Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.92
DOI: https://doi.org/10.24867/31SA01Tilinger

UPOTREBA VORONOI TESALACIJE ZA DEFINISANJE FUNKCIONALNIH ELEMENATA MODELA

USE OF VORONOI TESSALATION FOR DEFINING THE FUNCTIONAL ELEMENTS OF THE MODEL

Antonia Tilinger, Fakultet tehničkih nauka, Novi Sad

Oblast – RAČUNARSKA GRAFIKA

Kratak sadržaj – *U ovom radu ukratko je opisan pojam Voronoi tesalacije i način na koji se on može implementirati za definisanje funckionalnih elemenata modela u trodimenzionalnom prostoru. Implementacija je sprovedena korišćenjem skripte koja je napisana korišćenjem Python programskog jezika i njegovih biblioteka poput NumPy i SciPy. Vizuelizacija je sprovedena u softveru Blender u verziji 4.2 koja je kompatibilna sa Python verzijom 3.11.*

Ključne reči: *kompjuterska grafika, Python programski jezik,Voronoi, Blender*

Abstract – *This paper briefly describes the concept of Voronoi tessellation and how it can be implemented to define the functional elements of the model in three-dimensional space. The implementation was carried out using a script written using the Python programming language and its libraries such as NumPy and SciPy. Visualization was performed in Blender software version 4.2 which is compatible with Python version 3.11.*

Keywords: *computer graphics, Python programming language, Voronoi, Blender*

1. UVOD

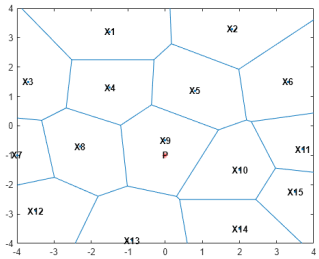
Cilj ovog rada je primena znanja koje je student stekao u toku studiranja kao i njegovo proširenje. U toku studija, student se susreo sa Voronoi tesalacijom u dvodimenzionalnom prostoru, nakon čega je student odlučio da proba da unapredi znanje koje je stekao i implementira Voronoi u trećoj dimenziju. Cilj je bio pisanje skripte kao dodatak koji automatizuje proces deljenja 3D objekta na segmente, koji se nakon obrade mogu koristiti u svrhu 3D štampe.

NAPOMENA:
Ovaj rad proistekao je iz master rada pod istim nazivom, čiji mentor je bila prof. dr Lidija Krstanović.

2. TEORIJSKA OSNOVA

2.1. Voronoi

Voronoi dijagram je vrsta tesalacionog šablona u kome se n broj tačaka raspoređenih po ravni deli na tačan broj konveksnih oblasti koje se nazivaju ćelije. Svaka od tih ćelija obuhvata deo ravni koji je najbliži istom sedištu. Granice između ovih oblasti čine tačke koje imaju više od jednog sedišta. U najjednostavnijem, dvodimenzionalnom slučaju, za konačni skup tačaka *{p1, ... , pn}* u euklidskoj ravni, svaka tačka predstavlja generator Pk, a njoj odgovarajuća Voronoi ćelija Rk sadrži sve tačke čija je udaljenost do Pk manja ili jednaka udaljenosti do bilo kog drugog generatora. Ćelije nastaju kao polupresek prostora što ih čini konveksnim mnogouglovima. Voronoi ćelije ne moraju uvek biti povezane, samim tim dobijaju se uvek konveksni mnogouglovi [1].



Slika 1. *Primer Voronoi dijagrama [11]*

2.2. Rastojanja

U većini slučajeva, za računanje rastojanja između tačaka može se koristiti Euklidsko ili *Manhattan* rastojanje. Euklidsko rastojanje je uobičajena razdaljina između dve tačke koju je moguće izmeriti lenjirom, odnosno ono predstavlja dužinu prave koja spaja te dve tačke. Računa se Pitagorinom teoremom koristeći kartezijanske koordinate tačaka. Za opšti n-dimenzionalni slučaj, Euklidsko rastojanje između dve tačke p i q se računa na sledeći način [2]:

$$d(p_i, p_j) = \sqrt{(p_{i\,x} - p_{j\,x})^2 + (p_{i\,y} - p_{j\,y})^2 + \cdots + (p_{i\,n} - p_{j\,n})^2}$$

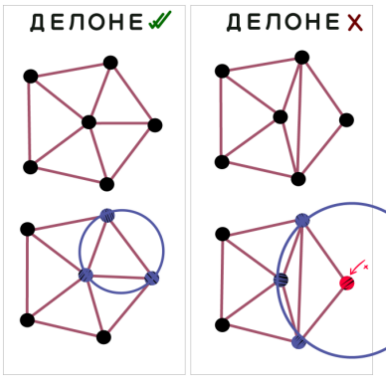
Manhattan rastojanje je dobilo ime po obliku mreže ulica na ostrvu Manhattan. Rastojanje između dve tačke definisano je kao zbir apsolutnih razlika njihovih odgovarajućih Dekartovih koordinata. Manhattan udaljenost između dve tačke p i q, u opštem n-dimenzionalnom slučaju se računa na sledeći način [3]:

$$d_T(p,q) = ||p-q||_T = \sum_{i=1}^{n} |p_i - q_i|$$

2.3. Delaunay triangulacija

Triangulacija je geometrijski proces razlaganja kompleksnih oblika, ili prostora, na jednostavnije geometrijske elemente.

Delone triangulacija dobila je ime po Borisu Deloneu koji je radio istraživanje o njoj 1934. godine. Kao što samo ime kaže, generiše se skup trouglova koji povezuju tačke. Generisani trouglovi su takvi da ukoliko se nacrta kružnica preko temena trouglova, unutar te kružnice se ne nalazi nijedna druga tačka. Delone triangulacija ima svojstvo maksimiziranja najmanjeg ugla, odnosno prilikom generisanja trouglova izbegava se generisanje trougla sa oštrim uglovima. Za izračunavanje Delone triangulacije postoji nekoliko algoritama. Većina njih oslanja se na operacije koje proveravaju da li se tačka nalazi unutar opisane kružnice trougla [4]. U matematičkom smislu, Deloneova triangulacija je dualni graf Voronoi dijagrama, odnosno za isti skup tačaka, vrhovi Voronoi dijagrama odgovaraju centrima opisanih kružnica Delone trouglova.



Slika 2. *Pravilna (levo) i nepravilna (desno) Delone triangulacija*

2.4.Python

Python je svestran programski jezik i spada u jezike višeg nivoa. Poprilično je jednostavan i čitljiv, zbog čega je i stekao svoju popularnost. Ima širok opseg biblioteka i podržava objektno-orijentisano i funkcionalno programiranje. Koristi jednostavnu sintaksu na engleskom, što ga čini lakim za čitanje i održavanje. Programi koji su pisani korišćenjem Python programskog jezika se interpretiraju, odnosno izvršavaju se red po red što omogućava brzo i efikasno testiranje i otklanjanje grešaka [5].

Python je kreiran od strane Holandskog programera *Guido van Rossum* – a, kao naslednik ABC programskog jezika na kojem je Rosum radio pre *Python*- a [6].

2.5. Korišćene biblioteke

Za izradu ovog projekta korišćeno je nekoliko Python biblioteka:

NumPy – popularna biblioteka koja se koristi za numeričko računanje i rukovanje sa višedimenzionalnim nizovima [7].

SciPy – biblioteka otvorenog koda, nadovezuje se na *NumPy* ali ima proširenu funkcionalnost [8].

Shapely – biblioteka koja se koristi za geometrijske operacije i prostornu analizu [9].

Mathutils – modul u Blenderu koji se koristi za rukovanje sa matematičkim operacijama u kontekstu 3D grafike.

BPy – Python API (*Application Programming Interface*) za Blender koji služi za razvijanje alata u Blenderu.

Bmesh – sistem za manipulisanje složenom poligonalnom geometrijom.

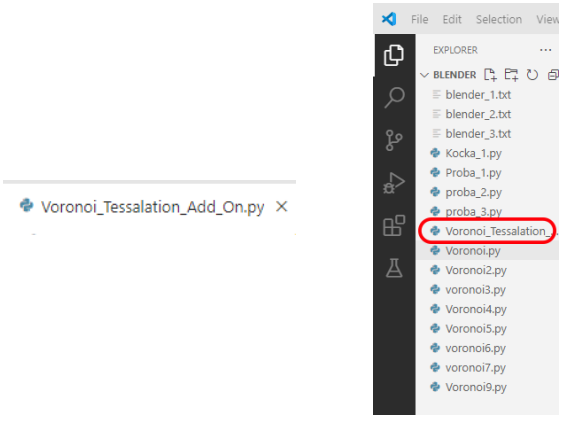
Random – biblioteka koja sadrži funkcije za generisanje nasumičnih brojeva.

2.5. Blender

Blender je besplatni softver otvorenog koda. Podržava ceo 3D proces – modelovanje, animiranje, simulaciju, renderovanje, praćenje pokreta i drugo. Koristi se za kreiranje animiranih filmova, vizuelnih efekata, umetnosti, 3D štampanih modela, a ranije se koristio i za kreiranje video igara. Blender je višeplatformski, što znači da radi podjenako dobro na *Windows*, *Linux* i *Machintosh* računarima (operativnom sistemu). Korisnički interfejs koristi *OpenGL* biblioteku [10].

3. PRAKTIČAN DEO

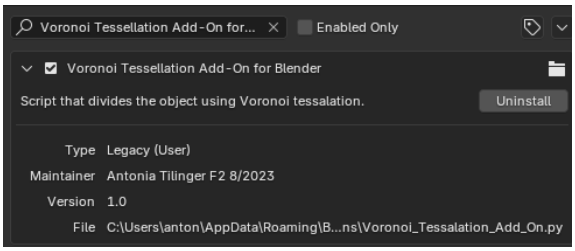
Praktičan deo projekta je podeljen u 3 glavne celine: generisanje nasumičnih tačaka koje predstavljaju generatore Voronoi ćelija, generisanje Voronoi dijagrama i podela objekta korišćenjem *Boolean* operacija. Skripta je pisana u programu *Visual Studio Code*.



Slika 3. *Visual studio projekat*

3.1. Generisanje tačaka

Prije nego što su se generisale tačke bilo je neophodno napisati rečnik za Blender skriptu podataka koji pruža osnovne metapodatke koje Blender koristi za identifikaciju i prikaz informacija o dodatku koji ise kreira. Rečnik sadrži naziv dodatka, minimalnu potrebnu verziju Blendera, kategorija gde se dodatak pojavljuje, opis dodatka, ime autora i verzija dodatka.

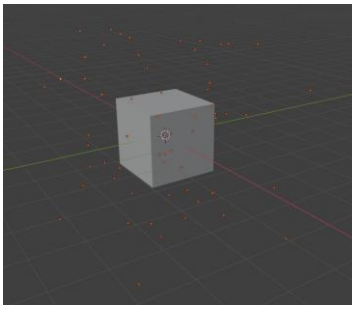


Slika 4. *Rečnik*

Da bi se tačke generisale prosleđuje se informacija o trenutno selektovanom / aktivnom objektu. Najpre se proverava da li je selektovan neki od objekata u sceni. Ukoliko jeste i proverava se tip objekta, i ukoliko je tipa *mesh* prosleđuje se ime izabranog objekta. Ukoliko nije, program nas obaveštava da nije izabran nijedan objekat i dodatak se zaustavlja.

Za generisanje tačaka uzimaju neophodni su podaci o poziciji objekta i njegovom graničnom okviru. Lokacija se uzima svetska a ne lokalna, jer nam treba lokacija objekta u odnosu na scenu. Granični okvir je definisan minimalnim i maksimalnim vrednostima temena objekta duž svake ose (x, y, z).

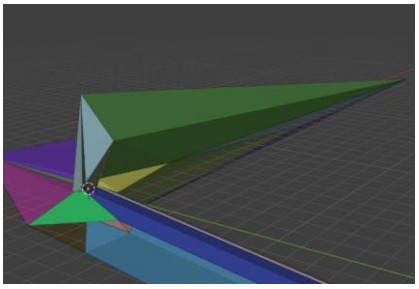
Pored lokacije i graničnog okvira objekta, prosleđuju se parametri za broj tačaka za generisanje i minimalna distanca između tačaka. Minimalna distanca osigurava da su tačke raspoređene. U zavisnosti od broja tačaka koji se generiše, prolazi se kroz petlju dok se sve tačke ne generišu. Tačke se mapiraju na veličinu graničnog okvira objekta po svakoj osi kako bi se nalazile unutar granica objekta. Na kraju se na novogenerisane tačke dodaje nasumični šum kako bi se sprečilo da se tačke nagomilavaju u jednom regionu, i sprečava pojavu degenerisanih regiona u Voronoi dijagramu. Za vizuelizaciju ovih tačaka korišćene su sfere. Na lokacijama tačaka kreirane su UV sfere poluprečnika 0.02, koje se ubacuju u posebnu kolekciju u Blenderu.



Slika 5. *Nasumične tačke i njihova vizuelizacija*

3.2. Generisanje Voronoi dijagrama

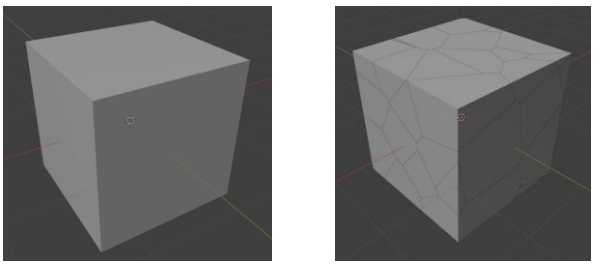
Za generisanje Voronoi dijagrama za nasumične tačke koristi se klasa „*Voronoi*“ iz *SciPy* biblioteke. Objekat „*vor*“ će da sadrži informacije o Voronoi regionima, odnosno ćelijama, koje uključuju poziciju vrhova, ivice i druge geometrijske karakteristike. Za otklanjanje degenerisanih regiona koristi se metoda „*filter_valid_regions*“, čime se obezbeđuje dalja obrada samo značajnih regiona. Svaki region sastoji se od liste indeksa koji se odnose na temena tog regiona. Ukoliko region sadrži indeks -1, koji ukazuje na to da se region proteže u beskonačnost, on se preskače i korisnik se obaveštava o preskočenom regionu. Za svaki važeći region preuzimaju se njegova temena iz niza koji sadrži sve stvarne 3D koordinate svim temena u Voronoi dijagramu. Da bi se formirala Voronoi ćelija neophodno je da region ima minimum četiri temena. Ukoliko region nema četiri temena on se odbacuje. Za regione koji su prošli proveru, svako teme važećeg regiona se ubacuje u posebnu listu. Na kraju se radi vizuelizacija validnih Voronoi regiona. Za svaku ćeliju se kreira nova mreža pomoću Bmesh modula. Koordinate svakog temena se konvertuju u Blenderov vektorski format. Oko ovih koordinata, za svaki region, pravi se konveksni omotač, samim tim formira se Voronoi ćelija. Na kraju se generisana mreža pretvara u običnu mrežu i od nje se kreira Blender objekat koji se ubacuje u kolekciju.



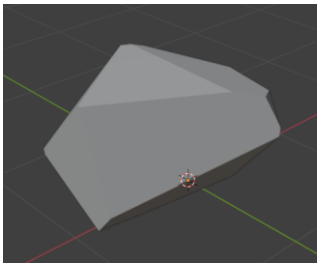
Slika 6. *Vizuelizovane Voronoi ćelije*

3.3. Boolean operacije

Za sečenje originalnog objekta kreiranim Voronoi ćelijama koristi se Boolean operacija „*intersect*“ koja osigurava da je rezultat operacije presek između originalne mreže objekta i Voronoi ćelije. Kreirana mreža u prethodnom koraku kreirana je radi vizuelizacije, te se postupak kreiranje mreže ponavlja kako bi se ta mreža iskoristila za sečenje objekta. Za svaku ćeliju se kreira logički modifikator, a zatim se njemu dodeljuje originalna mreža. To znači da će se Voronoi ćelija ukrštati sa originalnim objektom. Primenom Boolean modifikatora operacija preseka se završava, a rezultat je presek originalnom objekta i Voronoi ćelije.



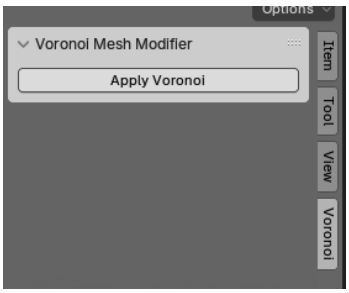
Slika 7. *Rezultat Boolean operacije*



Slika 8. *Jedan segment kocke*

3.4. Panel

Radi lakšeg rukovanja dodatkom unutar samog Blendera kreiran je panel koji jednim klikom poziva dodatak. Za to se koristi klasa „*OBJECT_PT_voronoi_panel*“ koja definiše panel u Blenderovom 3D prikazu. Panel će biti kreiran kao kartica u okviru korisničkog interfejsa pod imenom „*Voronoi*“, i sadržaće dugme koje pokreće klasu „*OBJECT_OT_voronoi_tessellation*“ u kojoj se nalazi Voronoi tesalacija.



Slika 9. Panel za Voronoi tesalaciju

4. MOGUĆA UNAPREĐENJA

Očigledno je da se ovaj projekat može znatno poboljšati kako bi se poboljšale performanse, upotrebljivost i pristupačnost. Ova poboljšanja bi omogućila umetnicima da manipulišu granicama proceduralnog modelovanja, a u isto vreme otključava nove kreativne mogućnosti.

5. ZAKLJUČAK

Cilj ovog rada bio je predstavljanje jednog od načina na koji se Voronoi dijagram može koristiti u računarskoj grafici, kako bi pružio umetnicima i dizajnerima alat za generisanje vizuelno složenih 3D modela. Skripta omogućava automatsku podelu objekta na nepravilne fragmente koji se mogu koristiti u različite svrhe, od apstraktne umetnosti do realističnih simulacija poput napuknute površine.

Glavni izazov pri izradi bile su neograničene Voronoi regije sa indeksom -1. Njihovo odbacivanje rezultiralo je neobrađenim delovima objekta, samim tim neuspešnom Voronoi tesalacijom. Kako bi se ovaj problem zaobišao, iskoristilo se rešenje proširivanja graničnog okvira objekta, kao i povećanje broja nasumičnih tačaka generatora. Na ovaj način, sigurno je da će regija koja je odbačena da bude zamenjena validnom regijom.

6. LITERATURA

[1] https://bozidarvladislav.wordpress.com/

[2] Tabak, John (2014), Geometry: The Language of Space and Form, Facts on File math library, Infobase Publishing, p. 150, ISBN 978-0-8160-6876-0

[3] Black, Paul E. "Manhattan distance". Dictionary of Algorithms and Data Structures. Retrieved October 6, 2019

[4] https://builtin.com/data-science/voronoi-diagram

[5] Python, 3.12.5 Documentation, The Python Tutorial, https://docs.python.org/3/tutorial/

[6] Pajankar Ashwin, Naučite Python 3: brzi kurs programiranja, Agencija Eho, 2022

[7] https://numpy.org

[8] https://scipy.org

[9] https://shapely.readthedocs.io/en/stable/manual.html

[10] https://docs.blender.org/api/current/mathutils.html#

[11] https://de.mathworks.com/help/matlab/math/voronoi-diagrams.html