RAČUNARSKA GRAFIKA

Oznaka predmeta: RG

Predavanje broj: 08

Nastavna jedinica: Krive, Hermitove krive, Bezier-ove krive, B-splajnovi, NURBS.

Nastavne teme: Predstavljanje krivih. Parametarske krive. Parametarske kubne krive. Hermitovi kubni splajnovi. Hermitove funkcije mešanja. Bezier-ove krive. Bezier vs. Hermite. Bezierove funkcije mešanja. Uniformni B-splajnovi. Neuniformni, racionalni B-splajnovi (NURBS).

Predavač: prof. dr Perica S. Štrbac, dipl. ing.

Literatura:

James D. Foley, Andries van Dam, Steven K. Feiner, John F. Hughes: "Computer Graphics: Principles and Practice", 2nd ed. in C, Addison-Wesley, 1996.

Predstavljanje krivih

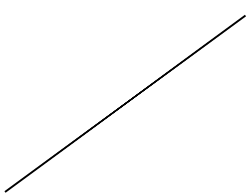
- Pomoću niza tačaka
  - Kriva je predstavljena približno, kao izlomljena linija – nije pogodno za glatke linije
  - Teško za manipulaciju jer se sve tačke moraju premeštati pojedinačno
- Umjesto toga, kriva se modeluje kao parametarski polinomom:

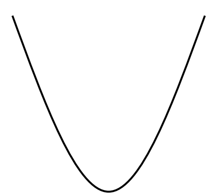
  x = x(t), y = y(t), z = z(t)
  - gde su x(), y(), z() polinomi, a t je parametar
- Polinomi:
  - Linearni
  - Kvadratni
  - Kubni

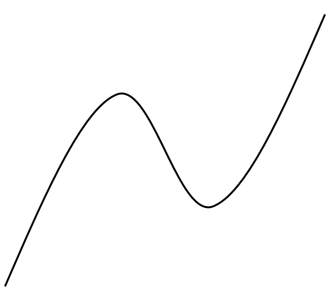
$$f(t) = at + b$$

$$f(t) = at^2 + bt + c$$

$$f(t) = at^3 + bt^2 + ct + d$$

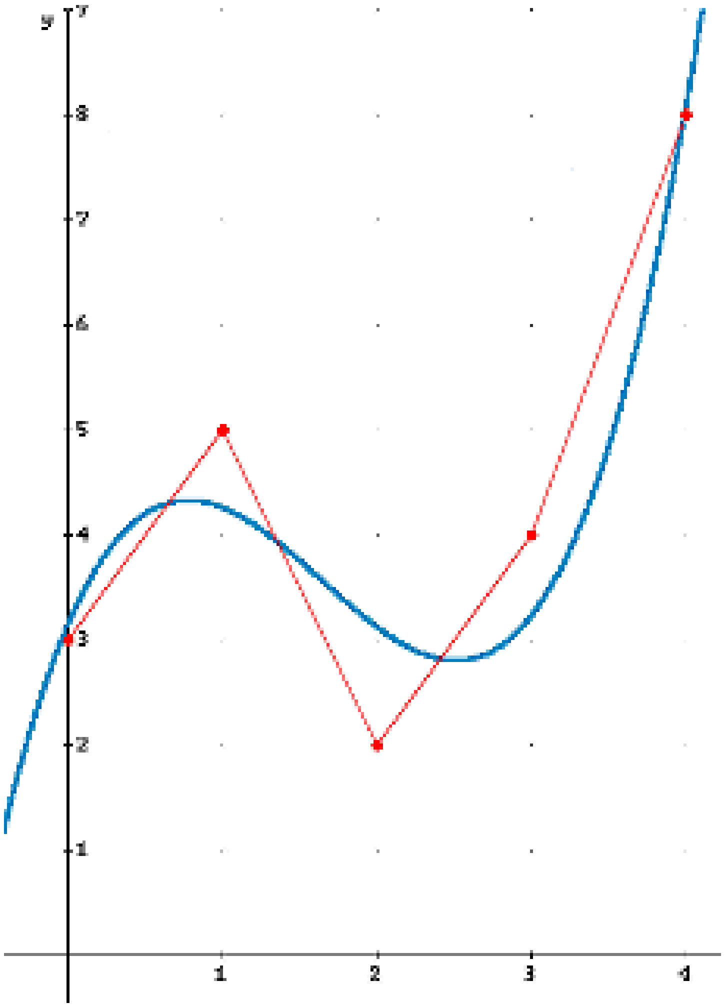






Predstavljanje krivih

- Kontrolne tačke
  - Set tačaka koje imaju uticaj na oblik krive.
- Čvorovi
  - Kontrolne tačke koje leže na krivoj.
- Splajnovi za interpolaciju (tačno poklapanje sa datim tačkama)
  - Splajn je specijalna funkcija definisana deo po deo preko polinoma.
  - Kontrolne tačke lokalno utiču na krivu.
  - Aproksimativni splajnovi (aproksimacija = približno poklapanje sa datim tačkama)



Parametarske krive

- Fleksibilno predstavljanje krive
- Ne moraju biti funkcije
  - Mogu imati više vrednosti u odnosu na bilo koju dimenziju

- Zadatak:

prikazati krivu zadatu parametarski,

ako su dati polinomi:

$$x(t) = t * \sin(t) + 150;$$

$$y(t) = t * \cos(t) + 150;$$

za t = 0, ... ,100

\- O kakvoj je krivoj reč ?

Kubni polinomi

- Kubni polinom je oblika

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$
$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$
$$z(t) = a_z t^3 + b_z t^2 + c_z t + d_z$$

- Neka je parametar t: (0 <= t <= 1), ako se uvede oznaka $T = [t^3\ t^2\ t\ 1]$
  - matrica koeficijenata C

$$\begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}$$

  - kriva: Q(t) = T*C

$$\begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} * \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}$$

Parametarske krive

- Kako odrediti tangentu na krivu u datoj tački?
- Neka je data kriva:

$$f(x) = x^2 - 4$$

  - tangenta za (x=3) je

$$f'(x) = 2x = 2(3) = 6$$

$$\frac{d}{dt}Q(t) = Q'(t) = \frac{d}{dt}TC = \begin{bmatrix} 3t^2 & 2t & 1 & 0 \end{bmatrix} C$$

➢ Izvod (derivacija) od Q(t) po t je vektor tangente u tački krive, gde je ta tačka određena vrednošću parametra t.

Izvodi parametarske funkcije

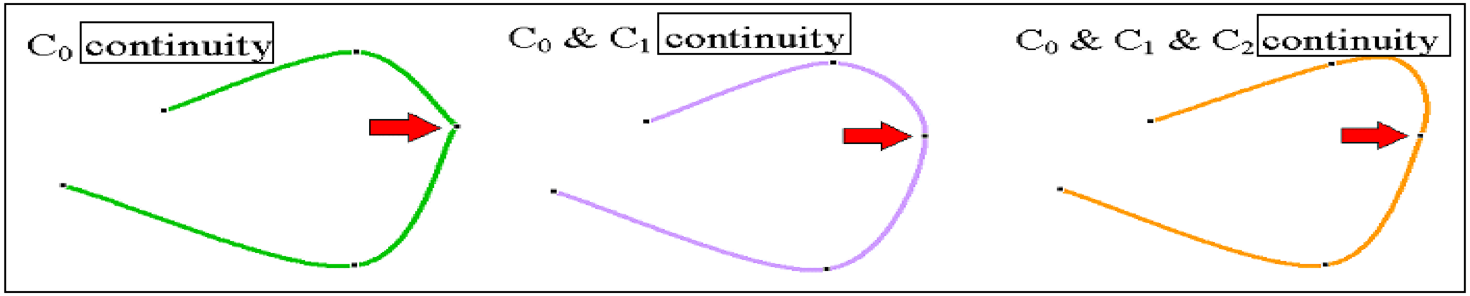
- Određivanje izvoda (tangenti) krive:

$$\mathbf{x} = \mathbf{a}t^3 + \mathbf{b}t^2 + \mathbf{c}t + \mathbf{d} \qquad \frac{d\mathbf{x}}{dt} = 3\mathbf{a}t^2 + 2\mathbf{b}t + \mathbf{c}$$

$$\mathbf{x} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} \qquad \frac{d\mathbf{x}}{dt} = \begin{bmatrix} 3t^2 & 2t & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix}$$

Segmenti krive

- Krive se mogu konstruisati povezivanjem krajeva više manjih segmenata
  - Moraju postojati pravila o tome kako se vrši povezivanje
- Kontinuitet opisuje vezu segmenata
  - parametarski kontinuitet
    - C_0 je kontinuitet linije (zajednička tačka)
    - C_1 je tangentni kontinuitet (brzina)
    - C_2 je kontinuitet drugog izvoda (ubrzanje)



  - geometrijski kontinuitet,
    - ako se položaji tačaka poklapaju, G^0 geometrijski kontinuitet
    - ako se smer (ne obavezno i intenzitet) tangenti poklapa, G^1 geometrijski kontinuitet
      - vrednost tangente na kraju jedne krive je proporcionalna vrednosti tangente na početku sedeće krive

Parametarske kubne krive

- Da bi se osigurao C_2 kontinuitet, krive moraju biti najmanje trećeg reda
- Data je parametarska definicija kubnog splajna (3. reda) u dve dimenzije

$$x = a_x t^3 + b_x t^2 + c_x t + d_x$$

$$y = a_y t^3 + b_y t^2 + c_y t + d_y$$

- Može se predstaviti i u matričnom obliku

$$\begin{bmatrix} x & y \end{bmatrix} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix}$$

Koeficijenti

- Kako izabrati koeficijente?
  - Vektori koeficijenata [a_x b_x c_x d_x] i [a_y b_y c_y d_y] moraju zadovoljiti ograničenja koja nameću čvorovi i uslovi kontinuiteta

- Krivu je teško konceptualizirati kao:

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$

umesto toga, kriva se definiše kao kombinacija kubnih polinoma

- Svaki tip krive definiše različite kubne polinome:
  - Hermitove
    - dve krajnje tačke i dva vektora tangenti u krajevima
  - Bezierove
    - dve krajnje tačke i dve druge tačke koje definišu vektore tangenti u krajevima
  - Splajnovi
    - četiri kontrolne tačke

      C0 i C1 kontinuitet u tačkama dodira (zajednička tačka, brzina), tačke krive se približavaju svojim kontrolnim tačkama, ali ih ne moraju uvek dodirnuti

Hermitovi kubni splajnovi

- Primer čvorova i kontinuiteta

- Po jedna kubna kriva za svaku dimenziju
  - kriva u x-y ravni određena je sa dve krive

$$f_x(t) = at^3 + bt^2 + ct + d = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

$$f_y(t) = et^3 + ft^2 + gt + h = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix}$$

Hermitovi kubni splajnovi

- 2-D Hermitov kubni splajn je definisan sa 8 parametara:
  - a, b, c, d,   e, f, g, h
- Kako na osnovu intuitivnih krajnjih tačaka i parametarskih izvoda u tim tačkama dobiti ovih 8 (relativno) neintuitivnih parametara?
- Poznato je, daju sa kao ulazni podaci za zadatak:
  - (x, y) položaj za t = 0, p_1
  - (x, y) položaj za t = 1, p_2
  - (x, y) izvod za t = 0   , dp_1/dt
  - (x, y) izvod za t = 1   , dp_2/dt

Hermitovi kubni splajnovi

- Poznat je: (x, y) položaj za t = 0, p_1

$$f_x(0) = a0^3 + b0^2 + c0 + d = \begin{bmatrix} 0^3 & 0^2 & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

$$f_x(0) = d = p_{1x}$$

$$f_y(0) = e0^3 + f0^2 + g0 + h = \begin{bmatrix} 0^3 & 0^2 & 0 & 1 \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix}$$

$$f_y(0) = h = p_{1y}$$

- Poznat je: (x, y) položaj za t = 1, p_2

$$f_x(1) = a1^3 + b1^2 + c1 + d = \begin{bmatrix} 1^3 & 1^2 & 1 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

$$f_x(1) = a + b + c + d = p_{2x}$$

$$f_y(1) = e1^3 + f1^2 + g1 + h = \begin{bmatrix} 1^3 & 1^2 & 1 & 1 \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix}$$

$$f_y(1) = e + f + g + h = p_{2y}$$

Hermitovi kubni splajnovi

- Za sada su 4 jednačine, a 8 nepoznatih
- Koriste se izvodi (koji su takođe poznati kao ulazni podaci) u tačkama p_1 i p_2:

$$f_x(t) = at^3 + bt^2 + ct + d$$

$$f_x'(t) = 3at^2 + 2bt + c$$

$$f_x'(t) = \begin{bmatrix} 3t^2 & 2t & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

$$f_y(t) = et^3 + ft^2 + gt + h$$

$$f_y'(t) = 3et^2 + 2ft + g$$

$$f_y'(t) = \begin{bmatrix} 3t^2 & 2t & 1 & 0 \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix}$$

Hermitovi kubni splajnovi

- Poznat je:
  - (x, y) izvod za t = 0, dp_1/dt

$$f_x'(0) = 3a0^2 + 2b0 + c$$

$$= \begin{bmatrix} 3\cdot 0^2 & 2\cdot 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

$$f_x'(0) = c = {dp_{1x}}/{dt}$$

$$f_y'(0) = 3e0^2 + 2f0 + g$$

$$= \begin{bmatrix} 3\cdot 0^2 & 2\cdot 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix}$$

$$f_y'(0) = g = {dp_{1y}}/{dt}$$

Hermitovi kubni splajnovi

- Poznat je i:
  - (x, y) izvod za t = 1, dp_2/dt

$$f_x'(1) = 3a1^2 + 2b1 + c$$

$$= \begin{bmatrix} 3\cdot 1^2 & 2\cdot 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

$$f_x'(1) = 3a + 2b + c = {dp_{2x}}/{dt}$$

$$f_y'(1) = 3e1^2 + 2f1 + g$$

$$= \begin{bmatrix} 3\cdot 1^2 & 2\cdot 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix}$$

$$f_y'(1) = 3e + 2f + g = {dp_{2y}}/{dt}$$

Hermitova specifikacija

- Uvrštavanjem dobijenih jednačina prema datim početnim vrednostima:
  - tačaka p_1, p_2
  - poznatim vrednostima prvih izvoda u ovim tačkama,

  dolazi se do matrične jednačine za Hermitovu krivu koja izgleda kao što sledi (radi lakšeg razumevanja dodatne informacije su označene crvenom bojom):

$$
\begin{array}{c}
 \\ p_1 \\ p_2 \\ p_1' \\ p_2'
\end{array}
\begin{array}{c}
\begin{array}{cccc} t^3 & t^2 & t^1 & t^0 \end{array} \\
\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}
\end{array}
\begin{bmatrix} a & e \\ b & f \\ c & g \\ d & h \end{bmatrix}
=
\begin{bmatrix} p_{1x} & p_{1y} \\ p_{2x} & p_{2y} \\ dp_{1x}/dt & dp_{1y}/dt \\ dp_{2x}/dt & dp_{2y}/dt \end{bmatrix}
\begin{array}{c}
t = 0 \\ t = 1 \\ t = 0 \\ t = 1
\end{array}
$$

Rešavanje Hermitove matrice

- Rešavanje inverzne matrice A^{-1}:
  1. izračunati determinantu det(A)
  2. ako je det(A)!=0
     onda naći transponovanu matricu A^T
  3. naći adjungovanu matricu A^*
  4. sada je $A^{-1}=(1/\det(A))\cdot A^*$

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} p_{1x} & p_{1y} \\ p_{2x} & p_{2y} \\ dp_{1x}/dt & dp_{1y}/dt \\ dp_{2x}/dt & dp_{2y}/dt \end{bmatrix} = \begin{bmatrix} a & e \\ b & f \\ c & g \\ d & h \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \qquad \det A = \begin{vmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{vmatrix} = (-1)\begin{vmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 3 & 2 & 1 \end{vmatrix} = -1$$

$$A^T = \begin{bmatrix} 0 & 1 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} \qquad A^* = \begin{bmatrix} -2 & 2 & -1 & -1 \\ 3 & -3 & 2 & 1 \\ 0 & 0 & -1 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix} \qquad A^{-1} = \frac{1}{\det A}\cdot A^* = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

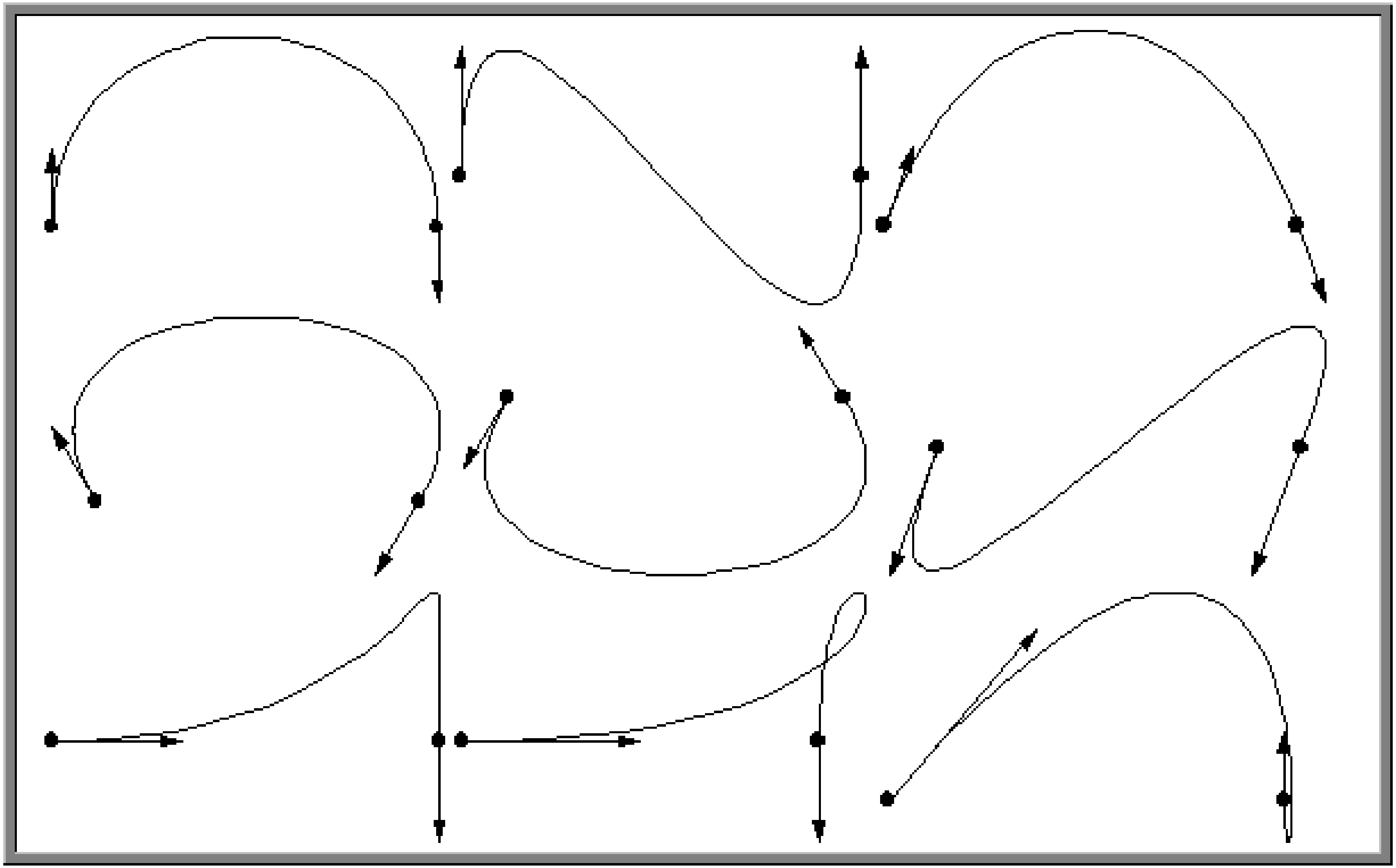
Matrice splajna i geometrije

$$\underbrace{\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathrm{M_{Hermite}}} \underbrace{\begin{bmatrix} p_{1x} & p_{1y} \\ p_{2x} & p_{2y} \\ dp_{1x}/dt & dp_{1y}/dt \\ dp_{2x}/dt & dp_{2y}/dt \end{bmatrix}}_{\mathrm{G_{Hermite}}} = \begin{bmatrix} a & e \\ b & f \\ c & g \\ d & h \end{bmatrix}$$

- **Rezultujuća jednačina Hermitovog splajna:**

$$[x \quad y] = [t^3 \quad t^2 \quad t \quad 1] \underbrace{\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{M}_{Hermite}} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \frac{dx_1}{dt} & \frac{dy_1}{dt} \\ \frac{dx_2}{dt} & \frac{dy_2}{dt} \end{bmatrix}}_{\mathbf{G}_{Hermite}}$$

Primeri Hermitovih krivih



Funkcije mešanja (Blending Functions)

- Množenjem prve dve matrice u jednačini hermitovog splajna, dobiju se 4 funkcije od 't' koje mešaju 4 kontrolna parametra

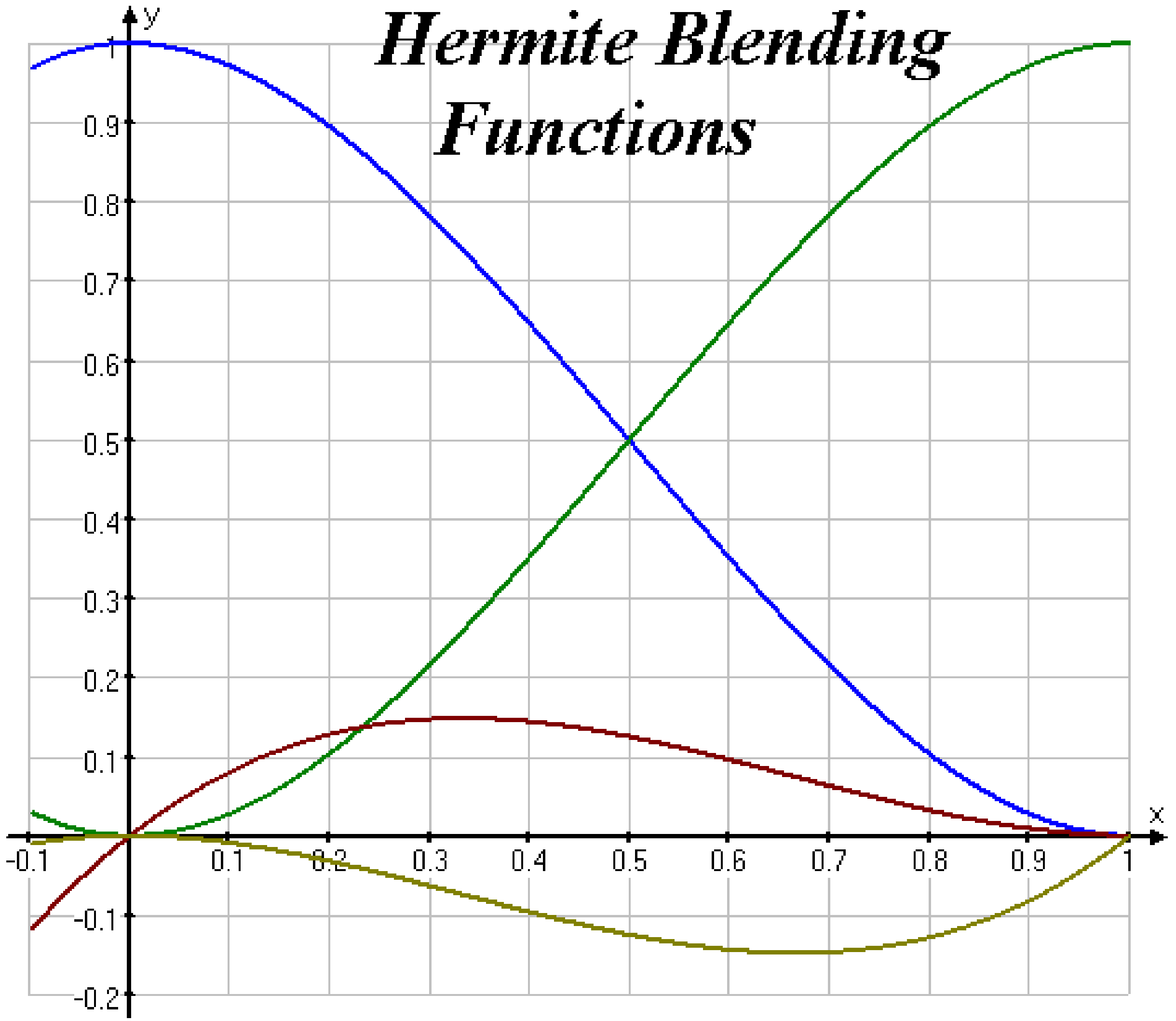
$$[x \quad y] = [t^3 \quad t^2 \quad t \quad 1] \underbrace{\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{M}_{Hermite}} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \frac{dx_1}{dt} & \frac{dy_1}{dt} \\ \frac{dx_2}{dt} & \frac{dy_2}{dt} \end{bmatrix}}_{\mathbf{G}_{Hermite}}$$

- To su funkcije mešanja:

$$p(t) = \begin{bmatrix} 2t^3 - 3t^2 + 1 \\ -2t^3 + 3t^2 \\ t^3 - 2t^2 + t \\ t^3 - t^2 \end{bmatrix}^{\mathbf{T}} \begin{bmatrix} p_1 \\ p_2 \\ \nabla p_1 \\ \nabla p_2 \end{bmatrix}$$

Hermitove funkcije mešanja

- Grafička zavisnost funkcije mešanja od parametra 't'
- Svaka funkcija mešanja reflektuje uticaj P_1, P_2, ΔP_1, ΔP_2 na oblik krive



Zadatak

- Date su tačke:

  P1(100, 20)

  P2(200, 150)

  - prvi izvod u tački p1 iznosi T1(-210, -80)
  - prvi izvod u tački p2 iznosi T2(180,-180)

koristeći sledeći pseudo kod nacrtati Hermitovu krivu koja spaja tačke P1 i P2.

```
moveto (P1); //postavljanje startne tacke
for (int i=0; i<= steps; i++)
{
  float t = (float)i / (float)steps;       //normalizacija 0<=t<=1
  float h1 =  2*pow(t,3) – 3*pow(t,2) + 1; //f-ja mesanja h1
  float h2 = -2*pow(t,3) + 3*pow(t,2);     //f-ja mesanja h2
  float h3 =    pow(t,3) - 2*pow(t,2) + t; //f-ja mesanja h3
  float h4 =    pow(t,3) -   pow(t,2);     //f-ja mesanja h4

  point p = h1*P1 + // racunanje interpolacione tacke duz krive
            h2*P2 +
            h3*T1 +
            h4*T2;
  lineto (p); // povuci liniju do interpolacione tacke
}
```

Primer 1/3

```
#pragma comment( lib, "opengl32.lib")
#pragma comment( lib, "glu32.lib")
#pragma comment( lib, "glut32.lib")
#pragma comment( linker, "/subsystem:windows /entry:mainCRTStartup")
#include <cmath>
#include <GL/glut.h>
class Point{
  public:
    double x;
    double y;
    Point(double x=0, double y=0){ this->x = x;this->y = y; }
};

class Line{
  public:
    Point a;
    Point b;
    Line(Point a=Point(0,0), Point b=Point(0,0)){this->a = a;this->b=b;}
};

float yellow[3] = {1.0,1.0,0.0};
float red[3]    = {1.0,0.0,0.0};
float blue[3]   = {0.0,0.0,1.0};
```

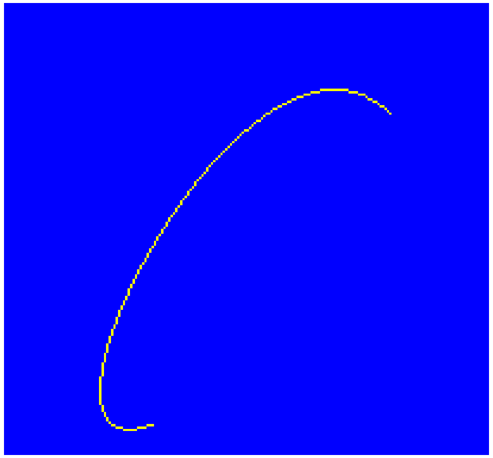
Primer 2/3

```
void ShowLine(double x1, double y1, double x2, double y2){
  glBegin(GL_LINES);glVertex2f(x1,y1);glVertex2f(x2,y2);glEnd();
}
void Hermit(Point P1, Point P2, Point T1, Point T2){//T1,T2 izvodi u P1,P2
  int steps=50;
  Point from(P1); //postavljanje startne tacke
  for (int i=0; i<= steps; i++) {
    double t = (double)i / (double)steps;       //normalizacija 0<=s<=1
    double h1 =  2.0*pow(t,3) - 3.0*pow(t,2)+1; //f-ja mesanja h1
    double h2 = -2.0*pow(t,3) + 3.0*pow(t,2);     //f-ja mesanja h2
    double h3 =      pow(t,3) - 2.0*pow(t,2) + t; //f-ja mesanja h3
    double h4 =      pow(t,3) -     pow(t,2);     //f-ja mesanja h4
    Point p;
    p.x = h1*P1.x + // racunanje interpolacione tacke duz krive
          h2*P2.x +
          h3*T1.x +
          h4*T2.x;
    p.y = h1*P1.y + // racunanje interpolacione tacke duz krive
          h2*P2.y +
          h3*T1.y +
          h4*T2.y;
   ShowLine(from.x,from.y,p.x,p.y); // povuci liniju do Interpol. tacke
   from=p;
}}
```

Primer 3/3

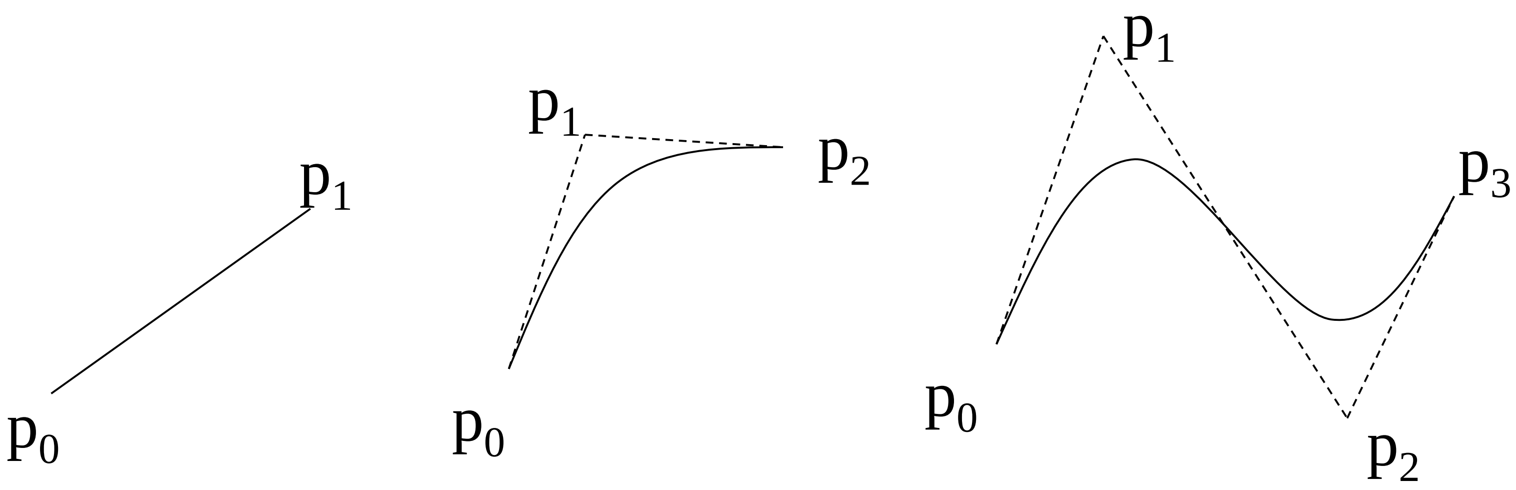
```
void displayCB(void){
  glClear(GL_COLOR_BUFFER_BIT);
  glColor3fv(yellow);
  Point A(100.0,20.0);
  Point B(200.0,150.0);
  Point T1(-210.0,-80.0);
  Point T2(180.0,-180.0);
  Hermit(A,B,T1,T2);
  glFlush();
}
void keyCB(unsigned char key, int x, int y){  if( key=='q' ) exit(0); }

int main(int argc, char *argv[]){
  glutInit(&argc, argv);
  glutInitDisplayMode(GLUT_RGB);
  glutInitWindowSize(500,500);
  int win = glutCreateWindow("Hermit");
  glClearColor(0.0,0.0,1.0,0.0);
  gluOrtho2D(0,500,0,500);
  glutDisplayFunc(displayCB);
  glutKeyboardFunc(keyCB);
  glutMainLoop();
  return 0;
}
```



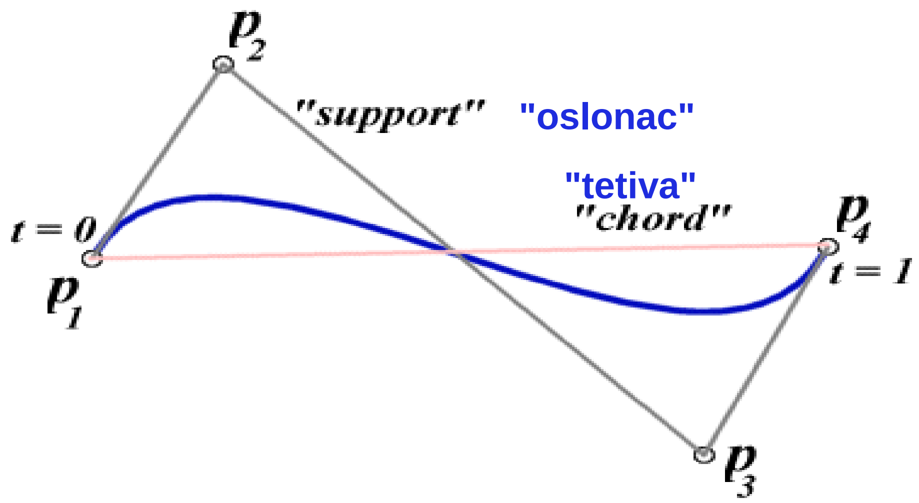
Bezier-ove krive

- Kontinuirane krive koje prolaze kroz zadate krajnje kontrolne tačke.
- Kontinuitet drugog reda (zajedničke tačke, tangente (prvi) i drugi izvodi)
- Bezier-ove krive se mogu posmatrati kao proširenje linearne interpolacije interpolacijom višeg reda



Bezier-ove krive

- Slične Hermitovim, ali imaju intuitivniju definiciju izvoda krajnjih tačaka
- Četiri kontrolne tačke, od kojih su dve čvorovi



- Vrednosti izvoda Bezier-ove krive u čvorovima zavise od susednih tačaka
- Neka je skalar 3 izabran samo za ovu krivu kao što sledi:

$$\nabla p_1 = 3(p_2 - p_1)$$
$$\nabla p_2 = 3(p_4 - p_3)$$

Bezier vs. Hermite

- Bezier se može napisati u zavisnosti od Hermitove krive
  - Ovo je samo matrična forma prethodnih jednačina

$$\underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \frac{dx_1}{dt} & \frac{dy_1}{dt} \\ \frac{dx_2}{dt} & \frac{dy_2}{dt} \end{bmatrix}}_{\mathbf{G}_{Hermite}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix}}_{\mathbf{G}_{Bezier}}$$

$$\begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix} = \underbrace{\underbrace{\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{M}_{Hermite}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix}}_{M_{bezier}} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix}}_{\mathbf{G}_{Bezier}}$$

Osnovna matrica Bezier-ove krive

- Matrična forma Bezierove krive

$$\begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix} = \underbrace{\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{M}_{Bezier}} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix}}_{\mathbf{G}_{Bezier}}$$

- Zadatak: implementirati program koji crta Bezier-ovu krivu određenu tačkama:

p1(10,10) p2(50,100) p3(250,100) p4(300,300)

Bezier-ova funkcija mešanja

- Funkcija mešanja definiše uticaj kontrolne tačke na svaku tačku krive.
- Vrednost 0 znači da kontrolna tačka nema uticaja na tačku krive.
- Ako funkcija mešanja ima vrednost 1, kriva seče kontrolnu tačku.
- Ova familija polinoma se naziva Bernstein polinomima 3. reda
  - $C(3, k)\ t^k\ (1-t)^{3-k} \qquad ; 0 \le k \le 3$
  - Svi su pozitivni na intervalu [0,1]
  - Njihov zbir iznosi 1

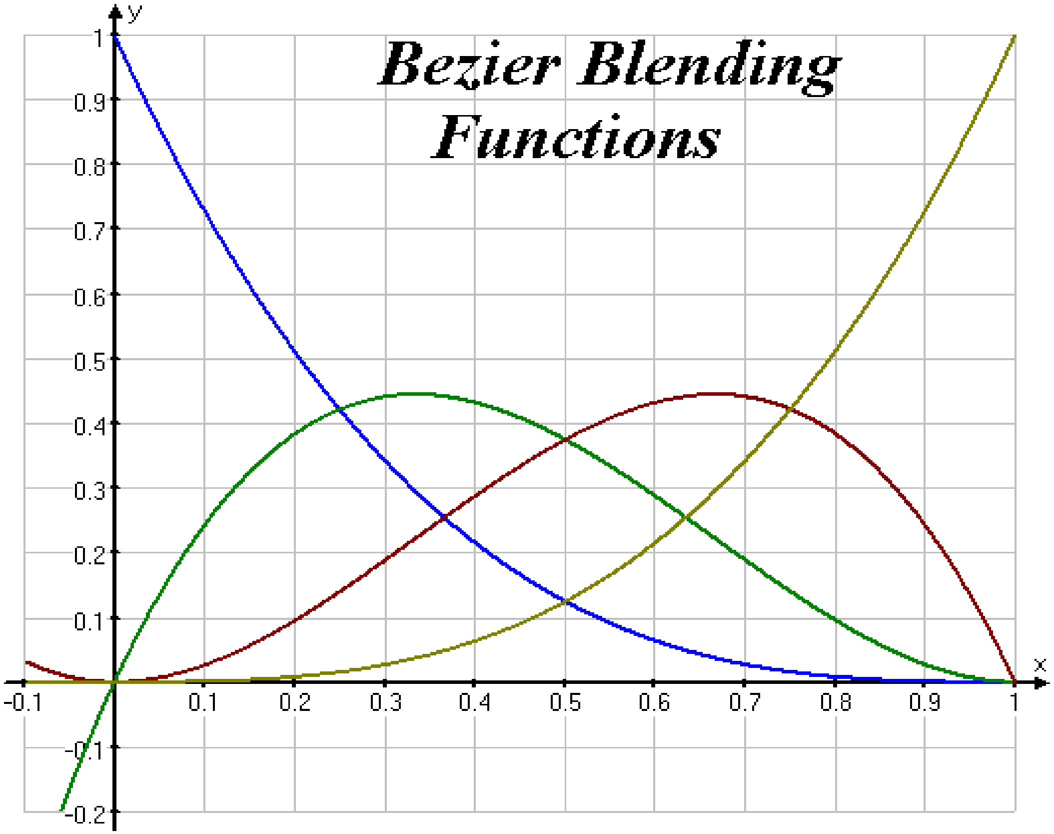
$$p(t) = \begin{bmatrix} (1-t)^3 \\ 3t(1-t)^2 \\ 3t^2(1-t) \\ t^3 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

$$B_{i,n}(u) = \binom{n}{i} u^i (1-u)^{n-i}$$

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i}$$

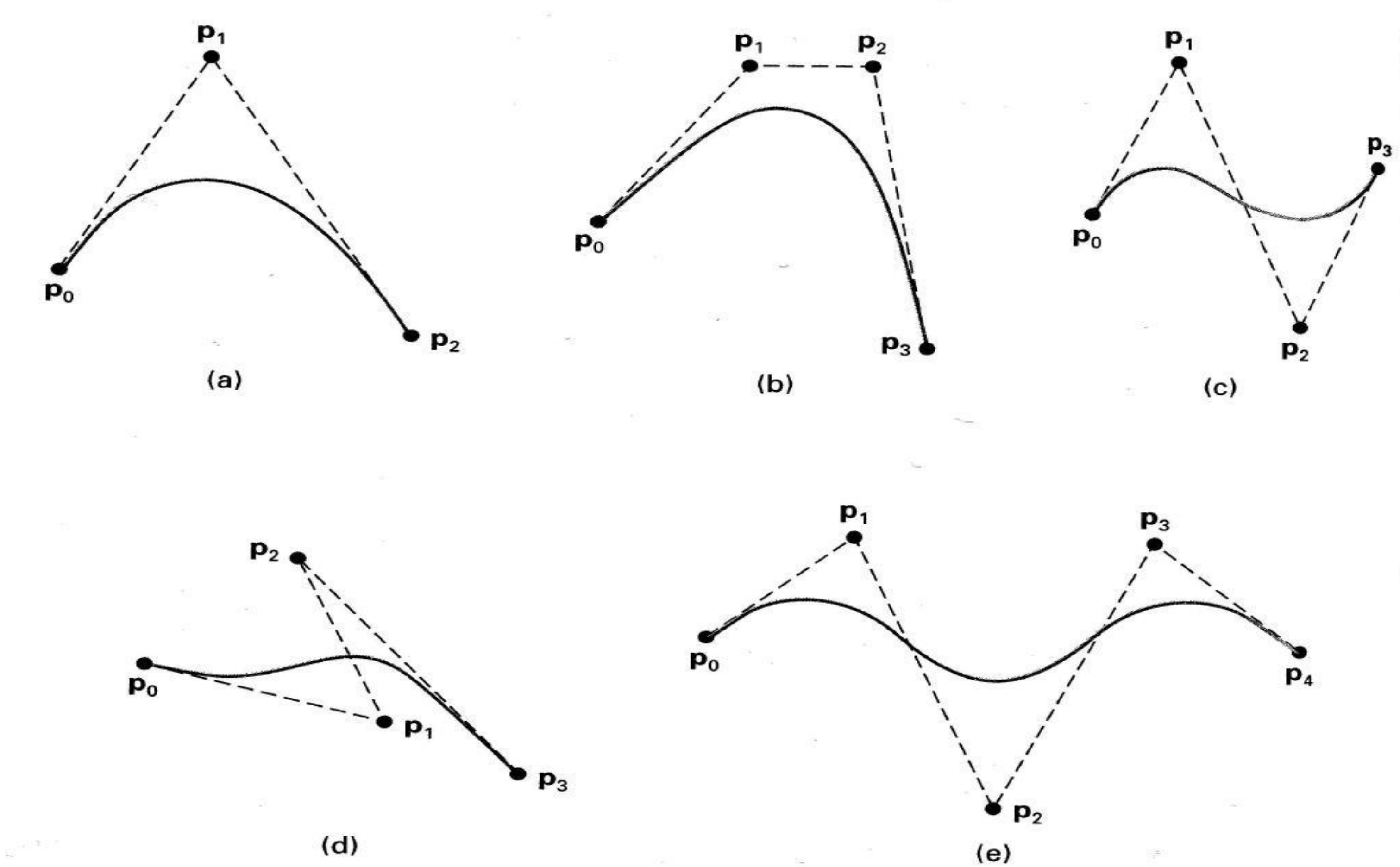
Bezier-ove funkcije mešanja

- Svaka tačka na krivoj je linearna kombinacija kontrolnih tačaka.
- Sve vrednosti kombinacija su pozitivne.
- Zbir vrednosti je 1.
- Kriva je konveksna kombinacija kontrolnih tačaka.

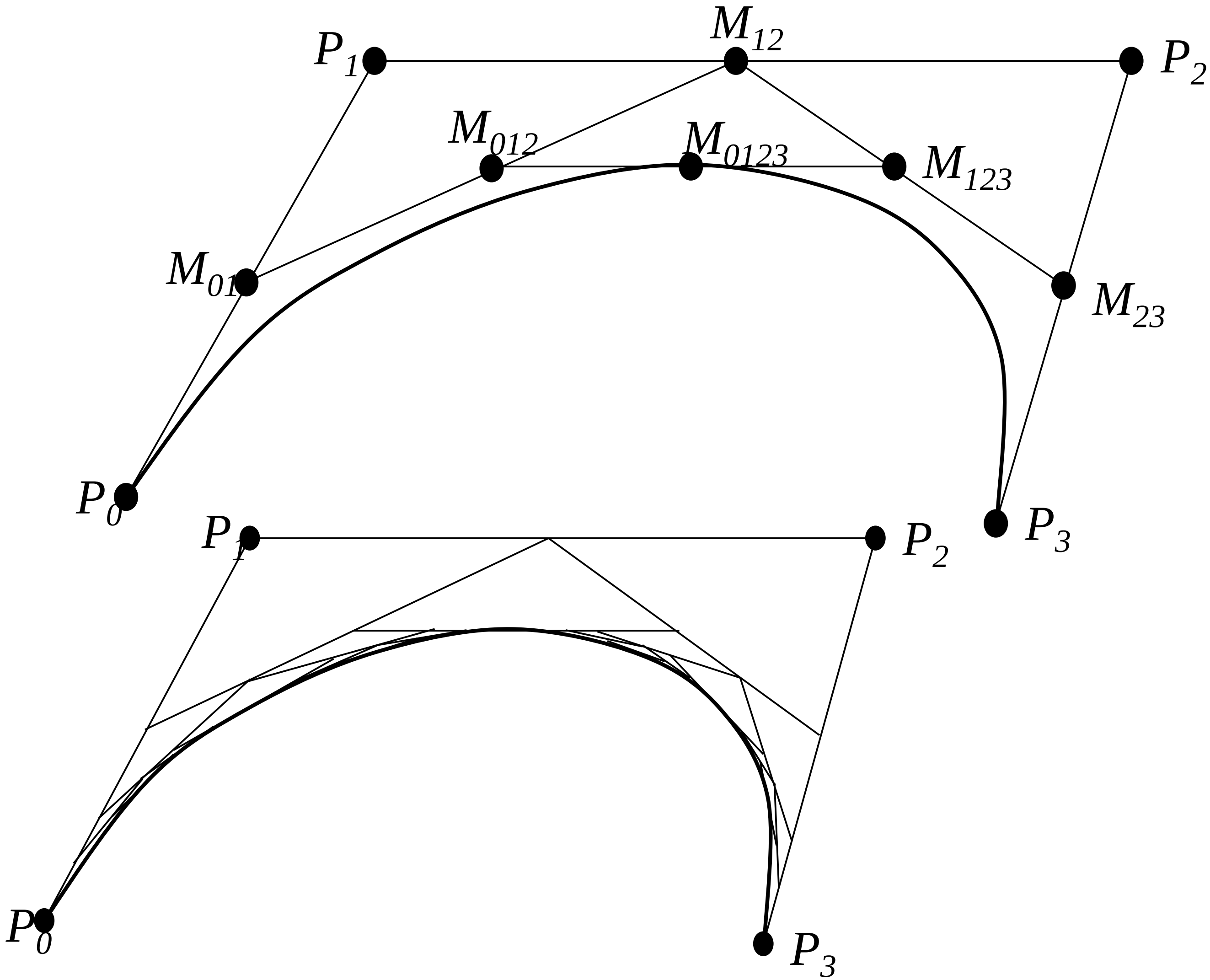


Konveksna kombinacija kontrolnih tačaka

- Uvek ostaje unutar graničnog regiona (konveksna ljuska) definisanog kontrolnim tačkama

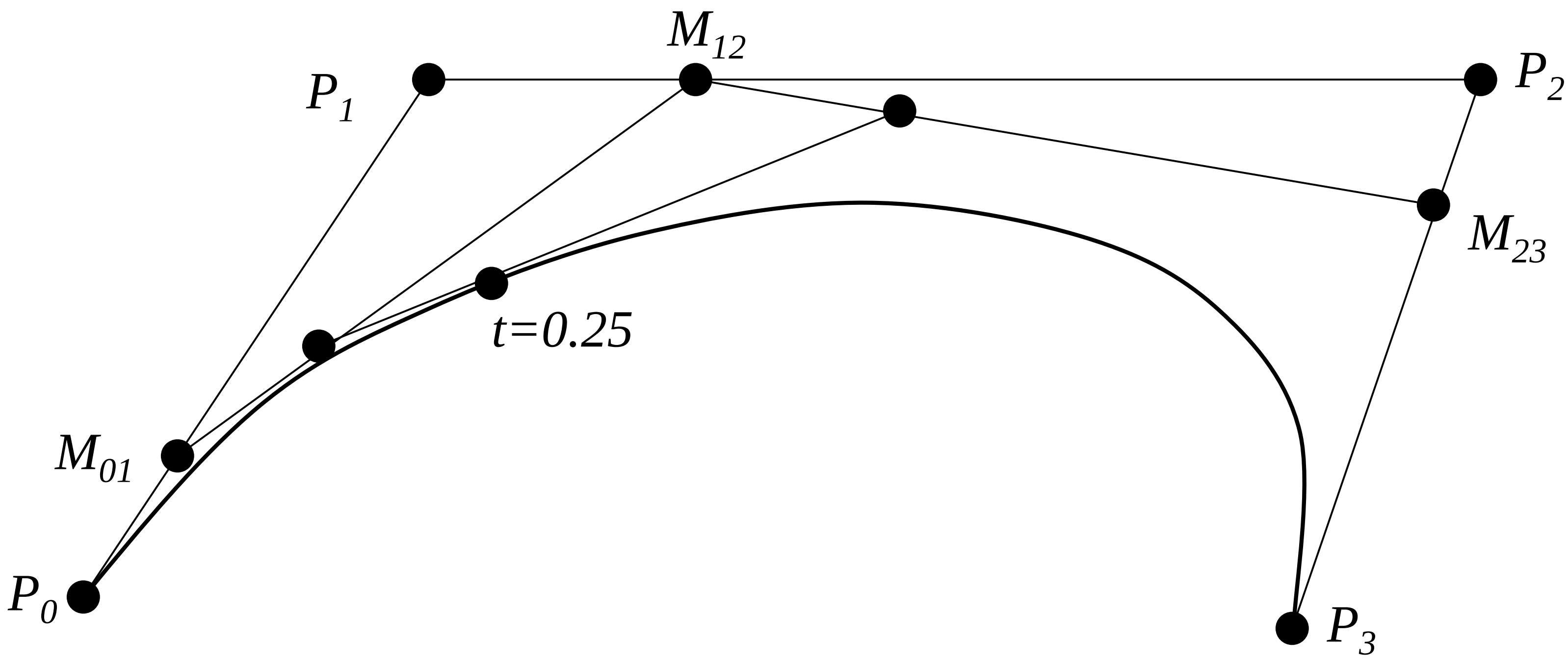


De Casteljau-ov algoritam

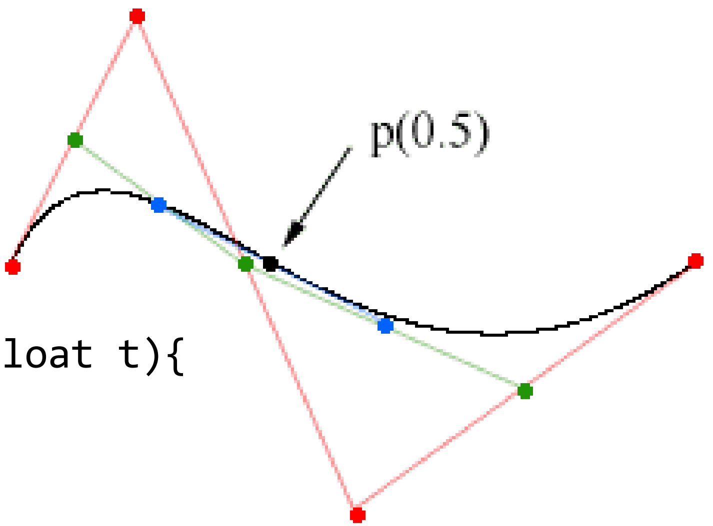


De Casteljau-ov algoritam

- Može se naći tačka na Bezier-ovoj krivoj za svaku vrednost parametra *t* prethodno pokazanim algoritmom.
- Ako se želi uzeti vrednost parametra *t=0.25*, umesto da se uzmu srednje tačke, uzimaju se tačke na 0.25 dužine kao na slici:



Primer




```
#include <stdio.h>

struct point { float x; float y; };
point a = {  40, 100 };
point b = {  80,  20 };
point c = { 150, 180 };
point d = { 260, 100 };


void mt(point &dest, point a, point b, float t){
  dest.x = a.x + (b.x-a.x)*t;
  dest.y = a.y + (b.y-a.y)*t;
}

void bezier (point &dest, float t){
  point ab,bc,cd,abbc,bccd;
  mt(ab, a,b,t);   mt(bc, b,c,t);   mt(cd, c,d,t);      //3 zelene tacke
  mt(abbc, ab,bc,t);  mt(bccd, bc,cd,t);                //2 plave tacke
  mt(dest, abbc,bccd,t); //tacka na Bezierovoj liniji – crna tacka
}
void main (void) {
  point p;
  for ( int i=0; i<=1000; i++ )
  {
    float t = (float)i/1000.0;
    bezier (p,t);
    printf ("%f %f\n", p.x, p.y);
    //glBegin(GL_POINTS); glVertex2f(p.x, p.y); glEnd();
  }
}
```

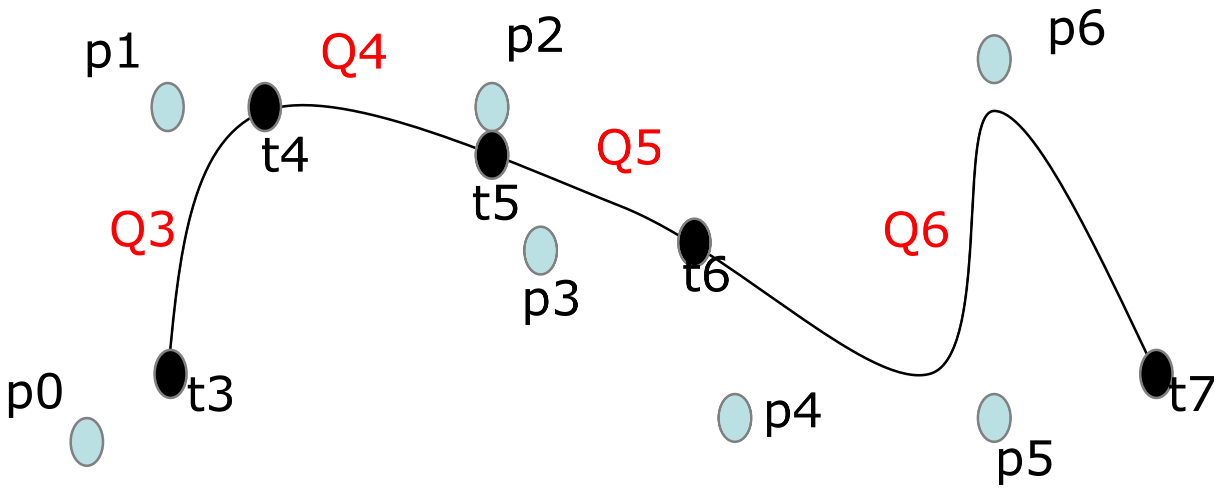
Zašto je potrebno još vrsta splajnova?

- Bezier-ovi i Hermitovi splajnovi imaju globalni uticaj
  - Može se kreirati Bezier-ova kriva kojoj bi trebalo 15 tačaka za definisanje krive
  - Pomeranjem jedne kontrolne tačke utiče se na celu krivu
  - Kod diskontinuiranih Bezier-ovih ili Hermitovih krivih se to ne dešava
    - ali zato kod njih nema obaveznog kontinuiteta izvoda u tačkama dodira
- B-splajnovi se sastoje od zakrivljenih segmenata čiji koeficijenti polinoma zavise samo od nekoliko kontrolnih tačaka
  - Kontrolna tačka ne utiče na celu krivu
  - Lokalna kontrola krive je izvedena preko grupe kontrolnih tačaka
  - Potrebno je podeliti crtanje po segmentima na koji utiču grupe kontrolnih tačaka, a ne sve kontrolne tačke.

B-Splajn krive (kubne periodične)

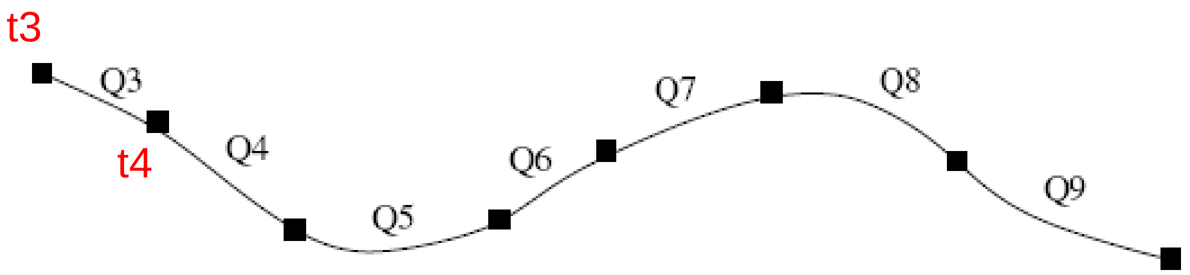
- Polazi se od niza kontrolnih tačaka.
- Izaberu se 4 tačke koje će biti kontrolne za odgovarajući segment

  $(p_{i-3}, p_{i-2}, p_{i-1}, p_i)$
  - Sada grupa tačaka utiče na formiranje odgovarajućeg segmenta
  - B-Splajn ne interpolira (ne dodiruje) niti jednu od njih ali aproksimira prolazak pored tih tačaka
- Na slici su dati:
  - kontrolne tačke: p_0, p_1, p_2, p_3, p_4, p_5, p_6.
  - segmenti: Q_3, Q_4, Q_5, Q_6.
  - parametri t koji predstavljaju spoj ili kraj datog segmenta: t_3, t_4, t_5, t_6, t_7.



B-Splajn krive (kubne periodične)

- Knot je krajnja tačka segmenta (knotovi na slici: t3, t4)



- Neka na segment utiču 4 kontrolne tačke, tada je kontrola tačaka na dati segment kao što sledi:

segment	kontrolne tačke
Q_3	p_0, p_1, p_2, p_3
Q_4	p_1, p_2, p_3, p_4
Q_5	p_2, p_3, p_4, p_5
Q_6	p_3, p_4, p_5, p_6
Q_7	p_4, p_5, p_6, p_7
Q_8	p_5, p_6, p_7, p_8
Q_9	p_6, p_7, p_8, p_9

B-spline osnovne funkcije

- Knot-ovi mogu biti jednako udaljeni po krivoj jedan od drugog gledano prema vrednosti parametra t
    - uniformni B-splajn

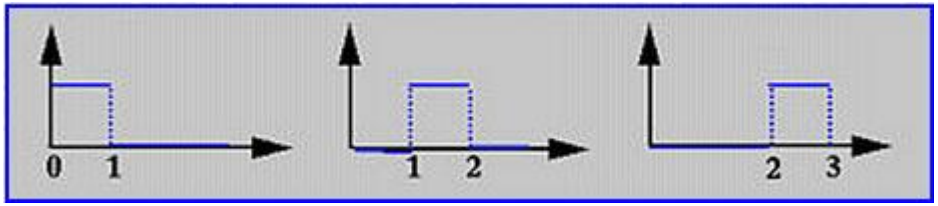
  ili suprotno mogu biti neuniformni.

- Cox de Boor algoritam rekurzivno izračunava osnovne funkcije za bilo koji tip B-splajna stepena m.
  - Vrednosti knotova su predstavljene sa $\boldsymbol{t_i}$ gde je $\boldsymbol{i}$ odgovarajći indeks knota

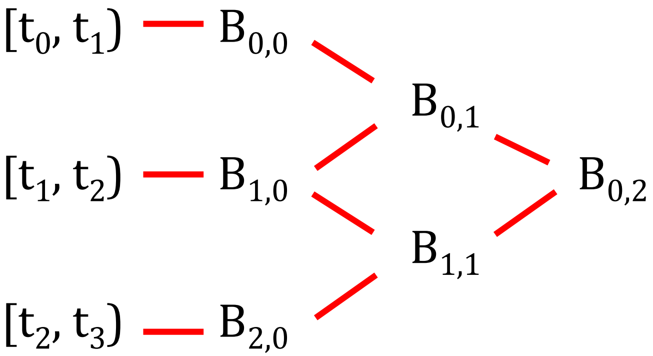
$$B_{i,0}(t) = \begin{Bmatrix} 1 & \quad \text{if } t_i \le t < t_{i+1} \\ 0 & \quad \text{otherwise} \end{Bmatrix}$$

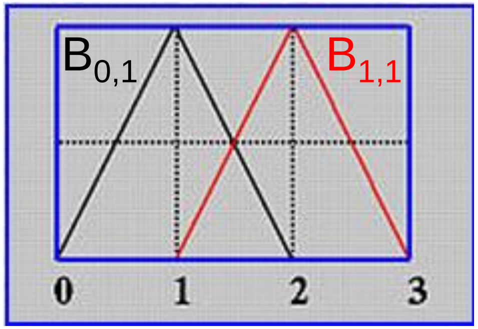
$$B_{i,m}(t) = \frac{t - t_i}{t_{i+m} - t_i} B_{i,m-1}(t) + \frac{t_{i+m+1} - t}{t_{i+m+1} - t_{i+1}} B_{i+1,m-1}(t)$$

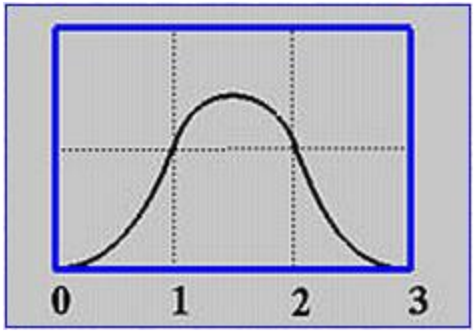
- Ako je m=0, onda su osnovne funkcije step funkcije, npr. za knot vector T = { 0, 1, 2, 3 } date na slici:



- Računanje osnovne funkcije za m>0 je kao prema slici:







- Sada je

$$B_{0,1}(t) = \frac{t - t_0}{t_1 - t_0} B_{0,0}(t) + \frac{t_2 - t}{t_2 - t_1} B_{1,0}(t) = tB_{0,0}(t) + (2 - t)B_{1,0}(t)$$

$$B_{1,1}(t) = \frac{t - t_1}{t_2 - t_1} B_{1,0}(t) + \frac{t_3 - t}{t_3 - t_2} B_{2,0}(t) = (t - 1)B_{1,0}(t) + (3 - t)B_{2,0}(t)$$

$$B_{0,2}(t) = 0.5 \cdot t \cdot B_{0,1}(t) + 0.5 \cdot (3 - t)B_{1,1}(t)$$

B-spline formulacija

- Postoji nekoliko različitih načina koji opisuju B-spline krive (imaju formu Bezier-ove formulacije):

$$W^m(t) = \sum_{i=0}^{n} P_i B_{i,m}(t)$$

  - P – vektor kontrolnih tačaka,
  - B – osnovna funkcija,
  - n – broj kontrolnih tačaka (indeksiran od 0)
  - m – stepen krive,

- Za svaki parametar ***t*** mora se izračunati osnovna funkcija za svaku kontrolnu tačku Pi.
- U slučaju kubne funkcije treba znati m+1 = 4 relevantne kontrolne tačke da bi se izračunao pripadni segment.

Uniformni B-splajnovi

- Kriva je *uniformna* ako se čvorovi nalaze na jednakim intervalima parametra t.
- **Aproksimirajući** splajnovi: kriva aproksimira n+1 kontrolnih tačaka (pošto indeksi kontrolnih tačaka idu od nule)

  $P_0, P_1, \ldots, P_n$

- Kriva se sastoji od n–2 segmenata kubnih polinoma

  $Q_3, Q_4, \ldots Q_n$

  t se menja duž B-splajna kao Q_i: $t_i \leq t < t_{i+1}$

  t_i (i = celi broj) su čvorne tačke koje spajaju segment Q_i sa segmentom Q_{i-1}

  primer:

za kubne polinome	: m=3
dato je 5 kontrolnih tačaka, n=4	: P_0, P_1, P_2, P_3, P_4
broj segmenata je n-2=2	: Q_3, Q_4
broj knotova je n-1=3	: t_3, t_4, t_5

što odgovara jer:

segment	kontrolne tačke
Q_3	p_0, p_1, p_2, p_3
Q_4	p_1, p_2, p_3, p_4

Uniformni B-splajnovi

- Prvi segment krive, Q_3, je definisan pomoću prve 4 kontrolne tačke.
- Poslednji segment krive, Q_n, je definisan pomoću poslednje 4 kontrolne tačke, P_{n-3}, P_{n-2}, P_{n-1}, P_n
- Kontrolna tačka može najviše da utiče na 4 segmenta krive.
- Potrebno je formulisati 16 jednačina da se nađe 16 nepoznatih
    - 16 jednačina forsiraju C_0, C_1, i C_2 kontinuitet između susednih segmenata Q
- Konačna matrica za B-spline je kao što sledi:

$$M_{B-spline} = \frac{1}{6}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}$$

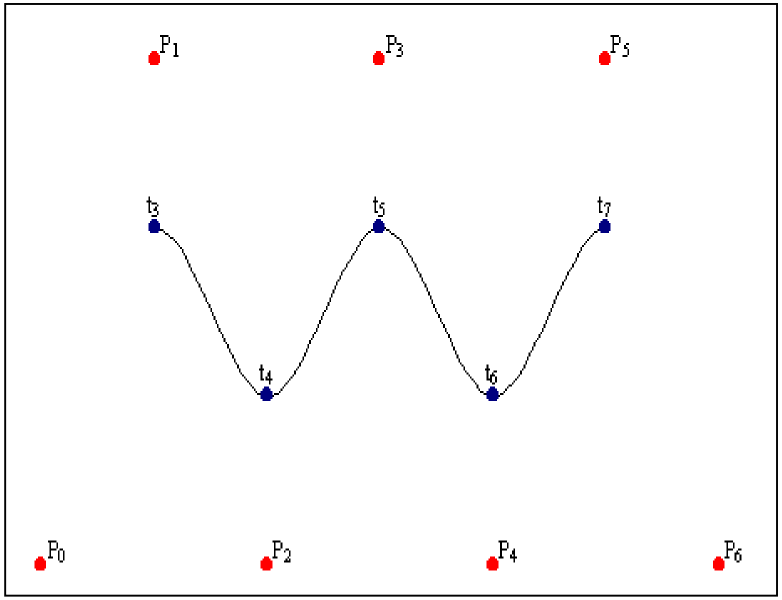
B-splajn

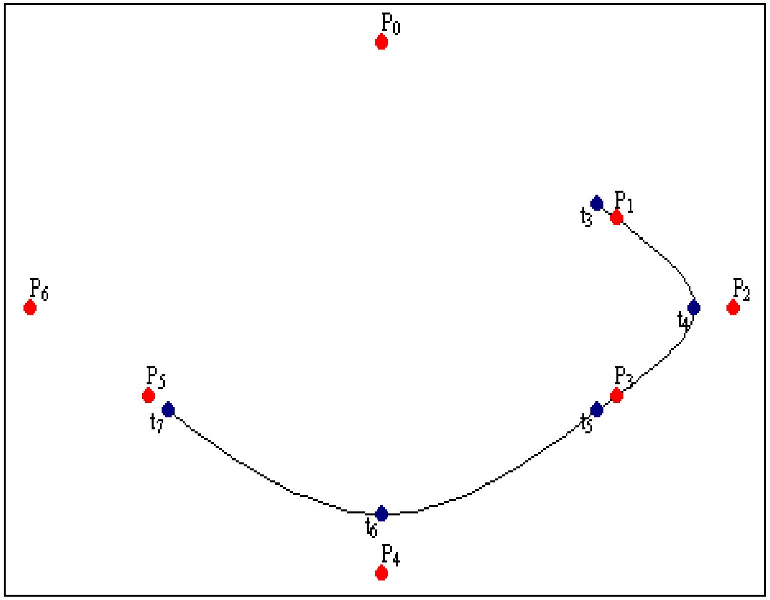
- Tačke duž B-splajna se izračunavaju kao i kod Bezier-ove krive:

$$Q_i(t) = TM_{B-Spline}P$$

$$Q_i(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} p_i \\ p_{i+1} \\ p_{i+2} \\ p_{i+3} \end{bmatrix}$$

- Ovo je najpopularniji splajn koji se koristi (ima C_0, C_1, i C_2 kontinuitet):





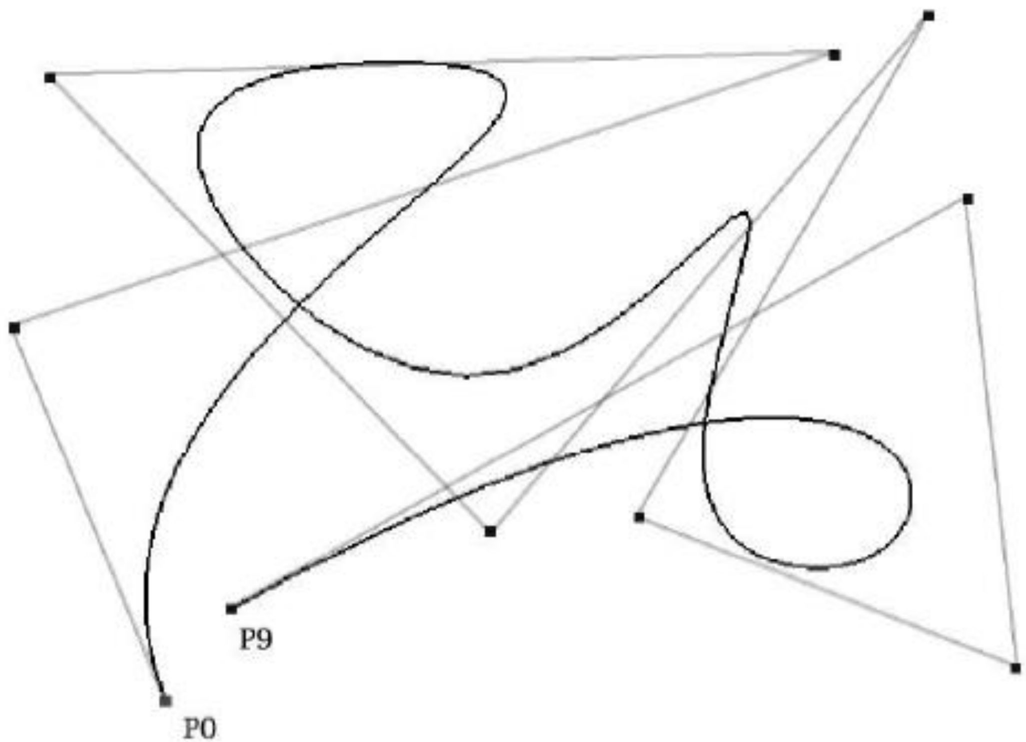
Primer

```
void b_spline(point p1, point p2, point p3, point p4, int divisions){
  double a[4];
  double b[4];
  a[0] = (-p1.x + 3 * p2.x - 3 * p3.x + p4.x) / 6.0;
  a[1] = (3 * p1.x - 6 * p2.x + 3 * p3.x) / 6.0;
  a[2] = (-3 * p1.x + 3 * p3.x) / 6.0;
  a[3] = (p1.x + 4 * p2.x + p3.x) / 6.0;
  b[0] = (-p1.y + 3 * p2.y - 3 * p3.y + p4.y) / 6.0;
  b[1] = (3 * p1.y - 6 * p2.y + 3 * p3.y) / 6.0;
  b[2] = (-3 * p1.y + 3 * p3.y) / 6.0;
  b[3] = (p1.y + 4 * p2.y + p3.y) / 6.0;
  double *splinex = new double[divisions];
  double *spliney = new double[divisions];
  int i;
  for (i = 0; i < divisions ; i++)
  {
    float t = (float)(i) / (float)(divisions);
    splinex[i] = t*( t* (t*a[0] + a[1]) + a[2] ) + a[3] ;
    spliney[i] = t*( t* (t*b[0] + b[1]) + b[2] ) + b[3] ;
    show(splinex[i],spliney[i]);
  }
  delete []splinex; delete []spliney;
}
```

NeUniformni, Racionalni B-Splajnovi (NURBS)

- NURBS je osnovni element geometrije u 3D programu Maya
- Modeli se sastoje od površina definisanih pomoću NURBS-a, a ne poligona.
- NURBS su glatke krive
- Termin NURBS znači Non-Uniform Rational B-Spline i najčešći je metod za modelovanje krivih u računarskoj grafici.

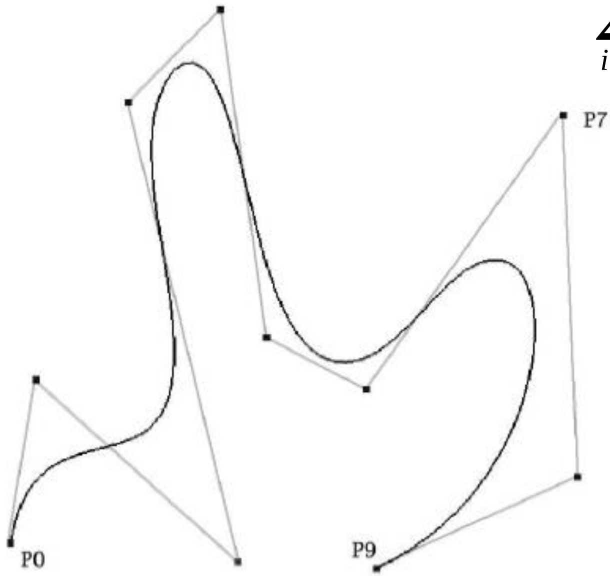
NURBS kriva, $w_i = 1.0 \ \forall i$

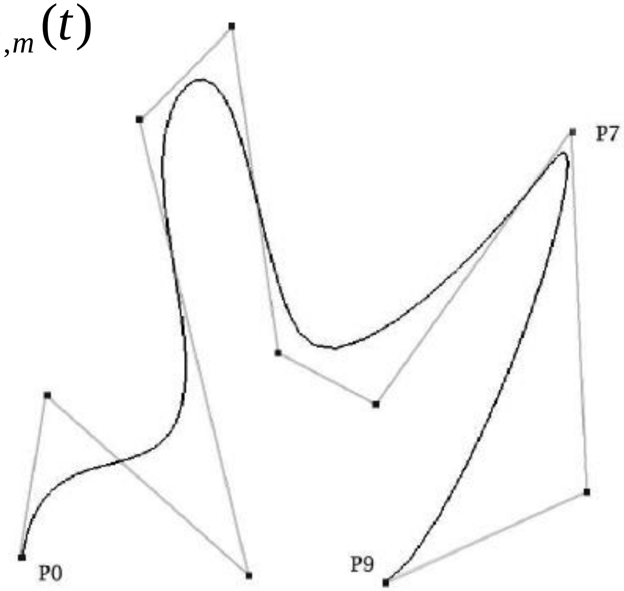


NURBS

- Racionalne krive se odnose na proširenje kontrolne tačke jednom koordinatom koja predstavlja težinu (weight) te kontrolne tačke.
- Na prethodnom slajdu dat je primer NURBS krive gde sve kontrolne tačke imaju težinu 1.0 (to je podrazumevana veličina, “weight-less").

$$W^m(t) = \frac{\sum_{i=0}^{n} w_i P_i B_{i,m}(t)}{\sum_{i=0}^{n} w_i B_{i,m}(t)}$$





NURBS krive: $w_7 = 1.0$, $w_7 = 15.0$ respektivno

Primer 1/2

```
#pragma comment( lib, "opengl32.lib")
#pragma comment( lib, "glu32.lib")
#pragma comment( lib, "glut32.lib")

#include <cmath>
#include <GL/glut.h>

struct point { float x; float y; };

point a = {  40, 100 };
point b = {  80,  20 };
point c = { 150, 180 };
point d = { 260, 100 };

void mt (point &dest, point a, point b, float t){
  dest.x = a.x + (b.x-a.x)*t;
  dest.y = a.y + (b.y-a.y)*t;
}

void bezier (point &dest, float t)
{
  point ab,bc,cd,abbc,bccd;
  mt (ab, a,b,t); mt (bc, b,c,t); mt (cd, c,d,t);
  mt (abbc, ab,bc,t);  mt (bccd, bc,cd,t);
  mt (dest, abbc,bccd,t);
}
```

Primer 2/2

```
void DeCasteljau(){
  point p;
  for ( int i=0; i<1000; i++ ) {
    float t = (float)i/999.0;
    bezier (p,t);
    glColor3f(0,0,0); glBegin(GL_POINTS); glVertex2f(p.x, p.y); glEnd();
  }
}

void displayCB(void){
  glClear(GL_COLOR_BUFFER_BIT);
  DeCasteljau();  glFlush();
}

void keyCB(unsigned char key, int x, int y){
  if( key == 'q' ) exit(0);
}

int main(int argc, char *argv[]){
  glutInit(&argc, argv);
  glutInitDisplayMode(GLUT_RGB);
  glutInitWindowSize(400,500);
  int win = glutCreateWindow("Kosteljau");
  glClearColor(1.0,1.0,0.0,0.0);
  gluOrtho2D(0,400,0,500);
  glutDisplayFunc(displayCB);  glutKeyboardFunc(keyCB);
  glutMainLoop();return 0;
}
```