TERRALIVE

the TERRAGEN™

Online Conference

18-20 January 2013

Working with the node network: Data- and work-flow

Presenter: Martin Huisman

Session content

- TerraLive survey results & conclusion:
  - Many common problems are related to difficulties with working with the node network.
- Brief introduction to the User Interface*
- Building a basic scene
  - Step by step explanation of scene creation and node network procedure to obtain desired result.

TerraLive survey results

- What aspect of Terragen 2 do you think holds back beginners most?
  - The User Interface (minimalistic, luckily not Maya!)
  - Working with nodes
  - The math wrong!
  - Lack of (simple) documentation

Sessions' goals and methods

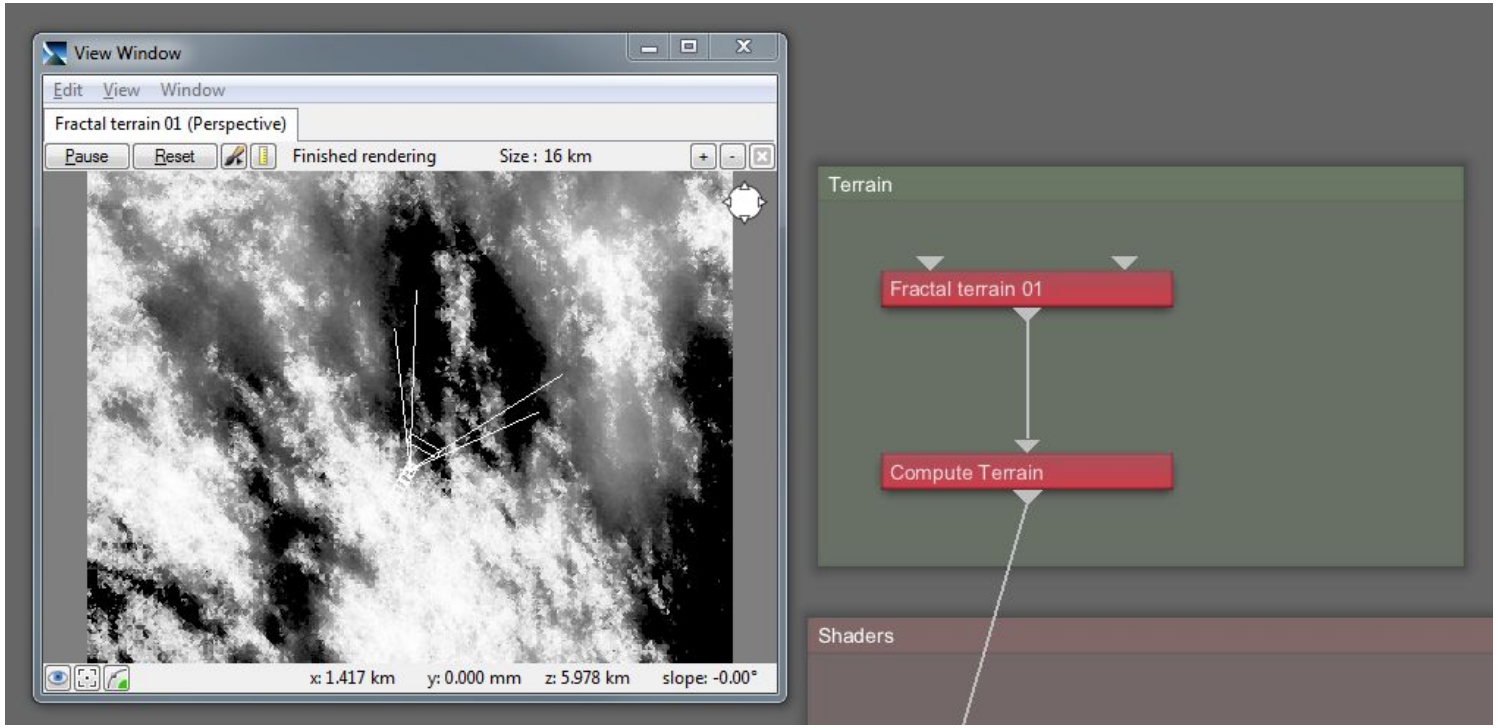
- Discuss the process of creating this scene:
  - Create basic terrain
  - Detail with displacements
    - » "Compute" nodes
    - » Mask displacements
    - » Blue node mini-network
  - Apply surface layers
    - » Mask surface layers
  - Add objects to the scene
  - Rendering our scene

Stage 1 of 7

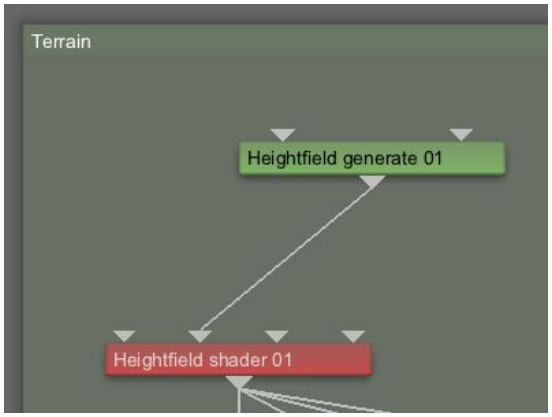
Adding a terrain in TG

Creating a basic terrain

- A fractal is a mathematical noise function with endlessly repeating “self-similar” patterns.
- In TG fractals have 3 dimensions and are not bound to a restricted area, like with height fields.
- Fractals also output/generate greyscale values, like height fields.
- Like height fields the gray scale value is “displaced” to create elevations.



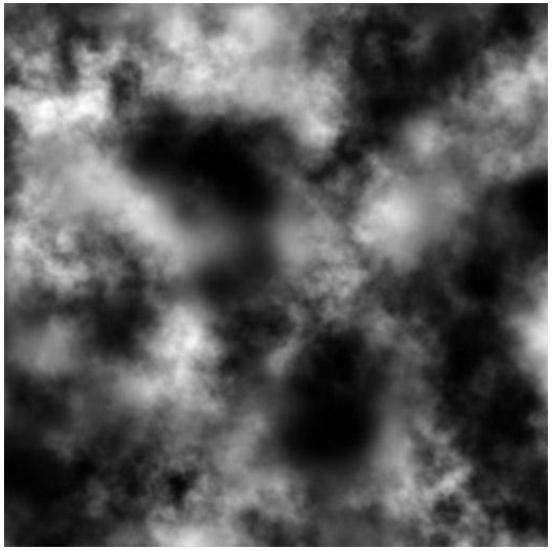
Creating a basic terrain

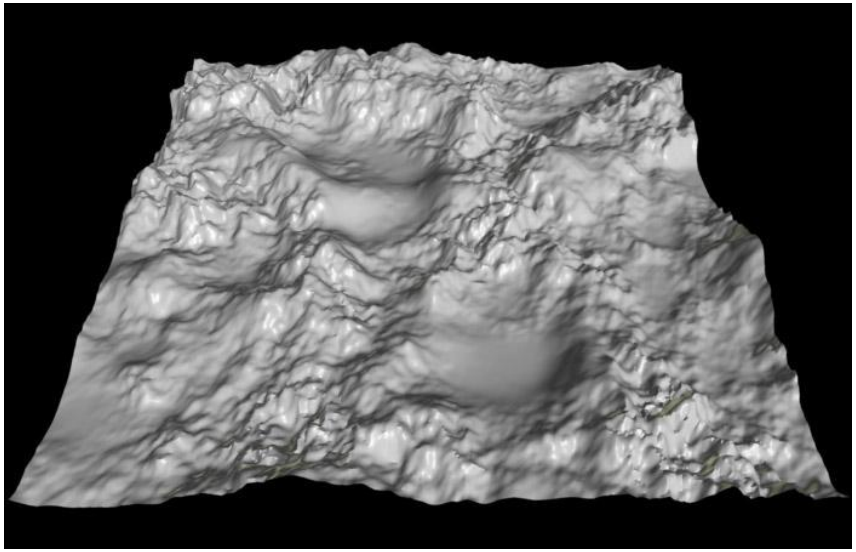


A heigthfield is a 2D image storing elevation data as greyscale values where:

- Black = no elevation
- White = max elevation

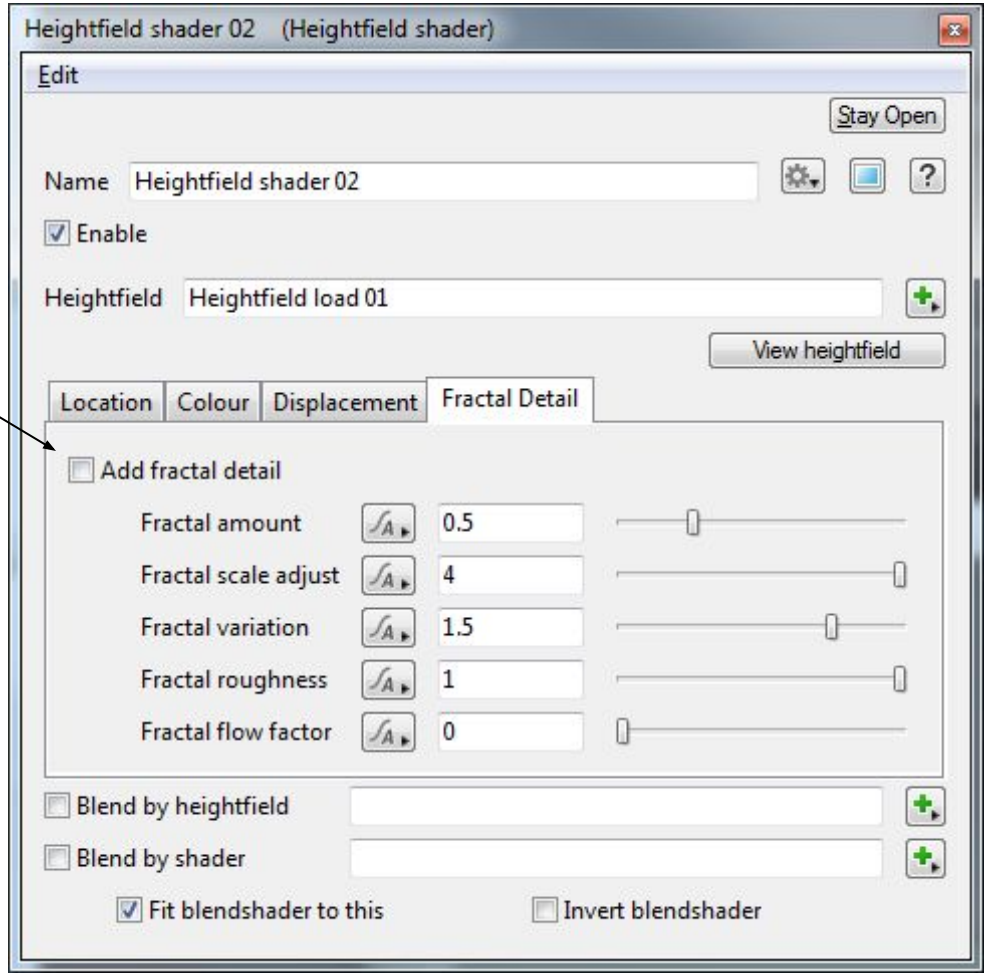
The elevation altitude range is stored as meta-data in the heightfield file



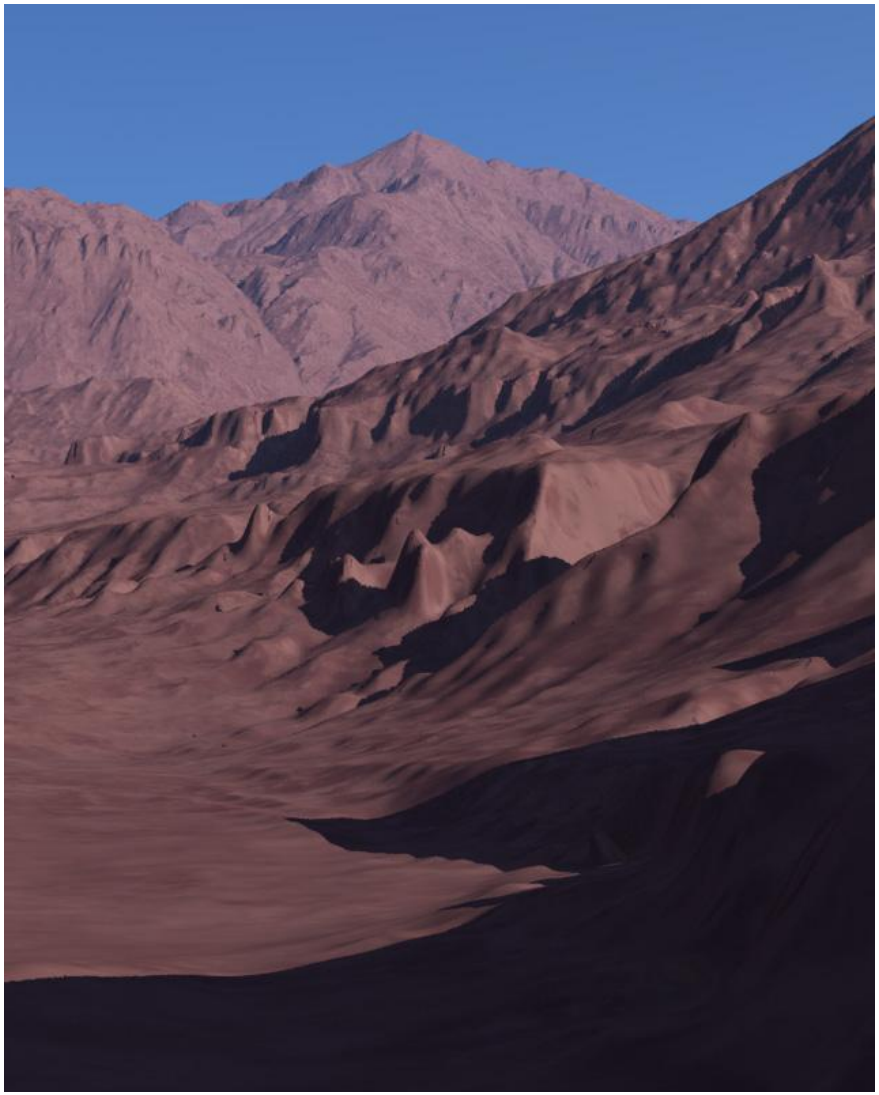


Disabling “fractal detail”

- Unchecking “add fractal detail” will disable fractal enhancements made to the height field



Disabling “fractal detail”



Stage 1 summary

- Height fields and fractals generate greyscale values.
- Height fields have boundaries
- Fractals have no boundaries, you can cover an entire planet with 1 fractal.

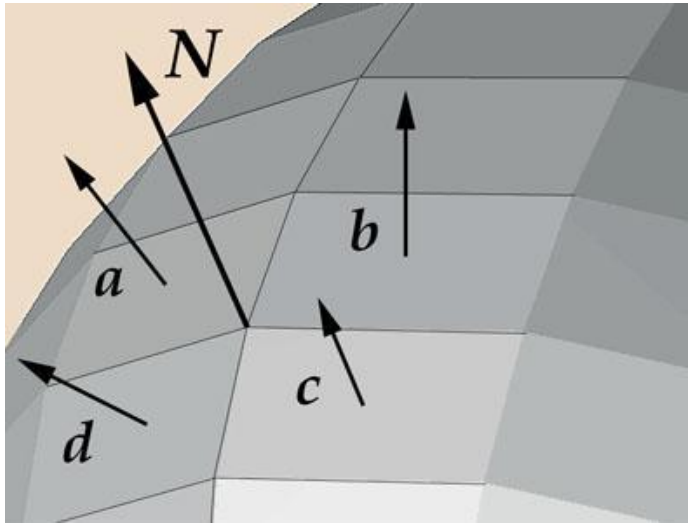
- In TG, height fields are “spiced up” with fractal detail by default.
- However, this can lead to problems when adding displacements or stones at a later stage
- Disabling “fractal detail” in the heightfield allows:
  - Smoother basis to start with
  - Full control on adding detail how and where you like

Stage 2 of 7

Adding outcrops/overhangs by using “redirect shaders”

Compute normal

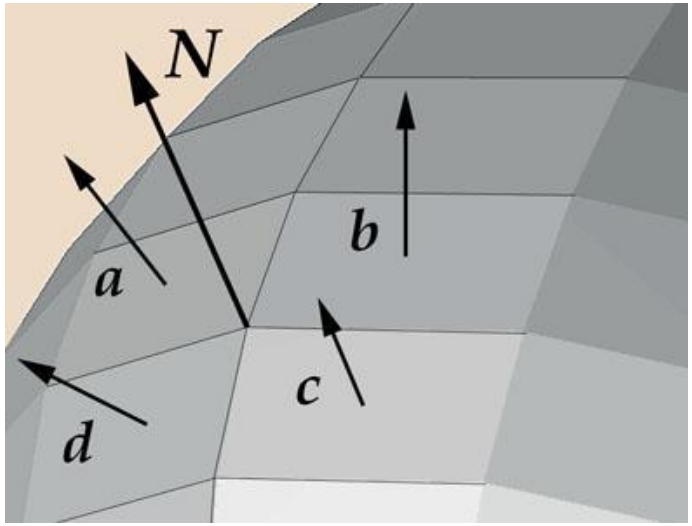
- The normal is a line or vector which is perpendicular to the surface.
- Every polygon has its own normal.
- Polygons intersect at a vertex (plural: vertices).
- The vertex normal is an averaged normal of its adjacent polygons.
- Without a computed normal TG uses the local “Up” vector, which is the vector pointing away from the centre of the planet.



http://en.wikipedia.org/wiki/Surface_normal
http://www.planetside.co.uk/forums/index.php?topic=1249.msg12539#msg12539

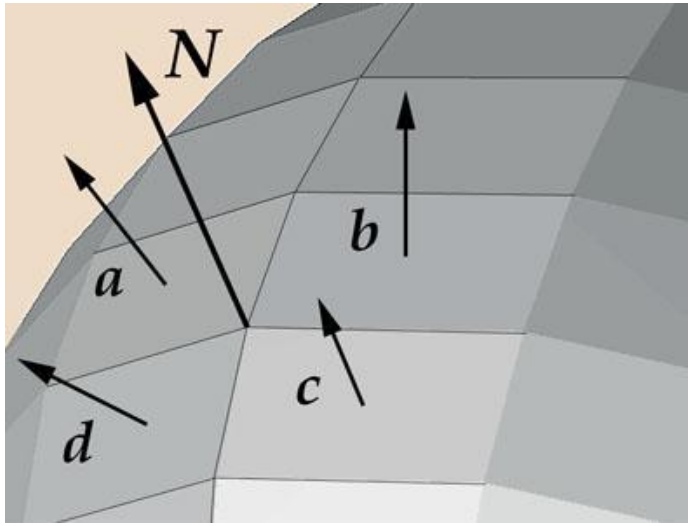
TEX coords from XYZ

- Abbreviation for “Texture coordinates from the X, Y and Z position” for the terrain.
- Needed for:
  - Aligning non-displacing shaders with the terrain (surface layers etc.) to make shaders “aware” of displaced surfaces.
  - Calculating altitude, allowing for restriction by altitude

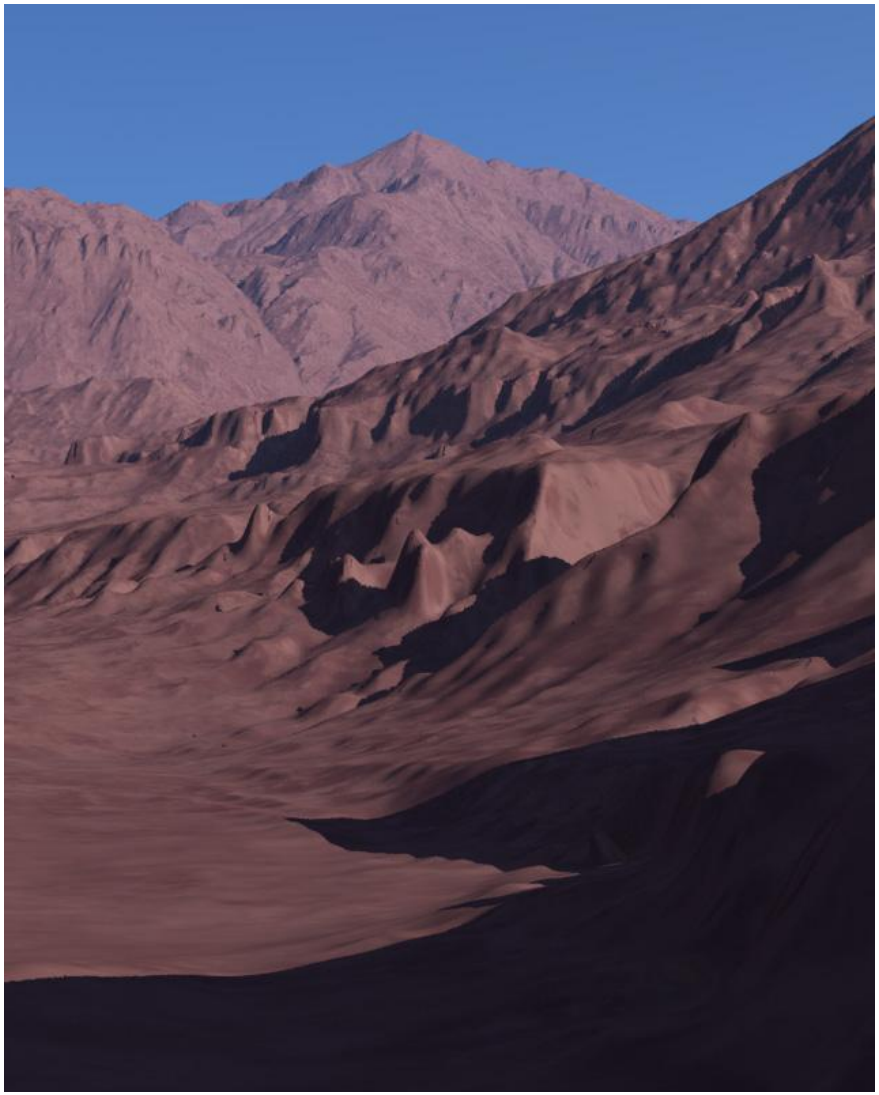


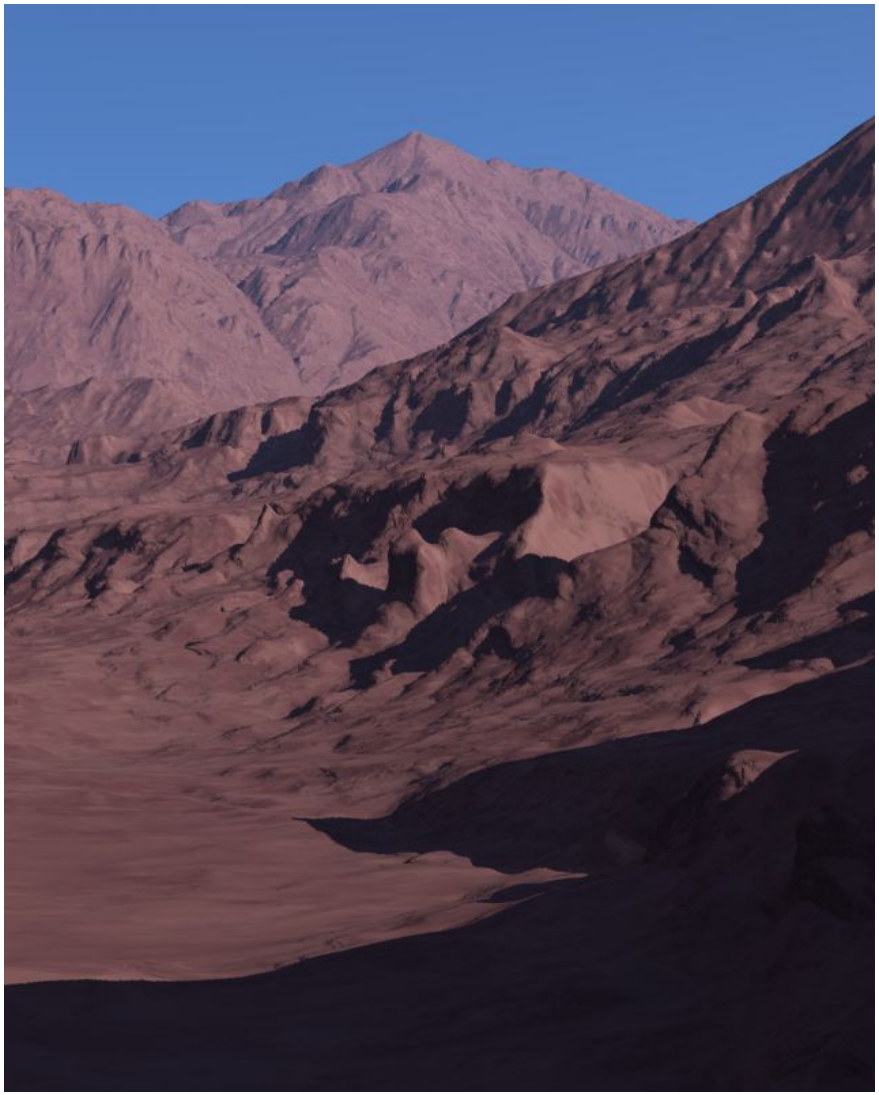
Compute Terrain

- Computes both the normals as well as the “texture coordinates”.
- Allows for:
  - Altitude/slope restriction by providing altitude and direction of the surface normals
  - Surface shaders to match the displaced geometry



Stage 2 result





Stage 2 summary

- Redirect shaders “trick” the displacement from a fractal to go into X, Y or Z direction, depending on where you plug the fractal in.

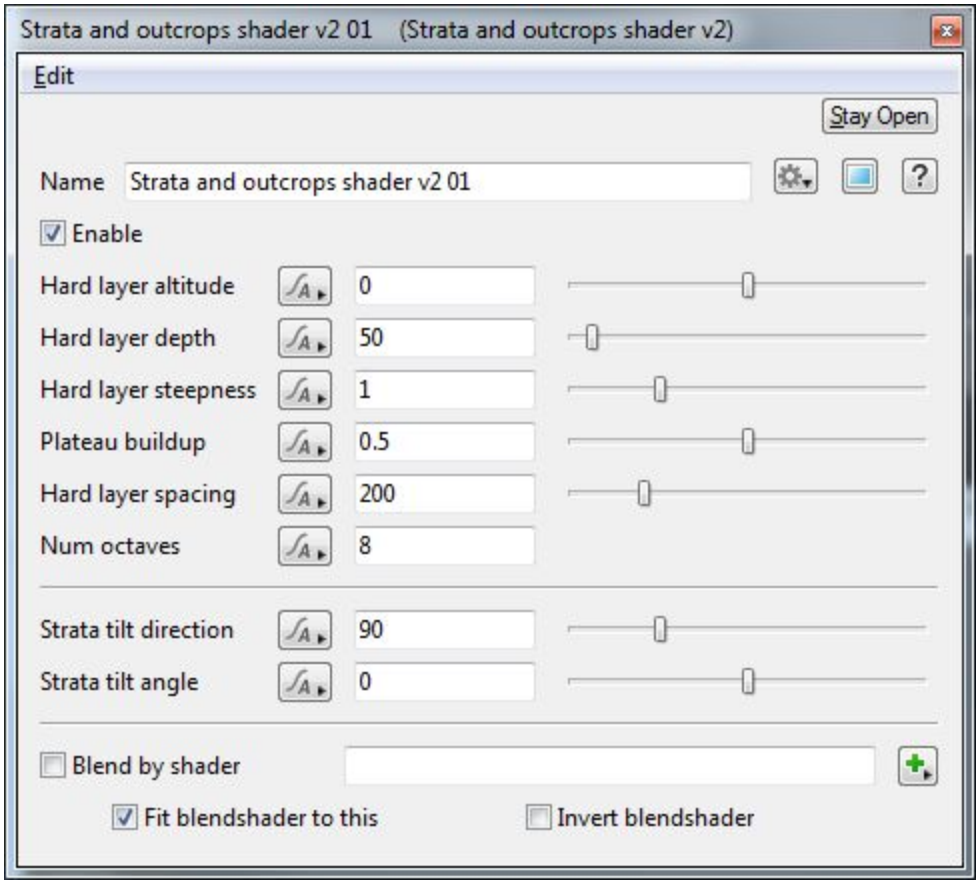
- When you restrict
  - for slope use “compute normal”
  - for altitude use “Tex coords from XYZ”
  - for slope and altitude use “compute terrain”

- The gradient patch size can be considered as filter which “averages” the normals for the “patch size” to prevent displacement (spikes) intersecting with each other.
  - A very small scale patch size ~ local normal
  - A very large scale patch size ~ average normal for great area

Stage 3 of 7

Adding strata by using the "Strata & Outcrops Shader"

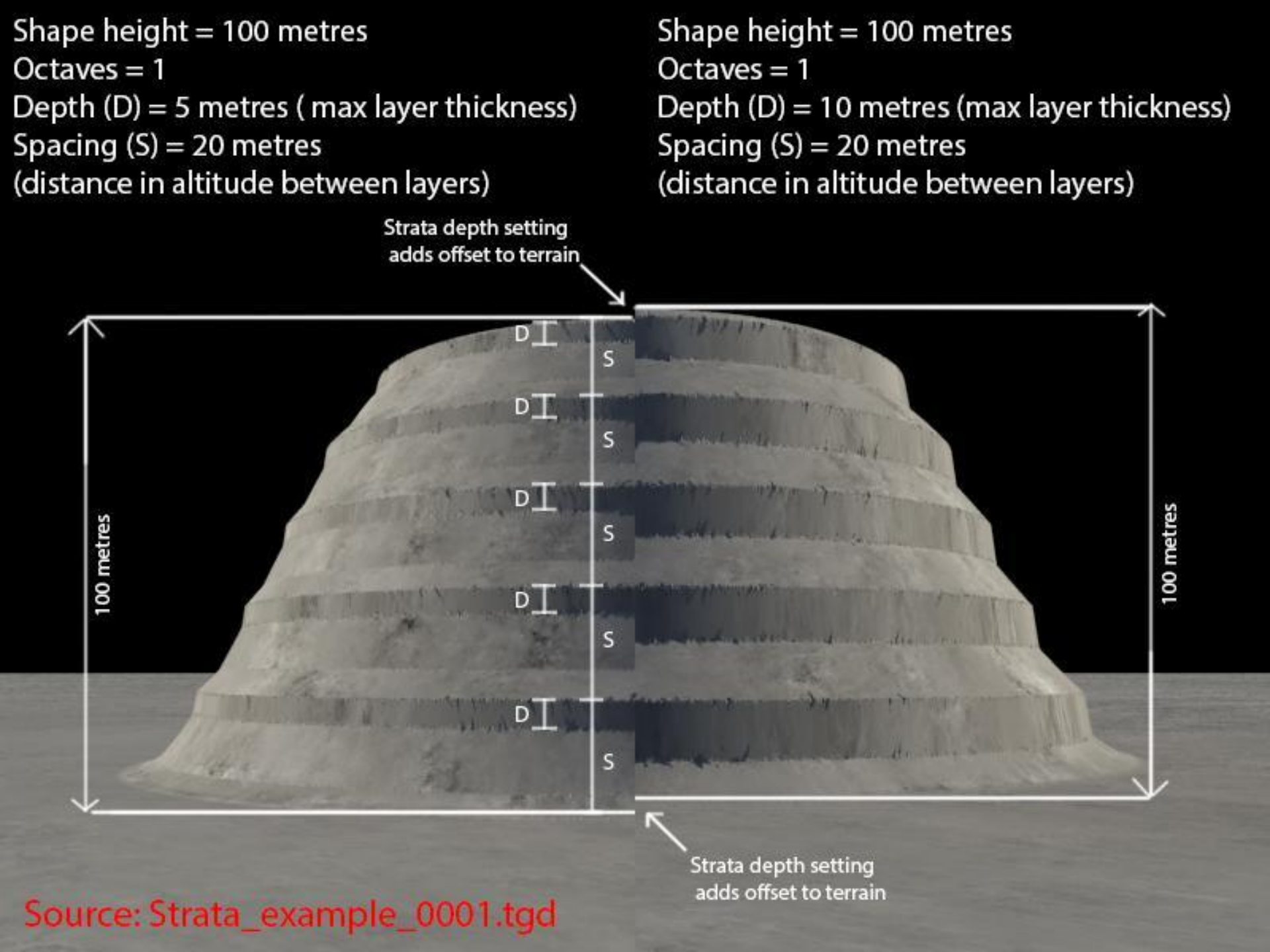
Strata & Outcrops Shader



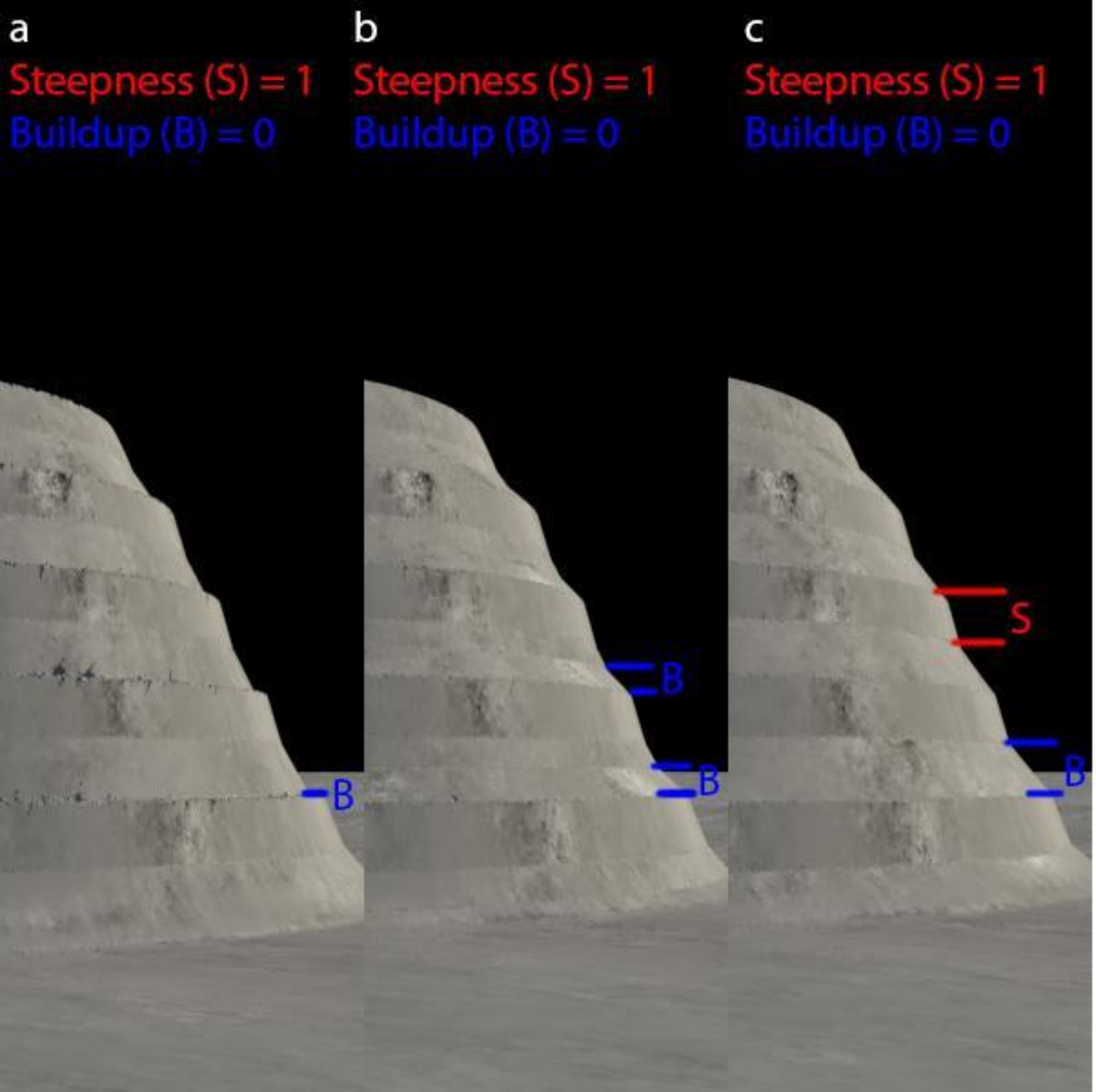
- Various settings for creating all kinds of effects…
- Let's have a look!

Shape height = 100 metres
Octaves = 1
Depth (D) = 5 metres (max layer thickness)
Spacing (S) = 20 metres
(distance in altitude between layers)

Shape height = 100 metres
Octaves = 1
Depth (D) = 10 metres (max layer thickness)
Spacing (S) = 20 metres
(distance in altitude between layers)



Source: Strata_example_0001.tgd



red lines define strata layer and its steepness

blue lines define slope buildup between the layers

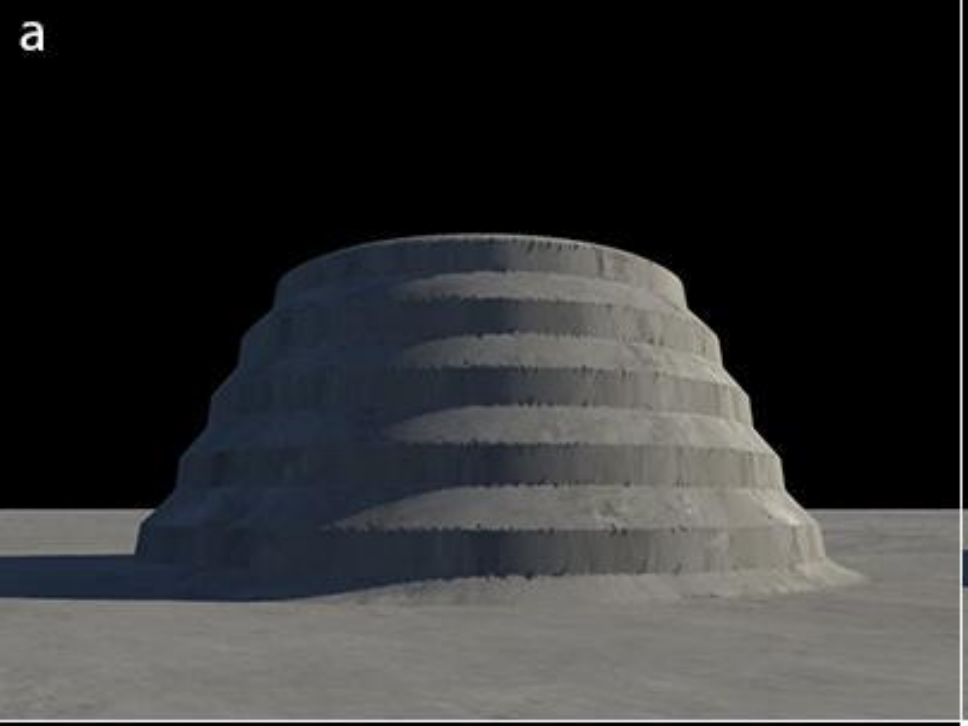
Steepness = steepness of strata layer defined by layer "depth"

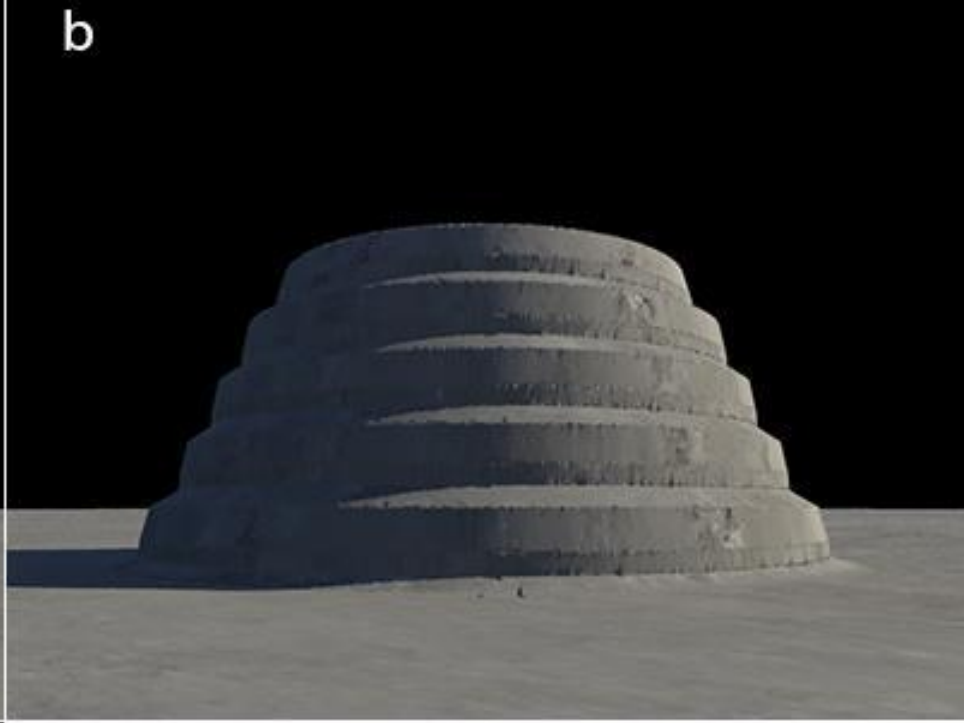
Buildup = simulates "talus" or slopes with deposition

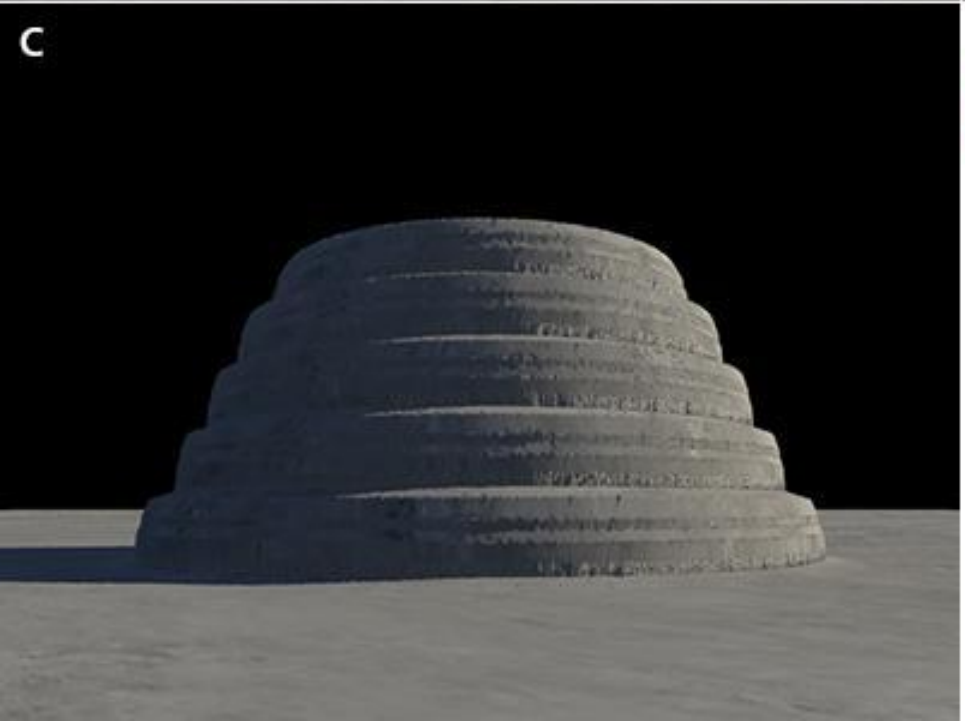
a) buildup 0 = flat slope

c) buildup 1 = single slope from lower layer's "top" to next layer's "bottom"

b) buildup 0.5 = buildup slope is half the layer's Steepness value and buildup will reach half-way to the next layer.







Adding 1 octave a time multiplies # of layers by 2 and added layers will be 2 x smaller compared to already existing layers

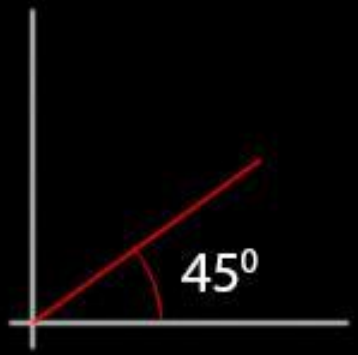
octave 1 = 1 layer
octave 2 = 2 layers
3=4, 4=8, 5=16, 6=32, 7=64 and 8=128 etc.)

a) Octaves @ 1 = each layer is 10m thick
b) Octaves @ 2 = 10m layer + 5m layer
c) Octaves @ 3 = 10+5+ 2.5+1.3+0.6+0.3m layers
-> = 19.7m of layers
-> spacing is 20m -> 1,25m "left" to "fill" with layers
-> adding octaves "fills" the "spacing" with layers

Camera rotation = 0 degrees (facing north)

Strata direction = 90 degrees (strata are facing east)

Strata tilt = 45 degrees (strata facing 45 degrees upwards to the east)



THUS, WE'll HAVE THE SAME RESULT WHEN:

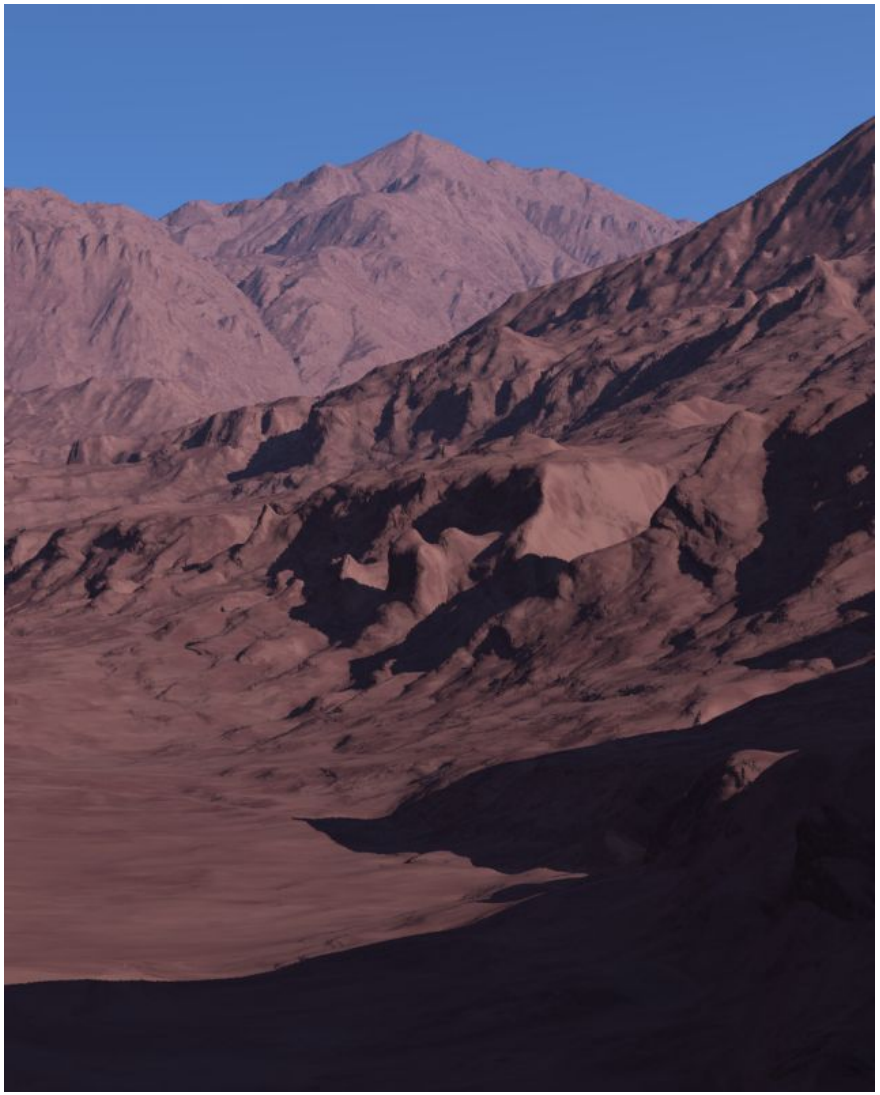
Strata direction = -90 or 270 degrees

Strata tilt = -45 degrees

Strata direction = 90 degrees

Source: strata_example_0002.tgd

Stage 3 result



Stage 3 summary

- Strata & Outcrops Shader is very versatile
- Mixing strata to have them “intersect” often gives very interesting results
- Don’t “over do” the settings to avoid staircasing and other unnatural effects
- Restrict strata by slope and altitude to break up appearance, as the shader can be visually quite dominant

Stage 4 of 7

Adding cracks by using a “blue node mini network”

Stage 4 result

Stage 4 summary

- Blue nodes aren't that scary, huh!? ☺
- The "Get position…" nodes asks TG
  - "what's the current state of the terrain at this point in the network?
  - What are the coordinates? □ "Get" coordinates
- Multiplying the result of "Get Position", using a vector can stretch the coordinates and thus will stretch anything downstream of it.
- To warp a function you need "Get position in texture" to start. Why?
  - Because the warp shader works in "texture space" and not in normals as obtained by "Get Position"

Stage 5 of 7

Adding surface shaders

Stage 5 result

Stage 5 summary

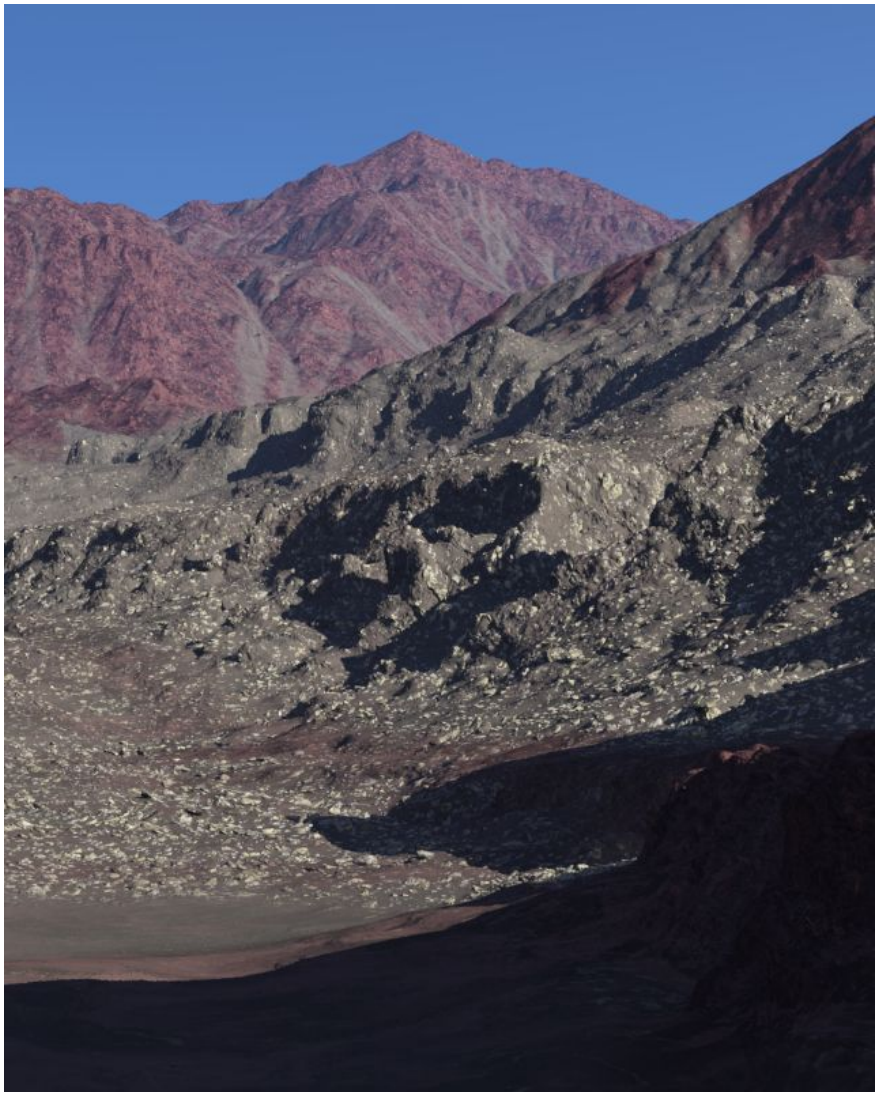
- Surface layers are key in shading your terrain, as they:
  - Add diffuse (colour) shading to your terrain
  - Restrict slope/altitude
  - Offer advanced coverage effects which interacts with the displacement (Sunday)
  - Offer stacking of shaders by having a “child layer” input port, which allows for:
    - Adding additional surface layers of colour
    - Acting as a “placeholder” to merge in blue node networks, fake stones or other shaders

- Everything connected to the “child layer” input port has the same restrictions as its parent surface layer!

Stage 6 of 7

Adding fake stones

Stage 6 result



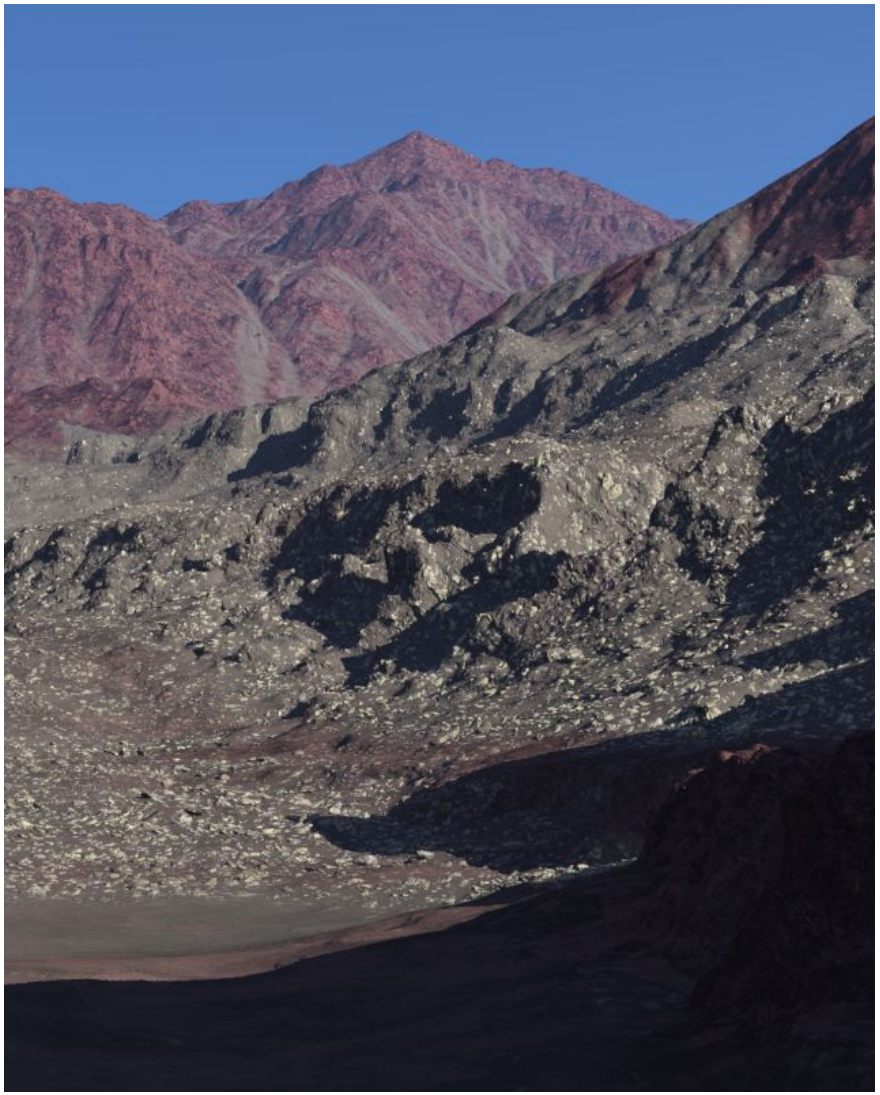
Stage 6 summary

- Fake stones can make/break your work
  - Difficult to work with due to settings and displacing them nicely
- Feed fake stone output into a surface layers child input
- Enable surface layer's smoothing to avoid many problems with displacing fake stones, as:
  - Stones will incorporate existing displacement
    - Smoothing reverts displacements to the point of the last compute terrain
    - The displacements calculated there will be smoothed
- Multiples layers with different sizes is the way to go
- Merge layers of stones using the "highest" mode avoids stones placed on each other

Stage 7 of 7

Adding water

Stage 7 result



Stage 7 summary

- Lake object is a “disc” object, but you can use a plane object as well.
- Keep the lake object as small as possible, why?
  - Water is rendered first, followed by the terrain and objects.
  - Keeping lake object small will reduce “overdraw” of your render and thus will reduce rendertime
- The best trick ever to minimize rendertime for water:
  - http://www.planetside.co.uk/forums/index.php?topic=8793.0
- If you don’t need transparency or can’t see it, then disable it to save rendertime.
- Wave scale is very important in communicating the scale of your scene with the viewer!

Workflow paradigm

- Create base terrain
  - Fractal based
  - Heightfield based
- Compute normal/XYZ/terrain, but when?
  - Normal □ when restricting for slope
  - XYZ □ when restricting for altitude
  - Terrain □ when restricting for slope+altitude
- Displace/detail terrain
  - Powerfractals
  - Voronoi
  - Etc.
- Compute terrain
- Shade/Texture terrain
  - Surface layer shaders
  - Image map shaders
  - Fake stone layers
  - Etc.
- Planet shader