Verkefni 2 í Tölvugrafík (fiskabúr)

Kristján Leó Guðmundsson, 14. október 2023

(*Ath. Þessi skýrsla er pínu löng, fannst mikilvægt að taka fram flest allt fyrir neðan en lýsing á hjarðhegðuninni er annars á bls. 3.*)

1 Skipulag

Uppsetningin er nokkuð svipuð og í **Verkefni 1**. Við erum með `index.html`, `main.js` og `styles.css` sem sjá um að púsla öllu saman.

Utils mappan inniheldur ýmsar skrár með hjálparföllum og breytum, m.a. skrárnar `MV.js`, `webgl-utils.js` og `initShaders.js` sem voru notaðar í fyrri heimadæmum og verkefnum en síðan var einnig bætt við

1. `utils.js`: Aukaföll sem hjálpa til við útreikninga.
2. `viewControls.js`: Hjálparföll til að hlusta eftir áhorfsstýringum.
3. `sliderControls.js`: Upplýsingar til að stilla stikastýringarnar, meira um þær fyrir neðan.

Objects mappan inniheldur

1. `GameObject.js`: Hlutir nýta klasana `GameObject` og `ObjectPart` sem síðan binda allt saman þegar það kemur að því að teikna og uppfæra, meira um þá fyrir neðan.
2. **BuildingBlocks** mappan inniheldur nokkrar skrár sem hjálpa til við að útbúa grunnhluti.
3. *Hlutaskrár*: Þessar skrár skilgreina hlutina sjálfa. Skránum er raðað eins og tré þar sem dýpsta skráin er lauf. (t.d. inniheldur fiskabúrið krossfiskinn og því er mappan **Starfish** undir **FishTank**).

2 `GameObject` og `ObjectPart` klasarnir

Þessar klasar eru mjög mikilvægir í útfærslunni, þeir binda saman aðra hluti þegar það er teiknað og uppfært stöðuna. Báðir klasarnir virka sem nóður í tré. Hver nóða getur haft börn sem eiga einnig að útfæra annan hvorn af þessum klösum. Þegar kallað er á **draw** eða **update** föllin er síðan farið í gegnum tréð.

Það sem skilur þessa tvo klasa að er að þær nóður sem nýta `GameObject` eru ekki teiknaðar sjálfar, þó svo þær geti skilgreint einhverjar varpanir fyrir sitt undirtré. Þegar `ObjectPart` hlutur er hinsvegar búinn til er nýtt hnútana og index-ana í að búa til buffera með x,y og z gildum hnútanna og normalvigrum þríhyrninganna sem þeir tilheyra (ef teikna á með `gl.TRIANGLES`). Bufferarnir eru síðan nýttir í að teikna hlutina með `drawArrays` en ekki `drawElements` (nema aðferðin sé önnur en `gl.TRIANGLES`).

Til þess að bæta við einhverjum vörpunum getur hlutur útfært föllin

- `__createInstanceMatrix`: varpanir sem verða bara gerðar á þessri nóðu. Hér þarf þá að stilla `this.instanceMatrix` breytuna með vörpuninni. (Hefði kannski átt að kalla þetta eitthvað annað en instance).
- `__createSubtreeMatrix`: varpanir sem verða gerðar á þessri nóðu og haldast í undirtré. Hér þarf þá að stilla `this.subtreeMatrix` breytuna með vörpuninni.
- `__createInstanceNormalMatrix`: varpanir á normal vigrinum (þarf aðeins að skilgreina ef verið er að skala mis mikið í víddirnar x, y, z).

Hér þarf þá að stilla `this.instanceNormalMatrix` breytuna með vörpuninni.

- `__createSubtreeNormalMatrix` varpanir á normal vigrinum (þarf aðeins að skilgreina ef verið er að skala mis mikið í víddirnar x, y, z). Hér þarf þá að stilla `this.subtreeNormalMatrix` breytuna með vörpuninni.

Til þess að sleppa því að teikna nóðu og undirtré er hægt að stilla

```
this.hidden = true
```

Til að bæta við einhverjum breytingum getur hlutur útfært fallið

- `__updateSelf`: fallið á að taka við breytunni `dt` sem segir til um tímabreytinguna frá síðasta `render` kalli í millisekúndum.

Til þess að koma í veg fyrir að hlutur sé uppfærður er hægt að stilla

```
this.freeze = true
```

Ef það á að bæta við barni/undirnóðu er hægt að nota `addChild` fallið.

3 Hjarðhegðun

Fiskarnir í búrinu sýna hjarðhegðun. Þeir reyna bæði að halda sér saman (c), ferðast í svipaða stefnu (a) og forðast að koma 'of' nálægt hver öðrum (s). Einnig reyna fiskarnir að forðast hákarlinn (h) (meira um hann fyrir neðan) og endamörk búrsins/sandinn (e) (einnig meira um hann fyrir neðan).

Þegar verið er að reikna stefnubreytinguna (Δd) eru þessir þættir reiknaðir hver fyrir sig, breytingin sköluð niður í einingarvigur og síðan er tekið vigtað meðaltal. Hægt er að stilla vigtirnar í gegnum stikana á skjánum.

Til þess að koma í veg fyrir of snöggar breytingar á hraðanum eru þær skalaðar niður með `smoothness` stikanum. Í heildina verður breytingin því

$$\Delta d = \frac{w_c \cdot c + w_a \cdot a + w_s \cdot s + w_h \cdot h + w_e \cdot e}{\texttt{smoothness} \cdot \sum w}$$

c er reiknað sem fjarlægð frá meðalstaðsetningu allra fiska, skalað niður í einingarvigur.

a er reiknað sem meðalstefna allra fiska, skalað niður í einingarvigur.

s er reiknað sem summa allra einingarvigra í áttina að fiskinum frá öðrum fiskum innan skilgreinds radíus r. Skalað niður í einingarvigur.

h er annað hvort einingarvigur í áttina frá hákarlinum eða $(0, 0, 0)$ eftir því hvort hákarlinn sé mögulega að elta fiskinn eða ekki.

e er einnig reiknað sem vigtuð summa. Hver hlið i fær vigtina $1/ln(d_i + 1)$ þar sem d_i er fjarlægðin frá hliðinni (logrinn var aðeins til að gefa hliðunum ekki of mikið vægi á móti hver annarri). Einnig skalað niður í einingarvigur.

Passað er uppá að allir fiskarnir taki mið af núverandi stefnu fiskanna í kringum sig með því að uppfæra síðan ekki stefnuna sjálfa fyrr en í `__updateSelf` fallinu.

4 Stefnuvigur í vörpun

Til þess að beina fiski í rétta átt er reiknuð viðeigandi snúningsvörpun út frá stefnuvigrinum $d = (d_x, d_y, d_z)$.

Við byrjum á að beina fiskinum í rétta stefnu í xz planinu með því að snúa um y-ás með

$$\theta_{xz} \text{ (hornið frá } x\text{-ás í } xz \text{ planinu)}$$

Það er hægt að reikna hornið með því að taka $\texttt{arctan}\left(\frac{d_z}{d_x}\right)$. Það þarf samt að passa að bæta við 180 gráðum ef $x < 0$ þar sem `arctan` gefur gildi á milli -90 og 90.

Nú stefnir fiskurinn í sömu átt og vigurinn $(d_x, 0, d_z)$. Til þess að snúa síðan frá $(d_x, 0, d_z)$ í (d_x, d_y, d_z) getum við notað hornið θ_y milli vigranna og snúið um ás sem er hornréttur á þá báða og liggur í gegnum $(0, 0, 0)$.

Hornið milli vigranna má reikna á svipaðan hátt með $\texttt{arctan}\left(\frac{d_y}{\sqrt{d_x^2+d_z^2}}\right)$ (þar sem lengd ofanvarps d á xz-planið er $\sqrt{d_x^2 + d_z^2}$). Hér ætti ekki að bæta 180 gráðum við útkomuna ef $x < 0$ þar sem hornið verður hvort eð er á bilinu -90 og 90 og við viljum halda því þannig.

Ásinn sem liggur í gegnum $(0, 0, 0)$ og er hornréttur á þá báða hefur stefnuvigurinn $(-d_z, 0, d_x)$ sem liggur einnig í xz-planinu (ætti að stefna til hægri þar sem við snúum síðan skv. hægri handarreglunni fyrir snúning).

Áður en það er snúið um ásinn ákvað ég síðan að margfalda θ_y með 0.75 þar sem fiskar horfa sjaldan beint upp eða niður.

Þegar búið er að snúa um $(-d_z, 0, d_x)$ með ætti fiskurinn að snúa í sömu átt og stefnuvigurinn. Þetta verður nokkuð svipað og *roll, pitch, yaw* nema *roll* verður eftir (væri kannski flott samt að snúa fiskinum 'örlítið' um stefnuvigurinn eftir því hver stefnubreytingin er).

5 Stikar

Á skjánum má sjá ýmsa stika og tvo takka. Hérna er smá lýsing á þeim:

- **Cohesion** stillir vigtina w_c fyrir samloðun
- **Alignment** stillir vigtina w_a fyrir uppröðun
- **Seperation** stillir vigtina w_s fyrir aðskilnað
- **Smoothness** Hversu hratt breytist stefnan
- **Edge avoidance** stillir vigtina w_e
- **Shark avoidance** stillir vigtina w_h
- **Light** stillir styrk ljóssins efst í fiskabúrinu
- **Daylight** stillir dagsbirtuna
- **Air bubbles** stillir fjölda sjáanlegra loftbóla
- **Water level** stillir hæð vatnsins í fiskabúrinu
- **Sand** stillir hæð sandsins í fiskabúrinu
- **Fish showing** stillir fjölda fiska sem sjást
- **Lid** takki sem opnar og lokar fiskabúrinu
- **Freeze** takki sem stöðvar eða kveikir aftur á breytingum hluta innan fiskabúrsins

Ath.

- Ef einhverjir stikar eða takkar sjást ekki gæti þurft að skrolla niður í stikaglugganum
- Ef **Edge avoidance** er hátt getur það virkað pínu eins og **Cohesion** líka þar sem fiskarnir leita meira í miðjuna.
- Þó **Shark avoidance** sé 0 fá fiskarnir samt meiri hraða ef hákarlinn er nógu nálægt þeim og gætu þeir því litið út fyrir að vera að flýja.

Það er hægt að finna fullt af skemmtilegum gildum en hérna eru tvær áhugaverðar stillingar til að byrja með:

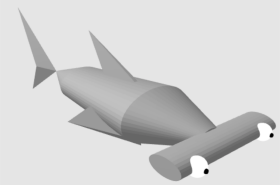
1. Fín þúfa:
    - Cohesion: 50
    - Alignment: 40
    - Seperation: 65
    - Seperation radius: 0.25
    - Smoothness: 7
    - Edge avoidance: 30
    - Shark avoidance: 0
    - Fish showing: 30
2. Fiskar í miðjunni og hákarl skiptir upp hópnum af og til:
    - Cohesion: 25
    - Alignment: 15
    - Seperation: 60
    - Seperation radius: 0.5
    - Smoothness: 7
    - Edge avoidance: 75
    - Shark avoidance 50
    - Fish showing: 30

6 Útfært aukalega

Einhverju hefur verið bætt við, hérna er það helsta:

6.1 Hlutir

6.1.1 Hamarhákarl

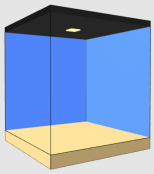


Hákarlinn er teiknaður með ýmsum formum, m.a. sívalningum fyrir búkinn og kúlum fyrir augun og augasteina. Hann eltir bæði fiskana og forðast einnig endamörk búrsins og sandinn. Hákarlinn byrjar að elta fisk ef hann er kominn nær en skilgreindur radíus. Þá eykst hraðinn á honum sjálfum og einnig hornhraðinn á sporðinum.

6.1.2 Fiskur

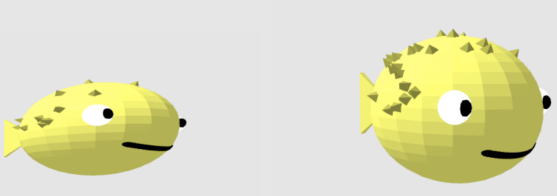
Fiskarnir eru einnig teiknaðir í þrívídd með hjálp ýmissa forma. Þegar hákarlinn byrjar að elta þá fá þeir einnig meiri hraða.

6.1.3 Fiskabúr



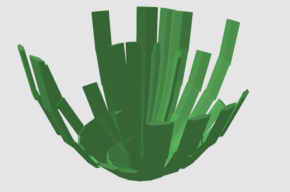
Búið er að bæta við loki með ljósi (meira um það fyrir neðan) á fiskabúrið ásamt sandi á botninn.

6.1.4 Kúlufiskur



Í fiskabúrinu má sjá kúlufisk. Kúlufiskurinn ferðast út um allt í fiskabúrinu og belgist út á slembnum tímapunktum.

6.1.5 Þari

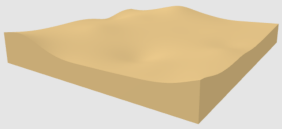


Á botni fiskabúrsins er þari. Hann er teiknaður með sívalningum.

6.1.6 Krossfiskur

Á botni fiskabúrsins er einnig einn krossfiskur.

6.1.7 Sandur



Sandurinn hefur ójafna sveiflukennda hæð. Til þess að ná því fram var skilgreint nokkra stýripunkta (c) á ákveðnum bilum í xz planinu með slembið y gildi. Þegar reiknað er hæð hnútanna sem mynda efsta lag sandsins er tekið mið af fjarlægðinni frá þessum stýripunktum í xz og vigtað milli hæðanna þeirra.

Þegar við færumst nær stýripunkti í xz planinu ættu áhrif hans á hæðina að aukast (ef við pælum ekki í hinum stýripunktunum á meðan). Þ.e. vigtin eykst í öfugu hlutfalli við fjarlægðina ($w_i \propto 1/d$).

Til þess að ná fram flottum kúrvum setti ég fjarlægðina í annað veldi

$$w_i \propto d^{-2}$$

Hæð punkts p verður því

$$p_y = \frac{\sum c_{iy} \cdot d_i^{-2}}{\sum d_i^{-2}} \quad \text{þar sem} \quad d_i = \sqrt{(c_{ix} - p_x)^2 + (c_{iz} - p_z)^2}$$

6.2 Annað

6.2.1 Ljós í fiskabúrinu

Efst í fiskabúrinu er punktljósgjafi sem er hluti af einhverskonar útfærslu af endurskinslíkani Phongs. Ljósstyrkurinn er reiknaður samkvæmt umhverfis-, dreifi- og depil-endurskins formúlunni nema útreikningurinn er gerður á hnútaleveli í hnútalitaranum og síðan er liturinn brúaður yfir á bútana.

Til þess að geta reiknað út litunina var bætt við nokkrum uniform breytum en einnig þurfti eiginleika sem segir til um normal vigur hnútanna. Normal vigurinn er reiknaður fyrir hvern þríhyrning þegar hlutur (sem er teiknaður með `gl.TRIANGLES`) er búinn til. Honum er síðan varpað í hnútalitaranum.

Þar sem að skölun sem hefur ekki sömu x, y, z gildi breytir normal vigrinum á öfugan hátt var bætt við auka fylki í útfærslunni sem varpar honum sérstaklega.

Vefsíða

Hérna er hlekkur á vefsíðuna:
https://tolvugrafik-test.netlify.app/v2/fiskabur/we/v2wproperreflection1/

Þar sem canvas-ið er stækkað upp í 100% getur það tekið soldið lengri tíma að teikna á stærri skjám. Þetta virðist virka fínt hjá mér en ef síðan er að lagga gæti verið að þessi gerð virki betur
https://tolvugrafik-test.netlify.app/v2/fiskabur/we/v2wproperreflection2/
en í þessari gerð byrja gæðin aðeins lægra niðri og þau eru hækkuð ef það gengur vel að rendera.

Betra að nota samt fyrri gerðina ef hún virkar vel.