Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Maša Planinc

Oblikovanje 3D scene in spletnega vodiča za njeno izdelavo

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Narvika Bovcon
Somentor: as. Blaž Meden

Ljubljana, 2018

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

V diplomski nalogi opišite ključne procese iz računalniške grafike, ki so zajeti v izdelavi 3D scene in 3D lika. Predstavite uporabljena orodja ter sam postopek modeliranja in upodabljanja različnih elementov scene v obliki spletnega vodiča za uporabnika začetnika.

Za vso pomoč in nasvete pri izdeavi diplomske naloge se zahvaljujem svoji mentorici izr. prof. dr. Narviki Bovcon in somentorju Blažu Medenu. Posebej pa bi se rada zahvalila tudi družini in prijateljem za vso podporo, ki so mi jo nudili tekom študija.

Kazalo

Razlaga pojmov

poimenovanje	**angleško**	**razlaga**
Oglišče	Vertex	točka, v kateri se stikajo stranice lika ali robovi ploskev telesa
Rob	Edge	povezava dveh oglišč na 3D objektu
Mnogokotnik, poligon	Polygon	premočrtni lik s tremi ali več stranicami iz katerega je sestavljen 3D objekt
3D	3D(three-dimensional)	tridimenzionalno, tri dimenzije (širina, višina, dolžina)
Objekt	Object	predmet, sestavljen iz mnogokotnikov
3D model	3D model	matematično predstavljen virtualni model, ki je lahko sestavljen iz več objektov
Poligonska mreža	Mesh	del 3D modela, ki definira oglišča objekta in s tem njegovo obliko
Upodabljanje	Render	proces, kjer računalnik izračuna vse elemente skupaj in tako upodobi končno 2D sliko, ki jo vidimo na ekranu
Piksel	Pixel (picture element)	osnovna in najmanjša enota barve na računalniškem zaslonu / sliki, kjer vsakega definira 2D koordinata in njena barva

Voksli	Voxels	piksli z volumnom ali mreža kock, ki predstavlja 3D prostor, kjer vsakega definira 3D koordinata in njena barva
Normala	Normal	Pravokoten vektor
NURBS	non uniform rational B-splines	matematične predstavitve 2D ali 3D predmetov, ki so lahko standardne (npr. kocka) ali proste oblike
BRDF	bidirectional reflectance distribution function	štiridimenzionalna funkcija kotov vpadne in odbite svetlobe
BTDF	bidirectional transmittance distribution function	štiridimenzionalna funkcija kotov vpadne in presevne svetlobe
BSDF	Bidirectional scattering distribution function	posplošen seštevek BRDF in BTDF
Senčilnik	Shader	deli kode, ki se izvajajo na grafičnem procesorju za vsako posamezno sliko na zaslonu
Barvanje vpliva	Weight painting	določevanje uteži vsakemu oglišču
Tekstura	Texture	2D slike kot npr. fotografije ali 3D teksture, ki vplivajo na barvo predmeta
Mapiranje	Mapping	proces, ki specificira, kako se teksture preslikajo na objekt
Šum	Noise	naključni med seboj interpolirani gradienti
Oddajnik	Emitter	sistem, ki oddaja / proizvaja delce

Animacija	Animation	navidezno oživljanje lutk, predmetov ali risanih figur s premikanjem, gibanjem
Manipulacija modela	Rigging	dodajanje kontrol za objekte po navadi za animiranje
Armatura	Armature	omogoča objektom, da imajo fleksibilne sklepe, po navadi jo sestavljajo kosti
Ključne oblike	Shape keys	ključne oblike objekta, med katerimi se oblika objekta s prehodom spreminja, ta tip animacije se imenuje morf, uporablja se npr. za animiranje obrazne mimike
Urejevalnik vozlišč	Node editor	urejevalnik za materiale, luči, ozadja, itd., ki so definirana z vozlišči
HTML	Hyper Text Markup Language	jezik za označevanje nadbesedila
CSS	Cascading Style Sheets	kaskadne stilske predloge
JavaScript	JavaScript	objektni, skriptni programski jezik
Python	Python	programski jezik visokega nivoja

Povzetek

Naslov: Oblikovanje 3D scene in spletnega vodiča za njeno izdelavo

Avtor: Maša Planinc

Vedno več stvari se izdeluje s pomočjo 3D modelov in veliko gradiva, ki smo ga prej našli le v knjigah, se pojavlja na spletnih straneh. V diplomski nalogi smo želeli raziskati področje 3D modeliranja. Opisali smo orodja, ki smo jih uporabili pri izdelavi 3D okolja, lika in spletne strani. Teoretično smo opisali procese, ki se uporabljajo pri 3D modeliranju, kjer smo se osredotočili predvsem na orodje Blender. Napisali smo vodič, ki nas vodi skozi postopek izdelave 3D scene, je del spletne strani in predstavlja komplementarni del pisnemu delu diplomske naloge. Predstavili smo tudi postopek načrtovanja, oblikovanja in izdelave spletne strani.

Ključne besede: 3D modeliranje, Blender, osvetljevanje, materiali in teksture, sistem delcev, spletna stran, vodič.

Abstract

Title: A web page tutorial on how to create a 3D scene and a 3D character

Author: Maša Planinc

More and more things are made using 3D models and a lot of material we previously found only in books now appears on web pages. In the diploma work, we wanted to explore the field of 3D modelling. We describe the tools used in the development of a 3D environment, a character and a website with tutorials. We theoretically describe the processes used in 3D modelling, where we primarily focus on the Blender tool. A complementary part of this thesis is also a tutorial, that guides us through the process of making a 3D scene and is a part of the website. We also present the process of planning, designing and creating a website.

Keywords: 3D modelling, Blender, Lighting, Materials and textures, Particle system, Web site, tutorial.

Poglavje 1

Uvod

V današnjem svetu vse bolj napredne tehnologije se je razvilo veliko različnih tehnik 3D modeliranja. Ker me je vedno zanimal postopek 3D oblikovanja, ki se lahko uporablja za igre, animirane filme, 3D tiskanje, itd., sem se odločila, da ga bom bolj podrobno raziskala. Na področju 3D modeliranja obstaja veliko različnih orodij in še več funkcij, kar pa je lahko za nekoga, ki se na to ne spozna, težavno. Zato sem se odločila, da pripravim spletno stran, ki bo vodila uporabnika korakoma skozi izdelavo 3D scene in mu tako omogočila lažji vstop v 3D modeliranje. Pri oblikovanju spletne strani je potrebno premišljeno oblikovati celostno grafično podobo, izbrati vsebino in komplementarno slikovno gradivo ter prilagoditi stran več različnim napravam. Del diplomske naloge je tako tudi spletna stran z vodičem za 3D modeliranje, ki se nahaja na spletni povezavi http://green.fri.uni-lj.si:9026/. V nadaljevanju diplomske naloge je predstavljena vsa programska oprema in pojasnjeni vsi procesi, uporabljeni pri izdelovanju 3D scene, prav tako je opisan tudi postopek izdelave spletne strani in njenega oblikovanja.

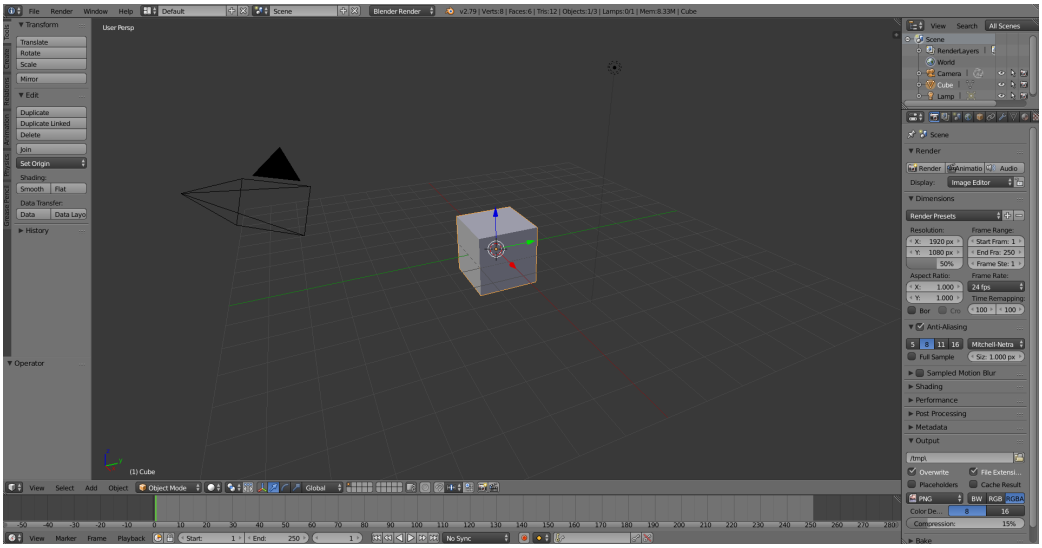
Poglavje 2

Programska oprema in tehnologija

V diplomski nalogi smo lik modelirali v ZBrushu in ga nato izvozili v Blender, kjer smo ustvarili tudi okolje, liku smo nato dodali okostje, ga postavili v okolje in celotno sceno upodobili. Postopek izdelave smo v obliki vodiča opisali na spletni strani, ki smo jo ustvarili z orodji HTML, CSS in JavaScript.

2.1 Blender

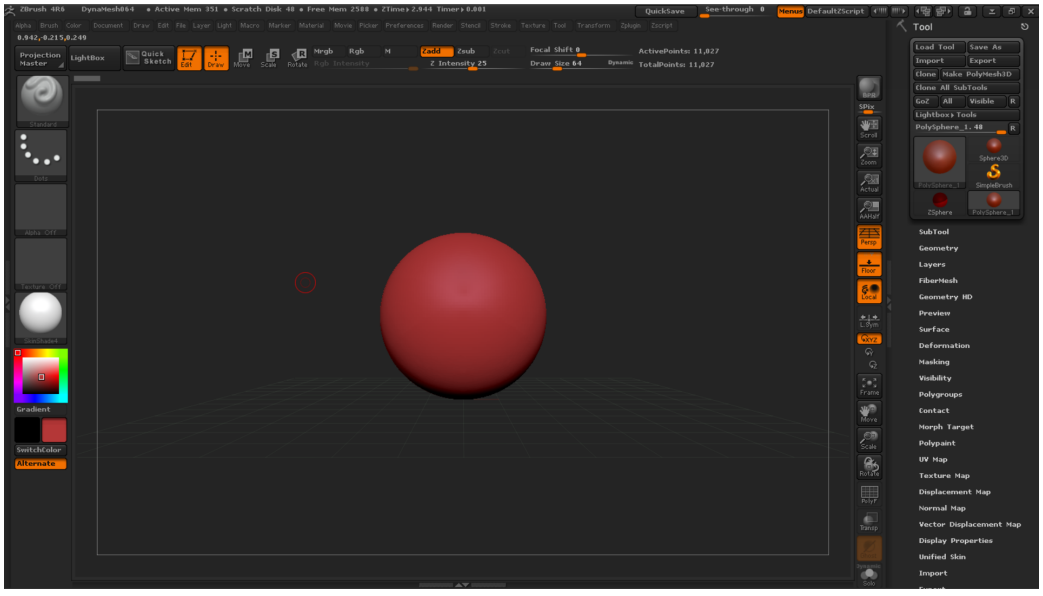
Blender je brezplačno odprto programsko orodje za grafično 3D modeliranje, animiranje, post produkcijo, 3D manipulacijo v realnem času in še mnogo drugega. Ima vgrajen programski jezik Python, s katerim lahko uporabnik še dodatno izboljša svoje delo. Ima sorazmerno majhne zahteve glede pomnilnika in pogona v primerjavi z drugimi 3D programi. Vmesnik uporablja OpenGL, ki zagotavlja dosledno izkušnjo na vseh podprtih strojnih platformah. Z novejšimi različicami se je Blender močno približal, če ne celo prehitel svoje konkurente, kot so Maya, 3D Max, Cinema 4D. Blender uporabniku ponuja tri vrste upodabljanja: Blenderjevo upodabljanje (Blender Render), ciklično upodabljanje (Cycles Render) in upodabljanje iger (Blender Game). Ker je bila moja scena narejena s cikličnim upodabljanjem, bom v diplomski nalogi opisovala postopke, ki jih uporablja ta način upodabljanja [1, 2].



Slika 2.1: Blender ogrodje [3].

2.2 ZBrush

ZBrush "je digitalno kiparsko orodje za 2.5D in 3D modeliranje, teksturiranje in slikanje"[4]. Uporablja posebno "pixol"tehnologijo, ki omogoči shranjevanje različnih informacij (npr. o materialih, razsvetljavi, barvi in globini) za vse prikazane predmete. Glede na ostale programe za modeliranje je ZBrush bolj podoben kiparjenju. Uporablja se lahko za ustvarjanje modelov z visoko ločljivostjo, ki se nato uporabljajo v filmih, animacijah in igrah. Najbolj je znan po tem, da je sposoben oblikovati visoko frekvenčne podrobnosti. Rezultate pa lahko nato uporablja tudi pri nižji poligonski različici istega modela. ZBrush ima dober izkoristek programske opreme, tako da lahko uporabnik dela z milijoni poligonov, brez preobremenitve računalnika. Tako kot Blender tudi Zbrush deluje na MacOS in MS-Windows sistemih, ne pa na Linuxu [4, 5].



Slika 2.2: Zbrush ogrodje [3].

2.3 Python

Python je programski jezik visokega nivoja, ki se uporablja za različne namene. Vsebuje konstrukte, ki omogočajo lažje programiranje, ima dinamičen sistem in samodejno upravljanje pomnilnika. V Blenderju lahko Python uporabimo za zagon orodja z lastnimi nastavitvami, ustvarimo elemente uporabniškega vmesnika, nova interaktivna orodja, nove motorje za upodabljanje in rišemo v 3D pogledu z ukazi OpenGL. [6]

```
>>> bpy.ops.mesh.flip_normals()
{'FINISHED'}
>>> bpy.ops.mesh.hide(unselected=False)
{'FINISHED'}
>>> bpy.ops.object.scale_apply()
{'FINISHED'}
```

Slika 2.3: Modul bpy.ops.

Primer upravljanja z operatorji, do katerih po navadi uporabnik dostopa preko menija ali z bližnjicami. Python jih lahko upravlja preko modula, imenovanega bpy.ops [7].

2.4 HTML

"HTML (jezik za označevanje nadbesedila) je označevalni jezik za izdelavo spletnih strani in spletnih aplikacij"[8]. Skupaj z CSS in JavaScriptom predstavlja temeljno tehnologijo za splet. Elementi HTML so gradniki spletne strani in s konstrukti HTML lahko v spletno stran vključimo slike in druge interaktivne oblike. HTML omogoča ustvarjanje strukturiranih dokumentov s pomočjo strukturne semantike za besedilo (npr. naslovi, odstavki, seznami, slike in drugi elementi). Dokument opisujemo s posebno sintakso, ki je sestavljena iz značk in njim pripadajočih atributov. Za izdelavo dokumenta ne potrebujemo nobenih orodij, saj zadostuje vsak urejevalnik besedila.

```
<ImeOznake ImeAtributa="VsebinaAtributa">Vsebina</ImeOznake>

<p class="naslov">Moj naslov</p>
```

Slika 2.4: Splošna sintaksa HTML.

Elementi HTML so sestavljeni iz značk, zapisanih v špičastih oklepajih. Značke vedno pišemo v parih začetnih in končnih značk, med njimi pa se nahaja vsebina spletne strani. Značkam lahko dodamo razne atribute, ki so povezani s CSS, lahko pa jih tudi gnezdimo [8].

2.5 CSS

CSS (kaskadne stilske podloge) so v obliki preprostega slogovnega jezika predstavljene podloge, ki se uporabljajo za predstavitev spletnih strani. Z njimi definiramo stil HTML elementov, ki jim lahko določamo barve, velikosti, odmike, poravnave, pozicije in še veliko drugih atributov. CSS pa nam omogoča tudi nadzor aktivnosti, ki jih uporabnik izvaja nad elementi strani. Namen podlog je podajanje informacij o stilu spletnega dokumenta, prav tako z njimi ločimo vsebino od oblikovanja.

```
selektor1 [, selektor2, selektor3 ...][:psevdo-razred ali :psevdo-element]{
        lastnost1: vrednost1;
        [lastnost2: vrednost2;
        lastnost3: vrednost3;
        ...]
}

.naslov {
    font-weight: normal;
    font-size: 40px;
    margin: 20px;
}
```

Slika 2.5: Splošna sintaksa CSS in oblikovanje naslova [9].

Sintaksa CSS je relativno enostavna. Sestavljajo jo angleške besede, ki predstavljajo stilske lastnosti. Podlogo sestavljajo različna pravila, ki so določena za enega ali več selektorjev. Selektorji nam povedo, katere elemente v HTML želimo oblikovati. Nato znotraj selektorja določimo, katere lastnosti bomo oblikovali, in jim podamo vrednosti [9].

2.6 JavaScript

JavaScript je odprtokodni objektni skriptni programski jezik, ki pomaga pri ustvarjanju interaktivnih spletnih strani. Po navadi ga uporabljamo v sodelovanju s HTML-kodo, kar nam omogoča delo z dinamičnimi vsebinami, torej naloge, ki jih ni mogoče izvesti s statično stranjo, npr. odpiranje raznih oken, izračuni, preverjanje pravilnosti vnesenih podatkov. V diplomski nalogi smo ga uporabljali za pomoč pri odzivnih dizajnih in s tem omogočili prilagojenost strani različnim napravam [10].

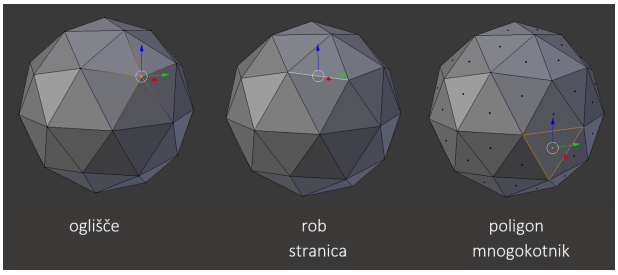
```
function open() {

    if(document.getElementById("mySidebar").style.display == "none")
        document.getElementById("mySidebar").style.display = "block";
    else
      document.getElementById("mySidebar").style.display = "none";
```

Slika 2.6: Funkicja open() za prikaz stranskega menija.

Poglavje 3

3D modeliranje

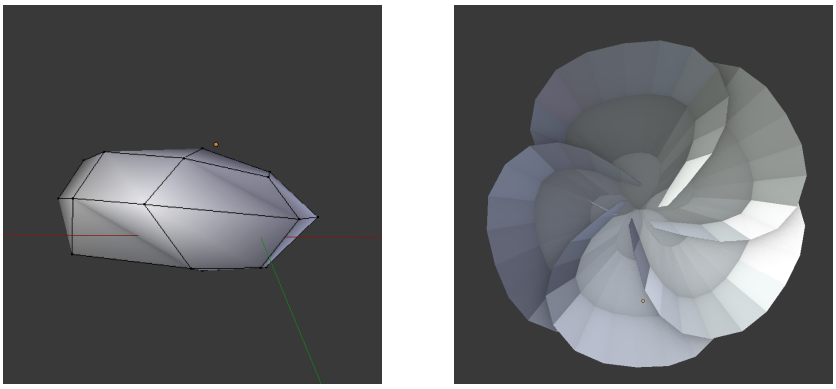
V 3D računalniški grafiki je 3D modeliranje (ali tridimenzionalno modeliranje) proces "razvijanja matematične predstavitve katerekoli površine predmeta v treh dimenzijah prek programske opreme"[11]. Izdelek se imenuje 3D model, ki ga lahko prikažemo kot dvodimenzionalno sliko s postopkom, imenovanim 3D-upodabljanje (3D rendering). 3D predmet predstavimo z množico poligonov (mnogokotnikov). Poligon je ploskev, ki jo oklepajo povezane daljice. Daljice, ki sestavljajo mnogokotnik, imenujemo stranice ali robovi mnogokotnika, točke, v katerih se stranici stikata, pa oglišča. Poligoni se uporabljajo za modeliranje, saj s premikanjem le teh preoblikujemo model. Obstaja veliko načinov za ustvarjanje 3D modela; v nadaljevanju bom opisala tri najbolj uporabljene [11].



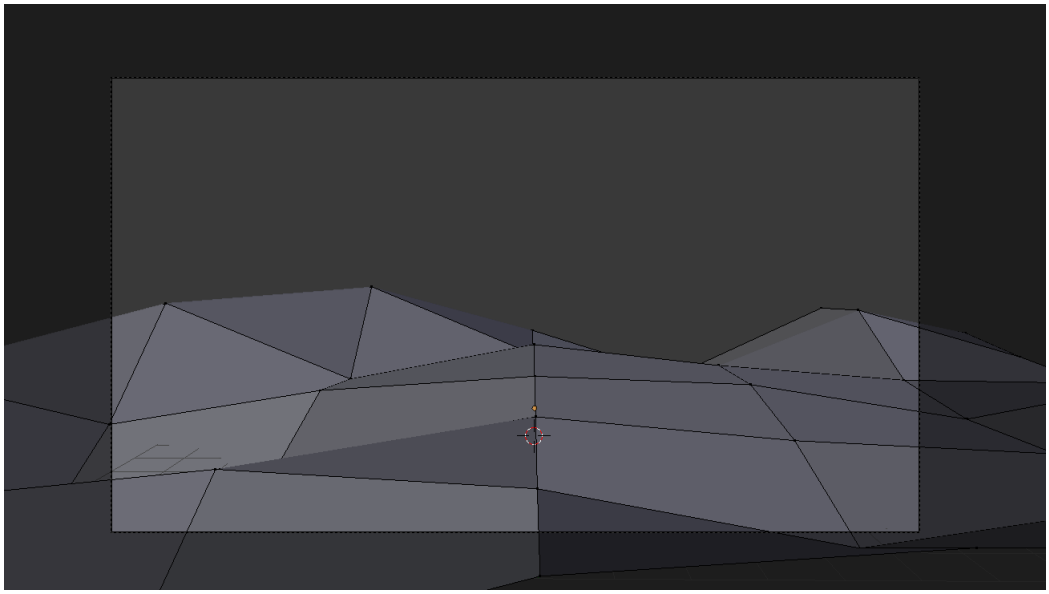
Slika 3.1: Načini označevanja poligonov v Blenderju [3].

3.1 Poligonalno modeliranje

Pri poligonalnem modeliranju (polygonal modeling) so točke v 3D prostoru, imenovane oglišča, povezane z linijami tako, da se ustvari poligonska mreža. Veliko 3D modelov je zgrajenih kot poligonalni modeli, ker so prilagodljivi in jih je lahko hitro narediti. Tak način modeliranja smo uporabili tudi pri izdelavi terena, cvetlic in kamnov, kot lahko vidimo na slikah 3.2 in 3.3. Ker pa so poligoni ravne ploskve, lahko ponazarjajo ukrivljenost le z uporabo mnogih poligonov. Zato se je tudi razvilo modeliranje s krivuljami [11].



Slika 3.2: Poligonalno oblikovanje cvetlice in kamna v Blenderju [3].



Slika 3.3: Poligonalno oblikovanje tal v Blenderju [3].

3.2 Modeliranje krivulj

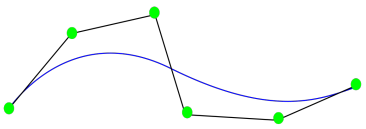
Ta metoda predstavlja predmete s krivuljami in ploskvami (curve modeling), ki jih krivulje definirajo. Najbolj standardne krivulje v 3D oblikovanju so Neenakomerni racionalni B zlepki - NURBS. Vsaka krivulja ima svoje kontrolne točke in vsaka kontrolna točka ima še utež, ki določa njen vpliv na krivuljo. Povečanje uteži točke bo krivuljo približalo točki. Tak način modeliranja je idealen za organsko modeliranje, saj imajo organske oblike veliko ukrivljenih linij [12, 13, 14].

Če je k število kontrolnih točk P_k in ustreznih uteži w_k ter želimo NURB krivuljo stopnje $m(k = m + 1)$, potem potrebujemo $n + k$ vozov. NURB krivuljo nato lahko predstavimo z naslednjo formulo:

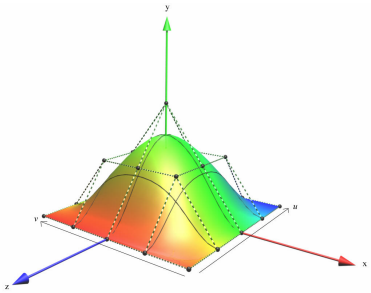
$$\mathbf{p}(t) = \sum_{k=0}^{n} R_{k,d}(t)\mathbf{p}_k = \sum_{k=0}^{n} \frac{w_k \mathbf{B}_{k,d}(t)}{\sum_{l=0}^{n} w_l \mathbf{B}_{l,d}(t)} \mathbf{p}_k$$

kjer je $B_k(t)$ k-ta funkcija B zlepka stopnje m. Če so vsi w_k enaki, potem se izničijo in dobimo polinom B zlepka:

$$\mathbf{p}(t) = \sum_{k=0}^{n-1} \mathbf{p}_k \mathbf{B}_{\mathbf{k,d}}(t)$$



(a) Zelene pike na sliki predstavljajo kontrolne točke, ki oblikujejo krivuljo v modri barvi [15].

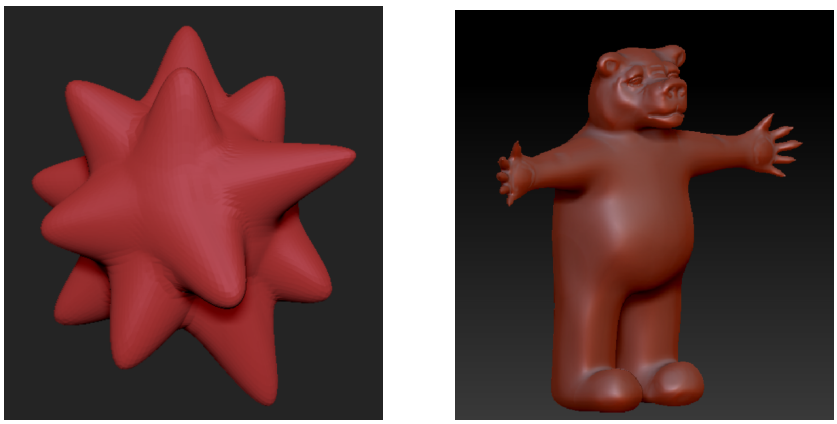


(b) Na sliki lahko vidimo, kako NURBS predstavljajo kompleksno, organsko obliko. Kontrolne točke vplivajo na obliko površine [16].

Slika 3.4: Krivuje uporabljene za 3D modeliranje.

3.3 Digitalno oblikovanje skulptur

Digitalno oblikovanje skulptur (digital sculpting) manipulira z digitalnim predmetom, kot če bi bil ta izdelan iz resnične snovi. Večina programov na trgu uporablja geometrijo poligonskih mrež, kjer je predmet predstavljen z medsebojno povezanimi poligoni, ki jih je mogoče potiskati in vleči naokoli kot vidimo na sliki 3.3. Dobra stran poligonskih mrež je, da podpirajo oblikovanje pri več ločljivostih na enem samem modelu. Zato imajo lahko območja modela z veliko podrobnostmi zelo majhne poligone, medtem ko imajo druga območja večje poligone. Vendar pa lahko ureditev mnogokotnikov omeji možnosti dodajanja in manipuliranja podrobnosti.
Drug način oblikovanja uporablja geometrijo na osnovi vokslov, kjer prostornino predmeta predstavlja sam objekt. Material lahko uporabnik dodaja in odstranjuje, podobno kot bi oblikoval z glino. Prednost oblikovanja z voksli je, da omogočajo veliko svobode pri oblikovanju raznih podrobnosti modela, so pa bolj omejeni pri oblikovanju z ravnimi ploskvami. Topologijo modela se zaradi dodajanja in odstranjevanja materiala med oblikovanjem (lahko) stalno spreminja. S takim načinom modeliranja je oblikovan tudi medved v ZBrushu [11].



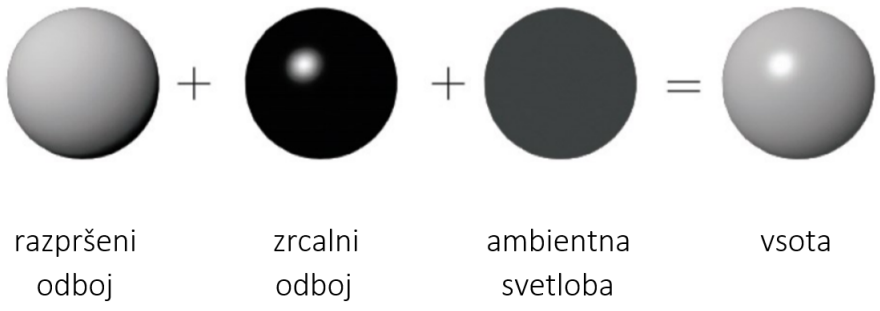
Slika 3.5: Digitalno oblikovanje skulptur v ZBrushu [3].

Poglavje 4

Osvetljevanje

4.1 Osvetljevanje

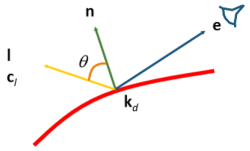
Svetloba je pri modeliranju zelo pomembna, saj je od osvetlitve odvisen izgled materialov in tekstur. Na barvo predmeta pa bo vplivala tudi posameznikova sposobnost zaznavanja različnih barv, medij, kjer je slika predstavljena, in kakovost slike. Pri osvetljevanju računamo osvetlitev neke površine. Odboj svetlobe nam pove, kakšno barvo ima predmet. Odbita svetloba je vsota razpršenega odboja, zrcalnega odboja in ambientne svetlobe [2, 17].



Slika 4.1: Seštevek različnih vrst odbojev in svetlobe[17].

Razpršeni odboj dobimo, ko svetloba pade na material, ki odbija svetlobo enakomerno na vse strani (npr. papir). "Odbita svetloba je proporcionalna s kosinusom kota med vpadno svetlobo in normalo na površino. Večji kot je kot, manj je površina svetla"[17].

- **n** ... normala na površino
- **l** ... smer svetlobe
- $\mathbf{k}_d$ ... razpršena odbojnost
- $\mathbf{c}_l$ ... intenziteta vpadne svetlobe
- $\mathbf{c}_d$ ... intenziteta odbite svetlobe

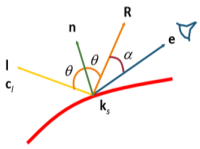


$$\mathbf{c}_d = \mathbf{c}_l \mathbf{k}_d (\mathbf{n} \cdot \mathbf{l}) = \mathbf{c}_l \mathbf{k}_d \cos\theta$$

[17]

Zrcalni odboj je zabrisan odsev vira svetlobe in je odvisen od smeri v katero gledamo. Pri ogledalu imamo idealen odboj, kjer je kot vpada enak kotu odboja. Pri zrcalnem odboju poznamo dva modela in sicer Phongov model, kjer "kot α med idealnim odbojem R in smerjo pogleda e določa količino odbite svetlobe, parameter zrcalnega odboja p določa velikost razpršitve"[17], in Blinnov model, ki je podoben Phongovemu, vendar ne potrebujemo izračuna odboja – imamo nek kompromisni vektor h, ki nam daje hitrejši izračun [17].

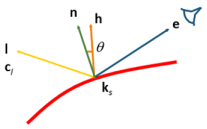
Phongov model



$$\mathbf{c}_s = \mathbf{c}_l \mathbf{k}_s (\mathbf{R} \cdot \mathbf{e})^p$$

$$\mathbf{R} = 2(\mathbf{l} \cdot \mathbf{n})\mathbf{n} - \mathbf{l}$$

Blinnov model



$$\mathbf{c}_s = \mathbf{c}_l \mathbf{k}_s (\mathbf{h} \cdot \mathbf{n})^s$$

$$\mathbf{h} = \frac{\mathbf{l} + \mathbf{e}}{|\mathbf{l} + \mathbf{e}|}$$

[17]

V realnosti je del svetlobe povsod, ker se le ta odbija od ostalih predmetov. Temu pri lokalni osvetlitvi pravimo ambientna svetloba in zato povsod dodamo konstantno osvetlitev $c = c_a k_a$. Ker imamo lahko več virov svetlobe, seštejemo vse prispevke in tako dobimo model lokalnega osvetljevanja:

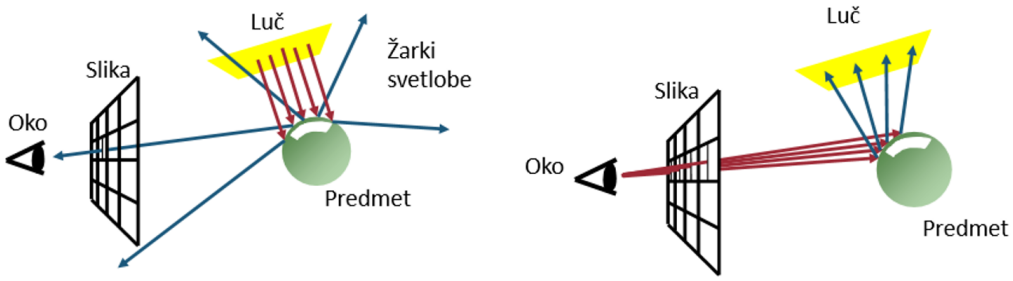
$$\mathbf{c} = \mathbf{c}_a \mathbf{k}_a + \sum_i \mathbf{c}_i \left(\mathbf{k}_d (\mathbf{l}_i \cdot \mathbf{n}) + \mathbf{k}_s (\mathbf{h}_i \cdot \mathbf{n})^s \right) \quad [17]$$

4.2 Senčenje

Pri senčenju računamo, kako izrisati osvetljeno površino. Za barvanje poligonov lahko računamo osvetlitev celega poligona, osvetlitev na ogliščih ali pa v vsaki posamezni točki. Pri ploskem senčenju "osvetlitev računamo za cel poligon in ga pobarvamo z eno barvo (npr. izračunamo osvetlitev v enem oglišču)"[17]. To je najbolj primitiven algoritem, rezultat pa nam daje nezvezen izgled. Pri Gouraudovem senčenju računamo osvetlitev v vsakem oglišču in nato v notranjosti interpoliramo med barvami v ogliščih. Za izračun osvetlitve za čim bolj mehke prehode potrebujemo normale v ogliščih, ki jih lahko računamo s povprečenjem normal ploskev, ki se stikajo v oglišču. Do problema pride pri odsevih, ko je število poligonov majhno, zato moramo v takih primerih za boljši rezultat povečati število poligonov. Pri Phongovem senčenju osvetlitev računamo v vsaki točki poligona. Za izračun osvetlitve v točki potrebujemo normalo v točki, ki jo dobimo z bilinearno interpolacijo in je boljše kvalitete kot Gouraudovo senčenje, vendar počasnejše [17].

4.3 Sledenje žarku

Da bi naredili podobo scene, moramo najprej ugotoviti, kakšna svetloba iz scene prihaja na vsako točko na projekcijski ravnini. Najboljši način je metoda sledenja žarku (Ray tracing), kjer namesto da sledimo žarkom od svetlobnih virov do gledišča, sledimo žarkom od gledišča, kot lahko vidimo na sliki 4.1. Tako prihranimo čas, saj veliko žarkov, ki je poslanih v sceno, ne pride do očesa. Najprej sledimo žarku svetlobe od očesa do presečišča s prvim predmetom, kjer gre en žarek skozi vsak piksel v končni sliki. V presečišču s predmetom se izračuna barva z osvetlitvenim modelom. V primeru, da žarek ne seka nobenega predmeta, je piksel črn. Proti vsaki luči pošljemo senčni žarek zato, da upoštevamo tudi sence v vsaki točki preseka. Če je na poti kak predmet, je točka v senci. Nato sledimo žarku v dve smeri. V smer popolnega odboja za materiale, ki imajo zrcalno komponento, in v smer prepuščenega žarka, kadar je predmet prosojen. Celoten postopek sledenja nato rekurzivno ponavljamo, dokler ne zadenemo luči ali pa ničesar, kar pomeni temo. Rekurzijo pa lahko tudi omejimo, saj z vsako iteracijo raste kompleksnost računanja [18].



Slika 4.2: Sledenje žarku od svetlobnega vira do gledišča in sledenje žarku od gledišča do svetlobnega vira [18].

Poglavje 5

Senčilniki, materiali in teksture

5.1 Senčilniki

Senčilniki (Shaders) so deli kode, ki se izvajajo na grafičnem procesorju za vsako posamezno sliko na zaslonu. To pomeni, da se koda spreminja glede na položaj piksla na zaslonu. Poznamo dve večji skupini senčilnikov; 2D in 3D. 2D senčilniki ali senčilniki fragmentov (fragment shaders) delujejo na teksturah in spreminjajo atribute pikslov. Poleg barve izračunavajo tudi druge lastnosti vsakega fragmenta. S senčilniki fragmentov lahko vračamo barve, svetlobne vrednosti, izbokline, sence, prosojnost in druge lastnosti. Prav tako lahko spreminjajo tudi globino fragmentov. 3D senčilniki po navadi delujejo na 3D modelih, vendar lahko spreminjajo tudi barve in teksture, ki so uporabljene na modelu. Poznamo tri glavne senčilnike in sicer senčilnik oglišč (vertex shaders), senčilnik geometrije (geometry shaders) in teselacijo (tessellation shaders). Senčilnik oglišč ja najstarejši tip 3D senčilnika in izvaja operacije, kot so transformacije oglišč, osvetljevanje na posameznih ogliščih. Geometrijski senčilniki lahko ustvarjajo novo geometrijo. Teselacija pa je najnovejša oblika 3D senčilnika in se uporablja za lepljenje odmika in kontrolo nivojev podrobnosti ter deljenje črt in poligonov na manjše dele [19, 20, 21].

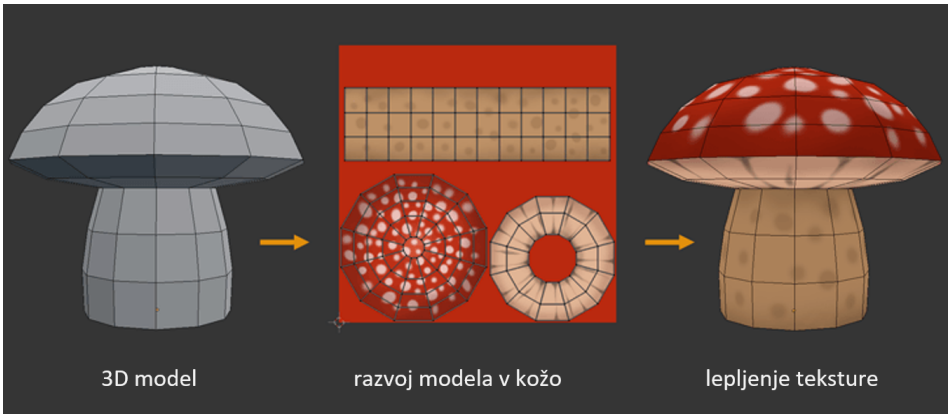
5.2 Materiali

Material predstavlja snov, iz katere je predmet narejen. Snov je po navadi predstavljena s površinskimi lastnostmi, kot so barva, odbojnost, sijaj, prosojnost, itd. Osnovni material v Blenderju je enak po celotni površini predmeta, vendar lahko piksli na različnih delih predmeta izgledajo drugače zaradi svetlobnih učinkov. Pri cikličnem upodabljanju (Cycles Render) lahko material sestavljajo trije senčilniki, ki so prikazani na sliki 5.2, in določajo videz površine, volumen ter oddaljenost površine od poligonske mreže. Površinski senčilnik (surface shader) definira interakcijo svetlobe s površino poligonske mreže. Senčilnik volumna (volume shader) opisuje interakcijo svetlobe, ko ta prehaja skozi poligonsko mrežo. Svetloba se lahko razprši, absorbira ali oddaja v neko točko v prostornini. Material ima lahko oba senčilnika, nobenega ali pa le enega od njiju. Oba skupaj se lahko uporabljata za materiale, kot so steklo, voda ali led. Senčilnik odmika (displacement shader) pa lahko spreminja obliko površine in prostornine. Na ta način se lahko nato teksture uporabijo za dodajanje podrobnosti na poligonsko mrežo [2].

Slika 5.1: Vrste senčilnikov, uporabljenih v cikličnemu upodabljanju [2].

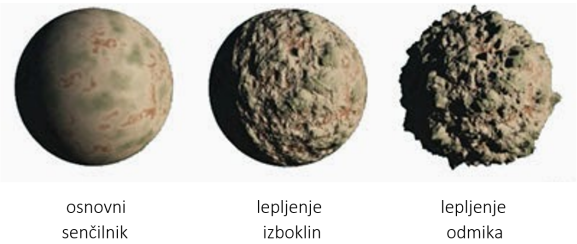
5.3 Teksture

Tekstura je lahko preprosta barva, vzorec, slika ali fotografija, ki s projiciranjem slik na površino dodaja predmetu podrobnosti. Teksture pa ne vplivajo samo na barvo, temveč tudi na senčenje, odsevnost, hrapavost in celo lažno 3D globino. Teksture so kot dodatni sloji na vrhu osnovnega materiala in z vstavljanjem v različne sloje v materialu tvorijo različne efekte. Zaradi lažjega računanja so modeli v računalniški grafiki predstavljeni s trikotniki. Preslikanje teksture na površino poteka tako, da se pikslom teksture določita koordinati u in v, "vsako oglišče trikotnika pa hrani u, v koordinati dela teksture, ki se nanj preslika"[22]. Z interpolacijo u in v koordinat lahko nato za vsako točko v trikotniku izračunamo točko teksture, kamor se le ta preslika. Poznamo več vrst lepljenja tekstur in sicer vzporedno, perspektivno, sferično in, najbolj zahtevno, kožno lepljenje, prikazano na sliki 5.1, kjer površino predmeta razvijemo v 2D kožo, jo pobarvamo in nalepimo nazaj. Tak način lepljenja teksture uporabljamo pri lepljenju teksture na oči medveda [22].



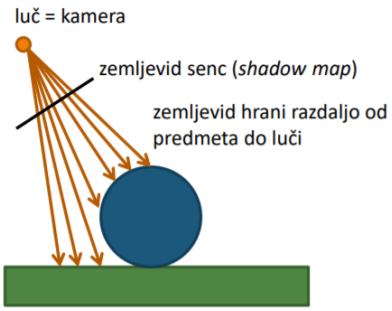
Slika 5.2: Kožno leplenje.[23]

Efekti, ki jih lahko dosežemo s teksturami in senčilniki, so lepljenje izboklin (bump mapping), odmika (displacement mapping), senc (shadow mapping) in še mnogo drugih. Pri lepljenju izboklin tekstura ne spreminja barve, temveč spreminja normale in tako vpliva na izračun osvetlitve. Tekstura je navadno sivinska slika - višinska slika, ki nam pove razliko v višini in koliko moramo spremeniti normalo. Pri tem procesu se geometrija modela ne spreminja. Pri lepljenju odmika je tako kot pri lepljenju izboklin tekstura sivinska slika, le da tu dejansko spreminja geometrijo in s tem položaj poligonov [20, 22].



Slika 5.3: Efekti različnih vrst lepljenja.[24]

Pri lepljenju senc lahko dodajamo občutek globine. Najprej izrišemo sceno s postavitvijo kamere na položaj luči in shranimo globinsko sliko, ki ji rečemo zemljevid senc in predstavlja razdaljo od luči. Nato izrišemo sceno s položaja kamere in na vsakem pikslu razdaljo, shranjeno v zemljevidu senc, primerjamo z razdaljo do luči. Če je razdalja daljša, smo v senci, drugače pa je piksel osvetljen [18].



Slika 5.4: Izris zemljevida senc.[18]

5.4 Urejevalnik vozlišč

Pri uporabi cikličnega upodabljanja (Cycles Renderja) so materiali, luči in ozadja definirani z vozlišči (nodes). Izhodne vrednosti vozlišč so vektorji, barve in senčilniki. S sistemom vozlišč opisujemo vrstni red obdelave podatkov. Podatki se začnejo obdelovati v vozliščih, ki opisujejo vire (input nodes), nato gredo skozi vozlišča, ki predstavljajo različne stopnje obdelave in filtriranja, in končajo v vozliščih, ki predstavljajo izhode ali cilje (output nodes). Vozlišča se povezujejo drugo z drugim na več različnih načinov in omogočajo prilagajanje lastnosti za vsako vozlišče posebej. Poznamo vhodna in izhodna vozlišča, vozlišča senčilnikov, barv, tekstur, skript, vektorjev, pretvorb in skupine vozlišč. Opisala bom nekaj najbolj uporabljenih vozlišč v moji sceni. Kratica BSDF opisuje način, kako se svetloba razprši na površino [2].

Razpršenost BSDF (Diffuse BSDF)
Predstavlja razpršen odboj, ki je prikazan na sliki 5.5. Uporablja se za materiale, kot sta papir ali les. Vhodni podatki so barva in hrapavost površine ter normala, ki se uporablja za senčenje. Izhod pa je standardi izhod senčilnika [2].

Slika 5.5: Razpršenost BSDF [3].

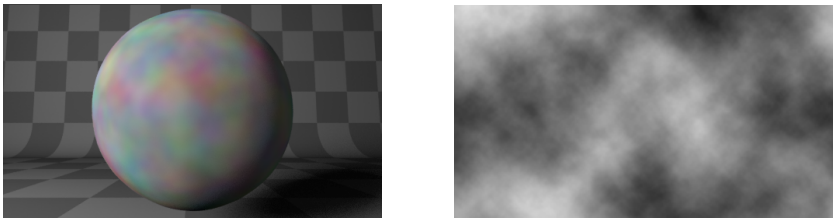
Sijaj BSDF (Glossy BSDF)

Uporablja se za dodajanje odseva z distribucijo mikrofaset in se uporablja za materiale kot so kovine ali ogledala. Vhodni podatki so barva in hrapavost površine ter normala, ki se uporablja za senčenje. Lahko pa izbiramo iz naslednjih lastnosti: distribucijo, ki nam pove porazdelitev mikrofaset, ostrino, ki se odraža v popolnoma ostrih odsevih kot ogledalo, in večkratno razpršenost, ki upošteva več odbojev in daje temnejši videz. Izhod pa je standardni izhod senčilnika [2].

Slika 5.6: Sijaj BSDF [3].

Tekstura šuma (Noise texture)

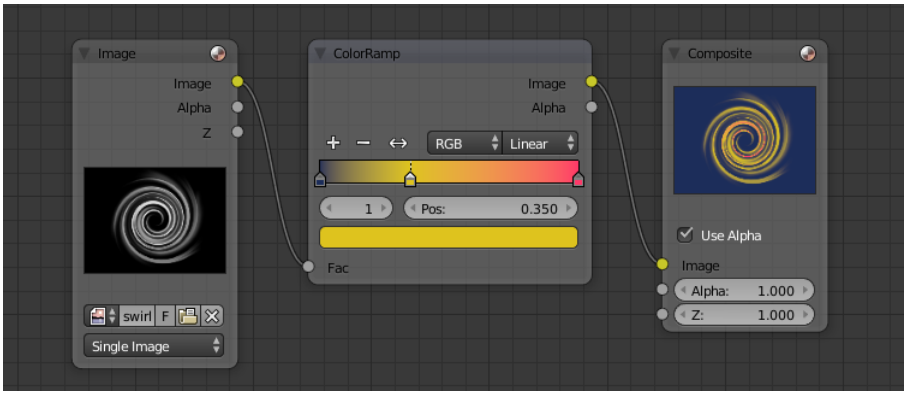
Uporablja se za dodajanje Perlinovega šuma, ki je gradientni šum z interpolacijo med naključnimi gradienti, in omogoča mehke prehode ter lahek nadzor nad frekvenco in fazo. Vhodni podatki so vektor za koordinate teksture, velikost teksture, količina podrobnosti in količina popačenja. Izhodni podatki pa so barva in intenziteta teksture [2, 22].



Slika 5.7: Tekstura šuma [3].

Barvna lestvica BSDF (Color Ramp)

Vsaka črnobela slika je sestavljena iz pikslov z vrednostmi med 0 in 1, pri čemer je 0 črna, 1 pa bela. Barvna lestvica vzame vhodno vrednost in jo v primeru, da ni med 0 in 1, pretvori ter vsakemu pikslu določi novo vrednost. Lahko se uporablja za dodajanje barve z uporabo gradienta, za določanje odmika glede na teksturo, ustvarjanje alfa mask. Barvna lestvica omogoča uporabniku, da določi več barv na podlagi barvnih postaj. Barvne postaje kažejo, kje naj bo izbrana barva. Intervali med postajami so rezultat barvne interpolacije in izbrane metode interpolacije. Izhodni podatki pa so obdelana slika in alfa vrednost [2].



Slika 5.8: Vozlišče Barvna lestvica [2].

Gama BSDF (Gamma)

Uporablja se za dodajanje gama popravkov. Vhodni podatki so slika in gama vrednost, ki predstavlja eksponentni faktor svetlosti. Izhod pa je obdelana slika [2].

Slika 5.9: Vozlišče Gama [2].

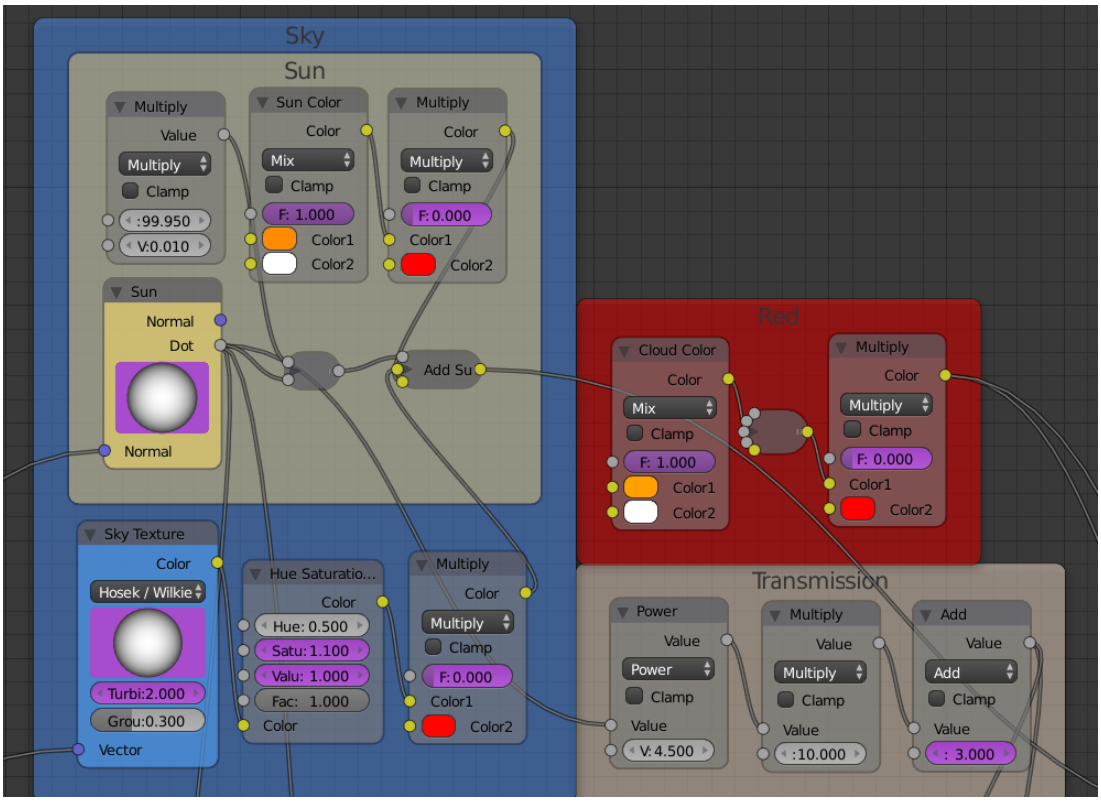
Poglavje 6

Generiranje neba

V vsaki sceni je potrebno dodati svetlobo in v določenih primerih še nebo. V našem primeru smo nebo uvozili. To nebo vsebuje sonce in oblake, ki jim lahko poljubno spreminjamo parametre [25]. Nebo je narejeno s pomočjo urejevalnika vozlišč (node editor), kjer so uporabljeni različni senčilniki, materiali, teksture, itd. Izdelava neba je razdeljena na več področij, ki jih lahko urejamo. Najprej imamo vozlišče za koordinate teksture, ki samodejno generira teksturne koordinate glede na pozicije vektorjev v sceni. Ta se potem razdeli na več podpodročij, ki vsebujejo še dodatna vozlišča, potrebna za generiranje neba.

6.1 Nebo in sonce

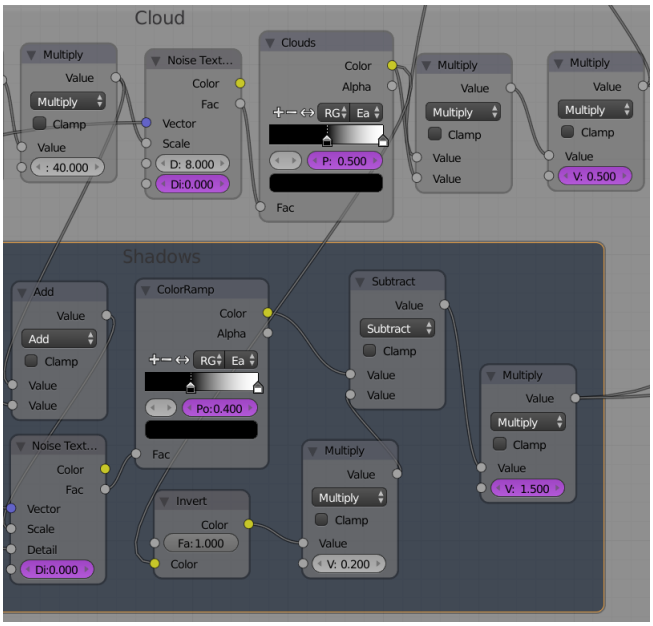
V sceno je postavljeno sonce, ki mu z vozlišči dodamo več lastnosti. Najprej dodamo nekaj barv, ki jih med sabo zmešamo (Mix RGB) in pomnožimo (Multiply) za večplastno svetlobo sonca. Za nebo uporabimo vozlišče za teksturo neba (Sun Texture), ki na sceno doda proceduralno teksturo neba. Lahko mu spreminjamo različne vrednosti, kot so smer sonca ali pa motnost neba in atmosfere (v našem primeru je motnost nizka, saj je nebo jasno). Na koncu uporabimo še vozlišče za odtenek in nasičenost, kjer upravljamo s tem, kako nasičene bodo barve neba, z odtenkom neba in osvetlitvijo scene (Hue Saturation). Dodana je tudi možnost, da je nebo bolj rdečkaste barve. Da lahko dosežemo bolj rdečkasto nebo, uporabimo vozlišče za mešanje (Mix RGB) in množenje barv (Multiply), kamor dodamo rdečo barvo, ki bo vplivala tudi na barvo oblakov.



Slika 6.1: Vozlišča uporabljena za sonce [25].

6.2 Oblaki

S pomočjo različnih matematičnih vozlišč lahko določamo, kako se bodo tvorili oblaki. Določamo lahko njihovo razporeditev, količino in velikost. Glavni del oblakov pa tvorita vozlišče za teksturo šuma (Noise Texture) in barvna lestvica (Color Ramp). Najprej tekstura šuma doda šum na nebo. Ko jo povežemo v barvno lestvico in tej dodamo črno in belo postajo, temnejši del šuma izgine in prikaže nebo pod njim, svetli del šuma pa predstavlja oblake. Z večanjem vrednosti popačenja v teksturi šuma lahko tudi simuliramo veter. Vidnost oblakov stopnjujemo z vozliščem za množenje (Multiply). Če so oblaki kakršne koli druge barve kot bele, je to zaradi svetlobe sonca. Podobno sestavljene so tudi sence oblakov, le da je barva obrnjena (Invert), pomnožena (Multiply) in odšteta (Subtract), tako da predstavlja sence na delih oblakov, kjer jih sonce ne doseže. Za pravilno rotacijo neba je uporabljeno vozlišče za mapiranje teksture (Texture Mapping), ki ga povežemo v teksturo šuma tako, da se oblaki gibljejo enakomerno z nebom.



Slika 6.2: Vozlišča uporabljena za oblake [25].

Cirusi so visoki, tanki oblaki, vlaknatega videza v obliki belih nežnih vlaken, ozkih trakov ali velikih kosmov. Vozlišča so podobna tistim v oblakih, le da tu uporabljamo še valovno teksturiranje (Wave Texture), ki dodaja na sceno proceduralne pasove in črte s pomočjo popačenja šuma. Spreminjanje količine pa nam omogoča barvna lestvica (Color Ramp) z dodajanjem bele barve.

Slika 6.3: Izgled neba [3].

Poglavje 7

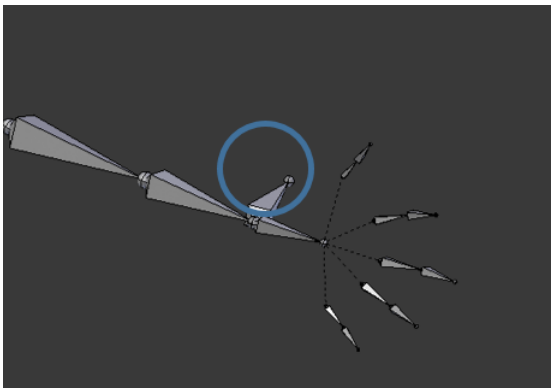
Animacija lika

V tem koraku dodajamo objektom določene kontrole, ki nam bodo omogočile manipulacijo objekta. Poznamo več vrst manipulacije objekta, vendar so za animacijo najbolj pomembne skelet (armature), omejitve (constraints) in ključne oblike (shape keys). Dodajanje skeleta omogoča objektom, da se premikajo v sklepih, zato se pogosto uporablja za skeletno animacijo. Omejitve se uporabljajo za nadzor premikanja, tako da je le to smiselno. Oblikovne tipke pa nadzorujejo določeno obliko modela, kot so npr. izrazi obraza [2].

7.1 Dodajanje skeleta

Če želimo like animirati, moramo lik premakniti in oživeti. Če premikamo le posamezne točke in poligone, bo animacija izgledala zelo nenaravno. Da bo izgled animacije boljši, uporabimo notranji skelet. Lik je torej sestavljen iz poligonske mreže, ki predstavlja kožo lika, in hierarhičnega sklopa med seboj povezanih kosti, ki mu pravimo skelet. Skelet nam torej omogoča, da se naš lik premika. Proces poteka tako, da dodajamo kosti in jih povežemo s kožo. Vsaka kost ima pozicijo, orientacijo in velikost, lahko pa je tudi vezana na drugo kost. S povezovanjem kosti tvorimo hierarhijo starševskih kosti, ki jim sledijo kosti, ki so vezane nanje.

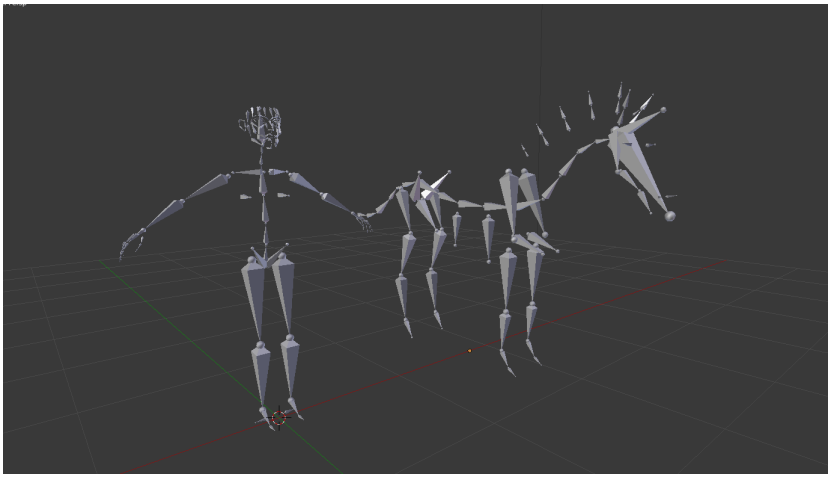
Kosti lahko delimo na dve vrsti in sicer kosti za deformacijo in kontrolne kosti. Prve so kosti, ki bodo ob premiku vplivale tudi na vsa oglišča, ki so povezana z njimi, tako da se bodo ta premaknila v podobni smeri. Deformacijske kosti so torej neposredno vključene v spreminjanje položaja oglišč. Druga vrsta kosti pa deluje podobno kot stikala, ki nadzorujejo druge kosti ali objekte ob premikanju. Te kosti niso neposredno uporabljene za spreminjanje položaja oglišč in po navadi nase nimajo vezanih oglišč [2].



Slika 7.1: Kosti na roki [3].

Kosti na sliki 7.1 so med seboj povezane v verigo. Vse kosti razen označene so deformacijske in bodo vplivale na oglišča. Obkrožena kost pa je kontrolna. Z njo si pomagamo pri premikanju roke, saj vpliva na celotno verigo.

Skeleti po navadi posnemajo prava okostja. Blender omogoča tudi uporabo že pripravljenega okostja človeka in nekaterih živali, ki jih vidmo na sliki 7.2, seveda pa lahko okostje naredimo sami. Kosti so med seboj lahko popolnoma neodvisne, vendar je za najboljše posnemanje realnega premikanja kosti dobro povezati v verige. To pomeni, da v primeru premika noge sledi temu tudi stopalo. Kosti povezujemo iz konice staršev na koren otroka. Določena kost je lahko starš več kostem in je del različnih verig [2].



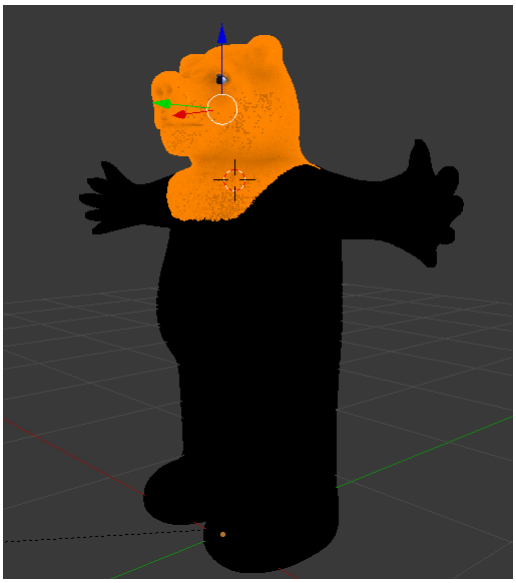
Slika 7.2: Okostje človeka in konja [3].

7.2 Omejitve

Omejitve (constraints) omogočajo nadzor lastnosti predmeta in se pogosto uporabljajo pri animaciji. Omogočajo, da na primer oči sledijo nekemu predmetu, da se kolesa na avtu vrtijo istočasno, pomagajo nogam živali pri animaciji galopa, tako da se samodejno nagibajo v kolenu, itd. Lahko jih uporabljamo na kosteh, tako da določamo razne transformacije ali vplive med njimi, ali pa na objektih, kjer lahko en predmet sledi drugemu. V diplomski nalogi smo jih uporabili za omejitev gibanja kosti v izbrani smeri in določitev vpliva kontrolne kosti [2].

7.3 Skupine oglišč

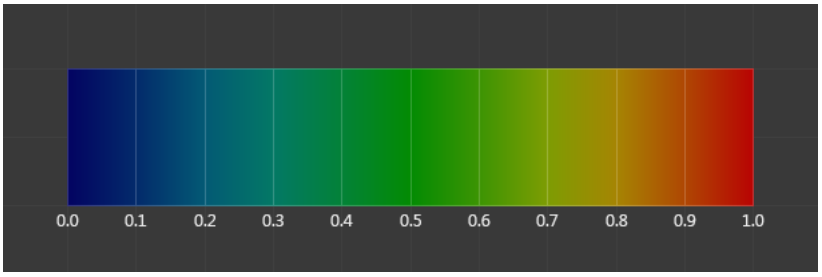
Skupine oglišč (vertex groups) se največkrat uporabljajo za označevanje oglišč, ki pripadajo nekemu objektu. Torej lahko tvorimo skupine oglišč, ki pripadajo npr. roki, nogi, trupu. Velikokrat se uporabljajo pri določanju vpliva okostja, lahko se uporabljajo tudi pri sistemih delcev. V našem primeru smo jih uporabili pri določanju vpliva kosti na model medveda in pri dlaki medveda, kjer smo z njihovo pomočjo določili, katera kost se bo povezala s čim ter kje ima medved dlako in kje ne.



Slika 7.3: Izbrana oglišča za glavo in vrat [3].

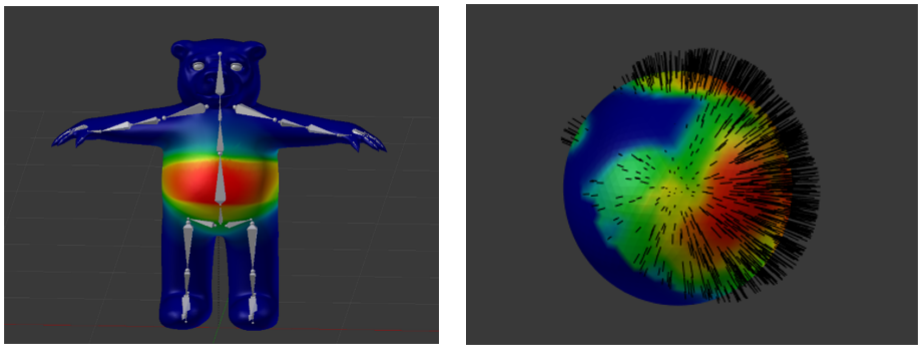
7.4 Označevanje vpliva z barvanjem

Skupine oglišč pa lahko uporabljamo tudi pri barvanju vpliva (weight paint). To je postopek določevanja uteži vsakemu oglišču (ena utež na oglišče) z barvanjem. Skupine oglišč imajo zelo veliko število povezanih točk in s tem veliko število uteži. Barvanje vpliva pa nam omogoča vzdrževanje velikih količin informacij o teži na zelo intuitiven način. Uporablja se predvsem za določanje vpliva določenih kosti na del objekta. Lahko pa se uporablja tudi za nadzor emisij delcev, gostote las, itd. Izbran objekt je prikazan z barvnim spektrom mavrice. Barva vizualizira težo, povezano z vsakim ogliščem v aktivni skupini oglišč. Teže se vizualizirajo z gradientom, ki ima sistem hladnega in toplega barvanja, tako da so območja majhne vrednosti (z utežmi blizu 0,0) pobarvana z modro barvo (hladno) in območja z visoko vrednostjo (z utežmi blizu 1,0) z rdečo barvo (toplo) [2].



Slika 7.4: Vrednosti, sestavljene iz mavričnih barv (modra, zelena, rumena, oranžna, rdeča) [2].

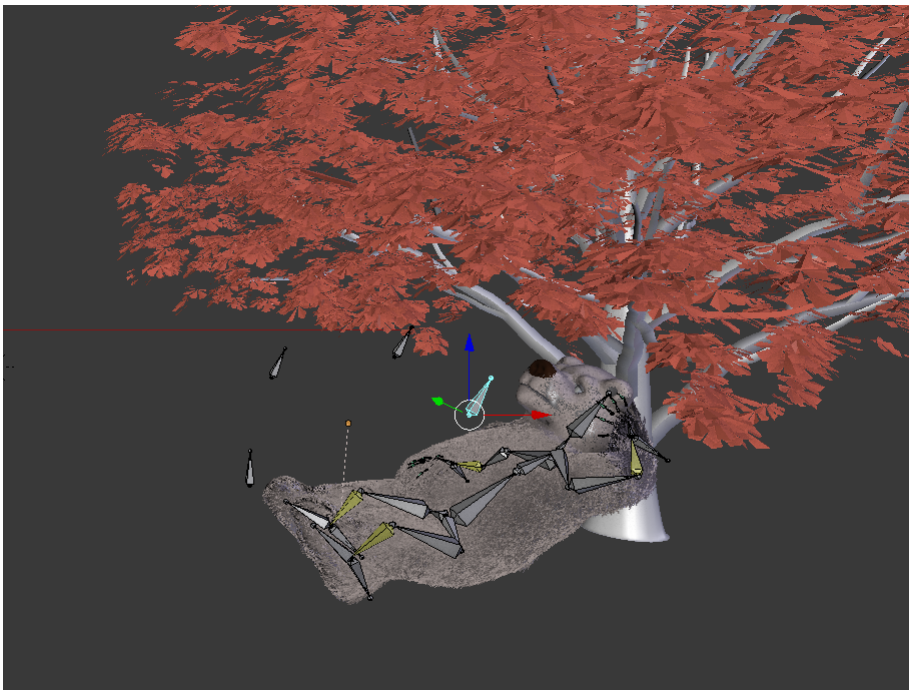
Določanje uteži za kosti je eden od glavnih načinov uporabe barvanja vpliva. Ko se premika kost, se morajo premikati tudi oglišča, tako da posnemajo raztezanje kože okoli sklepa. Obstajajo tudi načini za samodejno dodeljevanje uteži na armaturo, vendar niso zelo natančni. Barvanje vpliva pa lahko uporabimo tudi pri sistemih delcev, kjer to metodo uporabimo za razporeditev delcev po objektu. Število delcev se bo razporedilo na podlagi uteži; tam, kjer bo večja utež, bo več delcev [2].



Slika 7.5: Uteži za kosti in delce [2, 3].

7.5 Dodajanje kože in poziranje lika

Ko je skelet prilagojen liku in so določene uteži, združimo kožo lika s skeletom. Ta korak lahko opravimo tako, da povežemo kožo s skeletom v odnosu otrok/starš, kjer je starš skelet. To pomeni, da bo koža sledila skeletu. Lahko pa uporabimo tudi modifikator skeleta, ki pa je bolj kompleksna metoda. Ko smo zadovoljni z rezultatom, lahko postavimo naš lik v različne poze kot npr. na sliki 7.5. To delamo tako, da premikamo posamezne kosti in jih postavljamo v željene pozicije.



Slika 7.6: Medved v spremenjeni pozi [3].

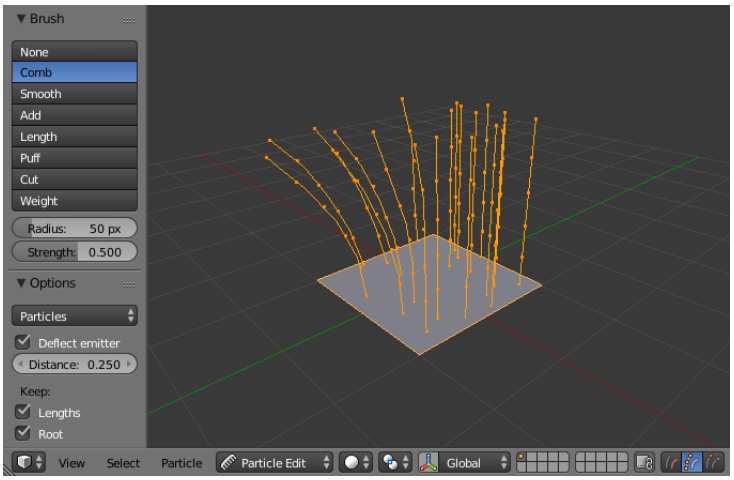
Poglavje 8

Sistem delcev

V diplomski nalogi smo sistem delcev (particles system) uporabili pri generiranju trave, dlaki medveda in razporeditvi kamenčkov in rož. Sistem delcev je tehnika, kjer se iz nekega objekta tvori veliko delcev. Vsak delec je lahko svetlobna točka ali poligonska mreža, ki pa je lahko dinamična ali združena. Odzovejo se lahko na številne različne vplive, sile in imajo celo življenjsko dobo. Na njihovo gibanje lahko vplivajo gibanje poligonske mreže, trki, gibanje glede na gravitacijo ali zračni upor v sceni, itd. Dinamični delci lahko predstavljajo ogenj, dim, meglico, prah, itd. Poznamo tudi delce tipa dlaka, ki po navadi predstavljajo lase, krzno, travo in ščetine. Tak tip delcev se po navadi izriše in nato uporablja kot del modela. Delce lahko manipuliramo tudi v 3D pogledu (česanje, dodajanje, rezanje, premikanje itd.). Vsak predmet lahko nosi številne sisteme delcev in vsak sistem delcev lahko vsebuje do 100.000 delcev [2].

8.1 Oddajnik delcev

Položaj in gibanje delcev nadzorujemo z oddajnikom delcev (emitter), ki deluje kot vir delcev, njegova lokacija v 3D prostoru pa določa, kje bodo delci nastali in kam se bodo premikali. Na sliki 8.1 kot oddajnik delcev deluje ploskev. Kot oddajnik lahko uporabimo 3D predmet, kot je kocka ali ravnina. Nato mu dodamo parametre obnašanja delcev. Ti parametri lahko vključujejo hitrost tvorjenja delcev, začetno hitrost, življenjsko dobo, barve delcev in še veliko več. Vsak od parametrov delcev se ustvari glede na parametre oddajnika. Pri vsaki posodobitvi program preveri življenjsko dobo obstoječih delcev in jih v primeru, da so jo presegli, odstrani iz simulacije. V nasprotnem primeru se položaj delcev in druge značilnosti razvijajo na podlagi fizične simulacije, ki je lahko tako preprosta kot prevajanje njihovega trenutnega položaja ali pa je zapletena kot opravljanje fizikalno natančnih izračunov poti [2].



Slika 8.1: Urejevalnik delcev, kjer je vir delcev ravnina, delci pa so prikazani kot krivulje [2].

8.2 Otroci

Otroci (children) so dodatni delci, ki so vezani na osnovne, matične delce in so od njih odvisni. Fizikalni vplivi se računajo le za osnovne delce, kar pomeni, da lahko brez računanja fizikalnih vplivov spremenimo število in vizualizacijo otrok. Zaradi njih lahko delamo s sorazmerno nizko količino matičnih delcev, kar nam omogoči lažje in hitrejše delo. Otroci se lahko oddajajo iz delcev ali ploskev. Emisija iz ploskev ima nekaj prednosti, ker so delci bolje razporejeni po ploskvi. Otroci iz delcev pa lahko bolje sledijo svojim staršem. Če uporabimo otroke, se starši ne upodabljajo več, ker se lahko oblika otroka precej razlikuje od oblike staršev. Otroci so iz enakega materiala kot njihovi starši in so obarvani glede na položaj oddajanja, zato imajo lahko otroci med seboj različne lastnosti [2].

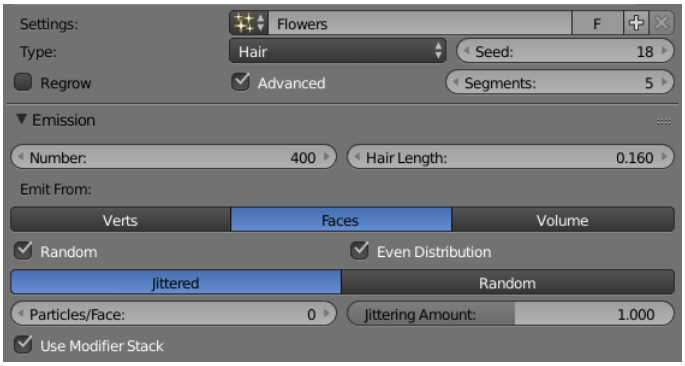
8.3 Dlaka

Sistem delcev za dlako se uporablja za predmete, kot so lasje, krzno, trava, itd. Najprej določimo količino in dolžino dlake. Celotna pot delcev se izračuna vnaprej. Naslednji korak je oblikovanje las, kjer lahko vplivamo na izgled s spreminjanjem fizičnih lastnosti. Naprednejši način spreminjanja videza las pa je uporaba otrok, kar doda dodatne dlake na prvotne in omogoča dodatne nastavitve. Prav tako lahko interaktivno oblikujemo dlake v načinu urejanja delcev [2].

Slika 8.2: Sistema delcev za travo, dlako in razporeditev rož [3].

8.4 Objekti kot sistemi delcev

Tudi objekti se lahko obnašajo kot sistemi delcev. Potrebujemo torej objekt, iz katerega se bodo tvorili delci, in objekt, ki bo predstavljal delce. Ta način lahko uporabimo za razporeditev nekega objekta po drugemu, npr. rože po travniku. Prav tako pa se parametri nastavljajo v oddajniku delcev [2].



Slika 8.3: Rože kot sistem delcev[3].

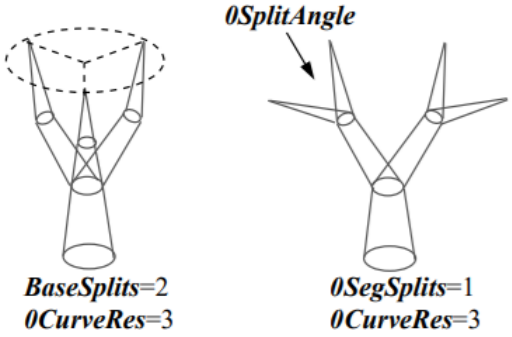
Poglavje 9

Generiranje dreves

Modeliranje realističnih dreves je lahko težavna naloga, še posebej, če je potrebno veliko število podobnih dreves z majhnimi razlikami. V našem primeru smo enostavna devesa naredili sami, bolj kompleksna pa uvozili in jih še dodatno uredili [26]. Za to smo se odločili, ker so bila drevesa že predhono optimizirana in nam je to olajšalo delo. Vsa drevesa pa so bila narejena po isti metodi, ki jo omogoča Blender. Blender ponuja uporabniku vtičnik za generiranje dreves, ki uporablja metodo, ki sta jo razvila Jason Weber in Joseph Penn. Model drevesa, ki sta si ga zamislila, je sestavljen iz debla in listov. Model ima niz parametrov, ki določajo število, verjetnost in kot, glede na katerega se tvorijo veje, ki imajo po navadi podobne lastnosti kot deblo. Model pa sestavljajo tudi veje, imenovane otroci, ki rastejo tako na deblu kot na drugih vejah in imajo popolnoma drugačne lastnosti kot starševske veje in deblo. Kljub drugačnim lastnostim pa večina parametrov otrok temelji na parametrih staršev [27].

9.1 Deblo in veje

Veje so sestavljene iz $nCurveRes$ segmentov in n globino rekurzije. "Lahko določimo tudi vrednost $nCurveBack$, ki bo vplivala na rotacijo veje. Če je vrednost $nCurveBack$ enaka nič, se z-os vsakega dela vrti stran od z-osi prejšnje veje za $(nCurve/nCurveRes)$ stopinj po x-osi. Če vrednost ni enaka nič, se prva polovica veje vrti za $(nCurve/(nCurveRes/2))$ stopinj, druga polovica pa za $(nCurveBack/(nCurveRes/2))$"[27]. Vsaki veji je dodana še posebna rotacija za bolj realističen izgled. Vrednost $0BaseSplit$ definira število vej, ki se tvori iz debla. Nato imamo še vrednost $nSegSprlits$, ki pove, na koliko delov se bo veja delila, $nSplitAngle$ pa pove, kakšni bodo koti med deli veje [27, 28].



Slika 9.1: Generiranje dreves [27].

- $BaseSplit$ nam pove, da se bosta iz debla tvorili dve veji.
- $0CurveRes$ nam pove, da bodo veje deljene na tri dele.
- $0SegSplits$ nam pove, da se bo veja enkrat delila.

Število vej nam določa tudi, koliko otrok se lahko tvori v globini rekurzije n. Povprečno število otrok se izračuna z naslednjo formulo za prvega otroka:

$$veje = veja_{max} * (0{,}2 + 0{,}8 * (\frac{dolžina_{otroka}}{dolžina_{starša}})/dolžina_{otroka,max}))$$

Za vse nadaljnje pa tako:

$$veje = veja_{max} * \left(1{,}0 - 0{,}5 * \frac{odmik_{otroka}}{dolžina_{starša}}\right)$$

Slika 9.2: Formuli za izračun povprečnega števila otrok [27].

Potrebno je izračunati tudi dolžino otrok, kar izračunamo z naslednjo formulo za prvega otroka:

$$dolžina_{otroka} = dolžina_{debla} * dolžina_{otroka,max} * RazmerjeOblike\ (Oblika, \frac{(dolžina_{debla} - odmik_{otroka})}{(dolžina_{debla} - dolžina_{baze})})$$

Za vse nadaljnje pa tako:

$$dolžina_{otroka} = dolžina_{otroka,max} * (dolžina_{starša} - 0{,}6 * odmik_{otroka})$$

Slika 9.3: Formuli za izračun dolžine otrok [27].

Lahko določamo tudi, kakšne oblike bodo veje. Poznamo koničaste, sferične, cilindrične, itd. V Blenderju imamo možnost določati resolucijo drevesa, kako bodo veje med seboj razdeljene, usmerjenost, debelino in dolžino vej, obliko, ki jo bodo tvorile, itd. [27].

9.2 Listi

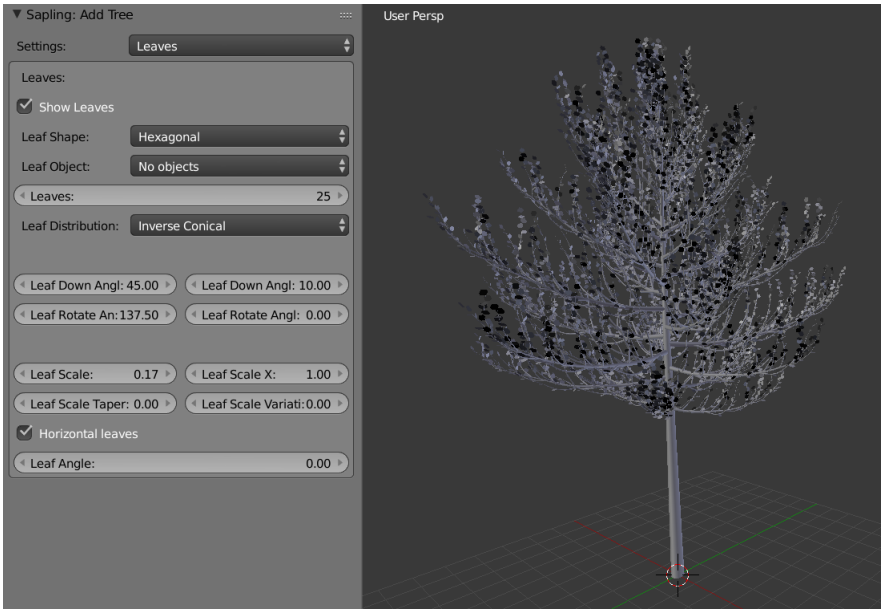
Ko imamo enkrat deblo z vejami, dodamo tudi liste. Najprej z naslednjo formulo izračunamo, koliko listov gre na vejo:

$$Listi_{na\ vejo} = Listi * RazmerjeOblike\left(4(konica), \frac{odmik_{otroka}}{dolžina_{starša}}\right) * kvaliteta$$

Slika 9.4: Formula za število listov na vejo [27].

Kvaliteta je nek faktor, ki ga doda program in je po navadi okoli 1. Nato lahko določamo še obliko listov, npr. v obliki pravokotnika, šestkotnika, lahko pa dodamo tudi svoj model lista, ki ga želimo uporabiti. Na sliki 9.2. lahko npr. vidimo drevo z listi v obliki pravokotnika. Velikost listov pa določimo tako, da najprej izračunamo dolžino listov z $(VelikostListov/\sqrt{kvaliteta})$, za širino po izračunamo $(VelikostListov*VelikostListovX/\sqrt{kvaliteta})$ [27].

V Blenderju lahko izbiramo med več vrst listi, številom listov, distribucijo in ukrivljenostjo, njihovo velikostjo in transformacijo.

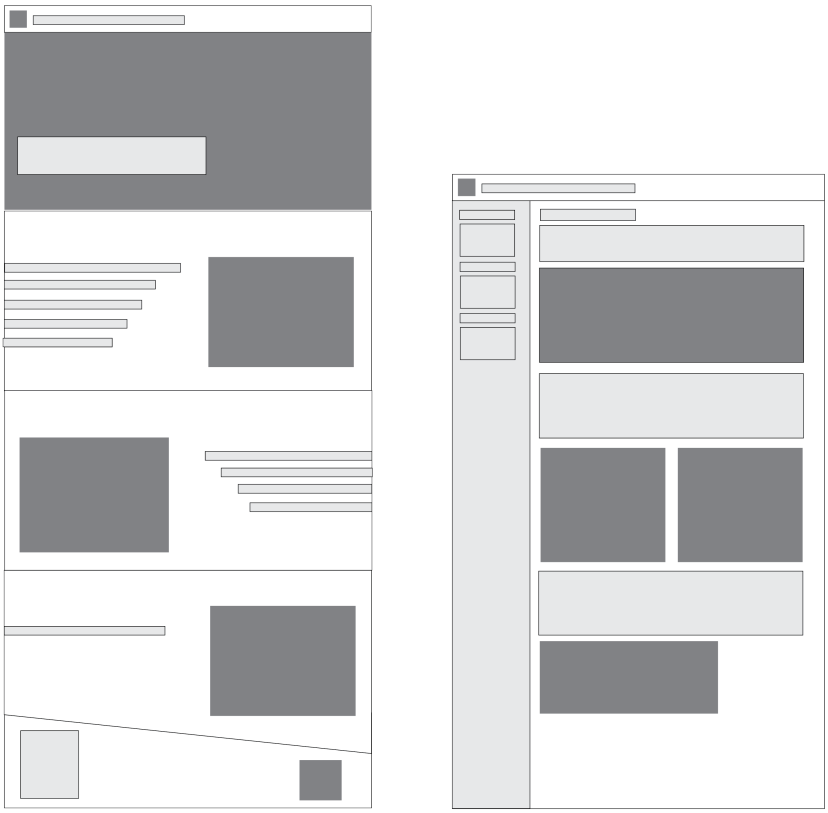


Slika 9.5: Drevo v Blenderju z listi [3].

Poglavje 10

Spletna stran

Postopek izdelave spletne strani se začne pri določitvi zahtev, smernic in ciljev, ki so bili v našem primeru čim boljše vodenje uporabnika skozi postopek izdelave 3D scene. Pri izdelavi spletne strani se moramo vprašati, kdo je naš ciljni uporabnik, kako mu čim bolje podati vsebino in v primeru, da jih je več, sestaviti profile uporabnikov. Nato sledi načrt strani, kjer določimo, kaj vse bo stran vsebovala, čemu bomo posvetili največ pozornosti, itd. Na podlagi tega načrta začnemo nato zbirati ideje za grafično podobo. Ko smo zadovoljni z zbranimi idejami, se lotimo žičnatih okvirjev, ki so tehnika za predstavitev strukture spletne strani. Po navadi so preprosti in črno bele barve, da se jasno vidi postavitev in da jih je lahko spreminjati. Žičnati okvirji, ki smo jih uporabili za našo spletno stran, so predstavljeni na sliki 10.1. Sledi prototipiranje, kar je interaktivna različica žičnatih okvirjev in tudi zadnja faza pred izdelavo spletne strani. S tem lahko prikažemo interakcijo uporabnika s stranjo.



Slika 10.1: Žičnati okvirji za spletno stran [3].

10.1 Oblikovanje

Pri oblikovanju se najprej določi tipografijo, barvno paleto, znake ter logotip. Nato se po navadi oblikuje zaglavje in noga strani. Ko pa je postavljeno približno ogrodje strani, se začne oblikovati posamezne komponente spletne strani.

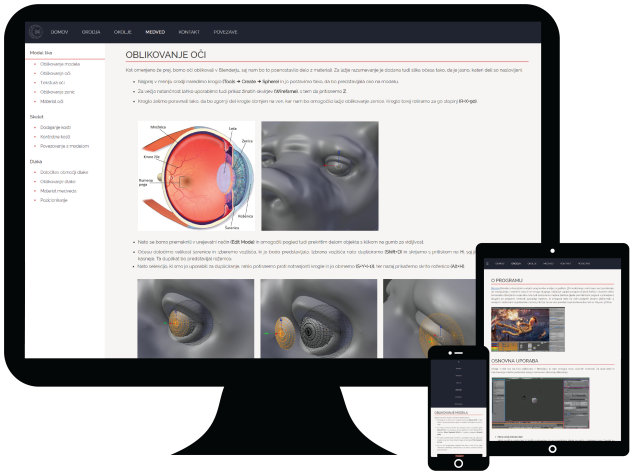
Za spletno stran smo uporabili le eno vrsto pisave in sicer Raleway. Tej pisavi smo nato spreminjali velikost in debelino glede na vlogo besedila. Kot barvno paleto smo uporabili temno modro barvo za meni, rdečo za poudarke, in bež za podlago strani. V okviru spletne strani je bil oblikovan tudi znak, ki predstavlja obris lune, ki uokvirja črko M. Inspiracija za uporabo trikotnikov je prišla iz računalniške grafike, kjer se ti uporabljajo za predstavitve modelov.

Slika 10.2: Tipografija, logo in barvna paleta moje strani [3].

10.2 Vsebina spletne strani

Spletna stran je namenjena začetniku v 3D modeliranju, ki želi bolje spoznati okolja, kot sta Blender in ZBrush. V sklopu spletne strani smo napisali tudi vodiče, ki uporabnika vodijo skozi vse korake izdelave 3D scene. Glavna razlika med pisnim delom diplome in vodiči je, da prvi opisuje teoretsko ozadje 3D modeliranja, medtem ko so vodiči namenjeni praktični uporabi. Vsebina vodičev najprej povzame teorijo in nato z nazorno prikazanimi koraki vodi uporabnika skozi izdelavo. Celoten proces pa je opremljen z veliko slikami, ki omogočajo lažje sledenje procesu.

Spletna stran je sestavljena iz treh večjih delov. V prvem delu so opisana orodja, osnovna uporaba Blenderja in ZBrusha (osnovna postavitev, materiali, čopiči). Drugi del predstavi izdelavo okolja (terena, trave, dreves in podrobnosti). Tretji del pa se osredotoči na medveda (izdelavo lika v Zbrushu, postavitev okostja in izdelavo dlake). Uporabnik me lahko preko spletne strani tudi kontaktira in dostopa do različnih slik, virov in videov pod povezavami. Celotna stran je odzivna glede na uporabljeno napravo, kot lahko vidimo na sliki 10.3. Prilagojena je mobilnim napravam, tabličnim napravam, prenosnikom in namiznim računalnikom različnih resolucij.



Slika 10.3: Spletna stran na različnih napravah [3].

Poglavje 11

Sklepne ugotovitve

V diplomski nalogi smo oblikovali in izdelali 3D sceno in spletno stran, ki uporabnika vodi skozi proces izdelave te 3D scene. Pri 3D sceni smo želeli uporabiti več različnih tehnik in tako zajeti splošen postopek 3D oblikovanja. Sceno in lik smo skušali oblikovati tako, da se lahko uporabita tudi kasneje za animacijo. Pri spletni strani smo dali poudarek preprostosti, obliki in vsebini. Moj cilj je bil, da se uporabnik čim lažje znajde, da je proces izdelave opisan, kar se da razumljivo, in da je prijetna tudi vizualna izkušnja. Tekom izdelave 3D scene smo spoznali pomembnost optimizacije, ki lahko občutno vpliva na hitrost dela. Prav tako smo ugotovili, kako močan vpliv na končno upodobitev ima osvetljava scene, saj so od nje odvisni izgled tekstur, materialov in atmosfera celotne scene. Pri izdelavi spletne strani pa smo spoznali pomen načrtovanja tako oblike kot vsebine, saj z njim prihranimo čas pri programiranju in se izognemo napakam.

Literatura

[1] Blender - Wikipedija, prosta enciklopedija. Dosegljivo: https://sl.wikipedia.org/wiki/Blender, 2018. [Dostopano 19. 6. 2018].

[2] Blender Documentation Team. Blender 2.79 Manual. Dosegljivo: https://docs.blender.org/manual/en/dev/, 2018. [Dostopano 25. 5. 2018].

[3] Maša Planinc. Posnetki zaslona, 2018.

[4] ZBrush - Wikipedija, prosta enciklopedija. Dosegljivo: https://en.wikipedia.org/wiki/ZBrush, 2018. [Dostopano 19. 6. 2018].

[5] Pixologic. Dosegljivo: http://docs.pixologic.com/, 2018. [Dostopano 18. 5. 2018].

[6] Blender 2.77.1 - API documentation. Dosegljivo: https://docs.blender.org/api/blender_python_api_2_77_1/info_quickstart.html, 2018. [Dostopano 1. 6. 2018].

[7] Python (programming language) - Wikipedia, The Free Encyclopedia. Dosegljivo: https://en.wikipedia.org/w/index.php?title=Python_(programming_language)&oldid=846675333, 2018. [Dostopano 1. 6. 2018].

[8] HTML - Wikipedia, The Free Encyclopedia. Dosegljivo: https://en.wikipedia.org/w/index.php?title=HTML&oldid=846079688, 2018. [Dostopano 15. 5. 2018].

[9] CSS - Wikipedija, prosta enciklopedija. Dosegljivo: https://sl.wikipedia.org/wiki/CSS, 2018. [Dostopano 17. 5. 2018].

[10] JavaScript - Wikipedia, The Free Encyclopedia. Dosegljivo: https://en.wikipedia.org/w/index.php?title=JavaScript&oldid=846711364, 2018. [Dostopano 9. 6. 2018].

[11] 3D modeling - Wikipedia, The Free Encyclopedia. Dosegljivo: https://en.wikipedia.org/w/index.php?title=3D_modeling&oldid=842233055, 2018. [Dostopano 20. 6. 2018].

[12] Non-uniform rational B-spline - Wikipedia, The Free Encyclopedia. Dosegljivo: https://en.wikipedia.org/w/index.php?title=Non-uniform_rational_B-spline&oldid=844728185, 2018. [Dostopano 15. 6. 2018].

[13] Matija Marolt. Parametrične predstavitve. Dosegljivo: https://ucilnica.fri.uni-lj.si/pluginfile.php/72610/mod_resource/content/0/32%20Parametri%C4%8Dne%20predstavitve.pdf, 2017. [Dostopano 10. 6. 2018].

[14] Les Piegl and Wayne Tiller. *The NURBS Book*. Springer, 1997.

[15] WulfTheSaxon. NURBS curve being created in NX Shape Studio. Dosegljivo: https://upload.wikimedia.org/wikipedia/commons/8/81/NURBstatic.svg, 2018. [Dostopano 24. 5. 2018].

[16] Extensible 3D (X3D). NURBS component. Dosegljivo: https://www.web3d.org/files/specifications/19775-1/V3.3/Images/NurbsPatchSurface.png, 2016. [Dostopano 26. 5. 2018].

[17] Matija Marolt. Osvetljevanje in senčenje. Dosegljivo: https://ucilnica.fri.uni-lj.si/pluginfile.php/70124/mod_resource/content/0/15%20Osvetljevanje%20in%20sencenje.pdf, 2017. [Dostopano 9. 6. 2018].

[18] Matija Marolt. Globalno osvetljevanje. Dosegljivo: https://ucilnica.fri.uni-lj.si/pluginfile.php/73065/mod_resource/content/0/42%20Globalno%20osvetljevanje.pdf, 2017. [Dostopano 11. 6. 2018].

[19] Shader - Wikipedia, The Free Encyclopedia. Dosegljivo: https://en.wikipedia.org/w/index.php?title=Shader&oldid=845848352, 2018. [Dostopano 24. 5. 2018].

[20] Matija Marolt. Senčilniki. Dosegljivo: https://ucilnica.fri.uni-lj.si/pluginfile.php/71888/mod_resource/content/0/18%20Sen%C4%8Dilniki.pdf, 2017. [Dostopano 8. 6. 2018].

[21] Patricio Gonzalez Vivo and Jen Lowe. The Book of Shaders. Dosegljivo: https://thebookofshaders.com/, 2015. [Dostopano 3. 6. 2018].

[22] Matija Marolt. Teksture. Dosegljivo: https://ucilnica.fri.uni-lj.si/pluginfile.php/71263/mod_resource/content/0/16%20Teksture.pdf, 2017. [Dostopano 11. 6. 2018].

[23] Widhi Muttaqien. Tutorial: Modeling, UV Unwrapping and Texturing a Mushroom. Dosegljivo: https://media.blendernation.com/wp-content/uploads/2017/04/BlenderNation_texturing_mushroom.png, 2018. [Dostopano 19. 6. 2018].

[24] Mapping. Dosegljivo: https://joannelaidlow.wordpress.com/2015/11/02/bump-normal-and-displacement-maps/, 2014. [Dostopano 13. 6. 2018].

[25] Wolf Sky Free. Dosegljivo: https://www.3d-wolf.com/products/sky.html, 2014. [Dostopano 16. 4. 2018].

[26] Archive 3D. Dosegljivo: https://archive3d.net/?a=download&id=b72c55db, 2014. [Dostopano 11. 5. 2018].

[27] Weber Jason and Penn Joseph. Creation and rendering of realistic trees. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 119–128. ACM, 1995.

[28] AiTi Cheng. Creation and rendering of realistic trees. Dosegljivo: `http://plaza.ufl.edu/chengai/file/tree.pdf`, 2018. [Dostopano 4. 7. 2018].