AngryPigs

Dokumentacija

1. junij 2011

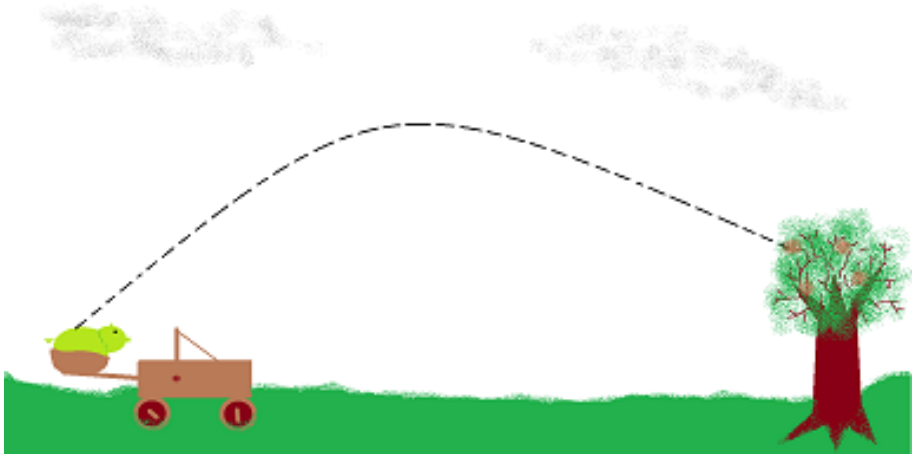
Avtorja

Matic Potočnik, Jaka Sivec

Poglavje 1

O igri

AngryPigs je zabavna 3D igra, v kateri igralec katapultira prašiče v drevesa, ter skuša z njih sklatiti ptice, ter njihova gnezda. Igra je bila ustvarjena v sklopu seminarske naloge za predmet Računalniška grafika na univerzitetnem programu Fakultete za računalništvo in informatiko v Ljubljani. Repozitorij z izvorno kodo je na voljo na: `https://github.com/HairyFotr/AngryPigs`



Poglavje 2

Tehnologija

2.1 Uvod

Igra je napisana v Scali, z izjemo dela, ki generira drevesa – ta je bil napisan v jeziku Clojure. Scala je razmeroma nov jezik, ki teče na JVM(Java Virtual Machine) in hkrati podpira objektno ter funkcijsko programiranje. Clojure je moderen dialekt Lispa, ki tudi teče na JVM. Oba sva bila pred začetkom projekta skoraj čista začetnika v obeh jezikih.

Za grafiko sva uporabila knjižnico LWJGL. Prevedla in priredila sva nekatere primere iz vaj. Igra ne uporablja naprednih tehnik in funkcionalnosti OpenGL-a, predvsem zato, ker ima Matic precej počasen prenosnik in za nameček na njem poganja Linux, ki ne podpira njegove vgrajene grafične kartice. Grafika mu teče v programsko-pospešenem načinu. (To ni le šala, ampak precej velik faktor pri nekaterih odločitvah)

2.2 Prevajanje

Prvi problem na katerega sva naletela je, kako hkrati prevajati Scalo in Clojure, ter kako v Scali uporabiti funkcije iz Clojura. Oba jezika sta v precejšnji meri kompatibilna z Javo, v smislu, da lahko uporabljata vse Javine razrede, z nekaj truda gre tudi obratno, z še nekaj več truda, pa lahko sodelujeta tudi Scala in Clojure.

Po približno tednu pisanja lastne skripte za prevajanje, ki sva jo poimenovala *bashbeans*, sva ugotovila, da za oba jezika obstaja način, za dinamično nalaganje kode med delovanjem programa. Delno vlogo pri razvoju bashbeans pa je imelo tudi dejstvo, da ima Matic precej počasen prenosnik, ter da ima Scala precej počasen prevajalnik – bashbeans tako vsakokrat shrani hash vrednosti izvornih datotek in ne prevaja ponovno tistih datotek, ki se niso spremenile. Kasneje sva odkrila, da ima Scala tudi fsc(fast scala compiler), ki naloži prevajalnik v ozadje kot daemon in tudi hrani delno prevedene datoteke in je tako le prvo prevajanje po zagonu sistema počasno. Tako je na žalost večina uporabnosti bashbeansa odpadla, imava pa še dovolj idej, kako ga poboljšati in morda uporabiti še na kakem drugem projektu.

2.3 Modeliranje

Zelo zgodaj v razvoju sva se odločila, da bodo vsi podatki in modeli generirani programsko – nekateri so skorajda ročno napisani v kodi, kar ni najlepše, ampak večina pa uporablja nekakšne naključne algoritme za generiranje.

Pri tem se je funkcijsko programiranje zelo izkazalo, saj je možno napisati razred, ki v konstruktor sprejme dve funkciji: eno ki generira objekte, ter drugo, ki iz zgeneriranih podatkov izriše grafiko – razred nato znotraj sebe avtomatsko shrani izris kot OpenGL DisplayList in so tako nadaljni izrisi precej hitrejši, brez da bi bilo treba za to kaj posebej narediti.

Uporabnost tega se je pokazala tudi, ko sva lahko zelo hitro razvila avtomatsko prilagajanje stopnje detajlov, glede na hitrost računalnika – ob spremembi stopnje detajlov, razredu preprosto povemo, naj posodobi svoj izris.

2.4 Generiranje dreves

Poglavje 3

Rezultati