Procedural Building Generation

COSC450 Assignment 1

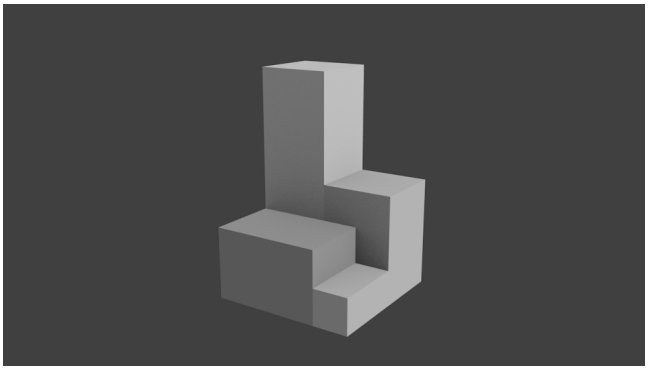
Due: Monday 3rd April 2017, 11:59pm

This assignment is worth 20% of your final grade.

1 Overview

While hand-crafted models and animation are common, it is not always practical. In many situations it makes more sense to algorithmically generate models for backgrounds and large-scale simulation. Examples of this include Weta Digital's MASSIVE software for crowd simulation; level and world generation for games such as Diablo, Minecraft, and the Civilization series; and Lindenmayer systems (L-Systems) for modelling plants from formal grammars.

Blender, an open-source 3D modelling package, includes the ability to script almost all of its operations through a Python interface. This makes it relatively easy to create procedural models. In this assignment you will write a script to generate procedural buildings. A sample script which generates rectangular blocks which might be seen as extreme examples of Brutalist architecture is shown below. More realistic buildings can be generated by adding more complex geometry, materials, textures, and so forth.



2 Assignment Requirements

For this assignment you must write a Blender script that procedurally generates buildings. For high marks your buildings should:

- Show plenty of variation when the script is run multiple times, or when several buildings are constructed.
- Have a consistent appearance, which may or may not be based on a real-world architectural style. If several buildings are generated, they should look like they belong together.
- Make use of a variety of techniques to achieve detailed and plausible buildings. Note that realism is not essential – it is OK to generate buildings that have a distinctive visual style.

Your buildings should be generated by a Python script from within Blender, without intervention from the user. You may wish to include parameters in your model. If you do so you should include clear instructions as to how to set these parameters in your report.

The sample script has a simple structure consisting of three main components:

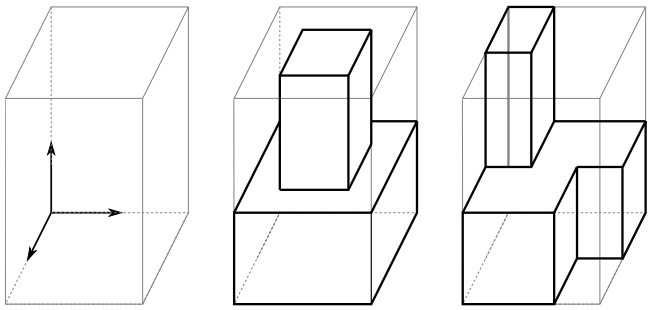
- `Init`, which is used to set up the scene and any data (such as materials) which are common across buildings.
- `Buildings` which contains the key function, `create()`.
- `Render`, which creates a camera and renders the scene.

The sample script gives examples of how to use these modules and their functions, but the main function you need to reimplement is `Buildings.create()`. This takes the following parameters:

- The size of a bounding box (`xSize, ySize, height`) that the building must fit within.
- The `name` of the building to create.

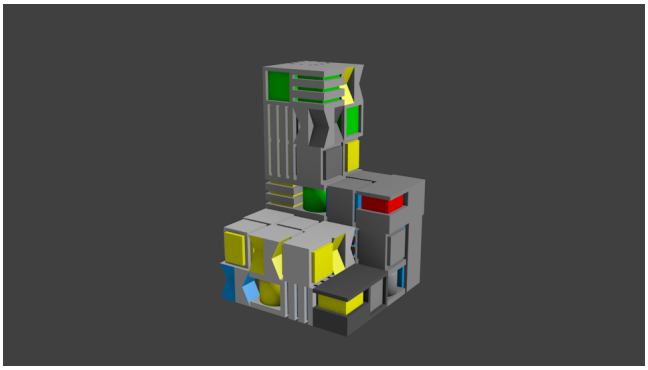
`Buildings.create()` should return a *single object* called `name` which can be transformed to position the building. In the example script, an empty object is used for this purpose, and the actual geometry of the building has this object as its parent.

The bounding box starts at the origin, and extends along each positive axis by the quantities specified. The dimensions of the box will be small positive integers. The building does not need to completely fill the bounding box, but should extend to all of the faces of the box at some point:

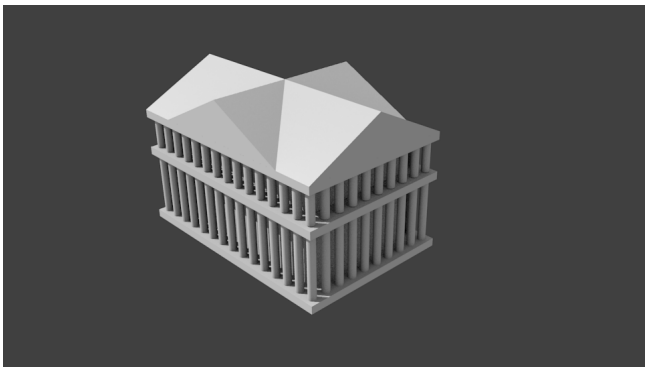


Your script may refer to image textures etc. as appropriate, and these should be included in your submission and referred to using relative paths, so that I can run your script. Your script should also include a significant random element so that different buildings are generated each time it is run.

The requirement for a random element does not mean that every aspect of your building needs to be generated at random. You can use pre-constructed elements where appropriate. For example, random selection from a set of pre-fabricated components can be used to create a Modernist appearance:



The interpretation of the dimensions of the building is up to you, but your script should work for a variety of dimensions, from $1 \times 1 \times 1$ up. One way to think of the height is as the number of storeys or floors for the building. This could be useful for styles of architecture that have natural layers, such as a more Classical look:



As well as your code you will need to write a short report describing your work. Your report should include:

- Clear directions for running your script, including any parameters that can be changed.
- A discussion of how your buildings are constructed, and how this is reflected in the design of your script.
- Examples illustrating the range of buildings that can be generated by your script, and a discussion of the types of variation that can be expected between different invocations of your code.

3 Example Scripts and Resources

Examples of Blender scripting are available on the departmental machines from `~steven/Public/COSC450/BlenderScripts`. The first, `script.blend`, is the example talked about in lectures. The second, `planets.blend`, is a more complex example that makes a random number of planets around a star. The second script demonstrates loops, noise- and image-based textures, lighting, and camera positioning. For the image-based textures, you will need the file `mercury.png` from the same directory. The final script. `buildingSample.blend` should be used as a basis for this assignment.

More information about Blender and its scripting interface can be found online. Remember that there were significant changes in Blender between versions 2.4 and 2.5, so older material may not be relevant. Some specific resources that may be of use are:

- The Blender reference manual – http://www.blender.org/manual/.
- Blender tutorials – https://www.blender.org/support/tutorials/.
- The Blender Scripts Cookbook – http://wiki.blender.org/index.php/Dev:Py/Scripts/Cookbook.

- Free books about Blender – http://wiki.blender.org/index.php/Doc:2.6/Books.

Inspiration might be found from a variety of sources, but there are many different architectural styles that you could base your buildings on. Here are some images that might be useful inspiration, but make whatever type of building interests you.

4 Deliverables

You should send your report (PDFs preferred), Blender file, and any supporting files (textures etc.) required to steven@cs.otago.ac.nz. You may find it convenient to archive your files as a `.tar` or `.zip`, but be aware that archives sent from outside the department often get caught in the University's spam filters. This can be easily circumvented by renaming your files to remove the archive extension. If your submission is zipped up as `astudent450ass1.zip` then just rename it to `astudent450ass1.piz` or something and tell me what you've done in the body of your email. I will reply to acknowledge receipt of your assignment in any case.

4.1 Marking Scheme

This assignment is fairly open, so a precise marking scheme is hard to give. I will be dividing marks roughly as follows:

- 30% for the visual appearance and complexity of the generated buildings. As well as interesting and varied buildings, I will be looking for good use of materials, textures, and lighting.
- 20% for your code. Clarity and correctness are the main concerns here. Comments, naming conventions, and appropriate division of code into functions and modules all help with this. While efficient methods are preferred, you should worry primarily about making sure that your script does what it should, and that this is clear to people reading your code.
- 40% of the marks will come from your report. Clarity of expression is again important, and this is where you demonstrate that you have fulfilled the requirements discussed in Section 2. You should make appropriate use of images and diagrams. References should be given to sources of inspiration, and to any third party libraries, code, algorithms, or other material used in your assignment solution. I would expect that reports will typically be about 6-8 pages long, although large numbers of images could extend this.
- 10% of the marks are for extras beyond a basic building. This is particularly open-ended, and these marks will come from a combination of the generated buildings, your code, and your report. Some possibilities for an 'extra' could be:
  - Modelling buildings with some interior detail and glass windows so you can see it.
  - Using a regular expression or other formal grammar to generate buildings. Ideally this would allow you to generate different types of buildings by changing the grammar.
  - Making a particular effort to model a specific style, such as Gothic, Art Deco, or Murghal architecture.

Late assignment submissions will be penalised at the rate of 10% per working day. Extensions to the deadline may be granted where appropriate, but should be sought well in advance. The usual university regulations relating to academic integrity apply (`http://www.otago.ac.nz/study/academicintegrity/`), and any work you submit must be your own, or be clearly attributed to the original author.