UNIVERZITA KOMENSKÉHO V BRATISLAVE

FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

KATEDRA ALGEBRY, GEOMETRIE A DIDAKTIKY MATEMATIKY

Modelovanie zväzku implicitne definovaných plôch so zameraním na rezy dát

D I P L O M O V Á P R Á C A

Bratislava 2009 **Tibor Chilý**

UNIVERZITA KOMENSKÉHO V BRATISLAVE

FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

KATEDRA ALGEBRY, GEOMETRIE A DIDAKTIKY MATEMATIKY

Študijný odbor 9.1.1 Matematika

Modelovanie zväzku implicitne definovaných plôch so zameraním na rezy dát

Diplomová práca

Autor:
Tibor Chilý

Vedúci diplomovej práce:
RNDr. Pavel Chalmovianský, PhD.

Čestne vyhlasujem, že uvedenú diplomovú prácu som vypracoval samostatne len s použitím literatúry, ktorá je uvedená v zozname

V Bratislave, dňa 24. apríla 2009

.....
Tibor Chilý

Ďakujem vedúcemu diplomovej práce, RNDr. Pavlovi Chalmovianskému, PhD. za venovaný čas, cenné rady, odborné vedenie, pripomienky pri tvorbe tejto práce. Poďakovanie patrí aj firme SIEMENS a jej zástupcom, pánovi Sedlačkovi, ktorí prispeli inšpiráciou pri zadaní témy.

Abstrakt

Cieľom diplomovej práce je preskúmať rôzne spôsoby vytvárania špeciálnych rezových plôch nazývaných zakrivené rezové plochy (Curved clipped ranges). Zároveň je cieľom aj navrhnúť algoritmy na vytváranie, editovanie a transformovanie, používateľné pri práci s týmito plochami, a výsledky implementovať v ukážkovej aplikácii. Diplomová práca vychádza z potreby zobrazovania špecifických rezov v medicínskom zobrazovaní. Samotný algoritmus využíva na vytvorenie rezových plôch implicitne definované objekty. Prínosom práce je aplikácia algoritmu, ktorý umožňuje nelineárnym spôsobom rezať dátový hranol podľa potreby užívateľa, keďže rezové plochy je možné relatívne ľahko vytvárať.

Kľúčové slová

Implicitne definované objekty, zväzok implicitne definovaných plôch, Bézierova krivka, šablónovanie

Obsah

Zoznam obrázkov

1. Predslov

Už od dávnej minulosti vznikala potreba nahliadnuť dovnútra tela človeka neinvazívnym spôsobom, teda bez toho, aby bolo pacienta treba operovať, rezať alebo iným spôsobom zbytočne zasahovať tam, kde to nie je nevyhnutné.
Táto spočiatku nemožná úloha sa s postupom času čoraz viac menila v skutočnosť. S pokrokom vedy a techniky pribúdajú nové spôsoby zobrazovania vnútra ľudského tela, najčastejšie prichádzajú z oblasti fyziky a matematiky, ktorá dáta analyzuje.

(Citáty v tejto časti sú z (1))

Azda najznámejšou metódou je röntgenový snímok.

> *„Boli to neviditeľné elektromagnetické lúče, najprv nazývané X lúčmi, ktoré po prvý raz umožnili pohľad dovnútra tela. Zaznamenali revolúciu pre medicínsku diagnostiku. Röntgenové lúče nazvané po svojom objaviteľovi umožnili medicíne úplne nové perspektívy.“*

Metódy založené na fotografii s vývojom výpočtovej techniky začali byť zastaranými. V inovovane podobe a so zlepšenou efektívnosťou sa ale práve na nich zakladá veľa v súčasnosti stále využívaných metód.

> *„Predtým sa ako zachytávajúci systém používali filmové kazety, dnes sa už väčšinou pracuje s digitálnou technikou a diagnostikuje sa na obrazovke.“*

V súčasnej dobe, keď sa vo väčšine oblastí života presadzujú počítače, ani medicínske zobrazovanie nie je výnimkou. Počítačová grafika a jej metódy ako napríklad modelovanie pomáhajú k lepším možnostiam vizualizácie.
Metódy ako **CT** (počítačová tomografia, computer tomography), **MRI** (zobrazovanie pomocou magnetickej rezonancie, magnetic resonance imaging), **PET** (pozitrónová emisná tomografia, positron emission tomography) alebo **ultrazvuk** patria medzi najznámejšie. Ich hlavné výstupné dáta si môžeme predstaviť ako dátový hranol, zložený z po sebe nasledujúcich snímkov.

„Jednotlivé snímky vznikajú otočným a posuvným pohybom rúry a detektoru, porovnateľné s pohybom skrutkového závitu.“

Na tomto stupni vývoja sa možnosti prístrojov na istú dobu pozastavili. V súčinnosti s počítačmi a špeciálnym softvérom možno tieto obmedzenia odstrániť.

„Moderná rádiológia nemôže existovať bez zložitých počítačových systémov. Tie vyrobia z dát, ktoré sú vlastne produktom znázorňujúcich prístrojov ostré snímky, alebo čoraz častejšie kompletné trojrozmerné simulácie.“

Najnovšou požiadavkou v oblasti medicínskeho zobrazovania ako aplikácie počítačovej grafiky je, aby na obrazovke, v zadanom dátovom hranole bolo možné vyselektovať a následne zobraziť zovšeobecnené valcové plochy. Tieto majú vytvoriť rezové plochy na nasnímanom dátovom hranole, ktoré sú za pomoci snímacích prístrojov nedosiahnuteľné.
Za pomoci „myši“ a „kliknutých bodov“ tak používateľ môže vytvoriť a editovať krivky, z ktorých sa výpočtom vytvorí množina požadovaných rezov.
Táto diplomová práca sa zaoberá vytvorením a implementáciou algoritmu vytvárajúceho takéto plochy.

2. Definície

Definície v časti 2.1, 2.2 a 2.3 sú prevzaté z (2 s. 13-15).

2.1. Body a vektory

Prvky n-rozmerného Euklidovského priestoru $\mathbf{E}^n$ značíme veľkými, tlačenými písmenami abecedy a nazývame *body*, napr. $A, B, \ldots$. Prvky n-rozmerného lineárneho priestoru $\mathbf{R}^n$ nazývame *vektory* a označujeme, malými tučnými písmenami napr. $\mathbf{a}, \mathbf{b}, \mathbf{c}, \ldots$

Body aj vektory reprezentujeme n-ticou reálnych čísel, sú však odlišné. Pre každé dva body A, B existuje práve jeden vektor $\mathbf{v}$ tak, že platí:

$$\mathbf{v} = B - A; \qquad A, B \in \mathbf{E}^n; \qquad \mathbf{v} \in \mathbf{R}^n. \tag{1.1}$$

Pre každý vektor $\mathbf{v}$ existuje nekonečne veľa dvojíc bodov A, B takých, že platí (1.1). Priradenie $a + \mathbf{v}$, ktoré každému bodu $A \in \mathbf{E}^n$ priradí bod B, nazývame posunutie.

Vektor sa dá chápať ako množina dvojíc bodov, ktoré ho vytvárajú t.j.:

$$D - C = \mathbf{v} \equiv \{(A; B) \in \mathbf{E}^n \times \mathbf{E}^n : B - A = D - C\} \tag{1.2}$$

2.2. Barycentrická kombinácia

Simplex v $\mathbf{E}^n$ je $(n+1)$-tica bodov neležiacich v lineárnej variete dimenzie $(n-1)$.

Nech $A_0, \cdots, A_n = \Sigma$ je ľubovoľný simplex v $\mathbf{E}^n$, každý bod $X \in \mathbf{E}^n$ sa potom dá jednoznačne vyjadriť ako $X = \alpha_0 A_0 + \cdots + \alpha_n A_n$, kde $\alpha_0 + \cdots + \alpha_n = 1$. Potom $(n+1)$-ticu $(\alpha_0, \cdots, \alpha_n) \in \mathbf{R}^{n+1}$ nazývame *barycentrickými súradnicami* bodu X vzhľadom na Σ.

Teda pre platí:

$$B = B_0 + \sum_{j=0}^{n} \alpha_j (A_j - A_0), \quad kde\ \alpha_0 + \cdots + \alpha_n = 1\ , \tag{2.1}$$

$$\mathbf{v} = \sum_{j=0}^{n} \alpha_j (A_j - A_0), \qquad kde\ \alpha_0 + \cdots + \alpha_n = 0\ . \tag{2.2}$$

Sumy vyjadrujú korektný zápis súčtu vektorov a bodu (2.1), alebo len vektorov (2.2). Špeciálnym prípadom barycentrických súradníc (ak pre všetky koeficienty α_i súčtu platí $\alpha_i \geq 0$) je konvexná kombinácia bodov.

2.3. Homogénne afinné súradnice

Majme afinné súradnice bodu $P = (x_1, \cdots x_n) \in \mathbf{E}^n$. Potom homogénne afinné súradnice bodu dostávame rozšírením jeho klasických súradníc o prvok w nazývaný váha nasledovne:

$$(x_1, \cdots, x_n) \rightarrow (x_1 : \ldots : \mathrm{x}_n : w); \quad w \neq 0 \tag{2.3}$$

Ak vektor chápeme ako rozdiel bodov, potom pre vektory platí:

$$(x_1, \cdots, x_n) \rightarrow (x_1 : \ldots : \mathrm{x}_n : 0), \quad ak\ x\ je\ vektor \tag{2.4}$$

$X = (x_1 : \ldots : \mathrm{x}_n : w);\ w \neq 0$ sú homogénne afinné súradnice bodu $X \in \mathbf{E}^n$ práve vtedy, keď sú $X = (\frac{x_1}{w}, \cdots, \frac{x_n}{w})$ jeho afinné súradnice.

Definície v nasledujúcej časti sú prevzaté z (3):

2.4. Afinné zobrazenie

Zobrazenie φ ($\varphi : \mathbf{E}^n \rightarrow \mathbf{E}^n$) nazývame afinné, ak pre všetky body $A, B \in \mathbf{E}^n$ a čísla $t \in \mathbf{R}$ platí:

$$\varphi(A + t(B - A)) = \varphi(A) + t[\varphi(B) - \varphi(A)] \tag{3.1}$$

vyjadrime bod $X = (a_0, \ldots, a_d)$, a jeho obraz $\varphi(X)$:

$$X = \sum \alpha_j a_j, \text{ kde} \sum \alpha_j = 1; \quad \varphi(X) = \sum \alpha_j \varphi(a_j) \tag{3.2}$$

Pozn. afinné zobrazenie zachováva barycentrické kombinácie. (3)

2.5. Lineárna interpolácia

Funkciu $\varphi : \langle 0;1 \rangle \to \mathbf{R}^d$ ktorá je daná predpisom $\varphi(t) = (1-t)A + tB,\ A, B \in \mathbf{E}^d, t \in \mathbf{R}$ nazývame lineárna interpolácia bodov A, B.

Pre lineárnu interpoláciu platí:

- $\varphi(0) = A,\ \varphi(1) = B$
- ak $0 \le t \le 1$ potom obrazy $\varphi(t)$ ležia na úsečke AB

Interpolantom bodov $A, B \in \mathbf{E}^d$ pre danú hodnotu parametra $t = t_C$ budeme nazývať bod $C = (1-t_C)A + t_C B$. Analogicky, pri interpolácii kriviek budeme interpolantom kriviek $\mathcal{P}, \mathcal{R}$ pre hodnotu parametra $t = t_C$ rozumieť krivku $\mathcal{S} = (1-t_C)\mathcal{P} + t_C \mathcal{R}$.

Pozn. lineárna interpolácia zachováva barycentrické súradnice.

2.6. Vzdialenosť v priestore

Vzdialenosťou v priestore rozumieme klasickú euklidovskú definíciu vzdialenosti:

$$\rho(A, B) = \sqrt{(a_1 - b_1)^2 + \ldots + (a_d - b_d)^2} \tag{4.1}$$

kde d je 2 alebo 3 a A, B sú body v rovine alebo priestore.

2.7. Krivka

V počítačovej grafike sa krivky v rovine resp. v priestore najčastejšie zadávajú *parametricky* prostredníctvom *bodovej funkcie jednej (číselnej) premennej*, teda pomocou zobrazenia:

$$\varphi : I \to \mathbf{E}^d, \quad \text{kde } d = 2,3. \tag{5.1}$$

I je ľubovoľný interval na číselnej osi $\mathbf{R}$. Vyjadrené súradnicami $P(t) = (x_1(t), \ldots, x_d(t))$, kde $x_1 = x_1(t), \ldots, x_d = x_d(t);\ t \in I$ sú číselné funkcie premennej t. Premenná t je *parametrom* bodu $P(t)$.

Derivácia bodovej funkcie $P(t) = (x_1(t), \ldots, x_d(t))$ vo vnútornom bode $t_0 \in I$ je vektor $P'(t_0) = \left(x_1'(t_0), \ldots, x_d'(t_0)\right)$. V hraničných bodoch musíme uvažovať jednostranné derivácie. Od parametrického určenia krivky sa spravidla vyžaduje splnenie podmienky *hladkosti*: existujú derivácie všetkých rádov (t.j. súradnicové funkcie $x_1(t), \ldots, x_d(t)$ sú hladké) a podmienky *regulárnosti*: $P'(t) \neq \mathbf{0}$ pre všetky $t \in I$.

Nasledovné definície sú prevzaté z (4)

2.8. Bézierova krivka

Bézierova krivka je opísaná rovnicou

$$P(t) = \sum_{i=0}^{n} V_i B_i^n(t);\ t \in \langle 0;1 \rangle \tag{6.1}$$

kde V_i sú riadiace vrcholy a $B_i^n(t)$ sú Bernsteinove polynómy a n je stupeň Bézierovej krivky. Bernsteinove polynómy definujeme vzťahom

$$B_t^n(t) = \binom{n}{i} t^i (1-t)^{n-i};\ \text{pre } i = 0,1,\ldots,n\ , \tag{6.2}$$

pričom koeficienty definujeme nasledovne:

$$\binom{n}{i} = \begin{cases} \dfrac{n!}{(n-i)!\,i!} & 0 \le i \le n \\ 0 & \text{inak} \end{cases}; \quad \binom{0}{0} \overset{def}{=} 1\ .$$

Špeciálnym prípadom je *Bézierova kubická krivka* daná predpisom:

$$P(t) = (1-t)^3 V_0 + 3t(1-t)^2 V_1 + 3t^2(1-t)V_2 + t^3 V_3 \ . \tag{6.3}$$

Nech S je lineárna funkcia ktorej predpis je:

$$S : t \to \frac{t - t_{min}}{t_{max} - t_{min}} \ .$$

S použitím substitúcie S vieme zovšeobecniť definíciu (6.1) na ľubovoľný interval $\langle t_{min}; t_{max} \rangle$, a nazývame *Bézierovou krivkou nad intervalom* $\langle t_{min}; t_{max} \rangle$.

$$P(t) = \sum_{i=0}^{n} V_i B_i^n \left(\frac{t - t_{min}}{t_{max} - t_{min}} \right); \ t \in \langle t_{min}; t_{max} \rangle \ . \tag{6.4}$$

2.9. Derivácie

Na výpočet r-tej derivácie Bézierovej krivky použijeme nasledujúci vzorec:

$$\frac{d^r B(t)}{dt^r} = \frac{n!}{(n-r)!} \sum_{j=0}^{n-r} \Delta^r V_j B_j^{n-r}(t) \ , \tag{7.1}$$

pričom symbolom $\Delta^r V_i$ označujeme nasledujúcu operáciu na riadiacich vrcholoch:

$$\Delta^r V_i = \sum_{j=0}^{r} \binom{r}{j} (-1)^{r-j} V_{i+j} \ . \tag{7.2}$$

Špeciálne prípady pre derivácie v koncových bodoch Bézierovej krivky, t.j. pre hodnoty parametra $t = 0$ derivácia sprava, resp. pre $t = 1$ derivácia zľava

$$\begin{aligned} B^{(r)}(0) &= \frac{n!}{(n-r)!} \Delta^r V_0 \ , \\ B^{(r)}(1) &= \frac{n!}{(n-r)!} \Delta^r V_{n-r} \ . \end{aligned} \tag{7.3}$$

Pre Bézierovu krivku nad intervalom $\langle a;b\rangle$ rôznym od $\langle 0;1\rangle$ pričom $t \to \frac{t-a}{b-a}$ platí:

$$\frac{d^r B(t)}{dt^r} = \frac{1}{(b-a)^r} \cdot \frac{n!}{(n-r)!} \sum_{j=0}^{n-r} \Delta^r V_j B_j^{n-r}(t) \,. \tag{7.4}$$

Označme $\Delta = b - a$ vo vzorci (7.4), dostávame:

$$\frac{d^r B(t)}{dt^r} = \frac{1}{\Delta^r} \cdot \frac{n!}{(n-r)!} \sum_{j=0}^{n-r} \Delta^r V_j B_j^{n-r}(t) \,. \tag{7.5}$$

Definície z nasledujúceho odseku sú prevzaté z (5)

2.10. Po častiach spojitá krivka

Krivka $P(t)$ sa nazýva čiastkovou Bézierovou krivkou na intervale $\langle a;b\rangle$ práve vtedy, keď existuje delenie $a = t_0 < \ldots < t_r = b$, a na každom intervale $I_j = \langle t_j; t_{j+1}\rangle;\ j = \{0,\ldots,r-1\}$ je definovaná intervalová Bézierova krivka ${}^jB(t)$ tak, že platí:

$$P(t) = {}^jB(t), \text{ pre } t \in \langle t_j; t_{j+1}\rangle, \tag{8.1}$$

$$P(t_j) = {}^{j-1}B(t_j) \text{ alebo } \mathrm{P(t}_j) = {}^jB(t_j),\ j = \{1,\ldots,r-1\}, \tag{8.2}$$

$$P(t_0) = {}^0B(t_0) \ a\ P(t_r) = {}^{r-1}B(t_r). \tag{8.3}$$

Hodnoty parametra $t_j, j = \{0,1,\ldots,r\}$ nazývame deliacimi bodmi. Stupňom $P(t)$ rozumieme $n = \max_{st}\{{}^jB(t), j = 0,\ldots,r-1\}$, pre $n = 2,3$ hovoríme o kvadratickej resp. kubickej čiastkovej Bézierovej krivke.

Vzťah (8.1) vyjadruje j-ty segment krivky. Vzťah (8.2) vyjadruje, že dva segmenty ${}^{j-1}B(t_j)$ a ${}^jB(t_j)$ sa stretajú v bode $t = t_j$:

$${}^{j-1}B(t_j) = P(t_j) = {}^jB(t_j)$$

Zápisom $B(t) = B\left[V_0,\ldots,V_n; t \in \langle 0;1\rangle\right]$ teraz označme Bézierovu krivku definovanú pomocou vrcholov $V_0,\ldots,V_n$ na intervale $t \in \langle 0;1\rangle$.

Ak budeme predpokladať že ${}^jB(t) = {}^jB\left[{}^jV_0,\ldots,{}^jV_n; t \in \langle t_j; t_{j+1}\rangle\right]$, $j = 0,\ldots,r-1$, potom je zrejmé, že krivka je spojitá v bode $t = t_j$ práve vtedy, keď:

$$ {}^{j-1}B(t_j) = {}^jB(t_j) \Leftrightarrow {}^{j-1}V_n = {}^jV_0 \ . \tag{8.4} $$

Podmienka pre C^k-spojitosť krivky $P(t)$ v bode $t = t_j$ má teda tvar:

$$ {}^{j-1}B^{(k)}(t_j) = {}^jB^{(k)}(t_j) \Leftrightarrow \frac{1}{\Delta_{j-1}^k}\,{}^{j-1}B^{(k)}(t_j) = \frac{1}{\Delta_j^k}\,{}^jB^{(k)}(t_j)\ , \quad k = 0,1,\ldots r\ , \tag{8.5} $$

kde symbol Δ_j^k vyjadruje číslo ako vo vzorci (7.5).

Ak ďalej podľa (5) označíme symbolom $\Delta^k\left[{}^jV_0\right]$, resp. $\Delta^k\left[{}^{j-1}V_n\right]$ rekurzívne aplikovaný rozdiel $\Delta = V_{i+1} - V_i$ riadiacich vrcholov, ako vo vzorci (7.2), dostávame podmienku pre vrcholy:

$$ \frac{1}{\Delta_{j-1}^k}\,{}^{j-1}B^{(k)}(1) = \frac{1}{\Delta_j^k}\,{}^jB^{(k)}(0) \Leftrightarrow \frac{1}{\Delta_{j-1}^k}\Delta^k\left[{}^{j-1}V_n\right] = \frac{1}{\Delta_j^k}\Delta^k\left[{}^jV_0\right], \quad k = 0,1,\ldots,r. \tag{8.6} $$

Dôsledkom je, že Bézierovu krivku nad intervalom (6.4) možno pri splnení rovností (8.4), (8.5) považovať za C^r spojitú zloženú krivku.

2.11. Euklidovská varieta

Euklidovská varieta je topologický priestor, ktorý je lokálne euklidovský. Presnejšie, n-rozmerná varieta je separabilný topologický priestor, v ktorom každý bod má okolie homeomorfné s podmnožinou $\mathbf{R}^n$. (6)

Definícia implicitne definovaného objektu podľa (6 s. 51)

2.12. Implicitne definovaný objekt

Podmnožina $\mathcal{O} \subset \mathbf{R}^n$ sa nazýva implicitne definovaný objekt ak existuje zobrazenie $f : U \to \mathbf{R}^n$, $\mathcal{O} \subset U$, a podmnožina $V \subset R^k$, $\mathcal{O} = f^{-1}(V)$ tak, že platí:

$$\mathcal{O} = \{p \in U : f(p) \in V\}\ , \tag{9.1}$$

Takáto definícia je vďaka ľubovoľnej množine U veľmi široká, zahŕňa aj objekty, ktoré nie sú varietou. Budeme teda navyše vyžadovať aj splnenie podmienky regulárnosti v zmysle (6 s. 35), t.j. regulárny implicitne definovaný objekt definuje varietu.

Implicitne definovaná varieta sa teda definuje ako množina bodov euklidovského priestoru vyhovujúca rovnosti typu $F(x) = 0$.

$$\mathcal{O} = \{p : F(p) = 0\} \tag{9.2}$$

2.13. Implicitná reprezentácia variet

Na popis variety používame hraničnú reprezentáciu (Boundary representation, B-rep.), ktorá popisuje hranicu objektu. Implicitná reprezentácia vyžaduje zadanie funkcie popisujúcej povrch telesa napr. guľa v priestore $x^2 + y^2 + z^2 - 1 = 0$

Pozn. v hraničnej reprezentácii môžeme použiť tiež polyhedrálnu reprezentáciu alebo parametrickú reprezentáciu na vyjadrenie povrchu telesa (7).

2.14. Izonadplocha

Izonadplochou $\mathcal{P}$ rozumieme body x priestoru $\mathbf{R}^d$, ktoré pri dosadzovaní do predpisu implicitného objektu $F(x)$ nadobúdajú rovnaké hodnoty.

$$\mathcal{P} = \{x \in R^d : F(x) = prah\} \tag{10.1}$$

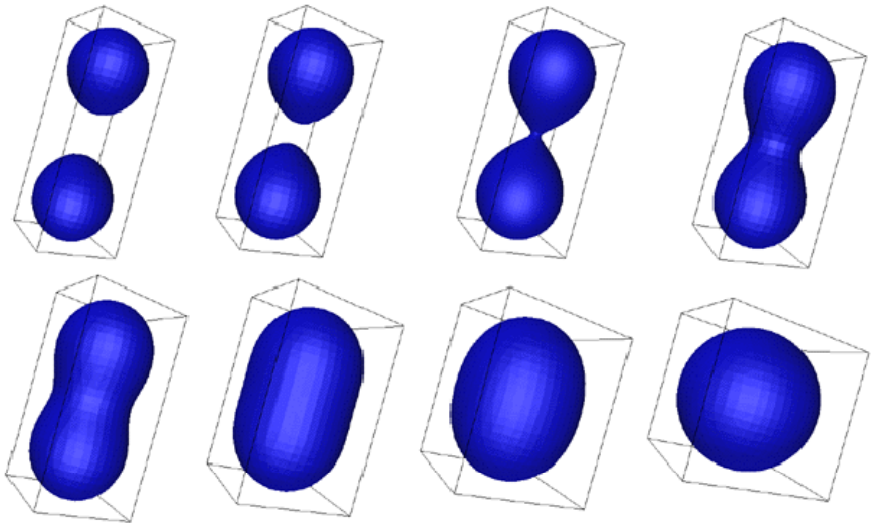
Ak $d = 3$ hovoríme o izoploche.

2.15. Metagule, mäkké objekty, kvapkové molekuly

Modelovanie parametrických objektov v 3D grafike využíva základné primitíva ako úsečky, trojuholníky, štvorce a pod. (kde body tvoria vrcholy, definujú hrany a steny).

K modelovaniu tzv. implicitných povrchov (implicit surfaces), t.j. povrchov kde obrys je definovaný ako izoplocha (isosurface) za pomoci nejakého skalárneho poľa v 3D existujú 3 bežne používané techniky. Tieto techniky sa odlišujú vo funkcii popisujúcej skalárne pole. Funkcia poľa $D(r)$ určuje hodnoty v každom bode priestoru vzhľadom na nejaký bázový primitívny objekt, napr. bod, úsečka a pod.

Obr. 1 (8) je ukážkou modelovania pomocou metaguľových primitívnych objektov, vidíme ako sa dva metaguľové objekty pri vzájomnom priblížení stmelia a vytvoria jeden výsledný metaobjekt.



Obr. 1

2.15.1. Kvapkové molekuly (Blobby molecules)

Za pomoci funkcie pripisovanej Jimovi Blinnovi a modelovanej na základe hustoty elektrónových polí definujeme model kvapkových molekúl (blobby molecules model)

$$D(r) = ae^{-br^2},$$

kde b vyjadruje odchýlku krivky, a a výšku. Graf priebehu funkcie poľa v porovnaní s referenčnou funkciou poľa a funkciami poľa pre metagule a mäkké objekty vidíme na obrázku 2.

2.15.2. Metagule (Meta balls)

Skalárne pole pre metaguľové objekty definujeme ako:

$$D(r) = \begin{cases} a(1-\frac{3r^2}{r^2}) \text{ , } ak \;\; 0 \le r \le \frac{b}{3}, \\ \frac{3a}{2}(1-\frac{r}{b})^2 \text{ , } ak \;\; \frac{b}{3} \le r \le b, \\ 0 \quad\quad \text{ , } ak \quad b \le r, \end{cases}$$

kde b je maximálna vzdialenosť, v ktorej ešte príspevok riadiaceho primitívneho objektu je nenulová, a je škálovací koeficient.

2.15.3. Mäkké objekty (Soft objects)

Vytvorenie mäkkých objektov sa pripisuje bratom Wyvillovým, ich funkcia poľa je definovaná nasledovne:

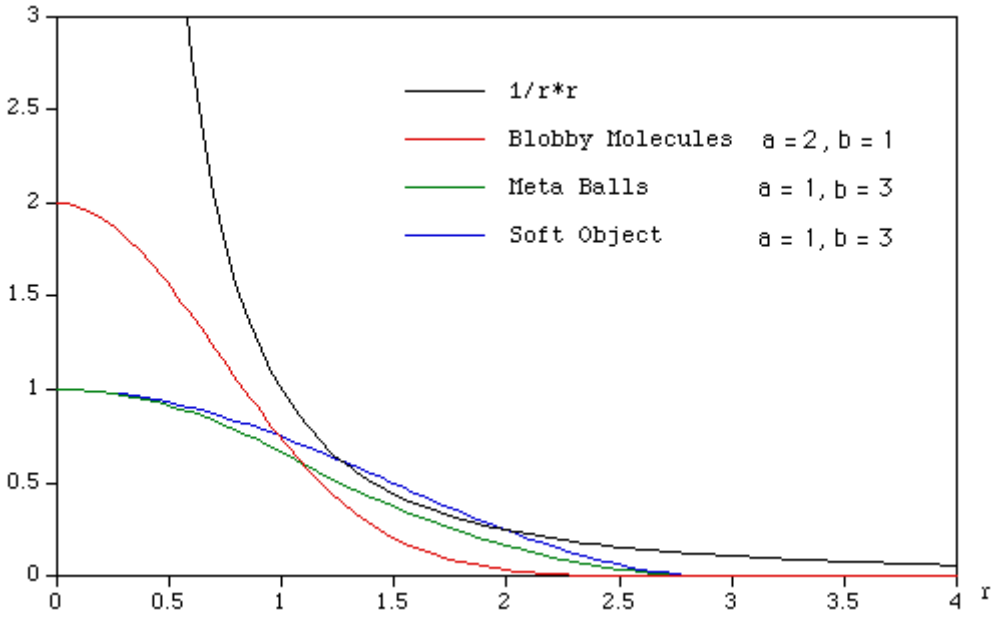
$$D(r) = \begin{cases} a(1-\frac{4r^6}{9b^6}+\frac{17r^4}{9b^4}-\frac{22r^2}{9b^2}) \text{ , } ak\, r \le b \\ 0 \quad\quad\quad\quad\quad\quad\quad\quad \text{ , } ak\, r > b \end{cases},$$

pričom a , b majú rovnaký význam ako v predchádzajúcom vzorci.

Nevýhodou modelu kvapkových molekúl je, že funkcia poľa má príspevky aj v nekonečnej vzdialenosti od riadiaceho primitívneho objektu. t.j. pre výpočet hodnoty poľa v danom

bode treba brať do úvahy príspevky od každého primitívneho objektu, čo je výpočtovo náročné s rastúcim počtom primitív. Tento nedostatok riešia ohraničené funkcie poľa, kde po istej vzdialenosti už má primitívny objekt nulový príspevok (metagule, mäkké objekty). Výpočtovo menej náročná je funkcia Wyvillových bratov, kde sa pri implementácii nemusí počítať odmocnina.

Graf na Obr. 2, prevzatý z (8), zobrazuje porovnanie funkcií poľa pre implicitne definované objekty metagulí, kvapkových molekúl a mäkkých objektov s referenčnou funkciou potenciálového poľa $1/r^2$. Takto definované funkcie zabezpečujú spojitý a hladký povrch implicitne definovaného objektu.

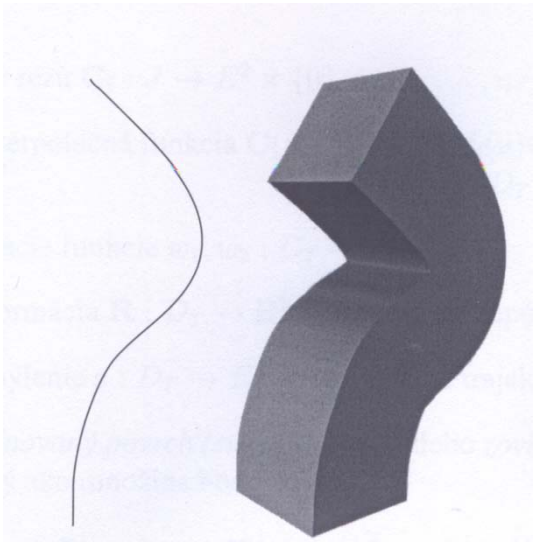


Obr. 2

2.16. Šablónovanie (sweeping)

Šablónovanie je spôsob popisu 3D objektu (varieta) v trojrozmernom priestore, za pomoci dvojrozmernej funkcie. Šablónovanie posúva po šablónovacej trajektórii (sweeping path, trajectory) šablónovaciu funkciu (označovanú ako rez (cross section), prípadne obrys (contour)). Geometrický vzťah medzi obrysom a trajektóriou sa nazýva šablónovacie pravidlo (sweeping rule). Šablónovanie je 3D reprezentácia, ktorej najrozšírenejším produktom je zovšeobecnený valec, (generalized cylinder).

Na Obr. 3 (9) vidíme vľavo kostru a vpravo šablónovanie štvorca pozdĺž kostry.



Obr. 3

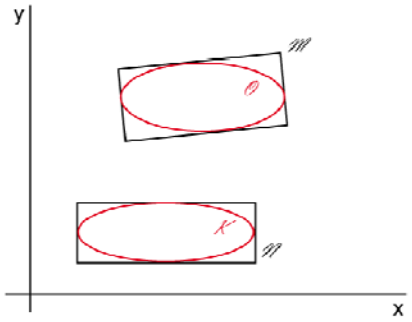
Ak budeme implicitne definovaný objekt chápať ako vzdialenosť ρ bodu od krivky pre trojrozmerné body, dostaneme tak zovšeobecnený valec (generalized cylinder). (9)

2.17. Obálka

Obálkou nazveme minimálny obdĺžnik opísaný objektu, čo zobrazujeme. (10) Pomocou objektu obálky vieme pri zobrazovaní objektov v počítačovej grafike urýchliť výpočet vynechaním časti priestoru, kde sa objekt nenachádza.

Špeciálnym prípadom obálky je osovo rovnobežne orientovaná obálka (Axis Aligned Bounding Box, ďalej len AABB), ktorá pre daný objekt vráti opísaný obdĺžnik rovnobežný s osami súradnicovej sústavy.

Na Obr. 4 vidíme 2 elipsy $\mathcal{O}$ a $\mathcal{K}$ a ich obálky $\mathcal{M}$ a $\mathcal{N}$. Obálka $\mathcal{N}$ je osovo rovnobežne orientovaná obálka (AABB)



Obr. 4

2.18. Polygonizácia

Polygonizáciou (6) nazývame proces, v ktorom pre daný implicitne definovaný povrch $\mathcal{M}$ nájdeme polygonálny povrch $\mathcal{N}$, ktorý aproximuje $\mathcal{M}$ a má rovnakú topológiu, t.j. existuje homeomorfizmus $h : \mathcal{M} \to \mathcal{N}$. Zároveň existuje $\varepsilon > 0$ tak, že $\rho(x, h(x)) < \varepsilon$, pričom ρ je funkcia vzdialenosti v priestore.

Algoritmus prevzatý z (11)

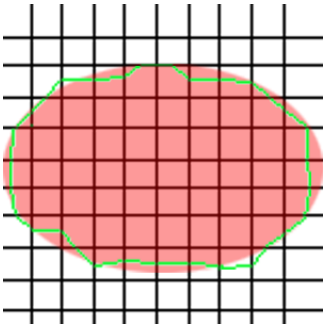
2.19. Marching cubes / squares algoritmus

Algoritmus marching cubes sa využíva na aproximáciu trojrozmerných skalárnych polí pomocou polygonálnej siete nazývanej mesh. (pre dvojrozmerné polia je to marching squares, jeho dvojrozmerný variant).

Algoritmus marching squares v pseudokóde:

- pre každý bod štvorcovej siete sprav:
  - nastav status bodu podľa predpisu implicitne definovanej funkcie aproximovaného objektu na DNU alebo VON
- pre každý štvorec v sieti sprav:
  - pre každú stranu štvorca, ktorá má jeden koniec DNU a jeden VON rekurzívnym delením na polovice, nájdi bod B, v ktorom implicitne definovaná funkcia aproximovaného objektu pretína stranu štvorca
  - pridaj bod B do zoznamu L
- Pospájaj body zo zoznamu L úsečkami a nakresli

Obr. 5 (11) zobrazuje elipsu a jej aproximáciu vytvorenú pomocou upraveného marching squares algoritmu.



Obr. 5

2.20. Tvoriace čiary

Tvoriacimi čiarami nazveme dvojicu vstupných C^1 spojitých po častiach kubických Béziérových kriviek $\mathcal{P},\mathcal{R}$. Dané sú body reprezentujúce riadiace vrcholy kriviek V_i, W_i

$$\mathcal{P} = \bigcup_j {}^jP(t);\ \text{kde } {}^jP(t) = P\left[{}^jV_0,\ldots,{}^jV_3; t \in \left\langle t_j; t_{j+1}\right\rangle\right],$$
$$\mathcal{R} = \bigcup_j {}^jS(t);\ \text{kde } {}^jS(t) = S\left[{}^jW_0,\ldots,{}^jW_3; t \in \left\langle t_j; t_{j+1}\right\rangle\right],$$
$$j = 0,1,\ldots,r-1,$$

kde j vyjadruje segment krivky a pritom platia pravidlá z odseku 2.10.

Pre riadiace vrcholy V_i, W_i kriviek $\mathcal{P},\mathcal{R}$ navyše platí:

$$V_i = (x_i, 0, z_i) \qquad x_i, z_i \in \mathbf{R}$$
$$W_i = (u_i, 0, w_i) \qquad u_i, w_i \in \mathbf{R}$$

T.j. $\mathcal{P},\mathcal{R}$ sú krivky v rovine definovanej súradnicovými osami xz. Posunutím po želanej trajektórii vzniká translačná plocha.

2.21. Kostra

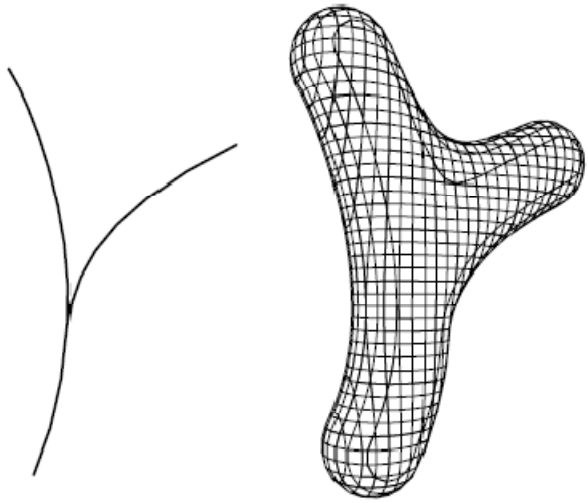
Implicitne definované povrchy, definované pomocou nejakej metriky sa tiež nazývajú kostrové (skeleton based) implicitne definované objekty.

Kostru definujeme ako množinu pozostávajúcu z nasledovných geometrických primitívnych objektov (12):

- bod: jednoduchá kostra, ktorá slúži ako stred implicitne definovaného objektu, napr. gule
- splajny: množina centrálnych osí pre zovšeobecnené valce, kde polomer sa môže meniť pozdĺž kostry
- polygóny: polygonálna sieť tvoriaca ofsetový povrch

Každý kostrový prvok pritom zodpovedá nejakej lokálnej implicitne definovanej funkcii, pričom jednotlivé funkcie sú stmeľované polynomiálnou váhovou funkciou, ktorá môže byť zadaná používateľom.

Na Obr. 6 (12) vľavo vidíme kostru implicitne definovaného objektu, vpravo vidíme príslušný implicitne definovaný objekt



Obr. 6

Zovšeobecnený valec je objekt z triedy šablónovaných objektov (9). Pre kostru pozostávajúcu z n úsečiek je implicitne definovaný zovšeobecnený valec s polomerom r daný ako:

$$f(P) = \frac{\min_i^n(\rho(P, úsečka_i)^2)}{r^2 - 1}.$$

Výsledný implicitne definovaný objekt teda chápeme ako vzdialenosť ρ bodu P od krivky – úsečky pre trojrozmerné body, ako už sme spomínali v odseku 2.16

3. Špecifikácia

Diplomová práca vznikla za účelom potreby vytvoriť špeciálne rezy – zväzok implicitne definovaných plôch na dátovom objeme. Okrem vytvorenia je cieľom práce:

- Preskúmať rôzne spôsoby vytvárania zväzku implicitne definovaných plôch (Curved Clip Ranges), ďalej len CCR
- Navrhnúť postup pri interakciách (posun a rotácia) a manipuláciách (škálovanie),
- Navrhované postupy implementovať

3.1. Spôsoby vytvárania CCR

Jedným zo spôsobov vytvorenia implicitne definovaného objektu je prechod od popisu parametrického, k popisu za pomoci implicitne definovanej funkcie. Prechod sa vykoná elimináciou parametrov v parametrickom vyjadrení.
Napr. v rovnici úsečky v rovine $X = A + t(B - A)$, kde $X = (x; y)$, $A = (a_x; a_y)$, $B = (b_x; b_y)$ odstránením parametra t dostaneme vyjadrenie typu $ax + by + c = 0$, presnejšie: $x(b_y - a_y) - y(b_x - a_x) - (a_x(b_y - a_y) + a_y(b_x - a_x)) = 0$. Teoreticky je možné presne previesť parametrické vyjadrenie na implicitné, v praxi sa však tento postup nevyužíva, pretože vedie na implicitne definované funkcie vysokých stupňov. (6 s. 151)
Metóda pre prevod (hoci ide iba o aproximáciu) je opísanú v článku (13). Prevod sa vykoná vytvorením volumetrickej reprezentácie v dvoch krokoch. Najprv sa vytvorí rasterizácia parametricky zadaného povrchu. V druhom kroku sa objem ohraničený povrchom vyplní. Po zistení charakteristickej funkcie sa aplikuje vyhladenie povrchu.

Konštrukčno-geometrické modelovanie (CSG-strom) (10) síce vytvorí výsledný implicitne definovaný objekt bez potreby počítania rovníc objektu elimináciou parametrov, prístup však nezohľadňuje požadovaný krivkový vstup. Prínosom pri použití konštrukčno-geometrického modelovania je, že z primitívnych implicitne definovaných objektov a booleovských funkcií skladaním vznikne výsledný objekt

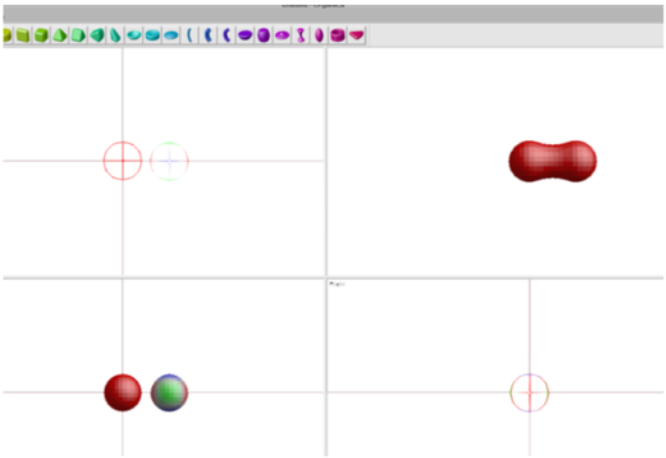
Riešením, ktoré by zodpovedalo požadovanému vstupu sú kostrové (skeleton-based) implicitne definované objekty (13), (12). Tieto izoplochy (metagule, mäkké objekty alebo kvapkové molekuly) umožňujú ľahko ovládať tvar kostry výsledného objektu už známymi technikami. Operátor tak intuitívne vie pochopiť význam zmeny tvaru kostry a primitívneho objektu na výsledný objekt.

Pre zložené implicitne definované povrchy musíme uvažovať o podmienkach spojitosti a hladkosti, resp. o stmeľovacej funkcii, ktorá uvedené podmienky zabezpečí. Ľahké modelovanie stmeľovania je zabezpečené práve s pomocou potenciálov modelujúcich izoplôch (metagule, mäkké objekty alebo kvapkové molekuly).
Rozhodli sme sa použiť 2 varianty :

- model kde primitívne implicitne definované objekty sú založené na modeli metagule
- a model s primitívnym implicitne definovaným objektom založeným na mäkkých objektoch.

Modely metagule (alebo mäkkého objektu) v literatúre (14), (15) majú vlastnosť, ktorá spôsobuje, že umiestnenie dvoch primitívnych objektov metagule (mäkkého objektu) do toho istého bodu v priestore vytvorí výsledný objekt s polomerom väčším, ako mali pôvodné objekty. Zámerom je vytvoriť implicitne definované objekty, ktoré uvedenú vlastnosť minimalizujú. Zároveň však chceme zachovať vlastnosti ľahkého stmeľovania sa primitívnych implicitne definovaných objektov.

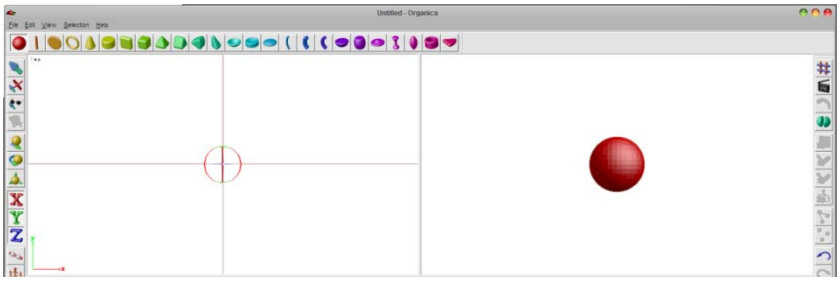
Obr. 7 je príkladom z aplikácie „organica“ (16), ktorá využíva na 3D modelovanie rôzne variácie objektu metagule.



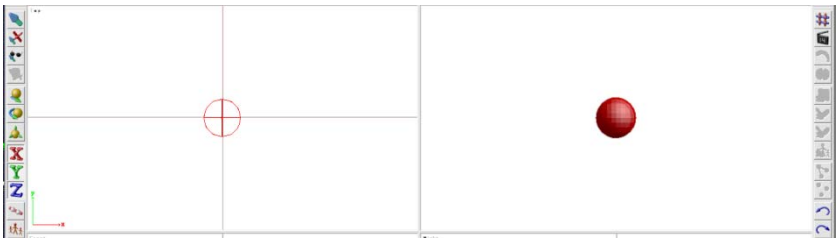
Obr. 7

Na Obr. 7 vidíme výstup z programu „organica“, stmelenie dvoch metaguľových primitív, v priestorovom pohľade vpravo hore, a zobrazenie primitívnych objektov v náryse, pôdoryse a bokoryse.

Na Obr. 8 vľavo vidíme dva objekty metagule vložené do toho istého bodu v priestore. Vpravo je výsledný metaobjekt. Na Obr. 9 vľavo a vpravo vidíme, ako vyzerá referenčný metaguľový objekt, v rovnakých zobrazeniach. Pri porovnaní obrázkov zároveň vidíme už spomenutý fakt, že umiestnením dvoch primitívnych objektov do rovnakého bodu v priestore vzniká výsledný objekt so zmeneným, väčším polomerom, čo je v prípade objektu CCR nežiaduce.



Obr. 8



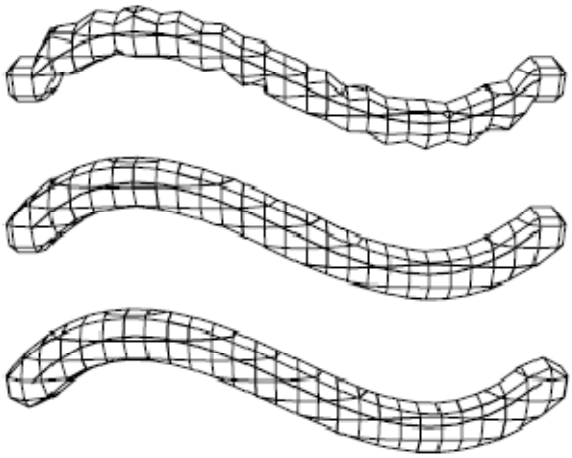
Obr. 9

Pri modelovaní objektov používame algoritmus založený na marching cubes algoritme. Nakoľko sa pri potenciáloch modelujúcich implicitne definovaných objektoch jedná vlastne o modelovanie funkcie poľa, model kvapkových molekúl sme vynechali pre jeho výpočtovú náročnosť, stúpajúcu s rastúcim počtom primitívnych objektov.

Modelovanie implicitne definovaných objektov vždy dáva do nepriamej úmernosti rýchlosť a presnosť modelovania (6). Pre zvýšenie rýchlosti modelovania preto uvažujeme o niekoľkých zjednodušeniach, konkrétne:

- upravený algoritmus marching squares
- modelovanie pomocou metakružníc
- vopred stanovený počet metaguľových primitívnych objektov
- zobrazovanie drôteným modelom
- upustenie od požiadavky modelovania v reálnom čase

Na Obr. 10 (12) vidíme zjednodušenie – modelovanie výsledného objektu prostredníctvom stanoveného počtu primitívnych objektov. Na obrázku hore vidíme aproximáciu s 15 metaguľami, v strede je aproximácia s 30 metaguľami a dole je exaktné riešenie.



Obr. 10

3.2. Interakcie a manipulácie CCR

3.2.1. Interakcia na dátach tvoriacich CCR

Postupy pri interakciách – rotácia, posun a škálovanie. Je treba uvažovať o dvoch úrovniach a to o úrovni zobrazovania a úrovni dát samotného objektu resp. jeho bodov. Pre uskutočnenie všetkých operácií používame homogénne afinné súradnice, zápis uvádzame v maticovom tvare. Pre implicitne definované objekty nie je vždy možné intuitívne pochopiť dôsledky zmeny parametrov vo vyjadrení funkcie, pomocou ktorej je objekt definovaný. Kvôli požadovanému krivkovému vstupu (bodovo zadané parametrické krivky tvoria-

cich čiar) sme sa rozhodli pre vytvorenie zväzku plôch v implementácii aplikovať uvedené operácie na tvoriace čiary zväzku plôch.

V zmysle (10) sú teda matica pre operáciu translácie $\mathbf{T}$ a matica pre škálovanie $\mathbf{S}$ na bodoch $\mathbf{X}$:

$$\mathbf{X} = (x, y, z, 1), \quad \mathbf{T} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{pmatrix}, \quad \mathbf{S} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Na riadiace vrcholy objektu, označené $\mathbf{X}$, aplikujeme transformácie, v maticovom zápise:

$$\mathbf{X}' = \mathbf{XT} \text{ resp. } \mathbf{X}' = \mathbf{XS}$$

dostaneme transformované body $\mathbf{X}'$.

Pre rotáciu v 3D okolo osi $x, y,$ alebo z o uhol φ opäť použijeme v zmysle (10) maticové operácie na riadiacich vrcholoch objektu, bodoch $\mathbf{X}$.

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi & 0 \\ 0 & -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{R}_y = \begin{pmatrix} \cos\varphi & 0 & -\sin\varphi & 0 \\ 0 & 1 & 0 & 0 \\ \sin\varphi & 0 & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{R}_z = \begin{pmatrix} \cos\varphi & \sin\varphi & 0 & 0 \\ -\sin\varphi & \cos\varphi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

$$\mathbf{X}' = \mathbf{XR}_i$$

Pozn. Maticové operácie možno skladať, treba však uvážiť poradie skladania.

3.2.2. Zmena parametrov izometrického zobrazenia

Na zobrazovacej úrovni rovnaký výsledok ako za pomoci úprav vlastností objektu dosiahneme správnym nastavením parametrov izometrického zobrazenia.

Nie je žiaduce zmeniť charakter dát definujúcich objekt CCR, len kvôli potrebe zmeniť pozíciu pozorovateľa. Zmenu pozície pozorovateľa (kamery) docielime nastavením parametrov zobrazovacej funkcie.

Parametre zobrazovacej funkcie (jednoduchá izometria), ktoré možno voliť sú jednotky súradnicových osí e_x, e_y, e_z, uhly ξ, η a súradnice počiatku – bodu $P = (p_x, p_y, 0,1)$ v priemetni. Z vlastností jednoduchej izometrie vyplýva, že pri zmene jednotky musí platiť $e_x = e_y = e_z = e.$ Nastavenie jednotky umožňuje škálovať zobrazenie, zmenou pozície bodu P sa vykoná posunutie, zmenou uhlov ξ, η sa nastaví natočenie súradnicových osí.

Pre bod X v priestore dostaneme súradnice zobrazovaného bodu X_ϵ vhodným nastavením matice zobrazenia $\mathbf{Z}$. (17)

$$\mathbf{X} = (x, y, z, 1),\ \mathbf{X}_\epsilon = (x_\epsilon, y_\epsilon, 0, 1),\ \mathbf{Z} = \begin{pmatrix} -e_x \cos(90-\eta) & -e_x \sin(\eta-90) & 0 & 0 \\ e_y \cos(\xi-90) & -e_y \sin(\xi-90) & 0 & 0 \\ 0 & e_z & 0 & 0 \\ p_x & p_y & 0 & 1 \end{pmatrix}.$$

V maticovom zápise:

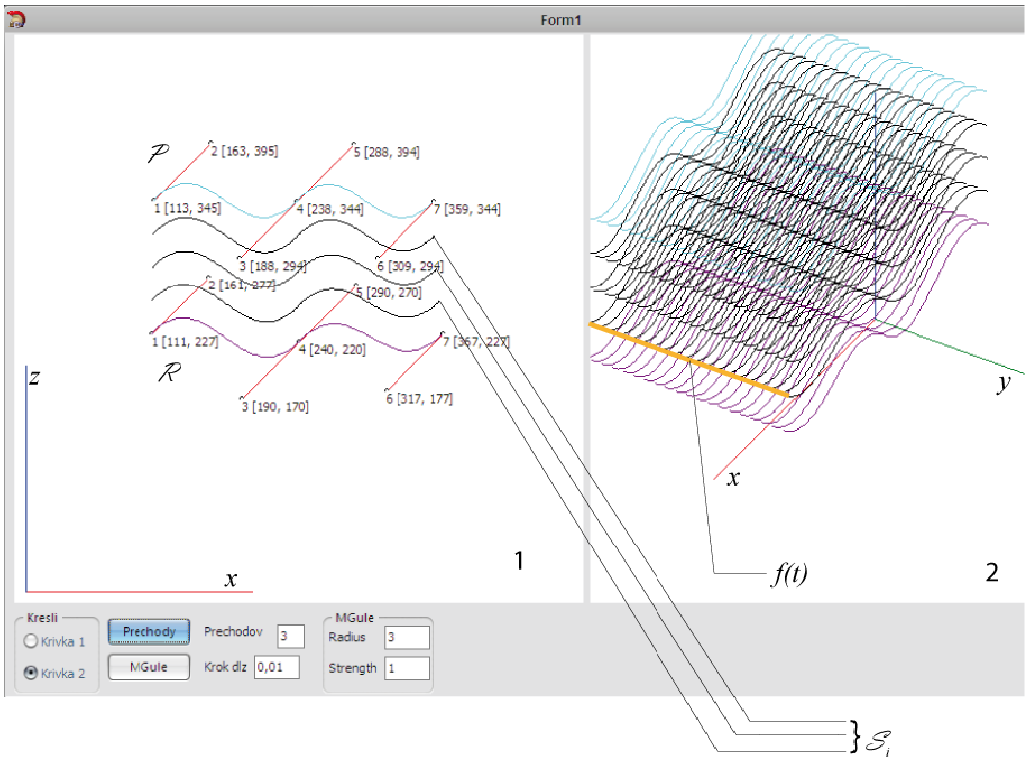
$$\mathbf{X}_\epsilon = \mathbf{XZ}.$$

Pozn. Operácie otočenia, posunutia a škálovania na bodoch tvoriacich objekty v zmysle odseku 3.2.1 a na nastavení zobrazenia možno kombinovať.

3.3. Návrh riešenia

Pri vytváraní CCR si používateľ zadá dve krivky – tvoriace čiary zväzku, počet prechodov medzi nimi. Zadané krivky sa prostredníctvom úchopov dajú tvarovať, a prispôsobiť dodatočne po nakreslení. Medzi zadanými tvoriacimi čiarami sa vytvorí používateľom zadaný počet prechodov. Prechody spolu s tvoriacimi čiarami sú základom – kostrou pre zovšeobecnené valcové plochy nazývané CCR. Implicitne definované objekty, ktoré sa vytvoria

zo zadaných kriviek sa šablónujú, intuitívne povedané „potiahnu sa“, po známej trajektórii.



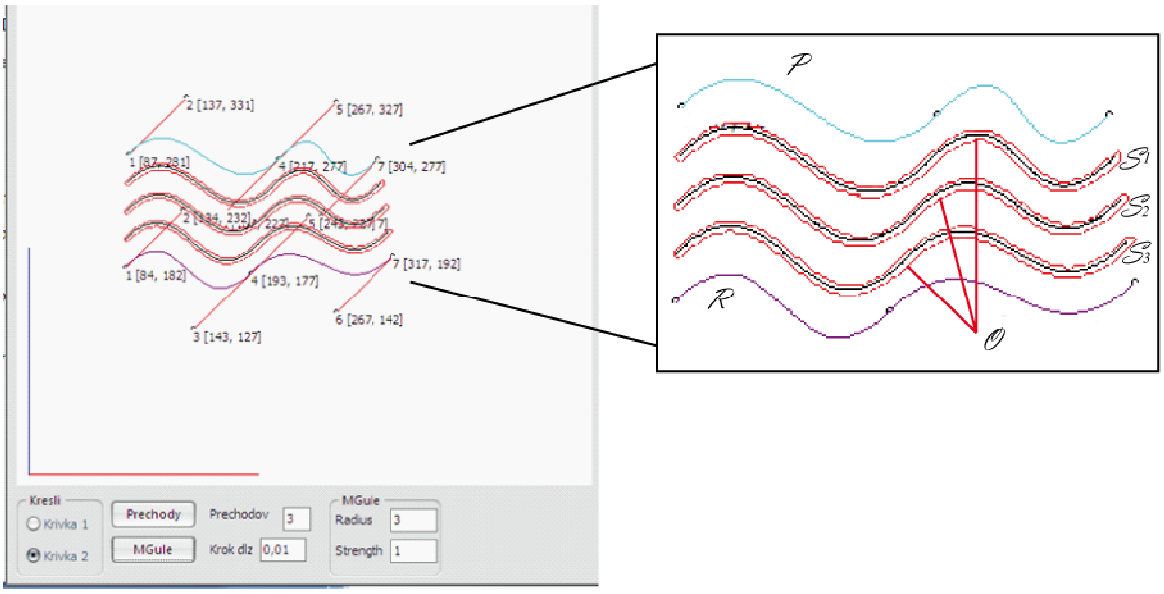
Obr. 11

Na Obr. 11 vidíme okno aplikácie, s ovládacími prvkami a dvomi kresliacimi plochami, označené 1 – ľavá resp. 2 – pravá. V ľavej, kresliacej ploche používateľ zadáva tvoriace čiary – krivky $\mathcal{P},\mathcal{R}$, znázornené svetlo modrou a fialovou farbou. Úchopy sú znázornené ako červené čiary, s príslušným riadiacim bodom ako čiernym krúžkom na konci. Prechody $\mathcal{S}_i$ sú znázornené ako čierne krivky medzi tvoriacimi čiarami. Ľavá kresliaca plocha pritom znázorňuje rovinu danú súradnicovými osami x a z. V pravej kresliacej ploche vidíme izometrické zobrazenie, šablónovanie kriviek $\mathcal{P},\mathcal{R},\mathcal{S}_i$ po oranžovo znázornenej trajektórii $f(t)$.

K takto vytvoreným kostrovým krivkám, používateľ priradí implicitne definované objekty zložené z primitívnych implicitne definovaných objektov metagulí (resp. mäkkých objektov). Vlastnosti primitívnych implicitne definovaných objektov pritom môže nastaviť.

Na Obr. 12 vidíme vľavo výstup z aplikácie, ktorý predstavuje rez zväzkom implicitne definovaných objektov CCR v rovine xy. Vpravo je detailné zobrazenie. Kostrové krivky sú čierne krivky označené $\mathcal{S}_1,\mathcal{S}_2,\mathcal{S}_3$ a zväzok implicitne definovaných prislúchajúcich

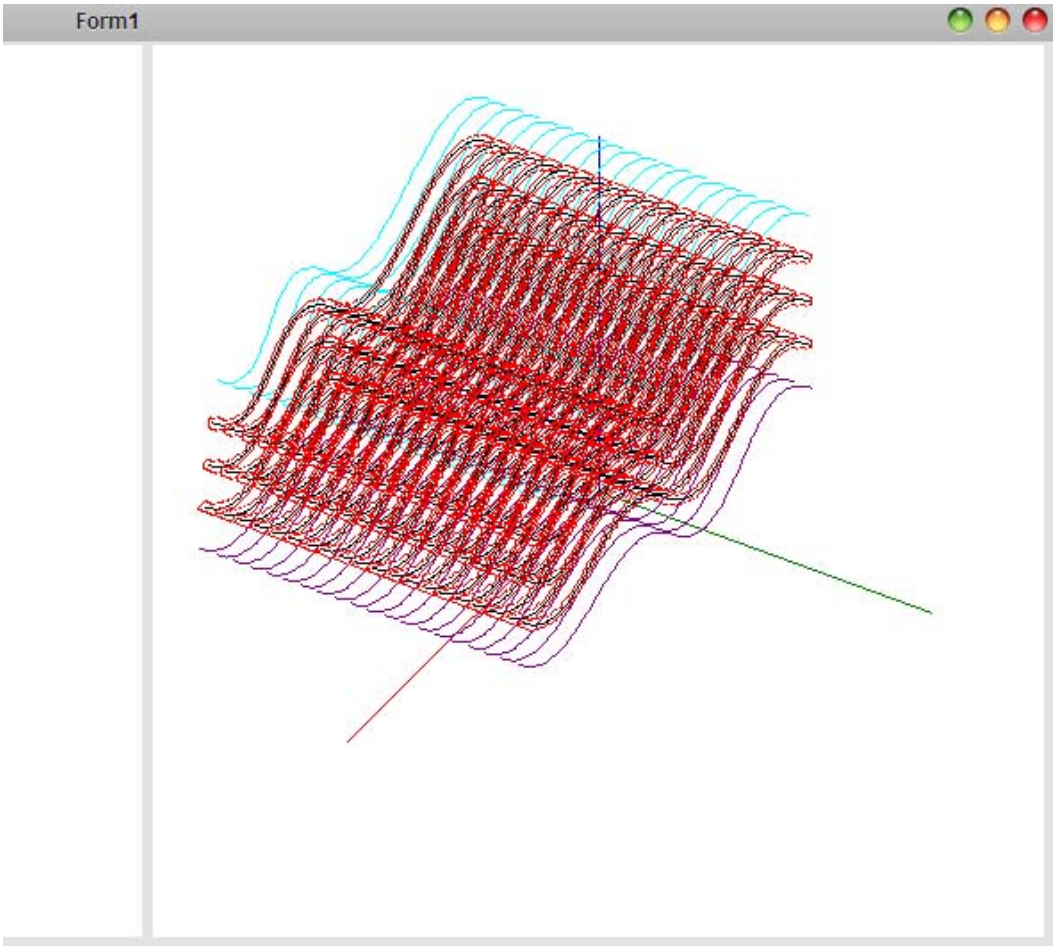
objektov je označený ∅ a červeno zvýraznený. Implicitne definované objekty vznikli stmelením primitívnych implicitne definovaných objektov metagulí.



Obr. 12

Z charakteru zväzku implicitne definovaných objektov CCR vyplývajú ťažkosti pri zobrazovaní. Keďže pri zobrazení v priestorovom náhľade vznikne „klbko“ množstva prepletených čiar, rozhodli sme sa proti použitiu drôteného modelu. Ilustrovanie tvaru objektu s použitím iba bodov (2) by sa stalo rovnako neprehľadné. Využitie nepriehľadných modelov by malo význam len pre jednotlivé plochy zväzku. Rozhodli sme sa pre ilustrovanie charakteru plôch tvoriacich zväzok CCR zobrazenie použiť rezy implicitne definovanými objektmi.

Výstup z aplikácie možno vidieť na Obr. 13, zväzok implicitne definovaných plôch CCR je zvýraznený červenou farbou. Zobrazené sú v rezy v jednotlivými implicitne definovanými plochami zväzku.



Obr. 13

4. Implementácia

4.1. Zväzok implicitne definovaných plôch (Curved Clip Ranges)

Zväzok implicitne definovaných plôch (Curved Clip Ranges), ďalej len CCR je 3D zložený implicitne definovaný objekt. Vytvorenie CCR je pritom dvojfázové. V prvom kroku sa z používateľom zadaných bodov vytvoria tvoriace čiary $\mathcal{P},\mathcal{R}$ a následne, lineárnou interpoláciou používateľom zadaný počet prechodov $\mathcal{S}_i$, vstup do druhého kroku. (Prechody sú pritom C^1 spojité po častiach kubické Bézierove krivky). Každý z prechodov $\mathcal{S}_i$ – interpolanty v zmysle odseku 2.5, tak tvorí kostru implicitne definovaného objektu $\mathcal{O}_i$. Implicitne definovaný objekt $\mathcal{O}_i$ vznikne ako stmelenie metagulí (alebo mäkkých objektov) $\mathcal{M}_j$ umiestnených do bodov kostry $\mathcal{S}_i$. Polomer a silu primitívneho objektu metagule $\mathcal{M}$ možno pritom definovať používateľom.

Druhá fáza vytvorenia CCR spočíva v šablónovaní každého z implicitne definovaných objektov $\mathcal{O}_i$ po zadanej trajektórii.

Výsledný zväzok implicitne definovaných plôch je objekt CCR, ktorý je zjednotením:

$$\mathcal{O} = \bigcup_i \mathcal{O}_i$$

$$\mathcal{O} = \bigcup_i \bigcup_j \mathcal{M}_j$$

4.2. Algoritmus vytvorenia objektu $\mathcal{O}$ (CCR)

Pri vytváraní CCR objektu postupujeme nasledujúcim spôsobom:

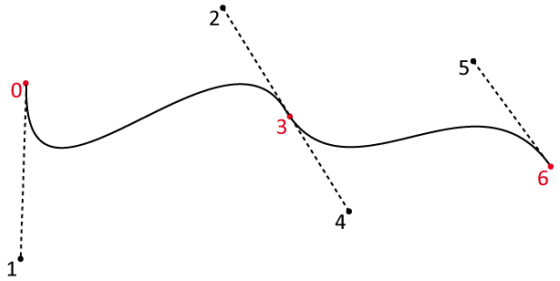
4.2.1. Tvoriace čiary - vytvorenie

Máme užívateľom zadané dve sady bodov $V_0,\ldots,V_n$ a $W_0,\ldots,W_n$. Body V_i sú riadiacimi vrcholmi krivky $\mathcal{P}$, rovnako B_i sú riadiacimi vrcholmi krivky $\mathcal{R}$.

Chceme: „tvoriace čiary“ pre implicitne definované $\mathcal{O}_i$ objekty, prvý krok vytvorenia objektu $\mathcal{O}$, CCR.

Na zadaných množinách riadiacich vrcholov vytvoríme C^1 - spojité po častiach kubické Beziérove krivky. (za pomoci aplikovania de Casteljau algoritmu alebo Bernsteinových polynómov)

Na Obr. 14 vidíme body $V_0,\ldots,V_6$, ktoré definujú C^1 spojitú po častiach kubickú Bézierovu krivku s dvomi segmentmi. Riadiace vrcholy sú dvoch typov. 0,3,6 sú body, ktoré krivku interpolujú. Ostatné vrcholy tvoria úchopy na tvarovanie krivky. Pri pridávaní vrcholov myšou vznikajú vrcholy, interpolačné, úchopy sú dopočítavané tak, aby boli zachované podmienky spojitosti. Všetky vrcholy možno editovať s pomocou myši.



Obr. 14

Obr. 15 je ukážkou z implementácie.

Obr. 15

Pozn. Keďže tvoriace čiary sú zložené z kubických segmentov Bézierových kriviek, je možné vypočítaním Bernsteinových polynómov vopred skrátiť čas výpočtu a pamäťovú

náročnosť. C^1 spojitosť zabezpečuje možnosť používateľsky zvoliť len prvý a štvrtý bod pre segment, ostatné sa dopočítajú tak, aby bola zachovaná rovnosť derivácií podľa podmienok z odseku 2.10 (po častiach spojitá krivka) .

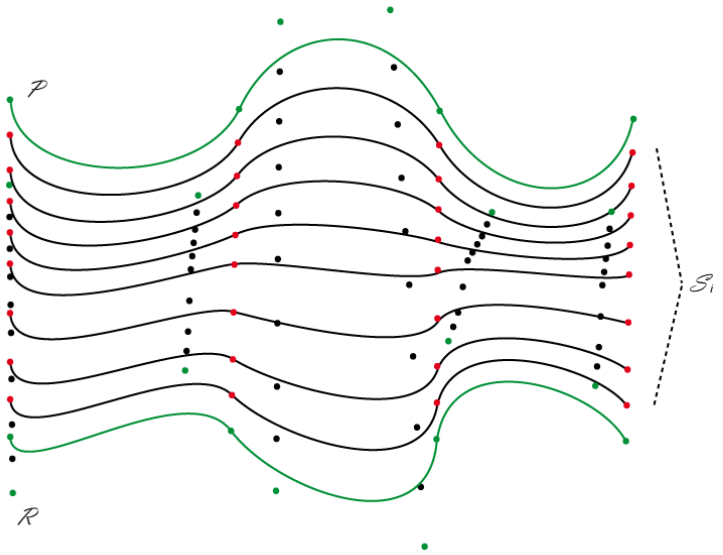
4.2.2. Prechody, kostrové krivky - vytvorenie

Máme dve krivky $\mathcal{P}$ a $\mathcal{R}$ s rovnakým počtom riadiacich vrcholov.

Chceme: prechody medzi $\mathcal{P}$ a $\mathcal{R}$, rez kostrou objektu CCR v rovine $y = 0$

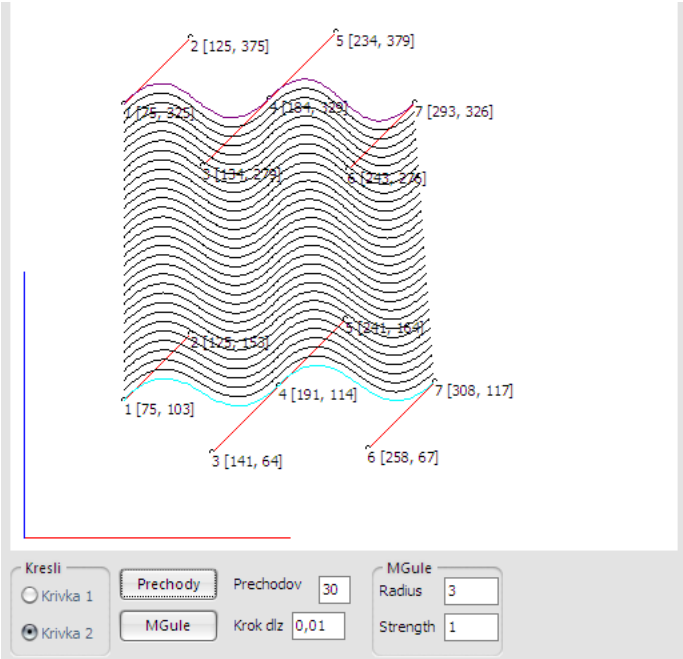
Interpoláciou medzi bodmi kriviek $\mathcal{P}$ a $\mathcal{R}$ dostávame prechodové krivky, nazývame ich kostrové čiary CCR objektu. (Interpolácia medzi jednotlivými bodmi sa javí ako zbytočne náročná na výpočet, zvolili sme preto interpoláciu medzi riadiacimi vrcholmi kriviek $\mathcal{P}$ a $\mathcal{R}$, ktorou získame riadiace vrcholy pre prechody)

Na Obr. 16 vidíme skonštruované prechody $\mathcal{S}_i$, medzi krivkami $\mathcal{P}$ a $\mathcal{R}$, kde počet prechodov volí používateľ. Pre dané t teda dostávame $\mathcal{S} = (1-t)\mathcal{P} + t\mathcal{R}$. Zeleno zvýraznené sú pôvodné zadané tvoriace čiary a ich riadiace vrcholy. Riadiace body, ktoré krivky interpolujú sú červeno zvýraznené.



Obr. 16

Ukážka z aplikácie, vytvorenie 30 prechodov medzi krivkami $\mathcal{P}$ a $\mathcal{R}$ je na obrázku



Obr. 17

4.2.3. Prechod k implicitne definovaným objektom

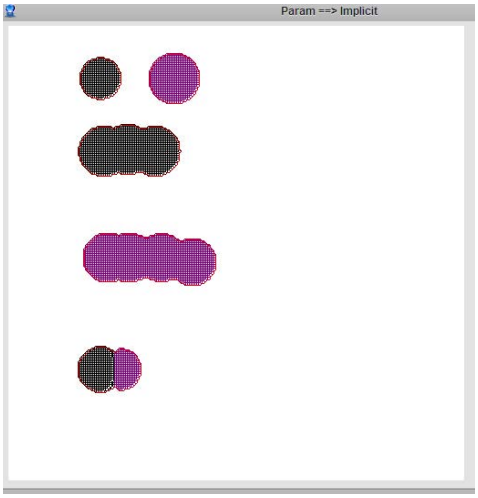
Máme tvoriace čiary – krivky $\mathcal{P}$ a $\mathcal{R}$ a prechody – kostrové krivky $\mathcal{S}_i$ pre CCR objekt.

Chceme: Implicitne definovaný objekt z parametricky zadaných kostrových kriviek, nulté štádium tvorby CCR,

Umiestnením primitívnych implicitne definovaných objektov (Metaballs, soft-objects alebo blobby molecule model) do vypočítaných bodov tvoriacich a základných čiar, a ich následného zlepenia získame vstup do metódy šablónovania.
Bodom, ktorých vzdialenosť $r \geq r_{max}$ od kostrových kriviek $\mathcal{S}_i$ primitíva sa nastaví nulový príspevok. Tvar výsledného implicitne definovaného objektu možno vytvorením aditívnych a subtraktívnych primitívnych implicitne definovaných objektov.

Na ukážke z aplikácie na Obr. 18 vidíme čiernou farbou nakreslené implicitne definované objekty zložené z aditívnych primitív, fialovou farbou kreslené sú objekty pozostávajúce

zo subtraktívnych primitív. Na obrázku dole vidieť aj zloženie vzájomné zloženie aditívneho a subtraktívneho primitívneho objektu.



Obr. 18

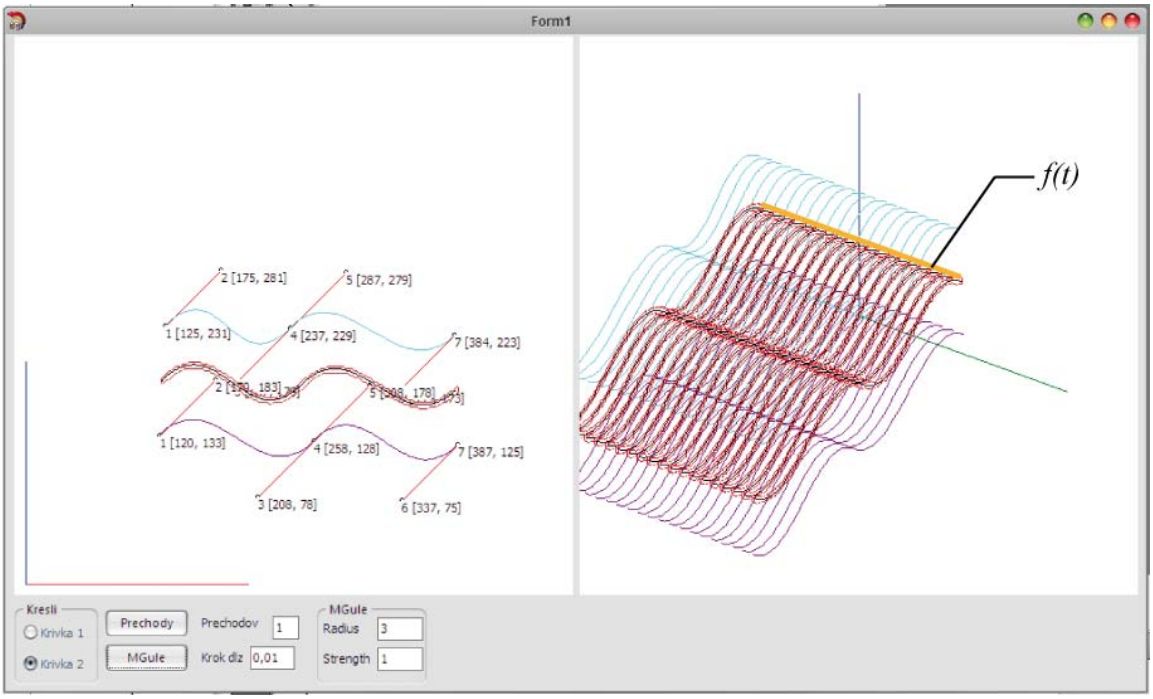
4.2.4. Využitie šablónovania

Na kostrových krivkách máme vytvorené implicitne definované objekty $\mathcal{O}$. Kostrové krivky v rovine xz majú body s nulou y súradnicou. V aplikácii sú zjednodušene zobrazované implicitne definované objekty pomocou rezu v rovine xz.

Chceme: Celkový CCR objekt

Metódou šablónovania „potiahneme“ implicitne definované objekty $\mathcal{O}$ po požadovanej trajektórii. Presnejšie body implicitne definovaných objektov $\mathcal{O}$ sú posunuté po trajektórii, zadanej v aplikácii napevno. Vznikne tak translačná plocha.

Na Obr. 19 vidíme výstup z aplikácie, vľavo tvoriace čiary (fialová a modrá farba) s úchopmi, prechody – kostrovú krivku (čierna farba) a rez implicitne definovaným objektom $\mathcal{O}$ (červená farba), vpravo vidíme kompletný implicitne definovaný objekt ktorý vznikol šablónovaním po trajektórii zvýraznenej oranžovou farbou, označenej $f(t)$.



Obr. 19

4.3. Programovacie prostredie a jazyk

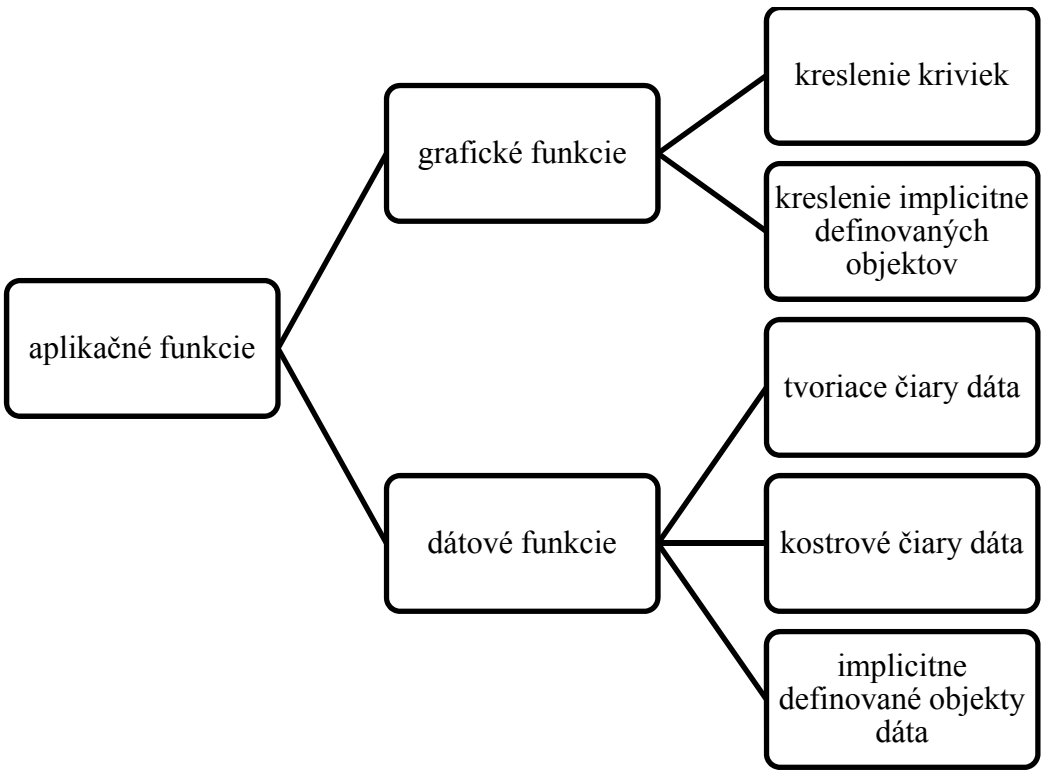
Pri výbere z možností Delphi, Java, C/C++, C#, FLASH a vývojových prostredí Microsoft Visual Studio Express 2008, Borland Builder, devcpp sme sa rozhodli použiť jazyk C/C++ a vývojové prostredie Borland C++ Builder.

Programovací jazyk C resp. C++ (ktorý vznikol ako rozšírenie jazyka C) spolu s grafickou knižnicou OpenGL patrí k najrozšírenejším nástrojom používaným na programovanie 3D grafiky. Jazyk C umožňuje písať programy na dostatočne nízkej úrovni tak, že sa pre takéto programy nemusí vytvárať ich prepis v jazyku assembler. Jazyk C++ zároveň umožňuje používať objekty, spojiť tak dáta a funkcie na ich modifikáciu (nazývané v objektovom programovaní metódy).

OpenGL knižnice zaručujú platformovú prenositeľnosť a ľahkú rozšíriteľnosť, zároveň poskytujú bohaté programové vybavenie funkcií. Keďže cieľom práce je preskúmanie rôznych spôsobov vytvárania CCR, rozhodli sme sa OpenGL a ani inú 3D knižnicu nepoužiť, nakoľko sú to zaužívané a známe spôsoby vytvárania 3D objektov.

Vizuálna knižnica (visual component library, „<vcl.h>“) vývojového prostredia Borland C++ Builder umožňuje rýchly a jednoduchý vývoj aplikácií na platforme WINDOWS, za pomoci win32 API, v prostredí Microsoft Visual Studio Express je to síce tiež možné, no náročnejšie. Nástroj na ladenie (debugger) programov je v Borland C++ Builder prostredí na vysokej úrovni, prehľadnejšie spracovaný ako v prostredí Microsoft Visual Studio Express 2008.

4.4. Programovanie – implementácia



Obr. 20

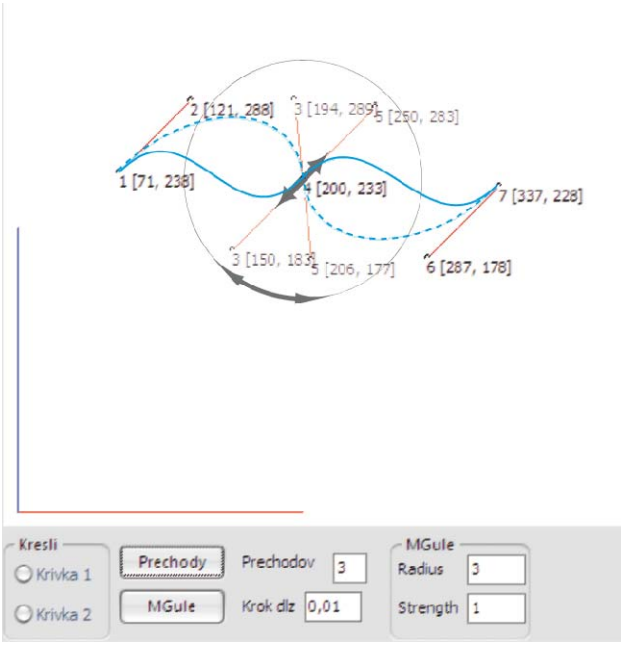
Diagram na Obr. 20 znázorňuje zjednodušenú hierarchiu funkcií využívaných aplikáciou.

Na ukladanie dát v aplikácii využívame štruktúry TBod, TMQBod a triedy TRV, MBall. V štruktúre TBod sa udržujú homogénne súradnice obrazových bodov. Prvky štruktúry TMQBod sa používajú na realizáciu marching cubes algoritmu. Trieda TRV ukladá dáta a metódy pre riadiace vrcholy Bézierových kriviek. Trieda MBall je reprezentáciou dát a metód pre metagule.

„Naklikávanie“ a modifikáciu riadiacich vrcholov pomocou tlačidla myši zabezpečujú funkcie obsluhujúce udalosť onMouseDown a onMouseMove, ktorých vstupy sú obrazové súradnice kliknutého bodu a stlačené tlačidlo myši. Pre ľavé tlačidlo sa parametre kliknutého obrazového bodu predajú ďalej funkcii PridajBod, ktorá zabezpečuje realizáciu pridania bodu do príslušného poľa, podľa zvolenej krivky. Pri stlačenom pravom tlačidle sa zavolá funkcia OdoberBod (s parametrami kliknutého obrazového bodu), ktorá body vymazáva. Stredným tlačidlom myši sa označí bod, ktorý používateľ chce presunúť.

PridajBod a OdoberBod sú funkcie, ktorých vstupom sú súradnice obrazových bodov. Funkcie modifikujú globálne pole zvolenej krivky (r_pole[] alebo u_pole[]). Okrem naklikaných riadiacich bodov sa do polí riadiacich vrcholov zapisujú (odoberajú) vypočítané riadiace vrcholy zabezpečujúce vytvorenie úchopov.

Realizácia úchopov je zabezpečená s pomocou Bézierových kubických segmentov kriviek. Úchopy zlepšujú ovládateľnosť tvoriacich čiar. Pretože pri Bézierových krivkách sú prvý a posledný bod krivkou interpolované, ak je krivka zložená z kubických segmentov, okolo každého krivkou interpolovaného bodu je dvojica bodov, (s výnimkou prvého a posledného bodu krivky kde už nenadväzujú ďalšie segmenty) umožňujúca vytvorenie riadiaceho prvku úchopov. Pri modifikácii úchopov sa tretí riadiaci vrchol jedného a druhý riadiaci vrchol druhého segmentu navzájom symetricky ovplyvňujú.



Obr. 21

Na Obr. 21 vidíme ukážku z aplikácie, modifikáciu úchopov. Body 3 a 5 sa pri zmene jedného z nich symetricky zmenia obidva, vzhľadom na bod 4.

Pretože komponent TImage, zabezpečujúci plochu na vykresľovanie má ľavotočivú súradnicovú sústavu s počiatočným bodom umiestneným v ľavo hore, a súradnice z euklidovskej roviny $\mathbf{E}^2$ majú pravotočivú súradnicovú sústavu s počiatočným bodom obvykle v strede resp. vpravo dole, je potrebné súradnice transformovať. Na transformáciu súradníc z obrazových na priestorové slúži funkcia transfObr_SurXY, ktorej parametrami sú súradnice X,Y kliknutého bodu a počiatok starej a novej súradnicovej sústavy. Na opačnú transformáciu slúži funkcia transfSurXY_Obr s rovnakými parametrami.

Na zobrazenie priestorového náhľadu slúži funkcia Izometria, ktorej parametre sú jednotky súradnicových osí x, y, z a uhly α, β. S pomocou jednoduchej izometrie zobrazujeme body v priestore vykreslené do obrazovej roviny. Okrem funkcie izometrie používame zobrazenie Nárys, ktoré vytvára priemet do roviny určenej súradnicovými osami xz.

```
TBod Izometria(ss, nO, ex, ey, ez, alfa, beta)
{
      TBod ns;
      ns.x = nO.x+(ey*ss.y*cos(beta*M_PI/180) -
                        ex*ss.x*sin(alfa*M_PI/180));
      ns.y = nO.y-(ss.z*ez - ss.x*ex*sin(alfa*M_PI/180) -
                        ss.y*ey*sin(beta*M_PI/180));

      return ns;
}


TBod Narys(ss, nO)
{
      TBod ns;
      ns.x = nO.x+(ss.x);
      ns.y = nO.y-(ss.z);
      return ns;
}
```

Funkcia MarchingCubes() je aplikáciou menovaného algoritmu. V literatúre (11) sme našli modifikovaný algoritmus, ktorý sme implementovali. Na implementáciu používa globálne pole p[][] prvkov typu TMQBod. Pri implementácii je nevyhnutné vhodne zvoliť konštantu delenie, ktorá vyjadruje jemnosť rozkladu priestoru do buniek.

Realizácia algoritmu MarchingSquares:

```
{
nastav delenie=konst;
nastav L=NULL;

pre y=0 až y=delenie+1;
      pre x=0 až x=delenie+1;
            ak (p[y][x] vyhovuje rovnici implicitne def. objektu)
                  nastav p[y][x]=DNU;
            inak
                  nastav p[y][x]=VON;

pre y=0 až y=delenie+1;
      pre x=0 až x=delenie+1;
            ak (p[y][x] alebo p[y][x+1] je DNU)
                  pridaj (p[y][x] + p[y][x+1])/2 do L;

            //analogicky pre ostatné úsečky štvorca delenia;

vráť zoznam bodov L pospájaných čiarami;
}
```

Pre zvýšenie efektívnosti a rýchlosti algoritmu Marching cubes sme použili funkciu Obálka AABB(). Pri pridávaní objektov metagulí do priestoru rozdeleného pomocou Marching cubes algoritmu sa tak obmedzilo zbytočné prechádzanie celého priestoru a pridávanie nulových príspevkov do buniek priestoru, kde sa objekt metagule nenachádza.

Kresli() je funkcia zabezpečujúca vykreslenie tvoriacich čiar CCR objektu (t.j. kriviek $\mathcal{P},\mathcal{R}$ z časti definície, tvoriace čiary). Ako vstup používa globálne polia a premenné r_body[], u_body[], pocet, poc, pocetu, pocu, a obrazové plochy pre vykreslenie nárysu resp. izometrického zobrazenia. Pole r_body[] obsahuje riadiace vrcholy krivky $\mathcal{P}$, pričom celočíselná premenná pocet obsahuje celkový počet riadiacich vrcholov a celočíselná pre-

menná poc obsahuje počet segmentov krivky. Pole u_body[], a premenné pocetu, pocu sú ekvivalentmi pre krivku $\mathcal{R}$.

Pomocou funkcie Prechody() sa lineárne interpolujú riadiace body z polí r_pole[] a u_pole[], vznikne tak pole interpolantov pkrivky[][], ktoré sú kostrou pre implicitne definované objekty. Funkciu Prechody() možno používateľsky ovplyvniť volením počtu interpolantov a tiež možno nastaviť jemnosť kostrových kriviek.

Po vypočítaní kostrových kriviek zabezpečí funkcia PridajMG() vloženie implicitne definovaných objektov metagulí do bodov kostrových kriviek a zároveň udržuje zoznam implicitne definovaných objektov, ktoré sa vykreslia po aplikovaní algoritmu Marching cubes.

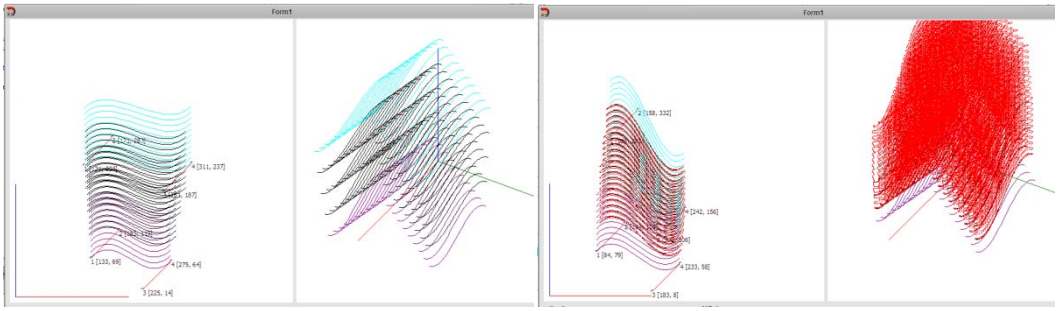
V aplikácii sú k dispozícii dve plochy na kreslenie, ľavá umožňuje pridávať body tvoriacich čiar, kontrolovať a upravovať ich priebeh, vytvoriť prechody – kostrové krivky. V pravej ploche sa zobrazuje výsledný produkt, tiež je možné ovládať zobrazenie, posúvať a otáčať a škálovať.(ešte nie je možné upravovať druhú kresliacu plochu)

5. Výsledky a pokusy

Pôvodná aplikácia CCR je zameraná na zovšeobecnené valcové plochy, ktorých trajektóriou je priamka, resp. úsečka rovnobežná s osou y. S minimálnym úsilím možno aplikáciu upraviť a nastaviť aj iné druhy kriviek. Nasledujú ilustračné obrázky pre ukážkové skúmané situácie.

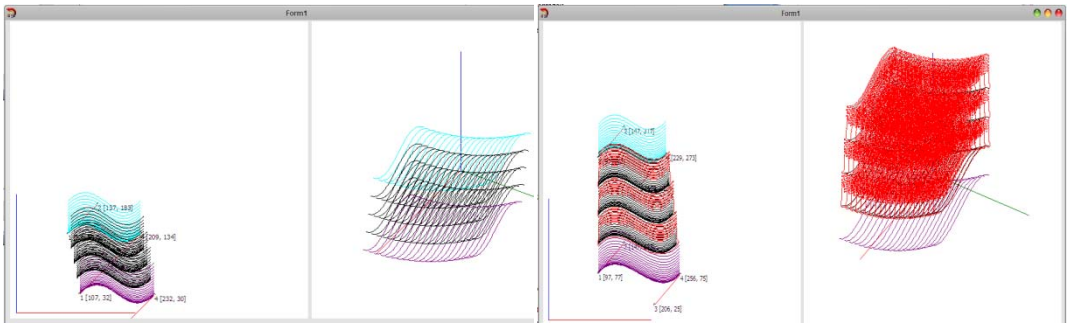
Na nasledujúcich obrázkoch vľavo sa nachádza výstup z aplikácie, ilustrujúci tvar tvoriacich a kostrových čiar v náryse a bokoryse, vpravo sú v rovnakých vyobrazeniach iné tvoriace a kostrové čiary, spolu s odvodeným zväzkom implicitne definovaných plôch.

- Lomená čiara



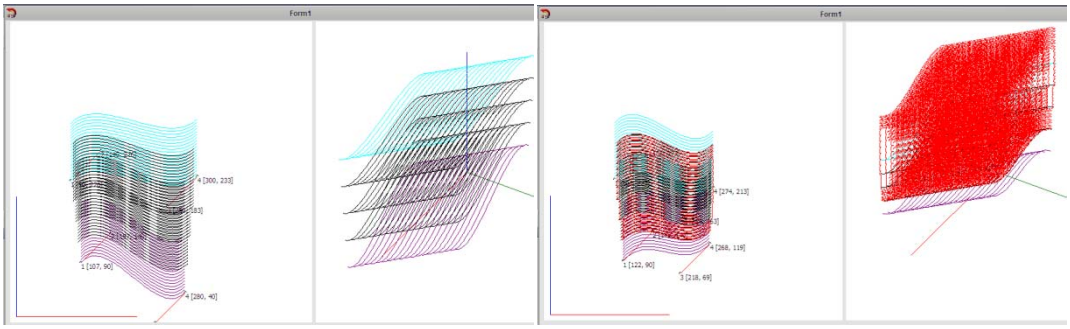
Obr. 22

- Parabola



Obr. 23

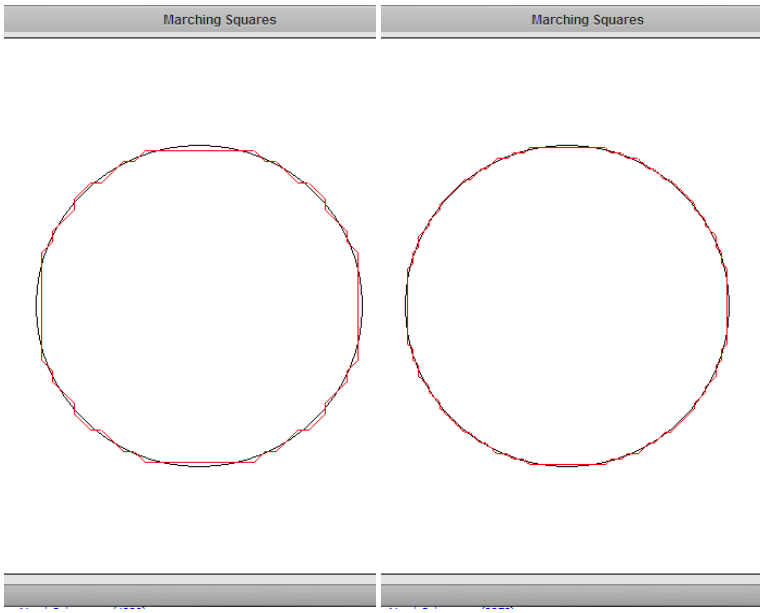
- Priamka vo všeobecnej polohe



Obr. 24

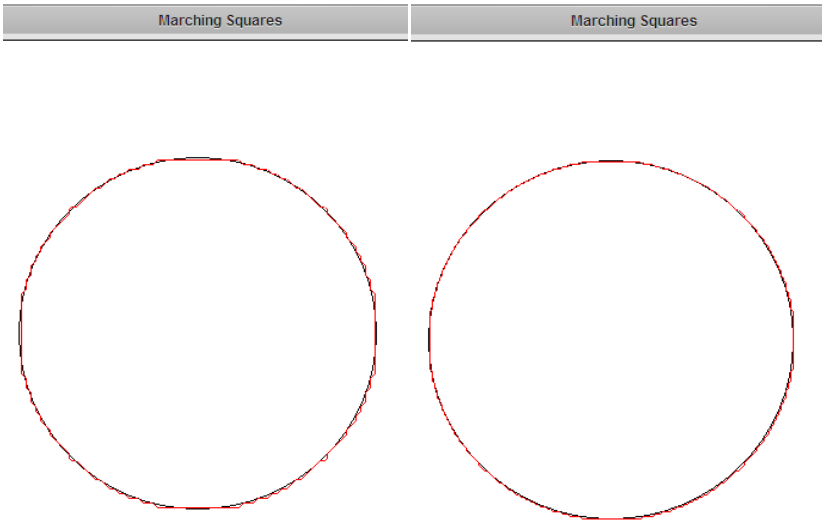
Pri vytváraní implicitných objektov sme použili „vylepšený“ algoritmus marching cubes, pri ktorom je kritickým krokom správne nastavenie konštanty delenia.
Ukážky pre rôzne nastavenia konštanty delenia:

Na obrázkoch vidíme čierno nakreslenú kružnicu a červenou farbou jej aproximáciu pomocou upraveného algoritmu marching squares s konštantou delenia nastavenou:
Vľavo 50, vpravo 100



Obr. 25

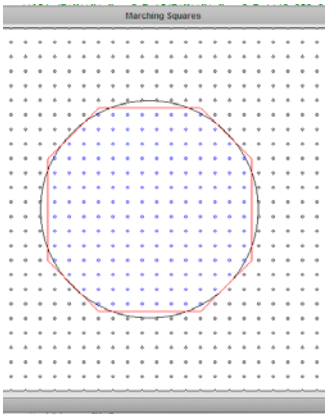
Vľavo 125, vpravo 250



Obr. 26

Ilustrovaný princíp vylepšeného algoritmu marching squares, spolu s ukážkou takto aproximovaných objektov:

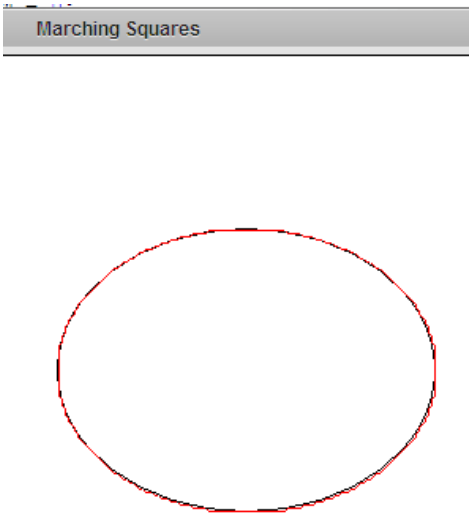
Na ukážke na Obr. 27 vidíme body mriežky pre vylepšený algoritmus marching cubes.



Obr. 27

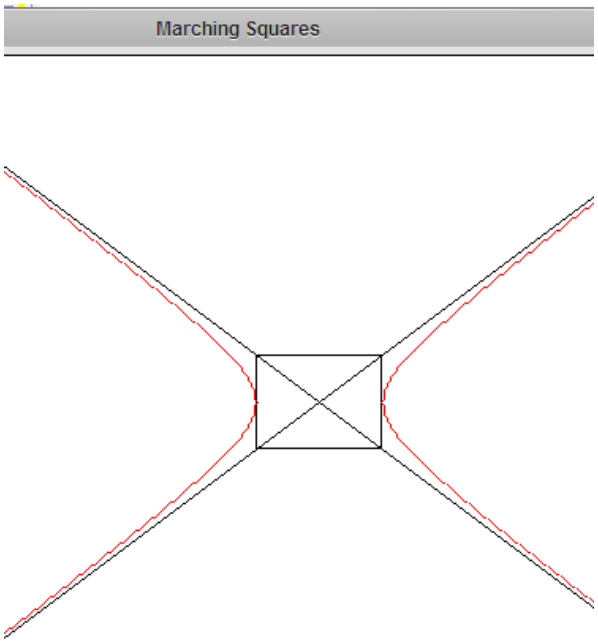
Pri prechode mriežkou sa podľa rovnice implicitne definovaného objektu nastaví príznak DNU (modré body) alebo VON (čierne body). Delením strán štvorcov v mriežke na polovice vznikajú body aproximácie, pospájané na obrázku červenými čiarami.

- Elipsa – Na obrázkoch je opäť červenou farbou zvýraznený objekt vykreslený pomocou upraveného marching squares algoritmu a čiernou farbou pôvodný exaktný objekt.



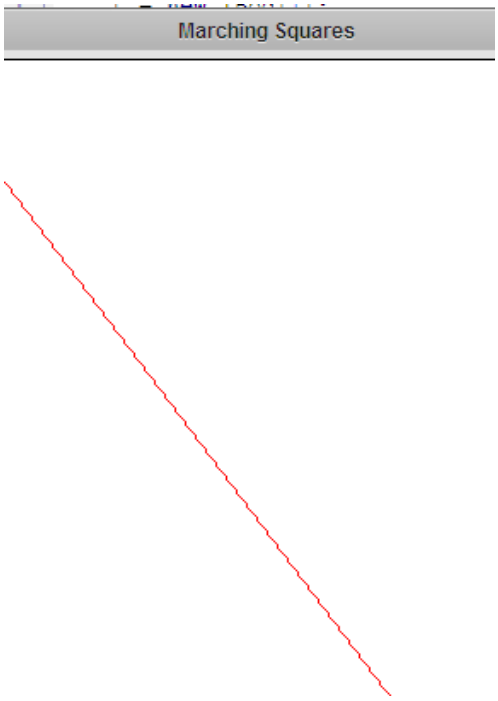
Obr. 28

- Hyperbola – Na obrázku sú čiernou farbou nakreslené asymptoty hyperboly, červenou nakreslená je aproximácia hyperboly pomocou upraveného marching cubes algoritmu



Obr. 29

- Priamka

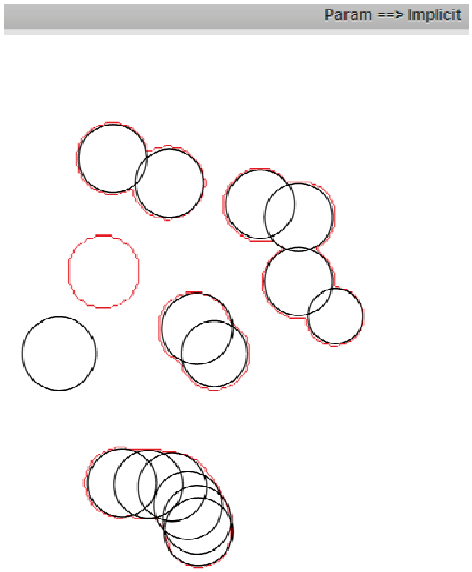


Obr. 30

- Parabola

Obr. 31

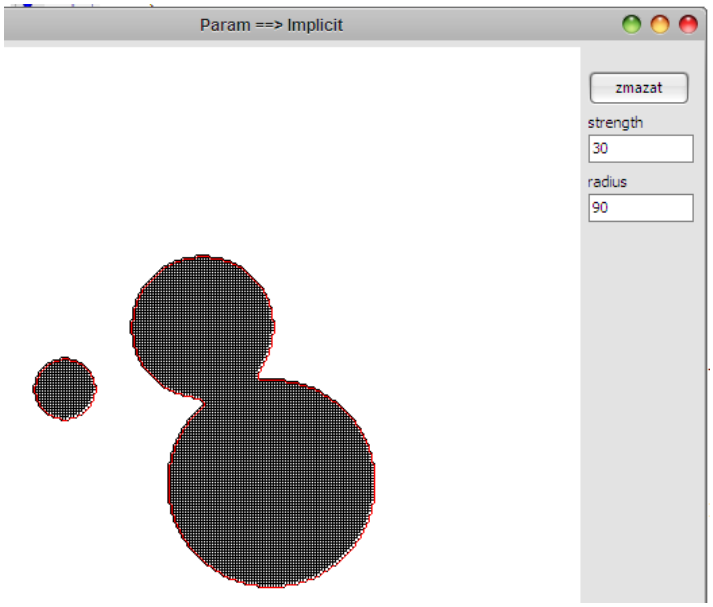
- Metaguľa (kružnica) & Stmelenie metagulí (s kružnicami pre ilustráciu stmelenia primitív)



Obr. 32

Na Obr. 32 vidíme červenou farbou nakreslené implicitne definované objekty zložené z primitívnych implicitne definovaných objektov metagulí. Čiernou farbou nakreslené sú kružnice, z ktorých každá reprezentuje primitívny objekt metagule. Na obrázku vľavo je aj referenčný objekt metagule a referenčná kružnica.

Veľkosť metagulí – t.j. sila a polomer sa dajú pre metagule nastaviť, pre plochy zväzku, ktoré sú zamerané na rezy sa javí ako vhodná sila rovná 1 a polomer rovný 3.



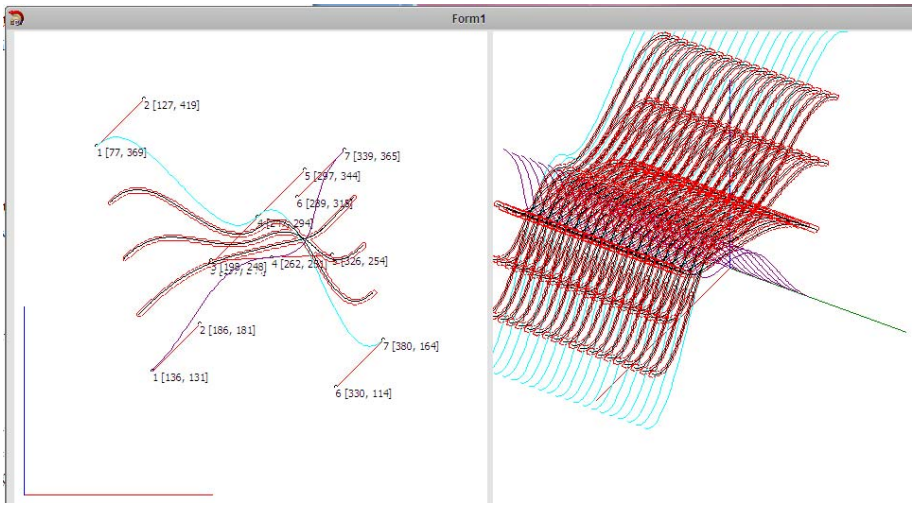
Obr. 33

Na Obr. 33 vľavo vidíme metaguľu s parametrami sila 10, polomer 30 a vpravo stmelenie dvoch metagulí s parametrami sila 20, polomer 60 a sila 30, polomer 90. Hranica stmelenia metagulí resp. metagule, za ktorou už funkcia poľa nemá príspevky je vykreslená červenou farbou.

5.1. Zistené obmedzenia

Nie je vhodné aby sa tvoriace čiary pretínali. Vzniknutý objekt v takomto prípade nebude mať charakter zväzku implicitne definovaných plôch, vznikne jeden implicitne definovaný objekt tvaru „X“, resp. „viacramenné X“.

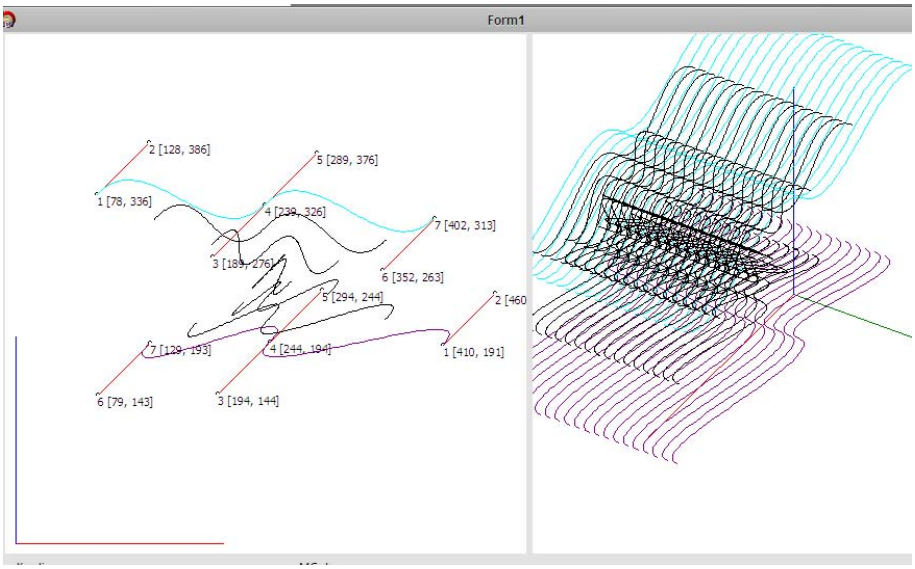
Na Obr. 34 vidíme uvedenú situáciu ilustrovanú výstupom z aplikácie. Vľavo na obrázku sú pretínajúce sa tvoriace čiary a kostrové krivky, v pravo je situácia v priestore.



Obr. 34

Nie je vhodné aby tvoriace čiary mali opačnú orientáciu. Orientáciou tvoriacich čiar rozumieme poradie riadiacich vrcholov (resp. x-ových súradníc riadiacich vrcholov) vzhľadom na os x. Pre ilustráciu uvádzame nasledovný obrázok, kde poradie riadiacich vrcholov vzhľadom na os x je pre tvoriace čiary opačné.

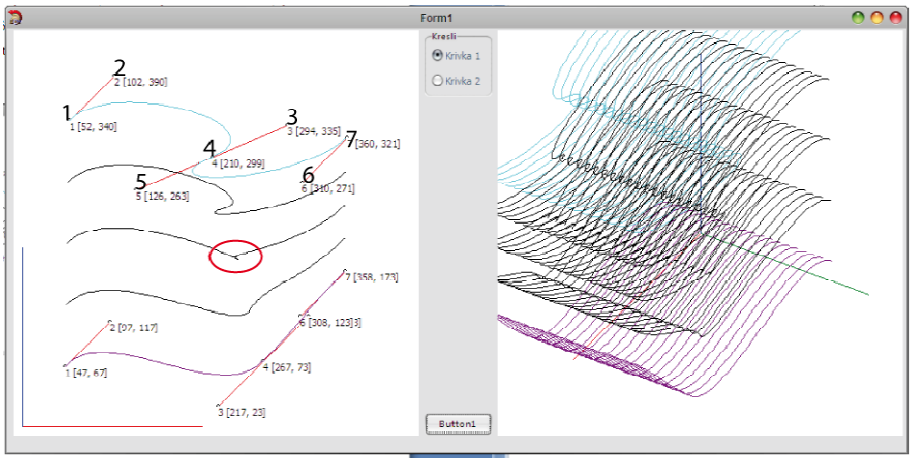
Vzniknuté prechody tak zodpovedajú skutočnosti, situácia je však nežiaduca, pretože vzniknuté kostrové krivky sa tak môžu pretínať. Ak sa kostrové krivky pretínajú, znemožní sa tak korektné vytvorenie implicitne definovanej plochy v zmysle odseku 2.11 a 2.12. Riešením je prečíslovanie vrcholov.



Obr. 35

Rovnako nevhodná situácia nastáva ak editovaním vrcholov pre niektorú z tvoriacich čiar vznikne na nejakom intervale opačná orientácia vrcholov vzhľadom na os x.

Situáciu ilustruje nasledujúci obrázok



Obr. 36

Vidíme, že poradie vrcholov vzhľadom na os x je v tomto prípade 1,2,5,4,3,6,7; teda v okolí bodu 4 je orientácia opačná. Červenou elipsou je označené miesto, kde sa na kostrovej krivke vytvorila problematická situácia. Objekt vzniknutý v tomto mieste nie je varietou.

Odporúčaný počet prechodov (kostrových kriviek) je 2 až 5. Maximálny počet kriviek prechodov je teoreticky obmedzený len veľkosťou kresliacej plochy, avšak pri testovaní sme narazili na fyzické obmedzenie pamäti pracovnej stanice. Ak nám stačí vytvoriť iba zväzok implicitne definovaných plôch CCR, najvyšší testovaný počet prechodov 50 sa bol vypočítaný a zobrazený bez problémov. Pre zväzok vykresľovaný aj s kostrovými čiarami uvedieme zistené obmedzenia neskôr v tabuľke.

Maximálny počet bodov na tvoriacich čiarach je teoreticky obmedzený tiež len veľkosťou kresliacej plochy. Pri testovaní sa však ukázalo vhodné používanie jedno, dvoj a troj segmentových tvoriacich čiar, t.j. max 10 riadiacich vrcholov, vrátane bodov tvoriacich úchopy. Pre situácie s jednou kostrovou krivkou resp. sme testovali až 10 segmentov, výsledky uvedieme neskôr v tabuľke.

Aplikáciu sme testovali na pracovných staniciach s nasledovnými konfiguráciami.

- CPU: Intel DualCore 1,6 GHz, RAM: 2GB, VGA: Intel GMA950
- CPU: Intel DualCore 1,55 GHz, RAM: 2GB, VGA: Intel X3100
- CPU: Intel Pentium4 1,7 GHz, RAM: 512MB, VGA: Intel

5.2. Testy časovej závislosti

Nasledujúca tabuľka zobrazuje časovú závislosť výpočtu (rýchlosť) kompletného zväzku implicitne definovaných plôch CCR v sekundách vzhľadom na počet segmentov tvoriacich čiar a počet kostrových čiar. Červené označené čísla predstavujú hraničné hodnoty, pri ktorých testovaná pracovná stanica vyčerpala systémové prostriedky.

# kostrových čiar	# segmentov						
	1	2	3	4	5	8	10
1	39	62	91	120	117	232	234
2	60	118	185	247	236		
3	87	173	234				
5	144	235					
8	231						
10	235						

tab. 1

Červenými číslami uvádzané hodnoty sú vypočítané bez vykreslenia kompletnej kostry. Pre hodnoty počtu kostrových čiar a segmentov vyššie ako sú červene uvedené v tabuľke červenými číslami teda výsledok neuvádzame, nakoľko metóda výpočtu je odlišná.

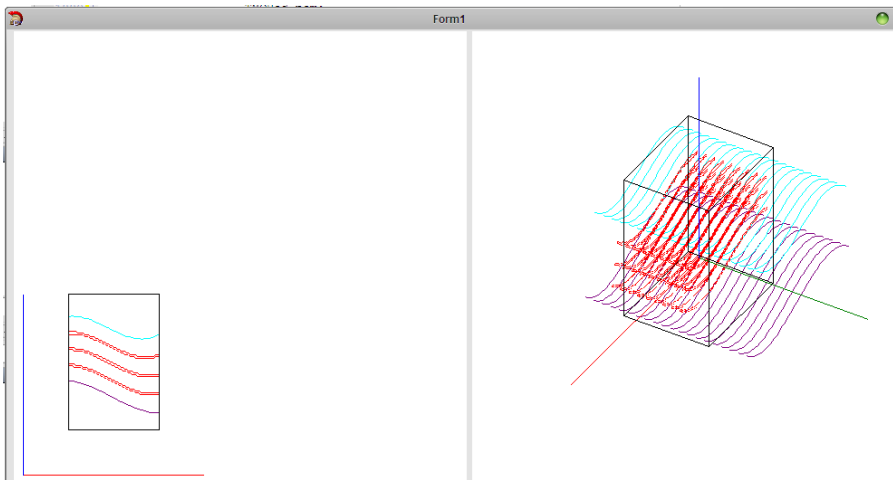
Nasledujúca tabuľka zobrazuje časovú závislosť výpočtu (rýchlosť) v sekundách, pre zväzok implicitne definovaných plôch CCR s jednou kostrovou čiarou vzhľadom na počet metaguľových komponentov na kostrovej čiare a počet segmentov. Počet segmentov 1 a 2 sme testovali ako najpravdepodobnejšie použité varianty.

# m-gulí	# segmentov	
	1	2
10	24	65
20	29	77
40	41	101
50	67	129
100	98	183

tab. 2

Pozn. v tabuľke uvádzané časy pre jednosegmentovú a dvojsegmentovú krivku sa môžu výrazne odlišovať od nameraných v závislosti od tvaru a dĺžky krivky.

Výsledný zväzok implicitne definovaných plôch použitý na rezanie dátového hranola je na nasledujúcom obrázku.



Obr. 37

5.3. Vylepšenia

Vytvorená aplikácia má charakter pokusnej aplikácie. V žiadnom prípade nemožno aplikáciu považovať za dokončené dielo. Aplikácia vznikla hlavne za účelom predvedenia funkčnosti algoritmu vytvárajúceho zväzok plôch CCR a na tento účel je podľa nás postačujúca aj v súčasnej forme. Popri implementovaní základnej funkčnosti sme odhalili ako najvýznamnejšie nasledovné možné vylepšenia, ktoré sme ale neimplementovali.

Pre jednotlivé primitívne objekty m_j sa pri stmelení používa môže využiť váhová funkcia h, ktorej vhodné nastavenie umožňuje lepšie kontrolovať príspevky bodov k funkcii poľa implicitného objektu. Pre body P mriežky algoritmu marching cubes (resp. priesečníkov na stranách pre štvorce siete v algoritme marching cubes) sa tak pre každý primitívny implicitne definovaný objekt nastaví koeficient váhy, s ktorým je násobený celkový súčet.

Konfigurácia aplikácie (napr. nastavenie trajektórie šablónovania, konštanta pre delenie v marching squares algoritme) je v súčasnom stave možná iba cez úpravu zdrojového kódu, preto by bolo vhodné napr. prostredníctvom konfiguračného súboru umožniť tieto parametre zmeniť.

Implementovať viaceré druhy pre tvoriace čiary napr. racionálne Bézierove krivky, NURBS krivky, splajny a dať používateľovi na výber.

Pre zobrazovanie zaujímavou možnosťou vylepšenia je hlavne implementácia iných nových zobrazovacích metód.

V aplikácii je v súčasnosti implementované šablónovanie tvoriacich čiar a kostrových čiar ako celku, t.j. po jednej trajektórii. Ak uvažujeme o šablónovaní každej z tvoriacich čiar po inej trajektórii, dostaneme tak viaceré možnosti vytvorenia rezových plôch.

V literatúre (16), (14), (18) sme našli príklady implementácie metaguľových objektov, ktoré sme použili (t.j. Marching cubes algoritmus + obálka AABB). Rozšírenie, ktoré sme našli a neimplementovali: pri modelovaní metaguľových objektov použiť na rozdelenie priestoru ako dátovú štruktúru Octree (7) .

6. Záver

Pri testovaní a skúmaní vzniknutého zväzku implicitne definovaných plôch sme nadobudli presvedčenie, že v medicínskom zobrazovaní je stále výhodnejšie na vytváranie 3D objektov počítačovej grafiky použiť parametricky zadané objekty. Implicitne definované objekty majú výhody, prinášajú však so sebou aj veľa problémov. V porovnaní s generovaním parametrických objektov vytvorenie implicitne definovaných objektov trvá dlhší čas a je výpočtovo náročnejšie. Na získanie rovnako relevantnej informácie považujeme za nezmyselné používať zložitejšie a náročnejšie prostriedky.
Nami vyvinutý algoritmus a jeho aplikácia nie je vhodná ak očakávame bleskové výsledky.

Kostrové implicitne definované objekty je skutočne možné vytvárať v reálnom čase a upravovaním kostrových prvkov ľahko dosiahnuť tvary ktoré sú s použitím parametrického prístupu ťažko vytvoriteľné. Spojenie kostrových implicitne definovaných objektov so šablónovaním sa tak javí ako stredná cesta medzi modelovaním s parametrickými a implicitne definovanými objektmi.

Ako najväčší problém prechodu od parametricky zadaných objektov ku implicitne definovaným objektom sa javí potreba veľkého množstva bodov pre implicitne definované objekty (rádovo 10^4).

Pre výpočty s veľkým počtom segmentov tvoriacich čiar, resp. veľkým počtom kostrových kriviek treba výkonnú pracovnú stanicu.

V súčasnom stave po základných optimalizáciách viac času, teda vytvorenie zväzku v reálnom čase považujeme za možné, avšak po výraznej optimalizácii od ktorej sme upustili.

Záverom treba uviesť, že aplikácia bola vytvorená len pre účel demonštrácie funkčnosti v práci navrhnutých postupov a isto obsahuje veľa neodladených chýb.

Zoznam použitej literatúry:

1. **SIEMENS.** výstava medicínskej techniky. *Poďte sa pozrieť dovnútra!Vývoj a pokrok v službách medicíny.* Bratislava, 2007.
2. **Kolinský, M.** *Diplomová práca - Objektovo orientovaná knižnica objektov CAGD.* Bratislava, 2000.
3. **Božek, M.** Geometria 1 - Prednášky. 2005.
4. **Polan, M.** Krivky, Plochy. *Internetová učebnica CAGD online.* [Online] 2005.
5. **Zaťko, V.** Krivky v CAGD - prednášky. *Krivky v CAGD - prednášky.* 2006.
6. **Velho, L., Gomes, J. a de Figueiredo, L. H.** *Implicit Objects in Computer Graphics.* New York : Springer-Verlag, 2002.
7. **Samuelčík, M.** Diskrétne geometrické štruktúry. *www.sccg.sk/~samuelcik/dgs.html.* [Online] 2008. http://www.sccg.sk/~samuelcik/dgs/8.pdf.
8. **Bourke, Paul.** Implicit surfaces. *http://local.wasp.uwa.edu.au/~pbourke/miscellaneous/implicitsurf/.* [Online] 1997.
9. **Rehák, M.** Teória a modelovanie šablónovaných objektov. Bratislava, 2001.
10. **Ružický, E. a Ferko, A.** *Počítačová grafika.* Bratislava : Sapientia, 1995.
11. **Lindh, Mats.** Marching cubes. *Østfold University College, Computer Graphics.* [Online] Marec 2003. http://www.ia.hiof.no/~borres/cgraph/explain/marching/p-march.html.
12. **Bloomenthal, J. a Wyvill, B.** *Interactive Techniques for Implicit Modeling.* Palo Alto, California : Xerox PARC, 1995.
13. *Approximate conversion of parametric to implicit surfaces.* **Velho, L. a Gomez, J.** s.l. : Elsevier Science Publishers, 1996. Computer Graphics Forum, 15. Zv. (5), s. 327-338.
14. **Anderson, J.** www.koders.com. *metaball.cpp.* [Online] November 1999. http://www.koders.com/cpp/fidE5BF3E30980A088AD5AA48D6B5C65412F11B11B3.aspx?s=cdef%3Aparser#L37.
15. **Chapell, G. G.** metaball.cpp. *Advanced Computer Graphics.* [Online] 2004. http://www.cs.uaf.edu/~cs481/index.html.
16. **Inc., Impulse.** *Organica - software.* [aplikacia] s.l. : Impulse Inc., 1997.
17. **Kudličková, S.** Zobrazovacie metódy - prednáška. 2005.
18. **Fischer, T. a Jacobs, M.** Metaball Class Reference. *Milton Renderer.* [Online] 2008. http://milton.mjacobs.net/docs/doxygen/classMetaBall.php#02ef538df7aa036a1833efd9948d7cc3.

Necitované materiály:

19. Google. [Online] http://www.google.com.
20. wikipedia. [Online] http://en.wikipedia.org/wiki/Main_Page.
21. **Hejný, M., Zaťko, V. a Kršňák, P.** *Geometria 1.* Bratislava : Slovenské pedagogické nakladateľstvo, 1985.
22. **Sederberg, T. W. a Chen, F.** *Implicitization using Moving Curves and Surfaces.* Brigham : Young University, 1995.
23. *The Displacement Method forImplicit Blending Surfaces in Solid Models.* **Rockwood, Alyn P.** s.l. : ACM Transaction on Graphics , 1989.
24. **Blinn, , James F.** A Generalization of Algebraic Surface Drawing. *ACM Trans. Graph.* 1982, Zv. 1, 3.
25. **Bloomenthal, J.** Implicit surfaces. Seattle : Unchained Geometry, 2002.

26. **Mueller, K.** Visual Medicine Foundations of Medical Imaging. Minneapolis : VIS OS, 2006.
27. **Crespin, B., Blanc, C. a C., Schlick.** *Implicit Sweep Objects.* s.l. : Blackwell Publishers, 1996.
28. *Modelling with Implicit Surfaces that Interpolate.* **Turk, G. a O'Brien, James F.** 4, s.l. : ACM Transactions on Graphics, 2002, Zv. 21.
29. **Shen, J. a Thalmann, D,.** Interactive shape design using metaballs and splines. Lausanne : Swiss Federal Institute of Technology.
30. **Opalach, A. Maddock, S.** An Overview of Implicit Surfaces. Sheffield : Department of Computer Science, The University of Sheffield,.