IZG – cvičení #4.
Zobrazování 2D křivek

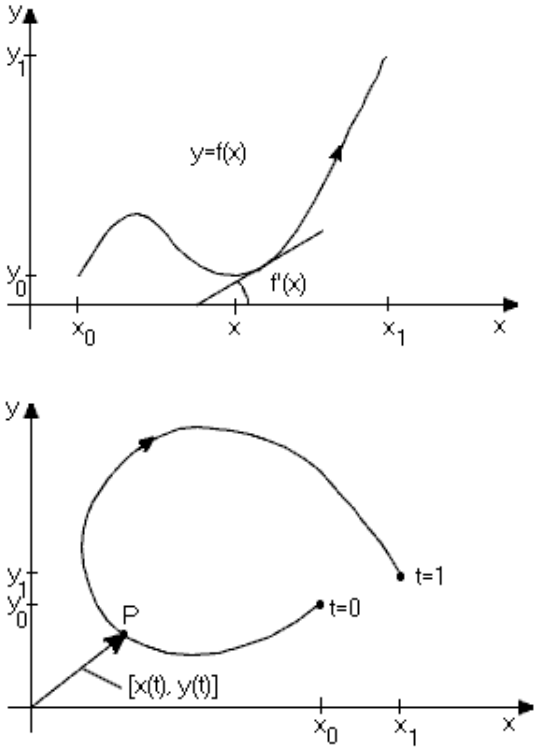
Autor: Jiří Venera
Upravili: Radovan Jošth, Pavel Svoboda

Obsah

- Úvodní část
  - Rekapitulace 2D křivek
- Cvičení - ukázka
  - Fergusnova kubika
- Cvičení samostatný projekt
  - Úvod
  - Racionální Bézierova křivka

Rekapitulace 2D křivek

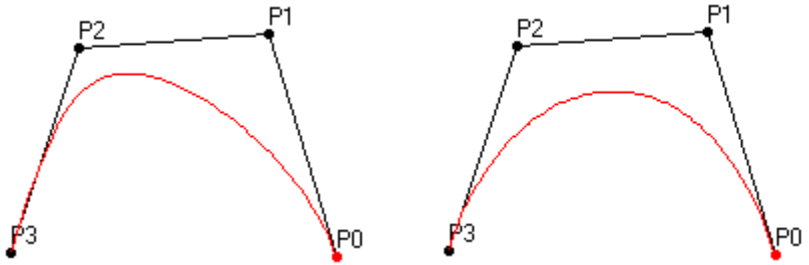
- Vyjádření křivek
  - Explicitní, Implicitní
  - Parametrické
- Použití 2D křivek
  - Definice fontů
  - Šablonování
  - Definice objektů
  - Animační křivky
  - …

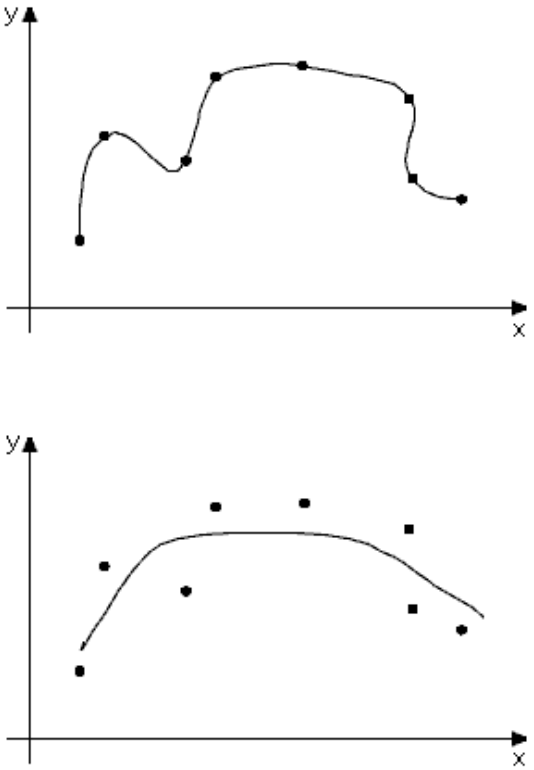


[F. Alexandr]

Rekapitulace 2D křivek

- Typy křivek
  - Aproximační (obecně neprochází řídícími body), interpolační (prochází řídícími body)
  - Racionální (váhové koeficienty řídících bodů), neracionální (váhové koeficienty rovny 1)



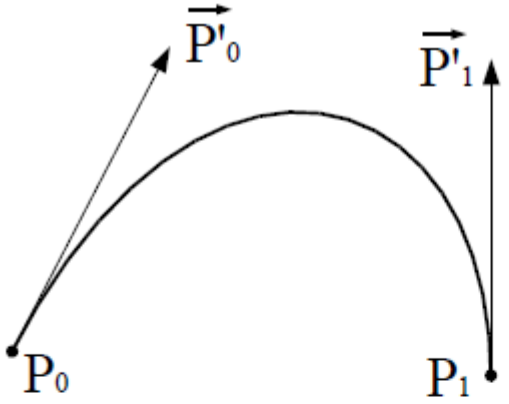


[F. Alexandr]

Co projedeme společně?

Fergusnovu kubiku!

Fergusnova kubika



- Nejčastější interpolační křivka
- Určena dvěma koncovými body
- $P_0; P_1$ (poloha)
- A dvěma tečnými vektory
- $\overrightarrow{P_0}; \overrightarrow{P_1}$ (vyklenutí)
- Neintuitivní řízení tvaru
- Nelokální změna tvaru

$$Q(t) = \mathbf{T} \cdot \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_0 \\ P_1 \\ \vec{P}_0' \\ \vec{P}_1' \end{bmatrix}$$

$$Q(t) = P_0 \cdot F_1(t) + P_1 \cdot F_2(t) + \vec{P}_0' \cdot F_3(t) + \vec{P}_1' \cdot F_1(t)$$

Hermitovy polynomy:

$$F_1(t) = 2 \cdot t^3 - 3 \cdot t^2 + 1$$
$$F_2(t) = -2 \cdot t^3 + 3 \cdot t^2$$
$$F_3(t) = t^3 - 2 \cdot t^2 + t$$
$$F_4(t) = t^3 - t^2$$

$$C_1 \rightarrow \begin{bmatrix} P_0 \\ P_1 \\ \vec{P}_0' \\ \vec{P}_1' \end{bmatrix} \leftrightarrow \begin{bmatrix} P_2 \\ P_3 \\ \vec{P}_2' \\ \vec{P}_3' \end{bmatrix}$$

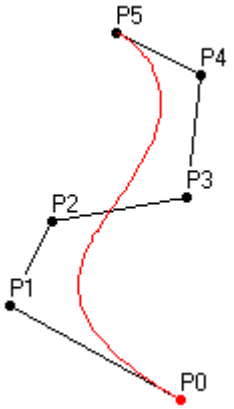
Samostatná práce?

Racionální bézierova křivka!

Neracionální Bézierova křivka

- Váhové koeficienty řídících bodů rovny jedné: w=1, ve vzorci proto nejsou zapsány
- Křivka n-tého stupně je dána *n+1* body $P_i = [x_i, y_i]$ řídícího polygonu $P_0, P_1, \ldots, P_n$

$$Q(t) = \sum_{i=0}^{n} P_i B_i^n(t)$$



kde B_i^n jsou Bernsteinovy polynomy *n-tého* stupně

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}, t \in \langle 0,1 \rangle, i = 0, \ldots, n$$

- Generování se provádí postupným dosazováním parametru *t*

Bézierova kubika

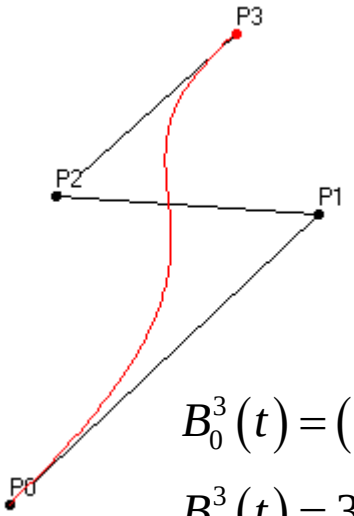
- Počet bodů řídícího polygonu je roven čtyřem P_0, P_1, P_2, P_3
- Maticový zápis

$$Q(t) = T \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

kde $T = [t^3, t^2, t^1, t^0 = 1]$

- Jiný zápis kde B_i jsou Bernsteinovy polynomy stupně 3

$$Q(t) = P_0 B_0^3(t) + P_1 B_1^3(t) + P_2 B_2^3(t) + P_3 B_3^3(t) = \sum_{i=0}^{3} P_i B_i^3(t)$$



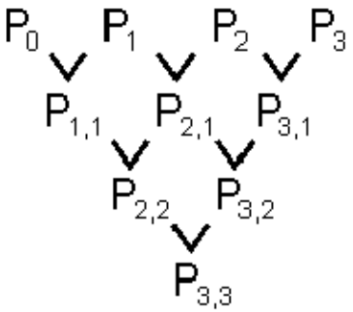
$$B_0^3(t) = (1-t)^3$$
$$B_1^3(t) = 3t(1-t)^2$$
$$B_2^3(t) = 3t^2(1-t)$$
$$B_3^3(t) = t^3$$

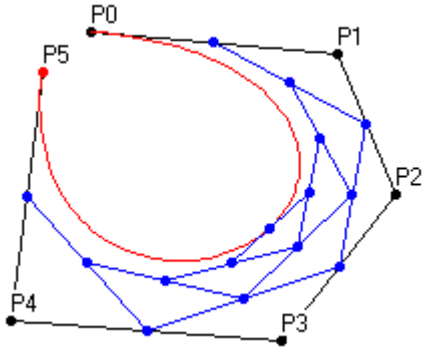
Algoritmus de Casteljau

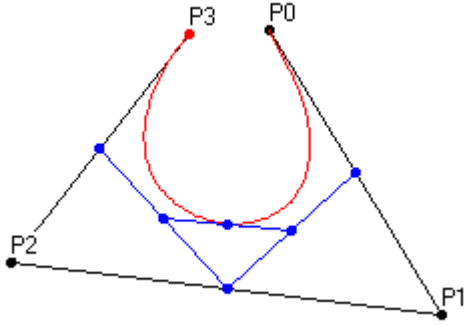
- Výpočet bodů Bézierovy křivky
  - Využívá rekurzivní definice Bersteinova polynomu

$$P_{j,i}(t) = (1-t)P_{j-1,i}(t) + tP_{j-1,i-1}(t)$$

kde $i = 1, 2, \ldots, n$; $j = i, i+1, \ldots, n$ a t určuje poměr dělení stran řídícího polynomu







Racionální Bézierova křivka

- Váhové koeficienty řídících bodů nabývají proměnných hodnot
- Křivka n-tého stupně je dána *n+1* body $P_i = [x_i, y_i]$ řídícího polygonu P_0, P_1, ... , P_n a váhovými koeficienty jednotlivých bodů ω_i
- Racionální Bézierovy polynomy

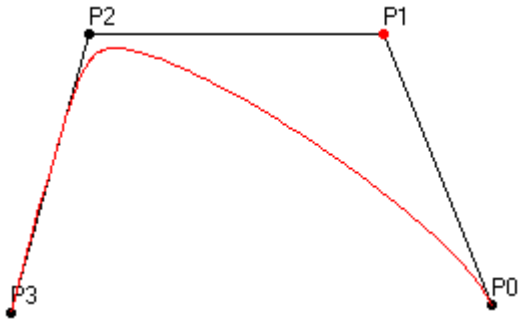
$$R_i^n(t) = \frac{\omega_i B_i^n(t)}{\sum_{i=0}^{n} \omega_i B_i^n(t)}$$

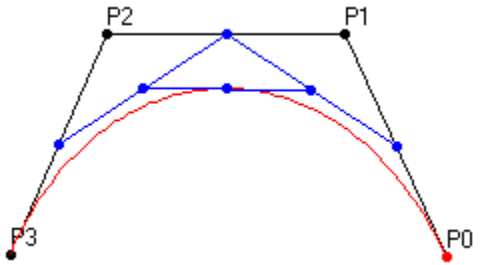
kde B_i^n jsou Bernsteinovy polynomy *n-tého* stupně definované stejně jako u neracionální Bézierovy křivky

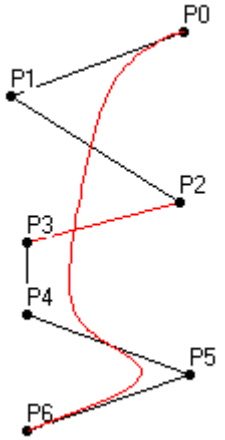
Racionální Bézierova křivka

- Výpočet bodů racionální Bézierovy křivky

$$Q(t) = \sum_{i=0}^{n} P_i R_i^n(t) = \frac{\sum_{i=0}^{n} \omega_i P_i B_i^n(t)}{\sum_{i=0}^{n} \omega_i B_i^n(t)}$$







Shrnutí a doporučení

Bernsteinovy polynomy *n-tého* stupně

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}, t \in \langle 0,1 \rangle, i = 0,\ldots,n$$

Racionální Bézierova křivka

$$Q(t) = \sum_{i=0}^{n} P_i R_i^n(t) = \frac{\sum_{i=0}^{n} \omega_i P_i B_i^n(t)}{\sum_{i=0}^{n} \omega_i B_i^n(t)}$$

Projekt – Kam psát zdrojový kód?

- Soubor student.cpp
  - Funkce void **Bezier**(**int** quality, **const T_PointVector** & points, **T_PointVector** & line_points)
  - **quality** – hladkost výsledné křivky (počet bodů)
  - **points** – pole bodů řídícího polygonu
  - **line_points** – výsledný vektor vygenerovaných bodů křivky
  - Pozn.: **Nic nevykreslujte, body ukládejte do vektoru!**

Základní struktury v main.h

- Bod ve 2D

```
struct S_Point {
        double x, y;        // souřadnice bodu
        double weight;      // váha bodu
                // Konstruktory (C++)
        S_Point() : x(0.0), y(0.0), weight(1.0) {}
        S_Point(double _x, double _y) : x(_x), y(_y),weight(1.0) {} };
```

- Vektor/pole bodů
  - **typedef vector**<S_Point> T_PointVector;
  - Šablona vector – knihovna STL
  - Dynamické pole – velikost lze měnit za běhu programu
  - **#include <vector>**
  - **direktivou using zpřístupnit jmenný prostor std - using namespace std**

STL šablona vector

- Nastavení velikosti pole/vektoru
  - Při vytváření – parametr konstruktoru
  - Př.: **T_PointVector P(n);**
  - Metoda **resize(n)**
- Indexace pole
  - Operátor **[]** – přístup k prvkům jako obyčejné pole
  - Př.: **P[i]** = prvek;
- Metody
  - Funkce definované „uvnitř“ struktury/třídy
  - Pro volání použijte '**.**' případně '**->**' notaci
  - **push_back(prvek)** – zvětší pole o 1 a vloží prvek na konec
  - **clear()** – vyprázdnění pole, nová velikost = 0
  - **size()** – vrací aktuální velikost pole

Literatura

- Přednášky IZG + uvedená literatura
- L. Alexandr: Výuka křivek formou WWW, 1999.
- http://lubovo.misto.cz/_MAIL_/curves/index.html
- www.google.com … BSpline

Kontakty

- ijosth@fit.vutbr.cz
- isvoboda@fit.vutbr.cz