Flying Vehicle Flying System Documentation

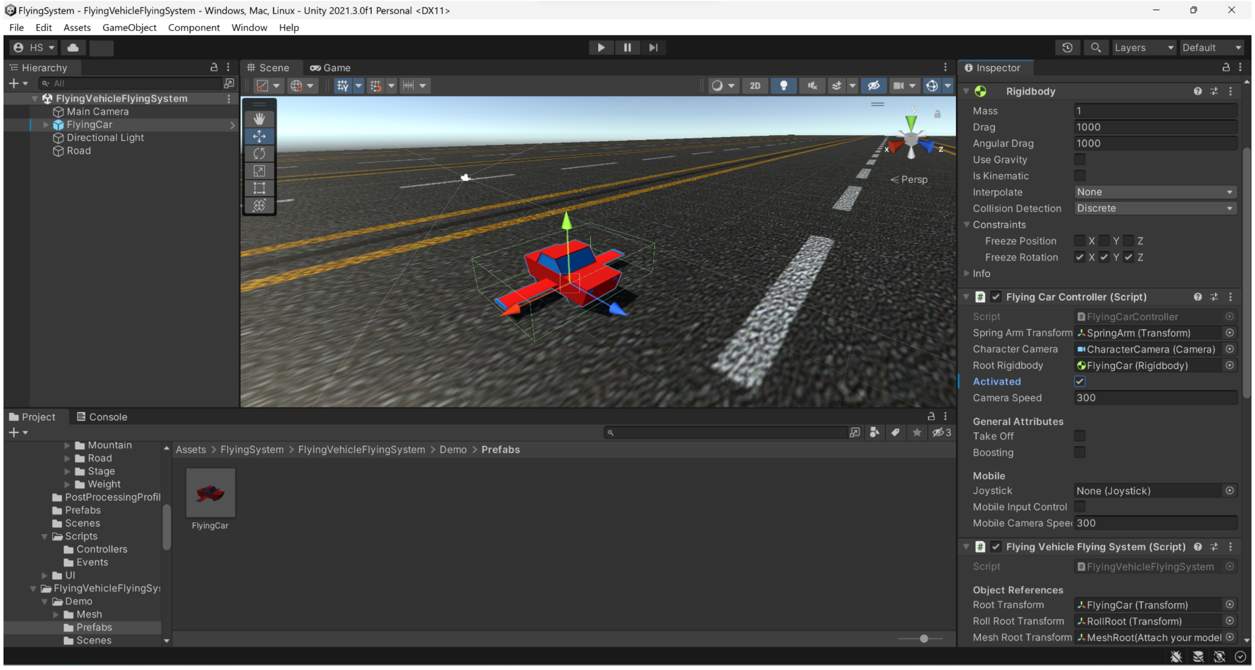
Description

This system is used for simulating flying behaviors of aircrafts like flying car.

Hover mode: Press C to turn it on/off. If it is on, it won’t be affected by the gravity.

Setup

1. Import the plugin.
2. Install “Post Processing” using Package Manager if it is not installed.
3. Create a new scene and make a simple terrain, navigate to Assets\FlyingSystem\FlyingVehicleFlyingSystem\Demo\Prefabs folder, drag the FlyingCar prefab into the scene, select “Activated” for the FlyingCarController script on the Inspector panel.



Rigidbody

Mass 1
Drag 1000
Angular Drag 1000
Use Gravity □
Is Kinematic □
Interpolate None
Collision Detection Discrete
Constraints
Freeze Position □ X □ Y □ Z
Freeze Rotation ☑ X ☑ Y ☑ Z
Info

Flying Car Controller (Script)

Script FlyingCarController
Spring Arm Transform SpringArm (Transform)
Character Camera CharacterCamera (Camera)
Root Rigidbody FlyingCar (Rigidbody)
Activated ☑
Camera Speed 300

General Attributes
Take Off □
Boosting □

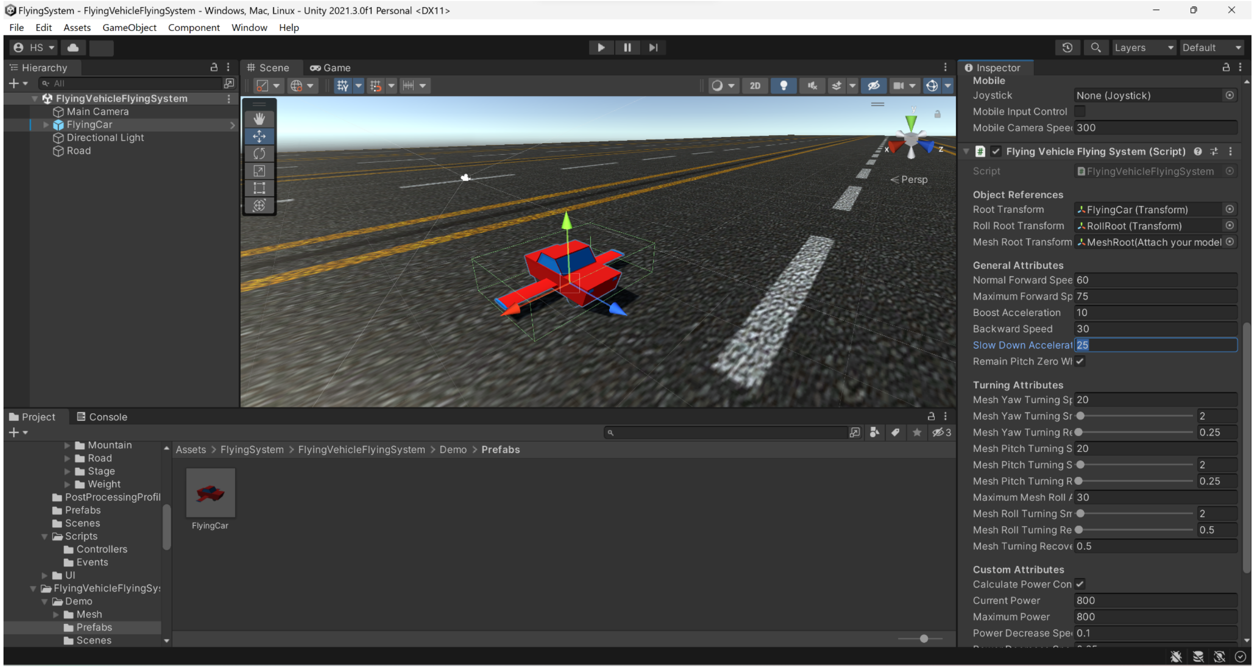
Mobile
Joystick None (Joystick)
Mobile Input Control □
Mobile Camera Spee 300

Flying Vehicle Flying System (Script)

Script FlyingVehicleFlyingSystem

Object References
Root Transform FlyingCar (Transform)
Roll Root Transform RollRoot (Transform)
Mesh Root Transform MeshRoot(Attach your model

4. Press the Play button, the flyer would be controllable. Depending on the need, the attributes and methods in Flying Vehicle Flying System component can be adjusted/executed.


FlyingSystem - FlyingVehicleFlyingSystem - Windows, Mac, Linux - Unity 2021.3.0f1 Personal <DX11>
File Edit Assets GameObject Component Window Help
HS
Layers
Default
Hierarchy
FlyingVehicleFlyingSystem
Main Camera
FlyingCar
Directional Light
Road
Scene
Game
2D
Persp
Project
Console
Mountain
Road
Stage
Weight
PostProcessingProfil
Prefabs
Scenes
Scripts
Controllers
Events
UI
FlyingVehicleFlyingSy
Demo
Mesh
Prefabs
Scenes
Assets > FlyingSystem > FlyingVehicleFlyingSystem > Demo > Prefabs
FlyingCar
Inspector

Mobile
Joystick: None (Joystick)
Mobile Input Control: □
Mobile Camera Spee: 300

Flying Vehicle Flying System (Script)
Script: FlyingVehicleFlyingSystem

Object References
Root Transform: FlyingCar (Transform)
Roll Root Transform: RollRoot (Transform)
Mesh Root Transform: MeshRoot(Attach your model

General Attributes
Normal Forward Spee: 60
Maximum Forward Sp: 75
Boost Acceleration: 10
Backward Speed: 30
Slow Down Accelerat: 25
Remain Pitch Zero Wl: ✓

Turning Attributes
Mesh Yaw Turning Sp: 20
Mesh Yaw Turning Sr: 2
Mesh Yaw Turning Re: 0.25
Mesh Pitch Turning S: 20
Mesh Pitch Turning S: 2
Mesh Pitch Turning R: 0.25
Maximum Mesh Roll A: 30
Mesh Roll Turning Sm: 2
Mesh Roll Turning Re: 0.5
Mesh Turning Recove: 0.5

Custom Attributes
Calculate Power Con: ✓
Current Power: 800
Maximum Power: 800
Power Decrease Spe: 0.1

Explanation

Setting the pitch, yaw, roll together for the same GameObject will cause gimbal lock problem(https://en.wikipedia.org/wiki/Gimbal_lock). To solve this problem, two GameObjects(RollRoot, MeshRoot) are used to separate them.

The prefab has the following hierarchy:

- FlyingCar
  - SpringArm
    - CharacterCamera
  - RollRoot
    - MeshRoot
      - FlyingCar(3d model)

1. FlyingCar: It only acts as the root, and doesn't represent the actual collision for the flyer. Rigidbody, C# scripts and audio source are attached to this GameObject.
2. SpringArm: For controlling the third-person camera.
3. CharacterCamera: The actual camera.

4. RollRoot: The roll value of the flyer will be set here.
5. MeshRoot: The pitch and yaw values will be set here.
6. FlyingCar(3d model): The actual flyer mesh, can be customized. The mesh should be attached to MeshRoot, since the flying rotation will be applied to its parent, not directly to the mesh.

Essential Object References

rootTransform	The Transform component of the root object.
rollRootTransform	The Transform component of the roll root object. There must be a RollRoot object above the MeshRoot object in the Hierarchy. See the example of “FlyingCar.prefab” in Assets\FlyingSystem\Demo\Prefabs\Controllers\AirTransportations folder.
meshRootTransform	The Transform component of the mesh root object.

Adjustable Attributes

General Attributes	
normalForwardSpeed	The speed of normal forward flying.
maximumForwardSpeed	The maximum speed of forward flying in boost mode.
boostAcceleration	It defines how fast the speed will increase from normalForwardSpeed to maximumForwardSpeed.
backwardSpeed	The speed of normal backward flying.
remainPitchZeroWhenHovering	Whether the pitch value remains 0 when hovering(no movement input) in the air.

Turning Attributes	
meshYawTurningSpeed	The speed of yaw(horizontal) turning.
meshYawTurningSmoothingFactor	The interpolated parameter for meshYawTurningSpeed. The smaller the value, the smoother and slower the turning will be.
meshYawTurningRecoverySmoothingFactor	The interpolated parameter for yaw back to 0. The smaller the value, the smoother and slower the turning will be.
meshPitchTurningSpeed	The speed of pitch(vertical) turning.
meshPitchTurningSmoothingFactor	The interpolated parameter for meshPitchTurningSpeed. The smaller the value, the smoother and slower the turning will be.
meshPitchTurningRecoverySmoothingFactor	The interpolated parameter for pitch back to 0. The smaller the value, the smoother and slower the turning will be.
maximumMeshRollAngle	The maximum of roll angle.
meshRollTurningSmoothingFactor	The interpolated parameter for roll turning. The smaller the value, the smoother and slower the turning will be.
meshRollTurningRecoverySmoothingFactor	The interpolated parameter for roll back to 0. The smaller the value, the smoother and slower the turning will be.
meshTurningRecoveryDelay	The delay(in seconds) of triggering mesh rotation back to 0 for pitch and roll.
Custom Attributes	
calculatePowerConsumption	Whether calculate the power consumption or not.
currentPower	The current power.
maximumPower	The maximum power.

powerDecreaseSpeed	It defines how fast the power drops when flying.
powerDecreaseSpeedWhenBoosting	It defines how fast the power drops when flying in boost mode.
speedRemainingPowerRatioAnimationCurve	An editable curve of the relation between speed and remaining power. Normally the lower power, the slower speed.
calculateCarryingWeight	Whether calculate the carrying weight or not.
currentCarryingWeight	The current carrying weight.
maximumCarryingWeight	The maximum carrying weight.
speedCarryingWeightRatioAnimationCurve	An editable curve of the relation between speed and carrying weight. Normally the higher carrying weight, the slower speed.
Other Public Attributes	
enabledFlyingLogic	Whether enable/disable the flying logic.
inAir	Whether it is in the air or not.
flyingDirection	The flying direction, in Vector3.
flyingSpeed	The flying speed.
flyingVelocity	The flying Velocity, in Vector3.
flyingAtNormalSpeed	Whether it is flying in normal speed mode or not.
boosting	Whether it is flying in boost mode or not.
powerPercentage	The percentage of power, equals 1 when power is full.
weightPercentage	The percentage of carrying weight, equals 1 when fully carried.

Methods

TakeOff()	Make the aircraft take off.
Land()	Make the aircraft land.
AddForwardInput(float value)	Make the aircraft move in forward direction, the value can be positive or negative.
StopMovingForward()	Make the aircraft stop moving in forward direction.
AddYawInput(float value)	Make the aircraft turn left/right.
StopTurning	Make the aircraft stop turning left/right.
AddPitchInput(float value)	Make the aircraft point up/down.
StopPitchInput()	Make the aircraft stop pointing up/down.
AddWeight(float increaseValue)	Increase the current carrying weight of the aircraft, the increaseValue can be positive or negative.

Universal Render Pipeline (URP) & High Definition Render Pipeline (HDRP)

The materials can be converted by following this tutorial:
https://www.youtube.com/watch?v=aJ1OpirisGM

References

The 3D model(FlyingCar.fbx) is the original creation by the developer of this project.