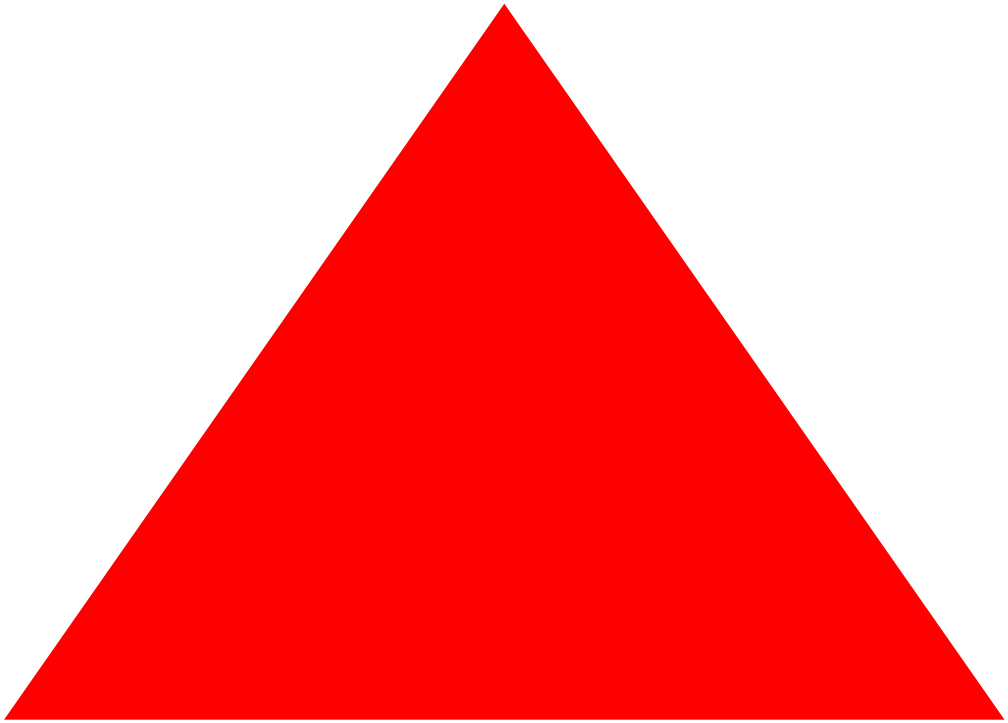
Insper

Computação Gráfica

Aula 2: Desenhando Triângulos e X3D

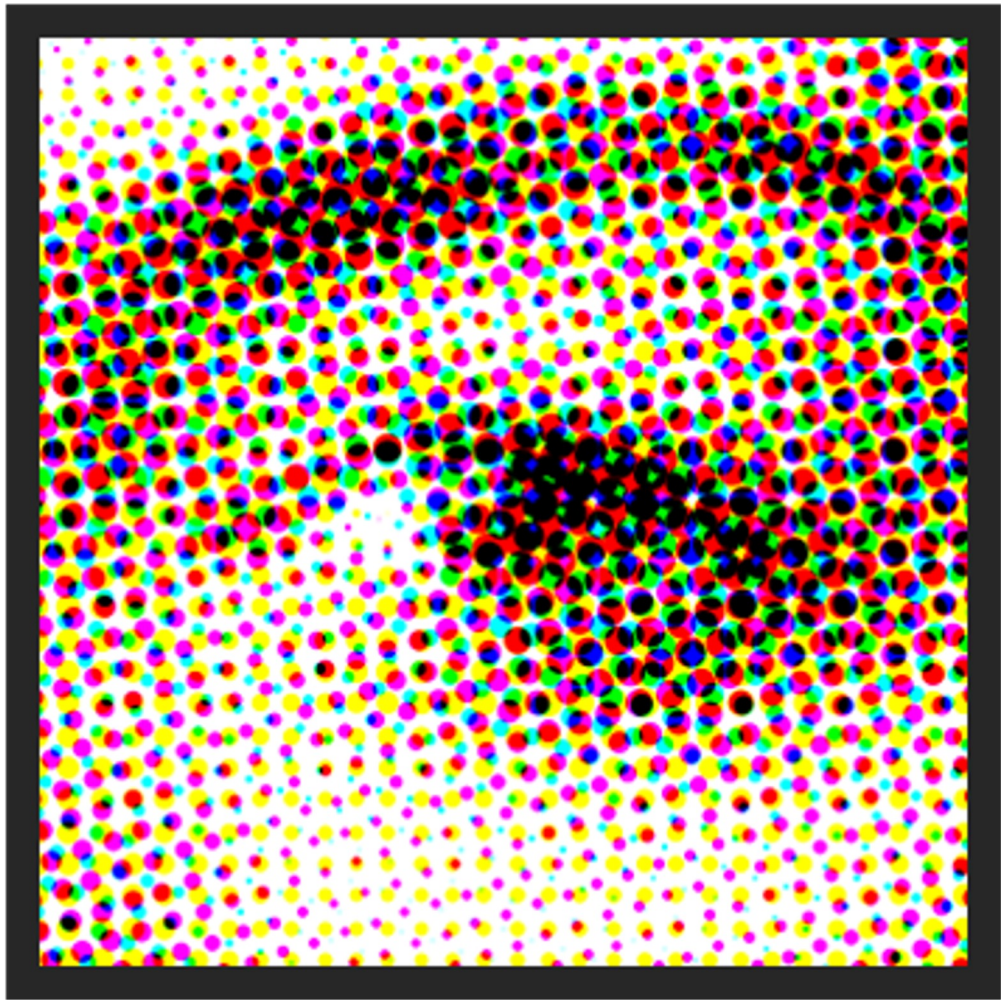
Computação Gráfica

Desenhando Triângulos



Insper

Como as imagens são apresentadas



Impressão colorida : CMYK com meio-tom(half-tone)

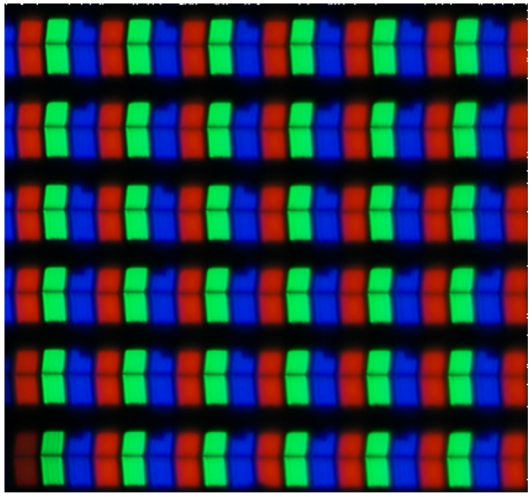
Insper

Telas / Diplays

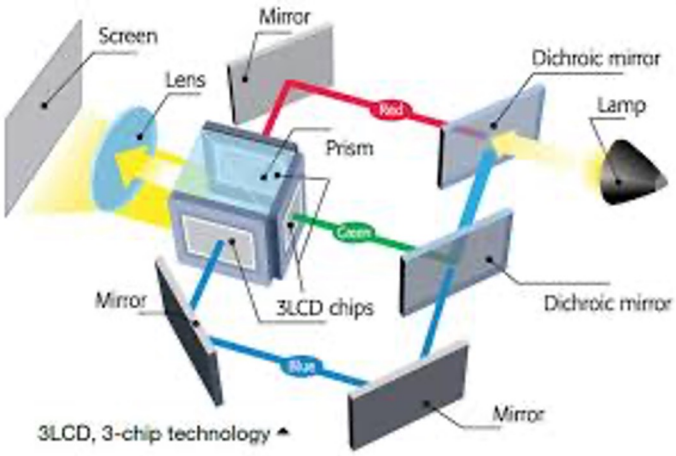
As telas são compostas de pixels
Iremos simplificar como quadradinhos
Cada "quadradinho" vai ter sua cor

Um pixel

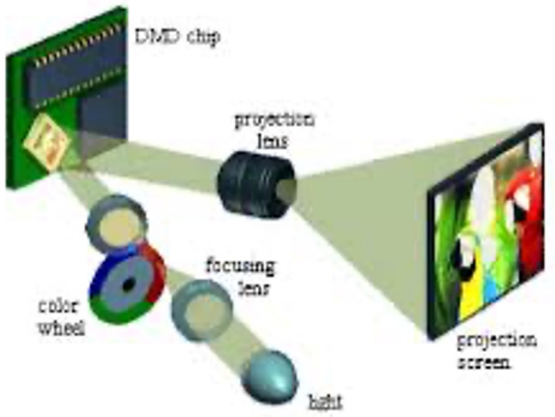
Tecnologias para gerar pixels



telas convencionais



projeção 3 chip

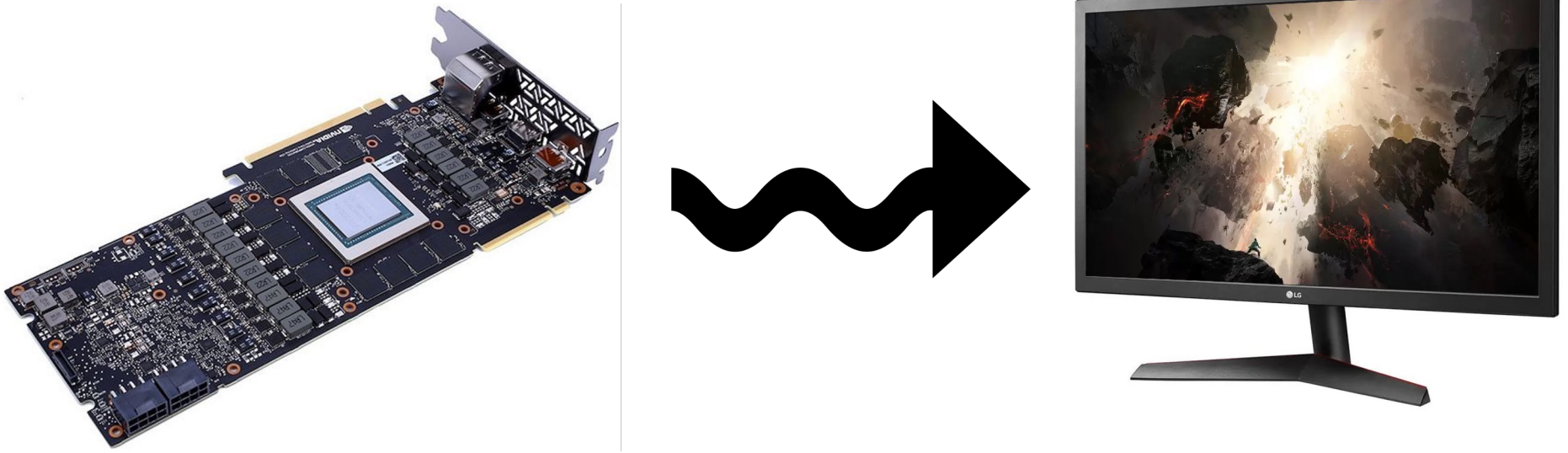


projeção single chip

Insper

Framebuffer

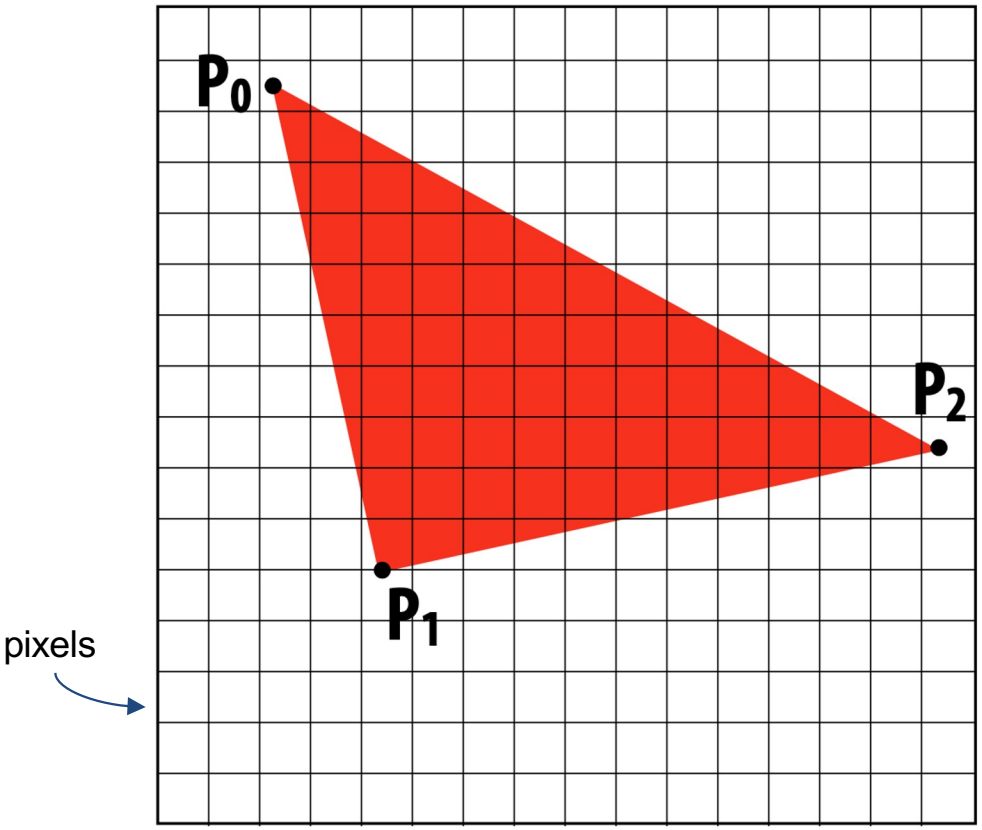
Memória na placa gráfica que será usada para produzir a imagem no monitor que estiver conectado.



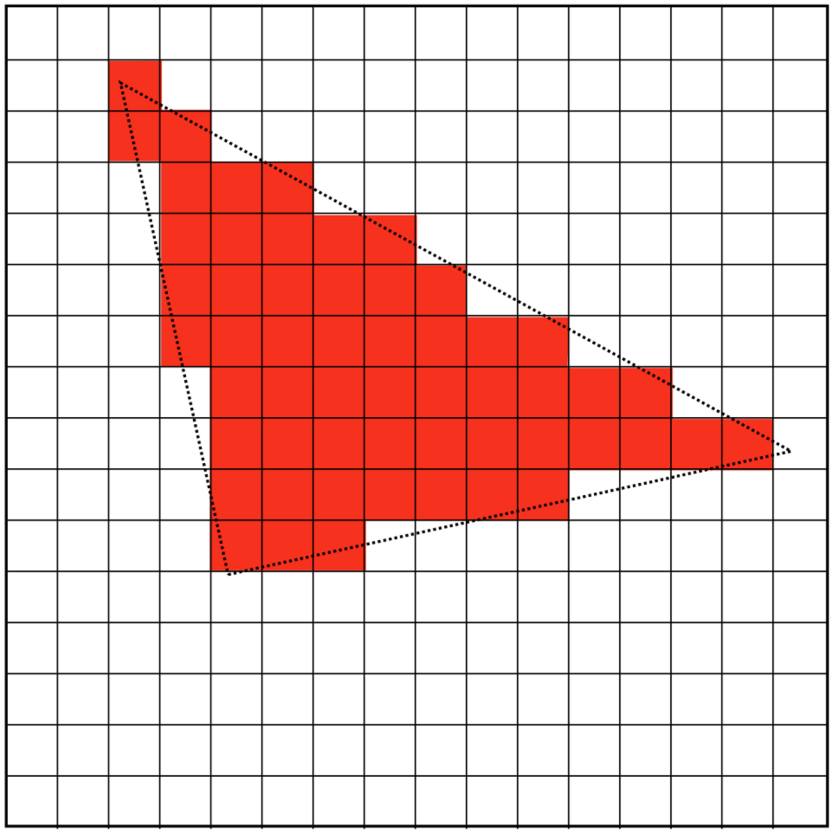
Desenhando um Triângulo no Framebuffer

Processo que chamamos de: Rasterização (triangle rasterization)

Entrada:
Posições projetadas dos vértices do triângulo



Saída:
conjunto de pixels coloridos pelo triângulo



Insper

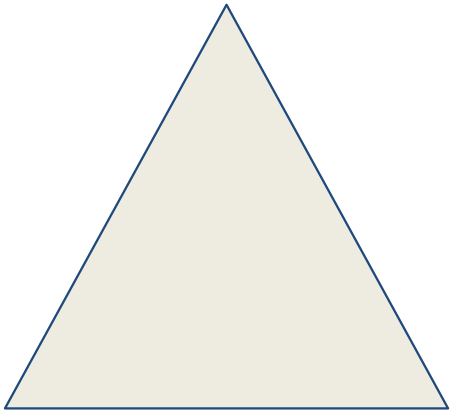
Triângulo – Primitiva Fundamental

Por que triângulos:

- Polígono mais simples possível
- Outros polígonos podem ser subdivididos em triângulos
- Podemos nos focar em otimizar um tipo de operação

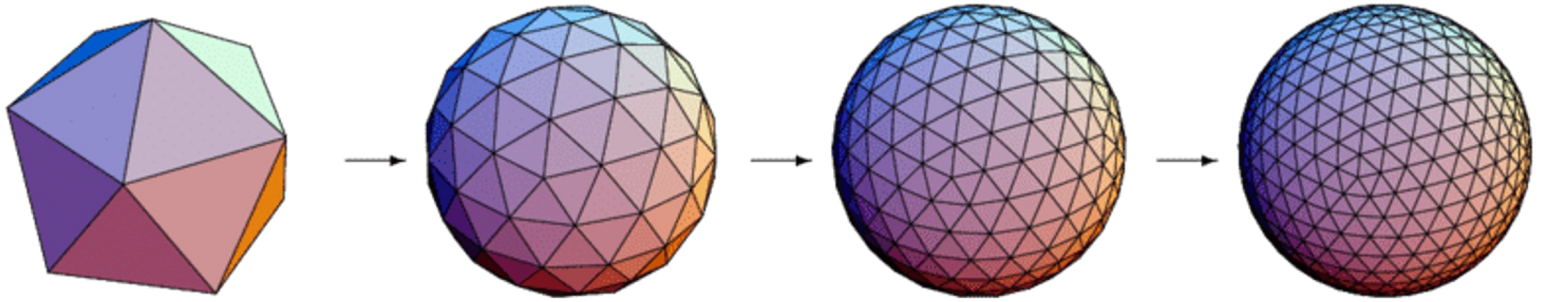
Triângulos têm propriedades únicas:

- São sempre planos
- Tem um interior bem definido
- Interpolar valores no seu interior é simples

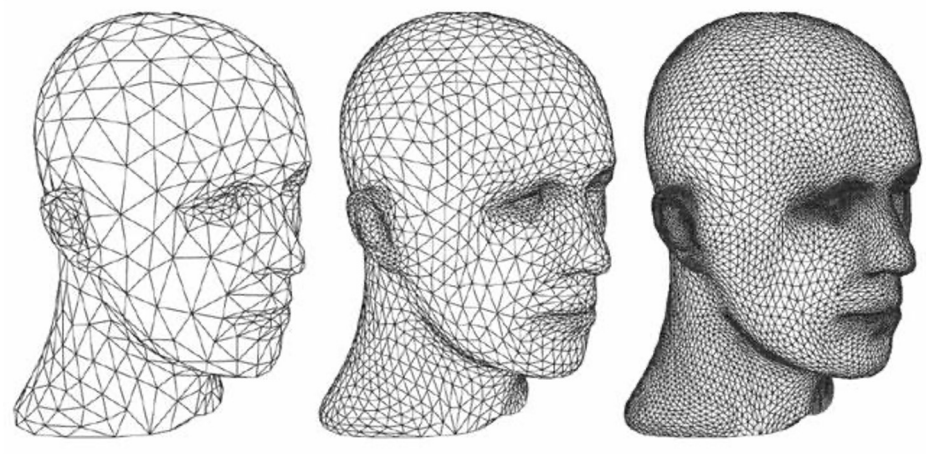


Triângulos (High e Low Poly Count)

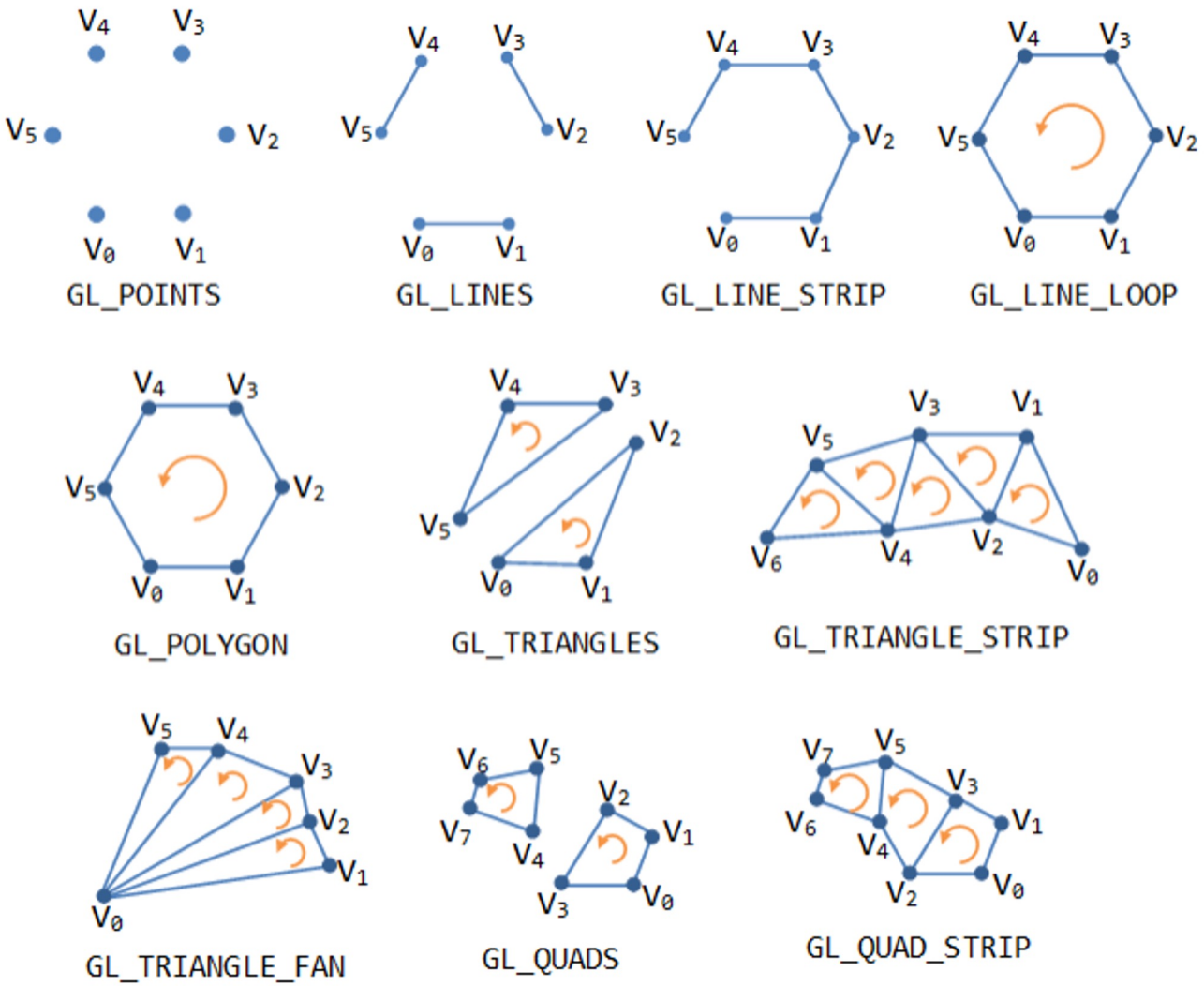
Os triângulos são um dos blocos básicos para criar formas e superfícies geométricas mais complexas.



por Martin Styner



Primitivas Gráficas em OpenGL

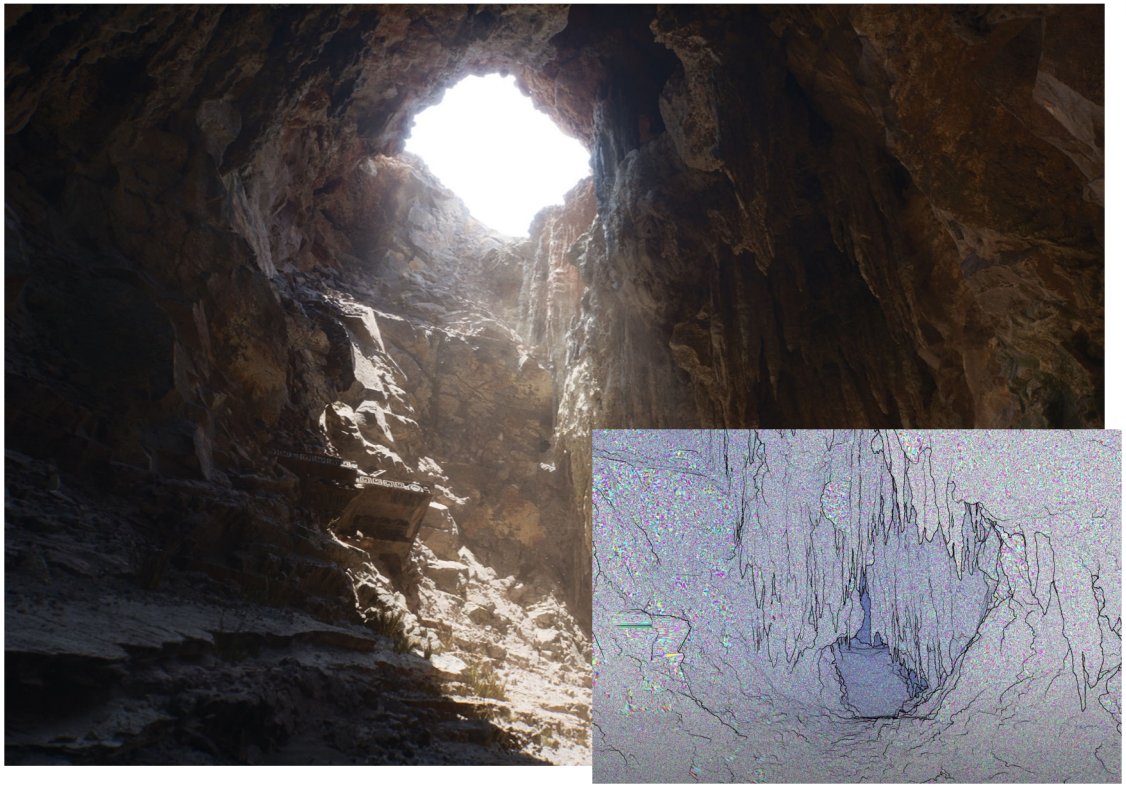


https://www3.ntu.edu.sg/home/ehchua/programming/opengl/cg_introduction.html

Insper

Low Poly e High Poly

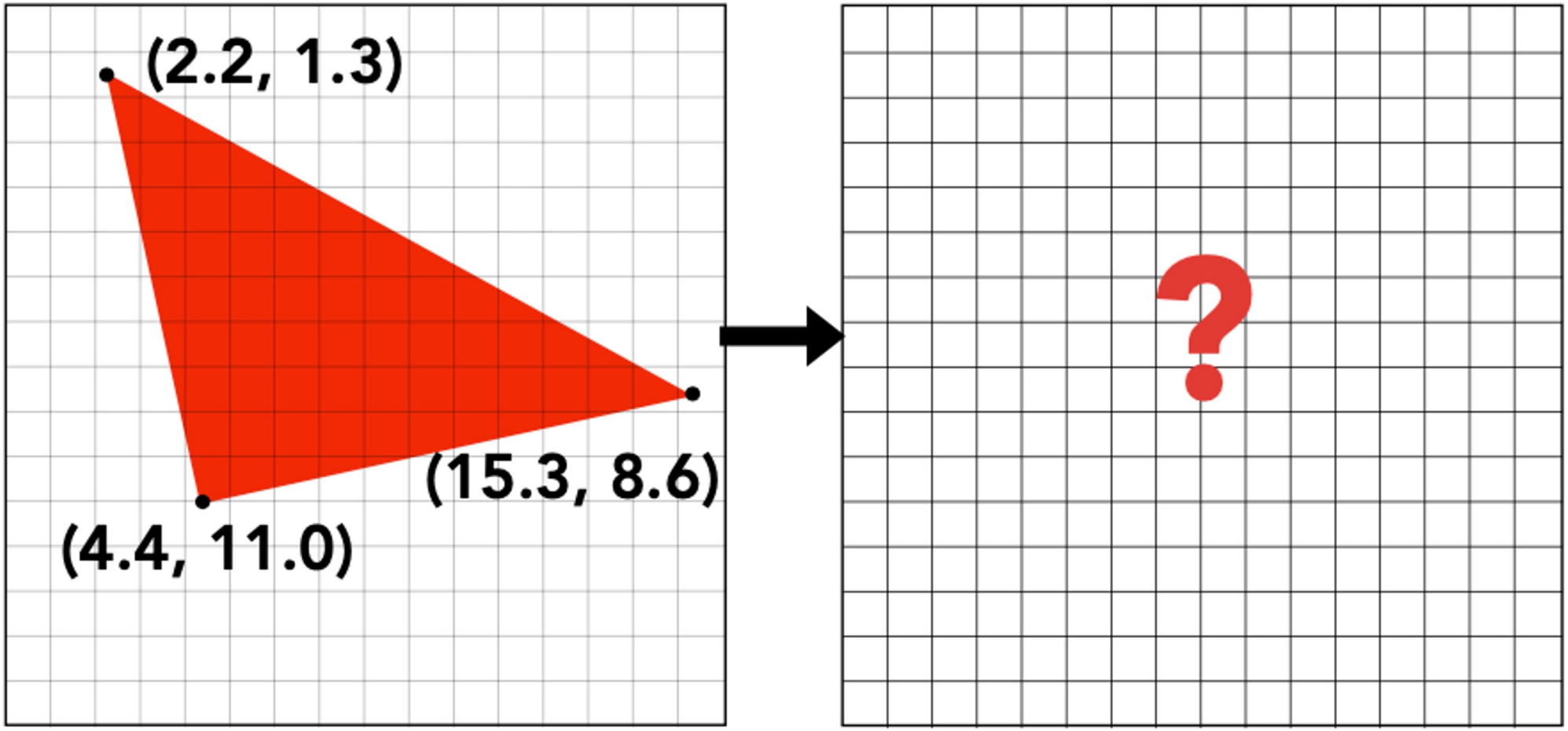
Star Fox / SNES (1993)



Unreal 5 / PS5 (2019)

Insper

Quais valores os pixels terão após renderização

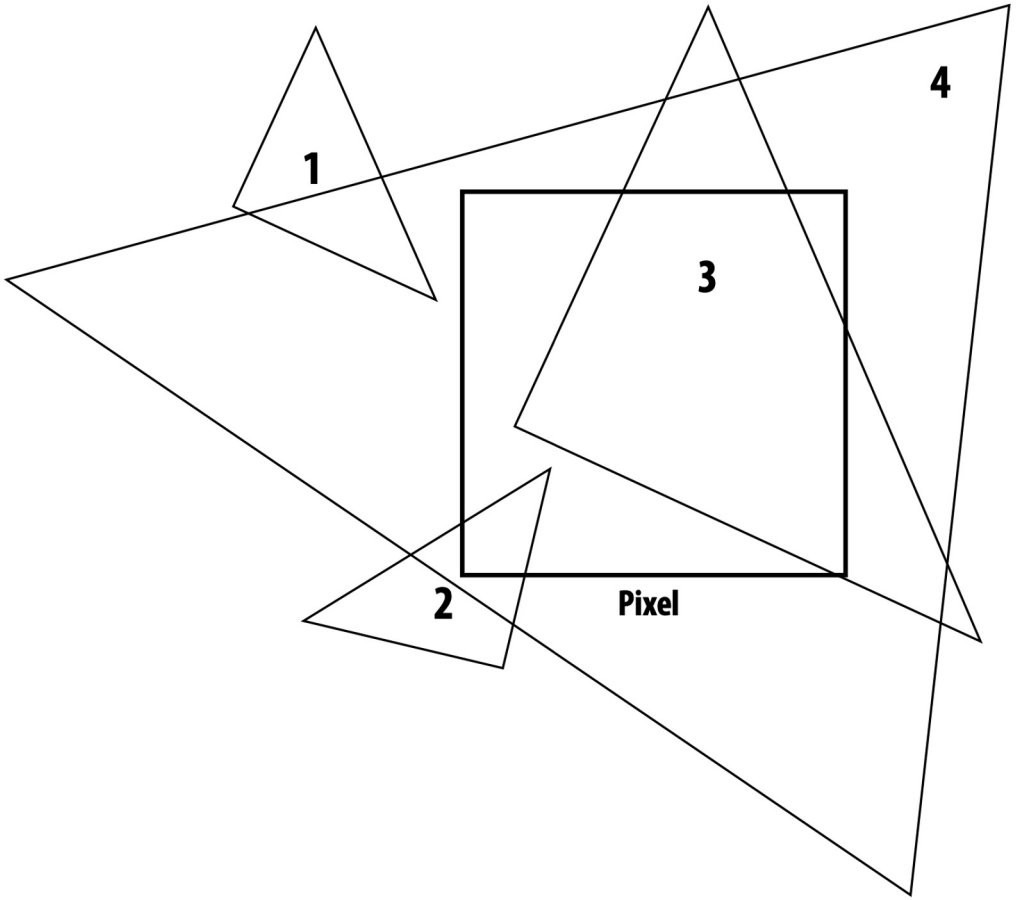


Alguns pixels são cobertos pelo triângulo outros não.

Insper

O que significa cobrir um pixel por triângulos?

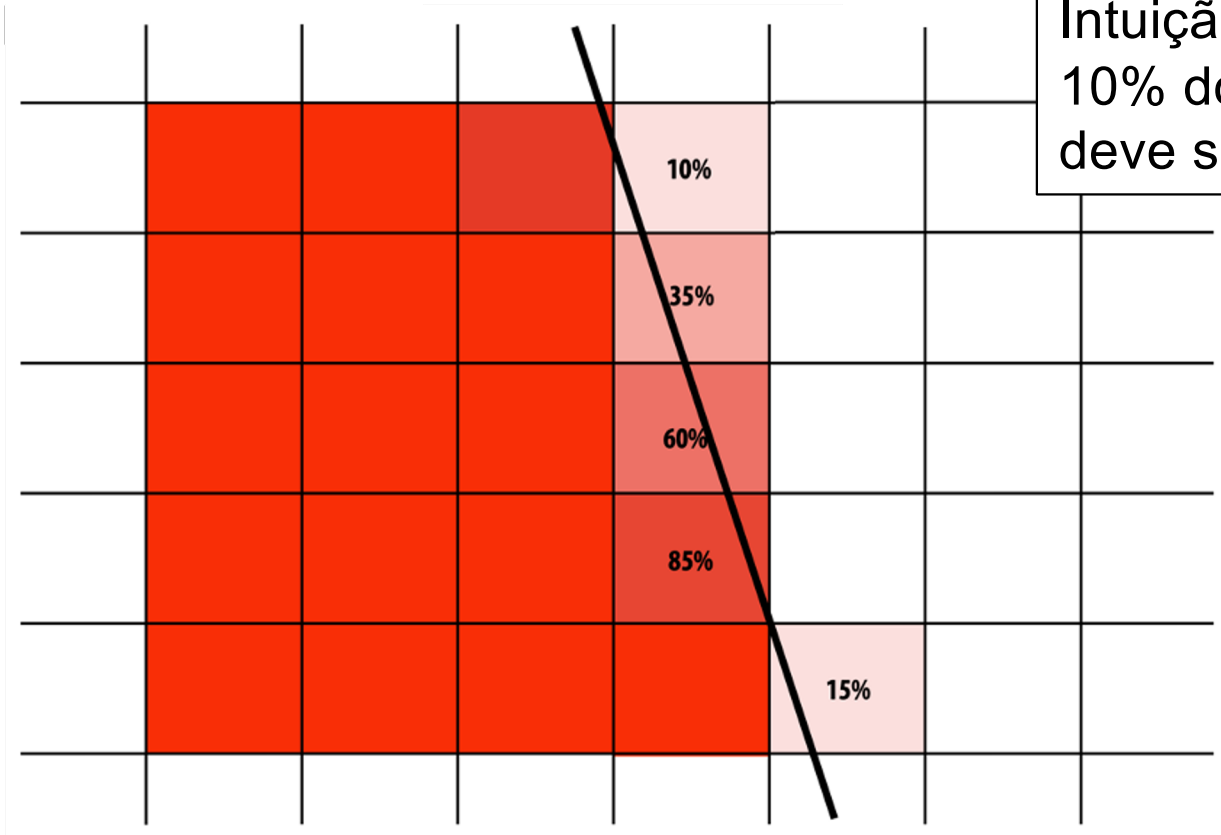
Pergunta: quais triângulos “cobrem” este pixel?



Insper

Pintando os pixels sobre as arestas

Uma opção: calcular a fração da área de pixel coberta pelo triângulo e, em seguida, colorir o pixel de acordo com essa fração.

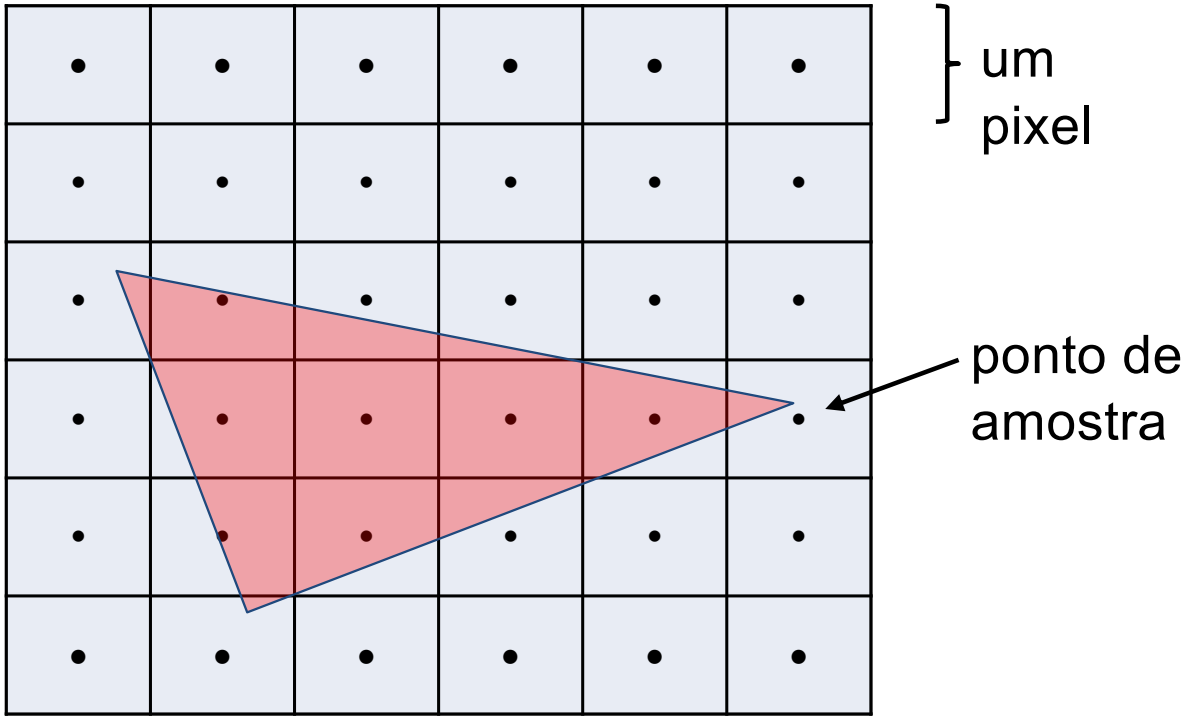


Intuição: se o triângulo cobre 10% do pixel, então o pixel deve ser 10% vermelho?

Insper

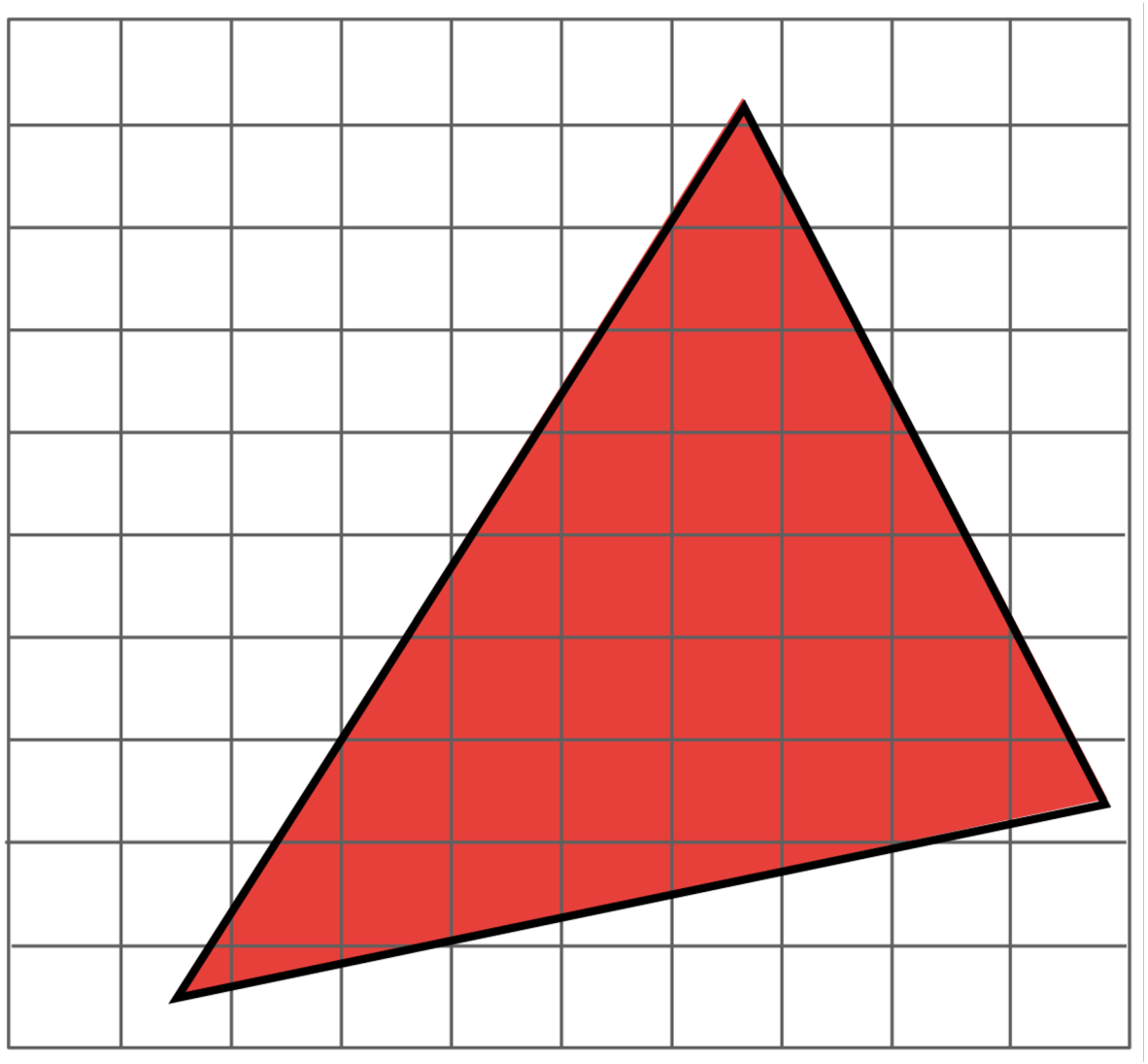
Desenhando os pixels dos triângulos

Uma forma de saber a cor do pixel é amostrando um ponto do pixel para coletar a cor do objeto.



Insper

Desenhando um triângulo por amostragem 2D



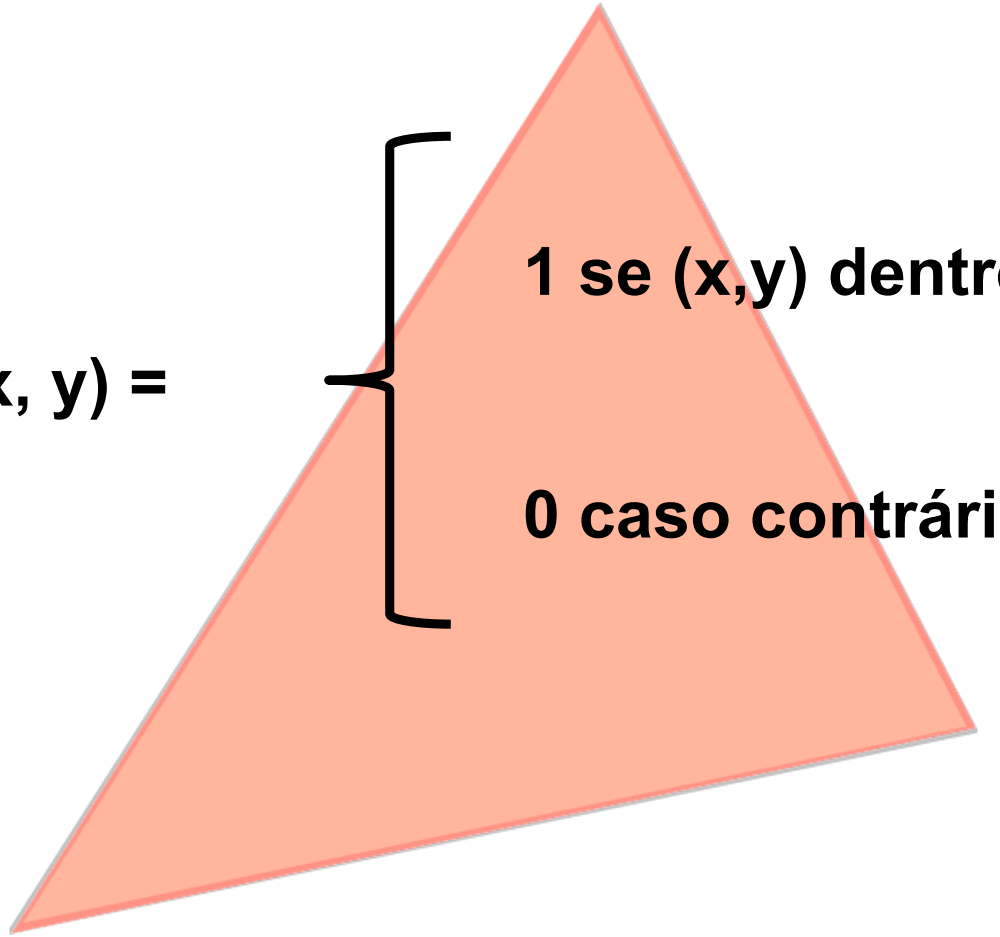
Insper

Definindo uma função binária

inside(tri ,x, y) =

1 se (x,y) dentro do triângulo

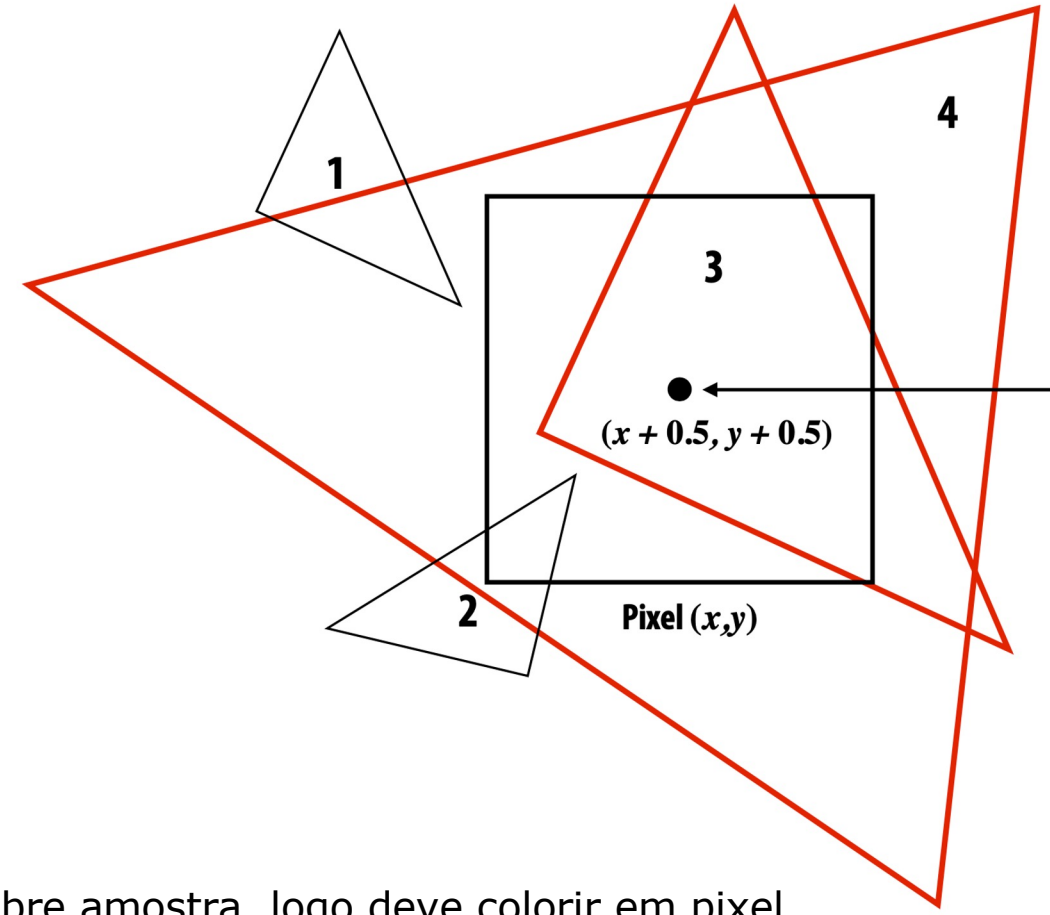
0 caso contrário



tri = triângulo

Nas APIs gráficas esse teste pode ser chamado de inside-outside test ou ainda coverage test.

Amostrando a função binária



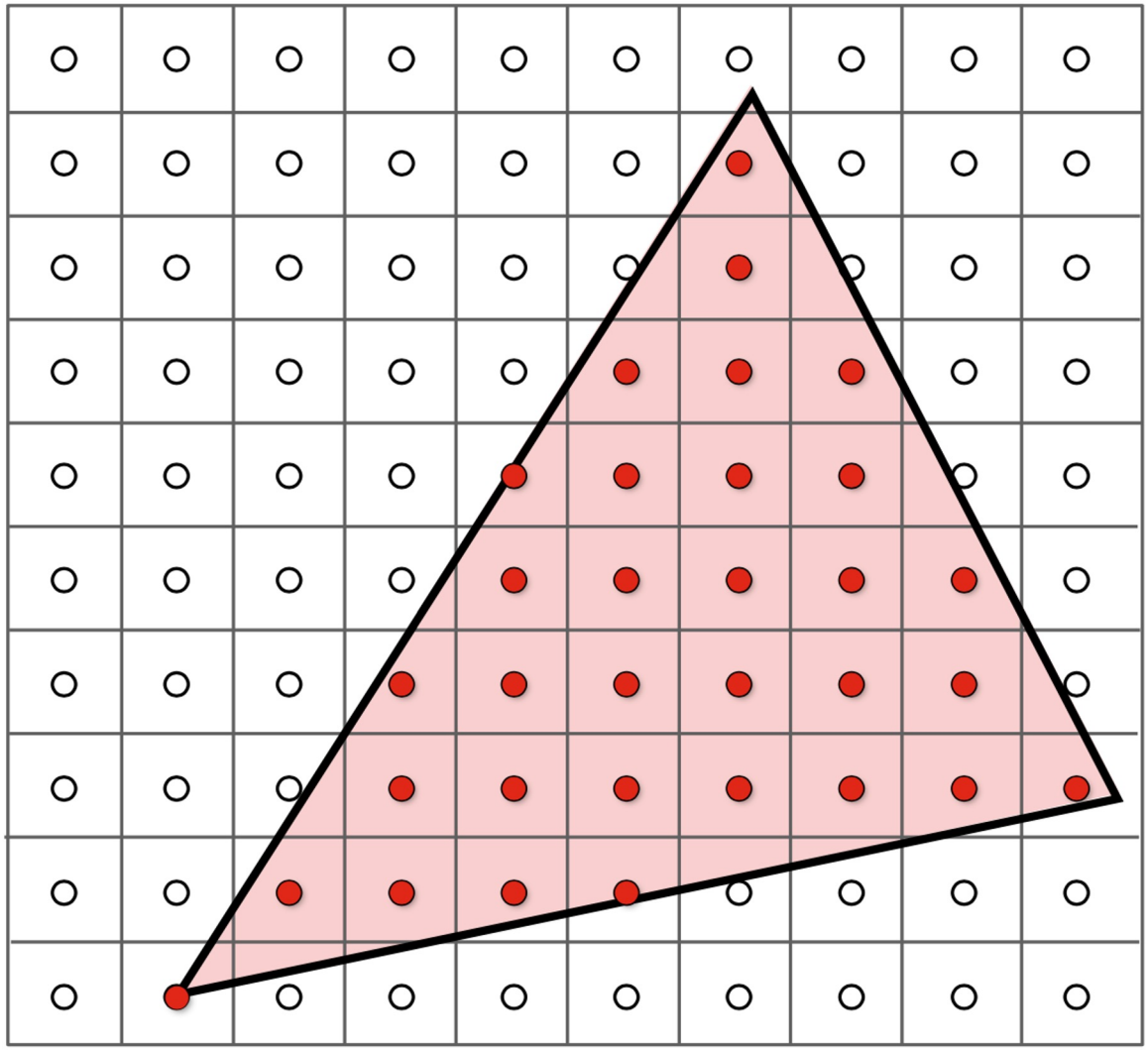
Escolhi o centro do pixel para ser o local da amostragem

= triângulo cobre amostra, logo deve colorir em pixel

= triângulo não cobre a amostra, não colore no pixel

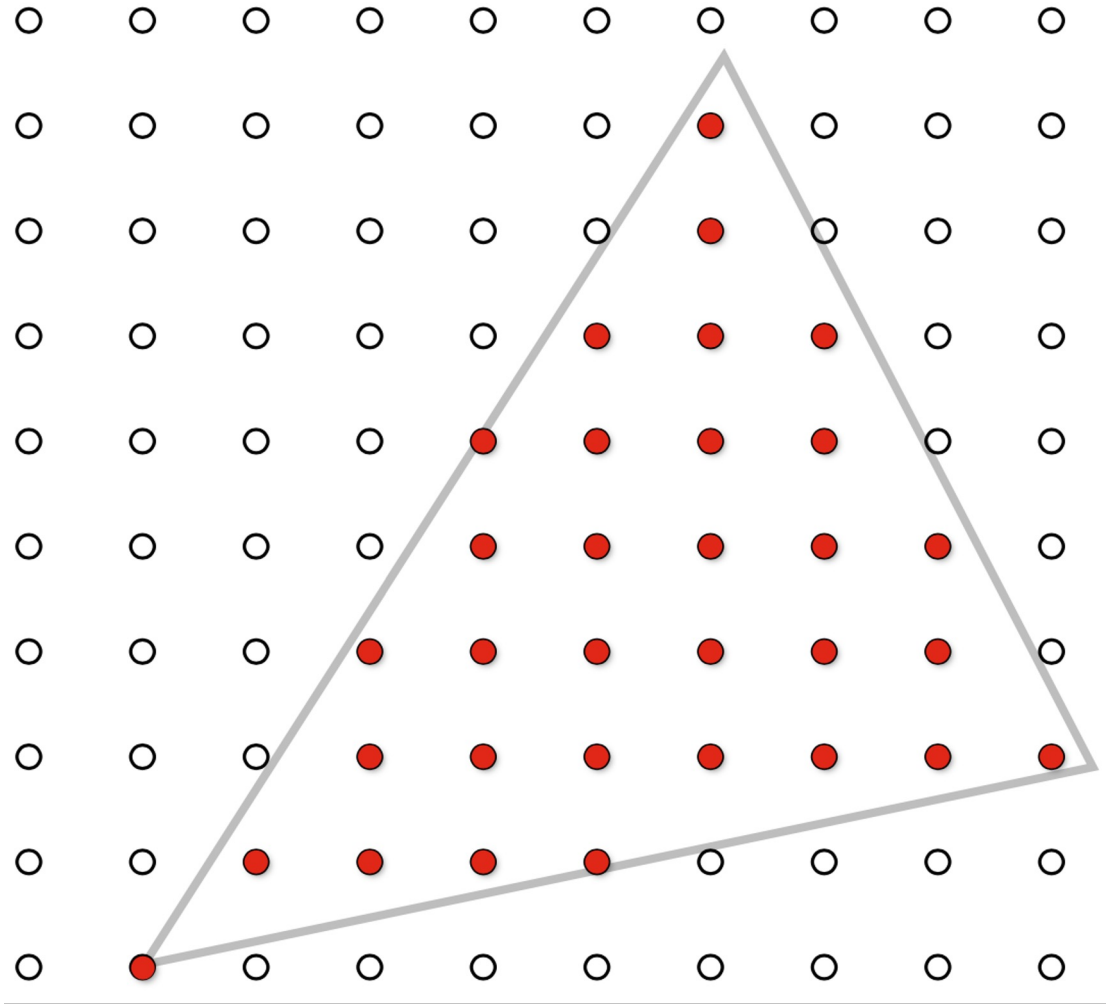
Insper

Amostras cobrindo o centro do pixel



Insper

Amostras identificando o triângulo



Insper

Rasterização

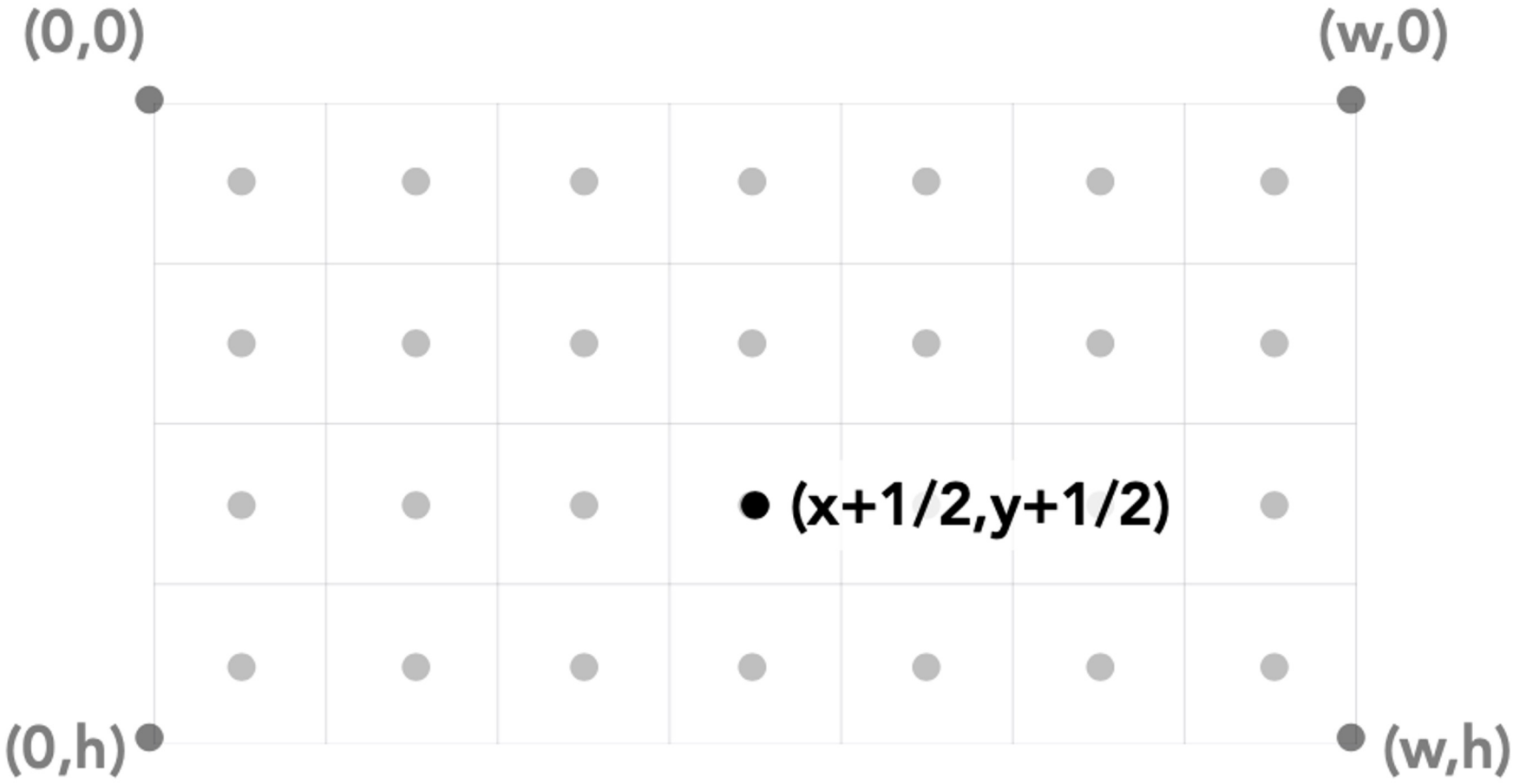
Amostrando os pixels de uma imagem

```
for (int x = 0; x < xmax; x++)
    for (int y = 0; y < ymax; y++)
        image[x][y] = f(tri, x + 0.5, y + 0.5);
```

Identifique o triângulo pela função inside()

```
f(tri, x, y) = inside(tri, x, y)
```

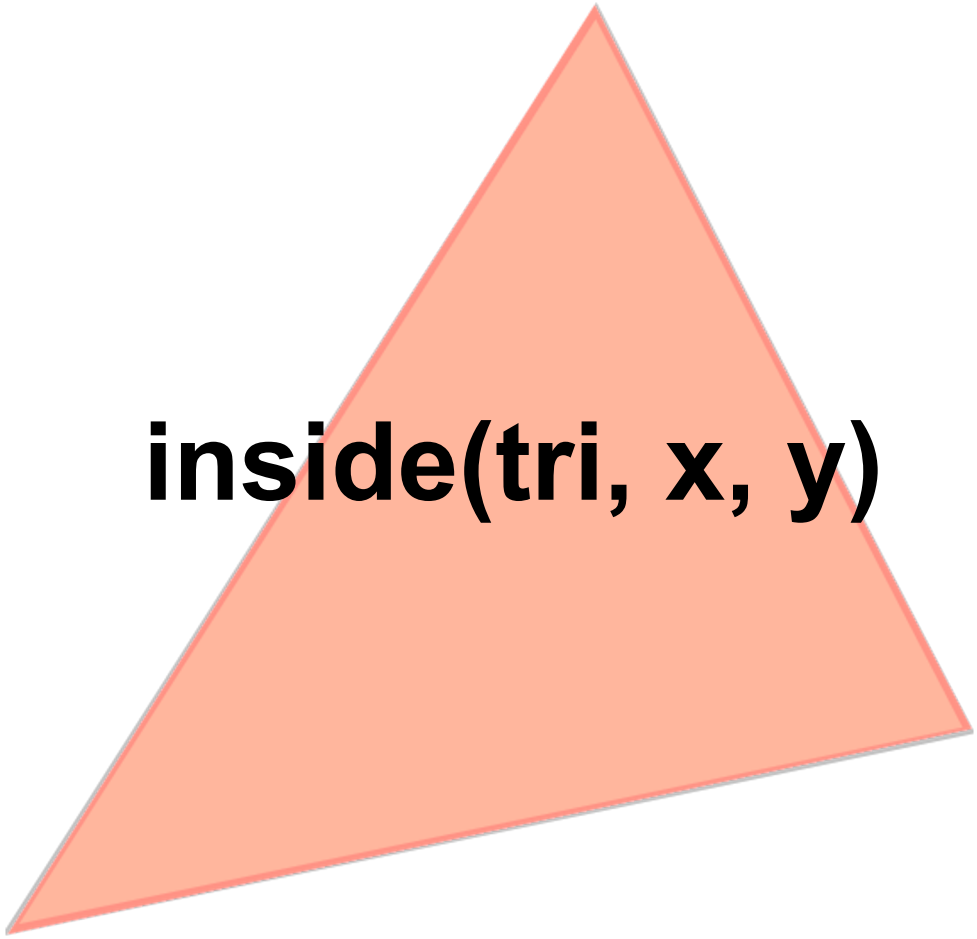
Detalhes: Localização da Amostra



Localização da amostra para pixel (x, y)

Insper

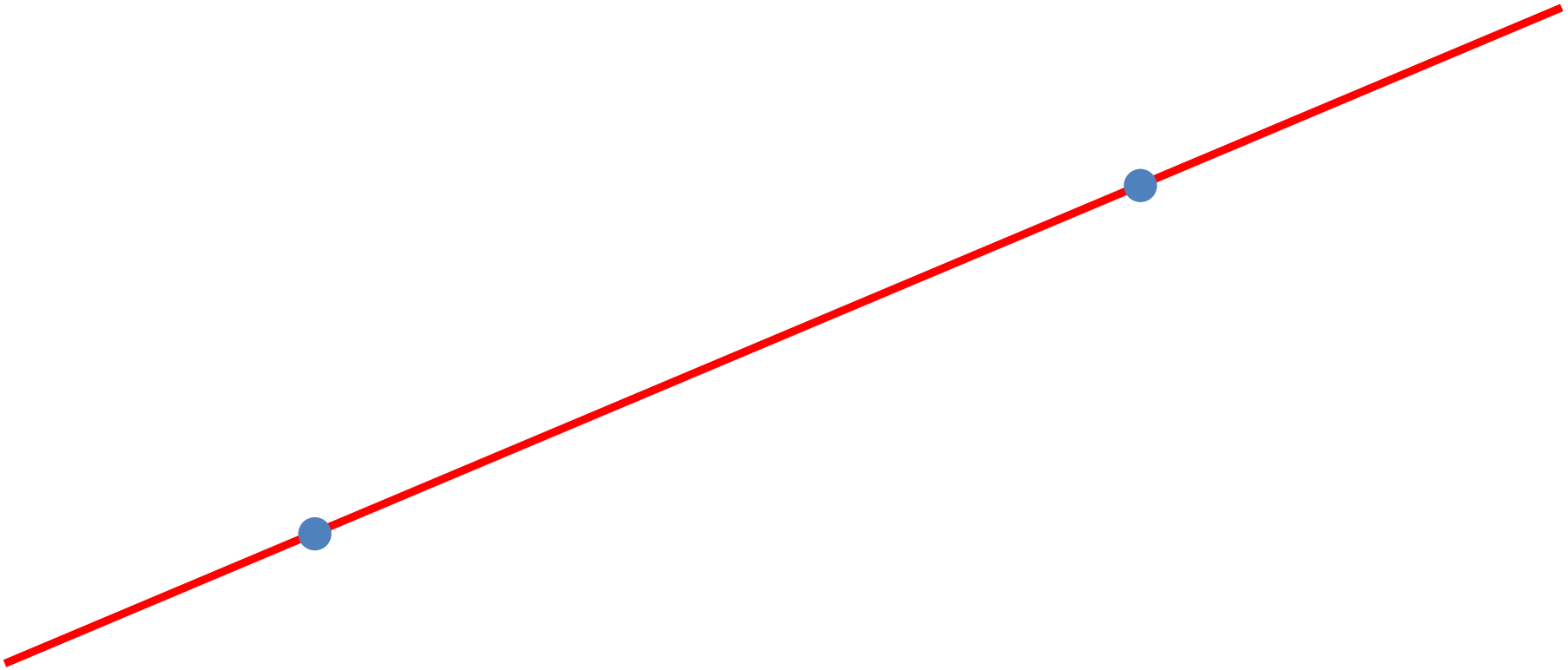
Avaliando se dentro do triângulo



Mas como saber se dentro do triângulo?
Antes disso. Como definimos um triângulo?

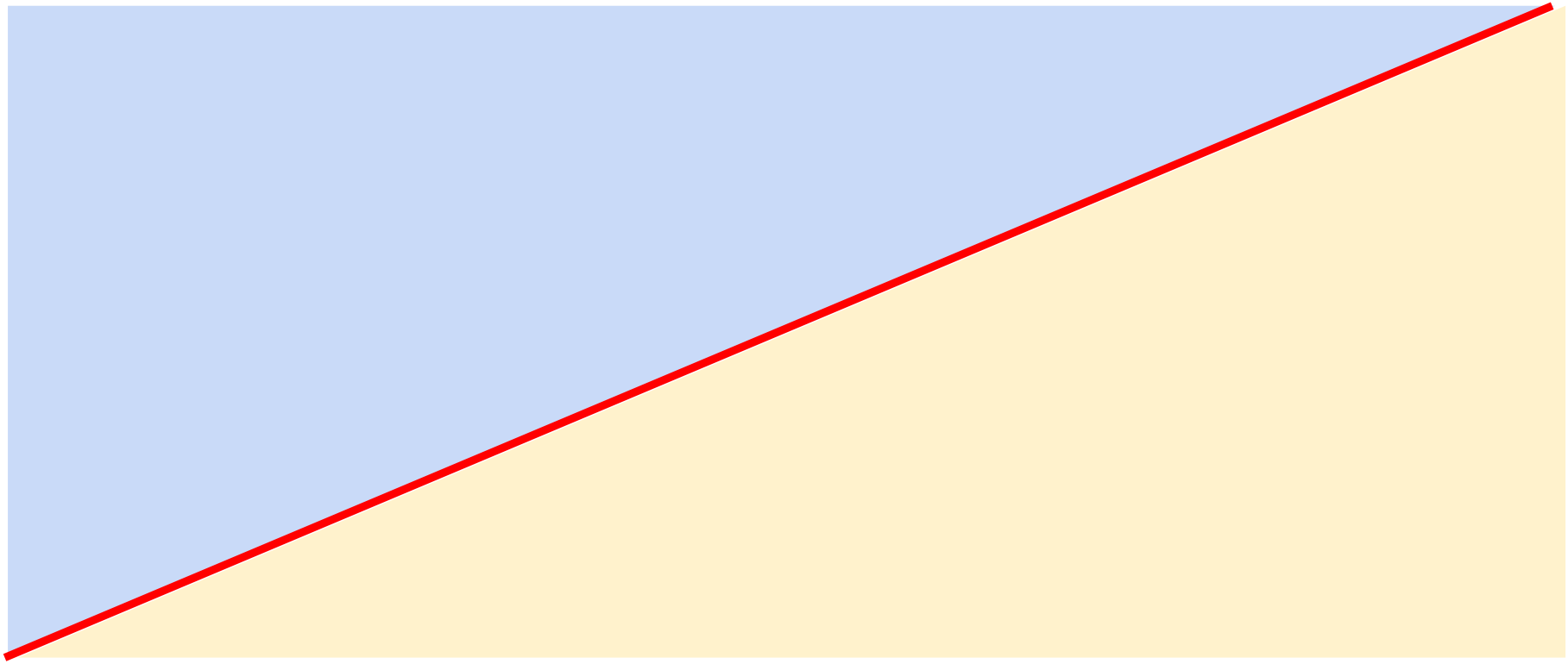
Reta

Dois pontos definem uma reta.



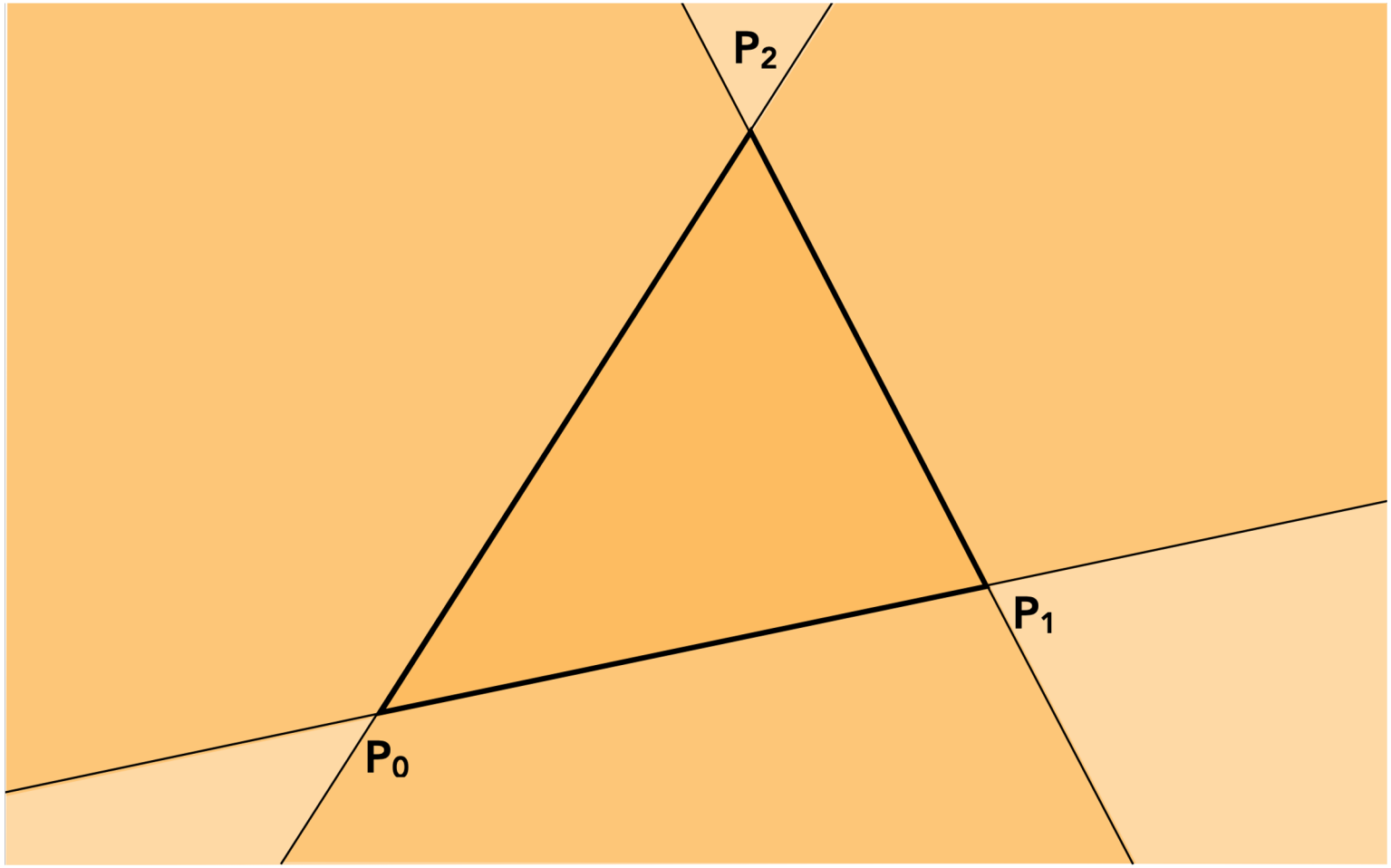
Triângulo

Toda reta divide o plano em que está contida em dois semiplanos



Triângulo

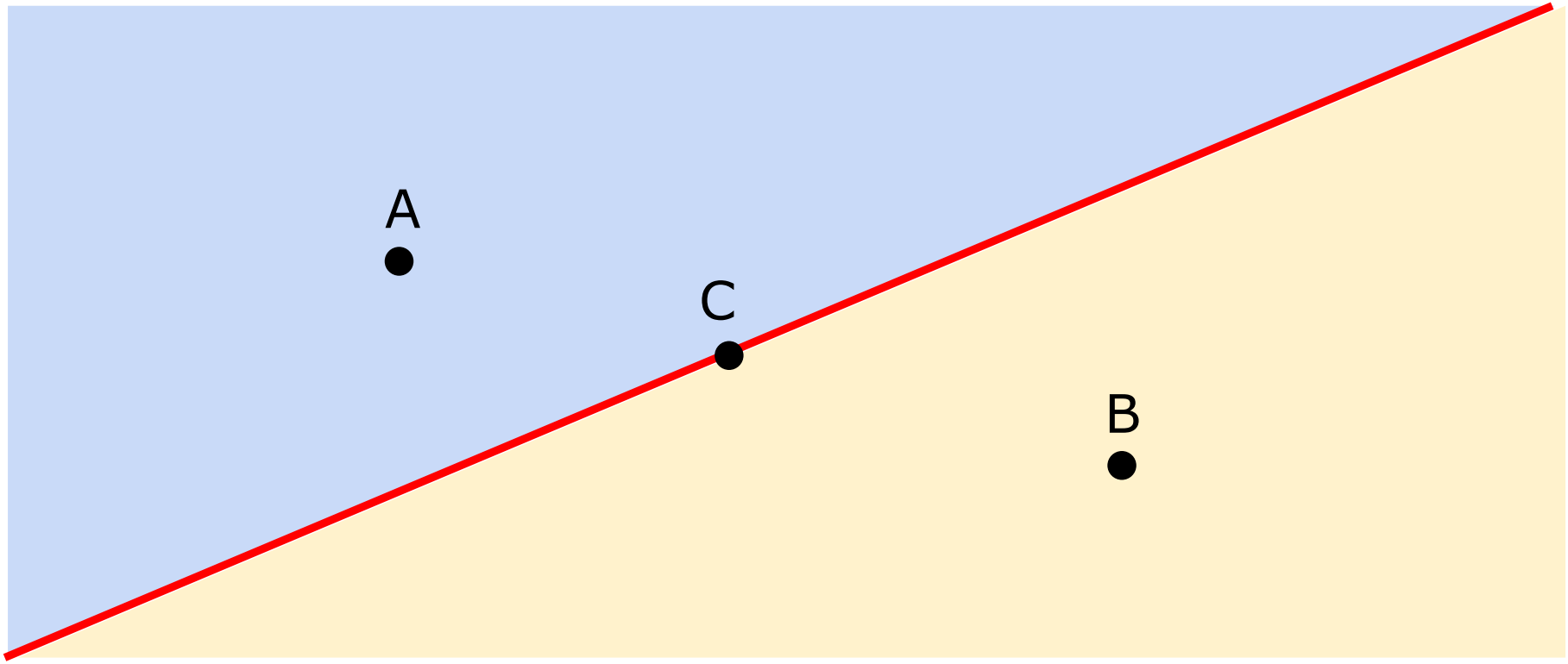
Podemos entender um triângulo como a interseção de três semiplanos



Insper

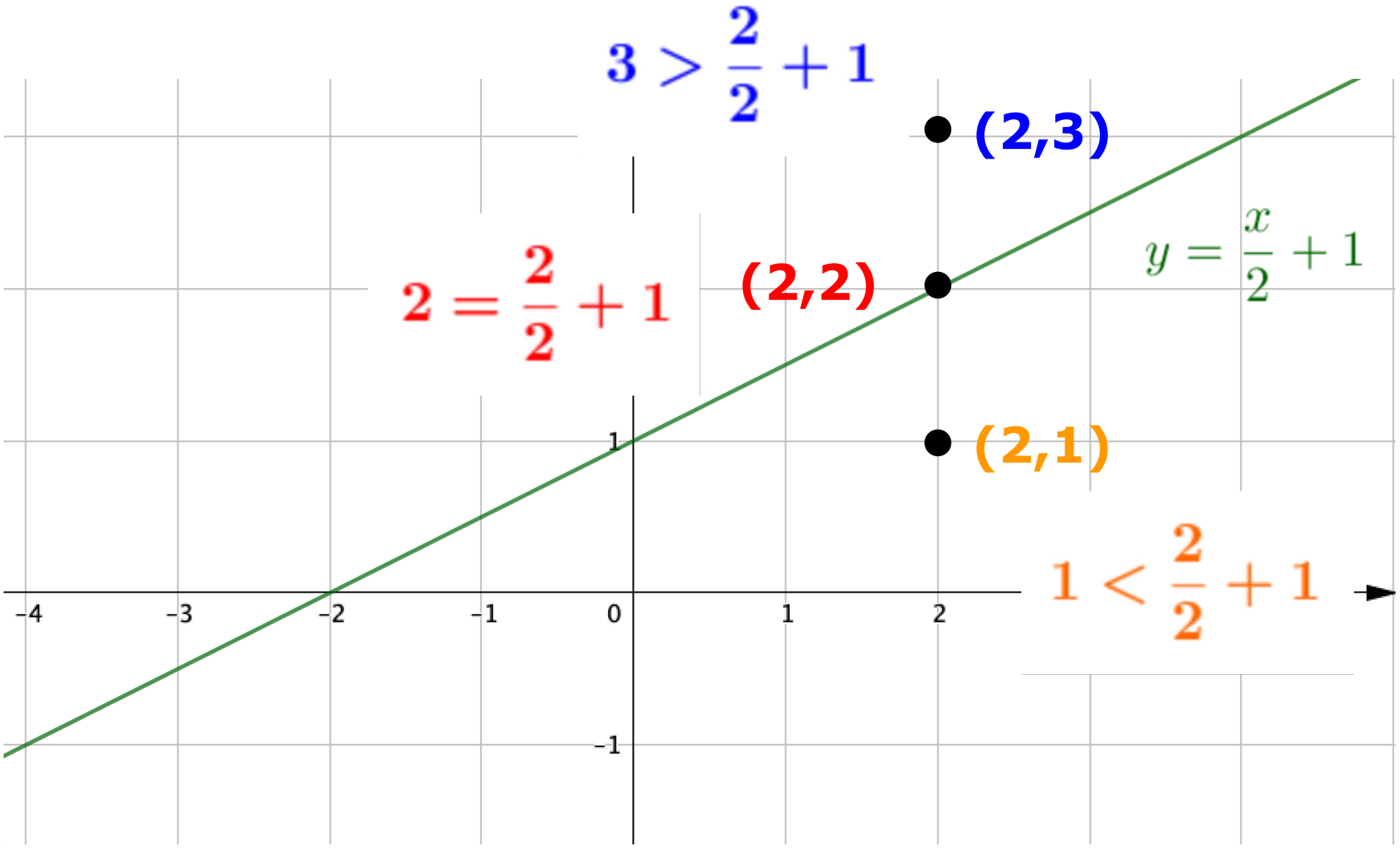
Triângulo

Como decidir em qual semiplano se encontra um ponto dado?



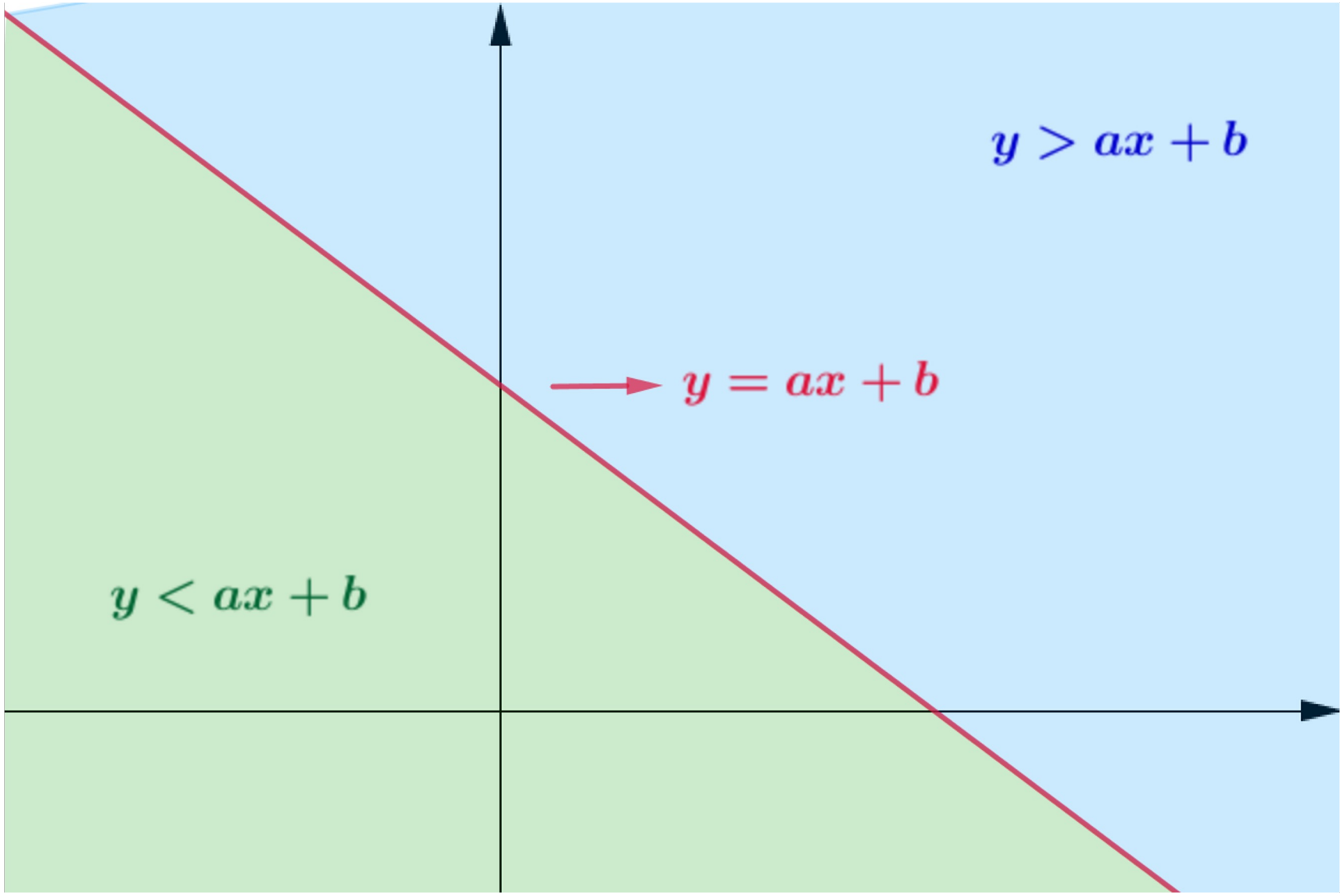
Insper

Geometria Analítica - Ensino Médio



Insper

Geometria Analítica - Ensino Médio

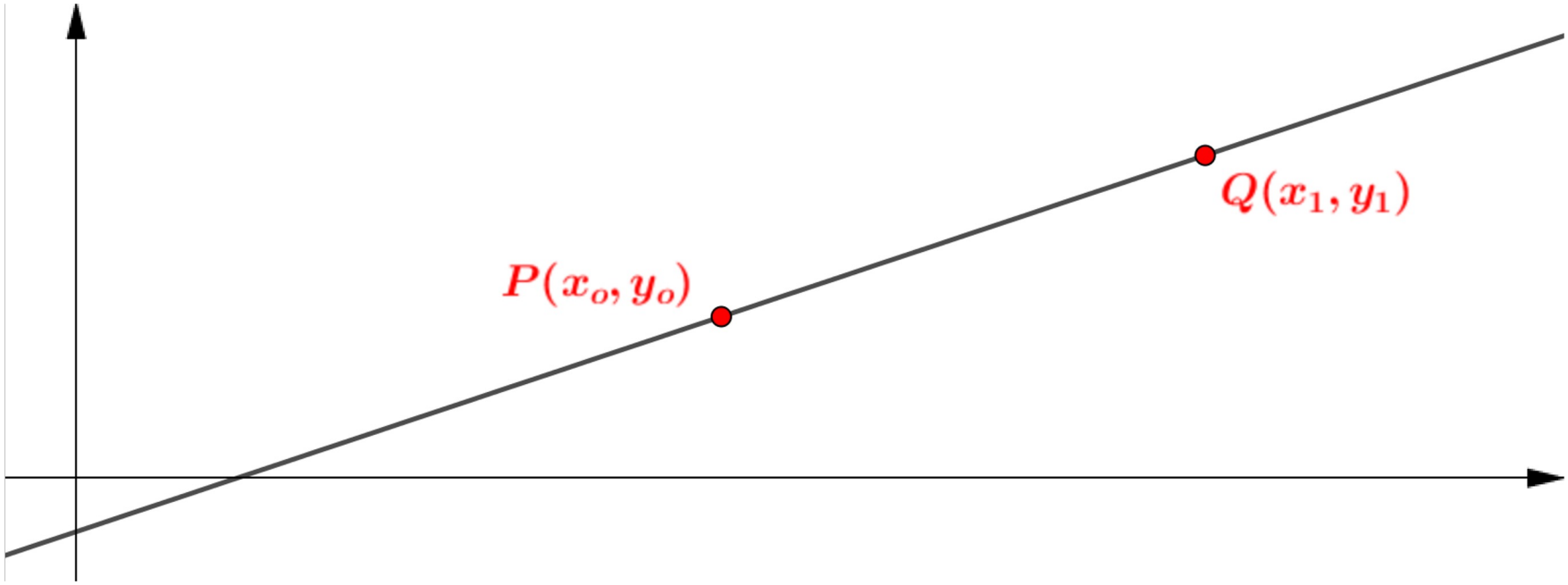


Insper

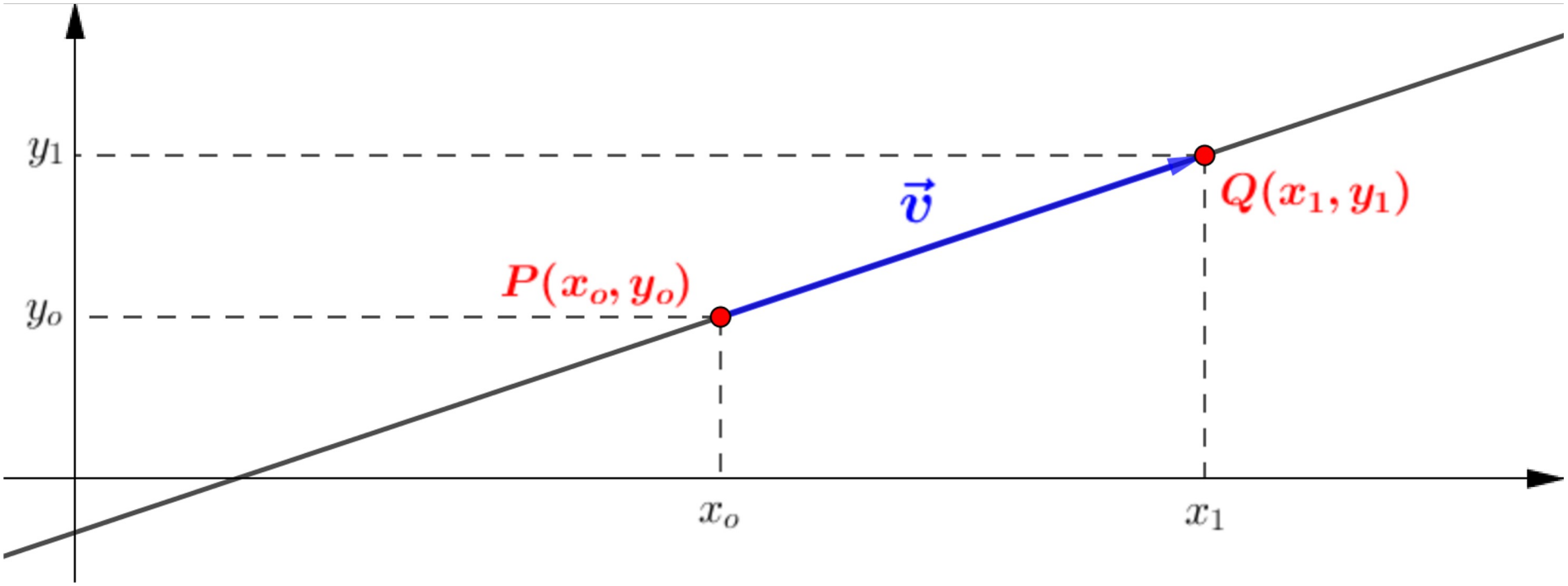
Vetores e a Equação da Reta

Neste curso, será fundamental lidar com vetores. Vamos, então, fazer uma abordagem vetorial para este problema.

Vetor diretor de uma reta



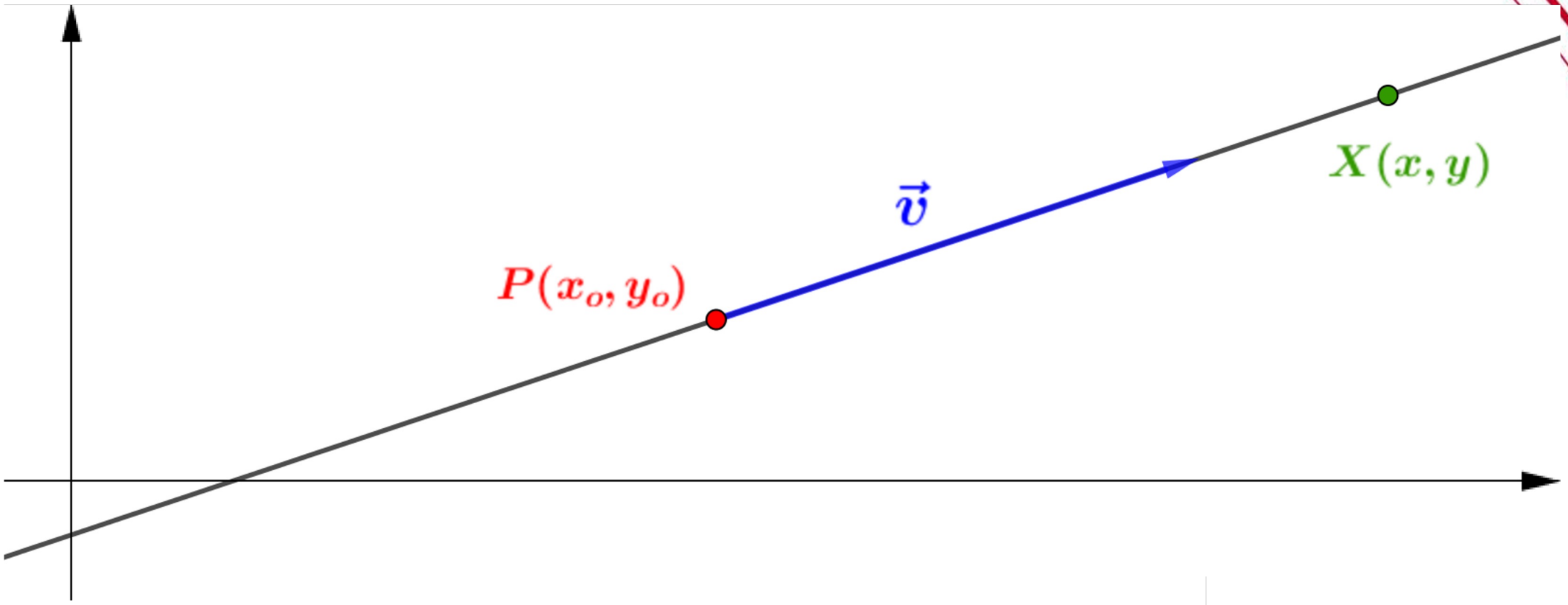
Vetores e a Equação da Reta



$$\vec{v} = (x_1 - x_0,\ y_1 - y_o)$$

Insper

Vetores e a Equação da Reta



$$X = P + \lambda \cdot \vec{v}$$

$$(x, y) = (x_o, y_o) + \lambda \cdot \vec{v}$$

Recordando o produto escalar

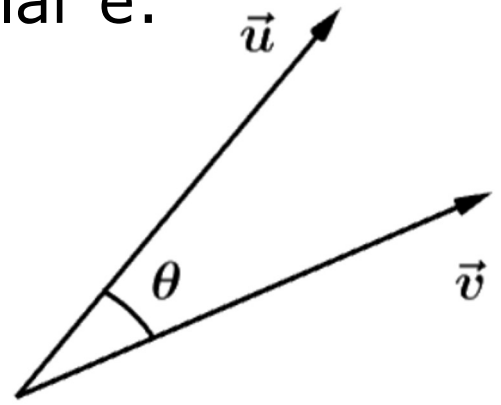
Todos se lembram disso?

O **produto escalar** (ou interno) entre dois vetores $\vec{u} = (u_1, u_2, u_3)$ e $\vec{v} = (v_1, v_2, v_3)$ é:

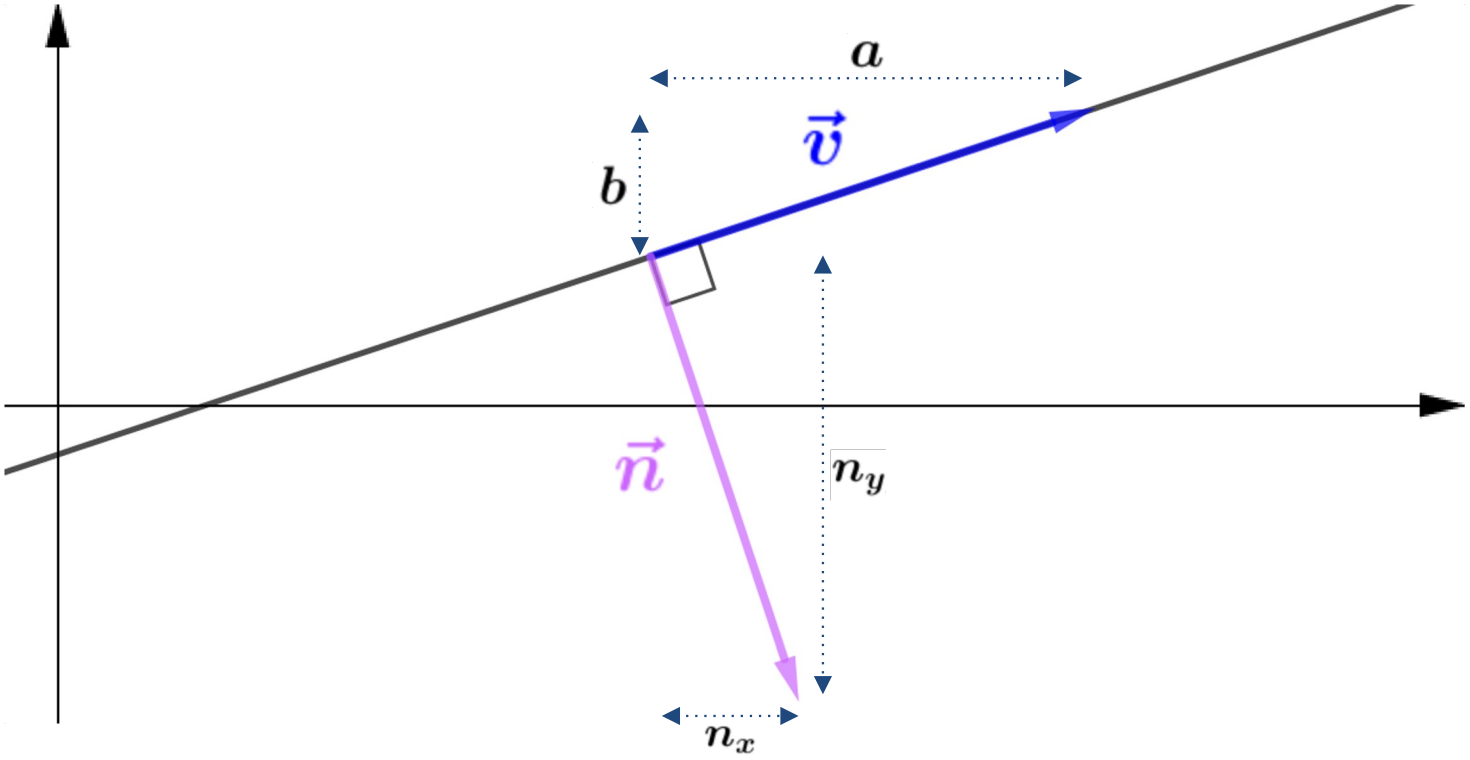
$$\boxed{\vec{u} \cdot \vec{v} = u_1 v_1 + u_2 v_2 + u_3 v_3}$$

Podemos também falar que o produto escalar é:

$$\boxed{\vec{u} \cdot \vec{v} = \|\vec{u}\| \, \|\vec{v}\| \, \cos\theta}$$



Vetor normal à reta no plano



$$\vec{v} \cdot \vec{n} = 0$$

$$(a, b) \cdot (n_x, n_y) = 0$$

Escolhendo $n_x = b$:

$$(a, b) \cdot (b, n_y) = 0$$

$$ab + b \cdot n_y = 0$$

$$n_y = -a$$

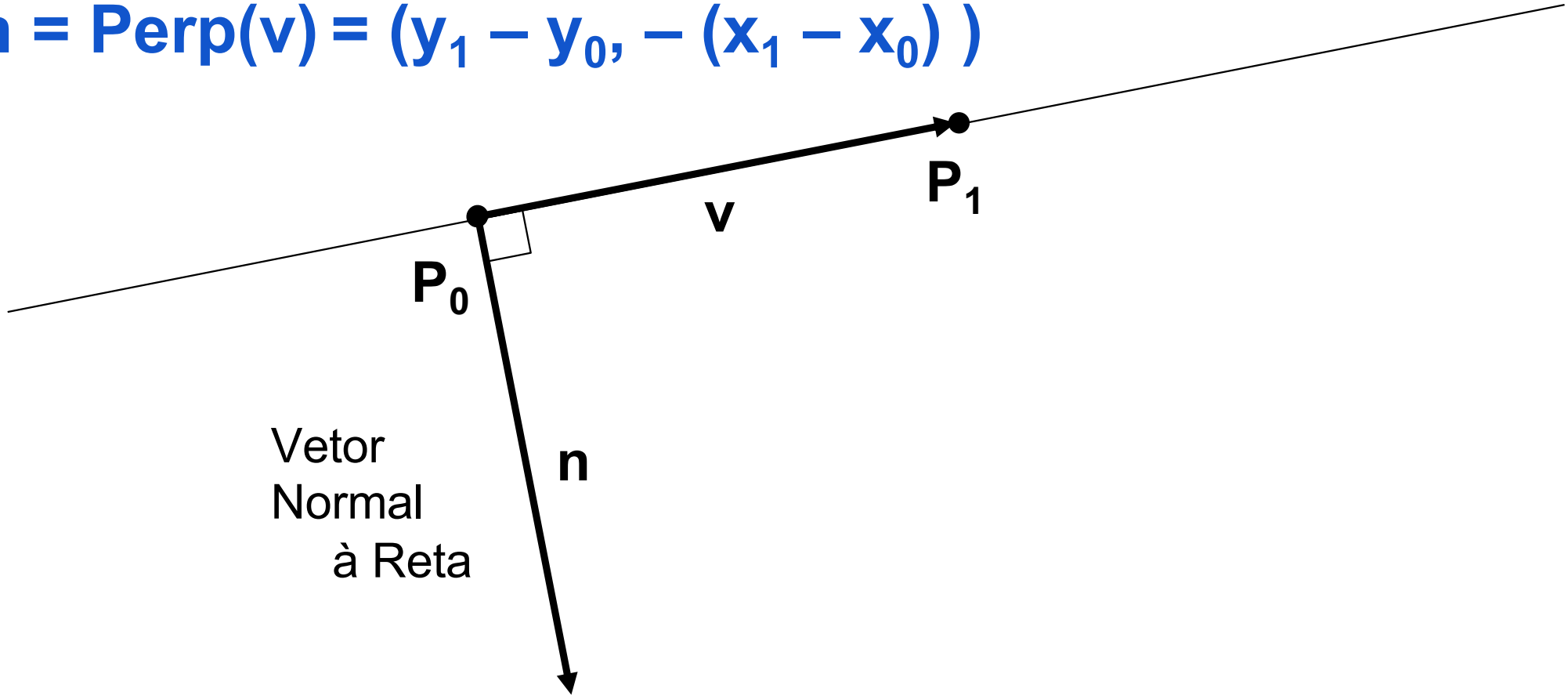
Então:

$$\vec{n} = (b, -a)$$

Trabalhando na Equação da Reta

$$v = (x_1 - x_0, y_1 - y_0)$$

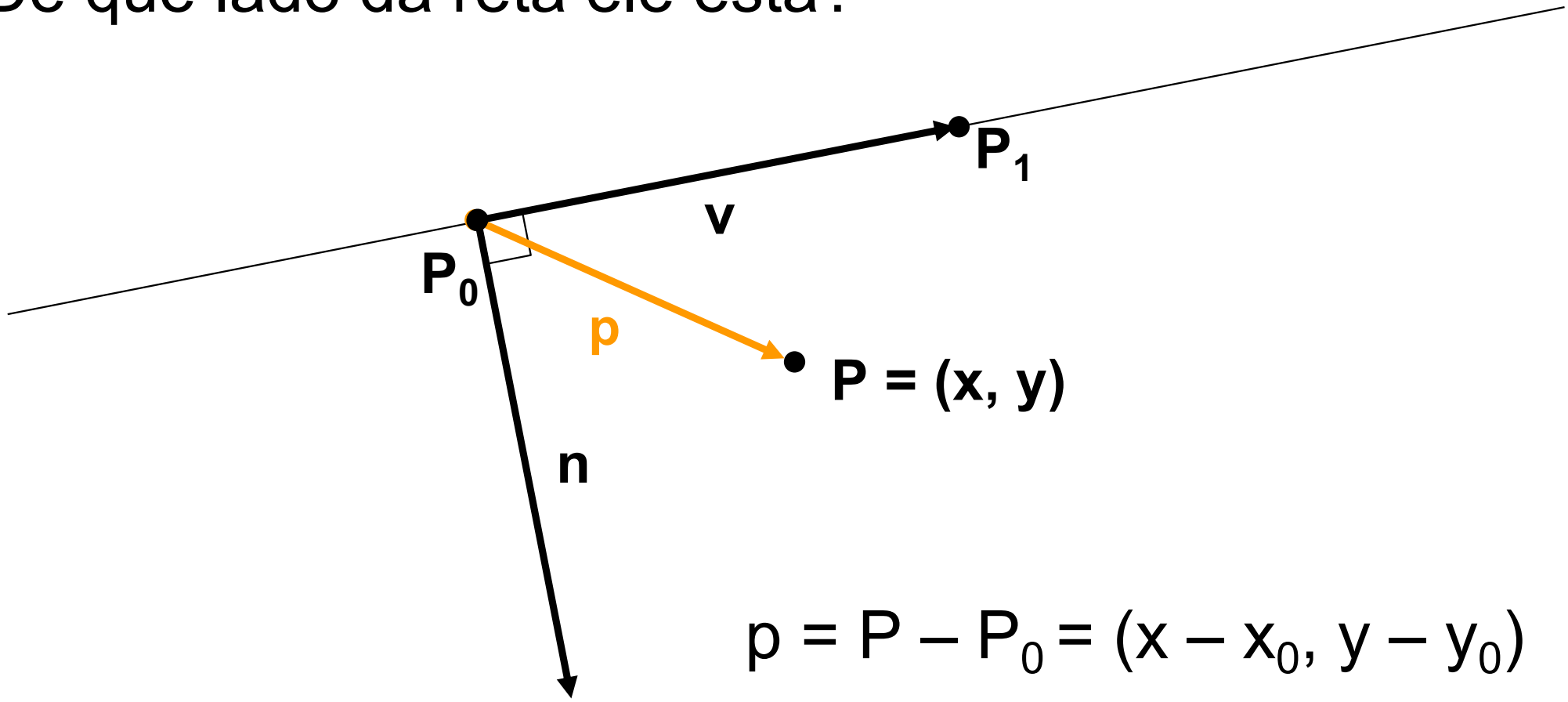
$$\mathbf{n = Perp(v) = (y_1 - y_0, -(x_1 - x_0))}$$



Insper

Trabalhando na Equação da Reta

Considere um ponto qualquer do plano, P = (x, y).
De que lado da reta ele está?

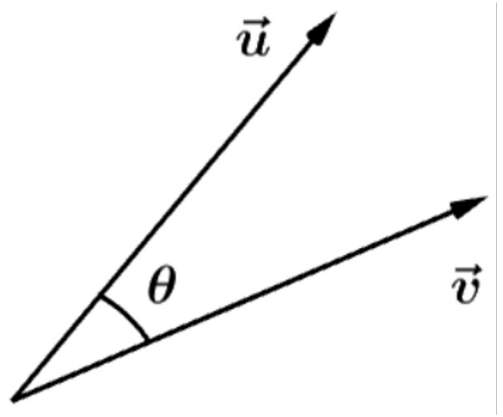


$$p = P - P_0 = (x - x_0, y - y_0)$$

Como já dissemos...

Os vetores $\vec{u}$ e $\vec{v}$ são ortogonais ($\theta = \pi/2$) se, e somente se, o produto escalar entre eles é zero:

$$\vec{u} \perp \vec{v} \Leftrightarrow \vec{u} \cdot \vec{v} = 0$$

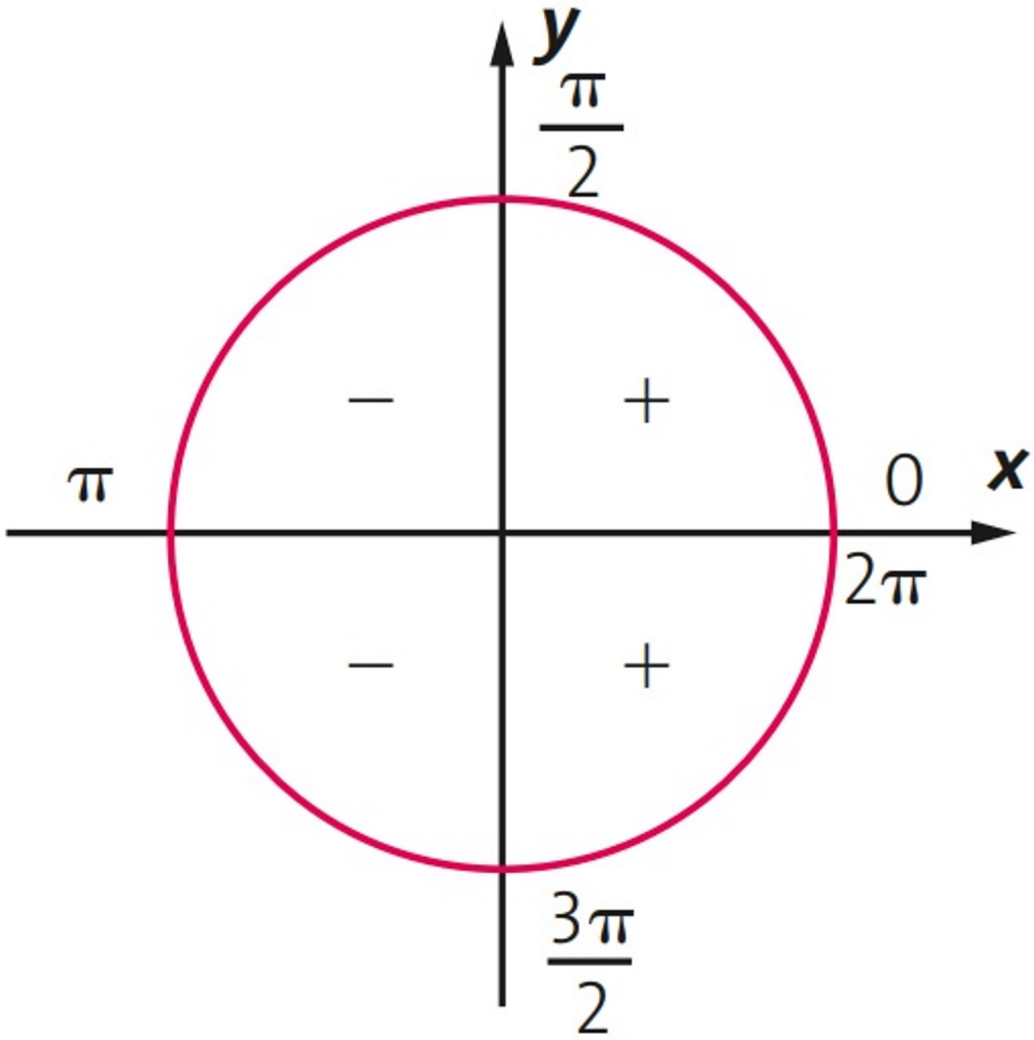


E se for positivo ou negativo?

$$\boxed{\vec{u} \cdot \vec{v} = \|\vec{u}\| \, \|\vec{v}\| \, \cos\theta}$$

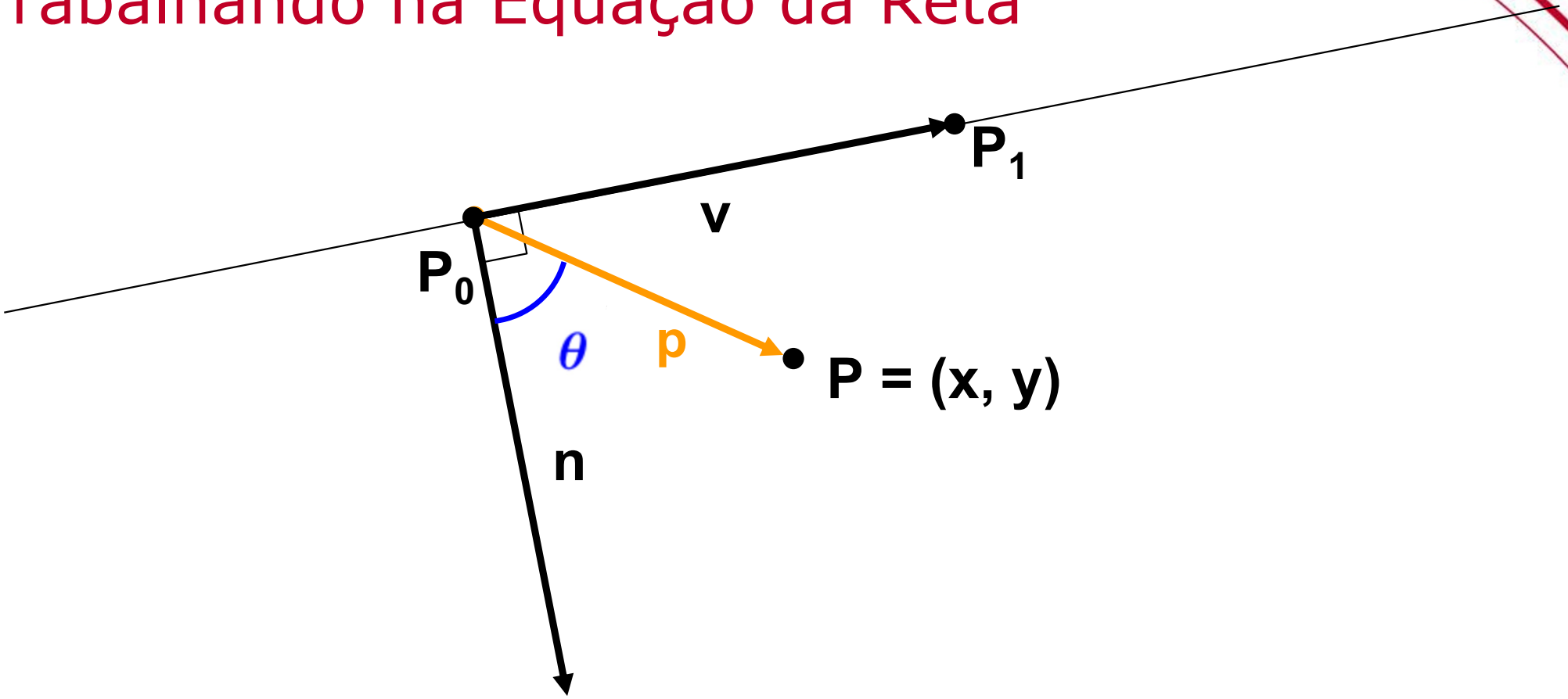
Insper

Cosseno



Insper

Trabalhando na Equação da Reta



- Se $-\pi/2 < \theta < \pi/2$, P está no mesmo semiplano que n.
- Se $\theta = \pi/2$ ou $\theta = -\pi/2$, P pertence à reta.
- Se $\pi/2 < \theta < 3 \cdot \pi/2$, P e n estão em semiplanos opostos.

Trabalhando na Equação da Reta

- Se $-\pi/2 < \theta < \pi/2$, P está no mesmo semiplano que n.

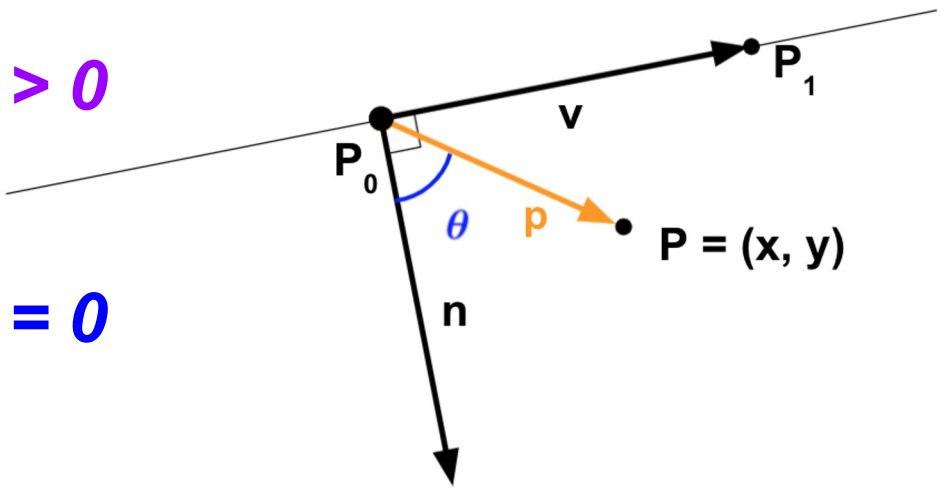
$p \bullet n > 0$

- Se $\theta = \pi/2$ ou $\theta = -\pi/2$, P pertence à reta.

$p \bullet n = 0$

- Se $\pi/2 < \theta < 3 \cdot \pi/2$, P e n estão em semiplanos opostos.

$p \bullet n < 0$



$$L(x, y) = p \bullet n = (x - x_0 ; y - y_0) \bullet (y_1 - y_0 ; -(x_1 - x_0))$$

$$= (x - x_0)(y_1 - y_0) - (y - y_0)(x_1 - x_0)$$

$$= (y_1 - y_0)x - (x_1 - x_0)y + y_0(x_1 - x_0) - x_0(y_1 - y_0)$$

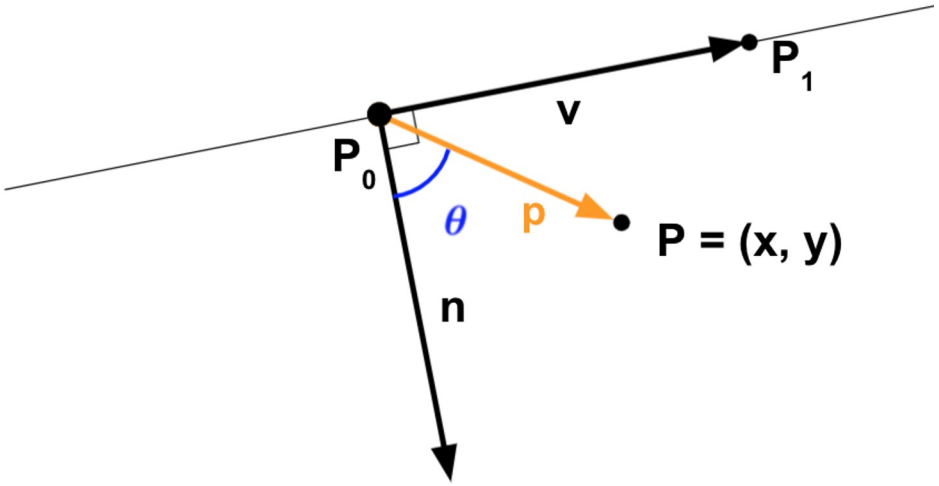
Trabalhando na Equação da Reta

$$L(x, y) = p \bullet n = (x - x_0 ; y - y_0) \bullet (y_1 - y_0 ; -(x_1 - x_0))$$

$$= (x - x_0)(y_1 - y_0) - (y - y_0)(x_1 - x_0)$$

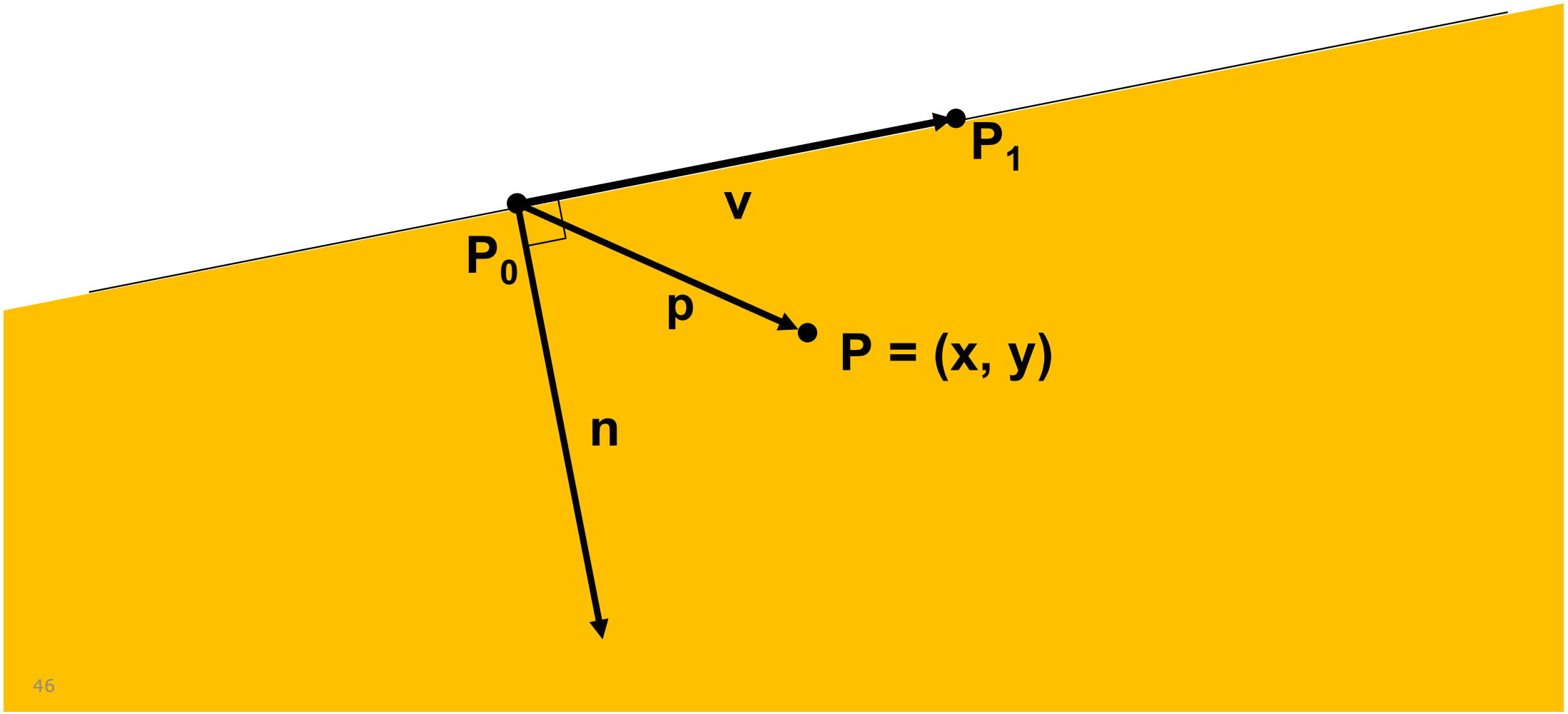
$$= (y_1 - y_0)x - (x_1 - x_0)y + y_0(x_1 - x_0) - x_0(y_1 - y_0)$$

$$L(x, y) = p \bullet n = \mathbf{Ax + By + C}$$



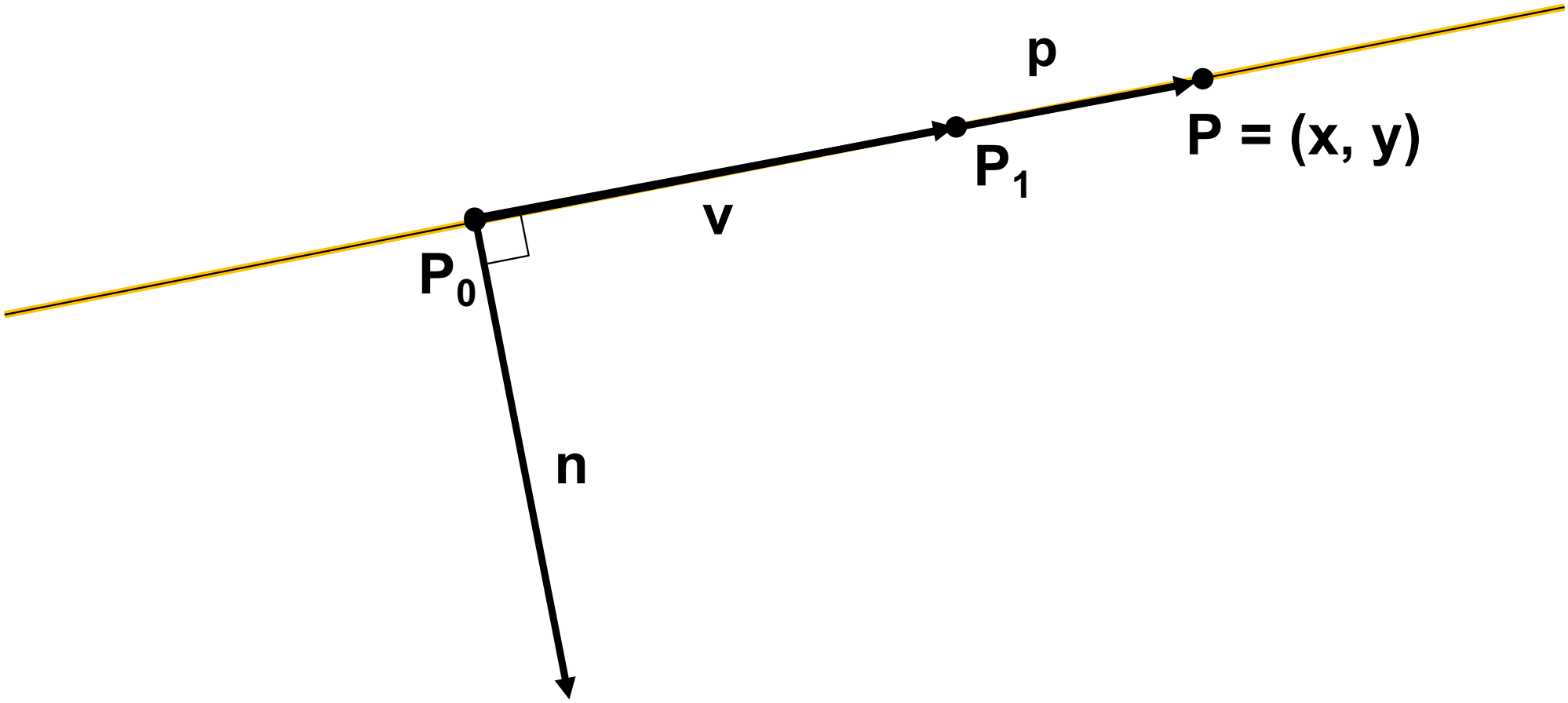
Testando o Ponto na Equação da Reta

$L(x, y) = p \bullet n > 0$



Testando o Ponto na Equação da Reta

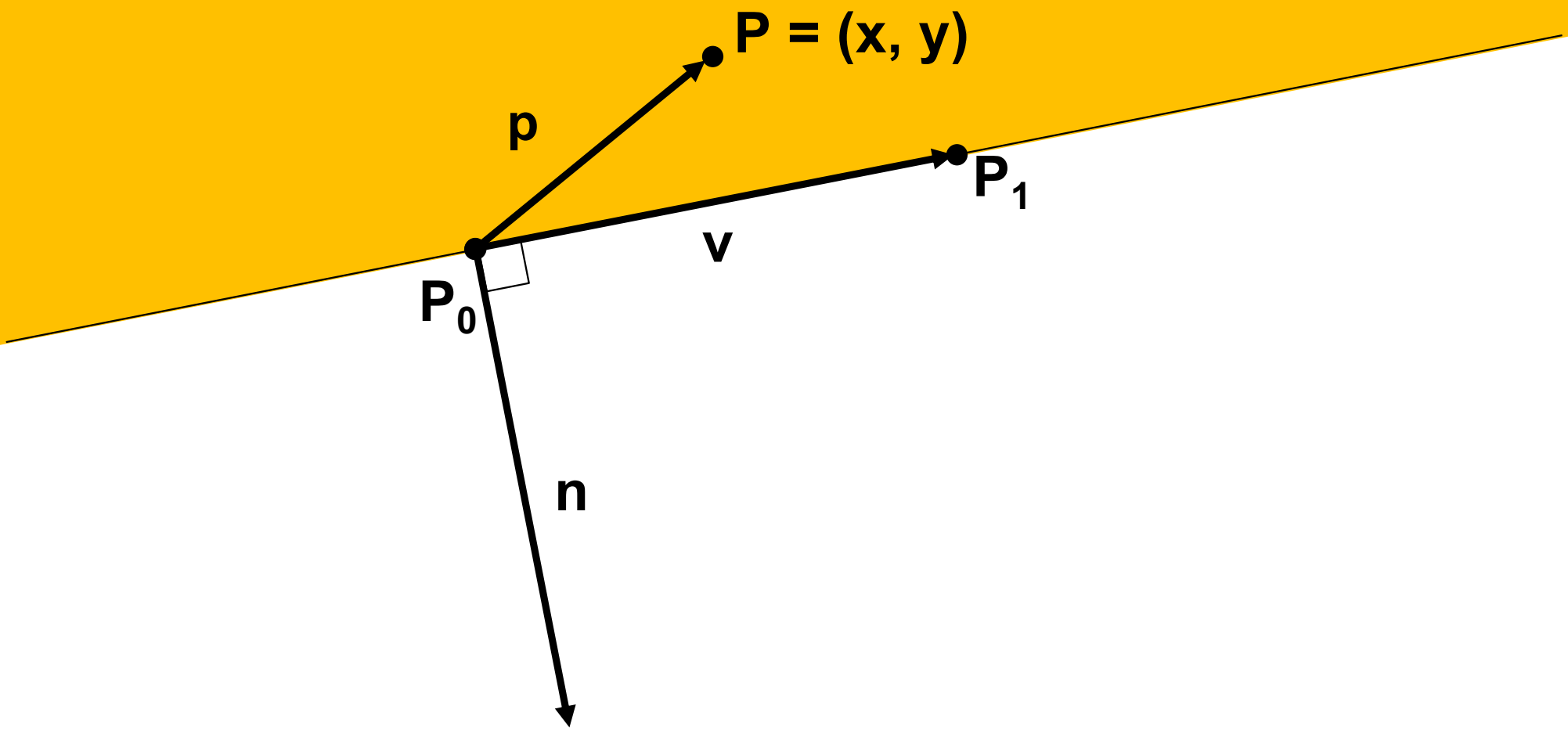
$L(x, y) = p \bullet n = 0$



Insper

Testando o Ponto na Equação da Reta

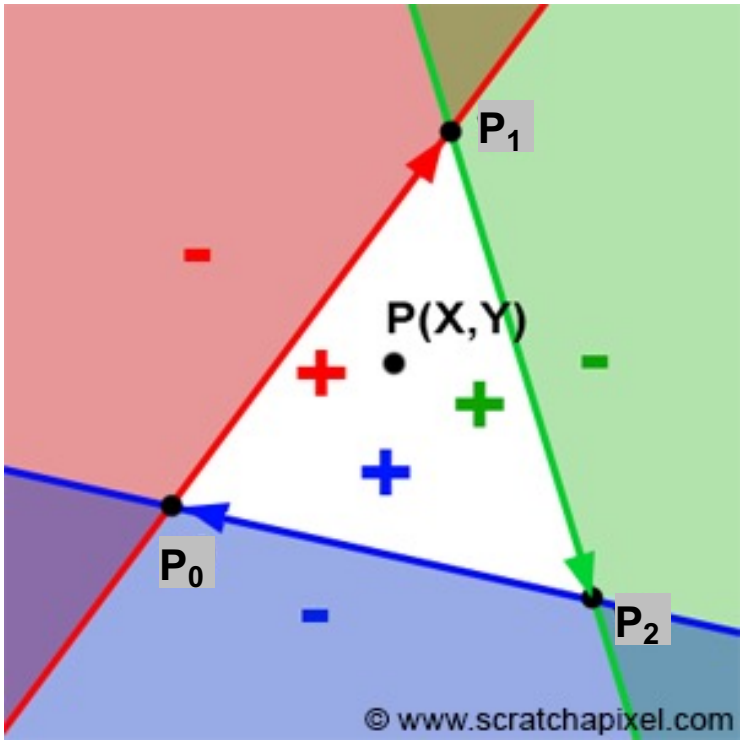
$L(x, y) = p \bullet n < 0$



Insper

Se todos os 3 testes forem positivos

Estamos dentro do triângulo



Esse teste é baseado na estratégia conhecida como Edge Function, proposto por Juan Pineda em 1988 no paper "**A Parallel Algorithm for Polygon Rasterization**".

Insper

Alternativa para organizar o teste de borda

$$L(x, y) = (x - x_0)(y_1 - y_0) - (y - y_0)(x_1 - x_0)$$

Essa equação também pode ser organizada como uma matriz:

$$\begin{bmatrix} (x - x_0) & (y - y_0) \\ (x_1 - x_0) & (y_1 - y_0) \end{bmatrix}$$

Se definirmos A = $(P - P_0)$ e B = $(P_1 - P_0)$

$$\begin{bmatrix} A.x & A.y \\ B.x & B.y \end{bmatrix}$$

O determinante dessa matriz é:

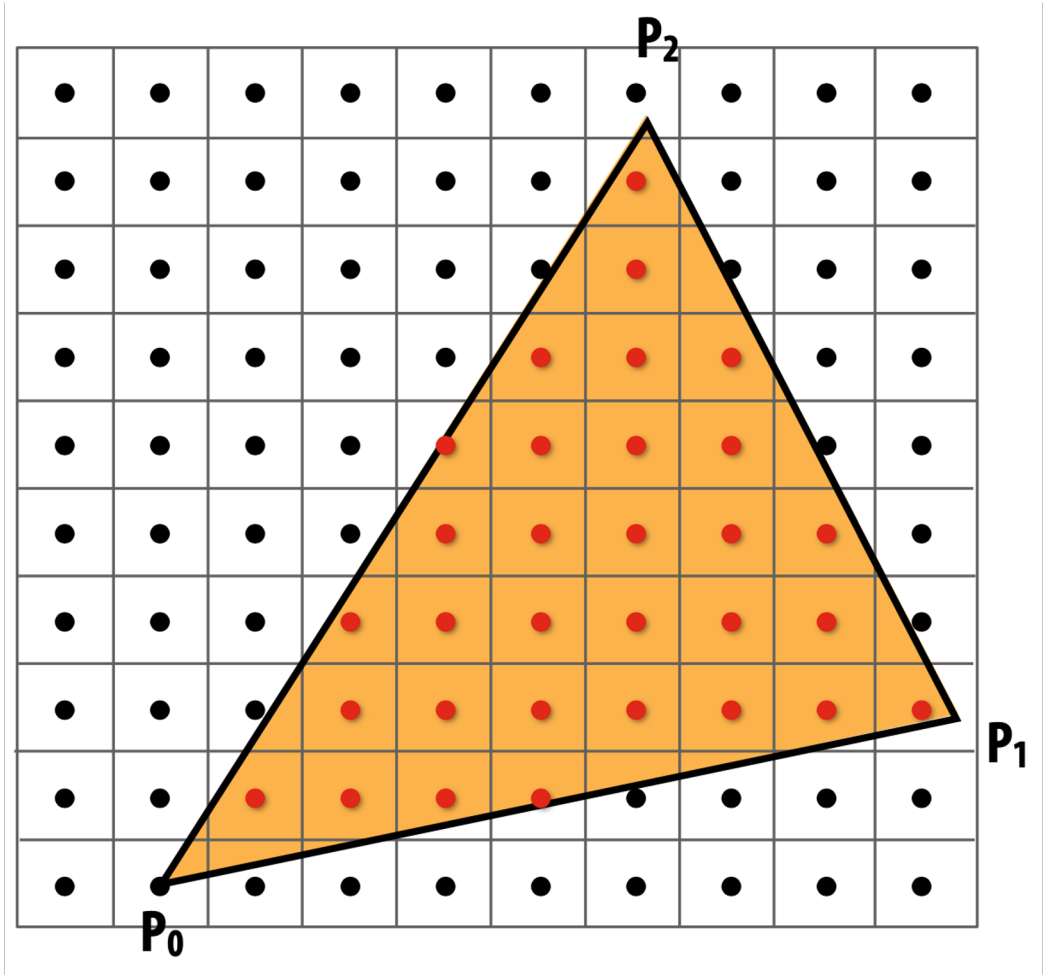
$$A.x * B.y - A.y * B.x$$

Teste do ponto no triângulo

O ponto amostrado s = (sx, sy) está dentro do triângulo se estiver dentro de todas as três arestas.

inside(*sx*, *sy*) =
$L_0(sx, sy) < 0$ &&
$L_1(sx, sy) < 0$ &&
$L_2(sx, sy) < 0$;

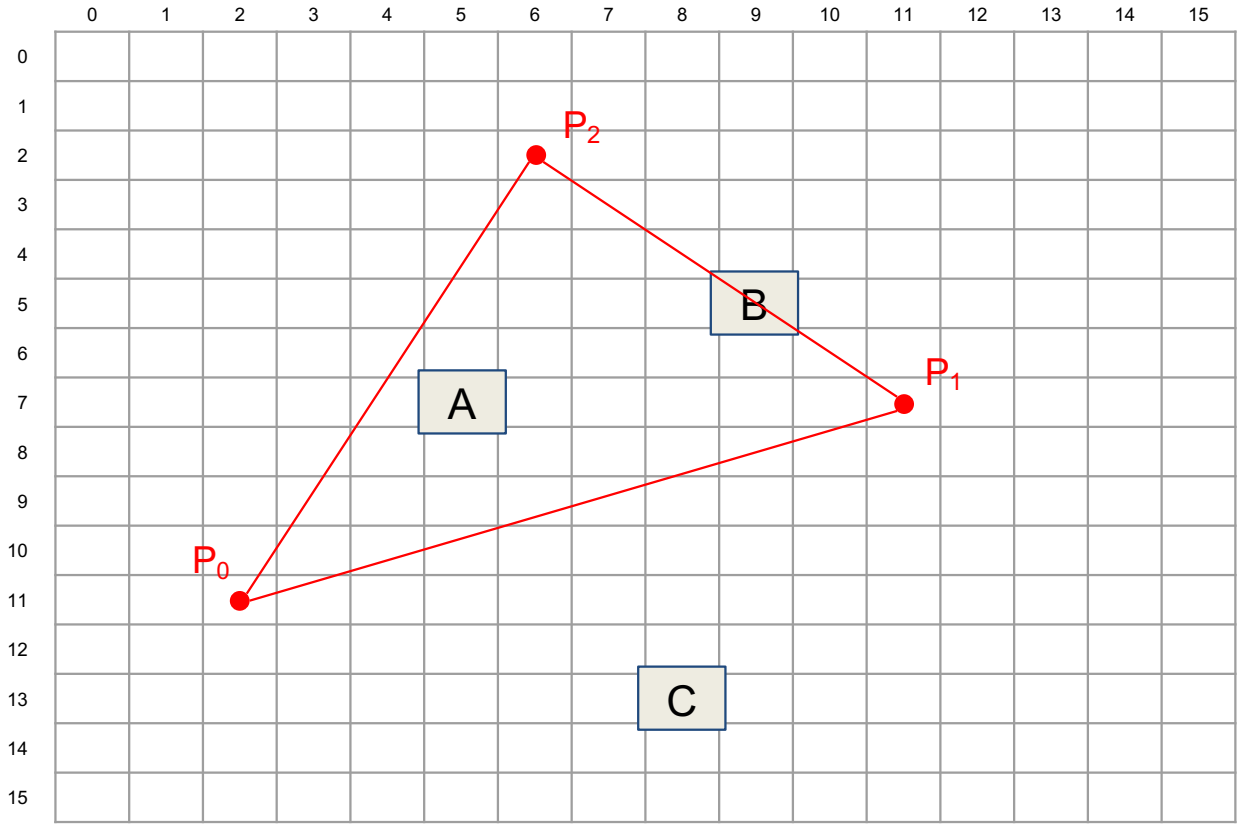
Nota: na prática não são feitos tantos testes assim, existem várias otimizações



Insper

Atividade

Identifique se o pixel está dentro ou fora do triângulo realizando os cálculos manualmente.



Obs:

1. devido o eixo y estar invertido, verifique se as equações L(x, y) são positivas.
2. considere que os pontos P_0, P_1 e P_2 estão exatamente no centro do pixel.

Insper

Resolvendo

$P_0 = (2,11)$ $P_1=(11,7)$ $P_2=(6, 2)$
$A = (5, 7)$ $B=(9, 5)$ $C=(8, 13)$

$$L(x, y) = (y_1 - y_0)x - (x_1 - x_0)y + y_0(x_1 - x_0) - x_0(y_1 - y_0)$$

$L_1(x, y) = (7 - 11)x - (11 - 2)y + 11(11 - 2) - 2(7 - 11)$
$L_1(x, y) = -4x - 9y + 107$

$L_2(x, y) = (2 - 7)x - (6 - 11)y + 7(6 - 11) - 11(2 - 7)$
$L_2(x, y) = -5x + 5y + 20$

$L_3(x, y) = (11 - 2)x - (2 - 6)y + 2(2 - 6) - 6(11 - 2)$
$L_3(x, y) = 9x + 4y - 62$

A = (5, 7) B=(9, 5) C=(8, 13)

$L_1(x, y) = -4x - 9y + 107$

$L_2(x, y) = -5x + 5y + 20$

$L_3(x, y) = 9x + 4y - 62$

A)

$inside(5, 7) = L_1(5, 7) \geq 0 \;\&\&\; L_2(5, 7) \geq 0 \;\&\&\; L_3(5, 7) \geq 0$

$inside(5, 7) = -4*5 - 9*7 + 107 \geq 0 \;\&\&\; -5*5 + 5*7 + 20 \geq 0 \;\&\&\; 9*5 + 4*7 - 62 \geq 0$

$inside(5, 7) = -20 - 63 + 107 \geq 0 \;\&\&\; -25 + 35 + 20 \geq 0 \;\&\&\; 45 + 28 - 62 \geq 0$

$inside(5, 7) = 24 \geq 0 \;\&\&\; 30 \geq 0 \;\&\&\; 11 \geq 0$

inside(5, 7) = True (dentro)

B)

$inside(9, 5) = L_1(9, 5) \geq 0 \;\&\&\; L_2(9, 5) \geq 0 \;\&\&\; L_3(9, 5) \geq 0$

$inside(9, 5) = -4*9 - 9*5 + 107 \geq 0 \;\&\&\; -5*9 + 5*5 + 20 \geq 0 \;\&\&\; 9*9 + 4*5 - 62 \geq 0$

$inside(9, 5) = -36 - 45 + 107 \geq 0 \;\&\&\; -45 + 25 + 20 \geq 0 \;\&\&\; 81 + 20 - 62 \geq 0$

$inside(9, 5) = 26 \geq 0 \;\&\&\; 0 \geq 0 \;\&\&\; 39 \geq 0$

inside(9, 5) = True (dentro, intersectando uma das arestas)

A = (5, 7) B=(9, 5) C=(8, 13)

$L_1(x, y) = -4x - 9y + 107$

$L_2(x, y) = -5x + 5y + 20$

$L_3(x, y) = 9x + 4y - 62$

C)

inside(8, 13) = $L_1(8, 13) \geq 0$ && $L_2(8, 13) \geq 0$ && $L_3(8, 13) \geq 0$

inside(8, 13) = $-4*8 - 9*13 + 107 \geq 0$ && $-5*8 + 5*13 + 20 \geq 0$ && $9*8 + 4*13 - 62 \geq 0$

inside(8, 13) = $-32 - 117 + 107 \geq 0$ && $-40 + 65 + 20 \geq 0$ && $72 + 52 - 62 \geq 0$

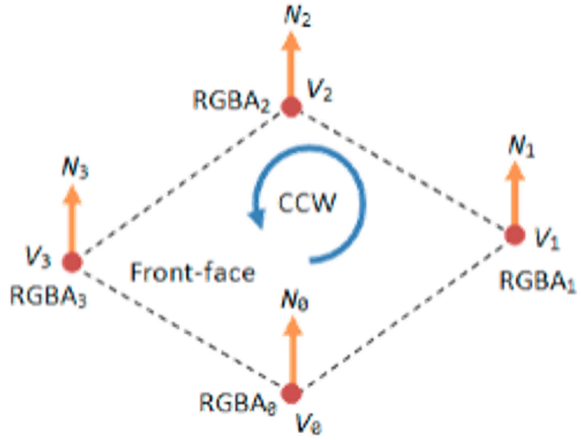
inside(8, 13) = $-42 \geq 0$ && $45 \geq 0$ && $62 \geq 0$

inside(8, 13) = False (fora)

Ordem dos Pontos

Perceba que a ordem dos pontos é importante, senão você poderá estar pegando semiplanos errados.

Ao se criar um polígono em OpenGL, a ordem padrão para se conectar os vértices é no sentido anti-horário. Isso é conhecido como a "winding order".



A estratégia de ordem pode ser alterada invocando a função:

void glFrontFace(GLenum *mode*);

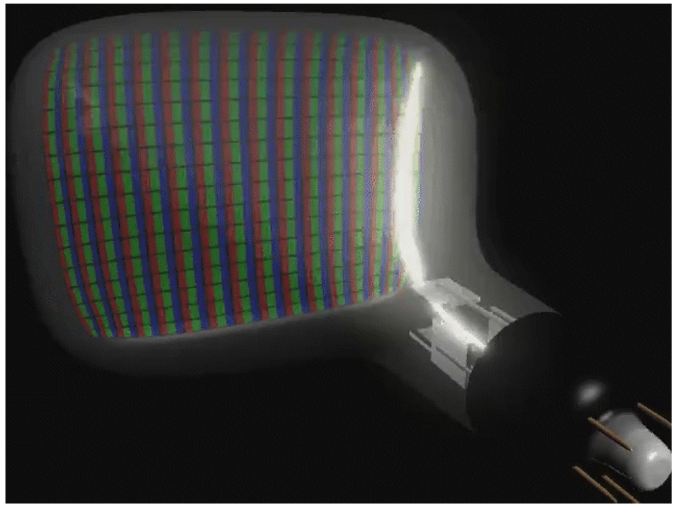
o parâmetro pode ser: GL_CW (horário) ou GL_CCW (anti-horário)

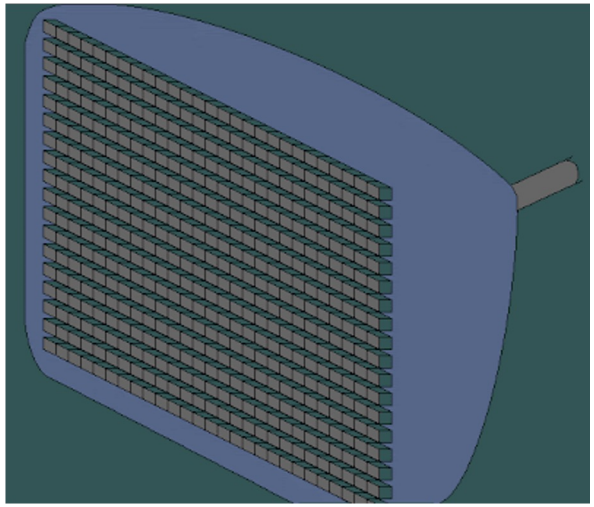
Insper

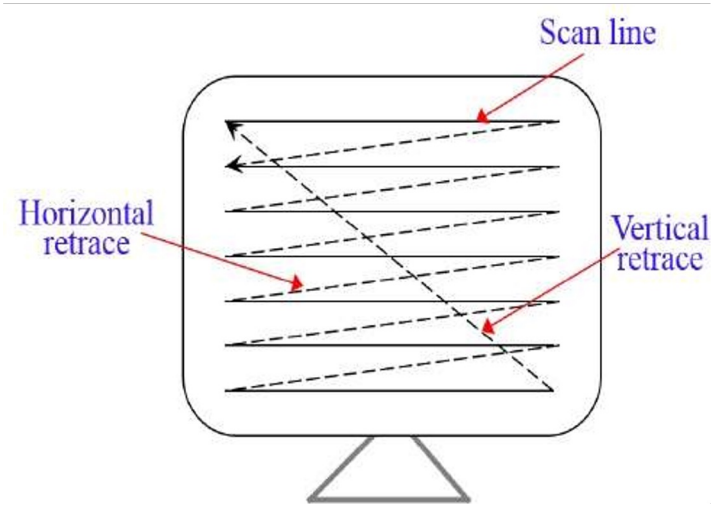
Eixos das Imagens

"However, in computer graphics and image processing one often uses a coordinate system with the y axis pointing down (as displayed on the computer's screen). This convention developed in the 1960s (or earlier) from the way that images were originally stored in display buffers."

Fonte: https://simple.wikipedia.org/wiki/User:Jeffwang/Cartesian_coordinate_system



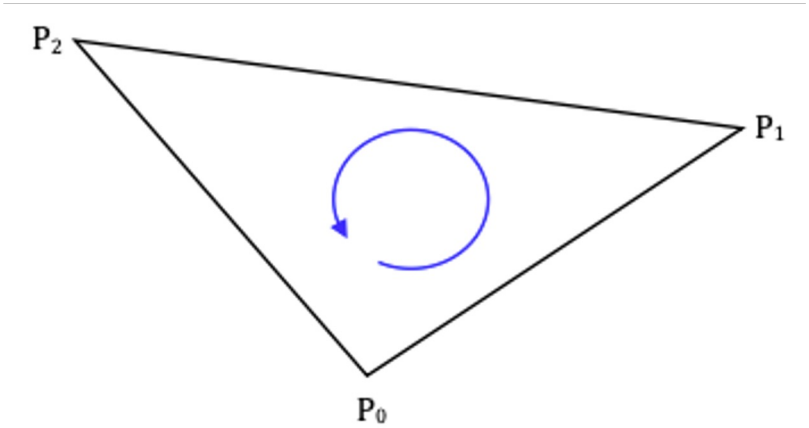




Insper

Cuidado

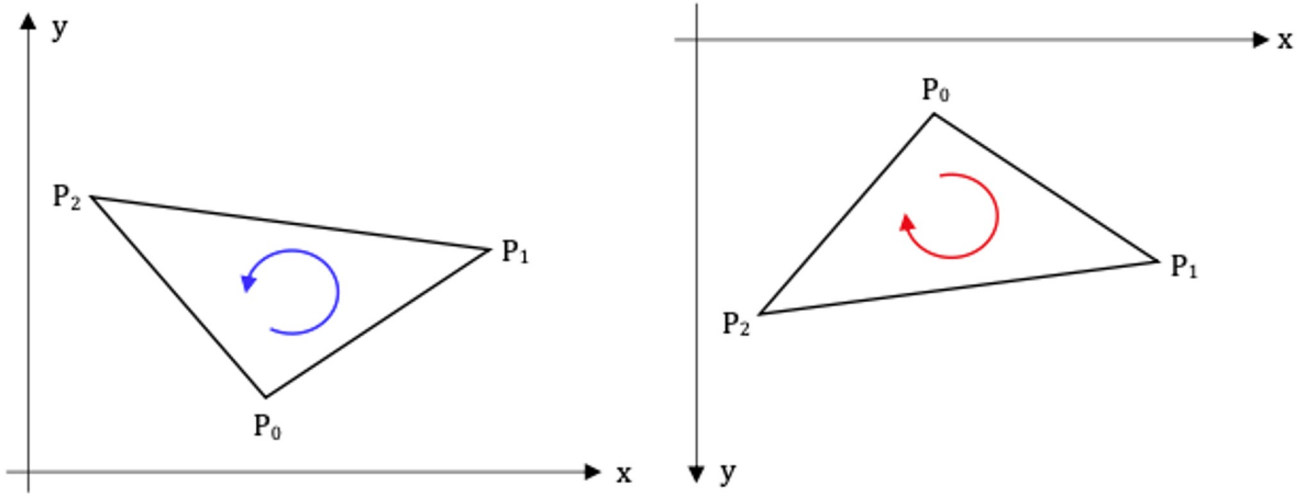
Pessoal, um ponto de atenção sobre as convenções para a ordem dos vértices do triângulo quando queremos decidir se um ponto está dentro ou fora desse triângulo. Convencionamos adotar sempre o sentido anti-horário, como ilustrado na figura abaixo.



Com isso, podemos afirmar que um ponto pertence ao interior do triângulo quando os valores numéricos das três expressões Li(x, y) são negativos. Chegamos a essa conclusão adotando a orientação convencional do plano cartesiano, ou seja, o eixo x para a direita e o eixo y para cima.

Invertendo

No exercício e também no projeto a ser entregue, usamos o sistema de coordenadas usual do sistema de pixels, ou seja, com o eixo x orientado para a direita, mas com o eixo y orientado para baixo. Observe o que acontece com o mesmo triângulo representado nesses dois sistemas.

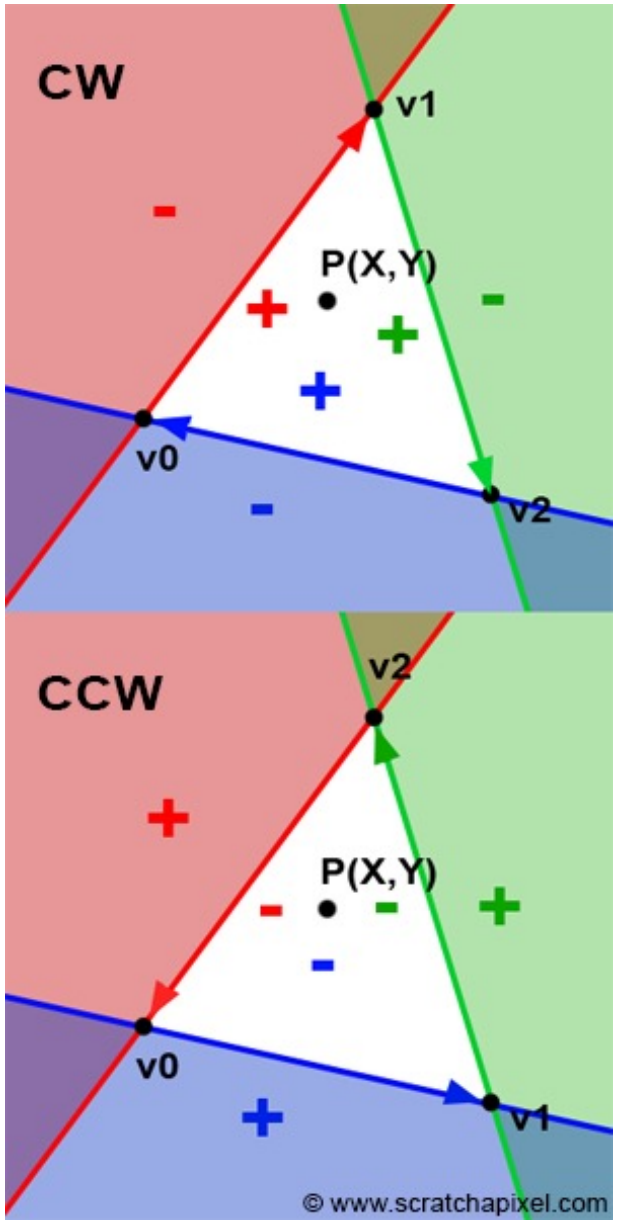


Isso mesmo! O sentido do polígono inverte, ou sejam, fica alterado de anti-horários para horário e vice-versa.

Insper

E o que fazer?

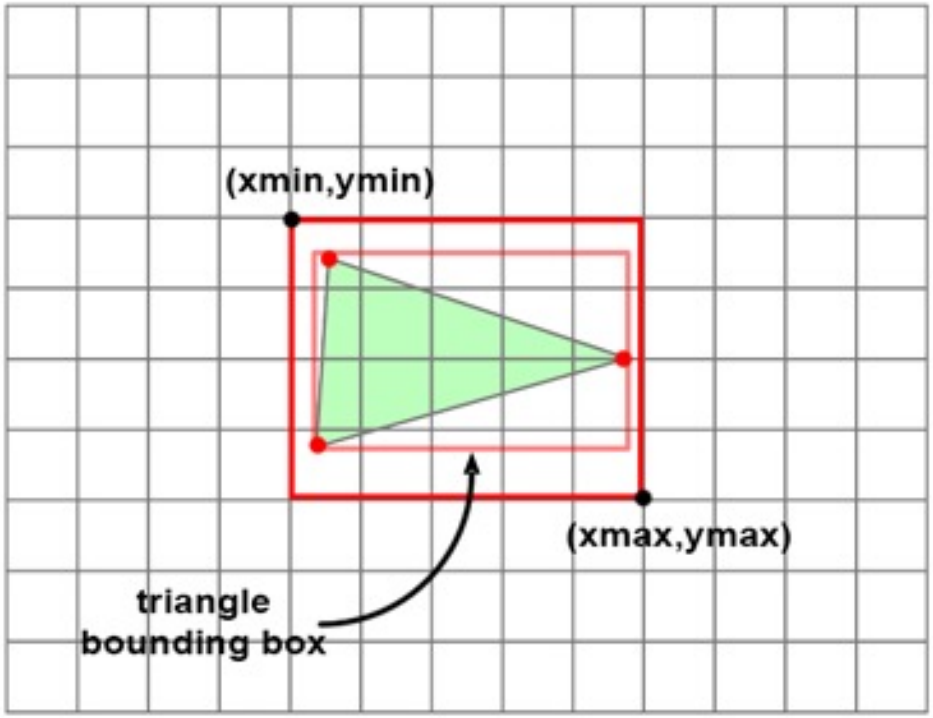
Calma, tudo continua valendo da mesma forma, só que os sinais das três expressões Li(x, y) ficarão invertidos. Então, usando esse sistema de coordenadas com o eixo y apontando para baixo, podemos afirmar que um ponto pertence ao interior do triângulo quando os valores numéricos das três expressões Li(x, y) são positivos. Fiquem atentos a essa questão e bom trabalho!



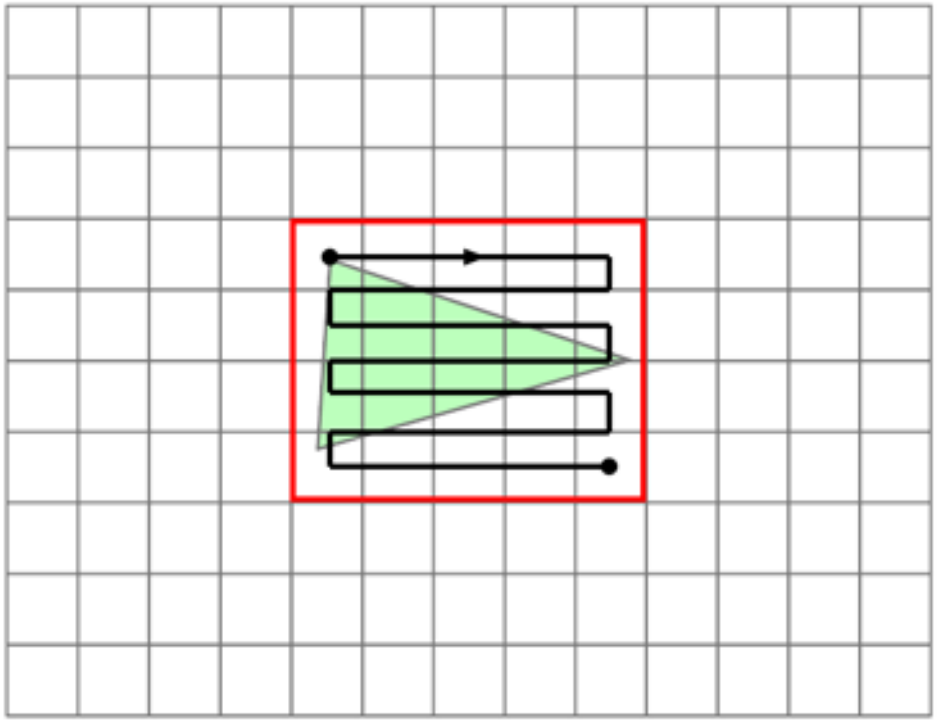
Insper

Otimizações: Bounding Box de Triângulos 2D

Para evitar a iteração em todos os pixels da imagem, podemos iterar em todos os pixels contidos na caixa delimitadora (Bounding Box) do triângulo 2D.



© www.scratchapixel.com



© www.scratchapixel.com

Insper

Otimizações: Traversal incremental

$P_i = (X_i, Y_i)$

$dX_i = X_{i+1} - X_i$

$dY_i = Y_{i+1} - Y_i$

$L_i(x, y) = (x - X_i)dY_i - (y - Y_i)dX_i$

$= A_i x + B_i y + C_i$

$L_i(x, y) = 0$: ponto na borda

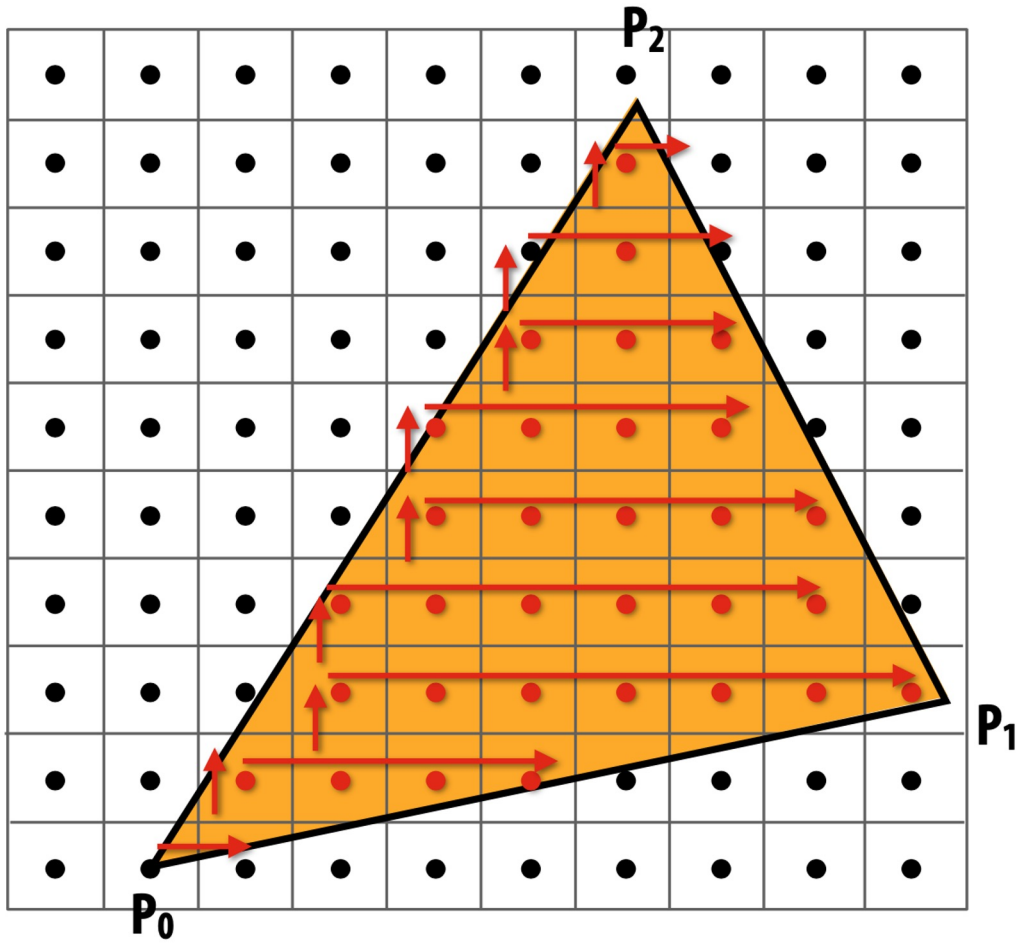
> 0 : fora da borda

< 0 : dentro da borda

Otimizações do Algoritmo:

$L_i(x+1, y) = L_i(x, y) + dY_i = L_i(x, y) + A_i$

$L_i(x, y+1) = L_i(x, y) + dX_i = L_i(x, y) + B_i$



Muitas formas de Traversal existem como: **backtrack, zig-zag, curvas Hilbert/Morton**

Traversal de Triângulos

Traversal de triângulos é implementado nas GPUs modernas.
Um artigo descrevendo o funcionamento se encontra em:
http://oa.upm.es/9184/1/INVE_MEM_2010_84947.pdf

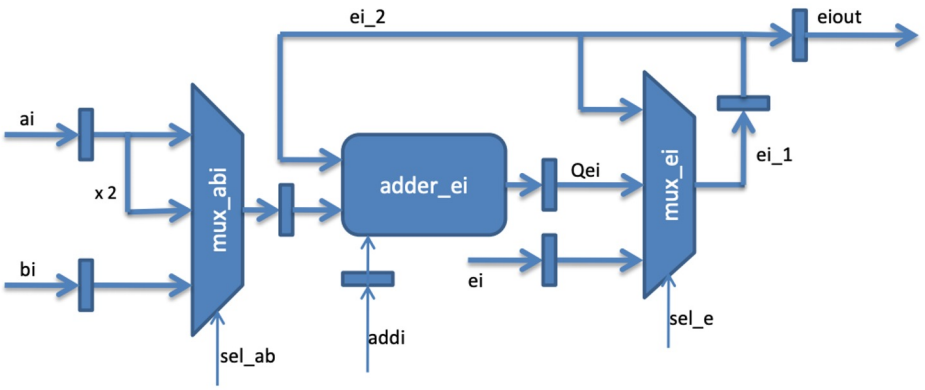


Fig. 6. Full schematic of an e_i adder.

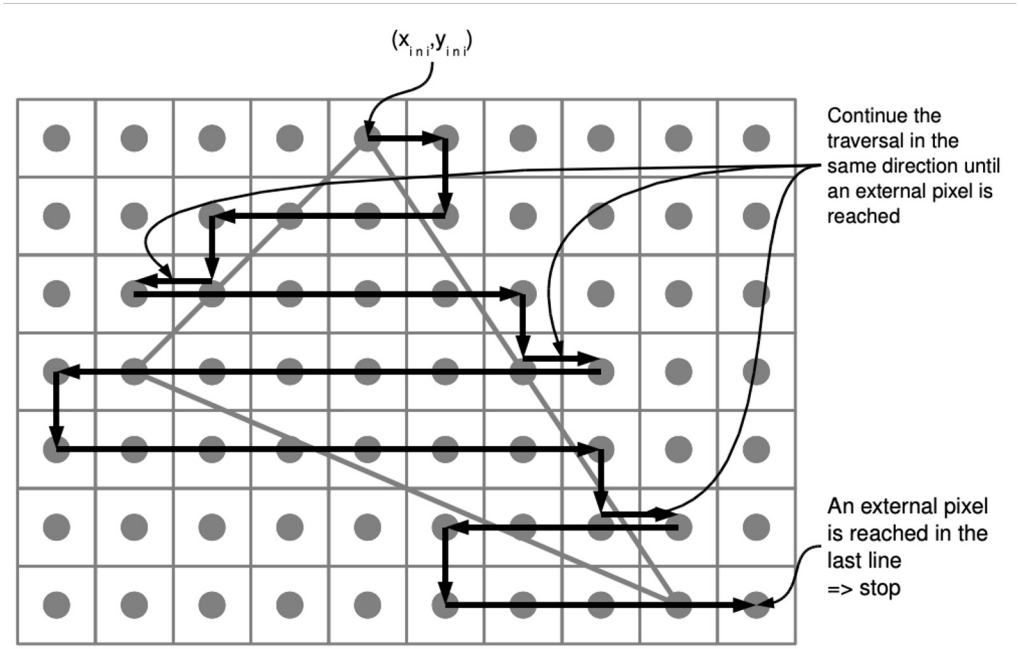
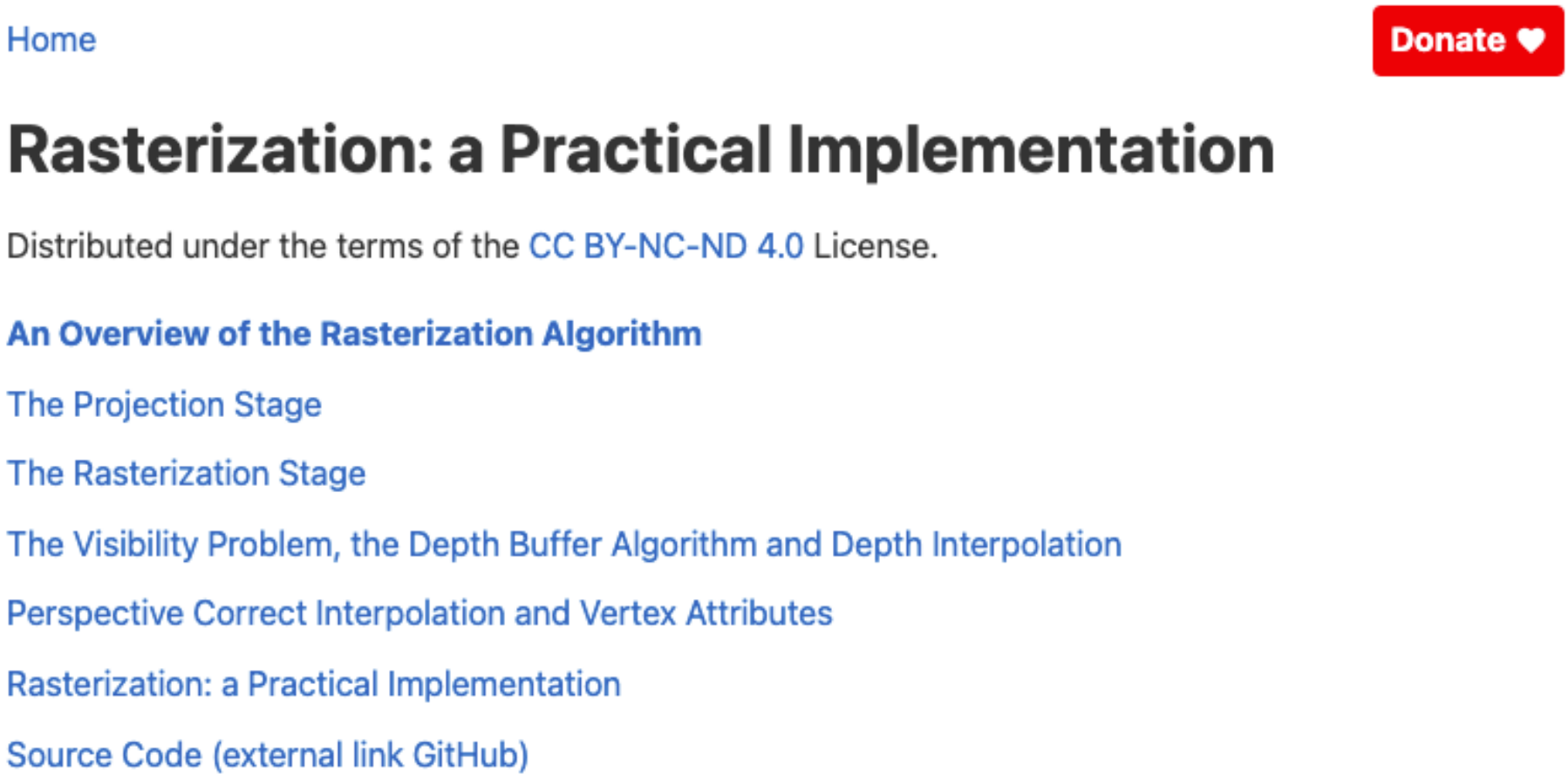


Fig. 7. Zig-zag traversal algorithm.

Insper

Referência recomendada



Home

Donate ♥

Rasterization: a Practical Implementation

https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation/overview-rasterization-algorithm.html

Computação Gráfica

X3D

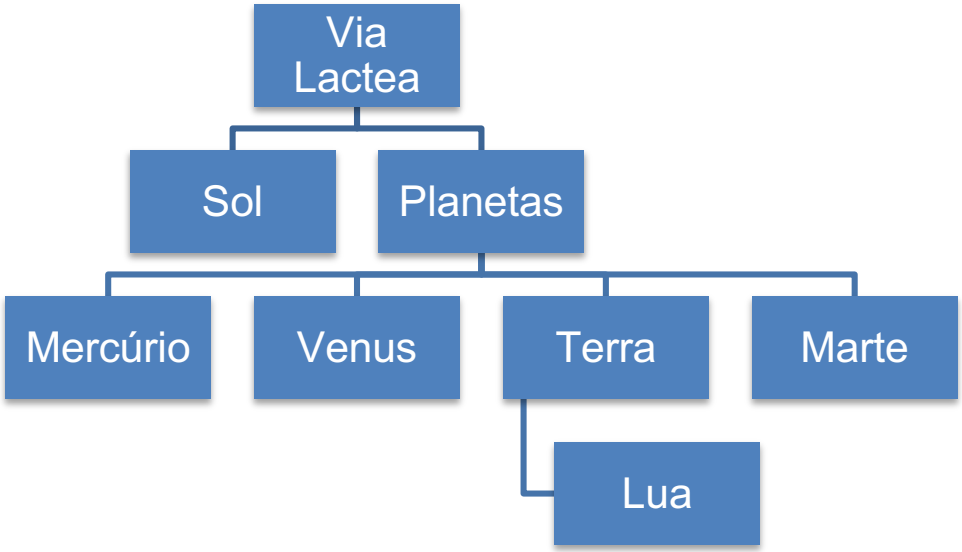
X3D

Formato Universal de Transferência de dados 3D

- Um padrão aberto
- Fácil de entender e modelar
- Portável entre plataformas
- Fácil de ensinar e programar

Grafo de Cena (Scene Graph)

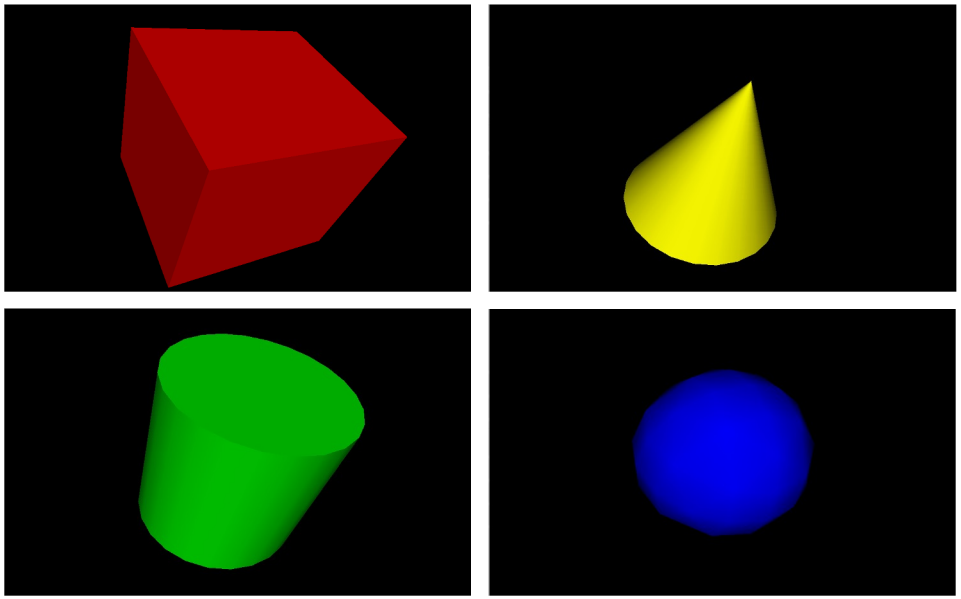
Estrutura de dados hierárquica de objetos gráficos para uma determinada cena. Objetos são representados como nós de um grafo, para posterior renderização.



Veremos mais sobre Grafo de Cena nas próximas aulas.

Exemplos de Tipos de Nós

- Geometrias
  - retas, curvas
  - cubos, esferas, cones, cilindros
  - malhas de polígonos
- Controle
  - Switch/Select
  - Group
- Propriedades
  - Cores
  - Materiais
  - Luzes
  - Câmera



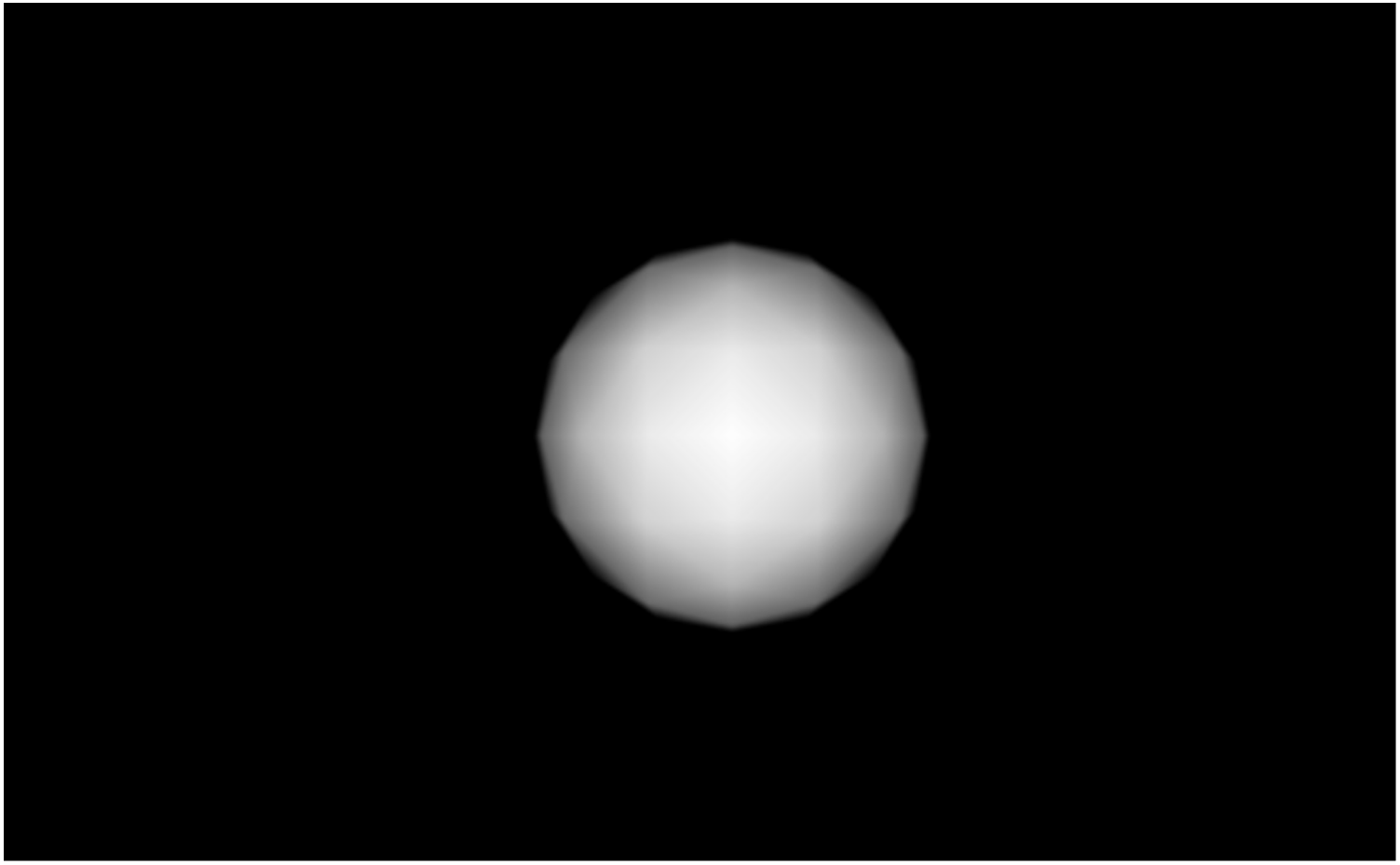
X3D XML (exemplo)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "http://www.web3d.org/specifications/x3d-3.0.dtd">
<X3D profile="Immersive">
<Scene>
        <NavigationInfo type="ANY"/>
        <Transform>
                <Shape>
                        <Appearance>
                                <Material diffuseColor=”1 1 1”/>
                        </Appearance>
                        <Sphere/>
                </Shape>
        </Transform>
</Scene>
</X3D>
```

Insper

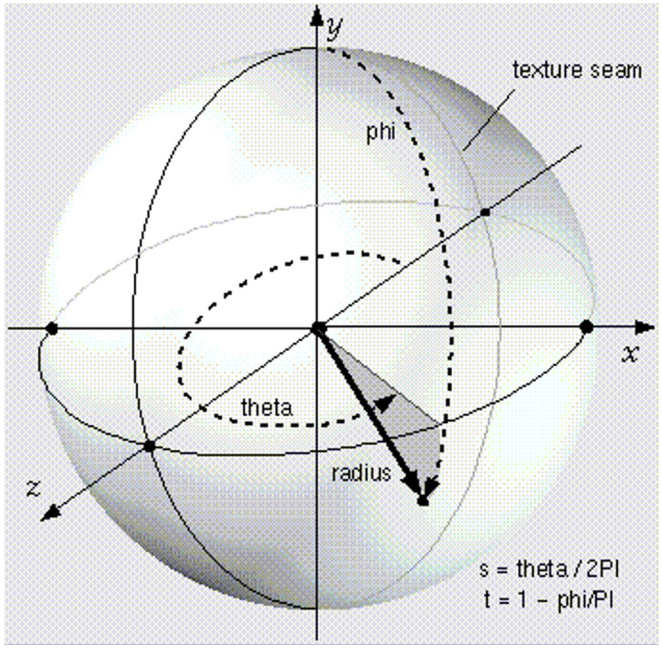
Exemplo

Esfera Branca



Abordagem do X3D (exemplo)

```
Sphere : X3DGeometryNode {
        SFNode [in,out]  metadata     NULL    [X3DMetadataObject]
        SFFloat [ ]      radius       1                (0,∞)
        SFBool  [ ]      solid        TRUE
}
```



Insper

Especificação X3D

https://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/Architecture.html

Extensible 3D (X3D)
Part 1: Architecture and base components

ISO/IEC 19775-1:2013

This document is Edition 3 of ISO/IEC 19775-1, Extensible 3D (X3D). The full title of this part of the International Standard is: *Information technology — Computer graphics, image processing and environmental data representation — Extensible 3D (X3D) — Part 1: Architecture and base components*. **When navigating within this document, it is possible to return to the beginning of the document by clicking on the X3D logo.**

alternativamente: https://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/nodeIndex.html

Insper

Componentes de Geometria 2D

https://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/geometry2D.html

Extensible 3D (X3D)
Part 1: Architecture and base components

14 Geometry2D component

14.1 Introduction

14.1.1 Name

The name of this component is "Geometry2D". This name shall be used when referring to this component in the COMPONENT statement (see 7.2.5.4 Component statement).

14.1.2 Overview

This clause describes the Geometry2D component of this part of ISO/IEC 19775. This includes how two-dimensional geometry is specified and what shapes are available. Table 14.1 provides links to the major topics in this clause.

Table 14.1 — Topics

Insper

Polypoint2D

O nó **Polypoint2D** especifica uma série de vértices no sistema de coordenadas 2D local em cada um dos quais é exibido um ponto. O campo **points** especifica os vértices a serem exibidos.

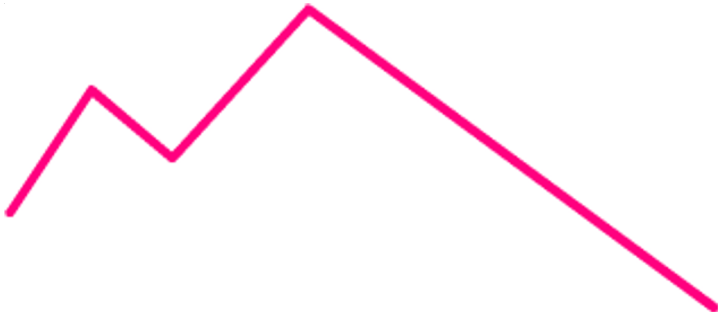
```
Polypoint2D : X3DGeometryNode {
        SFNode                  [in,out]    metadata            NULL
        [X3DMetadataObject]
        MFVec2f    [in,out]     point                           []              (-∞,∞)
}
```

exemplo de Polypoint2D
(com vértices ampliados)

Polyline2D

O nó **Polyline2D** especifica uma série de segmentos de linha contiguos no sistema de coordenadas 2D local conectando os vértices especificados. O campo **lineSegments** especifica os vértices a serem conectados.

```
Polyline2D : X3DGeometryNode {
    SFNode [X3DMetadataObject]    [in,out]    metadata       NULL
    MFVec2f []                                lineSegments   []          (-∞,∞)
}
```



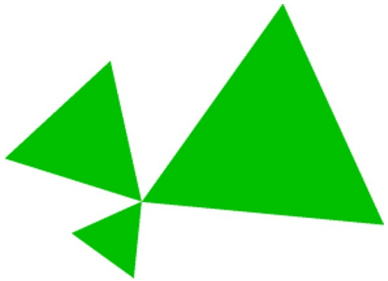
exemplo de Polyline2D

Insper

TriangleSet2D

O nó **TriangleSet2D** especifica um conjunto de triângulos no sistema de coordenadas 2D local. O campo **vertices** especifica os triângulos a serem exibidos. O número de vértices fornecidos deve ser igualmente divisível por três. O excesso de vértices deve ser ignorado.

```
TriangleSet2D : X3DGeometryNode {
        SFNode              [in,out]    metadata NULL      [X3DMetadataObject]
        MFVec2f  [in,out]   vertices                [] (-∞,∞)
        SFBool              []                      solid  FALSE
}
```

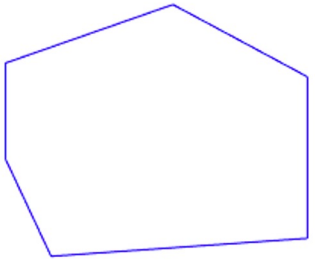


exemplo de TriangleSet2D

Exemplos

***X3D Example Archives:* X3D for Web Authors, Chapter 10 Geometry 2D, Polyline 2D**

Example of Polyline2D showing multiple 2D line segments.



X3D model | ClassicVRML | VRML97 | Canonical XML | annotated documentation | .py python

X_ITE | X3DOM | .json (check) | .x3db Binary | .java source (Javadoc) | .ttl Turtle (query)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.3//EN" "http://www.web3d.org/specifications/x3d-3.3.dtd">
<X3D profile='Immersive' version='3.3' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation =' http://www.web3d.org/specifications/x3d-3.3.xsd '>
    <head>
        <component level='2' name='Geometry2D'/>
        <meta name='title' content='Polyline2D.x3d'/>
        <meta name='description' content='Example of Polyline2D showing multiple 2D line segments.'/>
        <meta name='creator' content='Leonard Daly and Don Brutzman'/>
        <meta name='created' content='17 April 2006'/>
        <meta name='modified' content='20 October 2019'/>
        <meta name='reference' content=' http://X3dGraphics.com '/>
        <meta name='reference' content=' https://www.web3d.org/x3d/content/examples/X3dResources.html '/>
        <meta name='rights' content='Copyright 2006, Daly Realism and Don Brutzman'/>
        <meta name='subject' content=' X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com '/>
        <meta name='identifier' content=' http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter10Geometry2D/Polyline2D.x3d '/>
        <meta name='generator' content='X3D-Edit 3.3, https://savage.nps.edu/X3D-Edit'/>
        <meta name='license' content=' ../license.html'/>
    </head>
```

Index for DEF node : MagentaAppearance

Exemplo:
https://x3dgraphics.com/examples/X3dForWebAuthors/Chapter10Geometry2D/Polyline2DIndex.html

Projeto 1 : Primeira Parte

Fazer um renderizador:

1 – Capaz de desenhar Pontos
2 – Capaz de desenhar Linhas
3 – Capaz de desenhar Triângulos

Data de Entrega:

22/2/2023 às 23:59, via Blackboard (https://insper.blackboard.com/)

Detalhes:

Página da Disciplina (https://lpsoares.github.io/ComputacaoGrafica/)

Insper

Computação Gráfica

Luciano Soares
<lpsoares@insper.edu.br>