Continuous Simulation

Announcements

- Tonight:
  - PA 9
  - Exam 2 Review Session
- Tomorrow:
  - Lab
  - Reading
- Thursday: Exam 2

Example: Flu Virus Simulation

- Goal: Develop a simple simulation that shows how disease spreads through a population; provide graphic visualization.

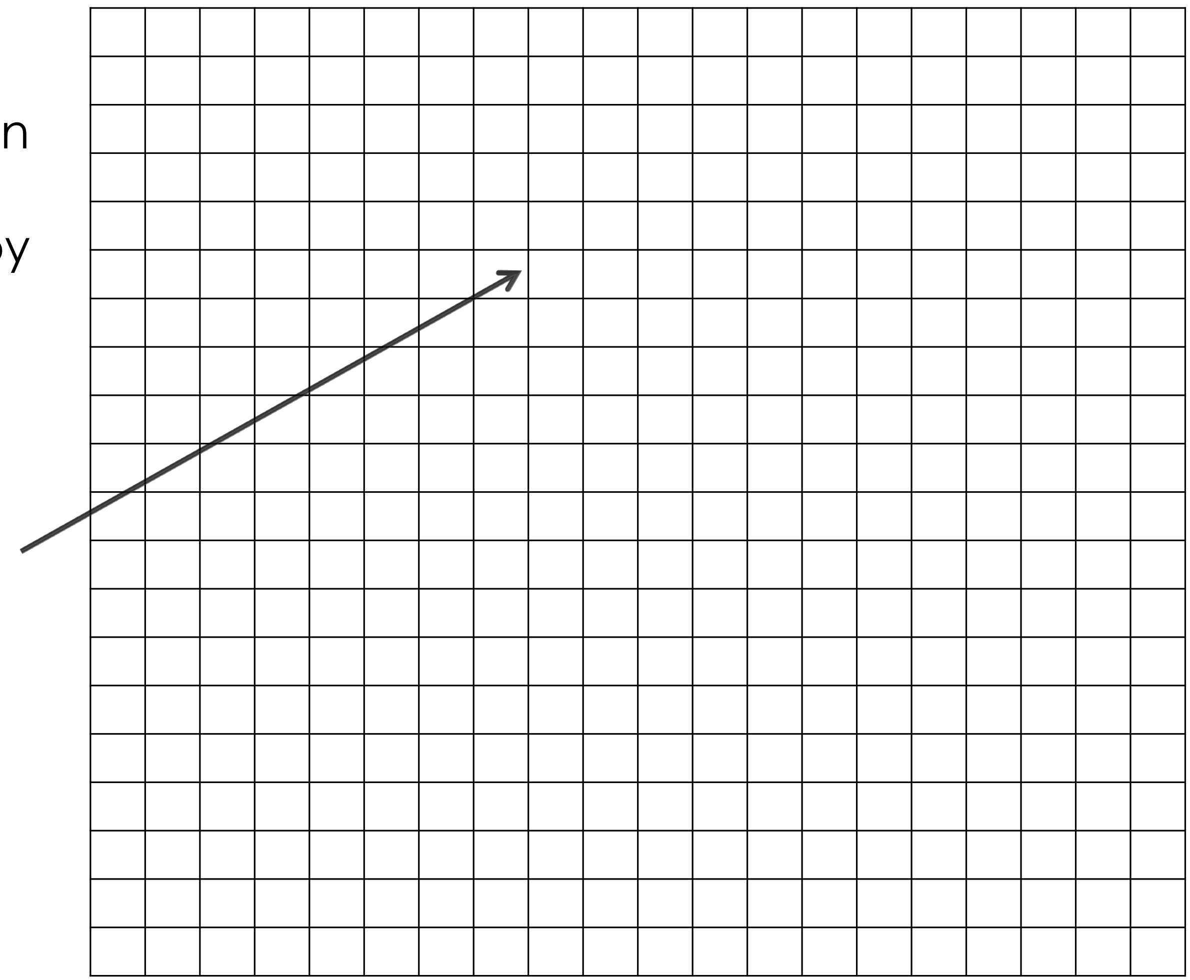
Modeling the Spread of Flu Virus

- Every person is either healthy, infected, contagious or immune. We assume that "infected" means infected but not contagious.
- Each day, a healthy person comes in contact with 4 random people. If any of those random people is contagious, then the healthy person becomes infected.
- It takes one day for the infected person to become contagious.
- After a person has been contagious for 4 days, then the person is non-contagious and cannot spread the virus nor can the person get the virus again due to immunity.

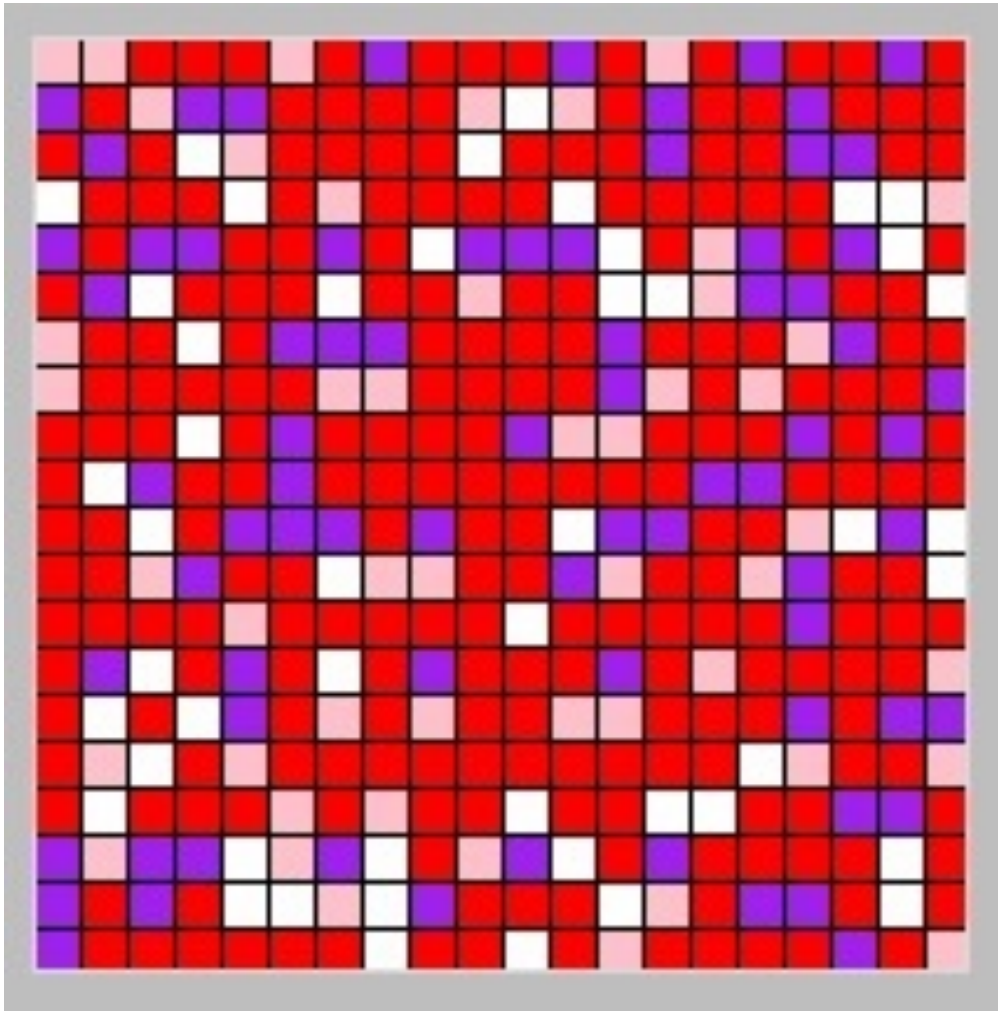
Representing the Population

Each person uniquely identified by a row and column

Person identified by row = 5, column = 7

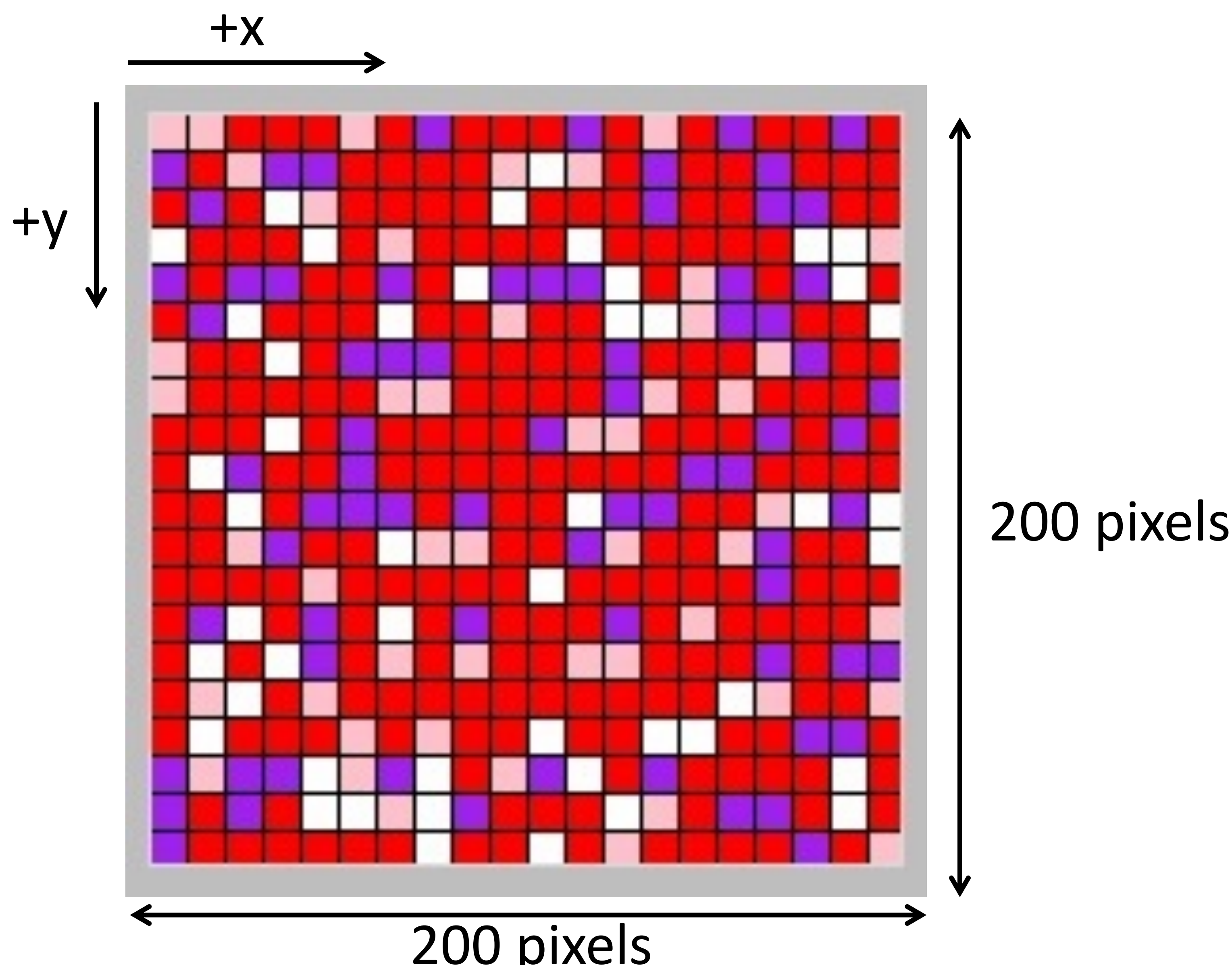


Displaying the Population

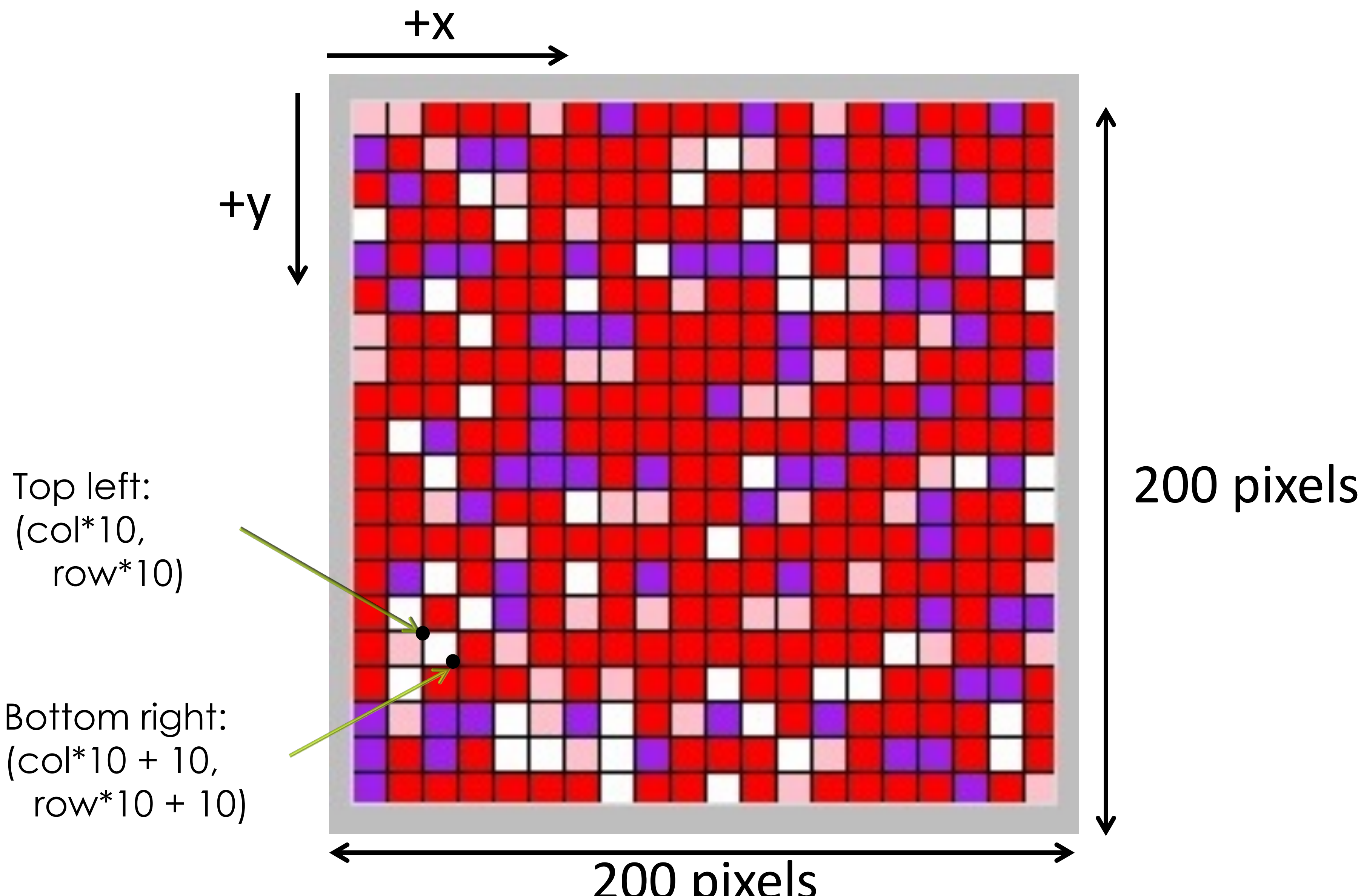


Color od each
cell indicates
the health state
of the person
corresponding to
that cell

Graphics

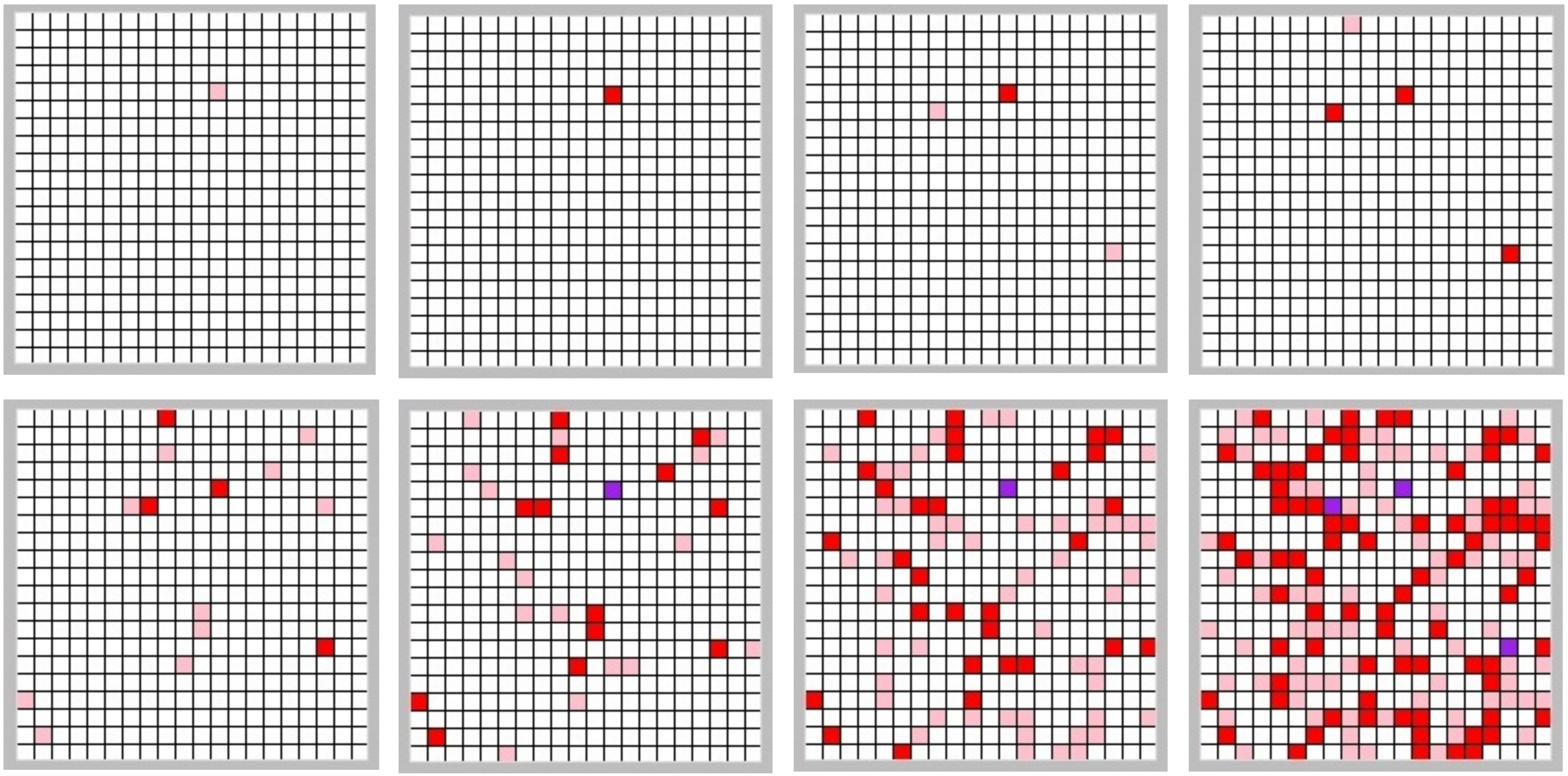


Coordinates of each cell



Graphical Simulation

Simulation captures the evolution of the health state of the population over time.
It evolves in discrete steps: change occurs instantaneously as a new day begins.



Overview of the code

- Constants used for representing health states of individuals
- Update function: Given the state of the population on some day (input `matrix`), calculates the state of the population for the next day (output `newmatrix`)
- Since we have a 2-dimensional input (`matrix`) a natural way to traverse the input structure for creating an output structure is a nested loop. We go over the input row by row, for each row column by column

Health States

0	white	healthy
1	pink	infected
2	red	contagious (day 1)
3	red	contagious (day 2)
4	red	contagious (day 3)
5	red	contagious (day 4)
6	purple	immune (non-contagious)

```
HEALTHY = 0
INFECTED = 1
DAY1 = 2
DAY2 = 3
DAY3 = 4
DAY4 = 5
IMMUNE = 6
```

The health state of the population will be represented using a 20 by 20 matrix where each entry has one of the values above.

Checking Health State

```
def immune(matrix, i, j):
    return matrix[i][j] == IMMUNE


def contagious(matrix, i, j):
    return matrix[i][j] >= DAY1 and matrix[i][j] <= DAY4


def infected(matrix, i, j):
    return matrix[i][j] == INFECTED


def healthy(matrix, i, j):
    return matrix[i][j] == HEALTHY
```

Updating the matrix

```
def update(matrix):
    # create new matrix, initialized to all zeroes
    newmatrix = []
    for i in range(20):
        newmatrix.append([0] * 20)
    # create next day
    for i in range(20):
        for j in range(20):
            if immune(matrix, i, j):
                newmatrix[i][j] = IMMUNE
            elif infected(matrix, i, j) or
                            contagious(matrix, i, j):
                newmatrix[i][j] = matrix[i][j] + 1
            elif healthy(matrix, i, j):
                for k in range(4):  # repeat 4 times
                    if contagious(matrix,
                                   randrange(20),randrange(20)):
                        newmatrix[i][j] = INFECTED
    return newmatrix
```

Updating the matrix

```
def update(matrix):
    # create new matrix, initialized to all zeroes
    newmatrix = []
    for i in range(20):
        newmatrix.append([0] * 20)
    # create next day
    for i in range(20):
        for j in range(20):
            if immune(matrix, i, j):
                newmatrix[i][j] = IMMUNE
            elif infected(matrix, i, j) or
                            contagious(matrix, i, j):
                newmatrix[i][j] = matrix[i][j] + 1
            elif healthy(matrix, i, j):
                for k in range(4):  # repeat 4 times
                    if contagious(matrix,
                               randrange(20),randrange(20)):
                        newmatrix[i][j] = INFECTED
    return newmatrix
```

Note the programming idiom.
We use an expression
that already has a Boolean value
instead of
`immune(matrix, i, j) == True`

Displaying the matrix

```
def display(matrix,c):
    for row in range(len(matrix)):
        for col in range(len(matrix[0])):
            person = matrix[row][col]
            if person == HEALTHY:
                color = "white"
            elif person == INFECTED:
                color = "pink"
            elif person >= DAY1 and person <= DAY4:
                color = "red"
            else:          # non-contagious or wrong input
                color = "purple"
            c.create_rectangle(col*10, row*10, col*10 + 10,
                                         row*10 + 10, fill = color)
```

Create_rectangle (topleft_x, topleft_y, bottomright_x, bottomright_y, optional params)

Testing `display`

```
def test_display():
    window = tkinter.Tk()
    # create a canvas of size 200 X 200
    c = Canvas(window,width=200,height=200)
    c.pack()
    matrix  = []
    # create a randomly filled matrix
    for i in range(20):
        row = []
        for j in range(20):
            row.append(randrange(7))
        matrix.append(row)
    # display the matrix using your display function
    display(matrix,c)
```

```python
def test_update():
    window = tkinter.Tk()
    # create a canvas of size 200 X 200
    c = Canvas(window,width=200,height=200)
    c.pack()
    # initialize matrix to all healthy individuals
    matrix= []
    for i in range(20):
        matrix.append([0] * 20)
    # infect one random person
    matrix[randrange(20)][randrange(20)] = INFECTED
    display(matrix,c)
    # Canvas.delay = 3
    sleep(0.3)
    # run the simulation for 10 "days
    for day in range(0, 10):
        c.delete(tkinter.ALL)
        matrix = update(matrix)
        display(matrix,c)
        sleep(0.3)
        c.update() #force new pixels to display
```

Running the Code

```
import tkinter
from tkinter import Canvas
from random import randrange
from time import sleep


# Constants for health states of an individual

HEALTHY = 0
INFECTED = 1
DAY1 = 2
DAY2 = 3
DAY3 = 4
DAY4 = 5
IMMUNE  = 6
```

What if Our Model Changes?

- If a healthy person contacts a contagious person, she gets sick 40% of the time.

```
if contagious(matrix,randrange(20),randrange(20))
                 and randrange(100) <40:
     newmatrix[i][j] = INFECTED
```

What if Our Model Changes?(cont'd)

- The current model does not capture neighbor relationship. The adjacency of 2 cells does not indicate that they are neighbors.

- What if we used to grid to capture neighbor relationship and assumed that a healthy person gets infected if they have at least one contagious neighbor.

Neighbors

```
cell = matrix[i][j]

north = matrix[i-1][j]          NO!


if i == 0:                      YES!

  north = None

else:

  north = matrix[i-1][j]
```

Continuous Simulation

N-Body Problem

Continuous-Time Simulations

- Often used to model physical phenomena involving forces acting on objects.
- Is "time" really continuous?
  - Philosophical question. No one knows.
  - Just pretend it is.
- Is simulated time continuous?
  - No. It's divided into discrete time steps.
  - But they can be as small as we like.

N-Body Problem

- Newton's theory: Planets and other bodies move according to the gravitational effects of the objects around them
- N-body problem: Predicting the individual motions of a group of objects interacting with each other gravitationally
- With just two bodies, we can write a simple formula to calculate their positions at any future time, given their starting positions.
- *But with 3 or more bodies, no formula exists for this, because the system is highly nonlinear, and potentially chaotic.*
- Our only recourse is simulation.

N-Body Simulation

- Using simulation to predict future locations of bodies
  - Astronomers use simulations to predict locations of satellites, plan space travel, track dangerous asteroids etc.

- Main idea of the simulation: Start with the current location and heading of each planet. Then repeatedly
  - Determine where the planets would be a short time later if they move according to a straight line
  - Calculate adjustments to headings

Simulating Gravitational Attraction

Newton's law of universal gravitation:

$$F = G \cdot m_1 \cdot m_2 / d^2$$

where G = gravitational constant,
m_1 and m_2 are the masses, and
d is the distance between them.

Force and Acceleration

- Newton's second law: if some external force is applied to a body then the body accelerates (its velocity changes)

$$F = ma$$

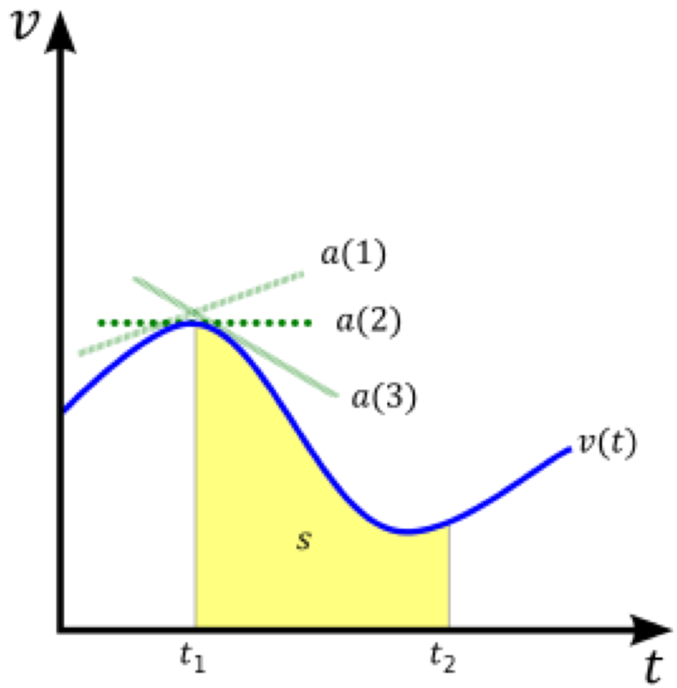
mass acceleration

Moving A Single Body

- Calculate the force and acceleration influencing the body at a given time

- Suppose that acceleration is constant for a given interval of time and calculate the velocity and distance moved

Velocity versus Time graph



a lines represent the values for acceleration at different points along the curve and the yellow area under the curve represents displacement ***s***

Source: Wikipedia

Integrating Acceleration

- When an object accelerates, its velocity $v(t)$ changes. How can we model this?
- Divide time into tiny steps Δt.
- Re-calculate the velocity at each time step.

  $$v(t + \Delta t) = v(t) + a(t) \cdot \Delta t$$

- Smaller Δt brings greater accuracy. Why?

Velocity Is Rate of Change of Position

- If an object has non-zero velocity, its position is changing.

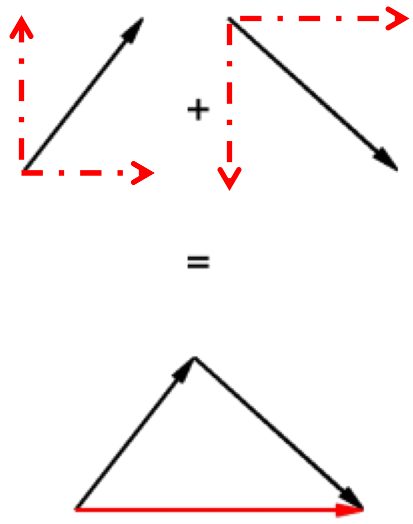
- We can use the same integration trick to update the body's position based on velocity.

  $$x(t + \Delta t) = x(t) + v(t) \cdot \Delta t$$

Force Vectors

- We can use vectors to keep track of positions, velocities, and accelerations: (x, y, z) coordinates
- Forces are additive and vector addition performs ordinary addition on each component:

$$(x1, y1, z1) + (x2, y2, z2) = (x1+x2, y1+y2, z1+z2)$$



The vectors in this example has 0 for the z coordinate.

The north and south vectors cancel out each other

The east vectors add up

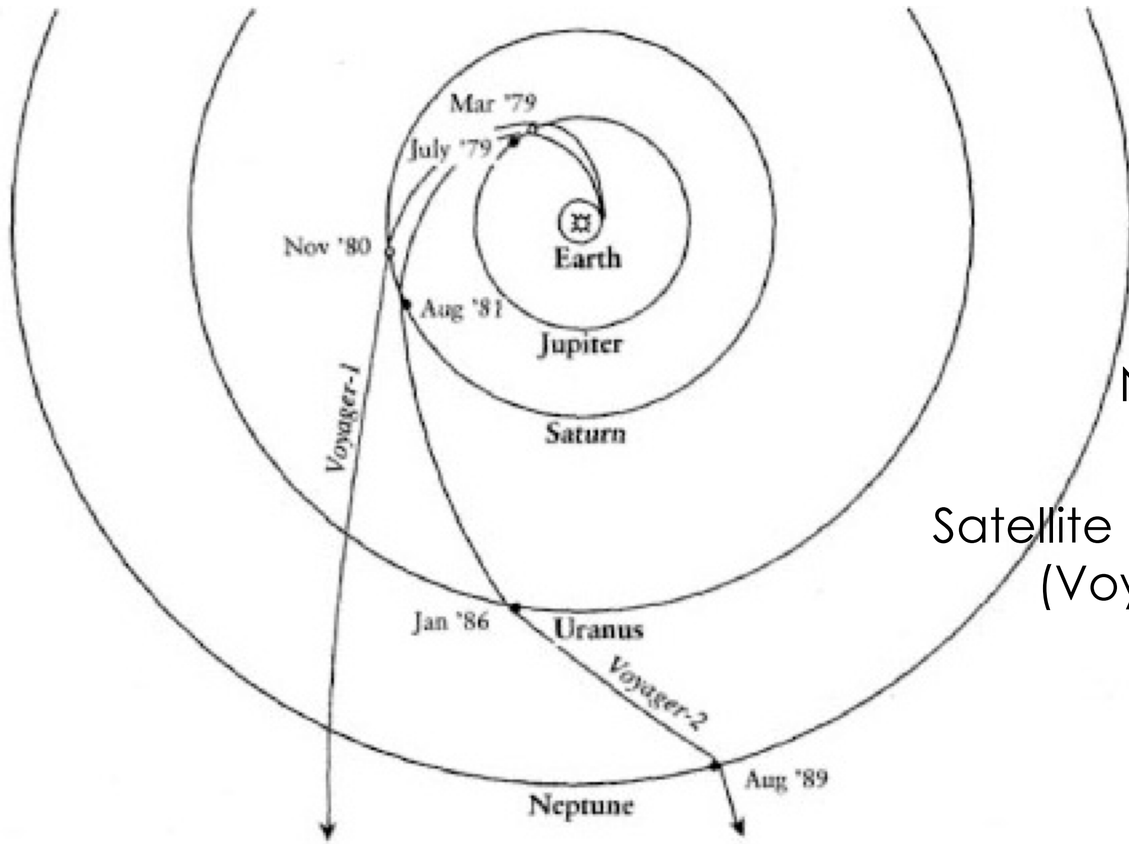
Force Action on a Single Body

- Calculate all the force vectors influencing the body
- Add the vectors together to determine the cumulative force

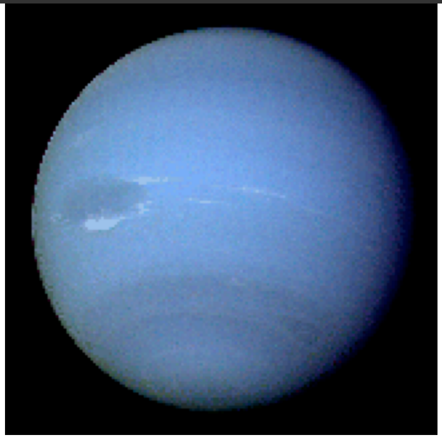
Moving Multiple Bodies

- At each time step move each body by calculating the force vectors in each direction
- Suppose we are given a method interaction(a,b) that calculates the gravitational force between the bodies a and b
- We need to compute all pairwise interactions.
- How many force calculations are there?
  - For the first body interactions with each of the remaining N-1 bodies, for the second one interactions with each of the remaining N-2 bodies because we already took into account its interaction with the first one etc.
  - $N-1 + N-2 + \ldots 1 = N \times (N-1)/2 \Rightarrow O(N^2)$

Paths of Voyager 1 and 2

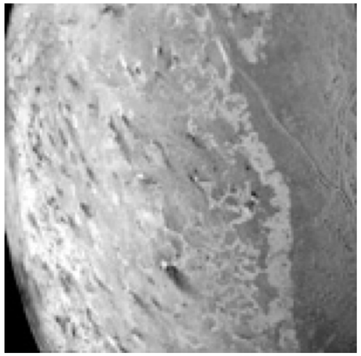


frankie0701.wordpress.com



Neptune (Voyager 2, NASA)

Satellite 1989N1, (Voyager 2, NASA)



Triton, (Voyager 2, NASA)

Simulation At Extreme Scales

- Cosmologists use simulations to study the formation of galaxies (clusters of stars), and even clusters of galaxies.
- At the other extreme, physicists simulate individual atoms and molecules, e.g., to model chemical reactions.