StrayTech Camera System

Version 1.0

StrayTech Camera System is a professional quality camera solution designed to allow developers to quickly implement and refine a wide array of camera setups and transitions.

This package contains eight different camera types. Set up camera state definitions with ease, place trigger volumes for the camera states where you want them and the rest is handled by the system. There are three different ways to transition between states; interpolation, crossfade, and instant. Camera state modifiers allow you to quickly and easily set up behaviors to modify the current camera state's base behavior.

Playmaker extensions are included as a separate unity package.

This document is a brief overview of the features contained in this package. Please check out the Sample scene and refer to the implementations of each the included Camera States and Modifiers in that scene. All Camera State's settings can be modified at runtime, but as with all unity components, those changes will be lost once play mode is stopped. The common strategy for saving the changes to a component at runtime is to copy the component and paste the component's values over the old ones outside of play mode.

Note from the developer: *This product was developed by a single person, . I have done the best that I can to make sure that the product is bug free and that it is as simple to use as possible. Though I have strived to make the best product I can, I understand that there may be defects and edge case bugs. Please contact me by email with any issues you may face with my product and I will do my best to support your needs. I will fix bugs as soon as possible and release updates as often as I can. Thank you for buying StrayTech Camera System.*

Camera System Component:

The Camera System component is a singleton that controls the entire system and is required to be present for camera states and triggers to function.

- **Camera Target** - The target transform for all cameras to look at. This value can be set in code. If a different camera target is required for a short duration, then the ‘Look At Target’ or ‘Zoom In On Target’ modifiers can be used.
- **Use Fixed Update** - The camera system as a whole can be set to update within the fixed update interval. This can be used if your camera target’s position is updated in the fixed update interval.
- **User Defined Flags** - User defined Flags are set in code and are either true or false. These flags can then be used to gate trigger logic. An example would be a “PlayerInput_Use” flag being set when the player presses and holds the ‘E’ button on the keyboard, and false when the button isn’t pressed. User Defined Flag Trigger Gate components can then be used to gate (allow or prevent) the trigger from firing based on the state of the User Defined Flag.

Camera Collision Component:

The Camera Collision component is a singleton that if present in the scene will provide camera collision support.

- **Use Camera Collision** - A global toggle for camera collision support.
- **Test Type** - The type of testing to use (Raycast or SphereCast).
- **Sphere Radius -** If SphereCast is used, this is the radius of the sphere.
- **Collision Layer Mask** - Allows one to set the layers to test against. The cameras will collide with these layers.

Camera State Definition Component:

The Camera State Definition component is where a camera state is defined. All of the parameters for the camera are stored in this component.

- **Camera State** - The desired camera state to use.
  - **Isometric -** An isometric locked perspective camera.
  - **Spline -** A spline following camera.
  - **First Person -** Basic first person perspective camera.
  - **Third Person -** Basic third person camera.
  - **Animated -** A camera which is controlled by animation.
  - **Pivot -** Camera that pivots on a fixed point and looks at the target.
  - **Parented -** Camera that is parented to a transform.
- **Transition Type** - The type of transition that will be used to transition into this state.
  - **Interpolation** - Smoothly moves the camera from the previous state's position to this state's position.
  - **Crossfade** - Fades the camera from the previous state to this state.
  - **Instant** - An instant cut transition.
- **Transition Duration** - The duration of the transition from the previous state to this one.
- **Camera** - (Optional) - Allows one to override the default camera and instead use the provided camera for this camera state.
- **State Settings** - (Listed Below).

Camera States:

1. **Isometric Camera**
   - **Rotation** - World space Euler rotation to lock the camera’s view to.
   - **Distance** - The distance the camera will be from the target.
   - **Use Camera Collision** - Whether to use camera collision or not. (Requires Camera Collision Component mentioned above)

2. **Spline Camera**
   - **Spline** - A reference to the Bezier Spline to use.
   - **Spline Position Offset** - Offset the camera on the spline from the start in world units of length.
   - **Camera Line Of Sight Offset** - Offset along the line of sight to the target.
   - **Camera Max Distance** - Maximum distance the camera can be from the target.
   - **Spline Travel Max Speed** - The maximum speed the camera can travel along the spline in world units of length per second.
   - **Use Camera Collision** - Whether to use camera collision or not (Requires Camera Collision Component mentioned above)

3. **First Person Camera**
   - **Position Root Transform** - The transform that the camera’s position will be parented to.
   - **Character Transform** - The root transform of the character. (Y axis camera rotation is applied to the root)
   - **Position Offset** - Position offset from the Position Root Transform
   - **Pitch Range** - The range of vertical rotation.
   - **Mouse Look Sensitivity** - Sensitivity of mouse movement on each axis.
   - **Mouse Smoothing** - Smooths the mouse input. (High value = faster interpolation)

4. **Third Person Camera**
    - **Mouse Orbit** - Use the mouse to control the camera's orbit.
    - **Mouse Orbit Distance** - The minimum and maximum distance the camera can be from the target.
    - **Pitch Range** - The range of vertical rotation.
    - **Mouse Sensitivity** - Sensitivity of mouse movement on each axis.
    - **Mouse Invert Y** - Invert Y axis input.
    - **Target Offset** - The position offset from the target.
    - **Motion Smoothing** - Smooths the mouse input. (High value = faster interpolation)
    - **Use Camera Collision** - Whether to use camera collision or not (Requires Camera Collision Component mentioned above)

5. **Animated Camera**
    - **Animation Clip** - The animation clip to play. (Needs to be a Legacy Animation Clip)
    - **Parent Override** - Use the parent override to override the root of the animation.
    - **Y Rotation Fix** - Y axis rotation adjustment (Some animations from Maya need adjustment)

6. **Pivot Camera**
    - **Pivot Host** - The transform to pivot on.
    - **Pivot Host Offset**- The offset from the pivot host position.
    - **Use Camera Collision** - Whether to use camera collision or not (Requires Camera Collision Component mentioned above)

7. **Parented Camera**
    - **Parent** - GameObject to parent to.
    - **Position Offset** - Position offset from parent.
    - **Rotation Offset** - Rotation offset from parent in Euler angles.
    - **Use Camera Collision** - Whether to use camera collision or not (Requires Camera Collision Component mentioned above)

Camera State Modifiers:

Camera State Modifiers can be used to modify the behavior of Camera States in many ways or override the behavior entirely. Modifiers are layered on top of the base behavior according to their priority.

- **Transition Into Enabled Duration** - Duration of the transition into the enabled state.
- **Transition Into Disabled Duration** - Duration of the transition into the disabled state.
- **Priority** - The priority of this modifier.

Types of Modifiers:

1. **Camera Shake Modifier**
    - **Shake Intensity Multiplier Curve** - Defines the duration and intensity multiplier of the camera shake.
    - **Shake Intensity** - The base intensity of the camera shake.

2. **Look At Target Modifier**
    - **Look At Target** - The target Transform to look at.

3. **Zoom In On Target Modifier**
    - **Target** - The target to zoom in on.
    - **Target Offset** - The offset from target.
    - **Distance From Target** - The distance to zoom into from target.

4. **Field Of View Modifier**
    - **Field Of View** - The Field of View that this modifier adjusts the camera to when it's active.

Triggers:

Camera States and Modifiers can be transitioned to in many different ways. One way is through the use of the numerous triggering components that are provided.

- **Trigger Collider Type** - Type of collider to use.
  - o **Box** - Standard Box collider.
  - o **Sphere** - Standard sphere collider.
  - o **Convex Mesh** - Uses the provided mesh as a convex collider.
- **Volume Color** - The color of the trigger volume in the editor.
- **Render Solid Volume** - Should this volume render as a solid in editor?
- **Render Only When Selected** - Render volume only when selected?
- **Tag Filter** - Filter collision by tag? (Blank means no tag filter)
- **Layer Mask** - The layers that will trigger the volume.
- **Single Use Trigger** - Use once then disable.

Types of Triggers:

1. **Camera State Transition Trigger**
   - **Target Camera State Definition** - The target Camera State Definition to transition to.

2. **Camera State Modifier Trigger**
   - **Camera State Modifier Target** - The target Camera Modifier to enable and/or disable.
   - **Ignore Trigger Exit -** Ignore the OnTriggerExit event?

3. **Camera State Modifier Timed Trigger**
   - **Camera State Modifier Target** - The target Camera Modifier to enable and/or disable.
   - **Enabled Duration -** The modifier will be disabled after this duration.

Trigger Gates:

All Camera System triggers can be gated. This means that specific logic in a script can prevent or allow the trigger’s OnTriggerEnter event to be listened to. This is useful for setting up directional triggers or having a trigger ‘zone’ that only operates when a User Defined Flag is true.

Types of Gates:

1. **User Defined Flag Trigger Gate:**
    - **User Defined Flag Name** - If this user defined flag is false, OnTriggerEnter logic will be bypassed.

2. **Directional Trigger Gate:**
    - **Angle** - The primary direction the volume can be triggered from.
    - **Angle Span** - The span of the primary direction the volume can be triggered from.

Change Log:

Version 1.0 - Initial Release