Lab 8: Color I

Intro
Color is an essential part of how we perceive our world, real and virtual. In this lab, we’ll explore the relationship between colors, our visual system, and digital representations of color.

Getting started
This lab will have two parts: a short answer + coloring part and a coding part. Before we start coding, let’s try to understand color concepts with a couple short answer questions (you can write your answers on the back of this page)

1. What is your favorite color? What is its complementary color?
2. How does human anatomy inform our understanding of color?
3. Name a color space that is perceptually uniform. Why is it perceptually uniform?

Once you’ve answered these, give your Stormtrooper some color! Color it in by hand! Don’t use your computer. You’ll see why in part two.
Lab 8: Color II

Intro
As you have learned in class, there are numerous ways of representing colors digitally. The most popular digital color representation today is RGB, since it’s the natural hardware representation.

Today, we’ll be trying to reproduce your colored drawing on the computer, by converting the colors of your Stormtrooper into two different color spaces, RGB and HSV.

Getting started
Copy the support code from /course/cs123/src/labs/lab08 into a personal directory and open ‘index.html’ in your favorite web browser. You should see a white background with an FPS counter in the top left and color controls on the right.

Now open up ‘index.html’ in your favorite text editor. We will be working entirely in the fragment shader today. The fragment shader begins with:

```html
<script id="fragmentShader" type="x-shader/x-fragment">

Take a moment to look over the fragment shader, particularly the uniforms.

RGB colors
To help us convert our drawing’s colors, we want to have our fragment shader display the color specified by the sliders in the dropdown menu. As you may have noticed, the sliders’ values are already tied to uniforms in our fragment shader (appropriately titled red, green, and blue).

Task 1:
In the main method of the fragment shader, set gl_FragColor to the color specified by the uniforms red, green, and blue. Refresh your page and try moving the RGB sliders – you should see the page’s background change accordingly.
**Task 2:**
We will now convert all your drawing’s colors into the RGB colorspace. For each color in your drawing, **write down** your best estimate of its RGB value using only the RGB sliders. This takes a while and is quite cumbersome! There must be a better way (hint, HSV colors!)

**Task 3:**
Now that you have the RGB values of your drawing’s colors, it’s time to color your digital drawing! Your uncolored drawing is located in the lab’s root directory. Open up ‘drawing.png’ in GIMP and using the RGB values you wrote down, color your drawing with the paintbucket tool.

Once you’re done, remember to **export (File -> Export)** your re-colored drawing as a PNG file!
HSV colors

As you may have noticed, the RGB color space can be confusing and hard to use. Although the RGB color space is based on the three types of light receptors in the human visual system (trichromaticity), the RGB color space is not exactly intuitive.

To convert our drawing's colors more easily and with higher accuracy we will be using another common color space, HSV. HSV is commonly represented as a color wheel, with complementary colors appearing on opposite sides of the wheel.

Drawing a circle

To visualize our HSV color picker, we need to draw the color wheel. We have provided a method stub called `drawColorWheel` that draws a blue circle (for now). This function takes the pixel location, `uv`, and returns the color of the color wheel at that location. The opacity of the returned color depends on whether `uv` is within the color wheel.

**Task 4:**
In the main method of the fragment shader call `drawColorWheel` and store the color returned. Use this color's opacity (4th channel) to mix (see this OpenGL doc) the background and the color wheel.

You should now see a blue circle with a target at its center surrounded by a background color controlled by the sliders in the drop-down menu.
Coloring the color wheel

Now we need to make our wheel colorful! HSV is commonly represented as a color wheel, with the first component, hue, representing the angle (theta), and the second component, saturation, representing the magnitude (r) of the polar coordinates of the color wheel. The third component, value, is already tied to a uniform in our fragment shader, titled hsvValue.

We have provided methods for converting between polar and pixel coordinates, uv2polar and polar2uv. Additionally, we have provided a method for converting HSV to RGB called hsv2rgb.

**Task 5:**
In the drawColorWheel method of the fragment shader, calculate the polar coordinates of the pixel. Scale these polar coordinates to [0 1] to calculate the HSV values of the pixel (hint: the last channel is given by the uniform hsvValue).

Since GLSL uses RGB, convert this HSV value using hsv2rgb() and set the color of the pixel in question.

You should now see a color wheel!
Positioning the target

Now we have a color wheel but we want to place the target on the color wheel where it should be (positioned by the sliders in the drop-down). This means we need to convert the RGB values to HSV and use those HSV values in turn to calculate the polar coordinates and uv coordinates of the target.

Task 6:
In the drawColorWheel method of the fragment shader, convert the RGB values in the uniforms red, green, and blue into HSV. Scale the HSV values to calculate the polar coordinates of the target and convert these polar coordinates into uv-coordinates. Use these uv-coordinates to draw your target.

Task 7:
We will now convert all your Stormtrooper’s colors into the HSV colorspace. For each color in your drawing, write down your best estimate of its HSV value using only the HSV sliders.

Task 8:
Now that you have the HSV values of your Stormtrooper’s colors, it’s time to color your digital Stormtrooper! Your uncolored Stormtrooper is located in the lab’s root directory. Open up ‘drawing.png’ in GIMP and using the HSV values you wrote down, color your Stormtrooper with the paintbucket tool.

Once you’re done, remember to export your re-colored Stormtrooper! Open both the RGB and HSV versions of your Stormtrooper in addition to your physically colored copy so a TA can check you off. The TA will also check your color wheel and its controls.