Maze Solving

(Control theory – Determinism – Rule based AI
Graph theory – Limits of Computation)

Steve Wattam — Carl Ellis

Topics

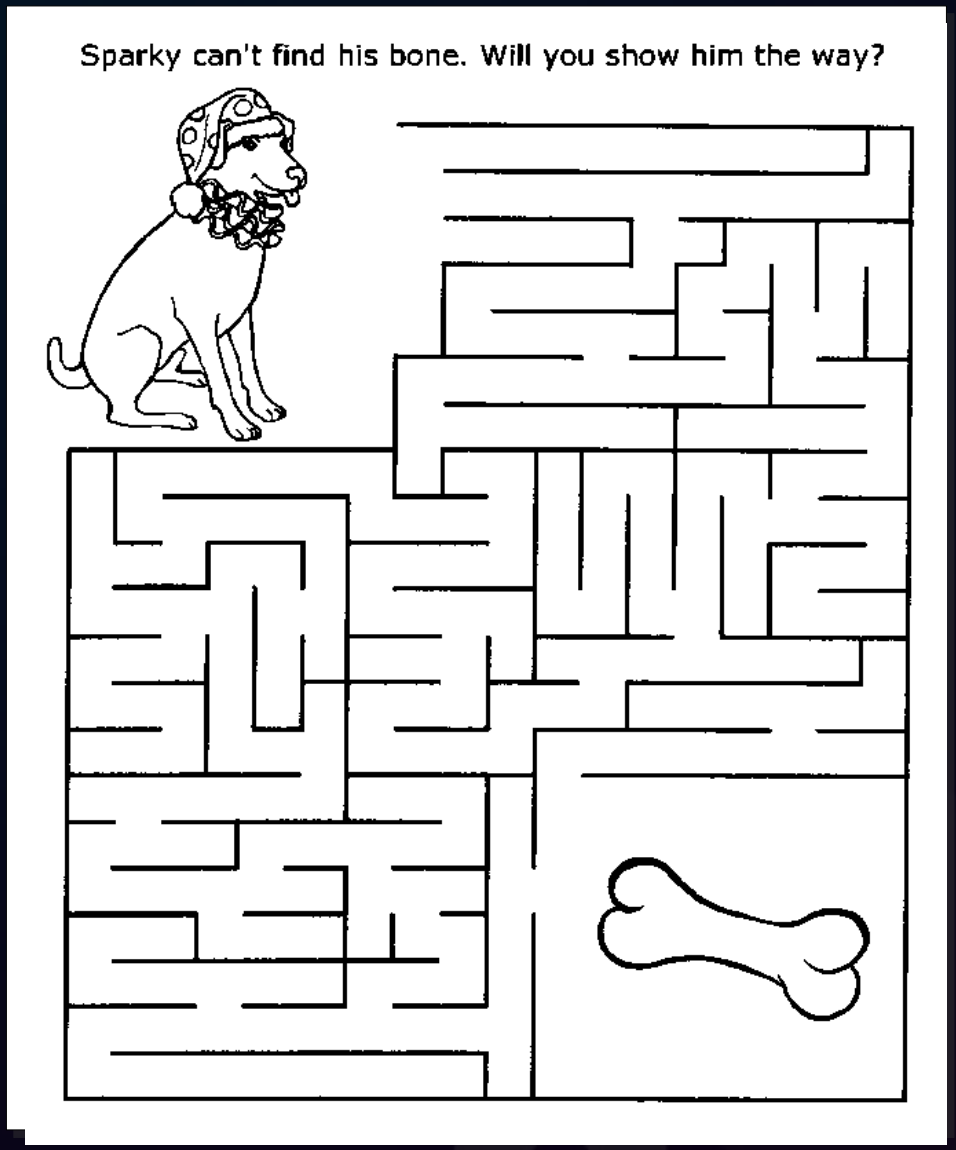
- Mazes – an overview
- Mice – an overview
- Control theory and AI
  - Stateful vs. Stateless (FSMs)
  - Limits
  - Determinism
- Our Framework – demo and introduction
- Suggested work (inc. info on maze types)
- Competition
- Reading/ further stuff

What is a Maze?

Space with Walls;

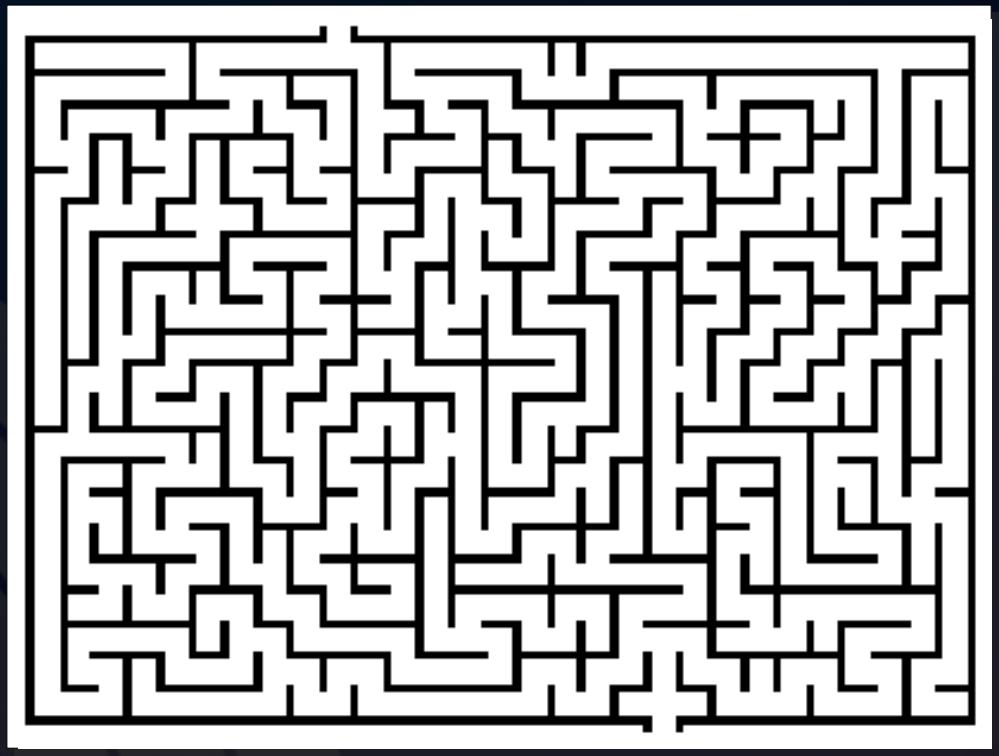
Start and Finish
(or Entrance/Exit);

Stops you finding
your bone.



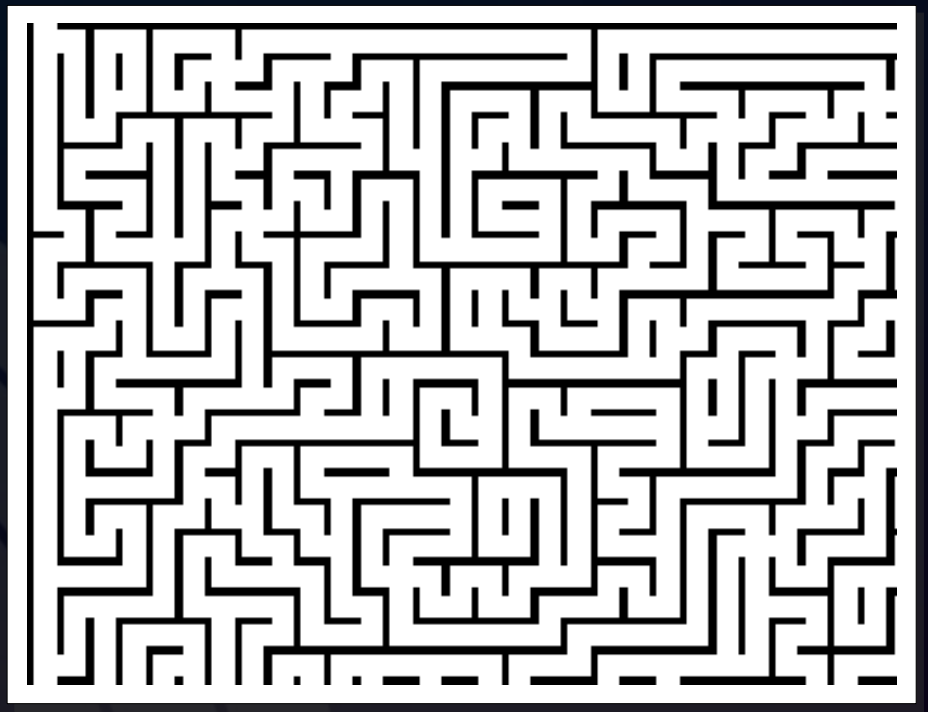
Types of Maze - Perfect

- This type of maze has no loops or inaccessible areas.
- There is a route to all areas of the maze.



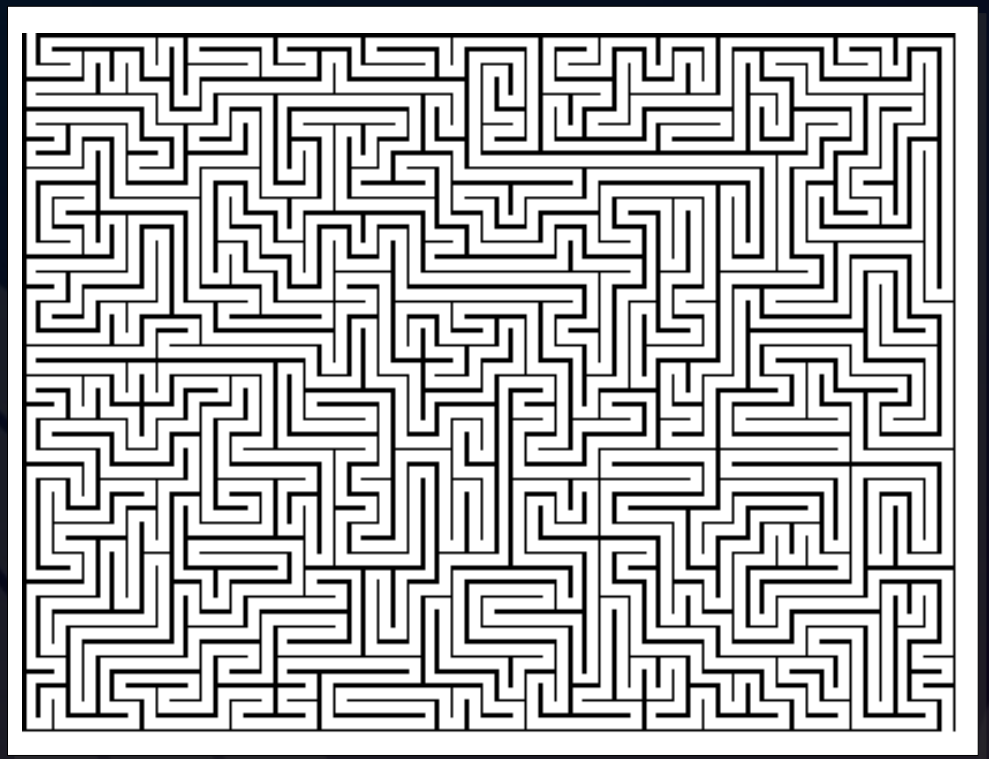
Types of Maze - Braid

- This type of maze has no dead ends.
- Uses passages that coil around and run back into each other.
- Much harder to solve (and generate).



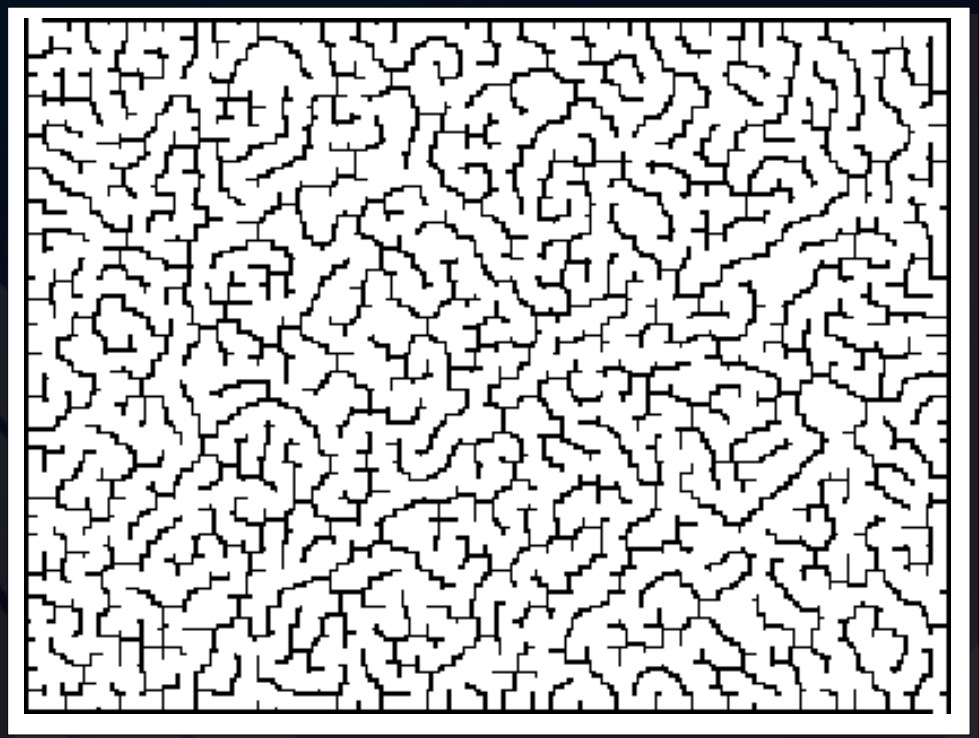
Types of Maze - Unicursal

- Has no junctions, sometimes called a Labyrinth.
- Trivial to solve with a bot, hard as a person as you get turned around.



Types of Maze - Sparseness

- Irregular layout of rooms: can be very heterogeneous
- Also classed as an irregular maze with wide passages (a bit like a relationship with your mother).



Properties of mazes

- Bias
  - A biased maze tends to have more passages in a horizontal or vertical direction. This makes it hard for a bot to travel 'against the grain'.
- Run
  - A large run is when the maze passages are longer than four rooms.
- Elitism
  - The elitism rating is the length of the optimum path in relation to maze length.

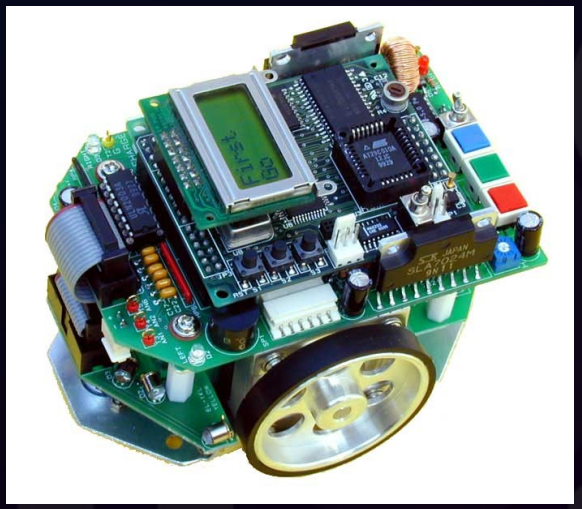
Mice

Maze-solving bots (micromice)

Micromouse competition

- 16x16 maze

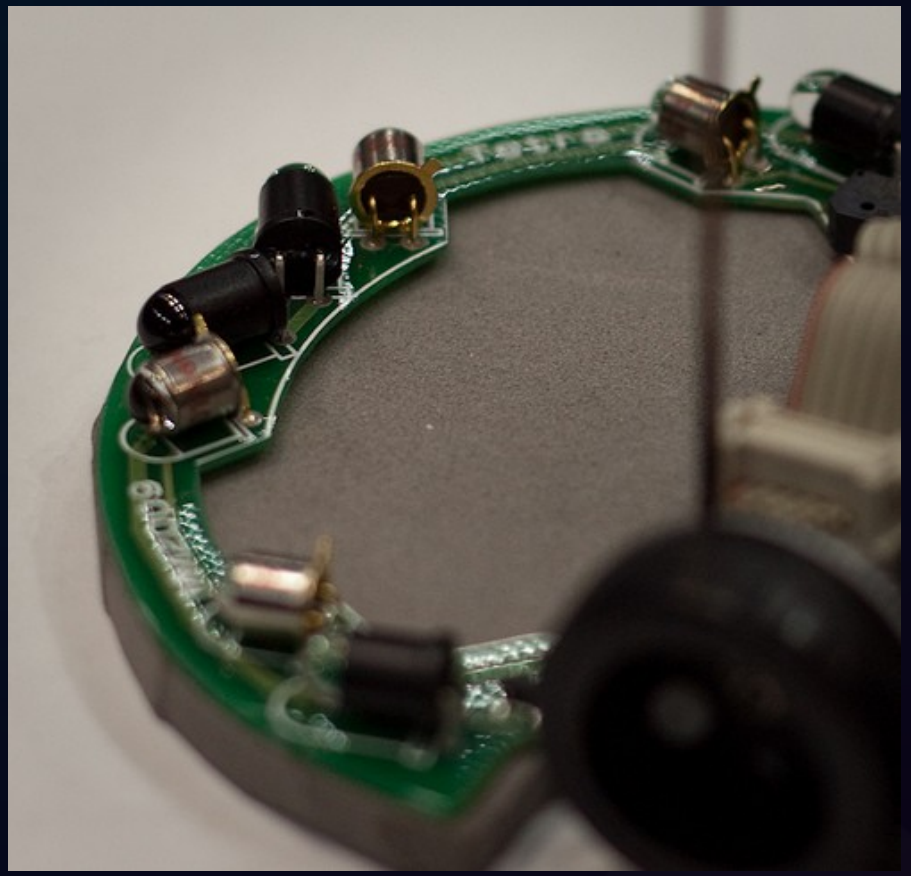
Limited view, processing power



No awareness of maze other than what is observed

Mice 2 – Abilities

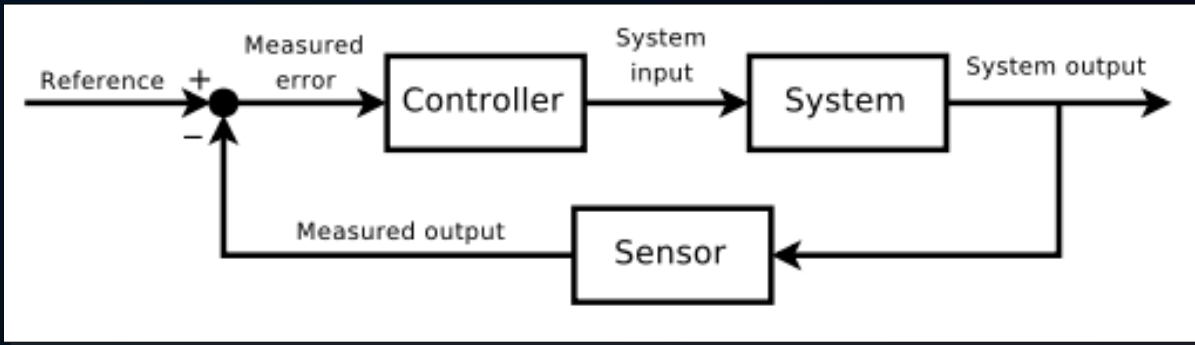
- Solving a maze algorithmically is not so hard – if you have the whole map
- MicroMice rely on 'practical' input, like us in hedge mazes.
- Click 'ere

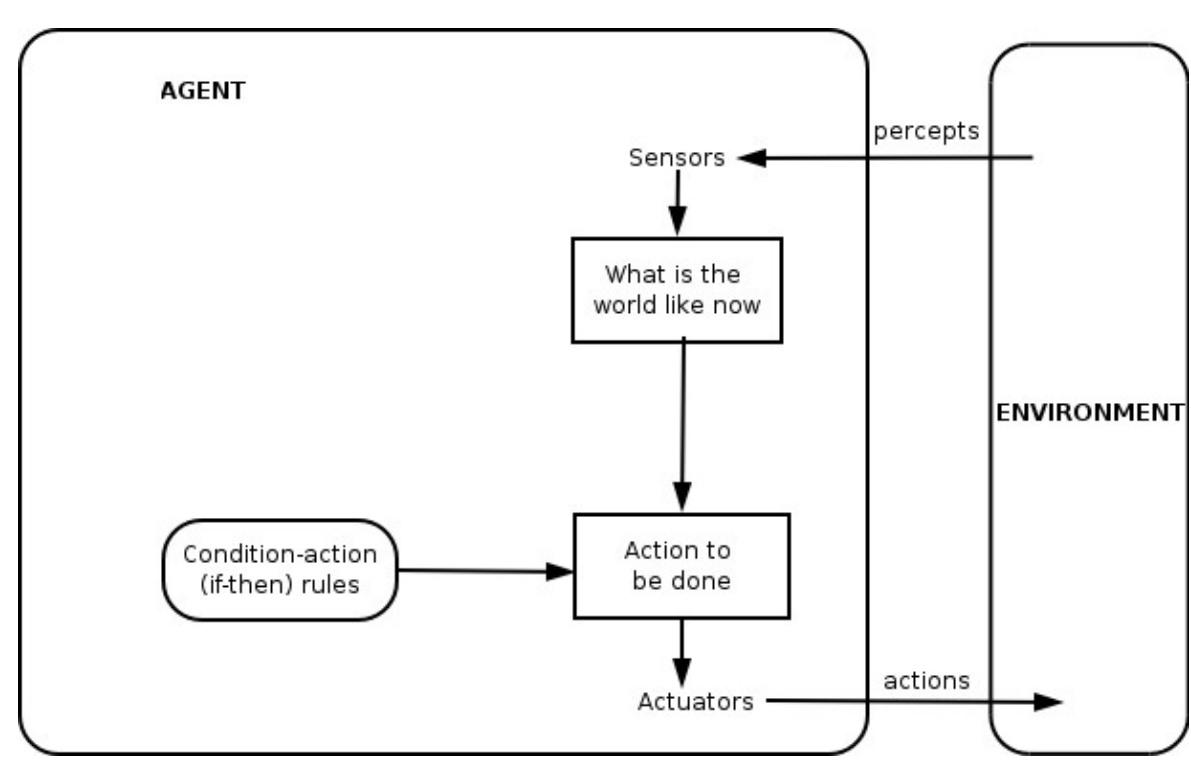


Sets of IR emitters and sensors on the front of a maze-solving robot.

Some theory

- Each mouse is an 'Intelligent Agent'
- Similar to control theory (but cooler)
- Measure – model – respond loop



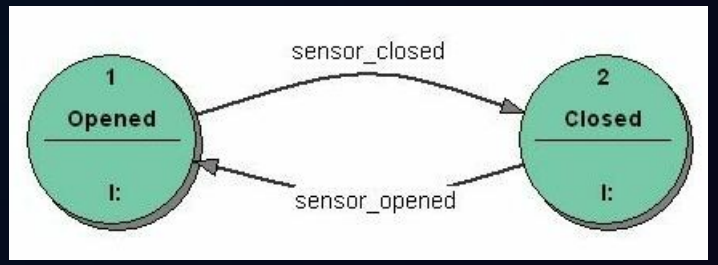


Types of Agent

- Stateful or Stateless
  - Does the bot remember information about its actions? (i.e. 'I am looking for the exit', 'I am backtracking')
- Deterministic or Stochastic
  - Will the bot do the same thing under the same conditions?
  - Determinism is reliable, but can be reliably shite
- Discrete or Continuous
  - Does the bot have to infer things like movement?
  - Can it react 'smoothly'?

Keeping State – Finite State Machines

- Model state based on events
  - Making a coherent view of how the world is, rather than how it has changed



- Can be interrogated at any point, even if there is no sensor data

- Make state changes based on arbitrary rules
  - Including probabilities and 'fuzzy' rules

Determinism

- Will this bot always perform the same actions if in the same situation
- Determinism results in reliable bots
  - But if the rules are flawed then they can be poor
- Non-determinism is closer to human judgement
  - But can be very hard to get 'right'

Discrete and Continuous

- All computerised agents are discrete
- Some may be able to run their actuators continuously (i.e. open valve until flow balances)
- This can offer better control, better feedback
  - 'Unsure' bots can creep slowly and check more
  - Confident ones can rush ahead blithely

Demos!

- Deterministic vs. Stochastic
  - DaveBot is stochastic, has no rules
  - LeftBot is deterministic, has only rules
- Discrete vs. Continuous response
  - LeftBot is discrete, moves and re-assesses
  - AdvancedBot is faux-continuous, may queue up
- Stateful vs Stateless
  - AdvancedBot is stateful, in order to do more complex actions
  - LeftBot is not

Limits to bot ability

- The best solution is to simply take the right path every time, but:

The maze is a big tree of choices

- Without enumerating every choice, the best option is thus equivalent to a binary tree search algorithm.

- O(log(n)) complexity for most *good* searches

Our Framework

- Bots get only a small amount of sensor data
  - Their immediate surroundings
  - Bot co-ordinates, orientation
  - Finish point co-ordinates

  (Those in yellow are not necessary except for advanced techniques)

- Bots can only move in limited ways
  - Left, Right, Forward, Back only (but it's enough!)

Our Framework 2 – Bot Classes

- Bot
  - Interface, simplest API available
  - Makes decisions every time it's called
- AdvancedBot
  - May queue up sequences of moves
    - (But they are only done once per tick)
  - Makes decisions each time it runs out of queue items

Demo of Bot Code

This should be a quick intro to the bots we've made

http://extremetomato.com/projects/maize/

Challenges

- Beat DaveBot on any maze type
  - An EmptyMaze may be easiest, a FullDFSMaze may be hardest
- Solve a DFS Maze without using a wall follower
  - Requires modelling the choices
- Write a wall follower that can also solve empty, scatter mazes.
- Write a wall follower that, through heuristics, can beat left/right following bots.

Getting Started

- Decide which maze(s) to run on
  - Not all bots will beat all mazes (probably)
- Decide which technique to use
  - Or to improve one of our bots
- Code, or pair up with someone who wishes to
- Submit to us for a glorious battle of awesome.

References

- http://en.wikipedia.org/wiki/Agent-based_model
- http://en.wikipedia.org/wiki/Intelligent_agent
- http://micromouse.cs.rhul.ac.uk/