Planetsimulator

Johannes Olegård Anton Olsson

25 april 2013

1 Programbeskrivning

Vi ska skriva ett program som simulerar ett 2D-universum. Användaren kan skapa planeter i olika storlekar, massor, färger, hastigheter, accelerationer, etc. och planeter kommer sedan interagera med varandra (t.ex. skapa omloppsbanor).

2 Användarbeskrivning

Applikationens användargrupp är seende mellan 13-90 med åtminstone lite datorvana som är intresserade av att experimentera med fysik. Användaren förutsätts ha viss kunskap i relevanta fysiskaliska begrepp (t.ex. vet ungefär vad massa och radie är) då programmet ej förklarar dessa. Vidare förutsätts användaren förstå engelska då text i programmet är på engelska.

Programmet skulle t.ex. kunna användas av högstadiebarn för att lära sig experimentera med fysik för att förstå hur gravitation, etc. påverkas av de olika storheterna.

3 Användarscenarier

3.1 Scenario 1

Användaren är en datalog på kth som går första året. Fysikkunskaperna är goda. Användaren vill simulera ett solsystem med 3 planeter i omloppsbana runt en stjärna.

Programmet startas med ett fönster. Vänstra delen av skärmen består av en meny. Längst ner i mitten av skärmen är ett tickande tidtagarur (visare men inga siffror). Resten av skärmen är svart.

I menyn finns parametrar som t.ex. massa och radie att ställa in. Användaren ställer in stjärnans egenskaper och klickar sedan i mitten av skärmen där det då skapas en ifylld cirkel.

Användaren ställer in nästa planet och håller inne mus-knappen där planeten ska skapas och drar sedan pekaren åt det hållet initial hastighet ska vara. Medans användaren drar med musen så syns en pil från planeten till pekaren med text brevid om hur hög hastigheten blir när användaren väl släpper.

En streckad ellips visar hur omloppsbanan kommer bli. När användaren släpper så börjar den nyskapade planeten röra sig i en ellips runt stjärnan. Användaren repeterar mönstret för 2–3 planeter men märker denna gång att när hen håller på att placera (musknapp nertryckt och vektorn för hastighet synes) en ny planet så pausas simulationen för att underlätta placeringen.

3.2 Scenario 2

Användaren har viss erfarenhet av datorer (bl.a. spel) och fysikkunskaperna är minimala. Programmet start likt beskrivet i Scenario 1. Även denna användare ska skapa ett solsystem med en stjärna och några planter i omloppsbana runt den.

Användaren testar att klicka ut lite planeter utan att ändra några parametrar. De rör sig inte så användaren försöker dra i en av dem. Planeterna och klockan nere i mitten pausas samtidigt, planeten markeras och en hastighetpil visas från den till muspekaren.. Användaren släpper musknappen och planeten får fart - ut från skärmen. Användaren försöker få skärmen att följa efter och upptäcker att det går att pause med space-knappen och att zooma med mushjulet.

Planten ges nu lite fart tillbaka - lite mindre denna gång.

Användaren provar nu att ändra parameterarna på planeten. Ökad storlek verkar inte ge gravitation, men massa verkar göra det.

Användaren ökar massan rejält planeten och alla andra planter börjar åka snabbare och snabbare mot den.

Vissa planeter missar planeten, får extra hastighet och flyger förbi, bortåt. Andra planeter åker rakt in i planeten och så att de smälter samman. Användaren noterar att massan ökar i menyn till vänster.

Användaren stannar den stora planeten genom att sätta dess hastighet till noll i menyn till vänster och börjar sätta ut planeter en sträcka ifrån den - med olika hastigheter, åt olika håll och med låga massor.

Efter några försök har användaren lyckats få några planeter att åka i en elliptisk bana runt den stor planeten.

4 Testplan

Vi testar programmet framförallt genom användartestning.

Användartestningen ska ske genom att ta tag i lämplig person i godtycklig kth korridor och få denne att utföra en eller flera av följande uppgifter:

1. Skapa ett solsystem med en planet och en tillhörande måne.
2. Justera omloppsbanan på en yttre planet i ett existerande solsystem med 2 orbitaler så att planeten hamnar innanför den andra.
3. Skapa ett dubbelstjärnigt solsystem eventuellt med en tillhörande planet.
4. Skapa en lite galax. Många stjärnor i mitten och fler hela solsystem som kretsar runt dessa.

De två senare uppgifterna är lite svårare och vi räknare inte med att användarna klarar dessa. Däremot vill vi gärna se hur de försöker använda programmet för att lösa problemen.

Innan användaren börjar ska vi kort förklara med ord (utan att visa) vad vi vill att hen ska försöka utföra. Vi ska också tydligt förklara att eventuella misslyckanden beror på brister i det grafiska gränssnittet och inte användarens bristande intelligens (även om det säker bidrar). Vi behöver tydligt anmärka att programmet är under utveckling och därför är det fullt möjligt att buggar gör uppgiften omöjlig. Det finns otydligheter i det grafiska gränsnittet och vi ber därför användaren att säga högt vad hen tänker så vi kan förstå vad som behöver ändras/förtydligas för en användarvänlig design.

5 Programdesign

Programmet ska använda opengl för det grafiska med hjälp av bibloteket libgdx[1]. Artemis entity system[2] kommer användas för att lättare separera koduppgifter i mindre klasser där varje klass gör en (1) enskild sak.

Artemis använder sig av komponenter och system, där komponenter håller data och systemen bearbetar dom. Ett system använder endast några få komponenter och arbetar endast på objekt som har dom komponenterna.
Planeterna ska ha följande komponenter:

- Position
- Hastighet
- Acceleration
- Massa
- Storlek
- Färg

De system som behövs är:

Gravitation behöver använda *position*, *massa* och *acceleration* för att räkna ut gravitations krafter mellan planeterna.

Acceleration behöver *acceleration* och *hastighet* för att förändra hastigheten.

Hastighet behöver *position* och *hastighet* för att uppdatera positionen.

Ritande behöver *position* och *storlek* för att rita planeterna på skärmen.

6 Tekniska frågor

1. Vilken algoritm ska användas för att justera planeters position, hastighet och acceleration? Ska t.ex. alla planeter jämföras med alla andra planeter i universumet, eller ska algoritmen vara selektiv och inte räkna planeter för långt borta?
2. Hur hanteras kolissioner? Studsar, spricker eller smälts planeter samman? Hur kommer kollisionsystemet påverka resten av programmet? Behövs systemet snabbas upp?
3. Behöver programmet snabbas upp med parallellisering? I så fall, hur? Bör låsning av delade variabler undvikas?

[1] http://libgdx.badlogicgames.com/features.html
[2] http://gamadu.com/artemis/index.html

7 Arbetsplan

Arbetet pågår under vecka 17 t.o.m. 20. Under denna tid är det 3 övningstillfällen: fre 26 april (v17), fre 3 maj (v18) och ons 15 maj (v20).

Projektplanen har deadline vid den första övningen. Slutrapporten och programmet har deadline vid den sista övningen. Vid varje övningstillfälle ges en muntlig lägesrapport.

Vecka 17 spenderas på att skapa ett git-repository med ett kodskelett och att skriva projektplanen. Om det finns tid över läggs den på att till viss del att börja fylla kodskelettet.

Vecka 18 ska användas till att skriva så mycket av programmet som möjligt. Om det finns tid över börjar användartestningarna v18 och fullgörs annars (senast) v19.

Vi räknar med ett fullt fungerande, debuggat och användartestat programmet i mitten av v19, 1 maj. Slutrapporten ska skrivas så mycket som möjligt parallellt med resten av arbetet, men framförallt slutet av vecka 19 används till rapporten. Om det finns tid över kan det användas till debugging i programmet.

Vi räknar med att slutrapporten är klar i slutet av vecka 19, fre 11 maj.

Vecka 20 (totalt 2 dagar innan inlämning) används endast om något moment tog längre tid än förväntat.