WPI

Procedural Dungeon Generation Algorithm

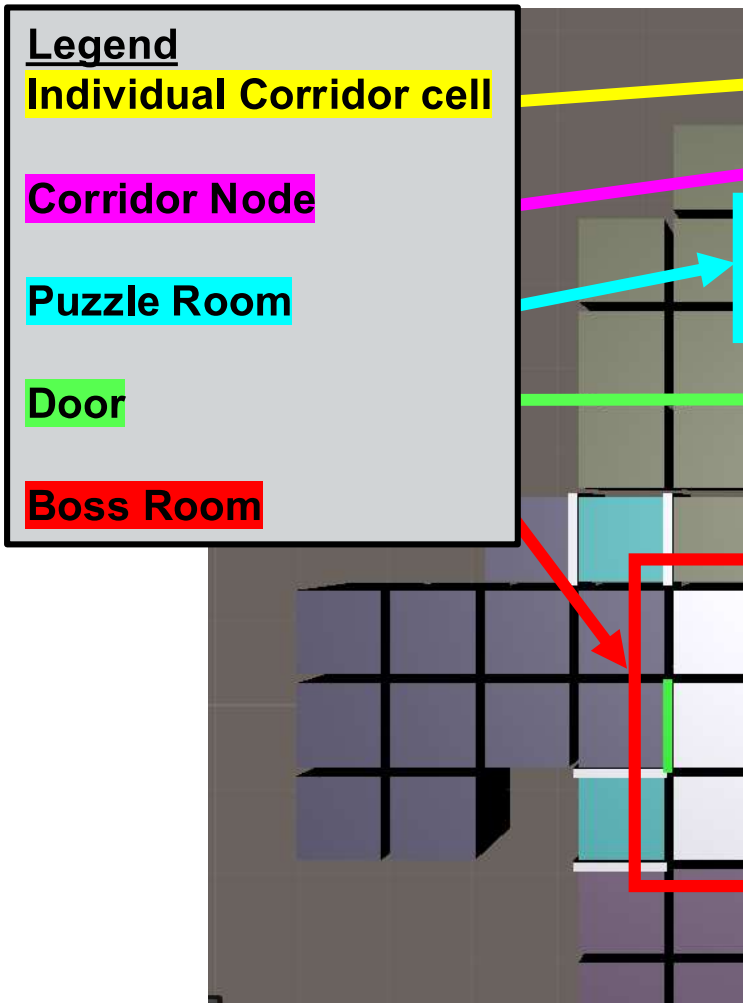
Jaskrit Singh, Luke Sanneman

Problem Statement

This project aims to use **Motion Planning** to create a **Proce**
Dungeon Generation algorithm for an educational game. E
algorithms lack the **flexibility** to account for custom room sh
the high number of **direction choices** required by a game to
an **authentic, unique user experience.** The creation of this
algorithm will be the first step in the process of embedding
educational content as a physical space that players can exp

Methodology - Definitions

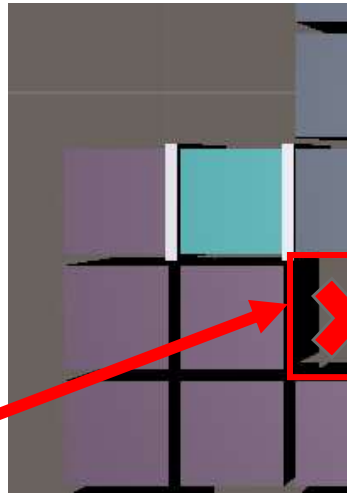
- Define a discrete environment with Corridor Cells and Doors
- Randomly generate locations for X amount of "Boss" Rooms
- Represent the map as a graph of Corridor Nodes, with Puzzle Rooms as edges

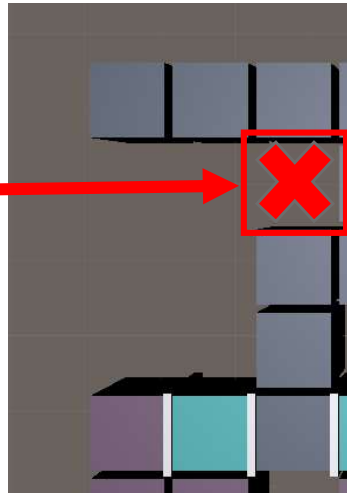


Methodology – Validity Checker

- Check that the cells don't collide with existing cells
- Check that doors don't conflict with walls and doors of other puzzle rooms
- Check for Invalid Puzzle Connections
  - for corridors, check that the corridor does not connect supercorridors that are already connected via puzzle room
  - for puzzlerooms, check that the puzzleroom start and end are not the same supercorridor

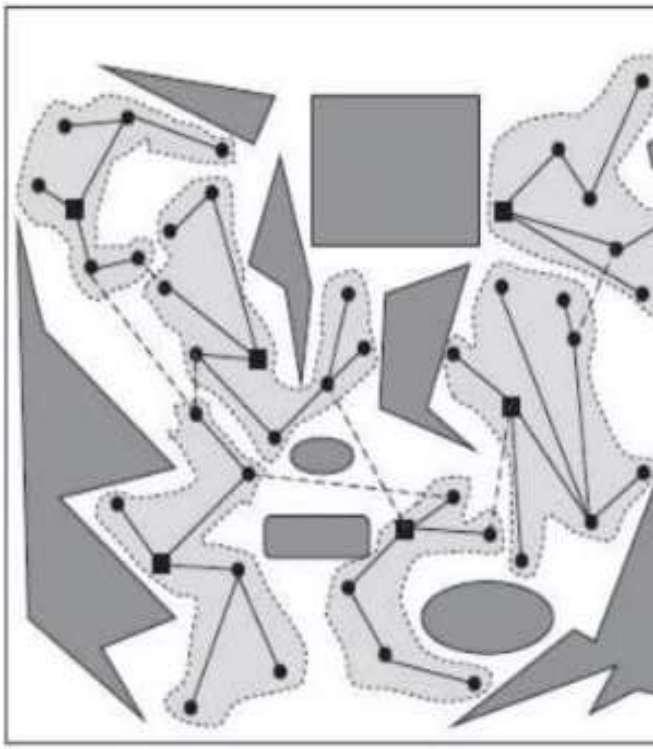
The validity checker is called every time a new component is added





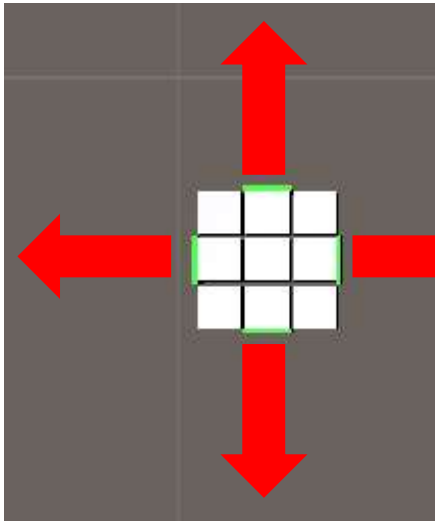
Methodology - Overview

- Connect boss rooms using the Sampling Based Roadmap of Trees algorithm
- Local Planner Must
  - Generate Corridors that with lots of connections to other Corridors
  - Explore the space well
- Global Planner Must
  - Connect all boss rooms in multiple ways



Methodology – Local Planners

- Several different planners were used to generate the dungeon around each boss room
  - EST
  - KPIECE
  - RRT
  - Combinations



Methodology - EST

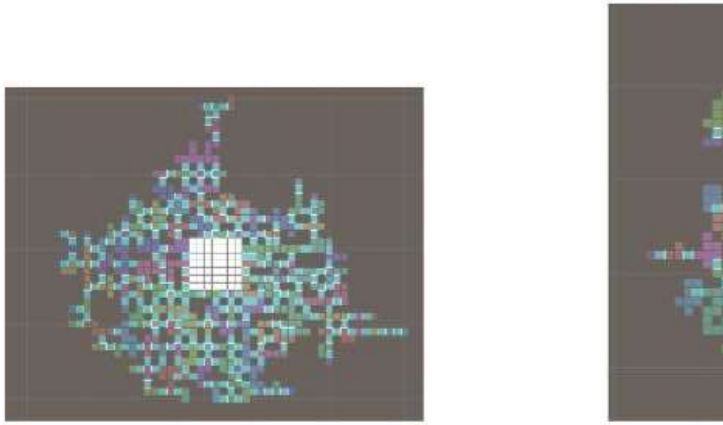
- Expand each Corridor Node based on weight

$$w = \frac{1}{1 + \#neighbors}$$

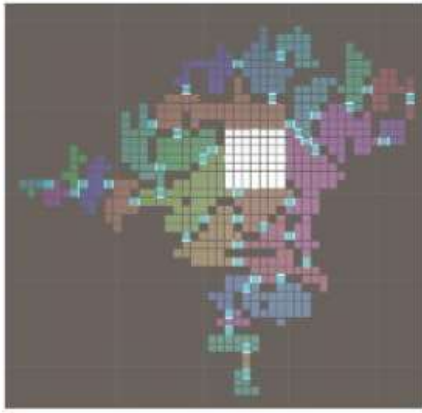
- Choose between adding a Cell and adding a Puzzle Room.

$$P(AddCell) = e^{-rn}$$

n: number of cells
r: decay parameter



(a) r=0.5



(c) r=0.01

Running EST for 1000 iterations with different decay parameters.

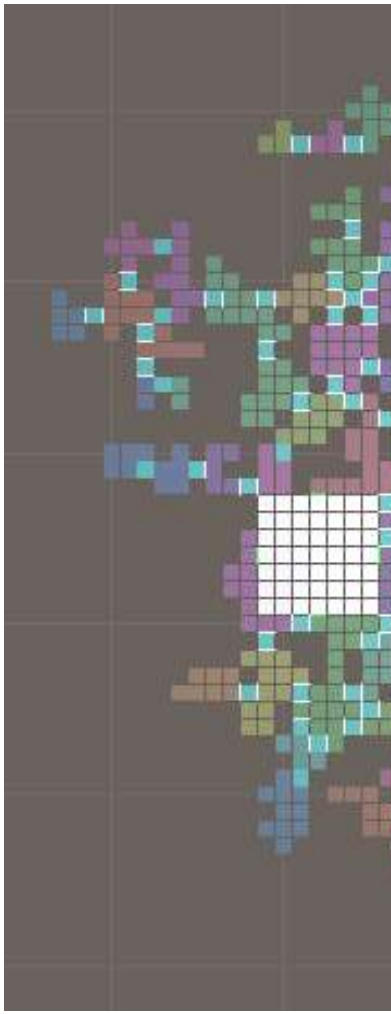
Methodology – KPIECE (simplified)

- Expand each Corridor Node based on weight

$$w = \frac{\log(i)}{sn}$$

i: iteration first cell added
s: number of times expanded
n: number of neighbors

KPIECE dungeon generation after 1000 iterations with r=0.05



Methodology - RRT

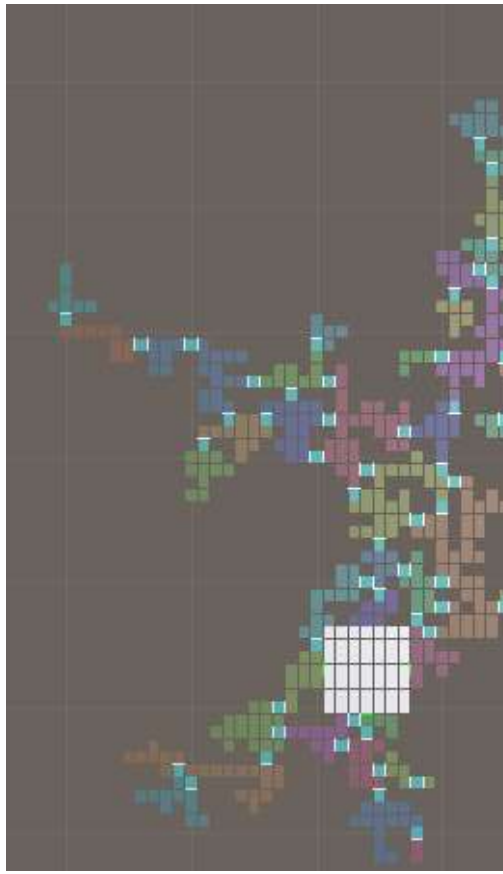
- 1. Sample a random point within our generation area.
- 2. Find closest corridor node to the sampled point
  - Faster to determine than by individual cell, allows for same decay function based on node size
- 3. Extend node by one cell or one puzzle room in the direction of the point.
- Most corridors appear in a line of corridors and there are no loops in the dungeon.
- EST and KPIECE – good at connecting nodes, bad at exploring
- RRT – good exploring, bad at connecting nodes

RRT dungeon generation after 1000 iterations with r=0.05

Methodology – Mixed Planner

- Use both RRT and KPIECE/EST with some chance
- Expansive aspect of RRT
- Connective aspect of KPIECE/EST
- Room size isn't too blobby or too long & narrow

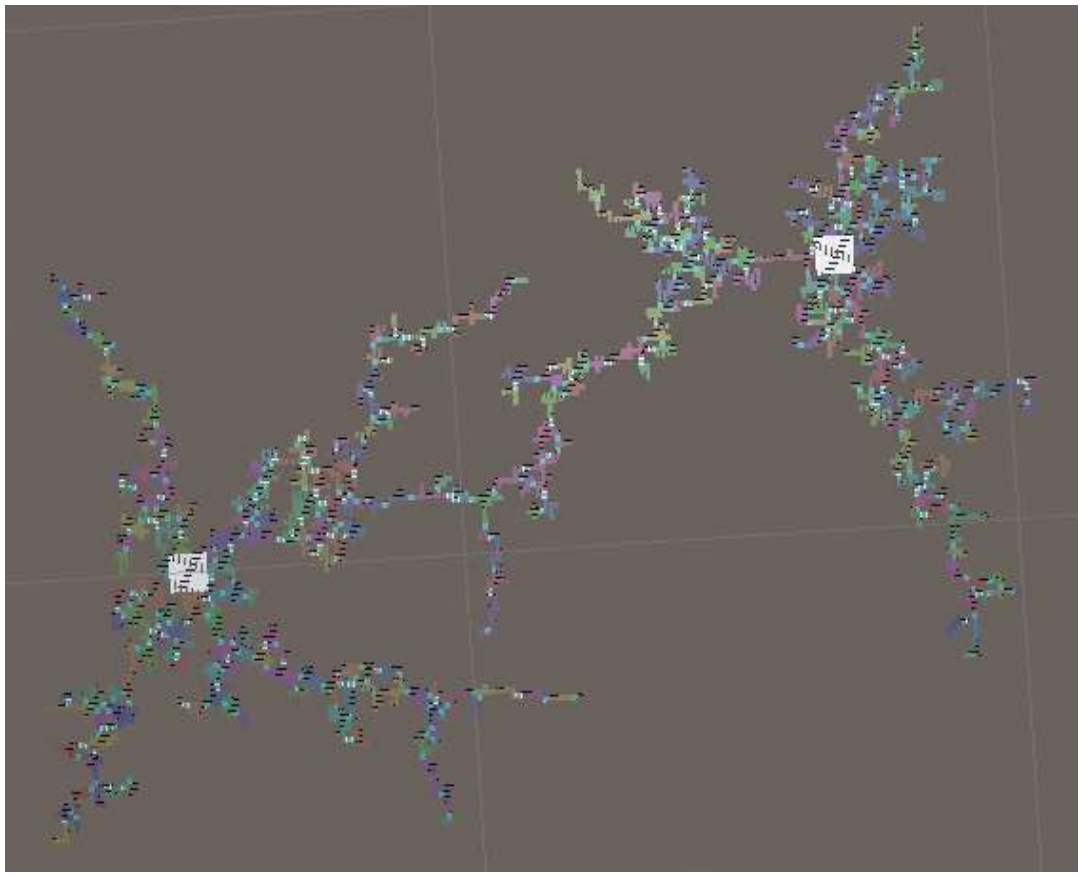
Expansion: 80% chance to use KPIECE
20% chance to use RRT



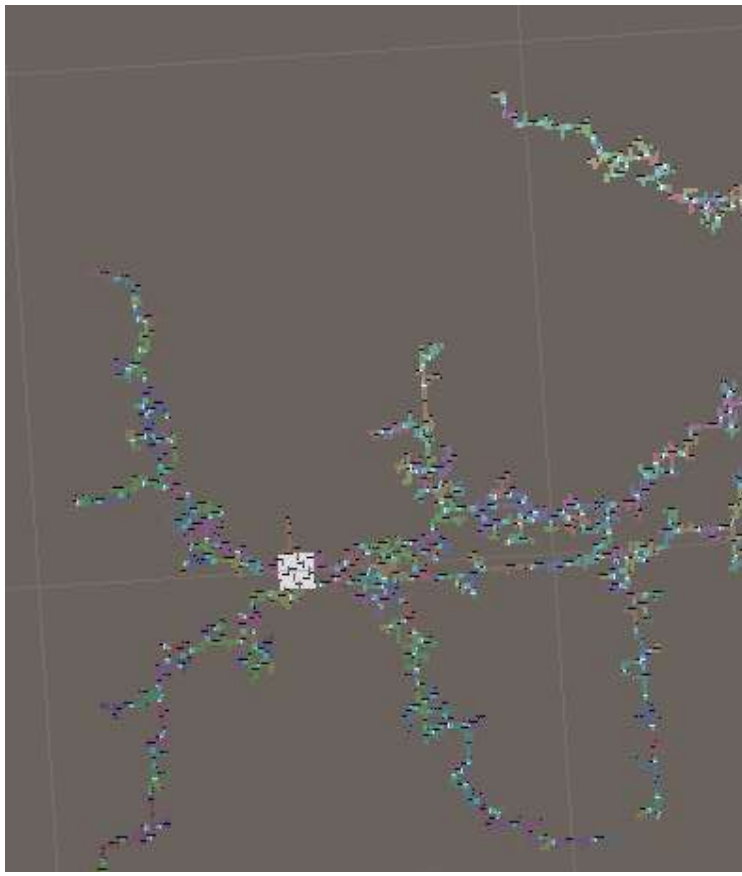
Methodology – Connecting Boss Rooms

- Grow a dungeon around each Boss Room
- Sample Corridor Nodes from nearby Boss Rooms as locations to expand towards
- Iteratively trim Corridor Nodes that are under-connected

Methodology – Sampling Nodes from other

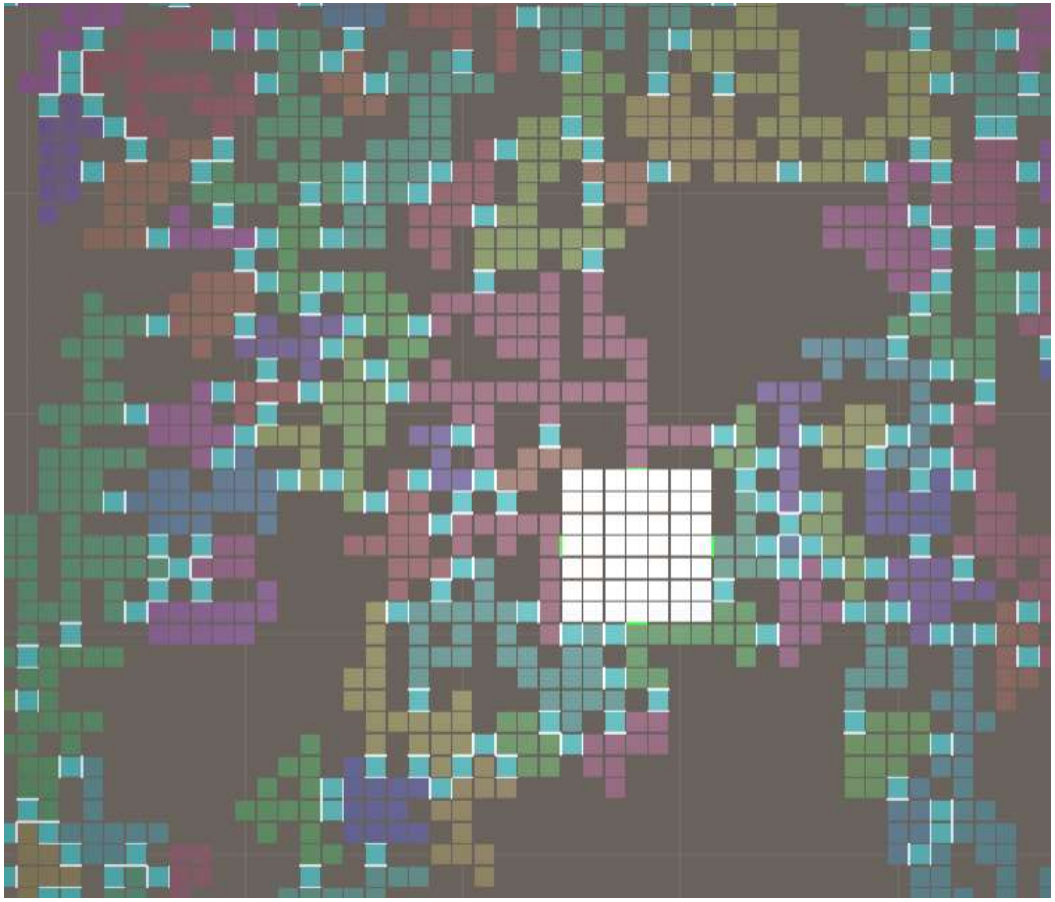


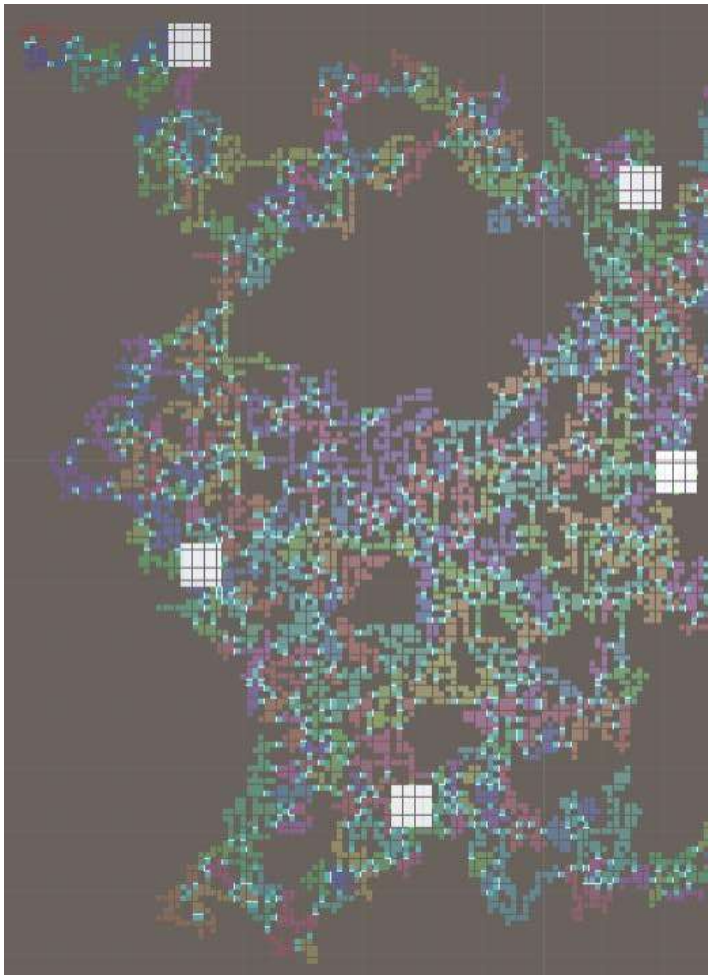
0% chance to sample other dungeon nodes



10% chance to sample other du

Methodology – Trimming Under-Connected





RRT-KPIECE run for 20,000 iterations followed by trimming of Corridors with less than 3 neighb

Results

- Connects all ***N*** Boss Rooms. 
- Does not connect two sides of a Puzzle Room to the same Corridor 
- Player has at least 3 choices of Puzzle Rooms at every location 
- Runtime ~ 1 second per 1000 iterations 

Conclusion

- Novel Dungeon Generator Approach using Motion Planning
  - Control over the connectivity of each room
    - Using EST and KPIECE
  - Control over expansiveness of Dungeon
    - Using RRT
  - Ability to add custom dungeon components
  - Implemented in Unity
    - Ready to add to a game
    - Ready to become a Unity Asset

Thank You!

Questi