

Kimmo Riihiaho

Punosvarjostin

Tietotekniikan

13. toukokuuta 2018

Jyväskylän yliopisto

Tietotekniikan laitos

Tekijä: Kimmo Riihiaho

Yhteystiedot: kimmo.a.riihiaho@student.jyu.fi

Ohjaaja: Jarno Kansanaho

Työn nimi: Punosvarjostin

Title in English: Straw weave shader

Työ:

Sivumäärä: 11+0

Tiivistelmä: Tiivistelmä on tyypillisesti 5-10 riviä pitkä esitys työn pääkohdista (tausta, tavoite, tulokset, johtopäätökset).

Avainsanat: avain1, avain2, avain3

Abstract: Englanninkielinen versio tiivistelmästä.

Keywords: avainsanat englanniksi

Kuviot

Kuvio 1. Blenderillä mallinnettu huone, jossa on käytetty proseduraalista punostekstuuria.	1
Kuvio 2. Skaalausten ja pyöristyksen vaikutus funktion käyttäytymiseen.	3
Kuvio 3. Alustava luokkakaavio.	6
Kuvio 4. Alustava komponenttikaavio.	6

Sisältö

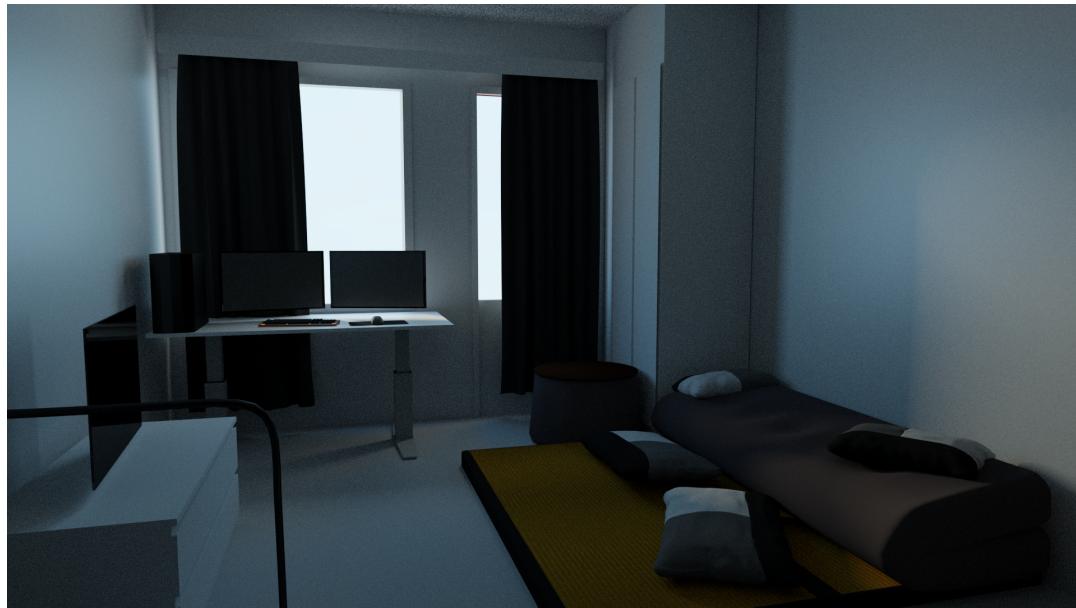
1	IDEA	1
2	MATEMAATTINEN PERUSTA.....	2
3	KÄYTÄNNÖN TOTEUTUS	5
3.1	Verteksivarjostin	5
3.2	Pikselivarjostin	5
4	OHJELMISTO	6
	LÄHTEET	7

1 Idea

Tarkoituksena on luoda varjostinohjelma, jolla voi kuvata punosta. Alunperin ajatuksena oli kuvata olkipunosta, mutta varjostin soveltunee myös esim. kankaan kuvaamiseen. Punoksen kuvaaminen $x - y$ -tasossa on suhteellisen yksinkertaista, mutta jotta sitä voisi käyttää kaareville pinnoille, tarvitaan koordinaattimuunnoksia varten malliavaruuden lisäksi pinnan paikallinen normaali - luulisin.

Tavoitteena olisi saada varjostin toimimaan edes yhteen suuntaan kaaretuvalle pinnalle.

Olen toteuttanut punoksen Blenderin Cycles-renderöijän node-ohjelmoinnilla kuvaamaan tatamin pintaa, joka näkyy kuvassa 1. Käytin samaa tekstuuria suuremmalla skaalaussella myös kuvassa näkyvään patjaan.



Kuvio 1. Blenderillä mallinnettu huone, jossa on käytetty proseduraalista punostekstuuria.

2 Matemaattinen perusta

Yksinkertaisen punosta kuvaavan funktion saa kohtalaisella vaivalla määriteltyä $x - y$ -tasoon. Määritellään aluksi skaalausfunktio f_a x -akselin suuntaan.

$$f_a(x) = \frac{xs_o}{s_a} = a, \quad (2.1)$$

jossa s_o on yleiskaalaus, s_a on loimen suuntainen skaalaus ja x on malliavaruuden x -koordinaatti. Alaindeksillä a tarkoitetaan loimeen (engl. warp) liittyviä asioita. Loimiskaalausen s_a kasvaessa loimilangat siirtyvät kauemmaksi toisistaan ja punos loivenee. Yleiskaalausen s_o (engl. overall) kasvaessa koko punos skaalautuu pienemmäksi.

Vastaavasti määritellään y -akselin suuntaan skaalaava funktio f_e

$$f_e(y) = \frac{ys_o}{s_e} = e, \quad (2.2)$$

jossa s_e on kudelangan (engl. weft) leveys.

Seuraavaksi määritellään funktio f_c kuvaamaan kudelangan pyöreyttä

$$f_c(e) = r|\sin e|, \quad (2.3)$$

jossa kerroin r määritetään pyörityksen voimakkuuden. r :n avulla 0 saadaan tuotettua kulmikas kude.

Funktio f_b kuvailee kanttaaltoa (engl. box), jota käytetään vuorottelemaan vierekkäisiä sinialtoja, jotta ne näyttäisivät erillisiltä kudelangoilta

$$f_b(e) = \pi|\sin e|. \quad (2.4)$$

Käyttämällä funktioita (2.3) ja (2.4) saadaan punoksen yhtälö f_w a :n ja e :n suhteen

$$f_w(a, e) = \sin(a + f_b(e)) + f_c(e),$$

joka laajennetaan x :n ja y :n suhteen syöttämällä sisään yhtälöt (2.1), (2.2), (2.3) ja (2.4)

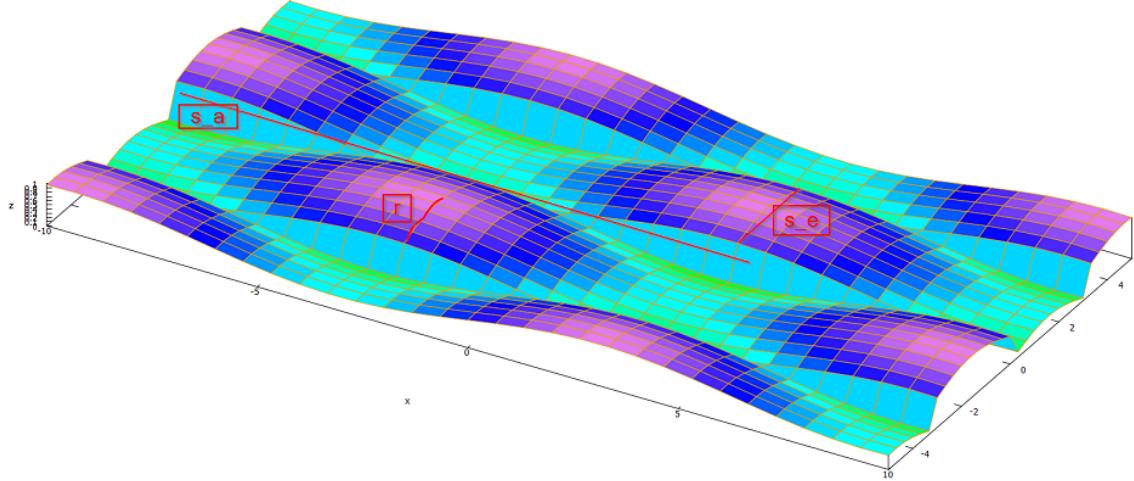
$$f_w(x, y) = \sin\left(\frac{xs_o}{s_a} + \pi \lfloor \sin \frac{ys_o}{s_e} \rfloor\right) + r|\sin \frac{ys_o}{s_e}| : \mathbb{R}^2 \rightarrow \mathbb{R}. \quad (2.5)$$

Yhtälö 2.5 on määritelty koko \mathbb{R}^2 :ssa, mutta itseisarvo- ja lattiafunkitoiden takia se ei ole jatkuva.

Lopuksi normalisoidaan yhtälö min-max -skaalausella (Tommi 2018) välille $[0, 1]$, jotta kun sitä käytetään pinnan normaalina, se ei painu negatiiviseksi.

$$f_{wn} = \frac{f_w - \min(f_w)}{\max(f_w) - \min(f_w)} : \mathbb{R}^2 \rightarrow \mathbb{R}. \quad (2.6)$$

Funktion f_{wn} Maxima-plottaus kuvassa 2 näyttää skaalausten ja pyöristyksen vaikutuskohdat.



Kuva 2. Skaalausten ja pyöristyksen vaikutus funktion käyttäytymiseen.

Jotta funktiota voi käyttää normaalimäppäykseen, pitää laskea funktion normaali pisteessä

(a, e) osittaisderivaatan avulla. Yhtälön 2.5 osittaisderivoointi a :n suhteeseen on yksinkertaista:

$$\frac{\partial f_w(a, e)}{\partial a} = \cos(a + \pi \lfloor \sin e \rfloor) \quad (2.7)$$

Osittaisderivaatta e :n suhteeseen vaatii lattiafunktion derivoinnin, mikä aiheuttaa pieniä hankaluksia, sillä lattiafunktio ei ole jatkuva. Lattiafunktion $\lfloor x \rfloor$ derivaatta ei ole määritelty, kun $x \in \mathbb{N}$, ja kaikkialla muualla se on 0, joten käytännön tarkoitukseissa voimme pitää sitä aina nollana. Itseisarvofunktion $|x|$ derivaatta on $\text{sgn}(x)$, kun $x \neq 0$.

$$\begin{aligned} \frac{\partial f_w(a, e)}{\partial e} &= \cos(a + \pi \lfloor \sin e \rfloor) \overbrace{D[\sin e]}^{=0} + r \cdot \text{sgn}(\sin e) \cos e \\ &= r \cdot \text{sgn}(\sin e) \cos e \end{aligned} \quad (2.8)$$

Lopullinen normaalivektori pisteessä (a, e) on tällöin $(\frac{\partial f_w(a, e)}{\partial a}, \frac{\partial f_w(a, e)}{\partial e}, f_w(a, e))$.

3 Käytännön toteutus

3.1 Verteksivarjostin

Jotta saamme laskettua normaalivektorit muillekin kuin tasasille pinnoille, täytyy käyttää tangenttiavaruutta. Muunnessa mallia varuudesta tangenttiavaruuteen lasketaan verteksivarjostimessa jokaiselle vertekslle erikseen. Tangenttiavaruus käyttää verteksin normaalina z-suuntana, ja vastaavat x- ja y-suunnat (tangentti ja bitangentti) lasketaan ristitulolla niin, että ensin asetetaan päävektori osoittamaan johonkin suuntaan. Mikäli päävektori on saman suuntainen verteksin normaalilin kanssa, päävektori käännetään johonkin muuhun suuntaan.

Jotta siirtymä kolmion reunojen välillä olisi pehmeä, normaali pitäisi interpoloida reunan läheisyydessä, mutta tästä toiminnallisuutta ei toteuttanut. Lisäksi x- ja y-koordinaatit pitäisi jatkaa sulavasti reunan yli.

3.2 Pikselivarjostin

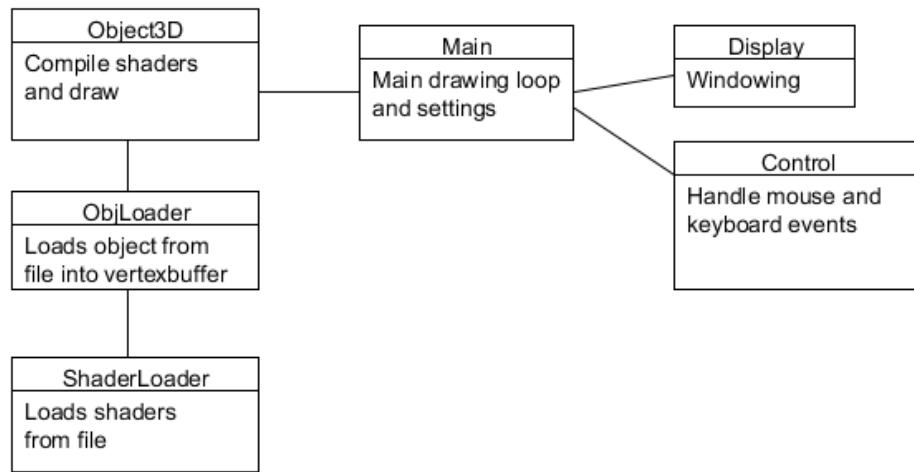
Pikselivarjostin laskee pinnan normaalilin suunnan käyttäen verteksivarjostimen antamia tangenttiavaruuden arvoja.

Jostain syystä a:n suuntainen osittaisderivaatta näyttää paremmalta, kun termi a jäätetään kokonaan pois.

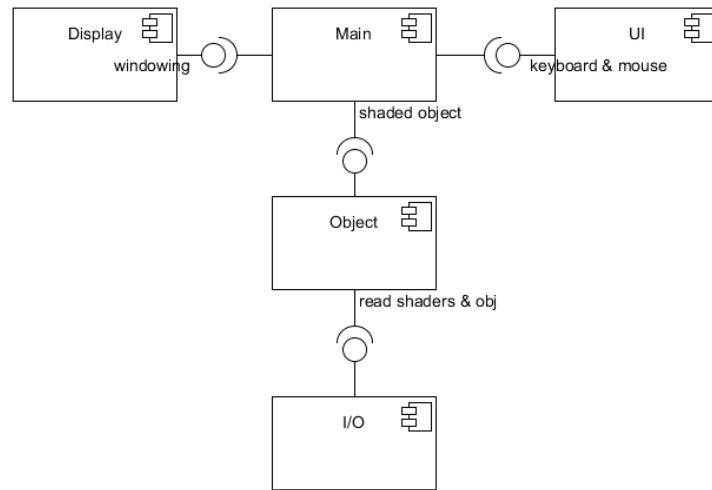
E:n suuntaisen derivaatan vaikutus on hyvin pieni, ja sillä voidaan hyvin käyttää arvoa nolla.

4 Ohjelmisto

Varjostimen testaamista varten kehitetään yksinkertainen kehyssovellus. Kehyssovellus jaa- taan alustavasti luokkiin kuvan 3 mukaisesti. Ohjelman karkea rakenne komponenttitasolla on esitetty kuvassa 4.



Kuvio 3. Alustava luokkakaavio.



Kuvio 4. Alustava komponenttikaavio.

Lähteet

Tommi, Kärkkäinen. 2018. *TIES445 Lecture 5*. Luentodiat.