SVEUČILIŠTE U SPLITU

FAKULTET ELEKTROTEHNIKE, STROJARSTVA I BRODOGRADNJE

Matematičko modeliranje forme broda

(manual and software test)

BRANKO BLAGOJEVIĆ

Zadatak:

Za aproksimaciju neke hidrodinamičke plohe definirane na pravokutnoj mreži točaka (x_i, y_j), i = 0,1,…,m ; j = 0,1,…,n , potrebno je definirati dvodimenzionalnu spline funkciju u slijedećoj formi:

$$S(x,y)=\sum_{i=0}^{m}\sum_{j=0}^{n} y_{i,j}\cdot\psi_i(x)\cdot\psi_j(y)$$

gdje se za osnovnu funkciju koristi funkcija

$$\psi_i(x)=\begin{cases}0,\ za\ x\in[x_{i-1},\ x_{i+1}]\\ (x\text{-}x_{i-1})/(x_i-x_{i-1}),\ za\ x\in[x_{i-1},\ x_i]\\ (x_{i+1}\text{-}x)/(x_{i+1}-x_i),\ za\ x\in[x_i,\ x_{i+1}]\end{cases}$$

pri čemu je :

$$\psi_i(x_j)=\begin{cases}0,\ j\neq 1\\ 1,\ j=1\end{cases}$$

Algoritam testirati na:

a) jednoj odabranoj analitičkoj površini (elipsoid, rotacioni paraboloid,..)
b) jednom tipičnom dijelu brodske forme sa nekom drugom metodom (NURBS)

Programi su pisani u **Turbo C** kompajleru i izvedeni su na računalu **486/80/8 s DOS** operativnim sustavom.
Program **MATEMATICA 3.0** korišten je za rješavanje sustava jednadžbi (datoteke s ekstenzijom *.nb) i radi pod operativnim sustavom Windows 95.

1. Testiranje spline funkcije preko rotacionog paraboloida

U prvom dijelu zadatka algoritam za računanje vrijednosti zadane spline funkcije testiran je usporedbom s vrijednostima z koordinata rotacionog paraboloida s tjemenom u ishodištu koordinatnog sustava, zapisanog u formi:

$$Z = x^2 + y^2 \qquad (1)$$

Zbog simetričnosti paraboloida s obzirom na koordinatne osi za mrežu točaka (x_i, y_j), i = 0,1,…,m ; j = 0,1,…,n , uzeto je m = 50 i n = 50 (nije se provjeravalo područje s negativnim vrijednostima x i y koordinata mreže).

Kod pokretanja programa potrebno je ubaciti koordinate točke x_i i y_j (0 < x < 50 i 0 < y < 50), a program ispisuje rješenje za funkciju (1) i za zadanu dvodimenzionalnu spline funkciju, kao i razliku rješenja.

Koeficijenti $y_{i,j}$ u zadanoj spline funkciji su u ovom slučaju jednaki z koordinati rotacionog paraboloida, pa ova dvodimenzionalna spline funckija interpolira paraboloid u svim točkama x_i i y_j, pa su odstupanja spline-a od paraboloida za cjelobrojne vrijednosti x_i i y_j (od 0 do 50) jednaka nuli.

Odstupanja rješenja po spline funkciji se kreću između nule (za točke x i y na mreži) do 0.5 za vrijednosti u sredini segmenta x, x+1 i y,y+1.

/* PROGRAM 1 - parabola.exe - testiranje 2d spline funkcije preko rotacionog paraboloida */

```
#include <stdio.h>
#include <conio.h>

#define n 50  /*definiranje domene x koordinate*/
#define m 50 /*definiranje domene y koordinate*/
float f[n],g[m];          /* f[n], g[m] - bazne funkcije */

main()
{
        int i,j;            /* pomoćne varijable */
        float xk,yk,s;  /*xk=x koordinata, yk=y koordinata, s=rješenje po spline funkciji*/
        float zk,d;      /*zk=z koor. rotacionog paraboloida, d=razlika rješenja */
        float x[n],y[m];
        float z[n][m];

        xk=0;
        yk=0;
        s=0;

        clrscr();

/* Unos podataka */
        printf ("\n Unesite x koordinatu (broj između 0 i 50) xk=");
        scanf  ("%f",&xk);
        printf ("\n Unesite y koordinatu (broj između 0 i 50) yk=");
        scanf  ("%f",&yk);

/* Pridruživanje vrijednosti x i y čvorovima mreže */
        for (i=0;i<n;i++)
                x[i]=i;
        for (j=0;j<m;j++)
                y[j]=j;
```

```
/* Vrijednosti z koordinata parova (x,y) = koeficijenti spline funkcije*/
        for (i=0;i<n;i++)
                for (j=0;j<m;j++)
                        z[i][j]=x[i]*x[i]+y[j]*y[j];

/* PRORAČUN BAZNIH FUNKCIJA */
        for (i=1;i<50;i++)
                for (j=1;j<50;j++)
                {
                        if (xk > x[i+1] || xk < x[i-1])
                                f[i]=0;
                        else if (xk < x[i] && xk > x[i-1])
                                f[i]=(xk-x[i-1])/(x[i]-x[i-1]);
                        else if (xk > x[i] && xk < x[i+1])
                                f[i]=(x[i+1]-xk)/(x[i+1]-x[i]);
                        else if (xk==x[i])
                                f[i]=1;
                        if (yk > y[j+1] || yk < y[j-1])
                                g[j]=0;
                        else if (yk < y[j] && yk > y[j-1])
                                g[j]=(yk-y[j-1])/(y[j]-y[j-1]);
                        else if (yk > y[j] && yk < y[j+1])
                                g[j]=(y[j+1]-yk)/(y[j+1]-y[j]);
                        else if (yk==y[j])
                                g[j]=1;
                        s=s+z[i][j]*f[i]*g[j];
                }
/* Ispis rezultata */
        printf ("\n \n z koordinata po spline funkciji: S(x=%.4f, y=%.4f)=%.4f",xk,yk,s);
                zk=xk*xk+yk*yk;
        printf ("\n \n Egzaktno rješenje (rot.paraboloid): z=%4.4f",zk);
                d=s-zk;
        printf ("\n \n Odstupanje rješenja spline funkcije: Greška =%f",d);
getch();
}
```

2. Opis dijela forme broda dvodimenzionalnom spline funkcijom

Za potrebe ovog dijela zadatka izabran je ribarski brod projektiran u brodogradilištu "Greben" , Vela Luka , glavnih dimenzija (priložen je plan rebara broda):

Loa = 23,19m
Lpp = 21,04m
B = 6,86m
H = 3,70m

Dio forme broda koji je opisan nalazi se na pramcu između vodnih linija WL2 i WL8, te između rebara R5 i R9. Razmak vodnih linija je 0.3m, a razmak rebara je 2.1m. Vodnih linija ima 4 (uzima se svaka druga, tj. WL2,4,6 i 8), a rebara ima 5 (gleda se svako rebro 5,6,7,8 i 9). Presjecišta rebara i vodnih linija daju niz točaka koje čine mrežu od 20 točaka u ravnini x-z. Rješanja koja se dobijaju su vrijednosti y , tj. poluširine rebara.

Prilikom izvršavanja programa potrebno je unijeti:

- x koordinatu u metrima i to između 0m (R5) i 8.4m (R9)
- z koordinatu i metrima i to između 0.7m (WL2) i 2.5m (WL8)

Program ispisuje vrijednost poluširine y za unijete točke, te traži unos odgovarajuće poluširine s nacrta linija i ispisuje razliku tih dviju vrijednosti.

/* PROGRAM 2 - spline2d.exe - opis forme broda 2d spline funkcijom */

```
#include <stdio.h>
#include <conio.h>

#define n 7    /*definiranje broja x koordinata mreže*/
#define m 6   /*definiranje broja z koordinata mreže*/

float f[n],g[m];          /* f[n], g[m] - bazne funkcije */

main()
{
/* Deklaracija varijabli */

        int i,j;  /* pomoćne varijable */
        float xk,yk,s;  /* xk=x koordinata, yk=z koor., s=rješenje po spline funkcijii */
        float zk,d;      /* zk=y koor. očitana sa nacrta, d = razlika rješenja */

        float x[n]={0,0,2.1,4.2,6.3,8.4,10.5};        /* polje x koordinata = rebra broda */
                        /* NAPOMENA!!  linije broda */
                        /* se gledaju od druge 0 u polju do 8.4 */
                        /* Prva vrijednost 0 i vrijednost 10.5 ne spadaju u područje */
                        /* definiranih linija broda već su potrebne */
                        /* kao potrebni preduvjeti kod programiranja*/
                        /* for petlje za proračun baznih funkcija. */
```

```
        float y[m]={0,0.7,1.3,1.9,2.5,3.1}; /* polje z koordinata= WL broda */
                        /* NAPOMENA! kao gore */

        /* polje z[ ][ ] = vrijednost y koordinata za gore navedene parove (x,z) */
        float z[n][m]={{0,0,0,0,0,0},      /* NAPOMENA! Niz {0,0,0,0,0,0} kao */
                {0,0.89,2.36,3.16,3.385,3.42},  /* i svaka prva 0 u svakom */
                {0,0.84,2.25,2.99,3.245,3.34},  /* sljedećem retku definicije */
                {0,0.65,1.72,2.435,2.81,3.04},  /* polja z[ ][ ] je napisana */
                {0,0.345,1.065,1.63,2.055,2.33},/* zbog gornjih napomena, */
                {0,0.15,0.45,0.75,1.045,1.39}}; /* i nema utjecaja na točnost */
                                                /* rezultata, već postoji samo */
                                                /* zbog mogućnosti programiranja */
                                                /* for petlje baznih funkcija */
        clrscr();

/* Sljedeća for petlja daje ispis vrijednosti y za svaki par koordinata (x,z)*/
/* Potrebno je izbrisati oznake komentara */

/*      for (i=0;i<n;i++)
                for (j=0;j<m;j++)
                {       getch();
                          printf ("\n y[x=%2.3f, z=%2.3f]=%f",x[i],y[j],z[i][j]);
                }*/

        s=0; /* Inicijalizacija vrijednosti bazne funkcije */
        clrscr();

/* Unos koordinata x i z , redom , za koje se računa vrijednost y */

        printf ("\n Unesite x (između 0m (Rebro 5) i 8.4m (Rebro 9) ) xk=");
        scanf  ("%f",&xk);
        printf ("\n Unesite y (između 0.7m (WL 2) i 2.5m (WL 8) yk=");
        scanf  ("%f",&yk);
```

```
/* Sljedeće for petlje -u komentarima -daju mogućnost unosa vrijednosti x, z i y jednu
po jednu s ekrana. U tom slučaju je potrebno obrisati vrijednosti polja unutar{}
   za x[ ], y[ ] i z[ ][ ] kod deklaracije varijabli i obrisati komentare */

/*      for (i=0;i<n;i++)*/ /* Unos vrijednosti x i z čvorova mreže
        {       printf ("\n upišite koordinatu x%d =",i);
                scanf ("%f",x[i]);
        }
        for (j=0;j<m;j++)
        {       printf ("\n upišite koordinatu z%d = ",j);
                scanf ("%f",y[j]);
        }*/

/*      for (i=0;i<n;i++)*/ /*vrijednosti y koordinata parova (x,z) = koeficijenti*/
/*              for (j=0;j<m;j++)
                {       printf ("\n upišite y koordinatu za (x=%2.2f, z=%2.2f )=",x[i],y[j]);
                        scanf ("%f",z[i][j]);
                }*/

/* PRORAČUN BAZNIH FUNKCIJA */
        for (i=1;i<6;i++)
                for (j=1;j<5;j++)
                {
                        if (xk > x[i+1] || xk < x[i-1])
                                f[i]=0;
                        else if (xk < x[i] && xk > x[i-1])
                                f[i]=(xk-x[i-1])/(x[i]-x[i-1]);
                        else if (xk > x[i] && xk < x[i+1])
                                f[i]=(x[i+1]-xk)/(x[i+1]-x[i]);
                        else if (xk==x[i])
                                f[i]=1;

                        if (yk > y[j+1] || yk < y[j-1])
                                g[j]=0;
                        else if (yk < y[j] && yk > y[j-1])
```

```
                    g[j]=(yk-y[j-1])/(y[j]-y[j-1]);
                else if (yk > y[j] && yk < y[j+1])
                    g[j]=(y[j+1]-yk)/(y[j+1]-y[j]);
                else if (yk==y[j])
                    g[j]=1;
                s=s+z[i][j]*f[i]*g[j];
            }

        /* Ispis vrijednosti y proračunatog po spline fji */
        printf ("\n \n y koordinata po spline funkciji: S(x=%.4f,z=%.4f)=%.4f",xk,yk,s);

        /* Unos koordinate y s nacrta (za unesene vrijednosti x i z) */
        printf ("\n \n Upišite y koordninatu s nacrta za  (x=%2.2f, z=%2.2f)=",xk,yk);
        scanf ("%f",&zk);

        /* Računanje i ispis greške */
        d=s-zk;
        printf ("\n \n Odstupanje rješenja spline funkcije = %3.1f%",d);

getch();
}
```

3. Opis dijela forme broda NURBS plohom

Analitička formulacija NURBS plohe je:

$$Q(u,v) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} W_{i,j} \; B_{i,j} \; N_i^k(u) \; M_j^l(v)}{\sum_{i=1}^{m} \sum_{j=1}^{n} W_{i,j} \; N_i^k(u) \; M_j^l(v)} \qquad (2)$$

gdje su:

- $u \in [u_{k-1}, u_{m+1}]$, $v \in [v_{l-1}, v_{n+1}]$ parametarske varijable
- $B_{i,j}$ kontrolne točke 3D mreže
- $W_{i,j}$ težine kontrolnih točaka
- $N_i^k(u)$, $M_j^l(v)$ bazne funkcije reda k, odnosno l

Vrijednosti težina kontrolnih točaka su rezultat racionalne definicije plohe, tj. kao omjera dvaju polinoma. Svrha težina je da omoguće točne opise koničnih oblika i da daju dodatnu kontrolnu nad oblikom plohe. Inicijalno su vrijednosti težina jednake 1.0, što znači da sve točke imaju jednak utjecaj na izgled plohe. Vrijednosti težina različite od 1.0 ne moraju nužno dati željene promjene na izgledu plohe, te u praksi promjene težina mogu uzrokovati probleme kod dobivanja glatkih krivulja. Stoga se u ovom slučaju uzima vrijednost težina jednaka 1.0 pa NURBS ploha postaje NUBS ploha definirana relacijom:

$$Q(u,v) = \sum_{i=1}^{m} \sum_{j=1}^{n} B_{i,j} \; N_i^k(u) \; M_j^l(v) \qquad (3)$$

Bazne funkcije se računaju kako slijedi:

$$N_{i,1}(u) = \begin{cases} 1 & za\ x_i \le u \le x_{i+1} \\ 0 & za\ ostale\ vrijednosti \end{cases}$$

$$N_{i,k}(u) = \frac{u - u_i}{u_{i+k-1} - u_i} \cdot N_{i,k-1}(u) + \frac{u_{i+k} - u}{u_{i+k} - u_{i+1}} \cdot N_{i+k,k-1}(u)$$

$$M_{j,1}(v) = \begin{cases} 1 & za\ y_j \le v \le y_{j+1} \\ 0 & za\ ostale\ vrijednosti \end{cases}$$

$$M_{j,l}(v) = \frac{v - v_j}{v_{j+l-1} - v_j} \cdot M_{j,l-1}(v) + \frac{v_{j+l} - v}{v_{j+l} - v_{j+1}} \cdot M_{j+l,l-1}(v)$$

- x_i i y_j su elementi vektora čvorova **[X] i [Y];**
- k i l su redovi baznih funkcija N i M u smjeru u i v parametra; u ovom slučaju je uzeto $k = l = 3$;
- n i m su brojevi kontrolnih točaka $B_{i,j}$ u smjeru parametara u i v
- $B_{i,j}$ su vrhovi kontrolnog poligona (kontrolne točke)

Za poznati skup točaka kroz koje treba interpolirati (ili aproksimirati) NURB plohu točke $B_{i,j}$ su nepoznanice.

PROGRAM 3 ispisuje rješenja za matricu elemenata N*M potrebnu za izračunavanje nepoznate matrice $B_{i,j}$ točaka.

Definicija vektora čvorova, poznatih točaka na površini kao i broj parametra u i v nalazi se unutar komentara u samom ***PROGRAMU 3***.
Produkti baznih funkcija N*M (matrica sa 320 elemenata) ispisani startanjem ***PROGRAMA 3*** (nubs.exe) uneseni su u program MATEMATICA 3.0 (*basfinal.nb*) i izračunata je nepoznata matrica kontrolnih točaka (matrica sadrži 3*16 elemenata), tj. x, y i z koordinate za 16 kontrolnih točaka. Ta matraica može se pogledati u datoteci Proračun_Bij.nb programom MATEMATICA 3.0).

Testiranje rezultata NUBS plohe izvodi se pomoću ***PROGRAMA 4*** (test.exe). Potrebno je unijeti podatke za po jedan parametar u i v (u je zapravo x koordinata u metrima, a v je z koordinata u metrima unutar ranije definiranog područja). Program kao rezultat ispisuje 16 produkata N i M baznih funkcija za unesene parametre u i v, i množenjem te matrice s prethodno izračunatom matricom točaka $B_{i,j}$ dobije se vrijednost poluširine y za unesene parametre. Priloženo je nekoliko testnih datoteka (test*.nb) napravljenih programom MATEMATICA 3.0.

U programu MATEMATICA 3.0 otvori se *test.nb* (mathematica notebook file) datoteka priložena na disketi, gdje je samo potrebno na mjestu prve matrice unijeti produkte N*M dobivene programom test.exe (i to redom N1*M1, N1*M2, N1*M3, …, N4*M3, N4*M4).Kao rješenje dobije se vrijednost NUBS plohe za unesene u i v , a to je upravo poluširina broda y.

Na kraju ovog rada mogu se usporediti testni rezultati za nekoliko točaka plohe.

/* PROGRAM 3 - Nubs.exe Računa produkate N*M baznih funkcija 3. reda (k=3) */

```
#include <stdio.h>

float knot[20]; /* knot vector može imati 20 čvorova */
float z[8]={2,2,2,6,8,8,10}; /* knot vector po z koor. osi=razmak po dvije vodne linije*/
float x[8]={3,3,3,6,7,7,9,10}; /* knot vector po x koordinatnoj osi=razmak rebara*/
float u[6]={3,4,5,6,7}; /* 5 parametara u1 do u5 za x os, tj. r=5 */
float w[5]={2,4,6,8}; /* 4 parametra w1 do w4 za z os, tj. s=4 */
float basisfja(float,int,int); /* prototip bazne funkcije */

main()
{
        float u1,a; /*u1=parametar u, a=vrijednost bazne fje*/
        int k1,i1; /*k1=red bazne funkcije k, i1=broj kontrolnih točaka Bi*/
        int b,c,j,l; /*lokalne varijable za FOR petlje i pomoćne varijable programa */
        float N[10][10],M[10][10]; /*polja za vrijednosti baznih funkcija N i M*/
                                   /*prvo polje je podatak o kontrolnoj točci*/
                                   /*a drugo polje je parametar u, tj. w*/
                                   /*tj. N[1][3] znači da bazna funkcijaja pripada*/
                                   /*kontrolnoj točki B1 - polje [1], i*/
                                   /*izračunata je za parametar u3-polje[3]*/
        float produktnm; /* produkt N(u)*M(w) */
```

```
        clrscr();
/* Inicijalizacija parametara i varijabli*/ /*Izbrisati komentare */

        /* Unos podataka, red bazne fje i broj kontrolnih točaka*/
        /* printf("\n Unesite red bazne funkcije k=");
        scanf ("%d",&k1);*/
   k1=3; /* red funkcije je fiksan =3*/

        /*printf ("\n Unesite broj kontrolnih točaka B(i)=");
        scanf ("%d",&i1);*/
   i1=4; /* broj kontrolnih točaka za ovaj zadatak je fiksno 4*/

        /*printf ("\n Unesite parametar u=");
        scanf ("%f",&u1);*/
        i1=i1-1; /* zato jer polja u c compileru počinju od 0*/

for (c=0;c<=8;c++) /*definiranje knot vectora za proračun baznih funkcija*/
        knot[c]=x[c]; /*za točke na osi x - rebra broda*/

        for (c=0;c<=4;c++) /* petlja za parametre u - od u1 do u5 */
        {       for (b = 0; b <= i1; b++) /*petlja za kontr. točke B1 do B4*/
                  {
                  a=basisfja(u[c],b,k1); /* poziv potprograma za računanje bazne fje*/
                  N[b][c]=a;
                  /*printf ("\n N %d,%d [u%d=%f] = %f",b+1,k1,c+1,u[c],a);*/
                  }
/*      printf ("\n \n");
        getch();*/
        }

for (c=0;c<=20;c++)
        knot[c]=z[c];

        for (c=0;c<=3;c++) /* petlja za parametre w - od w1 do w5 */
        {       for (b = 0; b <= i1; b++) /*petlja za kontr. točke B1 do B4*/
```

```
                {
                a=basisfja(w[c],b,k1); /*poziv potprog. za računanje bazne fje*/
                M[b][c]=a;
                /*printf ("\n M %d,%d [w%d = %f] = %f",b+1,k1,c+1,w[c],a);*/
                }
/*      printf ("\n \n");
        getch();*/
        }

for (b=0;b<=4;b++) /* parametar u - u1 do u5 */
        for (c=0;c<=3;c++) /* parametar w - w1 do w4 */
                for (l=0;l<=3;l++)
                for (j=0;j<=3;j++)
                { produktnm=N[l][b]*M[j][c];
                printf ("\n N%d [u%d]*M%d [w%d]=%f",l+1,b+1,j+1,c+1,produktnm);
                getch();
                        }
        printf ("\n");
getch ();
}

/* PRORAČUN BAZNIH FUNKCIJA */
        float basisfja(float t,int i, int k)      /* donose se parametar t, */
        {                                         /* broj kont.točaka i, */
                                                  /* red bazne fje k */
        float n; /* n je vrijednost bazne fje koja se vraća u program */
        /* printf("\n %f %f %f", x[i],x[i+k-1],x[i+k]); */

        if (k==1)
                {
                if (knot[i]<=t && t<knot[i+1])
                        return(1);
                else
                        return(0);
                }
```

```
        if (knot[i+k-1]-knot[i]==0)
                n=((knot[i+k]-t)/(knot[i+k]-knot[i+1])*basisfja(t,i+1,k-1));
        else if (knot[i+k-1]-knot[i]==0 && knot[i+k]-knot[i+1]==0)
                n=0;
        else if (knot[i+k]-knot[i+1]==0)
                n=((t-knot[i])/(knot[i+k-1]-knot[i])*basisfja(t,i,k-1));
        else
                n=((t-knot[i])/(knot[i+k-1]-knot[i])*basisfja(t,i,k-1))+((knot[i+k]-t)/
                 /(knot[i+k]-knot[i+1])*basisfja(t,i+1,k-1));
                return (n);
        }
```

/* PROGRAM 4 test.exe - Računa produkte N*M baznih funkcija 3. reda (k=3) */

/* i to za po jedan parametar u i v koji se unose po pokretanju programa */

```
#include <stdio.h>

float knot[20];
float z[8]={2,2,2,6,8,8,10};
float x[8]={3,3,3,6,7,7,9,10};

float basisfja(float,int,int);

main()
{
        float u1,w1,a; /*u1=parametar u, w1=parametar v, a=vrijednost bazne fje*/
        int k1,i1; /*k1=red bazne funkcije k, i1=broj kontrolnih tožaka Bi*/
        int b,c,j,l;
        float N[10][10],M[10][10];
        float produktnm;

        clrscr();
```

```
        /* Unos podataka, red bazne fje i broj kontrolnih tocaka*/
        /* printf("\n Unesite red bazne funkcije k=");
        scanf ("%d",&k1);*/
    k1=3; /* red funkcije je fiksan =3*/

        /*printf ("\n Unesite broj kontrolnih tożaka B(i)=");
        scanf ("%d",&i1);*/
    i1=4; /* broj kontrolnih točaka za ovaj zadatak je fiksno 4*/

        printf ("\n Unesite parametar u=");
        scanf ("%f",&u1);
        printf ("\n Unesite parametar v=");
        scanf ("%f",&w1);

        i1=i1-1;

for (c=0;c<=8;c++)
        knot[c]=x[c];

        for (b = 0; b <= i1; b++) /*petlja za kontr. točke B1 do B4*/
                {
                a=basisfja(u1,b,k1);
                N[b][u1]=a;
                /*printf ("\n N %d,%d [u=%f] = %f",b+1,k1,u1,a);*/
                }
/*      printf ("\n \n");
        getch();*/

for (c=0;c<=20;c++)
        knot[c]=z[c];

        for (b = 0; b <= i1; b++) /*petlja za kontr. točke B1 do B4*/
                {
                a=basisfja(w1,b,k1);
                M[b][w1]=a;
```

```
                /*printf ("\n M %d,%d [w = %f] = %f",b+1,k1,w1,a);*/
                }
/*      printf ("\n \n");
        getch();*/

        for (l=0;l<=3;l++)
                for (j=0;j<=3;j++)
                {       produktnm=N[l][u1]*M[j][w1];
                        printf("\n N%d[u=%f]*M%d[w=%f]=%f",l+1,u1,j+1,w1,produktnm);
                        getch();
                }

        printf ("\n");
/*      getch ();*/
}

        /* PRORAČUN BAZNIH FUNKCIJA */

        float basisfja(float t,int i, int k)      /* donose se parametar t, */
        {                                         /* broj kont.točaka i, */
                                                  /* red bazne fje k */

        float n; /* n je vrijednost bazne fje koja se vraća u program */

        /* printf("\n %f %f %f", x[i],x[i+k-1],x[i+k]); */

        if (k==1)
                {
                if (knot[i]<=t && t<knot[i+1])
                        return(1);
                else
                        return(0);
                }
        if (knot[i+k-1]-knot[i]==0)
                n=((knot[i+k]-t)/(knot[i+k]-knot[i+1])*basisfja(t,i+1,k-1));
```

```
        else if (knot[i+k-1]-knot[i]==0 && knot[i+k]-knot[i+1]==0)
                n=0;
        else if (knot[i+k]-knot[i+1]==0)
                n=((t-knot[i])/(knot[i+k-1]-knot[i])*basisfja(t,i,k-1));
        else
                n=((t-knot[i])/(knot[i+k-1]-knot[i])*basisfja(t,i,k-1))+((knot[i+k]-
t)/(knot[i+k]-knot[i+1])*basisfja(t,i+1,k-1));
                return (n);
        }
```

4. Usporedba rezultata

Rezultati za zadanu spline funkciju dobiveni su programom spline2d.exe, za NUBS funkciju programima nubs.exe, test.exe i programom Matematica 3.0.
Stvarna očitanja izmjerena su sa originalnog nacrta linija broda u M 1:20.

	Spline 2d funkcija	NUBS ploha	Očitanja s nacrta linija
x = 1.05 m z = 0.7 m	0,865	0,8797	0,91
x = 4.2 m z = 0.7 m	0,65	0,6536	0,65
x = 5.25 m z = 0.7 m	0,4975	0,515	0,505
x = 5.25 m z = 2.5 m	2,4325	2,469	2,47

Tablica testnih rezultata

Iz tablice se vidi veća točnost rezultata dobivenih NUBS funkcijom nego dvodimenzionalnom spline funkcijom.

Detalj koji je posebno zanimljiv su vrijednosti dobivene za točku x = 4,2 (R7) i z=0.7 (WL 2), gdje se vidi da je dvodimenzionalna spline funkcija dala točno rješenje y=0.65, što je i bilo za očekivati jer ona interpolira zadani skup točaka, do je NUBS funkcija dala približno rješenje y = 0.6536 , iako i ova funkcija interpolira plohu u zadanim točkama.
Razlog ovoj pogrešci je u tome što matrica N*M nije kvadratna (16*20 za 4*5 zadanih točaka plohe) pa se rješenje matrice kontrolnih točaka B_{ij} može dobiti samo približno.
Bolji rezultati bi se mogli postići višestrukom iteracijom (do željene točnosti), vrijednosti parametra u i v za proračun produkata N*M.

PRILOG 1

Nacrt rebara ribarskog broda dimenzija:

L_{oa} = 23,19 [m]

L_{pp} = 21,04 [m]

B = 6,86 [m]

H = 3,70 [m]