Jump Point Search
(10x faster than pure A*)
**Student Postmortem: Escape Velocity**

**[04.20.17]**

— David Rosano III

*Escape Velocity* is a vertical, endless scrolling mobile game made for Android tablets using the Unity engine. The player controls a spaceship trying to avoid obstacles in order to evade capture from an alien ship. The player wins the level by achieving Escape Velocity, the speed at which allows the player to warp to the next level. The player achieves Escape Velocity by filling an energy meter by collecting energy cores, depicted as floating canisters that appear on the screen as the player flies by.

Gameplay Screenshots: *Escape Velocity*: Art by Assistant Lead/Artist and Artist.

The project was created as a part of the curriculum at The Guildhall at Southern Methodist University (SMU Guildhall) in the first of three Team Game Projects. Development on *Escape Velocity* lasted from August-December 2015 for the student team, Crimson Lance Studios. I was a part of the project from the very beginning, serving as both an Assistant Team Lead and an Artist/Animator from pre-production onwards.

For myself and the majority of the team, this was the first game we had ever worked on. This project served as a valuable learning experience for all members of the team.
Reminder about the homework assigned
(Lots of people left before I mentioned it last class)
Jump Point Search

- This is based on a webpage that goes into much more detail. I recommend reading it!
- http://zerowidth.com/2013/05/05/jump-point-search-explained.html
Jump Point Search

Remember how many cells were visited for A*?

It was still nearly all of them... we can do better.
Assume navigation from Green to Red.

Each of these paths are optimal.

If you look at only one, you’re still using the optimal path!
Jump Point Search

- Optimization:
  - For any clear rectangle area, you can assume all nodes are connected to each other.
  - Only the boundaries are important!
Jump Point Search

Let’s apply that optimization here!
Jump Point Search

- To create these rectangles would require preprocessing.
- But we can do one better, we can build them “as necessary”
- And remember, the optimization needs to just include all “wide open” spaces.
Jump Point Search

- For straight movement, you can ignore everything except going straight.

- For diagonal movement, you can go straight or diagonal in the same direction.
Jump Point Search

Complete “rooms” get covered by combining the rules.
All of the above steps can be done without adding to the open set in A*
Jump Point Search

Questions?
If you encounter an obstacle, then you need to add an additional node to visit. This is called a “forced neighbor”.

Jump Point Search
Jump Point Search

- Straight movement has diagonal forced neighbors.
  (45 deg outside normal direction)

- Diagonal movement also has diagonal forced neighbors.
  (45 deg outside normal direction)
If a node has a forced neighbor along its path, you add it to the A* open list.
Jump Point Search

That’s it!
Let’s go through this with Jump Point Search!
Jump Point Search

Interactive demo from the webpage

http://zerowidth.com/2013/05/05/jump-point-search-explained.html