Bio-Inspired Artificial Intelligence - EMATM0029
Exercises - Cellular Systems

TED talks

Watch the following videos related to cellular systems, reflect on the potential and limitations of the approaches described:
`https://youtu.be/60P7717-X0Q?list=PLPwd13OaGbndtCwvT1hUieedttfKEbMmX`
`https://youtu.be/R9Plq-D1gEk?list=PLPwd13OaGbndtCwvT1hUieedttfKEbMmX`
`https://youtu.be/C2vgICfQawE?list=PLPwd13OaGbndtCwvT1hUieedttfKEbMmX`
PS, here is our full youtube channel, please feel free to propose your favorite videos by email to sabine.hauert@bristol.ac.uk:
`https://www.youtube.com/playlist?list=PLPwd13OaGbndtCwvT1hUieedttfKEbMmX`

Cellular Automaton

Implement you own cellular automaton.

- Start from a 1x9 grid with all values set to 0 except for the middle position set to 1. This should look like this: 0 0 0 0 1 0 0 0 0.
- On paper, apply the following rules for 5 generations.

000	001	010	011	100	101	110	111
↓	↓	↓	↓	↓	↓	↓	↓
0	1	0	1	1	0	1	0

- Make a Matlab program that does this automatically.
- What happens if you change the initial state?

Wolfram Rules

Run Wolfram's rules and determine the properties of different cellular automata. You can learn more about his rules here:
http://mathworld.wolfram.com/ElementaryCellularAutomaton.html

Download the Matlab code to simulate elementary CAs from BlackBoard (*elementaryCellularAutomata.zip*). Use the function *elementaryCellularAutomata()* to produce rules 40, 56, 10 and 110.

```
colormap(gray)
pattern=elementaryCellularAutomata(RULENUMBER,200)
pattern( pattern==1 )=255;
image(pattern)
```

Have a look at the code of elementaryCellularAutomata() and try to understand it.

Identify rules that lead to:

- Uniform final state
- Simple stable or periodic final state
- Chaotic, random, nonperiodic patterns
- Complex, localized, propagating structures

Game of Life

- Go to the webpage (http://pmav.eu/stuff/javascript-game-of-life-v3.1.1/) and try for yourself the Game of Life. Try to make your own patterns. Initialize it with random values.
- Write your own "game of life" rules on paper. The rules are for a 2D automaton and should follow the principles below. You don't need to be exhaustive, just write one rule per principle.
  - Death. If a cell is alive (state = 1) it will die (state becomes 0) under the following circumstances.
  - Overpopulation: If the cell has four or more alive neighbors, it dies.
  - Loneliness: If the cell has one or fewer alive neighbors, it dies.
  - Birth. If a cell is dead (state = 0) it will come to life (state becomes 1) if it has exactly three alive neighbors (no more, no less).
  - Stasis. In all other cases, the cell state does not change. To be thorough, let's describe those scenarios.
    - Staying Alive: If a cell is alive and has exactly two or three live neighbors, it stays alive.
    - Staying Dead: If a cell is dead and has anything other than three live neighbors, it stays dead.
- Optional: Implement your rules in matlab. What happens if you change the rules slightly?

Universal Computer

Check out this amazing cellular automaton that calculates prime numbers:
`http://www.wikiwand.com/en/Elementary_cellular_automaton`.

For more information on how this works, have a look at the explanations here:
`https://www.quinapalus.com/wi-index.html`
`https://www.logre.eu/wiki/Projet_Wireworld/en`

What can you compute with a cellular automata, why?