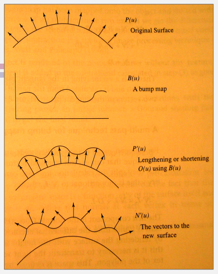
Implementing Bump Mapping



- to compute the perturbed normal N' for OC point (x,y,z)
  - map OC (x,y,z) to TC (u,v)
  - scale TC (u,v) to BC (u',v') – i.e. apply texture transform
  - compute B_u and B_v for (u',v')
    - convolve the bump map with kernels finding the gradient in the u and v directions (left to right, down to up)
  - compute tangent $N \times P_t$ and binormal $N \times P_s$ if needed
    - for a surface defined by parametric equations, can compute P_s, P_t directly (partial derivatives with respected to s and t)
    - for poly meshes, if both normal and tangent are provided, compute binormal as normal $\times$ tangent
  - compute N' = N + B_u `tangent` - B_v `binormal`
- use N' instead of regular surface normal for illumination and other lighting-related calculations