Assembling Environments with LEGOscape

Joseph Hegarty Bryan Smith Jens Jebens John Paul Molloy
Animal Logic*

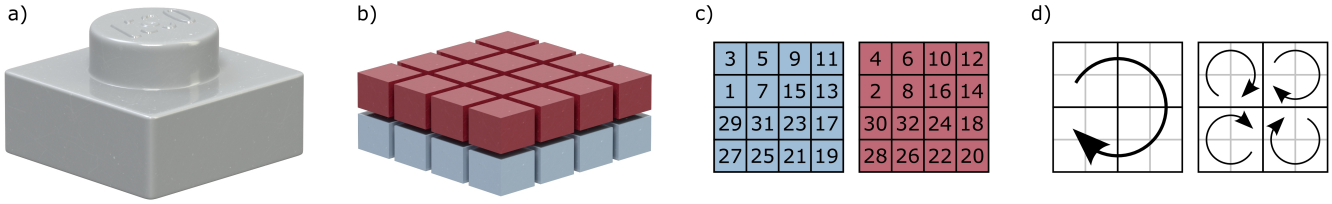

Figure 1: *a) 1x1 plate, b) 32 sub-voxels per voxel, c) Bit index for each sub-voxel, d) Two-level spiral ordering*

Abstract

When creating props and set pieces for *The LEGO Movie*, modelers would use *LEGO Digital Designer* to place each individual brick. While this gave complete control over the type and placement of every brick, it did not scale well for large objects such as terrains, canyons, or mesas. To allow users to rapidly create these large objects we created a procedural brick placement system called LEGOscape.

1 Overview

LEGOscape takes a mesh representation of an object and iteratively fills it with bricks. Each iteration fills the mesh using a pattern, which is an infinitely repeating arrangement of bricks of a single type. Patterns have a number of attributes used to control their arrangement such as shift, stagger, and spacing. Bricks defined by a pattern are inserted into the world if they are inside the mesh surface and either do not intersect with any existing bricks or, if they do intersect with existing bricks, result in a better fit. In the latter case, the newly added better fitting brick will evict any existing bricks which it intersects with.

2 Data Representation

Initially only rectangular bricks were supported, so it was natural to use the smallest LEGO brick as the voxel unit, the 1x1 plate (Fig.1a). Internally LEGOscape used integers rather than floating-point numbers to represent locations in space, with one unit along each coordinate axis mapping to the corresponding dimension of the 1x1 plate. The integer representation made indexing into the world precise and efficient. Bricks placed in the world would add themselves to a boolean-valued occupancy grid, while mesh volumes were rasterised into a boolean-valued terrain grid. Patterns would examine these two grids when adding bricks into the world.

Once we had the basic system up and running the next step was to add support for angular bricks, which are crucial in modeling flowing landscapes for example. Rather than move to a floating-point representation, we simply supersampled the voxels into sub-voxels. Conveniently the dimensions of the 1x1 plate, being roughly twice as wide and deep as it is tall, naturally lead to a 4x4x2 supersampling, which gave 32 sub-voxels per voxel (Fig.1b). The 1x1 plate remained the coordinate unit inside LEGOscape, however we now represented each voxel as a 32-bit integer, with each bit corresponding to a sub-voxel. This allowed us to perform comparisons between bricks and grids efficiently using standard bitwise operators:

```
bv1 & bv2; // do bricks intersect?
countBits(bv1 ^ bv2); // diff in sub-voxels
bv & ~tv; // does brick breach terrain?
```

In general, LEGO bricks can be placed in one of four orientations. While rectangular bricks often possess symmetries that make their appearance invariant to rotations, this is rarely the case for angular bricks. We needed a way to rotate voxels efficiently for use in comparisons. By using a two-level spiral bit order (Fig.1c and Fig.1d) voxels could be rotated using the following bit operations:

```
voxel <<  8 | voxel >> 24; //  90 degrees
voxel << 16 | voxel >> 16; // 180 degrees
voxel << 24 | voxel >>  8; // 270 degrees
```

3 Modeling Workflow

Although modelers wanted a procedural tool to help them fill large objects with LEGO bricks, they did not want to give up control of brick selection and placement. Generally it was sufficient for flat regions to have varying sized unaligned rectangular bricks, but the precise placement of angular bricks was very important when maintaining the flow of a model.

To do this, instead of manually placing each brick, modelers would manipulate the mesh surface interactively. Since LEGOscape fills the mesh volume as tightly as possible, by manipulating the shape of the mesh modelers could coax LEGOscape into placing specific bricks in specific locations very effectively.

4 Rendering

Although LEGOscape is used interactively through the use of region of interest bounds, for very large objects using many patterns it could take more than a minute to process. Rather than pay this cost per frame, the brick data was baked into a particle system, which was then rendered using our brick instancing PRMan procedural.

*{joeyh, bryans, jensj, johnpaulm}@al.com.au

SIGGRAPH 2014, August 10 – 14, 2014, Vancouver, British Columbia, Canada.

ACM 978-1-4503-2960-6/14/08