Discrete Simulation

Announcements

- Lab 10 tonight
- PS9 due tomorrow (9:00)
- PA8 due today (Noon)

- Exam 2 on 1st (Thursday!)
- Units 6, 7, 8, 9, 10

Changing Labs

- Lab 11 Now on Thursday Aug 1
- Lab 12 Now on Wednesday Aug 7

Exam 2

- Review Session Wed 4:30 – 6:30 Room 4307
- 80 minutes (full class period)
- Questions?

Last Week

- How to generate pseudo-random numbers
- Using randomness in interesting applications
- Monte Carlo simulations
  - Run many experiments with random inputs
  - Approximate an answer when an analytical solution is difficult/infeasible to obtain

Understanding Systems

- **Data Visualization** and Simulations are different
- We try to visualize the results of simulations to make it easy to see/understand the systems…
- ...because generally what we try to see/understand or predict is complicated because of the nature of systems.

Systems

- Collection of tracks, railway cars, infrastructure: railroad system
- Collection of hardware and software: computer system
- Collection of educations, students, infrastructure: school system

Dynamic, Interactive, Complicated

How Can we Study a System?

- Experiment with the **actual system**
- Experiment with a **model of the system**
  - **Physical model**
    - May not exist, be unsafe, be expensive to build and modify, or change too slowly over time
  - **Mathematical model**
    - **Analytical solution** (Equations or systems may be too complex for closed-form or analytical solution)
    - ***Simulation:*** The imitative representation of the functioning of one system or process by means of the functioning of another, for example a computer program.

Law and Kelton: Simulation, Modeling and Analysis

Computer simulation is a process of making a computer behave like a cow, an airplane, a battlefield, a social system, a terrorist, a HIV virus, a growing tree, a manufacturing plant, a mechanical system, an electric circuit, a stock market, a galaxy, a molecule, or any other thing. This is done with a specific purpose, mainly in order to carry out some **"what if" experiments** over the computer model instead of the real system.

Modeling and Simulation,
S. Raczynski

Uses of Simulation

- **Testing:** Performance optimization, safety engineering, testing of new technologies.
- **Predicting:** Gaining a better understanding of natural and human systems, and making predictions.
- **Training:** Providing lifelike experiences in training, education, games.

Large Scale Simulations

- Computing power of today enables large scale simulations. For example,
  - Department of Defense: Battle simulations
  - National Center for Atmospheric Research : 1,000 years of climactic changes http://www.youtube.com/watch?v=d8sHvhLvfBo
  - Blue Brain Project at EPFL to reverse engineer the human brain

  http://www.youtube.com/watch?v=ySgmZOTkQA8

Advantages of Using Simulation

- With simulation, we can
  - Control sources of variation
  - Choose the scale of time
  - **Stop and review**
  - **Replicate results** more easily

Models

- A *model* is an abstraction of the real system. It represents the system and the rules that govern the behavior of the system.

- The model represents the system itself, whereas the simulation represents the operation of the system over time.

Modeling Concerns

- Abstraction: Accuracy vs. Complexity
  - Most relevant factors
- How important is it to capture continuous behavior over time? (Discrete vs. Continuous models)
  - Discrete models: essential variables are enumerable, e.g., integers
  - Continuous models: essential variables range over non-enumerable sets such as real numbers
- Do parts of the system exhibit random behavior? (Deterministic vs. stochastic models)

our focus in this lecture

Computational Science

- Computational sciences use computational models (special kind of mathematical models) as the basis of obtaining scientific knowledge.
- Unifies
  - Modeling, algorithms, simulations
  - Computing environment developed to solve science, engineering, medicine, and humanities problems
- Helps explain and predict phenomena using a mechanistic view

Simulation Models are Descriptive

- They tell us how a system works under given conditions but not how to set the conditions to make the system work best
- Simulation does not “optimize” but it helps us in finding an optimal set of parameter settings.

DISCRETE SIMULATION: A Simple Example

Discrete Time and Discrete Events

- Real time vs. model time
  - In simulating the movements of a galaxy one hour simulation may cover billions of years
- In discrete simulation we assume time changes in discrete steps (ticks) and the states of simulated entities change instantaneously

Discrete Simulation of Disease Spread

- We are going to use a dynamic, discrete, stochastic simulation model
  - We want to capture how the disease spreads over time
  - We model time discretely as a sequence of days, and use discrete variables to capture the health state of each person
  - There is randomness in how the virus spreads
- Simulate the system execution as a sequence of discrete events that change the state of the system instantaneously at each time step

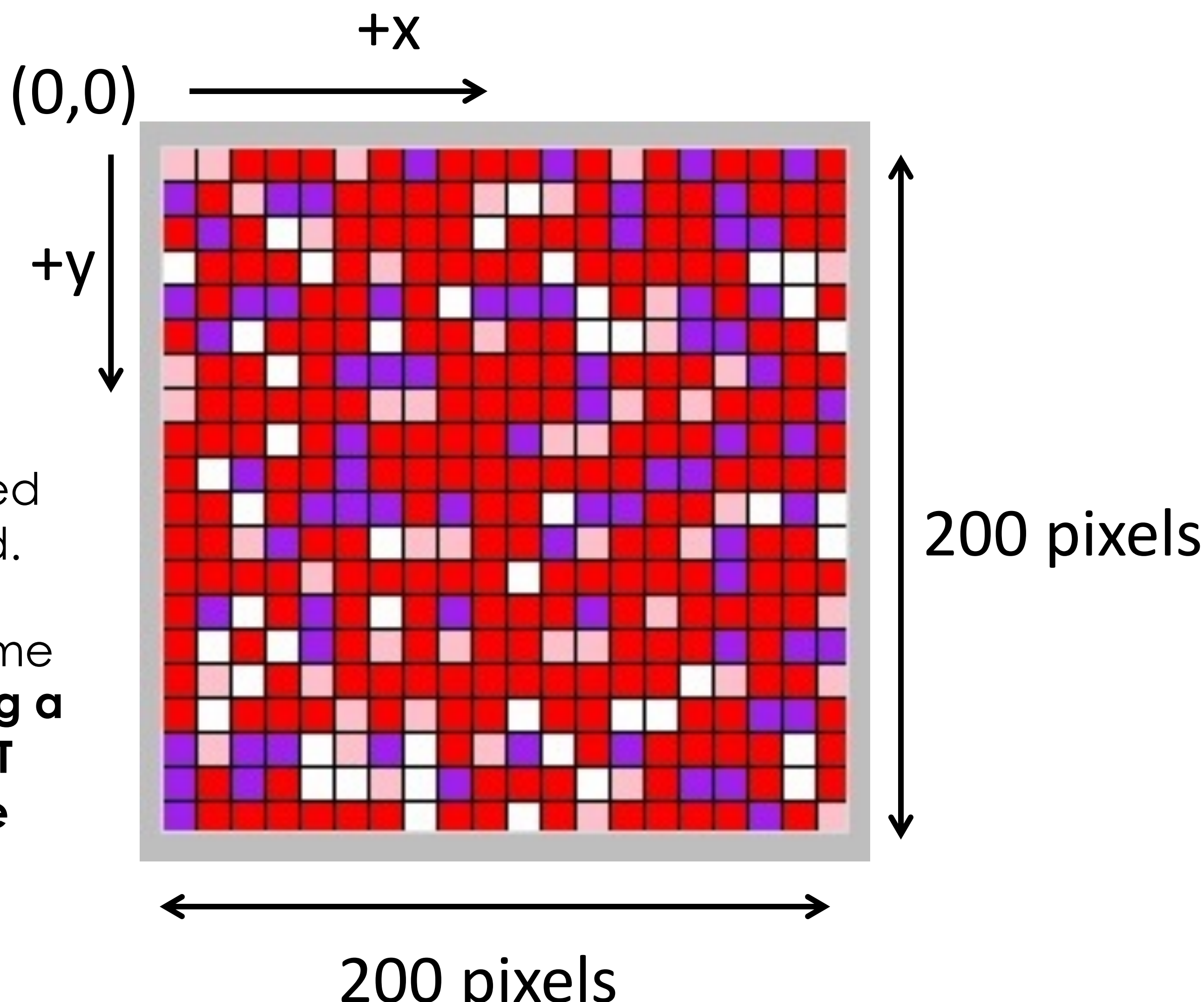
Example: Flu Virus Simulation

- Goal: Develop a simple simulation that shows graphically how disease spreads through a population.

Modeling the Spread of Flu Virus

- Every person is **healthy**, **infected**, **contagious,** or **immune**.
  - An infected person is 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 immune and cannot spread the virus nor can the person get the virus again due to immunity.

Displaying the Population

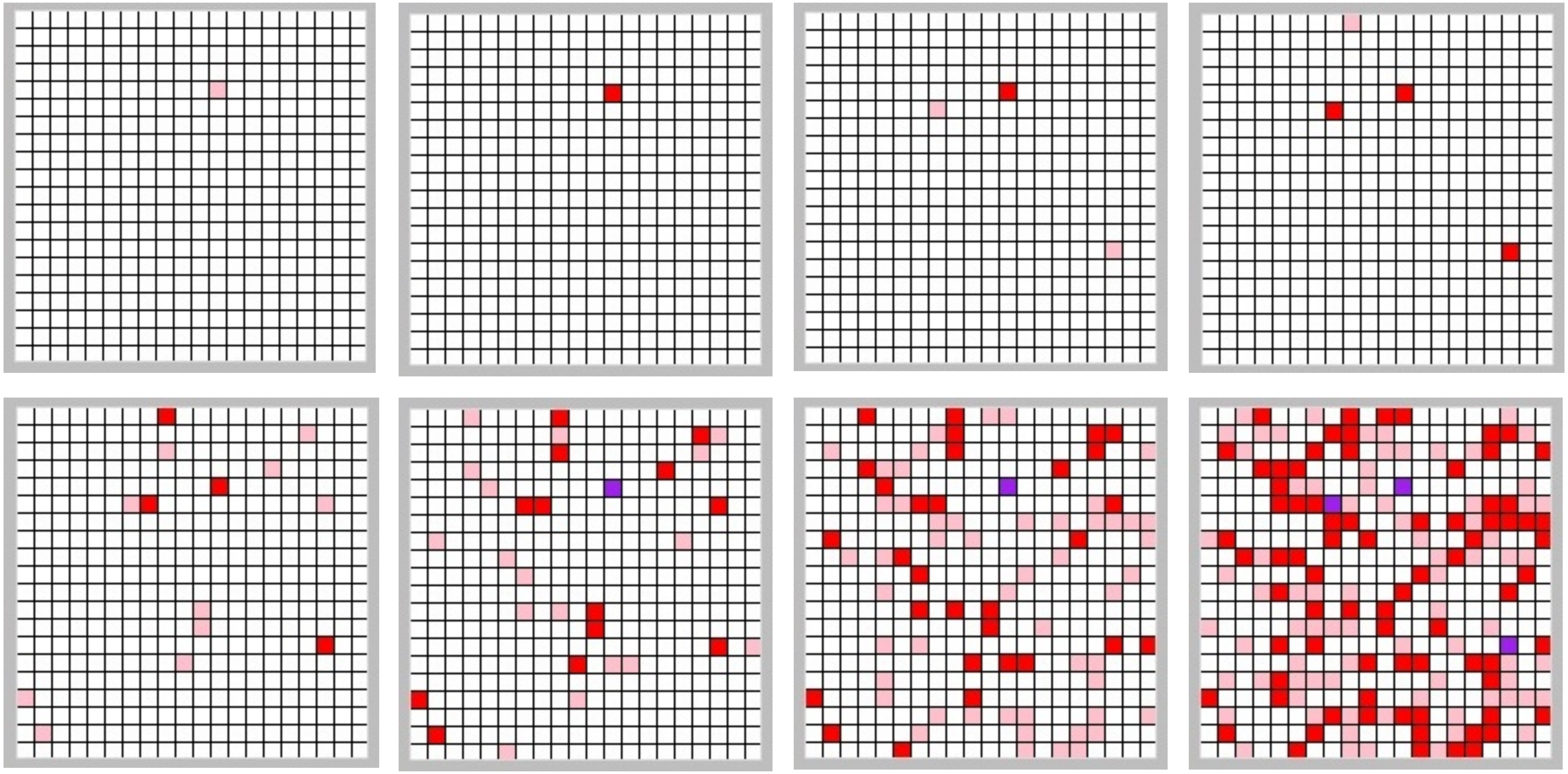


Assumption:
The population consists of 400 people each of which is represented by a cell in the grid.

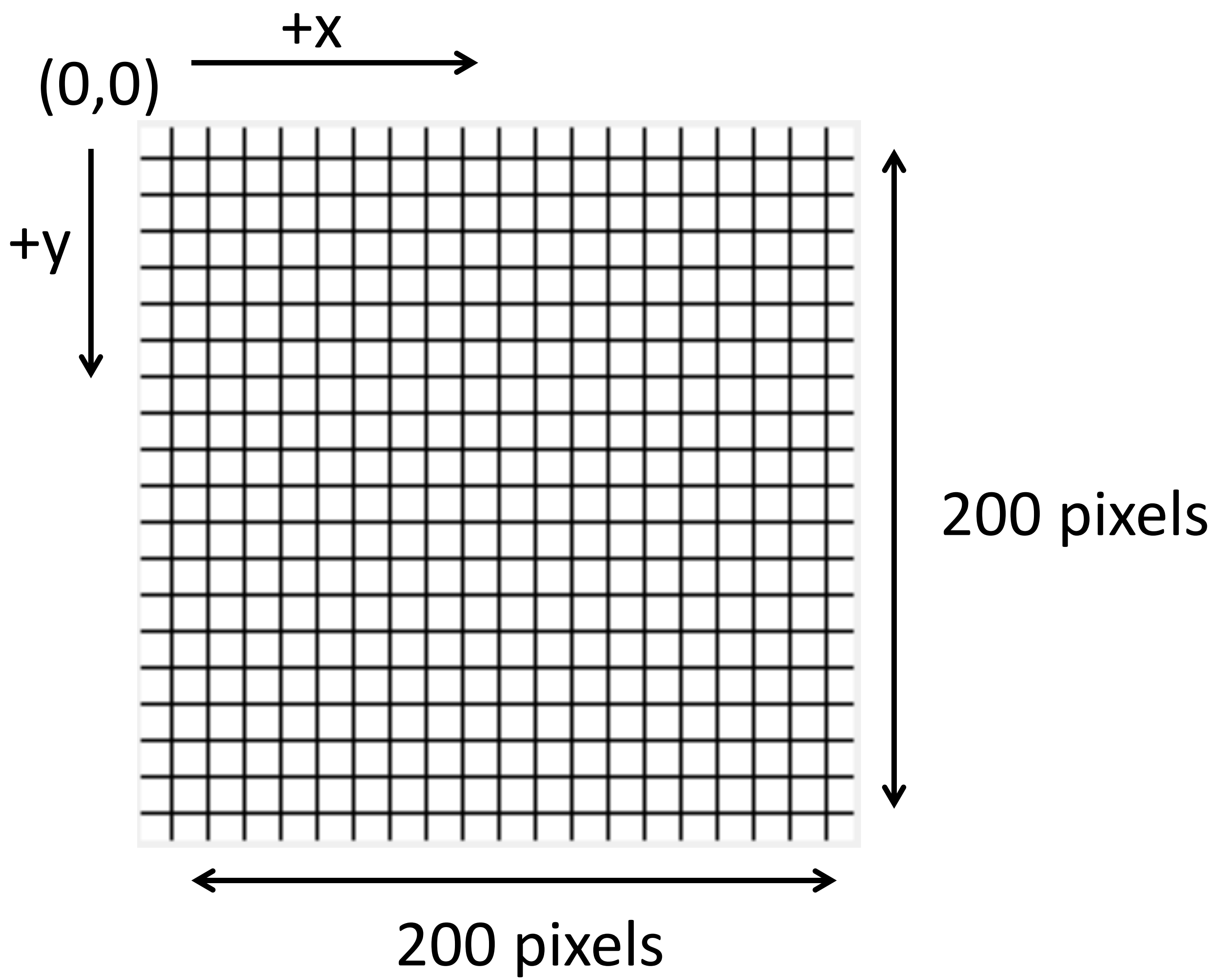
For simplicity, assume that **two cells being a adjacent does NOT mean that they are physically close**.

Graphical Simulation

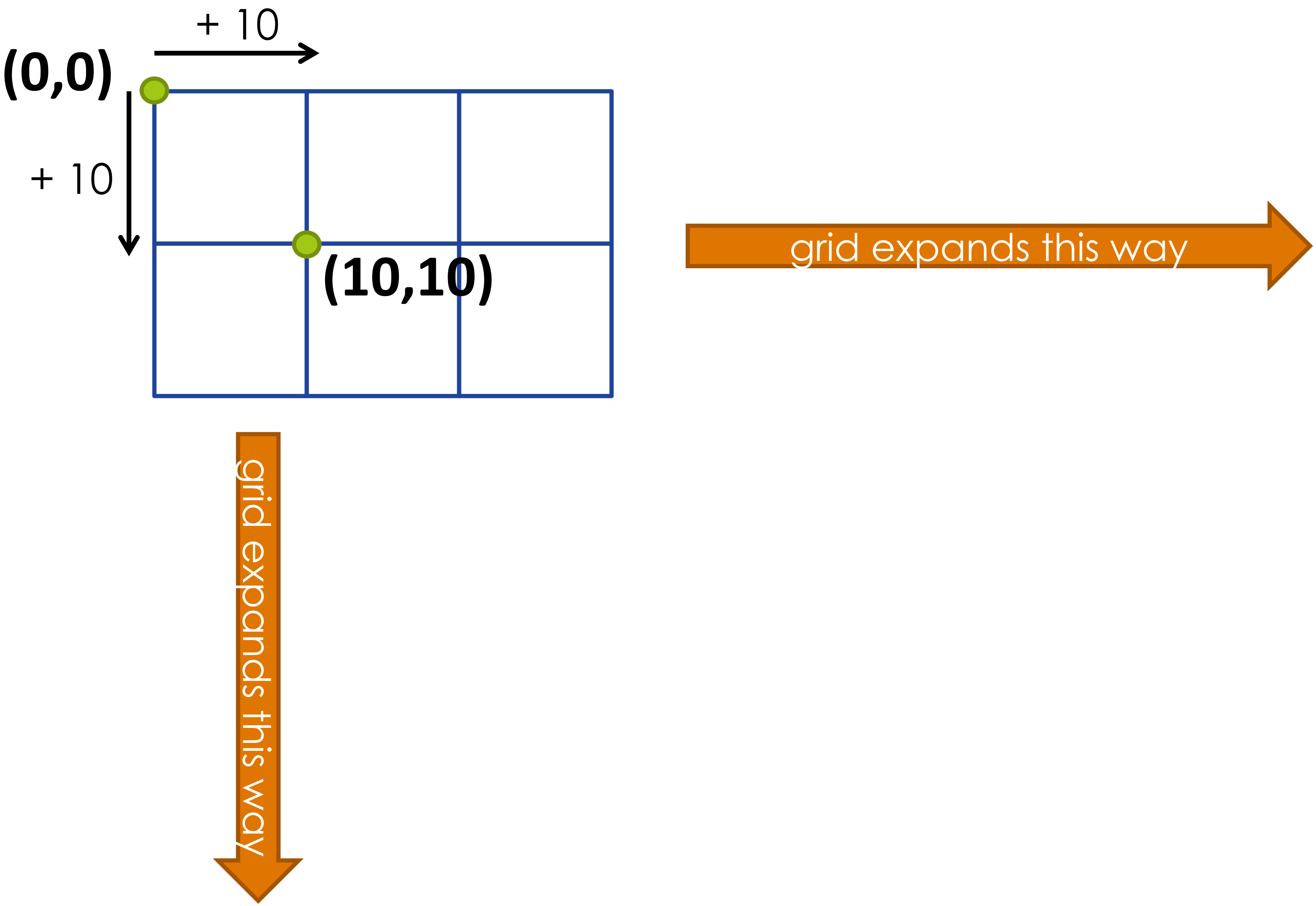
Simulation captures the evolution of the health state of the population over time.
It evolves in discrete steps: change occurs instantaneously as a new day begins.



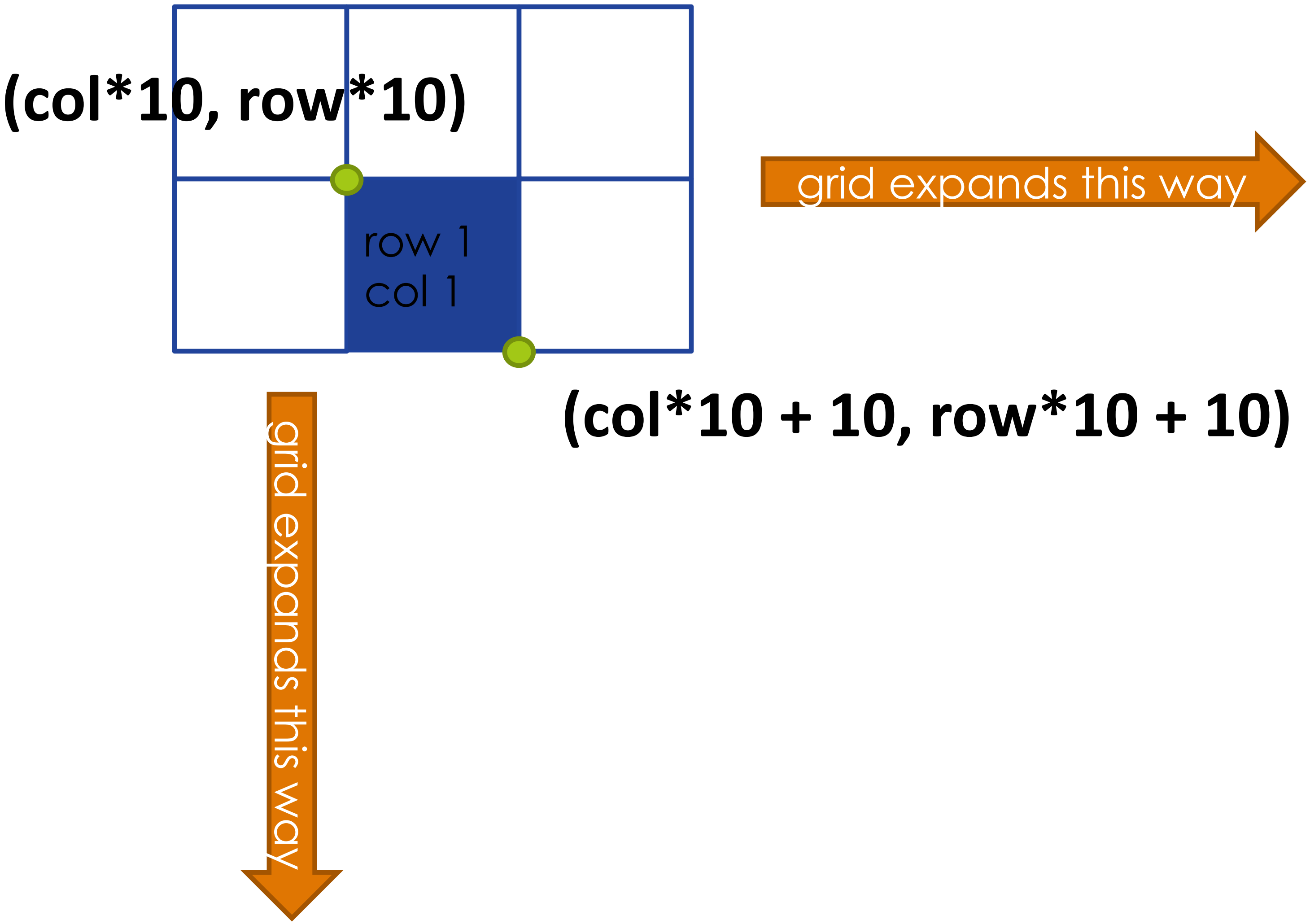
Displaying the Population



Displaying One Person



More Generally For Any Person



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

```
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.

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
```

We use an expression that already has a Boolean value instead of a test with "=="

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)
```

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
```

```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 a 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?

- 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:
   north = None
else:
   north = matrix[i-1][j]
```

YES!

Next Time

- Continuous simulation