Computational Geometry: The Oids...

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

14/04/2010

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

Introduction

- Computational Geometry
- Image vs. Animation
- Motivation:
- Great fun!
- Testing Numerical Algorithms
- Linear Algebra
- Numerical Derivation

- Den som kun tager sjov for sjov
Og alvor kun alvorligt
Han har faktisk forstået begge lige dårligt
Piet Hein

Geometrical Transformations

- Linearity: $T(\underline{\mathbf{x}}) = \underline{\underline{\mathbf{A}}}\ \underline{\mathbf{x}}$
- Orthogonality: $\underline{\underline{\mathbf{A}}}^T \underline{\underline{\mathbf{A}}} = \underline{\underline{\mathbf{I}}}$
- Affinity: $T(\underline{\mathbf{x}}) = \underline{\underline{\mathbf{A}}}\ \underline{\mathbf{x}} + \underline{\mathbf{b}}$
- Isometry: $T(\underline{\mathbf{x}}_1) \cdot T(\underline{\mathbf{x}}_2) = \underline{\mathbf{x}}_1 \cdot \underline{\mathbf{x}}_2$
-

Type	Rigid	Orthogonal	Isometry	Linear	Affin
Translation	Y	-	Y	N	Y
Rotation	Y	Y Even	Y	Y	Y
Scaling	N	N	N	Y	Y
Reflection	N	Y Odd	N	Y	Y

- $\frown$: Translations Non-linear...
- Remedy: Projective Geometries

Projective Geometry

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

  $\smile$: 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: $\begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} \lambda_1 x \\ \lambda_2 y \\ 1 \end{pmatrix}$

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;
```

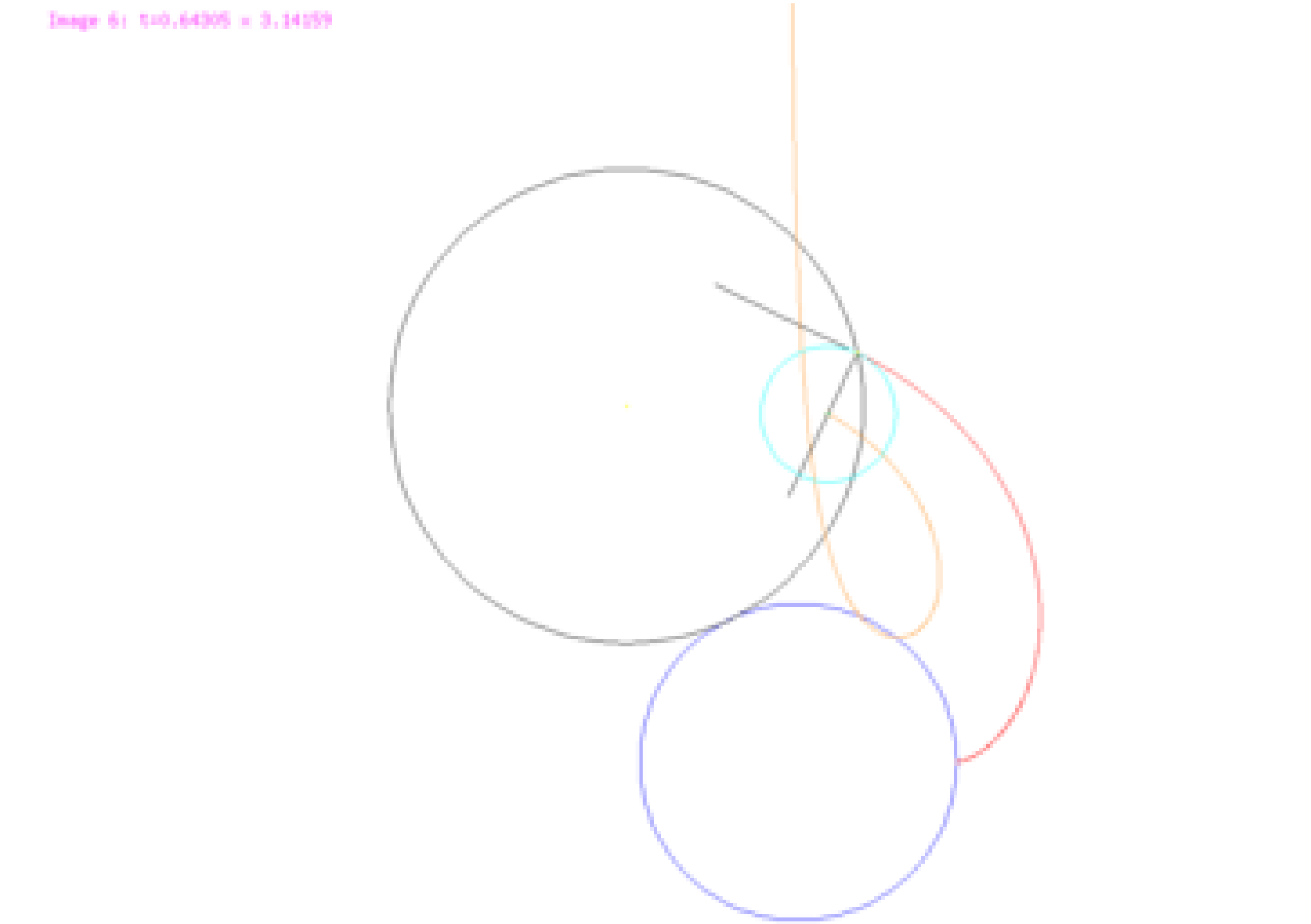


Image 6: 540,64305 = 3,14159

Evolute

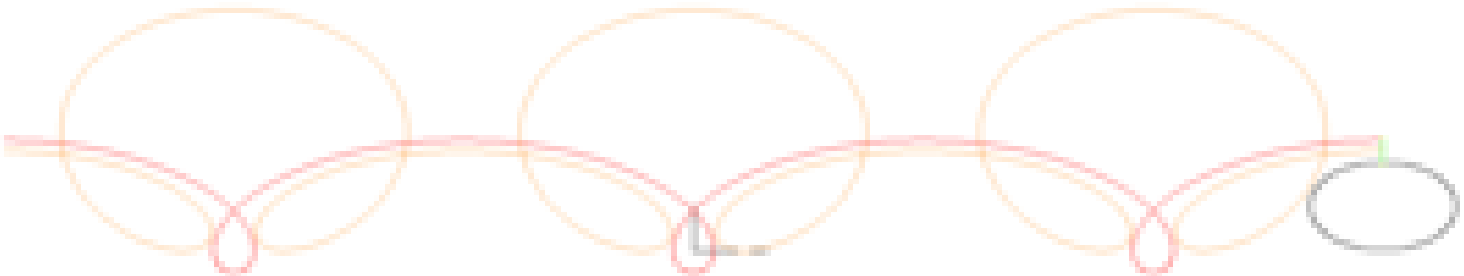
- Center of Curvature:

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

- 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
- $\underline{\mathbf{r}}(t) = \begin{pmatrix} at - a\sin t \\ a - a\cos t \end{pmatrix}$
- $\underline{\mathbf{r}}'(t) = \begin{pmatrix} a - a\cos t \\ a\sin t \end{pmatrix}$
- $\underline{\mathbf{r}}''(t) = \begin{pmatrix} a\sin t \\ a\cos t \end{pmatrix}$
- $\kappa = \dfrac{a^2(\cos t - 1)}{(2a^2(1-\cos t))^{3/2}} = \dfrac{-1}{2\sqrt{2}a(1-\cos t)^{1/2}}$
- $\underline{\mathbf{r}}_c = \underline{\mathbf{r}} + \kappa\underline{\mathbf{n}} = \begin{pmatrix} at - a\sin t \\ a - a\cos t \end{pmatrix} - \dfrac{1}{4a^2(1-\cos t)} \begin{pmatrix} -a\sin t \\ a - a\cos t \end{pmatrix}$
- 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!