INTRODUCTION

TOM WANG

2007 BS/MS CS
<table>
<thead>
<tr>
<th>*</th>
<th>KEY GAME MECHANICS</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>WORLD BUILDING</td>
</tr>
<tr>
<td>*</td>
<td>CONTROLS</td>
</tr>
<tr>
<td>*</td>
<td>CAMERA</td>
</tr>
<tr>
<td>*</td>
<td>AI</td>
</tr>
<tr>
<td>*</td>
<td>PERFORMANCE</td>
</tr>
</tbody>
</table>
WORLD BUILDING
WORLD BUILDING

- Set the atmosphere and tone of the game.
WORLD BUILDING

- Fill the world with engaging **characters** and **landmarks**.
WORLD BUILDING

- Teach players the rules of the world: concepts and interactions.
WORLD BUILDING

- Guide the player by defining **playable space** and boundaries.
WORLD BUILDING

- Guide the player by defining *playable space* and boundaries.
WORLD BUILDING

- Look for **content** to accelerate your development and iteration speed.
WORLD BUILDING

- Look for content to accelerate your development and iteration speed.
Within VR, consider human-accurate scale to preserve world immersion
WORLD BUILDING

- Within VR, use dimmer lighting & colors to ensure highest comfort
Take advantage of level design techniques

- Sightlines
- Landmarks
- Composition [Rule of Thirds]
- Symmetry

http://bobbyross.com/blog/2014/6/29/the-visual-guide-for-multiplayer-level-design
WORLD BUILDING

- For performance, utilize level of detail techniques
WORLD BUILDING

- Utilize plug-ins for specialized needs (sound engine, VR audio, physically based rendering, procedural trees, terrain generation)
Focus the attention in the world and highlight key areas and objectives.
CAMERA

- Position \([x, y, z]\)

- Rotation
  - Pitch / Roll / Yaw
  - Rotation Matrix
    - Beware Gimbal Lock
  - Quaternion

- Direction Vector
CAMERA [SIDE-SCROLLER]

- Appears to have depth with background or parallax, usually 2D playable space.
- Natural fit for mobile applications
CAMERA [TOP-DOWN]

- Represents information with limited perspective, useful for maps, game boards, user interface
- Natural fit for mobile applications
CAMERA [FIRST-PERSON]

- View that builds immersion and tactile feel for the player.
- Field of View (70 to 110 degrees)
- Enables more precise motion & controls
- Natural fit for PC or VR applications
CAMERA [THIRD-PERSON]

- Field of View: 90 to 130 degrees
- Best for action adventure games
- More objects on screen vs. first-person
- Highlights environment and character actions well: climbing, jumping, cover
CAMERA [VR]

- Use human field of view at eye height
- Keep camera motion smooth & steady - don’t shake or bob the camera
- Avoid motion blur, depth of field, etc.
- Adjust for interpupillary distance and keep straps tight
- Stationary more accessible than room-scale
CAMERA

- Spherical interpolation for smooth motion
- Use transparency for objects in near and far clip planes
- ‘Bump’ the camera to safe spots
- Try Unity Plugin iTween for splines and motion paths
For performance, enable **occlusion culling** to stop rendering occluded geometry.
CONTROLS
CONTROLS

- Help the player interact with the game world and characters.
CONTROLS [PC / WEB]

- Simulation
- Turn-Based Strategy
- Massively Multiplayer Online [MMO]
CONTROLS [MOBILE]

- Swipe
- Tilt
- Touch
CONTROLS TIPS

‣ Design controls to feel intuitive and match expectations for the player.

‣ Design controls to allow customization.

‣ Design controls to be consistent and responsive.

‣ Design controls to allow the player to build mastery and feel empowered.
CONTROLS DESIGN

- **Fitts’s Law**
  - Time required to rapidly move to a target area is a function of the distance to the target and the size of the target.

- **Hick’s Law**
  - Time required to make a decision when given a set of possible choices. More choices will increase decision time logarithmically.
AI
• Bring the game world to life with interesting NPCs and behaviors.
Decision Trees

- Finite State Machines are commonly used
- Search all child nodes in tree to evaluate choices
- Consistent, deterministic behavior
Decision Trees

- **Performance.**
  - Evaluating nodes can get costly with limited CPU
  - Need to budget against scripts, sound, physics
  - Can be improved with cached info, pruning

- **State Management.**
  - Every new game feature requires new states
  - Connecting states requires $O(n^2)$ transitions
Behavior Trees

- Hierarchical State Machines are often used
  - Fewer states to evaluate vs decision tree

- Can use information from self or group
  - Squads can share data and ‘roles’

- Evaluates dynamic behaviors by priority
  - Maslow’s Hierarchy of Needs
  - Can introduce probability for randomness
AI

Behavior Trees

› Heuristic Tuning.
  ◦ Requires constant re-balancing of priorities

› Indecision.
  ◦ AIs rapidly change strategies with new data
  ◦ Can be mitigated with hysteresis
Neural Network

- Learned Data Set & Identify Nearest Behavior
  - Radial Basis Function / Nearest Neighbor
- Can learn information over AI or human trials
- Can leverage cloud Big Data platforms
Neural Network

- **Data Size.**
  - Requires lots of experimentation for fitness
  - Selecting parameters can be tricky
  - May require a large amount of data storage
  - Data may invalidate as game changes

- **Implementation Complexity.**
  - Hard to debug with large # of parameters
  - Hard to reproduce poor behaviors

NAVIGATION
Straight Line with Local Avoidance

- Naïve and Simple - Plan a straight line from agent to goal
- If collision detected, select a random direction
- Paths are non-optimal and can get stuck in corners.
A* Gridmap

- Separate the world into grids and assign a travel cost to each grid.
- Keep an open set of each path, recursively search by adding neighbor grids and keep a running travel cost. Once a path found, cull other paths that are more expensive.
- Doesn’t handle dynamic obstacles well or obstacles that don’t fit neatly within a grid. Grid gets expensive to store in 3D with voxels.
NAVIGATION

Navigation Mesh

- Process the world and separate into navigable nodes and polygons (‘navmesh baking’)
  - Watershed Partitioning
  - Monotone Partitioning

- Strategy - find shortest path of nodes to goal

- Tactics
  - Apply path smoothing for realistic turns
  - Add off-mesh links for jumps, one-way drops
  - Support a navmesh per navigation agent size
  - Repeller fields to avoid obstacles or attractor fields to encourage flocking
  - Clever partitioning to avoid sliver triangles
Q&A

twang@riotgames.com