UMEÅ UNIVERSITET
Institutionen för datavetenskap
Projektrapport

January 25, 2010

Projekt

Emergenta system, 5DV017

Genetisk animation; rörelsemönster i 2D mha neuralt nät och genetisk algoritm

Andreas Jakobsson, dit06ajs@cs.umu.se
Anton Johansson, dit06ajn@cs.umu.se
Erik Rönnberg, dit06erg@cs.umu.se
Ludvig Widman, dit06lwn@cs.umu.se
Ragnar Asplund, dit04rad@cs.umu.se

Källkod: `http://github.com/blastura/genetic-reanimation/tree/master`
Projektsida: `http://sites.google.com/site/geneticreanimation/`

Handledare
Jonny Pettersson, jonny@cs.umu.se Anders Broberg, bopspe@cs.umu.se

Abstract

This report describes a simple simulation for evolving movements of a worm over time and generations using a genetic algorithm. A simplified and modified 2D version of Karl Sims work with Evolving Virtual Creatures is created. Creatures are created with a random genotype, the creatures movements are simulated and patterns arise from a Hopfield inspired neural network. The genotype is simply all the weights of the neural network which evolve over generations from a Genetic Algorithm. The fitness for each creature is determined by how far they are able to move within a specified number of steps in the simulation. A large number of tests are made and a rolling or a combination of rolling and jumping movement seems to be the most effective pattern for forward movement in our simulation.

dit06ajs@cs.umu.se, dit06ajn@cs.umu.se
dit06erg@cs.umu.se, dit06lwn@cs.umu.se, dit04rad@cs.umu.se

Innehåll

1 Introduktion **1**
1.1 Rapportens uppbyggnad . . . 1
1.2 Karl Sims . . . 1
1.3 Hugo de Garis . . . 1
1.4 Syfte och frågeställningar . . . 1

2 Metodbeskrivning **2**
2.1 Varelser . . . 2
2.1.1 Uppbyggnad av varelse . . . 2
2.1.2 Sensorer . . . 2
2.1.3 Effektorer . . . 2
2.1.4 Neuralt nät . . . 2
2.2 Simulering . . . 3
2.3 Genetisk algoritm . . . 3

3 Resultat **4**
3.1 Rörelsemönster . . . 4
3.2 Fitness . . . 4

4 Diskussion **5**
4.1 Slutsatser . . . 5
4.1.1 Fungerande rörelsemönster? . . . 5
4.1.2 Intressanta och effektiva? . . . 6
4.1.3 Övriga slutsatser . . . 6
4.2 Begränsningar . . . 6
4.3 Framtida arbete . . . 6

dit06ajs@cs.umu.se, dit06ajn@cs.umu.se
dit06erg@cs.umu.se, dit06lwn@cs.umu.se, dit04rad@cs.umu.se

1 Introduktion

I detta projekt har vi undersökt möjligheten att utveckla rörelsemönster för en enkel varelse med hjälp av genetiska algoritmer (GA). Varelsen har ett neuralt nät som hjärna och har formen av en ledad mask. En enkel modell har implementerats och försök har gjorts i denna modell för att undersöka om GAn kunde hitta rörelsemönster och hur bra dessa blir.

1.1 Rapportens uppbyggnad

Nedan introduceras ämnet kort och frågeställningarna läggs fram. Den modell vi byggt för att besvara våra frågeställningar presenteras under Modellbeskrivning. Därefter visas de resultat vi fått från våra körningar i modellen under Resultat. Slutligen drar vi slutsatser kring resultaten, försöker besvara frågeställningarna och resonerar kring framtida studier under Diskussion.

1.2 Karl Sims

Karl Sims utforskar i Evolving Virtual Creatures [Sim94] både hur varelsers form och deras beteenden kan utvecklas med hjälp av GA. Varelserna har sensorer och effektorer som är kopplade till in- och utnoder i ett neuralt nät.

I sina försök använder Sims en GA för att optimera både struktur och vikter i det neurala nätet. Det nät han använder har i varje nod en av många matematiska funktioner, exempelvis: sinus, summa, produkt, max, osv. Sims lät sina försök köra mellan 50 och 100 generationer och optimerade mot olika typer av beteenden som ex vandring, simmande och hoppande rörelser.

1.3 Hugo de Garis

Hugo de Garis forskning i *Genetic Programming: Evolutionary Approaches to Multistrategy Learning* [GT94] inriktar sig i störst utsträckning mot utveckling av hårdvara med hjälp av genetiska algoritmer. Han beskriver metoder som shaping (att i olika steg ändra definitionen för fitness) och talar om ”utvecklingsbara systemmed de avseendet att vissa system inte kan ge önskvärt beteende med hjälp av genetiska algoritmer. Han nämner att de inte finns något sätt att definitivt avgöra vilka system som är utvecklingsbara med fasta kriterier.

Garis har utvecklat en simulerad ödla som han kallar *LIZZY* [GT94]. Den virtuella ödlan *LIZZY* känner sin omvärld med två antenner och tar därigenom emot karaktäristiska signaler från tre sorters varelser i dess omgivning; Mate, Prey och Predator. Med hjälp av ett neuralt nät och en genetisk algoritm lär sig sedan *LIZZY* att söka upp Prey och Mate för att sedan vidta passande åtgärder och att fly undan Predators.

1.4 Syfte och frågeställningar

Inledningsvis valde vi ämnet ”Rörelsemönster i 2D med hjälp av Genetiska algoritmer”. Vi ville utforska ämnet GA lite djupare och tyckte Sims försök var en bra utgångspunkt.

I detta arbete har vi försökt återskapa en förenklad variant på Sims arbete. Istället för tre dimensioner använder vi bara två. Dessutom använder vi bara en fix varelse och en enklare hjärna.

Vi vill se om denna förenklade modell kommer hitta fungerande rörelsemönster. Hur effektiv den är på detta och hur intressant resultatet blir kommer också studeras.

2 Metodbeskrivning

Vi har utvecklat ett system som utvecklar kroppars rörelsemönster med hjälp av en genetisk algoritm. Kroppen simuleras i en fysikmotor och rörelsen bestäms av ett neuralt nät där indata kommer från simuleringsvärlden och utdata ansätts till varelsens effektorer. Fitnessfunktionen beräknas enligt längden varelsen har tagit sig under sin del av simuleringen, ett förinställt antal steg.

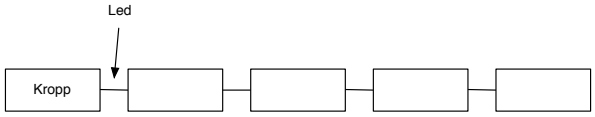
2.1 Varelser

I följande stycke presenteras en översikt av hur en varelse är uppbyggd och hur den reagerar/agerar på omvärlden.

2.1.1 Uppbyggnad av varelse

Varelsen består av en mask med 4 leder och 5 kroppsdelar, se figur 1. Lederna har ingen begränsning på hur de kan böjas, men kraft kan inte ges att böja leden över 180 grader. Varje kroppsdel har en sensor som ger hjärnan indata om dess y-position i förhållande till delen utmed.

Att en maskliknande varelse har valts beror på att kroppen är relativ enkel i sin sammansättning och komplexiteten sjunker således. Kroppsdelarnas vikt, ledernas typ och ledernas rörelsefrihet har valts efter eget tycke då varelsen ses som ett inledande exempel på hur systemet kan fungera. Valet av indata för varelsen grundas i en önskan om ledernas vetskap om varandra.



Figur 1: Representation av varelse.

Källkod för den implementerade varelsen finns att hitta på projektsidan.

2.1.2 Sensorer

Varelsens sensordata räknas ut som differensen mellan kroppdelarnas y-position. Antalet sensorer/indata blir ett mindre än antalet kroppsdelar då de räknas ut mellan intilliggande kroppsdelar.

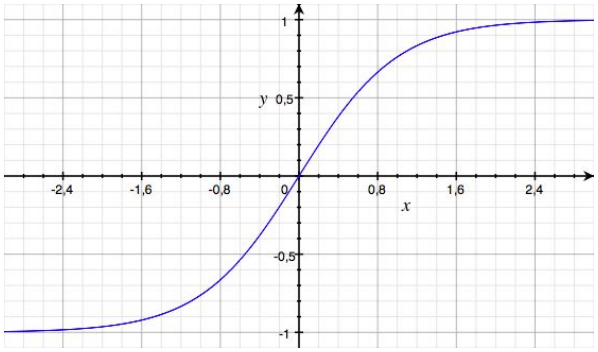
2.1.3 Effektorer

Varelsens effektorer utgörs i implementationen av en viss typ av leder som krafter kan appliceras på. Efter att varelsens hjärna har bearbetat indata så hämtas utdata som nu kommer att vara mellan -1 och 1. Denna utdata multiplicaras med π för att få en vinkel mellan $-\pi$ och π, det resulterande värdet påverkar leden, som i sin tur påverkar de två kroppar leden är fäst vid.

2.1.4 Neuralt nät

Varelsens hjärna består av ett neuralt nät som styr dess beteende. Det neurala nätet är vagt implementerat enligt Hopfields modell för neurala nät, se kapitel 13 i boken *Neural networks: a systematic introduction* [Roj96]. Varje nod i det neurala nätet har bågar till och från alla andra noder. Varje båge har en vikt.

För att mata nätet med indata tas värden emot och körs genom en sigmoid funktion



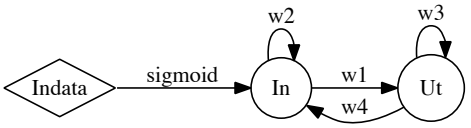
Figur 2: Plot av sigmoidfunktionen.

$1/\frac{1}{1+e^{-x}}$. Denna funktion är modifierad för att ge resultat mellan -1 och 1. Sigmoid-funktionen är uppritad i figur 2. Javakod för sigmoid-funktionen som används visas i kodsnutt 1. Vid varje simuleringssteg i hjärnan beräknas ett värde i varje nod. Värdet beräknas som en summa av värdet från varje båge gånger dess vikt. På denna summa appliceras sedan sigmoidfunktionen så värdet hamnar mellan -1 och 1.

Det görs ingen skillnad bland noderna i nätetet med avseende på vilka som tar emot och ger ifrån sig in- och utdata. Det neuralal nätet tar emot indata i from av en vektor, positionen värdena har i vektorn avgör vilken nod i nätet som kommer att ta emot datan. Samma position i vektorn kommer alltid att motsvara samma nod i det neurala nätet. Utdata fås ur det neurala nätet genom att alla noders värden returneras som en vektor. Behövs inte all utdata är det upp till den anropande metod att välja ut vilka nod-värden som ska användas.

Kodsnutt 1 Sigmoid (java)

```
2 / (1 + Math.pow(Math.E, (-2 * x))) - 1
```



Figur 3: Neuralt nätverk med två noder.

Källkoden för implementationen av det neurala nätet finns att hitta på projektsidan.

2.2 Simulering

Fysiksimuleringen använder biblioteket *phys2d* [Gla17] för själva fysiken. Detta bibliotek är skrivet av Kevin Glass och är en Java-variant av Erin Catto's plattform *box2d* [Cat09]. Denna platform valdes på grund av att den bygger på Java och gruppen besitter kompetens inom språket. I simuleringen har ett markplan skapats och på detta en varelse. En vägg skapades även till vänster om masken för att förhindra att den försvinner bakåt ur fönstret.

2.3 Genetisk algoritm

Genotypen i varelsen kodar för vikter i det neurala nät som utgör dess hjärna. För att evaluera fitness för varje individ skapas ett nät med de vikter genotypen anger, detta får sedan under en begränsad tid styra en fast kropp i en fysiksimulering. Fitness beräknas på hur långt kroppen lyckats röra sig.

För selektion används en variant på tournament selection [Wik27]. Vid crossover sker en korsning mellan två individer, genotypen delas på en slumpmässig plats och barnet får en del av vardera förälder. Mutation

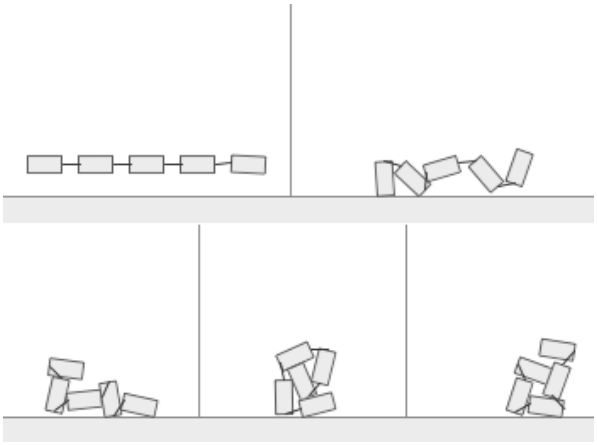
är implementerad så den individ som muteras har en bestämd sannolikhet att byta ut en vikt(kromosom) mot en ny slumpmässig, detta görs för alla vikter. Hela den genetiska algoritmen beskrivs nedanför.

1. Skapa en ny tom population
2. Kopiera över den individ med högst fitness från föregående generation till den nya populationen
3. Upprepa till den bestämda andel crossover till nya generationen är utförd
   (a) Välj ut två grupper om tre individer slumpmässigt
   (b) Utför crossover på den individ med högst fitness från respektive grupp och lägg över deras barn till den nya populationen
4. Upprepa till den nya populationen har samma storlek som den föregående generationen
   (a) Välj grupp om tre individer slumpmässigt
   (b) Kopiera över den individ med högst fitness från de tre
5. Mutera varje individ utom den första med högst fitness

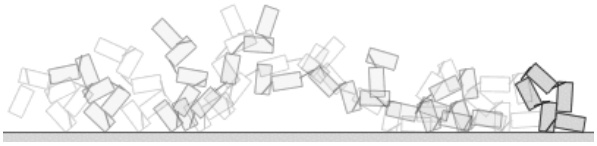
Källkoden till den genetiska algoritmen som används finns att hitta på projektsidan.

3 Resultat

Följande avsnitt presenterar resultat i form av typiska rörelsemönster och typisk utveckling för fitness.



Figur 4: Ett typiskt rörelsemönster där masken rullar fram.



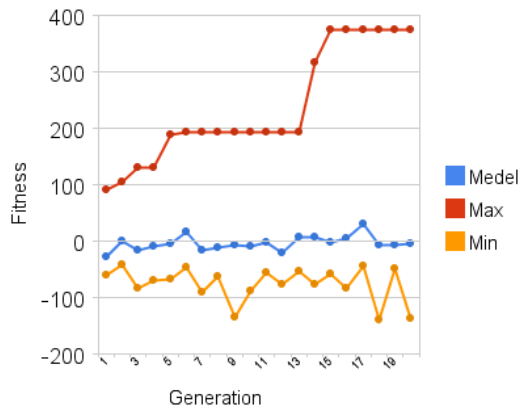
Figur 5: Hopp-rullande rörelse. Dessa rörelser är ofta lite effektivare än bara rullande.

3.1 Rörelsemönster

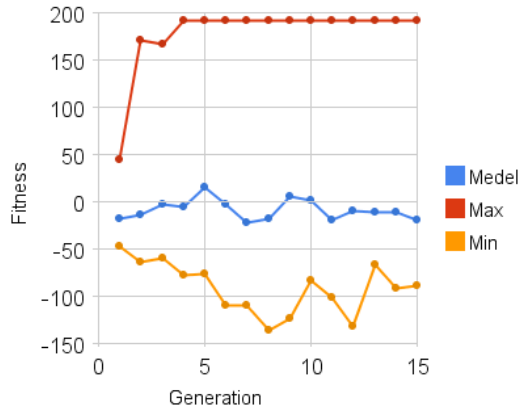
För att undersöka vilka rörelsemönster som får hög fitness har vi gjort ett stort antal testkörningar och spelat in den bästa varelsen i varje generation. Därefter har typiska rörelser med bra fitness valts ut och presenterats i form av bilder. Ett vanligt rörelsemönster är att masken rullar fram, det visas i figur 4. Ett annat rörelsemönster, hopp-rullande, som hade något högre fitness visas i figur 5.

3.2 Fitness

För att undersöka hur fitness utvecklas har vi har testat att köra simuleringen ett fler-

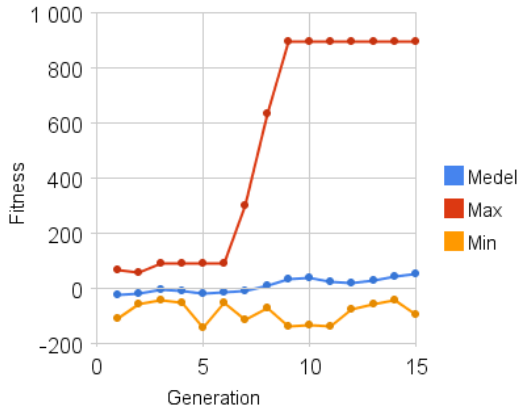


Figur 6: Fitnessens utveckling över 20 generationer vid en godtycklig testkörning.



Figur 7: Fitnessens utveckling över 15 generationer vid en godtycklig testkörning.

tal gånger och låta den köra mellan 15 och 20 generationer. Därefter plottat medel, max och min för fitness i varje generation. Detta syns i figur 6- 8. De olika figurerna skiljer sig ganska mycket, men visar ändå karakteristiska egenskaper vi sett i de flesta av våra körningar så som att max-fitness ökar etappvis. Oftast stiger medelvärdet sakta över tiden.



Figur 8: Fitnessens utveckling över 15 generationer vid en godtycklig testkörning.

4 Diskussion

I denna diskussionsdel presenteras först slutsatser vi dragit från resultatet. Därefter påpekar vi de brister och begränsningar som finns i arbetet och modellen. Slutligen resoneras det kring hur man skulle kunna bygga vidare på arbetet.

4.1 Slutsatser

Här nedan försöker vi besvara de frågeställningar vi presenterat i inledningen. Andra slutsatser som kan dras redovisas även lite längre ner.

4.1.1 Fungerande rörelsemönster?

Ett flertal olika rörelsemönster hittades av vår algoritm. Många av dem var bara spasmiska eller icke-cykliska (dvs varelsen fastnar och stannar). Några rörelsemönster fungerade väl och förflyttade masken effektivt framåt.

4.1.2 Intressanta och effektiva?

Man kan konstatera att vår enkla modell fungerar för att hitta rörelsemönster, men resultatet blir inte lika intressant som för Sims. Detta kan dock delvis förklaras med att en mask inte kan röra sig på så många olika sätt.

Det rörelsemönster som gett mest framgång i våra studier är ett rullande eller rullhoppande beteende. En förklaring till att detta mönster är vanligt i våra simuleringar, men inte i verkligheten kan vara att 2D-maskar inte kan tippa i djupled. Det är även tänkbart att våra simulerade maskar är starkare än naturliga motsvarigheter. I simuleringarna har det visat sig att rörelsemönsterna vi får ut är extremt beroende av de val som gjorts i fysiksimuleringen och uppbyggnaden av varlsen i denna simulering.

4.1.3 Övriga slutsatser

Vanligt förekommande resultat av simuleringarna är att varelsers fitness går upp väldigt snabbt i början och ligger sedan stabilt. Utvecklingen hos populationen kan ske i etapper istället för kontinueligt. Detta kan liknas vid naturlig evolution som också kan ske etappvis.

4.2 Begränsningar

Ett stort problem med vår lösning är att fysiksimuleringen inte är helt deterministisk. Simuleras samma individ flera gånger kan man få olika resultat. Detta medför även att filmerna inte alltid visar samma rörelse som fitness beräknats på.

Ibland försvinner masken ur simuleringen. Vi misstänker någon bugg i fysiksimuleringen som får masken att upphöra att existera. Fitness för maskar som försvunnit har exkluderats ur resultatet.

4.3 Framtida arbete

Om man vill bygga vidare på detta arbete bör man nog initialt hitta lösningar till begränsningarna. Det kanske en annan fysikmotor för att kringgå några av begränsningarna.

Intressanta saker att utforska vore att låta hjärnan få fler sorters indata. Exempelvis om en kroppsdel rör i marken, någon sorts färdriktning och vinklar mellan kroppsdelar är indata man skulle kunna testa.

För att återskapa en varelses rörelsemönster på ett mer *naturligt* sätt hade en noggrannare studie av vald varelse (exempelvis mask) behövts. Med information om den naturligt förekommande varelsens effektorer för rörelse skulle snarliga effektorer behövt implementeras i fysiksimuleringen. Fysikmotorn vi valt har ingen riktigt tillfredställande led/effektor för att simulera de naturliga rörelsemönster vi tänkt oss. Vissa artefakter kan tänkas uppkommit som en bieffekt av den specifika led och fysikmotor vi valt.

Det vore även intressant att undersöka hur metoden fungerar på ett mer komplext djur.

Sims använde i sina försök sinus och andra komplexa funktioner i sitt neurala nät. Det vore intressant att se om det skulle hjälpa hjärnan om den hade en sinusvåg som indata.

Referenser

[Cat09] Erin Catto. Box2d physics engine. World Wide Web electronic publication, 2009.

[Gla17] Kevin Glass. Phys2d - the 2d game physics engine in java. World Wide Web electronic publication, 2009-03-17.

[GT94] Hugo De Garis and Electrotechnical Lab Tsukuba. Genetic programming: Evolutionary approaches to multistrategy learning. In *Vol.4, R.S. Michalski & G. Tecuci (eds)*, pages 549–577. Morgan Kaufmann, 1994.

[Roj96] Raúl Rojas. *Neural networks: a systematic introduction.* Springer-Verlag, Berlin, 1996.

[Sim94] Karl Sims. Evolving virtual creatures. *Computer Graphics*, pages 15–22, 1994.

[Wik27] Wikipedia. Tournament selection. World Wide Web electronic publication, 2009-03-27.