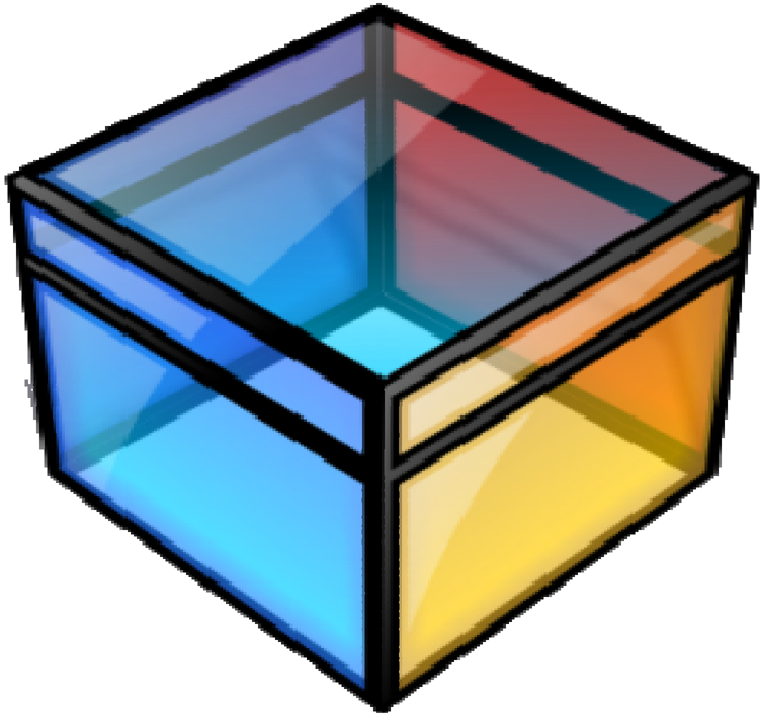
Geometria



COM11087-Tópicos Especiais em Programação II

edmar.kampke@ufes.br

Introdução

- Geometria é uma disciplina inerentemente uma disciplina visual.
- Desenhar figuras e analisá-las.
- Algumas operações que são facilmente resolvidas à mão requerem programação avançada para serem feitas pelo computador.
- Exemplo: Como determinar a interseção entre linhas.

Retas

- Representam a menor distância entre dois pontos.

- Possuem comprimento infinito e em ambas as direções.

- Não confunda com “Segmentos de retas” que são finitos.

- Retas podem ser representadas por pares de pontos ou equações.

Retas

- Pares de pontos (x_1, y_1) e (x_2, y_2) que pertencem a reta.

- Equações do tipo y = mx + b, onde *m* é o coeficiente angular (inclinação da reta) e *b* é o único ponto (0, b) em que a reta intercepta o eixo y.

$$m = \frac{\Delta y}{\Delta x} = \frac{(y1 - y2)}{(x1 - x2)} \; e \; b = y_1 - m x_1$$

- Problema: Retas verticais.

Retas

- Para se resolver o problema utiliza-se a equação no formato ax+by+c=0, pois dessa forma retas verticais podem ser representadas por equações do tipo x = c.

```
typedef struct {
   double a;  /* x-coefficient */
   double b;  /* y-coefficient */
   double c;  /* constant term */
} line;

#define DIMENSION 2
#define X         0
#define Y         1

typedef double point[DIMENSION];
```

Retas

```
points_to_line(point p1, point p2, line *l)
{
   if (p1[X] == p2[X]) {
      l->a = 1;
      l->b = 0;
      l->c = -p1[X];
   } else {
      l->b = 1;
      l->a = -(p1[Y]-p2[Y])/(p1[X]-p2[X]);
      l->c = -(l->a * p1[X]) - (l->b * p1[Y]);
   }
}
point_and_slope_to_line(point p, double m, line *l)
{
   l->a = -m;
   l->b = 1;
   l->c = -((l->a*p[X]) + p[Y]);
}
```

Retas

➢ Interseção entre Retas:

Duas retas distintas se interceptam em um único ponto se não são paralelas.

Retas paralelas possuem a mesma inclinação e diferentes pontos de interceptação com os eixos.

```
bool parallelQ(line l1, line l2)
{
   return ( (fabs(l1.a-l2.a) <= EPSILON) &&
            (fabs(l1.b-l2.b) <= EPSILON) );
}

bool same_lineQ(line l1, line l2)
{
   return ( parallelQ(l1,l2) && (fabs(l1.c-l2.c) <= EPSILON) );
}
```

Retas

- Um ponto (x',y') faz parte de uma reta se ao substituirmos x por x' na equação y=mx+b obtivermos y = y'

- Duas retas y = $m_1x + b_1$ e y = $m_2x + b_2$ se interceptam no ponto:

$$x = \frac{b_2 - b_1}{m_1 - m_2} \quad \text{e} \quad y = m_1 \frac{b_2 - b_1}{m_1 - m_2} + b_1$$

Retas

```
intersection_point(line l1, line l2, point p)
{
   if (same_lineQ(l1,l2)) {
      printf("Warning: Identical lines, all points intersect.\n");
      p[X] = p[Y] = 0.0;
      return;
   }
   if (parallelQ(l1,l2) == TRUE) {
      printf("Error: Distinct parallel lines do not intersect.\n");
      return;
   }
   p[X] = (l2.b*l1.c - l1.b*l2.c) / (l2.a*l1.b - l1.a*l2.b);
   if (fabs(l1.b) > EPSILON)    /* test for vertical line */
      p[Y] = - (l1.a * (p[X]) + l1.c) / l1.b;
   else
      p[Y] = - (l2.a * (p[X]) + l2.c) / l2.b;
}
```

Retas

- Duas retas não paralelas se interceptam e formam um anglo θ

Se as retas estiverem na forma: ax+by+c=0

$$\tan(\theta) = \frac{a_1 b_2 - a_2 b_1}{a_1 a_2 - b_1 b_2}$$

Se as retas estiverem na forma: y=mx+b

$$\tan(\theta) = \frac{m_2 - m_1}{m_1 m_2 - 1}$$

Retas

- Antes de verificar o valor de tan(θ) verificar se elas não são perpendiculares.
- Duas retas são perpendiculares se formam um ângulo reto entre elas, ou seja, $m_1=(-1/m_2)$.
- Outra aplicação constante é encontrar a menor distância de um ponto p a uma reta r.
- O ponto de r que está mais próximo de p forma uma linha perpendicular a r
- Utiliza-se assim as definições anteriores

Retas

```
closest_point(point p_in, line l, point p_c)
{
	line perp;			/* perpendicular to l through (x,y) */
	if (fabs(l.b) <= EPSILON) {		/* vertical line */
		p_c[X] = -(l.c);
		p_c[Y] = p_in[Y];
		return;
	}
	if (fabs(l.a) <= EPSILON) {		/* horizontal line */
		p_c[X] = p_in[X];
		p_c[Y] = -(l.c);
		return;
	}
	point_and_slope_to_line(p_in,1/l.a,&perp); /* normal case */
	intersection_point(l,perp,p_c);
}
```

Retas

- Raios (vetores) são partes de retas que se originam a partir de um ponto v, chamado de origem.

- Qualquer raio é completamente descrito por:

- Uma equação de reta;
- Origem
- Destino
- Direção ou origem e outro ponto no raio

Triângulos e Trigonometria

- Um ângulo é a união de dois raios que compartilham um *endpoint*.

- Trigonometria é a parte da matemática que estuda os ângulos e as suas medidas

- Ângulos possuem duas unidade de medidas: radianos e graus

- Radianos: 0 a 2π
- Graus: 0º a 360º

Triângulos e Trigonometria

- A geometria dos triângulos está diretamente relacionado à trigonometria

- Três funções trigonométricas básicas:
seno, *cosseno* e *tangente*

- Essas funções são importantes pois relacionam ângulos e comprimento de lados de um triângulo

- Lembre-se que num triângulo a soma dos ângulos é 180º ou π radianos.

Triângulos e Trigonometria

- Vale relembrar que num triângulo retângulo (com um ângulo reto = 90º ou π/2 radianos) vale as seguintes relações:

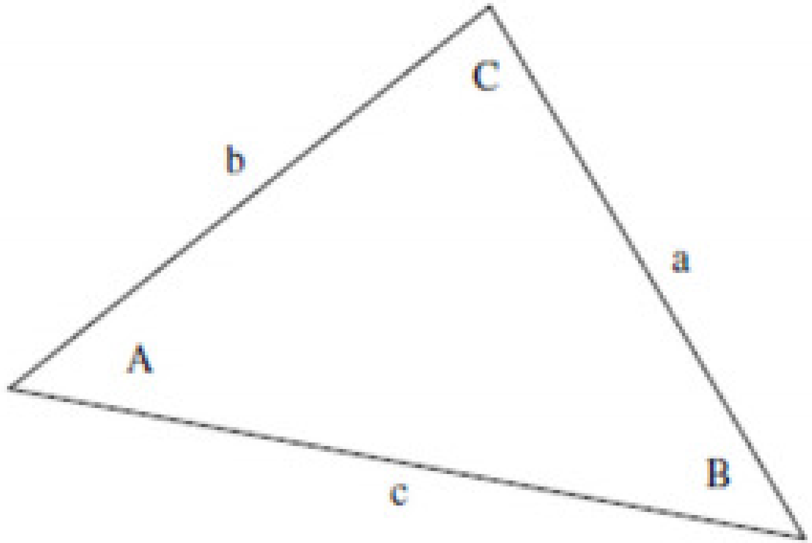
$$a^2+b^2 = c^2 \text{ (teorema de Pitágoras)}$$

c é a hipotenusa (lado oposto ao ângulo reto)
a e b são catetos (lados adjacentes ao ângulo reto)

Seja α um ângulo não reto de um triângulo retângulo.

$$\sin(\alpha) = \frac{cat.op.}{hip.}, \cos(\alpha) = \frac{cat.adj.}{hip.}, \tan(\alpha) = \frac{cat.op.}{cat.adj.}$$

Triângulos e Trigonometria



- Lei dos Senos

$$\frac{a}{\sin(A)} = \frac{b}{\sin(B)} = \frac{c}{\sin(C)}$$

- Lei dos cossenos

$$a^2 = b^2 + c^2 - 2.b.c.\cos(A)$$

Triângulos e Trigonometria

- Aplicações:

- Dado dois ângulos e um lado ache o resto

- Dado dois lados e um ângulo ache o resto

- Para calcular a área de um triângulo pode-se usar álgebra linear, sabendo-se os vértices do triângulo

$$2 \cdot A(T) = \begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix} = a_x b_y - a_y b_x + a_y c_x - a_x c_y + b_x c_y - c_x b_y$$

Triângulos e Trigonometria

```
double signed_triangle_area(point a, point b, point c)
{
   return( (a[X]*b[Y] - a[Y]*b[X] + a[Y]*c[X]
         - a[X]*c[Y] + b[X]*c[Y] - c[X]*b[Y]) / 2.0 );
}
double triangle_area(point a, point b, point c)
{
   return( fabs(signed_triangle_area(a,b,c)) );
}
```

Círculos

- Conjunto de pontos que estão a uma distância *r* (raio) de um ponto (x_c,y_c) chamado de centro.

- Disco é um círculo mais o seu interior, ou seja são todos os pontos cuja distância ao centro seja menor ou igual a *r*.

- Um círculo pode ser representado de duas formas: 1) Tripla de pontos fronteiriços e 2) Centro e raio.

```
typedef struct {
   point c;   /* center of circle */
   double r;  /* radius of circle */
} circle;
```

Círculos

- Equação do Círculo:

$$r^2 = (x - x_c)^2 + (y - y_c)^2$$

- Circunferência

$$C = 2\pi r$$

- Área

$$A = \pi r^2$$

- Diâmetro (maior distância entre dois pontos do círculo)

$$d = 2r$$

* $\pi = 3{,}1415296$

Círculos

- Tangente:

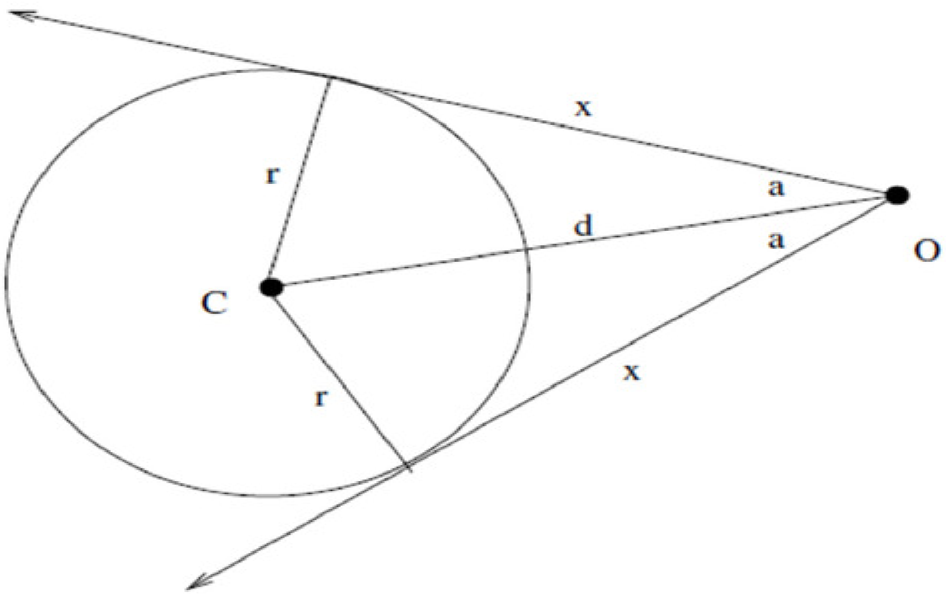
Uma reta pode interceptar um círculo em 0, 1 ou 2 pontos.

- No primeiro caso a reta não intercepta o círculo
- No último caso a reta cruza o interior do círculo
- Quando a reta intercepta o círculo em apenas um ponto diz-se que ela é tangente ao círculo

Círculos

➢ Construção de retas tangentes:

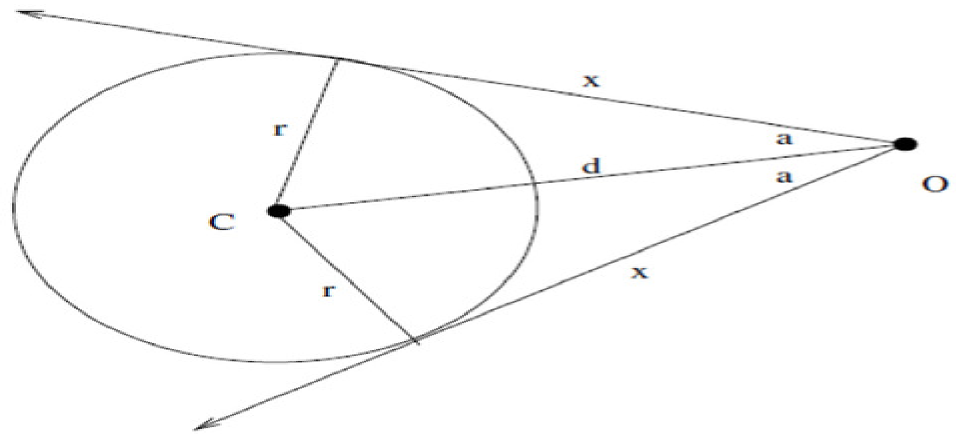
A reta que liga o centro à tangente (correspondente ao raio) formam um ângulo de 90º entre si, ou seja são perpendiculares.



Círculos

- O triângulo formado por *r*, *d* e *x* é retângulo, sendo que *r* e *d* são conhecidos, uma vez que *d* é a distância entre dois pontos (*O* a *C*). Pode-se calcular o valor de *x* pelo teorema de pitágoras.

- A partir disso, podemos determinar os pontos de tangência e o ângulo *a*.



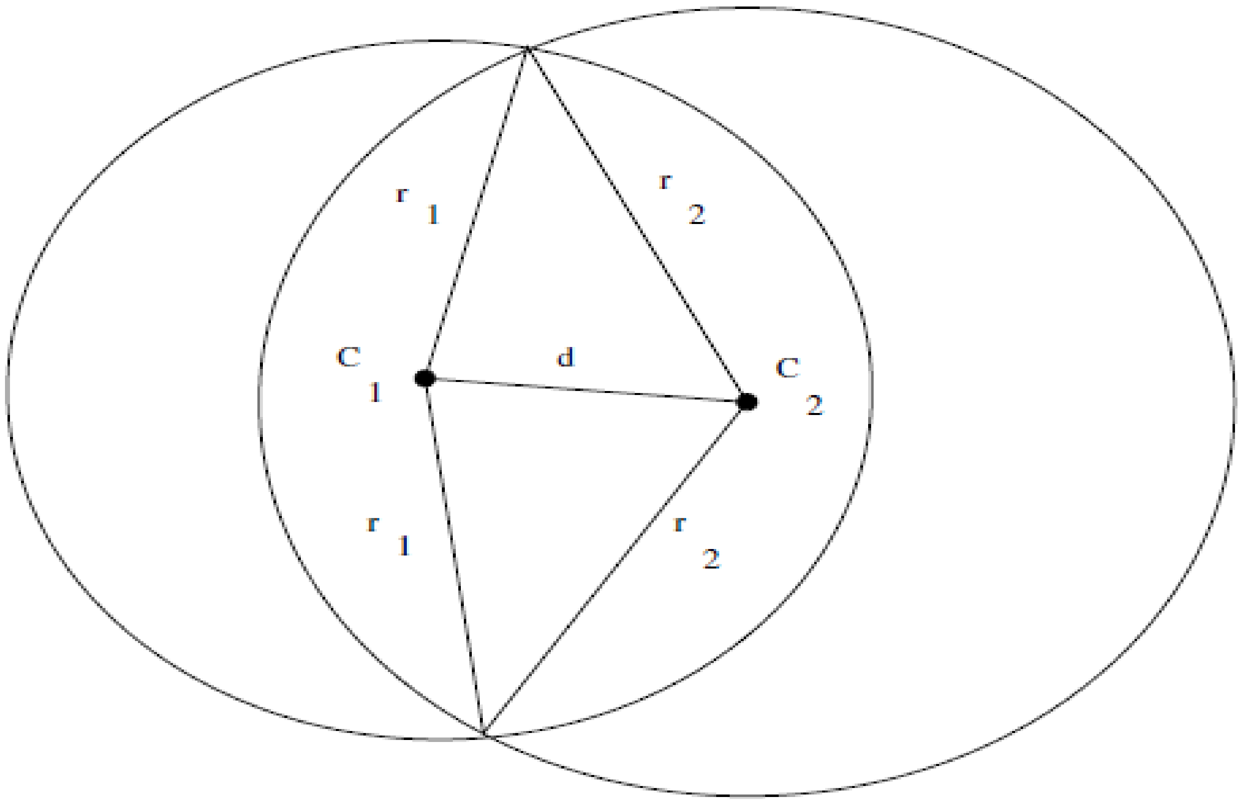
Círculos

- Dois círculos c_1 e c_2 de raios r_1 e r_2, respectivamente podem se relacionar de três formas:

- Não possuem nenhuma interseção (Distância entre centros for maior que r_1+r_2
- Um círculo menor (c_1)está totalmente dentro do outro. Se a distância entre os centros + r_1 for menor que r_2.
- Caso contrário os círculos se interceptam em dois pontos que formam triângulos com a distância entre os centros.

Círculos

- Os ângulos e as coordenadas das interseções podem ser determinadas!



Bíblioteca

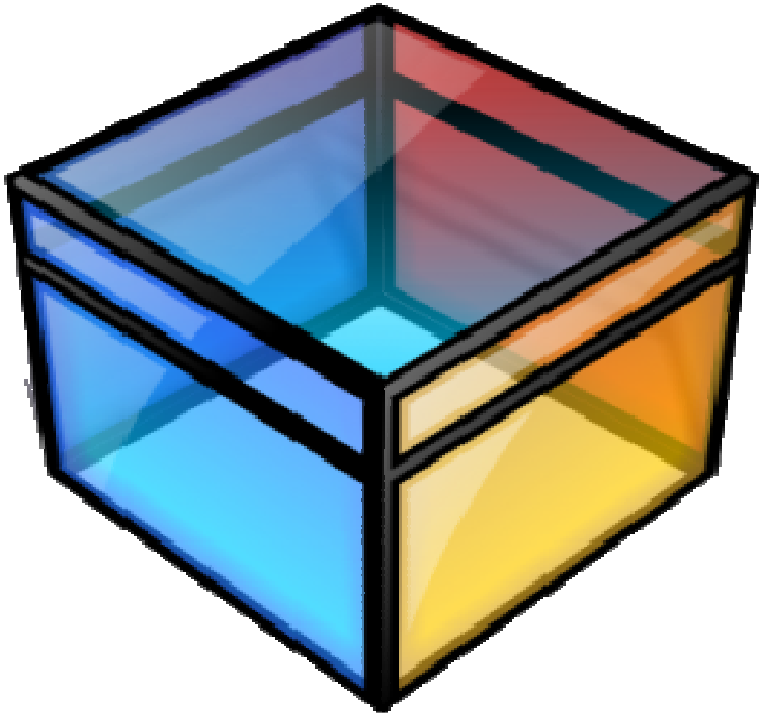
- Nota: Existem duas funções arctan para identificar corretamente em qual quadrante está o ângulo. O que depende dos sinais de *x* e *y*.

```
#include <math.h>

double cos(double x);    /* compute the cosine of x radians */
double acos(double x);   /* compute the arc cosine of [-1,1] */

double sin(double x);    /* compute the sine of x radians */
double asin(double x);   /* compute the arc sine of [-1,1] */

double tan(double x)     /* compute the tangent of x radians */
double atan(double x);   /* compute the principal arctan of x */
double atan2(double y, double x); /* compute the arc tan of y/x */
```

Exercícios

- *Dog and Gopher*
- *Is This Integration?*
- *Rope Crisis in Ropeland!*
- *The Knights of the Round Table*
- *The Largest/Smallest Box ...*
- *Birthday Cake*

Geometria



COM11087-Tópicos Especiais em Programação II

edmar.kampke@ufes.br