Interactive Computer Graphics
Coursework – Task 4

December 13, 2020

Task 5: Colour

Colour space conversion is an important tool in any modern computer graphics applications. The RGB colour space has the disadvantages that it is device specific, it is not useful for human description of colour (e.g., we do not describe color as RGB percentages) and it is highly redundant and correlated (*e.g.*, all channels hold luminance information, which reduces coding efficiency). In Computer Graphics is often required to separate colour from intensity information. This is difficult using the RGB model but straight forward when using an alternative color space like HSV. This space separates hue, saturation, and value, which makes it easier to classify colours irrespective of, e.g., local illumination conditions or to generate an adaptive target colour according to a changing projection surface.

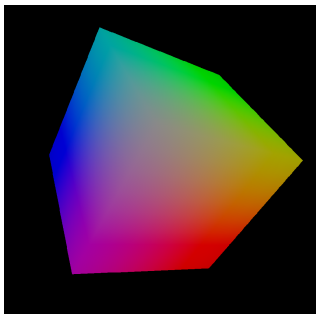

Figure 1: Example visualisation of the RGB colour space

Your task is to build a HSV to RGB converter as discussed during the lecture. You can do this in the *per-polygon* fragment shader and by defining a `uniform` variable as input HSV value. You can visualise the RGB colour space by using the currently rendered object's vertex position as output colour in the vertex shader (e.g. RGB colour space when rendering the cube model as shown in Figure 1). However, the cube for example is centred at the origin, hence you need to translate the position values with `vec4(0.5,0.5,0.5,0)` to get non-negative values.

Why is it difficult to visualise the HSV colour space using the cube model? Can you think of a different geometry that would be more suitable to sample the HSV space?

HAVE A LOT OF FUN!!