Computer Graphics Math

Sandro Spina
Computer Graphics and Simulation Group

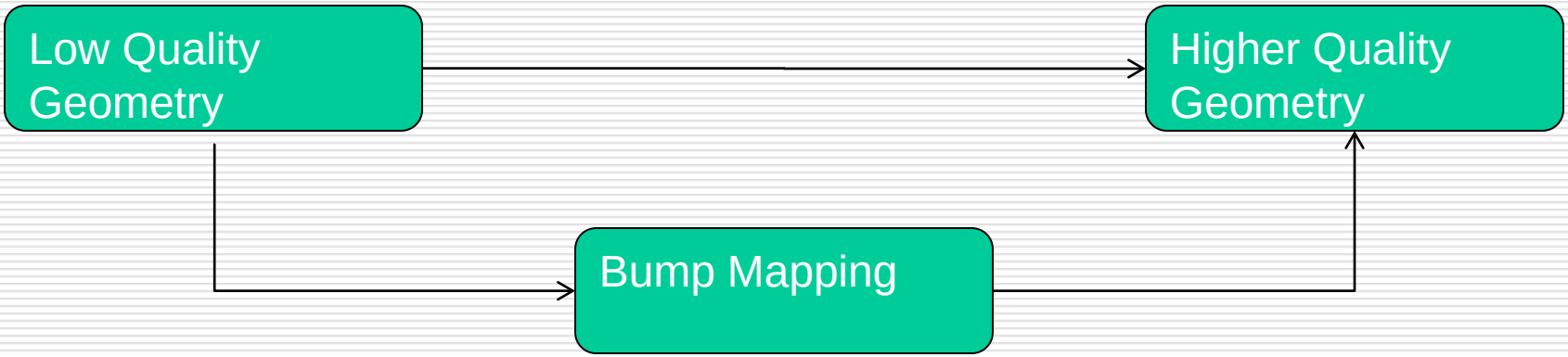
Computer Science Department
University of Malta

Texture Mapping ...

- So far we have seen how we can use an image texture in order to improve the appearance of our virtual objects.
- However, the rendered objects still do not seem to be realistic especially if the surface we are rendering is in reality not smooth.
- Texture mapping is like applying a wallpaper to a flat surface.
- In this module we shall see how we can trick the viewer into believing that the geometry is actually not smooth. Module content based on Mueller's Real-Time Rendering.

Bump Mapping ...

- In order to obtain the best possible 3D representation one can always increase the number of vertices used to sample the original object.
- However one can try and come-up with a trade-off between size of geometry and performance.
- Bump mapping offers one such trade-off.



Bump Mapping ...

- The term bump mapping describes a class of small-scale detail representation techniques.
- These methods are typically implemented by modifying the per-pixel shading routine (in the pixel shader)
- Their aim is to give a more three-dimensional appearance than texture mapping alone.
- Object Details:
  - Macro-features - cover many pixels
  - Meso-features – covers a few pixels
  - Micro-features – potentially smaller than a pixel

Macro, Micro and Meso Geometries

- Macro-geometries are represented by vertices and triangles, or other geometric primitives.
  - For eg. When modelling a virtual character (person) the limbs, head, etc are modelled at a macroscale.
- We talk about micro-geometries when we describe microscopic aspects of a geometry.
  - For eg. Shiny objects are microscopically smooth whereas diffuse objects are microscopically rough. The difference when rendering is achieved through the use of different shaders which simulate the interaction with the microscopic surface.
- Meso-geometry describes everything between these two scales.

Meso Geometries (some detail)

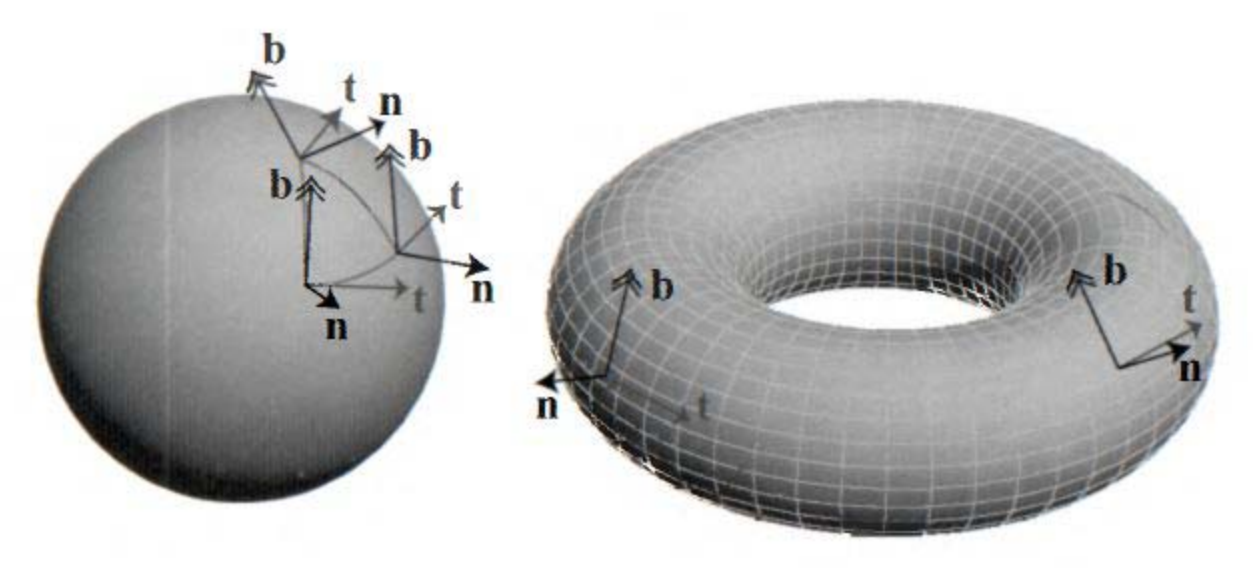
- Meso geometries include: Surface details that are too complex to be rendered efficiently using different polygons (for eg triangles) .... And ...
- That is large enough for the viewer to distinguish individual changes in surface curvature over a few pixels.
- Examples include:
  - A pebble floor
  - Wrinkles in a face
  - The brick wall (mentioned ealier)
- Bump mapping techniques are used (mainly) with meso-structures (geometries)

Bump Mapping Techniques (common idea)

- We'll see there are a number of common (widely used in games) bump mapping techniques.
- They vary mainly on:
  - Levels of realism
  - Complexity of detail features and how this is stored.
- These techniques are all based on one 'simple' idea …
- Instead of using the texture to change the colour of the surface being rendered, the texture (or bump map) is used to change the surface normal used in the lighting equation.
- Thus the perception of the surface changes (improves and becomes more realistic) while the surface geometry itself actually remains flat !!

Tangent Space Basis

- Changing the normal per pixel changes the perception of the polygon surface itself …
- The normal however must change with respect to some fixed frame of reference.
- A tangent space basis (recall what a basis is??) made up of the tangent, normal and binormal (or bitangent) is used in some bump mapping techniques.



Tangent Space Basis Matrix

- Before performing any calculations we need to make sure that the light and normal vectors are defined in the same coordinate space, i.e. The TBN space.
- The normal map already has it's normals defined in this space … So we need to transform the light direction vector from world space to TBN space, i.e. We need the light to be relative to the map.
- We do this by multiplying the light's direction (for a given vertex) by the following matrix:
- The three vectors, form a basis matrix

$$\begin{pmatrix} t_x & t_y & t_z & 0 \\ b_x & b_y & b_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Blinn's Original Bump Mapping Methods (1st)

- In this method two values are stored within each texel, b_u and b_v.
- These values correspond to the amount to vary the normal along the **u** and **v** axis.
- **u** and **v** can be thought of as the tangent and bi-tangent vectors on the surface geometry.
- These two vectors are added to the normal in order to change its direction.
- Conceptually b_u and b_v describe the direction in which the surface should be facing at that point.
- This type of bump map texture is called an offset map.

Blinn's Original Bump Mapping Methods (2nd)

- This method uses a texture which encodes a height field.
- Essentially (in the monochrome image) white is the high area and black is the low one. The gray areas describe altitude in between.
- Introduced by Blinn in 1978 (>30 years ago)
- The height field is used to derive the **u** and **v** component used to change the direction of the normal.
- This is a really convenient way of encoding the perturbations of the surface normal.
- Difference between neighbouring columns = u
- Difference between neighbouring rows = v

Blinn's Original Bump Mapping Methods (diagram)

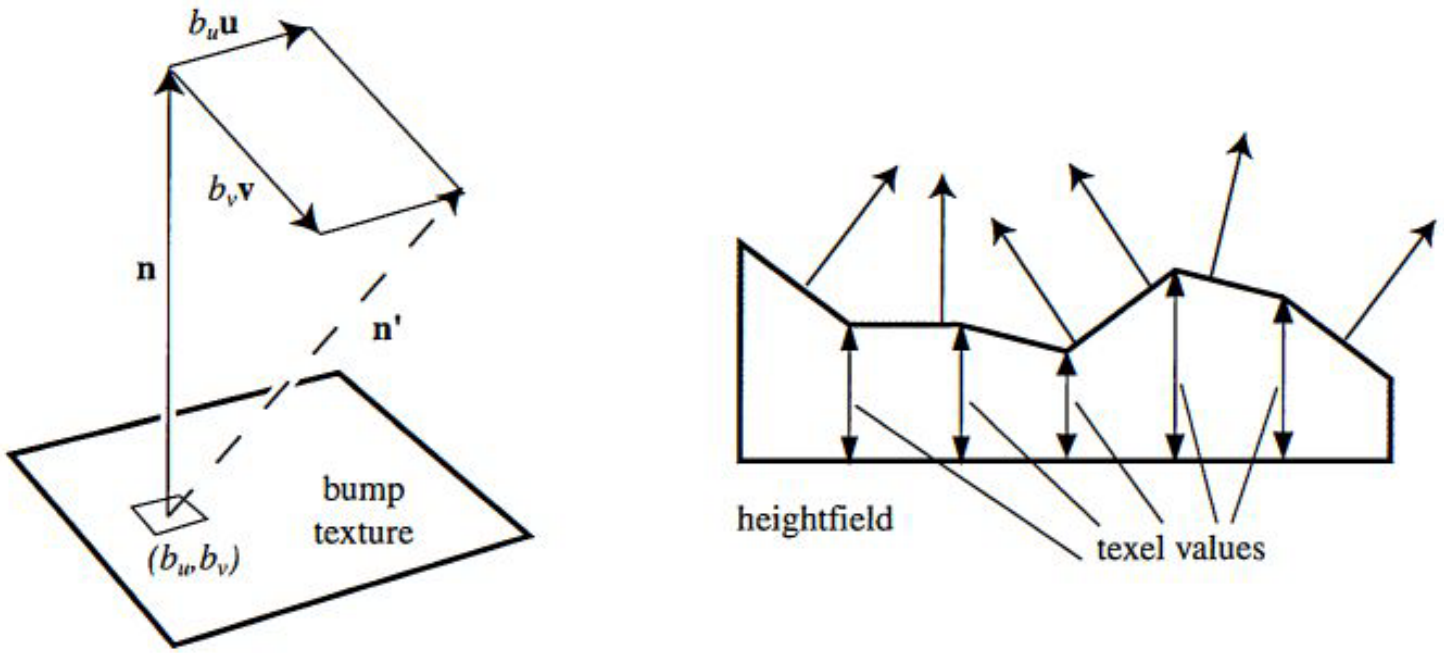


Figure 6.27. On the left, a normal vector **n** is modified in the **u** and **v** directions by the (b_u, b_v) values taken from the bump texture, giving $\mathbf{n}'$ (which is unnormalized). On the right, a heightfield and its effect on shading normals is shown. These normals could instead be interpolated between heights for a smoother look.

Normal Mapping

- So bump maps, as originated by Blinn, are texture maps that store a perturbation of the underlying geometric surface at each pixel.

- A normal map, originally described in Cohen et al., is an image that has the full surface normal stored at each pixel, not just a perturbation of the normal of the underlying geometric surface.

- IMP: Both bump and normal mapping exploit the fact that the light intensity of variations on a surface are more strongly affected by surface orientation than by distance variations for small surface features.

- Note also that normal mapping can produce the same results as bump mapping (and vice-versa). What's changing is simply the map (storage) format.

Normal Mapping (An Example in XNA)