

Rovinné křivky

Křivky typu $y = f(x)$

Konstrukce pomocí Line:

```
var x,hx:Double;
    A,B :TPoint;
function f(x:Double):Double;
begin if x=0 then f:=0
      else f:=x+10*x*x*sin(1/x); end;
```

```
begin
  With Draw2D do
  begin
    .....
    if CheckBox1.Checked then
    begin
      GaugedXAxis(x1,x2,0,Red,Green,Blue);GaugedYAxis(y1,y2,0,clRed);
      end;
      .....
      hx:=(x2-x1)/NumberOfSegments;x:=x1;A[1]:=x;A[2]:=f(x);
      Repeat
        B[1]:=x+hx;B[2]:=f(x+hx); Line(A,B,slBlue);A:=B;x:=x+hx
      Until x>x2
      end;
    end;
```

{Uživatelská kreslicí plocha}

{souřadné osy a jejich popis}

{barva křivky}

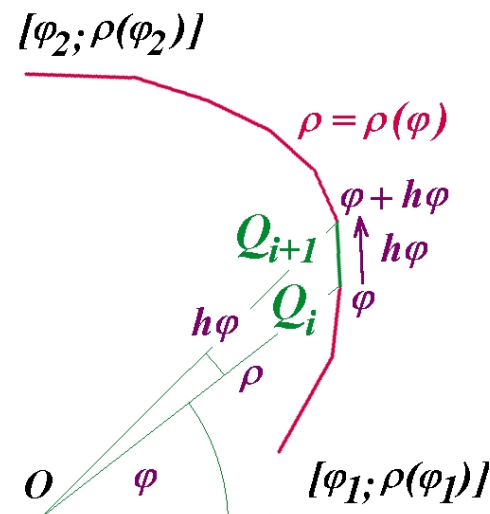
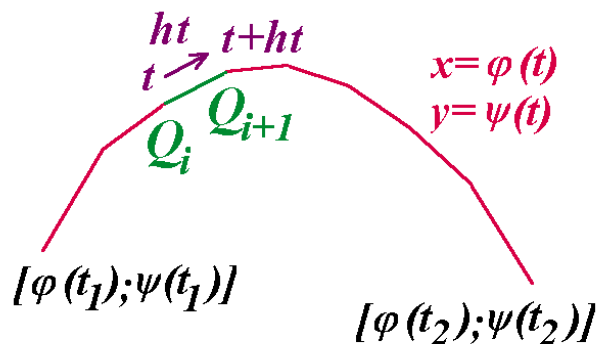
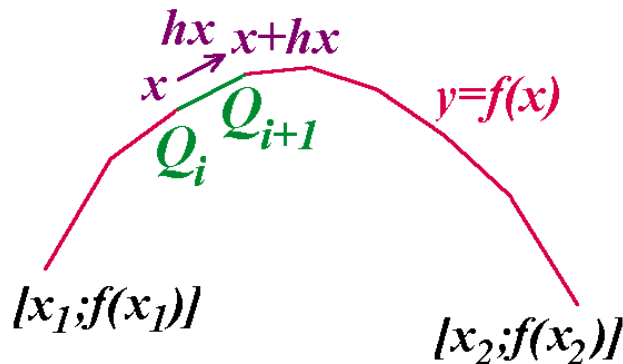
{konstrukce křivky jako lomené čáry}

[Editovat příklad](#) [Zdrojový kód](#) [Spustit](#)

Použití typu Polyline:

```
var x,hx:Double;  Q  :TArrayOfPoints;  i  :Word;
```

```
function f.....
begin  ....
    hx:=(x2-x1)/NumberOfSegments;  i:=0;x:=x1;
    Repeat
        {konstrukce křivky jako lomené čáry}
        Q[i,1]:=x;Q[i,2]:=f(x);x:=x+hx;i:=succ(i);
    Until  x>x2+hx;
    PolyLine(Q,i,Color);
end;
```



parametricky

$x, x1, x2, hx \rightarrow t, t1, t2, ht$

[Editovat příklad](#) [Zdrojový kód](#) [Spustit](#)

Parametrický systém křivek:

Částečné součty Taylorových a Fourierových řad:

Křivky zadané v polárních souřadnicích

$x, x1, x2, hx \rightarrow \varphi, \varphi1, \varphi2, h\varphi$

[Editovat příklad](#) [Zdrojový kód](#) [Spustit](#)

Screen Saver: [Spustit](#)

Křivky
zadané

Křivky zadané

$x, x1, x2, hx \rightarrow \varphi,$

Lagrangeův interpolační polynom. Z numerické matematiky víme, že polynom procházející body $[x_i; y_i]; i=1, \dots, n$ je

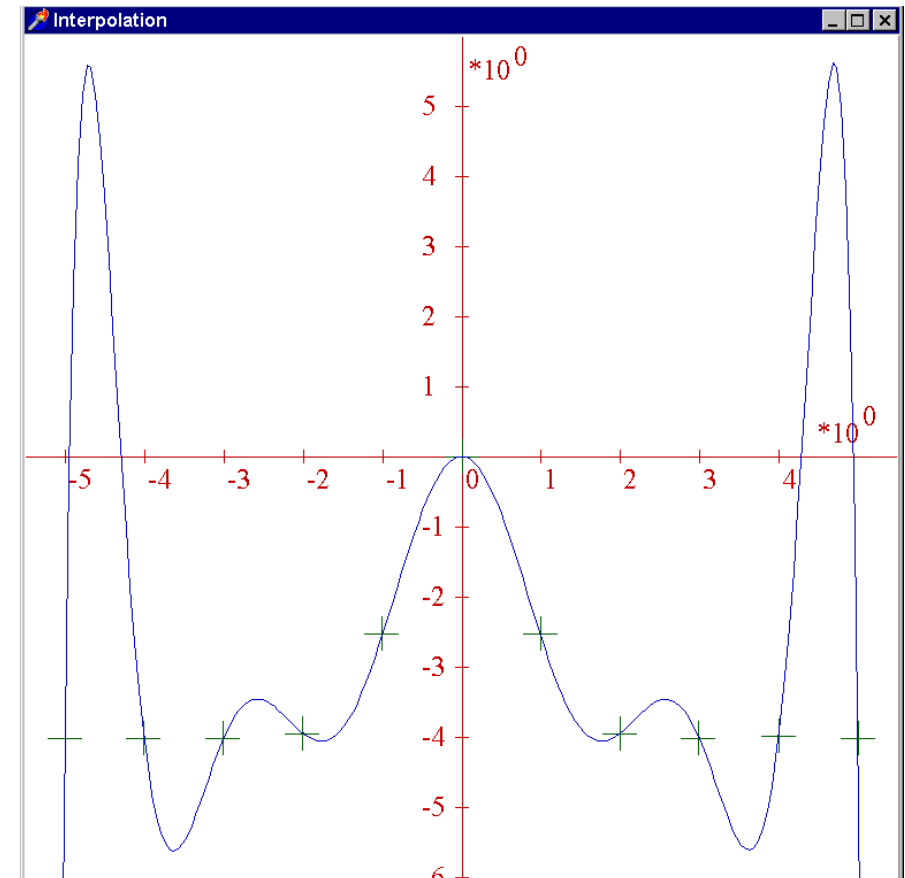
$$y = \sum_{i=1}^n y_i \frac{\prod_{j=1; j \neq i}^n (x - x_j)}{\prod_{j=1; j \neq i}^n (x_i - x_j)}$$

Tento vzorec je vyčíslován funkcí Lagrange:

```

Function Lagrange(z:Double):Double
var i,j           :Integer;
    Lagr,Citatel,Jmenovatel:Double;
begin.
    Lagr:=0;
    for i:=1 to n do
        begin
            Citatel:=1;Jmenovatel:=1;
            for j:=1 to n do
                if i<>j then
                    begin
                        Citatel:=Citatel*(z-x[j]);
                        Jmenovatel:=Jmenovatel*(x[i]-x[j]);
                    end;
            Lagr:=Lagr+y[i]*Citatel/Jmenovatel;
        end;
    Lagrange:=Lagr;  end;

```



[Spustit](#)

Fergusonovy křivky

Technická praxe často vyžaduje křivky, určené tzv. řídicími body nebo o řídicím polygonem. Jednu z nejjednodušších takových křivek používal od r. 1964 J. C. Ferguson. Křivka je určena dvěma krajními body P_0 , P_2 a dvěma tečnými vektory $\overrightarrow{P_0P_1}$, $\overrightarrow{P_2P_3}$. Velikost těchto vektorů ovlivňuje zároveň druhou derivaci křivky. Křivka je definována parametrickými rovnicemi

$$Q(t) = \sum_{i=0}^3 P_i F_i(t); \quad t \in \langle 0; 1 \rangle; \quad F_0(t) = 2t^3 - 3t^2 + 1; \quad F_1(t) = -2t^3 - 3t^2; \quad F_2(t) = t^3 - 2t^2 + t; \quad F_3(t) = t^3 - t^2$$

Funkce F_i jsou polynomy 3. stupně, Fergusonova křivka je tedy kubická parabola.

```
Procedure Ferguson(t:Double; var Q:TPoint);
```

begin

$$Q[1] := P[0,1]^*(2*t*t*t-3*t*t+1) + P[1,1]^*(-2*t*t*t+3*t*t) + P[2,1]^*(t*t*t-2*t*t+1) + P[3,1]^*(t*t*t-t*t);$$
$$Q[2] := P[0,2]^* \{ \dots \} + P[1,2]^* \{ \dots \} + P[2,2]^* \{ \dots \} + P[3,2]^* \{ \dots \}$$

end;

begin

$$x1:=5;x2:=300;y1:=10;y2:=300;$$

```
Scale(x1,x2,y1,y2);
```

$$P[0,1] := 10; \quad P[0,2] := 10; \quad P[1,1] := 30; \quad P[1,2] := 150;$$
$$P[2,1] := 90; \quad P[2,2] := 250; \quad P[3,1] := 140; \quad P[3,2] := 170;$$

```
Line(P[0],P[1],200,0,0);Line(P[2],P[3],200,0,0);
```

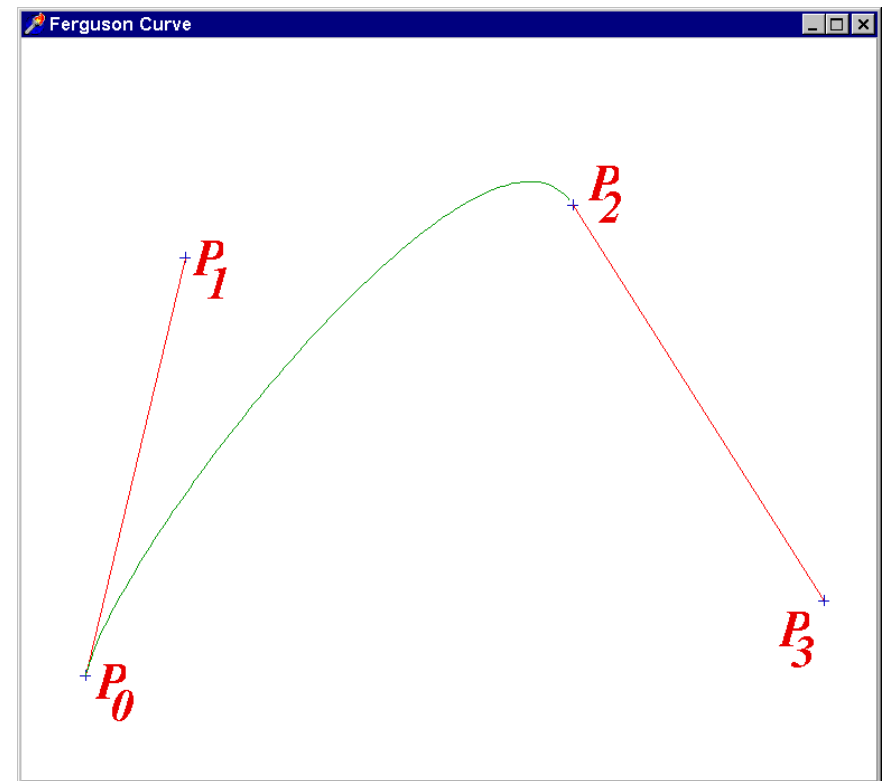
```
ht:=0.01; t:=0;i:=1;
```

While $t \leq 1$ **do**

```
begin Ferguson(t,Q[i]);i:=succ(i);t:=t+ht; end;
```

```
PolyLine(Q,i-1,0,200,0);
```

end;



Fergusonova křivka: [spustit](#)

Napojování: [spustit](#)

Bézierovy křivky

V letech 1959 - 1962 navrhli nezávisle na sobě P. E. Béziere a P. de Casteljau křivku, která je určena čtyřmi body P_0, P_1, P_2, P_3 . Je definována parametrickými rovnicemi

$$Q(t) = \sum_{i=0}^3 P_i B_i(t); \quad t \in \langle 0; 1 \rangle;$$

$$B_0(t) = (1-t)^3; \quad B_1(t) = 3t(1-t)^2; \quad B_2(t) = 3t^2(1-t); \quad B_3(t) = t^3$$

Procedure Beziere(t:Double;var Q:TPoint);

begin

$Q[1] := P[0,1] * (1-t) * (1-t) * (1-t) + P[1,1] * 3 * t * (1-t) * (1-t) + P[2,1] * 3 * t * t * (1-t) + P[3,1] * t * t * t;$

$Q[2] := P[0,2] * \{...\} + P[1,2] * \{...\} + P[2,2] * \{...\} + P[3,2] * \{...\}$

end;

begin

$x1 := 5; x2 := 300; y1 := 10; y2 := 300; \text{Scale}(x1, x2, y1, y2);$

$P[1,1] := 10; P[1,2] := 10; P[2,1] := 30; P[2,2] := 150;$

$P[3,1] := 90; P[3,2] := 250; P[4,1] := 140; P[4,2] := 170;$

$\text{Line}(P[1], P[2], \text{Red}, \text{Green}, \text{Blue}); \text{Line}(P[2], P[3], \text{Red}, \text{Green}, \text{Blue});$

$\text{Line}(P[3], P[4], \text{Red}, \text{Green}, \text{Blue});$

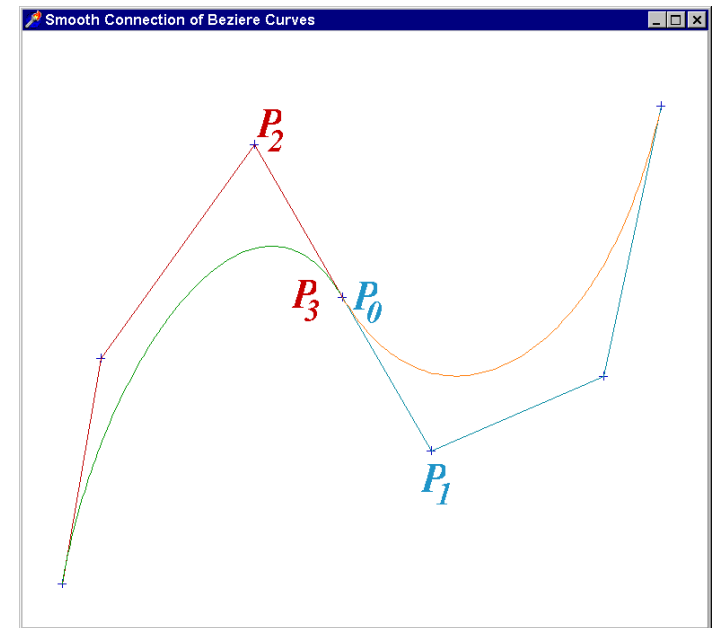
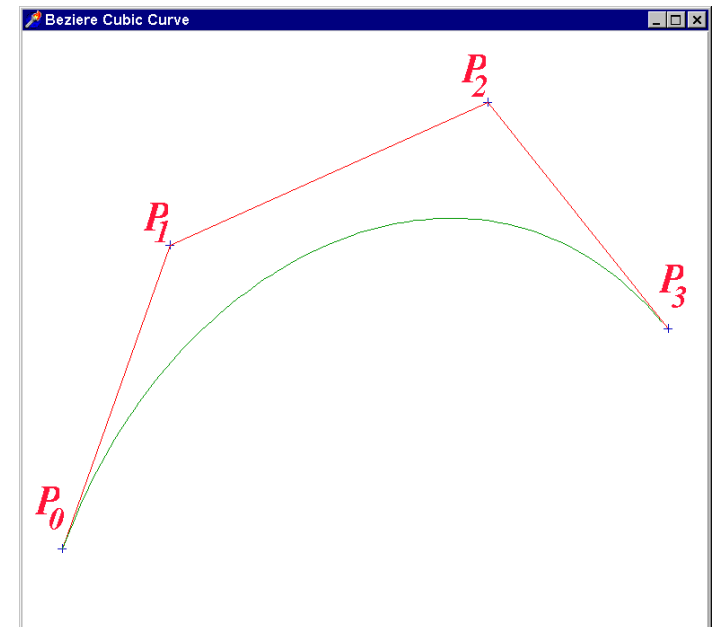
$ht := 0.01; t := 0; i := 1;$

While $t < 1$ **do**

begin $\text{Beziere}(t, Q[i]); i := \text{succ}(i); t := t + ht; \text{end};$

$\text{PolyLine}(C, i-1, 0, \text{Red}, \text{Green}, \text{Blue});$

end;

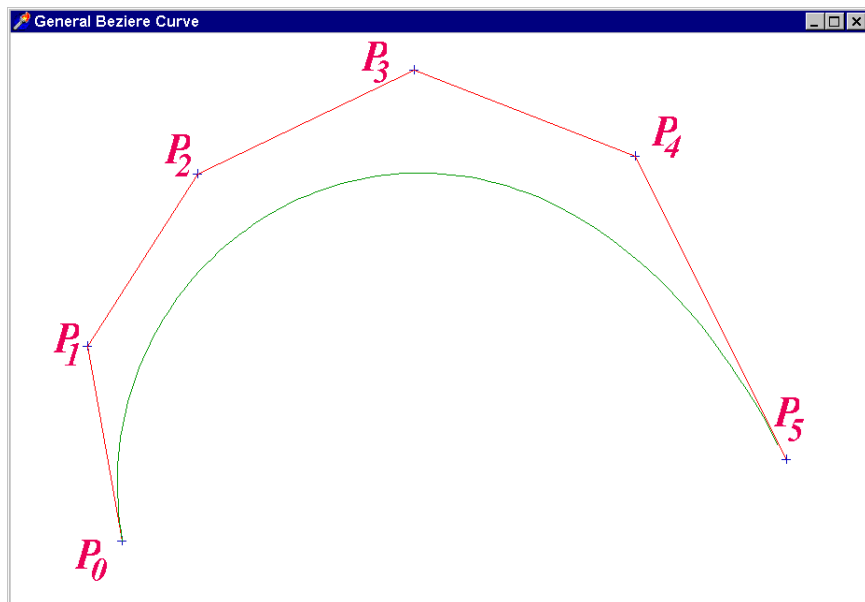


Bezierova křivka: [Zdrojový kód](#) [Spustit](#) Algoritmus de Casteljau: [spustit](#)

Hladké spojení: [spustit](#)

Hladké spojení v DESIGN CAD : [spustit](#)

Obecná Bézierova křivka: Křivka n tého stupně vznikne pomocí $n+1$ řídících bodů a je určena vztahem



$$Q(t) = \sum_{i=1}^n P_i B_i^n(t); \quad t \in \langle 0; 1 \rangle; \quad B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

V programovém zpracování přibývá výpočet faktoriálu

a kombinačního čísla $\binom{n}{i} = \frac{n!}{(n-i)!i!}$

Function Factorial(n:Integer):LongInt;

var Fa,i:LongInt;

begin

 Fa:=1;

if n>1 **then for** i:=2 **to** n **do** Fa:=Fa*i;

 Factorial:=Fa;

end;

if abs(Exponent)>0 **then**

for i:=1 **to** Abs(Exponent) **do**

 Pow:=Pow*Basis;

if Exponent<0 **then** Pow:=1/Pow;

 Power:=Pow;

end;

Function Bernstein(n,i:Integer):Double;

begin

 Bernstein:=Combin(n,i)*Power(t,i) *Power(1-

t,n-i);

end;

Function Combin(n,i:Integer):LongInt;

begin

 Combin:=Trunc(Factorial(n)/Factorial(n-

i)/Factorial(i));

end;

Function

Power(Basis:Real;Exponent:Integer):Extended;

var Pow:Extended;

 i :Integer;

begin

 Pow:=1;

[Zdrojový kód](#)

[Spustit](#)

Racionální Bezierova křivka: Předchozí Bezierovy křivky jsou řídicími body určeny jednoznačně. Ke změně tvaru nutná změna řídicího polygonu

Zobecnění: každému řídicímu bodu P_i přiřadíme nezáporné reálné číslo m_i , které ovlivňuje tvar křivky. Ta má potom tvar:

$$Q(t) = \sum_{i=0}^n P_i R_i^n(t); \quad t \in \langle 0; 1 \rangle; \quad R_i^n(t) = \frac{B_i^n(t)}{\sum_{j=0}^n B_j^n(t) m_j};$$

$R_i^n(t)$ jsou tzv. racionální Bernsteinovy polynomy

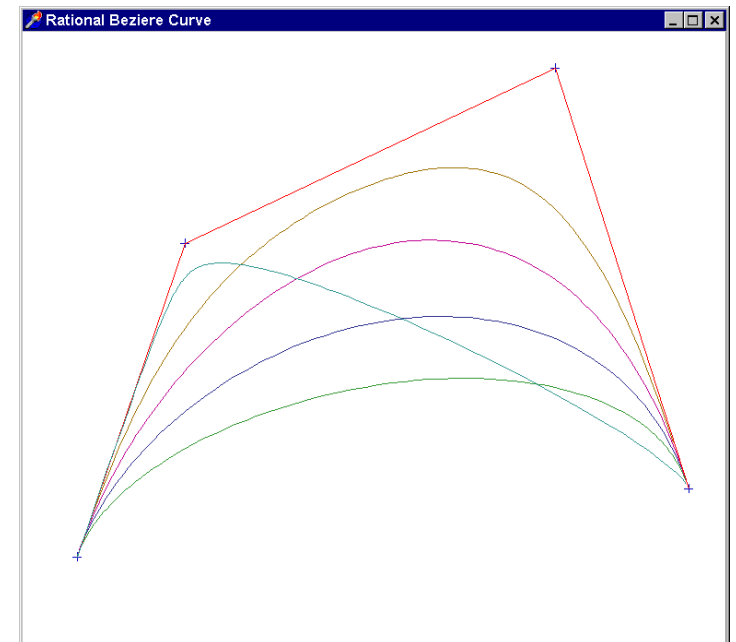
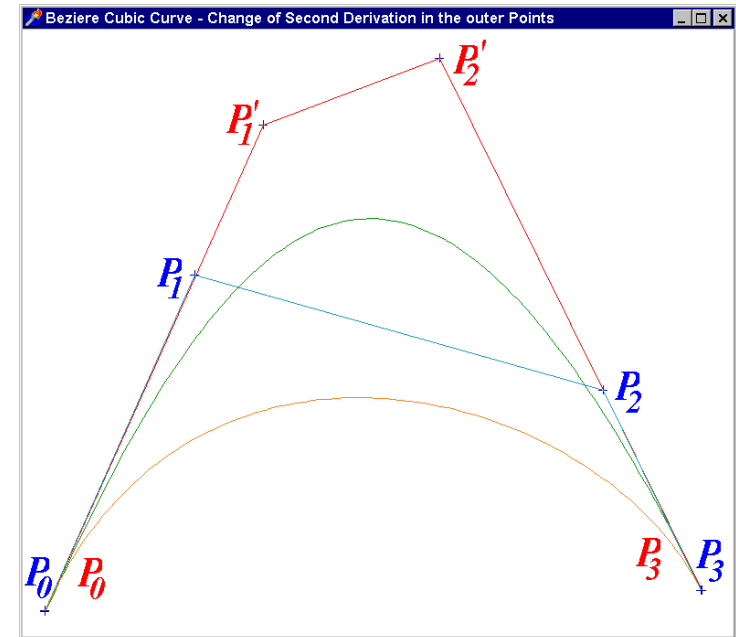
```
begin
  A[1]:=0;A[2]:=0;
  for i:=0 to n do
    begin
      R[i]:=Bernstein(n,i)*m[i];Value:=0;
      for j:=0 to n do Value:=Value+Bernstein(n,j)*m[j];
      A[1]:=A[1]+R[i]*P[i,1]/Value;
      A[2]:=A[2]+R[i]*P[i,2]/Value;
    end;
  end;
```

Změna tvaru Bezierovy kubiky $m_0 = m_3 = 1$, dále $m_1 = m_2$,
a to postupně

0,2; 0,4; 1; 4;

u poslední je pak $m_1 = 0.3$; $m_2 = 6$;

Bezierova racionální křivka [Zdrojový kód](#) [Spustit](#)



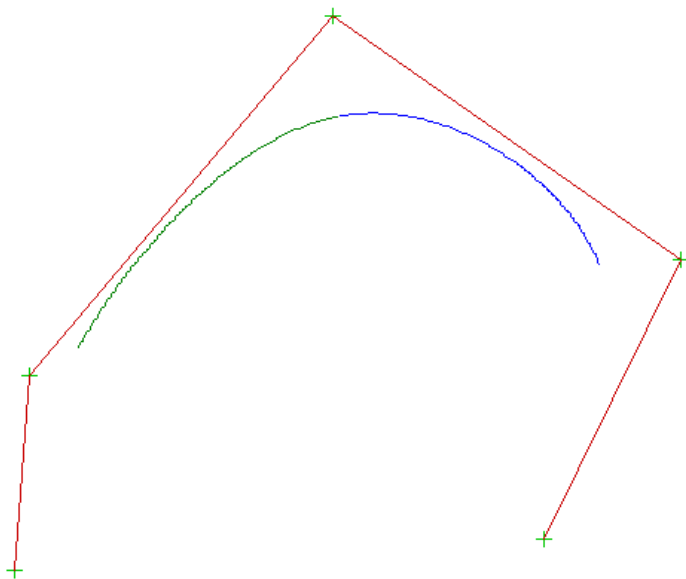
Coonsovy křivky a B-splajny Definoval S. A. Coons opět čtyřmi řídicími body P_0, P_1, P_2, P_3 a kubickými

polynomy C_0, C_1, C_2, C_3 :

$$Q(t) = \frac{1}{6} \sum_{i=0}^3 P_i C_i(t); \quad t \in \langle 0; 1 \rangle;$$

$$C_0(t) = (1-t)^3; \quad C_1(t) = 3t^3 - 6t^2 + 4; \quad C_2(t) = -3t^3 + 3t^2 + 3t + 1; \quad C_3(t) = t^3$$

Hladké spojení – opakování prvních tří bodů řídicího polygonu:



```

Procedure Coons(t:Real;var Q:TPoint);
begin
    Q[1]:=(P[0,1]*(1-t)*(1-t)*(1-t)+P[1,1]*(3*t*t*t-6*t*t+4)
            +P[2,1]*(-3*t*t*t+3*t*t+3*t+1)+P[3,1]*t*t*t)/6;
    Q[2]:=(P[0,2]*(1-t)*(1-t)*(1-t)+P[1,2]*(3*t*t*t-6*t*t+4)
            +P[2,2]*(-3*t*t*t+3*t*t+3*t+1)+P[3,2]*t*t*t)/6;
end;

Procedure CoonsArc(Red,Green,Blue:Byte);
begin
    t:=0;i:=1;
    While t<1 do
        begin Coons(t,C[i]);i:=succ(i);t:=t+ht;end;
        PolyLine(C,i-1,200,0,0);
    end;

```

begin

x1:=10;x2:=340;y1:=0;y2:=250;Scale(x1,x2,y1,y2);ht:=0.01;

P[1,1]:= 10; P[1,2]:= 10; P[2,1] = 30; P[2,2]:=150;

P[3,1]:= 90; P[3,2]:=250; P[4,1]:=140; P[4,2]:= 70;

CoonsArc(0;200,0);

P[1]:=P[2];P[2]:=P[3];P[3]:=P[4]; P[4,1] := 150;P[4,2] :=0;{posun řídicích bodů}

CoonsArc(0,200,0);

end;

{měřítka, krok parametru}

{řídicí body}

{první oblouk}

{druhý oblouk}

Coonsův oblouk:

[zdrojový kód](#) [spustit](#)

hladké spojení:

[zdrojový kód](#) [spustit](#)