Tygsimulering

TNM085
Grupp 4

Linnea Bergman
Rebecca Cedermalm
Yrsa Lifvergren
Erik Rudhe

13 mars 2017

Innehåll

1 Sammanfattning

Den här rapporten beskriver och redogör hur en grupp på fyra personer har gått till väga för att simulera tyg på ett realistiskt sätt. Med hjälp av kunskap från tidigare modelleringskurser har gruppen kunnat gå från en simpel modell i Matlab till en animation i OpenGL.

Rapporten tar upp en redogörande beskrivning av systemet samt vilka numeriska metoder som användes och vilka verktyg som krävdes för att genomföra animationen. I diskussion och slutsatskapitlet beskrivs det vilken numerisk metod som lämpade sig bäst samt vilka beslut som togs i projektet.

2 Inledning

I detta avsnitt beskrivs bakgrund och syfte med projektet.

2.1 Bakgrund

Inom filmer och animering är det stora mängder tyg som visualiseras. Allt från Stålmannens mantel till kläderna på en bybo modelleras till dessa projekt. Att modellera tyg manuellt är svårt att få verklighetstroget och därför används ofta simuleringar för att visualisera tyg.

Detta projekt behandlar hur tyg kan simuleras på ett realistiskt vis genom beräkningar om hur tyg rör sig. Ekvationerna som används vid beräkningarna bestäms genom modellering. Ett vanligt sätt att modellera tyg är genom ett mass-fjäder-dämparsystem där tyget är representerat av flera massor som är sammankopplade med hjälp av fjädrar och dämpare.

2.2 Syfte

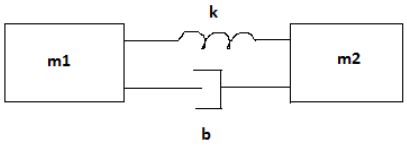
Syftet med projektet är att, utifrån framtagna matematiska beräkningar, simulera och animera ett tygstycke. Målet är att få tygstycket att upplevas verklighetstroget och reagera på de fysikaliska krafter som finns runt omkring.

3 Redogörelse

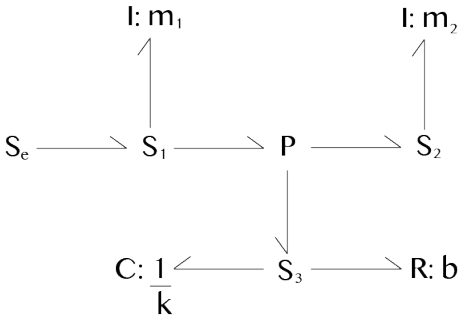
I detta kapitel redogörs hur arbetet i projektet har gått till. Systemet beskrivs grundligt och implementationen skrivs om i stora drag utan att gå in på detalj i hur programmeringen gick till.

3.1 Beskrivning av systemet

Till en början skissades en fysikalisk modell bestående av två sammankopplade partiklar som representerade två massor i ett tygstycke, se Figur 1. Med hjälp av en fjäder med fjäderkonstanten k samt en dämpare med dämpningskonstanten b kopplades massorna m_1 och m_2 ihop. Utifrån den framtagna modellen ritades en bindningsgraf upp, se Figur 2, för att på ett annat sätt visa hur systemet är uppbyggt. S_e symboliserar kraften som kommer in i systemet, m_1, m_2, b och k står för massorna, dämpningskonstanten respektive fjäderkonstanten.



Figur 1: Skiss över två partiklar i ett tygstycke.



Figur 2: Bindningsgraf över systemet i figur 2.

Systemet använder sig av Newtons andra lag om rörelse som säger att det behövs en kraft för att accelerera ett föremål (1).

$$F = ma \tag{1}$$

F är den resulterande kraften, m är massan på föremålet och a är accelerationen. Då en kraft appliceras på en av partiklarna i systemet påverkas de andra partiklarna eftersom de är sammankopplade. Det resulterar i en acceleration för varje partikel i systemet. Accelerationen sätter partiklarna i rörelse och ger upphov till en hastighet som gör att varje partikel uppdaterar sin position. Den nya positionen för varje partikel i systemet orsakar en längdskillnad i det sammankopplade fjäder -dämparsystemet.

Systemet använder sig också utav Hookes lag som säger att förlängningen av en fjäder är proportionell till den applicerade kraften (2).

$$F = -kx \tag{2}$$

F är den resulterande kraften, k är fjäderkonstanten och x förskjutningen från jämviktsläget. För att reducera att systemet oscillerar tillsätts också en dämpare mellan partiklarna. Den sammansatta ekvationen blir då enligt ekvation 3.

$$F = -k_s(|L| - L_0)(\frac{L}{|L|}) - k_d(v_1 - v_2) \tag{3}$$

F är den resulterande kraften, k_s fjäderkonstanten respektive k_d dämparkonstanten. L är i det här fallet en vektorn mellan de två partiklarna och L_0 längden på fjädern i viloläge. v_1 och v_2 anger hastigheten för de båda partiklarna. [1]

Genom att applicera Hookes lag kan en ny kraft beräknas och tas med senare i simulationen. Det leder till en ny acceleration för varje partikel. Den upprepade processen av att beräkna krafterna och uppdatera positionen gör att simulationen känns levande och ger intrycket av tyg.

3.2 Numeriska metoder

Två olika numeriska metoder, Euler och Runge-Kutta, implementerades för att beräkna tygpartiklarnas nästa position [2]. De numeriska metoderna används två gånger. Först för att beräkna nästkommande hastighet på partiklarna för att sedan kunna använda den för att kunna beräkna nästa position.

Först implementerades Euler (4),

$$v(t + \Delta t) = v(t) + (\Delta t)v'(t) \tag{4}$$

där v står för position om det är nästa position som beräknas, eller för hastighet om det är nästa hastighet som beräknas. $v'(t)$ står för acceleration om det är hastigheten som beräknas, eller för hastighet om det är nästa position som beräknas. Δt representerar förändringen i tid. För att se noggrannheten hos positionsberäkningen kunde förbättras implementerades också Runge-Kutta av fjärde ordningen (5-6).

$$v_1(t + 1) = v_1(t) + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{5}$$

$$\begin{aligned} k_1 &= v_2(t) \\ k_2 &= v_2(t) + \frac{h}{2}k_1 \\ k_3 &= v_2(t) + \frac{h}{2}k_2 \\ k_4 &= v_2(t) + hk_3 \end{aligned} \tag{6}$$

Är det nästa position som beräknas representerar $v_1(t + 1)$ den kommande positionen, $v_1(t)$ den nuvarande positionen och v_2 hastigheten. Är det istället nästa hastighet som beräknas representerar $v_1(t+1)$ nästa hastighet, $v_1(t)$ den nuvarande hastigheten och v_2 accelerationen. h är steglängden, hur stora steg metoden ska ta i sina beräkningar. k_1, k_2, k_3 och k_4 anger ökningen från föregående hastighet eller position.

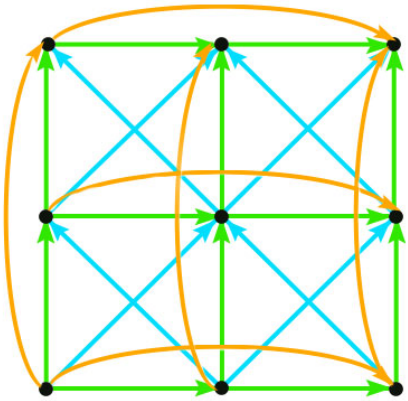
I Runge-Kutta tas fler värden med i beräkningarna än i Euler vilket minskar felet och ökar noggrannheten.

3.3 Implementering av animationen

Ett rutnät av partiklar skapades. Varje partikel fick en egen massa, position, hastighet och acceleration. Massan valdes till samma för varje partikel eftersom varje del av tyget väger lika mycket. Ursprungshastigheten och ursprungsaccelerationen sattes till noll. Fjäder- och dämpningskonstanten bestämdes.

För varje bild i simuleringen beräknades en ny acceleration för varje partikel genom att räkna ut de olika fjäder- och dämpningskrafterna mellan partikeln och de närliggande partiklarna. Tre olika typer av fjädrar användes: strukturella-, tvär- och böjningsfjädrar, se figur 3, för att få tyget att se realistiskt ut. I figuren representeras de strukturella fjädrarna av gröna pilar, tvärfjädrarna av blåa och böjningsfjädrarna av orangea.

På varje partikel lades även tyngdkraften till i krafterna. Den nya hastigheten och positionen på partikeln beräknades sedan med hjälp av Runge-Kutta. Positionen för varje partikel användes sedan för att rita upp tygstycket på skärmen.



Figur 3: Skiss över de tre olika typerna av fjädrar som användes.

En kamera implementerades så att användaren kan röra scenen och se tyget från flera olika håll. Det lades även till att användaren kan, med hjälp av klick med musen, lägga till ytterligare en kraft på en partikel i mitten av tygbiten för att få tyget att röra sig.

3.4 Vertyg

Det verktyg som användes för att simulera partiklarna i tygstycket var datorprogrammet Matlab. Där implementerades de två olika numeriska metoderna och jämfördes med varandra för att se vilken av dem som var bäst lämpade.

Till animationen av tygstycket användes OpenGL som är ett plattforms-oberoende API för att skriva applikationer med datorgrafik i två eller tre dimensioner [3]. OpenGL skrevs i programmeringsspråket C++. Utöver OpenGL hämtades diverse bibliotek: GLFW, GLUT och GLM. GLFW och GLUT användes för att sätta upp ett fönster, GLM för matematiska operationer. För att den skrivna koden skulle fungera på alla plattformar användes programmet CMake, som hanterar byggnadsprocessen av en kod genom att använda en oberoende kompileringsmetod [4].

För versionshanteringen under projektet användes verktyget Git och GitHub. Där skapades olika grenar med olika områden inom projektet. Det medförde att all

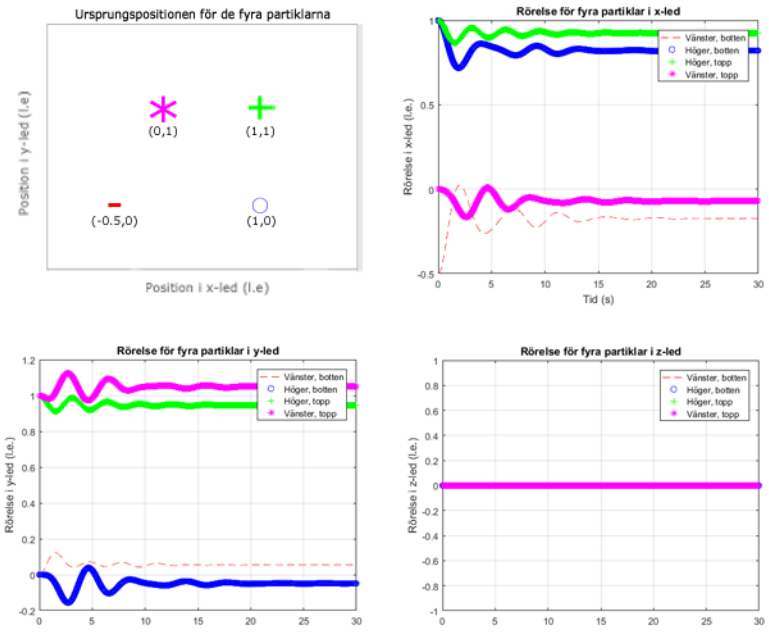
kod som skrevs kunde spåras och att medlemmarna i gruppen kunde arbeta på olika saker samtidigt.

3.5 Instruktioner för körning

Det krävs att användaren bygger projektet med hjälp av CMake för att kunna köra programmet. Programmet startas med hjälp av mellanslag. Användaren kan röra runt på scenen med tangenterna W, A, S, D och musen. Vid klick med vänster musknapp förflyttas en partikel i mitten av tyget i positivt z-led. Var försiktig med att röra musen för mycket då det kan vara lätt att "tappa bort" tygstycket. En inspelad version av resultatet går att hitta på Youtube: https://youtu.be/mx1k_C8rC9Y.

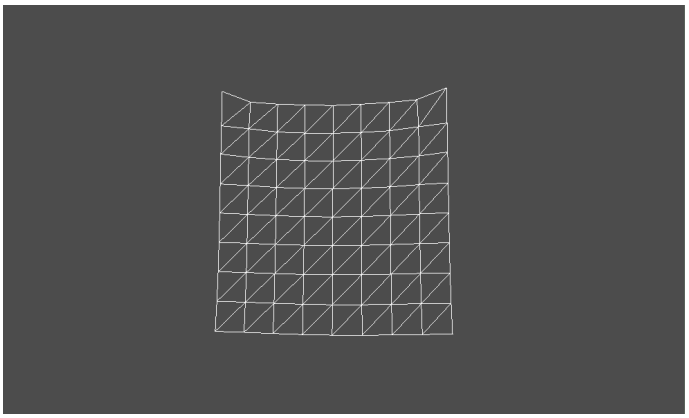
4 Resultat

Figur 4 visar resultatet från simuleringen av fyra partiklar i MATLAB när partikeln i vänstra bottenhörnet blivit utdragen en bit i x-led. Partiklarna sätts i rörelse av att krafterna mellan dem vill dra tillbaka partikeln till viloläget för fjädrarna. Vid förflyttningen i x-led av partikeln rör sig de andra partiklarna endast i x- och y-planet, därav är rörelsen i z-planet noll.



Figur 4: Grafer över fyra partiklar i ett tygstycke.

Resultatet av implementationen i C++ och OpenGL visas i Figur 5. Varje partikel representeras av en vertex. Tillsammans skapar partiklarna ett tvådimensionellt plan i en 3D värld. De partiklarna i översta hörnet sattes till stationära punkter, vilket gör att tyget stannar kvar i bild trots att en tyngdkraft är tillagd. Användaren har en möjlighet att interagera med tyget via klick med musen.



Figur 5: Rutnät av partiklar ritade i OpenGL.

5 Diskussion

Genom att endast titta på graferna skapade i MATLAB var det svårt att avgöra om resultatet var korrekt eller ej eftersom det inte fanns något att jämföra med. Det ansågs dock som korrekt då det var naturligt att den partikel som blev förflyttad rörde sig mest och att de andra partiklarna rörde sig i förhållande till den. Partiklarna stannade även upp efter ett tag. I OpenGL var det lättare att se att resultatet var det eftersökta. Partiklarna gav intrycket att röra sig som ett tyg gör.

Till en början visualiserades partiklarnas positioner med kuber. För att simuleringen skulle efterlikna tyg implementerades streck mellan positionerna. Dock visade sig detta vara problematiskt att kombinera tillsammans med kuber, därför förkastades kuberna helt och tyget visualiseras endast genom linjerna mellan partiklarnas positioner.

Tygstycket som simuleras har få punkter men tar mycket prestanda och simulationen tar lång tid att visualisera på grund av alla beräkningar som utförs. Om tygstycket ökar i storlek blir det fler krafter som tas med i beräkningen vilket resulterar i fler beräkningar. För att minska beräkningsprocessen kan Euler användas för approximationen, istället för Runge-Kutta. Genom att minska beräkningarna går animationen av simuleringen snabbare. I projekt som kräver en större noggrannhet på approximeringarna, till exempel för att simulera hur större tygstycken rör i vinden, är Runge-Kutta ett bra alternativ. Snabbt eskalerar beräkningarna till en sådan mängd att en kraftig dator krävs för att simulera systemet utan att det blir för stora fördröjningar.

Ett annat alternativ för att påskynda beräkningarna är att använda API:n OpenCL. OpenCL utför beräkningarna på datorns GPU istället för CPU. Genom att dela upp arbetet mellan GPU:n och CPU:n kan fler beräkningar utföras på samma tid. Dock var OpenCL dåligt dokumenterat och krångligt att implementera så det alternativet förkastades.

Några funktionaliteter som diskuterades att vidareutveckla projektet med är bland annat att sätta en textur på rutnätet och lägga till en lampa i scenen. En textur och lampa gör simuleringen ännu mer verklighetstrogen. Vidare kan kollisionshantering och interaktion mellan tygstycket och andra objekt implementeras. Detta ökar användbarheten av simuleringen till mer praktiska situationer. Om mer tid till projektet funnits och ovannämnda funktionaliteter implementerats hade nästa steg varit att lägga till en funktionalitet som tillåter användaren att skära i tyget med musen på ett trovärdigt vis.

6 Slutsats

Simulering av tyg genom beräkningar av differentialekvationer är möjligt med de kunskaper som gruppen har. För att simulera ett större och mer omfattande stycke tyg kan det krävas en dator med hög prestanda eftersom animationen annars kan bli väldigt långsam. Projektgruppen är i sin helhet nöjda med resultatet. Tyget anses röra sig verklighetstroget.

Referenser

[1] Kieran E, Harrison G. *Cloth Simulation* [Internet]. Bournemouth: NCCA Bournemouth University; 2005. 2017-03-13, hämtad från: https://nccastaff.bournemouth.ac.uk/jmacey/MastersProjects/Msc05/cloth_simulation.pdf

[2] Ljung L, Glad T. *Modellbygge och simulering*. Upplaga 2:5. Lund: Studentlitteratur; 2011.

[3] *OpenGL* [Internet]. 2017-03-09, hämtad från: https://www.opengl.org/

[4] Wikipedia. *CMake* [Internet]. 2017-03-08. 2017-03-09, hämtad från: https://en.wikipedia.org/wiki/CMake