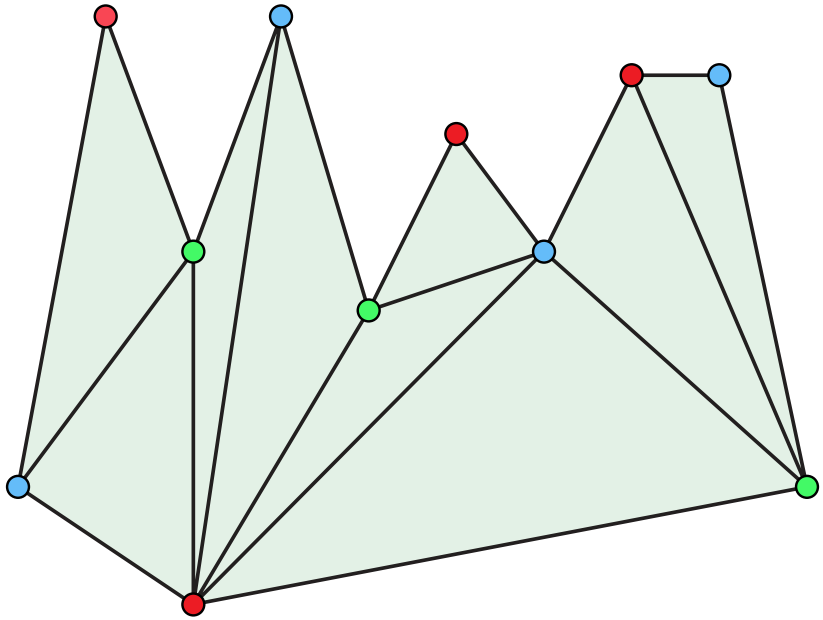


Definição

Uma orelha da triangulação do polígono P é um triângulo $\triangle pqr$ tal que $\overline{pr}$ é uma diagonal de P.

Algoritmo de Triangulação de um Polígono

Assumiremos que os vértices de P estão ordenados no sentido **anti-horário**.



Definição

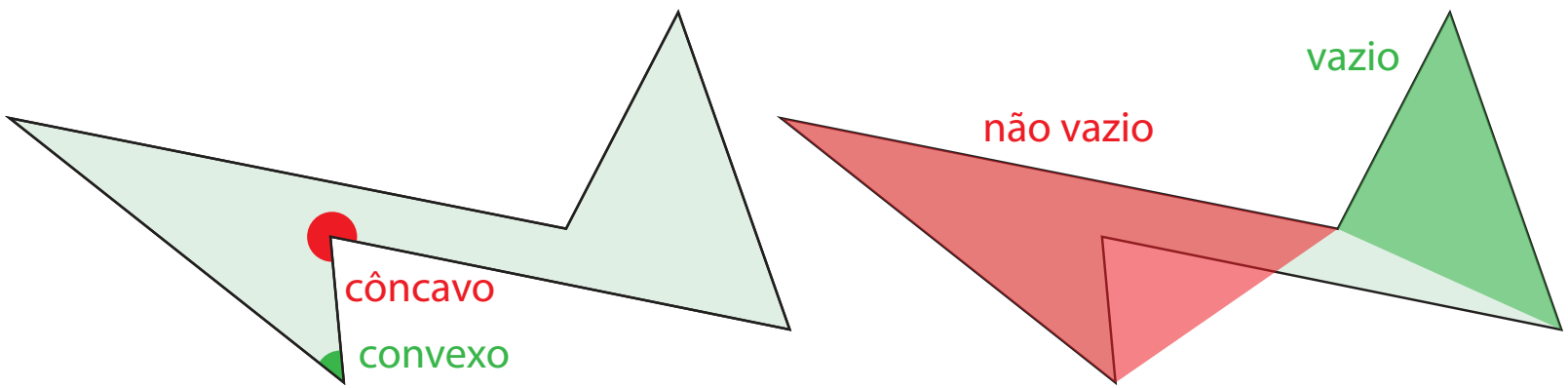
Uma orelha da triangulação do polígono P é um triângulo $\triangle pqr$ tal que $\overline{pr}$ é uma diagonal de P.

Ideia: cortaremos as orelhas de P assim que forem processadas (**algoritmo de remoção de orelhas**).

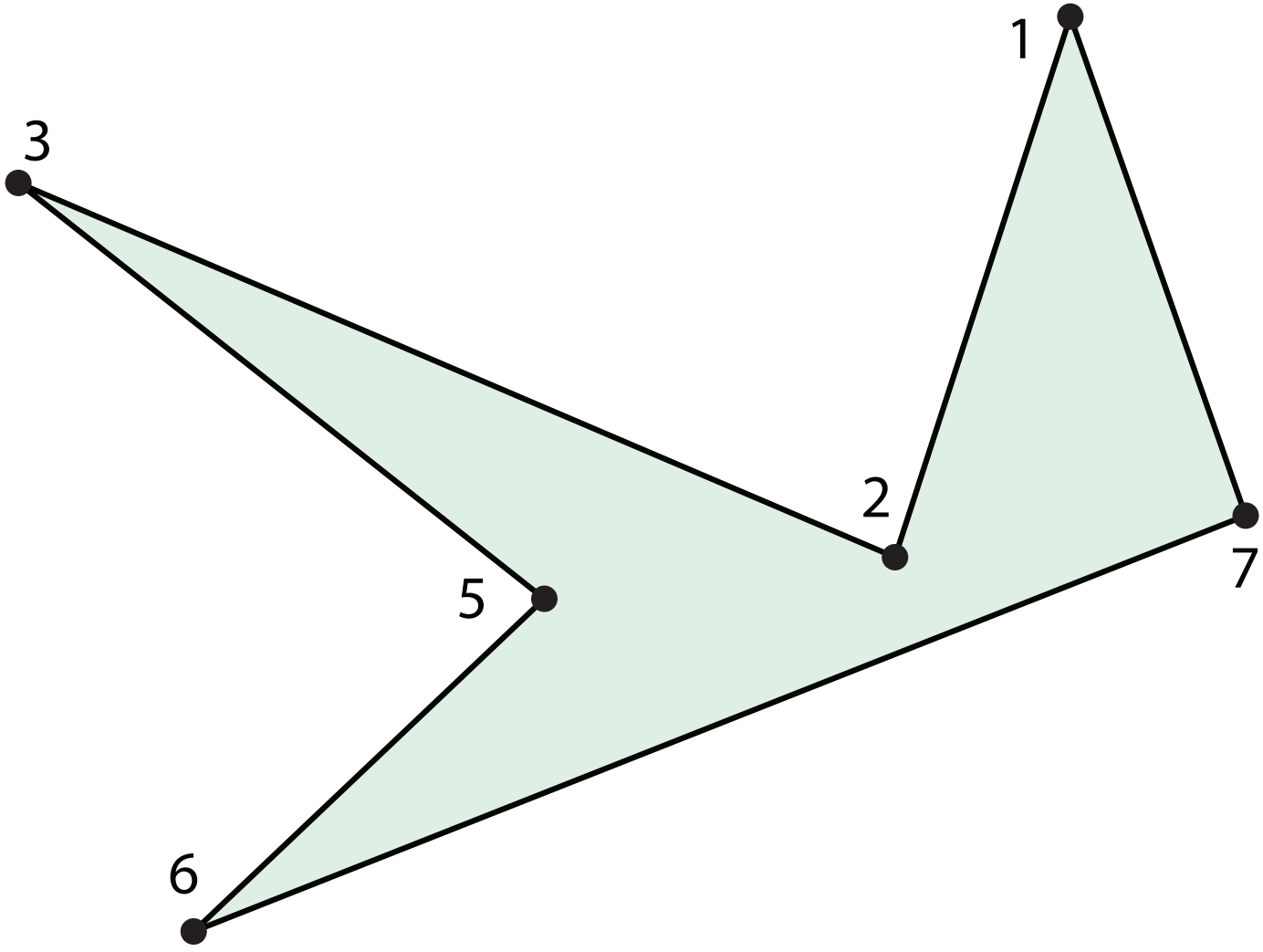
Algoritmo de Triangulação de um Polígono

Para verificar se 3 vértices consecutivos $p_0, p_1, p_2 \in P$ é uma orelha, precisamos checar duas coisas:

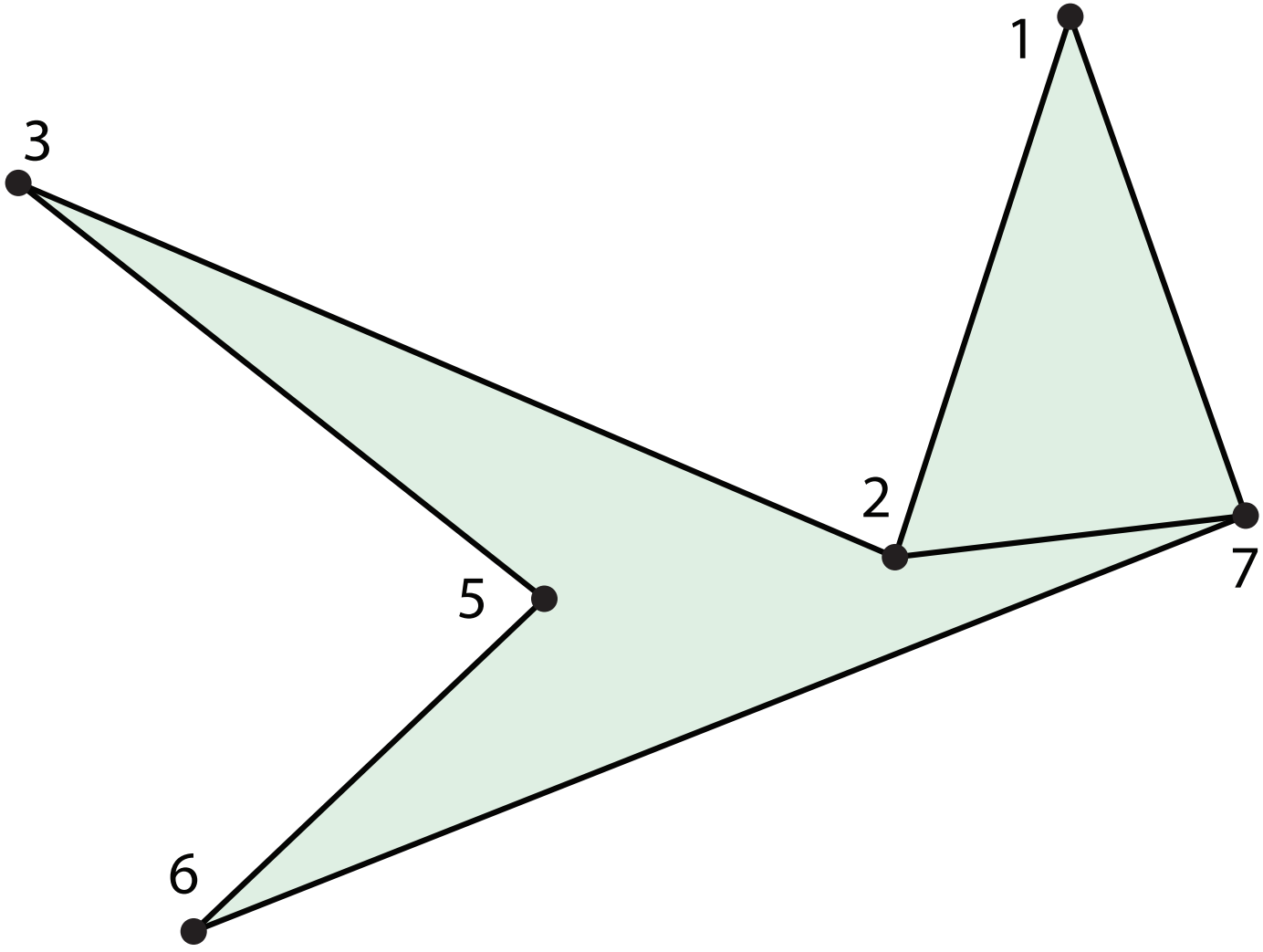
1. se o ângulo $\angle p_0 p_1 p_2 < 180^\circ$ é **convexo** (regra da mão direita);
2. o triângulo $\triangle p_0 p_1 p_2$ é **vazio**, i.e., não existe vértice de P no interior do triângulo.



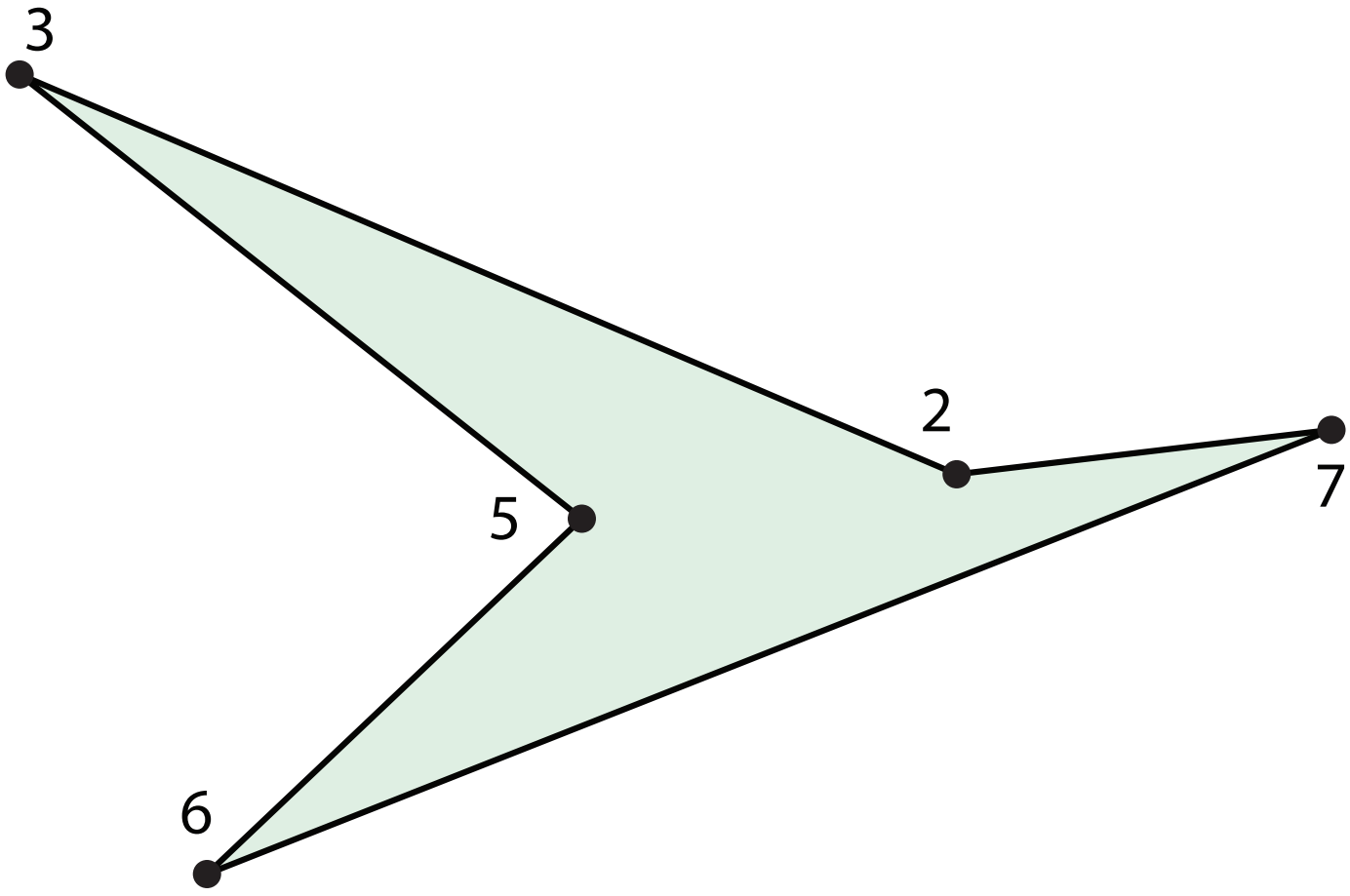
Algoritmo de Triangulação de um Polígono



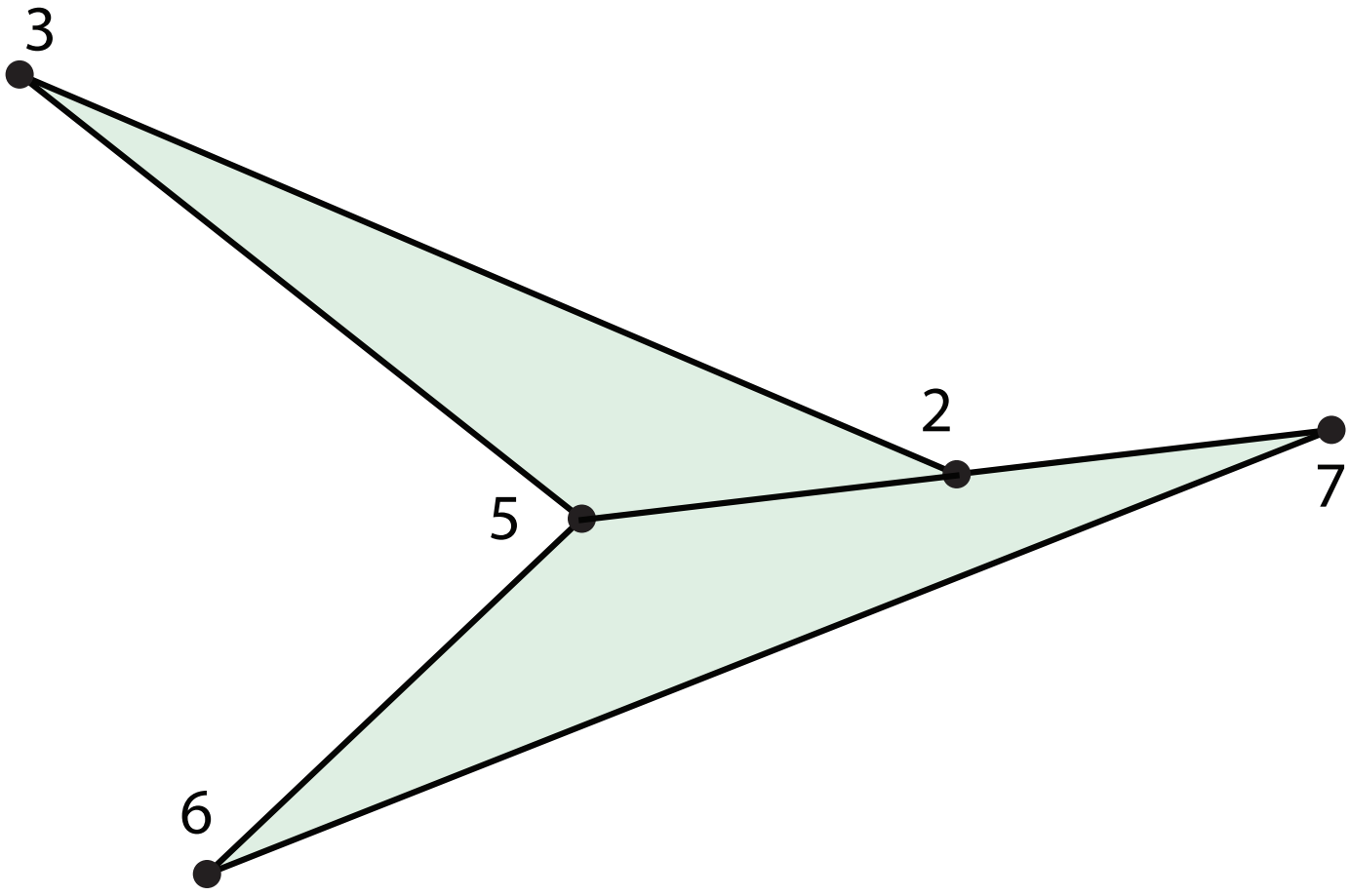
Algoritmo de Triangulação de um Polígono



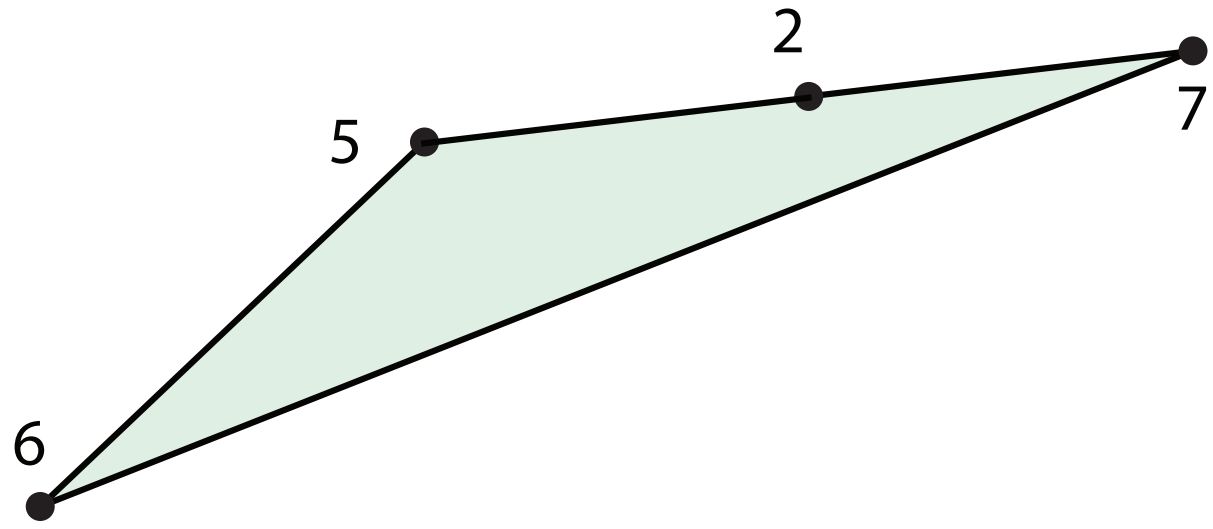
Algoritmo de Triangulação de um Polígono



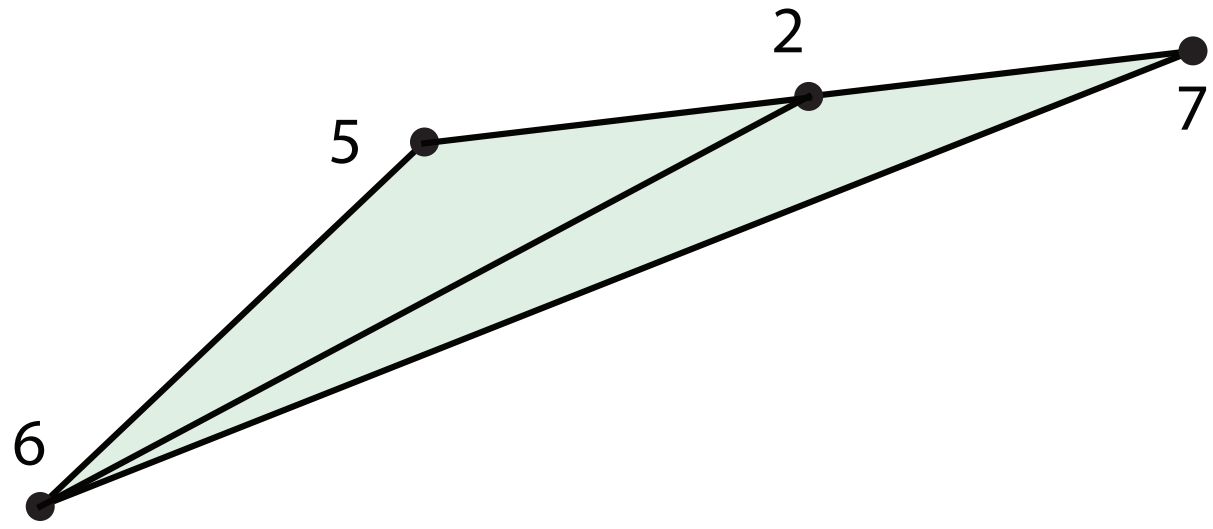
Algoritmo de Triangulação de um Polígono



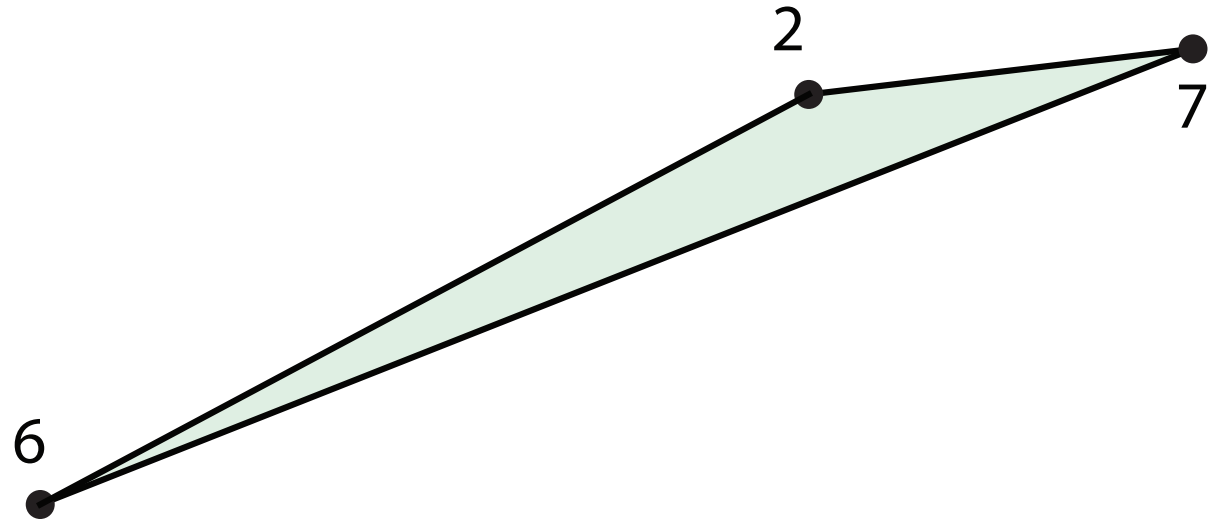
Algoritmo de Triangulação de um Polígono



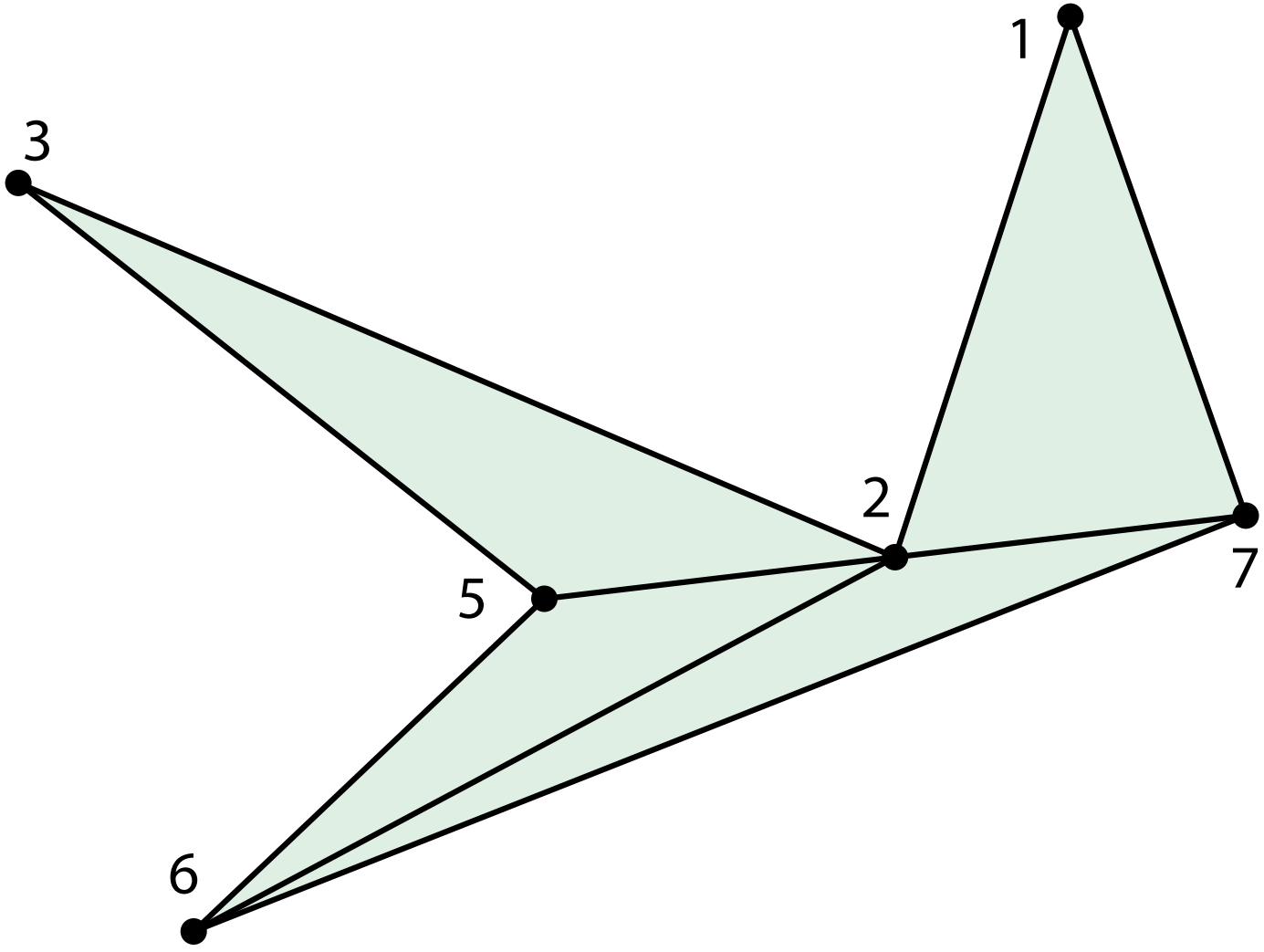
Algoritmo de Triangulação de um Polígono



Algoritmo de Triangulação de um Polígono



Algoritmo de Triangulação de um Polígono



Teorema da Galeria de Arte

Teorema 2

Para um polígono simples P com n vértices, $\lfloor \frac{n}{3} \rfloor$ câmeras são suficientes (no pior caso) para que cada ponto interior de P seja visível por pelo menos uma câmera.

Teorema da Galeria de Arte

Teorema 2

Para um polígono simples P com n vértices, $\lfloor \frac{n}{3} \rfloor$ câmeras são suficientes (no pior caso) para que cada ponto interior de P seja visível por pelo menos uma câmera.

Solução do Problema da Galeria de Arte

1. Triangule P com o algoritmo de remoção de orelhas – $O(n^2)$;
2. Faça uma 3-coloração da triangulação;
3. Escolha os vértices da cor que aparece menos.

Teorema da Galeria de Arte

Teorema 2

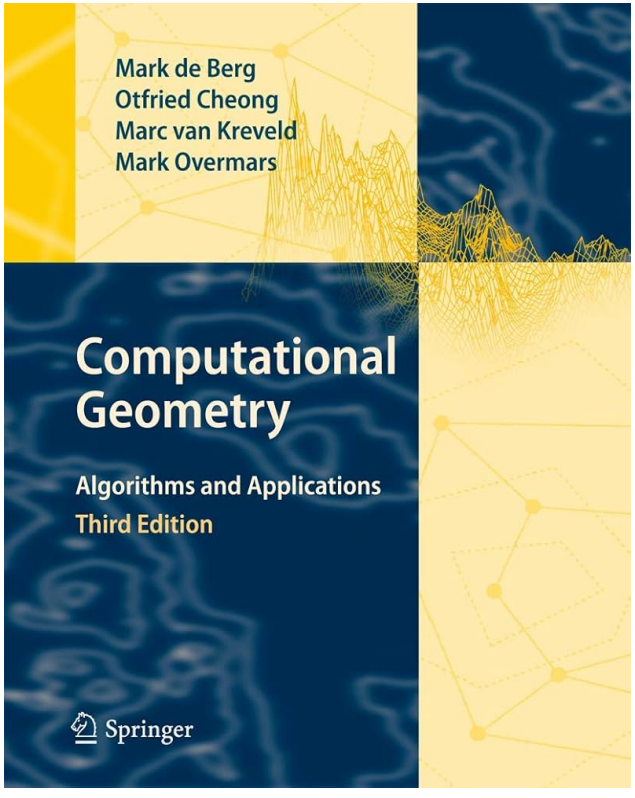
Para um polígono simples P com n vértices, $\lfloor \frac{n}{3} \rfloor$ câmeras são suficientes (no pior caso) para que cada ponto interior de P seja visível por pelo menos uma câmera.

Solução do Problema da Galeria de Arte

1. Triangule P com o algoritmo de remoção de orelhas – $O(n^2)$;
2. Faça uma 3-coloração da triangulação;
3. Escolha os vértices da cor que aparece menos.

- Existe um algoritmo rápido de triangulação de polígonos – complexidade $O(n \log(n))$;
- Use uma estrutura de dados para salvar a relação de adjacência dos triângulos – consultas geométricas em $O(1)$.

Geometria Computacional



Disciplinas:

- SME0250 – Métodos Numéricos para Geração de Malhas
- SME0271 – Modelagem Geométrica

Livros:

- Introdução à Geometria Computacional, L.H. Figueiredo e P.C.P. Carvalho, 18° CBM, IMPA, 1991