CarAI-Unity

A C# script that uses NavMesh for path finding and simulates vehicle movement using unity's physics components

Table of content:

Requirements:

1. Unity 2018 or higher
2. NavMeshComponents (This package is different from the built-in NavMesh)

Package contents:

1. CarAI.cs: This is the main script. It's responsible for generating a path using NavMesh and simulating vehicle movement using unity's physics engine

2. controllingCarAI.cs: this script has examples of controlling the AI through code

3. CarAISensor.cs: This script is an example of controlling the AI through Code (It is also useful for creating Traffic system).

Setup The Scene:

1- Bake NavMesh Surface:

Note: It's highly recommended to use the NavMeshComponents package from GitHub rather than the built-in package.

After importing NavMeshComponents into your project, Create an Empty GameObject then "add component" and choose NavMeshSurface.
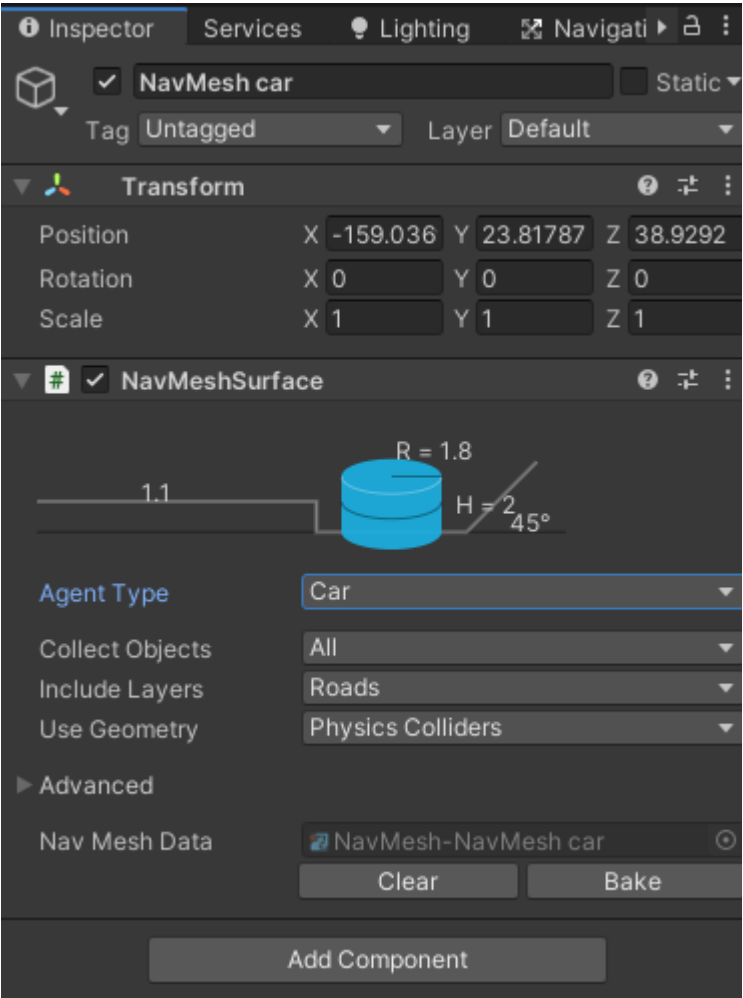
Then use these settings:

1- Agent Type: Select the Agent type (Or create a new agent with the following settings)
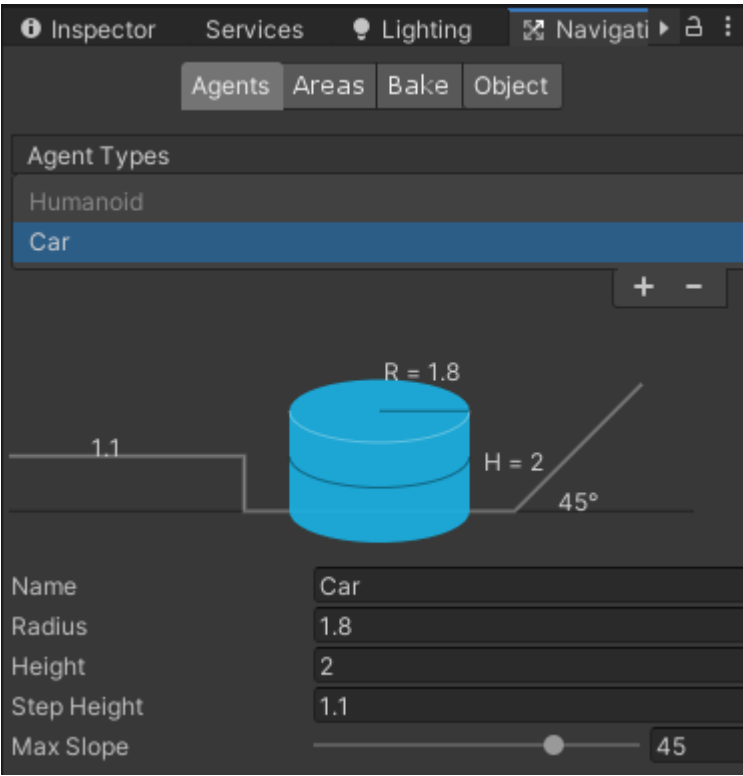
> Name: "Car"
> Radius: 1.8
> Height: 2
> Step Height: 1.1
> Max Slope: 45
>
> **Note**: These values may differ from one vehicle to another.

2- Collect Objects: All

3- Include Layers: Choose the layer you want the agent to move in. (Create a layer and assign all the roads to it)

4- Use Geometry: I prefer "Render Meshes" but you can use "Physics colliders" as well

Then click on "Bake" and wait for it to finish

After The NavMesh is Backed, A color (purple color) will appear on the street which represents the area where the AI can move (This means that the AI's pivot won't leave this area

Note: As you see in the image above. The car's center point(pivot) won't leave the purple area(NavMeshSurface). But, if the whole street is purple. The car might move on the sidewalk. To prevent this use NavMesh Modifier to Override all the sidewalks as a Non-Walkable area. Then bake the surface again.

2- Car Model Components:

For the CarAI package to work the car model must have these components:
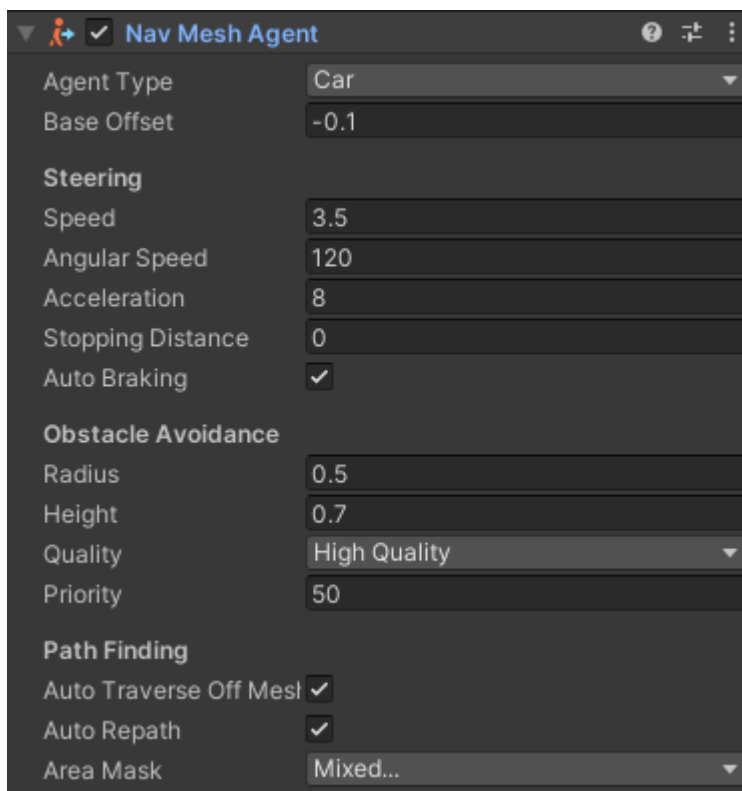
1- Rigidbody:
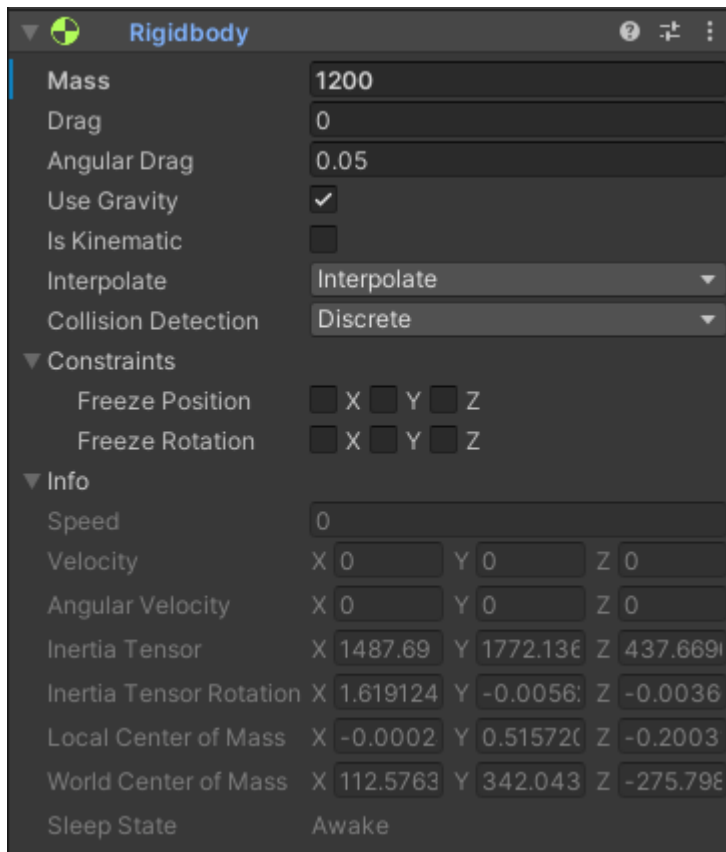
> 1- Mass: Set the value from 1000 to 1500 (It depends on the vehicle. Ex:A Truck will be 1500)
>
> And you can leave everything else as it is

2- NavMeshAgent:

> 1- Agent Type: "Car" (the same agent type that is used in backing the NavMeshSurface)

2- Steering settings won't affect anything since the Agent is used for generating the path only.

3- Obstacle Avoidance similar to Steering Settings it won't affect anything

4- Auto Transverse Off mesh & Auto Repath: Set both of them to true

5- Area Mask: choose the NavMesh Layers you want the Car to move in (the same Layer that's used in Backing the NavMeshSurface)

3- WheelColliders:

Note: it's highly recommended to watch this tutorial on wheel colliders

Select all the wheels then "Add component" and choose "Wheel Collider"

Then apply these values:

Wheel Collider	
Mass	80
Radius	0.35
Wheel Damping Rate	1
Suspension Distance	0.01
Force App Point Distan	0
Center	X 0 Y 0 Z 0
Suspension Spring	
Spring	5
Damper	100
Target Position	0.5
Forward Friction	
Extremum Slip	0.4
Extremum Value	1
Asymptote Slip	0.8
Asymptote Value	0.5
Stiffness	20
Sideways Friction	
Extremum Slip	0.2
Extremum Value	1
Asymptote Slip	0.5
Asymptote Value	0.75
Stiffness	10

**Note**: These values may differ according to the Vehicle type. So, adjust it carefully.

1- Mass: From 80 to 100

2- Radius: Depends on the car model (wheel radius)

2- Wheel Damping Rate: From 0.01 to 1 (Higher value means slower rotation)

3- Spring: 5

4- Damper: 100

5- Target Position: 0.5

3- Forward Friction Stiffness: From 15 to 20

4- Sideways Friction Stiffness: From 7.5 to 10

CarAI Package

1- Setup CarAI.cs:

You need to attach "CarAI.cs" Script to the car. Then Follow these steps:
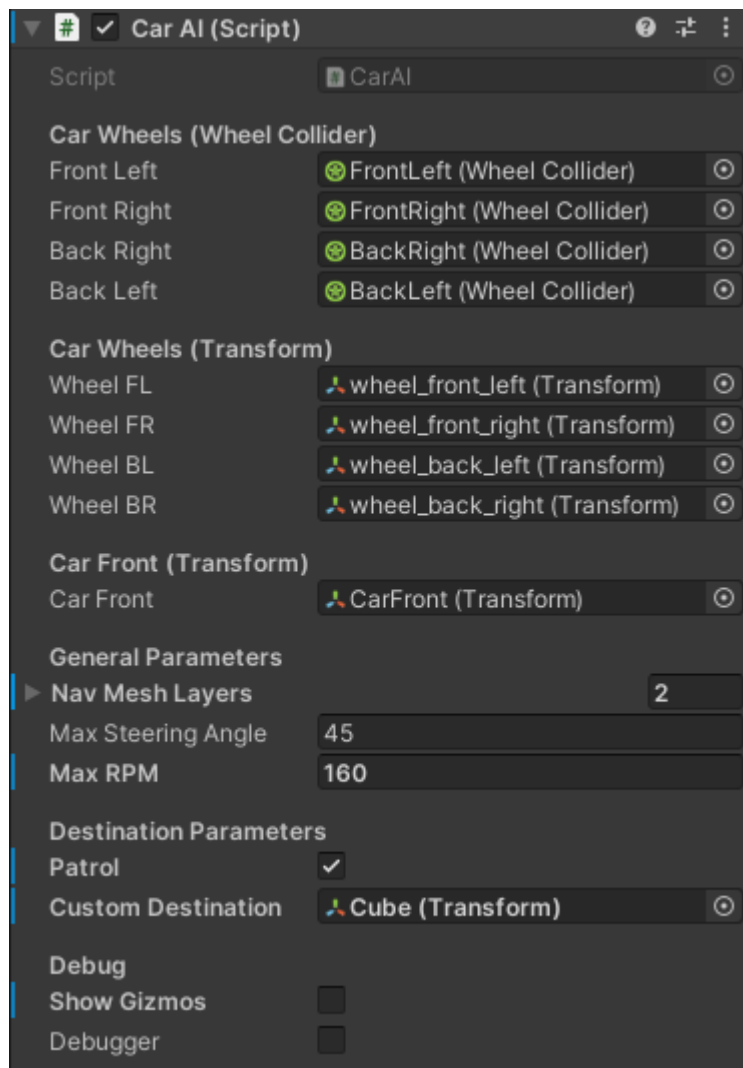
1- Car Wheels (Wheel collider): Assign each wheel collider to its corresponding location

2- Car Wheels (Transform): Assign each Wheel model to its corresponding object

3- Car Front: Create a new empty GameObject as a child of the vehicle then place this object at the front of the vehicle and assign this Gameobject to `Car Front` Field

Note: the following tables are very important. So, read them carefully and I will explain every point in details

4- General Parameters:

1- Nav Mesh Layers: Enter the name of the layers you want the AI to move in

2- Max Steering Angle: The Maximum steering angle for the vehicle

3- Max RPM: You can use it to control the speed of the vehicle the lower the value the slower the car

5- Destination Parameters:

1- Patrol: This is useful for traffic systems as the AI will keep generating a random destination after it reaches the active one.

2- Custom Destination: If a transform is assigned then the AI will keep generating a path to that destination

Note: if both `Patrol` is True and `Custom Destination` is assigned, then the AI will keep generating a path to the custom destination. (In Other words, `Custom Destination` has priority over `Patrol`)

6- Debug:

1- Show Gizmos: Show the FOV of the AI and wire spheres representing the waypoints of the current path

What is the FOV of the AI?

In unity, the normal AI can turn around, and this behavior can't be applied to cars because they have steering angles. So, I created a field of view for the AI to prevent it from turning more than `Max Steering Angle` Away from his forward position.

2- Debugger: Display debugging info in the console

Programming

Here are some examples of controlling the AI with scripts

You will find these examples in a script called "controllingCarAI.cs" in the package files

1- Setting Variables through code:

```
//1- Patrol = true
carAI.Patrol = true;
// Patrol = false
carAI.Patrol = false;

//2- Set the maximum steering angle
carAI.MaxSteeringAngle = 45;

//3- Set the maximum RPM(Speed)
carAI.MaxRPM = 150;

//4- Set the NavMesh Layer name
carAI.NavMeshLayers[index] = "LayerName";

//5- Show Gizmos
carAI.ShowGizmos = true;
//or hide Gizmos
carAI.ShowGizmos = false;

//6- Allow thr car to move
carAI.move = true;
//or apply brakes
carAI.move = false;
```

2- Calling methods through code:

```
//1- Generates a path to a new random Destination
carAI.RandomPath();

//2- Generates a path to a new custom Destination
carAI.CustomPath(gameObject.transform);
```
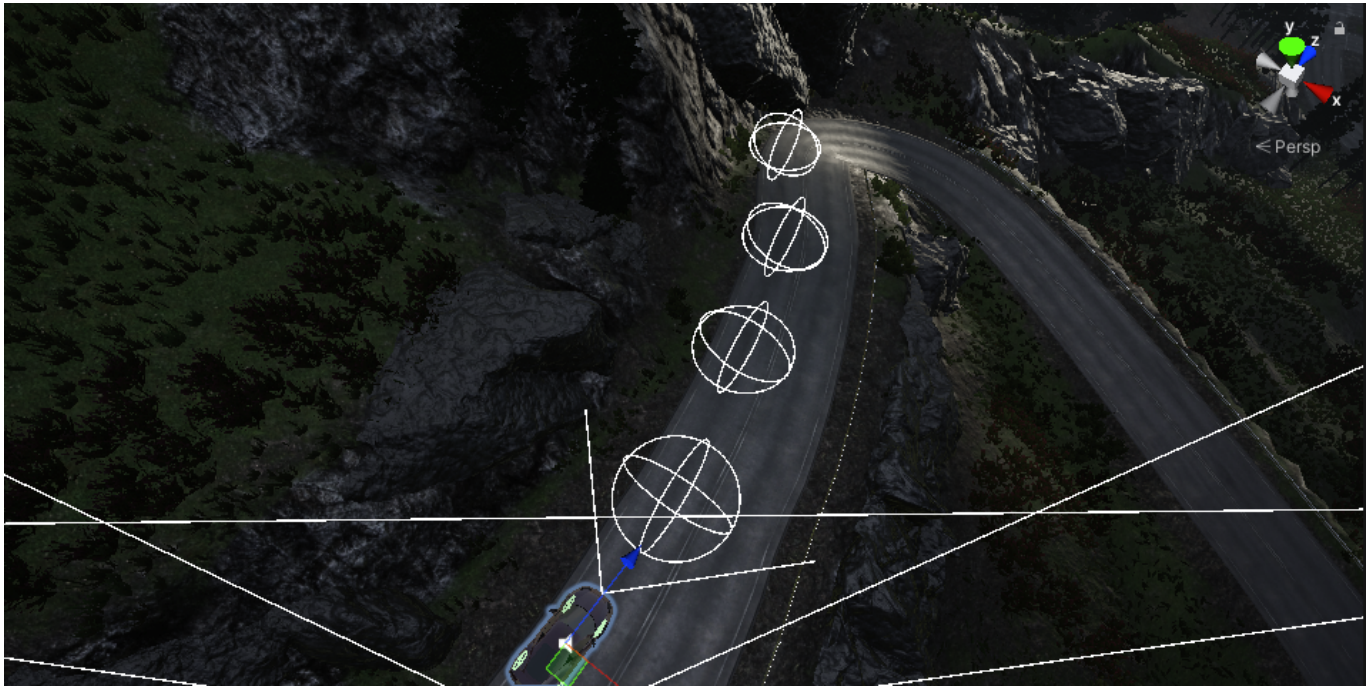
Common Issues And How To Fix Them

1- AI can't create the Path

Case 1: The car is not in the NavMesh area

**Fix:** Never place the car outside the NavMesh area

Case 2: There are no possible paths available

As you see in the image above the AI was able to create a random destination. But, The problem here is the last waypoint

To understand the problem you need to understand how the AI generates a path:

After the AI generates a random destination it checks if the first waypoint is located in the FOV or not. (If not, then the AI will try to generate another path). So, When The AI reaches the final waypoint it will be facing that giant rock. (Like in the image below)
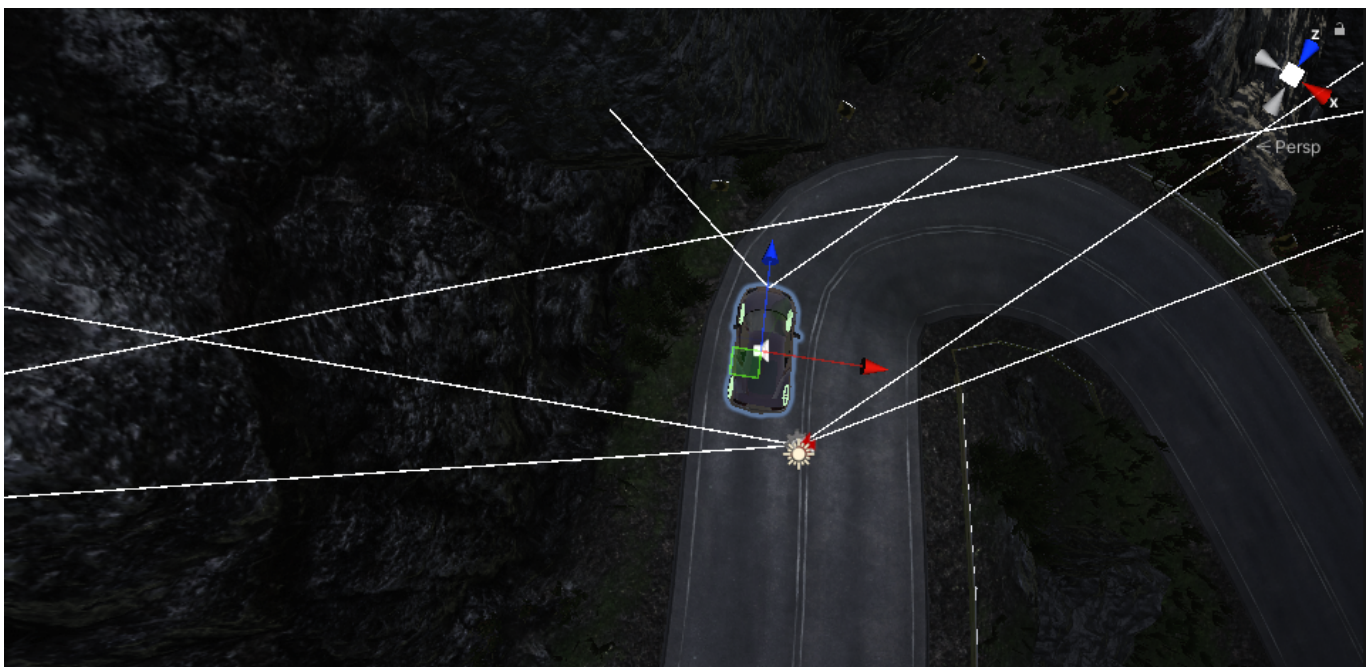
When it tries to generate a path. it will be stuck in a loop of generating a path and refusing it (because it's not in the FOV)

Fix:

(The Best Solution) 1- Change the area type of roads that have sharp turns using the NavMesh Modifier component (And don't forget to add the new area type to the Area Mask in the NavMeshAgent Component).

So, The AI can move in this area but can't create destinations in it.

2- Modify the behavior of the AI to move backward if it gets stuck for some time

3- By increasing the FOV: But this is not a great idea because increasing the FOV without increasing the MaxSteering Angle will make the car move in circles around the waypoint.

> Anyway if you want to increase the FOV. Open CarAI.cs at line 40 you can change the default value of AIFOV.

The same problem happens when the car reaches a dead end.
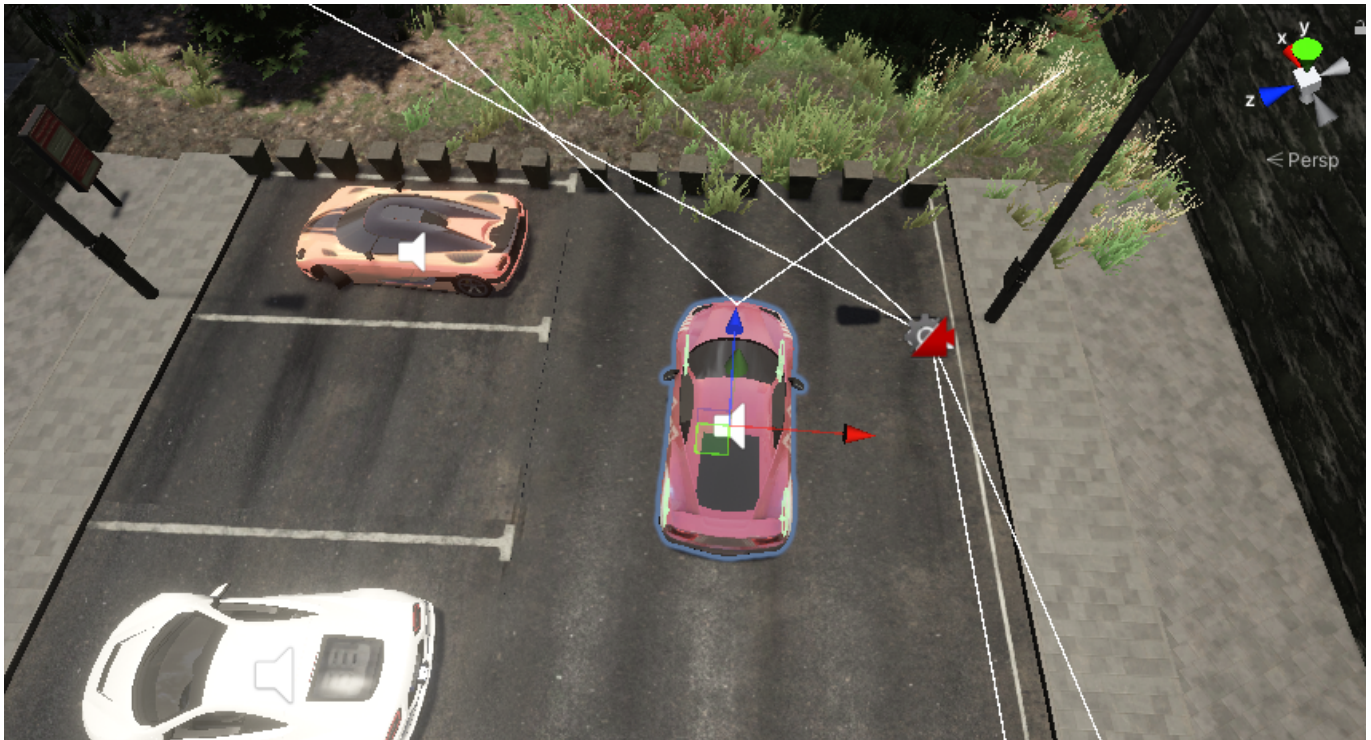
Fix: You can fix this by excluding all Dead-Ends from the NavMeshSurface (Using NavMesh Modifier). So, the AI won't go there