Alan Turing, 1952, "The Chemical Basis of Morphogenesis", phil. Trans R Soc B, 237 37-72 (transactions of the royal society)

Turing proposed the idea that two or more reacting chemicals in a diffusion would reach a stable state and thus reveal a pattern. Andrew Adamatzky in his introduction to ***reaction-diffusion*** models uses the example of the grass fire to explain the general ideas about reaction diffusion.
Initially the idea is that you have a system at rest, consisting of an excitable medium – in this case burnable grass; after perturbation the system moves in some phase space along a fixed trajectory until it returns to the fixed point.

- setting on fire is a *perturbation* action
- a fire front is an *excitation front*
- a burnt zone is the part of the medium in a *refractory state*
- growth of grass is the *recovery of the substrate*

The items in italics are technical terms derived from chemistry, which we need not go into here, but the point is that the process consists of waves (or fronts) of excitation which travel across the chemical and as they do so, where they meet they stop. So if you chuck some matches into an area of dried grass, where the matches land becomes the centre of a circle (if there is no wind!) of burnt grass. This will go on until there is no more grass to burn, or if the front of burning grass meets another front. The fact that once the grass is burnt it can't immediately be reignited is the fact behind the bushmen's trick of stopping a fire by starting another one ahead of the front.

This final project in our 3 lessons on netlogo is a combination of the two we have looked at, the cellular automata (magic carpet/life game) and the turtles/agents/particles the we looked at last week.

In today's project we are building on the repel procedure that we have already discussed, where a set of turtles do the following:

1 - find nearest turtle
2 - back off by repel-strength

We have observed that once the turtles have quietened down they are all the same distance from each other, and as a consequence represent a triangular lattice. If you look at the distribution of the points you can see that

1 - they can all be seen to lie on this triangular grid
2 - they can all be seen to lie at the centre of a small region bounded by their immediate neighbour's regions

The technical term for developing these regions is delaunay triangulation/ voronoi diagram, and there are many methods of constructing such things (we can show you a version written in VBA later on) which rely on complicated mathematical formulae and constructive geometry, but today we will look at how we can use these points to initiate a reaction diffusion mechanism so as to show these regions. We will look at how to do it the simplest possible way – the way nature does it in the cracks in dried mud and the other material methods such as Frei Otto looks at (and we will next week in the soap-film workshop).

The program works by using the patches (like in the magic sponge) to do the diffusion. At the beginning the patches are all at zero – growing grass as it were.

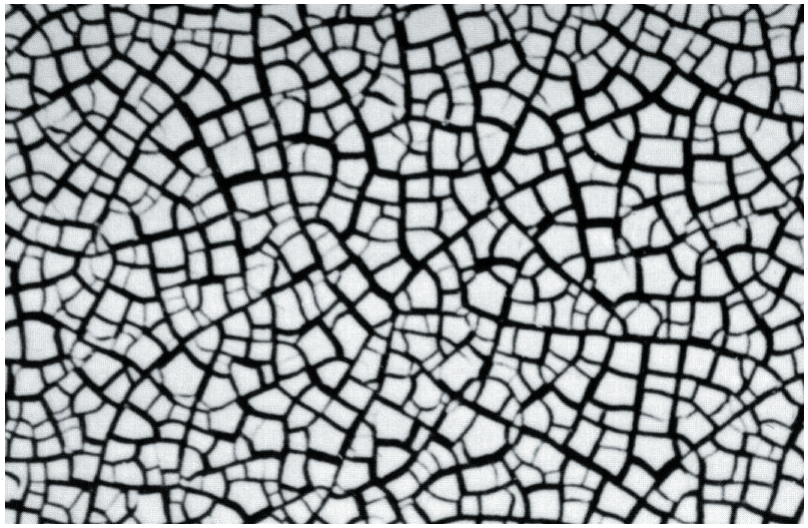
```
to leak
  without-interruption  ;use patchcolour to set patches own variables
        [
        ask patches [if not boundary [checkdiffuse]]
  ]
         if recording [movie-grab-graphics ]
 without-interruption    ;;use patches own to set patch colour
        [
      ask patches [ifelse boundary [set pcolor white set s1 0 ]
                           [if s1 > 0  [set pcolor s1]]
   ]
  ]
end

  to checkdiffuse
  if pcolor = 0                                ;this patch is currently blank

  [
      set s1 sum values-from  neighbors3d  [pcolor ]
      set s2 count neighbors3d with [ pcolor > 0]
      if s2 > 0                          ; any coloured patches in the neighbourhood ?
      [
      ifelse ( int(s1  / s2) * s2 ) = s1            ;check all colours are the same ?
            [set s1 ( s1 / s2 ) ]                ;then just turn the same colour
            [set boundary true]    ;but if some are different be a boundary
      ]

  ]

  end
```

How it works

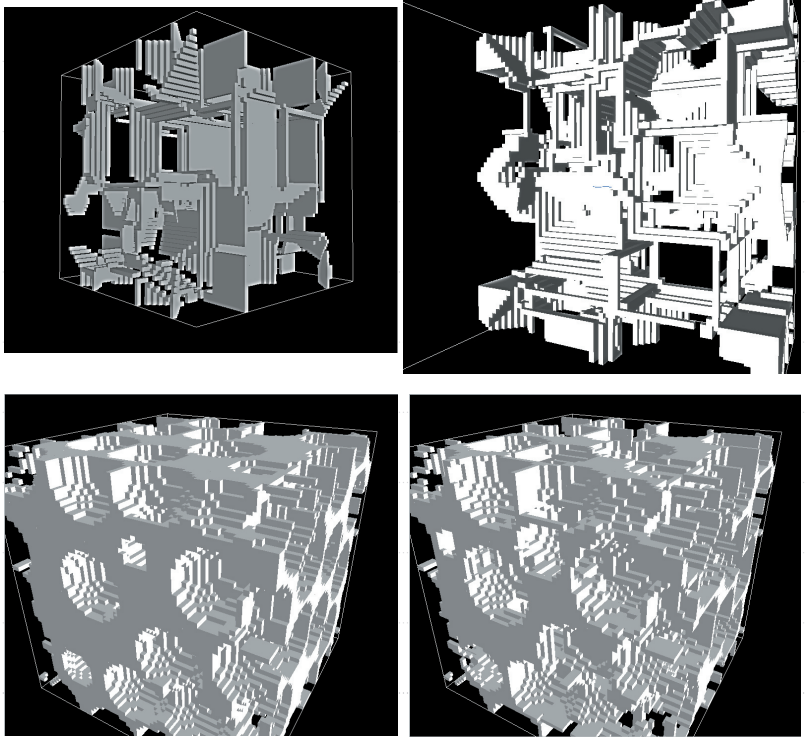
We need a way of initiating the reaction diffusion mechanism (dropping the match somewhere as it were), and we do this by getting the turtles to colour in the patch they are standing on. Using 'stamp' will do this, but if we only do that the patches stamped on will all be the same colour. For ease of programming we need the initiators to be different colours so we add 'who' (turtle's ID) to the stamp to get different colours for every turtle (because they all have different ids). Then we tell them to die because we don't need them any more - 'die'.

The algorithm

The general idea is that any empty patch will change to the colour of any neighbouring patch that isn't zero. That will work to diffuse the initial coloured patch (where the turtle stamped) but we don't have any way to stop it. We want to stop the diffusion when one colour meets another different colour, and we will set this patch to be a 'boundary' patch. In the 'checkdiffuse' procedure below we set patches to be either the colour of the general diffusion or if there are 2 or more colours in the neighbourhood we set it to be a boundary. This works by the slightly devious method of checking how many neighbours are coloured at all (s1 =count neighbours with …) and also summing all the (possibly different) colour values in the neighbourhood (s2 = sum values-from). If all the colours are the same then taking the whole number value of the division of howmany(s1) by the howmuch (s2) and comparing that with the real result we can infer whether all the colours are the same .



mud crack pattern



Images on the right show the effect of visualising different concentrations of chemical as solid.

The 'leak' procedure calls 'checkdiffuse' and then colours them in appropriately, white for boundary cells and the diffusing colour for the others.

In the resulting pattern you will see that the white cells form lines between the original turtle points. The geometry is rather heavily influenced by the orthogonality of the patches grid, so we don't see good hexagons, but only 0, 45 and 90 degree lines. There is another version of this program which uses the 'diffuse' command in netlogo, which results in slightly better tilings, but with gaps because the diffusion is difficult to control.

Tasks

1 - during today's workshop

You should combine the two programmes: 1) circles from last week with repel and 2) this weeks reaction-diffusion, into one.
After that you should use this program with your multiple strength repulse procedures so as to create large and small cells in your emergent voronoi. Don't forget to RECORD your results for Fridays DTP session.

2 - assignment for next week

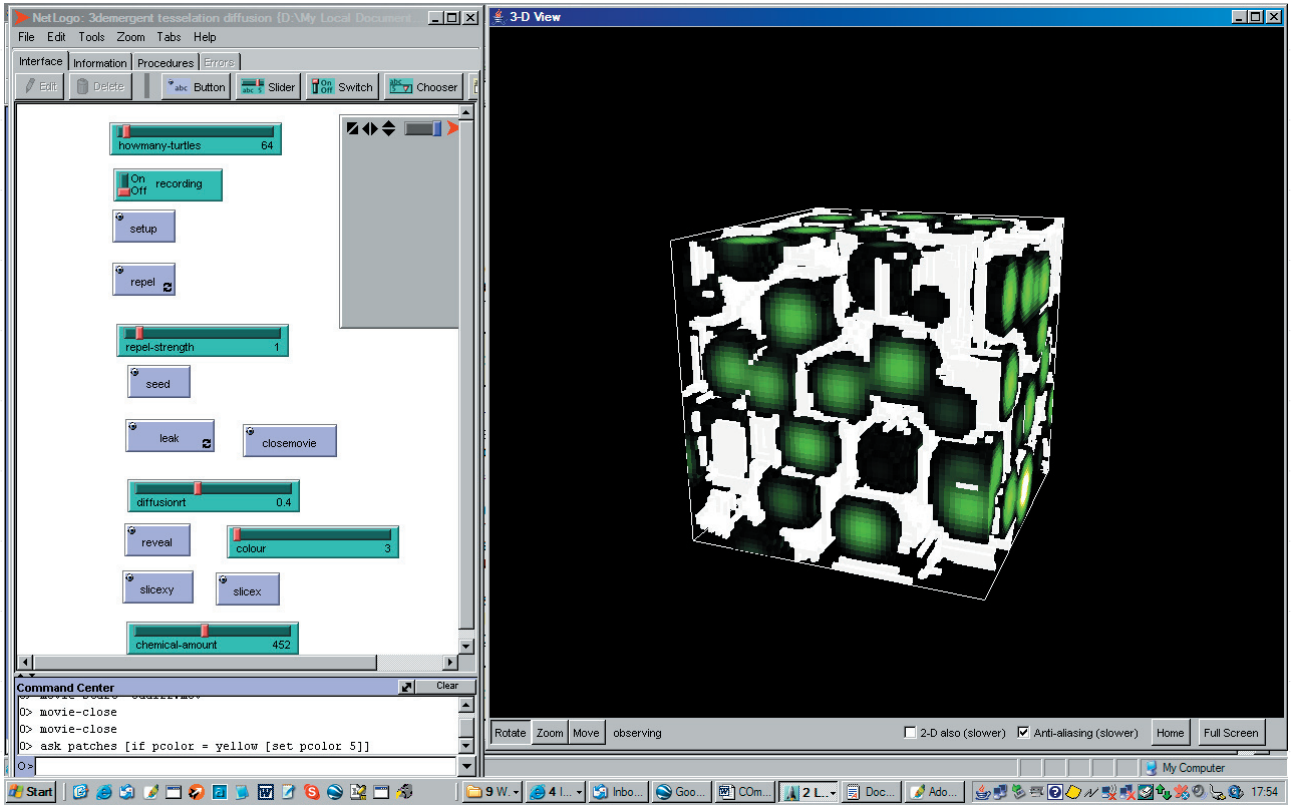
Collect images from the web of natural/ social examples of these reaction-diffusion processes and emergent patterns through forces.
Come up with a program of your own that combines the last three workshops scripts - handed-out and written by yourselves. Keep it simple and look for the graphic effect rather than heavy programming!

form script. workshop

2010 netlogo.react_diffuse 03

1310 netlogo.agents 02

0610 netlogo.CA 01



form script. workshop

2010 netlogo.react_diffuse 03

1310 netlogo.agents 02

0610 netlogo.CA 01

form script. workshop

2010 netlogo.react_diffuse 03

1310 netlogo.agents 02

0610 netlogo.CA 01

form script. workshop

2010 netlogo.react_diffuse 03

1310 netlogo.agents 02

0610 netlogo.CA 01