SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1337

Prikaz fraktalnih objekata u stvarnom vremenu

Tomislav Tunković

Zagreb, lipanj 2016.

Umjesto ove stranice umetnite izvornik Vašeg rada.
Da bi ste uklonili ovu stranicu obrišite naredbu `\izvornik`.

SADRŽAJ

Popis slika **v**

1. Uvod **1**

2. Fraktali **2**

2.1. Dvodimenzionalni fraktali 2

2.1.1. Trokut Sierpinskoga 2

2.1.2. Kochova pahuljica 3

2.1.3. Mandelbrotov skup 4

2.2. Trodimenzionalni fraktali 5

2.2.1. Mengerova spužva 5

2.2.2. Mandelbulb 6

2.2.3. Mandelbox 7

3. Prikaz 3D fraktala **8**

3.1. Koračanje duž zrake 8

3.2. Računanje udaljenosti 8

3.3. Bojanje i sjenčanje 9

4. Crtanje u stvarnom vremenu **11**

4.1. Postepeno koračanje i povećavanje kvalitete 11

4.2. Reprojiciranje udaljenosti 13

5. Zaključak **15**

Literatura **16**

POPIS SLIKA

1.1. Uvećanje dijela Mandelbrotovog skupa 1

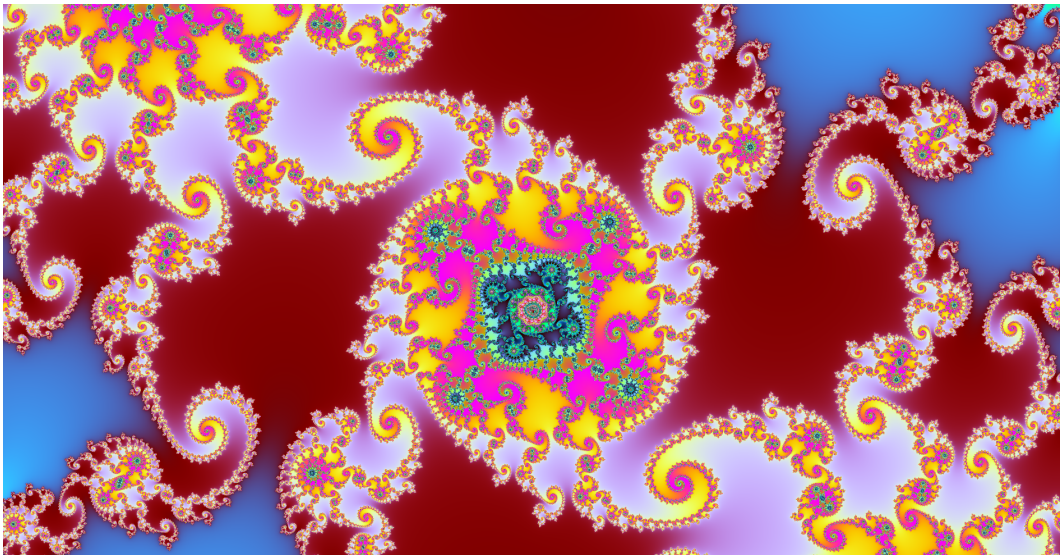
2.1. Prvih šest iteracija trokuta Sierpinskoga 2
2.2. Prvih šest iteracija Kochove pahuljice 3
2.3. Prvo preslikavanje 3
2.4. Drugo preslikavanje 3
2.5. Mandelbrotov skup 4
2.6. Mengerova spužva 5
2.7. Mandelbulb 6
2.8. Mandelbox 7

3.1. Dio obojanog i sjenčanog Mundelbulba 10
3.2. Dio obojanog i sjenčanog Mandelboxa 10

4.1. Početna slika nakon samo nekoliko koraka 12
4.2. Slika nakon više koraka 12
4.3. Konačna slika 12
4.4. Prije i poslije reprojekcije 13
4.5. 30 fps bez reprojekcije 14
4.6. 30 fps sa reprojekcijom 14

1. Uvod

Fraktali su geometrijski objekti s nekom razinom samosličnosti koja se proteže u beskonačnost kako mu se približavamo. Obično im je površina vrlo hrapava i što bliže dođemo tim naborima, otkrivamo da ih unutra ima još više koji su slični prijašnjima, ali opet na neki način drugačiji. Takvi fraktali postoje i u prirodi. Neki od primjera su munje, stabla i morske obale. Ovaj rad će obrađivati prikaz matematički zadanih fraktala. Neki od njih izgledaju doista impresivno i naizgled ne moguće za opisati, no njihova matematička definicija je često nevjerojatno jednostavna. No svejedno, iscrtavanje slike na računalu može biti vremenski vrlo zahtjevno. Ovaj rad će uz osnovni uvod u računalni prikaz fraktala predstaviti i neke obećavajuće metode za prikaz relativno visoke kvalitete u stvarnom vremenu.



Slika 1.1: Uvećanje dijela Mandelbrotovog skupa

2. Fraktali

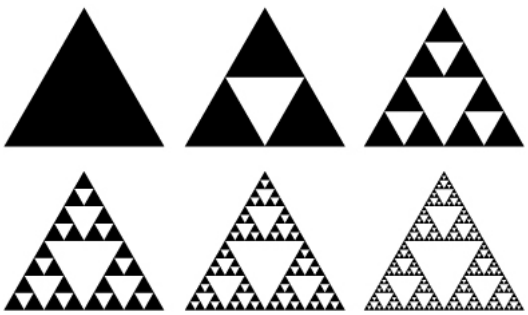
U ovom poglavlju ćemo spomenuti neke od poznatijih fraktala i reći ponešto o njima.

2.1. Dvodimenzionalni fraktali

Prije nego krenemo na trodimenzionalne fraktale, razmotrimo prvo neke osnovne ideje potrebne za definiranje dvodimenzionalnih. Neke od tih ideja će se kasnije relativno lako poopćiti na tri dimenzije, a za neke druge će biti potrebno malo kreativnosti.

2.1.1. Trokut Sierpinskoga

Jedan od najjednostavnijih je trokut Sierpinskoga. Jednostavan način za opisat konstrukciju tog fraktala bi bio da se prvo nacrta jednakostranični trokut, podijeli na 4 jednaka trokuta, srednji se obriše i postupak se ponovi u beskonačnost s preostala tri.

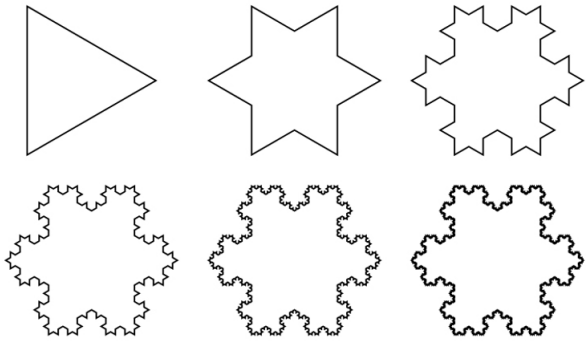


Slika 2.1: Prvih šest iteracija trokuta Sierpinskoga

Ako bismo ga htjeli nacrtati, trebamo se prvo odlučiti koliko iteracija želimo provesti, odnosno koliko ga u dubinu želimo rezati. Za svaku točku u ravnini možemo provjeriti pripada li jednom od tri trukuta iz prve iteracije. Ako ne pripada, onda je izvan fraktala. Ako pripada, onda ravninu transformiramo tako da trokut kojem pripada odgovara početnom trokutu i ponavljamo postupak zamišljeni broj iteracija.

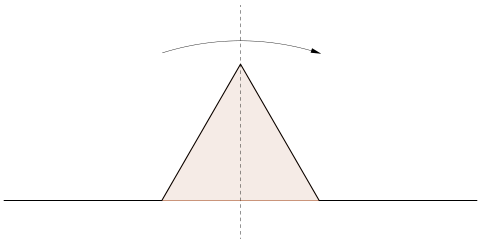
2.1.2. Kochova pahuljica

Jos jedan jednostavan fraktal je Kochova pahuljica koja se sastoji od triju Kochovih krivulja.

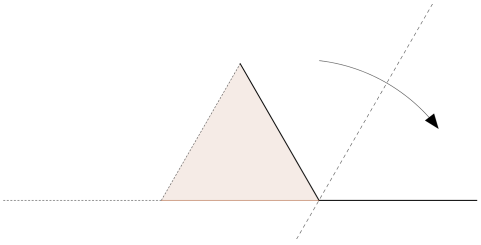


Slika 2.2: Prvih šest iteracija Kochove pahuljice

Postupak testiranja pripada li točka ovom fraktalu je malo složeniji, ali i općenitiji tako da bi se mogao primijeniti i na prethodni fraktal, kao i mnoge druge. Zasniva se na preslikavanju ravnine s jedne strane pravca na drugu. Prvo provjerimo pripada li točka početnom trokutu. Ako ne pripada, onda odredimo s koje strane se nalazi i napravimo transformaciju tako da ta strana legne na x os, sa središtem u ishodištu, a trokut bude s donje strane. Sada možemo početi ponavljati slijedeći iterativni postupak. Dužinu podijelimo na tri jednaka dijela i postavimo trokut na središnji dio. Ako točka pripada tom trokutu, onda je dio fraktala. Ako ne, onda trebamo napraviti preslikavanje ravnine preko dva pravca kao na slikama 2.3 i 2.4, tako da se rub fraktala preslika ponovo u jednu dužinu na kojoj ponavljamo postupak.



Slika 2.3: Prvo preslikavanje



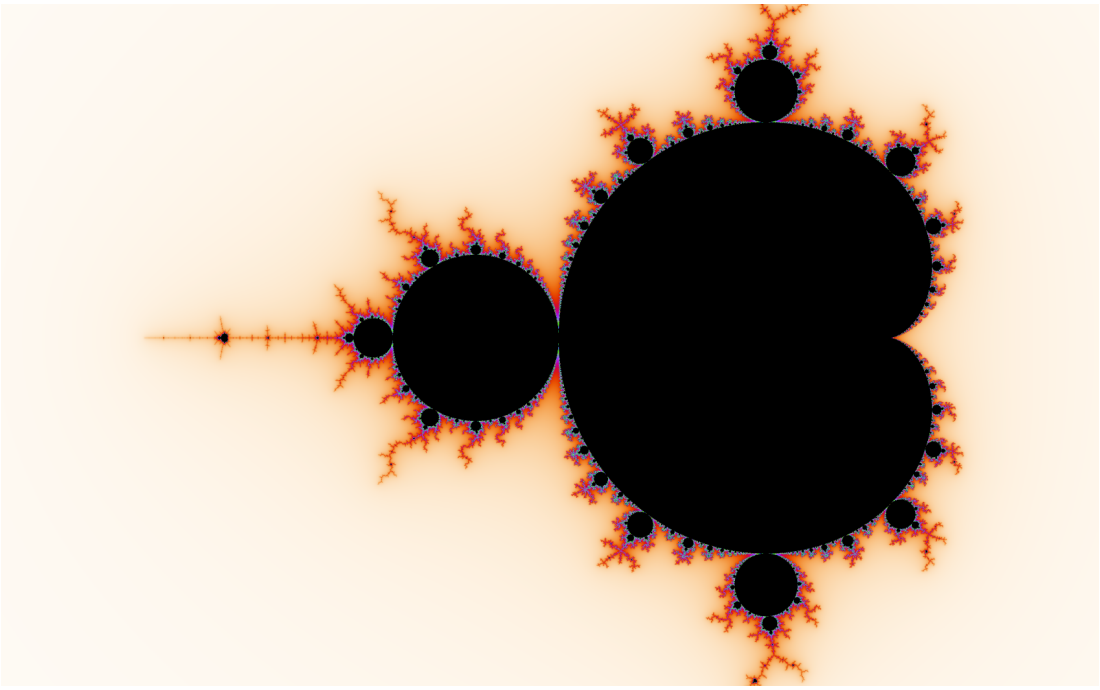
Slika 2.4: Drugo preslikavanje

2.1.3. Mandelbrotov skup

Mandelbrotov skup se definira bitno drugačije od prethodnih fraktala. Ravnina se promatra kao kompleksna ravnina i za svaki kompleksni broj c u njoj se iterira izraz. Ako je modul vrijednosti niza kojeg generira iteriranje izraza ograničen, onda točka pripada Mandelbrotovom skupu. Izraz koji se iterira je:

$$z_n = z_{n-1}^2 + c$$

$$z_0 = 0$$



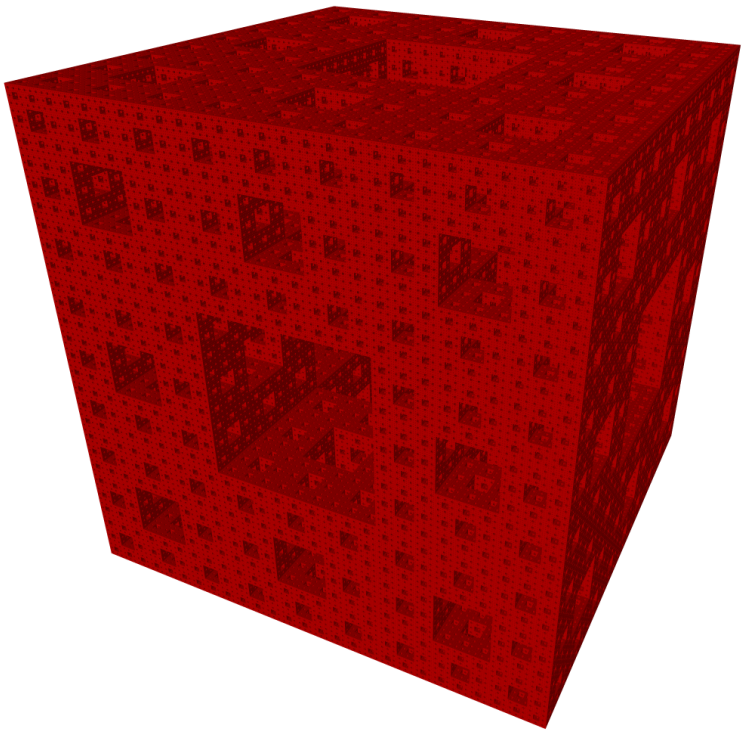
Slika 2.5: Mandelbrotov skup

Taj izraz možemo promatrati i kao udvostručavanje kuta u odnosu na x os, kvadriranje udaljenosti od ishodišta i translatiranje za c. Slični fraktali bi se dobili i za druge eksponente. Takva ideja će biti potrebna za definiranje trodimenzionalnih fraktala sličnih ovome. Za crtanje ovog fraktala, izraz naravno možemo iterirati samo konačan broj puta, a dovoljan uvjet za dokazati da generirani niz nije omeđen je $|z_n| > 2$ za bilo koji n.

2.2. Trodimenzionalni fraktali

2.2.1. Mengerova spužva

Donekle sličan fraktal trokutu Sierpinskoga. Ovdje umjesto trokuta imamo kocke. Svaka kocka se podjeli na $3 \times 3 \times 3$ manjih kocki i obrišu se one na sredinama strana i unutrašnjosti početne kocke.



Slika 2.6: Mengerova spužva

Slično kao kod Kochove pahuljice, za testiranje nalazi li se točka u fraktalu možemo koristiti niz zrcaljenja preko ravnine.

2.2.2. Mandelbulb

Mandelbulb je na neki način poopćenje Mandelbrotova skupa na treću dimenziju. Nažalost ne postoji lijepo proširenje kompleksnih brojeva s još jednom imaginarnom jedinicom, ali možemo koristiti istu ideju množenja kutova, odnosno sfernih koordinata, i potenciranje udaljenosti kako bismo dobili izraz za iteriranje. Jedan od zanimljivijih takvih izraza je slijedeći:

$$\mathbf{v}_n = (x, y, z) = S(r, \theta, \phi)$$

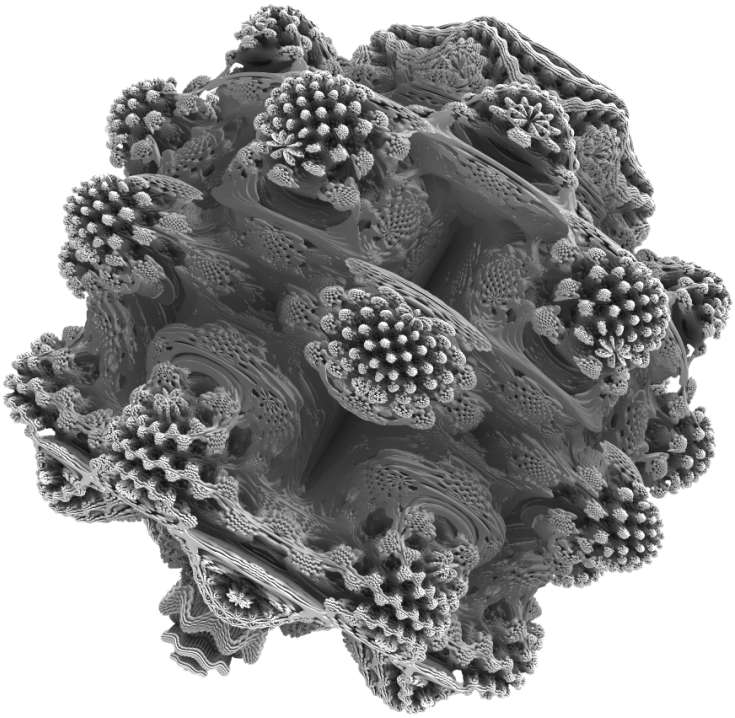
$$r = \sqrt{x^2 + y^2 + z^2}$$

$$\theta = acos(\frac{y}{r})$$

$$\phi = atan2(x, z)$$

$$\mathbf{v}_{n+1} = S(r^8, 8\theta, 8\phi) + \mathbf{c}$$

Gdje je $\mathbf{c}$ početna točka, a $z_0 = (0, 0, 0)$. Time dobivamo najpoznatiju varijantu Mandelbulb fraktala prikazanu na slici 2.7.



Slika 2.7: Mandelbulb

2.2.3. Mandelbox

Za generiranje Mandelboxa se također iterira izraz i testira je li omeđen. Transformacije koje se koriste su $boxFold$ i $sphereFold$.

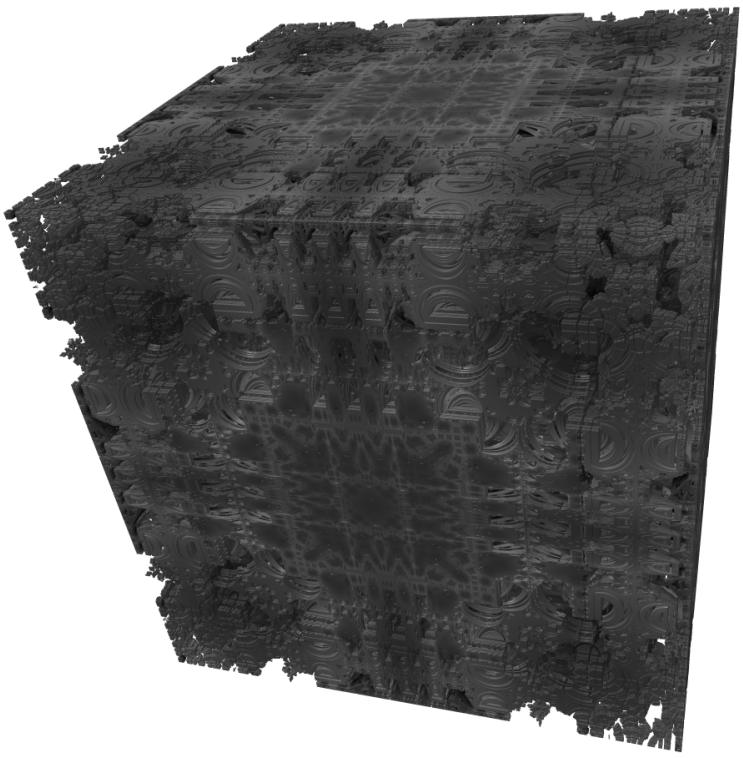
$$boxFold(\mathbf{v}) = clamp(\mathbf{v}, -L, L) * 2 - \mathbf{v}$$

$$sphereFold(\mathbf{v}) = \mathbf{v} \cdot \frac{R_f^2}{clamp(|\mathbf{v}|^2, R_m^2, R_f^2)}$$

Gdje funkcija $clamp(\mathbf{v}, a, b)$ ograničava komponente $\mathbf{v}$ na interval $[a, b]$. Konačni izraz koji se iterira glasi:

$$\mathbf{v}_{n+1} = M \cdot sphereFold(boxFold(\mathbf{v})) + \mathbf{c}$$

Paramteri L, R_f, R_m i M se mogu podešavati kako bi se dobile razne varijacije Mandelbox fraktala. Jedna od najčešćih kombinacija je $L = 1$, $R_f = 1$, $R_m = 0.5$ i $M = 2$, prikazan na slici 2.8.



Slika 2.8: Mandelbox

3. Prikaz 3D fraktala

Crtanje dvodemenzionalnih fraktala je prilično direktno ako imamo funkciju koja testira pripadnost točke. Jednostavno za svaki piksel unutar zadanog područja testiramo je li dio fraktala ili ne te ovisno o tome obojimo. U trodimenzionalnom slučaju to baš i ne možemo tako raditi. Moglo bi se cijeli prostor koji promatramo podijeliti na voksele i onda testirati centar svakoga pa ovisno o rezultatu testa ih različito obojiti. To bi bilo ili vrlo sporo, ili vrlo neprecizno, a možda i oboje. No, postoje i bolje metode.

3.1. Koračanje duž zrake

Glavna ideja je za svaki piksel slike imati po jednu zraku koja kreće iz pozicije kamere u smjeru tog piksela. Najjednostavnije bi bilo onda po svakoj zraci se kretati fiksnim korakom i stati jednom kad detektiramo da smo ušli u fraktal. Tada možemo još metodom bisekcije pokušati pronaći gdje se točno sjecište nalazi. Međutim, takav pristup je također daleko od idealnog. Fraktali su obično vrlo hrapavi, s puno rupa i nabora. To znači da takvim koračanjem lako možemo u potuponosti preskočiti neki manji dio fraktala i time dobiti pogrešnu sliku.

3.2. Računanje udaljenosti

Kako bi izbjegli greške koje mogu nastati prethodnom metodom i ubrzali računanje, bilo bi dobro kad bi postojala funkcija kojom možemo izračunati udaljenost do najbliže točke na površini fraktala. Iako možda zvuči nemoguće, za neke fraktale ih uopće nije teško dobiti, a za neke druge je moguće izvesti procjene koje nikad neće premašiti stvarnu udaljenost.
Promotrimo kako bismo to učinili za Mengerovu spužvu. U konstrukciji tog fraktala koristimo kocke kao osnovne elemente te transformacije koje uključuju translaciju, skaliranje i zrcaljenje poluprostora s jedne strane ravnine na drugu. Prilikom translacije, udaljenosti se ne mijenjaju. Kada skaliramo za s u svim smjerovima, udaljenost

u originalnom prostoru je s puta manja nego u skaliranom. Ako dio fraktala s jedne strane ravnine reflektiramo na isti takav s druge strane, dovoljno je promatrati samo udaljenost na toj jednoj strani jer možemo biti sigurni da će se najbliža točka nalaziti upravo na njoj. Koristeći te jednostavne zaključke, isti kod koji se koristi za testiranje pripadnosti točke Mengerovoj spužvi možemo modificirati da vraća udaljenost. Ako zaključimo da se nalazimo izvan fraktala, prestajemo iterirati i vraćamo udaljenost do najbliže točke na referentnoj kocki, koju dijelimo s ukupnim skaliranjem koje smo napravili. To je dovoljno jer sve ostale transformacije ne mijenjaju udaljenost.
Situacija je nešto kompliciranija kod fraktala gdje iteriramo izraz i testiramo omeđenost kao što su Mandelbrotov skup ili Mandelbox. Ispostavlja se da je i za njih moguće izvesti formule kojima dobivamo donju granicu za udaljenost. Međutim, to je izvan opsega ovog rada.

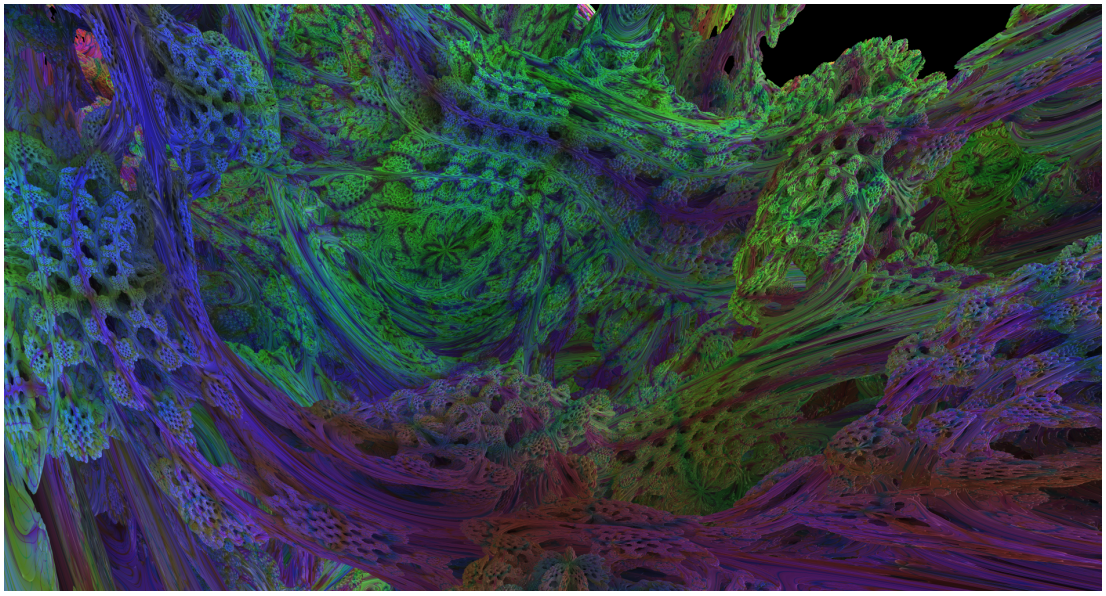
3.3. Bojanje i sjenčanje

Postavlja se pitanje kako obojiti piksel na slici jednom kada smo pronašli presjecište zrake s fraktalom. Jedan od čestih načina je vođenje evidencije koliko blizu smo došli nekim točkama ili ravninama tokom računanja udaljenosti od fraktala. Udaljenosti od triju glavnih ravnina koordinatnog sustava obično daju zanimljive uzorke, a udaljenost od ishodišta se pokazuje dobrim za sjenčanje površina koje su više zaklonjene. Normale se mogu aproksimirati tako da izračunamo kako se udaljenost mijenja u obližnjim točkama.

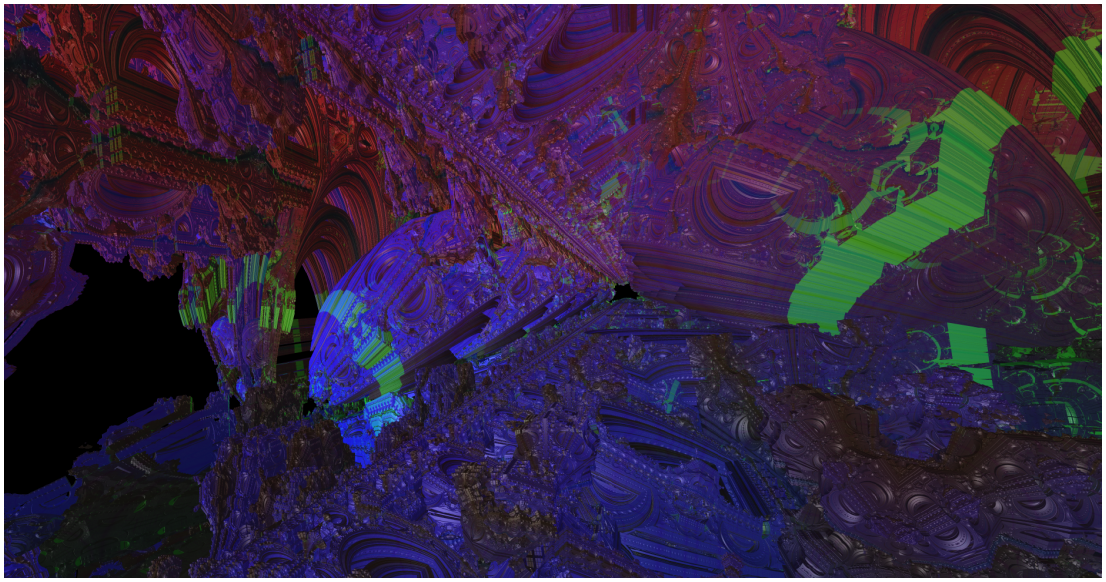
$$\begin{aligned}\mathbf{n}' = (&d(\mathbf{v} + (\epsilon, 0, 0)) - d(\mathbf{v} - (\epsilon, 0, 0)),\\ &d(\mathbf{v} + (0, \epsilon, 0)) - d(\mathbf{v} - (0, \epsilon, 0)),\\ &d(\mathbf{v} + (0, 0, \epsilon)) - d(\mathbf{v} - (0, 0, \epsilon)))\end{aligned}$$

$$\mathbf{n} = \frac{\mathbf{n}'}{|\mathbf{n}'|}$$

Gdje je d funkcija udaljenost od fraktala. Kad imamo normalu, možemo raditi Phongovo sjenčanje, računati refleksije i sjene ponavljanjem postupka koračanja u novom smjeru i mnoge druge stvari. Ambijentalna zaklonjenost se također može računati tako da se napravi nekoliko koraka u smjeru normale. Ako smo se uspjeli jako udaljiti onda znači da nema ničega u blizini pa je to mjesto dobro otkriveno, inače nije.
Slike 3.1 i 3.2 prikazuju primjere dobivene korištenjem nekih od navedenih metoda.



Slika 3.1: Dio obojanog i sjenčanog Mundelbulba



Slika 3.2: Dio obojanog i sjenčanog Mandelboxa

4. Crtanje u stvarnom vremenu

Prikaz fraktala može biti računski vrlo zahtjevan problem. Za svaki piksel na slici treba koračati malo po malo po zraci i u svakoj točki iterirati izraz za računanje udaljenosti od fraktala. Očito je da tu ima puno posla čije se vremenske složenosti množe. Računanje takvo nečega u stvarnom vremenu na CPU je nezamislivo, ali se vrlo lako može paralelizirati po pojedinom pikselu tako da je vrlo prilagodljivo za grafičku karticu. No čak i modernim grafičkim karticama treba vremena da izgeneriraju sliku koju bismo mogli smatrati dovoljno dobrom.

4.1. Postepeno koračanje i povećavanje kvalitete

Prva ideja koju bismo mogli primijeniti je da odvojimo kod za hodanje po zrakama od onoga za prikaz samog fraktala. Tada možemo rezultate koračanja spremiti zasebno, na temelju njih prikazati fraktal i za iduću sliku iskoristiti već prethodno izračunate udaljenosti i tako nastaviti dalje. To je vrlo jednostavna ideja i lako ju je implementirati.

Također možemo postepeno povećavati razinu antialiasinga. Pri tome treba paziti da kada prelazimo na višu razinu, više piksela na višoj rezuluciji će čitati isti piksel u nižoj, za zraku koja baš i ne ide u točno istom smjeru. Na rubovima fraktala se može dogoditi da time pročitamo vrijednost koja se nalazi već daleko iza prvog presjecišta s trenutno promatranom zrakom. Time zapravo nećemo popraviti nazubljene rubove. Taj problem možemo riješiti tako da se ne približavam u potpunosti fraktalu, nego stanemo dovoljno rano tako da možemo garantirati da povećavanjem rezolucije slike zrake susjednih piksela još uvijek pogađaju fraktal na približno odgovarajućem mjestu.

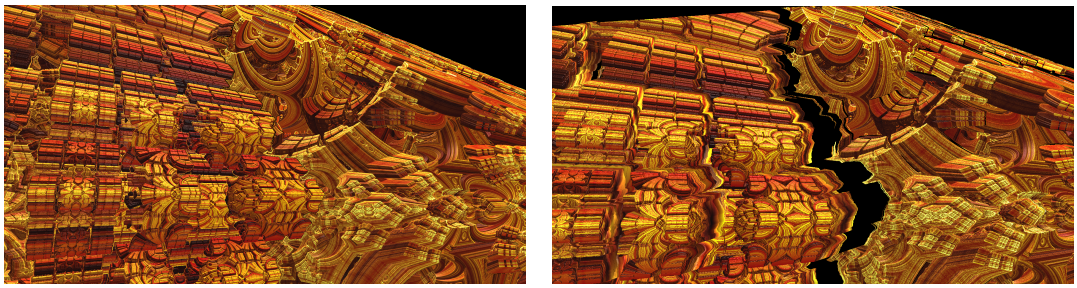
Slika 4.1: Početna slika nakon samo nekoliko koraka

Slika 4.2: Slika nakon više koraka

Slika 4.3: Konačna slika

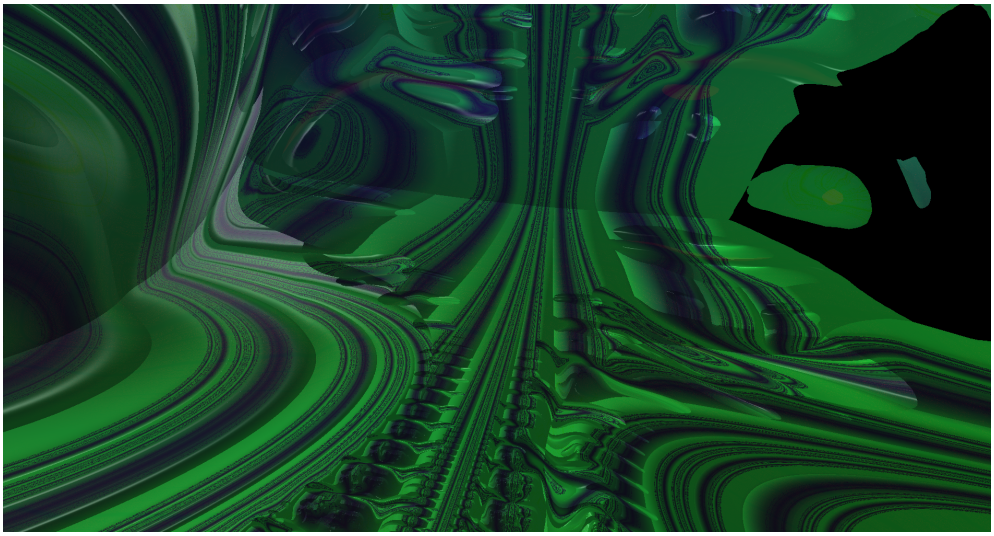
4.2. Reprojiciranje udaljenosti

Do sada, svaki put kad bi se pomaknula kamera, morali bi ponovo iz nule krenuti koračati svakom zrakom jer se zrake više ne poklapaju. Time gubimo sve prethodno izračunate rezultate. No nameće se da bi ih ipak trebalo nekako iskoristiti, barem u slučajevima kad se kamera nije puno pomaknula i slika je ostala skoro pa ista.
To možemo postići reprojiciranjem. Koristeći mrežu kvadrata, po jedan za svaki piksel. Vrhove kvadrata postavljamo na točke do kojih smo uspjeli doći na odgovarajućim zrakama. Tada koristeći novu projekcijsku matricu projiciramo tu mrežu i zapišemo nove udaljenosti. Ako se neki od tako projiciranih kvadrata preklope, koristimo onaj koji je bliži. Slika 4.4 ilustrira postupak reprojekcije. Crna traka na sredini se pojavila jer bi tu trebao biti dio fraktala koji nije bio vidljiv u izvornoj slici, a gore lijevo smo dobili crni trokut jer je to područje bilo izvan okvira. No ipak, vidimo da je dobar dio ostao očuvan i time smo uštedili puno računanja za iduće koračanje.

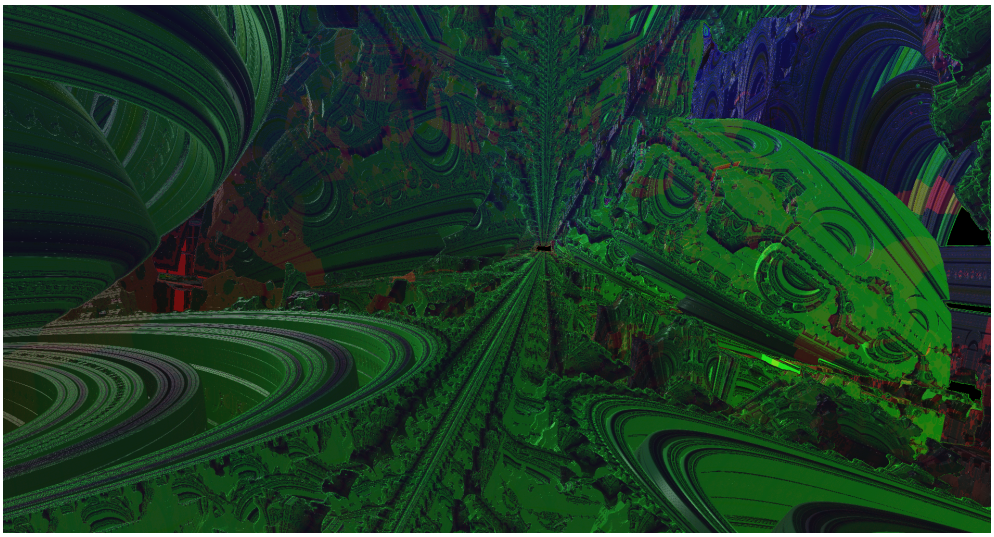


Slika 4.4: Prije i poslije reprojekcije

Problemi nastaju na rubovima fraktala gdje postoji velika razlika između udaljenosti do kojih se došlo na zrakama susjednih piksela. Ako se pri rasterizaciji konstantno događa da dobivamo udaljenost udaljenijeg piksela, a trebao nam je bliži, djelovi fraktala će početi nestajati izvana prema unutrašnjosti. To se može riješiti tako da prilikom reprojekcije, za svaki od vrhova mreže, umjesto da pročitamo udaljenost samo jednog piksela, uzmemo ih više iz okoline i koristimo minimalnu udaljenost. Drugi pristup je da nakon reprojekcije, udaljenost za svaku zraku zamijenimo s minimalnom udaljenošću od susjednih zraka.
Reprojekcija se pokazala iznenadujuće uspješnom. Broj koraka po zraci po slici je bio namješten na 30, s ciljem da mogu iscrtati oko 30 slika po sekundi, rezolucije 1920×1080. To sam radio na laptopu sa GTX 980M grafičkom karticom. Na temelju usporedbe slika 4.5 i 4.6 se može vidjeti poboljšanje.



Slika 4.5: 30 fps bez reprojekcije



Slika 4.6: 30 fps sa reprojekcijom

5. Zaključak

Crtanje trodimenzionalnih fraktala je vrlo zahvalan posao jer često rezultira impresivnim slikama. Postoje online forumi koji se bave time, gdje ljudi razmjenjuju fraktale koje su pronašli i pišu programe za crtanje. Isprobao sam ih nekoliko i izgleda da su svi primarno napravljeni kako bi crtali slike jako visoke kvalitete sa naprednim opcijama za sjenčanje, što zahtjeva nekoliko sekundi da se izračuna. Za pozicioniranje kamere se obično koristi prikaz vrlo niske rezolucije koji je i dalje bio spor. Jedan manji dio je imao brzi i relativno kvalitetan prikaz implementiran na GPU, ali izgleda da je korišten osnovni algoritam koji je još uvijek dosta spor. Nisam pronašao niti jedan koji koristi reprojekciju ili slične optimizacije za prikaz u stvarnom vremenu. Rezultati postignuti u ovom radu su obećavajući, ali uvjeren sam da se može i bolje. Uz još malo napredaka u algoritmima i grafičkim procesorima, smatram da se može otvoriti mogućnost za prikazivanje ovakvih fraktala u video igrama što bi se moglo pokazati vrlo atraktivnim.

LITERATURA

URL http://www.fractalforums.com/index.php.

Mikael Hvidtfeldt Christensen. Syntopia. URL http://blog.hvidtfeldts.net/index.php/2011/06/distance-estimated-3d-fractals-part-i/.

Inigo Quilez. URL http://www.iquilezles.org/www/index.htm.

Prikaz fraktalnih objekata u stvarnom vremenu

Sažetak

Ovaj rad sadrži kratak uvod u fraktalne objekte kroz nekoliko poznatih primjera. Objašnjava osnovne metode računalnog prikaza. Iznosi neke ideje kako ubrzati proces računanja i omogućiti prikaz visoke kvalitete u stvarnom vremenu. Uvodi se algoritam reprojiciranja izračunatih udaljenosti koračanjem zraka kako bi se već postojeci rezultati u dobroj mjeri mogli iskoristiti za iduće slike.

Ključne riječi: fraktali, koračanje zrakom, stvarno vrijeme, reprojekcija

Real time rendering of fractal objects

Abstract

This paper contains a short introduction to fractal objects through several well known examples. It explains basic methods of rendering. Some ideas are proposed on how to accelerate the rendering of high quality images in real time. The reprojection algorithm is employed to reuse already calculated raymarched distances for subsequent frames.

Keywords: fractals, raymarching, real time, reprojection