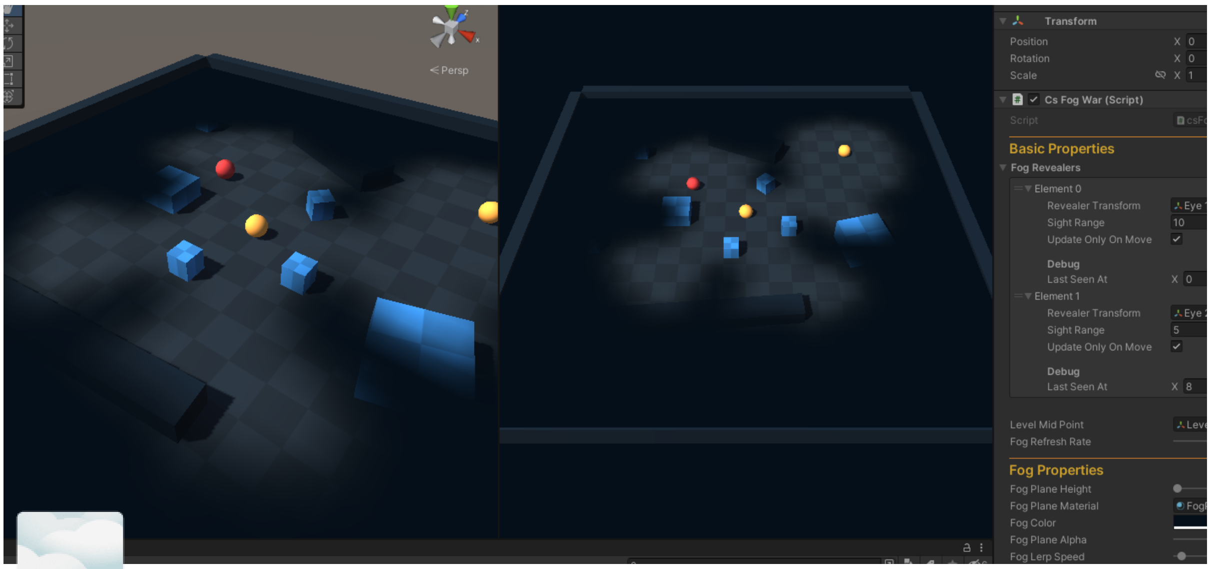


Transform
Position X 0
Rotation X 0
Scale X 1
Cs Fog War (Script)
Script csF
Basic Properties
Fog Revealers
Element 0
Revealer Transform Eye
Sight Range 10
Update Only On Move ✓
Debug
Last Seen At X 0
Element 1
Revealer Transform Eye
Sight Range 5
Update Only On Move ✓
Debug
Last Seen At X 8
Level Mid Point Leve
Fog Refresh Rate
Fog Properties
Fog Plane Height
Fog Plane Material Fog
Fog Color
Fog Plane Alpha
Fog Lerp Speed

Docs - AOS Fog of War (v1.2)

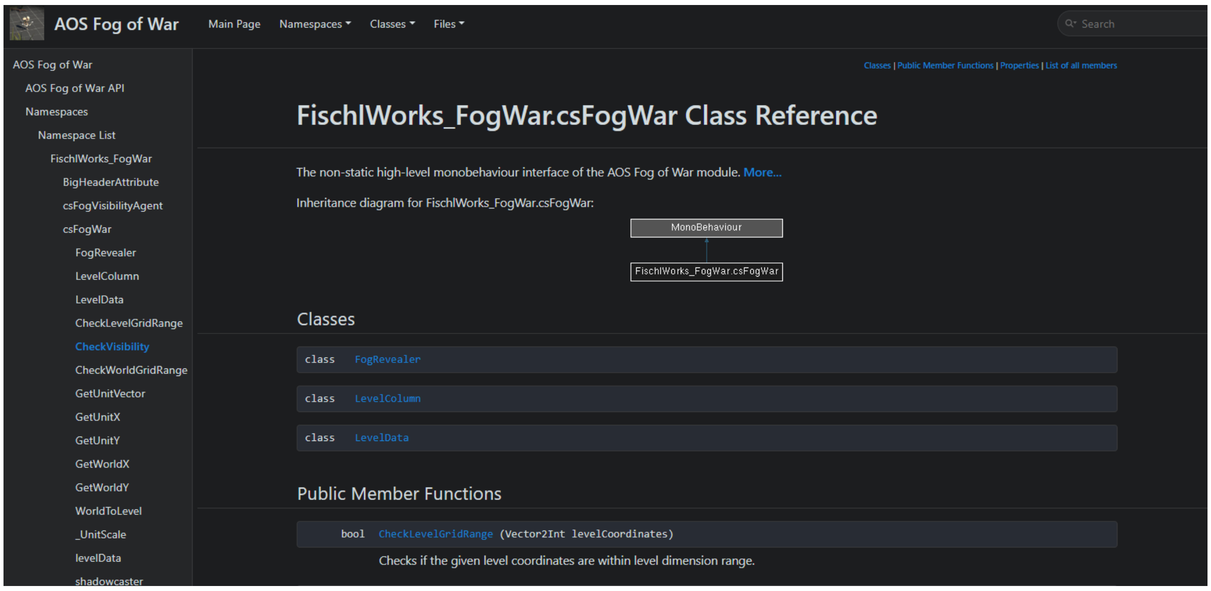
📽️ Youtube Tutorial

https://youtu.be/MnER3bD7LbA

⚙️ How it works

https://youtu.be/0xB7C-jrI9I

🛠️ API



FischlWorks_FogWar.csFogWar Class Reference

The non-static high-level monobehaviour interface of the AOS Fog of War module. More...

Inheritance diagram for FischlWorks_FogWar.csFogWar:

Classes

```
class   FogRevealer
class   LevelColumn
class   LevelData
```

Public Member Functions

```
    bool  CheckLevelGridRange (Vector2Int levelCoordinates)
```

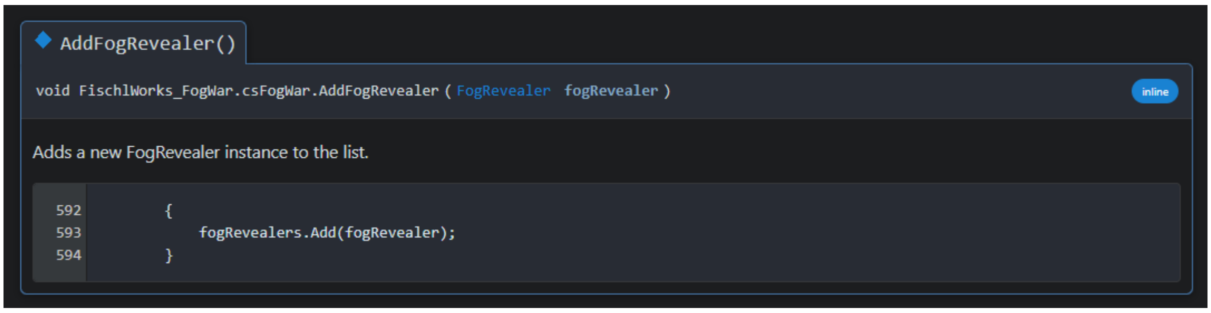
Checks if the given level coordinates are within level dimension range.

AOS Fog of War: AOS Fog of War API

https://fischlworks.github.io/

💬 Common Questions

Q: How do I add a revealer through code during runtime?


AddFogRevealer()

```
void FischlWorks_FogWar.csFogWar.AddFogRevealer ( FogRevealer  fogRevealer )
```

inline

Adds a new FogRevealer instance to the list.

```
        {
            fogRevealers.Add(fogRevealer);
        }
```

AOS Fog of War: FischlWorks_FogWar.csFogWar Class Reference

https://fischlworks.github.io/class_fischl_works___fog_war_1_1cs_fog_war.html#a41589c87dadd842ae4b8227d54fdf33b

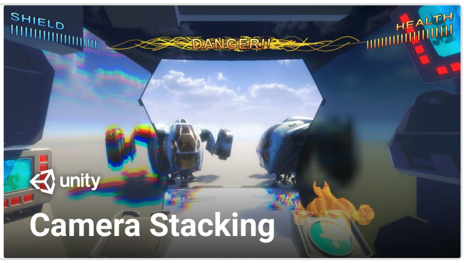
A : While I added a revealer inside the tutorial video the primitive way just for demonstration, you can use the public ***AddFogRevealer()*** interface for that problem.

Q: How do I overlay an object / UI element over the fog?

Camera Stacking in Unity with URP! (Tutorial)

In this video we're going to take a look at Camera Stacking with the Universal Render Pipeline, in short URP, using Unity! Using Camera Stacking you can layer game UI in Unity very quickly,

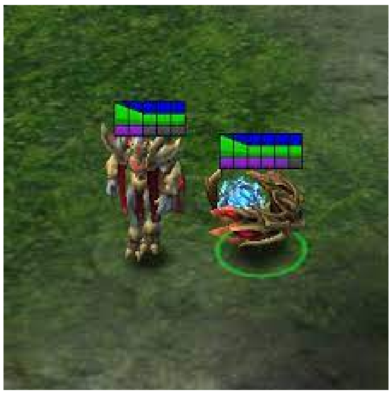
https://youtu.be/OmCjPctKkjw



Camera Stacking | Universal RP | 7.2.1

https://docs.unity3d.com/Packages/com.unity.render-pipelines.universal@7.2/manual/camera-stacking.html

A : For non-world space **canvas-based UIs,** they will be drawn after the fog, so no worries.

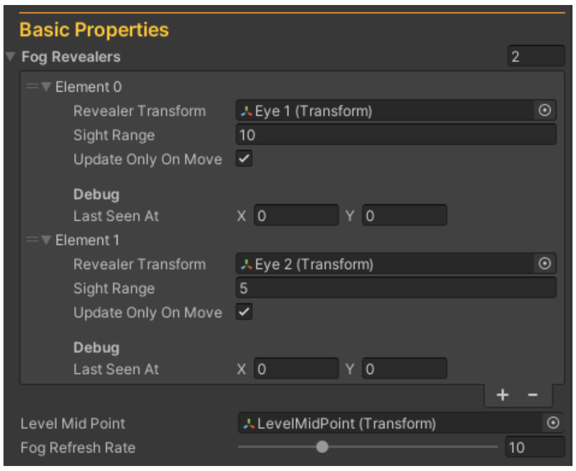


However, for some certain **mesh-based instances** that lies over the world space, (ex) healthbars, 3D UI elements…) you will have to use a technique called **Camera Stacking** of unity**.** Above are official tutorial and documentation for that feature.

Keep in mind that there are multiple fancy techniques that can be utilized to achieve the same behaviours, but the camera stacking technique is the simplest, most effective, and easiest option that is easy to maintain.

⚙️ Adjustable Properties

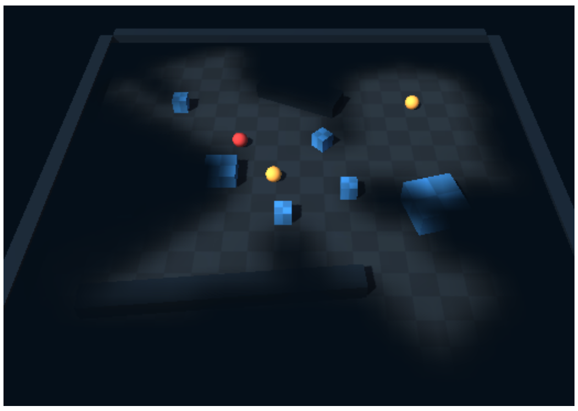
Basic Properties



Basic Properties
Fog Revealers 2
Element 0
Revealer Transform: Eye 1 (Transform)
Sight Range: 10
Update Only On Move: ✓
Debug
Last Seen At: X 0 Y 0
Element 1
Revealer Transform: Eye 2 (Transform)
Sight Range: 5
Update Only On Move: ✓
Debug
Last Seen At: X 0 Y 0
Level Mid Point: LevelMidPoint (Transform)
Fog Refresh Rate: 10

Fog Revealers are entities that provides vision to the player, which usually refers to player characters, minions, turrets, or wards. Having zero revealers is OK.

Note that through ***AddFogRevealer()*** public method of ***csFogWar,*** you can add more entities as the revelaers dynamically during runtime.

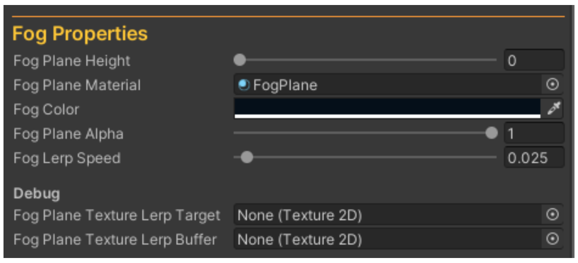


Changes in **Sight Range** property during runtime is OK, and the **Update Only On Move** property will stop the module from being updated (it will just return shortly after each ***Update()*** call) when the player is not moving.

Level Mid Point property **MUST** be set to the middle of the level, and it is recommended that it is not moved during runtime. All scans and unit conversion happens around this specific transform's position, so setting it right is very important.

Fog Refresh Rate designates the number of checks that is done per second. It is based on ***Time.deltaTime***, so it is independant from target framerates.

Fog Properties



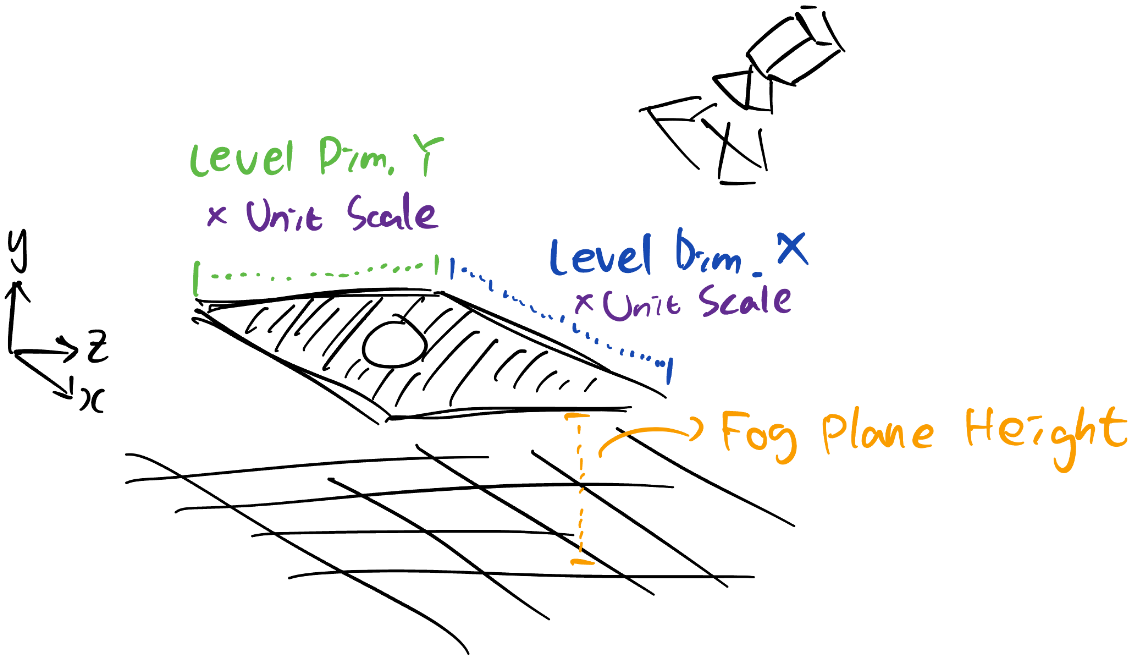
Fog Properties

Fog Plane Height	0
Fog Plane Material	FogPlane
Fog Color	
Fog Plane Alpha	1
Fog Lerp Speed	0.025

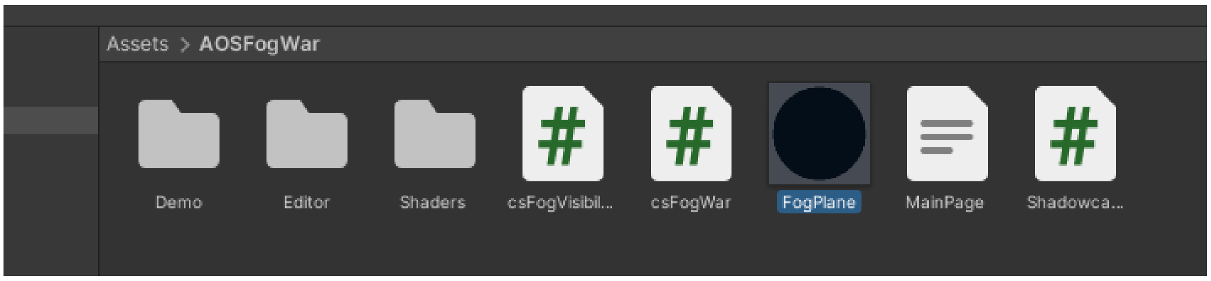
Debug

Fog Plane Texture Lerp Target	None (Texture 2D)
Fog Plane Texture Lerp Buffer	None (Texture 2D)

Fog Plane Height property designates the height of the fog plane.



This drawing may help you understand better.

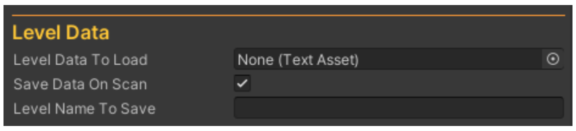


Fog Plane Material is a property section that **REQUIRES** the specific material named ***"FogPlane"*** that comes along with the module to be assigned manually.

With the **Fog Color** and **Fog Plane Alpha** property, you can control the color and the alpha value of the fog plane texture.

Fog Lerp Speed literally designates the speed that the shape of the fog texture changes. The closer it is to 0, the smoother the transition will be.

Level Data



Level Data

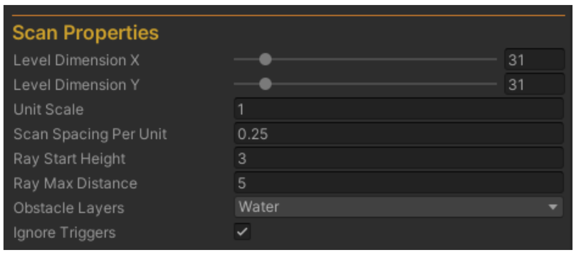
Level Data To Load	None (Text Asset)
Save Data On Scan	✓
Level Name To Save	

Inside the **Level Data To Load** property, you can assign a pre-scanned & saved level data located in ***Assets/LevelData/*** path. The private path name property is called **levelScanDataPath.**

Save Data On Scan option designates wheter to save the scanned data or not. Note that if there are no pre-assigned level data, the csFogWar module will perform a scan based on the **Level Mid Point** property and the **Scan Properties**.

If there's already an existant file having the same name, the file will be overwrited by default.

Scan Properties



Scan Properties

Level Dimension X	31
Level Dimension Y	31
Unit Scale	1
Scan Spacing Per Unit	0.25
Ray Start Height	3
Ray Max Distance	5
Obstacle Layers	Water
Ignore Triggers	✓

Level Dimension X and **Level Dimension Y** designates the dimension of the level. Keep in mind that larger dimension will result in a much heavier computational load for the CPU.

Unit Scale property determines the width and height of each cell, and **Scan Spacing Per Unit** property determines the space between each **BoxCast** passes.

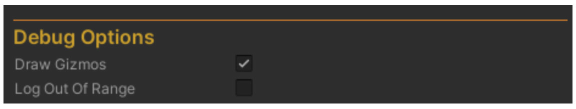
Unity - Scripting API: Physics.BoxCast

https://docs.unity3d.com/ScriptReference/Physics.BoxCast.html

Ray Start Height and **Ray Max Distance** property are also passed onto the ***BoxCast()*** function as parameters, and you may have to optimize this value carefully the level is not flat. Note that the ray is casted downwards.

Obstacle Layers property is used to determine wheter a hit object is an obstacle or not. **Ignore Triggers** option is pretty much self-explanatory.

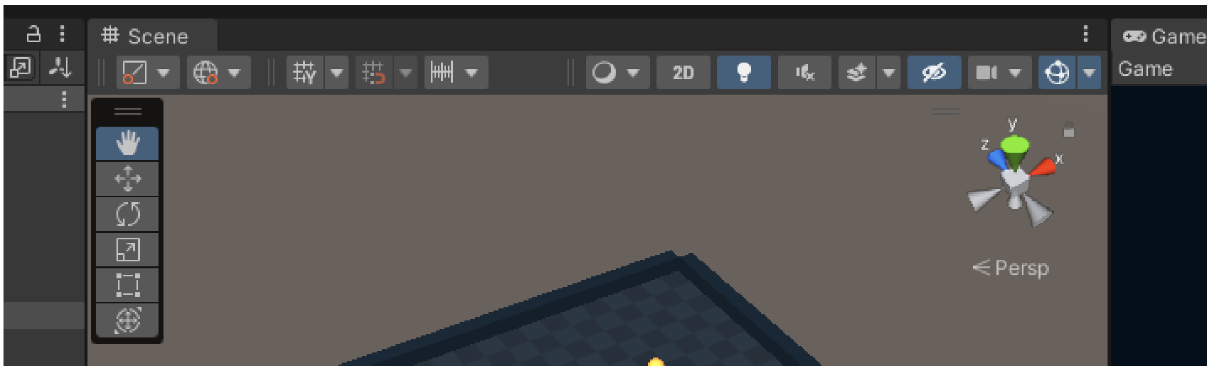
Debug Options



Debug Options
Draw Gizmos ✓
Log Out Of Range

Toggling the **Draw Gizmos** option will let the module draw gizmos in the Scene view. **Log Out Of Range** option lets the module to log all level cell indexing request that is out of grid range.

Drawing these gizmos takes a lot of computational resources, so only enable them if you really want them.



Note that you have to toggle unity's show gizmos option on in the upper-right side of the scene view to actually display the gizmos.

📫 Support

keithrek@hanmail.net

🍻 Contact

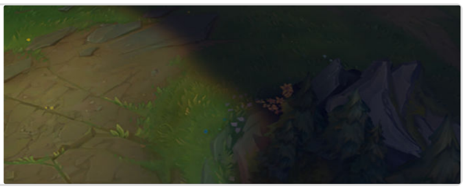
https://www.linkedin.com/in/keithrek/

🙏 Major Credits & References

A Story of Fog and War

http://riot.com/1NvPXyT #RiotTechBlog

https://technology.riotgames.com/news/story-fog-and-war



Dota 2 Interactive Map v5

Dota 2 Interactive Map. View, pan, and zoom in on any part of the Dota 2 map. Turn on various overlays for more information, place wards to see sight ranges, and measure distances.

https://devilesk.com/dota2/apps/interactivemap/?x=-349&y=406&zoom=1&mode=observer&observer=-1485,94;-1960,2520;574,568

Symmetric Shadowcasting

https://www.albertford.com/shadowcasting/

Implementing Fog of War for RTS games in Unity 2/2

As I said in the previous blog post, some time ago I started working on a new Fog of War / Vision System solution aiming the following features: Being able ...

https://blog.gemserk.com/2018/11/20/implementing-fog-of-war-for-rts-games-in-unity-2-2/

유니티 - 전장의 안개(Fog of War)

목차 1. 개념 2. 구현 방법 3. 타일맵을 이용한 구현 4. 구현 결과 5. 프로파일링, 최적화 6. Reference

https://rito15.github.io/posts/fog-of-war/