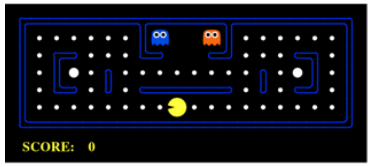
Solve a Maze via Search

An Intro-AI Exploration

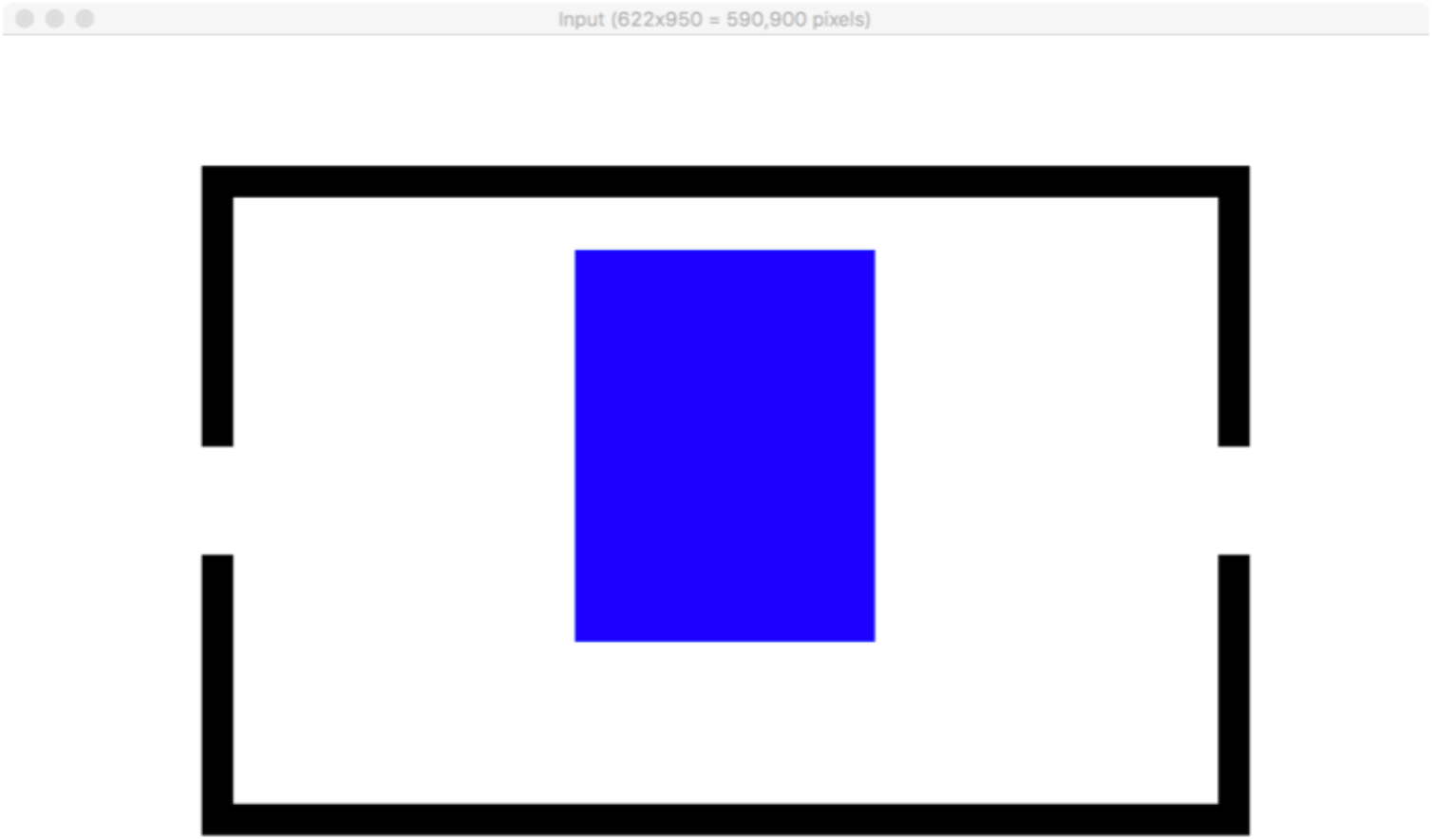
Context

- Intro AI end-of-course project
  - Used as "default" option for several semesters
    - And starter for undergrads interested in AI research!
  - End-to-end application with guidance & base
  - About 150 (well-written) lines

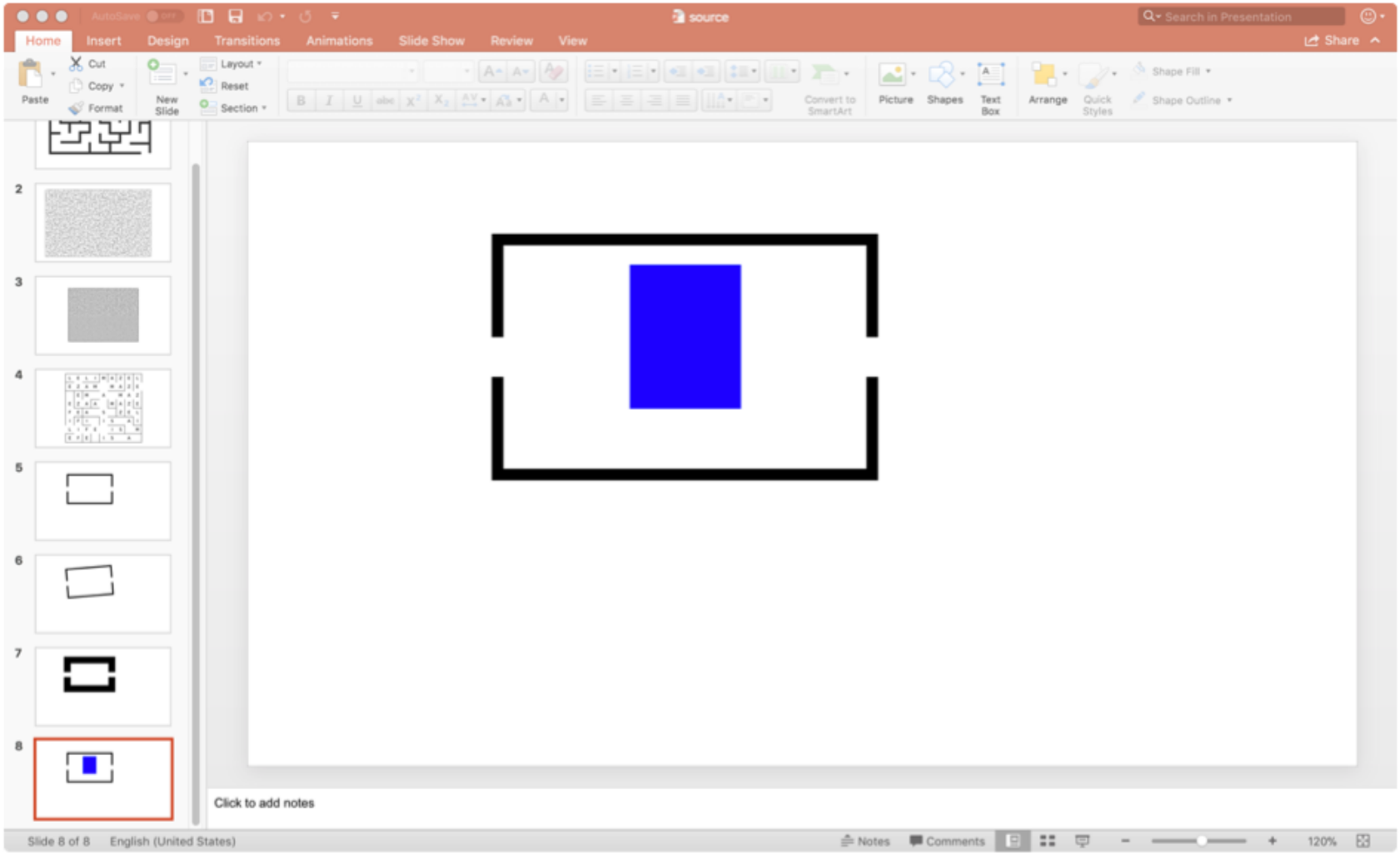
- Builds on Pacman Projects
  - While still maze-based, adds significant room for exploration/growth via OpenCV



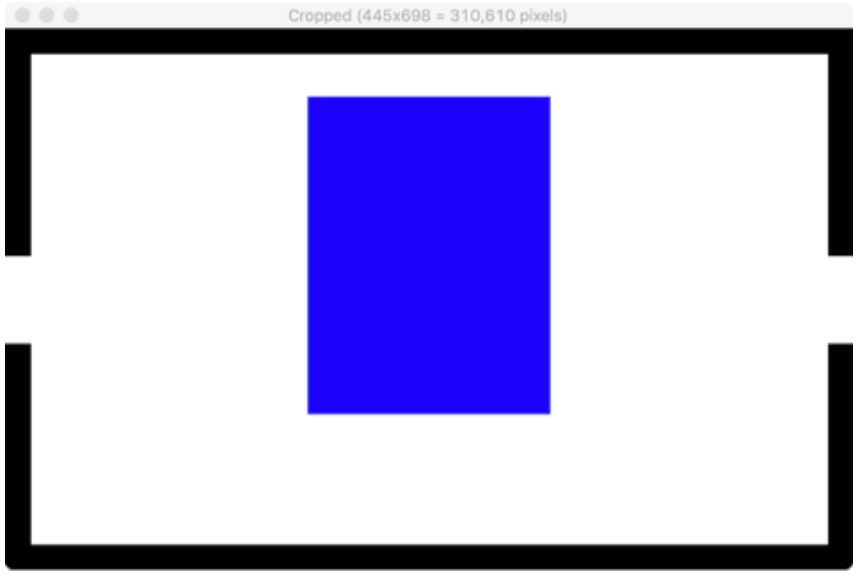
Input: Image

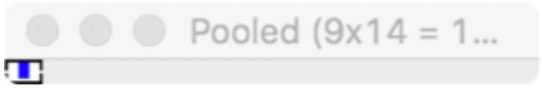


Source: You!


source
Search in Presentation
Home
Insert
Design
Transitions
Animations
Slide Show
Review
View
Share
Paste
Cut
Copy
Format
New Slide
Layout
Reset
Section
Convert to SmartArt
Picture
Shapes
Text Box
Arrange
Quick Styles
Shape Fill
Shape Outline
Click to add notes
Slide 8 of 8
English (United States)
Notes
Comments
120%

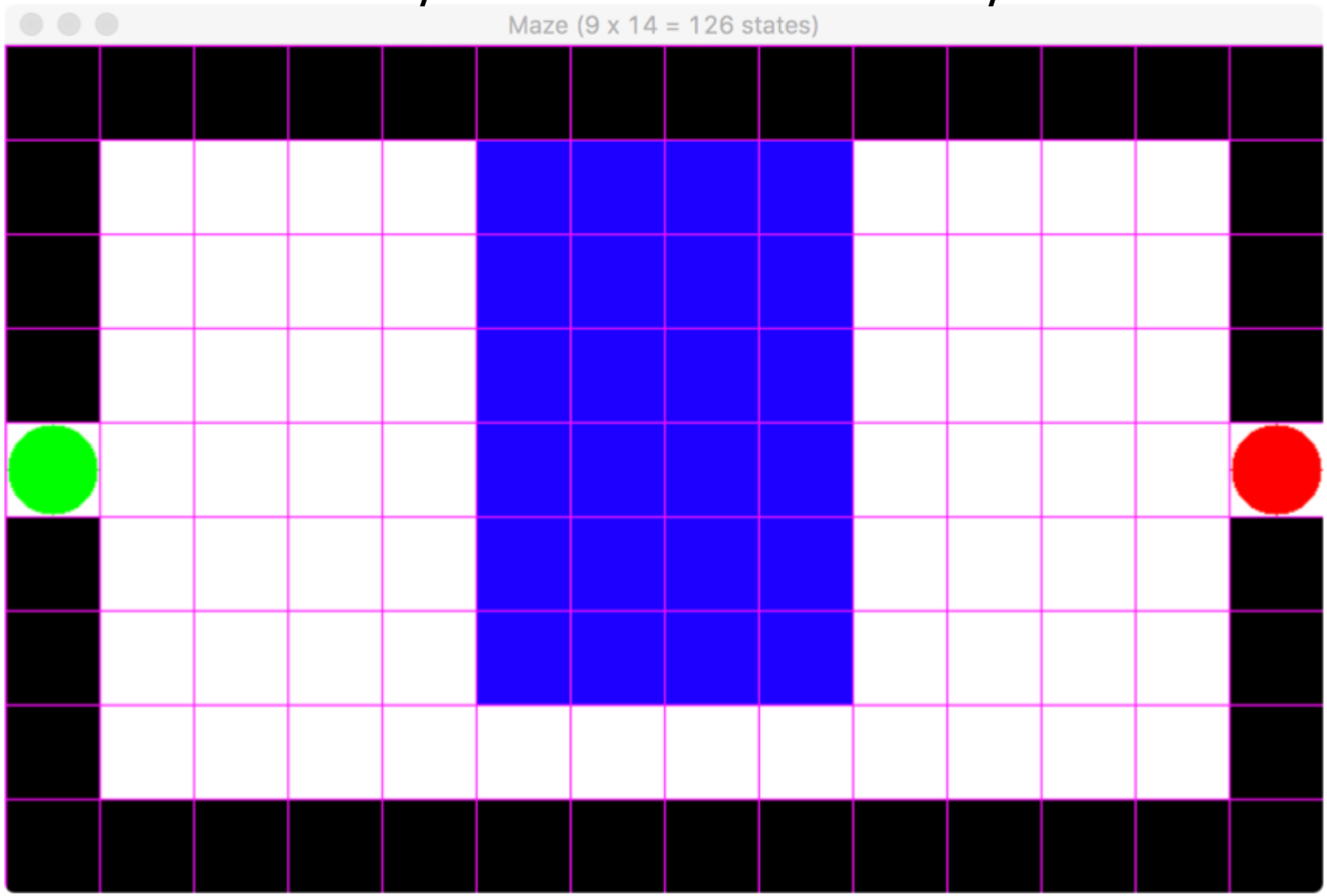
Preprocessing via OpenCV



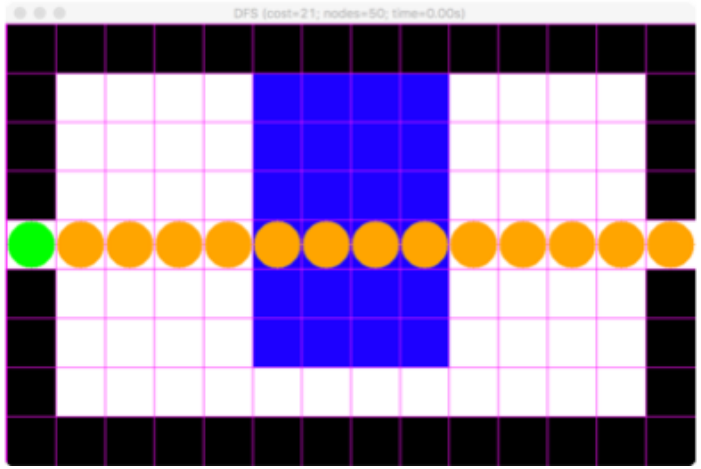


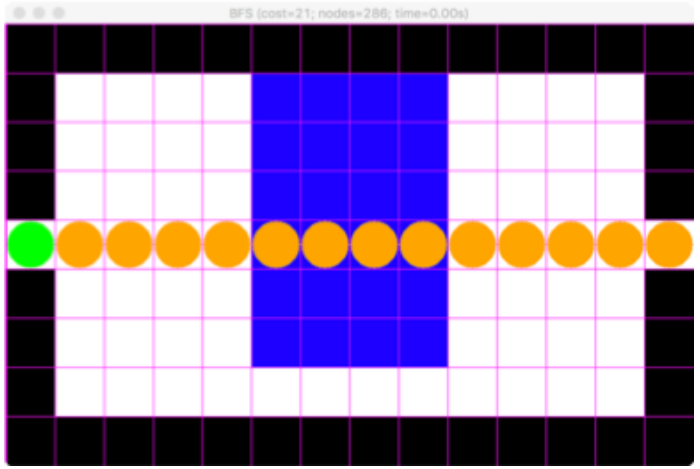
Grid via Nearest-Color

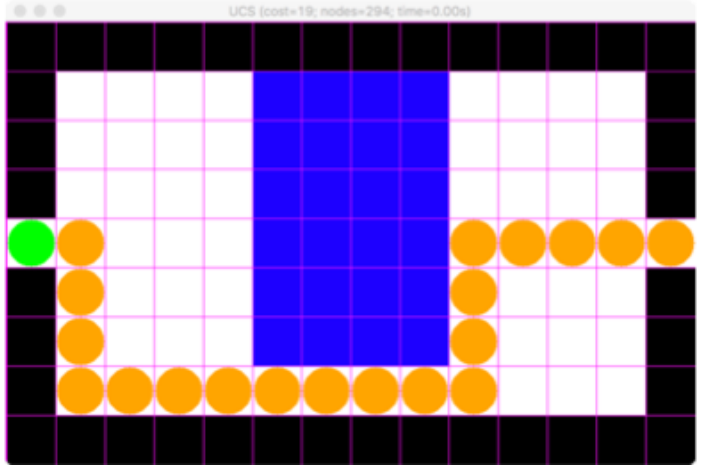
Start/Stop via Border Assumptions

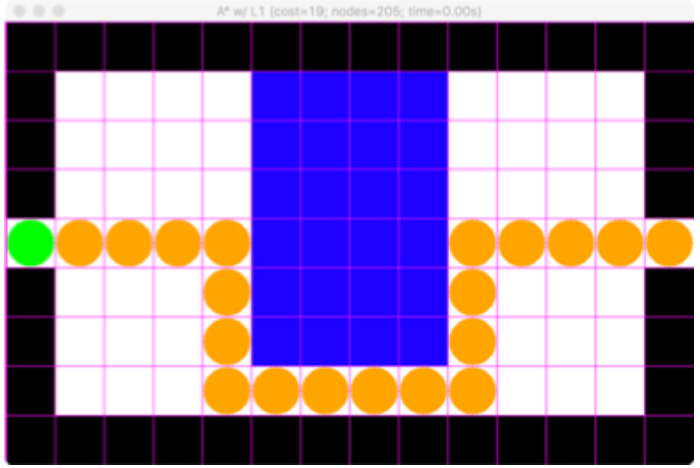


Solve via DFS, BFS, UCS, A*

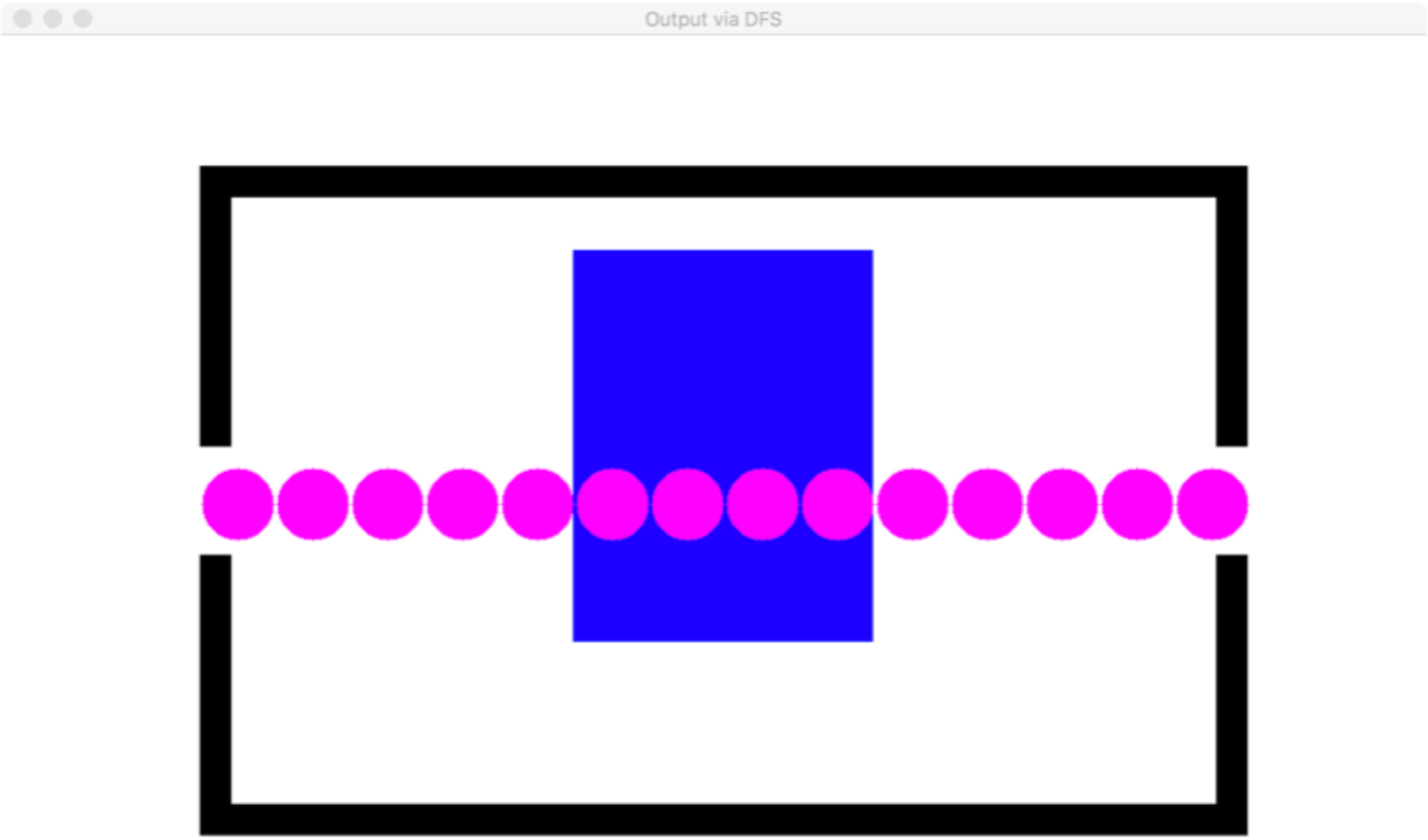




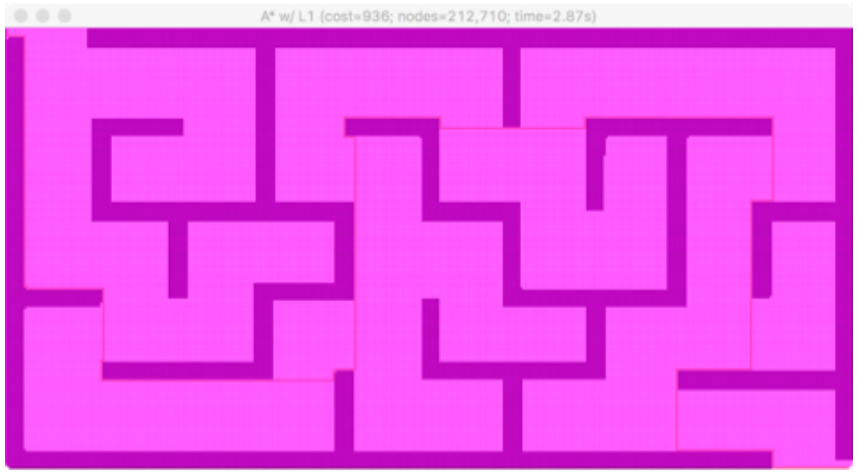


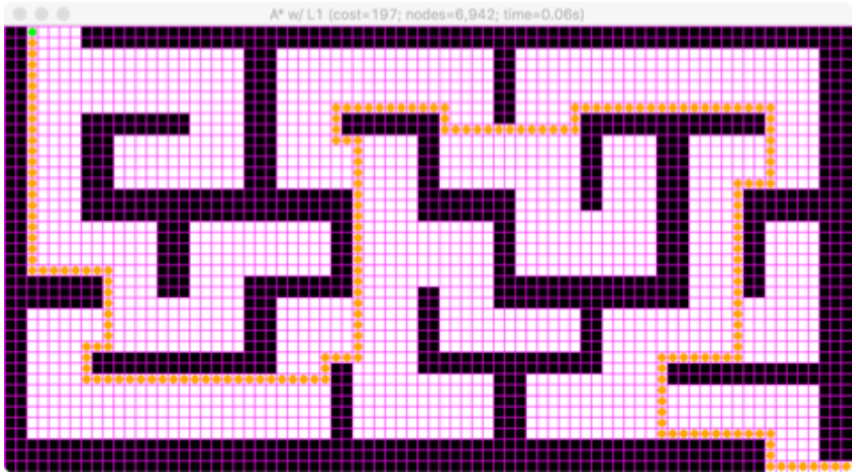


Overlay on Original

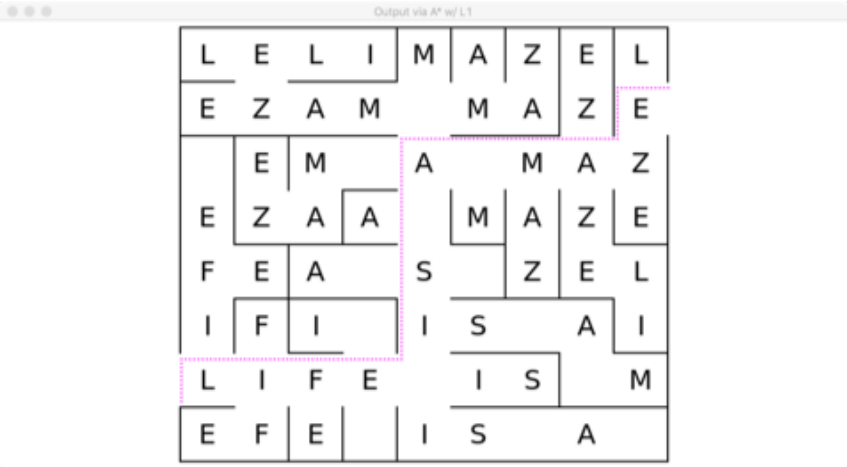


Representation/Performance





Scaling Up


Output via A* w/ L1
L E L I M A Z E L
E Z A M M A Z E
E M A M A Z
E Z A A M A Z E
F E A S Z E L
I F I I S A I
L I F E I S M
E F E I S A


Output via A* w/ L1

Given

Assignment Write-up

- Software installation
  - OpenCV in 1-3 lines
- Pointers to concepts/ tutorials/documentation
- Sequencing + examples with invocation/images
- Ideas for extension

Starter Code

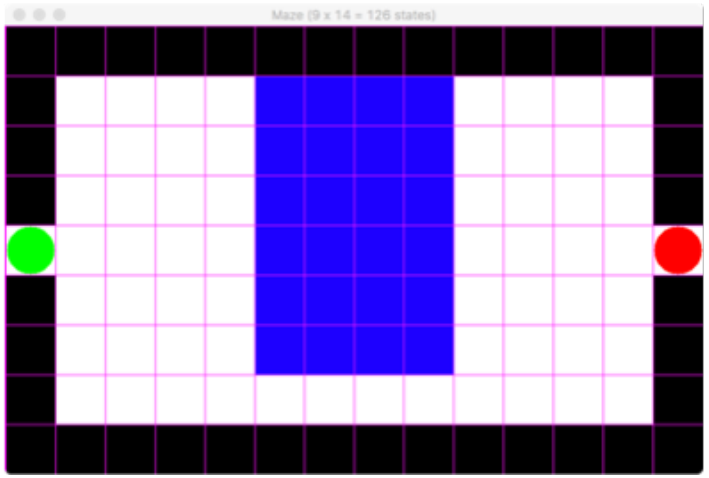
- Extracted portions of Pacman (w/ license)
- Scaffolding, utility code
- Fill-in-the-code with comments
- Example inputs, PowerPoint for more

1. Visualize a Maze (~15 lines)

- Given TXT, output a visual representation

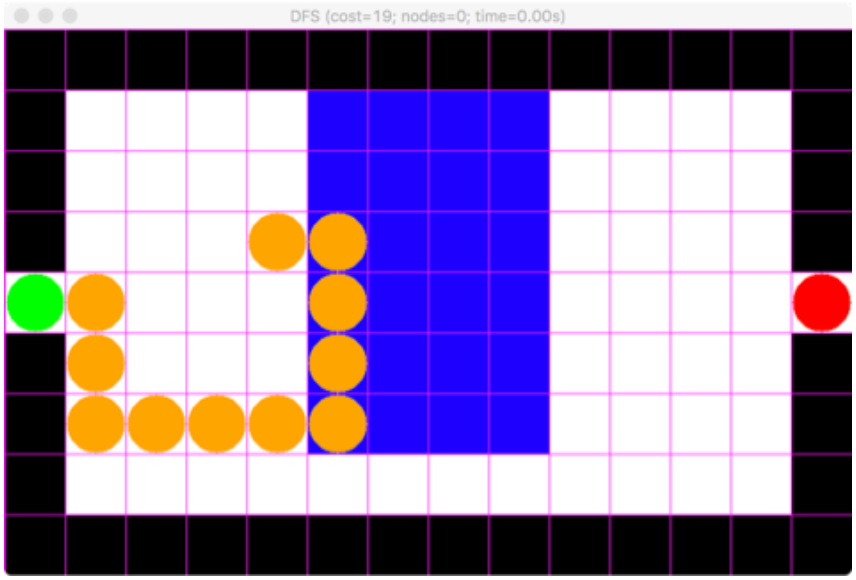
- Encourages familiarity with OpenCV data structures and functions + modular programming

```
(4, 0)
(4, 13)
KKKKKKKKKKKKK
KWWWWBBBBWWWK
KWWWWBBBBWWWK
KWWWWBBBBWWWK
WWWWWBBBBWWWW
KWWWWBBBBWWWK
KWWWWBBBBWWWK
KWWWWWWWWWWWK
KKKKKKKKKKKKK
```



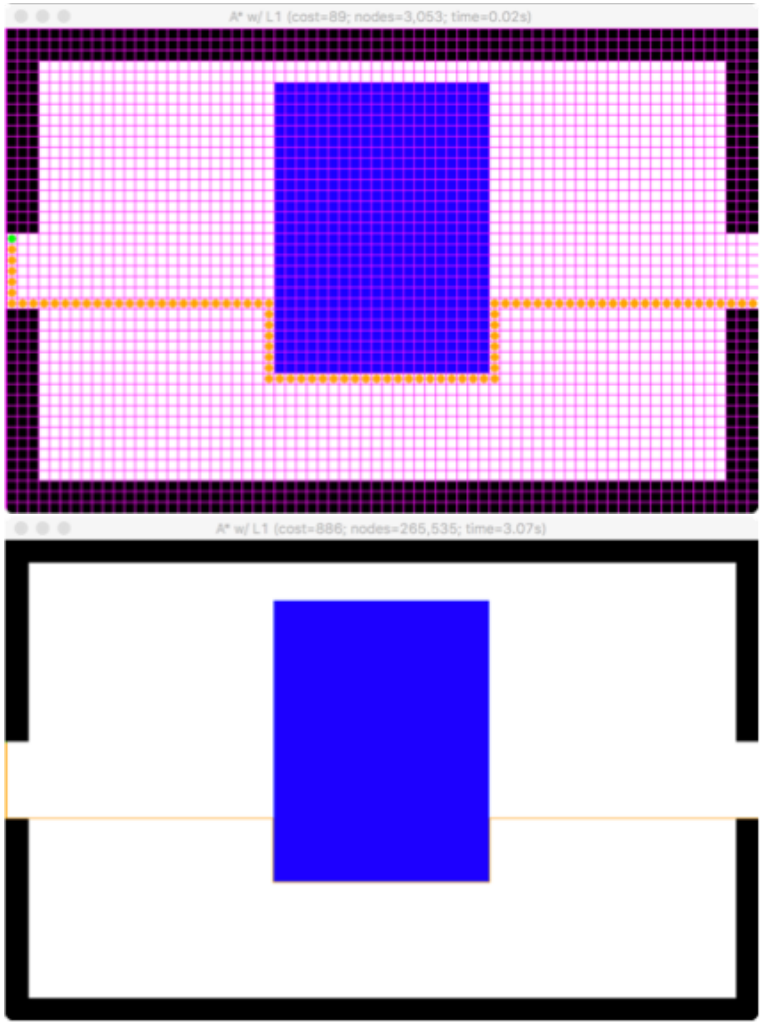
2. MazeProblem w/ Viz (~25 lines)

- Implement MazeProblem start, goalTest, successor (ala Pacman)
- Visualize path
  - Dummy solution provided to test



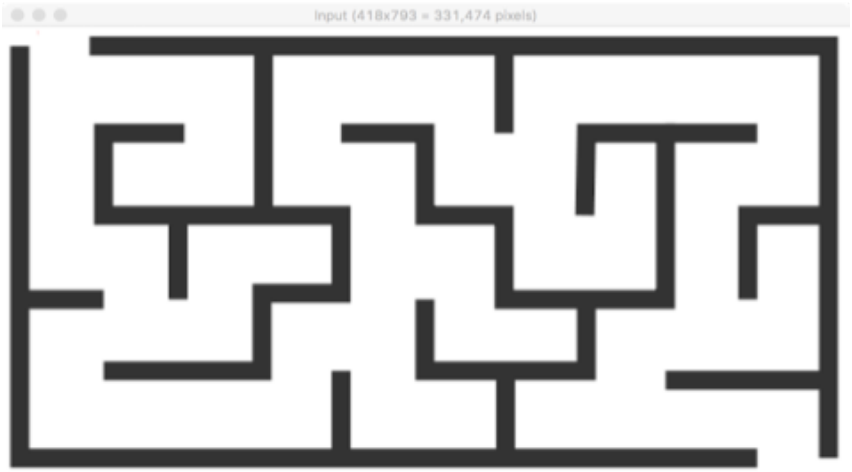
3. GraphSearch + Analysis (~15 lines)

- Encourages unified graphSearch
  - ala AI:MA
- Encourages data collection for a set of supplied mazes
  - Some are known to be identical with different representation



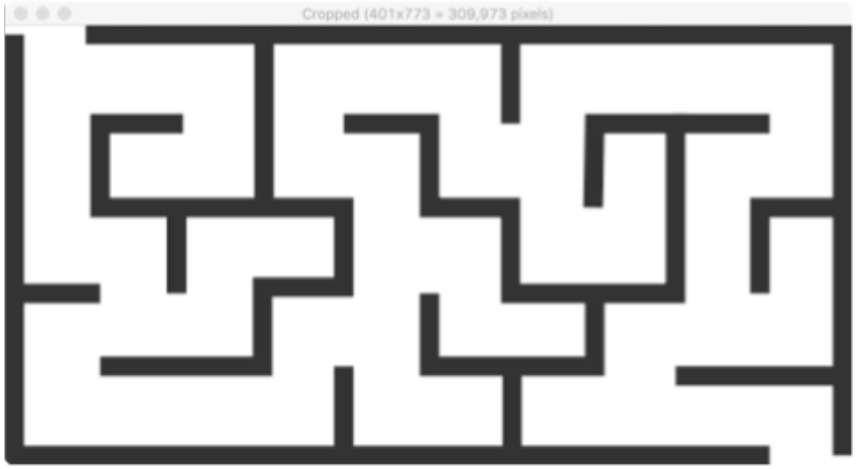
4a. Show Input Image (~5 lines)

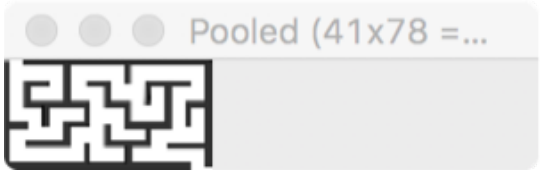
- Given PNG, present image in a window with size information
- Scaffolding provided, including show window and wait for key press



4b. Preprocess (~40 lines)

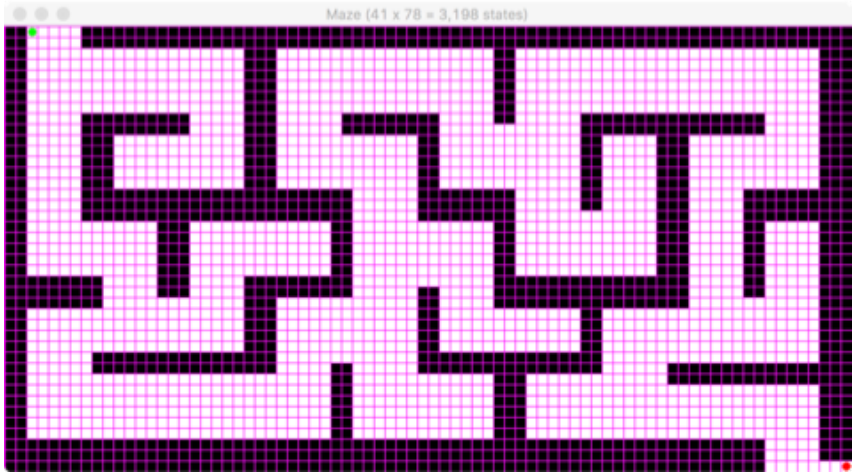
- Assume axis-aligned + black border
- Crop via Gaussian Blur + threshold
- Pool via user-supplied multiplier





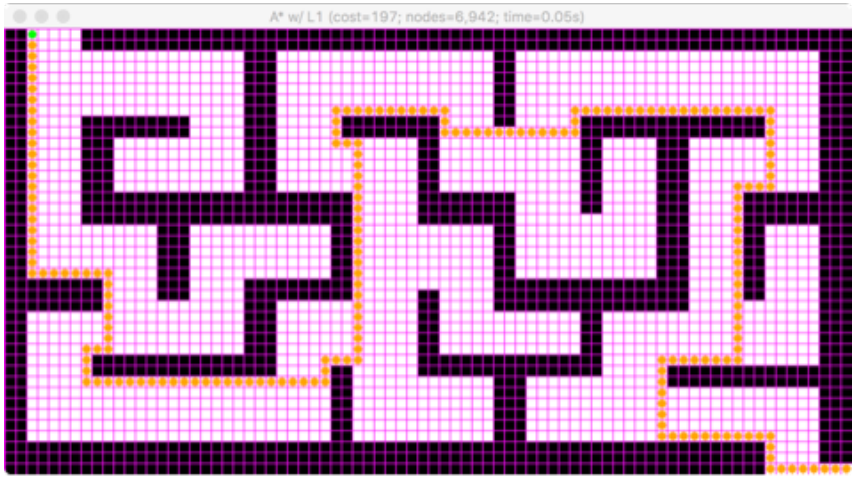
4c. Extract Maze (~30 lines)

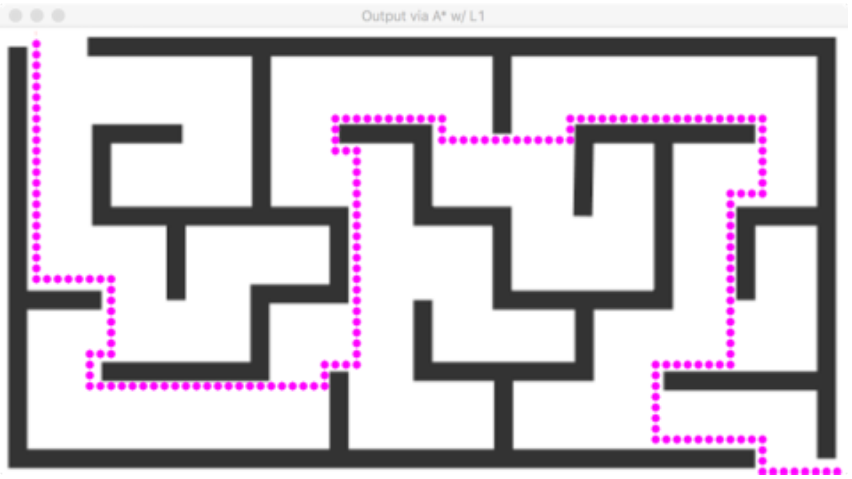
- Using distance to closest known color, extract row-of-rows
- Using simplifying assumptions, detect reasonable start/ finish locations
- Visualize!



4d. Solve! (~10 lines)

- Solve using #3
  - Loop through methods unless supplied a method
- Given cropping offset, overlay solution path on original image
- Analyze methods w.r.t. optimality, performance





Extra-Credit Exploration

- Different maze shapes
- Maze rotation
- Video

…

Strengths

- Integration with Pacman Projects
- Easy OpenCV setup across platforms
  - Anaconda (+ 1-3 lines)
- PowerPoint method for maze creation
- Representation vs performance analysis
- In the past, student teams have surprised me with their quality of work

Weaknesses

- No auto-grading
  - But emphasis on intermediate visualization
- This=Python3, Pacman=Python2
- Similar search problem to Pacman
- Limited image/vision aspects

Thank You :)

Questions?