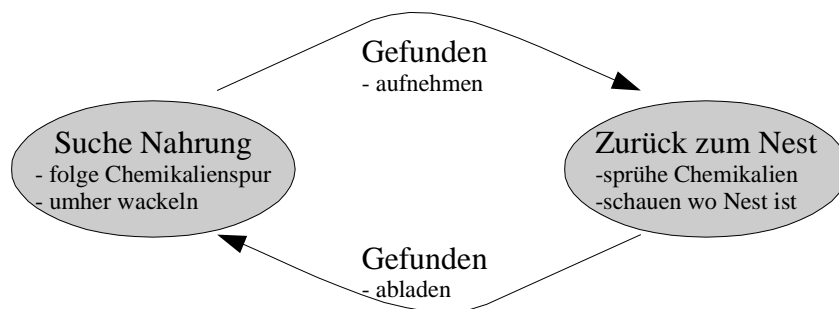


# Wände / Labyrinth für das Ameisen Modell

## 1. Einführung

Das Ameisen Modell besteht aus einem Spielfeld welches über die Ränder zyklisch ist. In der Mitte befindet sich das Nest der Kolonie. Aus diesem Nest kommen die Ameisen und suchen nach Nahrung. Finden sie ein Stück Nahrung, so tragen sie es zurück ins Nest und sprühen dabei Chemikalien auf den Boden. Wenn andere Ameisen diese Chemikalie riechen so folgen sie der Spur. Dadurch, dass mehrere Ameisen nun die Nahrung finden, wird die Chemiekalienspur weiter ausgebaut und lockt noch mehr an.

Der innere Status einer Ameise kann wie folgt beschrieben werden:



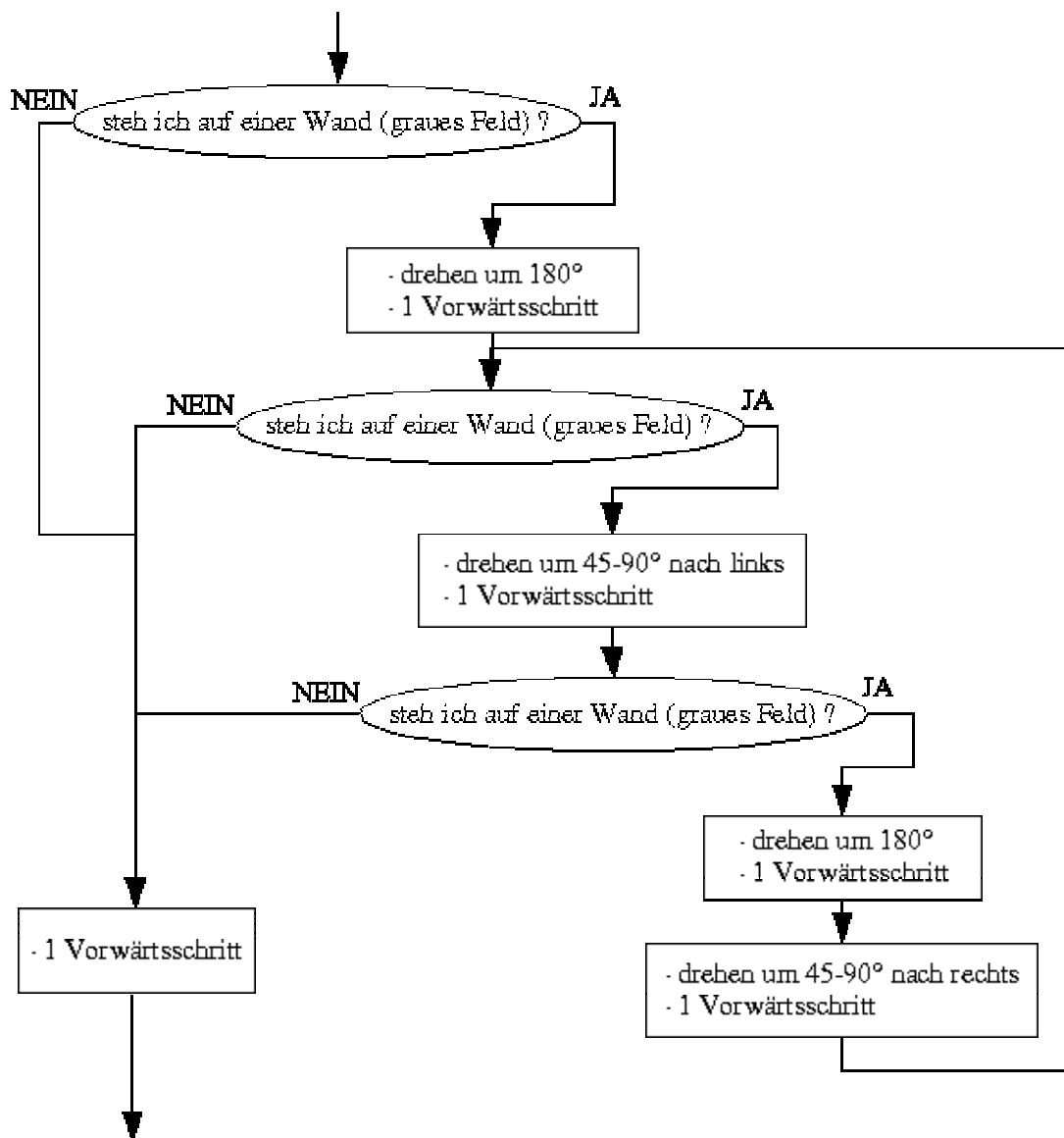
Meine Aufgabe bestand darin dieses Ameisenmodell in StarlogoT<sup>1</sup> zu erweitern, damit man auch unüberwindbare Wände einfügen kann. Es sollte die Möglichkeit geben Wände mittels dem grauen Farbpinsel aus dem Malwerkzeug von StarlogoT auf das Spielfeld zu malen. Man sollte dies bevor man die Aufstellung der Nahrungsquellen macht und auch während der Simulation machen können. Die Ameisen sollten sofort reagieren wenn sie auf eine Mauer (graues Feld) stehen und umdrehen.

---

<sup>1</sup> StarlogoT: <http://ccl.northwestern.edu/cm/starlogoT/>

## 2. Konkrete Implementation

Die Ameisen werden im original Ameisen Modell, wie in Turtlegraphic üblich, aus der Perspektive des Akteurs (Ameise) gesteuert, und mittels einem Schritt vorwärts bewegt. Ich habe nun alle Vorkommnisse dieses Vorwärtsbewegungs-Befehl durch einen Funktionsaufruf ersetzt, der die folgende neue Befehlsstruktur enthält:

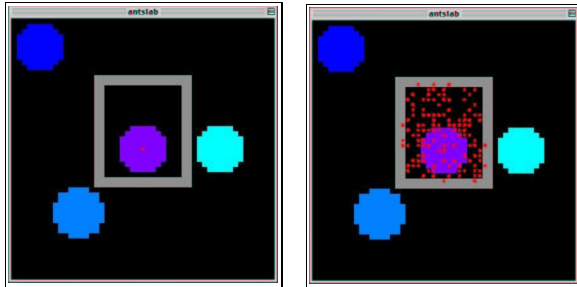


Oder einfach Erklärt: Ist die Ameise auf einem grauen Feld, einer Wand, so muss sie umkehren und einen Forwärtsschritt machen. Steht sie immernoch auf einer Wand so probiert sie einen Schritt etwas nach links und nachher etwas nach rechts, bis sie nicht mehr auf der Wand steht.

### 3. Resultate

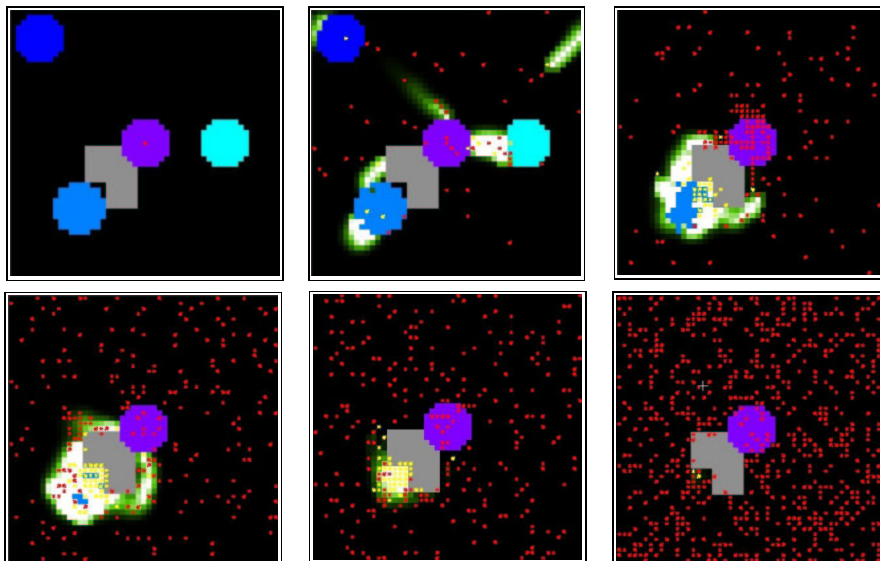
#### 3.1. Beispiele

##### 3.1.1. Ameisen einsperren



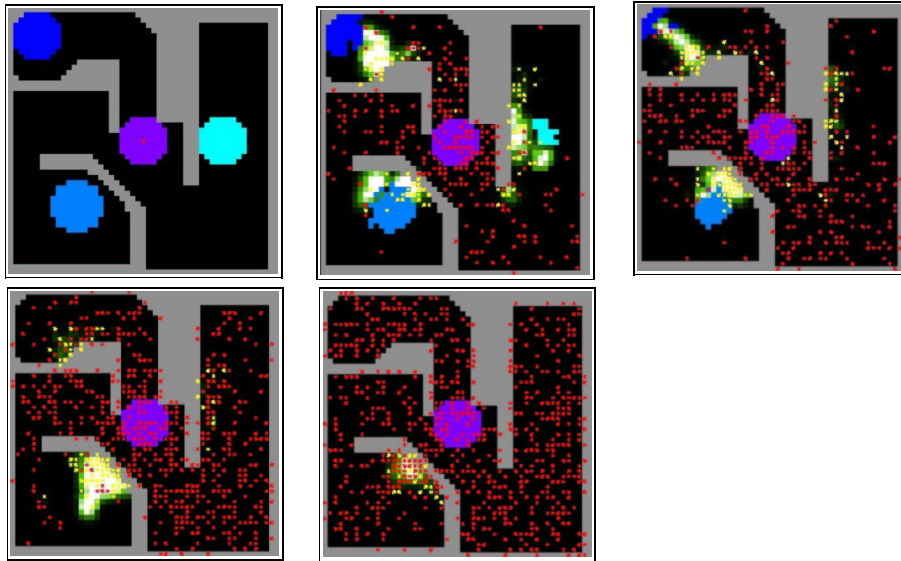
Mit einer Mauer um das Nest können die Ameisen eingesperrt werden, ohne dass sie je eine Futterquelle erreichen.

##### 3.1.2. Einfaches Hindernis



Mauern können, wie in diesem Fall, einfach ein Hindernis sein, das aber von den Ameisen über Umwege überwunden werden können. Die Futterquelle mit der Mauer wird zu letzt aufgebraucht, da die Ameisen diese nicht sofort gefunden haben und es nicht so schnell zu einer Chemikalienspur kommt, welcher andere Ameisen folgen, da die Futter beladenen Ameisen länger festsitzen.

### 3.1.3. Komplexeres Labyrinth mit einer Falle



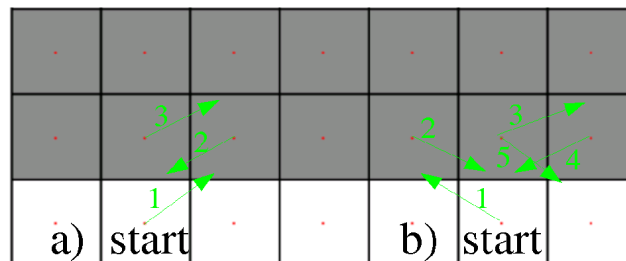
Bei diesem Beispiel wurden alle möglichen Arten von Hindernissen bis zur Falle aus dem es kein Entrinnen gibt, ausprobiert (unten links).

## 3.2. Erklärungen

3.2.1. Die einfachste Strategie um Ameisen beizubringen, dass sie nicht eine Wand überqueren dürfen wäre:

Sobald die Ameise auf der Wand steht, muss sie sich um  $180^\circ$  drehen und einen Vorwärtsschritt machen.

Dadurch würde sie dorthin zurück kehren wo sie her kam und sollte von dort wieder weiter machen. Es hat sich herausgestellt, dass das nicht generell funktioniert in StarLogoT. Die Erklärung ist relativ einfach:



Da das Spielfeld in Felder unterteilt ist, kann es durch die vereinfachte Berechnung in StarlogoT vorkommen, dass ein Schritt ganz am Rand auf ein Feld kommt. Wenn nun der nächste Schritt von der Mitte des Feldes weiter gemacht wird, so kann es vorkommen, dass man nicht auf das Ursprungsfeld zurück

kommt ( a ). Ich habe das probiert etwas abzufangen, in dem die Ameise links und rechts schaut ob sie dort ab der Wand kommt, bis sie das schafft ( b ). Diese Strategie hat sich ziemlich gut bewährt und es kommen nur noch 1-2% der Ameisen über die Wand. Was ja möglicherweise der Realität entspricht, dass man Ameisen nicht unüberwindbare Hindernisse in den Weg stellen kann.

**3.2.2.** Was ich nicht verändert habe ist die Funktion die der Ameise sagt in welcher Richtung sich das Nest befindet. Das heisst, die Ameise übersieht dabei die Mauern und kann so auch in eine Falle geraten [siehe 3.1.3]. Was das Finden der Nahrung angeht funktioniert das Problemlos, da die Ameisen ja der Chemikalienspur folgen und sonst einfach herum wackeln bis sie was riechen.

## **4. Weiterführende Überlegungen**

**4.1.** Die Chemikalie diffundiert durch die Wand hindurch, was nicht realistisch ist. Eine Lösung dieses Problems müsste eine tiefgreifende Veränderung der Chemikalienfunktionen die von StarlogoT zur Verfügung gestellt werden enthalten.

**4.2.** Die schauen-wo-Nest-ist Funktion müsste adaptiert werden, damit die Ameisen nicht einfach stecken bleiben, wenn eine Wand zwischen ihnen und dem Nest ist. Beispielsweise dass sie sich länger von der Wand entfernt bevor sie wieder umschaue wo sich das Nest befindet und sich in diese Richtung bewegt. Da das den Rahmen meiner Arbeit übersteigt, lasse ich das offen für zukünftige Versuche.

## 5. Anhang: Veränderter StarlogoT-code

```
patches-own    [chemical food nest? nest-scent food-source-number]
turtles-own    [carrying-food? drop-size ahead scent-left scent-right]
globals        [clock]
constants      [ food1 [1] food2[2] food3 [3]]

to setup
  setup-turtles
  setup-patches
  setclock 0
end

to setup-turtles
  crt 100
  setxy 0 0
  setc red
  seth random 360
  setcarrying-food? false
end

to setup-patches
  setchemical 0
  setup-nest
  setup-food
  update-display
end

to setup-nest
  setnest? ((distance 0 0) < 5)
  setnest-scent 200 - (distance 0 0)
end

to setup-food
  setfood 0
  setfood-source-number -1
  if ((distance (0.6 * screen-edge) 0) < 5)
    [setfood 1 + random 2
     setfood-source-number food1]

  if ((distance (-0.5 * screen-edge) (-0.5 * screen-edge)) < 5)
    [setfood 1 + random 2
     setfood-source-number food2]

  if ((distance (-0.8 * screen-edge) (0.8 * screen-edge)) < 5)
    [setfood 1 + random 2
     setfood-source-number food3]
end

to update-display
  ifelse nest?
    [setpc violet]
    [ifelse food > 0
      [if(food-source-number = food1) [setpc 85]
       if(food-source-number = food2) [setpc 95 ]
       if(food-source-number = food3) [setpc 105]]
      [if pc != 5 [scale-pc green chemical 0.1 5]]]
end

to step
  if pc = 5 [rt 180 fd 1 unstep]
  fd 1
end

to unstep
  if pc = 5 [
    lt 45 lt random 45
    fd 1
    if pc = 5 [
      rt 180
      fd 1
      rt 45 rt random 45
      fd 1
      unstep ]]
end

to go
  go1
  go2
```

```
end

to go1
  if who > clock [stop]
  ifelse carrying-food?
    [setc yellow return-to-nest]
    [setc red look-for-food]
end

to go2
  diffuse chemical diffusion-rate / 100
  setchemical chemical * (100 - evaporation-rate) / 100
  update-display
  setclock clock + 1
end

to return-to-nest
  if nest?
    [setcarrying-food? false
     rt 180 step stop]
  tsetchemical chemical + drop-size
  setdrop-size (drop-size - 1.5)
  if drop-size < 1 [setdrop-size 1]
  uphill-nest-scent
  wiggle
  grid-step
end

to look-for-food
  if food > 0
    [setcarrying-food? true
     tsetfood food - 1
     setdrop-size 60
     rt 180 step stop]
  ifelse chemical > 2
    [step]
    [ifelse chemical < 0.05
     [wiggle step]
     [uphill-chemical grid-step]]
end

to uphill-chemical
  setahead next-chemical
  rt 45
  setscent-right next-chemical
  lt 90
  setscent-left next-chemical
  rt 45
  if (scent-right > ahead) and not (scent-right < scent-left) [rt 45 stop]
  if (scent-left > ahead) and not (scent-left < scent-right) [lt 45]
end

to next-chemical
  output chemical-at dx dy
end

to uphill-nest-scent
  setahead next-nest-scent
  rt 45
  setscent-right next-nest-scent
  lt 90
  setscent-left next-nest-scent
  rt 45
  if (scent-right > ahead) and not (scent-right < scent-left) [rt 45 stop]
  if (scent-left > ahead) and not (scent-left < scent-right) [lt 45]
end

to next-nest-scent
  output nest-scent-at dx dy
end

to grid-step
  step
  setxy round xcor round ycor
end

to wiggle
  rt random 40
  lt random 40
end
```