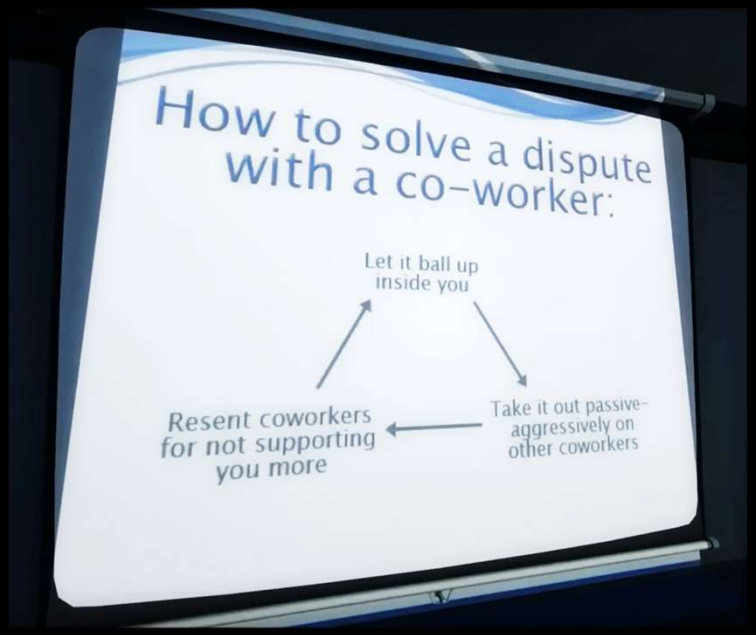
AI FOR GAMES

CSC404 Lecture Slides

But first, an announcement

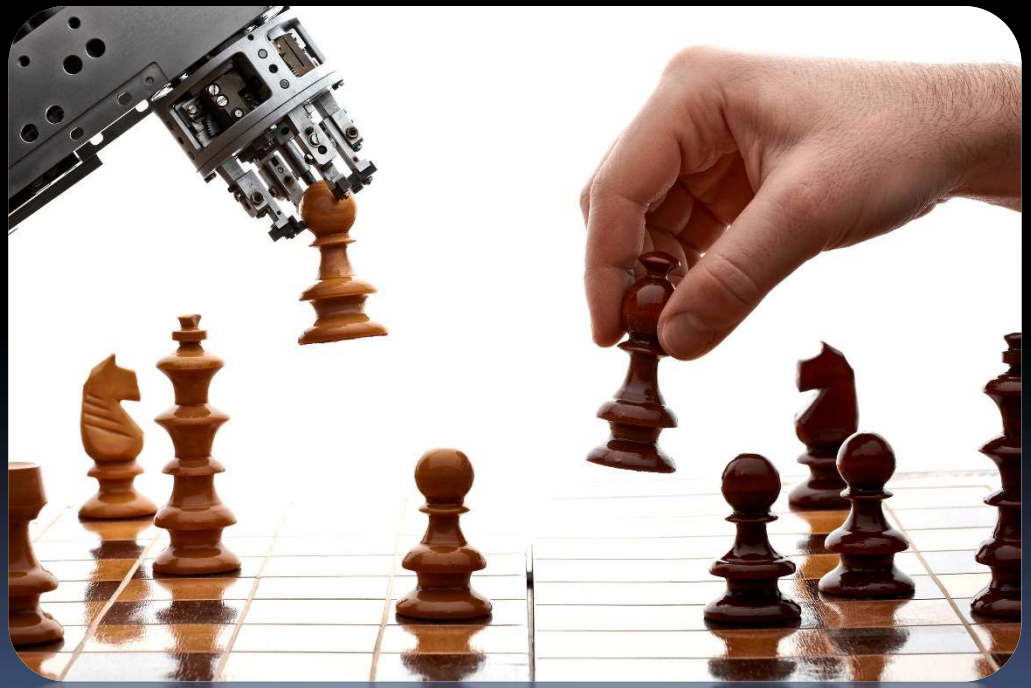
- The project is 80% of your course mark.
  - A large factor on how you do depends on your peer evaluations.
- Have you been a good group member so far?
  - Have you listened to the ideas of others?
  - Have you done your share of the work? More?
  - Have you checked with others, to be sure?



Artificial Intelligence

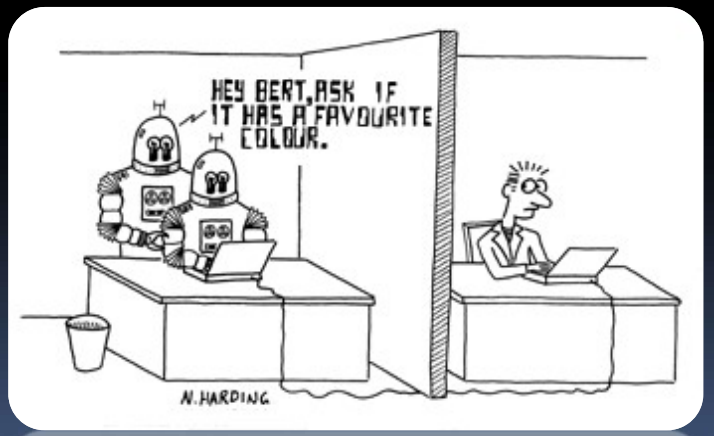
What is AI?

- Artificial intelligence (AI) is the field of creating intelligent behaviour in machines.



Okay, so what is intelligence?

- Typically, a combination of perception, processing and expression.
- Most often measured using the Turing Test.
  - Main principle: Can you tell if you're interacting with a human or a computer?
  - Alternate: Can computers complete tasks with levels of performance?




BostonDynamics
Boston Dynamics

What is AI for games?

- Game AI isn't concerned with the same tasks and objectives as mainstream AI.
- The goal of game AI is to create a good experience for the player, by having the game respond properly to a situation.
- Commonly referred to as gameplay.

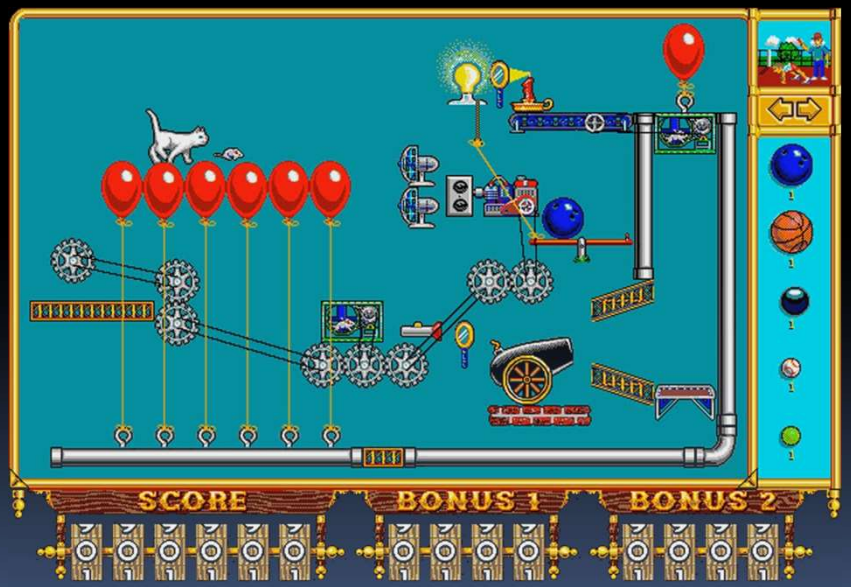
AI for Games

- Typical assumption is that game AI deals with the behavior of NPCs (non-player characters):
  - Movement
  - Strategies
  - Player interaction
- Best to start with how AI modifies our existing model of games (and past 404 topics):
  - Object & environmental AI
  - User interface AI

AI + Level Design

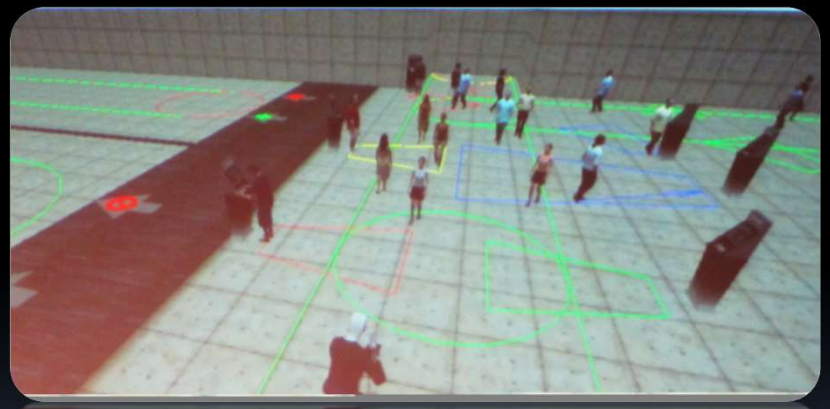
Intelligent Objects

- At the most basic level, AI == UX.
- Objects need to "feel" right.
  - This can change in response to situation and player behaviour.
  - Example:
    - Objects that fall nicely into place when they need to be stacked and/or climbed.



Intelligent Objects

- Some objects have intelligence and behaviour of their own.
- Example: Action stations
  - e.g. benches, ATMs.
  - Stations "capture" characters in given area, taking over their brains & animations.
    - Once done, release character.
    - For NPCs, resume normal operation (scripts).



AI Example: Assassin's Creed

Intelligent NPCs

- Intelligence needs to extend to both things and people.

What this means for you

1. Objects in your level can be responsible for activating and executing scripted behaviours.
2. Some of you should stop using Unity physics.

Intelligent Terrain

- Obstacle avoidance
  - Case-sensitive steering behaviour.
  - Social rules, self-organizing lanes.
- Flow
  - Dynamic splines, dynamic lane forming.
  - Problems: twitching, piling up.
- Result: creates more nuanced characters.

Procedural Generation

- Very fashionable now to use procedural level generation.
- But remember!

Procedural ≠ Random

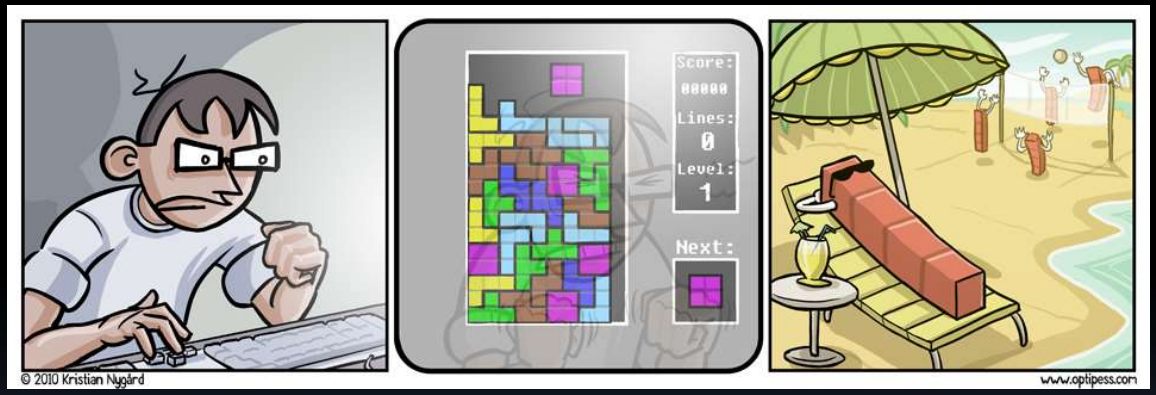
Procedural vs Random

- Example: Binding of Isaac
  - Levels are different every time:

  - But…each level also follows certain rules:
    - 4 core rooms (treasure, boss, shop, secret)
    - 5-20 non-core rooms, drawn from a preset pool.

What this means for you

- In general, avoid truly random behaviour.
  - Randomness creates situations that make the player feel like the computer is cheating.

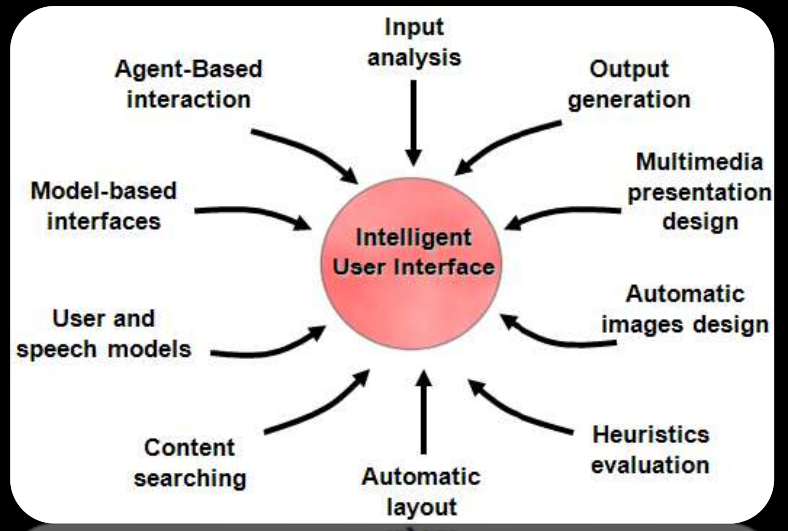


- Make sure any random elements are controlled by a set of purposeful rules.

AI + User Interfaces

Intelligent interfaces

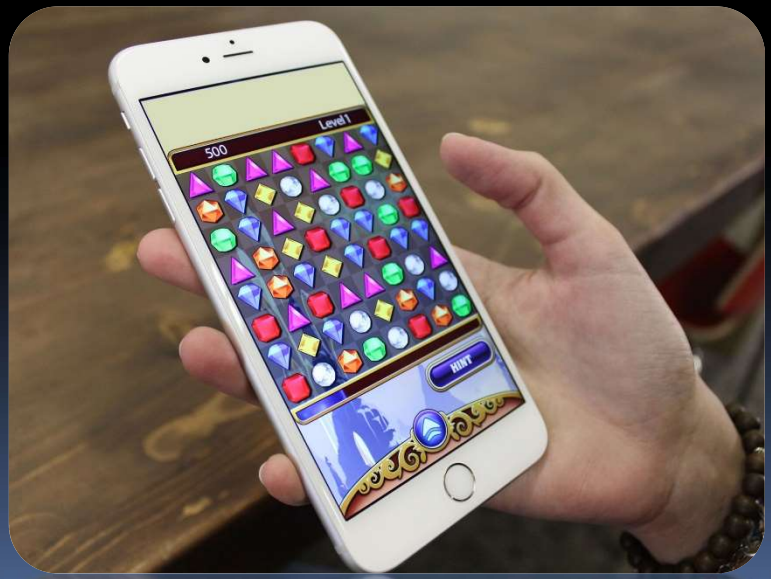
- Designing a good UI is hard, and the focus of other courses in CS.
- The main challenge of implementing game UI is intentionality.



- Need to look beyond what the player does (i.e. run, jump, shoot), and determine what they player means to do.

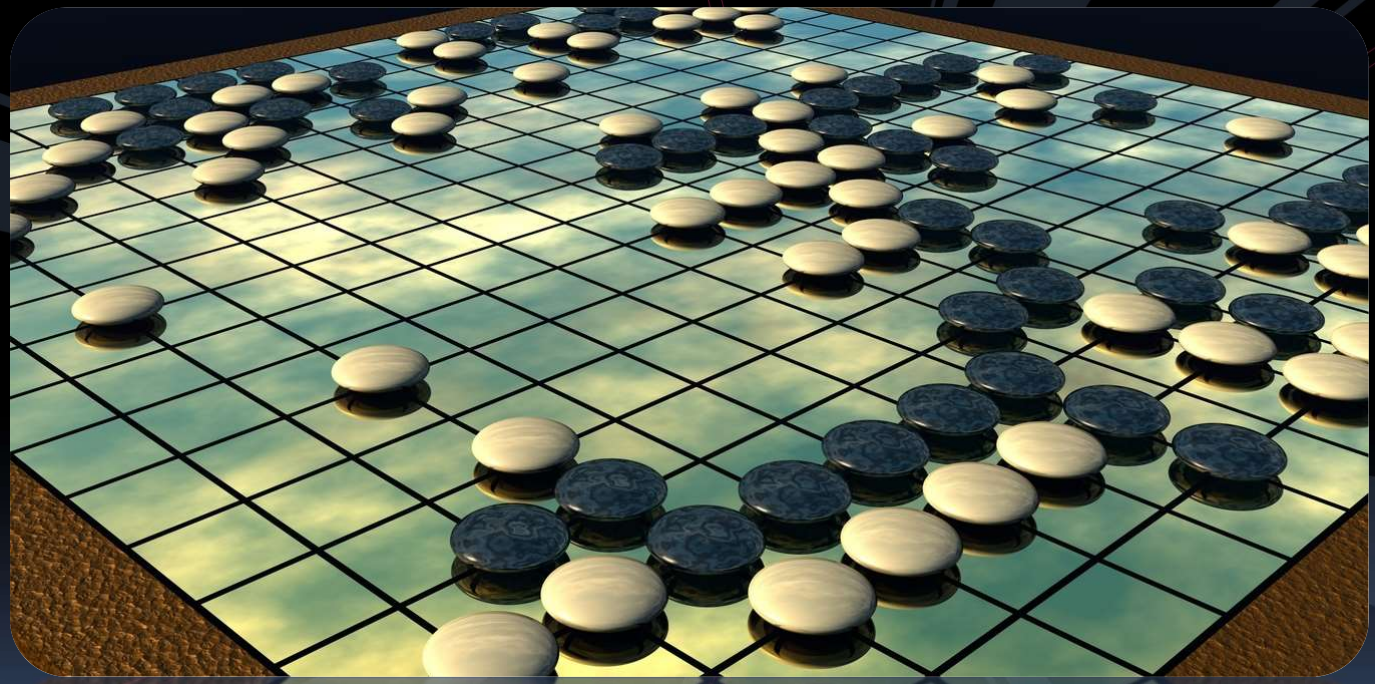
Case study: Bejeweled

- When a player selects a point on the screen, what should the game do in response?
- Developers at PopCap Studios look at the surrounding region to find the best move, and assume that was the player's intention.
- Games need to infer what the player wants to do, and allow them to do it.




9:45 AM / CLEAR
752
CABARET
Return to Dandelion.
Gallop [Press Twice]
Canter [Hold]
Dismount [Hold]
HORSE STAMINA

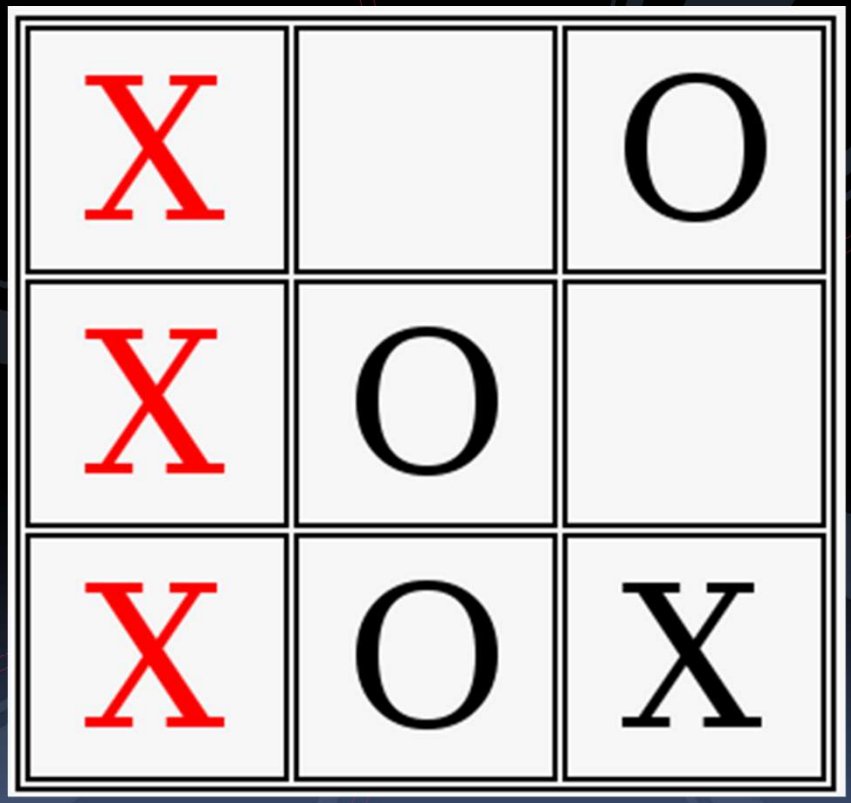
Games + Mainstream AI



The AI Spectrum

- Search/Pathfinding
- Planning
- Expert Systems / Decision Trees
- Reasoning
- Speech Recognition
- Natural Language
- Computer Vision
- Knowledge Representation
- Machine Learning
- Neural Networks
- Data Mining
- Social intelligence
- Robotics
- Multi-agent systems

Classic Game AI

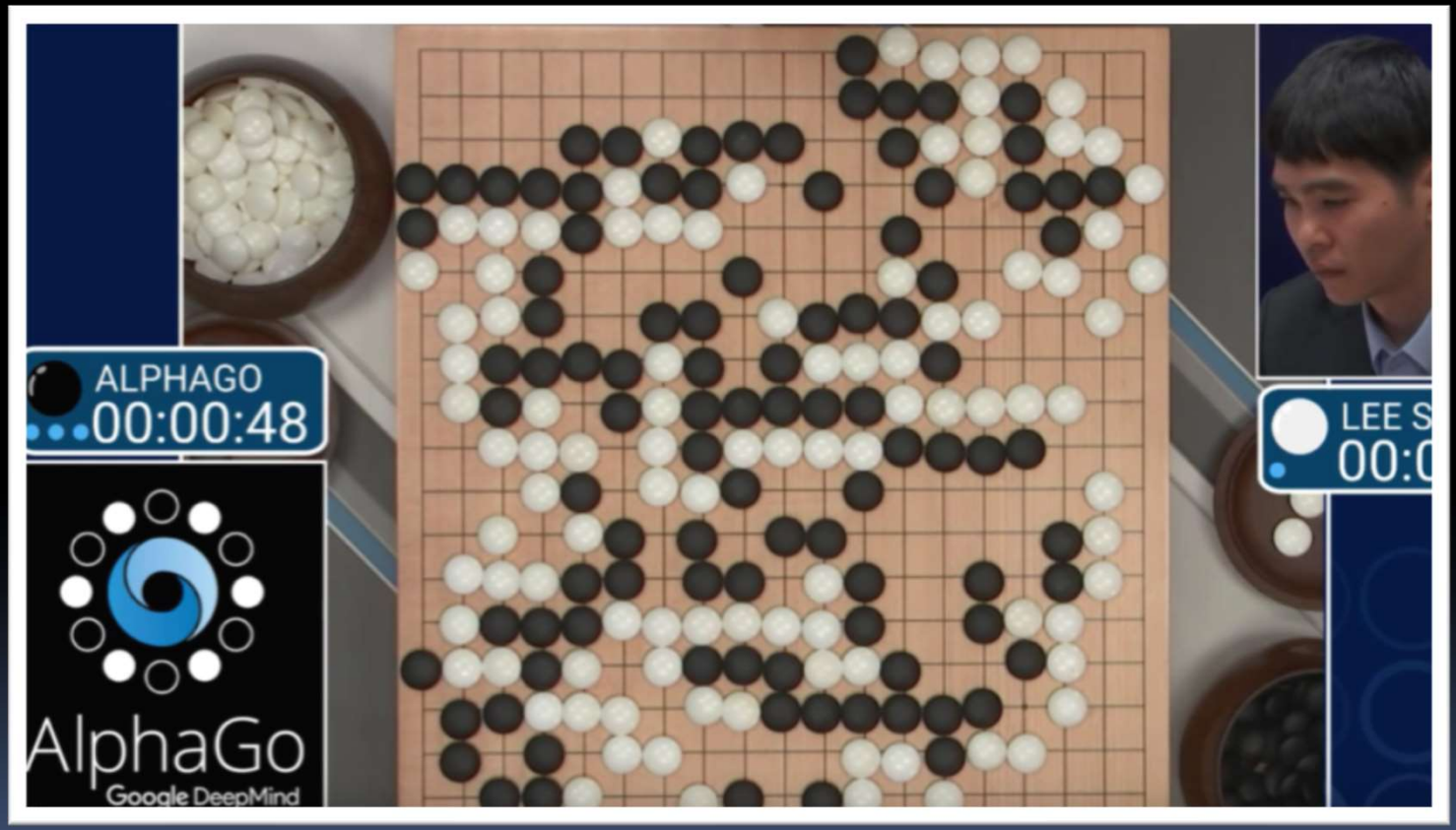


Measuring Intelligence

- Historically, games were used as tests of computer intelligence, because only smart humans performed well at certain games.

1997: Deep Blue

2017: AlphaGo



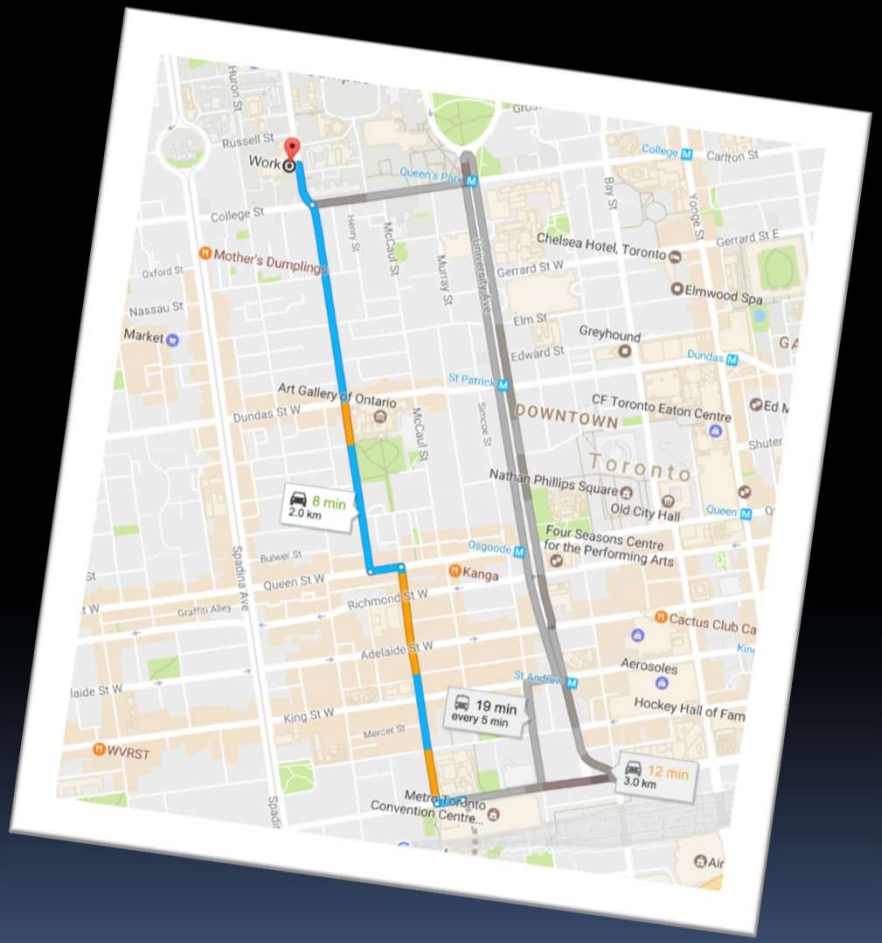
Thinking about AI

- The simplest AI problems don't even seem like AI problems.
  - Example: Sudoku solver.
- Tasks that seem more like algorithms than AI.
- At its heart, game AI is more about algorithms than anything magical or intelligent.

7		6		5		3		8
		3				2		
	1			9			4	
6			3		1			4
		1				5		
3			4		5			9
	6			8			1	
		7				9		
2		4		1		8		6

Example: Google Maps

- Searching is still considered a big problem in game AI.
- Touches on a lot of gameplay tasks:
  - Navigation
  - Decision making
  - Exploration



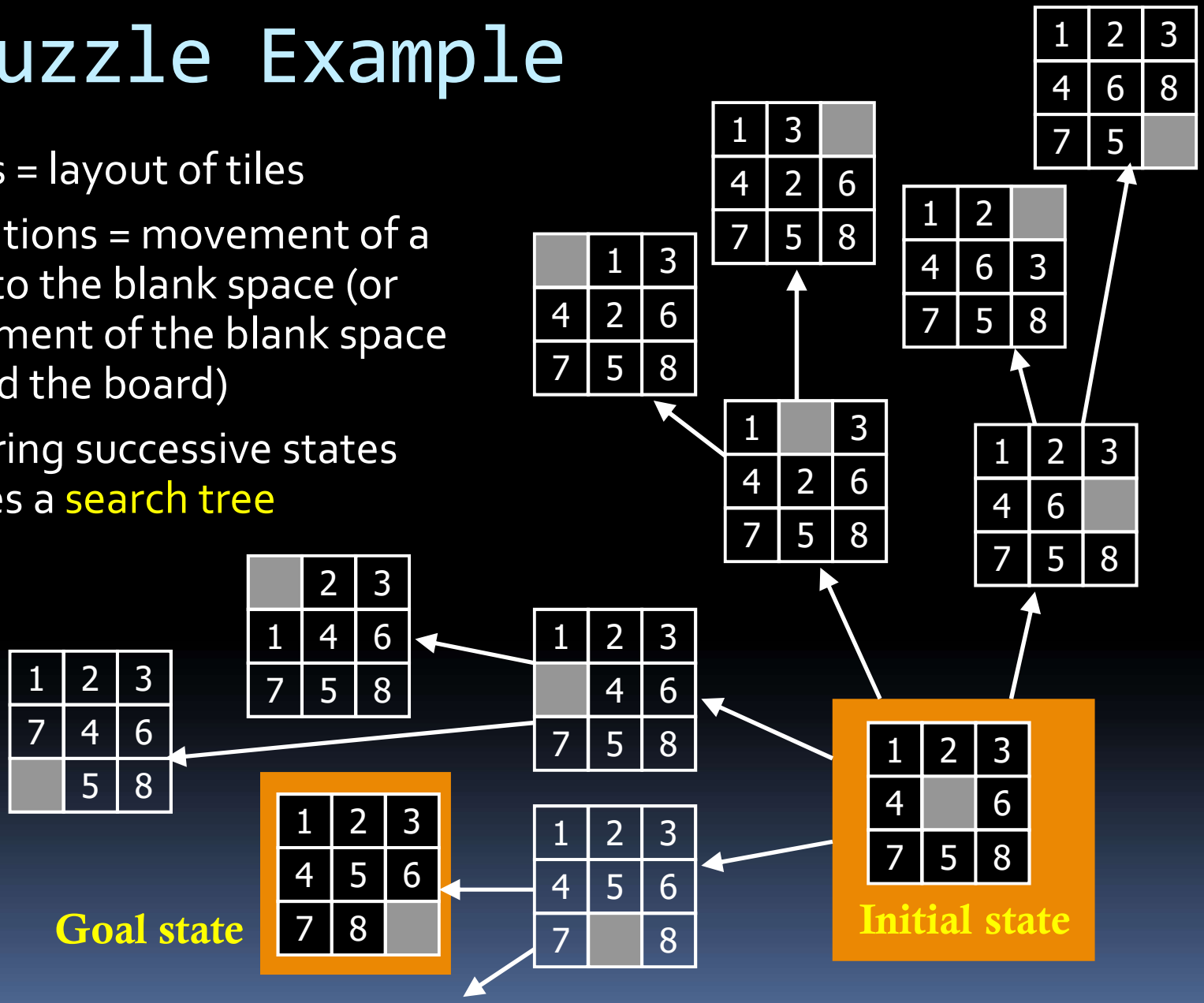
Example #2: Monument Valley

Searching / Pathfinding

- Searching is the act of exploring possible states in a game or environment, to reach a specified goal
  - State in a game of chess = a layout of pieces on a board
  - State in an adventure game = your current position, where you're facing, what you're carrying and what you've done
  - State in a sports game = your current score, position, direction and player condition
- The actions that allow you to move from one state to another is called an operation
  - Operations in chess = moving a piece
  - Operations in an adventure game = moving your character, picking up an item, using an item, changing your clothes
  - Operations in sports game = move, throw, hit, jump, etc.

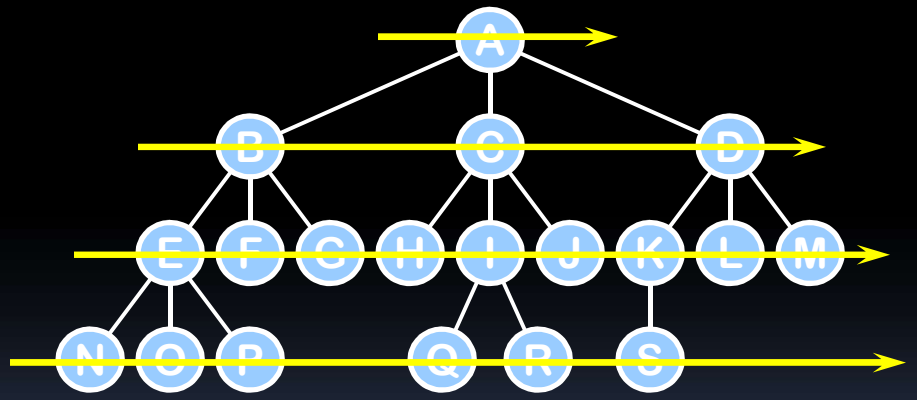
8-Puzzle Example

- States = layout of tiles
- Operations = movement of a tile into the blank space (or movement of the blank space around the board)
- Exploring successive states creates a search tree



Breadth-first Search

- Breadth-first search expands the start state first, and then expands successive states, one level at a time.
- Slow, but guaranteed to find the closest solution (if one exists)
- Complexity analysis:
  - Time: ~ b^d
  - Space: ~ b^d
  - (b is the branching factor, the maximum number of successor states possible. d is the depth of the solution in the search tree)

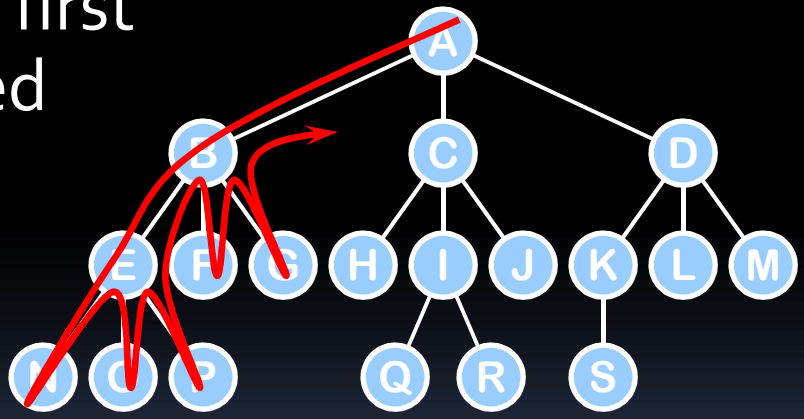


Expanded nodes:

A B C D E F G H I J K L M N O P Q R S

Depth-first Search

- Depth-first search expands states down a single branch of the search tree until the goal or a dead end is reached (requires backtracking)
- Faster, but dangerous. Can explore far past solution depth, and the first solution isn't guaranteed to be the best possible.
- Complexity analysis:
  - Time: ~ b^d
  - Space: ~ b*d
  - (d is the maximum search depth)



Expanded nodes:

A B E N O P F G C H I Q R J D K S L M

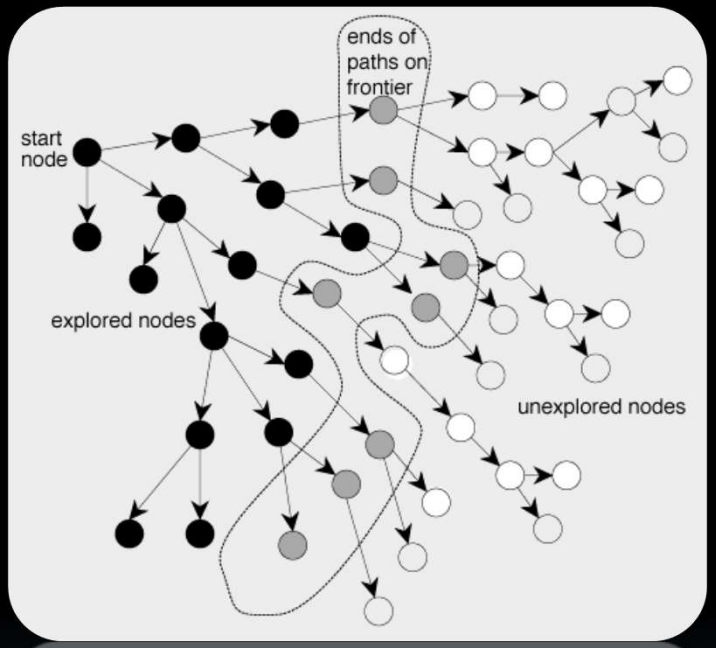
Heuristic Searches

- Also known as informed searches
- A heuristic is a "rule of thumb" that allows an agent to estimate the distance between a particular state and the goal
- Heuristic function = estimated cost of path from current state `n` to goal state → `h(n)`
  - e.g. straight-line heuristic: direct distance between current position and destination on a map
- Heuristic searches ignore path cost information, and focus instead on seeking the solution

Heuristic Example: Mercenaries

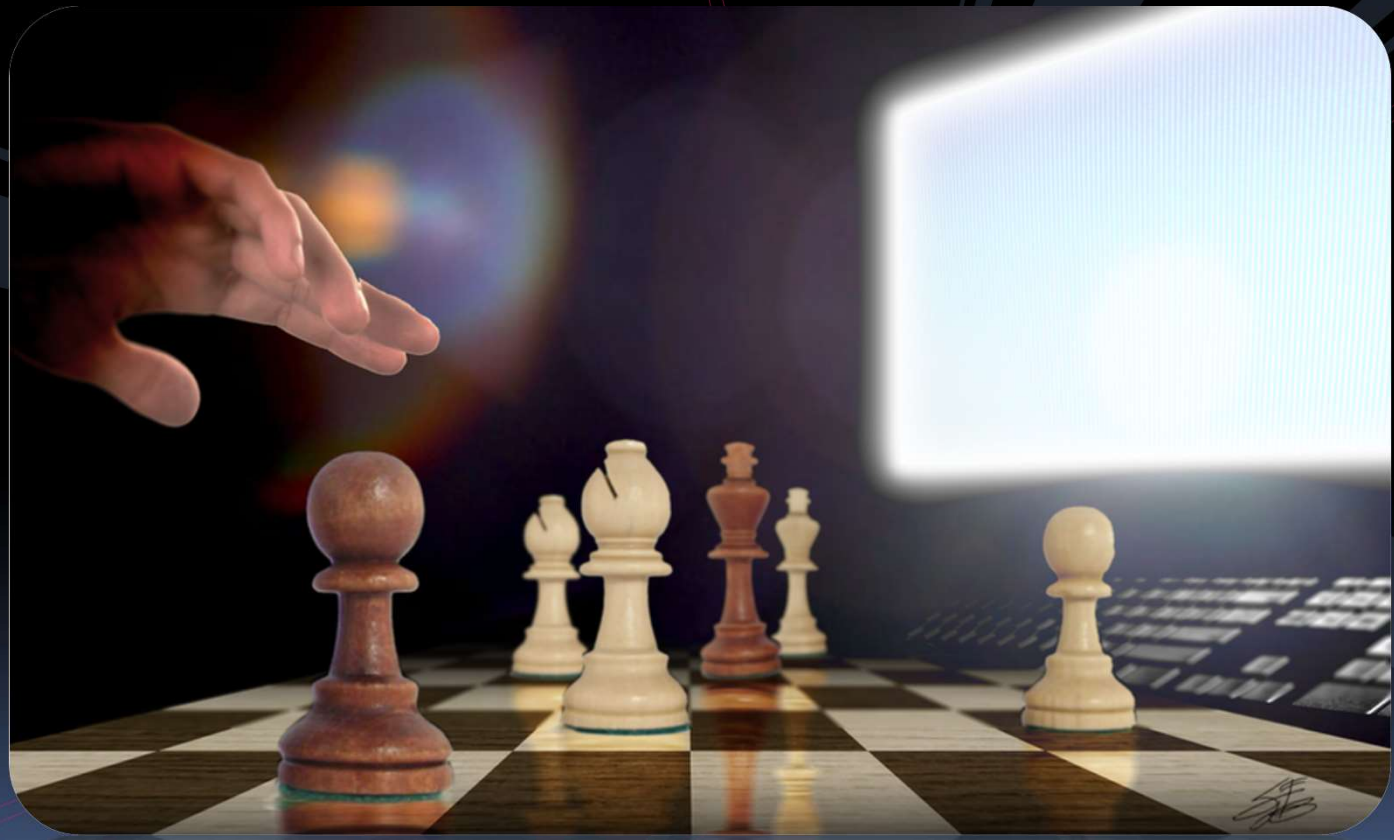
A* Search ("A-star")

- Still the basic standard for search algorithms in game AI (and in general).
  - Combine path cost with heuristic cost, and expand nodes based on smallest combined value.



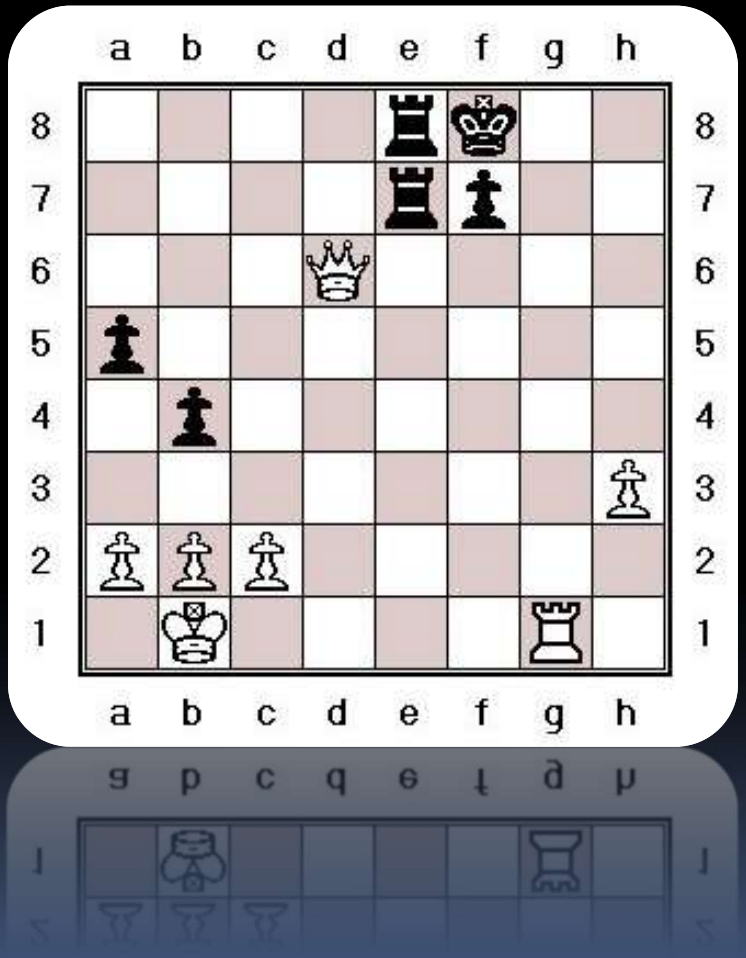
- Assuming good heuristic is used, A* guarantees finding optimal solution in reasonable time.

Classic Adversarial AI



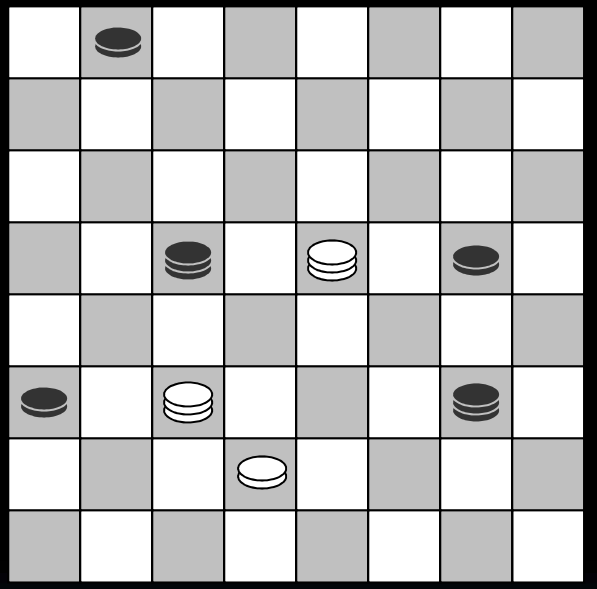
Adversarial Game AI

- Approach is slightly different when searching for solutions in an adversarial domain
  - Must always assume that opponent is trying to undo or undermine your position.
- Need to take opponent's possibly actions into account when planning one's own.



Assumptions

1. Domain can't be exhaustively explored
2. No strictly dominating strategies
3. Both the player's and opponent's strategies and positions are knowable
4. Both the player and the opponent are rational
5. Strategies are comprised of heuristics that can measure the "goodness" of any position with a numerical result
6. Game searches are pursuing a single goal
7. Zero-sum games being examined



What move by white guarantees victory?

Game Example: Othello

- In Othello, rows of opponent pieces are captured by surrounding them with two pieces of your own colour.
- As a result, positions on the edges and corners are more valuable than the middle.
- The value of an arrangement of pieces can be the sum of the weights of the player's pieces, minus the weight of the opponent's pieces.

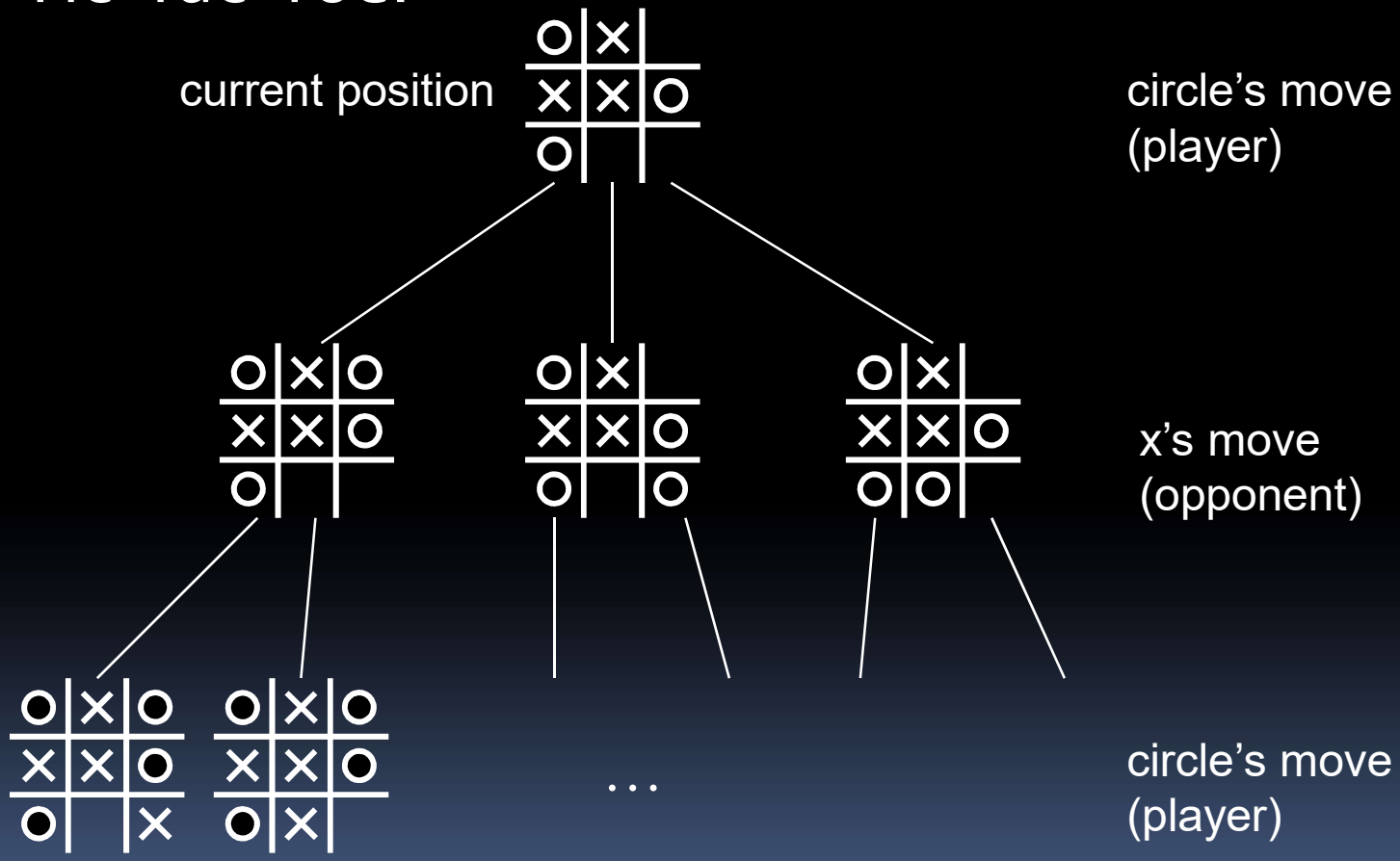
8	4	4	4	4	4	4	8
4	1	1	1	1	1	1	4
4	1	1	1	1	1	1	4
4	1	1	1	1	1	1	4
4	1	1	1	1	1	1	4
4	1	1	1	1	1	1	4
4	1	1	1	1	1	1	4
8	4	4	4	4	4	4	8

Game Trees

- Game trees are used to reflect all two-player game scenarios
  - multi-player scenarios are an extension.
- Same as search trees, but take the opponent's goals into account as well
- Game trees form the foundation for the traditional game-playing competitions:
  - Othello/Reversi: AI beat world champion (1980s)
  - Checkers: AI beat world champion (1994), solved in 2007
  - Chess: AI beat world champion (1997)

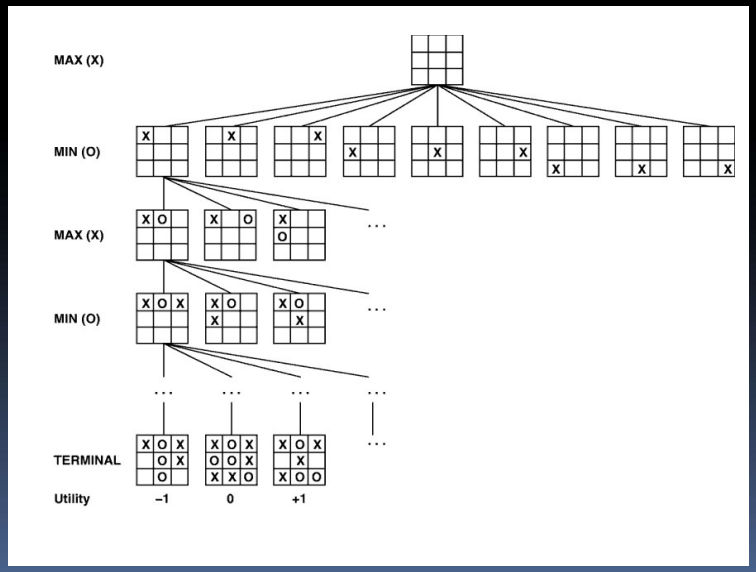
Game Tree Example

- Tic-Tac-Toe:



Branching in Game Trees

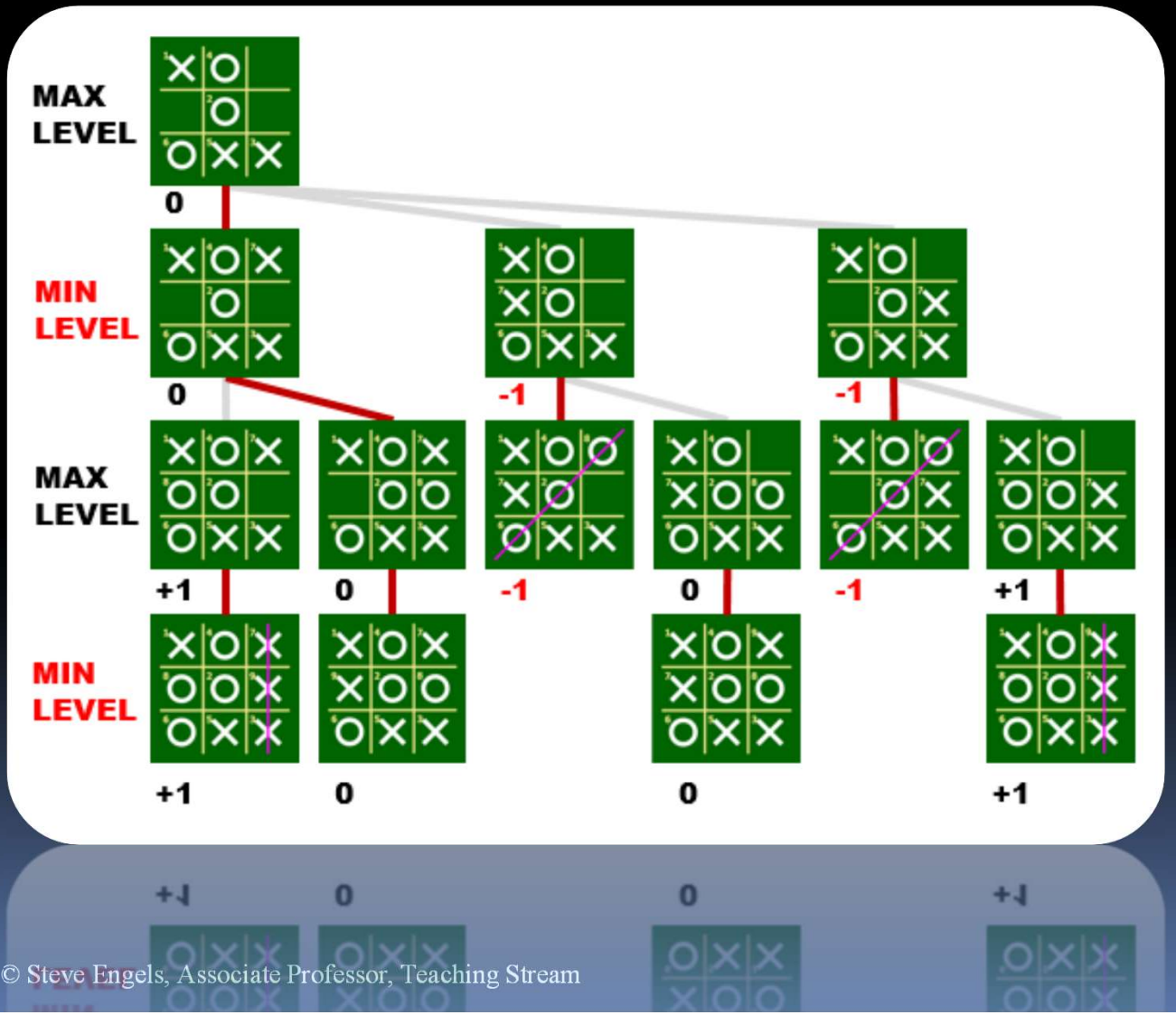
- Like any search tree, game tree branching can get out-of-hand very quickly
  - tic-tac-toe has 97,162 potential game positions to consider
  - chess has an average branching factor $b = 42$
    - two moves on each side produces over 3 million possible positions!
- Solution:
  - look limited moves ahead
  - use good heuristic function
    - keep extensive databases
  - use minimax principle
  - prune search tree



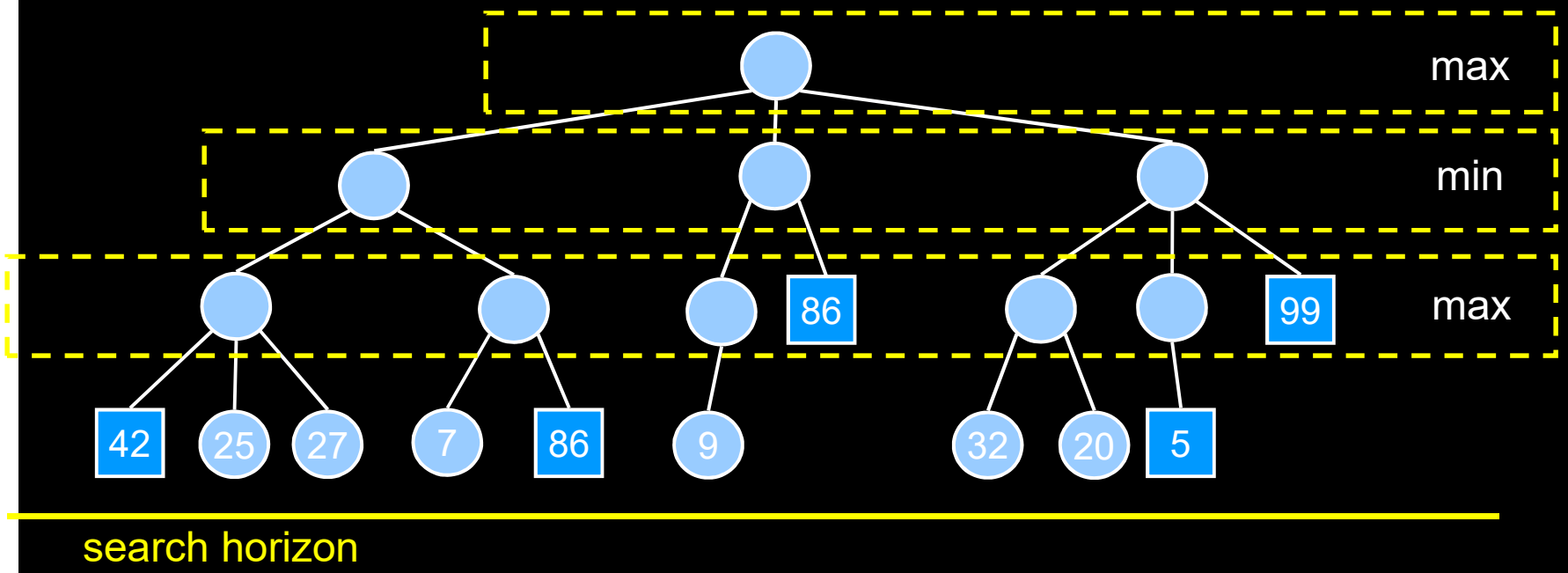
Minimax Principle

- Invented by von Neumann and Morgenstern in 1944 as part of game theory.
- Involves growing a game tree to the search horizon.
  - The search horizon is defined by the number of moves the computer looks ahead. If the computer look `n` moves ahead for itself and `n - 1` moves for the opponent, we say the computer is playing `2n - 1` ply.
- Within the tree bounded by the search horizon, apply the heuristic function to all leaves to calculate the utility of the position at each leaf.
- Idea behind minimax is that the AI player tries to maximize the utility while the opponent tries to minimize it.
- Note:
  - Leaf states are not necessarily end states
  - End states do not necessarily satisfy the goal conditions
  - Heuristic values at end states can vary

Minimax for Tic-Tac-Toe



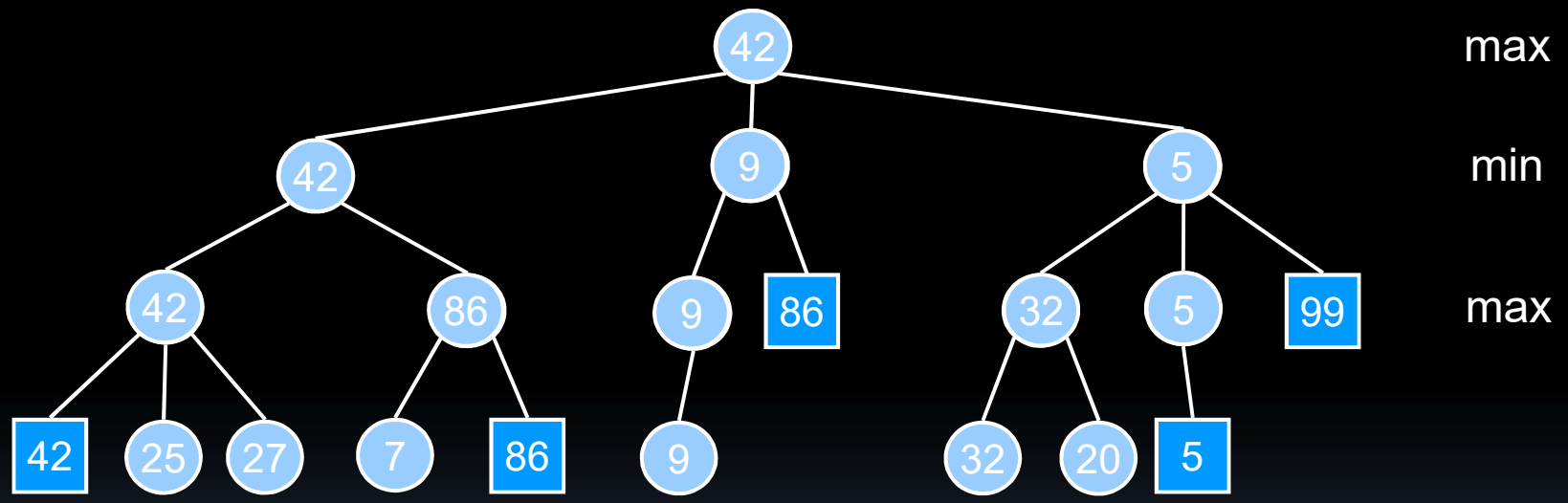
Minimax Example



- From root position, what branch should the AI player pursue to achieve the highest eventual score?
  - Note: heuristic values are from the player's perspective, not the opponent's. It is assumed that the opponent wishes to minimize the player's score at each min stage.

Minimax Example (cont'd)

- Minimax algorithm:
  - from bottom level to top, fill in node with either the maximum of its children (on max levels) or the minimum of its children (on min levels)



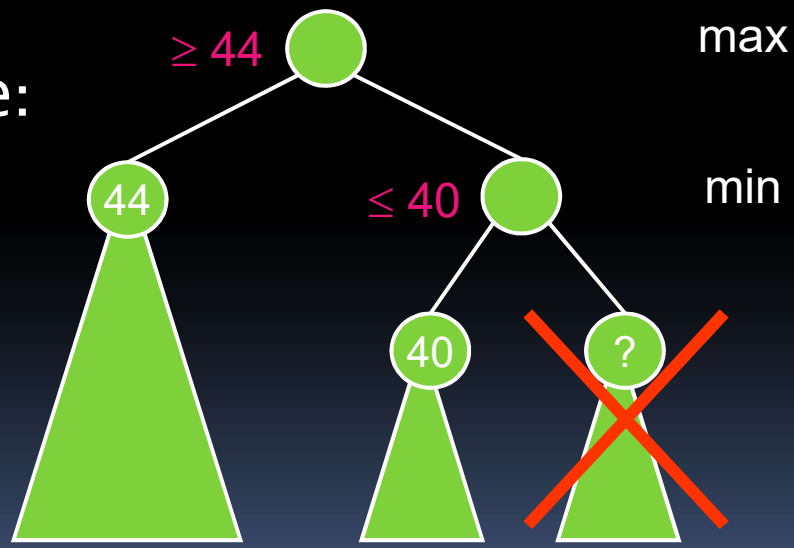
→ Best path is left-most branch

Minimax & Pruning

- Assumptions:
  - Heuristic is known by both player and opponent
  - both will make the most sensible move, given their goals
- Minimax alone will not solve the search problem
  - need reduction in search space in order to increase chances of arriving at the goal
- Pruning is a major technique used to reduce the branch density and speed up the search
  - e.g. prior knowledge about certain positions can allow any further positions to be discounted
- $\alpha\beta$-pruning can cut down the number of nodes searched *without losing information*

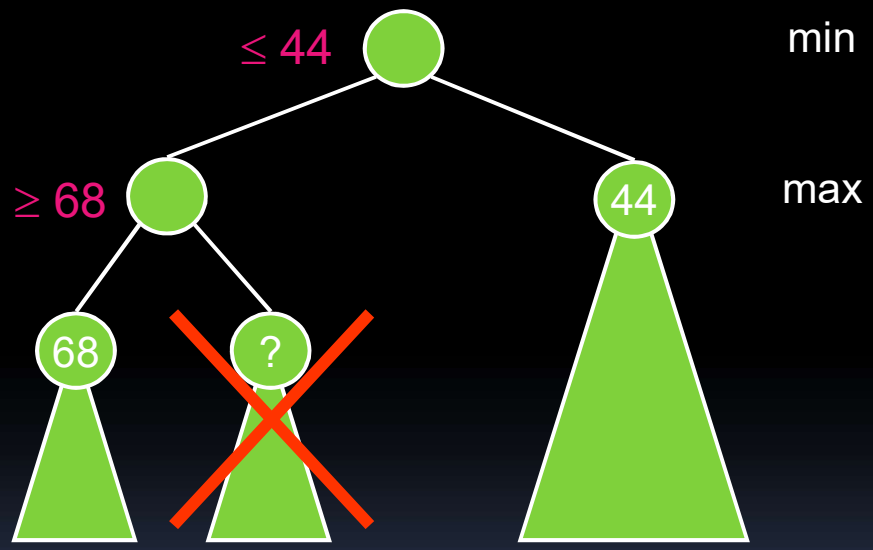
α-Pruning

- When exploring nodes in game tree, assumption is a limited depth-first search, from left to right.
- α-pruning keys on the max level, and discards any branches that won't affect the max level value.
- Example:



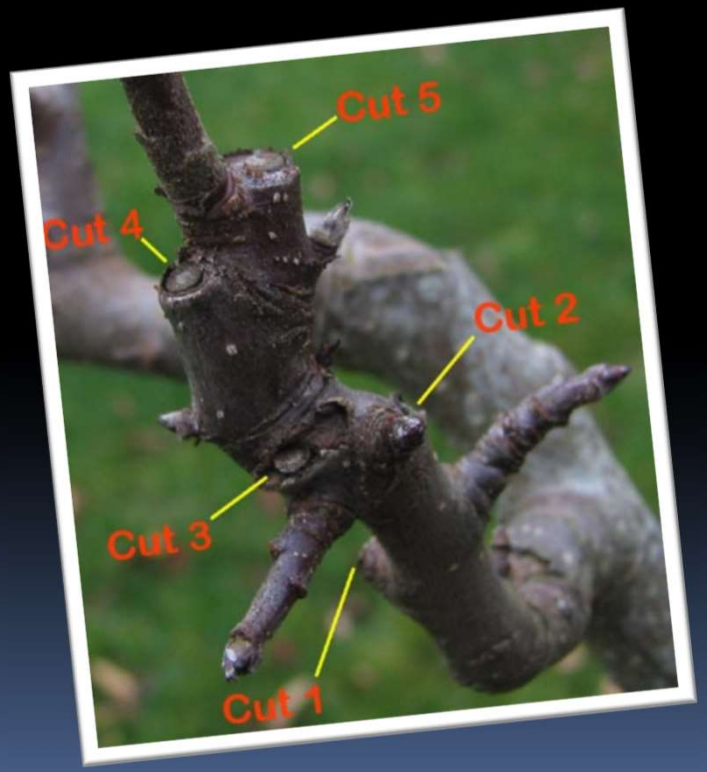
β-Pruning

- β-pruning is the same as α-pruning, only from the perspective of the min levels



αβ-Pruning (cont'd)

- Before a node and its subtree can be discarded, the algorithm requires tentative values for parent and grandparent nodes in tree
- Complexity reduction:
  - αβ-pruning reduces searching to $\sim n^{1/2}$ states
  - search is able to explore twice the depth of a regular search
    - very helpful with real-time game applications



Search Enhancements

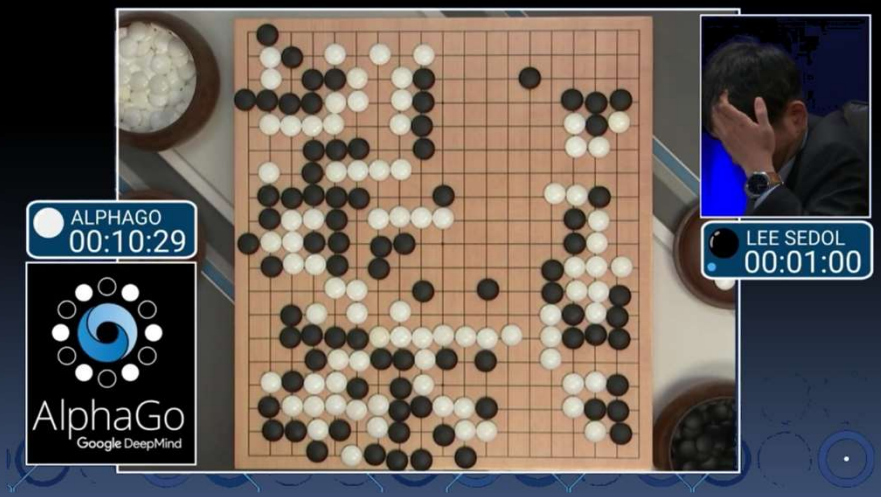
- Other techniques exist for speeding up search through game space
- Transposition tables
  - tables that record past positions, to avoid searching previously-explored subtrees
  - also used to eliminate subtrees that are permutations of other positions
    - e.g. Tic-tac-toe: initial branching reduces from b=9 to b=3
  - Disadvantage: most effective with iterative deepening search, which undermines use of $\alpha\beta$-pruning
  - must be careful not to equate two states whose positions are similar but situations are different
    - e.g. castling in chess, inventory in first-person shooters

Search Enhancements (cont'd)

- Opening book
  - games with set initial positions tend to have predetermined patterns (i.e. chess)
  - opening books reduce the search space in initial situations by performing one of a set of opening actions
- Endgames / Killer moves
  - certain situations that match recognized patterns can trigger a series of actions that guarantee an increase in the utility of the game state
  - endgames in particular lead to outcomes that guarantee a win
- Variable depth
  - the search horizon can be adjusted if a particular path requires more exploration (i.e. to realize a sub-goal)

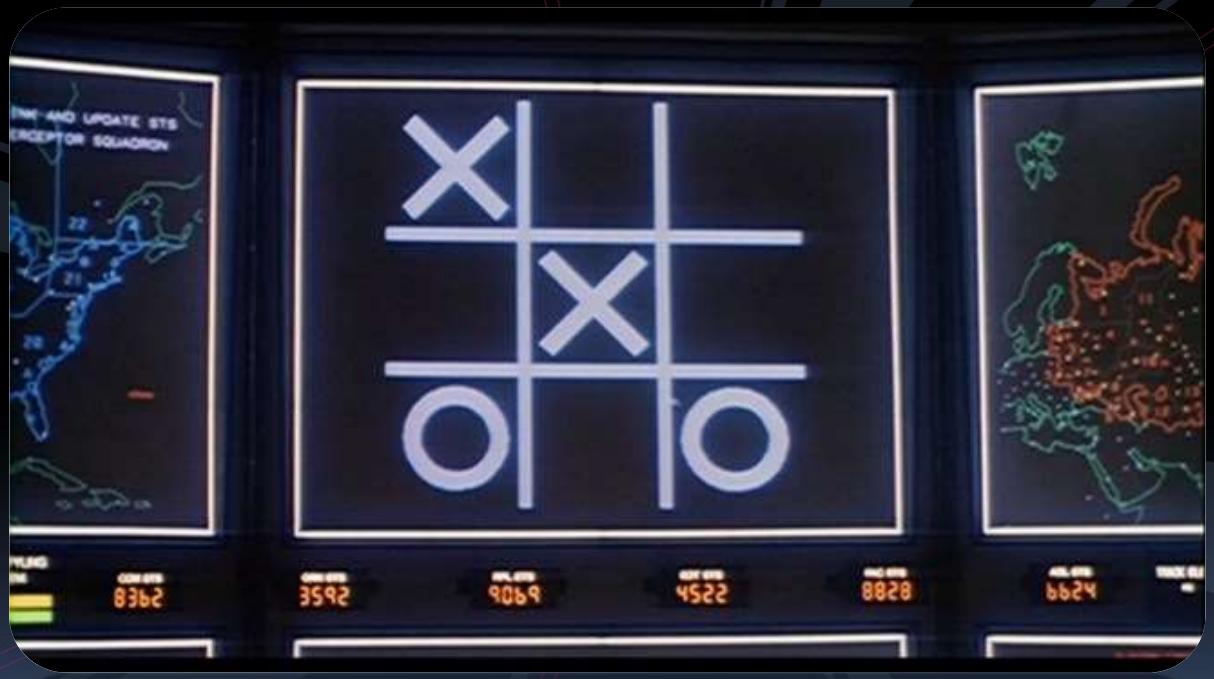
AlphaGo

- Going beyond the game tree rule.
  - (and a possible future for game AI?)
- AlphaGo: Beat world champion (2016)
  - Monte Carlo tree search, guided by neural network.
  - Trained off previous matches, then off matches with itself, using reinforcement learning.



Modern Game AI

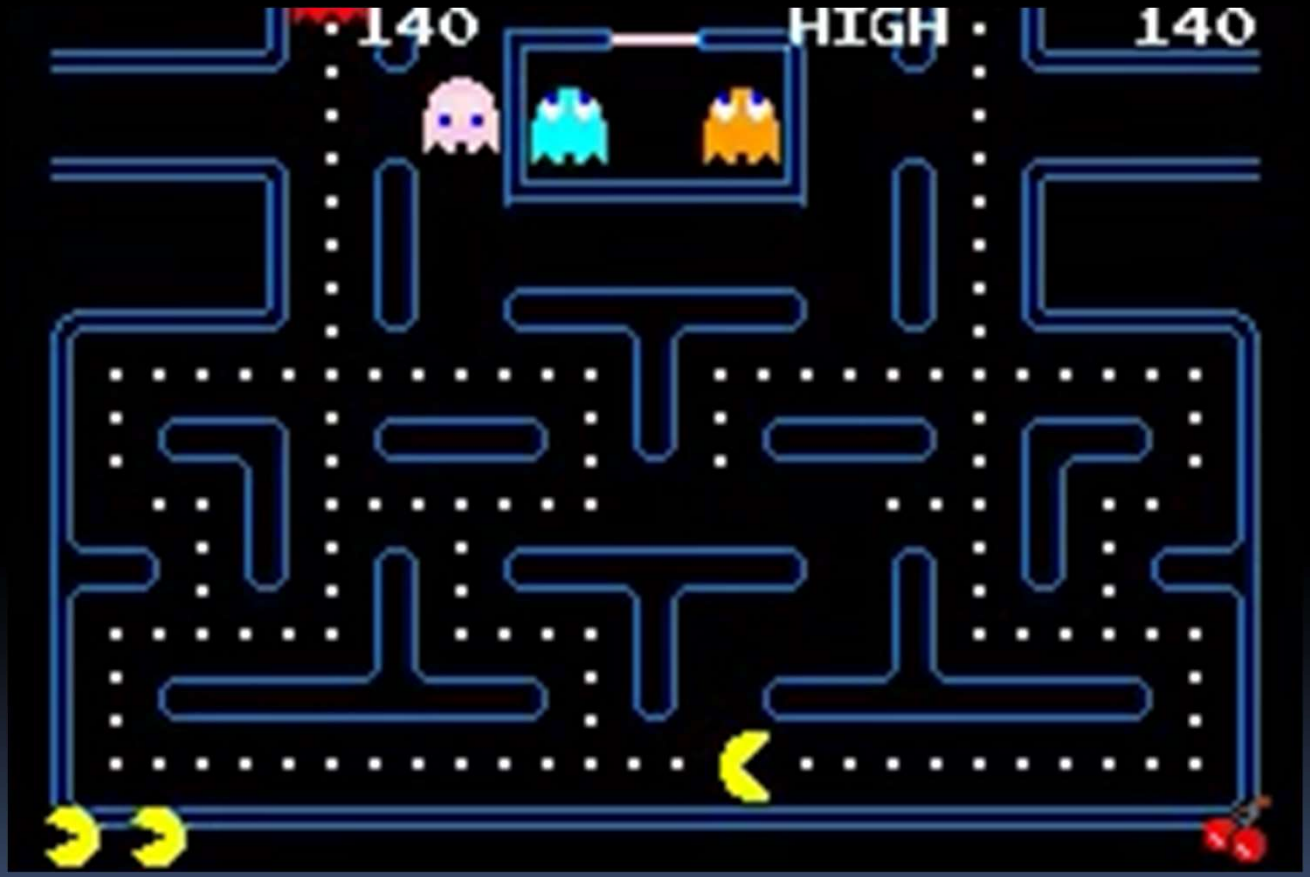
aka NPCs



AI in Modern Games

- Classic board games (chess, go) are no longer standards for measuring human intelligence over computers.
- Standards for modern AI involve modeling other human behaviours.
- Start by extending some of the classic AI algorithms…

Pac-Man



Burger Time

Extending Search

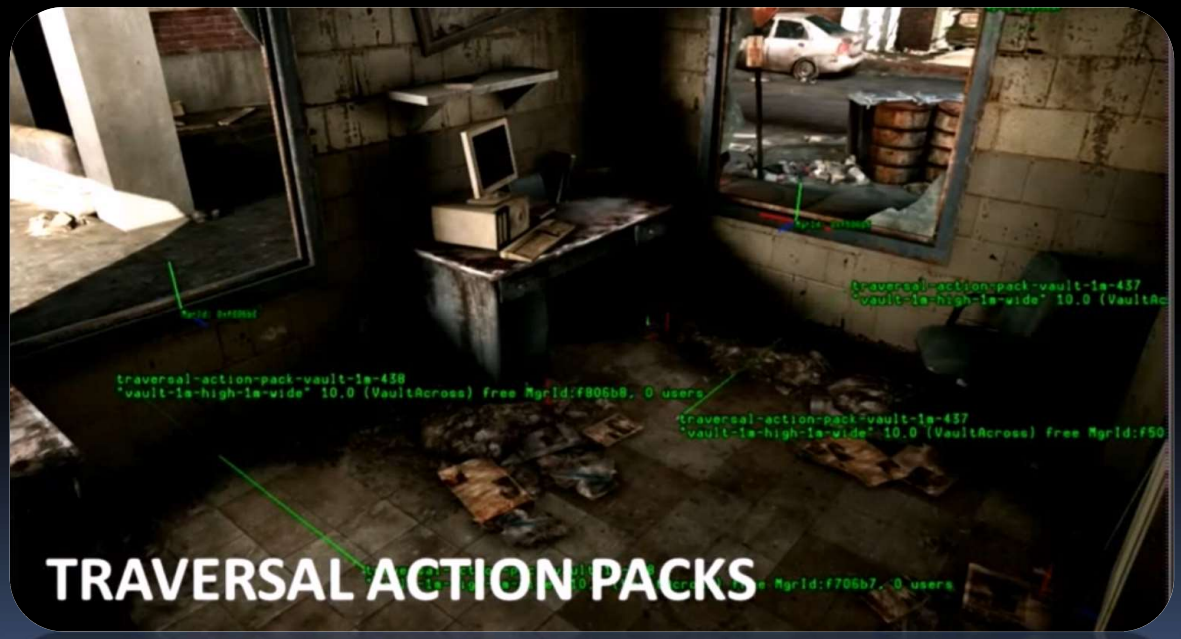
- Search continues to be a challenge in modern games, often because of extra constraints.
  - Need to take place in real-time
  - Environments can change
  - Enemies can alter the direction you take (either to seek them out or avoid them)
- Example: Pac-Man
  - Initial AI made red ghosts seek the player, while blue and pink ghosts positioned themselves in front of Pac-Man's mouth.

Extending A*

- A* continues to be a reliable way to navigate through a space.
  - Developers find ways to enhance it.
  - Example: Iterative A*
    - While calculating A*, use heuristic to begin navigating the player.
    - Once best route has been calculated, adjust current path to new route.

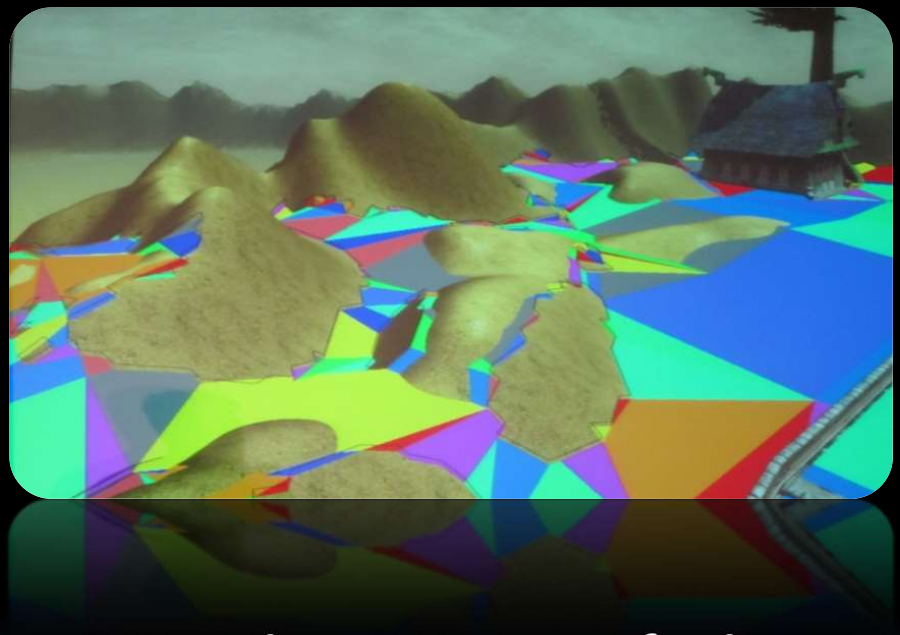
A* Variations

- Allow players to navigate over certain barriers
  - "Action packs" → Last of Us, F.E.A.R., etc.



A* Variations

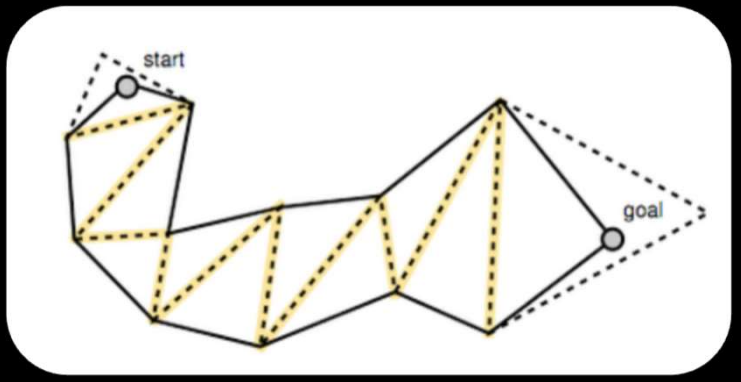
- Pathfinding
  - Key idea: Reduce search space.
  - Solution: Break up geography into sections, and search through the sections.

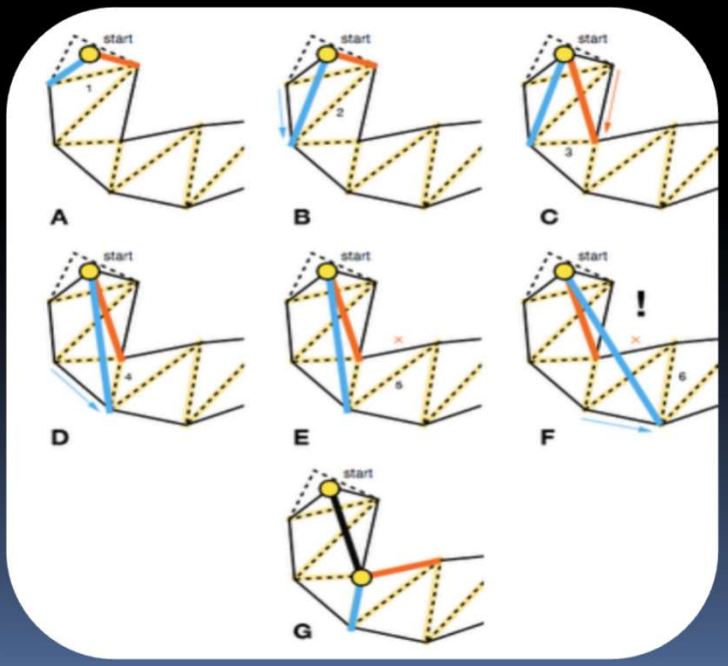


  - If target is within your section, then use A* to find an immediate route.
  - Form sections manually, or automatically by extracting significant boundary points, and drawing polygons by connecting nearest boundary points.

Funnel Algorithm

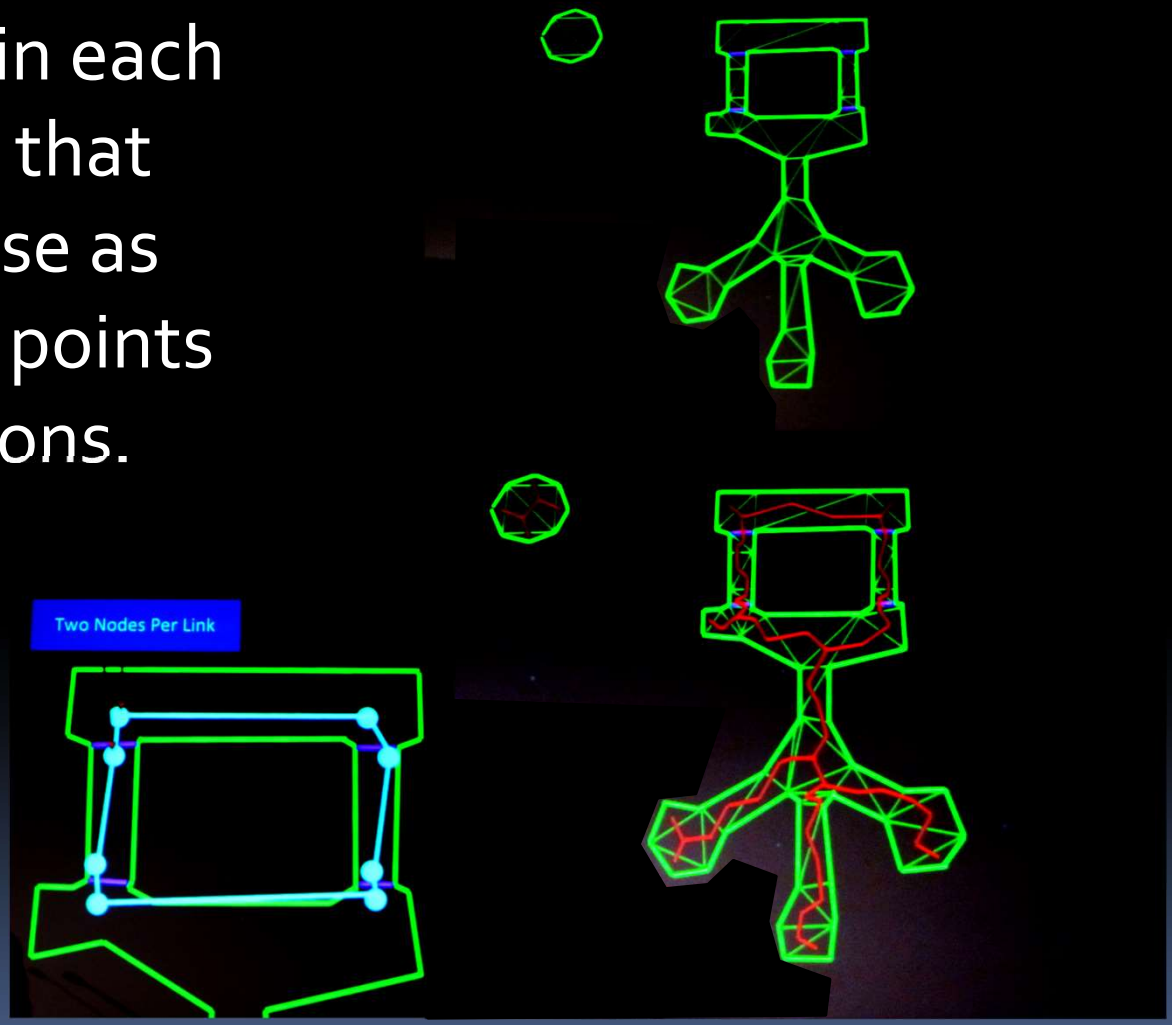
- Used to find quick paths through levels.
- Assumes that level has been decomposed into large polygons.
- Iterate through polygon corners to find narrowest funnel through passage.
- Multiple levels with different granularity
- Note: Always search for straight-line path first ☺





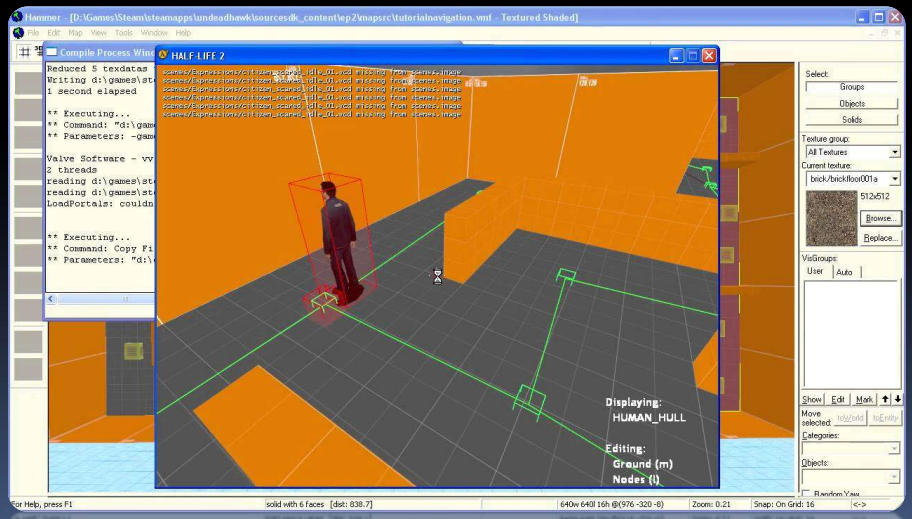
Pathfinding: Portals

- Create spots in each triangle edge that pathfinders use as intermediate points between regions.
- Example:
  - Playstation Move Heroes



What does this mean for you

- If your game has NPCs that need to traverse your level, consider dividing your level into regions, and setting up portals between regions to help your NPC's navigation.



NPC Opponents

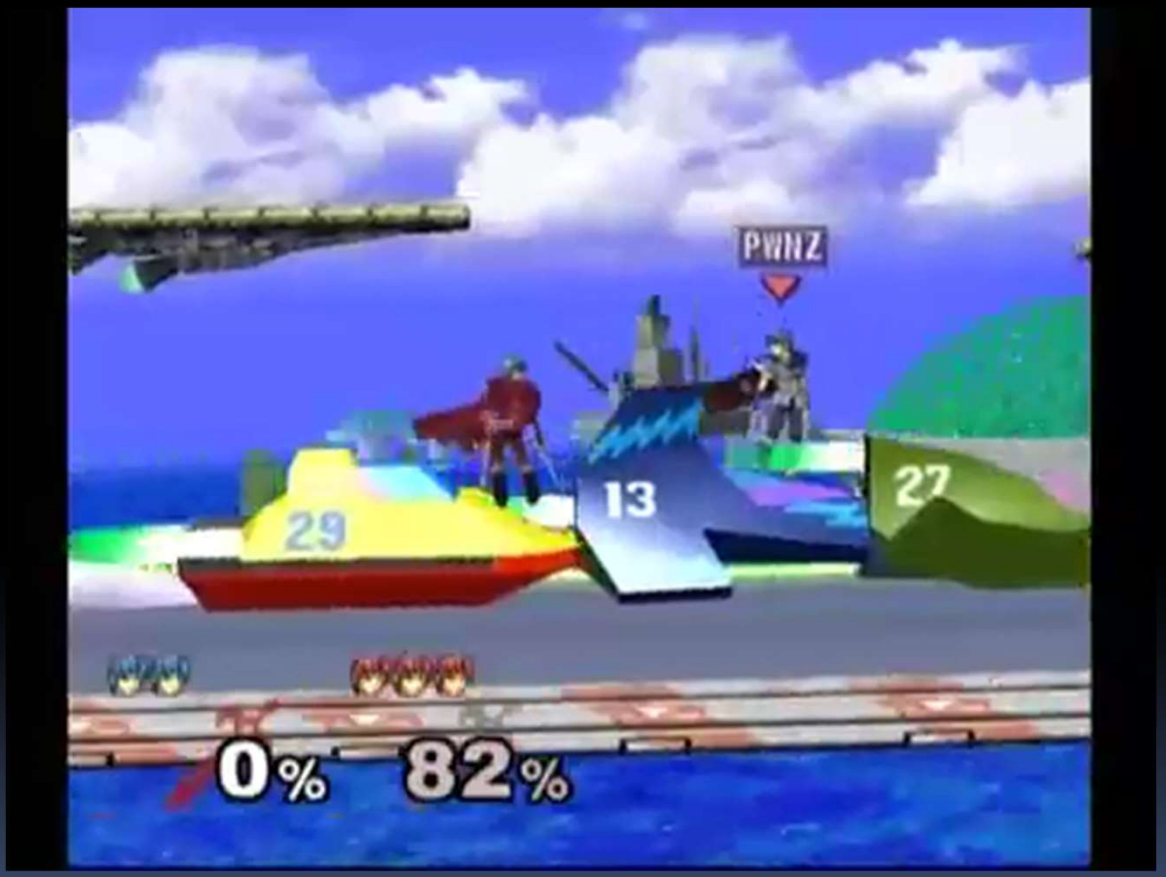
Intelligent opponents

- The presence of NPCs are meant to enhance the experience of the game.
  - Opponents should present challenges that test the player's abilities but are ultimately surmountable.
- As part of the overall gameplay experience, NPCs are meant to interact intelligently with the player and the world.

AI Example: Crysis

AI Example: Force Unleashed

AI Example: Smash Bros



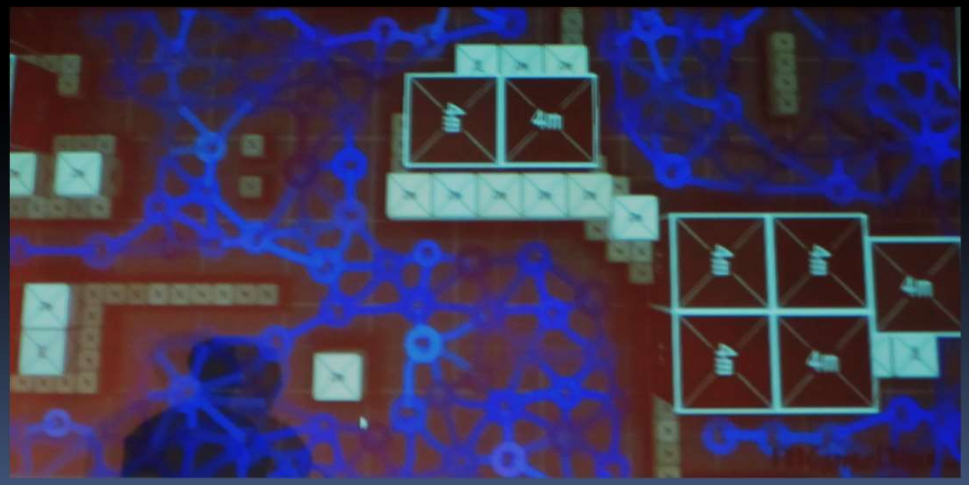
AI Example: Splinter Cell

Behaviours to explore

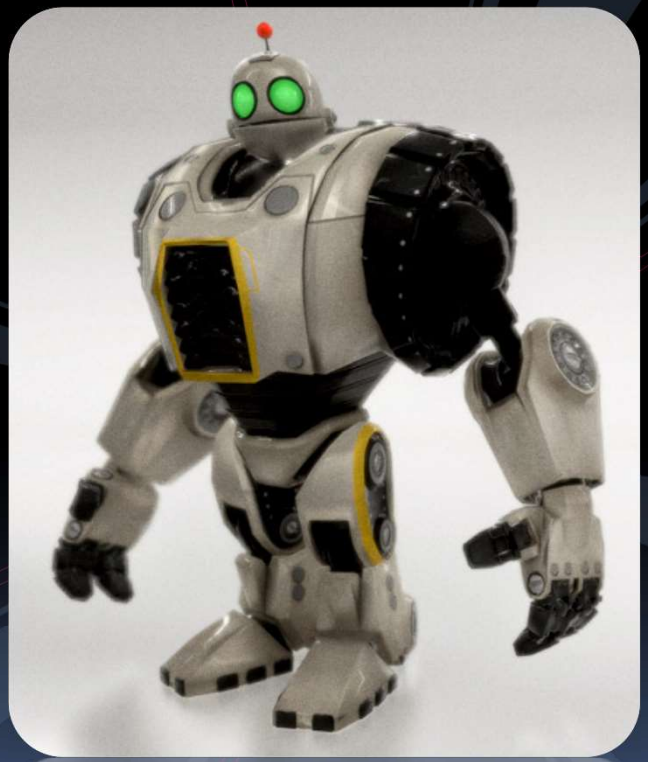
- Creature behaviour (following, flocking, grouping, separation, arrival, avoidance, etc)

Influence Maps

- Shows areas of control and influence for players.
- Implications:
  - Shows possible actions, future moves.
  - Defend where threatened, attack where weakest.
  - Emergent feigns and feints, teamwork.
- Based off spatial function:
  - Travel time, line-of-sight, A* penalty, path speed, target bias, weapon choice, multipliers.



Assistive AI



Intelligent NPCs

- This is what people typically think about when they hear "Game AI".
- Game developers want to make characters interesting, realistic, immersive and challenging.

Intelligent NPCs

- Animal behaviour!

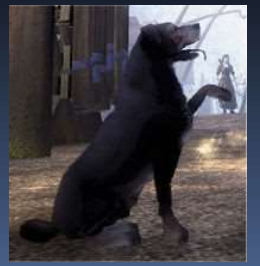
Helper NPCs

- Instead of providing part of the challenge of the level, NPCs have taken on a greater role in enhancing the player experience.
- Example: Buddy AIs

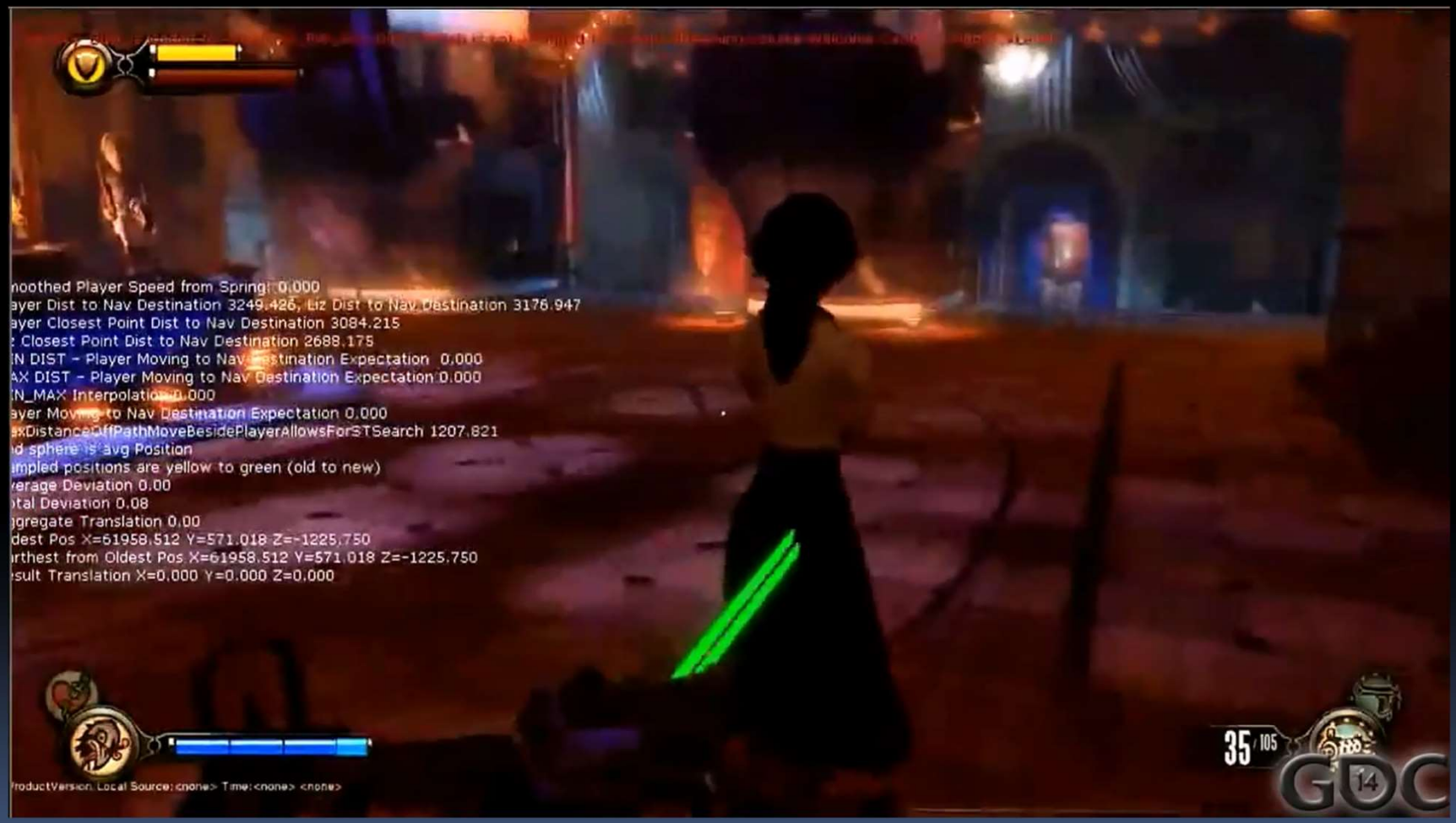






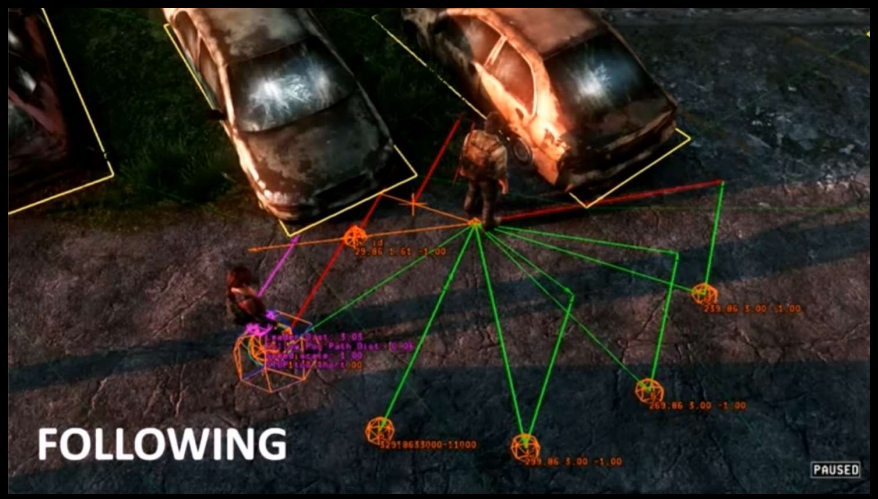


AI Example: Bioshock Infinite

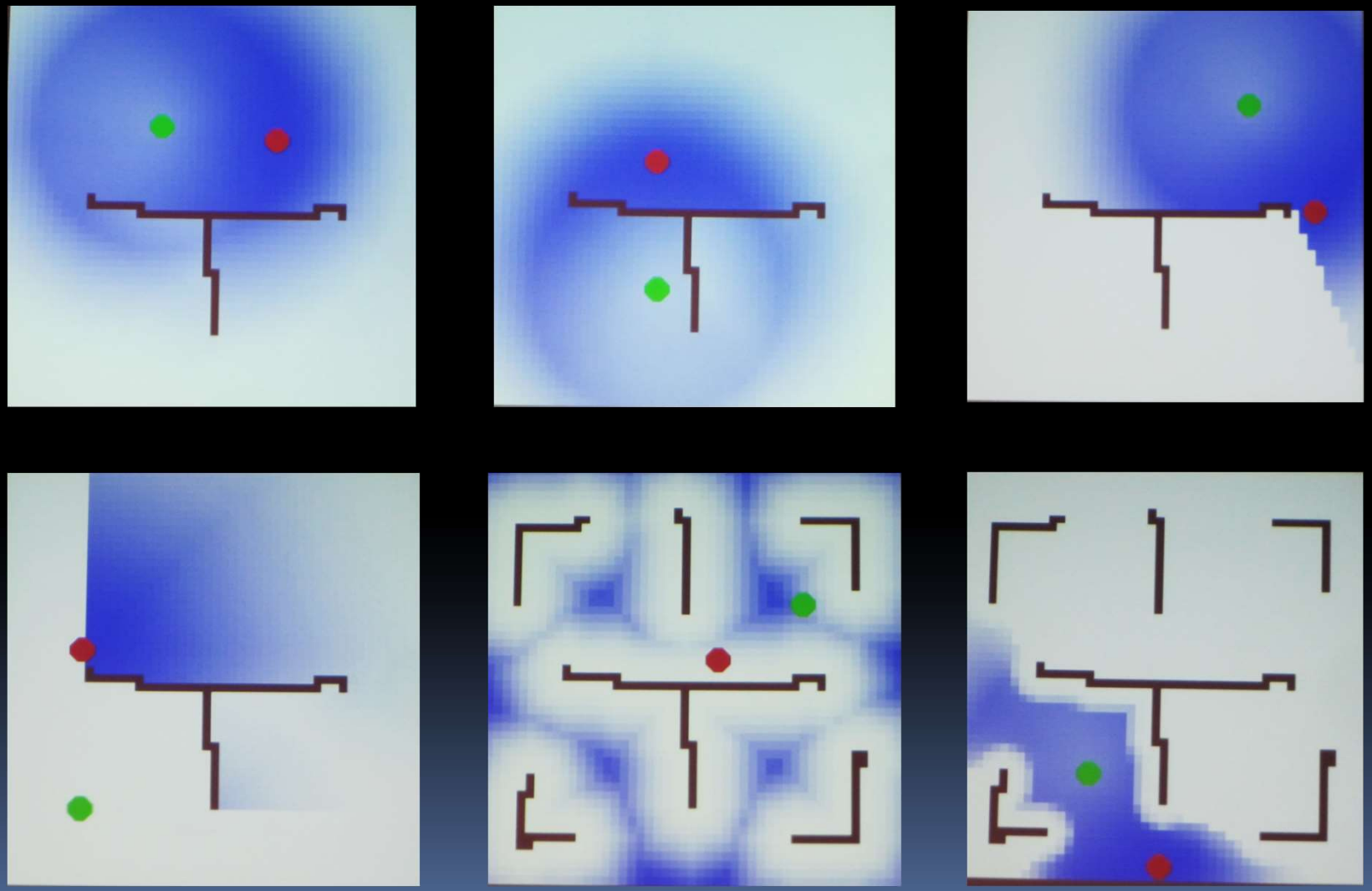


Navigation for Buddy AI

- Ellie in Last of Us:
  - Raycasting
  - Cover action packs



Influence Maps, Visualized



The Power of Game AI

- Obviously, some AI still needs work.
- Why is it important?
  - Immersion / Realism
  - Communication
  - Optimal challenge
- Let's revisit Elizabeth from Bioshock Infinite

IRRATIONAL GAMES

LEADING THE PLAYER – BLOCKING AND IMPROVISATION

Introducing "Goal Side"

- Fixed positions in the world for goals in Bioshock Infinite
- Use the "golden path" pathfind from player to next goal
- Move with the player

GDC 14

What artificial intelligence was taking place there?

Some of the AI at work

- Navigation
- Guidance
- Animation
- Scripted interaction
  - Attention
- Transmitting information
  - Puzzle-solving clues
  - Mood

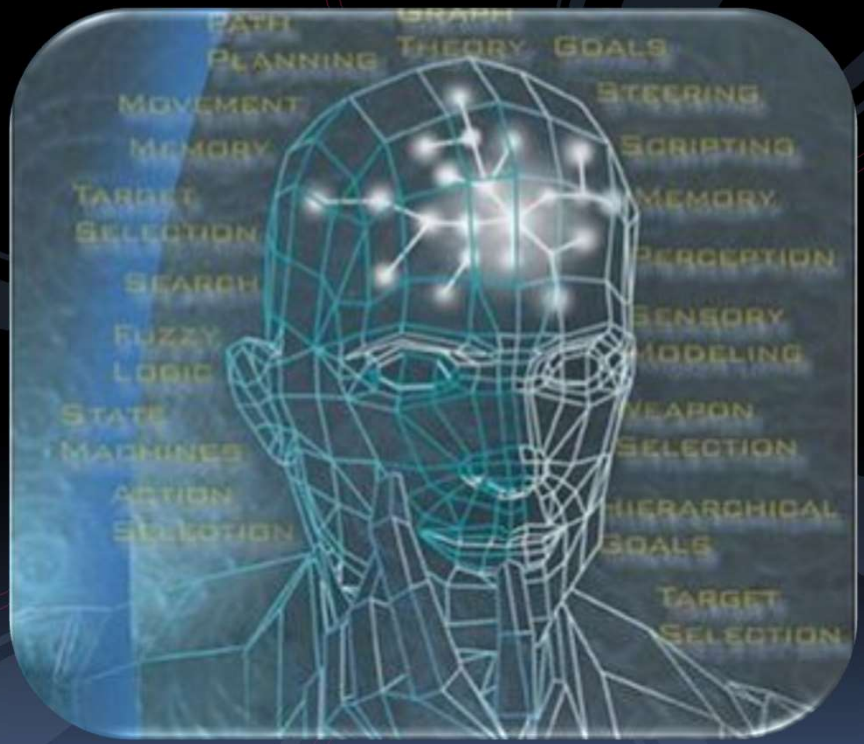
IRRATIONAL GAMES

Item tossing – More blocking & improvisation!

GDC 14

How do you solve the problem of the buddy AI running into the line of fire?

Advanced NPC Interaction



Artificial Intelligence Examples

- Perception
  - Language
  - Vision
- Processing
  - Searching
  - Planning
  - Game Trees
- Learning
  - Neural networks

Vision

- Character Vision

- Computer Vision

Natural Language

- Façade

Planning

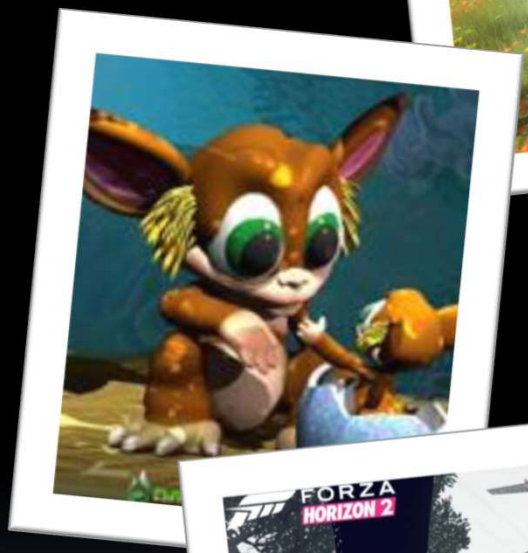
- F.E.A.R.

F.E.A.R.

Machine Learning

- Black & White

- Creatures



- Forza Drivatars

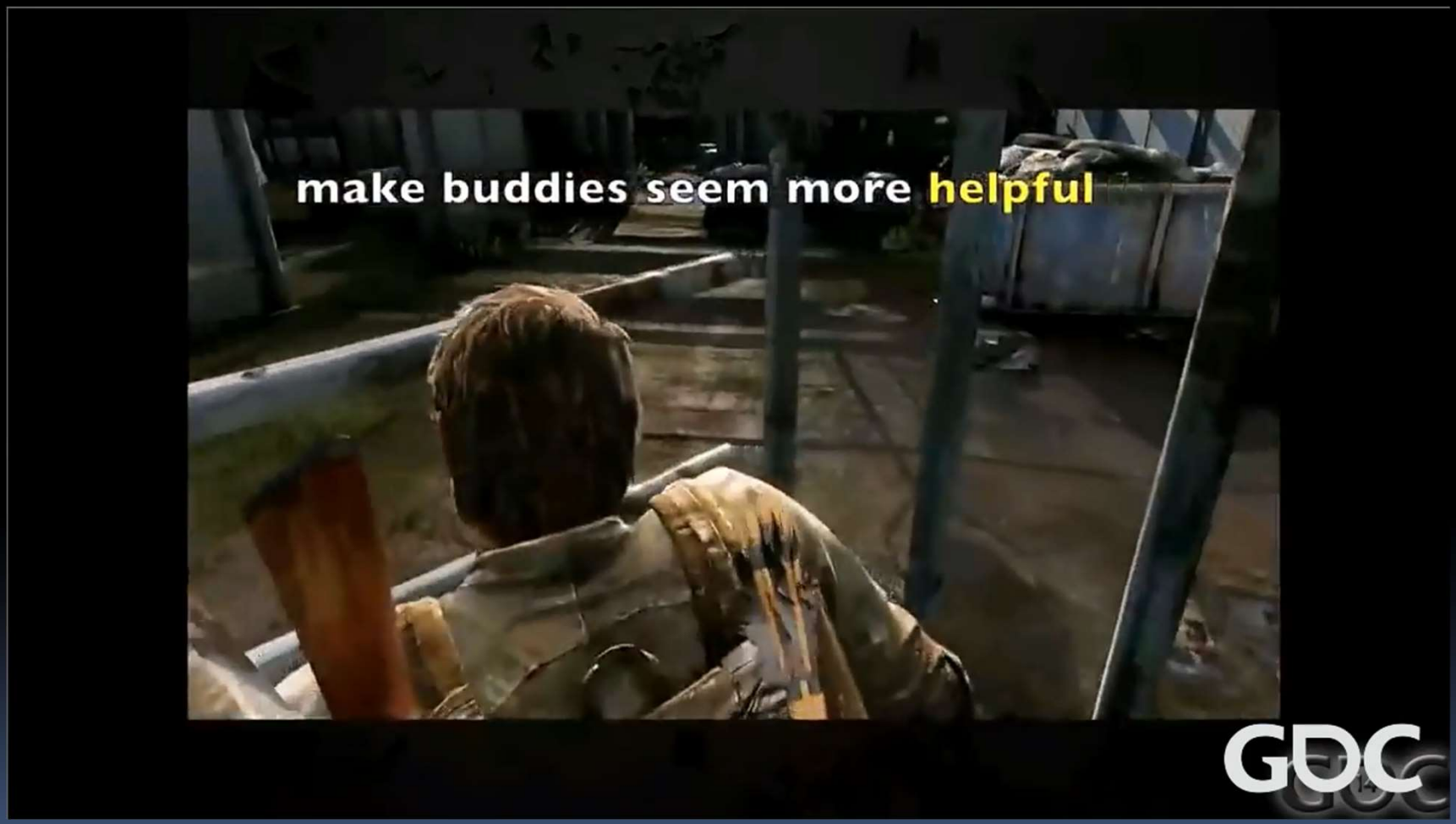
Creatures

Multi-Agent Applications

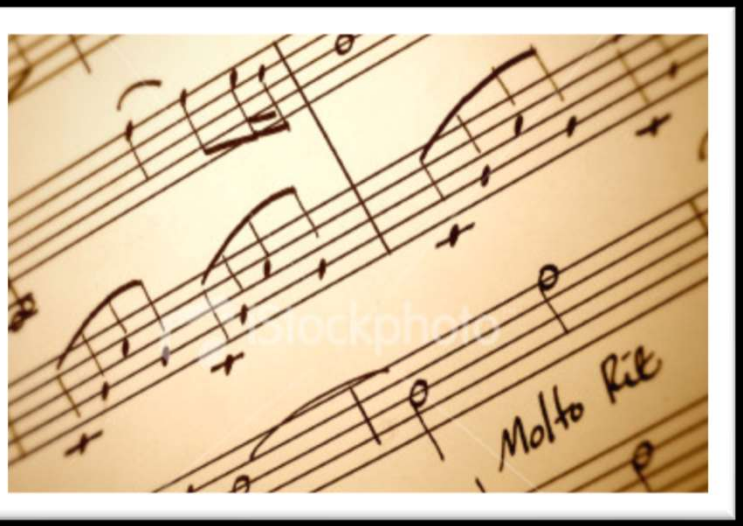
- Example: RoboCup
  - robot soccer league
  - international competition
  - also offers search & rescue, RoboCup junior, and a dance competition

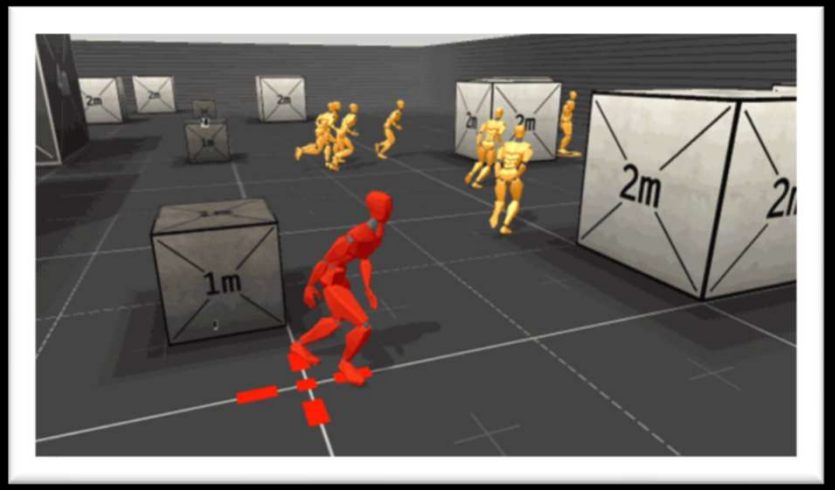
- Game example: Sports Games
  - Game AI has to coordinate multiple team members for a common goal, not just for their individual goals.

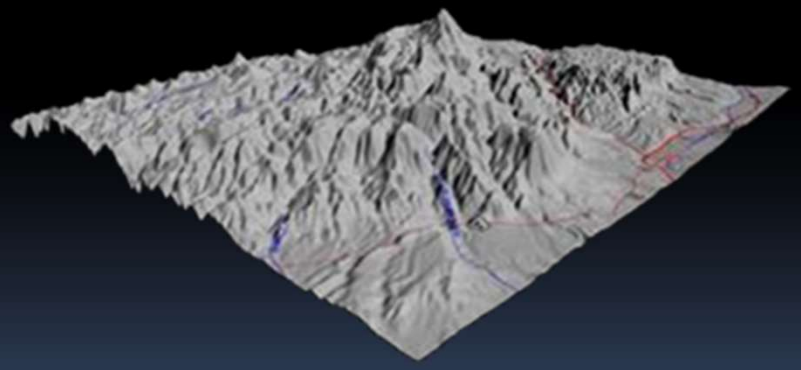
AI Dialog: Last of Us



Steve's Game AI Research


Molto Rit


1m
2m




VIDEO GAME SELECTION CENTER
Just a Touch Selects the Game of Your Choice
VIDEO GAME SELECTION CENTER

Game AI Issues

Game AI Issues

- Nearest neighbour searches are slow
- Player intent
  - What does a click mean?
- Destructive interference (conflicting goals)
- Grid resolution
  - Grid elements < body size
- Hierarchical searching
  - Problems with aiming for section, then searching in section.
- Randomness
  - Can produce seemingly oppressive behaviour.
  - Use Gaussians, filter out results (especially in near-win conditions).

End of Intro to Game AI

- Questions?