KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PRAKTINĖS INFORMATIKOS KATEDRA

AUDRIUS ŠEŠKAS

ERDVĖS VAIZDAVIMO ALGORITMŲ TYRIMAS IR ANALIZĖ

Magistro darbas

Vadovas:
doc. A. Lenkevičius

KAUNAS 2005

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

PRAKTINĖS INFORMATIKOS KATEDRA

TVIRTINU
Katedros vedėjas
doc. D. Rubliauskas
2005 05

ERDVĖS VAIZDAVIMO ALGORITMŲ TYRIMAS IR ANALIZĖ

Informatikos magistro baigiamasis darbas

Kalbos konsultantė
Lietuvių k. katedros lekt.
dr. J. Mikelionienė
2005 05

Recenzentas
doc. L. Nemuraitė
2005 05

Vadovas
doc. A. Lenkevičius
2005 05

Atliko
IFM-9/1 gr. stud.
A. Šeškas
2005 05 23

KAUNAS, 2005

Kvalifikacinė komisija

Pirmininkas:	Laimutis Telksnys, akademikas
Sekretorius:	Stasys Maciulevičius, docentas
Nariai:	Rimantas Barauskas, profesorius, Raimundas Jasinevičius, profesorius, Jonas Mockus, akademikas, Rimantas Plėštys, docentas, Henrikas Pranevičius, profesorius.

Summary

The new approach of volume data visualization is presented in this paper. The proposed algorithm is based on point-based volume visualization. The main idea of it is to approximate volumetric dataset with surface made from points rather than approximate it with polygon-based iso-surface.

This work covers a survey of volumetric data types, main principles of volume visualization and areas where improvements on algorithms of volume visualization are likely to be done. There is analysis of meshless iso-surface generation from multiblock data algorithm presented, suggesting dynamic iso-surface generation modification and some experiments based on these suggestions.

Santrauka

Taškais paremtas erdvės vaizdavimas pagrįstas nauju požiūriu į erdvinius duomenis. Sąlygos šiuo principu paremtiems algoritmams vystytis atsirado techninės įrangos tobulėjimo dėka, kuomet vaizdo plokščių galimybės leido atvaizduoti didelius grafinių primityvų kiekius. Taškais paremto vaizdavimo idėja – iš vokselių išskirto trikampiais paremto dengiančiojo paviršiaus pakeitimas iš taškų sudarytu paviršiumi.

Šis darbas yra erdvinių duomenų tipų, pagrindinių erdvės vaizdavimo metodų, sričių, kuriose algoritmai dažniausiai yra tobulinami, apžvalga. Taip pat buvo atliktas tinklelio nereikalaujančio dengiančiojo paviršiaus išskyrimo iš keletą blokų turinčių duomenų algoritmo analizė bei ištirta algoritmo modifikacija, kuomet dengiančiojo paviršiaus taškų aibės tankis generuojamas atsižvelgiant į paviršiaus srities kreivumą.

TURINYS

ĮVADAS ... **9**

1 BENDROJI DALIS ... **10**

1.1 Erdviniai duomenys ... 10

1.1.1 Vokseliai ... 10

1.1.2 Tinkleliai ... 10

1.2 Vaizdavimo metodai ... 12

1.2.1 Tiesioginis erdvės vaizdavimas ... 12

1.2.2 Netiesioginis erdvės vaizdavimas ... 13

1.2.2.1 Dengiantieji paviršiai ... 13

1.2.2.2 Daugiakampiais paremtas vaizdavimas ... 15

1.2.2.3 Taškais paremtas vaizdavimas ... 18

1.2.2.4 Taškymas ... 21

1.2.2.5 Erdvės vaizdavimo algoritmų trūkumai ... 21

1.2.2.6 Hierarchinių duomenų struktūrų panaudojimas erdvės vaizdavime ... 22

1.3 Išvados ... 24

2 EKSPERIMENTINĖ DALIS ... **26**

2.1 Betinklis dengiančiojo paviršiaus išskyrimas iš keletą blokų turinčių duomenų ... 26

2.1.1 Algoritmas ... 26

2.1.2 Algoritmo modifikacijos ... 28

2.1.3 Algoritmo aprašymas ... 29

2.1.4 Rezultatai ... 31

IŠVADOS ... **37**

LITERATŪRA ... **38**

TERMINŲ IR SANTRUMPŲ ŽODYNAS ... **40**

LENTELIŲ SĄRAŠAS

1 LENTELĖ. PAGRINDINIAI VAIZDAVIMO METODŲ PRIVALUMAI BEI TRŪKUMAI 24
2 LENTELĖ. TESTAVIMUI NAUDOTI DUOMENŲ RINKINIAI 31
3 LENTELĖ. ALGORITMO VYKDYMO SU VIENO TINKLELIO DUOMENIMIS REZULTATAI 32
4 LENTELĖ. PRASMINGŲ CELIŲ DUOMENŲ RINKINIUOSE KIEKIAI 34

PAVEIKSLŲ SĄRAŠAS

1 pav. Tinklelių tipai. 11
2 pav. Pagrindinės celių aktyvių viršūnių kombinacijos 16
3 pav. Užpildančių kubų algoritmo vykdymo rezultatai 17
4 pav. Poslinkio kriterijus 21
5 pav. Originalaus algoritmo dalių vykdymo laikinė diagrama 33
6 pav. Modifikuoto algoritmo vykdymo laikinės diagramos 34
7 pav. Originalaus ir modifikuoto algoritmų vykdymo laikų palyginimas apdorojant „NEGHIP“ duomenų rinkinį 35
8 pav. Originalaus ir modifikuoto algoritmų vykdymo laikų palyginimas apdorojant „FUEL“ duomenų rinkinį 36

Įvadas

Dabartinė kompiuterių grafinė įranga nėra pritaikyta erdvinių duomenų apdorojimui, taigi, yra reikalingi algoritmai, galintys savarankiškai arba su techninės įrangos pagalba atlikti vaizdavimą.

Pirmi erdvės vaizdavimo algoritmai atsirado prieš beveik 20 metų, ir nuo tada yra nuolatos vystomi. Iki šių dienų optimalus algoritmas nėra sukurtas, o praktikoje naudojamų duomenų rinkinių dydis vis auga. Per visa šios srities vystymo laikotarpį buvo sukurta keletas skirtingų erdvinių duomenų vaizdavimo metodų bei daugybė algoritmų, paremtų vienu ar kitu metodu, kartais, siekiant geresnių rezultatų netgi sujungiant skirtingas metodikas.

Erdvinių duomenų apdorojimo metodai priklauso nuo erdvės ir objektų, apie kuriuos yra saugoma informacija, pobūdžio.

Duomenų rinkiniai pasiekė tokį dydį, jog jų apdorojimas bei vaizdavimas tapo praktiškai neįmanomas įprastiniais metodais. Augančios apimtys įtakojo naujų duomenų formų atsiradimą. Erdviniai duomenys evoliucionuoja, ir jau senai nėra vien tik iš taisyklingų kubų sudarytos struktūros. Nereguliarių, kreivinių tinklelių naudojimas sumažino duomenų rinkinių apimtis, tačiau išaugo vaizdavimo algoritmų sudėtingumas.

Keletą tinklelių turintys duomenų rinkiniai – dar vienas būdas sumažinti duomenų rinkinių dydį. Neliko poreikio saugoti didelį tankų rinkinį, kuomet galima išsaugoti keletą tinklelių skirtingoms erdvės sritims – retesnį, skirtą mažiau detalumo reikalaujančioms sritims, bei tankesnius – detaliam objektų vaizdavimui.

Hierarchinės duomenų struktūros – vienas plačiausiai taikomų algoritmų optimizavimo būdų. Teisingai parinkta struktūra gali atminties sąskaita paspartinti ženkliai erdvinių duomenų apdorojimą.

Šiame darbe bus siekiama pagerinti vieną iš taškais paremto netiesioginio erdvės vaizdavimo algoritmų. Bus siekiama sumažinti skaičiavimų kiekį, reikalingą sukurti dengiančiajam paviršiui. Modifikuojamas algoritmas ypatingas savo universalumu – bendru atveju jis gali būti naudojamas paviršiaus išskyrimui erdvinių duomenų, išdėstytų įvairiais tinklelių tipais. Taip pat algoritmas gali būti taikomas duomenims, sudarytiems iš keleto tinklelių neatsižvelgiant į jų tarpusavio išsidėstymą.

1 Bendroji dalis

1.1 Erdviniai duomenys

1.1.1 Vokseliai

Erdviniai duomenys – tai aibė S, susidedanti iš elementų *(x, y, z, v)*, vaizduojančių kokią nors duomenų savybės reikšmę *v* trimatėje erdvėje koordinatėmis (x, y, z). Duomenys gali būti daugiareikšmiai. Tokiu atveju v žymi keletą išmatuojamų duomenų savybių, pavyzdžiui tankumas, karštį ar spaudimą [1]. Reikšmė v taip pat gali būti vektorius, žymintis, pavyzdžiui, greitį kuria nors kryptimi.

Duomenys gali būti paimti iš visiškai atsitiktinų erdvės vietų, bet daugumai atvejų aibė S yra vienoda visomis kryptimis arba izotropinė, turinti elementus, paimtus vienodais intervalais visomis trimis pagrindinėmis ašimis. Intervalai vienoje ašyje gali būti pastovūs, tačiau skirtingi skirtingose ašyse. Tokiu atveju aibė S yra anizotropinė [4].

Masyvas S nusako tik keleto išmatuojamų duomenų savybių reikšmes pavienėse erdvės vietose. Funkcija *f(x,y,z)* gali būti nusakyta erdvėje R^3, tokiu būdu aprašant reikšmes bet kurioje vietoje. Funkcija *f(x,y,z)=S(x,y,z)* jei *(x,y,z)* yra tinklelio pozicija, priešingu atveju *f(x,y,z)*, panaudojus kokią nors interpoliavimo funkciją masyvui S, tampa apytikriai lygi elemento, esančio pozicijoje *(x,y,z)*, reikšmei.

Aštuoni gretimi erdvinių duomenų elementai, sujungti į kubą sudaro paprasčiausią ir dažniausiai naudojamą erdvės vienetą, dar vadinamą vokseliu. Tokio pavidalo vokseliai yra plačiausiai naudojami erdvės vaizdavime, tačiau, atskirais atvejais, vokseliai gali būti ir kitokių formų, pavyzdžiui prizmės, gretasieniai.

1.1.2 Tinkleliai

Paprastai erdviniai duomenys išsidėstę pagal griežta struktūrą, vadinamą tinkleliu.

Taisyklingas tinklelis – tai toks tinklelis, kuriame duomenų elementai išsidėstę vienodais intervalais visomis trimis pagrindinėmis ašimis.

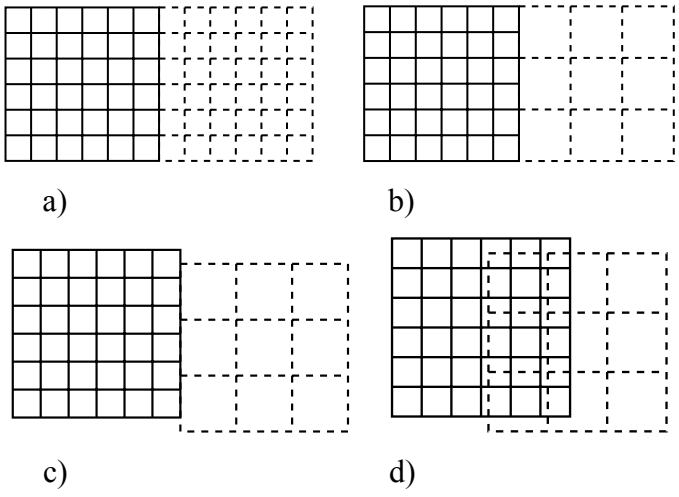
Tuo atveju, kuomet elementų aibė yra aprašyta taisyklingame tinklelyje, trimatis masyvas, saugantis informaciją apie erdvę, naudojamas elementų savybių saugojimui, o elemento poziciją nusako jo vieta masyve.

Dar yra naudojami linijiniai, kreiviniai arba nenuoseklūs tinkleliai. Nenuoseklūs tinkleliai sutinkami duomenų rinkiniuose, saugančiuose informaciją apie itin skirtingą erdvę, kuomet vienoje erdvės srityje erdviniai duomenys yra pastovūs, o kitoje – stipriai svyruoja [11]. Nenuoseklūs tinkleliai naudojami duomenyse, saugančiuose informaciją apie objektus, turinčius labai smulkių dalelių.

Linijiniame tinklelyje celės yra lygios ašyje, tačiau išlygiavimas tarp ašių yra skirtingas. Kuomet toks tinklelis yra netiesiškai transformuojamas išsaugant tinklelio topologiją, tuomet tinklelis tampa kreivinis. Kreivinis tinklelis – tai struktūrizuotas tinklelis, ištemptas erdvėje, išsaugant reguliaraus tinklelio sujungimus. Didesnio detalumo reikalaujantiems duomenų rinkiniams naudojami nestruktūrizuoti nenuoseklūs tinkleliai, kurie neturi pilnos informacijos apie atskirų elementų jungimą.

Paprasčiausias būdas dirbti su nenuosekliais tinkleliais yra juos perrinkti, tokiu būdu sukuriant apytikrį reguliarų tinklelį, kurį galima būtų apdoroti klasikiniais metodais. Norint gauti didelį tikslumą būtina atlikinėti labai didelio dažnio perrinkimą, reikalaujantį didelio skaičiavimų kiekio, o perrinktas tinklelis tampa pernelyg didelis, kad būtų patogus apdoroti. Įvairūs nenuoseklių tinklelių vaizdavimo algoritmai vietoj perrinkimo remiasi įvairiu indeksavimu bei buferiais.

Erdvinių duomenų rinkinys, sudarytas iš keleto savarankiškų tinklelių, vadinamas daugiablokiu [2]. Galimi tinklelių tarpusavio išsidėstymo variantai pateikti 1 paveiksle.



1 pav. Tinklelių tipai. (a) atitinkantys, (b) dalinai atitinkantys,

(c) neatitinkantys ir (d) persidengiantys

Dauguma vaizdavimo algoritmų tinkami darbui su dviem pirmaisiais būdais išdėstytais tinkleliais, kuomet neatitinkantiems bei persidengiantiems tinkleliams pritaikytų algoritmų yra kur kas mažiau. Dirbant su neatitinkančiais arba persidengiančiais tinkleliais iškyla problemos, kuomet

reikia apdoroti tinklelių kraštus, o konkrečiau – atskirų tinklelių susikirtimo arba susijungimo vietas. Persidengiančių tinklelių atveju taip pat sudėtinga apibrėžti kaip interpoliuojant vieno tinklelio duomenis įvertinti jį kertančio tinklelio duomenis.

1.2 Vaizdavimo metodai

Galima išskirti du pagrindinius erdvinių duomenų vaizdavimo metodus – vaizdavimas dengiančiaisiais paviršiais (netiesioginis erdvės vaizdavimas) bei tiesioginis erdvės vaizdavimas. Papildomai, kiekvienas iš šių metodų pagal vaizdavimo principą dar gali būti skaidomas į priklausomus ir nepriklausomus nuo stebėjimo parametrų. Priklausomas nuo stebėjimo parametrų vaizdavimo principas pasireiškia tuo, jog skaičiavimus stengiamasi atlikti tik tose erdvės vietose, kurios turės įtakos galutiniam matomam vaizdui. Šiuo principu paremtame vaizdavime skaičiavimai turi būti kartojami kiekvieną kartą pasikeitus stebėjimo parametrams. Vaizduojant pagal nepriklausomą nuo stebėjimo parametrų principą sukuriama geometrija, artima dengiančiajam paviršiui. Šiuo atveju pakartotinis skaičiavimas pasikeitus peržiūros parametrams nereikalingas, tačiau labai išauga atminties, reikalingos saugoti duomenis, kiekis didėjant vaizdo detalumui.

1.2.1 Tiesioginis erdvės vaizdavimas

Tiesioginio erdvės vaizdavimo metu bandoma perteikti viso duomenų rinkinio vaizdą skiriant kiekvienam erdvės elementui ar elementų intervalui skirtingas spalvų bei nepermatomumo reikšmes [3]. Skirtingai nuo netiesioginio erdvės vaizdavimo metodų, tiesioginiame vaizdavime dalyvauja visas duomenų rinkinys. Tiesioginio erdvės vaizdavimo metodai pagrysti supaprastintomis fizikinėmis šviesos sklidimo teorijomis, įvertinant šviesos spinduliavimo, absorbavimo efektus. Paprasčiausia tiesioginio erdvės vaizdavimo idėja yra pervesti spindulį per kiekviena erdvės tašką, apskaičiuoti funkcijų reikšmes atitinkamuose erdvės taškuose išilgai spindulio bei apjungti gautas spalvos bei nepermatomumo reikšmes.

Vienas pirmųjų tiesioginio erdvės vaizdavimo algoritmų buvo pasiūlytas Levoy M. [14]. Šiame algoritme tinkamai apdoroti ir suklasifikuoti duomenys yra šešėliuojami, įvertinamos spalvos, nepermatomumas. Iš stebėjimo taško vedami spinduliai per šiuos duomenų masyvus, įvertinant kiekvieno spindulio įtaką kiekvienam jo paveiktam vokseliui. Šiame algoritme buvo įvertinama ir ta erdvė, kuri neturi įtakos galutiniam duomenų rinkinio atvaizdui. Vėliau šis trūkumas buvo pašalintas hierarchinių duomenų struktūrų pagalba.

Tiesioginis erdvės vaizdavimas – tai vaizdavimo metodai, bandantys atvaizduoti visus trimačius duomenis į vieną dvimatį atvaizdą. Tiesioginis erdvės vaizdavimas išsaugo kur kas daugiau informacijos nei vaizdavimas dengiančiais paviršiais, tačiau tokio vaizdavimo algoritmų sudėtingumas kur kas didesnis , ir atitinkamai – didesnė skaičiavimo trukmė, kadangi vaizduojant įvertinamas kiekvienas erdvės elementas. Tiesioginio erdvės vaizdavimo metodai yra plačiai naudojami vaizduojant amorfinius kūnus.

1.2.2 Netiesioginis erdvės vaizdavimas

1.2.2.1 Dengiantieji paviršiai

Netiesioginio vaizdavimo metoduose erdvės elementai pirmiausiai konvertuojami į tarpinius juos atitinkančius paviršius, ir tuomet atvaizduojami ekrane naudojantis įprastiniais kompiuterinės grafikos metodais.

Dengiantiesiems paviršiams išrinkinkti iš erdvinių duomenų problema formaliai nusakoma taip: skaitmeniniai erdvės duomenų rinkiniai yra pora (V, W), kur $V = \{v_i \in R^3, i = 1,\ldots,n\}$ yra baigtinė srities $\Omega \subset R^3$ taškų aibė. $W = \{w_i \in R, i = 1,\ldots,n\}$ yra atitinkama skaitmeninių reikšmių aibė, gauta matuojant skaitmeninį lauką *f(x,y,z)* aibės V taškuose, pvz., $w_i = f(v_i)$.

Skaitmeninis erdvės duomenų rinkinio interpoliacinis modelis (V,W) yra pora (Σ, Φ), kur Σ yra Ω, padalinta į celes $\sigma_1,\ldots,\sigma_m$, o Φ yra ją atitinkanti funkcijų $\phi_j : \sigma_j \to R$, $j = 1\ldots m$ šeima, kuri interpoliuoja W reikšmes visuose V taškuose. Jei Φ funkcijos yra sutampančios bendroje gretimų Σ celių sienoje, tuomet ištisinė funkcija Φ yra apibrėžiama

$\phi(p) = \phi_j(p)$, jei $p \in \sigma_j, \forall j = 1,\ldots,m$

Duotai $q \in R$, aibė $S(q) = \{p \in \Omega | \phi(p) = q\}$ yra vadinama dengiančiuoju Φ paviršiumi reikšmei q. *S(q)* yra apibrėžta panariui Σ celėse – kiekviena celė, priklausanti $\sigma_j \in \Sigma$ bei $\phi_j \le q \le \max_{\sigma_j} \phi_j$ yra vadinama aktyvia q reikšmei ir prisideda prie paviršiaus *S(q)* gabalo priklausomai nuo taško padėties

$S_j(q) = \{p \in \sigma_j | \phi_j(p) = q\}$

Dengiančiojo paviršiaus išrinkimas susideda iš visų paviršiaus gabalų $S_j(q)$ priklausančių aktyvioms celėms radimo duotiems (Σ,Φ) bei q reikšmei.

Kompiuterinėje grafikoje yra dvi pagrindinės dengiančiųjų paviršių klasės – parametriniai ir nuspėjami paviršiai. Bet kurio erdvės taško koordinates galima traktuoti dvejopai – parametrine bei nuspėjama prasme.

Nuspėjami paviršiai yra dviejų dimensijų geometrinės figūros, egzistuojančios trimatėje erdvėje. Nuspėjamas paviršius gali būti įsivaizduojamas kaip be galo maža dalis kokio nors išmatuojamo dydžio, kaip kad spalva, temperatūra ir kt. Tas dydis kinta erdvėje, tačiau yra pastovus visame paviršiuje. Taigi nuspėjamas paviršius susideda iš taškų trimatėje erdvėje, kurie atitinka tam tikrus reikalavimus. Matematiškai reikalavimas yra nusakytas funkcija f, kurios argumentas yra trimatis taškas p [5].

Nuspėjamas paviršius yra taškų aibė, kuriai $f(p) = c$, kur c – paviršiaus dengiančio kontūro reikšmė. Funkcija grąžina 0 toms p reikšmėms, kurios yra aibės dalis, ir 1 – kitais atvejais. f gali būti nusakoma diskretizuotais elementais, vienodu atstumu išsidėsčiusiais ribotoje erdvėje, taip pat matematinėmis funkcijomis, kuriose viena ar keletas išraiškų nustato taško p koordinates, arba procedūriniais metodais, kuomet algoritminis procesas nustato p reikšmes.

Parametrinis paviršius paprastai nusakomas funkcija f, kuri atvaizduoja keletą dvimačių paviršių į trimatę erdvę, kuomet numanomi paviršiai paprastai yra trimačių skaliarinių laukų $f(x,y,z)$ erdvinis atvaizdas. Bendru atveju parametriniai paviršiai yra aprašyti kaip funkcijų intervalas.

Parametriniu atveju koordinates nusakomos pagal tai, kokia yra geometrinė kūno forma. Bet kuris trimatis taškas gali būti nusakytas (s,t) pora: $x = f_x(s,t), y = f_y(s,t), z = f_z(s,t)$.

Parametrinius paviršius yra lengva apskaičiuoti įvertinant funkciją f skirtingoms parametrų reikšmėms srityje Ω. Gretimi elementai $p_1 = f(u_1, v_1)$ ir $p_2 = f(u_2, v_2)$ gali būti identifikuoti įvertinant skirtumus tarp atitinkamų parametrinių reikšmių (u_1, v_1) ir (u_2, v_2), tačiau trimatėje erdvėje turint tašką p nėra paprasta identifikuoti, ar jis priklauso parametriniam paviršiui, ar ne. Iš kitos pusės, nustatyti, ar duotas taškas p priklauso nuspėjamam paviršiui, yra paprasta, kadangi tam tereikia įvertinti funkciją f(p).

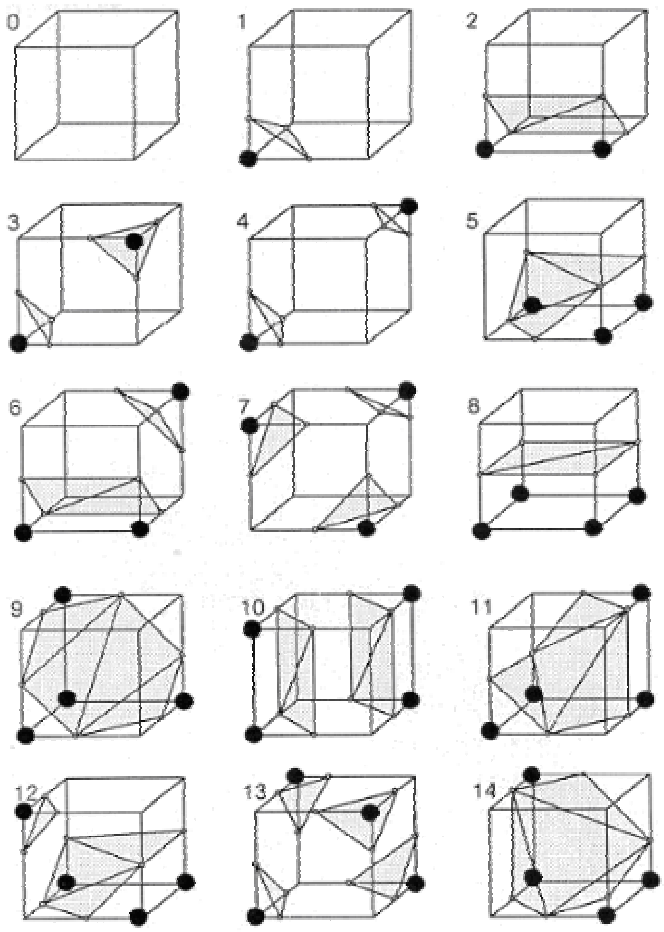
Vertimas iš parametrinių į spėjamus paviršius reikalauja apskaičiuoti paviršių atstumus, kuomet vertimas iš spėjamų į parametrinius dažniausiai atliekamas ieškant paviršių dalių ir sujungiant jas į daugiakampių tinklą. Kokios nors figūros gali būti nusakytos tiksliai tiek parametrine, tiek nuspėjama forma. Vertimas iš parametrinės į spėjamą formą ne visada lengvai realizuojamas. Vertimas iš spėjamos į parametrinę formą ne visuomet yra įmanomas, kadangi spėjami paviršiai, aprašyti aukštos eilės polinomais negali būti parametrizuoti racionaliomis funkcijomis. Vertimas į parametrinę formą visuomet įmanomas įprastiniams ketursieniams bei kubams.

1.2.2.2 Daugiakampiais paremtas vaizdavimas

Vaizduojant erdvę daugiakampiais, duomenų elementai sujungiami daugiakampių (dažniausiai trikampių) masyvu. Dauguma daugiakampiais paremto vaizdavimo algoritmų yra užpildančių kubų algoritmo (*Marching cubes*) modifikacijos. Šis bei kiti panašūs algoritmai traktuoja kiekvieną duomenų elementą kaip kokios nors geometrinės figūros viršūnę, paprastai kubo arba ketursienio. Tokia figūra laikoma viena erdvės cele.

Klasikinis užpildančių kubų algoritmas tikrina kiekvieną erdvę sudarančią celę ar ji nesikerta su dengiančiuoju paviršiumi. Vykdant algoritmą nusprendžiama, kaip apibrėžti paviršiaus dalį, kurią išskiria vienas kubas. Jei kiekvieną kampą apibrėšime kaip esantį aukščiau arba žemiau ribinės reikšmės, tuomet atsiranda 256 galimos kampų klasifikacijos kombinacijos. Iš jų dvi yra nereikšmingos – tai tuomet, kai visi taškai yra kubo viduje arba išorėje. Visiems kitiems atvejams reikalinga nustatyti, kur, išilgai kiekvienos kubo briaunos, eina ribinis paviršius, ir naudojantis šiais briaunų susikirtimo taškais sukurti vieną ar daugiau paviršiaus trikampių.

Įvertinus simetriškumus, iš esamu 254 kombinacijų lieka tik 15 unikalių. Šios pagrindinės būsenos pateiktos 2 paveikslėlyje.



2 pav. Pagrindinės celių aktyvių viršūnių kombinacijos

Kiekvienas iš prasmingų konfigūracijų rezultatų, susidedančių iš nuo 1 iki 4 trikampių įtraukiamas į dengiantįjį paviršių.

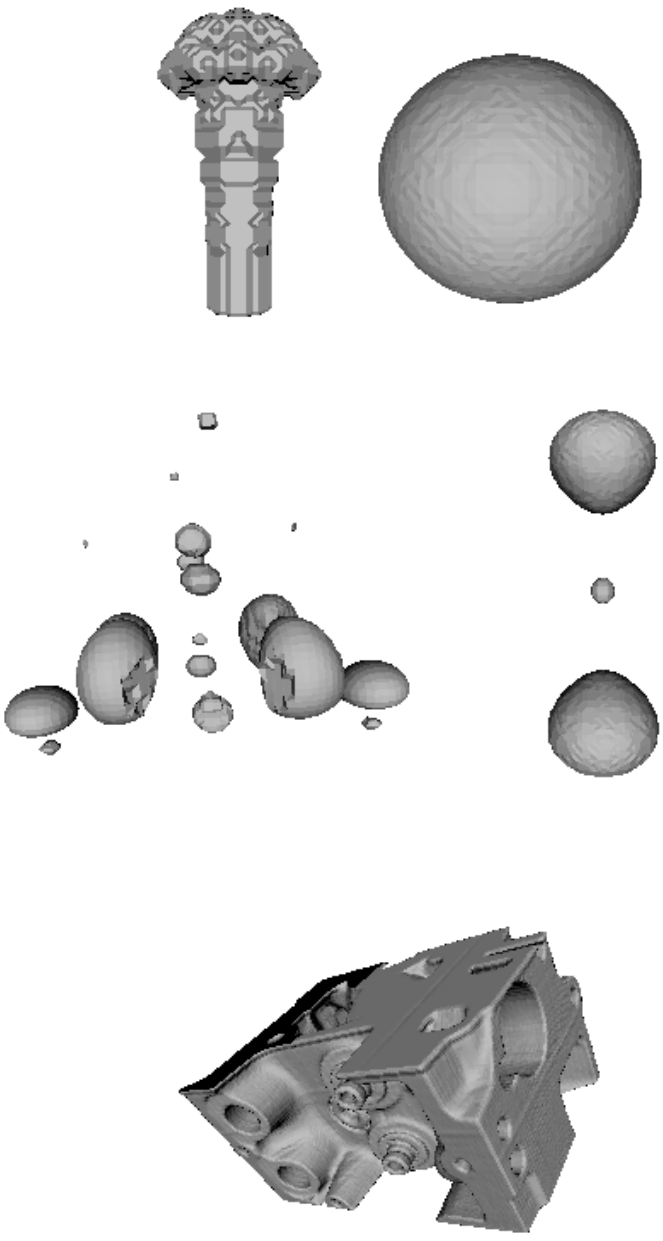
Klasikinio užpildančių kubų algoritmo principas:

1. Vartotojas nustato slenkstinę reikšmę
2. Nuskaitomos 8 duomenų dalys į atmintį
3. Iš duomenų dalių suformuojama celė
4. Klasifikuojamos aštuonios viršūnės ir sukuriami indeksai
5. Pasinaudojant indeksais apžvelgiamas kampų sąrašas
6. Tiesinio interpoliavimo būdu randamos 3 paviršiaus/kampų sankirtos
7. Išvedamos trikampių viršūnės bei viršūnių normalės

Viena šio algoritmo problemų yra rezultatų paviršiams saugoti reikalingas atminties kiekis. Kiekvienas ribojantis kubas gali sugeneruoti iki 4 trikampių. Kita iš problemų iškyla, jei nėra vokseliais užpildytos erdvės. Pagal tai, kokiu būdu buvo surinkti duomenys, čia gali atsirasti tuštumų, kuriose turėtų būti priskirtos reikšmės arba tuštumos turėtų būti apeinamos paviršiaus

generavimo algoritme. Be to, bet kokia interpoliuota reikšmė gali sumažinti galutinio gauto paviršiaus patikimumą.

Vykdant šį algoritmą būtina įvertinti dviprasmybes, atsirandančias tuomet, kai gretimų celių trikampių jungimas negali būti nusakytas vienareikšmiai. Toks netikslumas veda prie klaidų bei lūžių galutiniame paveiksle. Sprendžiant šia problema, vėlesnėse algoritmo modifikacijose buvo įvestos papildomos 6 būsenos.



3 pav. Užpildančių kubų algoritmo vykdymo rezultatai

1.2.2.3 Taškais paremtas vaizdavimas

Ilgą laiką standartinis tinkleliu paremtas vaizdavimo būdas buvo puikus pasirinkimas erdvės vaizdavime. Pagal šį principą erdvė išskaidoma į ketursienius, remiantis gautu tinkleliu suformuojami vokseliai, tolimesniuose etapuose remiantis vienu iš daugybės paviršių išskyrimo algoritmų vokseliai pakeičiami trikampių struktūromis. Tačiau atvaizduojant erdvinius duomenų rinkinius trikampiais neretai buvo susiduriama su nevientisumo problema, skylėmis galutiniame paveiksle ar atsirandančiomis objektų dalimis, kurių neturėtų būti pagal originalų vaizduojamą objektą. Šie netikslumai atsiranda dėl ne visuomet sėkmingo trikampių paviršių sujungimo į bendrą struktūrą.

Naujos techninės įrangos galimybės leido naujai pažvelgti į erdvinių duomenų rinkinių vaizdavimą. Vienas iš tokių naujų požiūrių tapo taškais paremtas erdvės vaizdavimas.

Kuomet trikampių skaičius viršija ekrano taškų skaičių, tokių didelių duomenų rinkinių vaizdavimas veda prie to, jog trikampis tampa projektuojamas į plotą, mažesnį nei vienas taškas. Šioje situacijoje tradiciniai taškų išgavimo metodai tampa neefektyvūs dėl pernelyg sudėtingo trikampių valdymo. Taškai tapo labiau tinkamas tokių didelių modelių vaizdavimo primityvas nei trikampis.

Taškais paremta geometrija gali būti traktuojama kaip ištisinio paviršiaus, esančio trimatės erdvės vietose p_i, paprastai susieto su normalės vektoriumi n_i bei papildomomis paviršiaus savybėmis , išrinkimas

Klasikiniuose tinkleliu paremtuose algoritmuose vokselius atitinkančios trikampių struktūros jungiamos remiantis tinkeliu, skaidančiu erdvę į vokselius. Pagrindinė taškais paremto vaizdavimo idėja yra ta, jog įprastinis tinklelis pakeičiamas taškų paviršiumi, kuris nėra griežtai apibrėžtas, o gauti iš taškų sudaryti paviršiai apdorojami spindulinėmis funkcijomis. Kiekvienam nustatytam erdvės taškui priskiriama sfera taip, kad gretimos sferos dalinai persidengtų. Tuomet atliekamas visų sferų projektavimas į ekraną, o visiems gautų diskų vidiniams taškams interpoliuojant apskaičiuojamas juos atitinkantis nuspėjamas paviršius.

Pagrindinė šio principo idėja yra paviršių vaizdavimui naudoti atskirus taškus nenaudojant jokių ryšių tarp jų vietoje tradicinio tinkleliu paremto vaizdavimo. Šis požiūris, kuriam vystytis leido didelis šiuolaikinių vaizdo plokščių atmintis kiekis, leidžia pasiekti kur kas didesnį vaizdo detalumą nei daugiakampiais paremtame vaizdavime, gauti aptakių formų trimatį vaizdą, o ryšių tarp taškų nebuvimas suteikia didesnį lankstumą jais manipuliuojant. Vaizduojant erdvę taškais

plačiai taikomos spindulinės bazinės funkcijos kaip priemonė išspręsti palaidų duomenų interpoliavimo problemą [7][13].

Taškais paremtame vaizdavime spindulinės bazinės funkcijos yra naudojamos duomenų išskaidymui bei patogiam prasmingų kaimyninių celių radimui [15]. Kuomet skaičiavimuose naudojamų duomenų elementų skaičius yra apribotas, sumažėja koeficientams apskaičiuoti reikalingų tiesinių lygčių sistemų dydis.

Nusakant nuspėjamą paviršių remiantis spindulinėmis bazinėmis funkcijomis, funkcija *f* turi tenkinti sąlygą

$$f(x_i) = h_i = 0,\ i = 1,...,n$$

visiems *n* prasmingų taškų x_i. Pagal susitarimą, dengiantis paviršius yra nusakytas nuline aibe $f(x) = 0$ ir turi teigiamas reikšmes viduje, o neigiamas – paviršiaus išorėje.

Paviršiuje nesančios reikšmės nusakomos taip

$$f(x_{n+j}) = h_{n+j},\ j = 1,...,m$$

visiems *m* paviršiuje nesančių taškų, kur $(h_i > 0)$, kuomet taškas yra viduje, arba $(h_i < 0)$, kuomet taškas paviršiaus išorėje.

Atkuriant paviršių atitinkama spindulinė bazinė funkcija $\phi : [0, \infty) \to R$ yra nekintama, ir, turint omenyje, jog $k = n + m$, interpoliavimo funkcija įgyja formą

$$f(x) = \sum_{i=1}^{k} w_i \phi(x - x_i) + P(x),$$

kur *P(x)* yra pirmos eilės polinomas, apskaičiuojamas tiesinėms bei pastovioms *f* dalims.

Remiantis šiomis trimis lygtimis galima sudaryti tiesinę lygčių sistemą, skirta apskaičiuoti spindulinių bazinių funkcijų svoriams w_i duotame duomenų taške

$$\begin{bmatrix} \phi_{11} & \phi_{12} & .. & \phi_{1k} & 1 & x_1^x & x_1^y & x_1^z \\ \phi_{21} & \phi_{22} & .. & \phi_{2k} & 1 & x_2^x & x_2^y & x_2^z \\ : & : & & : & : & : & : & : \\ \phi_{k1} & \phi_{k2} & .. & \phi_{kk} & 1 & x_k^x & x_k^y & x_k^z \\ 1 & 1 & .. & 1 & 0 & 0 & 0 & 0 \\ x_1^x & x_2^x & .. & x_k^x & 0 & 0 & 0 & 0 \\ x_1^y & x_2^y & .. & x_k^y & 0 & 0 & 0 & 0 \\ x_1^z & x_2^z & .. & x_k^z & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ : \\ w_k \\ 1 \\ p^x \\ p^y \\ p^z \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ : \\ h_k \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Šioje sistemoje $\phi_{ij} = \phi(\|x_i - x_j\|)$, kur $\|.\|$ žymi Euklido atstumą.

Taškais paremtame vaizdavime taip pat naudojamos riboto poveikio spindulinės bazinės funkcijos, pavyzdžiui $\phi(x) = 0$, kuomet $\|x\| \geq r$, kur r – poveikio spindulys. Naudojant riboto

poveikio spindulines bazines funkcijas, tiesinė lygčių sistema tampa išretinta tiesine lygčių sistema, kadangi $\phi(\|x_i - x_j\|) = 0$ visoms (x_i,x_j), labiau nutolusiems nei poveikio spindulys.

Vienas iš taškais paremto vaizdavimo algoritmų yra dengiantysis taškymas (*iso-splatting*). Klasikiniame trikampiais paremtame vaizdavime yra nustatinėjamos tinklelio celės, kurias kerta dengiantysis paviršius. Šioms celėms yra apskaičiuojami briaunų kirtimo taškai, toliau apskaičiuojami juos atitinkantys trikampiai ir įtraukiami į modelį. Dengiančiajame taškyme vietoj to, kad celės viduje būtų generuojama trikampių aibė, generuojamas taškas [8]. Šiam algoritmui kiekvienas duomenų taškas turi būti nusakytas padėtimi erdvėje, normalės vektoriumi bei laisvumo laipsniu, nusakančiu vidutinį atstumą iki kaimyninių duomenų taškų. Nuoseklūs tiesiniai erdviniai duomenys suteikia pilną informaciją apie erdvės elementų padėtį. Laisvumo matrica gali būti gauta žinant tinklelio intervalą.

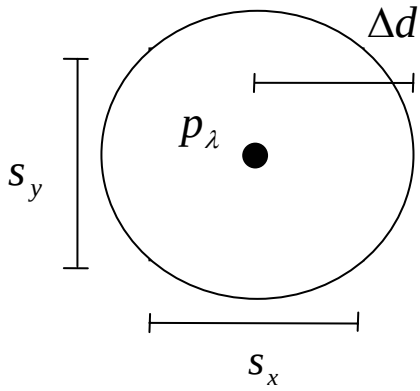
Taškai nebūtinai turi būti pageidaujamame dengiančiajame paviršiuje, todėl turi būti projektuojami į jį. Tam naudojama apytikris projektavimas. Taško p_λ aproksimacijos p'_λ radimas atliekamas pasinaudojant viena Niutono-Rafsono šaknies radimo metodo iteracija. Duotai funkcijai $f(x)$ bei ribinei reikšmei f_I yra ieškomos lygties $h(x) = f(x) - f_I$ šaknys. p'_λ nusakoma kaip

$$p'_\lambda = p_\lambda + \nabla f(p_\lambda)t\,.$$

Niutono-Rafsono metodas konverguoja kvadratiškai, kuomet pradinė spėjama x_0 reikšmė yra pakankamai artima šaknies reikšmei. Šis metodas gali apskaičiuoti šaknis, esančias toli nuo pageidaujamos reikšmės, kuomet x_0 yra artimas lokaliam minimumui arba maksimumui. Dėl šios priežasties yra atliekamas šaknų patikrinimas ir atmetamos klaidingos reikšmės. Šaknų atmetimas atliekamas remiantis poslinkio kriterijumi. Jei po vienos Niutono-Rafsono iteracijos taškas yra pernelyg toli nuo pradinės reikšmės, tuomet galutiniame vaizde šis taškas nėra įvertinamas. Poslinkio slenkstinė reikšmė Δd apskaičiuojama remiantis vienos celės išmatavimais. Turint vienos celės išmatavimus s_x, s_y, s_z, poslinkio slenkstinė reikšmė yra nustatoma pagal formulę

$$\Delta d = \frac{1}{2}\sqrt{s_x^2 + s_y^2 + s_z^2}\,.$$

4 pav. pateikta geometrinė poslinkio kriterijaus interpretacija. Kvadratas žymi celę, taškas – duomenų tašką p_λ. Šis duomenų taškas gali būti projektuojamas į celę kertantį dengiantįjį paviršių tuo atveju, jei projekcija neperžengia poslinkio ribų, nusakytų sfera, kurios spindulys yra Δd, apibrėžtas celės išmatavimais.



4 pav. Poslinkio kriterijus

1.2.2.4 Taškymas

Taškymas – tai erdvinių duomenų vaizdavimo būdas, paremtas persidengiančių bazinių funkcijų projektavimu į ekraną. Dažniausiai šiuo principu paremtame vaizdavime naudojamasi Gauso branduoliais, kurių amplitudės padaugintos iš vokselio reikšmės. Tokiu būdu vokseliai, kurių reikšmės yra didesnės atvaizduojami ryškiau nei mažesnes turintys mažesnes reikšmes. Tuomet šios funkcijos yra projektuojamos į ekraną ir gaunamas erdvinių duomenų vaizdas. Originaliame algoritmo variante įvertinamos visos vokselių reikšmės, labiausiai lygiagrečios paveikslo plokštumai. Patobulintuose šio algoritmo variantuose buvo įtrauktas duomenų plokštumų įvertinimas bei vaizdavimui pradėtos naudoti tik lygiagrečios plokštumos. Vokseliai taškomi remiantis jų atstumu nuo paveikslo plokštumos, taigi, arčiausiai esantys vokseliai taškomi pirmiausiai. Visu pirma tokiu būdu atvaizduojama visa arčiausiai esanti plokštuma, ir tik tada einama prie sekančios. Atlikus taškymą įvertinamas kiekvieno vokselio indėlis į galutinį paveikslą. Funkcijos branduolio projekcija į ekraną vadinama pėdsaku. Remiantis iš anksto paruoštomis lentelėmis spalvos bei nepermatomumo reikšmės yra suliejamos kiekviename taške, kuris patenka į Gauso funkcijos pėdsako plotą [9][10].

1.2.2.5 Erdvės vaizdavimo algoritmų trūkumai

Kuomet erdviniai duomenys yra vaizduojami paviršiais, prarandama daugybė informacijos. Nepermatomas paviršius, einantis per visas celes, kurios atitinka tam tikrą funkcijos reikšmę, atvaizduoja tik dalį duomenų rinkinio, kadangi erdvės dalys prie kitų funkcijos reikšmių yra tiesiog ignoruojamos. Netiesioginis erdvės vaizdavimas daugiakampiais yra patogus objektams su griežtai apibrėžtais kraštais, bet jis netinkamas amorfiniams objektams, turintiems mažus nuolydžius,

kuriuos sunku atvaizduoti smulkiais paviršiais. Vaizduojant daugiakampiais susiduriama su lūžių bei skylių vaizde problemomis. Šias problemas iš dalies sprendžia vaizdavimas taškais, tačiau čia iškyla nauji uždaviniai – didelis atminties kiekis bei gerokai didesnis, nei vaizduojant daugiakampiais, skaičiavimų poreikis. Vaizdavimas dengiančiais paviršiais yra nepatogus nežinomiems duomenų rinkiniams. Tokiu atveju duomenų rinkinys gali būti suprastas tik tuo atveju, jei duomenys apdorojami esant skirtingoms ribinėms reikšmėms ir vizualiai nustatoma, pagal kurią iš jų atkurtas vaizdas yra tiksliausias. Ribinės reikšmės keitimas yra dar viena probleminė sritis vaizdavime dengiančiais paviršiais – pakeitus ribinę reikšmę yra reikalinga iš naujo pereiti duomenų rinkinį ir atlikti dengiančiojo paviršiaus išrinkimą. Šiai problemai spręsti naudojami išankstiniai daliniai skaičiavimai bei indeksavimas. Duomenys pačioje pradžioje yra apdorojami, atliekant dalį skaičiavimų bei išrenkant bendrą informaciją apie duomenų rinkinį. Tuomet, pakeitus ribinę reikšmę, naudojamais ankščiau atliktų skaičiavimų rezultatais. Toks išankstinis duomenų apdorojimas padeda išvengti pasikartojančių skaičiavimų.

Tiesioginio erdvės vaizdavimo bei taškymo palyginimą galima rasti [6]. Taškymo algoritmo pranašumas prieš tiesioginį erdvės vaizdavimą akivaizdus, kuomet vaizduojami duomenų rinkiniai, kuriuose prasmingų celių skaičius sudaro mažą dalį bei vaizduojami objektai yra nepermatomi („Blood vessel“ duomenų rinkinys). Iš kitos pusės, tuo atveju, kai prasmingų celių duomenyse yra daug, tiesioginis erdvės vaizdavimas yra pranašesnis („Shockwave“ duomenų rinkinys). Kuomet vaizduojami nepermatomi objektai, taškymui tereikia keleto artimiausių paveikslo plokštumai duomenų plokštumų, kurios projektuojamos į ekraną, o tuo metu tiesioginiame erdvės vaizdavime vis tiek atliekamas spindulių trasavimas per ekrano taškus.

1.2.2.6 Hierarchinių duomenų struktūrų panaudojimas erdvės vaizdavime

Vaizduojant erdvę, daug resursų išnaudojama dirbant su celėmis, kurios nedaro įtakos galutiniam duomenų rinkinio atvaizdui. Šiai bei kitoms erdvės vaizdavimo problemoms spręsti sėkmingai buvo pritaikytos hierarchinės duomenų struktūros, tokios kaip aštuonetainiai medžiai, K-d medžiai.

Pirma kartą pilnai veikiantis paviršių išrinkimo algoritmas, paremtas hierarchinėmis duomenų struktūromis buvo pasiūlytas Wilhelm bei Van Gelder. Šis algoritmas paremtas efektyviu aštuonetainio medžio panaudojimu erdvinių duomenų saugojimui. Hierarchinė aštuonetainio erdvės dalinimo prigimtis leidžia paprastai atmesti didelius srities gabalus, nereikalaujant jokios užklausos į atmetamą sritį. Šis algoritmas, dar vadinamas BONO saugo maksimalią bei minimalią skaitmenines regionų, apimamų kiekvienos medžio dalies, reikšmes regiono tėvo mazge.

Algoritmas gali atsikartojimo būdu apeiti medį, išrinkdamas dengiančius paviršius tik tose medžio dalyse, į kurių intervalus patenka peržiūrima reikšmė. Pats tokio medžio principas leidžia lengvai atmesti didžiules erdvės dalis neatliekant jokių užklausų į žemesnio lygmens medžio atšakas. Klasikinio užpildančių kubų algoritmo vykdymo laikas naudojantis tokia duomenų struktūra gali būti sumažintas keletą kartų [12].

Kitas hierarchinės duomenų struktūros tipas yra K-d medžiai, arba balansuotas dvejetainis medis. Šio medžio šakninė celė atitinka visą vaizduojamą erdvę. Kitos celės atitinka stačiakampes dalines erdves, saugančias savyje įvairias erdvės charakteristikas. Kuriant K-d medį, darbas pradedamas nuo šakninės celės ir rekursiniu būdu dalinamos celės išilgai jos ilgiausios ašies, tam, kad kiekvienoje dalinėje erdvėje būtų vienodas dalių skaičius. Paprastai K-d medis naudojamas vietoje aštuonetainio medžio dėl paprastesnės struktūros bei itin sparčių algoritmų, leidžiančių itin efektyvų medžio sudarymą. Rodyklės nėra būtinos, kadangi kiekvienas medžio mazgas gali būti indeksuotas, taigi vaiko, tėvo ar broliško mazgo paieška tampa paprasta bitų stumdymo operacija.

Erdvės dalinimas – dar vienas metodas skirtas sumažinti skaičiavimų skaičių. Pagal šį metodą iš pradžių duomenų rinkinys atvaizduojamas grubiai. Toliau prasmingos sritys yra detalizuojamos, dalinant jas į smulkesnes dalis bei atvaizduojant jau didesne skiriamąja geba.

Hierarchinis dalinimas gali būti paremtas erdve, pvz., vokselio priklausomybė vienam ar kitam žemesnio lygmens medžiui nustatoma pagal vokselio viršūnės koordinates. Kitas atvejis – intervalu paremtas požiūris, kuomet vokseliai klasifikuojami pagal tai, koks skaitmeninių reikšmių intervalas egzistuoja celėse. Gauta perdengianti erdvė yra vėl apdorojama hierarchinėmis duomenų struktūromis.

1.3 Išvados

Kai kurie aukščiau paminėtų vaizdavimo metodų privalumai bei trūkumai pateikti 1 lentelėje:

1 lentelė. Pagrindiniai vaizdavimo metodų privalumai bei trūkumai

		Privalumai	Trūkumai
Vaizdavimo metodas	Tiesioginis erdvės vaizdavimas	Patogus vaizduoti amorfinius objektus	Algoritmo sudėtingumas Skaičiavimų kiekis
	Taškymas	Vaizdavime dalyvauja tik tie vokseliai, kurie turi įtakos galutiniam vaizdui	Netinkamas didelį prasmingų celių turinčių duomenų rinkinių vaizdavimui
	Daugiakampiais paremtas erdvės vaizdavimas	Paprastumas, bereikalingas didelis skaičiavimų kiekis	Reikalingas didelis atminties kiekis saugoti informacijai apie trikampius. Perteikiama tik dalis informacijos
	Taškais paremtas vaizdavimas	reikalinga informacija apie ryšius, gauto vaizdo vientisumas	Didelės atminties sąnaudos, kadangi naudojamas didžiulis grafinių primityvų kiekis.

1. Nėra nei vieno idealaus erdvės vaizdavimo algoritmo – kiekvienas iš esamų yra paremtas kompromisais, atminties sąskaita mažinant skaičiavimų kiekį arba atvirkščiai.
2. Pagrindinis tiesioginio erdvės vaizdavimo trūkumas – didelio skaičiavimų kiekio poreikis, algoritmo sudėtingumas.
3. Netiesioginis erdvės vaizdavimas paremtas apytikrių vaizduojamos erdvės projektavimu į ekraną. Jau pats tokio vaizdavimo principas neleidžia atvaizduoti duomenų rinkinio pilnai, dėl ko prarandama didelė informacijos dalis.
4. Netiesioginiame erdvės vaizdavime reikalingi dideli atminties kiekiai – vaizduojant daugiakampiais reikia saugoti informaciją apie viršūnių sujungimus, vaizduojant taškais – reikia saugoti didelį taškų kiekį.

5. Taškais paremtame vaizdavime plačiai naudojamos spindulinės bazinės funkcijos, kaip priemonė kuriant paviršių aproksimaciją.
6. Taškais paremtas vaizdavimas taikytinas tuomet, kai svarbiau yra ne galutinio duomenų rinkinio atvaizdo tikslumas, bet jo vaizdumas.
7. Hierarchinės duomenų struktūros (medžiai) – vienas plačiausiai taikomų priemonių erdvės vaizdavimo algoritmų optimizavimui. Medžių taikymas reikalauja papildomų operacijų darbo su duomenų rinkiniu pradžioje, tačiau leidžia ženkliai sumažinti skaičiavimų kiekį vėlesniuose darbo etapuose.
8. Kiekviena naujovė, pagerinanti vaizdavimą vienu aspektu, pareikalauja papildomų sąnaudų kitose srityse.
9. Nemažai naujesnių algoritmų sukurti sujungiant keletą skirtingais principais veikiančių, ankščiau sukurtų metodų.

2 Eksperimentinė dalis

2.1 Betinklis dengiančiojo paviršiaus išskyrimas iš keletą blokų turinčių duomenų

2.1.1 Algoritmas

Duota aibė pavyzdinių taškų S, aprašyta kaip taškų aibė $p_i = (x_i, y_i, z_i), i = 1,2,...,|S|$. Sakykime, kad F_i yra skaliarinė reikšmė, susieta su p_i. H reikšmė iterpoliuojama taške $p = (x, y, z)$ naudojantis multikvadratinį (*multiquadratic*) metodą [2].

Funkcijos reikšmei apskaičiuoti naudojama tokios formos iterpoliacija:

$$H(x, y, z) = \sum_{j=1}^{N} a_j B_j(x, y, z), \quad (1)$$

kur kiekvienas B_j yra multikvadratinė funkcija

$$B_j(x, y, z) = \sqrt{(x_j - x)^2 + (y_j - y)^2 + (z_j - z)^2 + R^2}.$$

Ryšiai:

$$H(x_j, y_j, z_j) = F_j, \text{ j= 1,2,…,N} \quad (2)$$

Funkcijos gradientas nusakomas

$$\nabla H(x, y, z) = (\frac{\delta H}{\delta x}, \frac{\delta H}{\delta y}, \frac{\delta H}{\delta z}),$$

kur dalinė išvestinė $\frac{\delta H}{\delta x}$ apskaičiuojama pagal formulę

$$\frac{\delta H}{\delta x} = \sum_{j=1}^{N} \frac{x - x_j}{B_j(x, y, z)}. \quad (3)$$

Dalinė išvestinė pagal y ir z apskaičiuojama atitinkamai pagal (3) formulę.

R yra vadinamas multikvadratiniu parametru, o a_j – perėjimo koeficientas. Multikvadratinis parametras paprastai yra fiksuotas dydis. Perėjimo koeficientai yra išskaičiuojami sprendžiant tiesinę lygčių sistemą, sudarytą iš (2) ryšių. N yra duomenų taškų skaičius, naudojamų reikšmės aproksimavimui. Bendru atveju N = |S|, tačiau kuriant lokalią schemą įvertinama tik N taškų aplink p tam tikroje apibrėžtoje kaimynystėje.

Dalimis nusakytas lokalus interpoliavimas negali garantuoti interpoliavimo vientisumo, todėl atskiros dalys turi būti sulietos. Tam paprastai naudojamos visumos dalinimo (*partition of unity*)

formulės, kurių pagalba lokaliai apibrėžtos interpoliavimo funkcijos suliejamos į vieną globalų iterpoliatorių.

Visumos dalinimo funkcijos Φ_k turi savybę, kad visoje funkcijos srityje

$$\sum_k \phi_k(x, y, z) \equiv 1$$

Duotoms M neneigiamų funkcijų W_k, galima apibrėžti visumos dalinimo funkcijų aibę kaip

$$\Phi_k(x, y, z) = \frac{W_k(x, y, z)}{\sum_{q=1}^{M} W_q(x, y, z)}$$

Globalus ištisinis iterpoliatorius F(x, y, z) nusakomas

$$F(x, y, z) = \sum_{k=1}^{M} \Phi_k(x, y, z) H_k(x, y, z),$$

kur H_k yra lokali SBF, apibrėžta kaip funkcija (1).

Tam, kad nusakyti lokalų interpoliatorių H_k, visų pirma panaikinami tinkelio ryšiai ir dirbama tik su duomenų taškais. Šie duomenų taškai tuomet yra apjungiami į globalų reguliarų tinklelį, kuris vadinamas SBF tinkleliu, nusakomu kaip ribojantys kubai aplink duomenų taškus. Kiekvienas lokalus interpoliatorius H_k yra susiejamas su kiekvienos celės centru c_k. Šis taškas vadinamas SBF centru. Įtakos sritis aplink kiekvieną SBF centrą apskaičiuojama pagal formulę:

$$R_w = 2\sqrt{{s_x}^2 + {s_y}^2 + {s_z}^2},$$

kur (s_x, s_y, s_z) yra vienos SBF tinklo celės dydis. Lokalaus SBF, susieto su centru c_k, skaičiavimo koeficientai yra apskaičiuojami įvertinant visus taškus sferoje, nusakytoje SBF centru c_k bei įtakos spinduliu R_w. Taškai, reikalingi apskaičiuoti šiuos parametrus yra gaunami tikrinant kiekvieną taškai gretimose 26 celėse – aštuoni viršūnių kaimynai, 12 briaunų kaimynų bei 6 sienelių kaimynai. Jei šis regionas yra tuščias, tuomet celėje SBF centras nekuriamas.

Kuriant SBF tinklą, celes stengiamasi padaryti kuo panašesnes į kubą. Tai pasiekiama apskaičiuojant SBF tinklo dydį n_x, n_y, n_z, naudojantis formulėmis

$$s = \sqrt[3]{(m\frac{w_x w_y w_z}{n})},$$

$$n_x = round(\frac{w_x}{s}),$$

$$n_y = round(\frac{w_y}{s}),$$

$$n_z = round(\frac{w_z}{s})$$

kur n yra duomenų taškų skaičius visame duomenų rinkinyje, (w_x, w_y, w_z) yra ribojančios dėžės dydis, apimantis visus duomenų tinklelius, ir *m* yra vartotojo nusakytas parametras, griežtai nusakantis kiek duomenų taškų patalpinti į kiekvieną celę.

Kuomet vertinama funkcija taške $p = (x, y, z)$, visų pirma nustatoma kurioje SBF celėje taškas p randasi tam, kad būtų galima apskaičiuoti lokalų SBF interpoliatorių H_k. Taip pat surenkami 26 gretimų celių lokalius SBF interpoliatorius. Taške *p* šie 27 interpoliatoriai yra įvertinami bei sujungiami naudojantis visumos dalinimo funkcija, iš kur gaunama galutinė funkcijos reikšmė bei gradientas.

Atvirkštinės atstumo funkcijos

$$W_k(x, y, z) = \left(\frac{(R_w - d_k)_+}{R_w d_k} \right)^2$$

$$(R_w - d_k)_+ = \begin{cases} R_w - d_k, jei\ d_k < R_w \\ 0 \qquad , jei\, d_k \geq R_w \end{cases}$$

naudojamos kaip svorinės funkcijos, skirtos sugeneruoti visumos dalinimo funkciją, kur d_k yra Euklido atstumas nuo SBF centro c_k iki taško p. Jei celė neturi su ja susieto lokalaus interpoliatoriaus, tuomet ji neįvertinama skaičiavimuose.

2.1.2 Algoritmo modifikacijos

Vykdant šį algoritmą kiekvienas užpildančių kubų algoritmo vykdymo metu gautas trikampis verčiamas jį atitinkančia taškų aibe. Kiekvienam iš gautų taškų apskaičiuojama funkcijos reikšmė, įvertinanti kiekvienos iš jį įtakojančių sferų poveikį. Gauta funkcijos reikšmė, esama taško pozicija bei su tašku susieta gradiento reikšmė naudojama Niutono-Rafsono metodu ieškant tikrosios taško pozicijos.

Vienas iš šio algoritmo trūkumų yra tas, jog visiškai nesistengiama įvertinti vaizduojamo objekto. Vienodai yra apdorojami duomenys, saugantys informaciją tiek apie aptakios formos kūnus, tiek apie plokščias figūras. Originaliame algoritme kiekvienas trikampis pakeičiamas vienodo intervalo taškų tinkleliu. Kiekvienas taškas priklausomai nuo jo poslinkio nuo pradinės pozicijos yra išplečiamas siekiant panaikinti atsiradusias skyles vaizde. Sugeneruojama taškų aibė yra pakankama atvaizduoti itin kreivoms sritims, tačiau mažiau kreiviems paviršiams toks primityvų kiekis gali būti perteklinis. Kadangi kiekvienam tokiam taškui atliekamas funkcijos reikšmės skaičiavimas bei vykdomos šaknies radimo procedūros, susidaro daugybė perteklinių

skaičiavimų. Greta to, aibės perteklinių taškų saugojimui reikalingas didelis papildomos atminties kiekis, jų vaizdavimui – papildomi vaizdo plokštės resursai.

Jeigu vykdant algoritmą būtų įvertinamas paviršiaus sričių kreivumas, būtų galima identifikuoti mažiau kreivus paviršiaus plotus bei juose generuoti retesnę taškų aibę. Sumažinus trikampį užpildančių taškų skaičių atitinkamai sumažėtų paviršių kūrimui reikalingų skaičiavimu kiekis, saugojimui / vaizdavimui reikalingos atminties kiekis bei sutrumpėtų taškų generavimui reikalingas laikas.

Paviršiaus plokštumas nustatomas skaičiuojant taškų funkcijų reikšmes. Iš pradžių apskaičiuojamos kiekvieno trikampio viršūnių tikrosios pozicijos. Remiantis šiais trimis taškais sudaroma plokštumos lygtis. Tuomet imamas taškas trikampio centre bei apskaičiuojama jo tikroji reikšmė. Jei galutinė taško pozicija yra ankščiau apskaičiuotoje plokštumoje, tuomet daroma prielaida, jog šioje vietoje paviršius yra plokščias bei apskaičiuojamas taško dydis, galintis padengti šią sritį. Jei taškas į plokštumą nepatenka, tuomet apskaičiuojama trikampį užpildanti taškų aibė, kiekvienam taškui skaičiuojama funkcijos reikšmė bei vykdoma Niutono-Rafsono procedūra. Laikoma, jog taškas patenka į plokštumą tuo atveju, jei jo atstumas iki plokštumos mažesnis nei pasirinkta paklaida.

2.1.3 Algoritmo aprašymas

Algoritmo etapai

1. Vykdomas duomenų nuskaitymas į struktūras

Pradiniai duomenys patalpinami į dvi struktūras, kurių viena saugo visų duomenų viršūnių koordinates bei reikšmes, o antroji – sugrupuotas viršūnes, kur viena grupė sudaro vieną vokselį. Viršūnių grupavimas į celes reikalingas saugant duomenis, sudarytus iš keleto tinklelių. Užpildančių kubų algoritmas apdoroja po vieną celę, neatsižvelgdamas nei į tinklelį, kuriame celė yra, nei į tinklelio charakteristikas.

2. Sukuriamas SBF tinklelis

Tinklelis formuojamas parinkus parametro *m* reikšmę lygią 3. Parametras *m* apytikriai nusako, kiek duomenų taškų turi būti patalpinama į viena SBF tinklelio celę. Kiekvienai celei sudaromas masyvas, saugantis informaciją apie celei priklausančias viršūnes.

3. Sukuriamos SBF sferos

Sferos kuriamos einant per visas SBF tinklelio celes, kiekvienai iš jų įvertinant gretimas celes. Gretimos celės tai aštuonios gretimos viršūnėms celės, dvylika gretimų kampams bei šešios

gretimos sienelėms celės. Jei šios celės pasirodo beesančios tuščios, tuomet sfera aktyviai celei nėra kuriama.

4. Įvykdomas užpildančių kubų algoritmas

Bendru atveju vykdant užpildančių kubų algoritmą gali atsirasti skylių paveiksle. Toks rezultatas gaunamas dėl tam tikrų gretimų kubų kombinacijų, kuomet susidariusi aktyvių viršūnių struktūra gali būti interpretuojama dvejopai. Nagrinėjamajame algoritme į tai neatsižvelgiama – užpildančių kubų algoritmas vienu metu apdoroja tik vieną celę, neatsižvelgdamas į aplinkines. Skyles uždengiamos atliekant vėlesniuose etapuose gautų trikampį užpildančių taškų postūmius bei dydžių keitimą.

5. Kiekviename trikampyje remiantis svorinėmis koordinatėmis sukuriama taškų aibė

Trikampiai, kuriuose kuriamos taškų aibės, gaunami įvykdžius užpildančių kubų algoritmą.

6. Kiekvienam iš šių taškų apskaičiuojamos svorių bei visumos dalinimo funkcijų reikšmės

Taškų svoriai apskaičiuojami priklausomai nuo jų atstumo iki konkrečios SBF sferos centro.

7. Kiekvienam taškui vykdoma Niutono-Rafsono procedūra ir apskaičiuojama tikroji jo pozicija
8. Pagal taško poslinkį nuo pradinės pozicijos apskaičiuojamas jo dydis

Taško spindulys apskaičiuojamas pagal formulę:

$$r = 2\left(1 + \frac{d}{l}\right)\sqrt{\frac{2A}{\pi n(n+1)}},$$

kur A – trikampio plotas, n – taškų skaičius, l – celės, kurioje yra trikampis, įstrižainė, ir d – Euklido atstumas tarp originalios taško pozicijos bei jo galutinės apskaičiuotos vietos. Trikampyje sukurtų taškų skaičius yra $\frac{n(n+1)}{2}$. Taško dydžio priklausomybė nuo poslinkio reikalinga užpildyti skyles paveiksle, taškui pasislinkus didesnius atstumu.

Modifikuotame algoritme 5 punktas yra pakeičiamas keletu papildomų operacijų. Vietoje to, kad būtų kuriama trikampį užpildančių taškų aibė, trikampis įvertinamas:

1. Apskaičiuojami trikampio viršūnių taškų poslinkiai;
2. Nustatoma plokštuma, einanti per šiuos taškus;
3. Remiantis svorinėmis koordinatėmis trikampis išskaidomas į tris mažesnius trikampius;

4. Nustatomi taškų, esančių kiekvieno vidinio trikampio centre bei pradiniame trikampyje esančio taško, žyminčio bendrą vidinių trikampių viršūnę, poslinkiai priklausomai nuo interpoliacinės schemos;
5. Jei visi šie taškai, įvertinus paklaidą, priklauso ankščiau nustatytai plokštumai, tuomet tolimesnis taškų kūrimas nutraukiamas, o esami taškai išplečiami;
6. Jei taškai nepriklauso plokštumai, tuomet daroma išvada, jog paviršius itin kreivas, ir generuojama vientisa tanki taškų aibė;

2.1.4 Rezultatai

2 lentelė. Testavimui naudoti duomenų rinkiniai

Duomenų rinkinys	Matmenys, viršūnėmis
Nucleon	41x41x41
Fuel	64x64x64
Neghip	64x64x64
Hydrogen Atom	128x128x128
Engine	256x256x128

2 lentelėje pateikti testavimui naudotų duomenų rinkinių sąrašas. Kiekvieno rinkinio celės yra taisyklingi kubai, tinklelio žingsnis yra 1. Visi šie duomenų rinkiniai sudaryti iš vieno taisyklingo reguliaraus struktūrizuoto tinklelio.

3 lentelė. Algoritmo vykdymo su vieno tinklelio duomenimis rezultatai

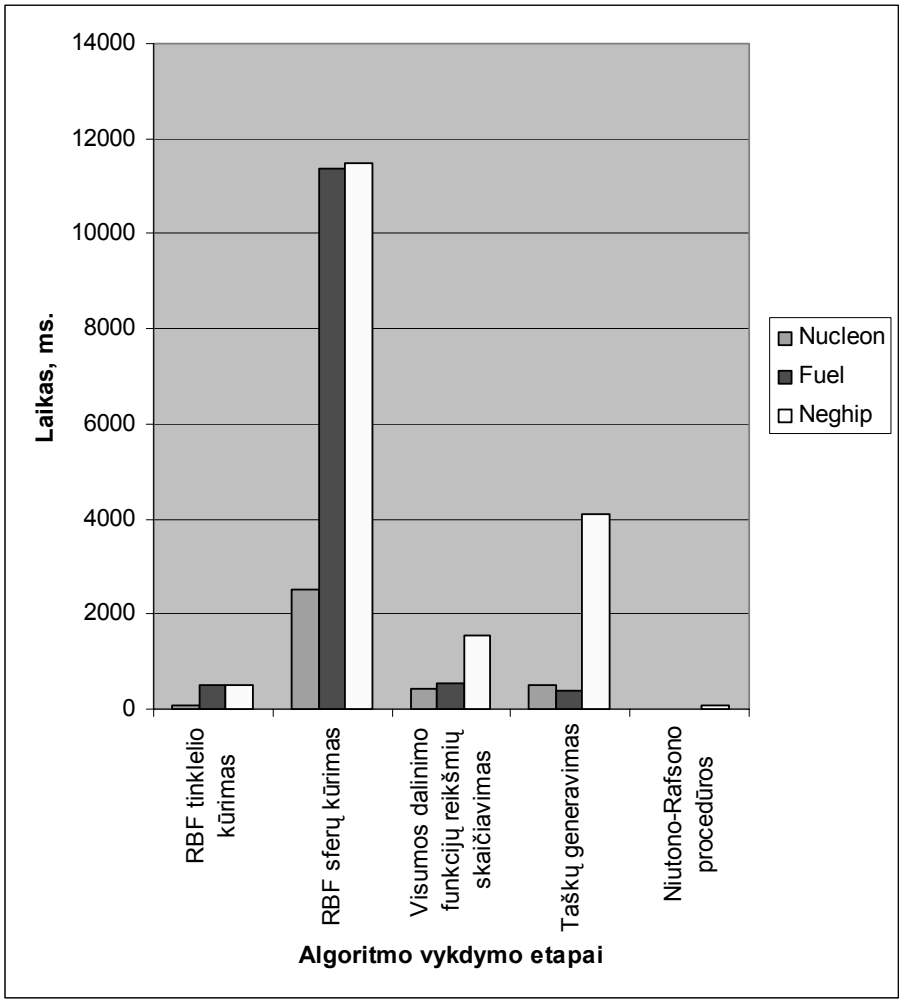
Etapai	Duomenų rinkinys				
	Nucelon	Fuel	Neghip	Hydrogen Atom	Engine
SBF tinklelio kūrimas	94ms	515ms	520ms	n.d.	n.d.
SBF sferų kūrimas	2500ms	11355ms	11500ms	n.d.	n.d.
Koeficientų skaičiavimas	~800000ms	~600000ms	~1500000ms	n.d.	n.d.
Visumos dalinimo funkcijų reikšmių skaičiavimas	422/420ms	550/545ms	1550/1500ms	n.d.	n.d.
Taškų generavimas	500/500ms	400/410ms	4100/3960ms	n.d.	n.d.
Niutono-Rafsono procedūros	18/18ms	17/20ms	60/57ms	n.d.	n.d.

SBF tinklelio kūrimo laikas skirtingiems vienodo dydžio duomenų rinkiniams užtrunka vienodai, kadangi kuriant SBF tinklelį duomenų rinkinys pereinamas vieną kartą, suskirstant visus duomenų elementus į jiems priklausančias celes. Panašus rezultatas gaunamas ir skaičiuojant interpoliacinės schemos elementus.

Kuriant SBF sferas, laikai net ir vienodiems duomenų rinkiniams svyruoja, nors kuriant vieną karta apeinamas visas SBF tinklelis. Vykdymo trukmė svyruoja todėl, kad į sferas priskiriami tik tie duomenų elementai, kurių reikšmės didesnės nei ribinė reikšmė.

Visumos dalinimo funkcijų reikšmių skaičiavimo rezultatai skiriasi net esant ir vienodo dydžio duomenų rinkiniams. Skaičiuojant šių funkcijų reikšmes skaičiavimo trukmė priklauso ne nuo pradinių, bet nuo prasmingų celių skaičiaus. Šio etapo vykdymo laiką įtakoja ir tai, kiek taškų generuojama kiekvieno trikampio viduje.

Taškų generavimas bei Niutono-Rafsono procedūrų vykdymo laikai taip pat priklauso nuo to, kiek trikampių gaunama atlikus užpildančių kubų algoritmą bei kiekvieną trikampį užpildančių taškų skaičių.

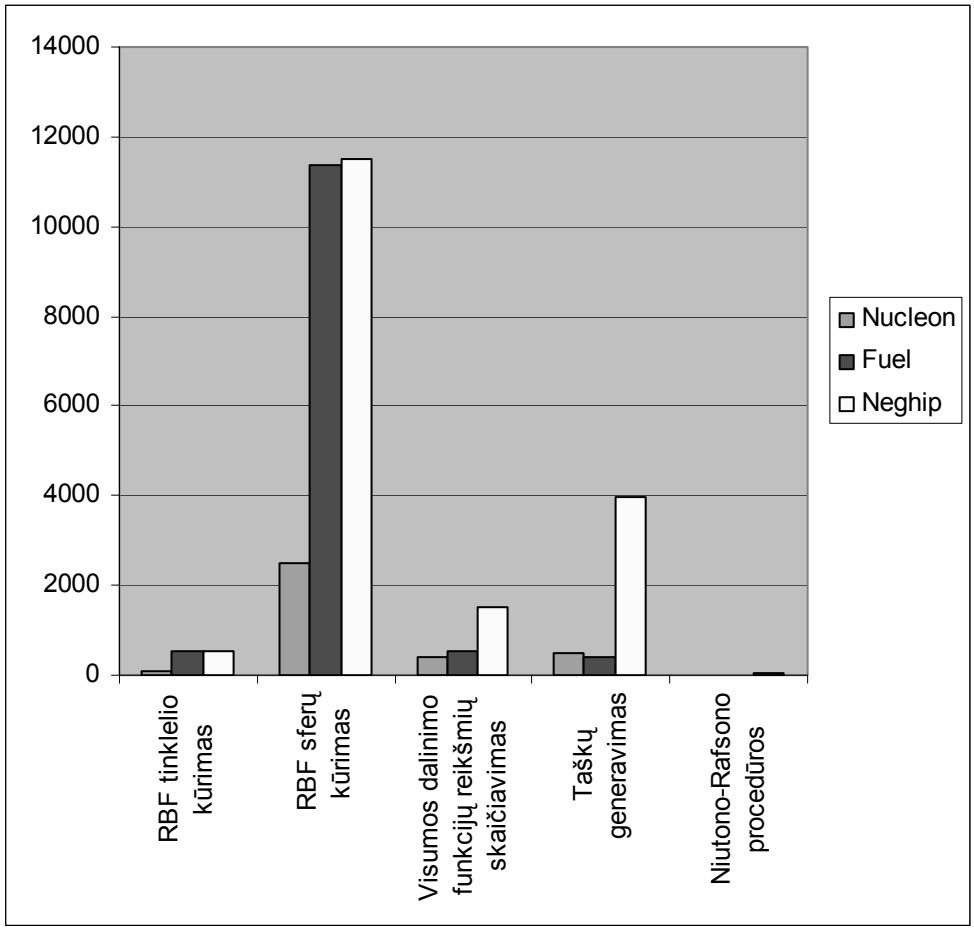


5 pav. Originalaus algoritmo dalių vykdymo laikinė diagrama

5 paveiksle pateikta algoritmo dalių vykdymo laikinė diagrama. Diagramoje, dėl pernelyg didelio vykdymo laiko, neįtrauktas koeficientų skaičiavimo etapas. Tarp likusių dalių daugiausiai trunka SBF sferų kūrimas. Šiame etape vieną kartą pilnai pereinamas visas SBF tinklelis. Kiekvienai tinklelio celei patikrinamos visos 26 gretimos celės, ieškant visų į sferos įtaką patenkančių duomenų elementų. Duomenų rinkinyje didėjant prasmingų celių skaičiui ženkliai šokteli visumos dalinimo funkcijų skaičiavimo bei taškų generavimo laikai, kadangi kiekviena prasminga celė į galutinį rezultatą gali įneši keletą naujų trikampių bei aibę papildomų skaičiavimų reikalaujančių taškų. Prasmingų celių kiekiai rinkiniuose pateikti 4 lentelėje.

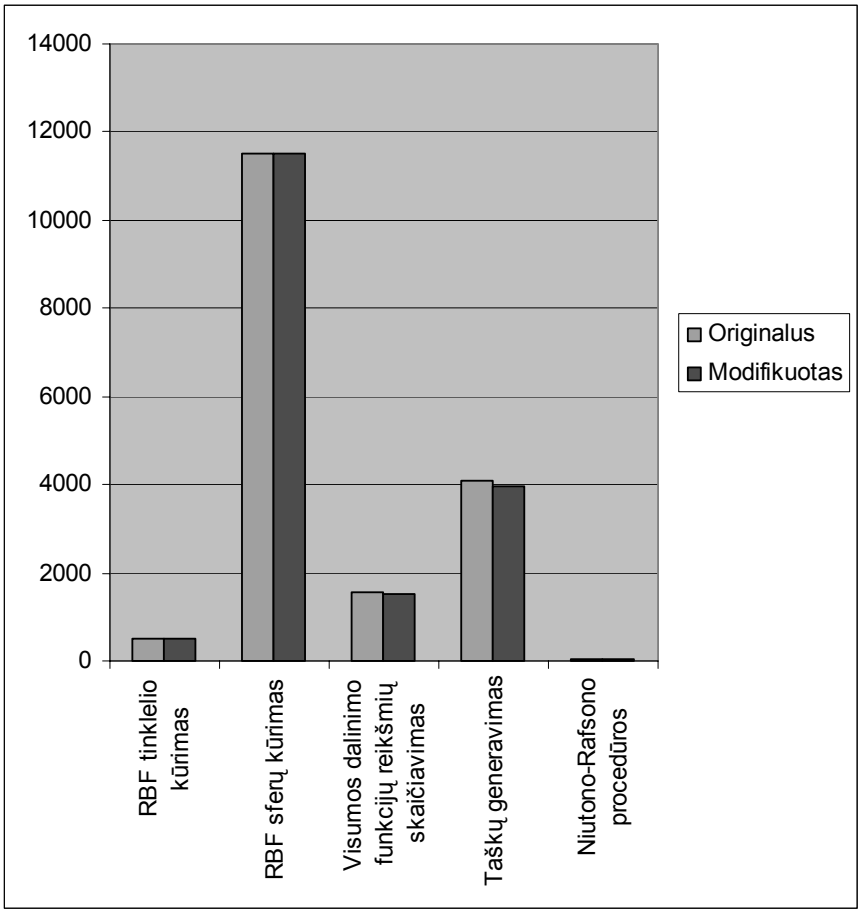
4 lentelė. Prasmingų celių duomenų rinkiniuose kiekiai

Duomenų rinkinys	Prasmingų celių sk./Bendras celių sk.
Nucleon	28272/64000 (ribinė reikšmė 20)
Fuel	16966/250047 (ribinė reikšmė 0)
Neghip	35579/250047 (ribinė reikšmė 50)
Hydrogen Atom	14120/2048383 (ribinė reikšmė 50)
Engine	1197439/8258175 (ribinė reikšmė 100)



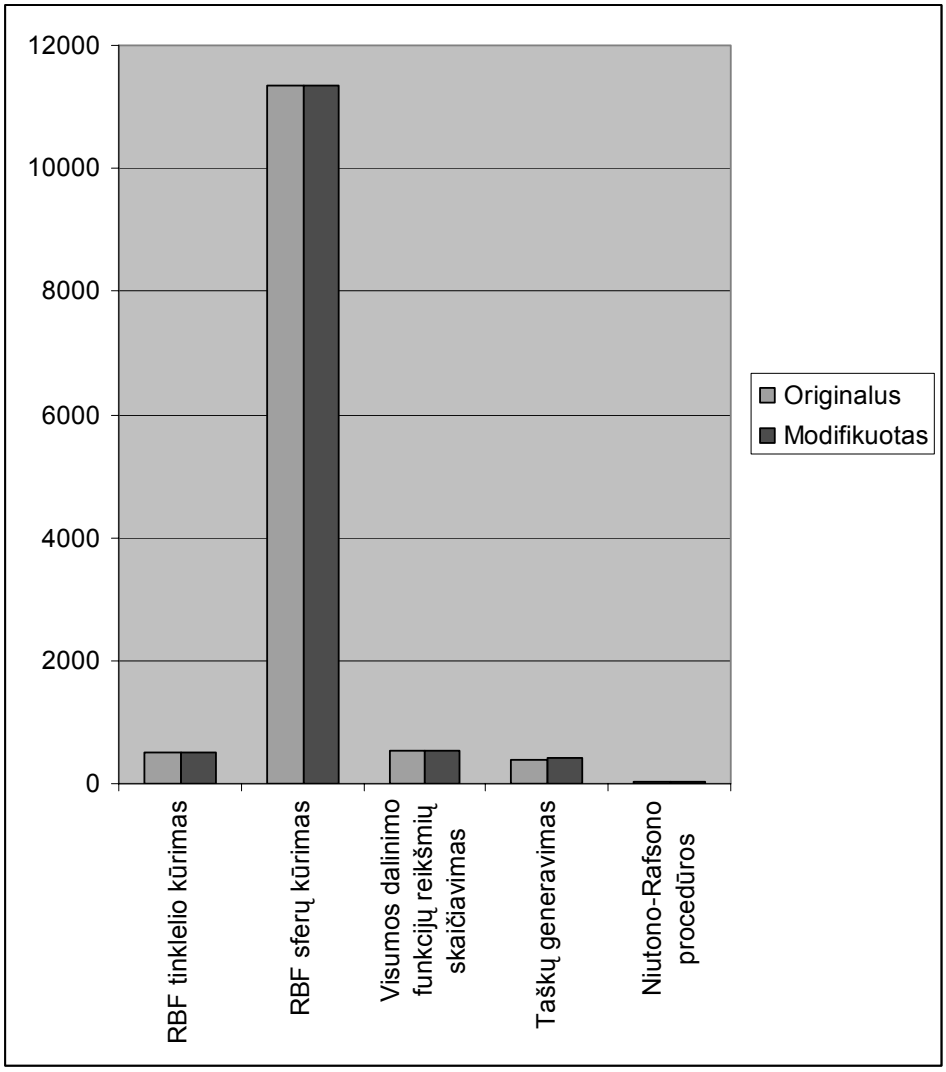
6 pav. Modifikuoto algoritmo vykdymo laikinės diagramos

„Fuel“ bei „Neghip“ duomenų rinkiniams SBF sferų kūrimo laikas skiriasi nedaug, kadangi šiame algoritmo vykdymo etape pereinamos pilnas duomenų rinkinys.



7 pav. Originalaus ir modifikuoto algoritmų vykdymo laikų palyginimas apdorojant „neghip“ duomenų rinkinį

Vykdant eksperimentus su modifikuotu algoritmu bei „Neghip“ duomenų rinkiniu vykdymo laiko sumažėjimas skaičiuojant visumos dalinimo funkcijų reikšmes bei atliekant taškų generavimą tampa pastebimas.



pav. Originalaus ir modifikuoto algoritmų vykdymo laikų palyginimas apdorojant „fuel“ duomenų rinkinį

Vykdant eksperimentus su „fuel“ duomenų rinkiniu pastebimas nedidelis algoritmo vykdymo laiko padidėjimas atliekant taškų generavimo žingsnį. Toks rezultatas gaunasi todėl, kad „fuel“ duomenų rinkinyje vaizduojamas objektas turi itin banguotą paviršių. Vykdant modifikuota algoritmą kiekvienam trikampiui yra atliekami paviršiaus sričių įvertinimai, tačiau dėl nelygumų atliekamas pilno užpildančio tinklelio generavimas.

Išvados

1. Pasiūlytas algoritmas, modifikuojantis betinklis dengiančiojo paviršiaus išskyrimo iš keletą blokų turinčių duomenų algoritmą;
2. Pasiūlytas algoritmas atlieka dinaminį dengiančiojo paviršiaus taškų aibės generavimą atsižvelgiant į paviršiaus tipą;
3. Nustatyta, jog vaizduojant mažus bei nelygius paviršius turinčius objektus modifikuota algoritmo versija skaičiavimų trukmę sąlygoja nežymiai;
4. Didėjant duomenų rinkiniui modifikuoto algoritmo rezultatai gerėja, paviršiaus taškų generavimo trukmė „Neghip“ duomenų rinkiniui paspartėjo 3,14%;
5. Nustatyta, jog tiek originaliame, tiek modifikuotame algoritme ilgiausiai užtrunka tiesinių lygčių sistemų, apskaičiuojančių interpoliacinių schemų koeficientus, sprendimai;
6. Nustatyta, jog efektyvus algoritmo optimizavimas negali būti atliekamas vienapusiškai, būtina atsižvelgti ne tik į skaičiavimų trukmę, bet ir į skaičiavimams atlikti reikalingos atminties kiekius;
7. Skaičiavimus paspartinti turinčios duomenų struktūros, saugančios SBF celėms bei sferoms priklausančius elementus, atliekant bandymus su „Neghip“ duomenų rinkiniu, pareikalavo ~20% daugiau atminties.

Literatūra

1. Kaufman A. E. Introduction to Volume Graphics [žiūrėta 2004-04-12]. Prieiga per Internetą: http://www.cs.sunysb.edu/~ari
2. Co C. S., Porumbescu S. D., Joy K. I. Meshless Isosurface Generation from Multiblock Data. 2004 [žiūrėta 2004-09-25]. Prieiga per Internetą: http://graphics.cs.ucdavis.edu/~schatzi/projects/MeshlessMultiblockIsosurfaces/
3. Kajiya J. T., Von Herren B. P. Ray Tracing Volume Densities. Iš Computer Graphics [interaktyvus]. 1984, Nr. 3 [žiūrėta 2003-01-29]. Prieiga per Internetą: http://www.cs.virginia.edu/~gfx/Courses/2003/ImageSynthesis/papers/Volume%20Rendering/Ray%20Tracing%20Volume%20Densities.pdf
4. Kaufman A. E. Volume Visualization: Principles and Advances. [žiūrėta 2004-04-14]. Prieiga per Internetą: http://www.merl.com/people/pfister/courses/Bonn2000/Papers/KaufmanVolumeVisualization.pdf
5. Bajaj Ch., Blinn J., Bloomenthal J., Cani-Gascuel M., Rockwood A., Wyvill B., Wyvill G. Implicite Surfaces. San Fransisco: Morgan-Kaufmann, 1997
6. Meißner M., Huang J., Bartz B., Mueller K., Crawfis R. A Practical Evaluation of Popular Volume Rendering Algorithms. [žiūrėta 2005-02-12]. Prieiga per Internetą: http://www.gris.uni-tuebingen.de/~/bartz/Publications/paper/volvis2000.pdf
7. Reuters P., Tobor I., Schlick Ch., Dedieu S. Point-based Modelling and Rendering using Radial Basis Functions. [žiūrėta 2004-12-15]. Prieiga per Internetą: http://www.labri.fr/Documents/Publications/Publi_Labri_2139.pdf
8. Co Ch. S., Hamann B., Joy K. I. Iso-splatting: A Point-based Alternative to Isosurface Visualization. Iš *Pacific Graphics 2003* [interaktyvus]. 2003, lapkritis [žiūrėta 2004-12-05]. Prieiga per Internetą: http://graphics.cs.ucdavis.edu/~schatzi/projects/IsoSplatting/coc_isosplatting.pdf
9. Elvins T.T. A Survey of Algorithms for Volume Visualization. [žiūrėta 2004-11-03]. Prieiga per Internetą: http://www.cs.duke.edu/courses/spring03/cps296.8/papers/elvins92surveyVolumeVisualization.pdf
10. Yagel R. Volume Viewing Algorithms: Survey. [žiūrėta 2004-09-18]. Prieiga per Internetą: http://www.cs.duke.edu/courses/spring03/cps296.8/papers/YagelViewingSurvey.pdf

11. Teixeira J. C. Jr. Parallel Volume Rendering Of Irregular Grids. 1996 [žiūrėta 2005-09-12]. Prieiga per Internetą: http://www.cs.utah.edu/~csilva/papers/phd96.pdf
12. Weinstein D.M, Johnson Ch. R. Hierarchical Data Structures for Interactive Volume Visualization. 1995 [žiūrėta 2004-03-12]. Prieiga per Internetą: http://www.cs.utah.edu/techreports/1995/pdf/UUCS-95-012.pdf
13. Ohtake Y., Belyaev A., Alexa M., Turk G., Seidel H. Multi-level Partition of Unity Implicite. [žiūrėta 2005-11-25]. Prieiga per Internetą: http://www.dgm.informatik.tu-darmstadt.de/pub/mpu.pdf
14. Levoy M. Display of Surfaces from Volume Data. 1987 [žiūrėta 2004-03-02]. Prieiga per Internetą: http://graphics.stanford.edu/papers/volume-cga88/volume.pdf
15. Kobbelt L., Botsch M. A Survey of Point-Based Techniques in Computer Graphics. [žiūrėta 2005-02-11]. Prieiga per Internetą: http://www-i8.informatik.rwth-aachen.de/publications/downloads/points.pdf

Terminų ir santrumpų žodynas

SBF – spindulinės bazinės funkcijos

Vokselis – mažiausias neskaidomas erdvės elementas