Fakultet elektrotehnike i računarstva

Računalna grafika

Samostalna vježba

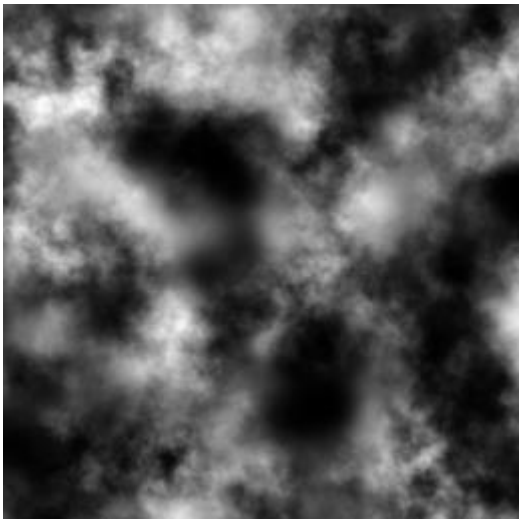
Proceduralno generiranje terena

Zvonimir Kučiš, 0036494839

Zagreb, 2020.

1. Proceduralno generiranje terena

Postoje mnogobrojni načini za generiranje terena, najlakši i zapravo temelj za većinu sustava je generiranje teksture visina(eng. *heigthmaps*). Teksture visina su 2D koordinatne mreže u kojoj svaka točka predstavlja visinu terena. Grafički prikaz teksture je, uglavnom, u sivim tonovima(eng. g*rayscale*) gdje vrijednost boje u pojedinom pikselu predstavlja visinu dotične točke na terenu(slika 1.).



Slika 1. Prikaz teksture visina,
(https://en.wikipedia.org/wiki/Heightmap#/media/File:Heightmap.png)

Generiranje teksture možemo izvesti da nasumično generiramo vrijednost visine te zapišemo to u teksturu. Kod ovakvog pristupa se javlja problem naglih prijelaza u visini terena, generiran teren bi mogao postati kaotičan, tj. s velikim promjenama na kratkim udaljenostima.

Da bi dobili bolji efekt terena, koriste se pseudo nasumične vrijednosti. Kod takvih metoda novo generirani broj nije sasvim nasumičan nego postoji neki odnos među njima. Najčešće korišteni algoritam za dobivanje takvih vrijednosti je Perlinov šum, nazvan prema njegovom kreatoru Ken Perlin-u. Perlinov šum može se koristiti za dobivanje raznih tekstura i oblika. Njime možemo generirati već spomenute teksture visine, ali i teksture vatre, teksture organskih tvari(slika 2.) itd.

Perlinov šum je relativno star algoritam i ima svoje nedostatke, pa je tako Ken Perlin 2001. godine razvio Simplex algoritam koji je u principu nadogradnja originalnog Perlinovog šuma. Simplex algoritam generira manje usmjerenih artefakata, radi bolje u prostoru većih dimenzija te koristi manje računalne moći. Kao prvi korak, algoritam stvara konstantnu mrežu trokuta te za svaki vrh trokuta generira nasumični gradijent. Veličina rešetke i gradijenti su konstante i mogu se izračunati u bilo kojem trenutku i mjestu, tako da jedini potrebni unosi su koordinate točke za koju nas zanima slučajna vrijednost. Algoritam vraća decimalan broj, obično vrijednost između 0 I 1 ili –1 I 1, što je rezultat zbroja utjecaja pojedinih vrhova trokuta u kojem se dana točka nalazi.

Budući da je Simplex algoritam zaštićen patentom razvijena je inačica koja se zove OpenSimplex. Taj algoritam je izvedba Simplex algoritma, ali otvorenog koda pa se slobodno koristi u bilo kojim aplikacijama. OpenSimplex se koristi I u ovom radu za generiranje teksture visine.

Slika 2. Prikaz teksture generirane Perlinovim šumom,

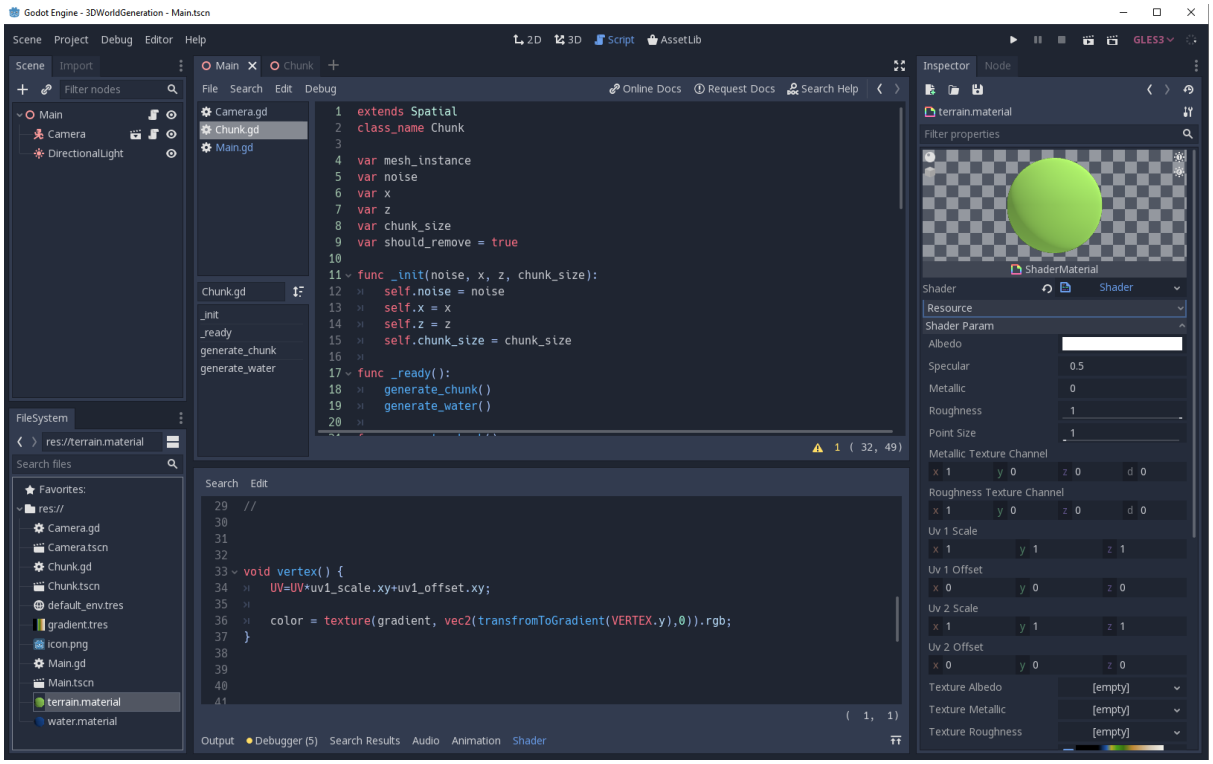
(https://en.wikipedia.org/wiki/Perlin_noise#/media/File:Pink_red_liquid_using_perlin_noise_+_bump_+_coloring_(2415197699).png)

2. Godot

Godot je 2D i 3D, besplatan i otvorenog koda pogonski sklop igre(eng. g*ame engine*). Godot želi ponuditi potpuno integrirano okruženje za razvoj igara. Omogućuje programerima da kreiraju igru od početka, a ne trebaju druge alate osim onih koji se koriste za stvaranje sadržaja (teksture, modeli, glazba itd.). Arhitektura je izgrađena oko koncepta stabla ugniježđenih "scena". Svi resursi za igre, od skripti do grafičkih sredstava, spremaju se kao dio datotečnog sustava računala. Podržava više platformi i omogućava specifikaciju postavki kompresije teksture i rezolucije za svaku platformu. Trenutno podržane platforme uključuju Linux, macOS, Windows, BSD, Android, iOS, BlackBerry 10, HTML5 i WebAssembly. Tu je i podrška za Windows Runtime i Universal Windows Platform.

Godot igre su kreirane pomoću C++, C#, jezicima s GDNative vezama, poput Rust, Nim, D, ili korištenjem vlastitog skriptnog jezika, GDScript, programskog jezika vrlo sličnog Pythonu. Suprotno Pythonu, GDScript sadrži strogo tipkanje varijabli i optimiziran je za Godotovu scensku arhitekturu.

Godot uključuje i uređivač skripti s automatskim uvlačenjem, isticanjem sintakse i dovršetkom koda. Sadrži i program za uklanjanje pogrešaka s mogućnošću postavljanja prijelomnih točaka i koraka programa.



Slika 3. Izgled Godot okruženja

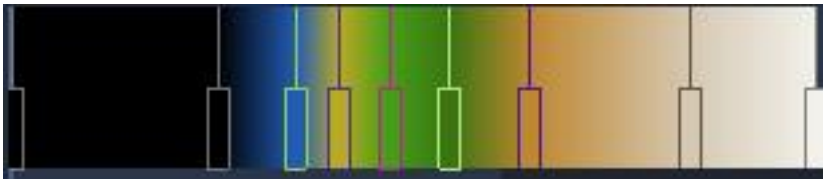
3. Implementacija

Implementacija je izvedena u tri scene: *Main, Camera, Chunk*. *Camera* scena je zadužena za samo kretanje igrača, tj. kamere po mapi. Kod za kretanje je preuzet sa službenog tutorijala od Godot-a (https://docs.godotengine.org/en/3.1/tutorials/3d/fps_tutorial/part_one.html). Kamera je postavljena iznad terena, tako da zapravo lebdimo nad terenom.

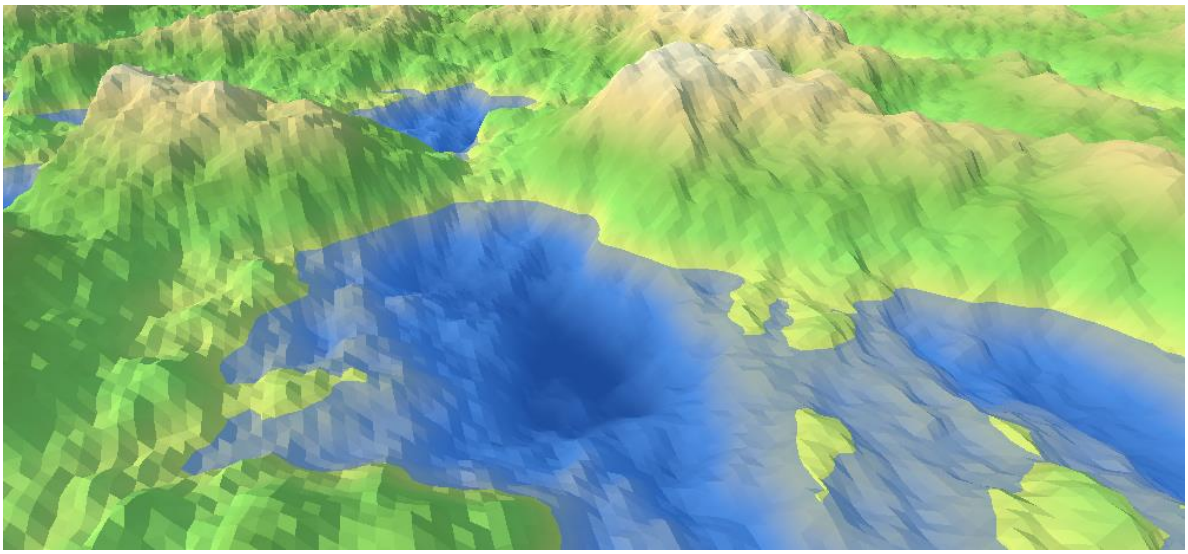
Chunk je jedan dio mape, on je zadužen za samo generiranje terana pomoću OpenSimplex šuma. Prvo se stvara mreža točki te se tada svakoj točki pridruži pseudo nasumična visina. Tada se ta mreža iscrtava čime dobijemo dojam terena. Unutar ove scene, tj. skripte generira se i ploha koja je postavljena na predefiniranu visinu te prozirne plave boje. Tom plohom se stvar privid površine vode.

Main je glavna scena u kojoj se sve odvija, ona je također zadužena za instanciranje *Chunk-ova.* Instanciranje se vrši oko pozicije igrača(kamere) u mreži od 4 puta 4 instance. Kretanjem igrača stvaraju se novi dijelovi mape, dok se oni od kojih se udaljava prestaju iscrtavati. Samo instanciranje se izvršava na zasebnoj dretvi.

Također dodan je sjenčar vrhova(eng. *vertex shader*) kojim se ovisno o visini pojedinog vrha postavlja boja samog vrha. Boja je određena gradijentom(slika 4.) pa tako možemo simulirati duboke vode, zelene površine te sve do bijelih visina.



Slika 4. Gradijent korišten u sjenčaru vrhova



Slika 5. Konačni izgled scene