VILNIAUS SIMONO DAUKANTO GIMNAZIJA

Adomas Bieliūnas

IIIu2

Fraktalų atvaizdavimas trimatėje erdvėje

Projektinis darbas

Darbo vadovai

Mokytoja Irma Gecevičiūtė

Mokytojas Dainius Martūnas

Vilnius, 2022

TURINYS

1. ĮVADAS 3

2. ĮPRASTI FRAKTALAI 4

2.1 Kompleksiniai skaičiai 4

2.2 Mandelbroto ir Julijos aibės 5

3. FRAKTALAI TRIMATĖJE ERDVĖJE 7

3.1 Įprasti 3D kompiuterinės grafikos būdai 7

3.2 Spinduliųžygiavimas 8

3.2.1 Žingsniavimo algoritmas 8

3.2.2 Vizualus algoritmo pritaikymas 9

3.2.3 Realistiška apšvieta 10

3.2.4 Operacijos su skirtingais objektais 10

4. IŠVADOS 13

5. NAUDOTŲ ŠALTINIŲ IR LITERATŪROS SĄRAŠAS 14

6. ANOTACIJA 15

7. PRIEDAI 16

# ĮVADAS

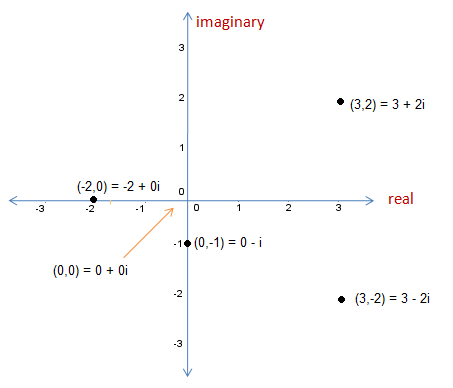
Šiame darbe aš, Adomas Bieliūnas, skaitmeniniu būdu atvaizduosiu specifinius matematinius objektus trimatėje erdvėje, vadinamus fraktalais. Norint sėkmingai pamatyti trimatį fraktalą kompiuterio ekrane man reikėjo pasitelkti daug metodų ir algoritmų iš matematinės bei informacinių technologijų pusės, taip man leidžiant giliau pažinti ir suprasti C++ vidinę ir išorinę kūrimo aplinką ir pritaikyti matematiką įvairiapusiškų problemų sprendimui.

# ĮPRASTI FRAKTALAI

## 2.1 Kompleksiniai skaičiai

Įprasti realieji skaičiai ℝ yra skaliarai – visi jie yra vienoje skaičių tiesėje nuo neigiamos iki teigiamos begalybės. Kompleksiniai skaičiai ℂ yra išreiškiami vektorine forma

kur *a* ir *b* yra realieji skaičiai, o *i* – menamasis vienetas, tenkinantis sąlygą

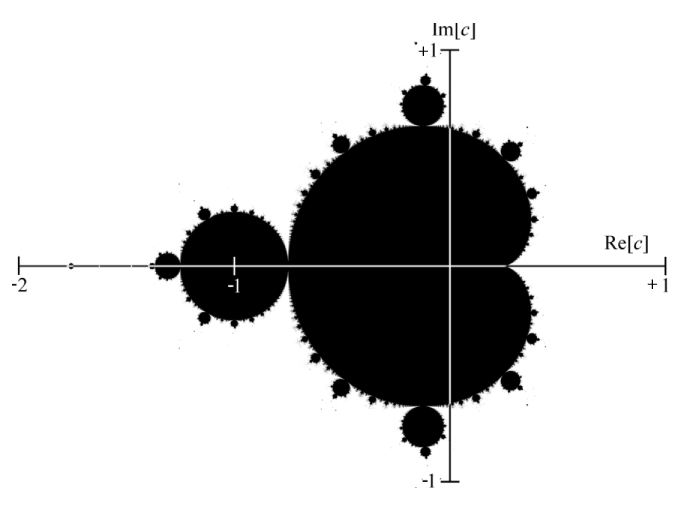
Menamoji ašis yra statmena realiajai ašiai ir ją kerta koordinačių pradžioje (0, 0), ir kartu jos sudaro kompleksinę plokštumą.

1 Vaizdas: Kompleksinė plokštuma su 4 taškais

Su kompleksiniais skaičiais galima atlikti tokius pat veiksmus kaip ir su realiaisiais:

## 2.2 Mandelbroto ir Julijos aibės

Vienas iš žymiausių fraktalų yra Mandelbroto aibė, esanti kompleksinėje plokštumoje, kuri turi baigtinį plotą, tačiau begalinį perimetrą. Kiekvienam plokštumos taškui *C* pritaikius iteratyvią formulę

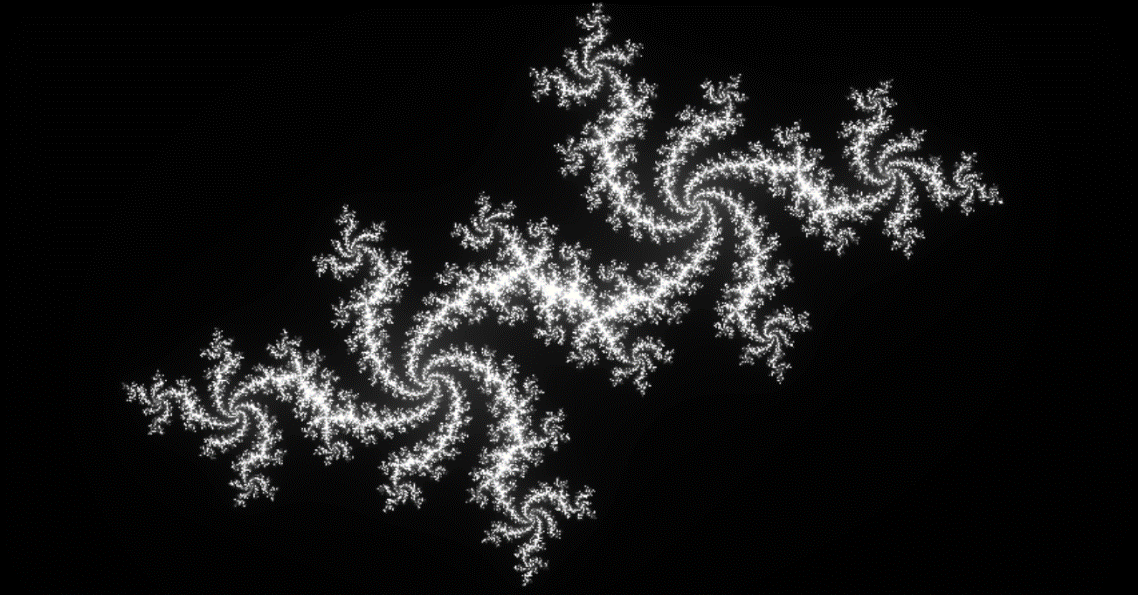
kur yra ateinantis skaičius ir yra gaunama Mandelbroto aibė. Jeigu po pasirinkto iteracijų kiekio taškas konverguoja, jis sekai priklauso. Jeigu su kiekviena iteracija taškas vis labiau tolsta nuo koordinačių pradžios, jis sekai nepriklauso.

2 Vaizdas: Mandelbroto aibė po tūkstančio iteracijų

Paveikslėlis, kuriame yra medis

Automatiškai sugeneruotas aprašymasJulios aibė yra panaši į Mandelbroto, tačiau vietoj vieno egzistuojančio rezultato galima gauti unikalią Julijos aibę su pasirinktu kompleksiniu skaičiumi kaip parametru. Šį kartą taškas įgauna vertę, o skaičius *C* yra pasirinktinas parametras.

3 Vaizdas: Julijos aibė kai c = (0, -0.622)



*4 Vaizdas: Julijos aibė kai c = (-0.54, -0.54)*

Šios dvimatės vizualizacijos leidžia pamatyti fraktalų chaotiškumą ir savęs pasikartojimą. Nors trimačių fraktalų atvaizdavimo metodai yra kitokie, juose yra matomos panašios chaotiškos savybės. Mažas pradinių parametrų pakeitimas gali duoti visiškai skirtingus rezultatus, bei artėjant prie fraktalo ribos pasirodo mažesnės, anksčiau matytos dalys.

# FRAKTALAI TRIMATĖJE ERDVĖJE

## 3.1 Įprasti 3D kompiuterinės grafikos būdai

Paprasčiausias būdas nupiešti trimačius objektus kompiuterio ekrane yra trikampių rastarizavimas (*angl. rasterisation)*. Šio proceso metu kompiuteris paverčia 3D modelio trikampių koordinates į matomus pikselius su atitinkama projekcija, spalva ir kitais efektais. Šis metodas naudoja mažai resursų, tačiau rezultatai dažnai yra nerealistiški ir žemos kokybės.

5 Vaizdas: Rasterizavimo pavyzdys

Moderniuose, realistiškuose modeliuose yra skiriama žymiai daugiau dėmesio apšvietai. Iš kameros kiekvienam pikseliui yra „išaunami“ spinduliai, kurie yra tikrinami ar susikerta su scenos objektu. Jeigu taip įvyksta, spindulys nuo objekto atsispindi žemesniu intensyvumu, prarasdamas šiek tiek energijos, ir vėl yra tikrinama ar jis su kažkuo kertasi. Šis procesas kartojasi tol, kol spindulys pasiekia šviesos šaltinį arba nustatytą žingsnių limitą. Šis algoritmas, vadinamas spindulių sekimu (*angl. ray tracing*), imituoja tikrą šviesą ir jo gaunami rezultatai atrodo kur kas labiau realūs. Tačiau kadangi reikia atlikti reikiamus skaičiavimus milijonams pikselių įvertinant atspindį, šviesos lūžius, tekstūrų įtaką šviesai ir kitus įvairius parametrus, vieno kadro atvaizdavimas gali užtrukti nuo kelių minučių iki kelių dienų.

## 3.2 Spinduliųžygiavimas

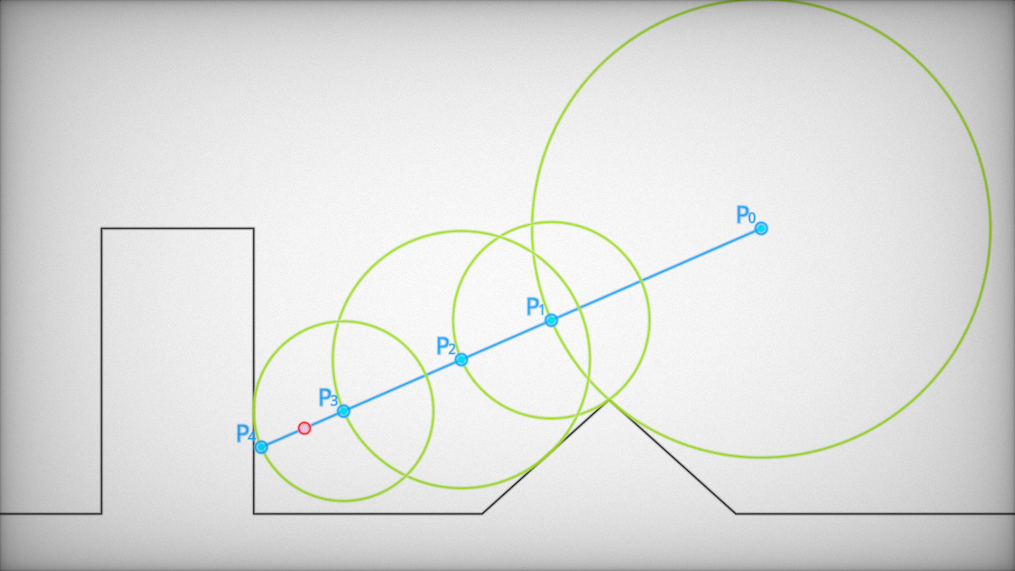
6 Vaizdas: 5-ojo vaizdo scena atvaizduota su spindulių sekimu

3 metodas trimačių objektų atvaizdavimui yra spindulių žygiavimas(*angl. raymarching*). Vietoj intensyvių ir ilgų skaičiavimų rasti spindulio ir daugybės trikampių susikirtimo vietą galima rekursyviai spindulį „žingsniuoti“ ir surasti sankirtos tašką kur kas greičiau.

Spindulių žingsniavimo algoritmuose kartais reikia perkelti reikšmę *v* iš vieno intervalo [*a, b*] į kitą [*c, d*], ką galima padaryti su šia funkcija:

### 3.2.1 Žingsniavimo algoritmas

Pirmiausia yra apskaičiuojamas atstumas nuo kameros iki objekto su pasirinkta funkcija DE. Tuomet taškas pajuda tuo atstumu į priekį, ir jo įgyta pozicija yra naujas pradinis taškas, nuo kurio vėl skaičiuojamas atstumas iki objekto ta pačia funkcija:

kur *p* yra taško pozicija, *n* yra iteracijos skaičius, *DE* yra atstumo iki objekto funkcija ir yra spindulio kryptis, priklausanti nuo pasirinkto matymo lauko (*angl. field of view FOV*) kampo. Šis algoritmas yra vykdomas rekursyviai, kol taško atstumas iki objekto yra lygus beveik 0, arba taškas nukeliauja per toli nieko nepaliečiant.

7 Vaizdas: Spindulio žingsniavimo schema

Spindulio kryptis priklauso nuo kameros pasisukimo ir matymo lauko kampo.

### 3.2.2 Vizualus algoritmo pritaikymas

Norint pamatyti norimos funkcijos sukurtą figūrą reikia 7 kameros parametrų – trimačių pozicijos ir pasisukimo vektorių ir matymo lauko kampo α. Pradžios taškas yra prilyginamas kameros pozicijai, o kryptis nustatoma pagal įprastas pasisukimo matricas:

Kur α yra matymo lauko kampas, o *p* – pikselio kordinatė nuo -1 iki 1. Jeigu kameros y pasisukimas nėra lygus nuliui, tuomet vaizdas atrodo pakreiptas, nepatogu orientuotis erdvėje, dėl to funkcija yra nenaudojama.

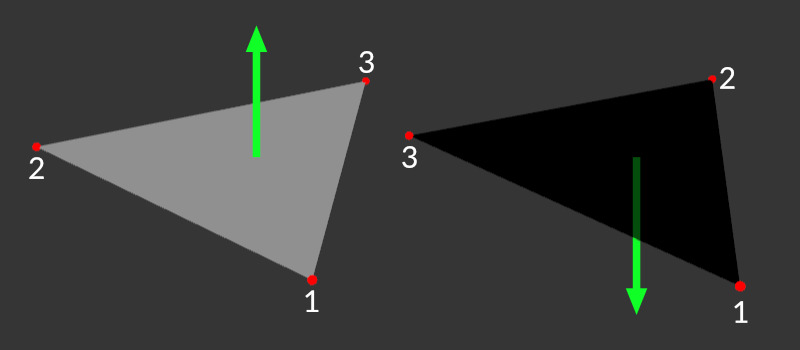
Paveikslėlis, kuriame yra siluetas

Automatiškai sugeneruotas aprašymasTaikant šį algoritmą ir nuspalvinant pikselius, iš kurių išėję spinduliai su objektu susikirto baltai, o kurie ne – juodai gaunamas toks rezultatas:

8 Vaizdas: Kubo atvaizdavimas su α = 80°

Nors ir galima matyti figūros siluetą, visos kitos savybės kaip spalva ar įgaubta geometrija yra prarandamos. Labiau detaliam atvaizdavimui naudojamos apšvietos funkcijos yra panašios kaip jau ir anksčiau minėtame spindulių sekimo metode, tik išgaunamos kitokiu būdu.

### 3.2.3 Realistiška apšvieta

Svarbus duomuo apšvietos skaičiavimuose yra normalė (*angl. surface normal*).

9 Vaizdas: Dvi trikampio normalės

Tai yra vektorius statmenas paviršiui. Tiesė eina iš abiejų pusių, todėl egzistuoja 2 priešingų krypčių normalės, todėl norint išvengti skaičiavimų su atvirkštine normale yra imamas tas vektorius, kuris yra labiau atsisukęs į kamerą. Įprastai žinant trikampio koordinates atitinkamą normalę apskaičiuoti yra gan paprasta, tačiau spindulių žygiavimo metodu trikampiai neatvaizduojami, ir tikslios normalės apskaičiuoti negalima. Reikia pasikliauti apytikslį baigtinių skirtumų metodą.

Jis veikia tokiu pačiu principu kaip ir elementarus funkcijos liestinės taške nustatymas – imama maža vertė, kuri yra pridedama prie pradinio taško. Su nauja verte funkcija yra vėl įvertinama ir pagal skirtumą tarp naujos ir senos funkcijos vertės apskaičiuojama liestinė. Kuo imama vertė yra arčiau 0, tuo gautas rezultatas yra tikslesnis. Taip pat ir yra ieškoma normalė – įvertinant DE funkciją su labai mažu pokyčiu ε ir palyginant skirtumą:

kur ε yra mažas skaičius arti 0, o *t* yra spindulio sankirtos su objektu koordinatė.

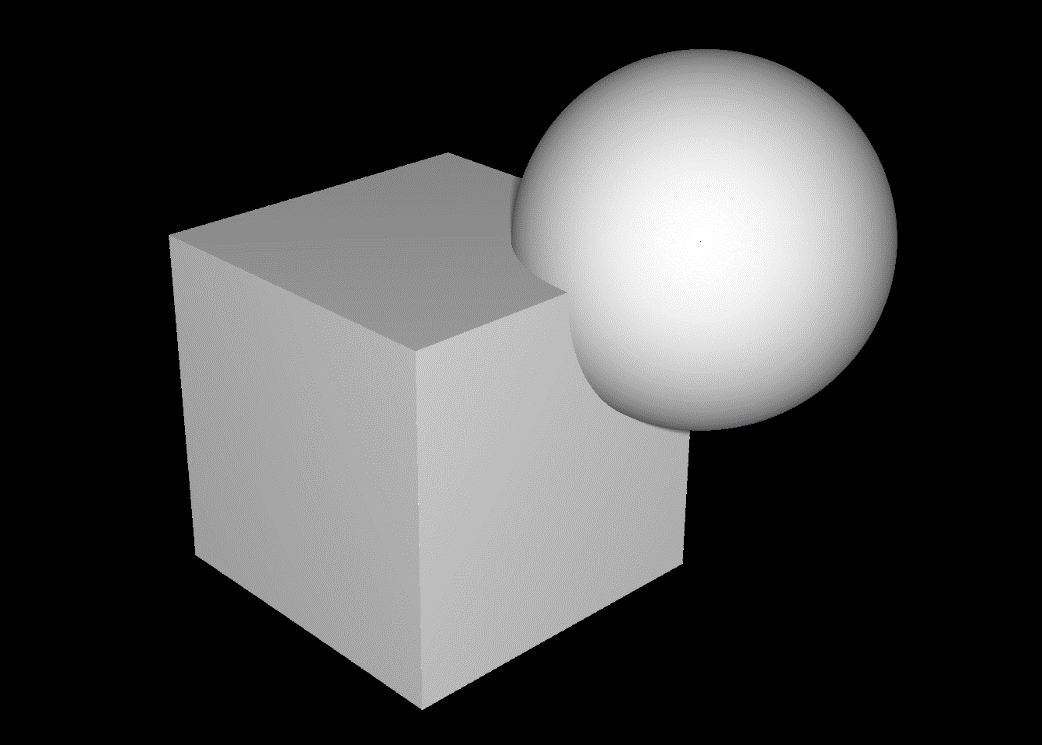
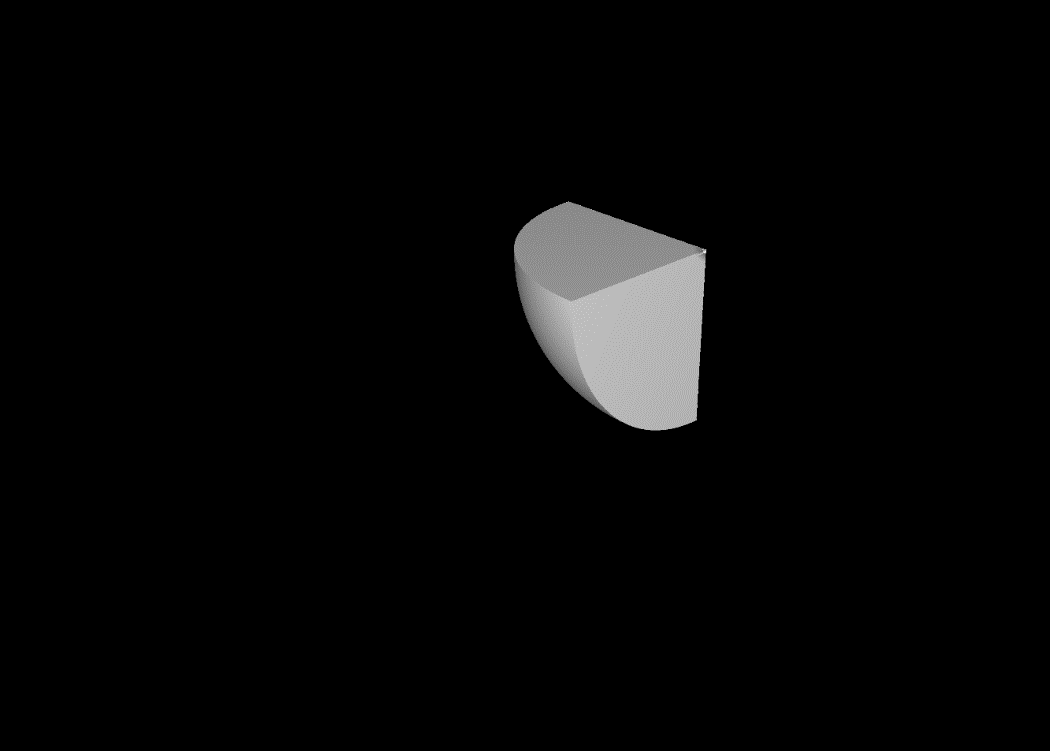
Turint normalę ir šviesų šaltinių pozicijas *s* galima apskaičiuoti taško apšvietos intensyvumą palyginant jo normalę su krintančiu šviesos šaltinio spinduliu bei padauginant gautą vertę iš šešėlių stiprumo, kuris gaunamas iš [formulės](https://iquilezles.org/articles/rmshadows/) , kur *k* nurodo aštrumo vertę. Kadangi neapšviestos vietos atrodytų visiškai tamsios, į apšvietos lygtį įeina minimalios apšvietos vertė *a* (*angl. ambient occlusion*) tarp 0 ir 1:

Jeigu i = 0, tuomet paviršius yra tiesiogiai neapšviestas ... [vaizdas]

### 3.2.4 Operacijos su skirtingais objektais

Tam, kad sužinoti atstumą iki objekto reikalinga funkcija DE(p). Paprastų objektų funkcijos egzistuoja ir nėra komplikuotos. Atstumų funkcijos gali turėti ir kitus parametrus be taško pozicijos, kaip dydį ar ilgį.

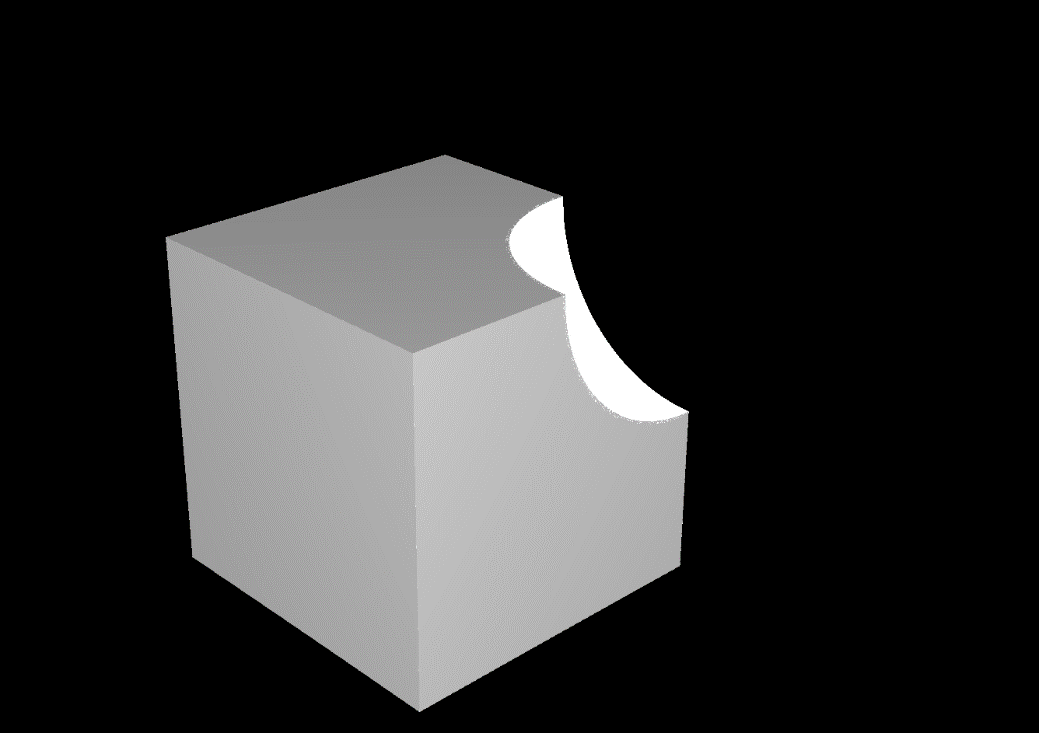
Esant keliems objektams su jais galima atlikti įvairius veiksmus. Keli objektai yra rodomi kartu kuomet naudojamas mažesnis iš 2 objektų duodamų funkcijos atstumų:

Taip pat galima matyti objektų sankirtos vietą su panašia formule:

11 Vaizdas: 2 objektų sankirta

10 Vaizdas: Dviejų objektų sąjunga

Arba objektų skirtumą:



12 Vaizdas: 2 objektų skirtumas (sfera yra atėminys)

Paveikslėlis, kuriame yra lubos, pastatas, stulpas, stogas

Automatiškai sugeneruotas aprašymasViena iš įdomesnių operacijų yra begalinis objekto pasikartojimas. Dėl funkcijoje esančios modulio operacijos erdvė, ir tuo pačiu objektas nuolat kartojasi 1, 2 ar visom 3 kryptim *c* atstumu. Pirma apskaičiuojamas naujas taškas su modulio funkcija ir tuomet jis yra įstatomas į *p* vietą objektą nusakančioje funkcijoje:

13 Vaizdas: Begalinio pasikartojimo pavyzdys

# IŠVADOS

# NAUDOTŲ ŠALTINIŲ IR LITERATŪROS SĄRAŠAS

<http://mathandmultimedia.com/wp-content/uploads/2012/02/complex-plane.png>

<https://upload.wikimedia.org/wikipedia/commons/thumb/5/56/Mandelset_hires.png/1280px-Mandelset_hires.png>

<https://www.mdpi.com/applsci/applsci-11-03264/article_deploy/html/images/applsci-11-03264-g001.png>

[https://www.racoon-artworks.de/cgbasics/images/normals/normals\_vertOrder.jpg](https://www.racoon-artworks.de/cgbasics/images/normals/normals_vertOrder.jpg%20)

<https://iquilezles.org/articles/distfunctions/>

<https://iquilezles.org/articles/rmshadows/>

# ANOTACIJA

# PRIEDAI