Procedural Mosaic Arrangement in *Rio*

Rhett Collier[*]
Blue Sky Studios

Josh Smeltzer[†]
Blue Sky Studios

Figure 1: *A sidewalk and cobblestone street.*

1 Mosaics: A Textural Challenge

The sidewalks of Rio de Janeiro are decorated with mosaic tiles arranged in designs known as *Portuguese Pavement*. The tiles range for miles along the Copacabana, Ipanema, and city streets. For the movie *Rio*, we wanted to reproduce this hand-crafted look, and in particular, we had to achieve the feeling of millions of tiles that had been placed in a non-repetitive, patternless way. We further had to be able to use the same technique whether the camera was focused tightly on the surface, or broadly in an overhead shot.

Our technique consists of two steps. In the first step, we generate tile positions, shapes, and orientations and write them out in vector form to a file. In the second step, we read in the vector data from a file, and use it along with noise shaders to create the finished surface. The tile surface would have been prohibitive to texture in traditional ways. An army of texture map artists would have been needed to create the first revision, let alone to rework the maps to achieve the evolving look from Art Direction. Cobblestone streets presented a similar problem. To our knowledge, *Rio* has one of the first uses of iterative, procedurally arranged mosaics in the history of feature film CGI.

2 Packing The Mosaic

Artists defined the color region boundaries with NURBS curves in order to have the maximum degree of control over the design. The packing process involves a sequence of passes of tile creation, deformation, translation and deletion. The process runs off-line and stores the tile vector data in a file as a disjoint set of quadrilaterals. The packing algorithm distributes tiles to areas of the region that are less likely to be occupied by existing tiles. The algorithm deletes any candidate tile that intersects with an existing tile or region boundary. The first pass populates the inner side of the region boundaries with edge-oriented tiles, expanding them until they collide with their neighbors. The second pass evenly distributes a set of oversize tiles throughout the region. The third pass populates tiles into the gaps between the oversize tiles, then subdivides them into 4 tiles each. Once the algorithm populates a moderately sparse set of tiles, it iteratively expands, collides, subdivides, and contracts the tiles until the grout width is the goal thickness and the tiles have the goal area. The algorithm steps each of the quadrilateral's vertices outward independently, checking for intersection with neighbor tiles at each step. Since the tiles are represented by quads in 2D, the algorithm approximates intersection testing by checking two neighboring tiles' sides for line intersection. At times during the process, is it necessary to defer some rules from being enforced until later. For example, the algorithm defers maintaining tiles' vertices within the NURBS curve boundary until the last step. The process then enforces the rule to bring the vertices back inside the region.

3 Rendering Mosaics as a Shader

An image map of sufficient resolution would have been memory-prohibitive due to the large surfaces to be covered and the proximity of the camera to the surface. The solution is to project the disjoint quadrilateral representation of the tiles onto the 3D surface by means of a vector shader. This is visually similar to shadow casting, and produces an alpha-like signal which the renderer processes ray-by-ray as an input to a shader network to perturb normals and introduce believable stone texture, etc. The vector shader produces a signal with rounded tile corners with a specified radius, in order to better approximate the natural corners of the tiles and cobblestones.

4 Material or Geometry

Once the packing algorithm establishes the footprint of the tiles, they can be rendered as a shader network or as procedurally generated subdivision-surface geometry. The footprint of the shader-based tiles can interlock with the geometry-based tiles. The algorithm "dithers" the transition zone between geometry and shader to make the transition more difficult to read than that of a hard line. For cobblestones, the process populated geometry-based cobbles the general location of the camera, transitioning to shader-based cobbles farther away.

[*]e-mail:rhett@blueskystudios.com
[†]e-mail:jsmeltzer@blueskystudios.com

SIGGRAPH 2011, Vancouver, British Columbia, Canada, August 7 – 11, 2011.
ISBN 978-1-4503-0921-9/11/0008