Unrealistic Arts

Draw and Drive

User Guide

Version: 1.6.0

Date: 17 March 2022

Table of Contents

List of Tables

List of Figures

Introduction

This system will allow you to create nice and functional game and do it really easy. You can easily touch screen and draw path in ground (or using Mouse button), then the object(s) can find they path and go ahead smoothly. In additional you can manage many node(s) as car or whatever in asynchronous or synchronous mode. It`s powerful to manage as many as nodes you want. Just enjoy it

Scripts list

There you have all scripts to work.

Table 1 - Main scripts

Main Scripts		
CarObject	Ground	
Other Scripts		
Extensions	Globals	EditorTools
TextWindow	CarObjectSituation	InputManager
PathNode		
Editors		
CarObjectEditor	GroundEditor	MainMenu

Note:

You don`t have to know each scripts or programming. This list is just for awareness.

How to use menu?

Here the menu items explained quickly but in Scripts title its explained with more details.

Ground

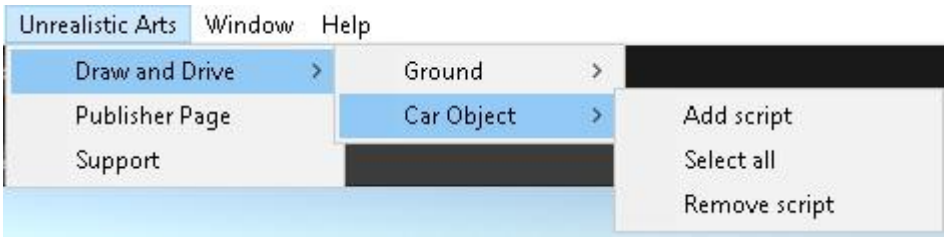
In “Ground” item you can add ground(s), attach ground script to your selection game object(s), or select all game objects with that component.

Figure 1 - Ground menu

Car Object

In “Car object” item you can attach car script to your vehicle or select them.

Figure 2 - Car Object menu



Basic Scripts and their parameters

Ground.cs

It`s the main script to handle your canvas/ground to draw a path. you can also have grounds as many as you want.

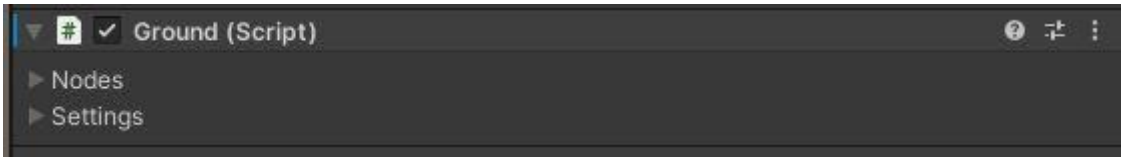
Path: Draw and Drive\Scripts\Ground.cs

Here we have some general options. Each section has its own variables.

Table 2 - Ground (General sections)

Name	Description
Nodes	Nodes contains car and any playable object
Settings	Game settings

Figure 3 - Ground inspector (General sections)



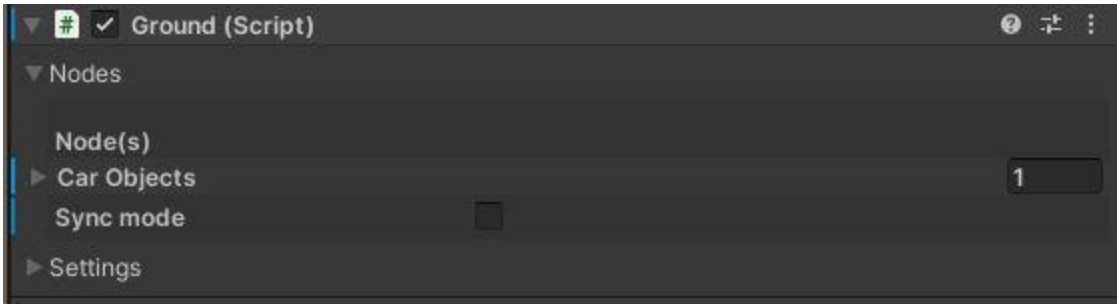
Nodes

Here we have Nodes variables.

Table 3 - Ground (Nodes)

Name	Description
Car Objects	List of all car object(s) or node(s)
Sync Mode	Checking this item make the cars synchronize, all node(s)/car(s) drive at the same time (after all paths created).

Figure 4 - Ground inspector (Nodes)



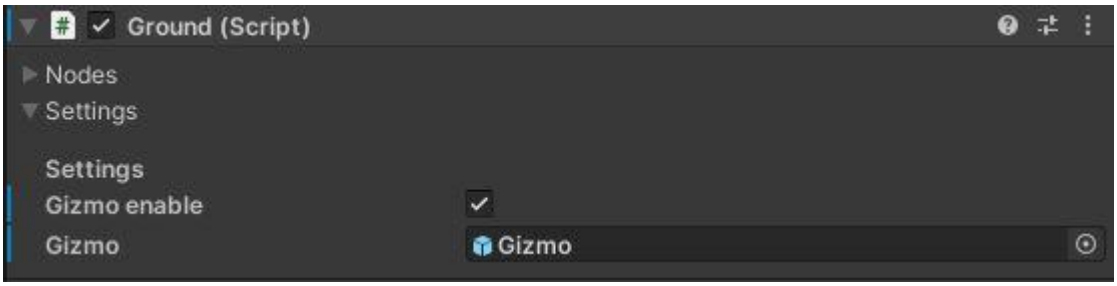
Settings

Here we have settings variables.

Table 4 - Ground (Settings)

Name	Description
Gizmo Enable	Checking this item make the helper gizmo shown.
Gizmo	The gizmo prefab

Figure 5 - Ground inspector (Settings)



Note:

Gizmo prefab should not have any collider.

It`s better to define car object(s) manually but if don`t, all car object(s) detected automatically.

CarObject.cs

It`s the main script to draw the path for handling your car (or any object).

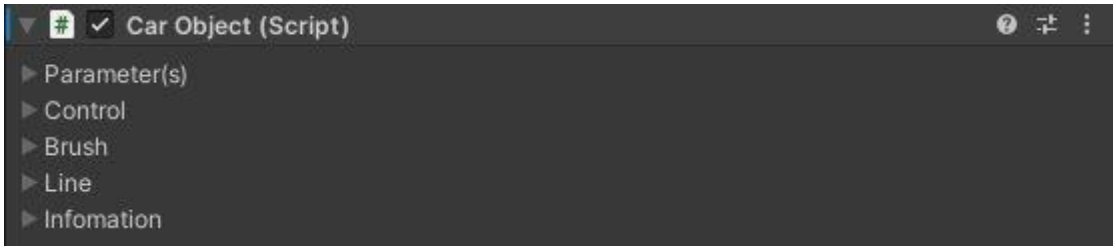
Path: Draw and Drive\Scripts\CarObject.cs

Here we have some general options. Each section has its own variables.

Table 5 - CarObject (General sections)

Name	Description
Parameters	Car object parameters
Control	Controllable settings
Brush	Brush setting
Line	Line setting
Information	Car object information

Figure 6 - CarObject inspector (General sections)



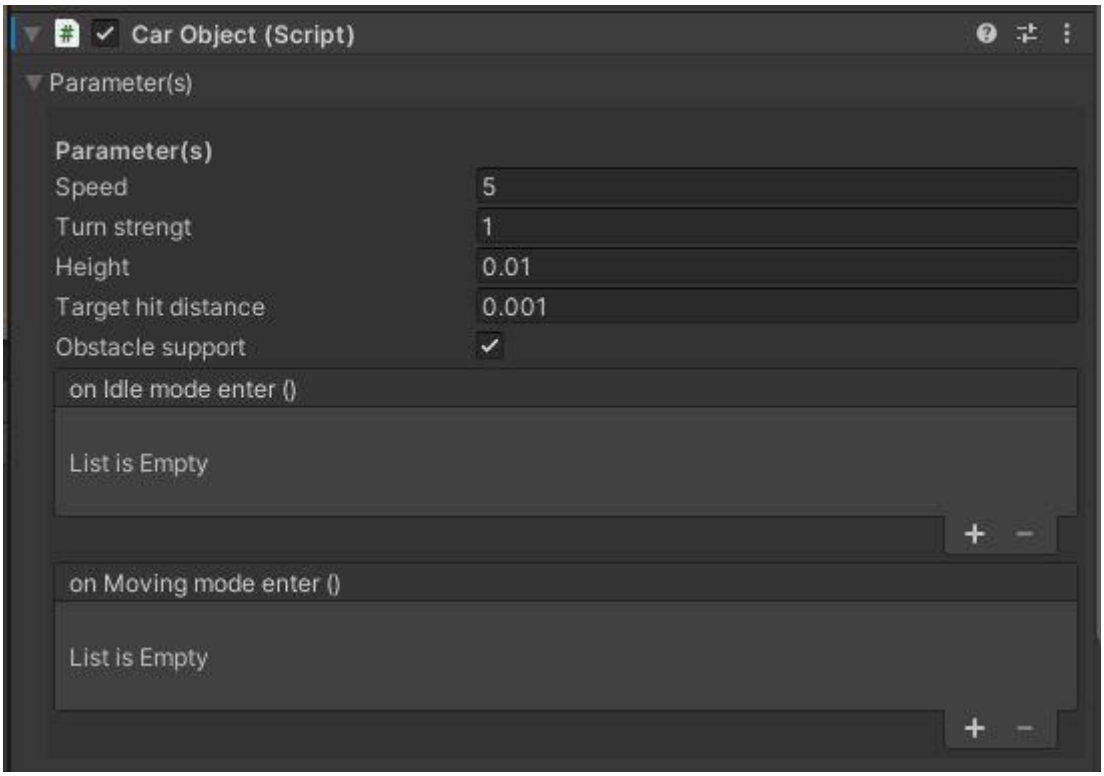
Parameters

Here we have parameters variables.

Table 6 - CarObject (Parameters)

Name	Description
Speed	Float number to handle your car speed. Higher number cause higher speed.
Turn strength	Float number to handle your car turnings. Higher number cause fast turning.
Height	The value of Y axis that car is above of ground.
Target hit distance	When the distance between car and target path is equal or lower than this value-car goes on next target.
Obstacle Support	Detect any obstacle when drawing a path
On Idle mode enter	In idle mode this Animation will be play
On moving mode enter	In moving mode this animation will be play

Figure 7 - CarObject inspector (Parameters)



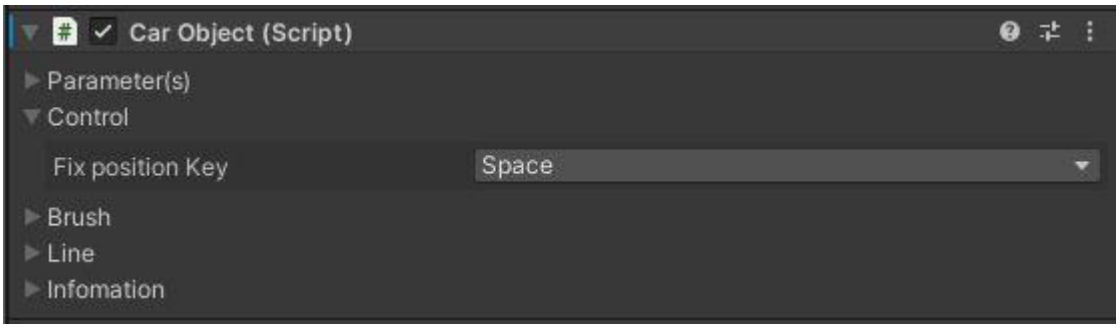
Controls

Here we have controls variables.

Table 7 - CarControl (Controls)

Name	Description
Fix position key	Save the last position.

Figure 8 - CarControl inspector (Control)



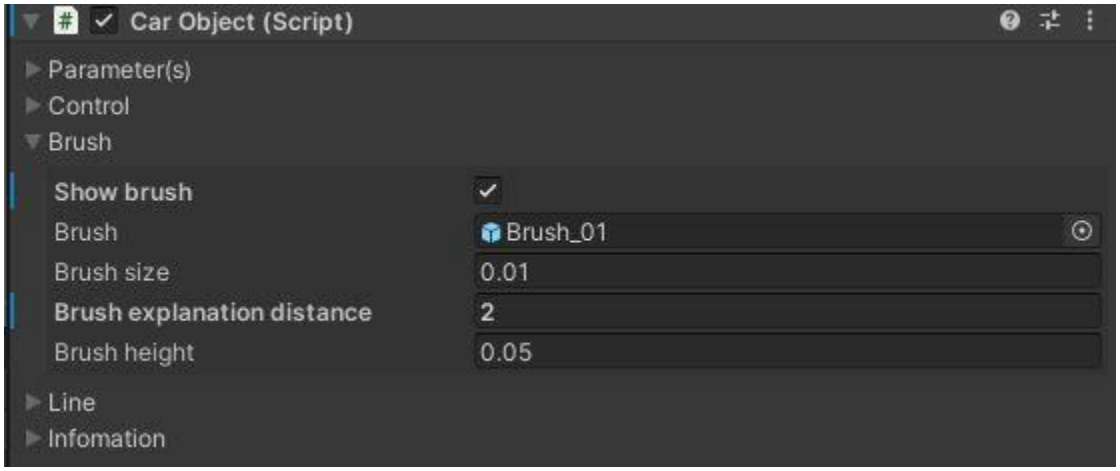
Brush

Here we have brush variables.

Table 8 - CarControl (Brush)

Name	Description
Show Brush	Boolean value to show brush in scene.
Brush	Brush prefab. This item shows on each target point.
Brush Size	Value of brush size in local scale axis.
Brush Explanation Distance	The distance between each two target nodes. The lower value makes the path more meandrous
Brush Height	The distance between brush and the ground.

Figure 9 - CarControl inspector (Brush)



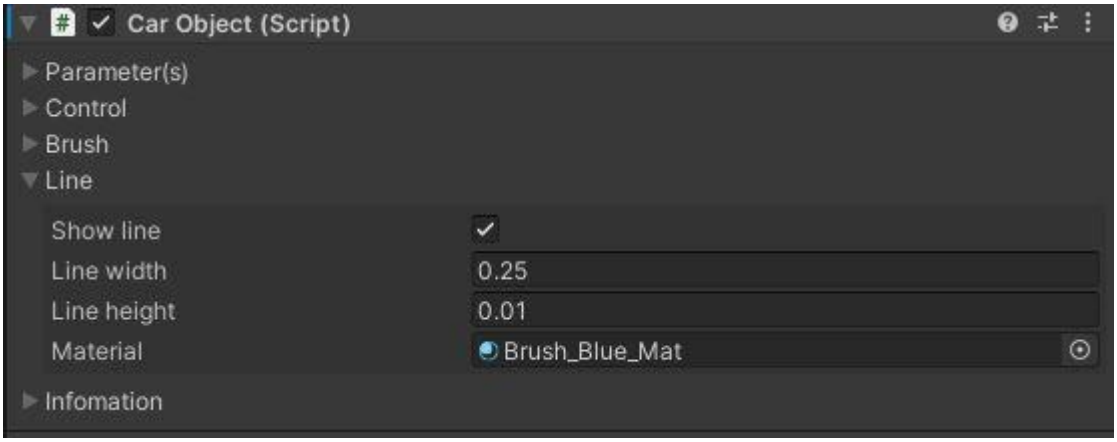
Line

Here we have line variables.

Table 9 - CarControl (Line)

Name	Description
Show Line	Boolean value to show line that connect between each of two targets in scene
Line Width	Line Width value
Line Height	The distance between line and the ground.
Material	The line material

Figure 10 - CarControl inspector (line)



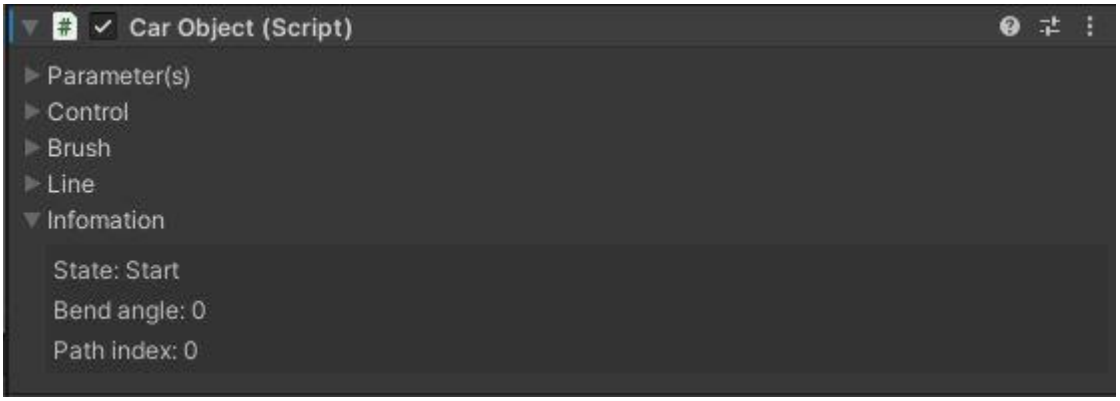
Information

Here we have information variables.

Table 10 - CarControl (Information)

Name	Description
State	The current state of car object
Path Index	The index of current target to hit (begins at zero)
Bend angle	Bend angle in movement

Figure 11 - CarControl inspector (Information)

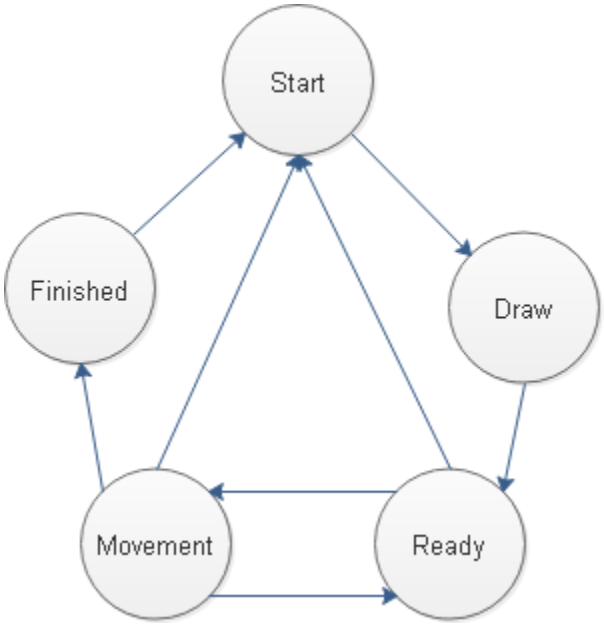


Note:
All prefabs and materials can be empty (script will correct them).
The Info values are just for analytics. It can help to understand game logic for game developer.
In case of using Rigid body for car(s)/node(s), Gravity must be “0” and deactivated.

Car Object states

For handling car(s), **state** exists to understand system and logic. Each state allows the object to do some jobs and prevent to do some others. Additionally, car(s)/node(s) can work together in a better way when you understand other car(s) state.

Figure 12 - CarObject States



There are all states to work:

Start	It`s like idle mode. In this state car is waiting to draw its own path.
Draw	Ray hit car and car`s own way is draw; the logic saves this position(s).
Ready	The path is drawn and position is saved. Now the car has own path and ready to go.
Movement	Car following each target position(s) one by one
Finished	All target position(s) are hits

Note:
By defining new steps and jobs, the system can do more complex works.

Functions

Table 11 - CarObject Functions

Name	Description
setCurrentState(State value)	Set current state to value
getCurrentState()	Return current state
Clear()	Clear all path and brush nodes
SaveCurrentPosition()	Set current position for car

Other scripts

CameraOrbit

This function is optional and handling the camera. Just assign this script to your camera and configure the parameters. This script gives you a nice camera orbit and zooming around the ground.

Table 12 - Camera Orbit (Parameters)

Name	Description
Enable	Enable or disable camera orbit
Target	The game object that camera orbit around target
Max distance	The maximum distance from camera and target
Min distance	The minimum distance from camera and target
Max height angle	The maximum angle from top of target and camera
Min height angle	The minimum angle from top of target and camera
Show camera area	Show the camera area in scene window
Show snap points	Show the camera snap points in scene window
Snap points	The array of snap points. If camera is unbounded and is in snap angle, the camera will auto orbit to snap point
Pivot	The angle in degrees that snap point will be get
Angle	The threshold of snap angle that if camera get in there and unbounded, the camera auto orbit to snap point
Orbit Damping	The damping of snap orbiting

InputManager

This function is optional and is the CameraOrbit requirement script. This script will be handling camera information from touchpad or mouse.

Table 13 - Input Manager (Parameters)

Name	Description
Input type	Type of input is mouse or touchpad

Online help

Watch Demo for PC, Mobile and script setup on YouTube:

Demo: https://www.youtube.com/watch?v=iiJb5FdoqKg

Setup: https://www.youtube.com/watch?v=5U92kxyhXbQ

Android test: https://www.youtube.com/watch?v=6cWD-TfTa8c

Contact us

You need help? Or want to improve this code?

If you need any further assistance, please contact us immediately.

unrealisticarts@gmail.com