gley

MOBILE TRAFFIC SYSTEM

1. WHY DO YOU NEED TO USE THIS PLUGIN

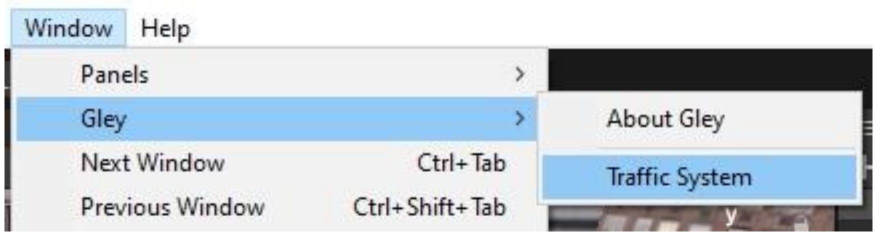
- **High performance** - 300 cars on a 2018 Android device (Test if yourself).
- **No coding skills required.**
- Traffic lights intersections support.
- Priority intersections support - cars decide by themselves to wait or to enter an intersection.
- **Roundabout support** - cars wait until the roundabout is free.
- **Narrow road support** - cars wait until the lane is free before changing it.
- **Overtake** - cars automatically overtake each other if the road has multiple lanes.
- Building avoidance - if a car hits a building, it will try to recover itself.
- Customizable car properties - acceleration, max speed, brake speed, steer angle etc.
- **Variable number of wheels** - from 3 to as many as you want.
- Automatically car assignments - made just by pressing a button.
- **Speed routes** - set speed restrictions for some lanes or areas and all cars will follow them.
- **Car types** - assign different types to cars and you can restrict access on some roads based on car types (ex: trucks are allowed only on the first lane and are not allowed inside the cities).
- **Lights support** - main lights, brake lights, blinkers, reverse lights that automatically work.
- Sound support - basic engine sound for each car based on acceleration and speed.
- Hard shadow support - useful is you need even more performance.
- **Custom editor tools** - a lot of editor windows to make the integration process as smooth as possible.
- Simple API for advanced functionalities.

2. TECHNICAL FEATURES

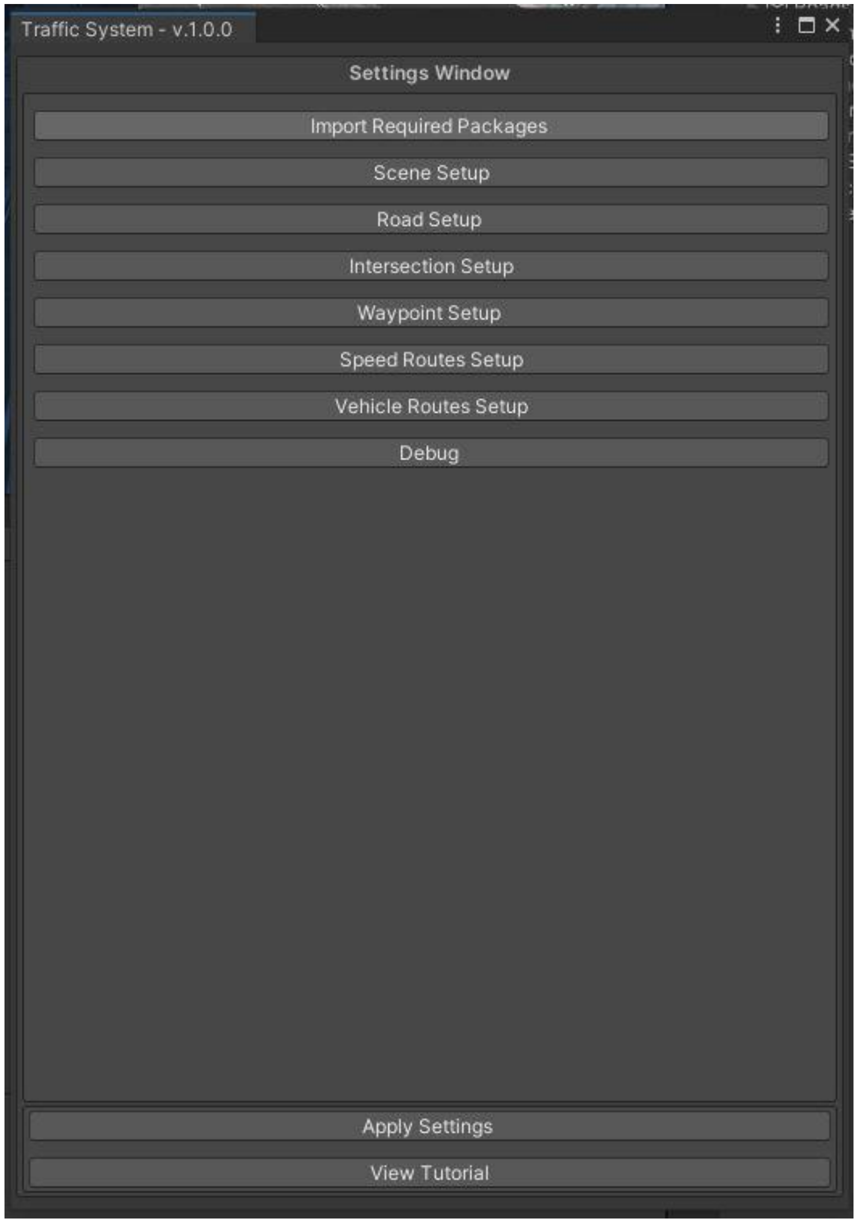
- **Parallel job system** - for better performance.
- **Burst compiler** - to speed up the computations.
- **No wheel colliders** - raycasts are used for better performance and high accuracy.
- **Pooling system** - spawns the cars around the player so you can create high density traffic with low number of used cars. Very scalable for huge environments.
- Layer management - for better optimization.
- Waypoints driving - it comes with an easy to use Waypoint editor, so waypoint generation is semi-automatically.
- Full editor script that automatically prepares your car.
- **Full code is provided** and fully commented, no dlls.
- Works with any version of Unity above 2019.1.0 with any Unity licence available.
- **Works with any supported Unity platforms.**

3. SETUP GUIDE

- Import **Mobile Traffic System** asset into Unity.
- Go to **Window->Gley->Traffic System** to open the Settings Window.



- Settings Window will open



3. SETUP GUIDE

Setup details are shown in the following youtube playlist:

https://youtube.com/playlist?list=PLKeb94eicHQtyL7nYgZ4De1htLs8lmz9C

Here is the detailed tutorial list:

1. Install and test
2. Prepare your scene
3. Create basic road
4. Test traffic
5. Connect roads
6. Priority intersection
7. Traffic lights intersection
8. Narrow roads
9. Roundabout setup
10. Highway exit setup
11. Speed routes
12. Vehicle routes setup
13. Waypoint setup window
14. Car Implementation
15. Traffic component and car pools
16. Scripting methods
17. Debug window

4. SCRIPTING GUIDE

- **Initialize the traffic**

// Transform activeCamera -> camera that follows the playe
// int nrOfVehicles -> max number of traffic vehicles active in the same time
// VehiclePool carPool -> available vehicles asset
// float minDistanceToAdd -> min distance from the player to add new vehicle
// float distanceToRemove ->distance at which traffic vehicles can be removed
// float masterVolume -> [-1,1] used to control the engine sound from your master volume
// float greenLightTime -> roads green light duration in seconds
// float yelloLightTime -> roads yellow light duration in seconds
GleyTrafficSystem.Manager.Initialize(activeCamera, nrOfVehicles, carPool,
minDistanceToAdd, distanceToRemove, masterVolume, greenLightTime, yelloLightTime);

- **Modify max number of active vehicles**

// int nrOfVehicles ->new max number of vehicles, needs to be less than the initialization max number of vehicles.
GleyTrafficSystem.Manager.SetTrafficDensity(nrOfVehicles);

4. SCRIPTING GUIDE

- **Update active camera that is used to remove vehicles outside of view**

```
// Transform activeCamera -> represents the camera or the player prefab
GleyTrafficSystem.Manager.SetCamera(activeCamera);
```

- **Remove all vehicles in a range**

```
// Vector3 center -> center position for removing the vehicles
// float radius -> radius of the circle
GleyTrafficSystem.Manager.ClearTrafficOnArea(center, radius);
```

- **Disable all waypoints in the specified area to stop vehicles to go in a certain area**

```
// Vector3 center -> center position for disabling the waypoints
// float radius -> radius of the circle
 GleyTrafficSystem.Manager.DisableAreaWaypoints(center, radius);
```

4. SCRIPTING GUIDE

- **Enable all disabled area waypoints**

GleyTrafficSystem.Manager. EnableAllWaypoints();

- **Turn all vehicle lights on or off**

```
//bool on -> if true, lights are on
GleyTrafficSystem.Manager.UpdateVehicleLights(on);
```

- **Control the engine volume from your master volume**

```
 //float volume -> current engine AudioSource volume
GleyTrafficSystem.Manager.SetEngineVolume(volume);
```

4. SCRIPTING GUIDE

- **Spawn vehicles mostly in front of the player for a greater vehicle density**

```
GleyTrafficSystem.Manager.SetSpawnWaypointSelectorDelegate(GetBestNeighbor.GetForwardSpawnWaypoint);
```

- **Remove a vehicle from traffic system control**

```
// When this method is called, the vehicle passed as param is no longer controlled by the traffic system
// until it is out of view and respawned

// GameObject vehicleGameObject -> The root gameobject of the vehicle
 GleyTrafficSystem.Manager.StopVehicleDriving(vehicleGameObject);
```

- **Set an intersection road to green**

```
// Force a road from an intersection to change to green

// string intersectionName -> name of the intersection to change
// int roadIndex -> the road index to change
// bool doNotChangeAgain -> if true that road will stay green until this param is set back to false
GleyTrafficSystem.Manager.SetIntersectionRoadToGreen(intersectionName, roadIndex, doNotChangeAgain);
```

- **Remove a vehicle from scene**

```
// When this method is called, the vehicle passed as param is removed from scene and respawned

// GameObject vehicleGameObject -> The root gameobject of the vehicle
 GleyTrafficSystem.Manager.RemoveVehicle(vehicleGameObject);
```

- **Add vehicle in point**

```
// Add a traffic vehicle to the closest waypoint from the given position
// This method will wait until that vehicle type will be available and the closest waypoint will be free to add a
// new vehicle on it.

// Vector3 position -> the position where to add a new vehicle
// VehicleTypes vehicleType -> the type of vehicle to add
 GleyTrafficSystem.Manager.AddVehicle(position, vehicleType);
```

5. EXAMPLE

You can find the example test scene here:
Assets/GleyPlugins/TrafficSystem/Example/City.unity

A video tutorial on how to test the scene is available here:
https://youtu.be/hjKXg6HtWPI

Version 1.0.8 / 2021