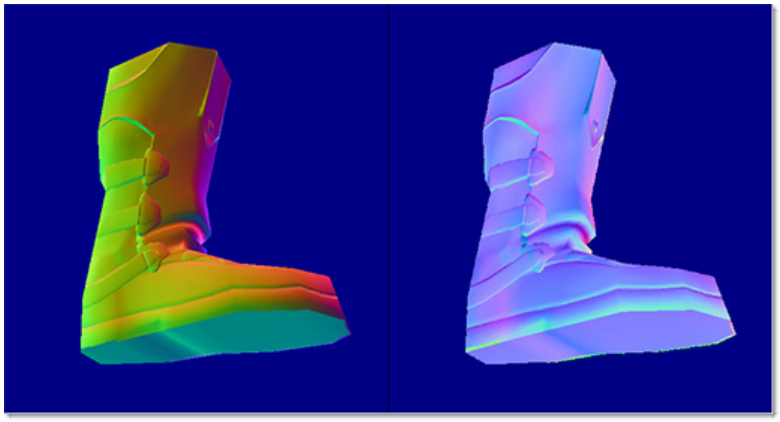
Tangent Space Normal Mapping

Created by Unknown User (martin), last modified by Adam Johnson [Crytek] on Jun 21, 2013

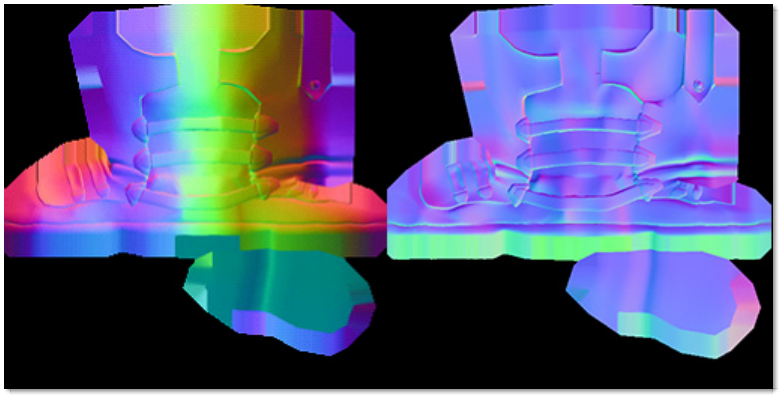
Normal Map Textures

Normal maps enable 3D graphics cards to apply bump mapping to the shading. Bump mapping is an efficient method to render surface structures like wrinkles, scratches or beveled edges much **faster than it could be done with fine tessellated geometry**. Unlike displacement mapping, it **only affects shading** and not the surface itself. The surface remains flat when seen from an angle.

In a normal map a color represents a certain **normal vector** (surface orientation of a point). For tangent space normal maps the information is relative to the underlying surface. For Object space normal maps the information is relative to the object orientation. The following pictures show the normal map texture of a low poly boot mapped as color map on the 3d object.



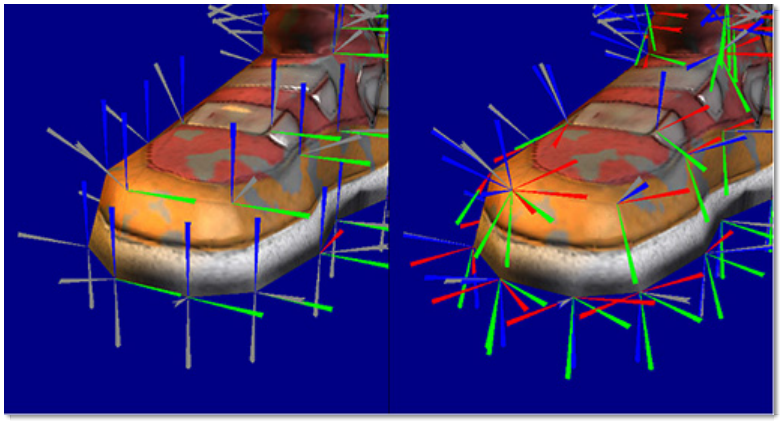
The following image shows the texture unwrapping:



A normal is a 3D vector in the range -1 to 1 and usually unit length. As conventional texture maps are in the range 0 to 1, the data needs to be adjusted to fit. **Object or world space normal maps** (left image) appear more colorful as here we store all normals on a unit sphere. The **tangent space normal map** only stores its **data relative to the surface**. Bright blue (0.5,0.5,1) means the surface is flat. The following texture represents the normals of two truncated 45 degree pyramids on top of each other:

Tangent Space Vectors Stored per Vertex

In the following image you can see the the local matrix that is required per vertex to implement object or world space normal mapping (left) or alternatively tangent space normal mapping (right):



The gray vector represents the surface normal. The red, green and blue vectors represent the basis vectors. For world and object space the data is all the same for each vertex and can be stored per object. In contrast, tangent space requires specific data per vertex. However, as the normal vector is aligned to this base, some compression is possible.

Benefits of Tangent Space Normal Maps

With the unit length property of the normal and restricting z to be always positive, we can reconstruct one component and only need to store two components (2 bytes instead of 4, one is usually wasted for alignment). There are several GPU supported formats (e.g. 3Dc in DX9 or BC5 in D3D10) for 2 channel textures (usually one byte instead of 2 bytes). As these formats **save memory and reduce memory bandwidth** and decompression costs are low as well, using them **can improve streaming and render performance**.

Tangent space normal maps are independent of the underlying geometry which means the **texture can be use on other geometry** as well. It will automatically align to the surface regardless of mirroring, rotation, scale or translation. Only the latter two are supported by object or world space normal maps. The geometry independence allows **efficient reuse**, for example in form of **tiled normal maps** that save further memory and reduce production time.

Unit Length Property

The normal length is not important for the shading but **can be a requirement for an efficient implementation**. In the Resource Compiler we re-normalize each vector and even mip maps. Normals that are smaller in length (appearing more grayish) can be used as the **RC normalized the data** but precision is slightly less (8 bit quantization issue). 8 bit normalized vectors still would cause artifacts but as the reconstruction on the z component is based on the unit length assumption we get some quality back.

Mip-mapped Normal Maps and Bilinear Filtering

Normal maps represent the surface orientation and traditional mip mapping computation is based on

weighted blends of the texture's pixels. That means with **each mip-map we lose surface details at a certain frequency**. This is often seen in speculars which are broad and noisy when nearby (high frequent normal details) but sharp and small when far away (blended normals represent a flat surface). Because of trilinear filtering or anisotropic filtering, this transition can be soft but it can be quite noticeable. The following pictures show the problem (same normal map with increased tiling):

The correct result should be a specular the has the same spread in average but instead the surface appears to be more and more polished.

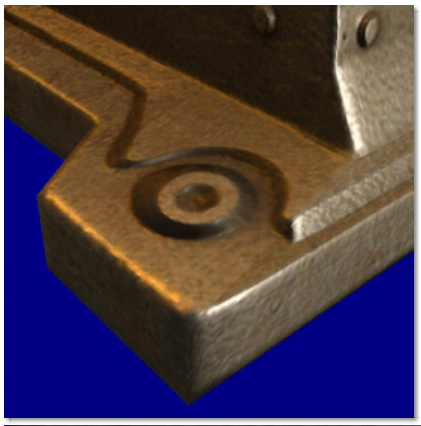
Normal Map Compression

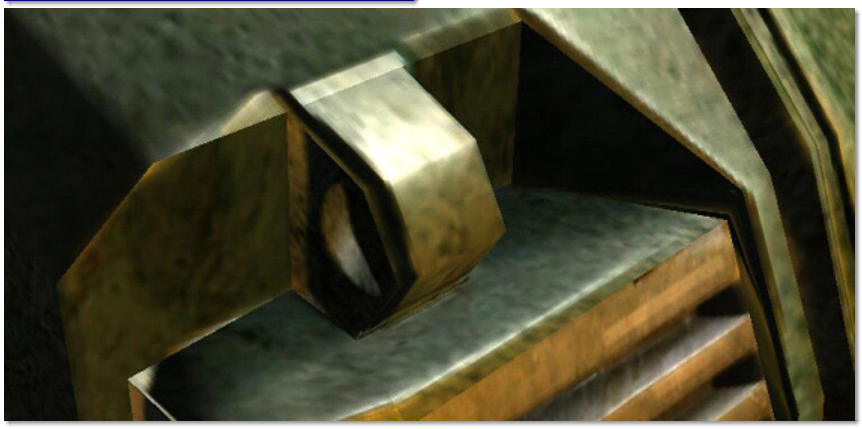
One advantage of tangent space normal maps is that the normals are always pointing outwards, so assuming unit length, the normal z coordinate can be reconstructed from the x and y components. After the coordinate expansion from 0..1 to the -1..1 range, the z component can be computed in the shader with this formula: `z = sqrt(1 - x*x + y*y)`. This makes it possible to use two-channel textures (2 bytes per texel) to store normal maps.

Usually the Resource Compiler compresses all normal maps in the 3Dc format (1 byte per texel). When no 3Dc hardware support is available, some graphics drivers can emulate it by converting back the texture to a different format (e.g. U8V8, 2 bytes per texel). On older hardware that does not have 3Dc support at all (e.g. ATI 9800), the engine will convert normal maps to the DXT5 format at loading time. However, using the DXT5 format requires different shader combinations and the quality is a bit worse compared to 3Dc.

Drawbacks of Tangent Space Lighting

Although a good implementation can limit problems, the fundamental idea of normal mapping has its flaws. Unwrapping complex models and putting seams in less exposed areas can be tricky for artists. The following images show object space normal maps (left) and tangent space normal maps (right). Here object space normal maps cause a lot less problems because bilinear filtering softens the edges and flat surfaces have simply one normal and are therefore perfectly flat. Seams on the left side are much less visible as the shading is independent from the UV layout and the triangle distortions.





Object space normal maps (left) and tangent space normal maps (right).

Further Reading

- Toksvig (NVidia) "Mip mapped normal maps"
  http://developer.download.nvidia.com/whitepapers/2006/Mipmapping_Normal_Maps.pdf
- Mittring, M. "Triangle mesh tangent space calculation"
  http://www.crytek.com/technology/presentations
- ATI 3Dc Whitepaper
  http://ati.amd.com/products/radeonx800/3DcWhitePaper.pdf