

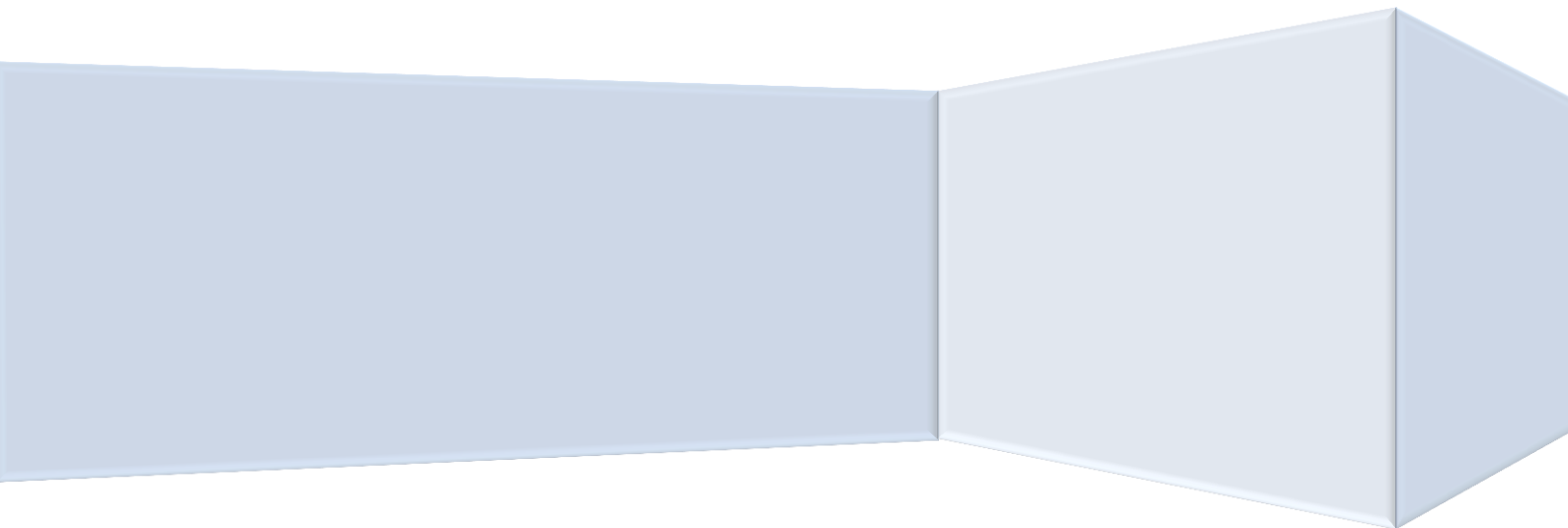
# Scale Space Manager

Version 0.1.0

---



Vampirasu Game Studios





## Index

1	Introduction.....	2
2	How it works.....	2
3	Components .....	5
3.1	Scale Space Manager.....	5
3.2	Scale Space Camera.....	5
3.3	Scale Space Object.....	6
3.4	Vector3Double .....	6
4	Getting started .....	3
5	Known issues .....	5



## 1 Introduction

The Scale Space Manager makes it possible to observe objects that are millions of units away from the camera even with shadows and all without jittering. You can also travel to places far beyond the visual borders. You can create the whole solar system in real scale and travel through it in real time. Normally to see objects that are far away from the camera the far plane has to be increased. This leads to inaccurate shadows and other visual clipping and jitter artifacts. It also decreases the performance of rendering and frustum culling. The Scale Space Manager eliminates the problem by creating multiple spaces and each space has his own scale. This and the built in floating origin mechanism makes it possible to travel long distances and observe huge objects that are millions of units away.


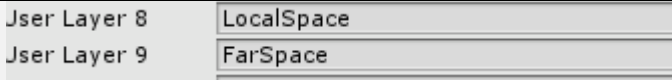
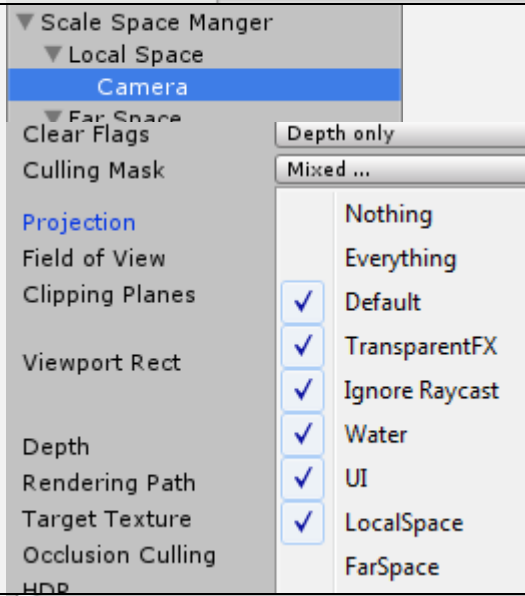
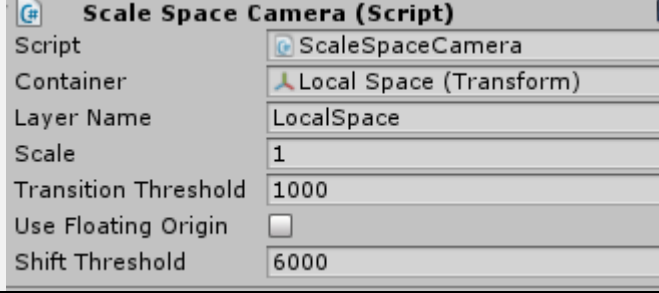
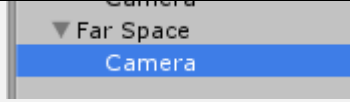
## 2 How it works

The Scale Space Manager Asset enables you to create different “Spaces” every space has his own scale and Camera. If an object reaches a specific distance from the camera it gets transitioned to the next higher Scale Space. This transition will also scale down, reposition the object and put him to a new layer. If the object gets below a specific distance it will transition back to a lower Scale Space. Every Scale Space needs at least 1 layer. To encounter the floating jitter problem there is an optional floating origin mechanism for every Layer. All key values are configurable for details see 4.



## 3 Getting started

To create a minimal setup just follow the steps:

First we need to prepare our scene. Create the structure as shown in the screenshot pay attention that every object its position set to (0,0,0). Create two Cameras. Also remove the "Main Camera"	
Next, add our layers that are needed for our Scale Space Layers.	
Now let's setup our "Local Camera" And change the Clear Flags to "Depth only" Also change the Culling Mask and remove the "FarSpace" layer.	
Add a Component "Scale Space Camera" to the Local Camera and change values like in the screenshot. Also attach the "Local Space" game Object to the "Container"	
For the Far Space Camera just change the Cullung Mask to only "FarSpace" and also remove the Component "GUI Layer" and " Audio Listener"	



Add a Component "Scale Space Camera" to the Far Space Camera and change values like in the screenshot. Also attach the "Far Space" game Object to the "Container"	
In the last step we Add the "Scale Space Manager" Component to the Scale Space Manager gameObject and set the size of the Cameras array to 2. Now link the Local Camera to slot 0 and your Far Space Camera to slot 1.	
This is the basic setup to get the Scale Space running. In addition to have something visible you can just add an object to your Local Space and add the "Cam Mover" Component from the Demo scene to your local camera to have some navigation.	
Add an Object like a Capsule and put under the Local Space container. Change the values for position and scale as shown in the picture.	
Also add the "Cam Mover" Component from the Demo Scene to the Local Space Camera. Now you can Move the camera with WASD and zoom in and out with the mouse wheel.	



## 4 Components

### 4.1 Scale Space Manager

The Scale Space manager represents the entry point when adding new object via code. It also controls the cameras.

AddObject(gameObject)	Adds an existing object to the observation of the Scale Space. It will automatically put it on the correct layer. Assuming that the current position is related to the space of the first layer called Local Space. It also takes the shift into account when using floating origin. It adds a new Behavior called "ScaleSpaceObject" that is needed.
DestroyObject(gameObject)	Checks if it a controlled object and will remove it from the context. It calls also Destroy.
MoveCameras(Vector3)	Takes the velocity that the local Camera needs to be moved. The other Cameras gets also moved accordingly to their scale
RotateCameras(Quaternion)	Rotate all cameras.
PrescanLayers	If it is activated the Manager will scan all layers for existing objects in the scene that are directly under the corresponding Layer Container. (called on Awake)
Cameras[]	Represents a sorted list of Scale Space Cameras that will be used. Cameras[0] will be the Local Camera. They need to be assigned manually in the Unity Editor

### 4.2 Scale Space Camera

The Scale Space Camera manages all objects on his Layer. It checks if an object gets over the distance threshold and will move it to the next layer (lower or higher). If enabled it tracks also the distance for the floating origin mechanism. It needs to be attached to an object with a camera. For every Camera Behaviour the Clear Flags needs to be set to "Depth only" except for the last one in the ScaleSpaceManager.Cameras array, this one need to be set to "Skybox" or similar. Every Camera should also only have its own layer rendered for this set the Culling Mask to the corresponding layer Except for the first (Local Camera) remove also the layer "UI". Finally set the depth descending from near to far camera.

Container	Holds the reference to the Container where all objects get linked to.
LayerName	Represents the name of the layer where all objects get rendered
Scale	Defines the amount of how much an objects gets scaled down or up. Depending of it comes from a lower or higher Scale Layer. The Scale is in relation to the previous Layer and NOT absolute!
TransitionThreshold	Sets the threshold at which distance an object gets moved to the next higher layer
UseFloatingOrigin	Indicates if floating origin should be used for this layer
ShiftThreshold	Sets the threshold at which distance from 0,0,0 the camera gets shifted back.
Shift	Holds the absolute shifting offset when floating origin is active. It is represented in a



	"Vector3Double" for accuracy reason.
ScaleFromOrigin	Represents the absolute scale factor in comparison to the Local Space.
ScaleFromOriginInverse	Represents the absolute scale factor in comparison to the Local Space as inverse. Its useful for performance mathematics
GetPositionInLocalSpace()	Gets the absolute position of the camera in Local Space in "Vector3Double" for accuracy reason.

## 4.3 Scale Space Object

The Scale Space Object is a simple MonoBehaviour that gets attached to every gameObject that is controlled by the ScaleSpaceManager.

FixedOnLayer	Sets if it should stay on a specific layer and will be ignored from the ScaleSpaceCamera and the transition
GetPositionInLocalSpace()	Gets the absolute position of the object in Local Space in "Vector3Double" for accuracy reason.
AddVelocityFromLocalSpace	Adds a specific Vector3 from the Local Space to the object under using the correct scale.

## 4.4 Vector3Double

Hold the position as doubles for more precision.

## 5 Known issues (Work in Progress)

- When floating origin is active and a layer will be shifted, particles can have a gap in there creation due to the interpolation feature from the shuriken system.
- Collision detection / Rigidbody only work for objects on the same Scale Layer.
- Manual does not cover all features
- Code documentation

## 6 Release Notes

### Release 0.1.0

- Added floating Origin per Scale Layer controllable
- Dynamic Scale Layers with independent thresholds
- Floating Origin for game objects, legacy/shuriken particles
- Unity Demo Scene
- Unity Tutorial Scene
- Added a SkyBox for the demo scene