**Final Project Proposal**

**Group Member:** Wei Wu (wuwei33)

As a digital artist, I always do my painting on the computer using software such as Photoshop or SAI. Being inspired by traditional art, we have a concept of “hiding colors” (藏色) in concept art: Adding colors to the painting while maintaining the values unchanged. To check if we “hide colors” successfully, the processed painting should have more color variety but the same value as the original one.

This technique promises that the value isn’t affected by adding colors. It’s significant since a correct value promises viewers’ understanding of the light source direction, shapes of the object, textures of the surfaces, etc. Artists like me usually do this process manually by picking a color on the patch where we want to add variety on and rolling the color wheel 80~200 degrees to find a color with similar luminosity but in a different hue. Then, we adjust the saturation/luminosity a little bit and try to add it to our painting. If it’s not suitable we then try again – This process is interesting but time-consuming. Also, it’s not easy for every artist to see what “value” each color has. Human eyes are more sensitive to colors such as green and less sensitive to blue, thus different hues have different values though in the same luminosity. When I was a beginner, I didn’t realize this and always mess up the values of my painting when I am adding colors.

Therefore, I want to make a tool that helps digital artists do the “hiding colors” process. I will mainly use python to write my program and OpenCV as the helper library. I will use mainly knowledge from the start of the class. Here are my rough ideas on how to do it:

1. Accept an input painting and extract patches that are relatively dark.
2. For these patches, record their colors, and change the hue to either: ~80 degrees away on the color wheel, or close to its complementary color (~180 degrees). Then, we calculate the luminosity/saturation changes we need to shift to make our new color the same value as the original color. Increasing saturation and decreasing luminosity both make the final value darker and vice versa. We’ll take some ratio of these adjustments combined. (If the original color’s saturation exceeds a threshold, we prefer adjusting values by making saturation lower, and vice versa). This prevents our painting to be too saturated and looking uncomfortable.
3. After we decided on the colors to add (respectively for each patch), we randomly pick some pixels in each patch (50%: pick a side and pixels closer to that side, 30%: close to the center of that patch, 20%: randomly over the whole patch). Modify the pixels so that they are in the exact same color as that we’ve picked.
4. For testing: Compare the greyscale of our new painting to that of the original painting to see if they are the same.
5. Output our colorful painting! Another choice is to add(modify) colors on an empty canvas in step 3 and output that PNG. This allows the artist to download the canvas and cover it directly to their painting using software, with the transparency they like.
6. No datasets will be used for this project since we are not training any ML things.