This image is made with a screenshot from Unity's scene viewport

SUPER-EARTH **Documentation V.1.2.0 by synapsemassage**

Always feel free to ask me if the documentation doesn't solve your problem! Don't hesitate to give me feedback.
helpmeplease***superearth.de (replace *** with @)

About (and some Blabla)
The images of our universe available on the net are very often just breathtaking. It teased me for a long time to make a move into this direction and create a reasonable 'cosmic something' on my own. It became some planet stuff.

This asset package is kind of standard, it consist of textures, materials, shaders and geometry, nothing that you would call fancy technology just typical Unity stuff. Yet, you' ll be able to achieve visually realistic results with some nice little features like pinch-free poles or planet surfaces rendered with 6 x 4096 x 4096 textures. **The textures are the heart of the package. So even if you don't need or don't like the shaders the textures might be still very useful in your space project.**

And well, I lost a bit control over this little project and ended up with approx. 12GB of assets. It contains mid- and high-resolution textures (2K*2K - 16K*16K). In addition you will get hundreds of materials and some shaders that I used for testing. This package is not a solar system maker (not yet), it is solely about creating realistically looking SUPER-EARTHs that look good from far to mid-distant orbits (related to size of our home-earth)!

There are enough ready-to-go planets/materials to jumpstart the planet part of your space opera.
The image at the top of this document and the images that you can find in the unity forum are screenshots from the Unity editor (WYSIWYG).

Please note that these planets are not physically correct but strive for a realistic look. The performance is very much depending on texture sizes and shader use. The textures should be compatible with most planet shaders outside this package as all textures include a cylindrically projected version. As everything in game tech this stuff is techniques to fake the real thing.

It was a huge effort to work on these large textures (16k x16k) with image editors. Using 64-Bit-Software to allocate enough system memory makes big sense. Please make sure to check the system requirements. Imagine these getting printed, at 96dpi it would measure around 433cm x 433cm (170inch x 170inch)!

Known issues:

- Materials/Textures: atmospheric turbulence without lightning
- Materials/Textures: Specular maps are not available yet
- Huge Textures ahead: Many textures are included as 4k or 8k (16k archived resource also available). If your hardware system is too weak it may become unresponsive! It may take 1 or 2 hours to import.
- Shaders **with** alpha channel are not recommended for mobile devices although they mostly work on these their impact on performance is too heavy in most situations. Instead use shaders without alphas and consider baking clouds into surface textures with an image editor (reject cloud sphere and transparency blocker).

Another known issue, the “Buy” and the “Download” :
Unity Asset Store supports packages up to **500MB**. Due to the package's sheer size of approximately **12.5GB** I will not be able to supply the full size version via the store. So, after purchasing you will need to email me your Asset Store receipt so that I can identify your purchase and give you download access to the full version. I will email you.
The difference between Asset Store version and “Full” is only in texture size. Many servers on the internet don't like data junks larger than 2GB. Thus the package will likely come in a split archive (.rar-parts), so you will need a compatible extractor like winrar (Windows) or unrar (OSX) to extract into one unitypackage. See system requirements below. **I apologize for this inconvenience!**

System requirements:

- **Unity 5.3.8f1 64-Bit is required for import. 32-Bit version will have big trouble to allocate enough memory for extracting and handling the textures. If you want to use large textures > 2048x2048 you will definitely have to continue to work with 64-bit.** Unity 5 (64-Bit) is limited to 8k textures. **If you would shrink textures in an image editor below 2K you could get away with 32-bit. I suggest using 64-bit.** Please consider not to import this large package directly into your current project because of its sheer size. Rather import it into a fresh new project and scale the textures to sizes **you** really need with a image editor. Please read also the “some advise” section below.

- 8GB RAM (16GB or more is always better).

- A reasonable GPU. But you're a game developer anyway, so you should have that.
- At least **50 GB of free Disc Space** during extracting and importing (also exporting of full projects), make sure you have sufficient hard drive space! Otherwise extracting or importing will fail due to lack of disc space. Extracting is very much depending on drive speed, internal SSDs will significantly speed up importing.
- A compatible extractor like Winrar (Windows) at http://www.rarsoft.com or unrarx (OSX) at http://www.unrarx.com , others may work too. The file will be delivered as a downloadlink to a split archive. Download the archive parts and start extracting with ".part1.rar". The parts will extract to SUPER-EARTH.unitypackage. Same with the highres textures.
- Tested on Win 7, OSX Yosemite (iMac early 2009), iOS 7+8 (iPhone 4+5), Android 5 Nexus 7 2012). Shaders work on Windows 7 and OSX Yosemite, actually even on iOS 7 + 8 and Android 5. Yet, for mobile devices I only recommend shaders without alpha channel and simple planet setup (one layer + baked cloud surface texture in image editor). Alpha channels have a heavy impact on performance of mobile devices and materials with alphas are not recommended for mobiles.

Some advise
At first you might want to avoid importing the large SUPER-EARTH package directly into your current project because of its huge textures. Working with these (especially cubemaps) could make your system unresponsive. Rather import it into a fresh new empty project and make yourself familiar with the asset. Think about the texture size that you will really need in **your** project and consider scaling down SUPER-EARTH textures with an image editor accordingly.

Alternatively you would start with the small package available in the asset store and replace the small textures later with your higher resolutions.

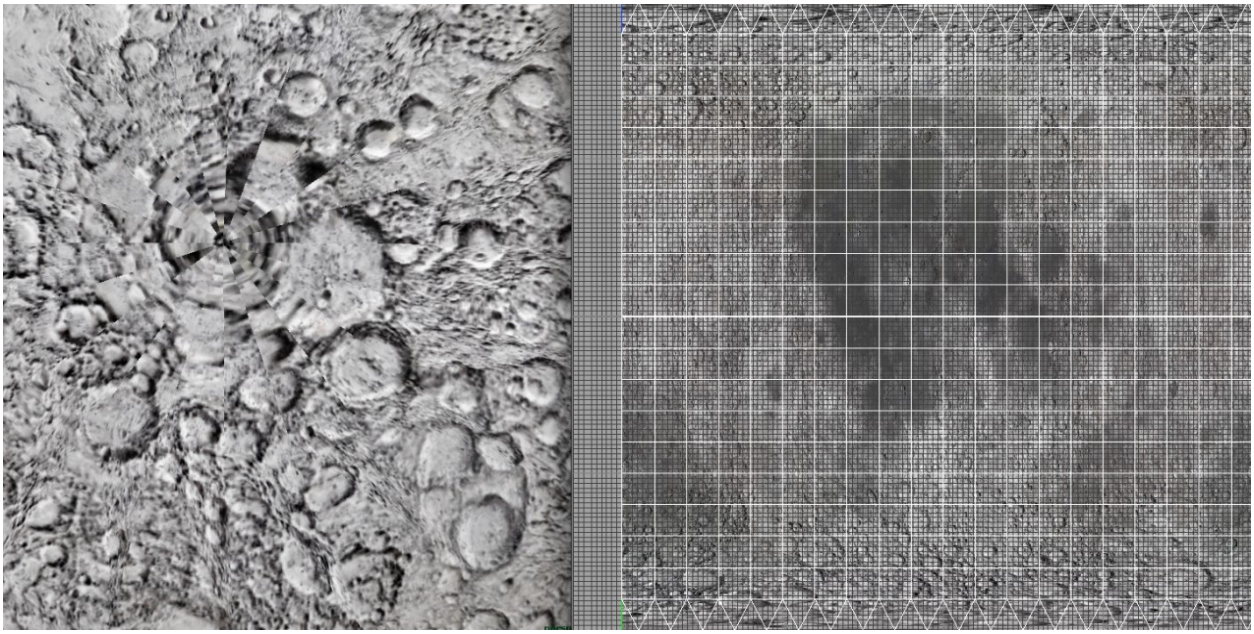
A wise decision will save you a lot of time**!** Weak hardware with big textures will likely result in lame editor performance and unresponsiveness. This unresponsiveness might be followed by Unity crashes. Restarting Unity might invoking reimporting of **all** the assets of your project. Imagine this happening several times per day, you will likely go nuts. So I'm saying this again "make a wise decision on your texture sizes" and if possible your hardware. Keep 50GB free hard drive space, because unitypackages are a subject to compressing/uncompressing and this is very much depending on your hard drive space and speed! This advise is not only related to SUPER-EARTH consider it general if you work on large scale projects. For example: My previous system consisted of a Intel Core 2 Quad Q6600 (CPU), Win 7 64 bit, Nvidia 650 GTX (GPU), nforce 680i LT (MBOARD), 8 GB RAM, old-school HDDs. It worked well with 2k textures. Without HDDs Windows Experience Index would be at 7.2. Now I'm using a Xeon E5 1650 v3 3.50GHz (6-Core CPU), Win 10 Pro, Nvidia 970 GTX, 64GB RAM, Solid State Drives.

Content
- Highres: 26 very realistic, highres, cylindrical cloud textures. In Unity available at 8192x8192, archive available with 16k textures.
- 26 highres, cylindrical textures for civilisation lights, e.g. advanced alien, coast line, ...
- Highres: Cylindrical texture set of our Earth (surface with/without clouds, bump maps, watermask). In Unity available at 8192x8192, archive with 16k textures

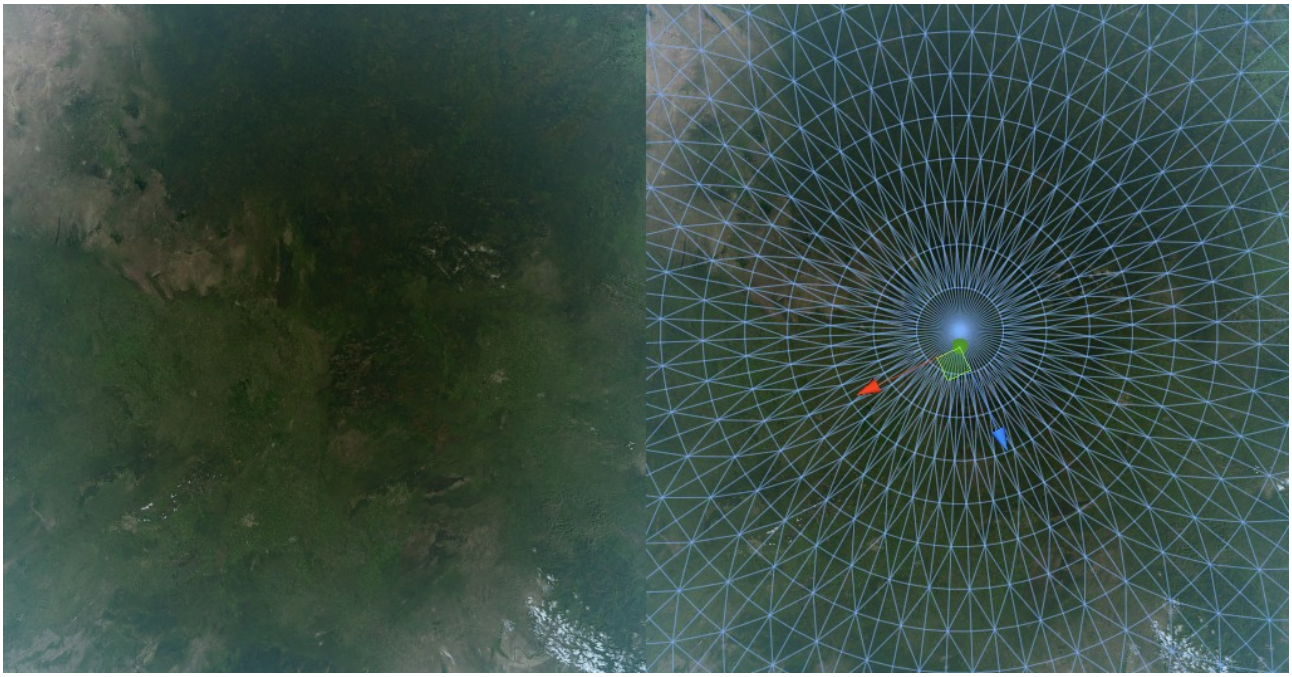
available for extra download.
- Highres: 8 highres cylindrical texture sets for earthlike planets (surfaces, bump maps, archived watermask). In Unity available at 8192x8192, archive with 16k textures available for extra download. **Note**: Currently I don't have a shader available that supports watermasks.
- 57 cylindrical texture sets (2048 x 2048) for earthlike planets (+bump maps).
- Shaders for planet surfaces and clouds.
- PSD-file to merge and adjust colors for surface, clouds and atmosphere. Available as archive for extra download.
- Specially unwrapped cylindrical spheres for almost pinch free poles. (vertex count: 420, 930, 1640, 3542, 6480, 14520, 29040)
- Geodesic spheres (vertex count: 42, 162, 642, 2562, 10242)
- Specially unwrapped spheres split into 6 sides (based on cube faces, vertex count per cube face: 289, 441, 961, 1681) supporting 6x4096x4096 rendering
- 26 + 8 highres texture-sets (clouds + surfaces) reprojected into 6 cube faces supporting 6x4096x4096 rendering
- 1 Alien + 1 Ice planet texture. In Unity available at 8192x8192, archive available with 16k textures.
- Hundreds of planet materials

Note: All highres textures should be crunched down to 8192x8192 or 4096x4096, so Unity can handle it. Higher resolutions wil be made available in a .rar file. Full resolutions are usually at 16384x16384)

Special spheres for planets - Why?
The problem with making planets based on spheres from 3d applications like Unity, are their poles. When mapping cylindrical projected textures (a most common projection for mapping textures onto spheres) to standard spheres you will get ugly pinching at the poles. If you look at the standard uv layout you will see that poles are unwrapped as saw tooths unlike cylindrical projected textures that are simply stretched at the poles (not saw tooth). Means that the UV layout will not match the texture correctly. Thus you will get pinching.



Pinching on planet surface at pole *UV layout sawtooth*



Pole without pinching *Geometry pole*

I offer 3 approaches for mapping onto spheres. You would choose the method that fits best to your project. Remove pinching with “cubemaps” and “6sided”. Reduce pinching with “cylindrical”. Numbers in the sphere name indicate verts count (based on 3d-package).

Approaches

A. **'Cylindrical' materials.** To keep pinching at a minimum you would avoid using mip maps. And if possible keep texture resolution as high as possible. Keep in mind that texture resolution and mip mapping are important factors for your app's performance.

Cylindrical spheres:
cylsphere_92, cylsphere_382, cylsphere_872, cylsphere_1562, cylsphere_2452, cylsphere_3542, cylsphere_4832, cylsphere_6322, cylsphere_8012, cylsphere_9902, cylsphere_14292, cylsphere_19462, cylsphere_25442, cylsphere_32759

Available shaders/materials:
SurfaceBumpRimLightrampHazeAlpha
SurfaceBumpRimHazeAlpha
SurfaceRimLightrampAlpha
SurfaceBumpRimAlpha
SurfaceRimHazeAlpha
SurfaceBumpRim, mobile
SurfaceRimAlpha
SurfaceRimHaze, mobile
SurfaceRim, mobile
SurfaceRimHaze, mobile
CloudsRimBumpLightrampDensityAlpha
CloudsRimLightrampDensityAlpha
CloudsRimBumpDensityAlpha
CloudsRimDensityAlpha
CloudsRimAlpha

CloudsAlpha

B. **'Cubemap' materials** remove pinching completely. You can use cubemap shaders/materials on any spheres (geospheres, cubed spheres or cylindrical spheres) but I prefer using **geospheres** as their geometry is spread evenly on a sphere surface. Typically, that means that you can get away with lower geometry resolution (less vertices).For cubemap materials you must import/apply cylindrical textures as cubemaps in the importer (→ adjust importer settings). Unfortunately it is not possible to use tangent space normalmaps created by Unity's importer. Please use the supplied object space normalmaps instead. Read up several steps below “how to create good object space normal maps. The disadvantage of cubemaps is a higher memory consumption. Unity (5.0 and older) does not support compression with cubemaps. In Unity 5.3.8f1 compression for cubemaps is available!

Geodesic spheres:
geosphere_42, geosphere_162, geosphere_642, geosphere_2562, geosphere_10242

Cube spheres (alternative for geodesic spheres):
cubedsphere_d1_26, cubedsphere_d2_98, cubedsphere_d3_218, cubedsphere_d4_386, cubedsphere_d5_602, cubedsphere_d6_866, cubedsphere_d7_1178, cubedsphere_d8_1538, cubedsphere_d9_1946, cubedsphere_d10_2402, cubedsphere_d15_5402, cubedsphere_d20_9602

Available shaders/materials:
SurfacecubemapBumpcubemapRimLightrampHazeAlpha
SurfacecubemapBumpcubemapRimLightrampHaze
SurfacecubemapRimLightrampHazeAlpha
SurfacecubemapRimHaze, mobile
Surfacecubemap, mobile
CloudscubemapBumpcubemapRimLightrampDensityHazeAlpha
CloudscubemapBumpcubemapRimLightrampHazeAlpha
CloudscubemapRimDensityHazeAlpha

C. **'6ided' materials.** These are based on cubed spheres but split into six faces (up, down, right, left, front, back). Each face is mapped with one material containing one texture for up, down, front, back, left, and right. This makes it possible to use a whopping 6x4096x4096 texture space. Note that such textures have to be derived from a cubemap projected texture like faces of a skycube.

Cube spheres 6 sided:
cubesphere_d8, cubesphere_d10, cubesphere_d15, cubesphere_d20

Available shaders/materials:
Same shaders as in A.

D. **'Shader Forge' Materials** (new!)
I've ported all the shaders over to Shader Forge for your (and my) convenience and added new shaders + materials accordingly. Future Shaders will be Shader-Forge-only from now on. Shader Forge Shaders should behave the same way as their original counter parts. Regarding the naming convention 'SF_' is added to the name e.g. SF_CloudscubemapBumpcubemapRimLightrampDensityHazeAlpha_V1. You should be able to switch between shader counterparts without noticing a difference.
New shader “SF_CloudscubemapBumpcubemapRimLightrampDensityHazeAlphaFlow” includes flowing cloud movement on cubemaps: turbulence (e.g. Hurricans) based on

flowmaps. You can edit/paint your own flowmaps in your prefered flowmap editor and match it precisly to your cloud map. Flowmaps are provided for each cloud map. Attention: Best results will be achieved with flowmaps up to 128x128 texture size. Larger textures will introduce artefacts unfortunately.
SF_CloudscubemapBumpcubemapRimLightrampDensityHazeAlphaFlow and SF_SurfacecubemapBumpcubemapRimLightrampHazeAlphaCivilisationlights are Shader Forge only.

General How to use...
It's important that you use materials/shaders on **corresponding** spheres. For example, using a sphere with **cylindrically** unwrapped UVs requires a material (including shader) that is made for cylindrical spheres. For example:
6sided sphere > 6sided material
geosphere > cubemapped materials
cylindrical spheres > cylindrical materials

You will find the materials and other components in folders named accordingly (e.g. “cylindrical”).

I am using a very detailed naming convention based on shader properties.
You will be able to identify the reference of planet examples, materials and shaders via their naming.
I also suggest using combinations of surfaces (game objects) and clouds (game objects) that have very similar names like <CloudscubemapBumpcubemapRimLightrampDensityHazeAlpha> with <SurfacecubemapBumpcubemapRimLightrampHazeAlpha>.

A typical planet would consist of a surface sphere, a cloud sphere and also, if alphas are involved, a inner sphere (transparency blocker) to circumvent z-sorting issues. For mobile devices don't use alphas instead use only one sphere and merge textures (surface + clouds) into one texture with the help of an image editor.

Everything else is pretty standard editing in Unity. It's easiest to take a look at the planet samples in the provided scenes and mimic what you find there.

Several shaders have a method for smoothing rims. These shaders contain „...Alpha...“ in their name (like in the naming example above). See the difference of soft rim vs. rim in the image below. If you make smoothing very subtle it will look like anti-aliasing. With normalmaps you might experience random flickering pixels around the edge caused by 2xmulti sampling (anti-aliasing) depending on normalmapping, switching off anti-aliasing or setting it to higher values will reduce or fix it.

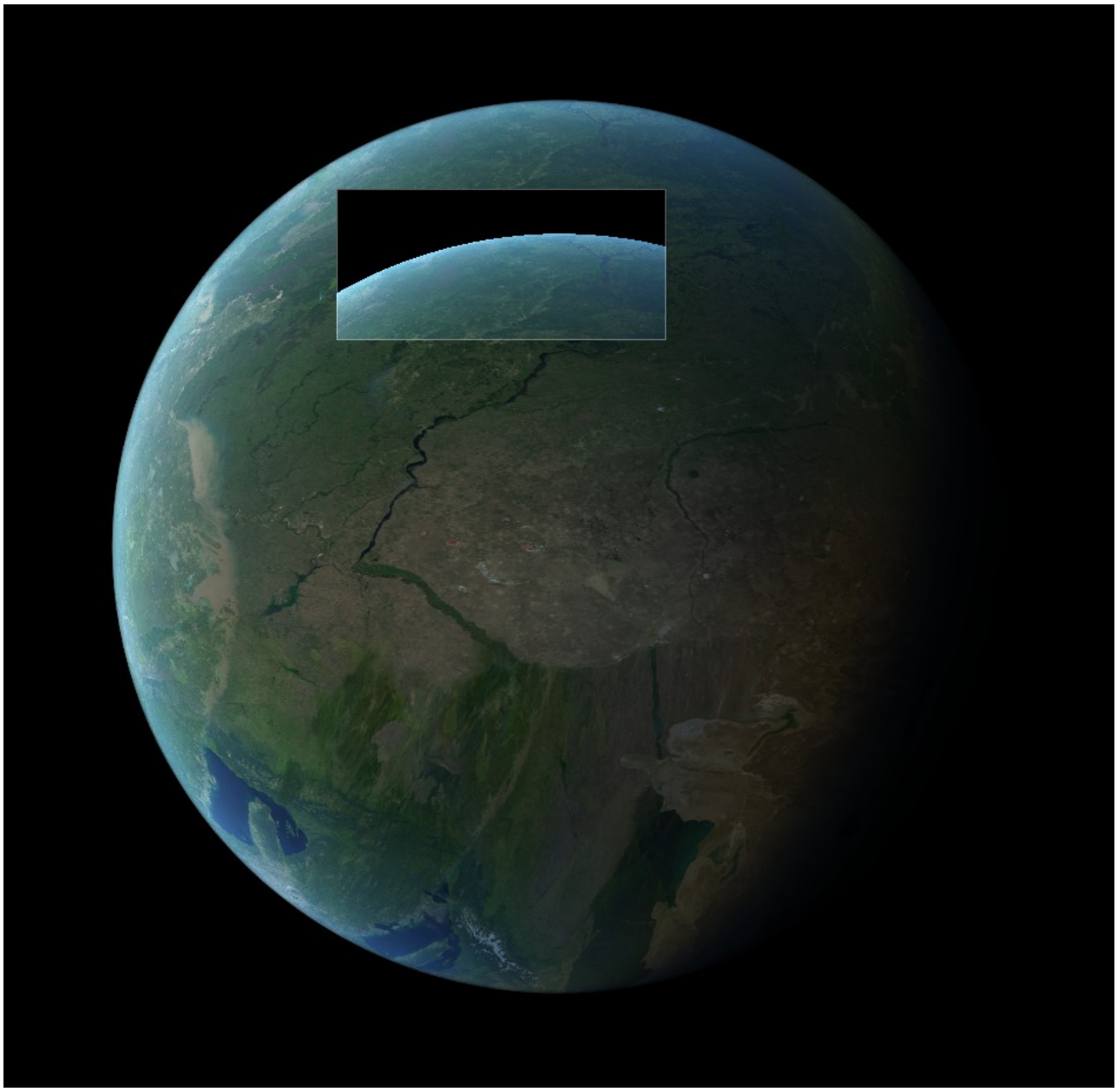


Abbildung 1: Smooth rim vs. non-smooth

How to use materials

1. My prefered way of creating a new material is duplicating an existing one. Apply the new material by following the naming convention. Dismissing this rule might lead to unexpected results ;-). Also keep in mind that cloud spheres and planet spheres differ slightly in size (scale factor 1.004 for cloud spheres). Take a close look at the examples.

2. Adjust parameters and/or exchange textures until you are happy with the result. Create a prefab with this planet. You could also animate materials if needed.

How to use shaders

You will also get a bunch of shaders that render either clouds or surfaces as seen in the screenshots. Some shaders come in different flavours and their differences can be very subtle. I'm trying to give you a smooth shader experience but I'm not a shader expert, so please consider shaders as work in progress.

Shaders have a naming convention that mirrors the containing features, means longer shader name = more features.

Shader naming convention example: SurfaceCubemapRimLightrampHazeAlpha
This shader renders

- surface (not clouds)
- cubemap texture (main texture)
- rim
- light ramp: light model based on ramp texture
- alpha (rim)
- haze

More features mean also more shader code thus (usually) a larger impact on performance. I have made a little prefab (FRAMES_PER_SECOND) that calculates the frame rate. Simply drop the prefab into your scene, build it and watch the rate. Increase number of planets until frame rate drops significantly and gets unacceptable. Try this with different planet shaders and choose according to your situation.

Note: As far as I can see shaders do work with perspective **and** orthographic cameras. For orthographic cameras set camera size in the camera component (inspector) correctly. For example to value '1'.

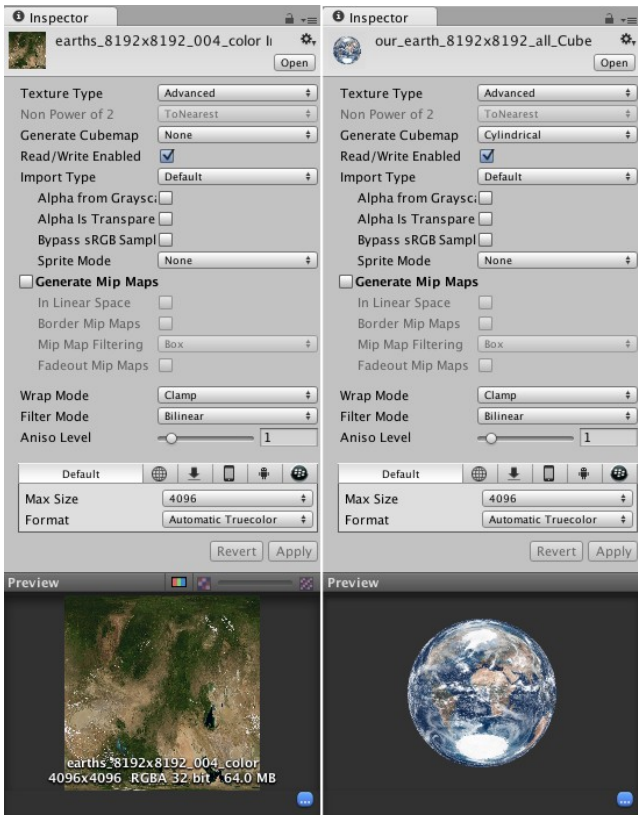
How to use textures
Textures are power of two. Projection is either cylindrical or cube (like skycube: each face one texture). Working with these textures is straight forward and Unity business as usual.

Please be aware that cylindrical textures might be imported as cubemaps for usage in the cubemap materials. Note the difference of cubemap materials and 6faces materials. Although 6faces materials are based on the idea of cubemaps, each cubed sphere face is linked with one material (texture) and is treated as separate geometry. Means each cubed sphere will require 6 materials (textures) thus 6 draw calls. Tangent space normalmaps for 6faces materials must be generated in xNormal not in Unity! Not doing so will introduce visible seams caused by Unity's handling of tangent space normalmaps. Unfortunately, adjusting bump height becomes less intuitive. I've already included precompiled normalmaps with 2,75 smoothing. If you want other height values grab **xNormal** at **xnormal.net** and farfarer's xNormal plugin **Unity3D Tangent Basis Calculator** at **farfarer.com** and recalculate the normal maps. Make sure you read farfarer's README.txt. It's possible that in newer versions of xNormal **Unity3D TBC** will be already integrated.

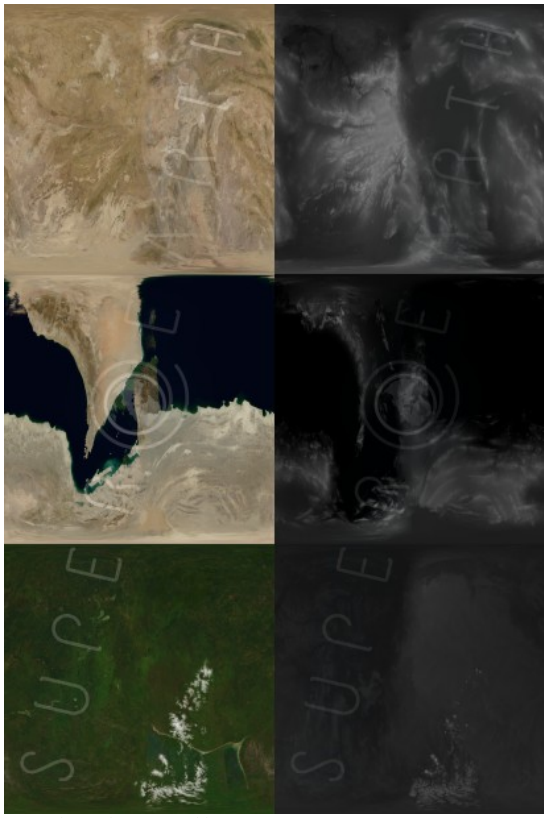
If you have tight performance requirements you may want to get rid of the cloud sphere and bake surface and clouds into one texture in photoshop. You can use the supplied photoshop file (ADJUST_COLOR_and_MERGE_LAYERS.psd) in the ZIP-archive. Follow the instruction that is **inside** the .psd-file. Basically it's placing textures onto new layers in the file, making changes as needed and merging/saving required layers as new textures.

If you need to rescale textures in Photoshop, you would **always** use **bipolar interpolation** to avoid semi transparent image borders that introduce ugly seams on the planets. Alternatively use Gimp to avoid this issue. Be aware that issues with seams can also have other reasons like tangent space normal maps.

You will need to change import settings based on your requirements. If your not familiar with Unity's workflow regarding materials and textures, please read this up in Unity's documentation. Basically you would change import settings in the inspector as you need. Make sure that **'Read/Write Enabled' = true.**
See below.



Cylindrical texture *Cubemap texture derived from cyl. texture*



Examples: color map and bump map

Attention: 4k cubemaps with mip maps consume 384MB of you memory. Keep this in mind! In Unity 5.0 (+older) **cubemaps** won't be compressed and are imported as RGB 24bit at max 4096x4096, Unity 5.3.8f1 supports cubemap compression. (In-between versions haven't been tested).

How to create objectspace normalmaps

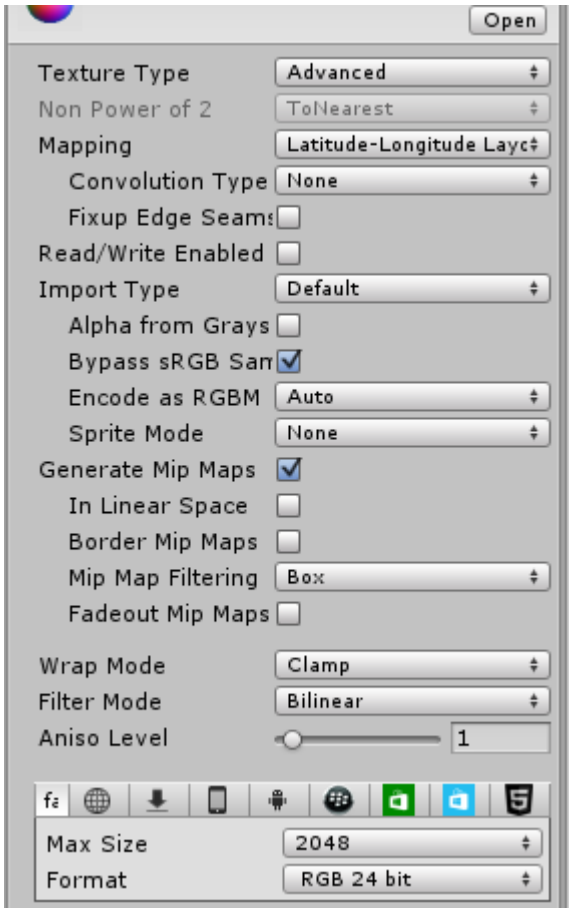
1. Download and install Xnormal by Santiago Orgaz (a must have tool for game makers downloadable at http://www.xnormal.net)

2. Start Xnormal and go to „Tools“ Section. Select Height Map to Normal Map. Load height map and generate normal map (right click) with default settings but smoothing at 2.5. Important: Do not save image with Xnormal but copy normal map (right click copy paste)!!! Open Gimp or Photoshop and paste normal map into new file. Save as .png with Gimp or PS. Attention: If you don't stick to this procedure you will most likely end up with object space maps corrupted with ugly seams/artifacts!
3. In Xnormal select Tools > Normal Maps to Object/Tangent Space Converter. Load previously saved Normal Map + load geometry from assets folder in SUPER-EARTH (cyl. Sphere). Set „convert tangent space to object space“. Set edge padding to min. 1 or more. Generate!

4. Unity import settings: Texture Type = Advanced, Import Type = Default, 'Read/Write Enabled' = true, Bypass sRGB = on, Wrap Mode = Clamp

5. Import settings objectspace normalmaps

See import settings below.

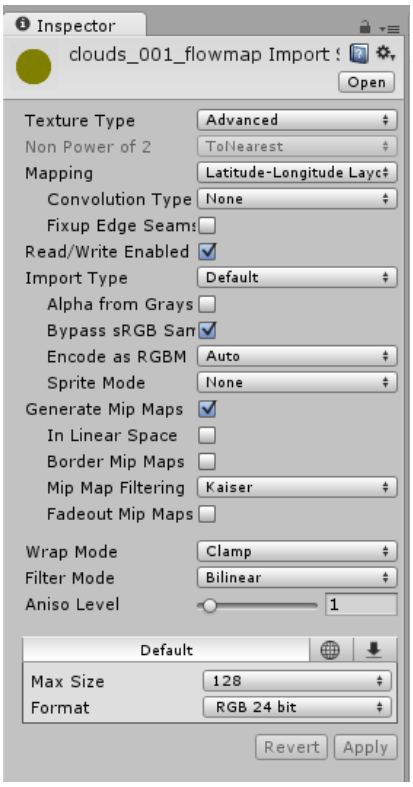


How to use flowmaps for cloud turbulence

1. If you need flowing clouds turbulence (e.g. Hurricans) use the shader "SF_CloudscubemapBumpcubemapRimLightrampDensityHazeAlphaFlow". It makes use of flowmaps. In the materials slot "flowmap" link to a texture inside SUPEREARTH>Textures>Flowmaps. I suggest using a flowmap number that matches the cloud texture as I have painted the flowmaps referencing its cloud map. You can edit/paint your own flowmaps in your prefered flowmap editor and match it precisely to your cloud map.
2. The flowmaps must be imported as cubemaps. Switch to Texture Type Advanced.
3. And Mapping "Latitude Longitude Layout".
4. Make sure you that you 'Read/Write Enabled' is true in the Import Settings of the flowmap.
5. My advise is Texture Max Size: 128. Attention: Best results will be achieved with flowmaps up to 128 x 128 texture size. Larger textures will introduce artefacts unfortunately (distorsion along grid-like lines). Currently I'm not sure why this happens :-(. I suspect that 8 bit-depth per channel is not enough and/or compression in .png produces this kind of characteristic data for gradients.

6. Format: RGB 24bit (if you prefer best texture quality)

Take a look at my import settings below.



How to use civilisation lights

1. If you need civilisation lights on the night side use the shader SF_SurfacecubemapBumpcubemapRimLightrampHazeAlphaCivilisationlights

2. The textures for these are imported as black/white cubemaps. Compare with the available textures.