Voxeland is a next generation of a cubic-style terrain seen in Minecraft and others, but a bit more than that: it is a subdivided and adroitly smoothed cubical structure.

Think of your terrain as 3D image of cubical pixels. Each “pixel” called Block contains a link to a data structure called Block Type. Block type has various data like block texture, grass, prefab, etc. Some of the blocks type are filled with terrain, while others are not (filled with “air”). Terrain surface is created on the border between filled and non-filled blocks.

As Voxeland deals with voxels data, it can calculate per-voxel ambient occlusion. Voxeland’s ambient is calculated immediately after terrain build or change, and it does not have a delay between change and AO apply. Voxeland ambient is applied additionally to the scene ambient.

Quick Start Guide

To create Voxeland terrain click GameObject -> 3D Object -> Voxeland Terrain. New game object named “Voxeland” will appear in the scene and will start building. Select it in a Hierarchy list, and you are ready to edit your terrain:

- To add block click left mouse button above the selected block.
- To remove a block click left mouse button with shift button pressed
- To blur the area click left mouse button with control button pressed (terrain could be blurred when brush extend is more than 0)
- To replace a block click left mouse button with control and shift pressed

Block type could be selected in a Block Types foldout by clicking a block type card. A new terrain has only one block type, it is named "Ground". To change the name click on "Ground" label. This block type has two toggles checked: “terrain” and “occlude ambient” - this means that it is filled with stratum and can occlude ambient light.

Block types that filled with terrain have terrain material properties. These properties are mostly the same as the properties of Unity standard material. However there some small differences:

- there is no separate “Smoothness” slider, smoothness is adjusted by specular/metallic color alpha.

- specular(metallic) color works even if specular(metallic) texture is assigned by multiplying its value.This is useful for fine-tuning your specular in Unity editor without constantly changing a texture.
- if no specular(metallic) texture is assigned, Voxeland uses albedo's alpha as specular(metallic) map.

Changing any of the properties requires rebuild operation (press "Rebuild" button above types list). To switch between from metallic setup to specular change material shader from "Voxeland/Standard" to "Voxeland/Standard (specular setup)". Material could found in Settings foldout.

To add new blocks click "+" button below blocks list. To select a block click it's texture icon to the left of block's name. For convenience only selected block's properties are displayed. To remove block press "✕" button below list.

Not all the block types should necessarily have terrain. For example, tree and light blocks from demo have no terrain, but have prefabs. For every block of such type new object is created in scene and removed if the block does not exists anymore. Combined blocks with prefab and terrain are also possible: this could be useful for creating tree roots or some kind of ore blocks.

As you may have noticed from the Voxeland gradual build, all the terrain is split into a separate meshes - chunks. Chunk size is 30 units in width and length by default. Each chunk is rendered in one draw call. When the terrain changes, only the changed chunks rebuild, leaving all other terrain untouched.

Adding Grass

Grass types list is made the same way as the terrain types list: to add a new block press "+" button, to remove it press "✕", to select - click on it's icon. To use the grass you should have a modelled grass mesh (however it could be taken from demo scene).

Grass mesh should have size of 1 unit in horizontal (X and Z) dimensions. It's pivot should be placed in it's center (i.e. 0.5 X and 0.5 Z), but on ground level (0 Y). To use Voxeland's nature shader with vertex animation mesh should have secondary uv channel. Vertex animation amplitude depends on it's secondary uv Y coordinate - the higher the vertex on mapping the more animated it will be.

Note that grass data structure is not 3D, but 2D. It means that if grass once placed in some X and Z coordinates it always be displayed even when terrain height will change.

Grass grows above terrain block types with "Grass" parameter turned on only. This is useful if some of terrain types such as cliffs should not have grass. Grass tint parameter of terrain type multiplies grass color: use it to create different grass tones using same grass texture.

Dynamic build

Turning off “Limited terrain” toggle in Settings will turn on terrain dynamic build algorithm. The terrain will not be build all at once, but will build new chunks as camera comes closer to them and will remove chunks that become far from camera. Chunk build and remove distances are controlled by “Build Distance” and “Remove Distance” respectively.

Dynamic build considers the position of scene view camera when the playmode is off, and and scene main camera when playmode is on. It does not matter if “Game” or “Scene” view is currently selected.

It is necessary to distinguish dynamic terrain generation from dynamic build. Dynamic build determines the appearance of the terrain only by hiding or showing it’s chunks, it does not create any new terrain or does not affect terrain data in any way.

On a new terrain only the small area is generated and filled with a “flat” land, but dynamic build can also show non-generated areas. When Voxeland encounters non-generated column (i.e terrain with one X and Z coordinates) it displays it like specific “default” column. “Display non-generated terrain level” parameter in settings will control the height of this column and it’s block type. Visually it will flat fill all non-generated terrain with given height and type.

A performant way to increase view distance is using “horizon plane” mesh. This is a special non-voxel terrain to be displayed at a far distance, and it works very similar to the standard Unity terrain. Voxeland takes a large flat mesh, extrudes it’s vertices to the height of the voxel terrain, and hides triangles that are filled with chunks that are already build. It is not a real voxel terrain, but a good mean for displaing the landscape in the distance. To use the horizon plane turn it’s toggle on in the Settings, assign mesh and set non-zero scale.

Terrain Generator

Voxeland has a terrain generation algorithm called “Generator”. It can create new land using fractal perlin noise, erosion, and other terrain, grass and forest algorithms.

Terrain generation is done per-area. Area is a terrain region about 300-500 units in length and width (the exact area size could be set in Settings). A limited terrain has only one area. Area does not wired to chunks in any way - one chunk can be set on the seam between two or even four areas. To select area for terrain generation open Generator foldout (by default currently selected area will be highlighted), place your scene view camera on a desired position, click “Get current area”. To generate an area using default settings press “Generate current area” button at the bottom of Generator foldout.

Each generation algorithm has one common parameter - type. This parameter determines which block type will be used to fill newly generated terrain. For all other parameters see this readme or tooltips.

Generator can create terrain automatically, when camera reaches new un-generated areas. This feature works both in edit mode and in playmode. Currently selected generator parameters are used to generate new areas. Note that by default auto-generated areas are not saved to data: each time you load your scene (or compile scripts) auto-generated areas will be generated again. This is done so that the size of the data did not exceed reasonable limits as you travel across the world.

However areas will be saved if “Save Auto Generated Areas” toggle is checked. The other way to save auto-generated area is to set a block anywhere in it - when the area terrain is changed area will be saved.

General properties

Brush Size: The extend of the brush. If size is more than 0 a volume brush is turned on, which allows to set multiple blocks in one click.
Spherify: Smoothes volume brush and shapes it in form of pseudo-sphere. Please note that Voxeland data is a cubical structure which cannot give clear spheres – all round objects will become a bit aliased.
Rebuild button Removes all chunks and creates new terrain mesh. This will not delete terrain data (unless terrain size is changed), it just re-creates same terrain with new settings. Rebuild operation is performed after each script compile. Every time settings or type parameters changed rebuild button should be pressed.

Terrain types

Terrain Check this the block is filled with land stratum, and uncheck if this block should not have any terrain filled (like a tree or light from demo). Every filled block has a list of parameters, that match Unity Standard material parameters. For parameter changes to take effect perform Rebuild.

> **Color** Color tint, multiplies texture color
> **Albedo** Material albedo (RGB), and Gloss/Metallic (A) if material does not have Specular/Metallic map
> **Normal Map**
> **Spec/Metal, Gloss Map** Specular or Metallic (RGB) (depending on shader setup) and Smoothness (A). Using Albedo alpha as Spec/Metal if Spec/Metal map is not assigned.
> **Spec/Metal, Gloss Color** Specular or Metallic (RGB) (depending on shader setup) and Smoothness (A)

Tile Number of times texture repeated per 1 world unit

Grass Turn this on if block has standing grass above. The type of grass is set using Grass section. You can place different grass types on the same blocks.

Grass Tint Grass growing above this block will be multiplied by this color. Use white to leave the grass unchanged.

Prefabs Objects could be placed or removed using terrain editor, for example tree or light from demo. Placing object with prefab will instantiate prefab in scene. Each prefab section has a list of different possible prefabs. Objects are taken from the list at random.

Occlude Ambient This block will occlude ambient if it is turned on.

Grass

Vertical grass meshes. Grass is set per-column (in 2D), not per-block (3D).

Grass meshes could be placed above block types that have Grass feature enabled. This feature is turned off by default.

Mesh Grass mesh. Recommended mesh is 1-unit in diameter with pivot positioned in center on ground level.

Material Grass material. Use Voxeland Grass shader to use the functionality of grass tint."

Take Terrain Normals If turned on grass mesh will use underlying terrain normals instead of original mesh normals.

Generator

Algorithms for terrain generation. Terrain generation is done per-area (are size could be set in settings). Depending on the generation mode generated area will or will not be saved to terrain data. If area is not saved it will be generated every time game started (or scripts compiled). Area will be saved if area block is set or remove. Saving area increases data size.

Get Current Area: Gets area where camera is currently positioned as currently selected area. Currently selected area will be generated on pressing Generate button.

Highlight Selected Area: Highlights area which is currently selected. Currently selected area will be generated on pressing Generate button.

Seed: Number to initialize random generator. With the same seed and noise size the noise value will be constant for each heightmap coordinate.

Clear Area before Generating: Will generate new terrain instead of instead of the existing one. If turned off generated terrain will be _added_ to the existing terrain.

Level:

Generates flat homogeneous bedrock.

Type: Type that will be used to fill terrain by current generator.

Level: Terrain will be filled with the selected type up to this level.

Texture

Imports a heightmap. Then this heightmap could be processed with noise, erosion and other generators.

Texture: Heightmap to import in PNG format. Red channel is used. Texture read/write attribute has to be enabled and texture compression should be set to Automatic Truecolor.

Noise:

Fractal perlin noise algorithm. It can help to create base terrain mountains and hollows.

Type: Type that will be used to fill terrain by current generator.

Noise Amount: Magnitude. How much noise affects the surface.

Noise Size: Wavelength. Sets the size of the highest iteration of fractal noise. High values will create more irregular noise. This parameter represents the percentage of brush size.

Detail: Defines the bias of each fractal. Low values sets low influence of low-sized fractals and high influence of high fractals. Low values will give smooth terrain, high values - detailed and even too noisy.

Uplift: When value is 0, noise is subtracted from terrain. When value is 1, noise is added to terrain. Value of 0.5 will mainly remain terrain on the same level, lifting or lowering individual areas.

Ruffle: Adds additional shallow (1-unit) noise to the resulting heightmap.

Glen:

Generates flat plains between mountains.

Type: Type that will be used to fill terrain by current generator.

Number: Number of flat plains.

Min Radius: Minimum radius of the plains.

Max Radius: Maximum radius of the plain.

Opacity: Flatness of the plains. Value of 1 will make plains absolutely flat, value of 0 will make plains not noticeable.

Fallof: Size of the noise gradient between plain and surrounding mountains. Value of 1 will make glens perfectly circular.

Fallof Noise Size: Size of the noise on the fallof area. Higher values will make glen forms more complex.

Depth: Underground depth willed with selected Type stratum under the glen.

Erosion:

Erosion that is caused mainly by water factors - rains and torrents. It will erode terrain (make a little canons where torrents flow) and return raised sediment in hollows. Moreover, it uses wind algorithm on convex surfaces - because hydraulic erosion does not work properly without wind. This is the most slow generation algorithm.

Type: Type that will be used to fill terrain by current generator.

Iterations: Number of algorithm iterations. Higher values will make terrain more eroded. Lowering this value and increasing amounts can speed up terrain generation, but will affect terrain quality.
Terrain Durability: Baserock resistance to water erosion. Low values erode terrain more. Lowering this parameter is mainly needed to reduce the number of brush passes (iterations), but will reduce terrain quality as well.
Fluidity Iterations: This parameter sets how liquid sediment (bedrock raised by torrents) is. Low parameter value will stick sediment on sheer cliffs, high value will allow sediment to drain in hollows. As this parameter sets number of iterations, increasing it to very high values can slow down performance.
Erosion Amount: Amount of bedrock that is washed away by torrents. Unlike sediment amount, this parameter sets the amount of bedrock that is subtracted from original terrain. Zero value will not erode terrain by water at all.
Sediment Amount: Percent of bedrock raised by torrents that returns back to earth) Unlike erosion amount, this parameter sets amount of land that is added to terrain. Zero value will not generate any sediment at all.
Wind Amount: Wind sets the amount of bedrock that was carried away by wind, rockfall and other factors non-related with water erosion. Technically it randomly smoothes the convex surfaces of the terrain. Use low values for tropical rocks (as they are more influenced by monsoon, rains and water erosion than by wind), and high values for highland pikes (as all streams freeze at high altitudes).
Smooth: Applies additional smoothness to terrain in order to fit brush terrain into an existing terrain made with Unity standard tools. Low, but non-zero values can remove small pikes made by wind randomness or left from water erosion. Use low values if your terrain heightmap resolution is low.

Forest:

Places trees using soil accordance. This generator literally plants some initial trees. If soil is suitable for tree it continues to grow and even gives sprouts, forming groups and groves.
Type: Type that will be used to fill terrain by current generator.
Initial Count: Initial tree count. Randomly places this amount of trees on terrain surface, no matter of soil.
Iterations: Number of iterations during which tree can die,survive or breed.
Tree Distance: How far trees can breed saplings.
Soil Type Quality: How soil is suitable for tree. Value of 0 will kill tree in first iteration. Value 1 will always make tree breed.

Grass:

Plants a grass using height and noise algorithms. It also creates grass under trees.
Type: Type that will be used to fill grass.
Height Min: Grass starting level. All the terrain below will have grass with maximum chance.

Height Max: Grass end level. All the terrain above will have grass with minimum chance.
Height Min Chance: Grass density on the Min Level.
Height Max Chance: Grass density on the Max Level.
Spread Iterations: Spread will select grass block (bush) at random and will create grass blocks around it. Iterations is a number of blocks that will be spread.
Noise Size: Additional fractal noise algorithm. Noise size determines size of the grass groups and spaces.
Noise Density: The intensity noise algorithm is influenced on final the grass density.

Generate Selected Area: Manually generates currently selected area. To select are use Get Current button. Generated area will be saved to data.
Auto Generate New Areas in Playmode: Will automatically generate new areas when they get in build distance range in playmode and in final game build. Generated areas will not be saved to data.
Auto Generate New Areas in Editor: Will automatically generate new areas when they get in build distance range in editor only. Generated areas will not be saved to data, unless toggle below is checked.
Save Auto Generated Areas: Areas generated **in editor** will be saved to data if this toggle is checked. Saving areas increases data size.

Settings:

Various Voxeland parameters. For the settings changes to take effect perform Rebuild operation.
For the settings changes to take effect perform Rebuild operation.
Data: Terrain voxels data. Think of it like a lossless-compressed 3d image, with block types instead pixel colors.
Save: Terrain data could be saved in separate .asset file. When it is not saved to file Data is stored in current scene.
Limited Size: When this toggle is checked terrain size is limited by the size of one area. This is useful for small terrains that do not require dynamic build or horizon plane. Moreover, only limited-size terrains could be baked to meshes.
Terrain Size: Size of the limited terrain. Internally it is the size of terrain area. Resizing terrain will destroy all the terrain data.
Area Size: Size of the data area. Terrain generation is done per-area. Big areas increase data size, small areas increase generation time.
Build Distance: Chunks that will get in this range from camera will be automatically build. When camera moves, new chunks will be built as they get in range.
Remove Distance: Chunks that are further from camera that this distance will be destroyed. As camera moves new chunks will be destroyed as they get out of destroy range. Remove Distance should be larger than Build Distance
LOD Distance: The distance of turning high-poly chunk mesh off and enabling it's low-poly version, which has 4 times less triangles.

Chunk Size: Dimensions of a terrain element mesh. Higher values will speed up the whole terrain building and reduce the number of draw calls but will slow down the terrain editing.
Terrain Margins: Chunk invisible faces overlap. Lowering this value will speed up chunk build, especially on small chunks, but will open gaps between chunks. Setting this value more than 2 does not have a sense.");
Playmode Edit: Enable terrain editing in playmode. Adds a possibility to edit terrain in-game the way it is done in editor. It is recommended to use special Edit() controller instead (see Demo VoxelandController example script)
Multithread (experimental): Can cause Unity crash. Use it on your own risc.
Save Meshes with Scene: Saves terrain mesh with scene. Turn this off if you do not plan to bake your terrain as terrain rebuilds its mesh any time the scene loaded anyway.
G Key Focuses on Brush: Works like an F key, but centers camera pivot on current brush position, not the whole terrain.
Switch LODs Removing Material: Slower, but more convenient (it does not show collider wireframe) way to switch lods. Turn this toggle off when creating release build.
Disabling will show collider wireframe.
RTP Compatibility: Makes Voxeland compatible with Relief Terrain Pack 3. For more detail see this manual below.

Display non-generated Terrain:

The way non-generated terrain is displayed. Sets the default height and type for all non-generated areas.

Ambient Occlusion:

Calculates ambient occlusion using block structure immediately on terrain change. Turn it off if you plan to use baked or realtime GI.
Fade: Ambient occlusion blur. Higher values will bright the dark hollows.
Margins: Increase this to avoid seams in ambient between chunks. Higher values will give less artifacts but will slow down calculations.

Horizon Plane:

Planar (non-voxel) terrain to display on far distances as lod. Horizon Plane could not be used without dynamic build on limited terrain.
Mesh: A flat mesh that is used as a horizon plane base.
Size: Scale of far mesh.

Terrain Material: Material of the terrain. Double-click it to set terrain material parameter, or assign a new terrain.
Highlight Material: Material of the highlight object. Highlight color or decal dist could be set by modifying this material.
Clear Terrain: Clearing terrain will remove all terrain data. This operation cannot be undone.

Import and Export

Center v3.0/3.1 Limited Data: When opening old limited terrain scene you can notice that it's position is shifted at half of the area. Use this to fix terrain position. This will clear all terrain areas except this one. This operation cannot be undone.
Import Heightmap: Obsolete. Use Texture Generator instead.
Load TXT: Loads string data from text format.
Bake meshes to Scene: Bakes terrain to Unity meshes and turns Voxeland script off. Meshes are saved to current scene. Note that terrain should not be infinite.
Bake meshes to Assets: Bakes terrain to Unity meshes and turns Voxeland script off. Meshes are stored in separate .asset files. Note that terrain should not be infinite.
Bake Lightmap: Will generate lightmaps when baking terrain. Enabling Lightmaps can greatly increase baking time.
Save Data Asset: Saves Voxeland data to .asset file.

Upgrading v3 terrain

On open your terrain saved using older Voxeland version you will see that it does not display. To fix this use these steps:

- For all of the terrain block types that should have a terrain:
    - Turn on “Terrain”;
    - Turn on “Occlude ambient”;
    - Set terrain color to white;
    - Set terrain tile to non-zero value;
    - Set grass tint to white;
    - Different top texture feature is not available anymore due to switch to standard shader, so you’ll have to choose what texture should be used in this type and reassign it if needed;
- For all of the prefabs type turn “Prefabs” on and assign prefab to list;
- Assign grass mesh to all the grass types. You can use Demo/Meshes/Grass. Turn on “Take terrain normals”;
- If you had an infinite terrain: Turn off “Limited size” toggle in settings;
- If your terrain was limited: Set chunk size to 32 or 16;
- Press “Revert” button next to Terrain Size / Area Size field;
- Set horizon plane mesh and size if this feature was on;
- Press Rebuild button.
- If your terrain is shifted on half of the area, press “Center v3.0/3.1 Limited Data” button in Import foldout, then rebuild.

Working with RTP

To make Voxeland work with RTP3 perform these steps:

- Turn on RTP Compatibility in Settings;
- Add Relief Terrain component to Voxeland object;
- Rebuild Voxeland terrain;
- Voxeland object now has a material panel. Assign **ReliefTerrainPMTriplanarStandalone** shader to this material.
- Edit the shader: comment the line 326: **#define WNORMAL_COVERAGE_X_Z_Ypos_Yneg.** To comment the line add **//** int the beginning: //#define WNORMAL_COVERAGE_X_Z_Ypos_Yneg

Most useful scripting functions

public byte VoxelandTerrain.GetBlock (int x, int y, int z)

Gets number of block type at specified coordinates.

public void VoxelandTerrain.SetBlock (int x, int y, int z, byte type)

Sets block of given type number at specified coordinates.

public void VoxelandTerrain.SetBlocks (int x, int y, int z, byte type=0, int extend=0, bool spherify=false, SetBlockMode mode=SetBlockMode.none)

Sets a cube (or sphere) of blocks given type with the center at specified coordinates. Extend determines the radius of the sphere (or half side of the cube). To set the single block use zero extend. Spherify parameter determines whether a brush would be a sphere or a cube. SetBlockMode is a mode used to set block:

- none: no block will be set, just terrain refresh
- standard: sets all the blocks within extend the way it is done in SetBlock function
- replace: sets block only at the coordinates filled with terrain. This will replace the existing blocks with Type, leaving non-filled blocks untouched
- blur: does not set any blocks, but blurs existing blocks within extend

public void VoxelandTerrain.Edit (Ray aimRay, int type, bool add=false, bool dig=false, bool smooth=false, bool replace=false)

Edits terrain using ray. This function is useful if you wish to remap default Voxeland playmode terrain editing controls. Edit brush is positioned at the intersection of aimRay with terrain. Type is currently selected edit type. Booleans determine an editing mode.

When none of the booleans is checked terrain is not edited, just a brush displayed. See Demo VoxelandController for usage example.

public Data VoxelandTerrain.data

Internal terrain data used to keep all the matrix of voxel types.

public List<byte> Data.SaveToByteList ()

Saves current data to a list of bytes. This list is a compressed and compact data information. Use it to to make game saves or synchronize data over Internet.

public void Data.LoadFromByteList (List<byte> byteList)

Loads saved list of bytes to data. This will remove all previous terrain data. Please note that data area size on saving and loading list should be the same.