RealSunAR™

v2.10 (manual revision 1)

Made by Antonis Savvidis

Welcome to RealSunAR a practical sun/shadow solution for your AR app/game.

Made to work for all AR SDKs/Toolkits

Asset import

Go ahead and import everything. You will find a new folder named “RealSunAR” on your Assets folder. This plugin does NOT include any demo scenes because that was impossible due to many possible combinations between Unity and your AR SDK that you are using. However I do write instructions just below to quickly build a testing scene for some popular AR workflows.

Upgrading

Go ahead and completely remove the RealSunAR folder in your assets. Then import the updated version from the asset store.

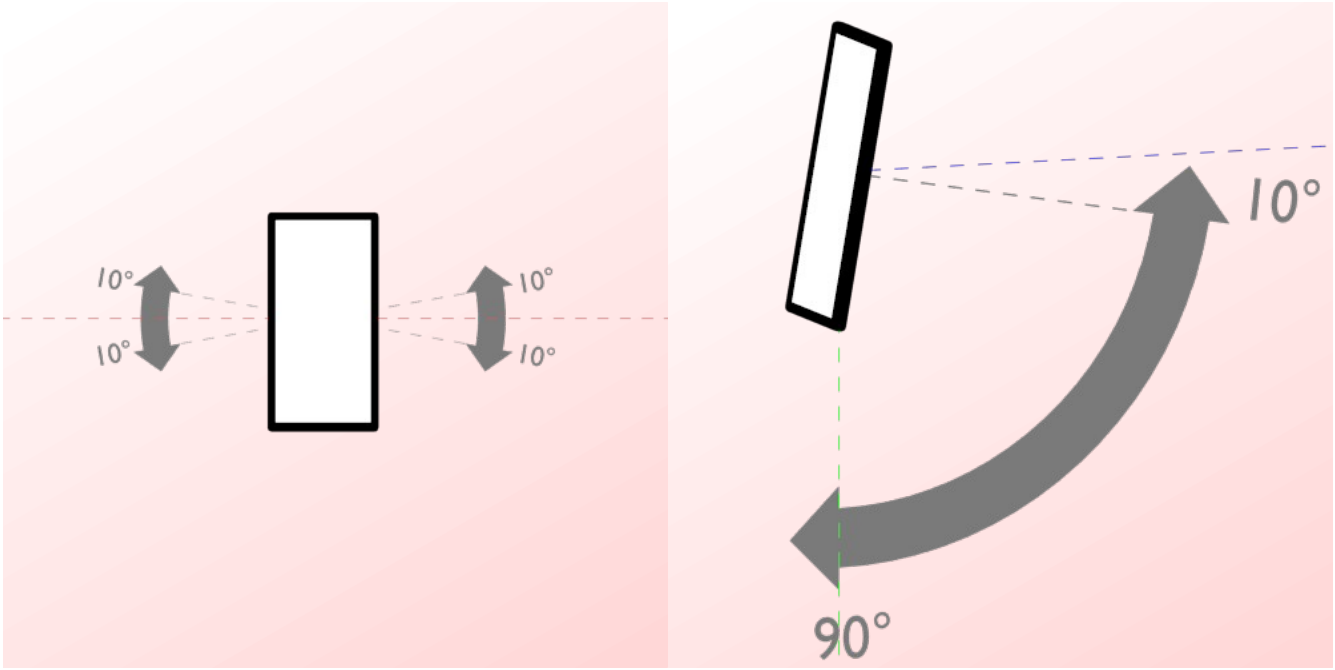
How it works (it’s a kind of magic!)

When the script runs it will try to determine

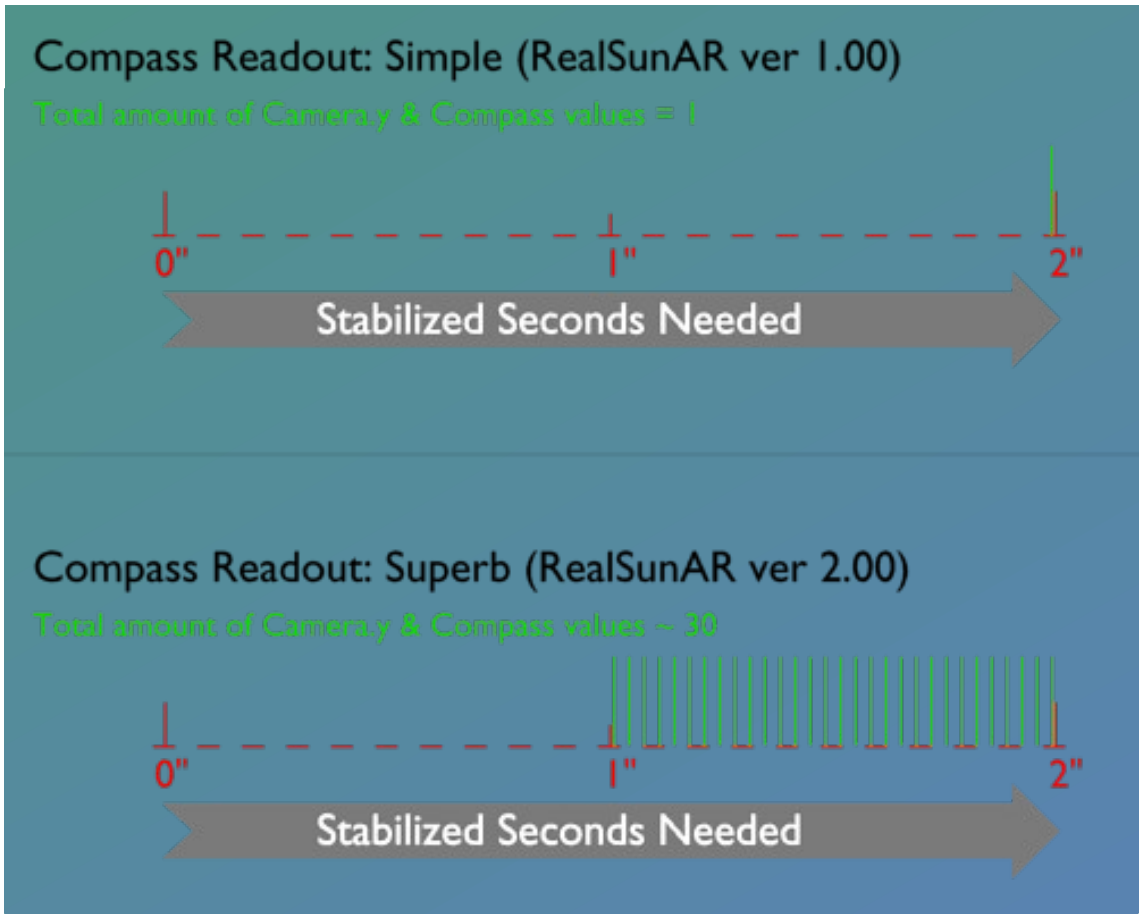
- The users location.
- The location cloudiness (optional)
- The local time & date.
- If the camera started giving usable rotation values
- If the compass is indeed active

Now after the conditions above are met, the engine will try to get it’s initial position of the sun. To do that it needs the device to maintain two conditions for the amount of time set in the **Stabilized Seconds Needed** inspector option**.**

1. Stay unshaken from very sudden movements
2. Stay inside that area of device rotation (for both portrait or landscape modes):



That will ensure a better compass reading. Now since the whole accuracy depends on the Camera.y rotation & Compass values, I created a new algorithm to get more precision and avoid “spiked” values that might occur. Here it is compared to the original method of reading the sensors (default values used):



After that initial measurement is done and the sun directional light is positioned, the engine will wait for "**compass drift timer**" seconds to pass. Then, the engine will go into that whole 2 seconds (or less, depending on the setting) procedure again and gather new data. If the new collected data show a north offset more than "**tolerate drift**" degrees from the active one, the new data will be used and reposition the sun, otherwise those data will be discarded and so on…

Usage

Short version

Drag and drop the "**RealSunAR**" script to a single gameobject in your scene. You will find that script here:

`Assets / RealSunAR / Scripts /`

The default values should work just fine but if you want to play around with them you will see what they do on a later part of this manual. Make sure you have Location permission enabled for you app/game. You might also want to use the material "**RealSunAR_Shadow**" on your floor which acts as an invisible shadow catcher that shows only shadows. That's it!!!

Example: How to create a demo scene for ARCore

Prepare a project for Google's ARCore as instructed on their site.

1. Open the HelloAR scene found in `Assets / GoogleARCore / Examples / HelloAR / Scenes`

2. Drag and drop the "**RealSunAR**" script to a root gameobject in your scene. You will find this here:

`Assets / RealSunAR / Scripts /`

3. Go to the **Plane Generator** gameobject and click on the **DetectedPlaneVisualizer** found on the inspector to go directly to that prefab in the assets.→

4. Now that you are in the **DetectedPlaneVisualizer** go to it's MeshRenderer and expand the materials part. Change the material from PlaneGrid to **RealSunAR_Shadow**

5. Go to the **ARCore Device** gameobject, click on it's Session Config and change the Light Estimation mode to anything **without** Environmental HDR (Environmental HDR tries to create it's own sun direction and fails, but it's great for interior applications).

Make sure you have Location permission enabled for you app/game. Also remember to tag your camera as **MainCamera**

Example: How to create a demo scene for AR Foundation

1. Open up the sample scene “SimpleAR” on the project found here.

2. Delete the “Directional Light” gameobject.

3. Drag and drop the “**RealSunAR**” script to a root gameobject in your scene. You will find this here:

`Assets / RealSunAR / Scripts /`

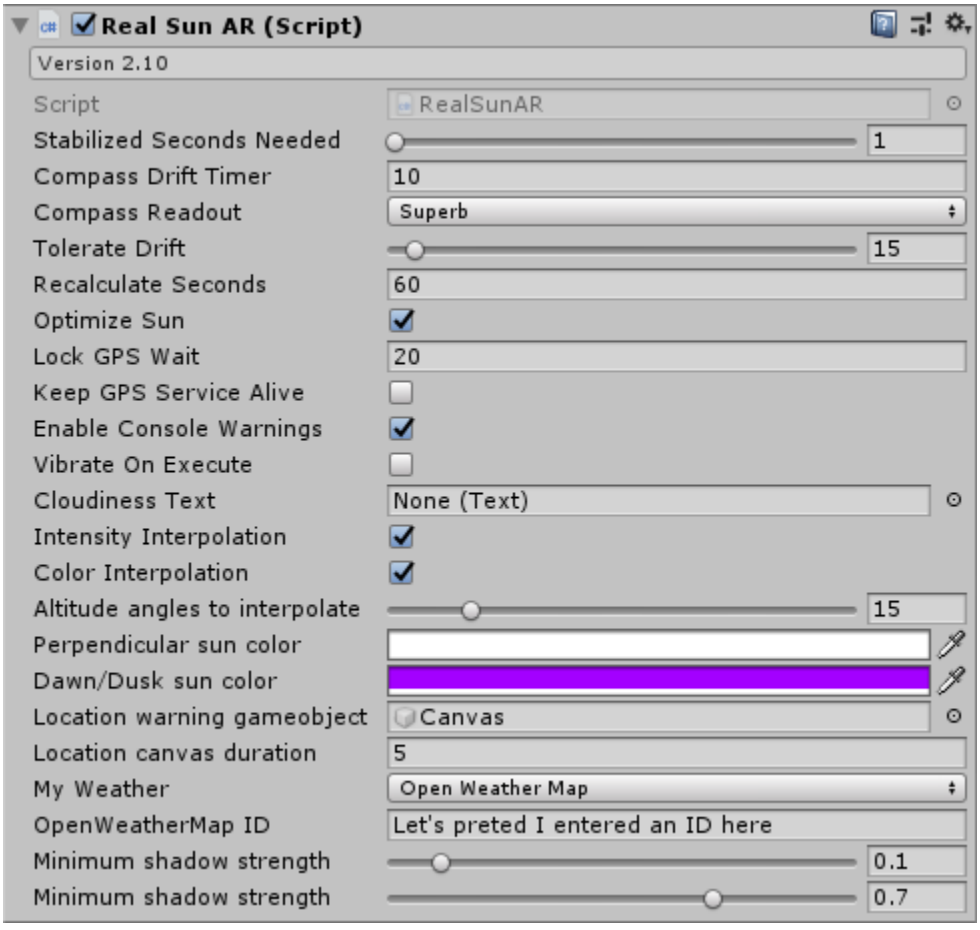
4. Go to the “AR Session Origin” gameobject and look at it’s inspector. Click on the “AR Plane Debug Visualizer” inside the “AR Plane Manager” component (or just look up “AR Plane Debug Visualizer” on the Project’s search bar.

5. Now that you are on the AR Plane Debug Visualizer go to it’s “Mesh Renderer” component and change the material from “PlaneMaterial” to “Realness_Shadow”

Make sure you have Location permission enabled for you app/game. Also remember to tag your camera as **MainCamera**

Usage (the detailed version)

All you need is one script, “**RealSunAR**” found here



`Assets / RealSunAR / Scripts /`

Notice that this script creates a light when attached to a gameobject. This is the light that will be used for the sun.

Stabilized Seconds Needed: The users phone must not produce values above 0,15f on the `Input.gyro.userAcceleration` on any axis for that many amount of consecutive seconds to make sure the phone is not shaking while measuring the compass. Also if “Superb” is used as compass readout method, the last half of this time will be used to gather data once per frame.

Compass Drift Timer: Every time this amount of seconds pass, new data will be gathered regarding camera.y rotation & compass in order to calculate if there was a compass drift or not.

Tolerate Drift: If the newly gathered data show a North-to-camera offset that exceeds that many degrees from the active one then those data will be used, otherwise they are discarded.

Recalculate Seconds: The amount of seconds it will take for the RealSunAR engine to reposition the sun according to compensate for it’s new orbital position due to time passed.

Optimize Sun: If checked, the script will make sure that the light attached to the same gameobject is directional, produces soft shadows and is set to very high shadow resolution.

Lock GPS Wait: Amount of time (in seconds) that the app will wait for a GPS lock before giving up.

Keep GPS Service Alive: If checked the GPS service will remain active even after RealSunAR figures out where the user is. Use this in case you have other scripts needing to access the GPS service.

Enable Console Warnings: This will display warnings & tips on the console. Take note that the warnings also show up on the inspector of the script either way.

Vibrate on Execute: Each time the RealSunAR engine positions the sun (besides time recalculation), it will make the device vibrate. Used for debugging.

Cloudiness Text: If filled with a TextUI component it will display the current cloudiness with the format of “Cloudiness =xx%”. Leave empty if you do not plan to use this.

Intensity Interpolation: This will force the sun to lower it’s intensity during dawn and dusk depending in it’s altitude.

Color Interpolation: This will force the sun to change color during dawn and dusk depending in it’s altitude.

Altitude degrees to interpolate: if any of the interpolations above are selected this field sets the threshold (in degrees above horizon) in which the sun goes from intensity 0-1 and also changes color (optional).

Perpendicular sun color: The color that the sun gets when it's above the degrees specified above.

Dawn/Dusk sun color: The color that the sun gets when it's on horizon level.

Location warning gameobject: Game object that gets activated if Location service or location permission is not active (usually a warning canvas). Leave empty if you do not plan to use this.

Location warning Duration: How many seconds will this gameobject stay active until it's deactivated.

My Weather: Weather API that requires internet use to figure out the cloudiness of the current location.

Weather API_ID: Enter your weather ID in case you use the **My Weather** option (if set to none it will also force MyWeather = none (used only with the **My Weather** option).

Shadow MIN & MAX: Shadow strength will be calculated using a linearly interpolation between those values (used only with **My Weather**).

Note: In case something goes wrong with the shadow strength (like there is no internet connection etc, the shadow strength will be the value of the light currently attached on the gameobject.

Public variables

```
public bool killSwitch = false
```
If set to **true** it will block the engine for doing anything more than determining the location.

```
public bool hasInitialized = false
```
If set to **false** the engine will go into it's data gathering procedure and position the sun using that data. (this becomes true after the first positioning of the sun is complete).

Public methods

```
public void Reseter()
```
This method will reset the current rotation to the original rotation of the gameobject.

Secret debugging mode

Just remove the "//" characters from line 2 of the RealSunAR script `and watch the screen of the device fill up with data for statistics maniacs!`

Shadow Catcher Material:

This material while transparent, it will catch the shadows cast into it, thus making it ideal to be used on your plane floors on an AR environment. You will find it here:

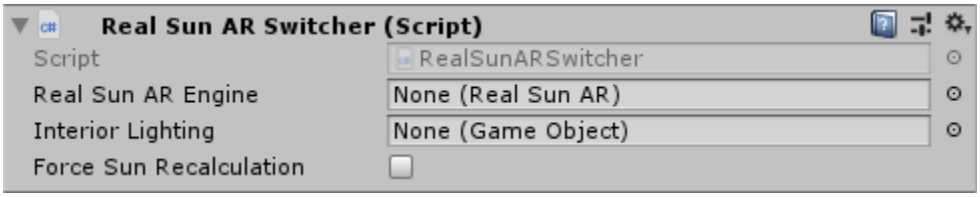
`Assets / RealSunAR / Materials /`

Additional Tool

RealSunAR Tester: If you do not own RealSunAR and you are just browsing the manual you could always try to use the free RealSunAR Tester tool to figure out if RealSunAR will work for you!

RealSunARSwitcher: This is a helpful example script that could be used to switch your app/game from sun light to internal light and vice versa. You will find it here:

`Assets / RealSunAR / Scripts /`



Real Sun AR Engine: Drag and drop your RealSunAR gameObject here.

Interior Lighting: (Optional) This gameobject here will be activated when switching off from the sunlight.

Force Sun Recalculation: If this is not checked, then when the lighting setup returns to RealSunAR it will use the angle it did when it first initialized, if checked it will force RealSunAR to recalculate the sun's position each time the "**RealSunARSwitcher"** returns to the Sun mode.

Public methods

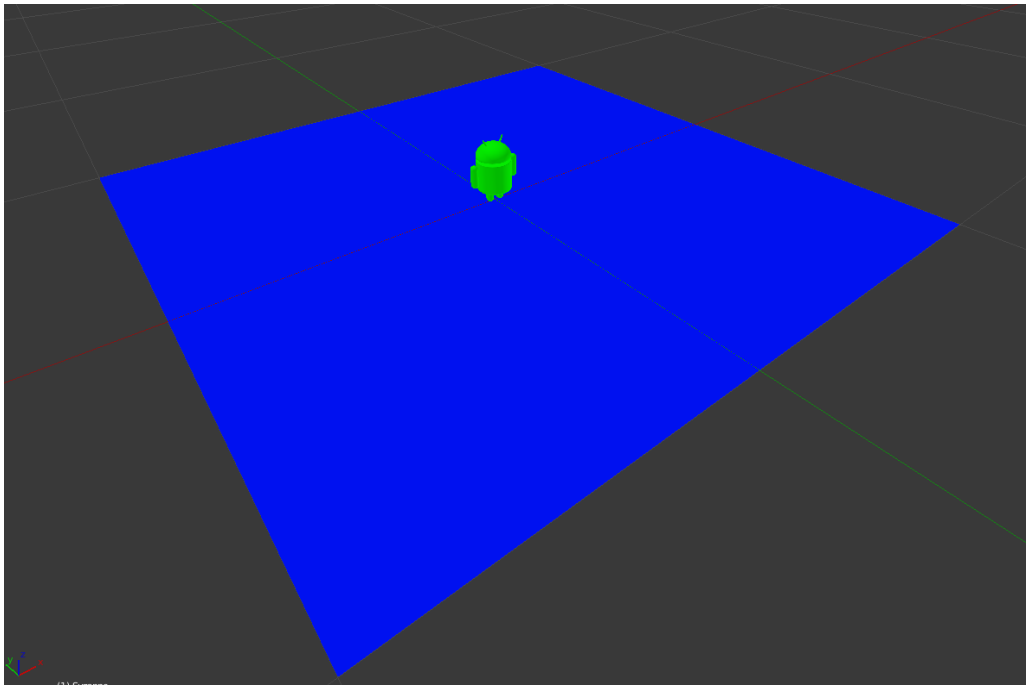
```
public void Disable_Sun_Enable_Lamps()
```
This method will disable the sun and activate the internal lighting gameobject (optional).

```
public void Disable_Lamps_Enable_Sun()
```
This method will disable the internal lighting gameobject(optional) and re-activate the sun.

Designing assets

There are two ways in which you could create your AR models ready for RealSunAR shadows

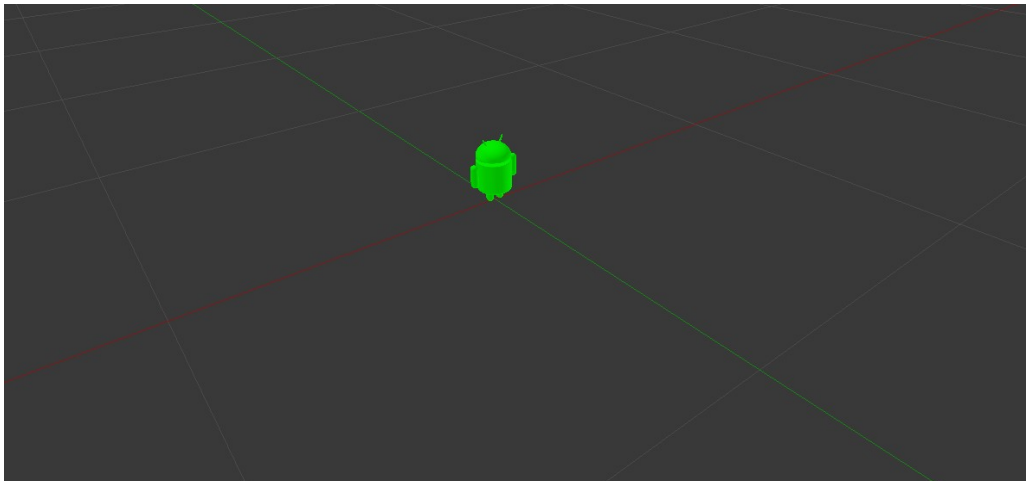
1. Create models that come with their own floor. Something like this:



Apply the “**RealSunAR_Shadow”** material to the blue surface and create a model (Green area) as you would normally do a model for AR.

OR

2. Do what you always did for creating a model



and make sure that any kind of plane generator your AR kit has, uses the RealSunAR_Shadow material**.**

Known Issues / Considerations:

- Some (usually legacy) shaders don't produce nice shadows
- Consider going to the quality settings and setting shadows to **stable fit.**
- Feel free to set the GPS accuracy to Low Accuracy (android player settings).
- Consider being a good person to others. Strive to be the best version of you!

Probable causes for RealSunAR not working:

- The app has no location permissions enabled.
- The compass gets influenced by the surrounding environment and gives out constant bad readings.
- Your AR engine lost track of the Camera.y rotation. Nothing to worry about here, just wait a bit and the compass drift detection will perceive that error in around 10 seconds and re-alight the sun properly.

Compatibility:

This plugin was submitted using Unity 2018.4.0 but has been verified to work with all more recent versions. You might face a small issue using an older version where the prefab is broken but this doesn't use any special settings or anything. Using the RealSunAR script alone on an empty root gameobject does the exact same thing.

Support

Product page

Forum page

Android demo

Email (feel free to use that for any question)

Developer LinkedIn

Special Thanks to my play testers

Himanshu Gupta

Abigail Mathews

Martin Wilter

Berenice Terwey

Magda Krystalli

Change log

RealSunAR 2.10 (changes since version 2.00)

ADDED: Android will now request coarse location permission so that the end user doesn't have to enable it manually.

ADDED: Inspector Setting to interpolate sun intensity according to it's altitude angle.

ADDED: Inspector Setting to interpolate sun color according to it's altitude angle.

ADDED: Inspector Setting that keeps the GPS service on after RealSunAR starts (maybe your own scripts demand GPS services too)

ADDED: Inspector help URL button on RealSunAR's gameobject component now links to the PDF manual.

ADDED: Script to exchange RealSunAR light to artificial light and vice versa.

ADDED: Singleton design doesn't allow more than one instances of RealSunAR (kudos to "dokterdok" for finding that bug)

FIX: RealSunAR would not fire up timed orbit recalculation in some cares

CHANGED: Location canvas is now deactivated instead of destroyed

IMPROVED: Script inspector now hides the properties that are not to be used according to other inspector settings.

IMPROVED: Script tool-tips.

IMPROVED: Minor GPS energy save improvement.

MANUAL: Improvements.

RealSunAR 2.00 (changes since version 1.00)

ADDED: Compass drift detection makes agnostic mode usable to ALL AR SDKs out there! All needed is a usable y-rotation value on the Main Camera but I can guess that all AR SDKs provide that.

ADDED: New "Superb" routine that gathers multiple data and then smooths out using a sophisticated algorithm to get rid of compass spike readings

ADDED: Debug mode (just remove the "//" character from line 2 of RealSunAR.cs and watch all that info flow the screen!)

ADDED: Editor script will add some nice boxes with warnings on the RealSunAR inspector

FIX: Every time RealSunAR calculated the sun position in would go into a deeper child of a new gameobject

FIX: Camera tilt sensor 2nd part should be z instead of x.

FIX: recalculateSeconds > 0 became timer > recalculate seconds

FIX: Counter for calculations will now get to zero if interrupted by phone tilt

FIX: if compass failed to get a reading the engine wouldn't retry again

FIX: Adds a routine to delay activation until camera rotation unfreezes (that was 1.5sec on my device)

IMPROVED: Changed default tilt values for stable phone. x rotation is now 10-90 and z rotation still 10

IMPROVED: Setting now available for canvas deactivation timer

IMPROVED: I added a 100ms delay on "Vibrate on Execute" in order to make sure it doesn't effect gyro/compass/camera

IMPROVED: Turned stable frames options into seconds. It makes more sense that way

IMPROVED: Compass readout added on Start() & GPS function to get the compass "warmed up"

IMPROVED: Cached the camera to make it more efficient

IMPROVED: Change all deltaTime to unscaledDeltaTime now if you change the time scale of your app, RealSunAR will not be influenced by that

CODE CLEANUP: Some System.DateTime can be DateTime.

CODE CLEANUP: Improved code comments

MANUAL: Better/cleaner instructions (Including new section: How it works)