# Real-Time Bump Map Deformations

Pawel Wrotek, Alexander Rice and Morgan McGuire* Brown University

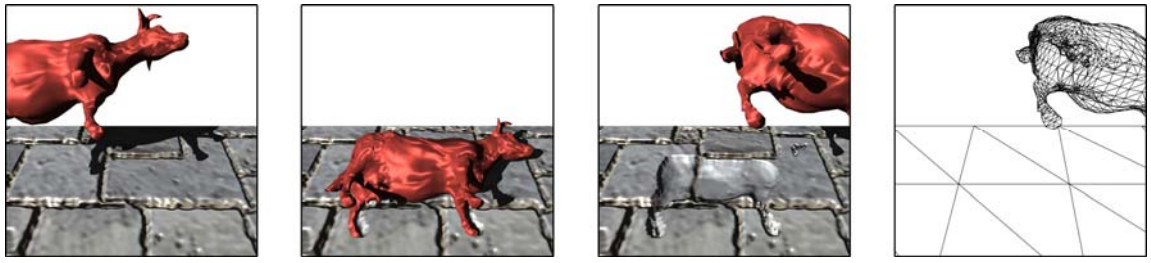

Figure 1: A cow collides with a bump mapped surface, resulting in a dent. As can be seen from the last frame, the geometry remains unchanged.

## OVERVIEW

- A method for efficiently generating plausible dents and scratches on object surfaces due to collisions.
- Uses bumps maps instead of mesh deformation.
- The GPU calculates the deformation to the bump map during a collision.
- Works in real time.
- Uses a rigid body simulator based on that of Guendelman et al. [2003], with collisions detected by interpenetration using the OPCODE and G3D libraries.

## PARAMETERIZATION AND RENDERING

As a pre-process, create a parameterization for each object with a 1:1 mapping from points on the object to points on the bump map.

Use Sheffer and de Sturler's parameterization technique [2001] to produce a roughly uniform bump map resolution across the surface.

Objects are rendered using parallax bump mapping, a recent hardware-rendering trick described by Welsh [2004] that approximates both self-occlusion and shading for a rough surface (Figure 2). This method works in the following way:

1. For each point $p_1$ being rendered, the view vector $v$ is formed from the eye to $p_1$. At $p_1$, the corresponding point $b_1$ in the bump map stores a height value.
2. A point $b_2$ is found by backtracking along the view vector until the height above the surface is equal to the height stored at $b_1$.
3. The point $b_2$ is projected onto the surface of the object at point $p_2$.
4. The texture and normal values at $p_1$ are set using the values from $p_2$.

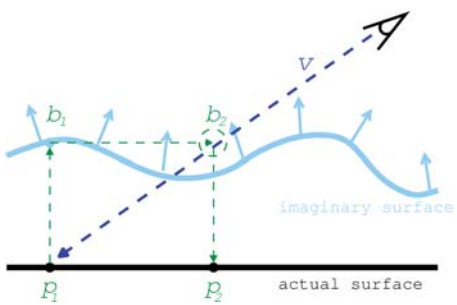


Figure 2: Parallax Bump Mapping.

## METHOD

When objects *A* and *B* collide, the location and shape of the deformation is computed by rendering their overlap to an offscreen depth-buffer as follows (all rendering done with no lighting or bump mapping):

1. Place an orthographic camera a small distance away and facing towards the collision (Fig. 3.2).
2. Clear the frame buffer.
3. Render the front faces of *A*, substituting for the color texture an address map, with color $(r, g, 0)$ at texel $(r, g)$ (Fig. 3.3).
4. Read back the depth buffer $D_0$ (which now holds the "highest" points on *A*) and the color buffer $C_A$.
5. Set the depth test to pass when the new pixel is farther from the camera than the old one (GL_GREATER).
6. Render the back faces of *B* using the address map, and set the stencil buffer to 1 wherever the depth test passes (i.e. wherever *B* penetrates *A*) (Fig. 3.4).
7. Read back the depth buffer $D_1$ (containing the "lowest" points on *B*), the color buffer $C_B$, and the stencil buffer *S*.

To modify the bump map, iterate over each pixel $C_A[i, j] = (r, g, b)$.

The color buffers give us the corresponding location in the bump map, since we rendered with the address map.

If $S[i, j] = 1$, we modify texel $(r, g)$ in the bump map, but only if this texel has not yet been modified for this collision.

The difference $D_1[i, j] - D_0[i, j]$ is the amount by which the bump map is altered at $(r, g)$.

Modifying the bump maps on the GPU is an open problem; the challenge is ensuring that the net change to a bump texel is independent of the area it affects during rendering.

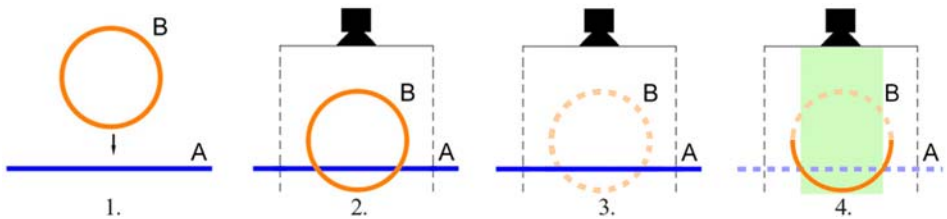


Figure 3: (1) *A* and *B* prior to collision. (2) Orthographic camera setup used to compute deformations on the GPU. (3) Front faces of *A* rendered. (4) Back faces of *B* rendered. The green box represents the pixels for which the stencil buffer is set to 1, i.e. those where *B* penetrates *A*.

Figure 4: Bump map deformations are used to create tracks in the ground and dents in the bunker's walls.

## REFERENCES

GUENDELMAN, E., BRIDSON, R., AND FEDKIW, R. 2003. Nonconvex rigid bodies with stacking. *ACM Trans. Graph. 22*, 3.

SHEFFER, A., AND DE STURLER, E. 2001. Parameterization of faceted surfaces for meshing using angle based flattening. *Engineering with Computers 17*, 3, 326-337.

WELSH, T. 2004. Parallax Mapping with Offset Limiting: A Per-Pixel Approximation of Uneven Surfaces. *Infiscape Corporation.* http://www.infiscape.com/doc/parallax_mapping.pdf.

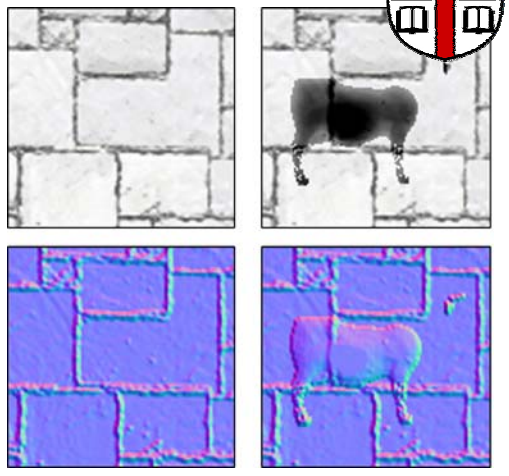

Figure 5: The bump and normal maps (above and below, respectively) for the surface in Figure 1, shown before (left) and after (right) the collision.

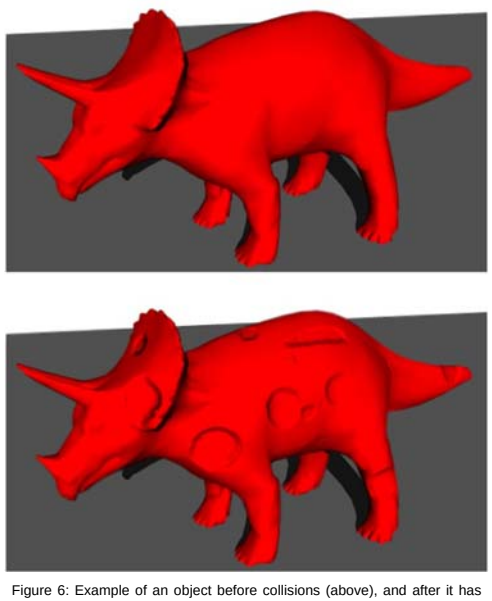

Figure 6: Example of an object before collisions (above), and after it has been hit by a number of other objects (below). All of the deformations have been performed on the object's bump map, the mesh remains unaltered.

*CONTACT ADDRESS: MORGAN@CS.BROWN.EDU