# CMPM 163 Notes

Malcolm Riley

Winter 2019 — April 9, 2019

## Vertex Displacement in Shaders

- There are two types of variables: **uniform** variables, which are the same for every vertex, and  variables, which are potentially unique per vertex.
- The `Properties` block can be used in Unity's ShaderLab syntax to define **uniform** variables.
- For clarity, it can be a good idea to denote variable fields in the actual code using the keyword `uniform`.
- In Unity, the `_Time` uniform is defined for you; it is a float array of various multiples of the current time.
- A **normal** is a unit vector perpendicular to the face of a corresponding polygon.
- The main takeaway is that the **Vertex Shader** is the program that operates per vertex.

## Phong Lighting Algorithm

- The most common shader - invented by a computer scientist from the University of Utah way back in 1975.
- There are different varieties of lighting model; the unique aspect of the Phong lighting model is that it operates per-pixel (per fragment) and that it incorporates specular highlights.
- "Matte" objects do not have specular highlights, but reflective objects do. Both types feature **Diffuse** and **Ambient** lighting.
- **Diffuse** lighting is, basically, surface lighting; **Ambient** light is a uniform light factor applied to the entire object.
- In Unity, point lights are only calculated during the "Forward Add" lighting pass.
- For multiple lights, changing the blending mode allows Unity to perform multiple passes.

- Unity provides the light color via the uniform `_LightColor0`. This vector is available in both the vertex and fragment shader.
- For the calculation of lighting, it is necessary to obtain the vertex position in world coordinates (since the vertex and lights both exist in world space, instead of local space).
- Unity exposes the `unity_ObjectToWorld` matrix for transforming local coordinates to world coordinates.
- Unity also exposes the worldspace camera coordinates as the uniform `_WorldSpaceCameraPos`.