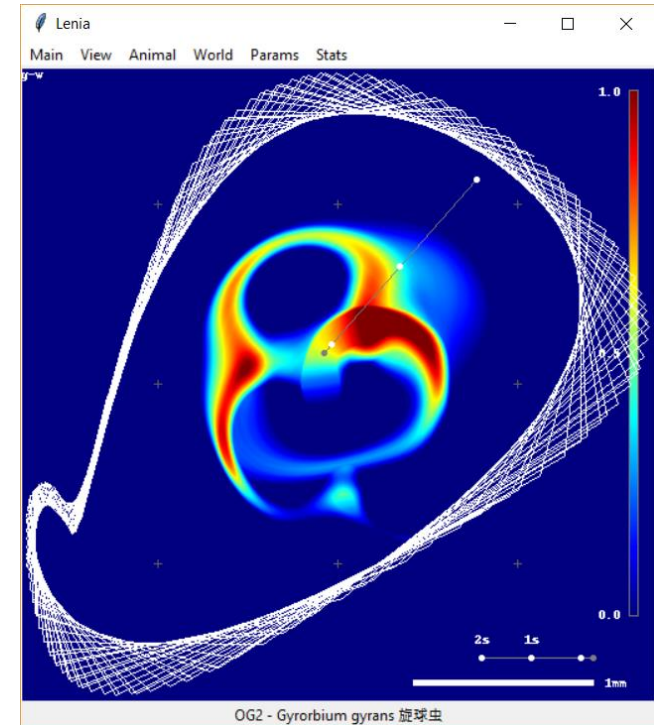


# How to Evolve Life in a Computer using Python

Bert Chan

*Big Data Consultant @ ASL*

PyCon HK / Code Conf 2018



# Programming in the 1990's

- PC: **80286** (8MHz, 8MB RAM)
- OS: MS-DOS (Win3.1 too slow!)
- **Pascal**
  - Simulate life
  - Simulate gravity, fractals
  - Hack & decode games
- **Assembly**
  - Main loop – very fast!
  - Direct write to video cache



*If you did coding and hacked stuffs in the 90's, you're a...*



# Simulate Life

- **Conway's Game of Life**

- John Conway 1970
- Cellular Automata – array of **cells** (0 or 1)
- **Neighborhood** (8 cells, sum)

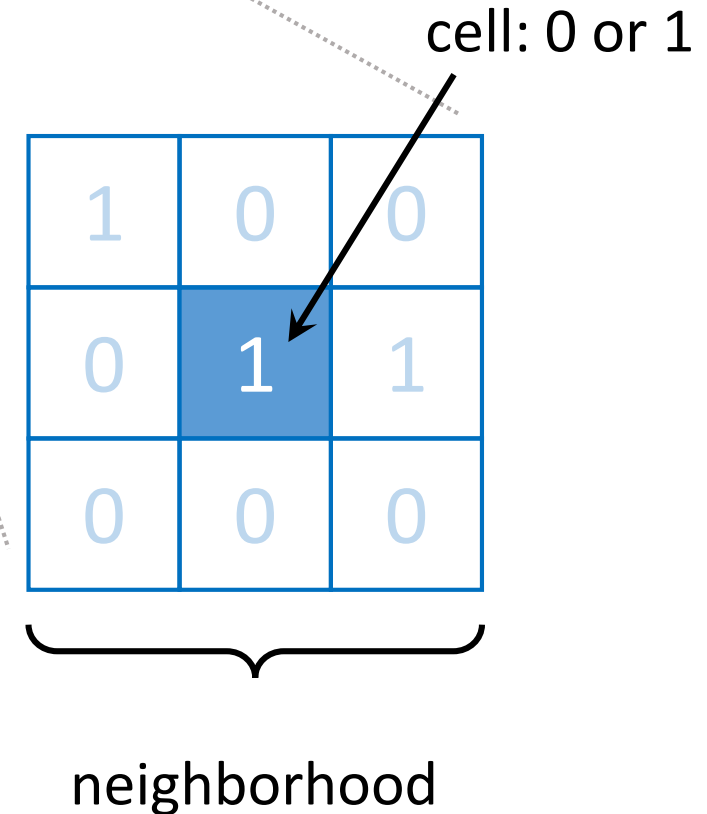
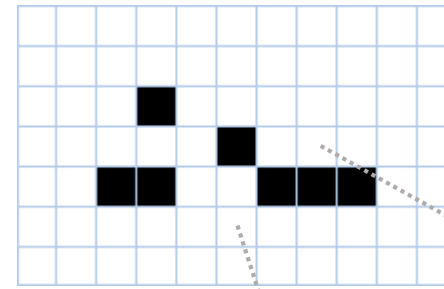
- Simple **if-then-else** rule

for cell in cells:

if cell==1 and sum in [2, 3]: cell = 1 #survive

elif cell==0 and sum in [3]: cell = 1 #born

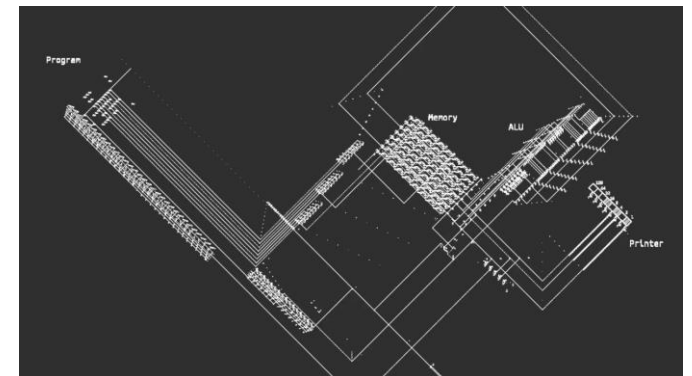
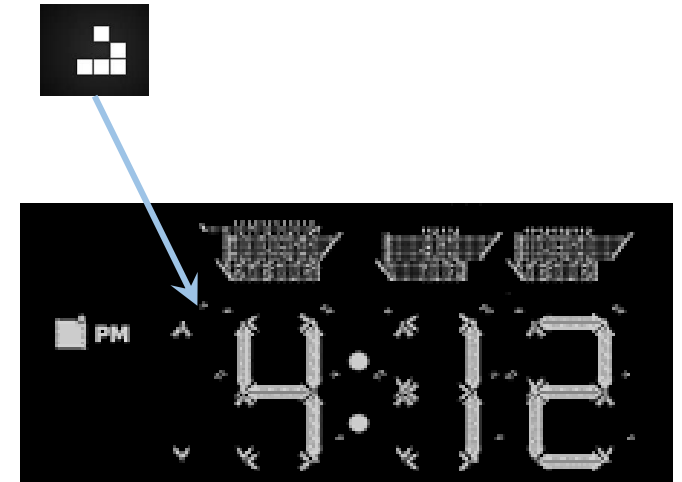
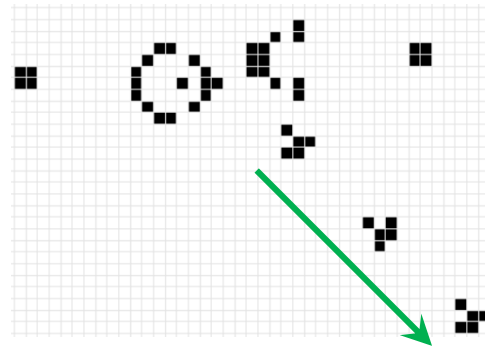
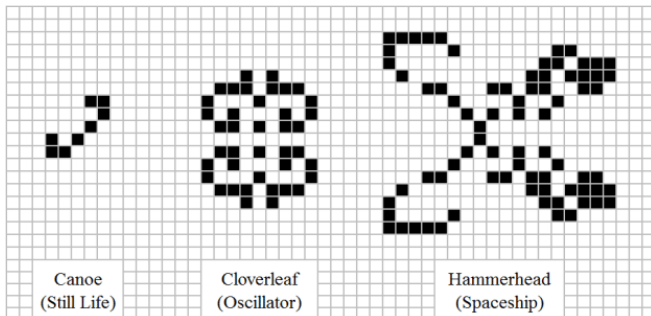
else: cell = 0 #die





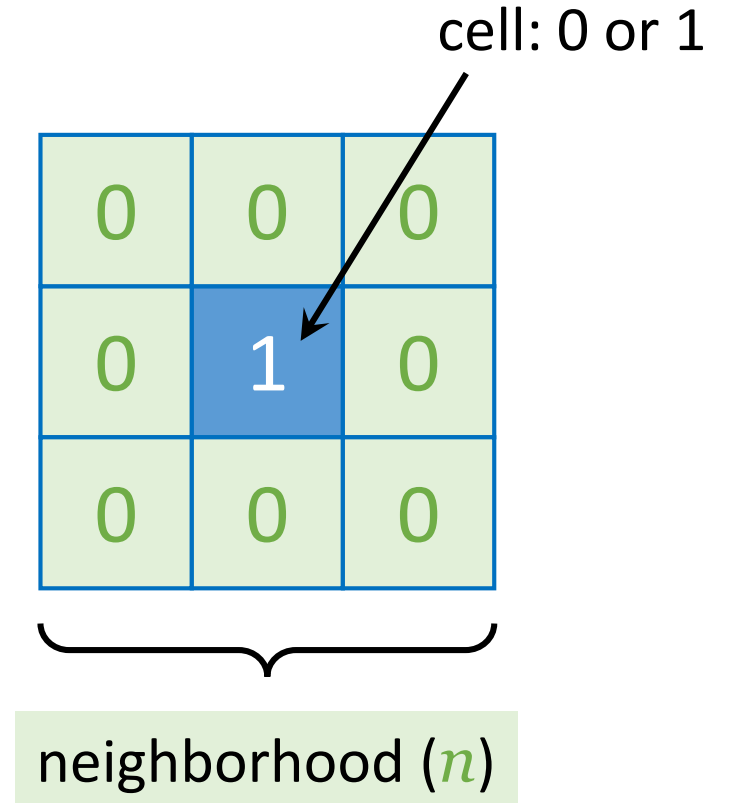
# Conway's Game of Life

- Spaceships, glider gun...
- Logic gate, clock, computer...
- Hackers love it!
- Good way to learn programming!



# Play with the rules

- What if we...

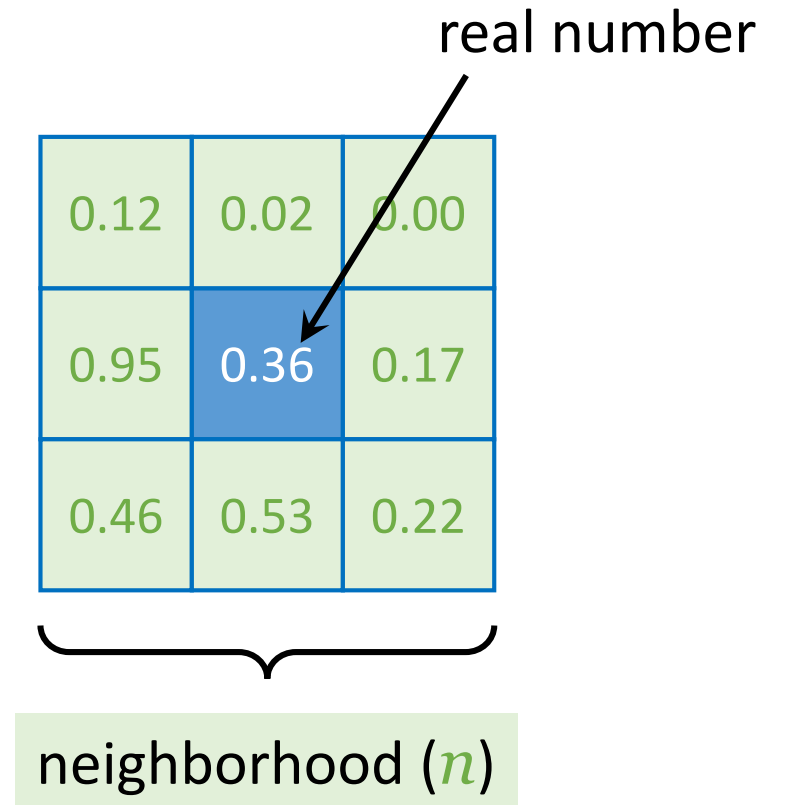


$$\text{sum} = \sum n$$

cell = (if sum ... then ... else ...)

# Play with the rules

- What if we...
  - Use floating point?



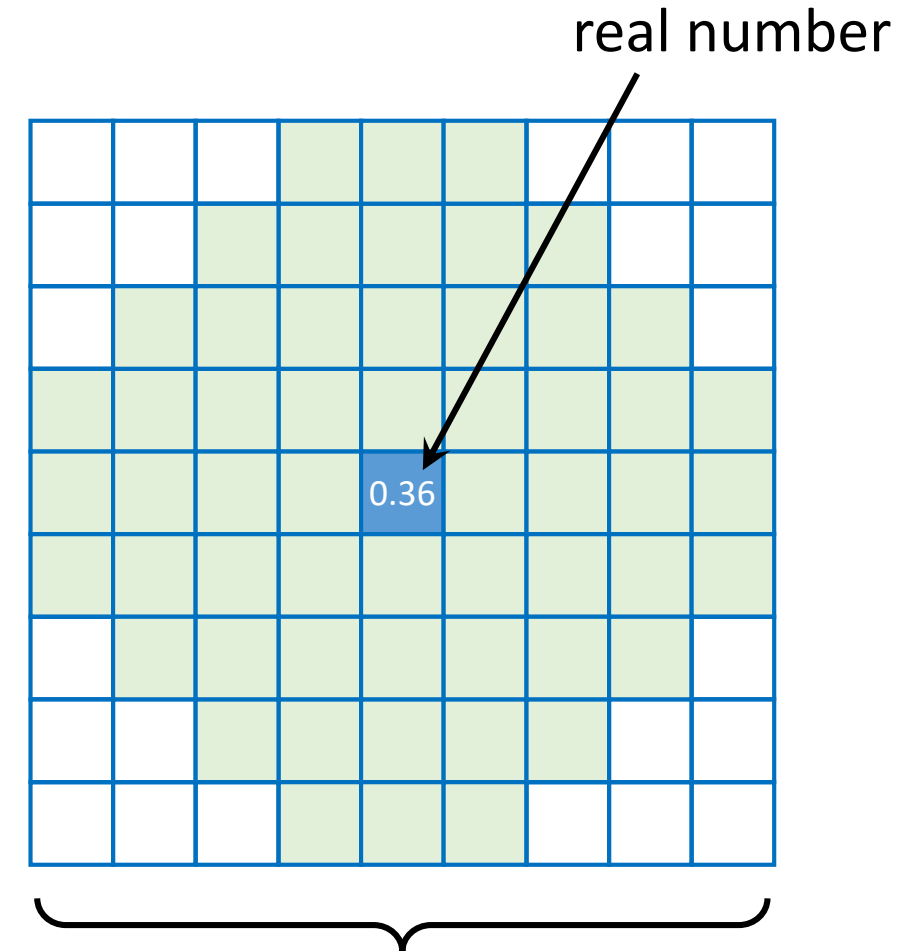
$$\text{sum} = \sum n$$

cell = (if sum ... then ... else ...)



# Play with the rules

- What if we...
  - Use floating point?
  - Bigger neighborhood? Circular?



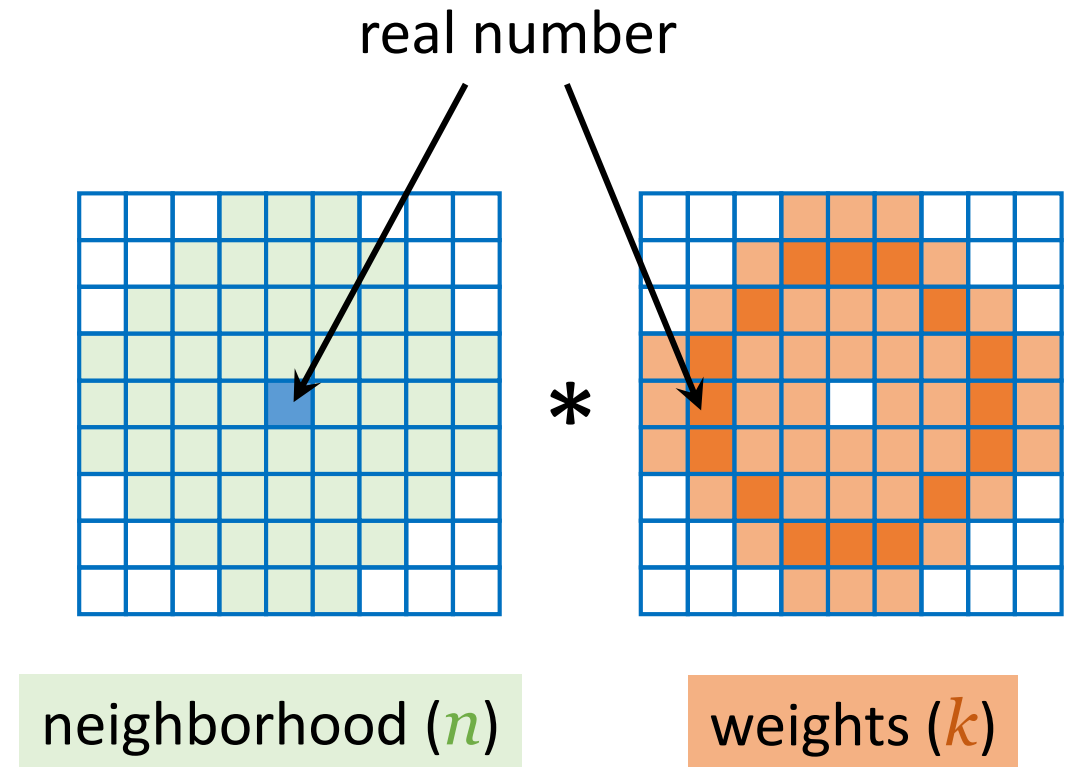
neighborhood ( $n$ )

$$\text{sum} = \sum n$$

cell = (if sum ... then ... else ...)

# Play with the rules

- What if we...
  - Use floating point?
  - Bigger neighborhood? Circular?
  - Weighted sum?

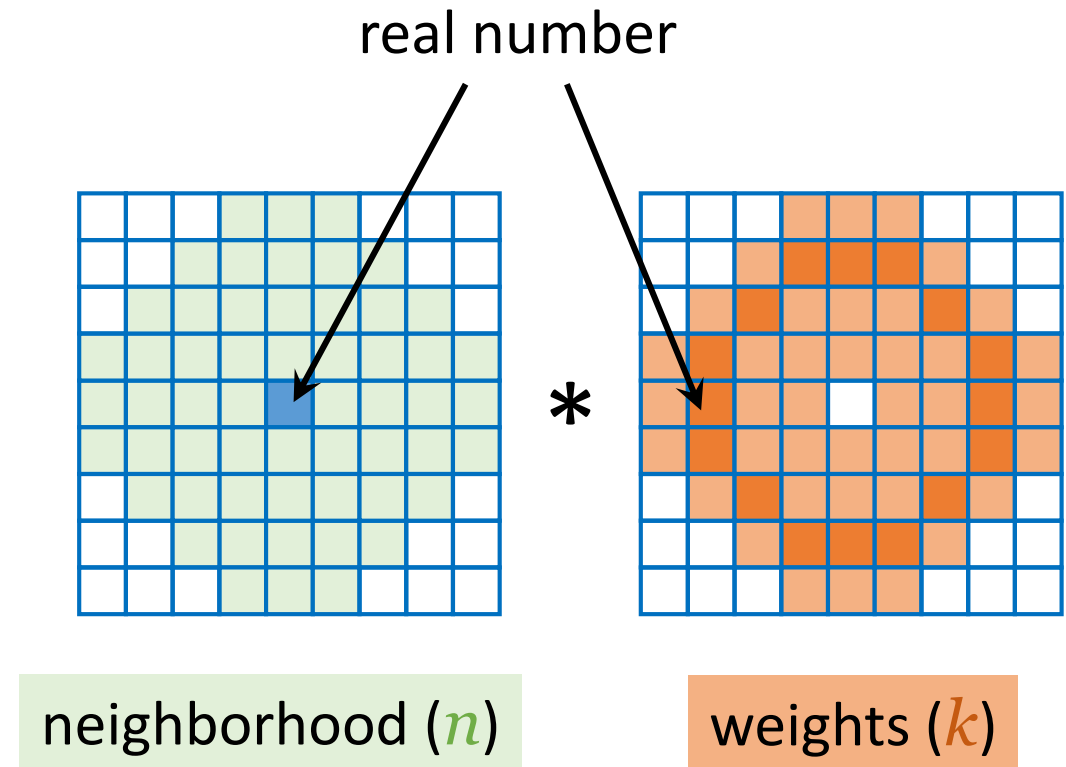


$$\text{sum} = \sum nk$$

cell = (if sum ... then ... else ...)

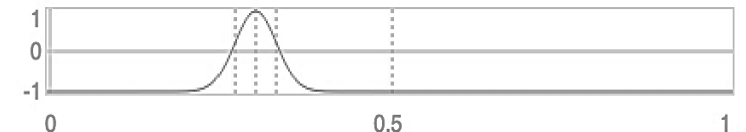
# Play with the rules

- What if we...
  - Use floating point?
  - Bigger neighborhood? Circular?
  - Weighted sum?
  - Smooth update?



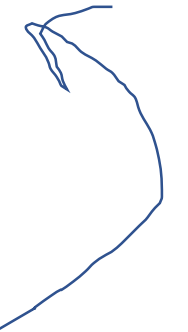
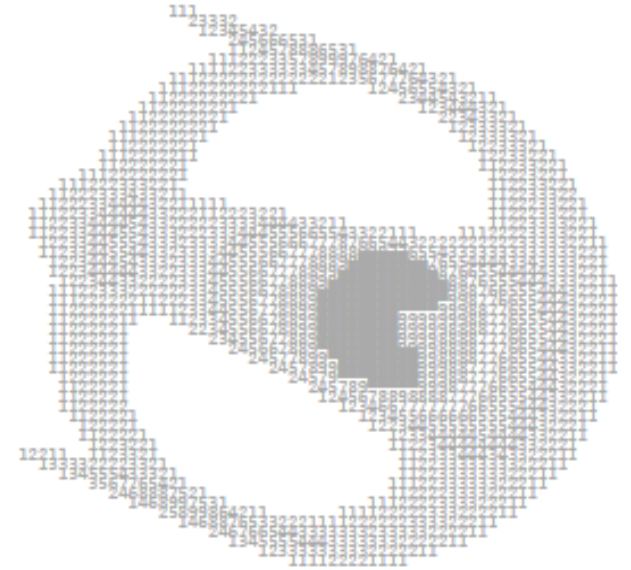
$$\text{sum} = \sum nk$$

$$\text{cell} = \text{cell} + 0.1 * f(\text{sum})$$



# Play with the rules

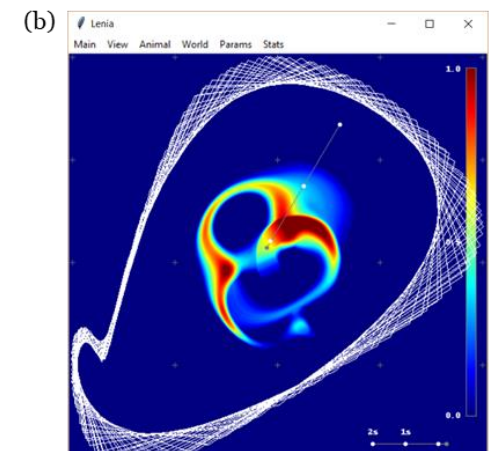
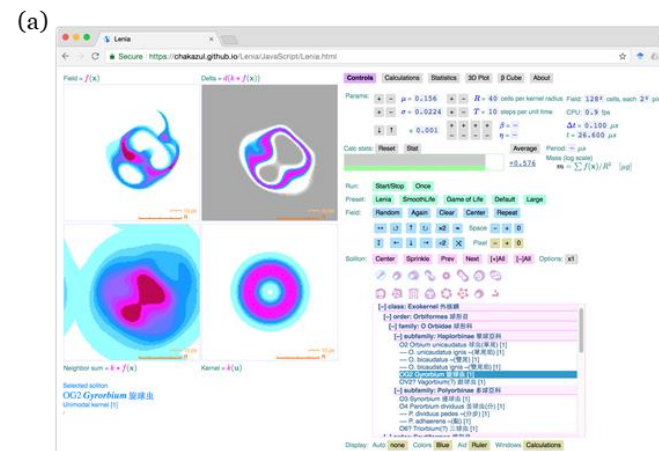
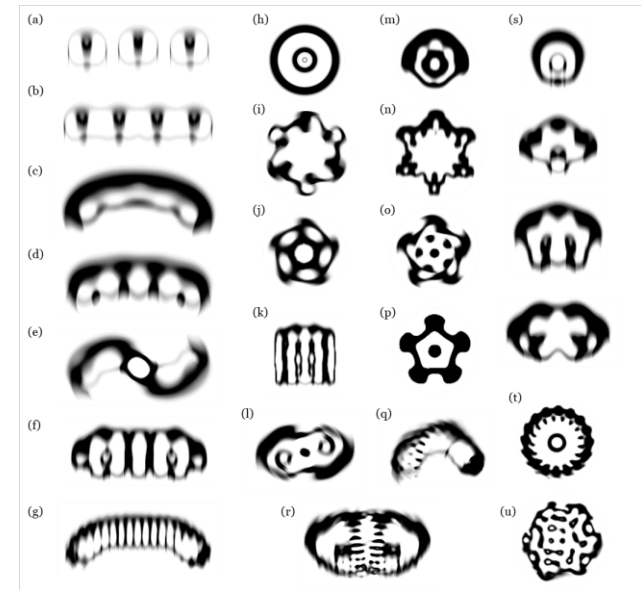
- What if we...
  - Use floating point?
  - Bigger neighborhood? Circular?
  - Weighted sum?
  - Smooth update?
- Spooky things happened...



OMG WHAT IS THIS??

# Lenia

- New kind of **Artificial Life**
  - Microorganism-like creatures
  - Discovered 400+ species
  - Study their anatomy, behavior, physiology...
- Good programming exercise
  - JavaScript, C#, MATLAB, Python



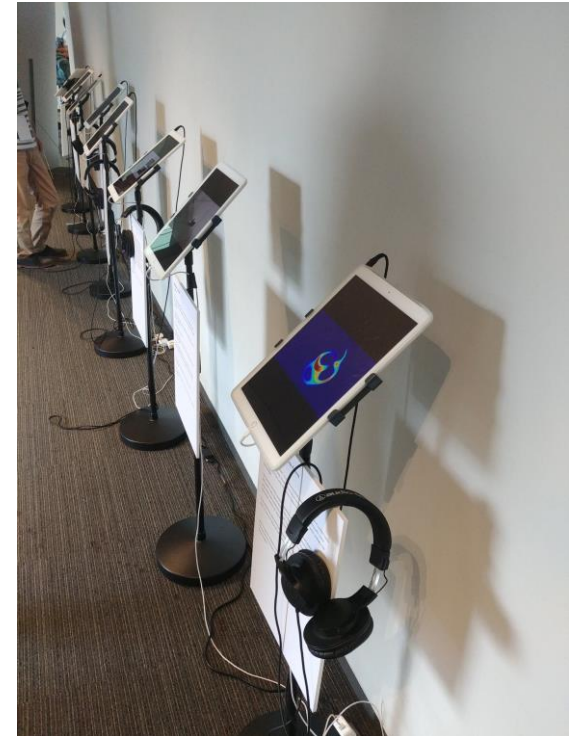
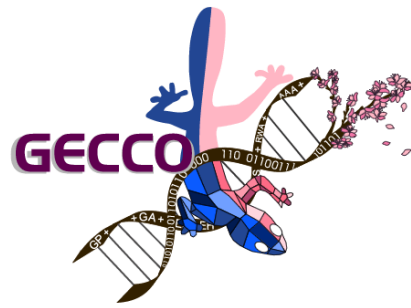
# Video



- Python → showcase video
- <https://vimeo.com/277328815>

# Kyoto

- Won GECCO Virtual Creatures Contest, Kyoto
- Honorable Mention in ALIFE Art Award, Tokyo
- Meet my AI hero – @hardmaru
  - David Ha (Google Brain Tokyo)



# Using Python

for PyCon HK



# Why Python?

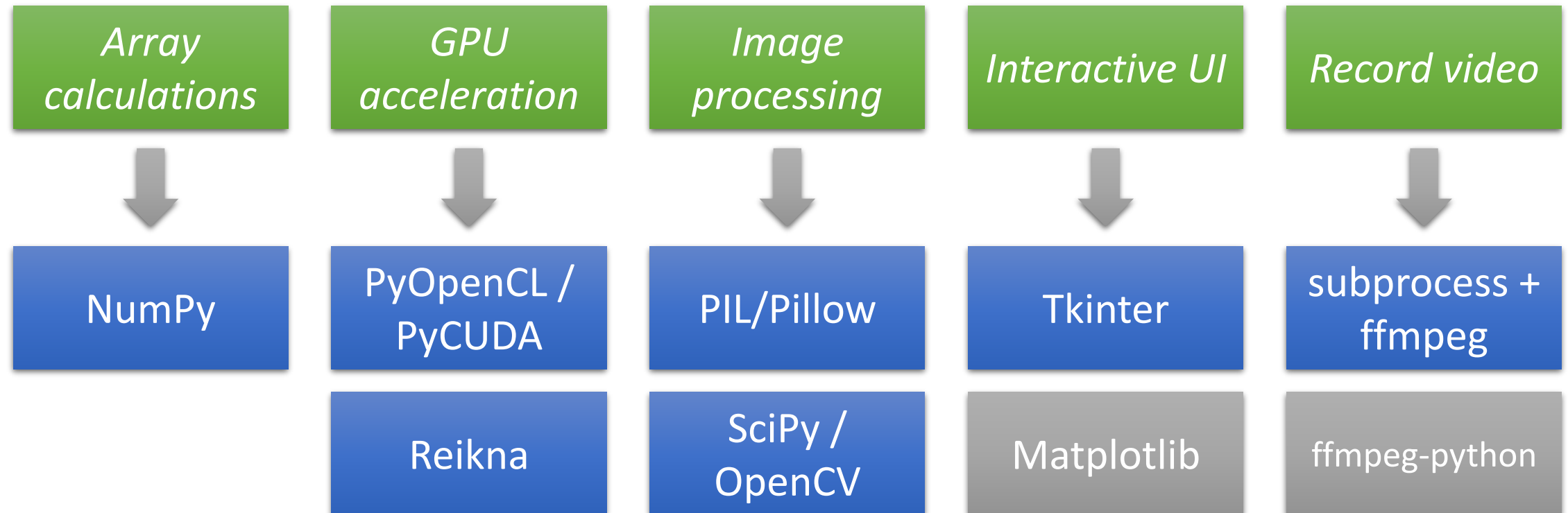
- Good performance
- Fast coding
- Nice syntax (indent, list comprehension, etc)
- Lots of useful libraries
- Vibrant community (PyCon, GitHub...)



# Python Libraries



- “Rule 34” of Python
  - *“If there is a need, there is a Python library for it.”*



# NumPy



- Fast array calculations
  - ✓ Machine learning, deep learning
  - ✓ Basis of image processing, time-series
  - ✓ Cellular automata (weighted sum using FFT)

- Main loop of Lenia in 3 lines

```
potential_fft = np.fft.fft2(cells) * kernel_fft  
potential = np.fft.fftshift(np.real(np.fft.ifft2(potential_fft)))  
cells_new = np.clip(cells + dt * g(potential, m, s), 0, 1)
```



# PyOpenCL/PyCUDA + Reikna

- GPU acceleration
  - (NVIDIA) CUDA → PyCUDA
  - (Apple) OpenCL → PyOpenCL

A screenshot of a web browser displaying the NVIDIA Developer Zone documentation page for CUFFT. The browser address bar shows the URL "https://docs.nvidia.com/cuda/cufft/index.html". The page header includes the NVIDIA logo, "DEVELOPER ZONE", and "CUDA TOOLKIT DOCUMENTATION". A search bar is visible on the right. The main content area contains a code example for computing a number of one-dimensional DFTs. The code is as follows:

```
Computing a number BATCH of one-dimensional DFTs of size NX using cuFFT will typically look like this:  
  
#define NX 256  
#define BATCH 10  
#define RANK 1  
...  
{  
    cufftHandle plan;  
    cufftComplex *data;  
    ...  
    cudaMalloc((void*)&data, sizeof(cufftComplex)*NX*BATCH);  
    cufftPlanMany(&plan, RANK, NX, &iembed, istride, idist,  
                 &oembed, ostride, odist, CUFFT_C2C, BATCH);  
    ...  
    cufftExecC2C(plan, data, data, CUFFT_FORWARD);  
    cudaDeviceSynchronize();  
    ...  
    cufftDestroy(plan);  
    cudaFree(data);  
}
```

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：<https://d.book118.com/018133133027006057>