Säikeistetty grafiikkamoottoriprojekti (ThreadedEngine)

Sakari Tanskanen
Miika Lehtimäki
Antti Hatanmaa

Git-repositorio

https://github.com/EeroHeikkinen/ThreadedEngine

SVN-repositorio

https://goblin.tkk.fi/svn13/projects/ownsubject8

1 Käyttö- ja kääntöohjeet

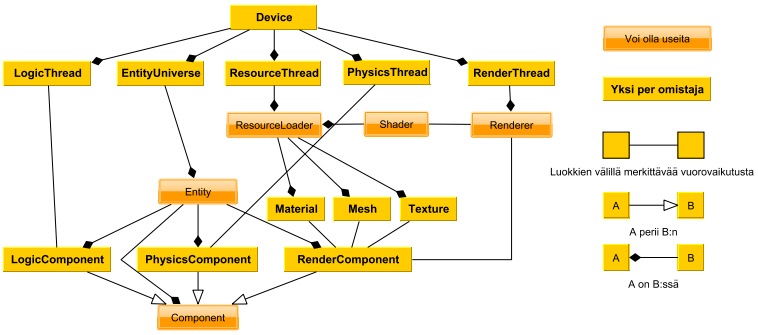
Ohjelma toimii ainakin Windowsilla ja kääntyy g++ 4.8.2:lla. Projektissa on käytössä glm, BulletPhysics, SFML, GLEW ja ThreadingBuildingBlocks -kirjastot. Ohjelman ajaminen vaatii vähintään OpenGL3.0:n. Shaders- ja res kansiot on oltava samassa kansiossa käännetyn ohjelman kanssa. Ajettaessa ohjelma näyttää demon, eikä mitään kontrolleja ole joten käyttöohjeita ei sen kummemin tarvita.

1.1 Bugit

Ohjelman toimivuudesta linuxilla ei ole tietoa. Ohjelma kääntyy, mutta testikone tukee vain OpenGL2.1:tä tai vanhempaa, joten emme pystyneet testaamaan. Lisäksi tällä hetkellä ohjelma segfaulttaa ajettaessa, tämä pyritään korjaamaan demoon mennessä...

2 Rakenne

Ohjelman rakenne on pääosiltaan samanlainen kuin suunnitelmassa. Ohjelma koostuu suunnitellun viiden ylätason komponentin sijaan neljästä komponentista, jotka toimivat omissa säikeissään. Viides säie on ohjelman käynnistyksen yhteydessä luotava pääsäie. EventThreadista on luovuttu eikä eventtejä ole toteutettu. Ne on suunniteltu toteutettavaksi pääsäikeeseen.



2.1 Säikeet

Säikeet ovat:

1. Device eli pääsäie
   - Käynnistää muut säikeet ja sisältää EntityUniversen eli entiteetticontainerin.
   - Muut säikeet kommunikoivat keskenään Devicen avulla, eli saavat Deviceltä referenssejä toisiin säikeisiin.
2. ResourceThread
   - Sisältää useampia erilaisia resurssinlataajia, jotka lataavat erilaisia resursseja (esim. eri formaatti).
   - Resurssinlataajat palauttavat referenssejä pyydettyyn resurssiin ja huolehtivat resurssien lataamisesta, poistamisesta sekä säilyttämisestä tarpeen mukaan, eli resurssinlataaja omistaa lataamansa resurssit ja pitää kirjaa niiden tarpeesta.
3. RenderThread
   - Sisältää rendererejä, joiden render-funktioita säie kutsuu. Yksittäinen renderer sisältää referenssit kaikkiin renderöintikomponentteihin joita se renderöi. Referenssejä säilytetään tilanteen mukaan erilaisissa tietorakenteissa, esim. lista tai octtree.
   - Mahdollistaa erilaisen käsittelyn erityyppisille komponenteille.
4. PhysicsThread
   - Luo ja ylläpitää BulletPhysicsin simulaatiomaailmaa ja ajaa Bulletin maailman simulaatiota, joka päivittää fysiikkakomponenttien translaatiomatriiseja. Säiltyttää referenssejä fysiikkakomponentteihin.
5. LogicThread
   - Säilyttää referenssejä logiikkakomponentteihin ja käy niitä läpi kutsuen logic-funktiota jokaiselle komponentille.
   - Logiikka voi sisältää oikeastaan mitä vain, vapaimmin käyttäjän käytössä oleva säie.

2.2 Entiteetit

Entiteetit koostuvat komponenteista sisältämällä pointtereita komponentteihin. Komponentteja on kolmea päätyyppiä: renderöintikomponentit, logiikkakomponentit ja fysiikkakomponentit. Näistä logiikka- ja fysiikkakomponentteja ohjelman säikeet käyttävät suoraan ja renderöintikomponentteja renderöien välityksellä. Tämän lisäksi käyttäjä pystyy luomaan minkälaisia komponentteja tahansa perimällä Component-luokan. Näitä komponentteja ei käytetä suoraan, mutta käyttäjä voi toteuttaa pääkomponentteja jotka hyödyntävät muita komponentteja. Esimerkkinä tästä kamerakomponentti pelaajalla, jota pelaajan renderöintikomponentti hyödyntää.

1. Renderöintikomponentti
    - Sisältää viittaukset renderöintiin tarvittavaan dataan, kuten käytettävään renderöijään, meshiin, tekstuureihin ja materiaaliin.
2. Fysiikkakomponentti
    - Sisältää kappaleen fysikaaliset ominaisuudet, kuten massan ja kimmoisuuden sekä sekä kappaleeseen vaikuttavat voimat. Tämä data annetaan Bulletille, joka laskee fysiikkasimulaatiota.
3. Logiikkakomponentti
    - Sisältää logic-funktion jota logiikkasäie suorittaa. Pääkomponenteista vapaimmin käyttäjän toteutettavissa oleva komponentti.

3 Lopullinen työnjako

Aikataulutus muuttui alkuperäisestä melko paljon, sillä Eero joutui lopettamaan kurssin kesken suunnitelman tekemisen jälkeen. Kolmen jäljellejääneen kesken kommunikaatio sujui pääosin kuitenkin hyvin ja työnjako oli selvä.

Antti:
Toteutti säikeistystä, refaktorointia, poikkeusturvallisuutta sekä osallistui paljon kaikkiin projektin osa-alueisiin.

Sakari:
Toteutti fysiikkasäikeen, tutustui bullettiin, teki kaikenlaista pientä sekä teki dokumentin.

Miika:
Toteutti renderöintisäikeen sekä resurssienlataajan.

4 Lähteet

BulletPhysicsin wiki `http://bulletphysics.org/mediawiki-1.5.8/index.php/Main_Page`
C++ reference `http://www.cplusplus.com/reference/`