Visualisering af funktioner af to variable i Minecraft

Philip Peder Hansen

December 20, 2013

Contents

Figure 1: En *Minecraft* verden

0.1 Funktioner af to variable

0.1.1 Afbildning af funktioner af to variable

0.2 Minecraft

Minecraft er et videospil lavel af *Markus "Notch" Persson*. Spillet er en blanding mellem et eventyr overlevelses spil, og et kreativt spil der går ud på at bygge verdener.

Spilleren har mulighed for at ødelægge blokke, samle dem op, og placere dem tilbage i verdenen. Man kan møde monstre der slår på en, skyder ild kugler og enda eksplodere, all med formål at gøre det svært at overleve.

I sin kamp mod det onde kan spilleren bygge et sikkert hus, grave efter metaller og smelte disse til jern, guld og diamant brynjer og svære.

Spillet har også et element der heder *redstone*, der til en vis grad fungere som ledninger. *Redstone* kan brugest til at lave kredsløb der automatisere skydning af pile, hældre lava ud over ens fjænder og er enda brugt til at lave *minigames* i *Minecraft*.

Indledene bliver spilleren placeret i en stor verden, der består af 1 m^3 blokke. Disse blokke har forskellige fysiske egenskaber, og bruges i forskellige sammenhænge.

Derudover genereres blokkene baseret på nogle algorytmer, der resultere i et mønster der til nogen grad minder og den virkelige verden. Trær genereres

i nærheden af hinanden, i skove, og ikke midt ude i havene. Ørkner og tundra findes generelt ikke umidelbart i nærheden af hinanden, og floder løber ofte gennem regnskove.

0.2.1 Generering af verdner

Algorytmerne der bruges til at generere verdener i*Minecraft* er en del mere komplicerede end en det simple eksemepel på den funktion af to parametre vi har kigget på. Ud over X og Z koordinated bruger minecraft også noget som kaldes et seed til at generere verdener, de genereres altså ud fra en mere kompliceret funktion af tre parametre.

Et *seed* er en tekst streng som brugeren kan give *Minecraft* når en ny verden genereres, hvis brugeren ikke giver denne streng bliver den automatisk genereret før verdenen laves. Funktionen af et seed er at selv med den samme verden genererings kode, kan vidt forskellige verdener laves.

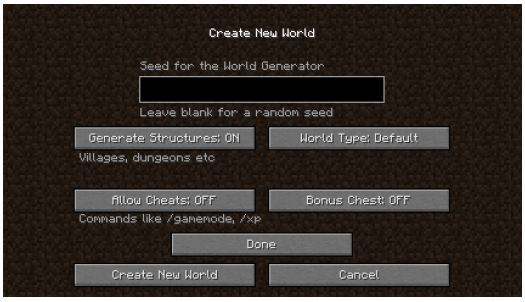

Figure 2: *Minecraft* verden genererings skærm

Forstil dig for eksempel dette scenarie, vi bruger den følgene formel til at generere en kurve

$$y = \sin(x)$$

Denne formel vil altid give en sinus kurve, som vi også ville forvente det, det passer perfect i matematik, men ikke så meget hvis vi prøver på at lave interesante mønstre.

Forstil dig nu hvis vi ændrede ligningen til at vre

$$y = \sin(seed * x)$$

Så længe *seed* ikke er lig nul, vil denne formel stadig give en sinus kurve, men afhængligt at hvad vi sætter *seed* til at være, vil perioden for vores kurve være forskellig.

På samme måde bruger *Minecraft* dette *seed* i generations koden til at skabe verdener der er unikke, på trods af at de alle er genererede fra den samme kode.

Ud fra dette seed og generations algorytmerne, bestemmer *Minecraft* hvilken blok der skal placeres på hvert punkt i verdenen.

0.2.2 Planlgning

For at implementere vores egen kode i *Minecraft*, og generere en verden baseret ud fra den følgene funktion, er der nogle overvejelser vi først må lave om hvordan dette skal forgå.

$$z = \cos(x^2) + \sin(y^2)$$

Den første, og måske vigtigste, overvejelse jeg har gjort min i forhold til at implementere denne formel i *Minecraft* er hvordan koden skal indkorporeres i spillet.

Da spillet ikke distribueres som kildekode, er det altså ikke muligt bare at skrive koden ind i den orginale kode, og kører spillet. Heldigvis findes der en uofficiel API til *Minecraft*, der gør det muligt at skrive noget kode som en separat file, som bliver kørt når *Minecraft* gør det.

0.2.3 Implementering

0.2.4 Vurdering

0.3 Perspektivering