Simple Railway Management System

Import the SimpleRailwaySystem package into your Unity3D project (version >= 2019.3.6f1)

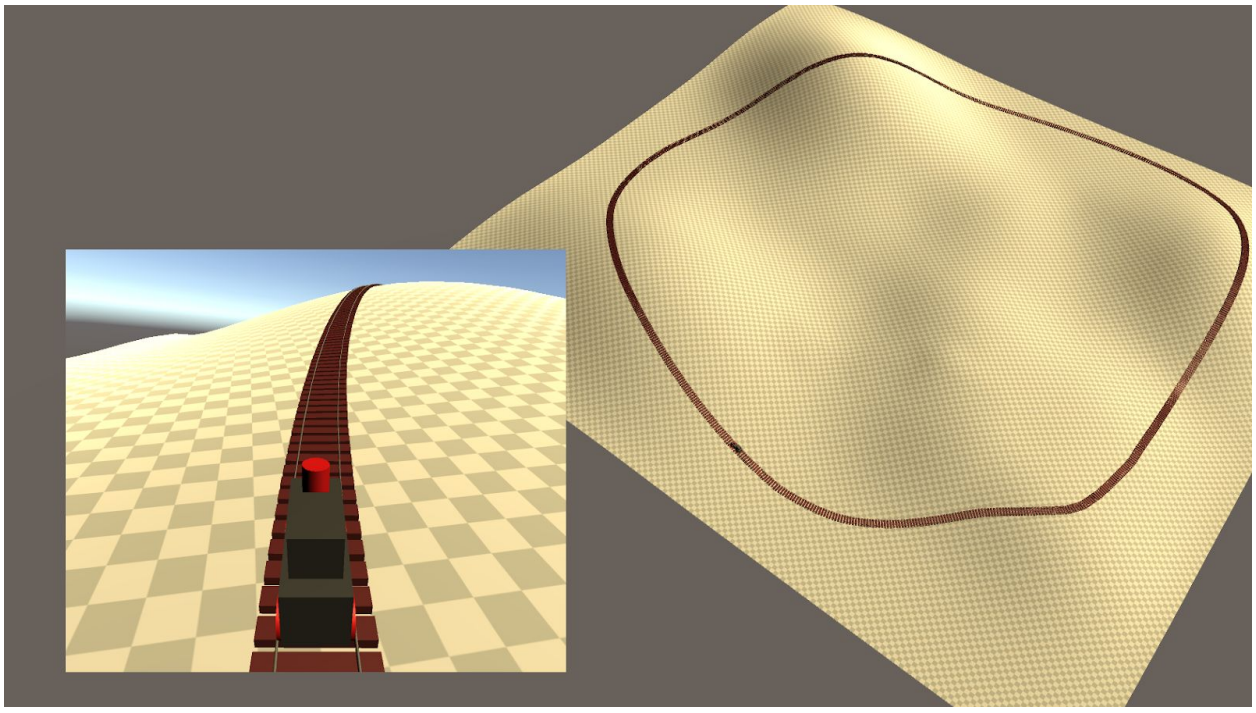
The “Simple Railway Management System” is aimed at:

- Generating a railway 3D mesh (rails & sleepers) from a spline-based path and a projective terrain
- Providing necessary tools for external scripts to steer objects on the railway

The system is based on three C# scripts:

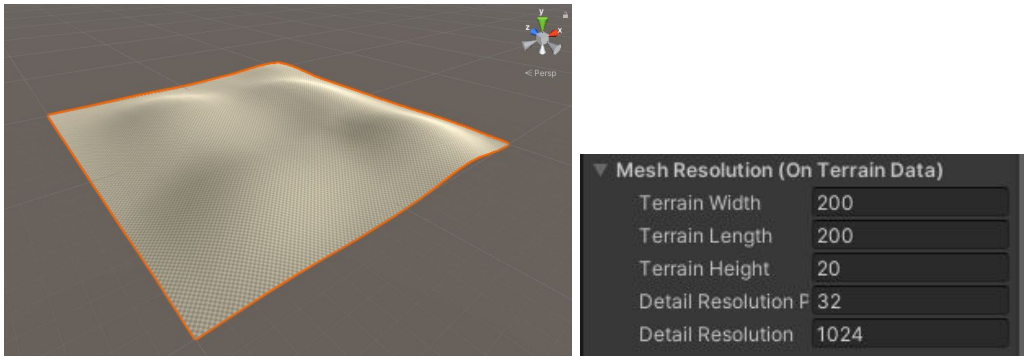
- *RailManager.cs*: rail mesh generation and navigation on rail
- *MySpline.cs*: spline system based on cubic Bezier curves
  Wikipedia article about Bezier Curves: https://en.wikipedia.org/wiki/B%C3%A9zier_curve
- *GeometryTools.cs* : some 2D and 3D intersection methods

The example scene found in Assets\SimpleRailwaySystem\ExampleScene**SimpleRailwaySystem** illustrates a basic railway loop with a schematic train driving the rails.



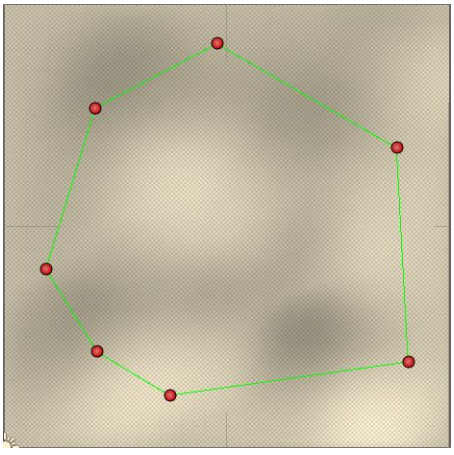
Steps to create a Simple Railway

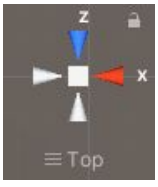
1. Open a new scene
2. Create a terrain and set its layer to "*Terrain*" (.... or whatever name you'll consider relevant). Favour a low altitude relief, as shown below:



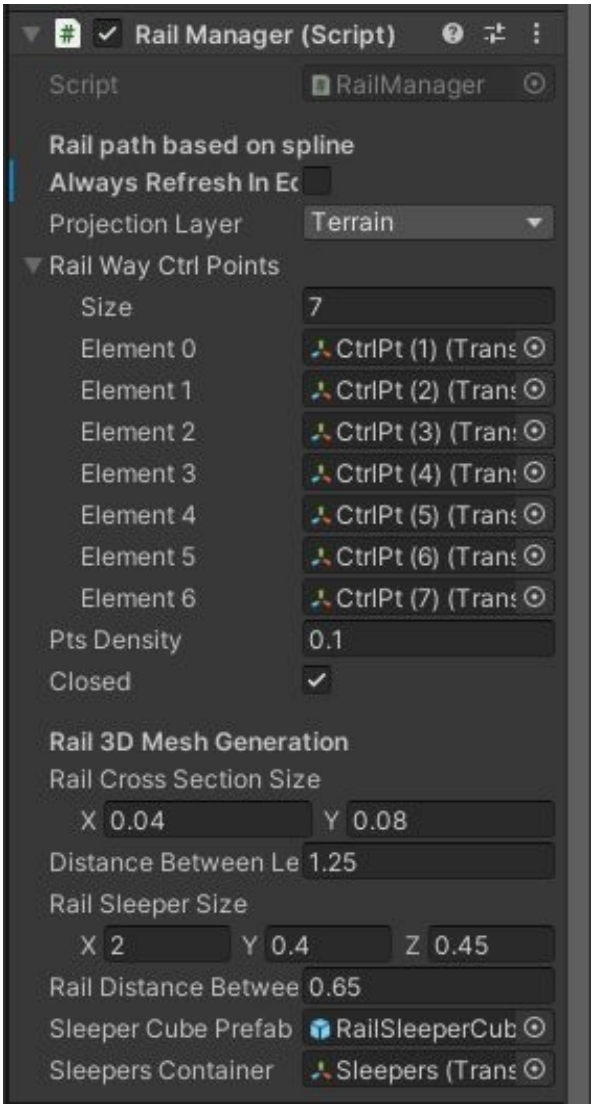
3. Drag the *RailManager* prefab onto the scene and position it at the origin (0,0,0)
    - Some control points have already been defined in the *RailCtrlPts* child object. These points define a rough railway path.
    - Uncheck the "3D Icons" Gizmos' attribute in order to see the red spheres highlighting the control points. Green lines connect the control points.

    - I recommend you set the Scene view to the top isometric view. Doing so, you'll get a precise view on the rail extensions.



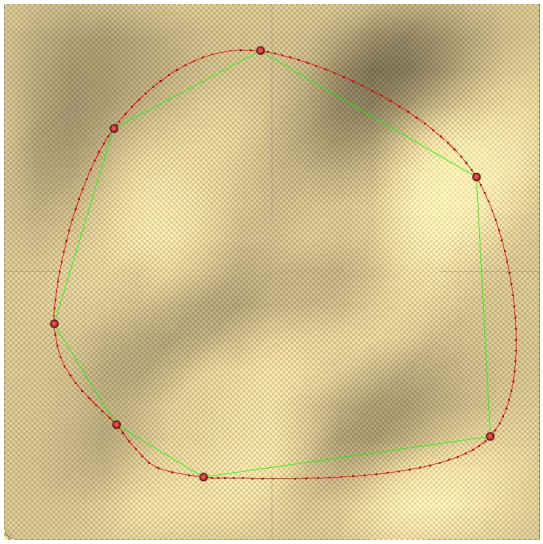


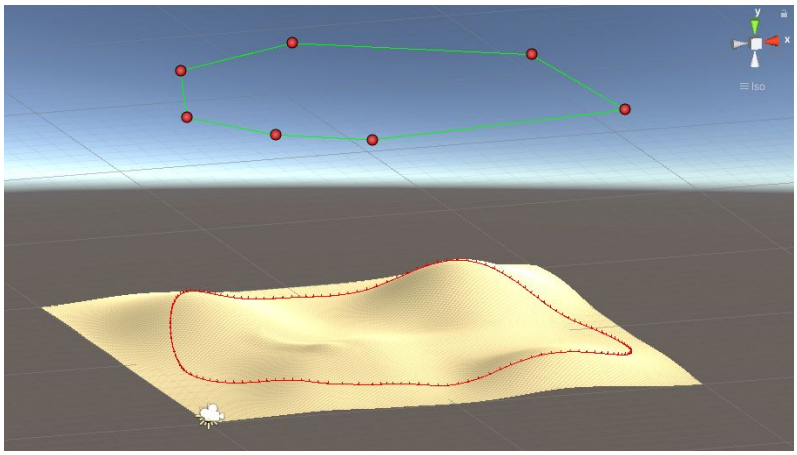
- If you want to expand the default path, add points by duplicating (Ctrl+d) existing ones, and move them around
- Do not forget to reference these control points, in the right order, into the *Rail Way Ctrl Points* list found in the *RailManager* component.



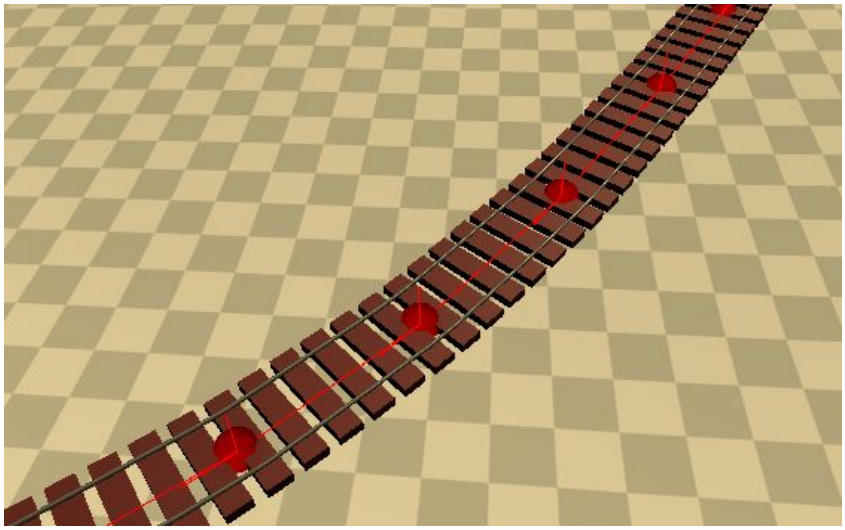
- Set the *Projection Layer* attribute to your terrain's layer.

4. With *RailManager* GameObject selected in the hierarchy, you should be able to see, drawn red, the waypoints of a more refined railway path projected onto the terrain. This path has been generated thanks to a mathematical spline based on the control points previously created.





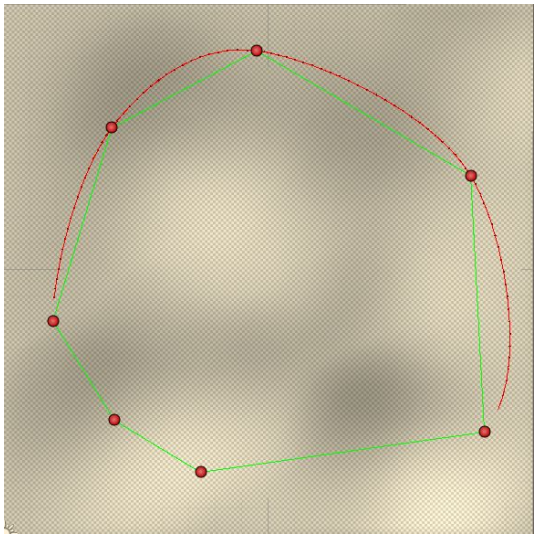
5. Press Play. A 3D railway mesh, made of rails and sleepers, is created along the path at runtime.



Open or closed Railroad track

If you check the “Closed” attribute of the *RailManager* component, the railroad track passes through all control points, and you must define at least 3 control points for the system to be able to create the railway.

If you uncheck the “Closed” attribute of the *RailManager* component, the first and last control points define the start and final tangents to the railroad track, and therefore the track goes from the second to the penultimate control points. You must define in this case at least 4 control points for the system to be able to create the railway.



Code interface

Public methods of the RailManager component will help steering objects along the rail.

public float Length { get ;}

- Returns the total length of the rail

public bool GetPositionNormalTangent(float t, out Vector3 pos, out Vector3 normal, out Vector3 tangent, out int segmentIndex)

- For a relative (between 0 and 100%) position *t* on the rail, gets the world position, normal and tangent, as well as the index of the corresponding rail segment. Returns true if successful.

public Vector3 this[float t]

- Accessor that retrieves, for a relative (between 0 and 100%) position *t* on the rail, the world position.

public float DistanceBetweenTerrainAndTopOfSleeper { get;}

- Returns the distance between the terrain and the top of the sleeper. Useful for setting precisely the position of the steered object above the rails.