Matematično modeliranje

2. projekt:

Bézierovi zlepki

Mentor
Martin Vuk

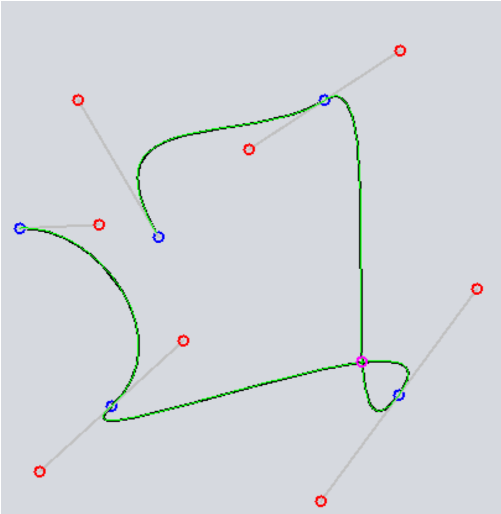
predmet Matematično modeliranje
Fakulteta za računalništvo in informatiko
11.6.2015, Ljubljana

Avtorji
Juš Debelak
Jure Jesenšek
Egidij Egej Vencelj
Tilen Fišer

Kazalo

Opis problema

Konstruirali bi radi gladek zlepek sestavljen iz Bézierovih krivulj, ki gre
skozi dane točke v ravnini. Za izbrani zlepek izračunamo še dolžino
zlepka in morebitna samopresečišča.



Slika 1: Bezierov zlepek

Naloga

V okviru naloge je potrebno konstruirati zlepek sestavljen iz Bézierovih krivulj iste stopnje, ki je načeloma majhna (med 2 in 5 – v našem primeru 3). Naloge so:

1. Izpeljati, katerim pogojem morajo zadoščati kontrolni poligoni krivulj v zlepku, da bo zlepek zvezno odvedljiv.
2. Napisati pomožne funkcije za računanje točk in izris Bézierove krivulje s podanim poligonom. Izračun naj bo izveden z De Casteljauovem algoritmom.
3. Za dane točke v ravnini poiskati zvezno odvedljiv zlepek sestavljen iz Bézierovih krivulj iste stopnje.

Izpeljava

- za lepljenje zlepkov s pomočjo 1. odvoda izračunamo mesto 1. kontrolne točke krivulje, ki jo zlepku priključujemo

$$B'(t) = 3(1-t)^2(P_1 - P_0) + 6(1-t)t(P_2 - P_1) + 3t^2(P_3 - P_2)$$

$$B_i'(1) = 3(P_{3,i} - P_{2,i})$$
$$B_{i+1}'(0) = 3(P_{1,i+1} - P_{0,\,i+1})$$

$$3(P_{3,i} - P_{2,i}) = 3(P_{1,\,i+1} - P_{0,i+1})$$
$$K_{i+1} = P_{3,i} = P_{0,i+1}$$

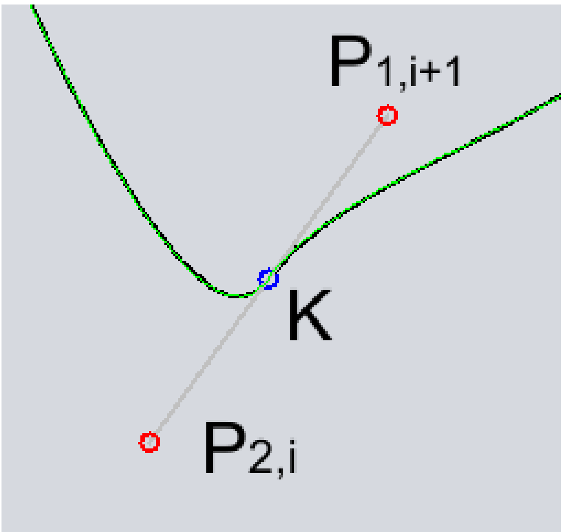
$$K_{i+1} - P_{2,i} = P_{1,i+1} - K_{i+1}$$

$$P_{1,i+1} = 2K_{i+1} - P_{2,i}$$

- ko izpeljemo 1. kontrolno točko naslednje krivulje, opazimo, da je to le preslikava 2. kontrolne točke iz trenutne krivulje čez točko K (zadnjo točko trenutne in prvo naslednje krivulje)
- za določanje 2. kontrolne točke, ki je v našem programu še prosta, se uporablja izpeljava iz 2. odvoda

$$P_{2,n-1} = \tfrac{1}{2}K_n + P_{1,n-1}$$

- iz zgornjih postopkov dobimo zvezno odvedljiv zlepek

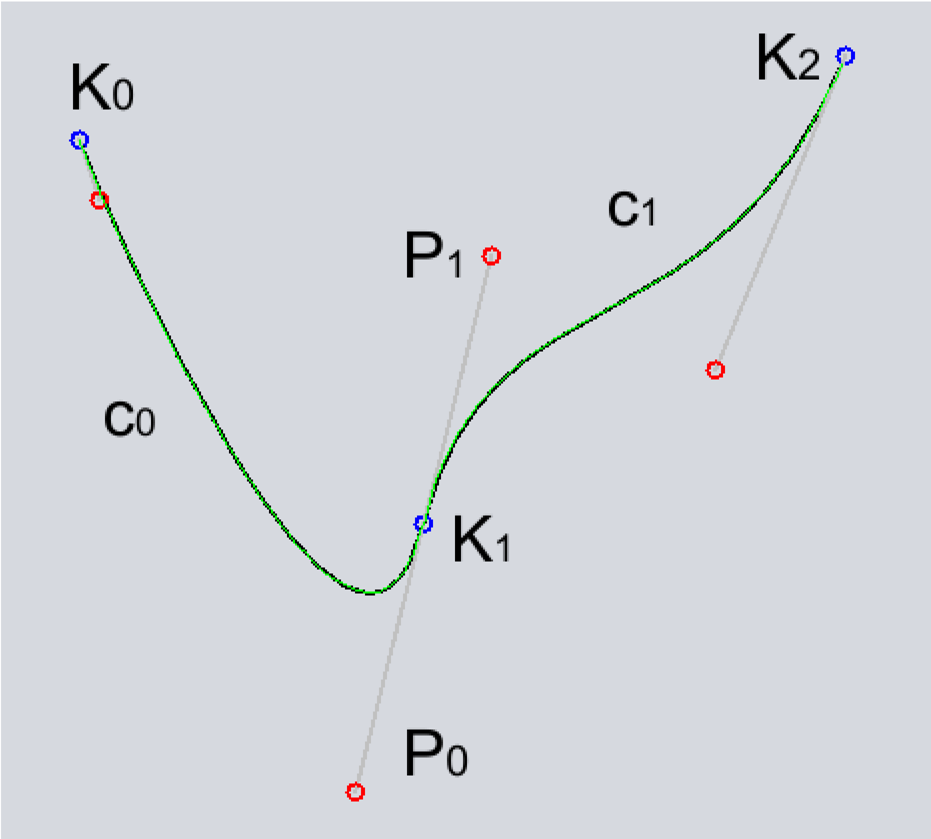


Slika 2: Položaj kontrolnih točk

Postopek

Uporabili smo programski jezik Java, ki ima soliden vmesnik za izrisovanje in interaktivno dodajanje točk, daljic, krivulj…

- Grafični vmesnik programa uporablja Java Swing – s pomočjo vgrajenih funkcij izrisujemo kontrolne točke in daljice, krivulje in točke skozi katere poteka krivulja.
- Za sam izračun točk krivulje uporabljamo De Casteljauov algoritem. Odločili smo se za iterativno različico, saj je napram svoji rekurzivni alternativi numerično stabilna
- Razvili smo tudi pomožne funkcije za izračun dolžine krivulje in izračun točk samopresečišč krivulje (če te obstajajo)
- V programu s kliki v ravnini določamo skozi katere točke (K_x) naj poteka Bezierjev zlepek
- Program določi usmerjenosti kontrolnih točk (P_x) in potem izriše krivuljo med točkama (c_x), glede na kontrolni poligon, določen s točkami
- Pri vstavljanju nadaljnjih točk K_x, program določi lego pripadajoče kontrolne točke po zgodaj dokazani enačbi, kar pomeni da morata biti daljici, ki se dotikata iste podane točke, na isti premici
- Program omogoča interaktivno prestavljanje kontrolnih točk, in s tem tudi spreminjanje oblike samega zlepka
- Pri vsaki postavitvi nove točke, ali pri spreminjanju kontrolnega poligona, program izračuna in izpiše skupno dolžino zlepka in število samopresečišč



Slika 3: Primer krivulje

Legenda

- K_x - podana točka
- c_x – krivulja, ki povezuje točki K_x in K_{x+1}
- P_x – izračunana kontrolna točka, ki določa obliko krivulje

Ugotovitve

1. Pri sosednjih kontrolnih poligonih (2 podani točki in pripadajoči kontrolni točki) morata biti stranici poligonov, ki imata skupno podani točko, na isti premici (Slika 3: daljici P_0K_1 in K_1P_1 morata biti na isti premici)
2. Funkcija v Java Swing pri izrisu krivulj vrača primerljive krivulje kot De Casteljauov algoritem (na slikah: črna krivulja je izrisana z vgrajenimi funkcijami, zelena pa z De Casteljauovem algoritmom)
3. Že pri generiranju majhnega števila točk s pomočjo De Casteljauovega algoritma dobimo dobre pribljižke, še posebej, če krivulja ni zelo ukrivljena

Razporeditev dela

- Juš Debelak: risanje podatkov, dodajanje in spreminjanje krivulj, izpeljava zveznosti zlepkov, poročilo
- Jure Jesenšek: uporabniški vmesnik, pomoč pri programiranju, poročilo
- Egidij Egej Vencelj: De Casteljauvov algoritem, računanje dolžine, iskanje samopresečišč, formatiranje poročila in popravki
- Tilen Fišer: predstavitev, testiranje implementacij

Viri

- Java dokumentacija
- Wikipedia - Bézier curve
- Wikipedia - De Casteljau's algorithm
- Particle In Cell Consulting LLC
- Michigan Tech course