Occlusion Culling and You

Occlusion culling is a complicated topic, and pretty easy to misunderstand. With this guide, I'm hoping I can give people a better idea of when and where to use it.

This is based on my guide at https://gitlab.com/s-ilent/SCSS/-/wikis/Other/Occlusion-Culling

- The basics of occlusion culling
- How occlusion culling affects objects
- How to set up a scene for occlusion culling
- Case study: Practical tips and tricks for world design
- Final notes

The basics of occlusion culling

Occlusion in a Unity scene is data that divides the world into chunks containing smaller and smaller chunks of space. In each of these sections, the data for what is and isn't visible from each section is recorded. Unity takes this data and uses it to hide things from being rendered.

It's important to remember that occlusion will only hide things visually. Most things will keep on running when hidden, so occlusion culling is strictly for reducing draw calls. Also, Unity won't hide objects it thinks are visible, no matter how far away they are. Occlusion culling doesn't care about distance.

Wait, occlusion culling is only visual?

That's right - occlusion culling only stops things from rendering. It doesn't stop them from doing any processing, so if you have a big world split into sections, you can gain performance by disabling parts you know aren't currently active.

How occlusion culling affects objects

Occlusion culling treats static objects (like the map) and dynamic objects (like players and pickups) separately.

- **Static** objects can hide and be hidden. (You don't always want static objects to do both.) They'll be baked into the data and can't be changed at runtime.
- **Dynamic** objects can't hide objects behind them, but they'll be hidden based on a rough calculation of whether they're behind a static object.

Unity uses the Static flags on an object to determine how occlusion affects them. There are 3 different things that occlusion thinks about.

- Static Occlu**ders**. The only things that can occlude are static things that don't move - specifically, Mesh Renderers and Terrain. If they're marked as Occluder Static, then things behind them get hidden. They'll be baked into the occlusion data.
- Static Occlu**dees**. This specifies static things that get hidden. Occludees can be anything with a Renderer, as long as it doesn't move.
- Dynamic Occludees. This covers anything else in the scene. If the "Dynamic Occlusion" property is enabled, then it can be hidden by Static Occluders using the baked data.

Dynamic occlusion is less precise than static occlusion. In practice, only large, solid objects will hide dynamic objects.

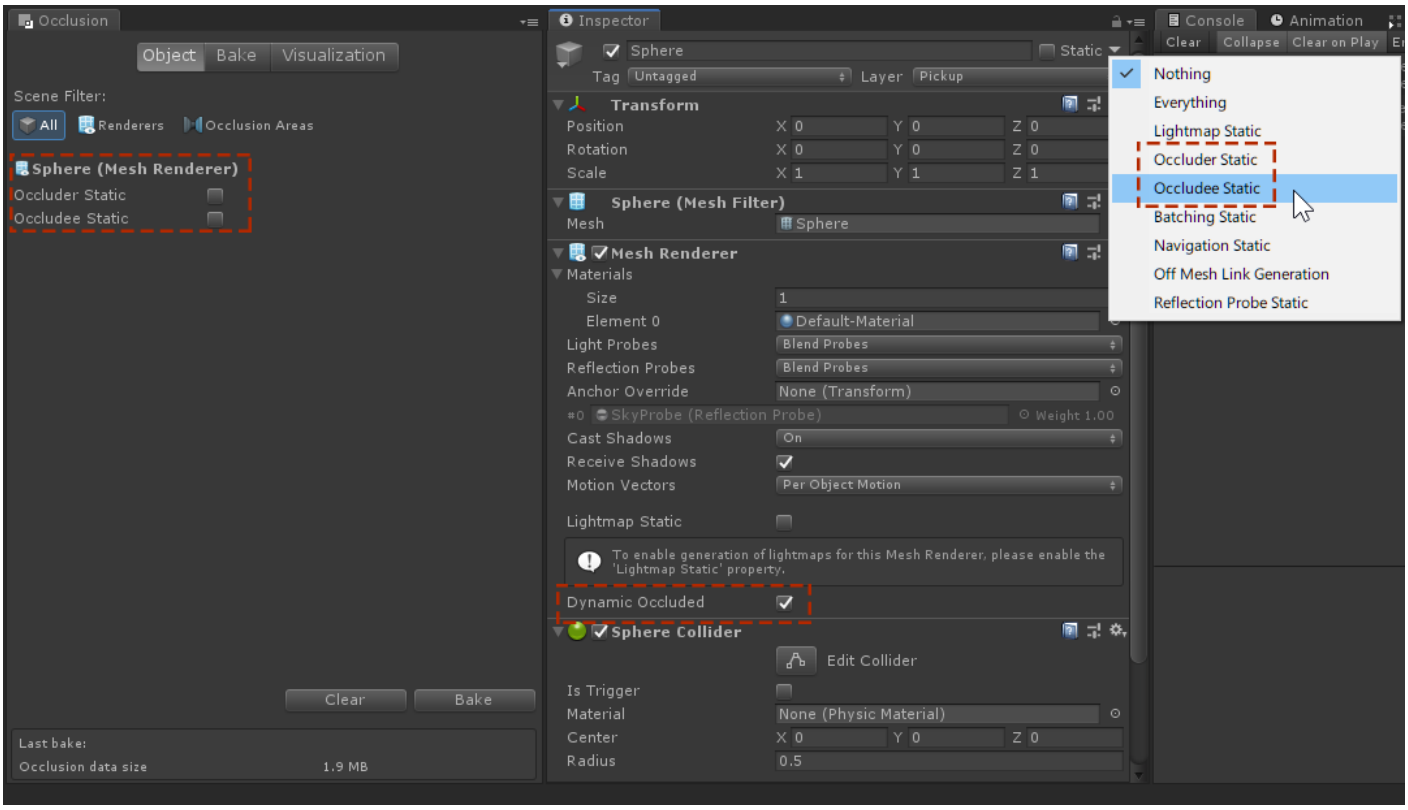
Occlusion culling works on top of static batching, and Unity can hide objects that are part of a static batch.

How to set up a scene for occlusion culling

By default, Unity will put the entire scene into one big chunk of space, called a View Volumes.

If you place an **Occlusion Area** into the scene, Unity will only calculate occlusion data for the inside of that space, which can make the occlusion data much smaller. If your scene is split into multiple seperate areas, consider covering them with Occlusion Areas instead of letting Unity calculate the entire scene as one big volume.

You can set occlusion settings from the Static flags on an object, or the Occlusion window.



Simple recommended settings for occlusion culling...

Only big, solid things should be occluders!

This is a simple rule of thumb. If it's small, or transparent, or it has a hole in it, it shouldn't be an occluder. Trees shouldn't be occluders because they have leaves which create lots of tiny holes which can't be represented by the occlusion. Complex edges are bad.

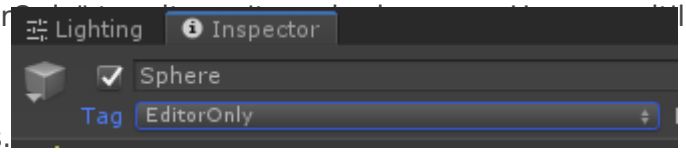
If it's always visible, don't make it an occludee!

If you have a skybox, or an ocean made of a gigantic plane, there is no point in allowing it to be occluded. The number of false positives will exceed the performance gain.

Make smaller objects occludees, but not occluders.

Just because something is small doesn't mean it won't benefit from occlusion culling.

Use the EditorOnly tag to create fake occluders!

If an object has the "Editor

ll still be considered during the baking process.

How do I preview occlusion culling?

In Unity 2018, occlusion culling affects the Scene camera. Besides that, if you select **Visualization** in the Occlusion window, you can see how the occlusion culling works from the perspective of cameras in the scene.

When you select Visualization, the scene Hierarchy will be *filtered* to only show objects with Camera components, and the occlusion culling for the currently active camera's perspective will be shown. You can see the raycasting used by the camera and the low-res scene the rays are cast against to determine what is being occluded and where.

To go back to normal, switch to the other tabs in the Occlusion window or close it.

What do the "Bake" settings mean?

In the Occlusion window, there's a "Bake" section with some settings and a block of text. As the text notes, you don't really need to change the settings because they're already tuned for a good balance between speed and performance.

Smallest Hole sets the threshold for a gap in geometry to be considered a "hole" which can be seen through. The smaller this is, the longer it will take to generate occlusion. Set this to a value that suits your scene, ideally between 0.05-0.5. (5cm to 50cm)

Smallest Occluder sets how much detail the baked occlusion data will take into account. The smaller this is, the more small things will be taken into account when calculating the occlusion, and the more calculations the occlusion will do in runtime when calculating what's visible. Set this to a value slightly larger than the average player's size, so 2-6. (2m to 6m)

If you want to know more about them, visit this article on the Unity blog which explains them in more depth.

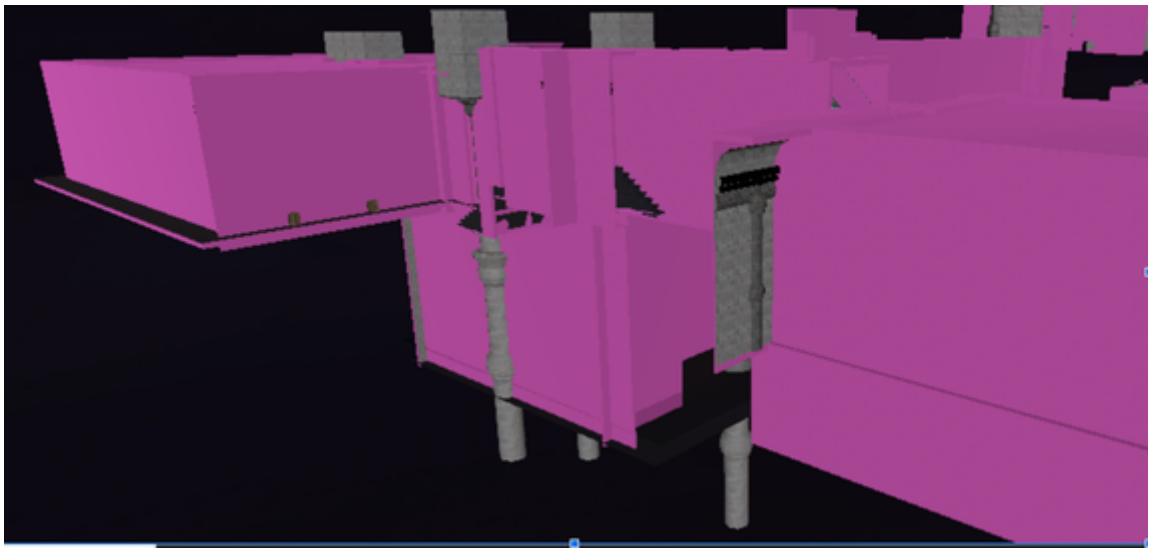
Case study: Practical tips and tricks for world design

The Devouring is a large map in VRC that uses occlusion culling to squeeze the most performance out of Unity. Here is some advice from its developers.

- Don't forget, occlusion is calculated using Opaque mesh renderers as occluders. Transparent materials will not occlude. Complex edges are bad.

- If you want to optimize your bake for a large and complex world, especially one with "zones", add big occluder cubes between areas that should not be mutually visible. Make these EditorOnly.

- One *fun* way to do occlusion is to not make any of your actual meshes occluder static, but only make occluders out of cubes, and use one obnoxious material for all these occluders (and set those objects to EditorOnly). This avoids issues that the occlusion baking has with more complicated geometry.



The occlusion prefabs used in The Devouring. Take note that they are behind the walls.

If those objects are visually distracting in your scene, change the shader on the material to an "invisible" shader as a way to switch off your occluders while working. (Remember to switch it back to an opaque shader for baking.)

In this case, they are just behind the wall, and same width. For the decorative pillars, they would be inside — for example, if you have a little jog/indentation in a pillar or wall, you don't want to have blinks when you look just at that area. In general, make your fake occluders a little smaller than exposed edges.

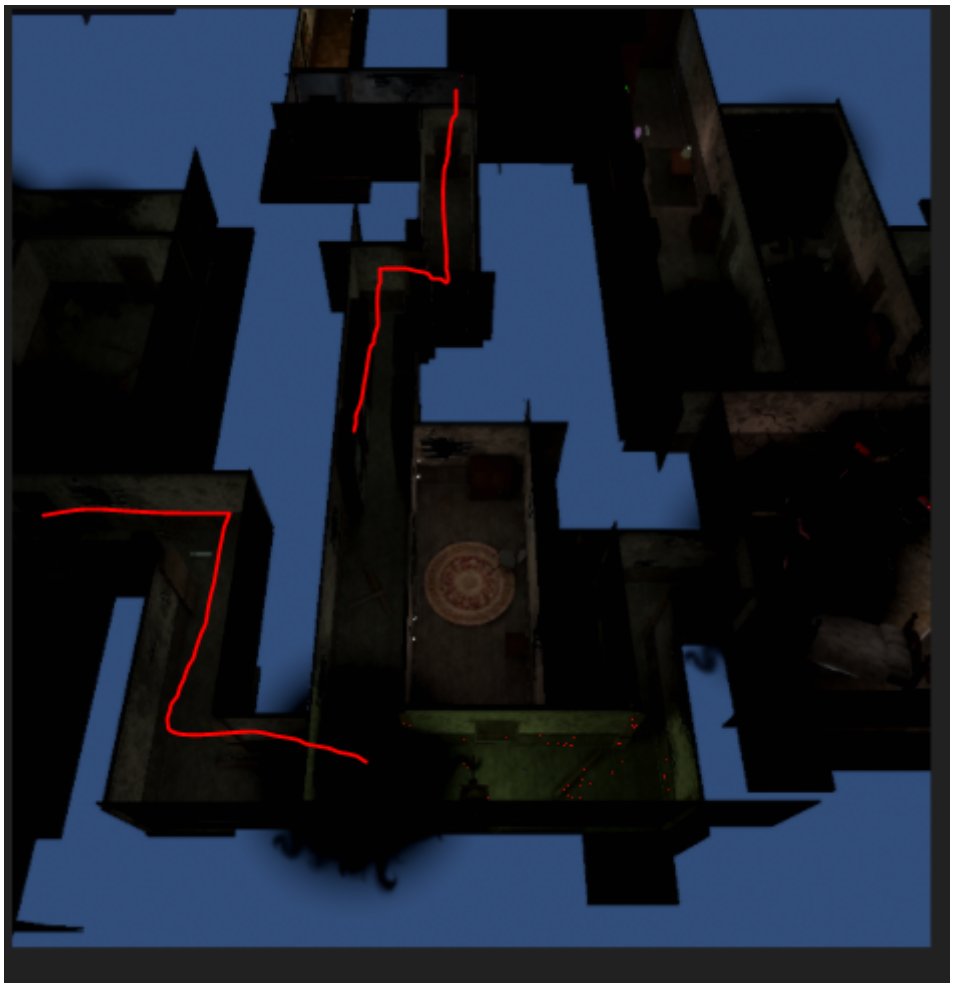
If you can see pink on the inside of your scene, you know you have a place where you might get false positive occlusion. If you don't see pink on the outside, you know that's a gap in your occlusion.

- The Devouring was made of modular assets. All the walls had occluder boxes in them as part of the prefab, which simplified the occluder setup, baking, and eliminated a lot of occlusion errors. You could also see them as pink walls in the scene and fairly easily visualize where they were working. This was combined with giant occluders in a separate GameObject, enabled only for baking.

- Occlusion areas are great for large maps in chunks, but beware that in 2018 you will sometimes get "blinks" in VR when passing between areas. Very frustrating, but we could find no way around it.

- Unity 2018 is very frustrating to work with when occlusion is baked, as it hides things from you in the editor. Change your bake settings to something less defined while working to make it easy to clear and rebake quickly. Do a more detailed bake once you are ready to optimize.

- Think about occlusion when designing worlds. If you don't need an open space, throw a wall there. Add corners and zigzags into hallways to occlude the other side. Block up big open views if they're not showing things off. Try to keep the player from seeing too much at one time — this is especially true for party instances, when occluding players is one of the biggest performance boosts you can make.



An example from The Devouring of occluding corridors.

Final notes

Occlusion Portals

Occlusion Portals are occluders that can be turned off and on and affect occlusion. If you have a large object in your scene that can open/close, like a door or toggle-able object, you can place an occlusion portal alongside it and open/close the portal to get the benefits of occlusion culling when the portal is closed.

References

Special thanks to Fionna for providing the details on The Devouring's occlusion planning. Further details were presented in the VRCPrefabs TLX videos.

For more information on how occlusion culling works, try the Unity documentation.

When I first read it, I didn't understand it at all. I learned a lot through trial and error. In the past year, Unity have also been taking feedback and reworking parts of their manuals to make more sense, which makes the documentation a lot more readable as a result!

- https://docs.unity3d.com/Manual/OcclusionCulling.html
- https://docs.unity3d.com/Manual/occlusion-culling-getting-started.html
- https://www.gamasutra.com/view/feature/164660/sponsored_feature_next_generation_.php
- http://web.archive.org/web/20131204230947/https://blogs.unity3d.com/2013/12/02/occlusion-culling-in-unity-4-3-the-basics/
- https://blogs.unity3d.com/2013/12/26/occlusion-culling-in-unity-4-3-best-practices/
- http://web.archive.org/web/20140105091320/https://blogs.unity3d.com/2014/01/02/occlusion-culling-in-unity-4-3-troubleshooting/
- https://medium.com/@Umbra3D/introduction-to-occlusion-culling-3d6cfb195c79
- https://cdn.discordapp.com/attachments/437816654492073994/924630771367559178/boosting-frame-rates-in-unity-games-using-umbra-3.pdf