Road Generator
Intersections Mockup

Notes

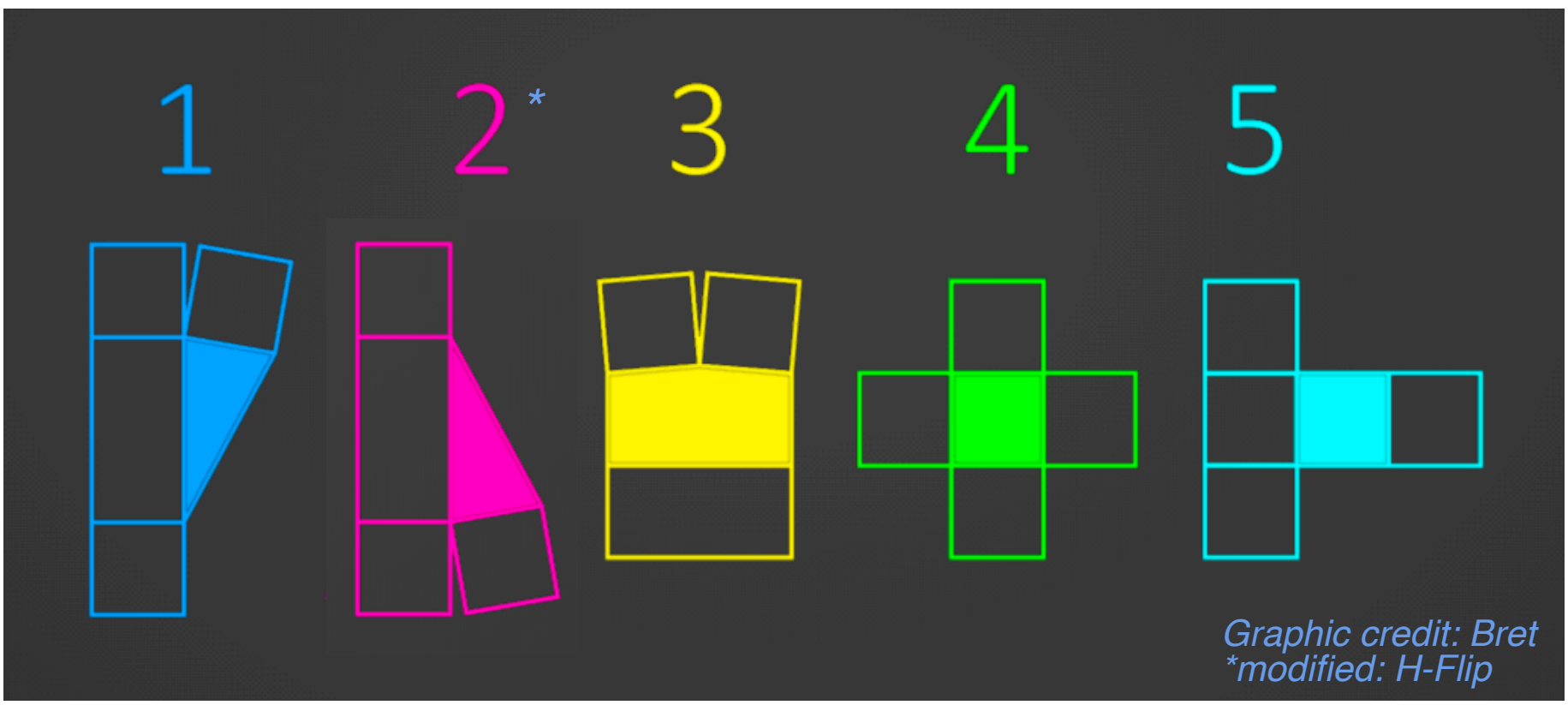
This document is written to plan for the UI and structure of intersections for the addon.

Functionality we need:
- Ability to connect a third RoadPoint to another one, converting it into an intersection
- Be able to insert + connect the next node as a complete intersection
- Ability to have both prefab models and dynamic models, for the most flexibility'
- RoadPoints must be able to seamlessly connect to intersections

Intersection operations
- Convert roadpoint into an intersection
- Drag a roadpoint onto another to dynamically turn into an intersection
- Be able to support prefab or multiple types of intersections

Out of scope
- Sidewalks
- 5-way, 6-way, 7-way... Intersections

Road Generator
Intersection types



At a starting point, we can imagine these kinds of intersections:

1. **Offramp** - An angled connective road piece that attaches to the left or right side of a Segment. It covers the span of the entire Segment and its Start Point is the Segment's Start Point. Its End Point is a separate RoadPoint situated off to the side of the Segment and adjacent to the Segment's Stop Point. Its appearance is somewhat different than a regular Segment in terms of lane markings or lack thereof.
2. **Onramp** - Largely just a flipped orientation of the offramp.
3. **Splitter** - A Segment that contains two side-by-side RoadPoints on one end and one RoadPoint on the other end. The total number of lanes in equals the total number of lanes out. A common use case is splitting a two-way road into two separate one-way roads. Although, there are many possible permutations. It would be good to narrow the scope on what this type will and won't do.
4. **4-Way Stop** - This takes the shape of a plus symbol and positions one RoadPoint on each side. It could, potentially, be a simple square. That really depends on whether we need to draw lines where the cars are supposed to stop. There are no lines in the middle of the Intersection.
5. **T** - This facilitates a regular Segment with a perpendicularly attached Intersection on one side. The user has the option to attach T-Intersections on both sides of a Segment. These Intersections will essentially be like stop signs. Whereas, the main Segment will contain unimpeded traffic flow with traditional lane markings. We could, arbitrarily, say that a white "Stop" line will be painted on any lanes that would be turning onto the main Segment (or not).
6. **Roundabouts (not pictured)** - Similar to T or 4-way, but custom textured and circular roundabout in the middle. Special textured onramps. Lane count limit.
7. **Composite intersections (not pictured)** - Not an actual intersection type, but an acknowledgement that very complex intersections like highway exchanges can be broken down into typically several Splitters + Off/onramps.

Road Generator
Intersection classes & data

High level structure
- So far, RoadSegments are created as the models between RoadPoints, and are greedy-generated by RoadPoints who have connections
- However, Intersections need to be fully responsible for their own roadpoints

Classes structure
- **RoadPoint**: Unchanged, other than skips trying to self generate/create a RoadSegment if the next connected node is an InterPoint.
- **RoadSegment**: Unchanged
- **InterPoint**: New class, does not inherit from RoadPoint. Points to all connected RoadPoints that are "part of the intersection", and then calls upon the InterSegment to generate the geometry. For hand crafted intersections, the scene necessarily contains these edge RoadPoints (already set up by the user).

  Inherits from: 3D Spatial (NOT RoadPoint)

  Export vars:
  > road_points: Array of Nodepaths to RoadPoints, can be increased/decreased
  > intersection_type: Defines the form and nature of the intersection
  > intersection_scene: Resource path? Only used if intersection_type == scene

- **InterSegment**: New class, possibly inherits from RoadSegment but not necessarily. In a way, InterPoint and InterSegment are tied 1:1, unlike RoadPoints/Segments which are more like nodes vs arcs. However, it will still be good to maintain the separation of user input controller (InterPoint) and the geometry delegate (InterSegment). InterSegment contains no data, all fetched via reference to it's InterPoint reference.

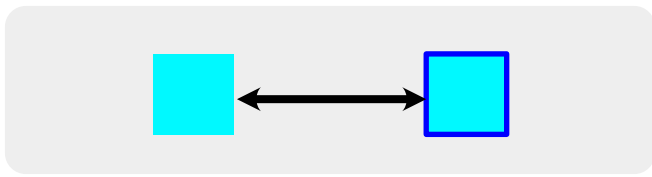
Rules
- Intersections cannot connect to other intersections. There must be a RoadPoint between their endpoints.
- The intersection is responsible for all geometry generation (or loading) for its segments. This is critical to allow for hand-crafted model importing to work.
- Intersections should be treated as the "ends" of roads, which would apply to any bulk or scale operations a user may try to perform (such as bulk roadpoint lane adjustment). This means that someone clicking "next/previous" point in the RoadPoint panel will end on the selection of an intersection (or, maybe the last RoadPoint before).

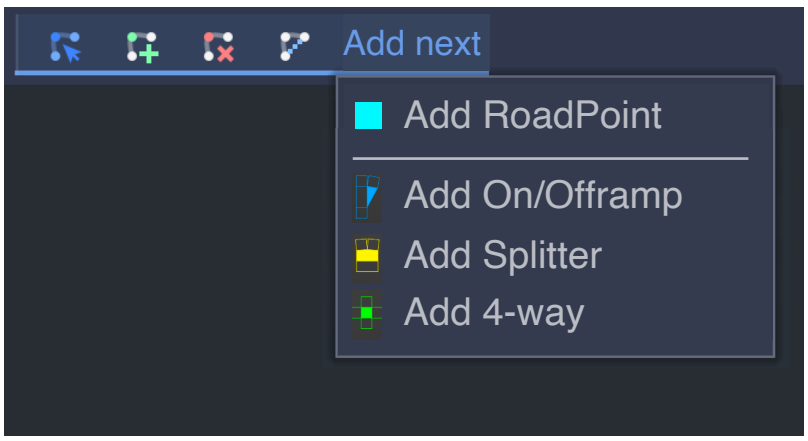
Road Generator
Intersection practical structure

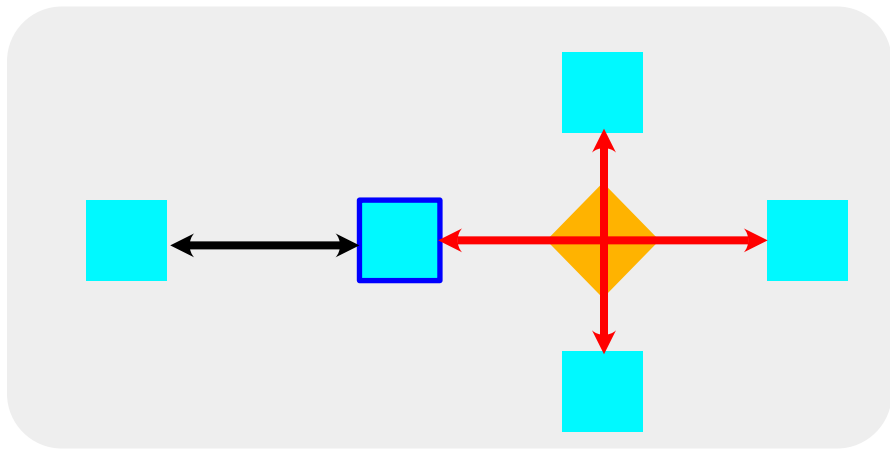
Based on what has been presented so far, that leaves us with a consideration for how one comes to this structure. We can think of the way that a user will in practice create intersections:

Create Intersection by "Next Node"

As a user has been drawing their road, they may get to a point where they want the "next" roadpoint to actually be an intersection. In this scenario, with that RoadPoint select, they could pick from the dropdown "add" menu to decide what to place next.

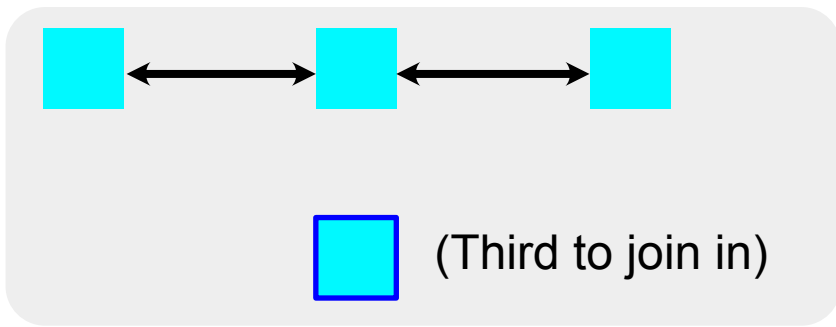






Create Intersection by Joining a third RoadPoint:

Imagine we have the scenario at right, and we (as a user) want to connect the bottom RoadPoint into the middle top one, thus converting the middle top into an intersection. We could do this in different ways:

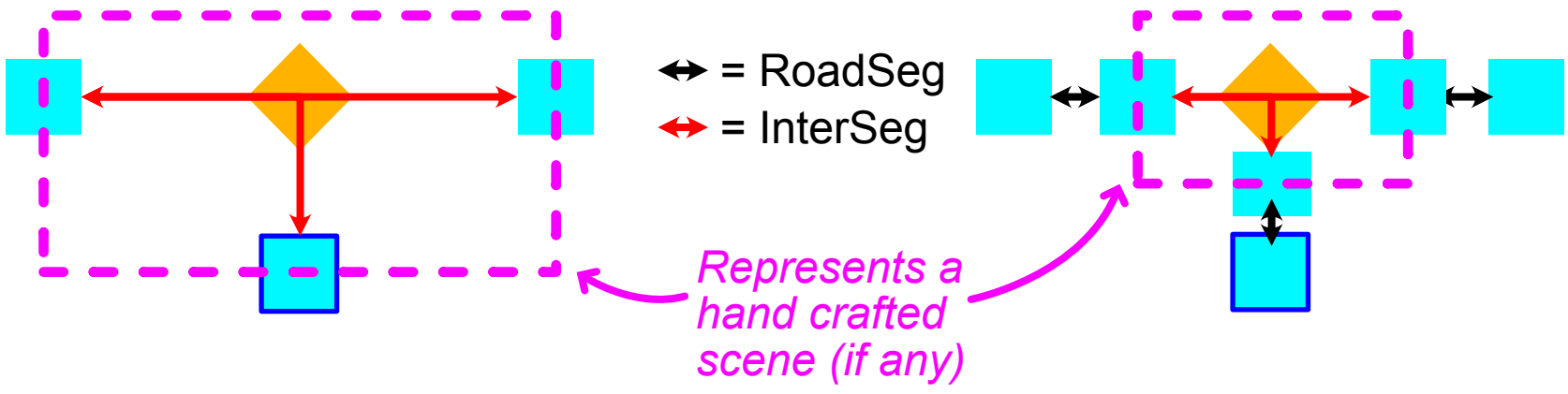


Option A:

Convert the target node, and treat the existing surrounding nodes as the new "edge" roadpoints which directly connect to InterSegs. Will conflict with hand crafted or modifying initial RoadPoints (lanes/size).

Option B:

Bring in an entire intersection with new RoadPoints contained (e.g. from scene), and try to place between existing RoadPoints. More likely to run into "not enough space", but this is possible in both scenarios.

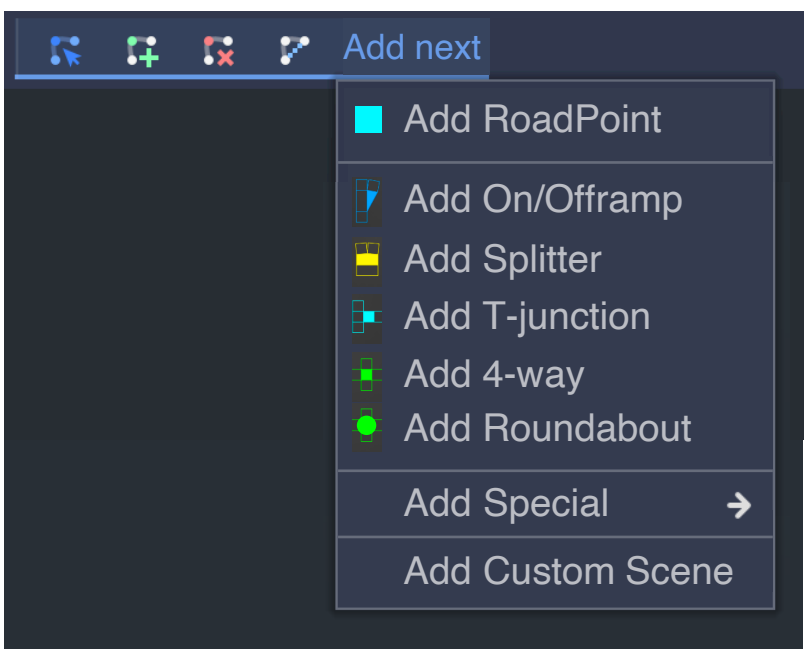


Road Generator
UI mockup

Summarizing how the UI can look for the end user

The 3D viewport panel

Visible if any RoadNetwork, RoadPoint, or InterPoint is selected. Icons are placeholders.

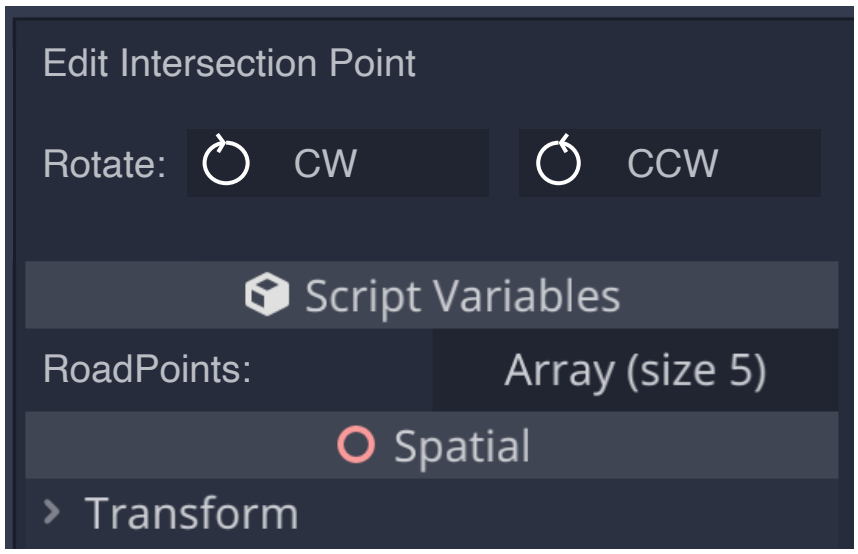


This makes it straightforward and intuitive to add intersections, and gives room for custom usage too (e.g. via the custom scene selection).

The tricky thing with intersection placement is that there isn't a single consistent orientation. If you want to add a splitter, are you making your current road split into two? Or is your current road representing one of the points to join together?

We could consider having "smart selections", whereby if you have multiple roadpoints selected, you can click "Create intersection". But even then, there's still orientation ambiguity.

Thus, it will be important to have an intersection "rotation" feature. See below.



The Intersection Point panel

The Inspector panel displayed when an InterPoint is the active selection.

Early on, it would not need to "do much" other than to have a utility to easily rotate the sequence of connected RoadPoints (so index 1 becomes 2, index 3 becomes 0 in a 4-way scenario).

RoadPoint connections should already be ordered in a clockwise fashion, starting from the top-closest at or after "12 'O' clock". In the Y-oriented splitter, index 0 is the top right branch, 1 is the bottom (joined part), and 2 is the top left split branch.

On pressing CW (clockwise), an RP initially connected to Index 1 is then connected to index 0. This looks cleaner if only one point is connected (rotate after just adding InterPoint).