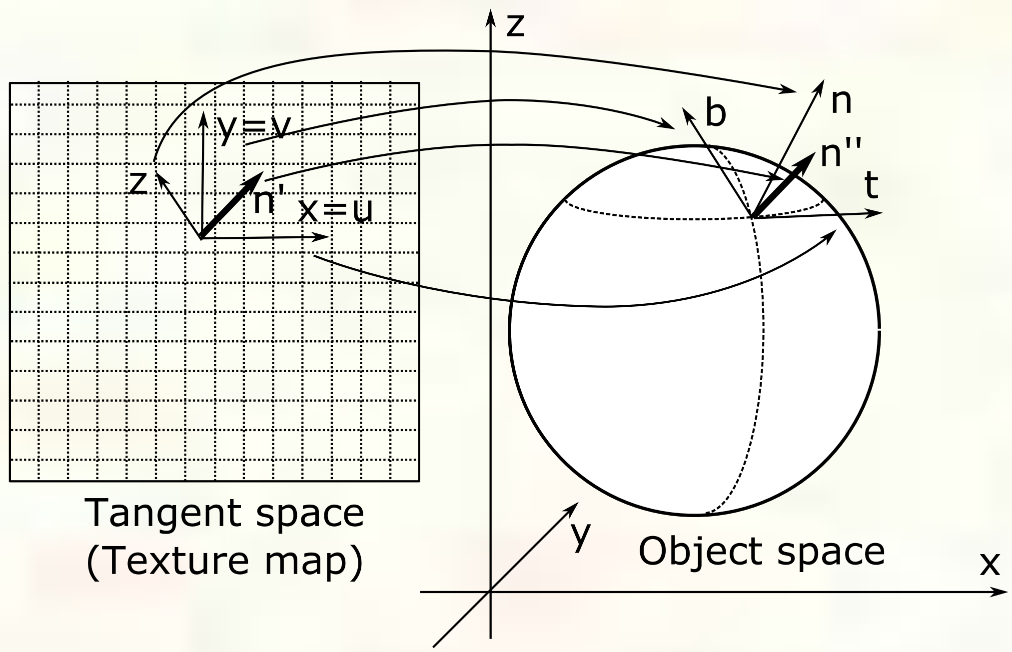
Normal Mapping and Tangent Spaces

Normal Maps

- Like bump maps but normals already computed
  - 3 elements per texel – x,y,z components of normal
  - Defined in texture (surface) space
  - Must map to object space to use
- Problem: What vectors in object space correspond to the x,y, and z axes in texture space?
  - Solution: Define two orthogonal vectors tangent to the surface and one normal to the surface
  - Surface space can now be called tangent space

Mapping normal to surface

- Step 1: Find texture coordinate of surface
- Step 2: Look up texel at that coordinate
- Step 3: Find rotation that maps tangent space normal to object space normal for the given pixel
- Step 4: Rotate tangent space normal defined in the texel by this rotation to define the normal at the surface point



Axes in object space

- For sphere in polar coordinates

$$x = r\sin\theta\cos\varphi \quad u = \frac{\varphi}{2\pi} \quad x = -r\sin\pi v\cos 2\pi u$$

$$y = r\sin\theta\sin\varphi \qquad\qquad y = -r\sin\pi v\sin 2\pi u$$

$$z = r\cos\theta \quad v = \frac{\pi-\theta}{\pi} \quad z = -r\cos\pi v$$

$$\mathbf{t} = \begin{bmatrix} \frac{dx}{du} \\ \frac{dy}{du} \\ \frac{dz}{du} \end{bmatrix} = \begin{bmatrix} \frac{dx}{d\varphi} \\ \frac{dy}{d\varphi} \\ \frac{dz}{d\varphi} \end{bmatrix} = \begin{bmatrix} -r\sin\theta\sin\varphi \\ r\sin\theta\cos\varphi \\ 0 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} \frac{dx}{dv} \\ \frac{dy}{dv} \\ \frac{dz}{dv} \end{bmatrix} = \begin{bmatrix} -\frac{dx}{d\theta} \\ -\frac{dy}{d\theta} \\ -\frac{dz}{d\theta} \end{bmatrix} = \begin{bmatrix} -r\cos\theta\cos\varphi \\ -r\cos\theta\sin\varphi \\ r\sin\theta \end{bmatrix}$$

$$\mathbf{n} = \mathbf{t} \times \mathbf{b}$$

need to normalize **t**, **b**, and **n**

Rotation – tangent and object space

- Object space to tangent space
  - For light vectors

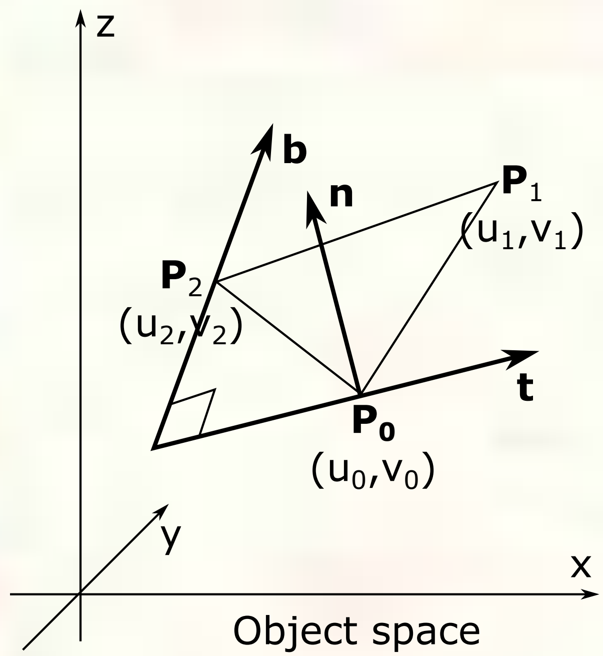
$$\mathbf{L'} = \begin{bmatrix} t_x & t_y & t_z & 0 \\ b_x & b_y & b_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{L}$$

- Tangent space to object space
  - For normals

$$\mathbf{n}'' = \begin{bmatrix} t_x & b_x & n_x & 0 \\ t_y & b_y & n_y & 0 \\ t_z & b_z & n_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{n}'$$

Messier for polygons (triangles)

- Given a triangle with vertices $\mathbf{P}_0,\mathbf{P}_1,\mathbf{P}_2$ and texture coordinates for those vertices
  - We need to solve for **t** and **b**
  - We already know how to get the triangle normal **n**



Solving for **t** and **b**

$$\mathbf{V}_1 = \mathbf{P}_1 - \mathbf{P}_0 = (u_1 - u_0)\mathbf{t} + (v_1 - v_0)\mathbf{b} = \Delta u_1 \mathbf{t} + \Delta v_1 \mathbf{b}$$

$$\mathbf{V}_2 = \mathbf{P}_2 - \mathbf{P}_0 = (u_2 - u_0)\mathbf{t} + (v_2 - v_0)\mathbf{b} = \Delta u_2 \mathbf{t} + \Delta v_2 \mathbf{b}$$

$$\begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix} = \begin{bmatrix} \Delta u_1 & \Delta v_1 \\ \Delta u_2 & \Delta v_2 \end{bmatrix} \begin{bmatrix} \mathbf{t} \\ \mathbf{b} \end{bmatrix}$$

$$\begin{bmatrix} \Delta u_1 & \Delta v_1 \\ \Delta u_2 & \Delta v_2 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{t} \\ \mathbf{b} \end{bmatrix}$$

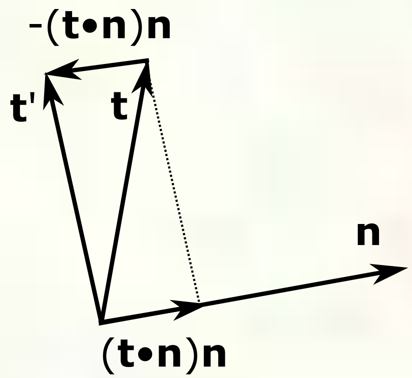
$$\frac{1}{\Delta u_1 \Delta v_2 - \Delta u_2 \Delta v_1} \begin{bmatrix} \Delta v_2 & -\Delta v_1 \\ -\Delta u_2 & \Delta u_1 \end{bmatrix} \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{t} \\ \mathbf{b} \end{bmatrix}$$

Why is it messy?

- For smooth shading you average the tangents and bitangents as well as normals of triangles sharing a vertex to get different **t**, **b** and **n** at each vertex.
- The you interpolate these across the triangle.
- At each pixel, the **t**, **b** and **n** vectors you get are no longer orthogonal.
- So you have to fix them, e.g. using Gram-Schmidt orthogonalization, before you can make a rotation matrix from them.

Gram-Schmidt Orthogonalization

- Start with unit vector **n**, make **t'** orthogonal to it



$$\mathbf{t}' = \mathbf{t} - (\mathbf{t} \cdot \mathbf{n})\mathbf{n}$$

- Normalize **t'**
- Now make **b** orthogonal to both **n** and **t'**

$$\mathbf{b}' = \mathbf{b} - (\mathbf{b} \cdot \mathbf{n})\mathbf{n} - (\mathbf{b} \cdot \mathbf{t}')\mathbf{t}'$$

- Normalize **b'**