Projeciranje Bézierjeve krivulje

Poročilo o projektu pri predmetu Matematično modeliranje

Gal Zakrajšek
27171031

September 5, 2021

Contents

1 Opis projekta

V projektu je opisan postopek primerjave dveh Bézierjevih krivulj, ki jih dobimo z naslenjimi postopki.

Prvo krivuljo dobimo tako, da imamo podane kotrolne točke v tridimenzionalnem prostoru s katerimi dobimo Bézierjevo krivuljo. To potem prezrcalimo v smeri izbranega vektorja na ravnino $z = 0$.

Drugo krivuljo pa dobimo tako, da najprej prezrcalimo kotronlne točke v smeri enakega izbranega vektorja na ravnino $z = 0$ in potem v dvodimenzionalme prostoru narišemo Bézierjevo krivuljo.

2 Bézierjeva krivulja

Bézierjeva krivulja je krivulja, ki jo pridobimo s pomočjo kontrolnih točk. Te se lahko nahajajo na ravnini ali pa tudi v prostoru. Krivulja je tako lahko ravninska ali pa prostorska. Do točk, ki bodo predstavljale Bézierjevo krivuljo pridemo s pomočjo De Casteljauovega algoritma. Le ta nam poda rekurzivno formulo, s katero pridemo do končnih točk na krivulji.

Imamo začetke kontrolne točke $b_0, b_1, ..., b_n$ in rekurzivno formulo:

$$b_j^k(t) = (1-t)b_j^{k-1}(t) + tb_{j+1}^{k-1}(t) \tag{1}$$

Kjer gre $k = 1, 2, ..., n$ in $j = 0, 1, ..., n - r$. Za $b_n^0(t)$ vzamemo kar kontrolne točke.

Ko poračunamo rekurzivno vse vrednosti, dobimo v $b_0^n(t)$ točko na Bézierjevi krivulji pri nekem parametru t. Celotno krivuljo oziroma vse točke krivulje dobimo, ko naš parameter t preteče celoten inteval $[0, 1]$. Podajmo še grafični prikaz računanja koeficientov.

$$\begin{array}{llll} b_0^0(t) & & & \\ b_1^0(t) & b_0^1(t) & & \\ b_2^0(t) & b_1^1(t) & \dots & \\ \vdots & \vdots & \ddots & \\ b_n^0(t) & b_{n-1}^1(t) & b_1^{n-1} \to b_0^n & \end{array} \tag{2}$$

Prvi stolpec imamo že podan, saj so to kar naše kotrolne točke. Potem pa z formulo (1) računamo naslednje stolpce, dokler ne pridemo do končne točke b_0^n, ki je točka na krivulji.

3 Ideja reševanja

Ideja reševanja je taka, da imamo v matlabu dve funkciji s pomočjo katerih lahko rešimo celoten problem.

3.1 Izracunaj_krivuljo(zacetne, gladkost)

Funckija nam bo z pomočjo rekurzivne formule (1) izračunala vse končne točke naše Bézierjeve krivulje. Začetke točke so podane v prvem parametru. Točke pa morajo biti podane vsaka v svoji vrsti matrike. Medtem ko drugi parameter določa gladkost krivulje oziroma koliko končnih točk bo imela. V definiciji smo to označevali z parametrom t. Parameter vzame celoštevilo in na koncu nam funckija vrne toliko točk kolikor je to število.

Funckija je napisana na način, da če ji podamo točke v ravnini, bo izračunala krivuljo na ravnini. V primeru, da imamo začetne točke v prostoru pa izračuna tridimenzionalno Bézierjevo krivuljo.

3.2 Prezrcali_tocke(tocke, smer)

Funkcija prezrcali dane točke v ravnino $z = 0$ v smeri izbranega vektorja. Prvi parameter so ponovno točke, ki so podane vsaka v svoji vrsti matrike. Drugi parameter pa je vektor, ki opisuje smer iz katerega preslikamo naše točke na ravnino. Preslikamo točke na naslednji način:

$[a_i, b_i, c_i]$: točke, ki jih preslikamo na ravnino $z = 0$
$[x, y, z]$: izbrani smerni vektor
Izračunamo koeficient, ki nam pove za koliko moramo točko premakniti navzdol.

$$koef\_zmanjsanja = c_i/z \tag{3}$$

Nato s pomočjo tega koeficienta in naslednje formule dobimo kočne koordinate točk.

$$koncna\_tocka_i = [a_i - x * koef\_zmanjsanja,\ \ b_i - y * koef\_zmanjsanja] \tag{4}$$

Kot rezultat nam funkcija vrne matriko točk, ki so sedaj v ravnini,vsaka točka je shranjena v svoji vrstici.

V našem projektu pa potrebujemo enkrat preslikati kontrolne točke, drugič pa krivuljo. Vendar ker je krivulja v našem primeru podana z končno mnogo točkami, lahko za obe preslikavi uporabimo isto funkcijo.

3.3 Ostala koda

V glavni datoteki main, pa je preostala koda, kjer rišemo krivulje v ravnini in v prostoru. Za risanje se uporabljata funkciji plot in plot3D. Za dodatno potrditev enakosti, pa je spodaj tudi stavek "isequal", ki primerja matriki točk, ki predstavljata krivulji in vrne 1, če vsebujeta vse enake točke.

4 Primerjanje krivulj in končne ugotovitve

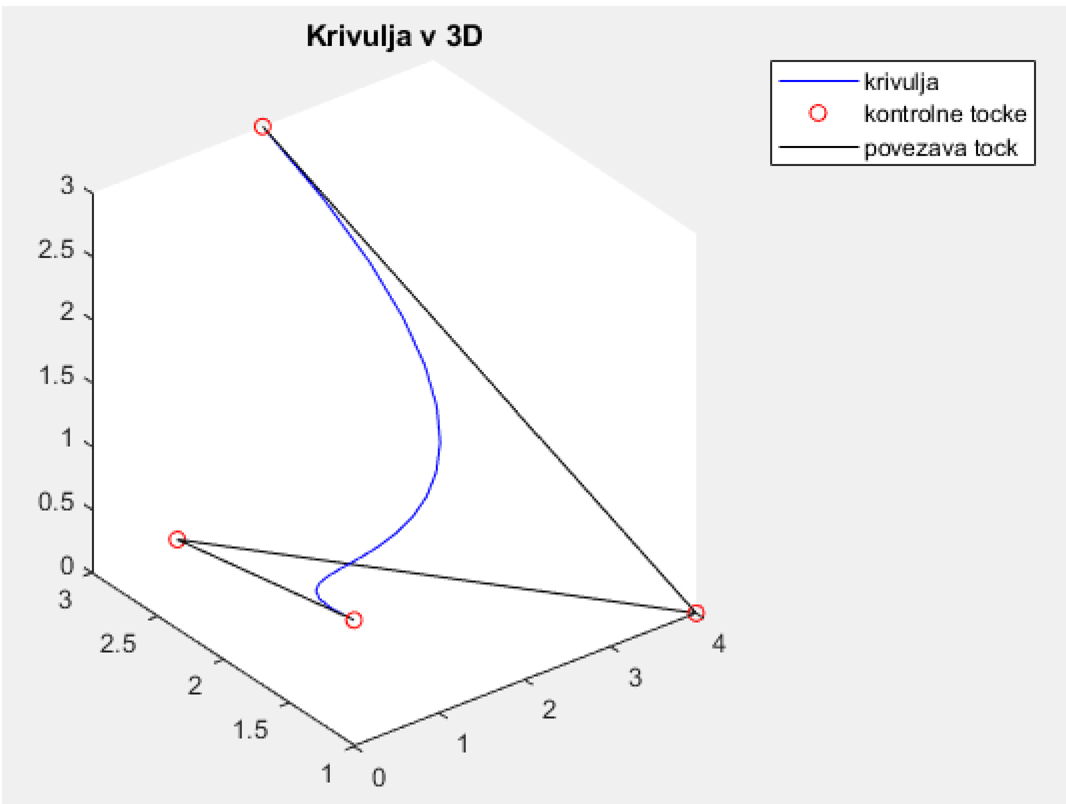
Zdaj, ko imamo definirana vsa potrebna orodja, pa si lahko pogledamo dva primera. Obe krivulji bomo preslikali najprej v smeri $[0, 0, 1]$ in pa $[0.2, -0.3, 0.5]$.

4.1 Krivulja iz 4 kontrolnih točk

Podane imamo koordinate štirih kontrolnih točk:

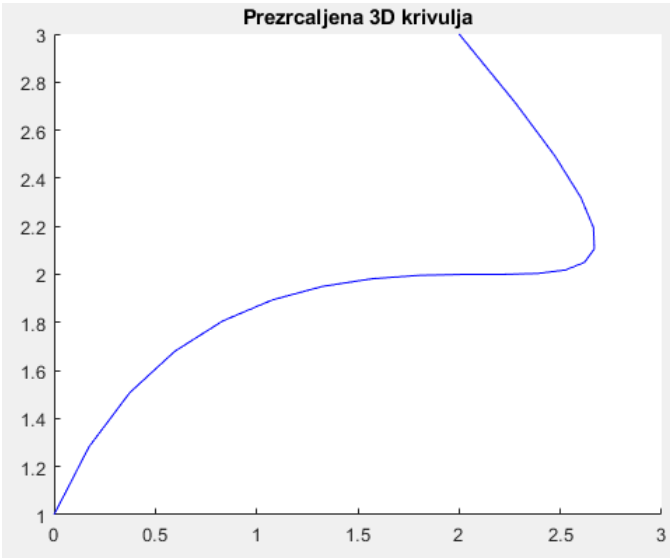
$$A = [0, 1, 1] \quad B = [1, 3, 0] \quad C = [4, 1, 0] \quad D = [2, 3, 3] \tag{5}$$

S pomočjo funkcije izracunaj_krivuljo, izračunamo vse točke Béizierjeve krivulje, z gladkostjo 20. Kar pomeni, da dobimo 20 točk, ki opisujejo našo krivuljo. Takšen je njen graf:

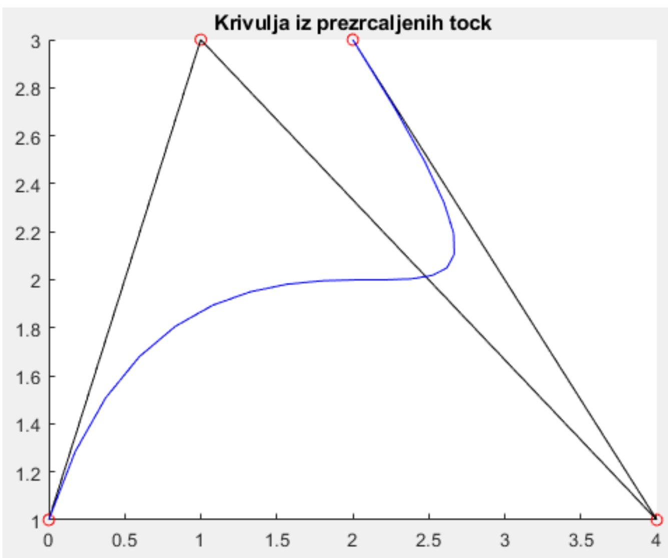


Poglejmo najprej, kako se ta krivuljo preslika v smeri vektorja $[0, 0, 1]$.

Uporabimo funkcijo prezrcali_tocke in kot parameter vstavimo točke, ki smo jih dobili v prejšnjem koraku. Tako dobimo nove točke, ki so sedaj v ravnini $z = 0$ in tako lahko krivuljo ponovno narišemo. Graf:

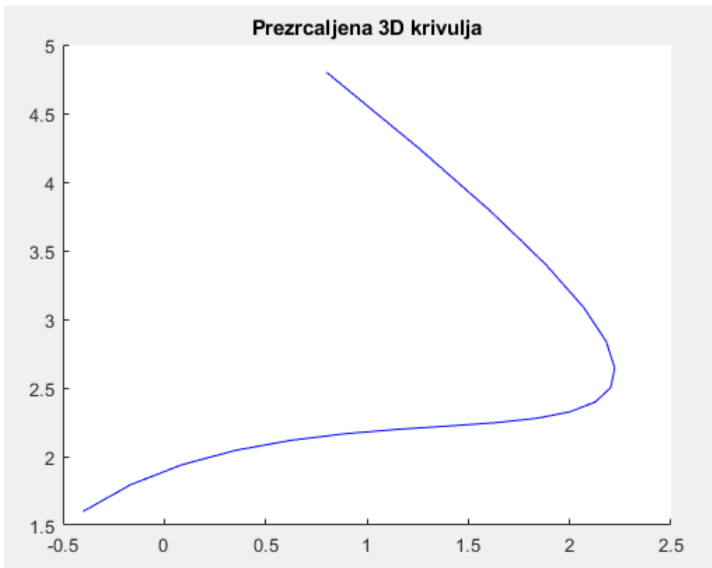


Za primerjavo pa potrebujemo še obratni vrstni red, in sicer prvo prezrcaljene kontrolne točke in nato izračunano krivuljo. To storimo tako, da prvo uporabimo funkcijo prezrcali_tocke na naših kontrolnih točkah in potem te točke vstavimo v funkcijo izracunaj_krivuljo. Dobimo naslednjo krivuljo.

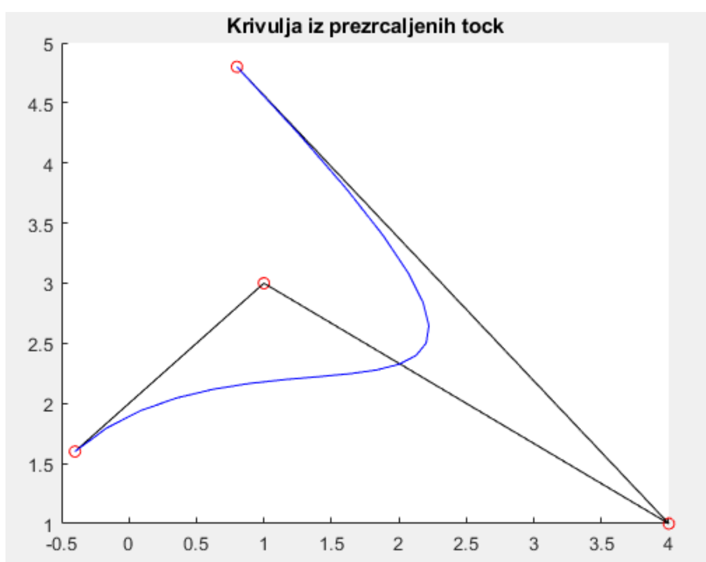


Že po slikah lahko vidimo, da sta krivulji isti. To pa nam tudi potrdi funkcija "isequal". saj če primerjamo matriki točk obeh krivulj, nam funkcija pove da se ujemata.

Zdaj poglejmo še primerjavo obeh grafov, če spremenimo smerni vektor zrcaljenja. Uporabimo vektor $[0.2, -0.3, 0.5]$ Graf krivulje, če jo najprej narišemo in nato prezrcalimo:



Graf, če najprej prezrcalimo točke:



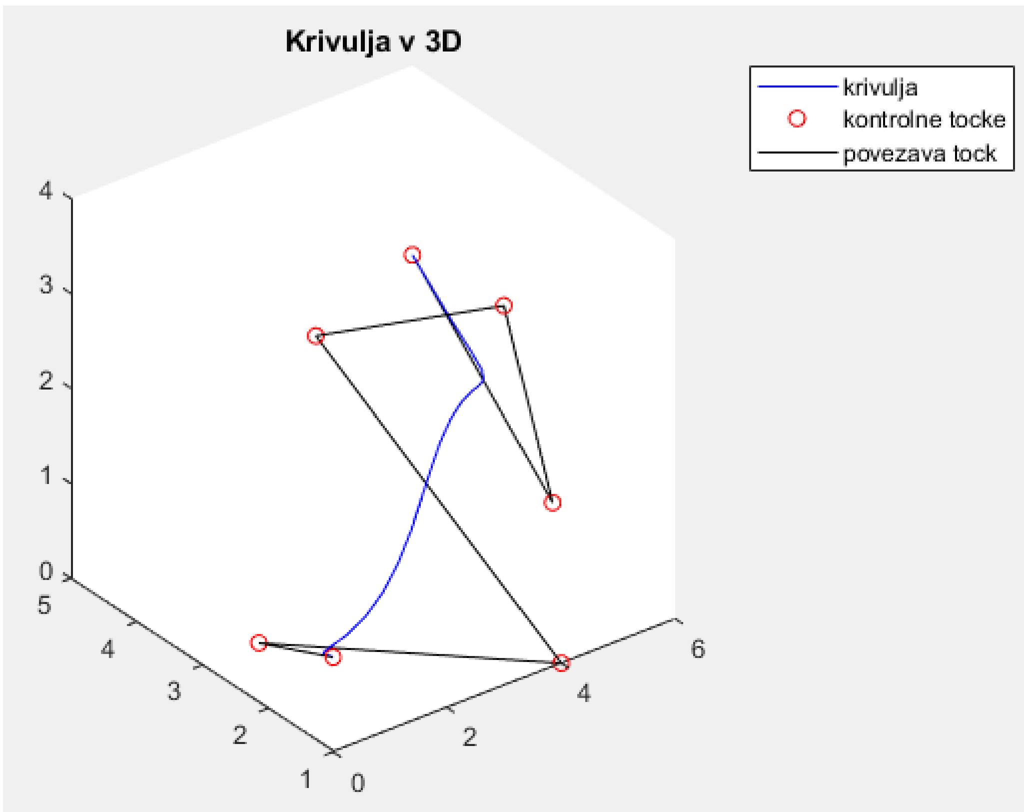
Ponovno iz grafov vidimo, da sta krivulji enaki.

4.2 Krivulja iz 7 kontrolnih točk

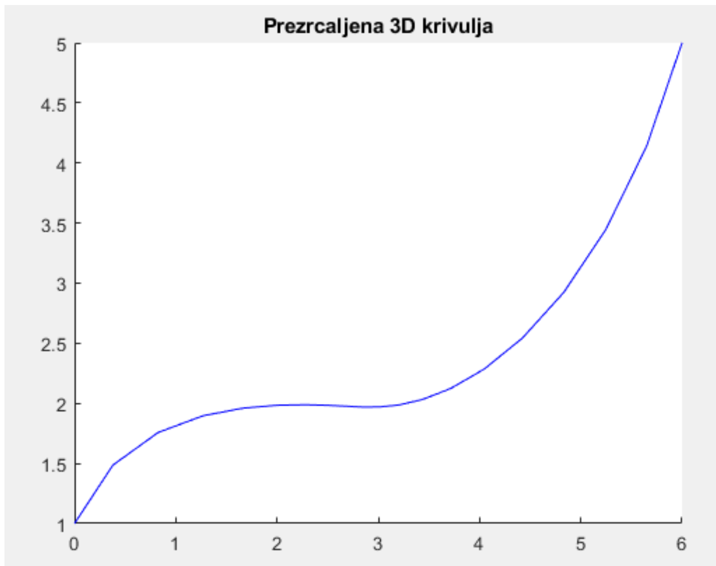
Poglejmo si še kako izgleda preslikava krivulje z večimi kontrolnimi točkami. Postopek reševanja je enak, zato ga ne bomo ponavljali. Podane imamo koordinate sedmih kontrolnih točk:

$$\begin{array}{llll} A = [0,1,1] & B = [1,3,0] & C = [4,1,0] & D = [2,3,3] \\ E = [3,1,4] & F = [5,2,1] & G = [6,5,2] & \end{array} \tag{6}$$

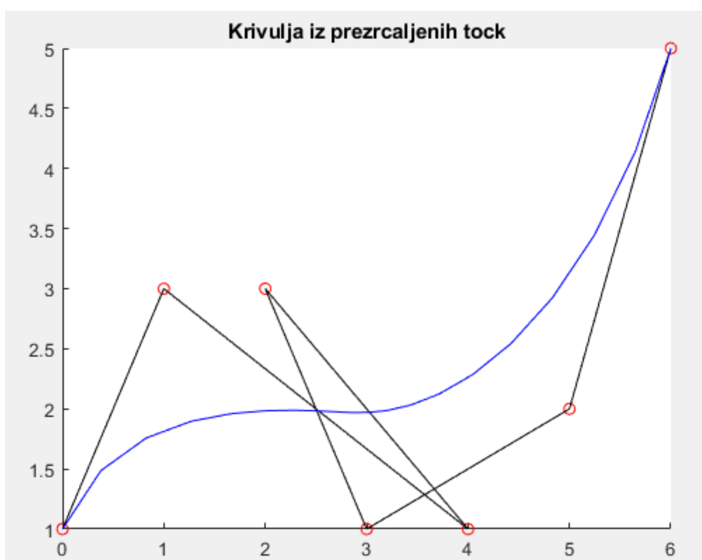
Graf 3D krivulje:



Preslikajmo najprej skozi smerni vektor $[0, 0, 1]$: Graf preslikane 3D krivulje:

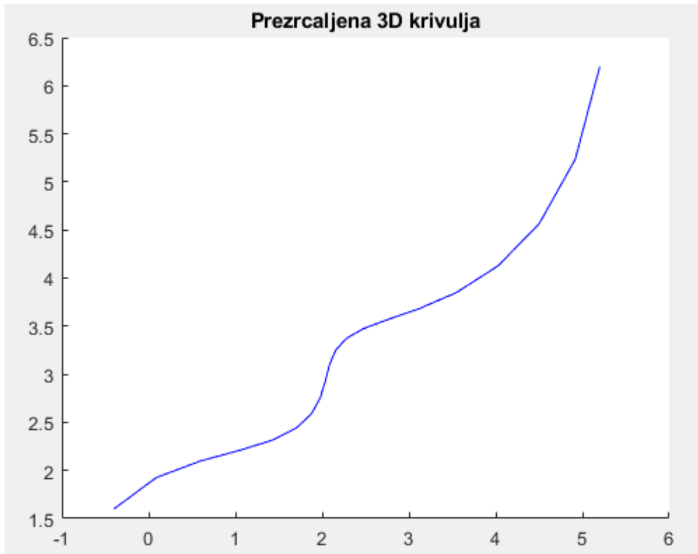


Če najprej preslikamo kontrolne točke, pa dobimo sledeči graf:

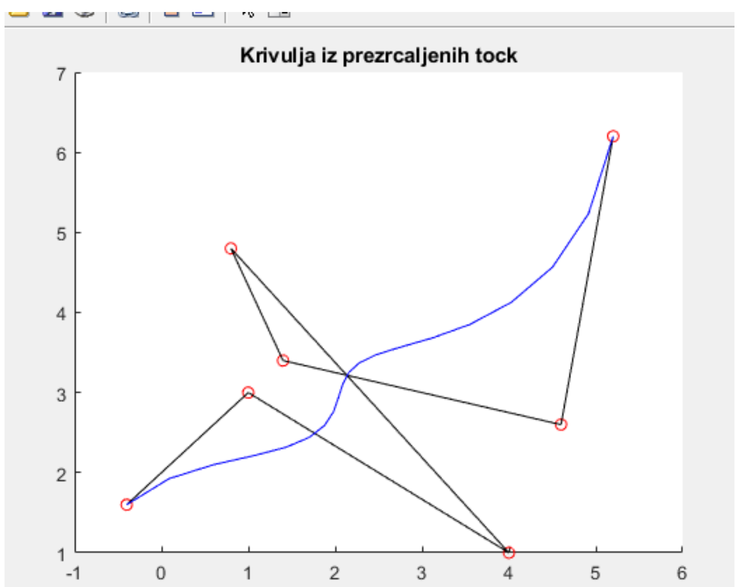


Iz grafa je ponovno razvidno, da se krivulji ujemata. Seveda vedno lahko to preverimo tudi z funkcijo "isequal".

Kot zadnji primer pa si poglejmo še preslikavo zgornje krivulje v smeri izbranega vektorja $[0.2, -0.3, 0.5]$. Graf preslikane 3D krivulje:



Če najprej preslikamo kontrolne točke, pa dobimo sledeči graf:



Tudi tukaj, če primerjamo grafa vidimo, da sta krivulji enaki.

5 Sklep

Pri risanju in primerjanju krivulj smo ne glede na to kakšno zaporedje med preslikavovanjem in računanjem Béizierjeve krivulje dobili enak rezultat. Zato lahko sklepamo, da je Béizierjeva krivulja v obej primerih zaporedja operacij enaka.

6 Viri

- Bezier Curve - 3D Curves - Computer Aided Design https://youtu.be/f1iMl-gXhQg dostop:(02.09.2021)
- Volaš V. Racionalne Bézierjove krivulje https://www.famnit.upr.si/files/zakljucna_dela_repo/90 dostop:(02.09.2021)
- Holmer F. The Beauty of Bézier Curves https://youtu.be/aVwxzDHniEw dostop:(02.09.2021)
- Jaklič G. Krivulje in ploskve v računalniško podprtem geometrijskem oblikovanju https://www.fmf.uni-lj.si/~jaklicg/CAGD.pdfdostop:(02.09.2021)