Principles of Computer Game Design and Implementation

Lecture 28
Outline for today

• Pathfinding 2
Tackling Paths

• Characters “live” in a computer world
  – Even developers may not know exact location
  • Physics simulations

• Pathfinders operate on discrete structures
World Representation

To use pathfinding

• Division Scheme
  – Quantisation and Localisation
    • Converting positions into nodes and back
  – Generation
    • Who and how define the mapping
  – Validity
    • Being able to fulfil the plan
Bad Quantisation

• Errors in quantisation can lead to invalid plans

• Plans have to agree with steering
Tile-Based Graphs

- $tileX = \text{(int)} \left( \frac{x}{tileSize} \right)$
- $tileY = \text{(int)} \left( \frac{y}{tileSize} \right)$

Works in square worlds
Tile-Based Graphs: Validity

• If wall are not parallel to tiles

• Will steering succeed?

• Not widely used in 3D games
Waypoints

Locations on map + edges
• Identified by designers
• Computed automatically
  – Corner waypoints
  – Points of visibility

Popular game AI technique
• Half-life
Navigation Mashes

- In modern games models are built from polygons (triangles)
- A character can always pass between adjacent polygons
- Fully automated generation of graphs
Correct Quantisation

• Several levels in the model
  – Take elevations into account when mapping to a graph node
Validity of Plans

- Character can always pass between adjacent polygons
- No direct pass between A and B
- Floor plan is done by designers and they avoid this
Chunky Paths

• Pathfinding may not produce a natural movement

• After a path is found
  – It needs to be smoothed
String Pulling

• Move A – B – C
  – If C can be seen from A, drop B

A – B – C

No change
Example

• Extreme case

• Even if there are obstacles, string pulling gives better paths
Splines

• Chunky paths can be further smoothed by converting them to *splines*
  – Curves that *approximate* paths
    • Some maths required (see wikipedia)
Passable Edges

• Not every agent can pass

• Need to adapt graphs for agents
Following Paths

• We assume that if a move is planned it can be executed
  – Validity of a division scheme

• What is the world changes
  – Other agents move about?
Possible Solutions

• Leave space between agents
  – Different pathfinding graphs for different agents
    • 😞
  – Centralised pathfinding
  – May not be natural (e.g. tanks)

• Assume there is no path
  – 😞

• Navigate around the obstacle
  – Steering / Pathfinding
Beware of the Pit

• Pathfinder requires to move X ➔ Y
  – Steering can fail
    • Navigation meshes are much better
      – Easier to re-plan (full information about passable areas)
    • Hierarchical pathfinding
Hierarchical Pathfinding

May not discover shortest path
Other Pathfinding Topics

• Cooperative pathfinding
  – Finding a path for a group of agents
• Variable terrain cost
  – Penalise paths near existing units
• Pathfinding using the GPU
• Pathfinding in dynamic environments
• ...

Pathfinding: Summary

• Algorithmically, not very difficult
  – A*
    – Choice of heuristics is important
      • Do not fear inadmissible heuristics!

• Linking model and graph can be tricky
  – A number of methods
  – Trend towards navigational meshes
    • Some developers disagree

• Paths often require smooting