

Outline	Overview ○○○○○	Mice ○○	Control Theory & Agent-based AI ○○○○○○○○	Introduction to Maize ○○○○	Challenges
---------	-------------------	------------	---	-------------------------------	------------

Mazes: An Introduction

Stephen Wattam

Computing Society of Lancaster University

November 1st, 2012

Notes

Outline	Overview ○○○○○	Mice ○○	Control Theory & Agent-based AI ○○○○○○○○	Introduction to Maize ○○○○	Challenges
---------	-------------------	------------	---	-------------------------------	------------

Overview

- What is a Maze?
- Types of Maze
- Properties of Mazes

Mice

- What's one of them?
- Abilities & Limitations

Control Theory & Agent-based AI

- Overview, Sensing and Actuating
- Types of Agent
- Limits to Computability

Introduction to Maize

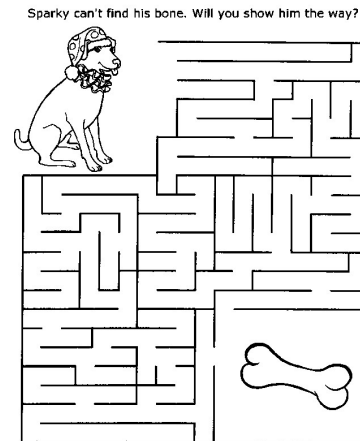
- The Framework
- Code Introduction

Challenges

Notes

WHAT IS A MAZE?

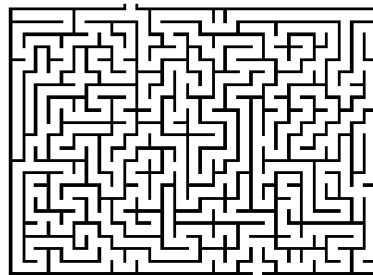
- ▶ A space with walls
- ▶ Has a start and finish (or entrance/exit)
- ▶ Stops you from finding your bone
- ▶ **Has some feature making it hard to follow the path**



Notes

PERFECT

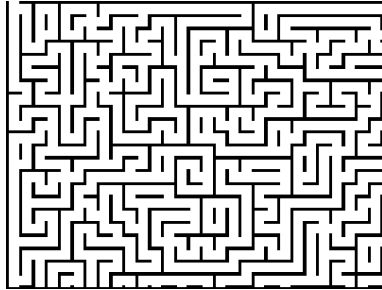
- ▶ Has no loops
- ▶ Has a route to every point in the maze



Notes

BRAID

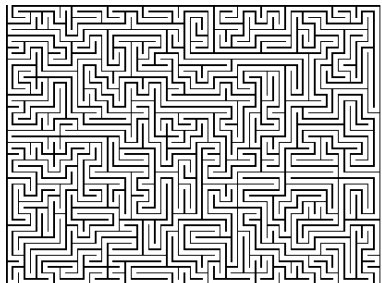
- ▶ Has no dead ends, instead using passages that run around and join up again
- ▶ Much harder to solve (and generate!)



Notes

UNICURSAL

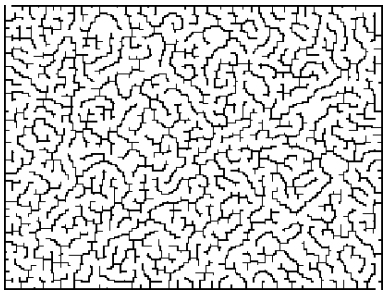
- ▶ Has no junctions, so is a single path
- ▶ Sometimes called a labyrinth
- ▶ Trivial to solve with automated means



Notes

SPARSE

- ▶ Wide, irregularly-spaced passages.
- ▶ Often come in cool designs (not shown here).



Notes

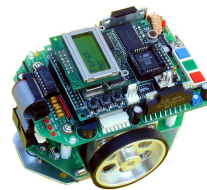
PROPERTIES OF MAZES

- ▶ **Bias**—The tendency of a maze to have horizontal or vertical passages. Makes it hard to go ‘against the grain’
- ▶ **Run**—A long, straight, uninterrupted passage
- ▶ **Elitism**—The ratio of the maze’s shortest path against its total path length

Notes

MICE

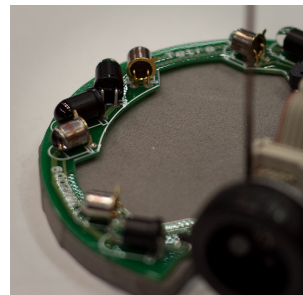
- ▶ Mice are **Maze-solving robots**
Mice are also mice
- ▶ Compete in many competitions, solving real-life mazes (Click me!)
- ▶ Limited view of the maze, limited processing power
- ▶ No awareness of the maze other than what is learned



Notes

ABILITIES & LIMITATIONS

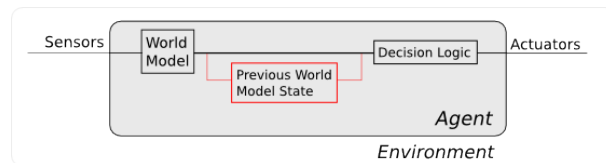
- ▶ Solving a maze is easy, if you can see it all (Click me!)
- ▶ Mice rely only on practical real-world input from sensors
- ▶ So how do they do it?



Notes

MEASURE–MODEL–RESPOND

- ▶ Similar to control theory—each mouse is an ‘intelligent agent’, judging the state of the world from its inputs
- ▶ Three basic stages, repeated *ad infinitum*:
 1. **Measure** the world about you;
 2. **Model** the rest of it to create a virtual world;
 3. **Respond** by using information from the model.



Notes

MEASURE–MODEL–RESPOND PROBLEMS

- ▶ Sensors are often inaccurate, or tell us confusing, simple things (data is not information)
- ▶ The world changes, even when we’re not the ones changing it
- ▶ **Modelling even simple worlds is difficult!**
- ▶ **We have to know what to do, even if our model is incomplete**

The items in red are all you have to worry about in Maize.

Notes

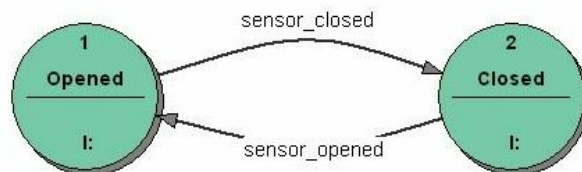
MANAGING THE WORLD

- ▶ **Stateful/Stateless**—Do we want to remember past events, or treat each sensor reading as a new event?
- ▶ **Deterministic/Stochastic**—Do we want to respond in a purely predictable manner?
- ▶ **Discrete/Continuous**—Are all of our values smoothly variable? Can we react partially/smoothly?

Notes

INFERRING STATE—FSMs

- ▶ Display how the world **is**, based on how it **has changed**
- ▶ Can be interrogated without needing new sensor data
- ▶ Make transitions between states based on arbitrary, even probabilistic, rules
- ▶ Underpin all of computing!



Notes

DETERMINISM

- ▶ Given the same (sequence of) input, will **always** produce the same output
- ▶ Very reliable
but also reliably bad if the rules are poor
- ▶ Nondeterminism can involve fuzzy logic, probability, or guesswork
Closer to human judgement, but very hard to get 'just right'

Notes

DISCRETE & CONTINUOUS CONTROL

- ▶ Continuous control allows for small tweaks and improvements
- ▶ Can be hard to rationalise logically—needs threshold values or statistics
- ▶ Can offer better control and progressive feedback
Confident bots can rush quickly; Unsure ones can creep to gather more information

Notes

DEMOS

- ▶ Deterministic vs. Stochastic
 - ▶ DaveBot is stochastic, has no rules
 - ▶ LeftBot is deterministic, has only rules
- ▶ Discrete vs. Continuous
 - ▶ LeftBot is discrete, one move per assessment
 - ▶ AdvancedBot has some continuous properties: command queueing
- ▶ Stateful vs. Stateless
 - ▶ LeftStateBot is stateful, remembers a long sequence of actions
 - ▶ LeftBot is not

Notes

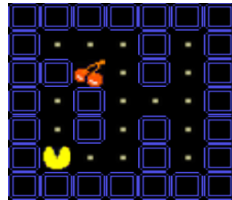
COMPLEXITY OF MAZE SOLVING

- ▶ Algorithmically, the best solution to a maze is to choose the best right path each time:
The maze is just a big tree of choices
- ▶ The best case is thus equivalent to a binary search, plus the time needed to move
- ▶ This becomes $O(n \log_2 n)$, where n is the number of choices (plus a bit of time to travel from start to finish).

Notes

OVERVIEW

- Manages Bots and Mazes, and simulates a virtual world
- Tile-based environment
- A tile can be a start, finish, space, or wall.



Notes

THE LIFE OF A BOT

- Bots get restricted data:
 - Their immediate surroundings (3x3 array around them)
 - Bot co-ordinates in the maze
 - Bot orientation
 - Finish point co-ordinates

(Technically, all that's needed is the red one, but the other things make it more fun)

- Bots can move only one tile per 'tick', and only Left, Right, Forward and Back (like kings in chess)

Notes

BOT CLASSES

► Bot

- Simplest API available
- Maize calls `nextMove()` on every tick.
- Simply return a direction to move.

► AdvancedBot

- Like `Bot`, but only requests a move when it has emptied its queue
- Has a queue commands can be added to, and a more powerful high-level API (`moveLeft()`, `turnRight()` etc)

Notes

DEMO, CODE RUNDOWN

A quick overview of the bots we have

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

Notes

CHALLENGES

1. Beat DaveBot on any Maze
(An EmptyMaze might be simplest, a ScatterMaze might be hardest)
2. Solve a DFS Maze without using a wall following algorithm
(this requires building a model of the decision tree)
3. Write a wall follower that can also solve Empty and Scatter mazes.
4. Improve an existing wall follower with heuristics

Notes

GETTING STARTED

- Decide on a maze type to run on
- Decide on a strategy to try
- Code, or pair up with someone who wishes to, and hack at it
- Submit to stephenwattam@gmail.com for a glorious battle, with prizes if I can be arsed.

Notes

Outline	Overview ○○○○○○	Mice ○○	Control Theory & Agent-based AI ○○○○○○○○	Introduction to Maize ○○○○	Challenges
---------	--------------------	------------	---	-------------------------------	------------

FIN

All resources available from

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

Notes

Notes
