Game AI

Friday, May 16, 2025

4:49 PM

Intro

* AI - Solves problems that can be solved by conventional algorithms.
* AI in Games - controls an agent, usually an opponent for a fun experience.
* Even camera systems could be considered AI
* Procedural content generation can be considered AI
* Its easy to make a system aim perfectly, it is considered an AI problem to make a system aim in a more "fun" way that is more modeled around humans
* Procedural created content can include things like generated quests as well as generated areas

Basic Agent Movement

* Continuous vs Discrete movement
* Kinematic movement algorithm for seeking
* Kinematic Wandering, could generate direction using normal distribution, or generate two numbers between 0 and 1.0 and add together
* Kinematic Path Following
* Better movement with "Steering Behaviors" topics.
* Could add a sense of momentum

Computational Geometry

* Interacting with geometry, converting the game world into a form that can work with path planning
* Converting floats to ints can help support robust computational geometry
* Multiply by a float like 1000 and cast to int.
* Bake involves computationally expensive things that are precomputed and cache. Precompute things like lighting, navmesh's

Intro to path planning and grid lattice:

* Seek moves towards target based on a unit vector
* For obstacles:
  + Can slide around the object
  + Go to spot in grid that reduces distance to goal. Agent can easily get stuck
  + Can improve this by setting up the scene as a graph
  + Games like risk can be represented as a graph structure. Board looks like a map but connectivity is represented as a graph
  + Graph from uniform structure (Grid), tile based games
* Planning
  + Move to a goal
  + Represent the world as a set of states
  + Change state by applying operators
  + Planning is not just for path planning, can include reaching target technologies in civ, for example
  + Graphs are well understood and suited to the planning problem
  + Voxel - equivalent to a grid but in 3d where it uses a cube like structure
  + Creating the graph - discretize the game world if necessary (not already tile based or similar)
  + Quantize the agent and goal locations to the graph
  + Perform path creation with a search algorithm
  + Localize the path to the game world
  + Optionally clean up the path to look more natural
  + Agent moves to follow path until goal reached
  + Blind search and Heuristic search
  + Blind search uses no domain knowledge
  + Heuristic search could include heuristic rules
  + Heuristics drive low-level decisions
  + Video games provide domain knowledge that can be leveraged
  + Depth First Search, go to deepest node in the tree down the node you looked down first
  + Breadth First Search, consider all children of starting point before moving to grandchildren
* Discretized representation of the world is important. It needs to represent physical capabilities or it could lead to bugs or strange results.
* Hex board gives more natural 6-way connectivity, but is harder to implement
* Squares give 4 or 8 way connectivity. It is easy to implement but 8-way is unequal movement.
* Can temporarily turn off collision detection when an agent is stuck
* Random movement to try to get unstuck
* Have them destroyed when they get stuck
* Teleport the agent
* Post process cleanup of found paths are often referenced as string pulling