EXOA

TOUCH CAMERA PRO - MANUAL

08/06/2022

Latest online Version is accessible here

INTRODUCTION

Touch Camera PRO is a really easy to use mobile and desktop camera controller with perspective switching!
It's working on both desktop and mobile devices! It supports translation, rotation around center, rotation around point, Zoom In/Out, Object Focus/Follow etc, on every camera type.

Other features : scene boundaries, camera reset in initial place, Move To Position/Rotation/Distance etc..

HOW TO INSTALL

- Download from the asset store and import in your project. There is no dependency anymore to install!

BASIC USE

1. Add both prefabs TouchCamera & TouchCameraInputs inside your scene

2. If you want to add boundaries to your camera, fill the CameraBoundaries component on the TouchCamera gameObject with a another gameObject (having a collider, see demos)

3. Edit parameters on the CameraPerspective & CameraTopDown components if needed.

CAMERA TYPES

Depending on the camera type you need, you can choose between:

- **Camera Perspective**: a 3D perspective camera focusing on the floor.
- **Camera Top Down Ortho**: a 2D top down orthographic camera working on the XZ plane
- **Camera Isometric Ortho**: a 3D isometric camera focusing on the floor.
- **Camera Side Scroll**: a 2D orthographic camera working on the XY plane
- **Camera Free**: a beta 360 perspective camera working with colliders

COMPONENTS & PARAMETERS

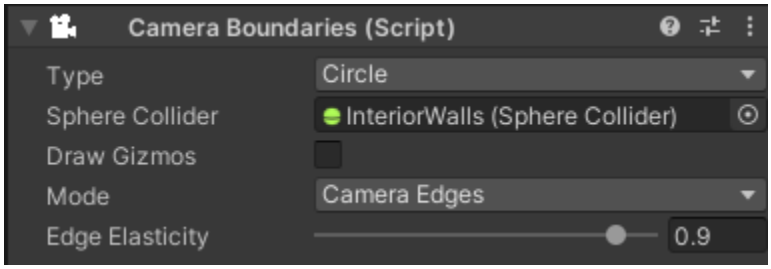
Camera Mode Switcher

This component is needed only if you need to have two camera modes on your camera.

You can add a perspective camera mode (CameraPerspective) and an orthographic camera mode (CameraTopDownOrtho or CameraIsometricOrtho) and use the perspective switch feature thanks to CameraModeSwitcher.

With 2 camera modes you need to specify which one is the default one by checking "default mode".

Camera Boundaries



Boundaries are meant to restrict your camera's position in the scene. Two shape types are supported, Rectangle and Circle. For Rectangle, you will have to fill the box collider field and for a circle, you will have to fill the sphereCollider field. Make sure that your sphere collider Y position is on the ground.

You can check the "draw gizmo" option to see how the boundaries are in the end calculated based on your collider.

The next important option is the boundary mode. "Camera Center" will restrict the camera view center on the ground inside the boundaries. "Camera Edges" will restrict the edges of the camera view projected on the ground inside the boundaries.

The second mode works great when the camera looks down (60°-90°) (top down or isometric modes) all four corners will be restricted. For the perspective mode, if you allow the camera to have an angle < 60° (Pitch Rotation) then only the bottom corners of the camera will be restricted in the boundaries.

The first mode (Camera Center) is calculated before the camera applies the movement, whereas the Edge Mode is calculated only after the camera has moved, as it's based on the frustum final state. This is why we apply an elastic effect in that case.

Camera Perspective

Inputs: here you can specify what the mouse buttons are doing and also what one finger drag is doing on mobile. Two fingers drag will always be dedicated to zoom and rotate (pinch and twist) as it is the default behaviour with 2 fingers.

Focus: When you call the FocusCameraOnGameObject() feature, you can specify here the duration of the animation and the kind of easing. The distance and radius multipliers help to be closer or not from the game object while focusing.

Distance: This is specifying the default distance of the camera from the ground. You can also clamp that distance with a min/max value.

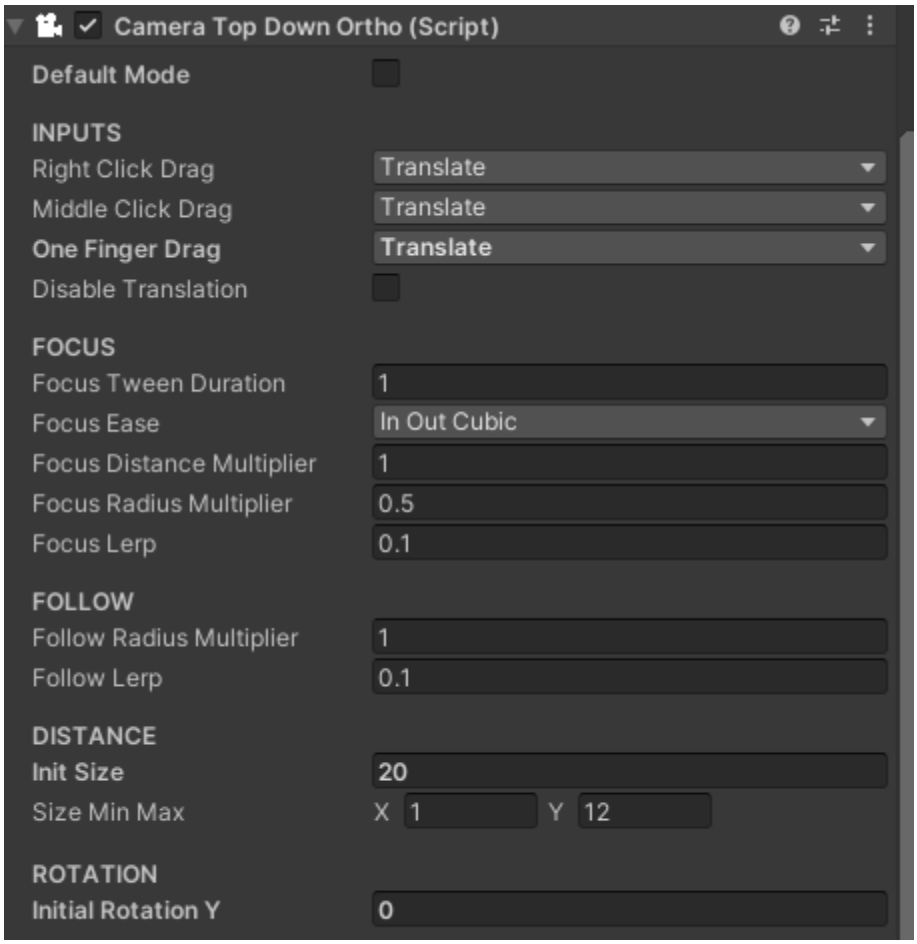
The distance changes when the user zooms in/out using the mouse scrollwheel or the pinch gesture.

Follow: Specifies how far the camera should be from the object we are following. This value will be multiplied by the object bounding box size.

Rotation: Yaw rotation is the rotation around the world Y axis, so there is no clamping as the camera can rotate all around Y.

The pitch rotation is around the X axis. 0° means that the camera is parallel to the ground, and 90° means that the camera is looking straight to the ground. I do not recommend to put the minimum pitch at 0° as that will cause issues with raycasting on the ground to move the camera.

Camera Top Down Ortho



Inputs: here you can specify what the mouse buttons are doing and also what one finger drag is doing on mobile. Two fingers drag will always be dedicated to zoom and rotate (pinch and twist) as it is the default behaviour with 2 fingers.

Focus: When you call the FocusCameraOnGameObject() feature, you can specify here the duration of the animation and the kind of easing. The distance and radius multipliers help to be closer or not from the game object while focusing.

Distance: This is specifying the default distance of the camera from the ground. You can also clamp that distance with a min/max value.

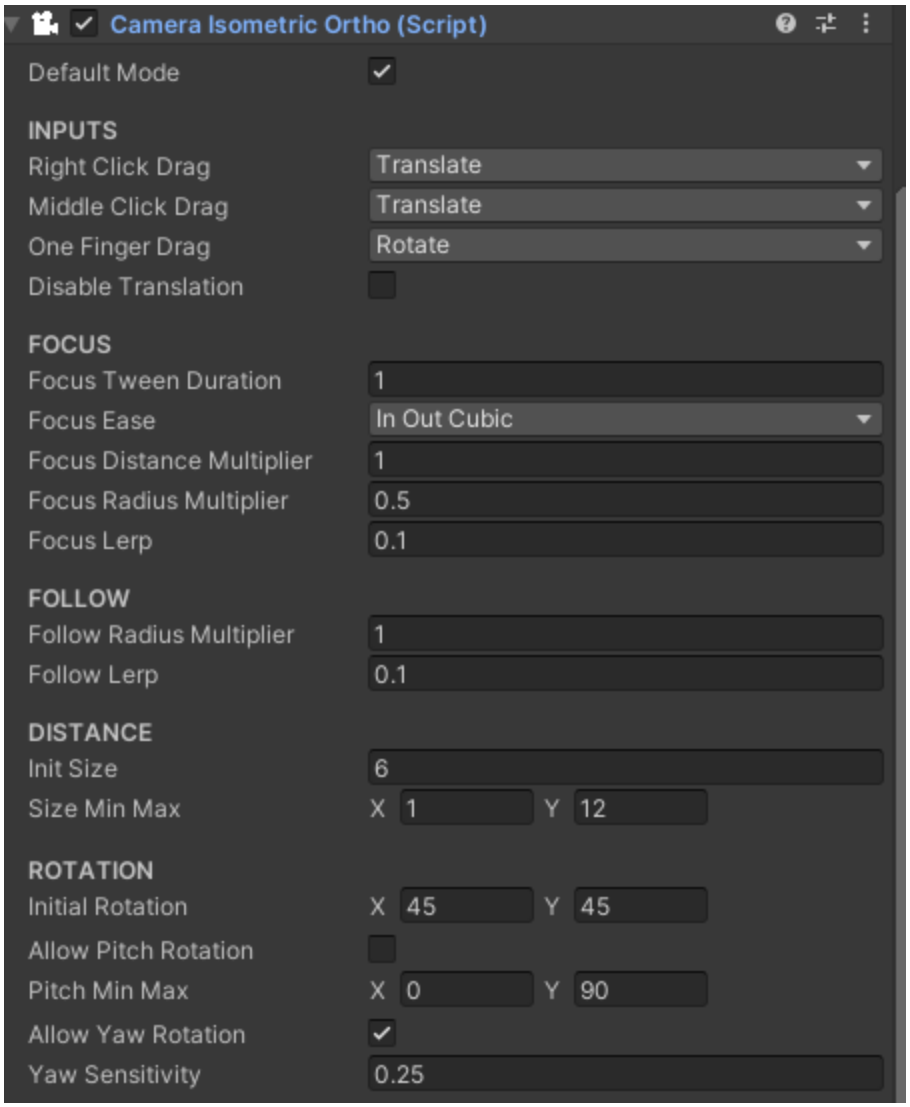
The distance changes when the user zooms in/out using the mouse scrollwheel or the pinch gesture.

In orthographic mode, the camera doesn't move on his Z axis while zooming, so this is why we are using the camera size here instead of an actual distance.

Follow: Specifies how far the camera should be from the object we are following. This value will be multiplied by the object bounding box size.

Rotation: the only setting here is to set the initial world Y rotation (Yaw)

Camera Isometric Ortho



Inputs: here you can specify what the mouse buttons are doing and also what one finger drag is doing on mobile. Two fingers drag will always be dedicated to zoom and rotate (pinch and twist) as it is the default behaviour with 2 fingers.

Focus: When you call the FocusCameraOnGameObject() feature, you can specify here the duration of the animation and the kind of easing. The distance and radius multipliers help to be closer or not from the game object while focusing.

Distance: This is specifying the default distance of the camera from the ground. You can also clamp that distance with a min/max value.

The distance changes when the user zooms in/out using the mouse scrollwheel or the pinch gesture.

In orthographic mode, the camera doesn't move on his Z axis while zooming, so this is why we are using the camera size here instead of an actual distance.

Follow: Specifies how far the camera should be from the object we are following. This value will be multiplied by the object bounding box size.

Rotation: Yaw rotation is the rotation around the world Y axis, so there is no clamping as the camera can rotate all around Y.

The pitch rotation is around the X axis. 0° means that the camera is parallel to the ground, and 90° means that the camera is looking straight to the ground. I do not recommend to put the minimum pitch at 0° as that will cause issues with raycasting on the ground to move the camera.

API DOCUMENTATION

The API doc is available here: http://monitor.exoa.fr/tcp-doc

The sections below will describe the main methods.

SWITCHING PERSPECTIVE

There are two ways to switch perspective in the demos, by pressing the “Space bar” or by clicking the 3D icon. To change the key you can edit Inputs.cs in the ChangePlanMode() function.

The other way is done by triggering an event as follow :

```
CameraEvents.OnRequestButtonAction?.Invoke( CameraEvents.Action.SwitchPerspective, true);
```

If you only need one mode in your scene you can remove either the “CameraPerspective” or “CameraTopDownOrtho” component and the CameraModeSwitcher component.

MOVE CAMERA TO

You can call MoveCameraTo() or MoveCameraToInstant() to move the camera with or without an animation. You can pass to these methods a position, a rotation and a distance. Here are the different variations you can call. (See the API Documentation)

void	**MoveCameraToInstant** (Vector3 targetPosition)
void	**MoveCameraToInstant** (Vector3 targetPosition, float targetDistance)
void	**MoveCameraToInstant** (Vector3 targetPosition, Quaternion targetRotation)
void	**MoveCameraToInstant** (Vector3 targetPosition, Vector2 targetRotation)
void	**MoveCameraToInstant** (Vector3 targetPosition, float targetDistance, Vector2 targetRotation)
void	**MoveCameraToInstant** (Vector3 targetPosition, float targetDistance, Quaternion targetRotation)
void	**MoveCameraTo** (Vector3 targetPosition)
void	**MoveCameraTo** (Vector3 targetPosition, float targetDistance)
void	**MoveCameraTo** (Vector3 targetPosition, Quaternion targetRotation)

void **MoveCameraTo** (Vector3 targetPosition, Vector2 targetRotation)

void **MoveCameraTo** (Vector3 targetPosition, float targetDistance, Vector2 targetRotation)

void **MoveCameraTo** (Vector3 targetPosition, float targetDistance, Quaternion targetRotation)

FOCUSING ON A GAME OBJECT

In the demos, clicking on a cube will trigger a focus on that object. Check out the "FocusOnClick.cs" script to see how it's done.

You basically just have to trigger an event like so :

CameraEvents.OnRequestObjectFocus?.Invoke(gameObject);

You can also call a method on a camera instance:

void **FocusCameraOnGameObject** (GameObject go, bool allowYOffsetFromGround=false)

FOLLOWING A GAME OBJECT

In the demos, clicking on a moving cube will trigger a follow on that object. Check out the “FocusOnClick.cs” script to see how it’s done.

You basically just have to trigger an event like so :

CameraEvents.OnRequestObjectFollow?.Invoke(gameObject, focusOnFollow);

You can also call a method on a camera instance:

```
void FollowGameObject (GameObject go, bool focusOnFollow, bool
     allowYOffsetFromGround=false)
```

The focusOnFollow parameter is a boolean. A value at false, is to follow only the position of the object, keeping the same distance that you can control with inputs. Setting “true” would also lock the distance regarding the object size on screen. You can control the focus multiplier on the camera’s component properties.

RESET THE CAMERA POSITION

The new “Camera Reset” feature, helps to put the camera back in its initial position and rotation. In the demos it is done by clicking the “R” icon.

```
CameraEvents.OnRequestButtonAction?.Invoke( CameraEvents.Action.ResetCamera, true);
```

You can also call a method on a camera instance:

```
void ResetCamera ()
```

The camera default position/rotation/ditance is calculated from where you placed the camera in the scene.

To change the camera default position/rotation/distance for the reset feature, you can use this method :

```
void SetResetValues (Vector3 offset, Quaternion rotation, float distance, float size)
```

CALLBACK EVENTS

In the class CameraEvents, a few callbacks are available:

OnFocusComplete / OnFocusStart: triggered when a follow/focus/moveTo animation starts and ends.

OnBeforeSwitchPerspective / OnAfterSwitchPerspective: triggered before an after the switch perspective animation happens.

SIMULATING FINGERS TWIST AND PINCH

Thanks to Lean Touch, you can simulate the finger inputs. Pressing ALT + CLick on the ground will place a virtual “fingers center point” then holding the Ctrl key + clicking and dragging will let you simulate the pinch (scale) and twist (rotation) around a point on screen.

SHORTCUTS

SWITCH PERSPECTIVE

Press "Space Bar", or press the top right "camera" button

IN TOP DOWN ORTHOGRAPHIC MODE

Mouse Wheel : Zoom In/out
Left/Right/Middle Mouse Button Press & Drag : Drag Camera

IN PERSPECTIVE MODE

Mouse Wheel : Zoom In/out
Left/Middle Mouse Button Press & Drag : Drag Camera
Right Mouse Button Press & Drag : Rotate Around Center

TOUCH SIMULATION IN BOTH MODES

Alt+Left Mouse Button Click : Set the center point of the simulated fingers
Alt+Left Mouse Button Press & Drag : Drag Camera
Ctrl+Left Mouse button Press & Drag : Two fingers simulation for Pinch (Zoom In/Out) and Twist (Rotate around fingers center point)

DEMOS

Android and PC Demos are accessible here

ROADMAP

- Integrate with the New Input System from Unity

OTHER PLUGINS

- Home Designer
- Floor Map Designer
- Level Designer
- Assets Manager Pro
- Packages Manager Free
- Tutorial Engine

SUPPORT

Please post your questions and issues on the new forum : https://support.exoa.fr/