DRIVING AI

Driving AI

- AI world representation
- Path finding
- AI driving
  - Navigation
  - Obstacle avoidance

World Representation

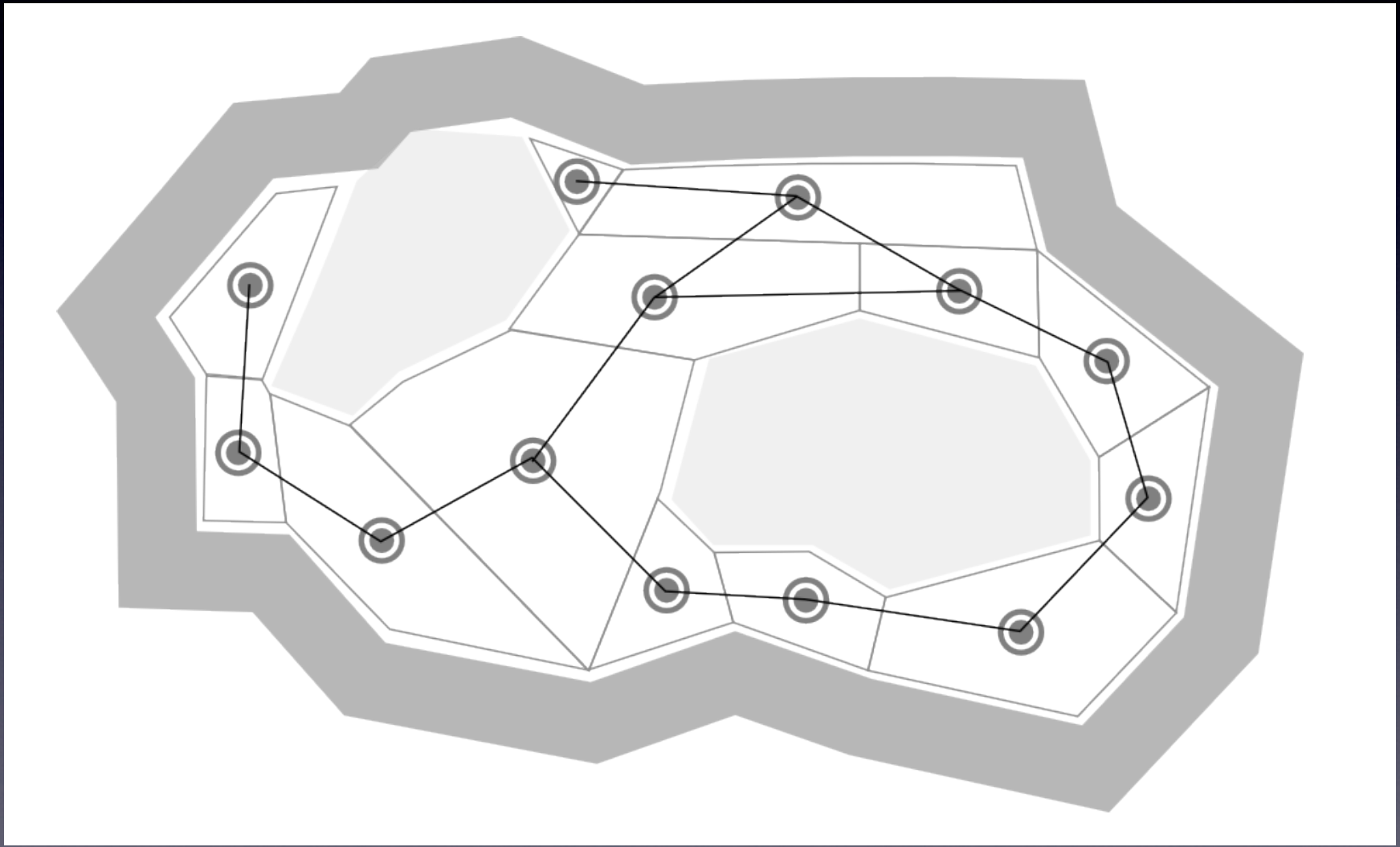
- Need some way to keep track of the world from a driving standpoint
  - Roads, intersections, etc.
  - Other vehicles and dynamic obstacles
- Three levels of operation
  - Navigation
    - I'm here, need to go there on other side of map, how do I get there around the static obstacles in the world?
  - Cruising
    - I'm driving along this road, how do I steer to stay on it?
  - Maneuvering
    - How do I get around this obstacle?
    - How do I get back on the road?

The Road Network in Prototype

The Road Network in Prototype

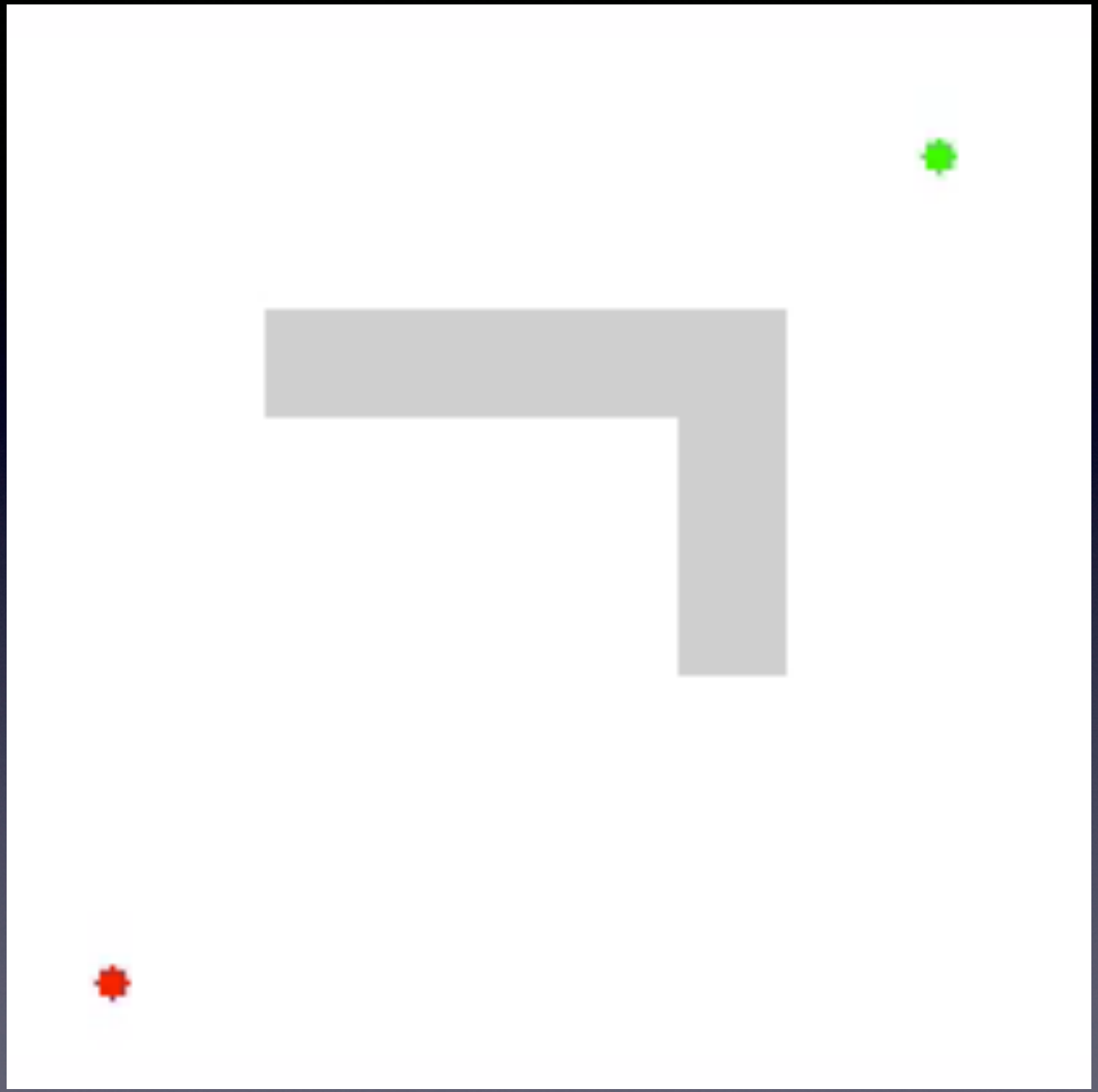
- Navigation mesh
  - A list of polygons covering all navigable areas
  - Polygons marked to be road, sidewalk, intersection, etc
  - Neighbour information in each polygon edge
- Roads are formed by adjacent polygons marked as “road”
- Every road segment has a specified number of lanes
  - Directional
  - Every lane must have a match in adjacent road segments
  - Lanes have a list of vehicles currently travelling along it
    - » Useful for querying if a lane is good to enter
- An intersection is a place where one or more roads meet
  - Knows what roads start and end at its edges
  - Has lanes connecting incoming lanes to outgoing ones
  - Roads (lanes) can only branch at intersections
- Road network can be used for A* pathfinding
  - Intersections as nodes and roads as edges

Navigation Mesh



A* Pathfinding

- The workhorse of path finding in games
- Basic algorithm
  - Two list of nodes “closed set” and “open set”
  - Heuristic for estimating cost from node to target
    - Straight line distance works pretty good
  - Nodes on the “edge” of area you have checked are the open set.
    - Initially only start point in open set.
  - Each iteration, take node with lowest combination of actual measured cost from start and estimated cost to target
    - Record actual path from start node
    - add to closed set
    - add all connected nodes not in closed set to open set
  - Stop when you hit the target



Other techniques

- Variable cost on paths
  - Good way to implement jumps, shortcuts, roadblocks, etc
  - Increase cost for nodes that you want to avoid
    - Shouldn't ever decrease cost, A* requires no overestimation of cost for correct results
  - Vary cost from time to time to implement random behaviour
- Moving through nodes
  - Need to get waypoints inside nodes for actual driving
  - Beeline should always work, but may not look right in "real world" scenarios like road and intersections
  - May want to generate curve of some kind if you have additional annotations for lanes, etc.

Driving AI Considerations

- High level types
  - Traffic : Could be quite different, not even use driving model
  - Opponents: Same basic capabilities as the player
    - Could have same goals and driving techniques, or just same driving techniques but quite different goals
- You only really care about opponents
- Considerations
  - Different states depending on current goals and situation
  - When to path-find
  - How to drive on roads
  - Obstacle avoidance

Opponents

- Opponents have same capabilities as player
- Generally want to use same input mechanism as player does
  - AI should steer a virtual gamepad, not modify things directly
  - Opponents using traffic style cheats will feel strange
- AI entity will generally have few high level states
  - Often based on proximity to player
  - Usually also depends on game mode
- Within states often have state specific goal
  - Often a point to pathfind to

Opponents

- A few (possible) high level strategies
  - Destination
    - Uses navigation graph to generate a path (list of waypoints)
    - Steers for the next point on the path or an interpolation between two adjacent waypoints
  - Intercept
    - Pick intercept point that should catch player (not necessarily point player is now, anticipation is better)
    - Path find same as destination
  - Avoid
    - Player is chasing you and close, want to drive more aggressively, make some random choices, fire weapons.
  - Chase
    - You are chasing player and close, beeline straight for the player, and engage (ram, fire weapons etc)

(Some) Lower level AI Behaviours

- Driving
  - Follow (relatively straight) navigation path
- Cornering
  - Like driving, but may need to brake/e-brake and modify turn parameters
- Passing
  - Get around another vehicle
- Off-road
  - Need to get back on

- Here's a rundown of how we handled some of these problems for Hit & Run
  - Not remotely the only solutions to these problems

Driving

- The path is a list of lane endpoints
  - Calculated from path finding
- “Steer to” points
  - Find closest point on path by checking distance to line segments
  - Extend forward along path by fixed distances
  - H&R used two with different distances (second used for cornering)
- Use the difference between the current facing and the vector to the “steer to” point to generate turning
  - Be mindful of corners (see next slide)
- Floor it
  - AI always uses full gas when just driving
  - H&R used vehicle speed to tune difficulty, could also have speed be fixed and give drivers an “aggressiveness”

Cornering

- Cruising logic doesn't work for corners
  - At speed, turning is hard
  - Tends to overshoot dramatically
- Need to detect when corner is approaching
  - Difference in angle ($\Delta\alpha$) between near and far steer to point
- Decelerate
  - Establish speed limits for various $\Delta\alpha$ ranges
    - Tunable per car and surface
  - If current velocity is above the threshold, slow down
- Change steering
  - Steer to far point instead
- Power-slide
  - If angle gets to big, try to power-slide

Passing

- Don't want to plough into other cars
  - We took very simple approach to this (you can too if you need it)
- Watch for nearby car(s)
  - Can get away with only handling one
  - Often when there are several cars there is no good solution anyway
  - Check if another car is within some volume in front
    - Can use the road segment's list of cars for this
- If you find a possible obstacle
  - If road network allows it
    - Check if the adjacent lane is free and change lanes
  - Otherwise
    - Shift steer to point to side and floor it
    - Once past the car return to the original pathfinding algorithm
  - To sell the effect try honking the horn and flashing the headlights!

Other AI Behaviours to Consider

- Off-road
  - If the navigation info is incomplete, may need different behaviour
- Collecting
  - Powerups, health, etc.
- Stalking
  - With ranged weapons, want to hold back once in range of player
- Shooting
  - Need to consider when to use weapons, if you have them
  - Some options: check distance / orientation, raycasts to target, Trigger volumes moving with vehicle
  - Possibly multiple weighted rules
  - Remember to make them imperfect
- Coordination with others on “team”

Conclusions

- World layout
  - Need some sort of graph for path finding
  - A* is your main navigation tool
- AI
  - High level behaviours (navigation / intercept / etc.)
  - Driving and cornering