Geometria Computacional

Ole Peter Smith, IME, UFG, ole@mat.ufg.br

24/05/2010

Life sure is a Mystery to be Lived
Not a Problem to be Solved...

These slides were made in LaTeX/Beamer

- Beautiful Scientific Documents
- Classical Free Software
- Markup Language
- Thesis, automatic:
  - · Bibliography
  - · Contents
  - · List of Tables, Figures
  - · Index, References
- Slideshows: Beamer
- MikTex, LyX, TexMaker, Kyle

Quero Desenhar $\mu\alpha\tau\epsilon\mu\alpha\tau\iota\kappa\alpha$...

- Usando SL...
- Graphical Library
  - GD
  - Unices - Ruindows
  - PHP - C - Perl - ...
  - http://www.libgd.org/Documentation
  - http://www.php.net/manual/pt_BR/book.image.php
- Formats: GIF - PNG - JPG
- $\smile$ Animated GIF's
  $\frown$ GIF's: poucas cores

Apache/CGI

- Apache: webserver
  - Documentos Estáticos
  - Documentos Dinâmicos
- PHP
  - Linguagem para gerar páginas dinâmics
- Perl, C, Java
- Minha maquina: 127.0.0.1
- Windows: Windows Apache Mysql Php (WAMP)
- Localizar 'ServerRoot'

Estrutura de uma Programa

- $image=CreateImageObject;
- $image− >Initialize;
- $image− >DrawSomething;
- $image− >DrawSomethingMore;
- ...
- $image− >WriteImage;
- $image− >Close;
- Pixels... ⌒

Init Image Object (PHP)

- ImageCreate:

```
$image = imagecreate (400,400)or die ("Cannot Create image");
#Black on White - Background first!
$bg_color = imagecolorallocate ($image,255, 255, 255);
$text_color = imagecolorallocate ($image,0,0,0);
```

- Tell Apache to Expect an Image

```
header("Content-type: image/png");
```

- Draw a Line

```
imageline($image,10,20,100,200,$text_color);
```

- Close and Write

```
imagepng ($image);
```

- Save in *ServerRoot*

Never Publish Emails on www

- ```
  $id=$_GET["ID"];
  $person=Mysql_SelectHash("People","ID='".$id."'");
  $email=$person[ "Email" ];
  $hgt=imagefontheight(3);
  $wdt=imagefontwidth(3*strlen($email);
  imagecolortransparent($image,$bg_color);
  imagestring ($image,$fontsize,5,5, $email,$text_color);
  ```
  ```

Line and Point

```
void imageline($image,
              int $x1, int $y1,     #first point
              int $x2, $int x2,     #last point
              int $color);

 void imagearc($image,
              int $cx, int $cy,     #center
              int $a, $int b,       #semi-eixos
              int $ang1, int $ang2, #angle (degrees)
              int $color);

void imagefilledarc(...);
```

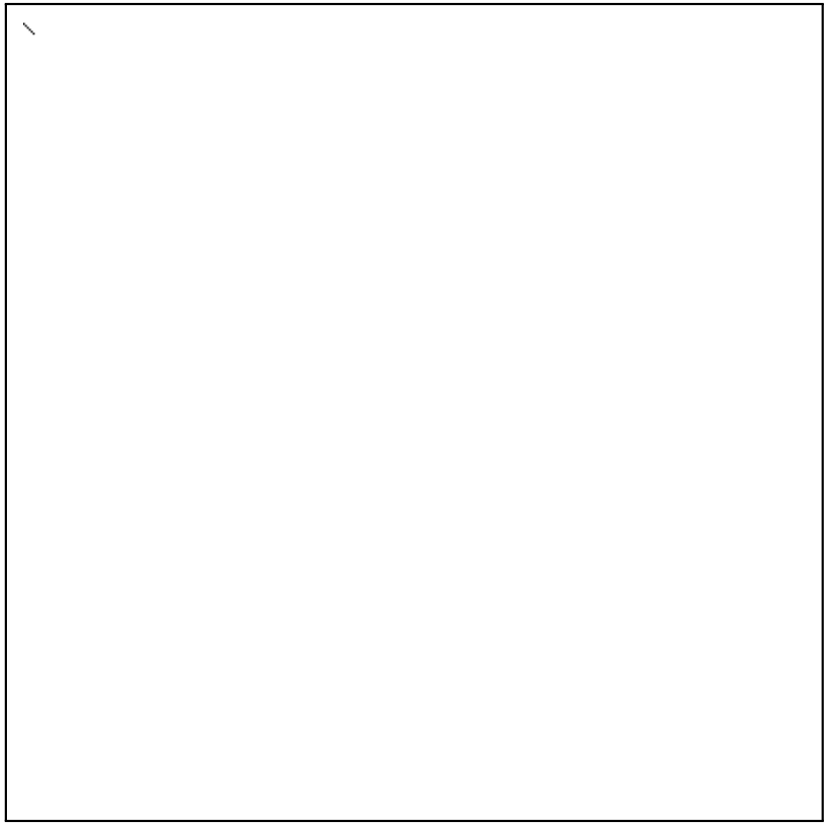
Draw Line

- Draw Line: $y = x + 1, x \in [0, 5]$

```
$image = imagecreate (400,400)
         or die ("Cannot Create image");
#Black on White - Background first!
$bg_color = imagecolorallocate ($image,255, 255, 255);
$text_color = imagecolorallocate ($image,0,0,0);

header("Content-type: image/png");
imageline($image,0,1,5,6,$text_color);
imagepng ($image);
```

Draw Line I - Result

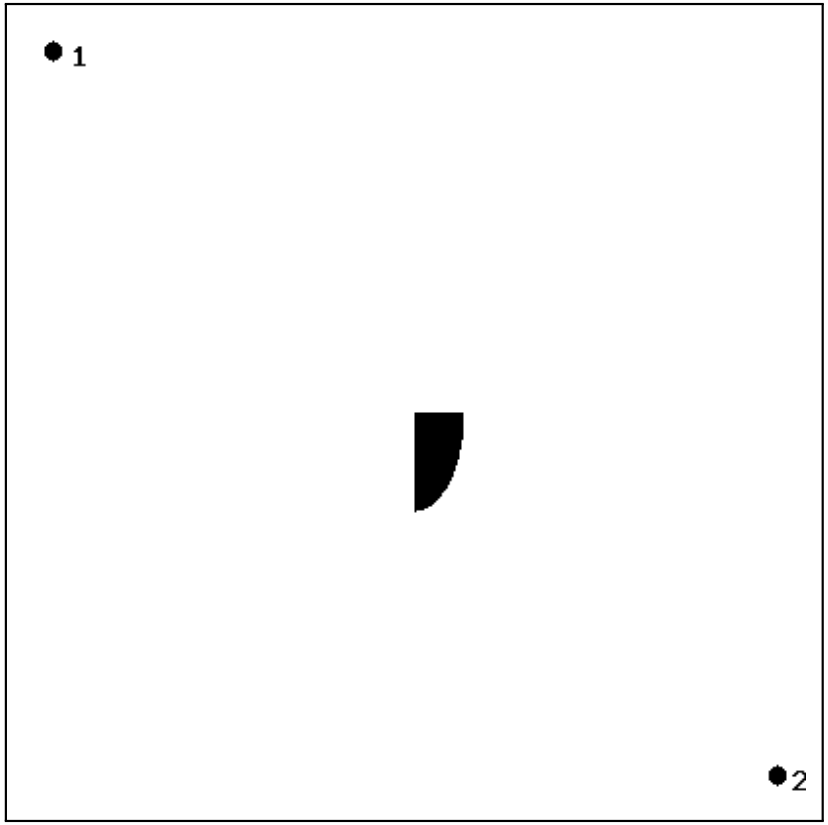


Draw Line I - Problems

- Escalonamento!
- Flipped: Cima por baixo...

```
$im=imagecreate (400,400)...
imagestring($im,5,15+8,15-5,"1",$txtcol);
imagefilledarc($im,15,15,10,10,0,360,$txtcol,IMG_ARC_PIE);
imagefilledarc($im,200,200,50,100,0,90,$txtcol,IMG_ARC_PIE);
imagestring($im,5,385+8,385-5,"2",$txtcol);
imagefilledarc($im,385,385,10,10,0,360,$txtcol,IMG_ARC_PIE);
```

Draw Line II - Result



Scaling

- $R_x, R_y \in \mathbb{N}$: Ex. 400,400
- $Min_x, Min_y \in \mathbb{R}$: 0.0,0.0
- $Max_x, Max_y \in \mathbb{R}$: 6.0,6.0
- Affin Application:

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} a_x x + b_x \\ a_y y + b_y \end{pmatrix}$$

- Conditions:

$$\begin{pmatrix} 0 \\ R_y \end{pmatrix} = \begin{pmatrix} a_x Min_x + b_x \\ a_y Min_y + b_y \end{pmatrix}, \qquad \begin{pmatrix} R_x \\ 0 \end{pmatrix} = \begin{pmatrix} a_x Max_x + b_x \\ a_y Max_y + b_y \end{pmatrix}$$

Scaling

- $a_x(Max_x - Min_x) = R_x - 0 \wedge a_y(Max_y - Min_y) = 0 - R_y \Leftrightarrow$
- $a_x = \dfrac{R_x}{Max_x - Min_x} \wedge a_y = -\dfrac{R_y}{Max_y - Min_y}$
- $b_x = -a_x Min_x$
- $b_y = -a_y Max_y$

Scaling - Init

```
function InitCanvas($R,$Min,$Max)
{
    global $a,$b;
    $a=array();$b=array();

    $a[0]=$R[0]/($Max[0]-$Min[0]);
    $a[1]=-$R[1]/($Max[1]-$Min[1]);

    $b[0]=-$a[0]*$Min[0];
    $b[1]=$a[1]*$Max[1];
}
```

Scaling - Convertion

```
function ScalePoint($p)
{
    global $a,$b;

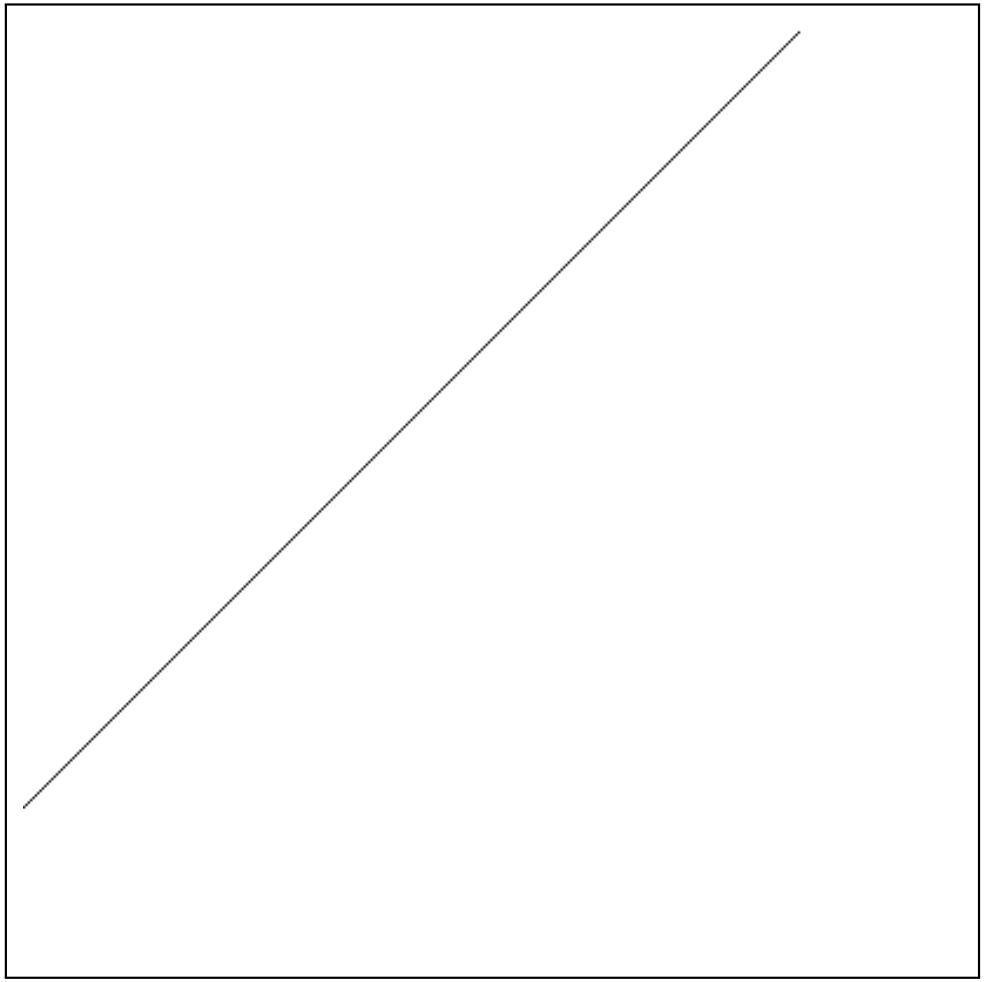
    $pp=array();
    for ($i=0;$i<2;$i++)
    {
        $pp[$i]=((int) ($a[$i]*$p[$i]+$b[$i]));
    }

    return $pp;
}
```

Now Draw the Line....

- ```
  $Res=array(400,400);
  $Min=array(0.0,0.0);
  $Max=array(6.0,6.0);
  InitCanvas($R,$Min,$Max);
  $p1=array(0.0,1.0);
  $p2=array(5.0,6.0);
  $pp1=ScalePoint($p1);
  $pp2=ScalePoint($p2);
  imageline($pp1[0],$pp1[1],$pp2[0],$pp2[1],$color);
  ```
  ```

Draw Line I - Result



Serial Convertion

- Maligma: Código Repetido
-
```
function ScalePoints($ps)
{
    $pps=array();
    for ($n=0;$n<count($ps);$n++)
    {
        $pps[$n]=ScalePoint($ps[$n]);
    }

    return $pps;
}
```

Drawing a Circle

- Equation:

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

- Parametrization:

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_c + r\cos t \\ y_c + r\sin t \end{pmatrix}$$

- $t \in [-\pi, \pi]$

Generating a Circle

```
function Circle($n,$pc,$r,$ang=0.0)
{
    $dt=2*3.1415927/(1.0*($n-1));
    $ps=array();
    for ($i=0,$t=$ang;$i<$n;$i++,$t+=$dt)
    {
        $p=array( $pc[0]+$r*cos($t),$pc[1]+$r*sin($t) );
        array_push($ps,$p);
    }
    return $ps;
}
```

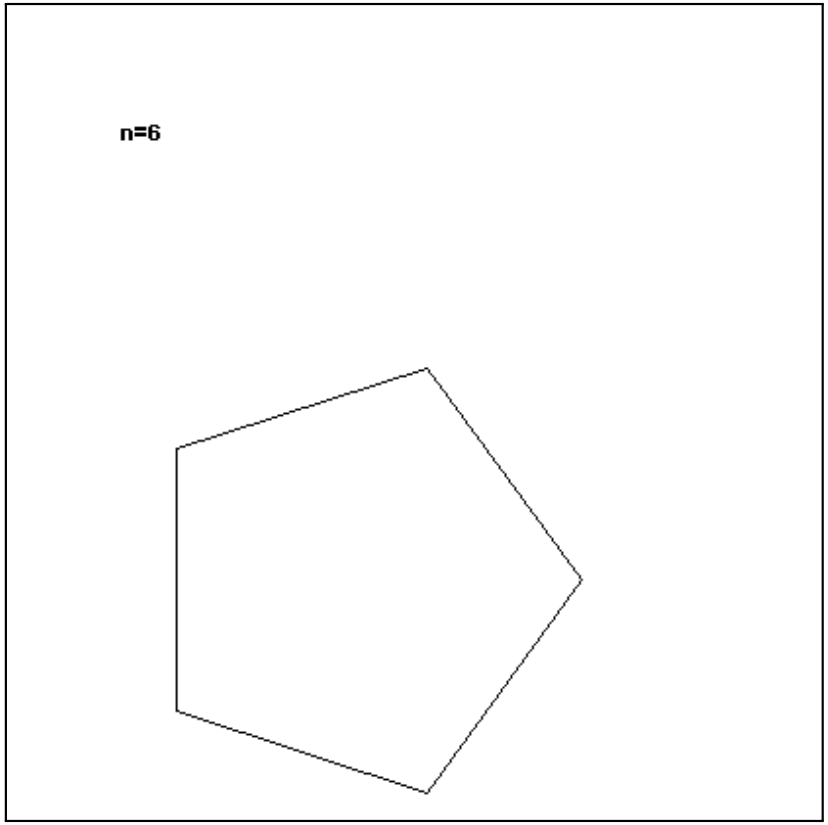
Curve as Line Segments

```
function DrawCurve($im,$ps,$color)
{
   $pps=ScalePoints($ps);
   for ($n=0;$n<count($pps)-1;$n++)
   {
      imageline($im,$pps[$n][0],$pps[$n][1],
                $pps[$n+1][0],$pps[$n+1][1],$color);
   }
}
```

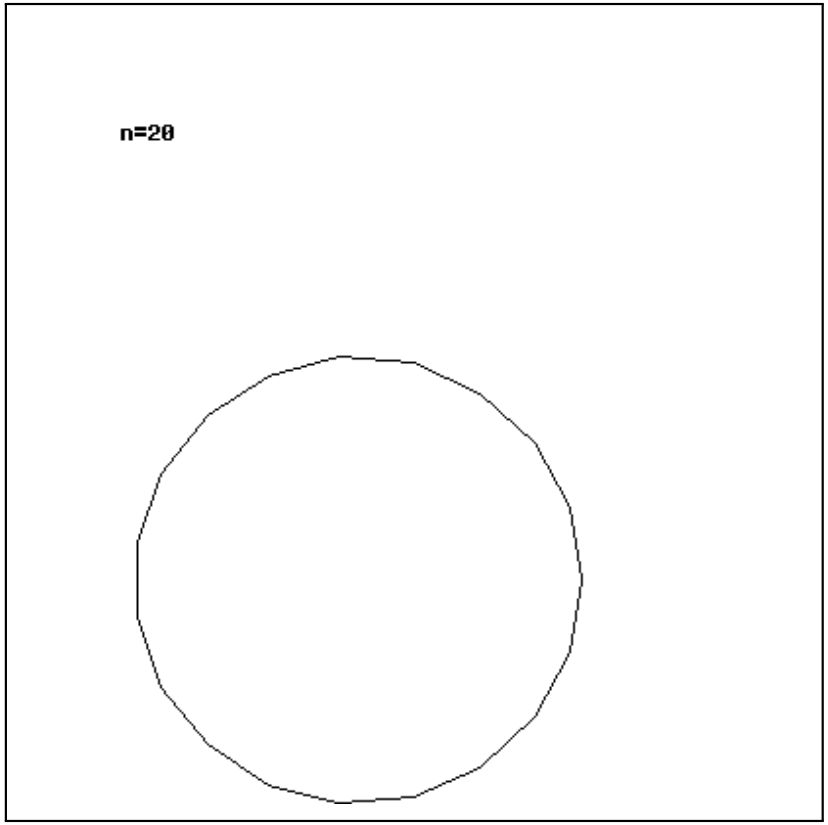
Drawing Circle

```
$n=6;
$Res=array(400,400);
$Min=array(0.0,0.0);
$Max=array(6.0,6.0);
InitCanvas($Res,$Min,$Max);
$center=array(2.0,1.0);
$ps=Circle($n,$center,2);
DrawCurve($im,$ps,$color);
imagestring($im,3,5,5,"n=$n",$color);
```

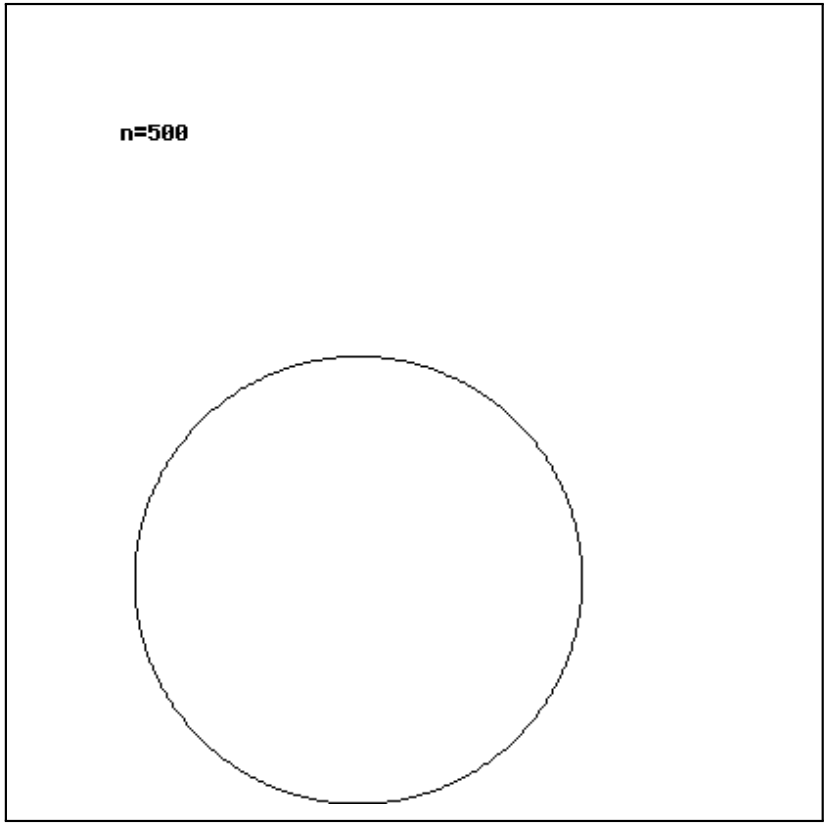
Draw Circle - Result $n = 6$



Draw Circle - Result $n = 20$



Draw Circle - Result $n = 500$



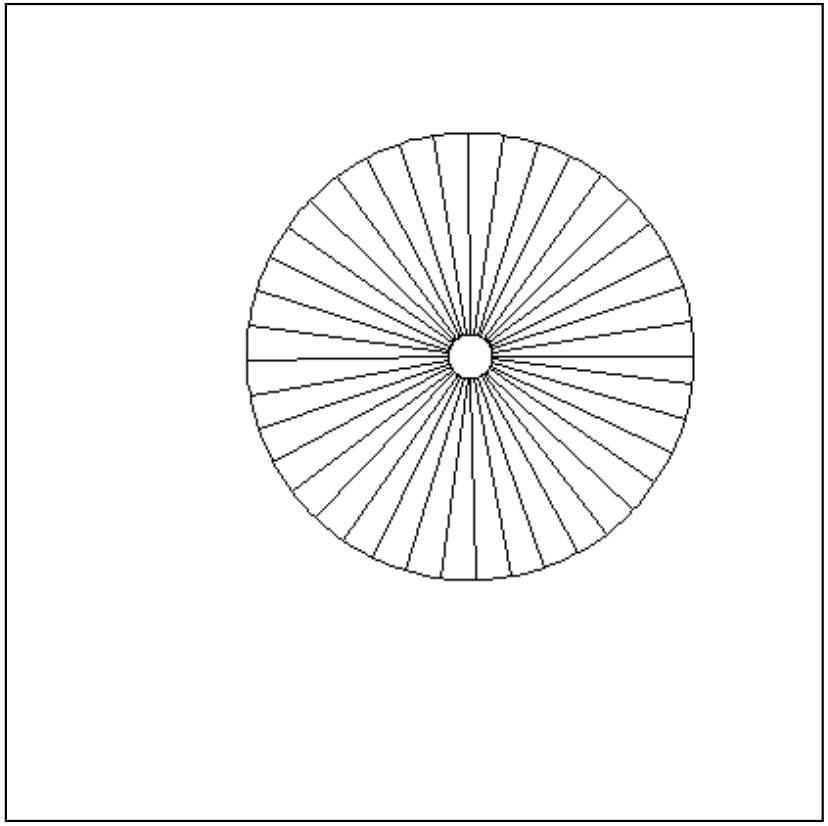
Drawing a Wheel

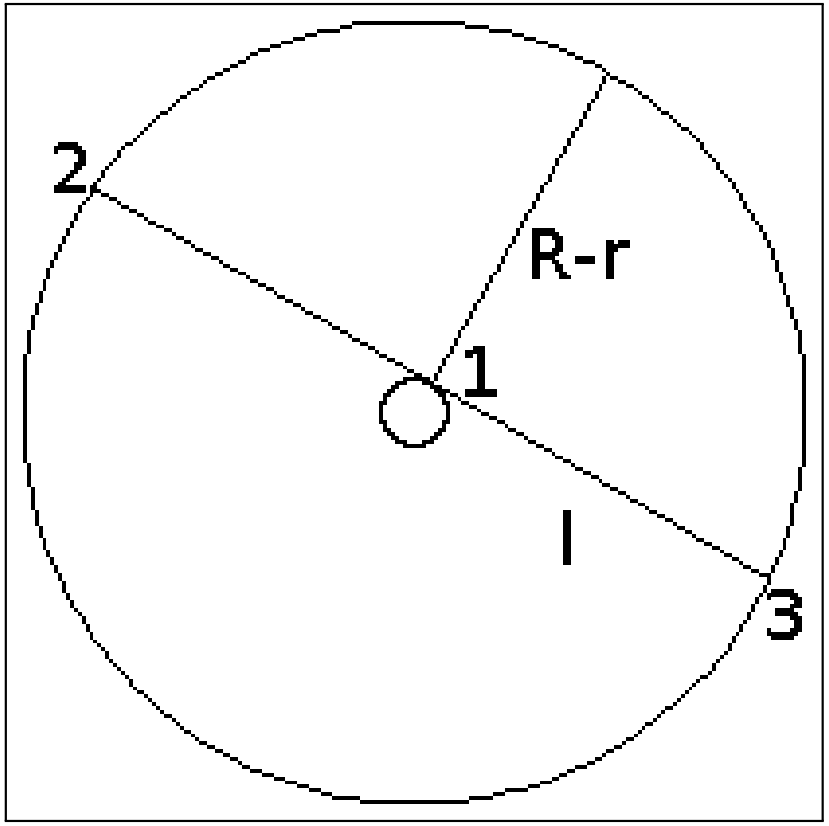
```
function DrawWheel($im,$n,$dn,$pc,$r,$R,$color,$ang=0.0)
{
   $p1s=Circle($n,$pc,$r,$ang);
   DrawCurve($im,$p1s,$color);
   $p2s=Circle($n,$pc,$R,$ang);
   DrawCurve($im,$p2s,$color);
   for ($i=0;$i<$n;$i+=$dn)
   {
      $p1=ScalePoint($p1s[$i]);
      $p2=ScalePoint($p2s[$i]);
      imageline($im,$p1[0],$p1[1],$p2[0],$p2[1],$color);
   }
}
```

Drawing a Wheel

- ```
  $n=200;
  $dn=5;
  $center=array(2.0,1.0);
  DrawWheel($im,$n,$dn,$center,$color,3.0,0.2);
  ```
- Modelo radial
- Melhorar: Modelo tangencial
  ```

Wheel - Result





Better Model

- Point 1:

$$\underline{\mathbf{r}}_1 = \begin{pmatrix} r\cos\theta \\ r\sin\theta \end{pmatrix}$$

- Tangente ou normal unitário(*versor*):

$$\underline{\mathbf{n}} = \begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix}$$

- Ponto 2:

$$\underline{\mathbf{r}}_2 = \underline{\mathbf{r}}_1 + l\underline{\mathbf{n}} = \begin{pmatrix} r\cos\theta - l\sin\theta \\ r\sin\theta + l\cos\theta \end{pmatrix}$$

- $|\underline{\mathbf{r}}_2|^2 = R^2 \Leftrightarrow l = \pm\sqrt{R^2 - r^2}$

Animate Wheel

- ```
  $nimages=79;
  $dang=2.0*3.1415927/$nimages;
  system("/bin/rm Wheel*.gif");
  for ($ang=0.0,$i=0;$i<$nimages;$i++)
  {
     $image = ImageCreate...
     $center=array(3.0,3.0);
     DrawWheel($image,200,5,$center,0.2,2.0,$color,$ang);
     imagegif($image,"Wheel.$i.gif");
     $ang+=$dang;
  }
  ```
  Merda! PHP não tem suporte para GIF animado...
  ```

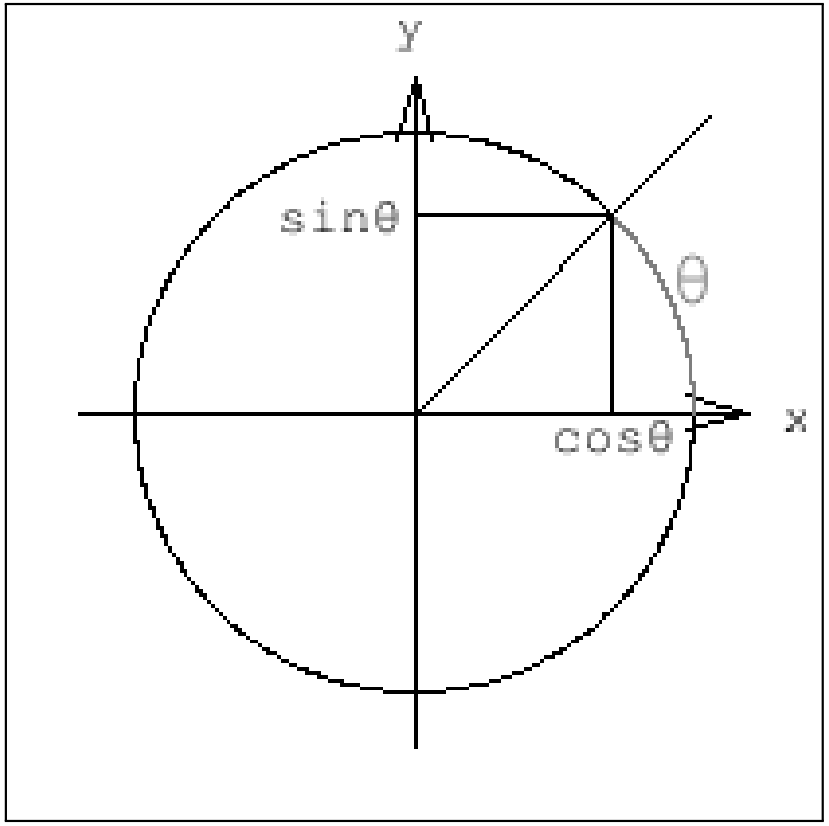
Animate Wheel

- ImageMagick...
- Continued...

```
system("/usr/bin/convert -delay 2 -loop 0 ".
       "Wheel.*.gif Wheel.gif");
header("Content-type: text/html");
print "<IMG SRC='Wheel.gif'>\n";
```

- http://127.0.0.1/uruacu/6.php

Funções Trigonométricas



Nada de triângulos...

Rotação

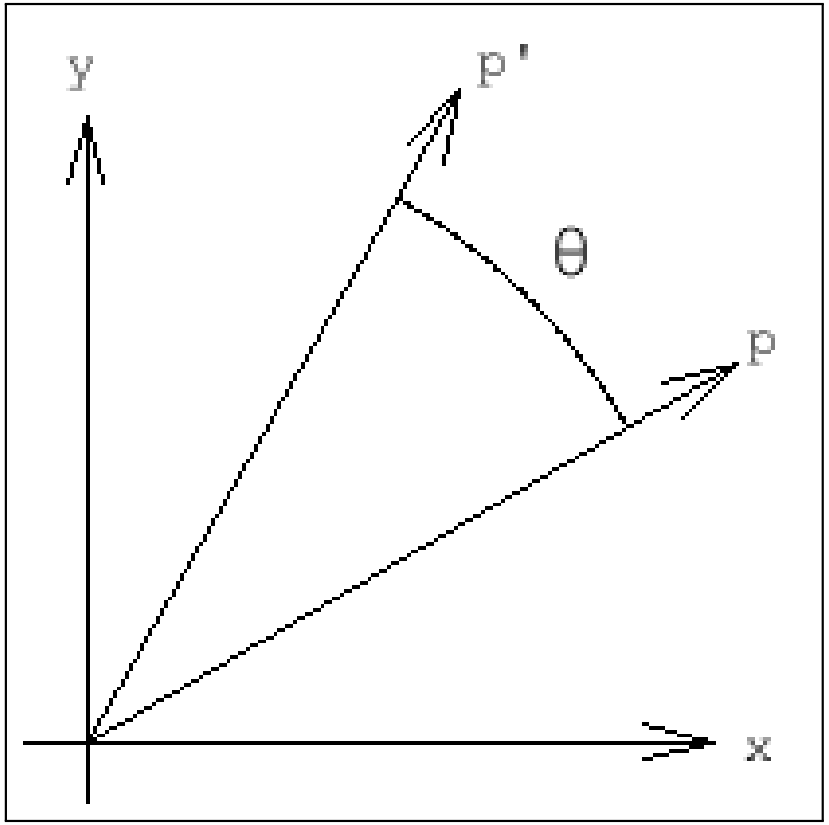
- Rotação, ângulo θ:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \underline{\underline{\mathbf{D}}}(\theta) \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} =$$

$$\begin{pmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{pmatrix}$$

- $\underline{\underline{\mathbf{R}}}$ *ortogonal*: $\underline{\underline{\mathbf{R}}}^T \underline{\underline{\mathbf{R}}} = \underline{\underline{\mathbf{I}}} \Leftrightarrow \underline{\underline{\mathbf{R}}}^{-1} = \underline{\underline{\mathbf{R}}}^T$
- $\underline{\underline{\mathbf{R}}}(\theta)^{-1} = \underline{\underline{\mathbf{R}}}(-\theta)$
- $\underline{\underline{\mathbf{R}}}(\theta)^n = \underline{\underline{\mathbf{R}}}(n\theta), \quad \forall n \in \mathbb{Z}$

Rotação



- Pontos fixos: (0.0)
- Retas em retas
- Círcumferências em Círcumferências
- Elipses em elipses
- Hipérboles em Hipérboles

Aplicação Linear: $\mathbb{R}^n \mapsto \mathbb{R}^n$

- $f(\underline{\mathbf{x}}) = \underline{\underline{\mathbf{A}}}\underline{\mathbf{x}} \quad - \quad \underline{\underline{\mathbf{A}}} = (\underline{\mathbf{a}}_1 \ldots \underline{\mathbf{a}}_n)$
- Composição: $f_1 \circ f_2 = \underline{\underline{\mathbf{A}}}_1 \underline{\underline{\mathbf{A}}}_2$
- Bijeção $\Leftrightarrow \det \underline{\underline{\mathbf{A}}} \neq 0$
- Núcleo, $\underline{\underline{\mathbf{A}}}\underline{\mathbf{x}} = \underline{\mathbf{0}}$: dim$= n - \rho_A$
- Imagen, ger$(\underline{\mathbf{a}}_1 \underline{\mathbf{a}}_2)$: dim $\mathrm{Im}(f) = \rho_A$
- Inversa seeeeee: $\underline{\underline{\mathbf{A}}}$ regular, f^{-1}: $\underline{\underline{\mathbf{A}}}^{-1}$
- Imagem inversa: Se $\underline{\underline{\mathbf{A}}}\underline{\mathbf{x}}_0 = \underline{\mathbf{y}}$:

$$f^{-1}(\underline{\mathbf{y}}) = \{\underline{\mathbf{x}}_0 + \ker f\}$$

Escalonamento

- Dilatação pelo eixo y:

$$\underline{\underline{\mathbf{E}}}\ \underline{\mathbf{x}} = \begin{pmatrix} \lambda & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \lambda x \\ y \end{pmatrix}$$

- $\lambda > 1$: Dilatação
- $\lambda = 1$: Identidade
- $0 < \lambda < 1$: Contração
- $\lambda = 0$: Projeção no eixo y
- $\lambda = -1$: Reflexão no eixo y

Escalonamento

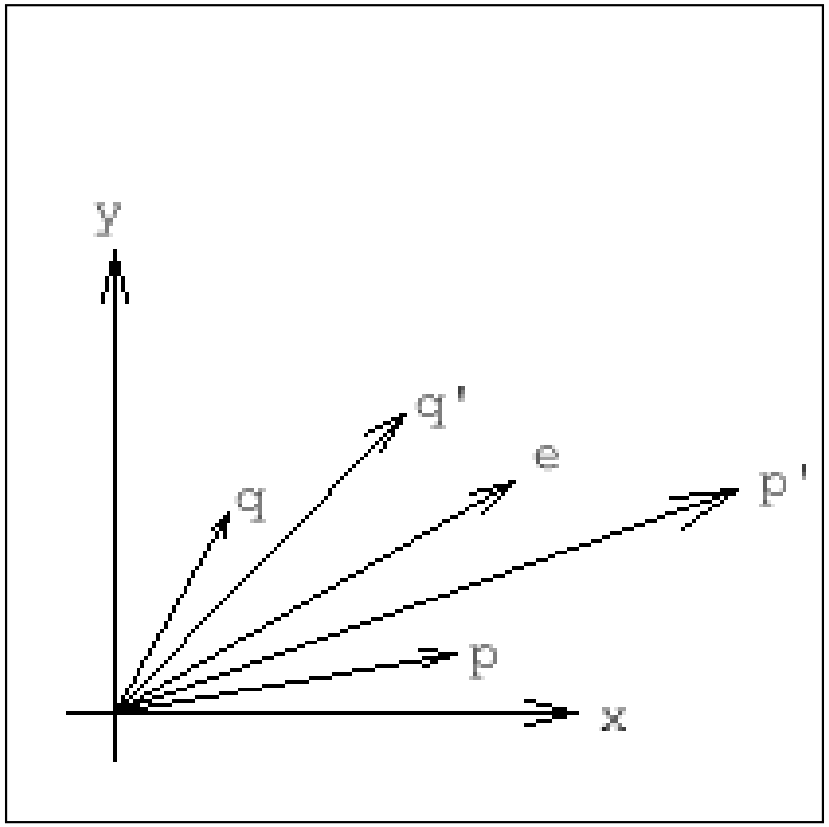
- Dilatação pelo eixo $\underline{\mathbf{e}} = (\cos\theta, \sin\theta)$:
  1: Rotação ângulo θ: $\underline{\mathbf{i}} \mapsto \underline{\mathbf{e}}$ e $\underline{\mathbf{j}} \mapsto \underline{\mathbf{f}} = \widehat{\underline{\mathbf{e}}}$
  2: Dilatação pelo eixo $\underline{\mathbf{f}}$
  3: Rotação ângulo $-\theta$
- $\underline{\underline{\mathbf{E}}}_\theta = \underline{\underline{\mathbf{R}}}^{-1}\underline{\underline{\mathbf{E}}}\underline{\underline{\mathbf{R}}} =$

$$\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \lambda & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} =$$

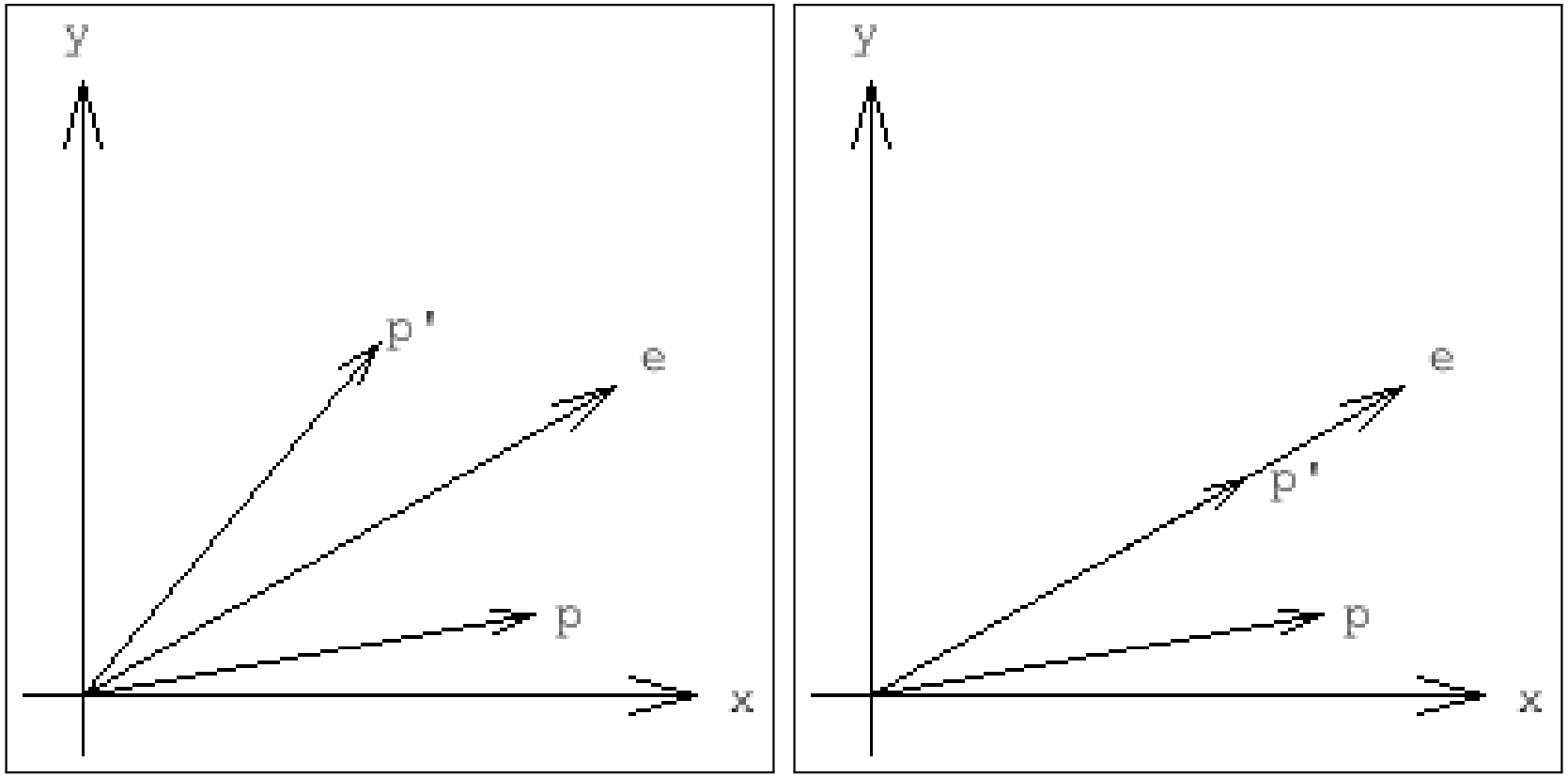
$$\begin{pmatrix} \lambda\cos^2\theta + \sin^2\theta & (1-\lambda)\cos\theta\sin\theta \\ (1-\lambda)\cos\theta\sin\theta & \lambda\sin^2\theta + \cos^2\theta \end{pmatrix}$$

- Linear
- $\underline{\underline{\mathbf{E}}}_\theta$ e $\underline{\underline{\mathbf{E}}}$ *similares*

Escalonamento e Reflexão



Projeção



Reflexão em $\underline{\mathbf{e}} = (\cos\theta, \sin\theta$

- $\underline{\underline{\mathbf{R}}} = \begin{pmatrix} -\cos 2\theta & \sin 2\theta \\ \sin 2\theta & \cos 2\theta \end{pmatrix}$
- $\underline{\underline{\mathbf{R}}}^2 = \underline{\underline{\mathbf{I}}}$
- $\underline{\underline{\mathbf{R}}}^{-1} = \underline{\underline{\mathbf{R}}}$
- Ortogonal: $\underline{\underline{\mathbf{R}}}^{-1} = \underline{\underline{\mathbf{R}}}$
- Ímpar: $\det \underline{\underline{\mathbf{R}}} = -1$
- Isometria: $\underline{\mathbf{x}}' \cdot \underline{\mathbf{x}}' = \underline{\mathbf{x}} \cdot \underline{\mathbf{x}}$
  Preserva comprimentos
  Reverte ângulos
- Não rígida

Projeção em $\underline{\mathbf{e}} = (\cos\theta, \sin\theta)$

- $\underline{\underline{\mathbf{P}}} = \begin{pmatrix} \sin^2\theta & \sin\theta\cos\theta \\ \sin\theta\cos\theta & \cos^2\theta \end{pmatrix}$
- Nilpotente: $\underline{\underline{\mathbf{P}}}^n = \underline{\underline{\mathbf{P}}}, \;\; n \geq 1$
- Singular: $\det \underline{\underline{\mathbf{P}}} = 0$
- $\ker f = t\underline{\mathbf{f}}, \quad t \in \mathbb{R}$

Translação, $\underline{\mathbf{t}} = (t_1, t_2)$

- $f(\underline{\mathbf{x}}) = \underline{\mathbf{x}} + \underline{\mathbf{t}} = \underline{\underline{\mathbf{I}}}\ \underline{\mathbf{x}} + \underline{\mathbf{t}}$
- $\frown$: não linear...
- Afim: $f(\underline{\mathbf{x}}) = \underline{\underline{\mathbf{A}}}\ \underline{\mathbf{x}} + \underline{\mathbf{b}}$
- $\Rightarrow$ Geometria Projetiva

Projective Geometry

- $\begin{pmatrix} x \\ y \end{pmatrix} \to \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$. E: $\begin{pmatrix} x \\ y \\ z \end{pmatrix} \to \begin{pmatrix} x/z \\ y/z \end{pmatrix}$, $z \neq 0$
- Translation: $\begin{pmatrix} 1 & 0 & t_1 \\ 0 & 1 & t_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + t_1 \\ y + t_2 \\ 1 \end{pmatrix}$

  ◡: Translation is Linear!

- Rotation: $\begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \\ 1 \end{pmatrix}$
- Scaling: $\begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} \lambda_1 x \\ \lambda_2 y \\ 1 \end{pmatrix}$

Projective Geometry

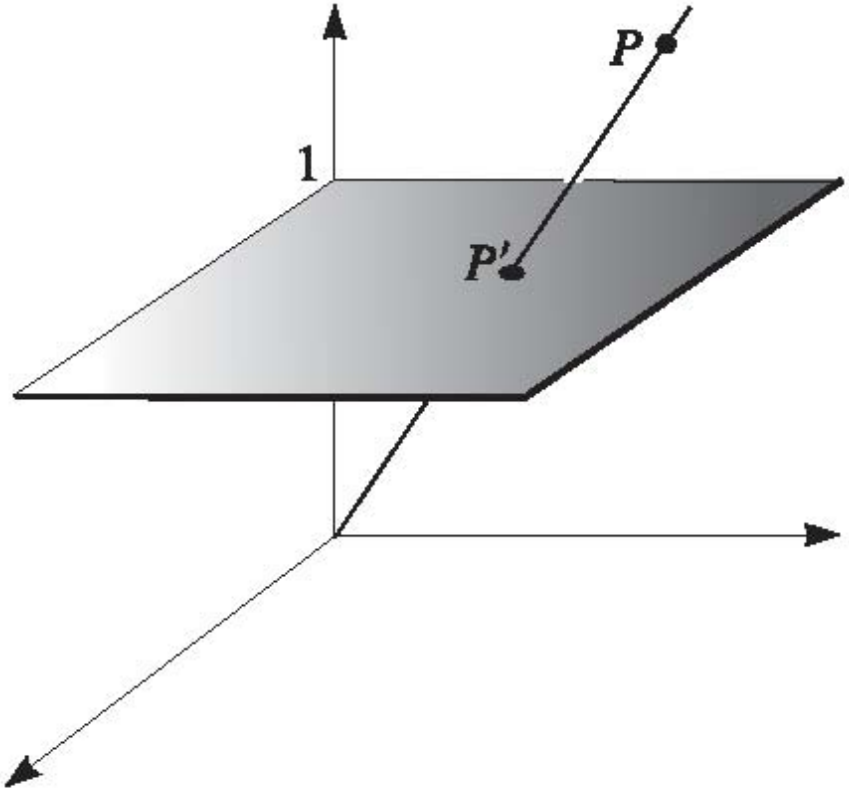


Figura 7. Plano projetivo.

Curvature

- Curve in $\mathbb{R}^2$:

$$\underline{\mathbf{r}}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} \quad - \quad \underline{\mathbf{r}}'(t) = \begin{pmatrix} x'(t) \\ y'(t) \end{pmatrix} \quad - \quad \underline{\mathbf{r}}''(t) = \begin{pmatrix} x''(t) \\ y''(t) \end{pmatrix}$$

- Accompanying Coordenate System:

$$\underline{\mathbf{t}}(t) = \frac{\underline{\mathbf{r}}'(t)}{|\underline{\mathbf{r}}'(t)|} \quad - \quad \underline{\mathbf{n}}(t) = \widehat{\underline{\mathbf{t}}}(t)$$

- Oscillating Circle, radius ρ - Curvature κ:

$$\frac{1}{\rho} = \frac{\widehat{\underline{\mathbf{r}}'}(t) \cdot \underline{\mathbf{r}}''(t)}{|\underline{\mathbf{r}}'(t)|^3} = \frac{\begin{vmatrix} x' & x'' \\ y' & y'' \end{vmatrix}}{(x'^2 + y'^2)^{3/2}}$$

- Curvature Vector $\kappa\underline{\mathbf{n}} =$

$$\frac{\widehat{\underline{\mathbf{r}}'}(t) \cdot \underline{\mathbf{r}}''(t)}{|\underline{\mathbf{r}}'(t)|^4} \; \widehat{\underline{\mathbf{r}}'}(t)$$

Calculate Curvature & Center

```
Vector tan=this->Tangent(t);
Vector acc=this->Acceleration(t);

double len=tan.Norm2();
double kappa=Determinant2(tan,acc)/(len*len*len);

tan.Normalize();
Vector n=tan.Normal2();

n*=kappa;
p+=n;

return p;
```

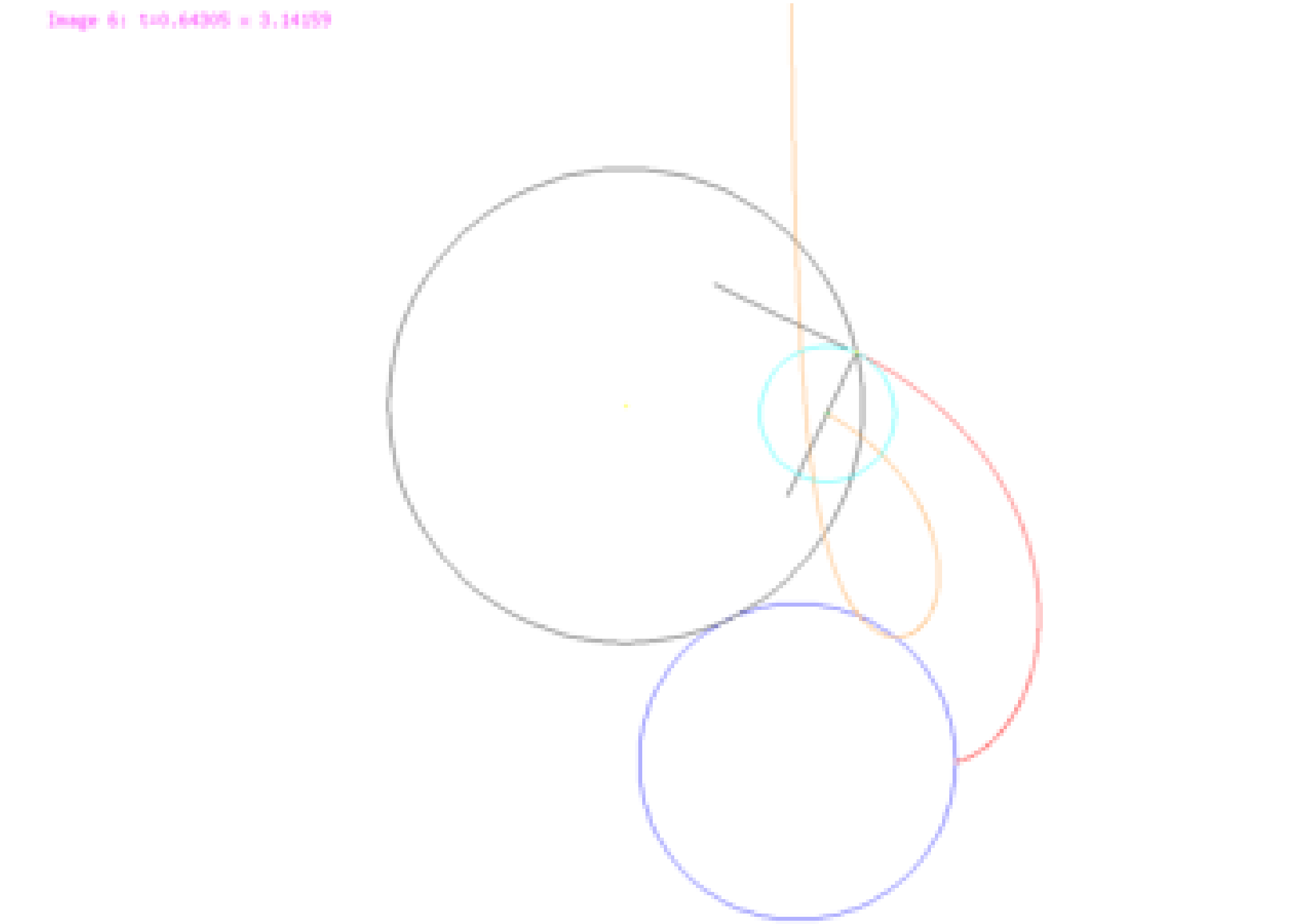


Image 6: t=0.64305 = 3.14159

Evolute

- Center of Curvature:

$$\underline{\mathbf{r}}_C(t) = \underline{\mathbf{r}}(t) + \kappa \underline{\mathbf{n}}(t)$$

- Center of Curvature:

$$\underline{\mathbf{r}}_C(t) = \underline{\mathbf{r}}(t) + \frac{\begin{vmatrix} x' & x'' \\ y' & y'' \end{vmatrix}}{(x'^2 + y'^2)^2} \begin{pmatrix} -y' \\ x' \end{pmatrix}$$

- Evolute: Curve of Centers of Curvature

Calculate & Draw Evolute

```
double dt=(t2-t1)/(1.0*(n-1));
//First point
Vector rant=this->CalcPoint(t1);
Vector revant=this->CenterOfCurvature2(t1);
double t=t1+dt;
for (int i=1;i<n;i++)
{
    Vector rnext=this->CalcPoint(t);
    Vector revnext=this->CenterOfCurvature2(t);
    graph->DrawLine(rant,rnext,color);
    graph->DrawLine(revant,revnext,colorev);
    rant=rnext;
    revant=revnext;
    t+=dt;
}
graph->DrawPoint(rant,color);
graph->DrawPoint(revant,colorev);
```

Cycloid

- Mark Point on Circle, radius a, and Let it Roll
- $\underline{\mathbf{r}}(t) = \begin{pmatrix} at - a\sin t \\ a - a\cos t \end{pmatrix}$
- $\underline{\mathbf{r}}'(t) = \begin{pmatrix} a - a\cos t \\ a\sin t \end{pmatrix}$
- $\underline{\mathbf{r}}''(t) = \begin{pmatrix} a\sin t \\ a\cos t \end{pmatrix}$
- $\kappa =$
$$\frac{a^2(\cos t - 1)}{(2a^2(1-\cos t))^{3/2}} = \frac{-1}{2\sqrt{2}a(1-\cos t)^{1/2}}$$
- $\underline{\mathbf{r}}_c = \underline{\mathbf{r}} + \kappa\underline{\mathbf{n}} = \begin{pmatrix} at - a\sin t \\ a - a\cos t \end{pmatrix} - \frac{1}{4a^2(1-\cos t)}\begin{pmatrix} -a\sin t \\ a - a\cos t \end{pmatrix}$
- Cycloids!!!

Define Cycloid

```
double a;
double X(double t) { return a*(t-sin(t)); }
double Y(double t) { return a*(1-cos(t)); }
double Kappa(double t)
{
    double div=2*sqrt(2)*a*sqrt(1-cos(t));
    return -1.0/div;
}
...
Curve *curve=new Curve(2,n);
curve->SetR(0,X);
curve->SetR(1,Y);
curve->SetCurvature(Kappa);
curve->DrawWithEvolute(graph,n,0.0,theta);
```

Trochoid

- Mark Point distance b from Center of Circle, radius a, and Let it Roll
- $a = b$: Cycloid
- $\underline{\mathbf{r}}(t) = \begin{pmatrix} at - b\sin t \\ a - b\cos t \end{pmatrix}$
- $\underline{\mathbf{r}}'(t) = \begin{pmatrix} a - b\cos t \\ b\sin t \end{pmatrix}$
- $\underline{\mathbf{r}}''(t) = \begin{pmatrix} a\sin t \\ -b\cos t \end{pmatrix}$
- $\kappa = \dfrac{ab\cos t - b^2}{(a^2 + b^2 - 2ab\cos t)^{3/2}}$
- $\underline{\mathbf{r}}_c = \underline{\mathbf{r}} + \kappa\underline{\mathbf{n}} =$
  $\begin{pmatrix} at - b\sin t \\ a - b\cos t \end{pmatrix} + \dfrac{ab\cos t - b^2}{(a^2 + b^2 - 2ab\cos t)^2} \begin{pmatrix} -b\sin t \\ a - b\cos t \end{pmatrix}$
- Trochoids!!!

Define Trochoid

```
double a,b;
double X(double t) { return a*t-b*sin(t); }
double Y(double t) { return a*1-b*cos(t); }
double Kappa(double t)
{
    double denom=a*a+b*b-2.0*a*b*cos(t);
    denom=sqrt(denom)*denom;

    return (a*b*cos(t)-b*b)/denom;
}
...
Curve *curve=new Curve(2,n);
curve->SetR(0,X);
curve->SetR(1,Y);
curve->SetCurvature(Kappa);
curve->DrawWithEvolute(graph,n,0.0,theta);
```

Unit Vectors

- $\underline{\mathbf{e}}(t) = \begin{pmatrix} \cos t \\ \sin t \end{pmatrix}$
- $\underline{\mathbf{f}}(t) = \begin{pmatrix} -\sin t \\ \cos t \end{pmatrix}$
- Derivatives:

$$\underline{\mathbf{e}}'(t) = \widehat{\underline{\mathbf{e}}}(t) = \underline{\mathbf{f}}(t)$$
$$\underline{\mathbf{f}}'(t) = \widehat{\underline{\mathbf{f}}}(t) = -\underline{\mathbf{e}}(t)$$

- Dotproducts:

$$\underline{\mathbf{e}}(t_1) \cdot \underline{\mathbf{e}}(t_2) = \underline{\mathbf{f}}(t_1) \cdot \underline{\mathbf{f}}(t_2) = \cos(t_1 - t_2)$$
$$\underline{\mathbf{e}}(t_1) \cdot \underline{\mathbf{f}}(t_2) = -\underline{\mathbf{f}}(t_1) \cdot \underline{\mathbf{e}}(t_2) = \sin(t_1 - t_2)$$

- Note:

$$\underline{\mathbf{e}}(\eta t) = \cos(\mu t)\underline{\mathbf{e}}(t) + \sin(\mu t)\underline{\mathbf{f}}(t)$$
$$\underline{\mathbf{f}}(\eta t) = -\sin(\mu t)\underline{\mathbf{e}}(t) + \cos(\mu t)\underline{\mathbf{f}}(t)$$

With: $\mu = \eta - 1$.

Epicycloid I

- Circle, Radius r, Rolls *outside* Circle, Radius R
- $\underline{\mathbf{r}}(t) =$

$$\begin{pmatrix} (R+r)\cos t - r\cos\left(\frac{R+r}{r}t\right) \\ (R+r)\sin t - r\sin\left(\frac{R+r}{r}t\right) \end{pmatrix} =$$

$$r\left[\eta\ \underline{\mathbf{e}}(t) - \underline{\mathbf{e}}\left(\eta t\right)\right]$$

Dimensionlesses: $\eta = \dfrac{R+r}{r} > 1$ e $\mu = \dfrac{R}{r} = \eta - 1 > 0$.

- $\underline{\mathbf{r}}'(t) =$

$$r\eta\left[\underline{\mathbf{f}}(t) - \underline{\mathbf{f}}\left(\eta t\right)\right]$$

- $\underline{\mathbf{r}}''(t) =$

$$r\eta\left[-\underline{\mathbf{e}}(t) + \eta\underline{\mathbf{e}}\left(\eta t\right)\right]$$

- $\widehat{\underline{\mathbf{r}}'}(t) =$

$$r\eta\left[-\underline{\mathbf{e}}(t) + \underline{\mathbf{e}}\left(\eta t\right)\right]$$

Epicycloid II

- $\widehat{\underline{\mathbf{r}}'}(t) \cdot \underline{\mathbf{r}}''(t) =$

$$(r\eta)^2 \left[1 + \eta - (1+\eta)\cos(\eta - 1)t\right] =$$

$$(r\eta)^2 (1+\eta) \left[1 - \cos \mu t\right]$$

- $\underline{\mathbf{r}}'(t) \cdot \underline{\mathbf{r}}'(t) =$

$$(r\eta)^2 \left[1 + 1 - 2\cos(\eta - 1)t\right] =$$

$$2(r\eta)^2 \left[1 - \cos \mu t\right] =$$

- $\kappa =$

$$\frac{(r\eta)^2 (1+\eta) \left[1 - \cos \mu t\right]}{\left(2(r\eta)^2 \left[1 - \cos \mu t\right]\right)^{3/2}} =$$

$$\frac{(1+\eta)}{2\sqrt{2} r\eta} \left[1 - \cos \mu t\right]^{-1/2}$$

Epicycloid III

- $\kappa = +\infty$ para $\mu t = 2p\pi, \ p \in \mathbb{Z}$
- $\kappa \underline{\mathbf{n}} = \frac{\kappa}{|\underline{\mathbf{r}}(t)|} \widehat{\underline{\mathbf{r}}'}(t) =$

$$\frac{(r\eta)^2(1+\eta)\left[1-\cos\mu t\right]}{\left(2(r\eta)^2\left[1-\cos\mu t\right]\right)^2} r\eta\left[-\underline{\mathbf{e}}(t)+\underline{\mathbf{e}}\left(\eta t\right)\right]$$

$$\frac{(1+\eta)}{4r\eta}\left[1-\cos\mu t\right]^{-1}\left\{-\underline{\mathbf{e}}(t)+\underline{\mathbf{e}}\left(\eta t\right)\right\}$$

- Evolute:

$$\underline{\mathbf{r}}_C(t) = \underline{\mathbf{r}}(t) + \kappa\underline{\mathbf{n}} =$$

$$r\left[\eta\ \underline{\mathbf{e}}(t) - \underline{\mathbf{e}}\left(\eta t\right)\right] + \frac{(1+\eta)}{4r\eta}\left[1-\cos\mu t\right]^{-1}\left\{-\underline{\mathbf{e}}(t)+\underline{\mathbf{e}}\left(\eta t\right)\right\}$$

- Epicycloids!!!

Define Epicycloid

```
double a,b;
double X(double t) { return (a+b)*cos(t)-r*cos( (a+b)/r*t ); }
double Y(double t) { return (a+b)*sin(t)-r*sin( (a+b)/r*t ); }
double Kappa(double t)
{
    double denom=2.0*sqrt(2.0)*b*(b+a)*sqrt(1-cos(a/r*t));
    double val=(2.0*b+a)/denom;

    return val;
}
...
Curve *curve=new Curve(2,n);
curve->SetR(0,X);
curve->SetR(1,Y);
curve->SetCurvature(Kappa);
curve->DrawWithEvolute(graph,n,0.0,theta);
```

Hipocycloid I

- Circle, Radius r, Rolls *inside* Circle, Radius R
- $\underline{\mathbf{r}}(t) =$

$$\begin{pmatrix} (R-r)\cos t + r\cos\left(\frac{r-R}{r}t\right) \\ (R-r)\sin t + r\sin\left(\frac{r-R}{r}t\right) \end{pmatrix} =$$

$$r\left[-\eta\ \underline{\mathbf{e}}(t) + \underline{\mathbf{e}}\left(\eta t\right)\right]$$

Dimensionlesses: $\eta = \dfrac{r-R}{r} \leq 0$ e $\mu = \dfrac{R}{r} = 1 - \eta \geq 1.$

- $\underline{\mathbf{r}}'(t) =$

$$r\eta\ \left[-\underline{\mathbf{f}}(t) + \underline{\mathbf{f}}\left(\eta t\right)\right]$$

- $\underline{\mathbf{r}}''(t) =$

$$r\eta\ \left[\underline{\mathbf{e}}(t) - \eta\ \underline{\mathbf{e}}\left(\eta t\right)\right]$$

- $\widehat{\underline{\mathbf{r}}'}(t) =$

$$r\eta\ \left[\underline{\mathbf{e}}(t) - \underline{\mathbf{e}}\left(\eta t\right)\right]$$

Hipocycloid II

- $\widehat{\underline{\mathbf{r}}'}(t) \cdot \underline{\mathbf{r}}''(t) =$

$$(r\eta)^2 \left[1 + \eta - (1+\eta)\cos(\eta - 1)t\right] =$$

$$(r\eta)^2 (1+\eta) \left[1 - \cos \mu t\right]$$

- $\underline{\mathbf{r}}'(t) \cdot \underline{\mathbf{r}}'(t) =$

$$(r\eta)^2 \left[1 + 1 - 2\cos(\eta - 1)t\right] =$$

$$2(r\eta)^2 \left[1 - \cos \mu t\right] =$$

- $\kappa =$

$$\frac{(r\eta)^2 (1+\eta) \left[1 - \cos \mu t\right]}{\left(2(r\eta)^2 \left[1 - \cos \mu t\right]\right)^{3/2}} =$$

$$\frac{(1+\eta)}{2\sqrt{2} r\eta} \left[1 - \cos \mu t\right]^{-1/2}$$

Hipocycloid III

- $\kappa = +\infty$ para $\mu t = \frac{\pi}{2} + 2p\pi, \ p \in \mathbb{Z}$
- $\kappa \underline{\mathbf{n}} =$

$$\frac{(r\eta)^2(1+\eta)\left[1-\cos\mu t\right]}{\left(2(r\eta)^2\left[1-\cos\mu t\right]\right)^2} r\eta\left\{-\underline{\mathbf{e}}(t)+\underline{\mathbf{e}}\left(\eta t\right)\right\} =$$

$$\frac{(1+\eta)}{4r\eta}\left[1-\cos\mu t\right]^{-1}\left\{-\underline{\mathbf{e}}(t)+\underline{\mathbf{e}}\left(\eta t\right)\right\}$$

- Evolute:

$$\underline{\mathbf{r}}_C(t) = \underline{\mathbf{r}}(t) + \kappa\underline{\mathbf{n}} =$$

$$r\left[-\eta\ \underline{\mathbf{e}}(t)+\underline{\mathbf{e}}\left(\eta t\right)\right] + \frac{(1+\eta)}{4(r\eta)^2}\left[1-\cos\mu t\right]^{-1}\left\{-\underline{\mathbf{e}}(t)+\underline{\mathbf{e}}\left(\eta t\right)\right\}$$

- Hypocycloids!!!

Define Hypocycloid

```
double a,b;
double X(double t) { return (a-b)*cos(t)+b*cos( (a-b)/r*t ); }
double Y(double t) { return (a-b)*sin(t)-b*sin( (a-b)/r*t ); }
double Kappa(double t)
{
    double denom=2.0*sqrt(2.0)*b*(b-a)*sqrt(1-cos(a/b*t));
    double val=(2.0*b-a)/denom;

    return val;
}
...
Curve *curve=new Curve(2,n);
curve->SetR(0,X);
curve->SetR(1,Y);
curve->SetCurvature(Kappa);
curve->DrawWithEvolute(graph,n,0.0,theta);
```

Epicycloid & Hipocycloid

- $\kappa_{Hypo} =$

$$\frac{(1+\frac{r-R}{r})}{2\sqrt{2}r\frac{r-R}{r}}\left[1-\cos\left(\frac{R}{r}t\right)\right]^{-1/2} =$$

$$\frac{2r-R}{2\sqrt{2}r(r-R)}\left[1-\cos\left(\frac{R}{r}t\right)\right]^{-1/2}$$

- $\kappa_{Epi} =$

$$\frac{(1+\frac{R+r}{r})}{2\sqrt{2}r\frac{R+r}{r}}\left[1-\cos\left(\frac{R}{r}t\right)\right]^{-1/2} =$$

$$\frac{2r+R}{2\sqrt{2}r(r+R)}\left[1-\cos\left(\frac{R}{r}t\right)\right]^{-1/2}$$

Hypotrochoid & Epitrochoid

- Hypotrochoid: $\underline{\mathbf{r}}(t) =$

$$\begin{pmatrix} (R-r)\cos t + d\cos\left(\frac{r-R}{r}t\right) \\ (R-r)\sin t + d\sin\left(\frac{r-R}{r}t\right) \end{pmatrix}$$

- Hypotrochoids...
- Epitrochoid: $\underline{\mathbf{r}}(t) =$

$$\begin{pmatrix} (R+r)\cos t - d\cos\left(\frac{R+r}{r}t\right) \\ (R+r)\sin t - d\sin\left(\frac{R+r}{r}t\right) \end{pmatrix}$$

- Epitrochoids...

Rolling

- Circle, Radius R, Rolls on Curve:

$$\underline{\mathbf{r}}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}$$

- $$\underline{\mathbf{r}}_R(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} + \frac{R}{(x'(t)^2 + y'(t)^2)^{1/2}} \begin{pmatrix} -y'(t) \\ x(t) \end{pmatrix}$$

- Oloids...

Defining Rolling

```
Vector Curve::CalcRollingCenter(double t,double a,Vector &rant,
{
    Vector tan=this->UnitTangent(t);
    Vector n=tan.Normal2();
    n*=a;
    Vector rc;
    if (inside==1) { rc=rant-n; }
    else           { rc=rant+n; }
    return rc;
}
Vector Curve::CalcRollingPoint(double s,double a,Vector &rc)
{
    Vector *p=Vector2(a*cos(s/a-PI),a*sin(s/a-PI));
    (*p)+=rc;
    return *p;
}
```

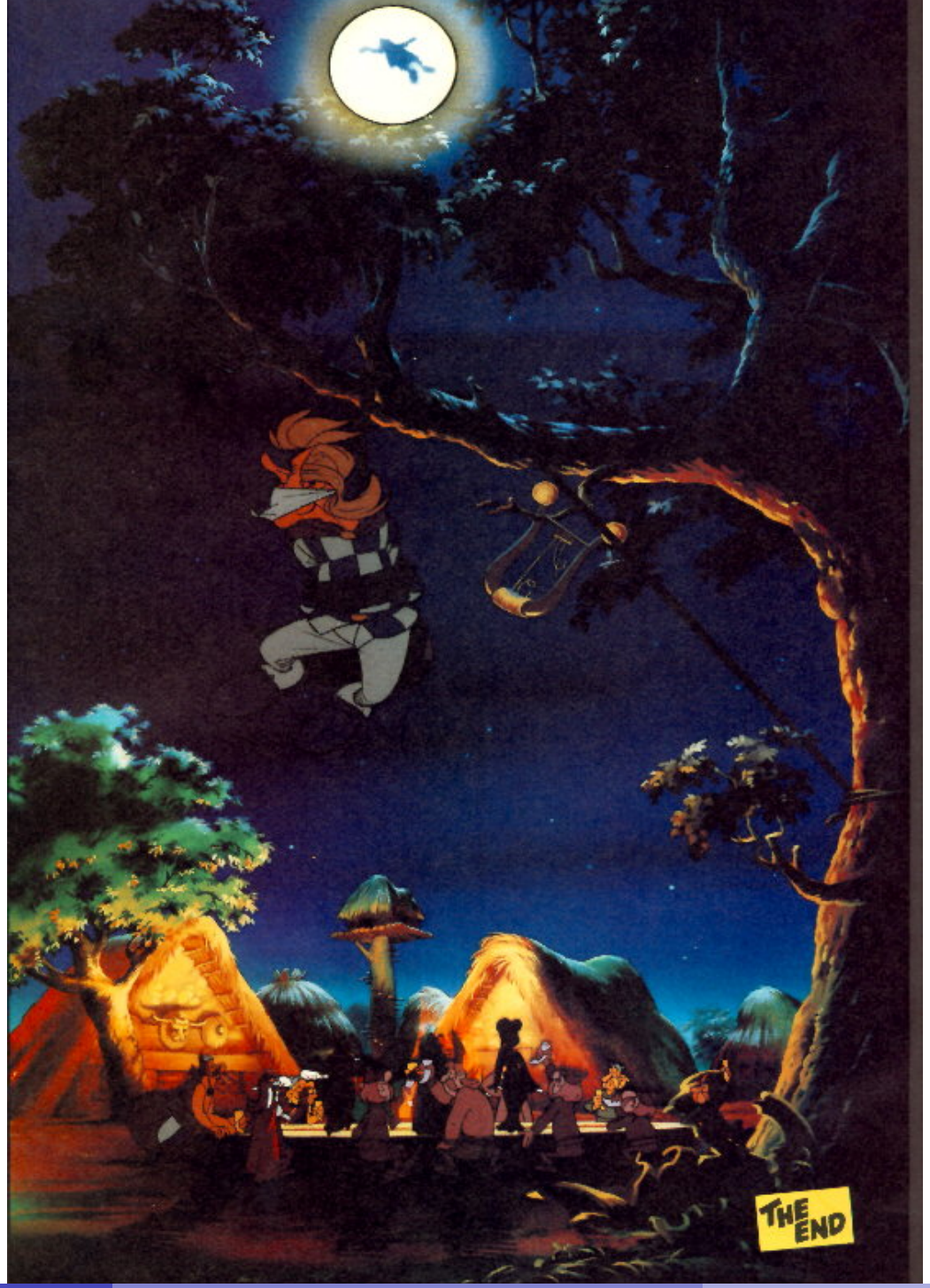
Defining Rolling

```
double dt=(t2-t1)/(1.0*(n-1));
Vector rant=this->CalcPoint(t1);
Vector rcant=this->CalcRollingCenter(t1,a,rant,inside);
Vector rpant=this->CalcRollingPoint(0.0,a,rcant);
double t=t1+dt, s=0.0;  //parameter & arc length
for (int i=1;i<n;i++)
{
    Vector rnext=this->CalcPoint(t);
    Vector rcnext=this->CalcRollingCenter(t,a,rnext,inside)
    Vector rpnext=this->CalcRollingPoint(s,a,rcnext);
    graph->DrawLine(rpant,rpnext,color);
    s+=Distance(rant,rnext);
    rant=rnext;
    rcant=rcnext;
    rpant=rpnext;
 }
```

Defining Rolling

```
double r;

double RX(double t)
{
    return t;
}


double RY(double t)
{
    return r*sin(t);
}
```

Thanks!