Rekonstrukcija superelipsoidov iz globinskih slik s pomočjo genetskih algoritmov

Jaka Krivic

DIPLOMSKA NALOGA
IZ
RAČUNALNIŠTVA IN INFORMATIKE

predložena
Fakulteti za računalništvo in informatiko
Univerze v Ljubljani
kot delna izpolnitev pogoja za pridobitev naslova
diplomirani inženir računalništva in informatike

September 1997

Mentor:

prof. dr. Franc Solina

Diplomska naloga je bila izdelana pod mentorstvom prof. Franca Soline in je last Fakultete za računalništvo in informatiko v Ljubljani. Za objavljanje in uporabo rezultatov diplomskega dela je potrebno soglasje zgoraj omenjene ustanove.

Rekonstrukcija superelipsoidov iz globinskih slik s pomočjo genetskih algoritmov

Jaka Krivic

Mentor:
prof. dr. Franc Solina

Povzetek

Diplomsko delo se ukvarja z rekonstrukcijo superelipsoidov iz globinskih slik. Superelipsoidi so tridimenzionalni modeli, določeni s parametri oblike, orientacije in pozicije. Definirani so osnovni in razširjeni superelipsoidi. Globinska slika je slika, pri kateri je v določeni točki podana globina, to je razdalja od gledišča. Prikazana je uporaba genetskih algoritmov pri rekonstrukciji superelipsoidov iz globinskih slik. Podane so različne funkcije ocenjevanja kvalitete rešitev. Napisan je bil program, ki realizira genetski algoritem, kakor tudi funkcije ocenjevanja. Podani so rezultati testiranja na sintetičnih slikah.

Reconstruction of Superellipsoids from Range Images Using Genetic Algorithms

Jaka Krivic

Supervisor:
prof. dr. Franc Solina

Abstract

A method for reconstruction of superellipsoids from a range image is presented. Superellipsoids are 3-D models, determined with parameters of shape, orientation, and position. There is also a definition of extending superellipsoids. A range image is an image produced by 3-D imaging sensors. Recognition is actualy minimization of a fitting function using genetic alghorithms. Several fitting functions are presented. The metod was implemented and its results are shown and discused.

Kazalo

Slike

1

Uvod

Računalniški vid naj bi omogočil strojem inteligentno interakcijo z zunanjim svetom. To predvsem pomeni planiranje poti, razpoznavanje objektov in manipuliranje z njimi. Tako bi bili stroji v svojem delovanju samostojni in uspešni tudi v spreminjajočem se okolju. Ker človek take naloge z lahkoto obvladuje, skušamo računalniški vid razviti po vzoru človekovega, ki pa ga zaradi njegove zapletenosti še ne razumemo popolnoma. Tako se omejujemo le na posamezne omejene naloge in računalniški vid gradimo iz njih. Ena takih nalog je vsekakor rekonstrukcija modelov.

1.1 Rekonstrukcija modelov

Pri računalniškem vidu torej razvijamo metode, ki iz slik izločijo le 'pomembne' informacije. Uporabljamo najrazličnejše vrste slik: intenzitetne, globinske, pare stereo slik itd. Kaj pa so 'pomembne' informacije, je od primera do primera različno. Te so lahko izražene npr. v obliki modelov. Zato želimo pri rekonstrukciji modelov izvedeti ustrezne vrednosti parametrov modela.

Modele, s katerimi predstavljamo $3-D$ točke lahko delimo na lokalne in globalne. Lokalni modeli predstavljajo objekte kot množico nestrukturiranih podatkov, kot so robovi ali površine. Globalni modeli pa poskušajo objekt obravnavati kot celoto v lastnem koordinatnem sistemu. Od slednjih so najpogosteje uporabljeni posplošeni cilindri in supervadriki, podmnožica katerih so superelipsoidi.

1.2 Rekonstrukcija superelipsoidov

Na področju rekonstrukcije superelipsoidov iz globinskih slik je bilo razvitih že kar nekaj metod. Prvi, ki je uporabil superelipsoide v računalniškem vidu, je bil Pentland [1]. Predlagal je rekonstrukcijo z uporabo eksplicitne enačbe. Metoda se razen za nekaj preprostih sintetičnih slik ni obnesla.

Solina in Bajcsy [2] sta rešila problem rekonstrukcije posameznega superelipsoida iz že segmentiranih slik. Rekonstrukcijo sta definirala kot minimizacijo funkcije prileganja z metodo najmanjših kvadratov. Za ta problem nelinearne optimizacije sta uporabila gradientno metodo. Funkcijo prileganja sta definirala na osnovi funkcije znotraj-zunaj.

Nato je bilo razvitih še kar nekaj metod, zato naj omenimo le nekatere. Yokoya s sodelavci [3] je predlagal novo funkcijo prileganja, katera poleg razlike med točkami na modelu in superelipsoidom uporablja še razliko v normalah. Za minimizacijo je uporabil algoritem ohlajanja.

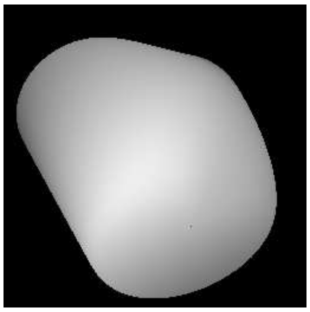
Vidmar [5] je preizkusila rekonstrukcijo superelipsoidov iz kontur.

Callari in Maniscalco [4] sta podala rekonstrukcijo s pomočjo senčenja oz. osvetlitve (shading flow field).

Leonardis [12] je podal segmentacijsko paradigmo 'gradi-in-izberi', v kateri se izmenjujeta segmentacija z rekonstrukcijo. Jaklič [13] jo je uspešno uporabil s superelipsoidi kot modeli.

Bralec najde podrobnejšo predstavitev metod v [6].

1.3 Postavitev problema



Slika 1.1: Primer globinske slike. Sivina točk nam pove razdaljo točke od gledišča: bližje kot je, bolj je svetla.

Globinska slika je v bistvu množica $3-D$ točk v prostoru. Zajemamo jih s t.i. 'range scannerji'. To so naprave, ki merijo razdalje do vidnih površin z različnimi metodami. Primer globinske slike vidimo na sliki 1.1.

Za cilj diplomskega dela smo si zadali sledeče:

- preizkusiti genetske algoritme na problemu rekonstrukcije superelipsiodov iz globinskih slik in
- preizkusiti družino deformacij, ki jih z običajnimi metodami ni mogoče obravnavati.

Delo je sestavljeno takole:

- Drugo poglavje z naslovom *Definicije* nas seznani z definicijo superelipsoidov. Opiše tudi predpostavke, na katerih je diplomsko delo narejeno.
- Tretje poglavje z naslovom *Genetski algoritmi* nam predstavi genetske algoritme kot metodo kombinatorične optimizacije. Opisana je knjižnica *PGAPack* za delo z njimi.

- V četrtem poglavju z naslovom *Rekonstrukcija superelipsoidov z genetskimi algoritmi* je opisana rekonstrukcija superelipsoidov in posebnosti genetskih algoritmov pri tem. Predstavljene so tudi tri različne funkcije ocenjevanja kvalitete rešitev.
- Peto poglavje *Rezultati* podaja rezultate metode pri običajnih in razširjenih superelipsoidih.
- V poglavju *Zaključek* je podan povzetek rezultatov dela. Navedene so tudi možnosti nadaljnjega dela.
- Sledijo še spisek uporabljene literature, *Zahvala* in *Izjava*.

2

Definicije

V tem poglavju bomo predstavili osnovne definicije in pojme, ki jih bomo uporabljali skozi vse diplomsko delo.

2.1 Osnovni superelipsoidi

Kot smo že v uvodu povedali, so superelipsoidi podmnožica družine parametričnih modelov superkvadrov. Definirali bomo vektor površine superelipsoida in funkcijo znotraj-zunaj, ki za dano točko v prostoru pove, ali leži znotraj, zunaj ali na površini superelipsoida. Podali bomo tudi definicije razširjenih superelipsoidov.

2.1.1 Površina superelipsoida

Superelipsoid je definiran z naslednjim 3-D vektorjem:

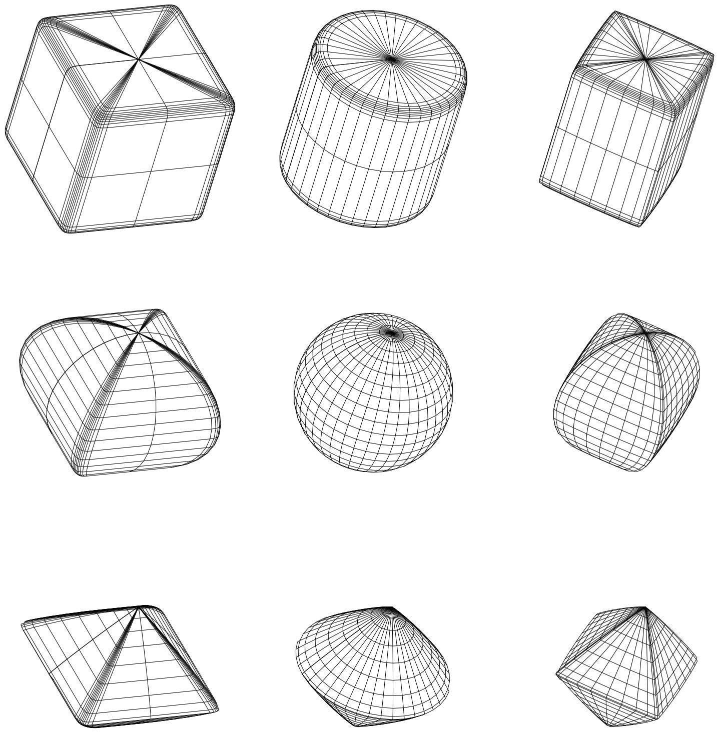
$$\vec{s}(\eta,\omega) = \begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} = \begin{bmatrix} a_1 \cos^{\varepsilon_1}\eta \cos^{\varepsilon_2}\omega \\ a_2 \cos^{\varepsilon_1}\eta \sin^{\varepsilon_2}\omega \\ a_3 \sin^{\varepsilon_1}\eta \end{bmatrix}, \qquad \begin{array}{ccccc} -\frac{\pi}{2} & \leq & \eta & \leq & \frac{\pi}{2} \\ -\pi & \leq & \omega & < & \pi \end{array} \tag{2.1}$$

Vektor površine $\vec{s}$ izhaja iz koordinatnega izhodišča in določa površino superelipsoida. Parameter ω je kot med x osjo in projekcijo vektorja $\vec{s}$ na x-y ravnino, medtem ko je parameter η kot med $\vec{s}$ in njegovo projekcijo na x-y ravnino. V sferičnih koordinatah ω ustreza dolžini (latitude angle), η pa polarni razdalji (longitude angle). Parametri a_1, a_2, a_3 določajo velikost superelipsoida v smereh x, y in z osi. ε_1 in ε_2 sta parametra, ki določata obliko superelipsoida. ε_1 jo določa v smislu z osi, ε_2 pa jo določa v smislu x-y ravnine. Kako vrednosti teh dveh parametrov vplivajo na obliko vidimo na sliki 2.1. Vse oblike dobljene pri $\varepsilon_1, \varepsilon_2 \leq 2$ so konveksne, ostale so konkavne.

Zgoraj navedeni zapis vektorja površine je nekoliko poenostavljen (osnovna notacija [7]). Potenciranje negativne osnove na realen eksponent namreč daje kompleksne rezultate. Temu se izognemo tako, da potenciranje x^y preoblikujemo v $\mathrm{sgn}(x)\,|x|^y$. Pri tem je $\mathrm{sgn}(x)$ funkcija predznaka:

$$\mathrm{sgn}(x) = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}$$

$\varepsilon_2 = 0.1$ $\varepsilon_2 = 1.0$ $\varepsilon_2 = 1.9$



$\varepsilon_1 = 0.1$

$\varepsilon_1 = 1.0$

$\varepsilon_1 = 1.9$

Slika 2.1: Odvisnost oblike superelipsoidov od vrednosti parametrov ε_1 in ε_2 (parametri, ki določajo velikost, so pri vseh superelipsoidih enaki)

V diplomski nalogi bomo uporabljali za preoblikovano potenciranje kar notacijo x^y . Kadar bomo želeli poudariti razliko med običajnim potenciranjem in našim, bo x^y označevalo preoblikovano potenciranje, običajno potenciranje pa bomo označili z $\boxed{x}^y$.

2.1.2 Normale

Vektorji normal igrajo pomembno vlogo pri 3-D geometrijskih telesih (npr. določanje vidne površine telesa). Poglejmo, kako so normale definirane pri superelipsoidih.

Vektor normale izračunamo kot vsak vektor normale ploskve – izračunamo vektorski produkt dveh vektorjev, ki ležita na tangentah na površino: $\vec{n} = \frac{\partial \vec{s}}{\partial \eta} \times \frac{\partial \vec{s}}{\partial \omega}$. Pri tem $\times$ označuje vektorski produkt. *Vektor normale* superelipsoida torej je:

$$\vec{n}(\omega, \eta) = \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} = \begin{bmatrix} \frac{1}{a_1} \cos^{2-\varepsilon_1}\eta \cos^{2-\varepsilon_2}\omega \\ \frac{1}{a_2} \cos^{2-\varepsilon_1}\eta \sin^{2-\varepsilon_2}\omega \\ \frac{1}{a_3} \sin^{2-\varepsilon_1}\eta \end{bmatrix} \tag{2.2}$$

2.1.3 Funkcija znotraj-zunaj

Superelipsoide lahko uvrstimo med *matematična telesa* (mathematical solids), ker prostor razdelijo na tri področja:

1. področje znotraj telesa,
2. področje zunaj telesa,
3. področje meje,

in ker obstaja funkcija znotraj-zunaj, ki za vsako točko pove, v katero od teh treh področij spada.

Po osnovni notaciji [7] se *funkcija znotraj-zunaj* (inside-outside function) glasi:

$$f(\vec{s}) = \left(\left(\frac{s_x}{a_1}\right)^{\frac{2}{\varepsilon_2}} + \left(\frac{s_y}{a_2}\right)^{\frac{2}{\varepsilon_2}} \right)^{\frac{\varepsilon_2}{\varepsilon_1}} + \left(\frac{s_z}{a_3}\right)^{\frac{2}{\varepsilon_1}} \tag{2.3}$$

Točka leži:

1. če $f(\vec{s}) < 1$, znotraj telesa,
2. če $f(\vec{s}) > 1$, zunaj telesa,
3. če $f(\vec{s}) = 1$, na meji telesa.

Pri zgornjem zapisu funkcije znotraj-zunaj moramo opozoriti na pomene posameznih potenciranj. Le-ti izhajajo iz načina izpeljave funkcije znotraj-zunaj – preoblikovanja parametrične enačbe superelipsoida (vektorja površine) v enačbo s kartezičnimi koordinatami [8]. Dosledni zapis funkcije je sledeč:

$$f(\vec{s}) = \left(\left(\boxed{\frac{s_x}{a_1}}^2\right)^{\frac{1}{\varepsilon_2}} + \left(\boxed{\frac{s_y}{a_2}}^2\right)^{\frac{1}{\varepsilon_2}} \right)^{\frac{\varepsilon_2}{\varepsilon_1}} + \left(\boxed{\frac{s_z}{a_3}}^2\right)^{\frac{1}{\varepsilon_1}}$$

2.2 Razširjeni superelipsoidi

Običajne deformacije superelipsoidov kot so ukrivljanje (bending), zvijanje (twisting), priostrevanje (tapering) in dolbenje (cavity deformation) imajo zagotovo to pomanjkljivost, da ne omogočajo lokalnih deformacij. Zato bomo definirali deformacije kot so jih predlagali v [11]. Namesto da sta parametra ε_1 in ε_2 konstantna, ju definiramo kot funkciji ω in η:

$$\begin{aligned} \varepsilon_1 &= \varepsilon_1(\omega, \eta), \\ \varepsilon_2 &= \varepsilon_2(\omega, \eta) \end{aligned}$$

Tako definirane modele bomo imenovali *razširjeni superelipsoidi* (extending superquadrics). Vidimo lahko, da so osnovni superelipsoidi poseben primer razširjenih superelipsoidov pri konstantnih $\varepsilon_1(\omega, \eta)$ in $\varepsilon_2(\omega, \eta)$.

2.2.1 Funkcije eksponentov

Glede na to, da so eksponenti v enačbi 2.1 pozitivni, morajo imeti funkcije eksponentov zalogo vrednosti $(0, +\infty)$. Da bi zagotovili zveznost površine superelipsoida, morajo biti tudi zvezne. Če pogledamo enačbo 2.1, opazimo, da vrednosti $\omega = -\pi$ in $\omega = \pi$ določajo isto točko. Za zveznost površine v točkah $\omega = -\pi, \pi$ mora torej veljati še pogoj [11]:

$$\begin{aligned} \varepsilon_1(-\pi, \eta) &= \varepsilon_1(\pi, \eta) \\ \varepsilon_2(-\pi, \eta) &= \varepsilon_2(\pi, \eta) \end{aligned} \tag{2.4}$$

Hoteli smo preiskusiti katere funkcije bi bile najboljše in smo tako preverili dve družini: eksponentne in polinomske. Pri obeh smo uporabili enako število parametrov.

Polinomske funkcije za ε_1 in ε_2

V [11] so za funkcije $\varepsilon_1(\omega, \eta)$ in $\varepsilon_2(\omega, \eta)$ predlagali kar polinome nad ω in η. Pri uporabi polinomskih funkcij zadostimo pogoju 2.4 npr. tako, da uporabimo samo sode potence parametra ω v funkcijah $\varepsilon_1(\omega, \eta)$ in $\varepsilon_2(\omega, \eta)$. Funkciji smo definirali tako:

$$\begin{aligned} \varepsilon_1(\omega, \eta) &= a_0 + a_1\omega^2 + a_2\omega^4 + \sum_{i=1}^{i=3} b_i\eta^i \\ \varepsilon_2(\omega, \eta) &= c_0 + c_1\omega^2 + c_2\omega^4 + \sum_{i=1}^{i=3} d_i\eta^i \end{aligned}$$

Pri tem so a_i, b_i, c_i in d_i parametri funkcij in tako tudi parametri superelipsoida. Nastopi pa problem, kako omejiti te parametre, da sta funkciji $\varepsilon_i(\omega, \eta)$ veljavni. Če se namreč omejimo na konveksne supereliposide mora veljati $0 < \varepsilon_i < 2$, to pa z omejitvami parametrov a_i, b_i, c_i in d_i težko dosežemo.

Eksponentne funkcije za ε_1 in ε_2

Preizkusili pa smo tudi družino eksponentnih funkcij. Definirali smo jih tako:

$$\begin{aligned} \varepsilon_1(\omega, \eta) &= b_0 + b_1 e^{b_2\omega^2} + b_3 e^{b_4\eta + b_5\eta^2} \\ \varepsilon_2(\omega, \eta) &= c_0 + c_1 e^{c_2\omega^2} + c_3 e^{c_4\eta + c_5\eta^2} \end{aligned}$$

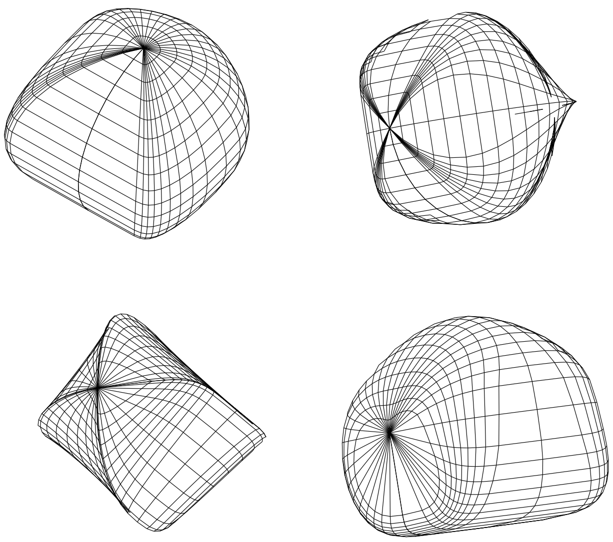
Lahko se prepričamo, da je pogoj 2.4 izpolnjen. Na sliki 2.2 vidimo nekaj primerov modelov, kjer smo uporabili te funkcije.

Parametra b_0 in c_0 pomenita nekako konstantno obliko, ki se spreminja z b_2 in c_2 v smislu $x - y$ ravnine, in z b_4, b_5, c_4 in c_5 v smislu z osi. Parametri b_1 in c_1 ter b_3 in c_3 pa pomenijo velikostni red spreminjanja oblike.

Odločili smo se, da uporabimo to družino funkcij in kjer sta v nadaljnem omenjeni $\varepsilon_1(\omega, \eta)$ in $\varepsilon_2(\omega, \eta)$ sta s tem mišljeni tako definirani funkciji.

2.2.2 Normale

Kot rečeno, dobimo normalo v dani točki tako, da izračunamo vektorski produkt $\frac{\partial \vec{s}}{\partial \eta} \times \frac{\partial \vec{s}}{\partial \omega}$. Tako jo tudi numerično izračunamo, saj se je ne da zapisati v dosti lepši obliki.



Slika 2.2: Razširjeni superelipsoidi z eksponentnimi funkcijami $\varepsilon_1(\omega, \eta)$ in $\varepsilon_2(\omega, \eta)$

2.2.3 Funkcija znotraj-zunaj

Funkcijo znotraj-zunaj dobimo iz eksplicitnih enačb tako, da se z uporabo lastnosti $\sin^2\alpha + \cos^2\alpha = 1$ (glej [8]) znebimo parametrov ω in η. Pri razširjenih superelipsoidih pa stvari niso tako enostavne. Ker ε_1 in ε_2 nista konstantna, temveč funkciji ω in η, se parametrov ω in η ne moremo znebiti. Funkcijo znotraj-zunaj tako zapišemo [11]:

$$f(\vec{s}) = \left(\left(\frac{s_x}{a_1}\right)^{2/\varepsilon_2(\omega,\eta)} + \left(\frac{s_y}{a_2}\right)^{2/\varepsilon_2(\omega,\eta)} \right)^{\varepsilon_2(\omega,\eta)/\varepsilon_1(\omega,\eta)} + \left(\frac{s_z}{a_3}\right)^{2/\varepsilon_1(\omega,\eta)} \tag{2.5}$$

Pri podani točki v prostoru, izračunamo začetna približka za ustrezajoča kota ω in η, nato pa iterativno s pomočjo enačb 4.2 izračunamo prava kota *omega* in *eta*. S podanima kotoma se nato izračuna funkcija znotraj-zunaj 2.5.

2.3 Superelipsoid v splošni legi

Dosedanje definicije (vektor površine in funkcija znotraj-zunaj) so bile zapisane s koordinatami lokalnega koordinatnega sistema. Če želimo določiti lego superelipsoida v prostoru, moramo poznati transformacijo, ki lokalni koordinatni sistem pretvori v svetovni koordinatni sistem. Ta transformacija je sestavljena iz rotacije v lokalnem koordinatnem sistemu in premika izhodišča lokalnega koordinatnega sistema v izhodišče svetovnega.

Rotacijo izvedemo z ZYZ-Eulerjevo rotacijo. Trije Eulerjevi koti φ, ϑ in ψ popolnoma določajo orientacijo transformiranega koordinatnega sistema. Kot φ pomeni rotacijo okoli z osi, kot ϑ okoli nove y osi in kot ψ okoli nove z osi. Izračunajmo *rotacijsko matriko* R:

$$R = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\vartheta & 0 & \sin\vartheta \\ 0 & 1 & 0 \\ -\sin\vartheta & 0 & \cos\vartheta \end{bmatrix} \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} \cos\varphi\cos\vartheta\cos\psi - \sin\varphi\sin\psi, & -\sin\varphi\cos\vartheta\cos\psi - \cos\varphi\sin\psi, & \sin\vartheta\cos\psi \\ \cos\varphi\cos\vartheta\sin\psi + \sin\varphi\cos\psi, & -\sin\varphi\cos\vartheta\sin\psi + \cos\varphi\cos\psi, & \sin\vartheta\sin\psi \\ -\cos\varphi\sin\vartheta, & \sin\varphi\sin\vartheta, & \cos\vartheta \end{bmatrix}.$$

Vektor površine $\vec{s}'$ superelipsoida, zapisan z globalnimi koordinatami, dobimo tako, da vektor površine, zapisan z lokalnimi koordinatami, z leve pomnožimo z rotacijsko matriko R in prištejemo *vektor premika* $\vec{p} = [p_x, p_y, p_z]^T$:

$$\vec{s}' = R\vec{s} + \vec{p}$$

Če želimo zapisati funkcijo znotraj-zunaj f' v svetovnem koordinatnem sistemu, moramo svetovni koordinatni sistem pretvoriti v lokalni koordinatni sistem. Funkcija znotraj-zunaj zapisana s svetovnimi koordinatami, torej je:

$$f'(\vec{s}') = f(R^{-1}(\vec{s}' - \vec{p}))$$

Za vsako rotacijsko matriko A velja $A^{-1} = \det(A)A^T$ [10]. Ker velja $\det(R) = 1$, je $R^{-1} = R^T$. Končna oblika funkcije znotraj-zunaj je:

$$f'(\vec{s}') = f(R^T(\vec{s}' - \vec{p}))$$

Če povzamemo, superelipsoid v splošni legi je določen z obliko, orientacijo in premikom:

$$\mathcal{P} = \langle a_1, a_2, a_3, \varepsilon_1, \varepsilon_2, \varphi, \vartheta, \psi, p_x, p_y, p_z \rangle$$

Razširjeni suprelipsoidi imajo namesto ε_1 in ε_2 parametre za obliko b_i in c_j:

$$\mathcal{P}' = \langle a_1, a_2, a_3, \{b_i\}, \{c_j\}, \varphi, \vartheta, \psi, p_x, p_y, p_z \rangle$$

2.4 Predpostavke

Diplomska naloga je zgrajena na nekaterih predpostavkah, ki v določeni meri poenostavijo reševanje zastavljenega problema.

2.4.1 Projekcija

V splošnem sta možni dve projekciji – perspektivna in paralelna. Zaradi večje enostavnosti (pri perspektivni projekciji moramo poznati oz. določiti tudi pozicijo gledišča), smo se odločili za paralelno projekcijo.

Uporabili bomo tako imenovano *šibko paralelno projekcijo* (weak parallel projection). To je različica paralelne projekcije, pri kateri je vektor smeri gledanja $\vec{v}$ konstanten: $\vec{v} = [0, 0, 1]^T$. To projekcijo lahko uporabimo, ker z rotacijsko matriko R lahko dosežemo poljubno orientacijo superelipsoida, s tem pa lahko tudi pri konstantnem $\vec{v}$ dobimo vse možne poglede na telo.

Kadarkoli je v našem delu naveden izraz projekcija, je s tem mišljena šibka paralelna projekcija.

2.4.2 Omejitve parametrov superelipsoida

Parametre smo zaradi lažjega dela omejili in sicer:

- parametre translacije glede na podano globinsko sliko velikosti $X \times Y$ in maksimalno globino Z:

$$\begin{array}{ccccc} 0 & < & p_x & < & X \\ 0 & < & p_y & < & Y \\ 0 & < & p_z & < & Z \end{array}$$

- parametre rotacije:

$$-\pi \quad < \quad \varphi, \vartheta, \psi \quad < \quad \pi$$

- parametre velikosti podobno kot pri parametrih translacije
- parametre oblike v dveh primerih:
  1. pri osnovnih superelipsoidih $0.1 \geq \varepsilon_1, \varepsilon_2 \geq 1.9$
  2. pri razširjenih superelipsoidih

$$\begin{array}{cccc} 0.1 & \geq & b_0, c_0 & \geq & 1.9 \\ -2 & \geq & b_1, b_3, b_4, c_1, c_3, c_4 & \geq & \\ -2 & \geq & b_2, b_5, c_2, c_5 & \geq & 0 \end{array}$$

  Ker pa se s tako omejitvijo ne izognemo vrednostim funkcije manjšim od 0.1 in večjim od 1.9 funkciji priredimo tako, da nam v teh primerih vrneta kar ti dve vrednosti.

2.4.3 Slika enega telesa

Predpostavljamo, da točke na dani globinski sliki pripadajo le enemu telesu – superelipsoidu. S tem se omejimo na problem razpoznavanja, izognemo pa se problemu segmentacije, ki ga je v splošnem najbrž potrebno rešiti skupaj z rekonstrukcijo oblike ([12], [13]).

3

Genetski algoritmi

Genetski algoritmi so preiskovalni algoritmi, ki temeljijo na načelih evolucije in genetike. Njihovo bistvo je simulirana evolucija množice rešitev danega problema. Tej množici pravimo *populacija*, posameznim rešitvam pa *osebki*. S pomočjo genetske rekombinacije in mutacije tvorimo iz starih osebkov nove. Kvaliteto rešitev ocenjujemo s funkcijo uspešnosti, ki jo uporabimo pri reprodukciji. Boljšim osebkom dajemo prednost in tako dobivamo čedalje boljše rešitve problema.

Genetske algoritme bomo v našem primeru uporabljali kot optimizacijsko metodo, lahko pa jih obravnavamo tudi kot preiskovalno metodo ali metodo avtomatskega učenja. Uporabni so na mnogih področjih [9]. Odlikujeta jih enostavnost in robustnost, pri reševanju problemov pa potrebujejo le ustrezno kodiranje rešitev in oceno njihove uspešnosti, ne pa tudi drugih informacij o problemu. Ta lastnost je v našem primeru odločilno vplivala na izbiro metode za minimizacijo, ker ravno iz tega vzroka odpadejo običajne metode, kot je naprimer Levenberg–Marquardtova.

Prvi korak pri reševanju problema z genetskim algoritmom je kodiranje rešitev, kar je tudi osnova za njihovo kombiniranje. Če hočemo kombinirati reštive, jih moramo obravnavati sočasno. To ima pred metodami, ki obravnavajo le eno rešitev, prednost v tem, da je manjša možnost pristanka v lokalnem minimumu.

Drugo, kar potrebujemo, je ocena kvalitete rešitev, metoda pa ne zahteva nobenih drugih informacij o problemu. Te je včasih težko ali pa celo nemogoče dobiti. V našem primeru so to odvodi funkcije prileganja, ki jih sicer lahko izračunamo le za osnovne, ne pa tudi za razširjene superelipsoide.

3.1 Enostavni genetski algoritem

Oglejmo si preprosto različico genetskega algoritma. Njegovo delovanje je presenetljivo enostavno, ker pa vsebuje vse osnovne elemente genetskega algoritma, je z njim mogoče reševati mnoge naloge.

Točke v prostoru, ki ga preiskujemo, naj bodo predstavljene z nizi. Tem pravimo *osebki*. Znake v nizu imenujemo *geni*. Vsaki točki v prostoru pripada vrednost kriterijske funkcije, to je funkcije, ki oceni kvaliteto osebka. Genetski algoritem s to funkcijo ne operira neposredno, temveč jo pretvori v t.i. *funkcijo uspešnosti*, s katero ocenjuje osebke. Za funkcijo uspešnosti zahtevamo, da je pozitivna in da osebkom, ki predstavljajo boljšo rešitev, priredi višjo uspešnost. Pri minimizacijskih problemih to pomeni, da se osebkom z manjšo vrednostjo kriterijske funkcije priredi večjo vrednost funkcije uspešnosti. Tej zadnji zahtevi je moč zadostiti z linearno transformacijo kriterijske funkcije v funkcijo uspešnosti.

Množici osebkov pravimo *populacija*. Osebke začetne populacije največkrat generiramo kar naključno, lahko pa uporabimo rešitve, ki jih dobimo s kako drugo metodo. Iz osebkov trenutne populacije *staršev* generiramo naslednike, ki pripadajo novi populaciji. Torej procesiramo v korakih, ki jim pravimo *generacije*. Velikost populacije se pri tem ne spreminja. Za generiranje osebkov nove generacije uporabljamo različne operatorje. V enostavnem genetskem algoritmu so trije: *reprodukcija*, *križanje* in *mutacija*.

Postopek traja, dokler ni izpolnjen ustavitveni pogoj. Ta je lahko podan s številom generacij, časom izvajanja, kvaliteto rešitev, njihovo konvergenco ali s kombinacijo teh pogojev. Najpogosteje predpišemo kar število generacij.

Za rezultat štejemo najboljšo rešitev, ki jo algoritem najde med izvajanjem. Algoritem je podan kot Algoritem 1.

Algoritem 1 (Enostavni genetski algoritem)

```
EnostavniGA(g, n, p_c, p_m, rešitev);
begin
    generacija := 0;
    Generiraj(populacija, n);
    Ovrednoti(populacija, povprečna_uspešnost, najboljši_osebek);
    rešitev := najboljši_osebek;
    repeat
        generacija := generacija + 1;
        Reprodukcija(povprečna_uspešnost, populacija, nova_populacija);
        Križanje(nova_populacija, p_c);
        Mutacija(nova_populacija, p_m);
        Ovrednoti(nova_populacija, povprečna_uspešnost, najboljši_osebek);
        if uspešnost(najboljši_osebek) > uspešnost(rešitev)
        then rešitev = najboljši_osebek;
        populacija := nova_populacija;
    until ustavitveni_pogoj(g, generacija, rešitev);
end
```

Pri tem g pomeni predpisano število generacij, n velikost populacije, p_c verjetnost križanja in p_m verjetnost mutacije.

3.1.1 Operatorji

V enostavnem genetskem algoritmu nastopajo trije operatorji: reprodukcija, križanje in mutacija. Reprodukcija izvaja selekcijo med osebki, tako da preživijo uspešnejši in odmrejo slabši osebki. Večja kot je uspešnost osebka, večja je verjetnost, da bo ta osebek prispeval določeno število potomcev v naslednjo generacijo. Ker se velikost populacije ne spreminja, morajo torej slabši osebki odmreti. Pričakovano število potomcev osebka definiramo kot razmerje med njegovo in povprečno uspešnostjo populacije. Reprodukcijo lahko potem implementiramo različno: osebek lahko pomnožimo v številu, ki je enak celemu delu pričakovanega števila, ostanek pa generiramo naključno. Temu pravimo *deterministična selekcija.* Pri izvedbi z 'ruleto' potomce določamo naključno. Verjetnost, da osebek da potomca je premosorazmerna njegovi uspešnosti. Kombinacija obeh izvedb je *selekcija s stohastičnim ostankom*, pri kateri se uporabi najprej determinističtna selekcija, nad ostankom pa 'ruletna' selekcija. Naj omenimo še, da glede reprodukcije obstajata dve vrsti genetskih algoritmov. Prvi, generacijski (*generational replacement genetic algorithm, GRGA*), vsako generacijo zamenja celotno populacijo in je tradicionalni genetski algoritem [14]. Drugi, t.i. *steady-state genetic algorithm (SSGA)*, pa vsako generacijo zamenja le nekaj osebkov in je novejše odkritje [15].

Križanje in mutacija variirata osebke. Križanje poteka tako, da naključno izberemo starševska niza. Nato naključno določimo mesto križanja in od tega mesta naprej zamenjamo sufiksa nizov. Tak način križanja je *enomestno* križanje. Pri dvomestnem se zamenja podniz, ki leži med dvema naključno izbranima mestoma križanja. Uporabljajo pa se tudi variante, pri katerih se zamenja več podnizov. Pri *uniformnem križanju* se za vsak par istoležnih genov sproti naključno odločimo za zamenjavo. Operator križanja se ponavadi uporablja z verjetnostjo p_c, ki je parameter genetskega algoritma.

Mutacija deluje tako, da naključno spreminja vrednost genov. Nanjo lahko gledamo kot na naključno preiskovanje prostora. Ponavadi jo uporabljamo z zelo majhno verjetnostjo p_m, ki je verjetnost, da nekemu genu spremenimo vrednost.

Različne izvedbe imajo tudi druge genetske operatorje, naprimer inverzijo, ki genu določi "obratno" vrednost. Na to pa lahko gledamo tudi kot na poseben primer mutacije.

3.2 PgaPack knjižnica za delo z GA

Pri sami implementaciji genetskega algoritma smo uporabili knjižnico PgaPack za delo z genetskimi algoritmi, zato naj jo na kratko opišemo. Dosegljiva je na ftp strežniku `ftp.mcs.anl.gov` kot `pub/pgapack/pgapack.tar.Z`. Paket vsebuje izvorno kodo, navodila za instalacijo, navodila za uporabo in primere. Napisana je v ANSI C in podpira tudi paralelno izvajanje. Na tem mestu jo bomo le na kratko opisali, za podrobnejši opis pa naj si bralec ogleda [16].

PgaPack je knjižnica, ki je nevtralna glede na stukturo podatkov, kar pomeni, da zmožnost skrivanja podatkov daje uporabniku celotno funkcionalnost ne glede na tip podatkov. Osnovne podatkovne tipe obvlada brez spreminjanja, brez težav pa se jo spremeni za kakršnekoli podatke.

PgaPack omogoča več nivojev uporabe. Najenostavnejši je tisti, pri katerem uporabnik samo nastavi parametre genetskega algoritma (različne verjetnosti, kriterijsko funkcijo, smer optimizacije...). Na naslednjem nivoju uporabnik sam eksplicitno kliče funkcije, ki so neodvisne od podatkovne strukture (selekcijo, križanje, mutacijo). To je uporabno, če želimo več kontrole nad samimi koraki v genetskem algoritmu ali pa želimo tega hibridizirati s kakšno hevristiko. Nadalje omogoča, da spremenimo algoritem tako, da podamo lastne funkcije za nek(aj) operator(jev), pri čemer pa še vedno uporabljamo osnovne podatkovne tipe. Če vse to združimo, dobimo še zadnji nivo, pri katerem si uporabnik definira podatkovni tip, napiše funkcije za operatorje nizkega nivoja, ki so odvisni od podatkovnega tipa, in jih izvaja s funkcijami višjega nivoja PgaPack-a.

PgaPack zahteva od uporabnika, da poda kriterijsko funkcijo, ki jo potem sam preslika v funkcijo uspešnosti. Uporabnik ima na voljo tri možnosti preslikave: identiteto, linearno rangiranje in linearno normalizacijo. Za minimizacijo se ta vrednost potem še odšteje od najslabše vrednosti kriterijske funkcije v populaciji ali pa se vzame njeno obratno vredost.

3.2.1 Možnosti

Na hitro bomo preleteli možnosti, ki jih PgaPack omogoča. Te so namenjene tako uporabi z osnovnimi kot tudi z uporabniško defniranimi podatkovnimi tipi.

PgaPack ima ločeno reprodukcijo od selekcije. Omogoča GRGA in SSGA s pomočjo parametra, ki določa, kolikšen del populacije se zamenja. Pri selekciji ima uporabnik na voljo vse že omenjene možnosti. Križanje je lahko uniformno, enomestno in dvomestno. Vrsta mutacije je odvisna od podatkovnega tipa. Na primer pri binarnih nizih se gen komplementira, pri realnih pa se vrednost gena spremeni za naključno vrednost z določeno verjetnostno porazdelitvijo. PgaPack ima definiran tudi t.i. restart operator, ki vsako n-to generacijo populacijo ponovno generira z osebki, ki so mutacije najboljšega predstavnika prejšnje generacije. Knjižnjica pa ima tudi mnogo dodatnih funkcij in pripomočkov, ki uporabniku olajšajo delo.

4

Rekonstrukcija superelipsoidov z genetskimi algoritmi

Rekonstrukcija superelipsoida iz globinske slike pomeni določanje superelipsoida, ki se prilega objektu na podani globinski sliki. Z določanjem superelipsoida imamo v mislih določanje vrednosti parametrov $\mathcal{P}$, s katerimi je superelipsoid v splošni legi natanko določen.

Določanje parametrov superelipsoida je v bistvu problem modeliranja podatkov – na voljo imamo množico podatkov (točk globinske slike), za katere moramo ugotoviti, kateremu od razpoložljivih modelov (superelipsoidov) pripadajo. Standardni pristop reševanja pri modeliranju je sledeč:

1. Določiti je potrebno *funkcijo prileganja* $g(\mathcal{T};\vec{p})$ (fitting function), ki pove kako dobro se določen model prilega vhodnim podatkom. Njene vrednosti so tem manjše, čim bolj se model približuje rešitvi. $\mathcal{T}$ je množica 3–D točk iz globinske slike in $\vec{p}$ parametri superelipsoida.

2. Izvedemo minimizacijo funkcije prileganja in dobimo *parametre najboljšega prileganja* (best–fit parameters).
$$\min_{\vec{p}} \; g(\mathcal{T};\vec{p})$$

Pri reševanju optimizacijskih problemov z genetskimi algoritmi sta potrebni dve stvari: kodiranje rešitev in funkcija prileganja. Ne potrebujemo pa nobenih drugih informacij o problemu, kot so npr. različni odvodi funkcije prileganja. To je bil tudi odločilen razlog, da smo se odločili za minimizacijo z genetskimi algoritmi, saj odvodi različnih funkcij prileganja za razširjene superelipsoide niso analitično izračunljivi. Zaradi tega odpadejo standardne metode za minimizacijo.

4.1 Kodiranje rešitev

Pri rekonstrukciji superelipsoida iz globinske slike iščemo superelipsoid, ki se najbolje prilega točkam iz slike. Rešitev je torej superelipsoid, ki je določen s parametri translacije, rotacije, velikosti in oblike. Ker so ti parametri vsi realni, smo se odločili, da superelipsoid zakodiramo z enajstimi oz. enaindvajsetimi geni (po tri za translacijo, rotacijo in velikost ter za obliko pri osnovnih dva, pri razširjenih pa deset). Geni so realna števila.

4.2 Funkcije prileganja

Kot rečeno, je potrebno določiti kodiranje rešitev in oceno njihove uspešnosti. To zadnje pa je ponavadi tudi najpomembnejše. Predstavili in preizkusili smo v grobem tri.

4.2.1 Funkcija prileganja na osnovi funkcije znotraj-zunaj

Funkcija temelji na funkciji znotraj-zunaj. Izračuna se povprečje kvadratov razdalj od točk iz globinske slike do modela superelipsoida. Definirana je tako:

$$\begin{aligned} g_1(\mathcal{T};\vec{p}) &= \tfrac{1}{n}\sum_{\vec{t}'\in\mathcal{T}} (|\vec{t}|(1-f(\vec{t})))^2 \\ n &= |\mathcal{T}| \\ \vec{t} &= R^T(\vec{t}'-\vec{p}) \end{aligned} \tag{4.1}$$

$\vec{t}'$ je torej točka globinske slike, $\vec{t}$ pa ta točka zapisana z lokalnimi koordinatami.

Ta funkcija ima to pomanjkljivost, da ne upošteva točk na modelu, ki so segale čez oblak točk na globinski sliki. Zato so se razvijali večji modeli, ki pa so se lokano dobro prilegali na točke globinske slike.

4.2.2 Funkcija prileganja na osnovi razlike v globini

Bistvo druge je izračun globinske slike modela in njena primerjava z dano globinsko sliko. Izračuna se vsota razlik v globini na nekaj točkah modela do ustreznih točk na globinski sliki (tistih z istima kooorditatama x in y). Te točke naj bi bile čimbolj enakomerno razporejene po superelipsoidu. Izračunamo jih tako, da je kot med lokalnima vektorjema, ki pripadata sosednjima točkama, konstanten. Če označimo z ω' kot med x osjo in projekcijo vektorja na x-y ravnino ter z η' kot med to projekcijo in vektorjem, dobimo parametra ω in η, ki določata točko na superelipsoidu s tem vektorjem:

$$\begin{aligned} \frac{\sin(\omega)}{\cos(\omega)} &= \frac{a_1 \sin^{\frac{1}{\varepsilon_2}}\omega'}{a_2 \cos^{\frac{1}{\varepsilon_2}}\omega'} \\ \frac{\sin(\eta)}{\cos(\eta)} &= \frac{a_3 \sin^{\frac{1}{\varepsilon_1}}\eta'}{a_2 \cos^{\frac{1}{\varepsilon_1}}\eta'} \end{aligned} \tag{4.2}$$

Če želimo na superelipsoidu izračunati $m \times n$ točk, določimo nabor vrednosti tako:

$$\begin{aligned} \omega' &= -\pi + k * \tfrac{2\pi}{m}, \quad k &= 0,1,...m-1 \\ \eta' &= -\tfrac{\pi}{2} + l * \tfrac{\pi}{n}, \quad l &= 0,1,...n-1 \end{aligned}$$

Množico točk $\mathcal{C}$, v katerih primerjamo globino, dobimo tako, da za vsak par vrednosti (ω', η') izračunamo iz gornjih enačb ω in η, iz teh pa točko superelipsoida v globalnih koordinatah. Če je točka vidna, (če je kot med vektorjem gledanja in normalo manjši od $\frac{\pi}{2}$) točko uvrstimo v $\mathcal{C}$, sicer ne.

Funkcija prileganja je potem vsota kvadratov razlik v globini med točkami iz C in ustreznimi točkami iz globinske slike. Če ustrezne točke na globinski sliki ni (temu ustreza točka v neskončnosti, se model kaznuje z vrednostjo K, ki je večja od največje možne razlike v globini. Ker ta funkcija nagrajuje zelo majhne modele, ki se prilegajo le nekaj točkam iz globinske slike, smo to vrednost potem še delili s kvadratnim korenom med seboj pomnoženih velikostnih parametrov, da bi spodbujali nastanek večjih modelov.

$$g_2(\mathcal{T};\vec{p}) = \frac{1}{\sqrt{a_1 a_2 a_3}} \sum_{\vec{c}\in\mathcal{C}} \begin{cases} K, & v\ globinski\ sliki\ ni\ točke\ [c_x, c_y, t_z]^T \\ (t_z - c_z)^2, & sicer \end{cases} \tag{4.3}$$

Ta ocena je odpravila problem prevelikih modelov, vendar se pri njej pojavi drug problem: če se upošteva več točk, je čas procesiranja mnogo predolg, če pa se izračuna manj točk, ima funkcija zaradi neupoštevanja velikega števila točk iz globinske slike več in globlje lokalne minimume, da bi bila praktično uporabna. Preizkusili smo še različico, pri kateri se manjkajoče točke interpolirajo iz izračunanih, vendar se zaradi numeričnih napak pri interpolaciji ta ni obnesla.

4.2.3 Kombinirana funkcija prileganja

Tretja funkcija prileganja je v bistvu kombinacija prvih dveh. Prvo oceno se poslabša za faktor, ki ga izračuna malo spremenjena druga. Če točka na modelu ustreza neskončni točki na globinski sliki, se faktor poveča za določeno konstanto (nekaj stotink). Ta ocena se je najbolje obnesla, saj združuje dobre lastnosti prve in druge.

Funkcija prileganja je takale:

$$\begin{aligned} g_3(\mathcal{T};\vec{p}) &= g_1(\mathcal{T};\vec{p}) * g_2'(\mathcal{T};\vec{p}) \\ g_2'(\mathcal{T};\vec{p}) &= 1 + \sum_{\vec{c}\in\mathcal{C}} \begin{cases} K', & v\ globinski\ sliki\ ni\ točke\ [c_x, c_y, t_z]^T \\ 0, & sicer \end{cases} \end{aligned} \tag{4.4}$$

Za najboljše so se pokazale vrednosti konstante K' od 0.02 do 0.08.

4.3 Posebnosti pri rekonstrukciji superelipsoidov

Pri testiranju se je genetski algoritem večkrat ustavil v lokalnem minimumu na način, kot ga podaja primer na sliki 5.3. To pomeni, da so bile zamenjane osi x in z oz. y in z, ustrezni velikostni parametri in parametra za obliko. Algoritem v večini primerov ni našel poti iz lokalnega minimuma, zato smo mutaciji dodali zmožnost, da zamenja osi.

Taka mutacija mora spremeniti parametre rotacije tako, da se osi x (oz. y) in z zamenjajo. Iz 'starih' parametrov φ, ϑ in ψ moramo dobiti nove φ', ϑ' in ψ'. Te dobimo tako, da 'stari' rotacijski matriki R zamenjamo ustrezne stolpce (ki so v bistvi koordinate zarotiranih osi superelipsoida) in jih primerjamo z 'novo' R'. Če osi (x, y, z) preslikamo v (z, x, y), dobimo:

$$\begin{aligned}
\frac{\sin(\varphi')}{\cos(\varphi')} &= \frac{-\cos(\varphi)\sin(\vartheta)}{-\cos(\vartheta)} \\
\frac{\sin(\vartheta')}{\cos(\vartheta')} &= \frac{-\sin(\psi)\sin(\varphi)\cos(\vartheta)+\cos(\psi)\cos(\varphi)}{\sin(\psi')\sin(\varphi)\sin(\vartheta)} \\
\frac{\sin(\psi')}{\cos(\psi')} &= \frac{-\sin(\psi)\sin(\varphi)\cos(\vartheta)+\cos(\psi)\cos(\varphi)}{-\cos(\psi)\sin(\varphi)\cos(\vartheta)-\sin(\psi)\cos(\varphi)}
\end{aligned}$$

Podobno dobimo nove kote za preslikavo osi (x, y, z) v (y, z, x).

Ko smo določili nove kote, moramo le še zamenjati velikostne parametre in parametre, ki določajo obliko. Z a_i označimo stare, z a_i' pa nove velikostne parametre. Imamo dva primera:

1. osi (x, y, z) se preslikajo v (z, x, y):

$$a_1' = a_3,\ a_2' = a_1,\ in\ a_3' = a_2$$

2. osi (x, y, z) se preslikajo v (y, z, x):

$$a_1' = a_2,\ a_2' = a_3,\ in\ a_3' = a_1$$

Parametre za obliko pri osnovnih superelipsoidih enostavno zamenjamo (${\varepsilon_1}' = \varepsilon_2$, ${\varepsilon_2}' = \varepsilon_1$), pri razširjenih pa zamenjamo istoležne parametre ($b_i' = c_i$, $c_i' = b_i$; $i = 0...5$).

5

Rezultati

Genetske algoritme smo testirali na sintetičnih slikah. Slike smo generirali preprosto tako, da smo za neke parametre superelipsoida izračunali točke s parametroma ω in η, katerih vrednosti smo enakomerno razporedili po intervalih (zalog vrednosti) dovolj na gosto.

V poglavju 5.1 je opisana rekonstrukcija osnovnih superelipsoidov. Naš namen je bil, da preverimo genetske algoritme kot metodo rekonstrukcije in dobimo parametre genetskega algoritma in funkcijo prileganja, ki se najbolje obnesejo. S tem smo hoteli zagotoviti, da bi rekonstrukcija razširjenih superelipsoidov potekala na 'zdravih' temeljih. Ta je opisana v poglavju 5.2.

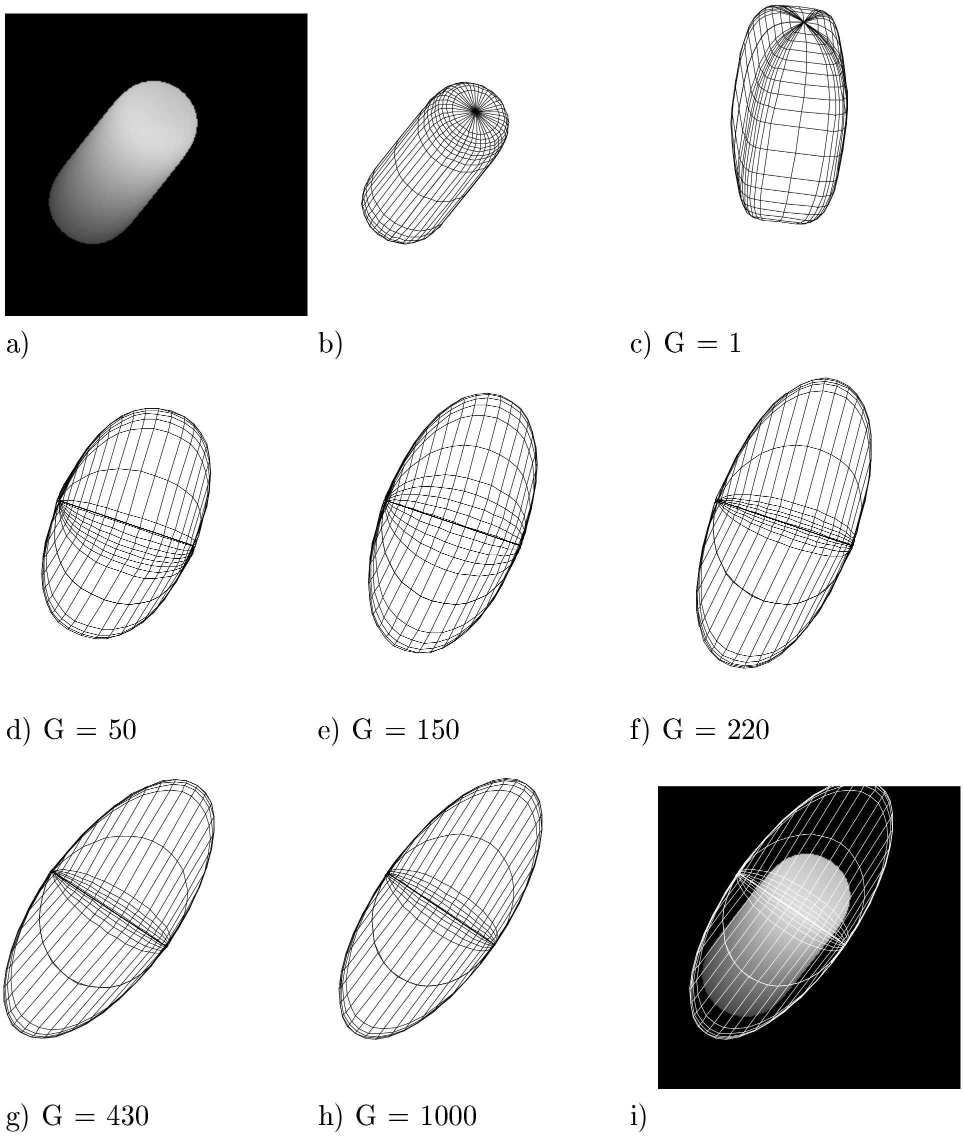
Genetski algoritem smo implementirali v jeziku C. Izvajali smo ga na Hewlett Packard Workstation 715/100, kjer je npr. izvajanje 1000 generacij z velikostjo populacije 1000 osebkov trajalo približno 30 minut.

5.1 Rekonstrukcija osnovnih superelipsoidov

Kot že rečeno, je bil naš namen najprej preveriti genetske algoritme. Zato smo vzeli za modele kar osnovne superelipsoide. Pri reševanju problemov z genetskimi algoritmi rabimo dve stvari (glej stran 13): kodiranje rešitev in funkcijo prileganja. S kodiranjem rešitev ni bilo težav, saj so vsi parametri superelipsoida realni in smo jih zakodirali kot realne gene. Funkcijo prileganja pa smo definirali podobno kot v [8].

Naj obrazložimo slike primerov rekonstrukcije. Pod a) se na sliki nahaja podana globinska slika, pod b) pa žični model originalnega superelipsoida. Točke c) do h) prikazujejo najboljše modele iz generacije, ki jo podaja G. Na koncu imamo še i), ki je kombinirana slika žičnega modela iz h) in podane globinske slike, za lažjo primerjavo.

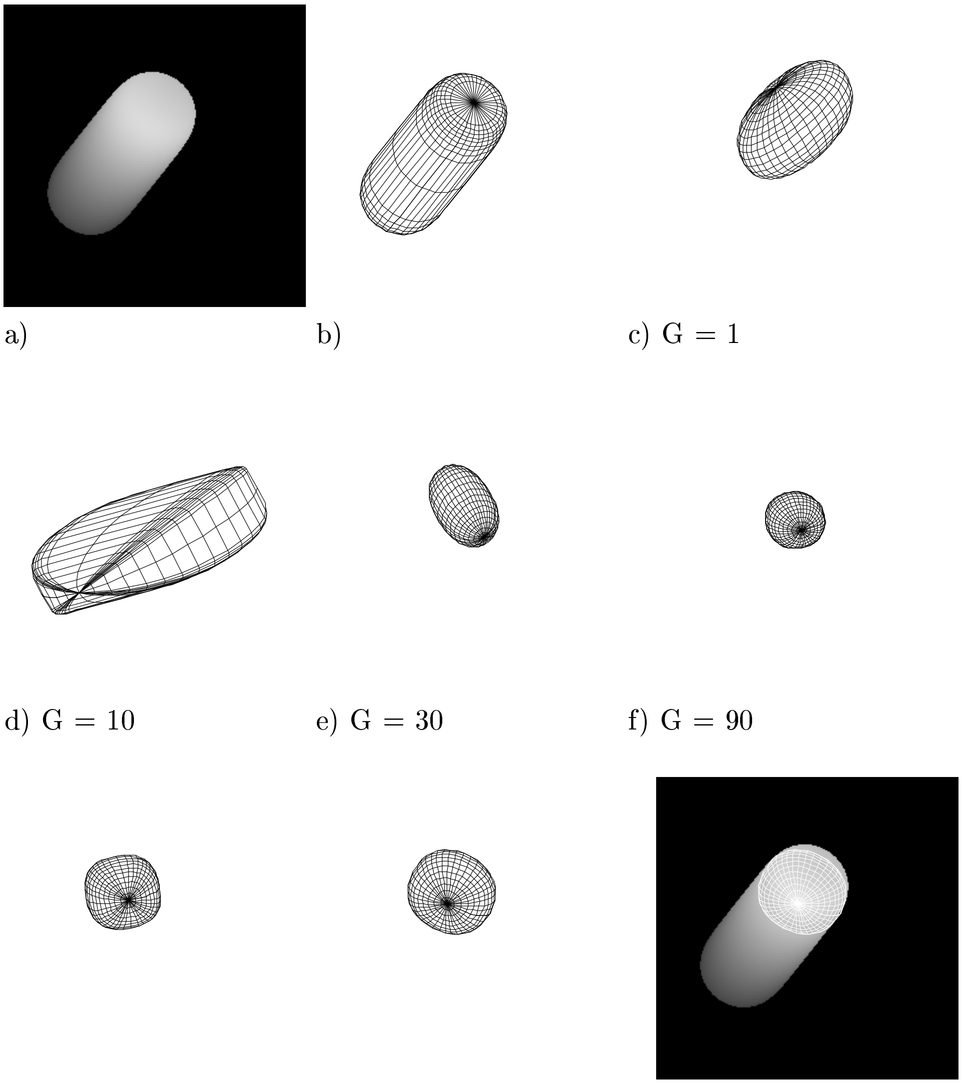
5.1.1 Rekonstrukcija s funkcijo na osnovi funkcije znotraj-zunaj



Slika 5.1: Primer rekonstrukcije superelipsoidov s funkcijo prileganja na osnovi funkcije znotraj-zunaj. Ocena kvalitete za originalen superelipsoid je bila 3.71, za model iz c) 501.4 in za rešitev (model iz h) 7.72

Za funkcijo prileganja smo najprej vzeli funkcijo $g_1(\mathcal{T}; \vec{p})$ (4.1). Tipičen primer rekonstrukcije s to funkcijo prileganja vidimo na sliki 5.1. Superelipsoid se prilega na točke iz globinske slike le lokalno. To je posledica neupoštevanja točk na superelipsoidu, ki jih na globinski sliki ni (oz. so v neskončnosti).

5.1.2 Rekonstrukcija s funkcijo prileganja na osnovi razlike v globini



a) b) c) G = 1

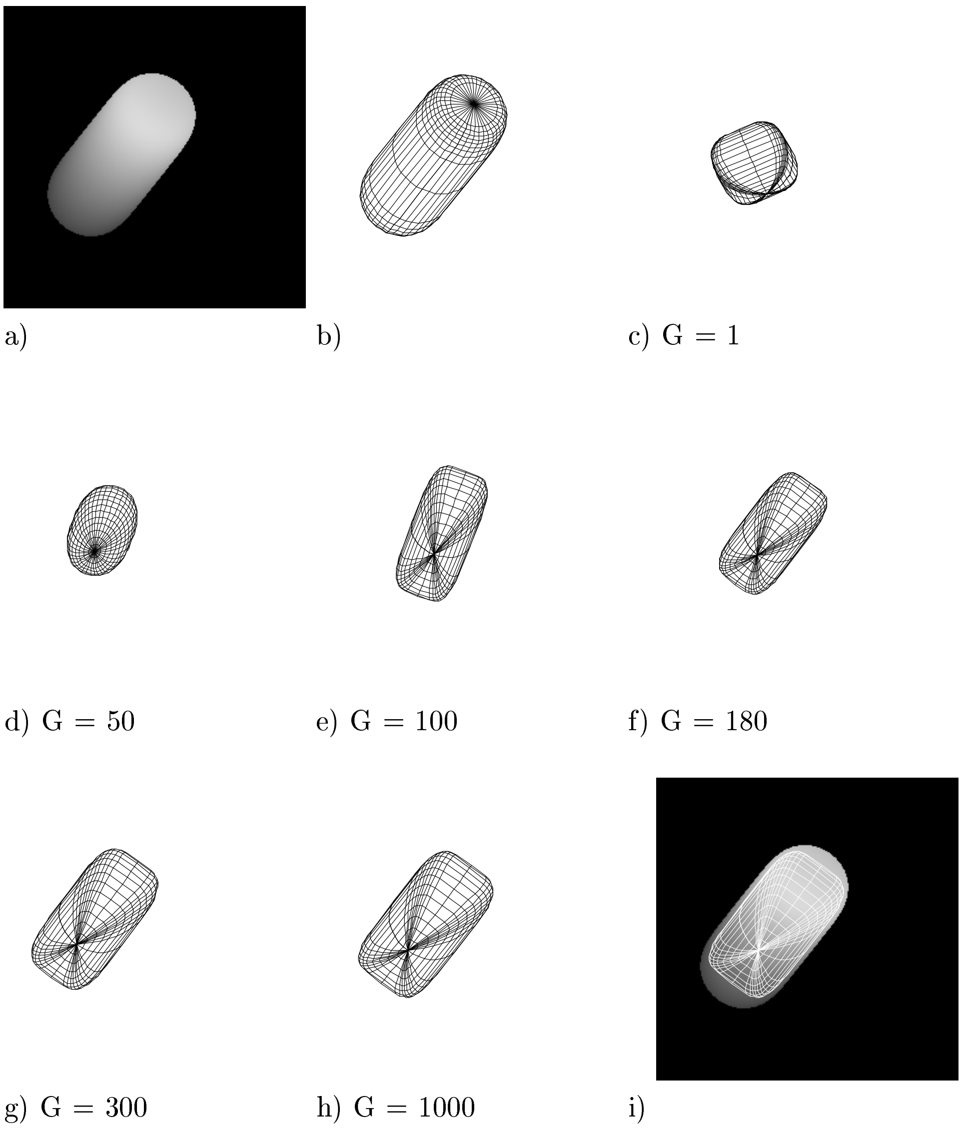
d) G = 10 e) G = 30 f) G = 90

g) G = 200 h) G = 1000 i)

Slika 5.2: Primer rekonstrukcije superelipsoidov s funkcijo prileganja na osnovi razlike v globini. Ocena kvalitete za originalen superelipsoid je bila 0.73, za model iz c) 11301.0 in za rešitev (model iz h) 2.51

Ker prva funkcija $g_1(\mathcal{T}; \vec{p})$ ni upoštevala točk, ki so bile vizualno zunaj oblaka točk globinske slike, smo se odločili za funkcijo $g_2(\mathcal{T}; \vec{p})$ (4.3). Ta je v tem pogledu teoretično boljša, saj kaznuje model, če mu točke segajo vizualno iz oblaka točk globinske slike. Pojavil pa se je drug problem. Ocena namreč ne upošteva vseh točk na globinski sliki, ampak samo tiste, ki jim ustrezajo izračunane na modelu. Zato so se razvijali modeli, ki so bili manjši, prilegali pa so se le na neko podmnožico točk globinske slike. Tipičen primer vidimo na sliki 5.2.

5.1.3 Rekonstrukcija s kombinirano funkcijo prileganja



Slika 5.3: Primer rekonstrukcije superelipsoidov s kombinirano funkcijo prileganja. Ocena kvalitete za originalen superelipsoid je bila 3.70, za model iz c) 643.1 in za rešitev (model iz h) 18.46

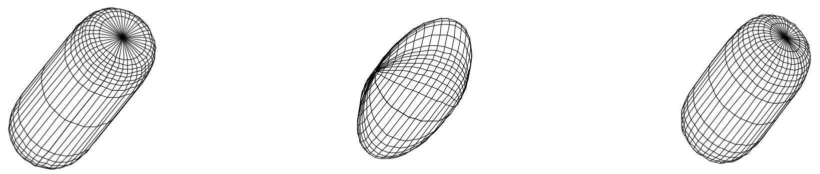
Ko smo spoznali dobre in slabe lastnosti obeh funkcij prileganja, smo se odločili, da ju nekako 'logično' združimo. To smo naredili tako, da smo model ocenili z $g_1(\mathcal{T}; \vec{p})$, nato pa smo ga še kaznovali s spremenjeno $g_2(\mathcal{T}; \vec{p})$. Dobimo novo funkcijo prileganja in sicer $g_3(\mathcal{T}; \vec{p})$ (4.4). Tipičen primer rekonstrukcije vidimo na sliki 5.3. Pri tej globinski sliki je bilo devet od desetih rekonstrukcij takih, ena pa taka s pravo rotacijo. Že na videz lahko opazimo, da ima rekonstruiran model glede na originalnega zamenjane osi z in x (ali y). Poglejmo še parametre na tabeli 5.1. Če primerjamo ε_1 in ε_2 ugotovimo, da

Model	t_x	t_y	t_z	φ	ϑ	ψ	a_1	a_2	a_3	ε_1	ε_2
Orig.	100	130	125	0.30	0.60	0.90	36.0	37.0	98.0	0.30	1.00
Rek.	101.8	132.6	142.2	-1.54	2.24	0.96	33.7	91.1	34.6	1.00	0.36

Tabela 5.1: Primerjava parametrov originalnega in rekonstruiranega superelipsoida s slike 5.3

sta zamenjana. Pri velikostnih parametrih vidimo, da so zamenjani tako: $(a_1, a_2, a_3) \rightarrow (a_2, a_3, a_1)$. Seveda razlike so, vendar so dokaj majhne. Parametri translacije se dokaj ujemajo. Pri rotaciji pa lahko vidimo, da so osi z in x (oz. y) zamenjane. Da bi genetskemu algoritmu 'pomagali' najti pravo rešitev, smo se odločili uporabiti mutacijo, ki bi nekako zamenjala te parametre.

5.1.4 Rekonstrukcija z novim 'obrnilnim' operatorjem

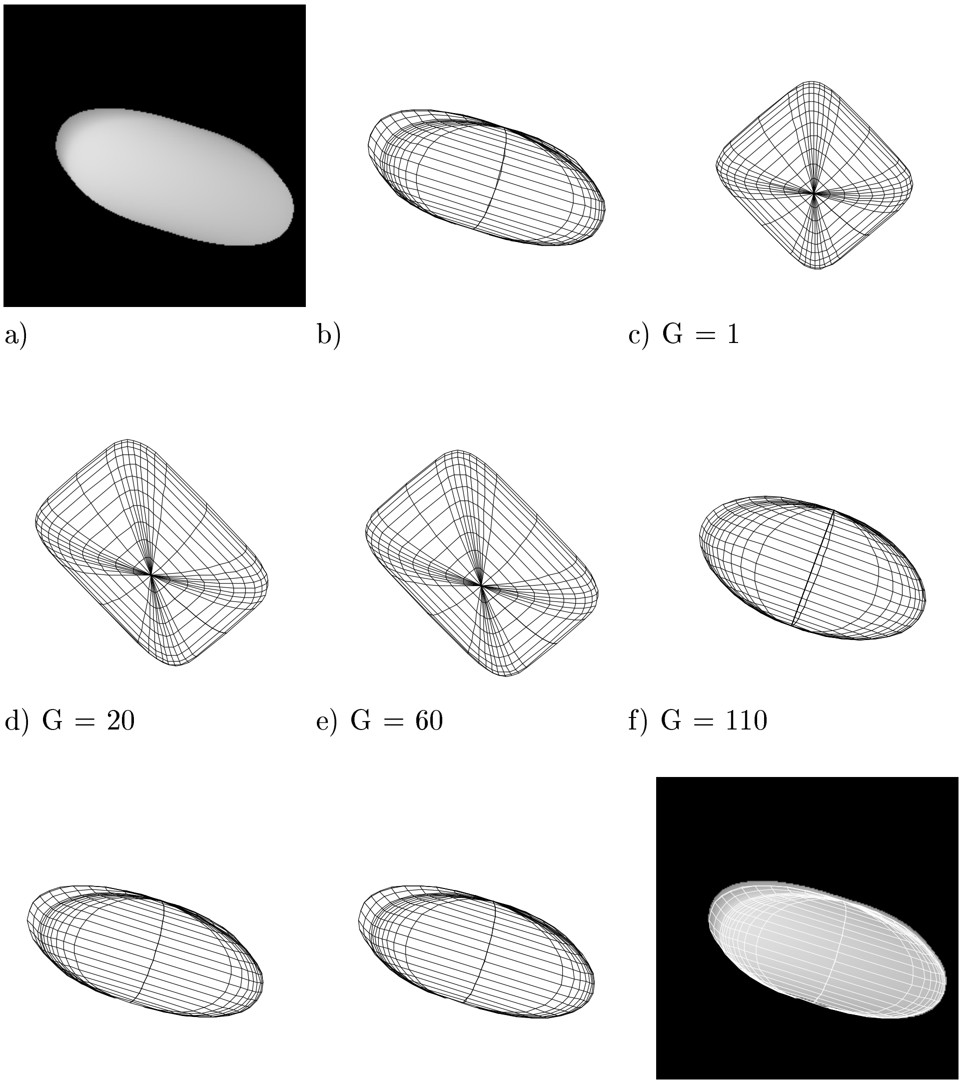


a) Ocena : 4.57 b) Ocena : 212.5 c) Ocena : 1049.7

Model	t_x	t_y	t_z	φ	ϑ	ψ	a_1	a_2	a_3	ε_1	ε_2
a)	100	130	125	0.3	0.6	0.9	36	37	98	0.3	1.0
b)	100.8	129.8	133.1	2.32	-1.33	-0.4	32	78	37	1.0	0.48
c)	100.8	129.8	133.1	-2.81	-2.29	-2.69	37	32	78	0.48	1.0

Slika 5.4: Primer zamenjave osi superelipsoida. a) je originalen superelipsoid, b) najboljši superelipsoid neke generacije, ki mu zamenjamo osi in c) superelipsoid z zamenjanimi osmi.

Zaradi zgoraj opisanih razlogov smo vgradili še mutacijo, ki je opisana na strani 19. Najprej smo jo vgradili v operator mutacije tako, da je nek osebek lahko mutiral tudi tako, da je zamenjal osi. Tak princip ni bil uspešen, saj se je najboljši osebek nekoliko razlikoval od originalnega modela. To je povzročilo, da je bil veliko slabše ocenjen in je tako odpadel že pri naslednji selekciji. Zato smo mutacijo izvedli kot poseben operator. Vsakih nekaj generacij (npr. sto) se je del najboljših osebkov (npr. desetina) prenesel v naslednjo generacijo, ostali pa so bili generirani iz teh tako, da so jim bile zamenjane osi in se jih je še naključno spremenilo (mutiralo). Vendar pa se tudi to iz istih razlogov ni obneslo. Primer vidimo na sliki 5.4. Superelipsoid z zamenjanimi osmi je na videz dokaj dobra rešitev, vendar če pogledamo ocene vidimo, da je slabše ocenjen kot predhodnik. Razlika je predvsem v parametru translacije v globini t_z, pa tudi v velikosti a_3. Kljub temu, da smo tako obrnjene superelipsoide nato še mutirali in z njimi zapolnili velik del populacije, se praviloma niso razvili v dobre rešitve.



a) b) c) G = 1

d) G = 20 e) G = 60 f) G = 110

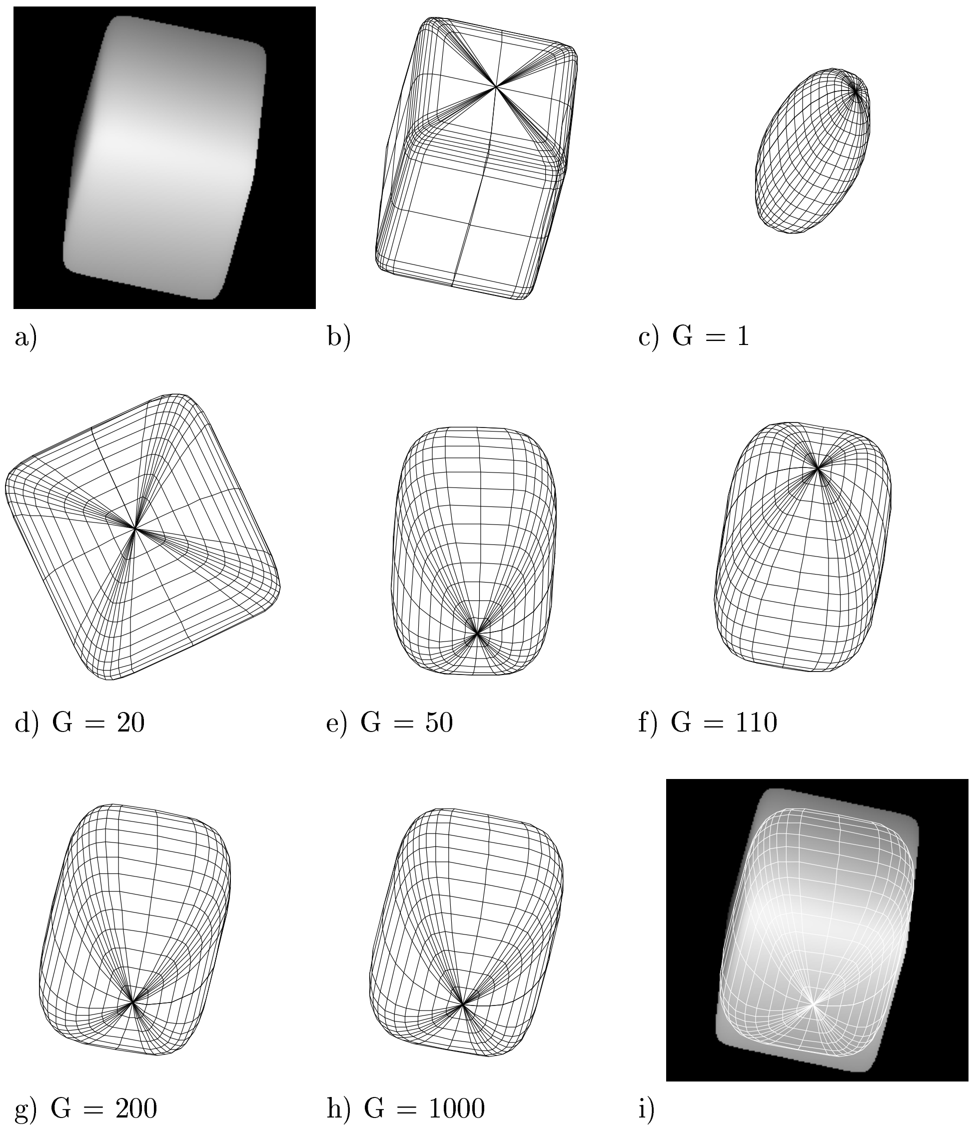
g) G = 700 h) G = 1000 i)

Model	t_x	t_y	t_z	φ	ϑ	ψ	a_1	a_2	a_3	ε_1	ε_2
Orig.	145	110	160	-0.2	1.7	-1.9	50.0	100.0	45.0	0.9	0.2
Rek.	145.6	108.7	165.5	0.2	1.45	1.24	44.1	100.5	45.0	0.93	0.20

Slika 5.5: Primer rekonstrukcije superelipsoida. Ocena kvalitete za originalen superelipsoid je bila 117.6, za rešitev (model iz h) pa 134.2

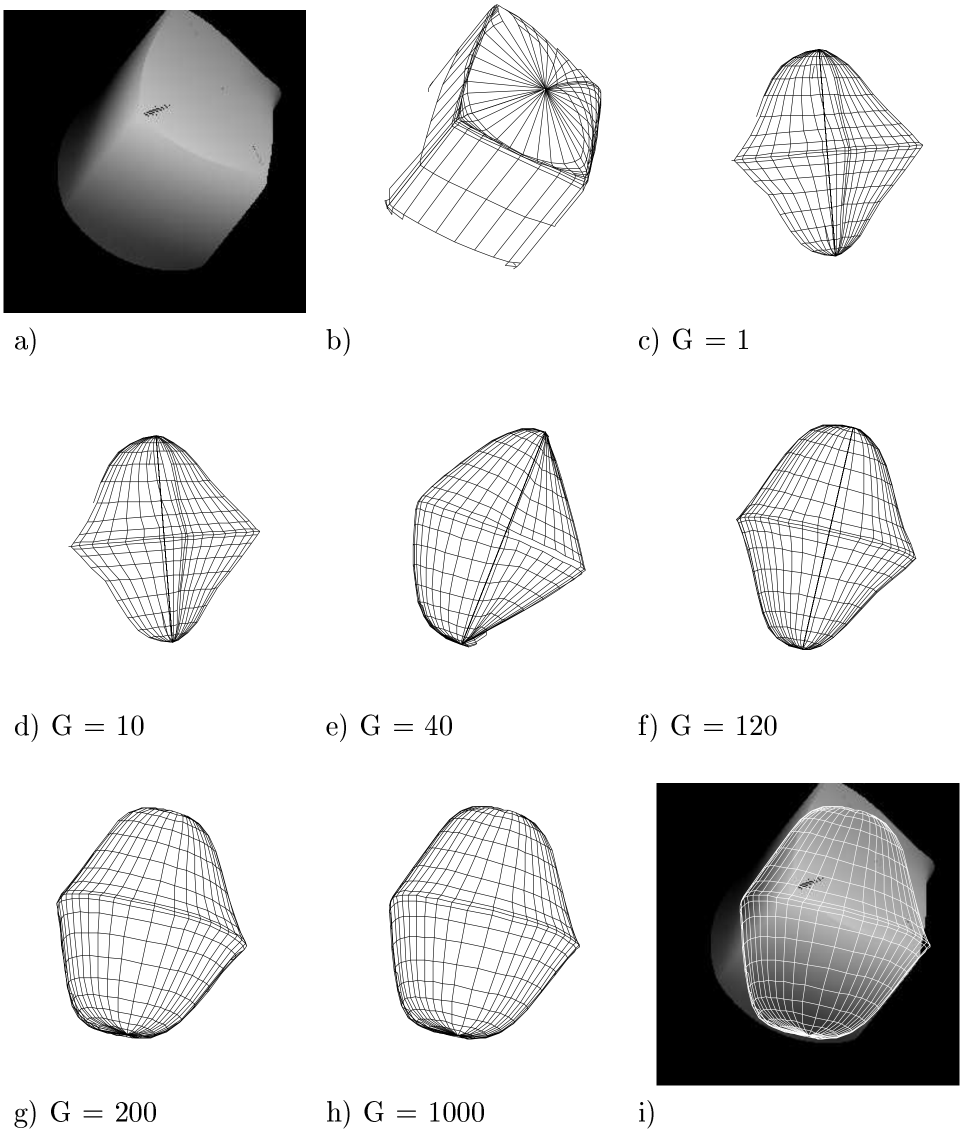
Na slikah 5.5 in 5.6 vidimo še dva primera rekonstrukcije. Na sliki 5.5 vidimo, da je genetski algoritem našel zelo dobro rešitev. Če pogledamo še ustrezne parametre na tabeli opazimo razliko le pri parametrih rotacije. Vendar vizualna primerjava pokaže, da se tudi v tem model in slika ujemata.

Za primer s slike 5.6 velja, da genetski algoritem ni našel ustrezne rešitve, temveč neko približno. Opazimo lahko, da je genetski algoritem dokaj hitro (v nekaj desetih generacijah) najde približno rešitev, ki pa je potem ni razvil v pravo.



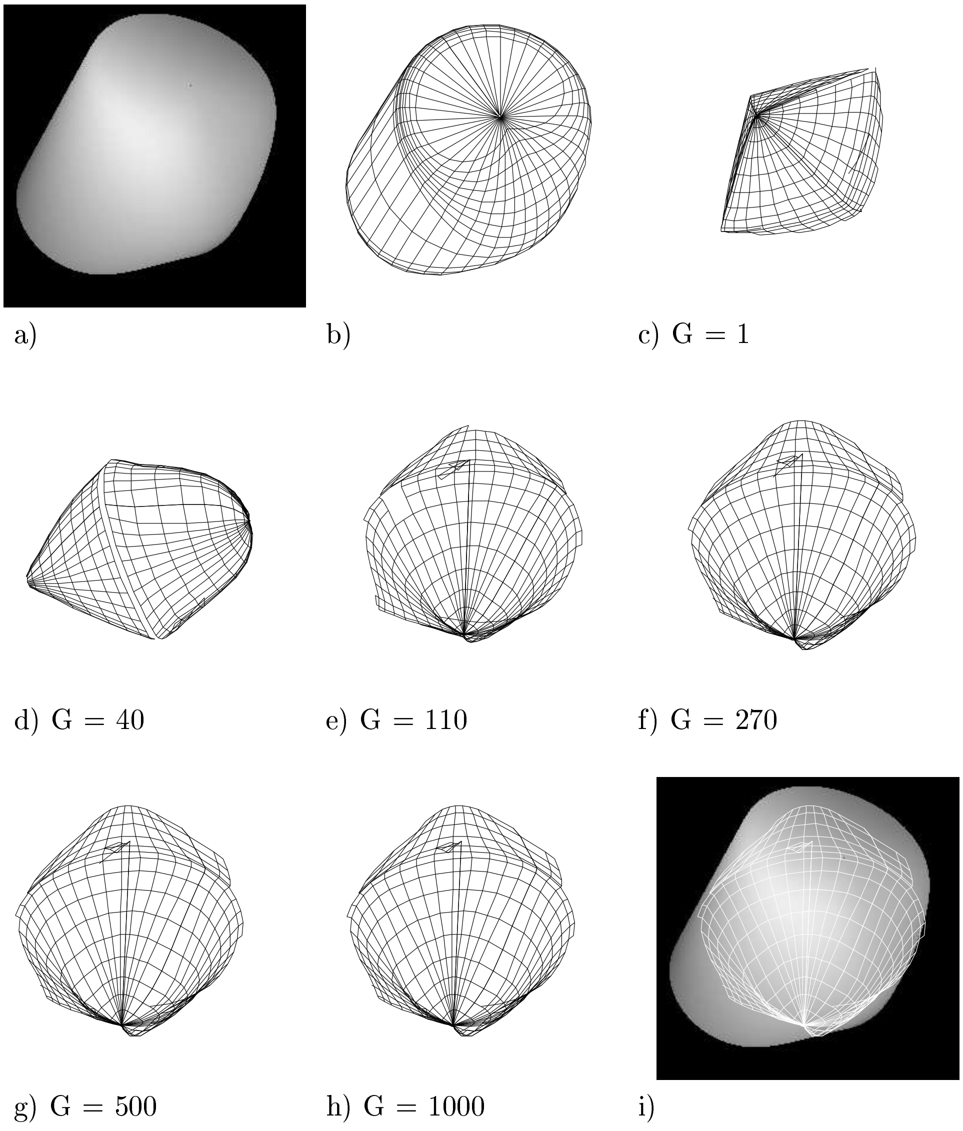
Slika 5.6: Primer rekonstrukcije superelipsoida. Ocena kvalitete za originalen superelipsoid je bila 3.85, za rešitev (model iz h) pa 105.9

5.2 Rekonstrukcija razširjenih superelipsoidov



Slika 5.7: Primer rekonstrukcije razširjenega superelipsoida. Ocena kvalitete za originalen superelipsoid je bila 401.2, za rešitev (model iz h) pa 2390.2

S tako dopolnjemim genetskim algoritmom smo hoteli preiskusiti še rekonstrukcijo razširjenih superelipsoidov. Na primerih s slike 5.7 in 5.8 lahko vidimo, da algoritem ni našel prave rešitve, lahko pa ugotovimo neko prileganje. Verjetno je funkcija prileganja preveč 'nagubana', to je, ima ogromno lokalnih minimumov, ki so tudi globoki. Tako algoritem izboljšuje rešitve samo do teh lokalnih minimumov.



Slika 5.8: Primer rekonstrukcije razširjenega superelipsoida. Ocena kvalitete za originalen superelipsoid je bila 598.5, za rešitev (model iz h) pa 2091.8

6

Zaključek

V diplomskem delu je prikazana metoda za rekonstrukcijo superelipsoidov iz globinskih slik. Rekonstrukcija je v bistvu določanje parametrov superelipsoda. To pa je problem modeliranja podatkov. Pri tem imamo neko kriterijsko funkcijo, ki nam pove, kako dobro parametri opišejo podatke.

Metoda temelji na optimizaciji kriterijske funkcije z genetskimi algoritmi. Pri reševanju problemov z genetskimi algoritmi potrebujemo le kriterijsko funkcijo, ki ji pri genetskih algoritmih pravimo funkcija uspešnosti, in kodiranje parametrov. S kodiranjem ni posebnih težav, saj so parametri superelipsoida realni. Podane so bile v grobem tri funkcije prileganja. Prva temelji na funkciji znotraj-zunaj, druga na primerjavi razlike v globini v nekaj točkah, tretja pa je kombinacija prvih dveh in ta se je pokazala za najboljšo. Podane so še posebnosti pri rekonstrukciji superelipsoidov. Opisana je posebna mutacija, ki zamenja osi superelipsoida in ki naj bi izboljšala delovanje algoritma.

Sledijo še rezultati izvajanja. Podani so primeri rekonstrukcije osnovnih superelipsoidov za vsako funkcijo prileganja in nato še s dodanim posebnim operatorjem, ki uporablja 'zamenjalno' mutacijo. Kljub temu pa pri nekaterih primerih metoda ni dala zadovoljivih rezultatov. Vzrok gre iskati v praviloma slabši oceni takih osebkov, tako da se pri naslednjih selekcijah le-ti zavržejo.

Na koncu so je prikazana še rekonstrukcija razširjenih superelipsoidov. Rekonstrukcija je bila neuspešna, saj algoritem ni nikoli našel ustrezne rešitve. Mogoče bi bilo potrebno genetski algoritem 'križati' z nekakšnim 'ogrevanjem'. Tako bi parametre, ki določajo deformacijo, na začetku omejili zelo ozko, da bi dosegli približen nedeformiran superelipsoid. Nato pa bi tem parametrom poščali vedno več 'svobode' in rekonstruirali razširjen superelipsoid.

6.1 Nadaljne delo

Pri delu se je pokazalo še nekaj možnih področij obravnave, ki pa niso bila predmet te diplomske naloge. Naj jih naštejemo:

- Rekonstrukcijo z genetskimi algoritmi bi bilo potrebno sistematično testirati na večji množici globinskih slik, tako sintetičnih kot realnih.
- Poizkusiti bi bilo treba hibridizirati genetske algoritme s kakšno lokalno optimizacijo, kot je npr. 'hill-climbing'.

- Razširjenim superelipsoidom bi bilo mogoče dodati še deformacije, kot so npr. ukrivljanje in zvijanje, in preveriti metode za rekonstrukcijo tako definiranih modelov.
- Preučiti bi bilo treba možnosti rekonstrukcije sestavljenih objektov, katerih dele opišemo z razširjenimi superelipsoidi.
- Ocena kvalitete rešitev je bila izračunana le na podlagi podatkov o točkah. Mogoče bi jo bilo razširiti še z podatki o normalah.
- Genetski algoritem pogosto ni našel prave rešitve, ampak tako, ki je imela osi zamenjane. Kljub temu, da smo dodali nekaj sprememb, ki naj bi problem rešile, se to ni zgodilo ker so bili osebki prehitro završeni in se niso mogli razviti v pravilne rešitve. Mogoče bi z 'otoško' varianto genetskih algoritmov to lahko rešili.
- Preizkusiti bi bilo treba križanje genetskega algoritma z 'ogrevanjem', kot je bilo opisano zgoraj.

Literatura

[1] A.P. Pentland, "Perceptual organization and the representation of natural form," Art. Intell. 28, str. 293-331, 1986.

[2] F. Solina in R. Bajcsy, "Recovery of parametric models from range images: The case for superquadrics with global deformations," IEEE Trans. Pattern Anal. Machine Intell. 12, str. 131-147, 1990.

[3] N. Yokoya et al., "Recovery of Superquadric Primitives from a Range Image Using Simulated Annealing," Proc. Int. Conf. Pattern Recognition, str. 168-172, 1992.

[4] F.G. Callari in U. Maniscalco, "A New Robust Approach to Image Shading Analysis and 3-D Shape Reconstruction," Proc. Int. Conf. Pattern Recognition, str. 103-107, 1994.

[5] A. Vidmar, "Razpoznavanje superelipsoidov iz kontur," diplomska naloga, Univerza v Ljubljani, Fakulteta za elektrotehniko in računalništvo, 1991.

[6] F. Solina, "Segmentation with Volumetric Part Models," Computing Suppl. 11, str. 201-220, 1996.

[7] A.H. Barr, "Superquadrics and Angle-Preserving Transformations,"IEEE Computer Graphics and Aplications, vol. 1, pp. 11–23, January 1981.

[8] F. Solina, "Shape Recovery and Segmentation with Deformable Part Models,"Ph.D. Thesis, University of Pennsylvania, 1987.

[9] B. Filipič, "Genetski algoritmi v kombinatorični optimizaciji," doktorska disertacija, Univerza v Ljubljani, 1993.

[10] J.N. Bronštejn in K.A. Semendjajev, "Matematični priročnik," str. 248–250 in 318–320, Tehniška založba Slovenije, 1984.

[11] L. Zhou in B. Yuan, "Extending Superquadrics: A New Approach of Modeling Smoothly Deformable Shapes,", v recenziji za objavo v IEEE Trans. Patt. Anal. Machine Vision, (1997).

[12] A. Leonardis, "Image analysis using parametric models: model-recovery and model-selection", doktorska disertacija, Univerza v Ljubljani, Fakulteta za elektrotehniko in računalništvo, 1996.

[13] A. Jaklič, "Gradnja CAD modelov iz globinskih slik," doktorska disertacija, Univerza v Ljubljani, 1996.

[14] J. Holland, "Adaptation in Natural and Artificial systems," MIT Press, Cambridge, 1992.

[15] D. Whitley in J. Kauth, "GENITOR: A different genetic algorithm," Rocky Mountain Conference on Artifical Intelligence, str. 118–130, Denver, 1988.

[16] D. Levine, "Users Guide to the PGAPack Parallel Genetic Algorithm Library," Argonne National Laboratory, 1996.
(`ftp://ftp.mcs.anl.gov/pub/pgapack/pgapack.tar.Z`)

Zahvala

Iskreno se zahvaljujem prof. dr. Francu Solini za mentorstvo.

Zahvaljujem se tudi Alešu za vse nasvete in informacije ter Bojanu, Boru in Jasni za pomoč pri TEXu.

Izjava

Izjavljam, da sem diplomsko delo samostojno izdelal pod vodstvom mentorja prof. dr. Franca Soline. Izkazano pomoč drugih sodelavcev sem v celoti navedel v zahvali.