# Notes on OpenGL

Asahina118

## Normal Mapping in Computer Graphics

A note referencing *https://learnopengl.com/Advanced-Lighting/Normal-Mapping* on the matrix formation. First, some background:
Let $V$ be a $\mathbb{F}-$Vector Space with basis $\{a_i\}, \{b_i\}$ where $\dim V = n$.
Let $v \in V : [v]_A = \sum_i \sigma_i a_i \iff v = [\sigma_1, ..., \sigma_n]_A^T$
Then the matrix for change of basis from $A \to B$ is

$$M = [[a_1]_B ... [a_n]_B]$$

Proof:
Note that $\forall i : [a_i]_B = \sum_j \omega_j b_j \implies \sigma[a_i]_B = \sigma \sum_j \omega_j b_j = \sum_j \sigma\omega_j b_j = [\sigma a_i]_B$

$$\begin{aligned}
v = [v]_A &= \sum_i \sigma_i a_i \\
\implies [v]_B &= \left[\sum_i \sigma_i a_i\right]_B \\
&= \sum_i \sigma_i \left[a_i\right]_B \\
&= \sum_i \sigma_i \left(\sum_j \omega_j^i b_j\right) \\
&= \sum_i \sum_j \left(\sigma_i \omega_j^i b_j\right) \\
&= \sum_j \sum_i \left(\sigma_i \omega_j^i\right) b_j \\
&= \left[\sum_i (\sigma_i \omega_1^i) \ ... \ \sum_i (\sigma_i \omega_n^i)\right]_B^T \\
&= \begin{bmatrix} \omega_1^1 & \omega_1^2 & ... & \omega_1^n \\ \vdots & \vdots & \ddots & \vdots \\ \omega_n^1 & \omega_n^2 & ... & \omega_n^n \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 \\ \vdots \\ \sigma_n \end{bmatrix} \\
&= [[a_1]_B \ ... \ [a_n]] \, \vec{\sigma} = [[a_1]_B \ ... \ [a_n]] \, [v]_A
\end{aligned}$$

Specifically, when $[v]_A = a_i$ for any $i$, $[a_i]_B = [\omega_1^i \ ... \ \omega_n^i]$. In other words, given $[a_i]_B$ which is the linear combination of basis of $B$ with coeff. $= \{\omega_j^i\}_j$, we can explicitly find the $i$-th column of $M$ (which is exactly the coefficients above. Sorry for being repetitive but it took me a whole 15 minutes to be able to see it from this perspective).

Then denote $\mathbb{T}, \mathbb{W} \subset \mathbb{R}^3$ be the texture and world space respectively, we have $\phi : \mathbb{T} \to \mathbb{W}$ where $\{aT, aB, aN\}$ is the basis of $\mathbb{T}$, so for any $w \in \mathbb{W}$,

$$w = (model) \cdot t$$

which is not that interesting, but when $t \in \{aT, aN, aB\} \implies t = e_i$, $(model) \cdot e_i \in \mathbb{W} \implies (model) \cdot e_i$ is a linear combinartion of basis of $\mathbb{W} \implies (model) \cdot e_i$ is (again) exactly the $i$-th column of the matrix of basis change by our above observation.

Thats why we are passing through the product of
$(model) * (aTangent/aBitangent/aNormal) := (T/N/B)$ into the fragment shader and form the $TNB\ matrix$ (matrix of change of basis from texture to world space) directly by attaching the product results together

$$TNB\ matrix = [T\ N\ B]$$

which we later can either (1) transform our normal map defined in texture space to the world space for shader calculation, or (2) do the inverse transformation of shader variables back to texture space to calculate our fancy lighting, using this $TNB\ matrix$ alone!

**Remark 1.** *notations followed from learnopengl.com.*

**Remark 2.** *I am typing out this note halfly for recalling the math for future reference, and halfly for fun. While this is not meant to be rigourous nor serious, feel free to correct me if I have written anything wrong.*