Stelkroppssimulering

Tomas Forsyth Rosin, Jonas Zeitler, Emil Axelsson, Karl Johan Krantz

5 mars 2013

Sammanfattning

Sammanfattning av projektet.

Innehåll

Figurer

Tabeller

1 Inledning

Fysiksimulering är en vital del av många moderna datorspel. I spelsammanhang kallas detta för ett spels fysikmotor. Att implementera en fysikmotor handlar om att skapa en miljö med regler vilka en uppsättning objekt måste följa. I denna rapport undersöks stelkroppssimulering som utgör basen i de allra flesta fysikmotorer. Begreppet antyder att en fysikmotor för stelkroppssimulering gör vissa antaganden och förenklingar av den fysiken i en riktig värld. Mest påtagligt är att systemets objekt antas vara okomprimerbara, eller stela. Även med dessa förenklingar i beaktning är implementation av en fysikmotor ett komplext problem. Det finns många olika metoder för att nå likvärdiga resultat och att låsa projektet till en metod i ett tidigt skede kan försvåra senare utveckling. I denna rapport implementeras och utvärderas en impulsbaserad, iterativ, fysikmotor med stelkroppssimulering.

2 Kollisionsdetektion

2.1 Detektionsmetod

Det första steget mot att simulera stelkroppars interaktion är att detektera alla kolliderande polygoner i scenen. Att två objekt i ett diskret system ligger dikt an varandra är sällsynt. Vanligare är att de interpenetrerar varandra, d.v.s. att ett hörn hos ett objekt ligger inuti ett annat objekt.

2.1.1 Separating Axis Test

För att utreda eventuell interpenetration mellan två givna objekt kan SAT [1, s. 29] *(Separating Axis Test)* användas. Testet kan avgöra om objekten inte kolliderar genom att hitta en vektor på vilken objektens punkter projiceras på disjunkta intervall (Figur **??**). Ett högre antal vektorer ökar beräkningstiden men också sannolikheten för ett korrekt resultat.

2.1.2 Gilbert-Johnson-Keerthi

För att inte göra avkall på noggrannheten har en annan metod, GJK [1, s. 30] *(Gilbert-Johnson-Keerthi)*, använts för detta projekt. Eftersom metoden kan utesluta fler möjliga kollisioner kommer senare beräkningar kräva mindre tid. GJK bygger på Minkovskidifferensen mellan de två objekten, definierad enligt (1).

$$\chi_1 \ominus \chi_2 = \{x_1 - x_2 | x_1 \in \chi_1, x_2 \in \chi_2\} \quad (1)$$

Polygonen som Minkovskidifferensen renderar i innehåller origo om, och endast om, de två objekten korsar varandra.

2.2 Optimering

3 Kollisionslösning

4 Kollisionsrespons

5 Krafter i impulsbaserade system

5.1 Kraftgeneratorer

Hittills har endast impulser mellan objekt diskuterats. Impulser är emellertid en förenkling av vad som egentligen sker i

stötar och kontakt. En kollision mellan två objekt orsakar kompression hos materialen i respektive objekt vilket i sin tur ger upphov till statiska, motriktade, krafter (kraft/motkraft enligt Newton 3 [referens?]). Dessa krafter verkar inte momentant utan under ett tidsspann. Impulser, som antas vara momentana, kan därför relateras till krafter enligt (2).

$$J = \int_{\Delta t} F \, dt \tag{2}$$

Vissa fysikmotorer delar upp tiden i varje tidssteg i mindre delar och kan på så sätt erhålla en bättre approximation för vissa krafter. Dessa metoder kallas tidsderiverande. ...

5.2 Friktion

Den allra vanligaste friktionsmodellen är Coloumbs modell. Den baserar sig i att den friktionskraften som påverkar ett objekt är proportionell mot normalkraften mellan objektet och underlaget. Modellen är egentligen en sammanslagning av två modeller, statisk och dynamisk friktion. Där den statiska modellen används för objekt i vila och den dynamiska för objekt i rörelse.

$$f_d = \mu_d \cdot N \tag{3}$$

$$f_s = \mu_s \cdot N \tag{4}$$

5.3 Fjäderkrafter

6 Resultat

7 Diskussion

Referenser

[1] Vella Colin, *Gravitas: An extensible physics engine framework using object-oriented and design pattern-driven software architecture principles*. 2008.

[2] Millington Ian, *Game Physics Engine Development*. 2007.