VIIKKORAPORTTI 3

Taas yksi kehitysviikko takana, töitä on riittänyt. Koodia on tullut kirjoitettua melko paljon ja testaamista on riittänyt. Eniten aikaa on kuitenkin ehkä vienyt 3D-grafiikan teorian opiskelu ja sen tarkistaminen kynällä ja paperilla laskiessa. Tässä muutamia keskeisiä kohtia viikolta.

Ohjelman testaaminen osoittautui todella vaikeaksi. Alkuperäinen idea oli käyttää NetBeansin CUnit-lisäkirjastoa testien kirjoittamiseen ja ajamiseen. CUnit-testit toimivat samalla tavalla kuin JUnit-testit javalla, ne ovat vain tarkoitettu testaamaan C:llä kirjoitettuja projekteja. Tämä ei kuitenkaan käynyt päinsä, vaan heti ensimmäisiä testiluokkia kirjoittaessani NetBeans sekosi täysin, eikä suostunut enää edes kääntämään koko ohjelmaa. Myöskään normaaleja Simple C-testejä en saanut toimimaan. Pienen tutkimisen jälkeen huomasin, että kyseessä on NetBeansin tunnettu bugi, jossa testien lisääminen aiheuttaa huonon makefilen luomisen, eikä ohjelma enää sen takia käänny. En lähtenyt selvittämään asiaa tämän enempää, vaan loin projektin uudestaan ja kirjoitin käsin muutaman pienen testiluokan. Aion myöhemmin kirjoittaa paremmat testit, kunhan saan piirtämisen ensin valmiiksi.

NetBeans on C:n kehittämisen kannalta todella kömpelö. Koodia kirjoittaessa törmää usein outoihin bugeihin, kirjastojen lisääminen on vaikeaa ja epäintuitiivista ja kun ne vihdoin saa asennettua, rikkovat ne koko projektin. Vaikka toisaalta ohjelman debugaaminen on helppoa, sillä NetBeansissä on hyvät koodin jäljitystoiminnot, kuluu aikaa ehkä kuitenkin enemmän vaan NetBeansin virittämiseen kuin bugien korjaamiseen. Tämän jälkeen en kyllä enää ikinä kirjoita riviäkään C:tä NetBeansilla. Pärjäisiköhän Vim:illä ja Valgrindilla paremmin?

3D-grafiikan teoria on melkeinpä pelkkää geometriaa ja lineaarialgebraa. Tällä hetkellä teoriapuoli on melkeinpä kokonaan hallussa. Grafiikan piirtämisen kulku on seuraavanlainen: Aluksi määritellään vain jokin malli määrittelemällä verteksien paikat suhteessa origoon. Tätä kutsutaan malliavaruudeksi (Object space). Tämän jälkeen malli skaalataan, käännetään haluttuun asentoon ja siirretään paikalleen globalissa avaruudessa (World space), jossa kaikki mallit sijaitsevat. Globaalin avaruuden origo on piste (0,0,0) ja sen kanta on standardikanta, eli normaalit yksikkövektorit. Tämän jälkeen valitaan jokin olio ”kameraksi”, jonka suhteen asioita kuvataan. Seuraavaksi muunnamme kappaleet kuva-avaruuteen (View space), jossa origoksi valitaan kameran koordinaatit globaalissa avaruudessa ja kannaksi ”kameran” malliavaruuden kanta siten, että z-suunta on suunta, johon kamera osoittaa, y-suunta osoittaa ylös ja x-suunta oikealle. Nyt meillä onkin jo tiedossa z-koordinaatin avulla kappaleiden etäisyys kamerasta, joten voimme piirtää jo yksinkertaisen kuvan. Viimeinen vaihe muunnoksissa on perspektiivimuunnos, jonka avulla tuotetaan valmis kuva, joka muistuttaa oikeata maailmaa.

Kaikki nämä laskutoimitukset ovat käytännössä koordinaattivektorin kertomista tietynlaisilla matriiseilla, sillä 3x3 matriisit ovat oikeastaan lineaarikuvauksia avaruudelta itselleen. Kuvaukset avaruuksista toisiin tehdään vektoriavaruuden kannanvaihtojen avulla, jotka ovat siis juuri näitä 3x3 matriiseja. Tämän osan teoriasta ymmärrän hyvin, mutta käytäntö on kuitenkin hieman monimutkaisempi. Oikeasti kolmiulotteiset kappaleet esitetään neliulotteisina *homogeenisina koordinaatteina*, jotka ovat projektioita neliulotteisesta avaruudesta kolmiulotteiseen avaruuteen. Tämä teoria ei auennut aivan täysin itselleni, mutta sen hyödyt ovat lukuisat: tämän ansiosta kaikki muunnokset ja projektiot voidaan esittää matriisien kertolaskuina ja jotkin muunnokset (kuten esim. translaatio ja rotaatio) voidaan jopa sisällyttää

samaan matriisiin. Lisäksi tällaiset laskutoimitukset ovat ilmeisesti myös helpommin mallinnettavissa grafiikkaprosessorilla, joten se tuo myös tehokkuusetuja.

Ensi viikolla kirjoitan piirtämisen valmiiksi. Oikeastaan ainoa, mikä tällä hetkellä puuttuu on perspektiivimuunnos. Tämän jälkeen kirjoitan kuva-avaruuteen tarvittavan muunnoksen, jotta voin liikutella kameraa. Koodia tulisi hieman siistiä ja ohjelmassa on ainakin yksi muistivuoto, joita en ole vielä löytänyt (vaikka koodia on suhteellisen vähän vielä ☺).