UMEÅ UNIVERSITET
Institutionen för datavetenskap
Laborationsrapport

2 februari 2009

Laboration 2

Emergenta system, VT-09

Boids

Andreas Jakobsson, dit06ajs@cs.umu.se
Anton Johansson, dit06ajn@cs.umu.se

`~dit06ajn/edu/emergenta-system/lab2/src`

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

Andreas Jakobsson, dit06ajs@cs.umu.se
Anton Johansson, dit06ajn@cs.umu.se 2 februari 2009

Innehåll

Andreas Jakobsson, dit06ajs@cs.umu.se
Anton Johansson, dit06ajn@cs.umu.se 2 februari 2009

1 Problemspecifikation

Laborationen gick ut på att göra ändringar i en befintlig NetLogo[1] modell som imiterar flockbete. Modellen utvecklades av Craig Reynolds på 80-talet och varje individ, så kallad *boid*, följer tre enkla regler:

- Undvik kollision med grannar.
- Håll samma hastighet och riktning som dina grannar.
- Försök ha en position så nära centrum av flocken som möjligt.

Denna modell ska utökas så att flockarna ska kunna undvika hinder i dess väg, och att de istället för att röra sig slumpvis omkring i miljön ska kunna röra sig mot ett gemensamt mål som bestäms av muspekarens position.

1.1 Frågor som ska behandlas

I problemspecifikationen finns följande frågor som denna rapport ska behandla.

Hinder:

- Vad krävs för att en flock ska kunna splittras upp när de undviker ett hinder och gå ihop till en samlad flock när hindret är passerat?
- Vilken metod valde ni och varför?
- Hur skulle er metod fungera med andra hinder än punktformade?
- Finns det någon gräns för hur stora hindren kan vara?

Mål:

- Modellen ska utvidgas till att inkludera ett gemensamt mål, alltså en punkt eller yta i världen som alla boids strävar efter att nå.
- Minimikravet är att man ska kunna ställa in målet genom att med muspekaren välja en punkt i världen
- Hur bibehålls flockbeteendet när ett mål finns?
- Hur stor vikt bör man lägga på mål, hinder och flockbeteende för att få ett beteende som ser naturligt ut?

[1] http://ccl.northwestern.edu/netlogo/

- Vilken sorts beteende anser ni är naturligt eller önskvärt?

Laborationsspecifikation finns i original på sidan:
`http://www.cs.umu.se/kurser/5DV017/VT09/lab/lab2.html`

2 Användarhandledning

Källkoden till implementationen Flocking.nlogo som diskuteras i denna rapport finns att hitta på:

`~dit06ajn/edu/emergenta-system/lab2/src`

Öppna filen i NetLogo för att köra den.

2.1 Förklaring av användargränssnittet

Nedan följer en förklaring av de knappar och reglage som förekommer i användargränssnittet:

- **population** - antalet individer i världen.
- **nr-of-obstacles** - antalet hinder i världen.
- **setup** - denna knapp initierar parametrarna till världen.
- **go** - denna knapp sätter igång animeringen i världen. Setup måste köras en gång innan denna knapp får användas.
- **vision-radius** - antalet rutor en individ kan se.
- **vision-angle** - vinkeln för en individs synfält angett i grader.
- **minimum-seperation** - det minsta avståndet en individ vill ha till sina grannar angett i rutor.
- **max-align-turn** - den maximala vinkeln en individ kan svänga för att komma i samma färdriktning som sina grannar.
- **max-cohere-turn** - den maximala vinkeln en individ kan svänga för att komma närmare mittpunkten av sina grannar.
- **max-seperate-turn** - den maximala vinkeln en individ kan svänga för att komma längre bort från sin närmsta granne.

- **max-avoidance-turn** - den maximala vinkeln en individ kan svänga för att undvika ett hinder.
- **moving-obstacles?** - om denna är på förflyttar sig hindren i slumpmässig riktning.
- **follow-mouse?** - om denna är på försöker individerna att följa efter muspekaren om den befinner sig inom världens gränser.
- **max-goal-turn** - den maximala vinkel en individ får använda sig av för att svänga mot muspekaren.

Avstånd är mätt i antal rutor i världen, vinklar är mätt i grader.

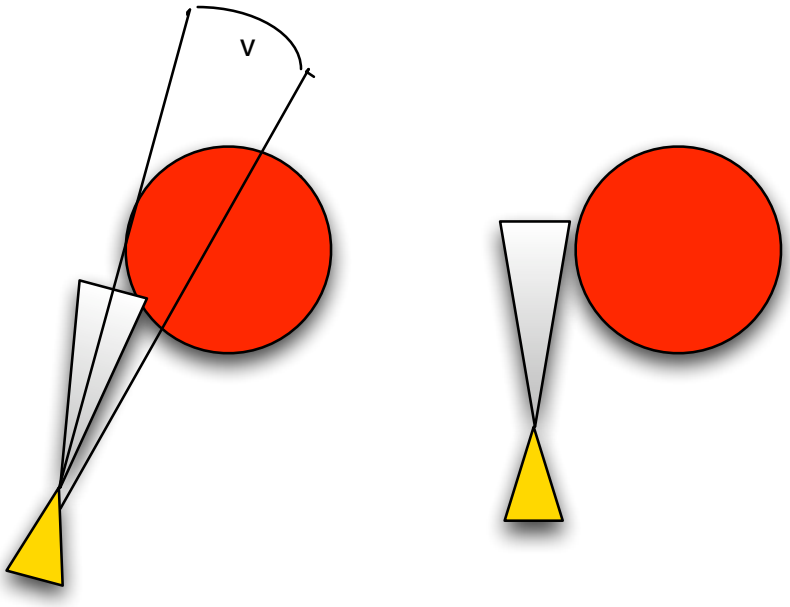
3 Algoritmbeskrivning

Nedan avsnitt beskriver de algoritmer som implementerats för att införa hinderavvikelse och målsökande i modellen.

3.1 Hinder

För att flockarna ska kunna undvika hinder har det införts nya agenter härefter refererade som *hinder*. Varje individ fått en ny procedur *obstacle-avoidance* som utförs före att de har utfört sitt flock-beteende.

Metoden *steer-to-avoid* implementerades där hinderundvikande löses genom att varje individ letar efter det närmaste hinder som befinner sig i en kon i deras färdriktning, se figur 1. Konens storlek bestäms av variablerna *vision-radius* och *vision-angle*. Om ett hinder upptäckts i konen svänger individen bort från denna med en vinkel som beräknas enligt *max-avoidance-turn* delat med avståndet till hindret. Detta innebär att enheten vid tidig upptäckt av hinder svänger minimalt, när enheten närmar sig hindret ökar vinkeln den försöker svänga bort med. Den initiala vinkeln, se v i figur 1, individen har mot målet i förhållande till sin färdriktning används för att bestämma vilket håll om hindret individen ska välja för att förhindra kollision. Om v är negativ svänger individen åt vänster, annars åt höger.



Figur 1: Kollisionshantering

3.2 Mål

För att få flockarna att söka sig mot ett gemensamt mål implementerades proceduren *find-goal* som räknar ut vinkeln mot målet, representerat av muspekarens position. Individerna svänger mot målet i varje tidssteg med maximalt tillåten vinkel, *max-goal-turn*.

4 Strategi för testning

Under laborationens gång används trial-and-error för att testa ifall koden genererar önskade beteenden. För att få svar på frågorna görs försök till att upprepa samma situation flera gånger med olika parameterinställningar för att subjektivt bedömma hur parametrarna påverkar beteenden. För återskapa beteenden som diskuteras i rapporten ombeds testkörningar med programmet där parametrar ändras på det sätt som beskrivs i avsnitt 5.

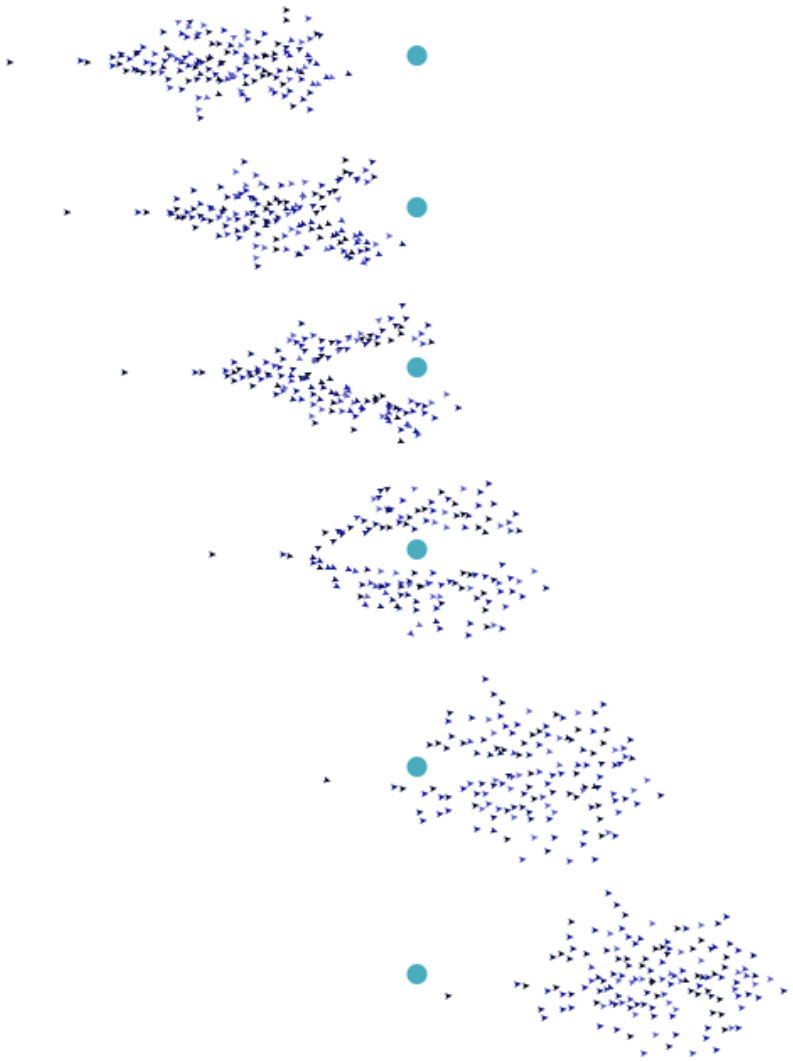
Några skärmdumpar presenteras som diskussionsunderlag till beteenden.

5 Reflektioner

Nedan avsnitt beskriver reflektioner som gjorts med avseende på frågorna från problemspecifikationen.

5.1 Hinder

För att en flock ska splittras vid ett hinder och sedan samlas ihop igen på andra sidan hindret krävs att individerna har tillräckligt stor sikt för att kunna upptäcka individer som tvingades välja en annan väg runt hindret. Anledningen till att individerna återförenas i en flock efter de har stött på ett hinder är deras *coherence*-beteende gör att de strävar mot en gemensam mittpunkt för flocken. Se figur 2 för en sekvens där en flock möter ett hinder, inställningar för simulering finns i bilaga A.1.



Figur 2: Kollisionshantering för flock

Många parametrar spelar in vid återförening av flocken efter hinder. Om vinkeln mellan de två flockarna är stor samtidigt som individernas *cohere-turn*-parameter är låg kan flockarna återförening utebli även fast de initialt såg varandra när hindret forcerats. Detta eftersom de inte hinner svänga mot varandra innan deras synfällt upphör att sammanfalla.

Vi valde att implementera ett *steer-to-avoid*-beteende, se avsnitt 3.1, till skillnad från *force-field*-beteende som skulle innebära att ett hinder stöter bort enheten oberoende av enhetens riktning. Beteendet *steer-to-avoid* innebär att en individ kan röra sig parallellt med ett hinder utan att påverkas av

det. Detta är viktigt för återförenande av flockar efter hinder. Notera dock att sätts vinkeln *vision-angle* till ett stort värde nära 360 grader kommer denna metod att likna *force-field*-metoden i och med att individen påverkas av hinder som inte enbart ligger framför dess färdriktning.

I nuläget är hindret implementerat som en cirkel med en diameter motsvarande storleken. När vinkel mellan en individ och ett hinder beräknas betraktas hindrets mittpunkt och hänsyn kommer inte tas till hindrets storlek. Däremot beräknas avståndet till hindret som mittpunkt på hindret - radien för att få ett hinder med en cirkulär yta.

Eftersom vår modell använder sig av den inbyggda metoden *in-cone* för att upptäcka hinder i en individs synfält används hindrens mittpunkt vid detektion. Detta innebär en begränsning som gör att storleken på hindrena inte får överstiga en enhets parametervärde på *vision-radius*. Om så är fallet kommer individen att krocka redan innan den har upptäckt hindret den krockar med.

5.2 Mål

När modellen utökades för att få individerna att söka sig mot ett gemensamt mål blev det svårare att avskilja vad som är flockbeteende och vad som är målsökarbeteende. Eftersom alla strävar mot samma mål får närliggande individer samma riktning. Om man minskar på parametern som anger en individs maxsväng mot mål, *max-goal-turn*, upplevs flockbeteendet starkare eftersom individernas svängar för flockbeteende får högre dominans än individernas målsökande svängar.

I våra testkörningar har vi noterat att balansen mellan mål- och flockbeteende är avgörande för hur naturlikt simuleringen verkar. Med naturlikt menar vi flockbeteende som kan ses bland fåglar och fiskar. Vi anser då att det bör läggas störst vikt vid att undvika hinder, det vill säga att i simuleringen tillåta individerna en stor vinkel på *max-avoidance-turn*. Därefter bör det läggas större vikt på flockbeteende än på målsökarbeteende. Vi gör jämförelsen med flyttfåglar där målsökandet är något som kontinuerligt sker över en större tidsperiod än flockbeteendet, som är en mer aktiv process.

Andreas Jakobsson, dit06ajs@cs.umu.se
Anton Johansson, dit06ajn@cs.umu.se 2 februari 2009

A Källkod

Härefter följer utskrifter från källkoden och andra filer som hör till denna laboration

A.1 Paremeterinställningar för figur 2

- **population** - 140
- **nr-of-obstacles** - 1
- **vision-radius** - 20
- **vision-angle** - 50
- **minimum-seperation** - 1
- **max-align-turn** - 2.75
- **max-cohere-turn** - 2.75
- **max-seperate-turn** - 1
- **max-avoidance-turn** - 90
- **moving-obstacles?** - off
- **follow-mouse?** - off
- **max-goal-turn** - 2.5

A.2 Flocking.nlogo

```
globals [
  turns
  align-turns
  cohere-turns
  separate-turns
]

breed [ obstacles obstacle ] ;; Breed to avoid
breed [ boids boid ] ;; Boids

boids-own [
  flockmates          ;; agentset of nearby turtles
  nearest-neighbor    ;; closest one of our flockmates
```

```
  obstacles-in-vision ;; obstacles in field of vision
  nearest-obstacle    ;; the nearest obstacle
]

to setup
  clear-all
  set-default-shape boids "default"
  set-default-shape obstacles "circle"

  create-boids population
    [ set color yellow - 2 + random 7  ;; random shades look nice
      set size 1.5  ;; easier to see
      setxy random-xcor random-ycor ]

   create-obstacles nr-of-obstacles
   [ set color red
     set size 2 + random 5
     setxy random-xcor random-ycor
   ]
   set-current-plot "turns"
   set-plot-y-range 0 10
end

to go
if moving-obstacles? [
 ask obstacles [
    fd 1
    rt random 10
    lt random 10
  ]
 ]
;  ask patches [
;    set pcolor black
;  ]
  if follow-mouse? and mouse-inside? [
    ask boids [ find-goal ]
  ]
  ask boids [ flock ]
  ask boids [ obstacle-avoidance ]

  ;; the following line is used to make the turtles
  ;; animate more smoothly.
  repeat 5 [ ask boids [ fd 0.2 ] display ]
  ;; for greater efficiency, at the expense of smooth
  ;; animation, substitute the following line instead:
  ;;   ask turtles [ fd 1 ]
  set-current-plot-pen "default"
  plot turns
```

```
  reset-globals
  tick
end

to reset-globals
  set turns 0
  set align-turns 0
  set cohere-turns 0
  set separate-turns 0
end

;;; OBSTACLE AVOIDANCE

to obstacle-avoidance
  find-obstacles
  if any? obstacles-in-vision
    [ find-nearest-obstacle
      avoid-obstacle
    ]
end

to find-obstacles
  set obstacles-in-vision obstacles in-cone vision-radius vision-angle
;  ask patches in-cone vision-radius vision-angle [
;    set pcolor white
;  ]
end

to find-nearest-obstacle
  set nearest-obstacle min-one-of obstacles-in-vision [distance myself]
end

to avoid-obstacle
  let angle towards nearest-obstacle
  ;;let dangle subtract-headings angle heading
  ;;show word "avoid-obstacle: vinkeln var " angle
  ;  show word "heading: " heading
  ;  show word "towards nearest-obstacle: " angle
  ;  show word "dangle: " dangle
  ;  show "  "
  turn-away-from-obstacle angle max-avoidance-turn
end


;;; FLOCKING

to flock  ;; turtle procedure
  find-flockmates
  if any? flockmates
```

```
    [ find-nearest-neighbor
      ifelse distance nearest-neighbor < minimum-separation
        [ separate ]
        [ align
          cohere ] ]
end

to find-flockmates  ;; turtle procedure
  set flockmates other boids in-radius vision-radius
end

to find-nearest-neighbor ;; turtle procedure
  set nearest-neighbor min-one-of flockmates [distance myself]
end

;;; SEPARATE

to separate  ;; turtle procedure
  turn-away ([heading] of nearest-neighbor) max-separate-turn
end

;;; FIND GOAL

to find-goal
  let angle towardsxy mouse-xcor mouse-ycor
  turn-towards angle max-goal-turn
end

;;; ALIGN

to align  ;; turtle procedure
  turn-towards average-flockmate-heading max-align-turn
end

to-report average-flockmate-heading  ;; turtle procedure
  ;; We can't just average the heading variables here.
  ;; For example, the average of 1 and 359 should be 0,
  ;; not 180.  So we have to use trigonometry.
  ;; Theoretically this could fail if both sums are 0
  ;; since atan 0 0 is undefined, but in practice that's
  ;; vanishingly unlikely.
  report atan sum [sin heading] of flockmates
              sum [cos heading] of flockmates
end

;;; COHERE

to cohere  ;; turtle procedure
  turn-towards average-heading-towards-flockmates max-cohere-turn
```

```
end

to-report average-heading-towards-flockmates  ;; turtle procedure
  ;; "towards myself" gives us the heading from the other turtle
  ;; to me, but we want the heading from me to the other turtle,
  ;; so we add 180
  report atan mean [sin (towards myself + 180)] of flockmates
              mean [cos (towards myself + 180)] of flockmates
end

;;; HELPER PROCEDURES

to turn-towards [new-heading max-turn]  ;; turtle procedure
  turn-at-most (subtract-headings new-heading heading) max-turn
end

to turn-away [new-heading max-turn]  ;; turtle procedure
  turn-at-most (subtract-headings heading new-heading) max-turn
end

to turn-away-from-obstacle [heading-to-obstacle max-turn]
  let angle (subtract-headings heading heading-to-obstacle)
  let distance-to-obstacle distance nearest-obstacle - [size / 2] of nearest-obstacle
  ;; Right turn, or straight on
  ifelse angle >= 0 [
    set angle (max-turn / distance-to-obstacle)
  ][
  ;; Left turn
    set angle (-(max-turn  / distance-to-obstacle))
  ]
  ;;show word "avoiding with angle: " angle
  turn-at-most angle max-turn
end

;; turn right by "turn" degrees (or left if "turn" is negative),
;; but never turn more than "max-turn" degrees
to turn-at-most [turn max-turn]  ;; turtle procedure
  ifelse abs turn > max-turn
    [ ifelse turn > 0
        [ set turns turns + max-turn
          rt max-turn ]
        [ set turns turns + max-turn
          lt max-turn ] ]
    [ set turns turns + abs turn
      rt turn ]
end


; *** NetLogo 4.0.4 Model Copyright Notice ***
```

```
;
; This model was created as part of the project: CONNECTED MATHEMATICS:
; MAKING SENSE OF COMPLEX PHENOMENA THROUGH BUILDING OBJECT-BASED PARALLEL
; MODELS (OBPML).  The project gratefully acknowledges the support of the
; National Science Foundation (Applications of Advanced Technologies
; Program) -- grant numbers RED #9552950 and REC #9632612.
;
; Copyright 1998 by Uri Wilensky.  All rights reserved.
;
; Permission to use, modify or redistribute this model is hereby granted,
; provided that both of the following requirements are followed:
; a) this copyright notice is included.
; b) this model will not be redistributed for profit without permission
;    from Uri Wilensky.
; Contact Uri Wilensky for appropriate licenses for redistribution for
; profit.
;
; This model was converted to NetLogo as part of the projects:
; PARTICIPATORY SIMULATIONS: NETWORK-BASED DESIGN FOR SYSTEMS LEARNING
; IN CLASSROOMS and/or INTEGRATED SIMULATION AND MODELING ENVIRONMENT.
; The project gratefully acknowledges the support of the
; National Science Foundation (REPP & ROLE programs) --
; grant numbers REC #9814682 and REC-0126227.
; Converted from StarLogoT to NetLogo, 2002.
;
; To refer to this model in academic publications, please use:
; Wilensky, U. (1998).  NetLogo Flocking model.
; http://ccl.northwestern.edu/netlogo/models/Flocking.
; Center for Connected Learning and Computer-Based Modeling,
; Northwestern University, Evanston, IL.
;
; In other publications, please use:
; Copyright 1998 Uri Wilensky.  All rights reserved.
; See http://ccl.northwestern.edu/netlogo/models/Flocking
; for terms of use.
;
; *** End of NetLogo 4.0.4 Model Copyright Notice ***
@#$#@#$#@
GRAPHICS-WINDOW
250
10
1064
445
100
50
4.0
1
10
1
```

```
1
1
0
1
1
1
-100
100
-50
50
1
1
1
ticks

CC-WINDOW
5
571
1295
666
Command Center
0

BUTTON
39
93
116
126
NIL
setup
NIL
1
T
OBSERVER
NIL
NIL
NIL
NIL

BUTTON
122
93
203
126
NIL
go
T
1
T
```

```
OBSERVER
NIL
NIL
NIL
NIL

SLIDER
8
10
231
43
population
population
1.0
1000.0
140
1.0
1
NIL
HORIZONTAL

SLIDER
10
249
243
282
max-align-turn
max-align-turn
0.0
20.0
2.75
0.25
1
degrees
HORIZONTAL

SLIDER
10
283
243
316
max-cohere-turn
max-cohere-turn
0.0
20.0
1.75
0.25
1
degrees
```

```
HORIZONTAL

SLIDER
11
318
244
351
max-separate-turn
max-separate-turn
0.0
20.0
1
0.25
1
degrees
HORIZONTAL

SLIDER
9
135
232
168
vision-radius
vision-radius
0.0
20.0
20
0.5
1
patches
HORIZONTAL

SLIDER
13
214
236
247
minimum-separation
minimum-separation
0.0
5.0
1
0.25
1
patches
HORIZONTAL

SLIDER
8
```

```
46
180
79
nr-of-obstacles
nr-of-obstacles
0
100
1
1
1
NIL
HORIZONTAL

SLIDER
10
356
243
389
max-avoidance-turn
max-avoidance-turn
0
180
90
0.5
1
degrees
HORIZONTAL

SLIDER
22
173
194
206
vision-angle
vision-angle
2
360
50
1
1
NIL
HORIZONTAL

SWITCH
17
467
165
500
follow-mouse?
```

```
follow-mouse?
0
1
-1000

SWITCH
14
431
188
464
moving-obstacles?
moving-obstacles?
1
1
-1000

SLIDER
27
524
227
557
max-goal-turn
max-goal-turn
0
20
2.5
0.5
1
degrees
HORIZONTAL

PLOT
1086
28
1286
178
turns
ticks
turns
0.0
10.0
0.0
10.0
true
false
PENS
"default" 1.0 0 -16777216 true

@#$#@#$#@
```

```
WHAT IS IT?
-----------
This model is an attempt to mimic the flocking of birds.  (The resulting motion also resembles

The birds follow three rules: "alignment", "separation", and "cohesion".  "Alignment" means th

The three rules affect only the bird's heading.  Each bird always moves forward at the same co


HOW TO USE IT
-------------
First, determine the number of birds you want in the simulation and set the POPULATION slider

The default settings for the sliders will produce reasonably good flocking behavior.  However,

Three TURN-ANGLE sliders control the maximum angle a bird can turn as a result of each rule.

VISION is the distance that each bird can see 360 degrees around it.


THINGS TO NOTICE
----------------
Central to the model is the observation that flocks form without a leader.

There are no random numbers used in this model, except to position the birds initially.  The f

Also, notice that each flock is dynamic.  A flock, once together, is not guaranteed to keep al

After running the model for a while, all of the birds have approximately the same heading.  Wh

Sometimes a bird breaks away from its flock.  How does this happen?  You may need to slow down


THINGS TO TRY
-------------
Play with the sliders to see if you can get tighter flocks, looser flocks, fewer flocks, more

You can turn off a rule entirely by setting that rule's angle slider to zero.  Is one rule by

Will running the model for a long time produce a static flock?  Or will the birds never settle


EXTENDING THE MODEL
-------------------
Currently the birds can "see" all around them.  What happens if birds can only see in front of

Is there some way to get V-shaped flocks, like migrating geese?

What happens if you put walls around the edges of the world that the birds can't fly into?
```

```
Can you get the birds to fly around obstacles in the middle of the world?

What would happen if you gave the birds different velocities?  For example, you could make bir

Are there other interesting ways you can make the birds different from each other?  There coul


NETLOGO FEATURES
----------------
Notice the need for the SUBTRACT-HEADINGS primitive and special procedure for averaging groups


CREDITS AND REFERENCES
----------------------
This model is inspired by the Boids simulation invented by Craig Reynolds.  The algorithm we u

To refer to this model in academic publications, please use:  Wilensky, U. (1998).  NetLogo Fl

In other publications, please use:  Copyright 1998 Uri Wilensky.  All rights reserved.  See ht
@#$#@#$#@
default
true
0
Polygon -7500403 true true 150 5 40 250 150 205 260 250

airplane
true
0
Polygon -7500403 true true 150 0 135 15 120 60 120 105 15 165 15 195 120 180 135 240 105 270 1

arrow
true
0
Polygon -7500403 true true 150 0 0 150 105 150 105 293 195 293 195 150 300 150

box
false
0
Polygon -7500403 true true 150 285 285 225 285 75 150 135
Polygon -7500403 true true 150 135 15 75 150 15 285 75
Polygon -7500403 true true 15 75 15 225 150 285 150 135
Line -16777216 false 150 285 150 135
Line -16777216 false 150 135 15 75
Line -16777216 false 150 135 285 75

bug
true
0
```

```
Circle -7500403 true true 96 182 108
Circle -7500403 true true 110 127 80
Circle -7500403 true true 110 75 80
Line -7500403 true 150 100 80 30
Line -7500403 true 150 100 220 30

butterfly
true
0
Polygon -7500403 true true 150 165 209 199 225 225 225 255 195 270 165 255 150 240
Polygon -7500403 true true 150 165 89 198 75 225 75 255 105 270 135 255 150 240
Polygon -7500403 true true 139 148 100 105 55 90 25 90 10 105 10 135 25 180 40 195 85 194 139
Polygon -7500403 true true 162 150 200 105 245 90 275 90 290 105 290 135 275 180 260 195 215 1
Polygon -16777216 true false 150 255 135 225 120 150 135 120 150 105 165 120 180 150 165 225
Circle -16777216 true false 135 90 30
Line -16777216 false 150 105 195 60
Line -16777216 false 150 105 105 60

car
false
0
Polygon -7500403 true true 300 180 279 164 261 144 240 135 226 132 213 106 203 84 185 63 159 5
Circle -16777216 true false 180 180 90
Circle -16777216 true false 30 180 90
Polygon -16777216 true false 162 80 132 78 134 135 209 135 194 105 189 96 180 89
Circle -7500403 true true 47 195 58
Circle -7500403 true true 195 195 58

circle
false
0
Circle -7500403 true true 0 0 300

circle 2
false
0
Circle -7500403 true true 0 0 300
Circle -16777216 true false 30 30 240

cow
false
0
Polygon -7500403 true true 200 193 197 249 179 249 177 196 166 187 140 189 93 191 78 179 72 21
Polygon -7500403 true true 73 210 86 251 62 249 48 208
Polygon -7500403 true true 25 114 16 195 9 204 23 213 25 200 39 123

cylinder
false
0
```

```
Circle -7500403 true true 0 0 300

dot
false
0
Circle -7500403 true true 90 90 120

face happy
false
0
Circle -7500403 true true 8 8 285
Circle -16777216 true false 60 75 60
Circle -16777216 true false 180 75 60
Polygon -16777216 true false 150 255 90 239 62 213 47 191 67 179 90 203 109 218 150 225 192 21

face neutral
false
0
Circle -7500403 true true 8 7 285
Circle -16777216 true false 60 75 60
Circle -16777216 true false 180 75 60
Rectangle -16777216 true false 60 195 240 225

face sad
false
0
Circle -7500403 true true 8 8 285
Circle -16777216 true false 60 75 60
Circle -16777216 true false 180 75 60
Polygon -16777216 true false 150 168 90 184 62 210 47 232 67 244 90 220 109 205 150 198 192 20

fish
false
0
Polygon -1 true false 44 131 21 87 15 86 0 120 15 150 0 180 13 214 20 212 45 166
Polygon -1 true false 135 195 119 235 95 218 76 210 46 204 60 165
Polygon -1 true false 75 45 83 77 71 103 86 114 166 78 135 60
Polygon -7500403 true true 30 136 151 77 226 81 280 119 292 146 292 160 287 170 270 195 195 21
Circle -16777216 true false 215 106 30

flag
false
0
Rectangle -7500403 true true 60 15 75 300
Polygon -7500403 true true 90 150 270 90 90 30
Line -7500403 true 75 135 90 135
Line -7500403 true 75 45 90 45

flower
```

```
false
0
Polygon -10899396 true false 135 120 165 165 180 210 180 240 150 300 165 300 195 240 195 195 1
Circle -7500403 true true 85 132 38
Circle -7500403 true true 130 147 38
Circle -7500403 true true 192 85 38
Circle -7500403 true true 85 40 38
Circle -7500403 true true 177 40 38
Circle -7500403 true true 177 132 38
Circle -7500403 true true 70 85 38
Circle -7500403 true true 130 25 38
Circle -7500403 true true 96 51 108
Circle -16777216 true false 113 68 74
Polygon -10899396 true false 189 233 219 188 249 173 279 188 234 218
Polygon -10899396 true false 180 255 150 210 105 210 75 240 135 240

house
false
0
Rectangle -7500403 true true 45 120 255 285
Rectangle -16777216 true false 120 210 180 285
Polygon -7500403 true true 15 120 150 15 285 120
Line -16777216 false 30 120 270 120

leaf
false
0
Polygon -7500403 true true 150 210 135 195 120 210 60 210 30 195 60 180 60 165 15 135 30 120 1
Polygon -7500403 true true 135 195 135 240 120 255 105 255 105 285 135 285 165 240 165 195

line
true
0
Line -7500403 true 150 0 150 300

line half
true
0
Line -7500403 true 150 0 150 150

pentagon
false
0
Polygon -7500403 true true 150 15 15 120 60 285 240 285 285 120

person
false
0
Circle -7500403 true true 110 5 80
```

```
Polygon -7500403 true true 105 90 120 195 90 285 105 300 135 300 150 225 165 300 195 300 210 2
Rectangle -7500403 true true 127 79 172 94
Polygon -7500403 true true 195 90 240 150 225 180 165 105
Polygon -7500403 true true 105 90 60 150 75 180 135 105

plant
false
0
Rectangle -7500403 true true 135 90 165 300
Polygon -7500403 true true 135 255 90 210 45 195 75 255 135 285
Polygon -7500403 true true 165 255 210 210 255 195 225 255 165 285
Polygon -7500403 true true 135 180 90 135 45 120 75 180 135 210
Polygon -7500403 true true 165 180 165 210 225 180 255 120 210 135
Polygon -7500403 true true 135 105 90 60 45 45 75 105 135 135
Polygon -7500403 true true 165 105 165 135 225 105 255 45 210 60
Polygon -7500403 true true 135 90 120 45 150 15 180 45 165 90

shark
true
0
Circle -13345367 true false 120 15 60
Polygon -13345367 true false 120 45 135 180 135 240 90 270 195 270 165 240 165 180 165 120 180

square
false
0
Rectangle -7500403 true true 30 30 270 270

square 2
false
0
Rectangle -7500403 true true 30 30 270 270
Rectangle -16777216 true false 60 60 240 240

star
false
0
Polygon -7500403 true true 151 1 185 108 298 108 207 175 242 282 151 216 59 282 94 175 3 108 1

target
false
0
Circle -7500403 true true 0 0 300
Circle -16777216 true false 30 30 240
Circle -7500403 true true 60 60 180
Circle -16777216 true false 90 90 120
Circle -7500403 true true 120 120 60

tree
```

```
false
0
Circle -7500403 true true 118 3 94
Rectangle -6459832 true false 120 195 180 300
Circle -7500403 true true 65 21 108
Circle -7500403 true true 116 41 127
Circle -7500403 true true 45 90 120
Circle -7500403 true true 104 74 152

triangle
false
0
Polygon -7500403 true true 150 30 15 255 285 255

triangle 2
false
0
Polygon -7500403 true true 150 30 15 255 285 255
Polygon -16777216 true false 151 99 225 223 75 224

truck
false
0
Rectangle -7500403 true true 4 45 195 187
Polygon -7500403 true true 296 193 296 150 259 134 244 104 208 104 207 194
Rectangle -1 true false 195 60 195 105
Polygon -16777216 true false 238 112 252 141 219 141 218 112
Circle -16777216 true false 234 174 42
Rectangle -7500403 true true 181 185 214 194
Circle -16777216 true false 144 174 42
Circle -16777216 true false 24 174 42
Circle -7500403 false true 24 174 42
Circle -7500403 false true 144 174 42
Circle -7500403 false true 234 174 42

turtle
true
0
Polygon -10899396 true false 215 204 240 233 246 254 228 266 215 252 193 210
Polygon -10899396 true false 195 90 225 75 245 75 260 89 269 108 261 124 240 105 225 105 210 1
Polygon -10899396 true false 105 90 75 75 55 75 40 89 31 108 39 124 60 105 75 105 90 105
Polygon -10899396 true false 132 85 134 64 107 51 108 17 150 2 192 18 192 52 169 65 172 87
Polygon -10899396 true false 85 204 60 233 54 254 72 266 85 252 107 210
Polygon -7500403 true true 119 75 179 75 209 101 224 135 220 225 175 261 128 261 81 224 74 135

wheel
false
0
Circle -7500403 true true 3 3 294
```

```
Circle -16777216 true false 30 30 240
Line -7500403 true 150 285 150 15
Line -7500403 true 15 150 285 150
Circle -7500403 true true 120 120 60
Line -7500403 true 216 40 79 269
Line -7500403 true 40 84 269 221
Line -7500403 true 40 216 269 79
Line -7500403 true 84 40 221 269

x
false
0
Polygon -7500403 true true 270 75 225 30 30 225 75 270
Polygon -7500403 true true 30 75 75 30 270 225 225 270

@#$#@#$#@
NetLogo 4.0.4
@#$#@#$#@
set population 200
setup
repeat 200 [ go ]
@#$#@#$#@
@#$#@#$#@
@#$#@#$#@
@#$#@#$#@
default
0.0
-0.2 0 0.0 1.0
0.0 1 1.0 0.0
0.2 0 0.0 1.0
link direction
true
0
Line -7500403 true 150 150 90 180
Line -7500403 true 150 150 210 180

@#$#@#$#@
```