Projektplan

Adrian Blanco och Casper Winsnes

April 16, 2012

0.1 Programbeskrivning. Beskriv detaljerat vad programmet gör:

Vårt program är en mandelbrotgenerator. Programmet kan generera bilder av mandelbrotset samt juliaset, och förhoppningsvis i framtiden andra användardefinierade funktioner. Programmet ska ha ett enkelt och minimalistiskt GUI med få knappar, i fokus ska de genererade bilderna ligga. Användare ska kunna bestämma graden av antialiasing, zoom, panorering och ha möjlighet att spara den generade bilden som en bildfil. Programmet ska dessutom ha fullskärmsfunktionalitet och fönsterfunktionalitet. Med hjälp av biblioteket Aparapi kan vi få våran javakod att konverteras till openCL och köras i grafikkortet för extra hastighet.

0.2 Användarbeskrivning. Vem kommer att använda ert program? Vilka antaganden gör ni om användarna? Är de vana datoranvändare, är de specialister, nybörjare, små barn?

Användare ska kunna vara vem som helst, då fokus ska ligga på enkelhet i framställningen. De användargenerade bilderna med egna funktioner lär dock antagligen kräva viss kunskap om mandelbrotset för att kunna användas fullt ut. Viss kunskap om inställningsmöjligheterna (framförallt antialiasing) kan behöva förklaras för en ovanare användare.

0.3 Användarscenarier. Ge minst två exempel på scenarier där en av era tänkta användare använder programmet. Beskriv i detalj vad de ser, vilken typ av input de måste ge, hur de ger sin input och hur programmets output ser ut.

1. Tänkt användare: tonåring/20-årsåldern med intresse för matematik. Har studerat fraktaler tidigare. Användaren vill se hur ett mandelbrotset ser ut och ha möjligheten att fokusera på de bitar som hen finner intressanta. Användaren vill även undersöka mer avancerade funktioner (exempelvis juliaset). Det första som användaren ser är det förinställda mandelbrotset som genereras vid startandet av programmet. Det enda input som krävs är musklickningar för att kunna zooma. För att spara bilder och ändra inställningar finns det självförklarande knappar att klicka på i ena hörnet, en av knapparna öppnar en meny för avancerade inställningar som sedan användaren kan välja mellan för att ändra mandelbrotsettet. För att generera mer avancerade funktioner finns några förinställda i inställningsmenyn samt en ruta för personligt input,

funktionalitet för dessa är detsamma som för mandelbrotset. Programmets output är en mandelbrotfraktal som visas i bild. Bildfiler är ett valbart output om användaren vill.

2. Tänkt användare: lågstadiebarn med viss datorvana med lärare/förälder vid sidan om. Lärare/förälder vill att barnet ska få se fina bilder och mönster och står bredvid för att hjälpa till vid eventuella problem/oförståelse. Input är väldigt enkelt, varför barnet instinktivt borde klara av att hantera de grundläggande delarna av programmet bara genom att klicka sig fram. Om barnet vill ändra inställningar med mer kan det dock behöva hjälp av en äldre person som kan hjälpa det att förstå vad de olika inställningarna betyder.

0.4 Testplan. Beskriv hur ni tänker testa programmet. I den här uppgiften ska ni lägga extra vikt vid användartestning. Beskriv vilka uppgifter som testanvändaren ska utföra. De två användarscenarierna ska ingå i användartestningen.

Scenario 1: (Tänkt användare nr. 1) Programmet ska testas genom att låta användaren får en kort beskrivning om vad programmet kan göra och överlåts sedan till att testa fritt. Några uppgifter ska genomföras av användaren: - Zooma - Ändra antialiasing - Ändra typ av bildgenerering - Spara en bild

Scenario 2: (Tänkt användare nr. 2) Programmet ska testas genom att vi visar användaren ett mandelbrotset (i programmet). Vi ber sedan användaren att gissa sig till hur man zoomar in på en punkt i programmet och ber henom även att zooma ut. Användaren ska även testa de mer avancerade funktionerna men då med viss assisstans från en mer erfaren användare av programmet.

0.5 Programdesign. Beskriv de grundläggande klasserna som ni avser att implementera och ge en beskrivning av de viktigaste metoderna i varje klass.

MandelbrotFrame: Klassen ärver från javax.swing.JFrame och använder sig således främst av dess ärvda funktioner. Denna klass tar hand om skärmen och ska implemenetera MouseListener för att hålla koll på användarens musklick samt andra interaktioner med skärmen. Av dess metoder kommer främst MouseClicked att användas. Klassen innehåller också setFullscreen vilken gör om fönstret så att den fyller hela skärmen.

MandelbrotCanvas: Klassen ärver från java.awt.Canvas och använder sig främst av metoderna paint, zoom och render. Dessa använder sig av en BufferedImage att generera bilder på och sedan ritas ut på Canvasen. Klassen är tänkt att ritas ut på vår MandelbrotFrame. De mest använda funktionerna är not zoomIn() och zoomOut() som hanterar zoomningen.

MandelbrotGenerator: Detta är en hjälpklass som sammanfattar funktioner för GPUKernel och Antialiasingkernel eftersom de inte är tänkta att användaren ska interagera med. De mest använda funktionerna är nog calculate() som säger åt GPUKernel och

AntialiasingKernel att arbeta, samt setAntialiasing som genom räknar ut vilka värden som ska ändras i båda kernels för att hantera ett nytt värde på antialiasing och changeSize() som används för att anpassa värdena i båda kernels till en ny fönsterstorlek.

GPUKernel: GPUKernel är en klass som ärver från com.amd.aparapi.Kernel och genererar en mandelbrotbild som (oftast) är större än skärmen för att kunna applicera antialiasing på den. Huvudalgoritmen sitter i run() metoden, och vi använder biblioteket aparapi som konverterar all kod i den metoden till OpenCL och exekverar den i grafikkortet (eller ifall den inte kan det så exekverar den algoritmerna i processorn i multipla trådar). Eftersom vi konverterar koden till OpenCL i run() är det väldigt många restriktioner som gäller, till exempel får man inte använda objekt eller använda funktioner som finns utanför run() i de flesta fall vilket gör att det blir mycket kod och variabelduplicering, däremot är det värt det eftersom koden går mycket snabbare. Förutom run() så är några använda funktioner setMagnification() som ändrar zoomen i bilden och erase() som ritar bilden helt svart.

AntialiasingKernel: AntialiasingKernel är mer eller mindre en kopia av GPUKernel förutom själva run() metoden. run() i GPUKernel genererar en stor mandelbrotbild, medan run() i AntialiasingKernel har som funktion att applicera antialiasing genom att ta medelvärdet av färgerna i vissa områden i bilden och sedan spara dem i en mindre bild. Förutom run() så är nog den mest använda funktionen setSource som berättar för klassen vart bilden som den ska applicera antialiasing på finns.

0.6 Tekniska frågor. En lista av tekniska frågor som ni måste hantera när ni bygger ert system. Var så detaljerad som möjligt. Ett viktigt steg mot en god design är att få ner så många frågor som möjligt på papper på ett organiserat sätt med så många förslag till lösningar som möjligt.

Bra algoritmer:

En stor del av vårt program bygger på matematiska algoritmer, och det krävs väldigt mycket för att få dem rätt. Framför allt så är algoritmerna som tar hand om mandelbrotsettet samt antialiasingen väldigt viktiga för programmet och därför är det mycket viktigt att inte bara få dem rätt utan att också göra dem bra eftersom de kommer itereras upp emot några miljoner gånger per sekund. I många fall blir det en kompromiss mellan kvalitet och snabbhet och det är någonting man måste ta hänsyn till

GPU-acceleration:

Att få grafikkortet att räkna är inte lätt. Det finns mycket problem med trådar, väldigt anonyma crasher, restriktiva kodningsregler, och specifika hårdvarubuggar som man inte alltid kan förutse överhuvudtaget.

GUI

Trots att GUIt inte kommer vara i fokus måste det ändå vara användarvänligt och inte förvirrande, ifall våra planer på GUIt uppfyller det återstår att se.

0.7 Arbetsplan. Beskriv hur arbetet kommer att delas upp mellan personerna i projektet. Gör en tidsplan som visar när olika delmoment i projektet ska vara klara.

Adrian arbetar med det grundläggande grafikarbetet och lågnivåkod och algoritmer/optimisering.

Casper arbetar med användarvänlighet, javaspecifika funktioner och högnivåkod samt hjälper till med grafikarbetet.

Vi har redan fått en fullt fungerande prototyp färdig, och det mesta som är kvar att implementera är användargränssnittet samt avancerade extrafunktioner.

UI är högsta prioritet i nuläget, vi måste få ett fungerande och lättanvänt gränssnitt som inte är förvirrande. Sedan efter det gäller det att koppla ihop UIt med funktionerna i MandelbrotGenerator, så att användaren själv kan ändra på bilden.

Lite mindre prioriterade funktioner är att kunna spara bilder, samt att ändra på mandelbrotalgoritmen så att den genererar ett juliaset.

Funktioner som vi har i åtanke att implementera, men inte är säkra på ifall en implementation är helt trolig är bland annat användarspecificerade funktioner, dragging av bilder.