Red Filter

Translate red filter algorithm to code

Now that you have implemented the grayscale image filter, it is time to try an unfamiliar image filter algorithm: a red hue filter. Below is one possible algorithm for the red filter.

For each pixel in the image:

- **Calculate the average of the pixel's RGB values**
- **If the average is less than 128:** set the pixel's red value to two times the average, set the pixel's green value to zero, and set the pixel's blue value to zero.
- **Otherwise:** set the pixel's red value to 255, set the pixel's green value to two times the average minus 255, and set the pixel's blue value to two times the average minus 255.

Using the DukeLearnToProgram JavaScript environment, translate the above red filter algorithm into JavaScript. Test with different images to make sure that your code is working correctly. What types of images would be good to test the code with?

If you would like to understand the logic behind this algorithm, see this reading explaining the red filter:

Red filter algorithm.pdf

Add red filter to web page

Once you have implemented the red filter in DLTP, add it to your web page. We suggest using the same approach as with the grayscale filter: have one function that actually modifies the image (in the grayscale example, this function was makeGray), then have another function (in the grayscale example, this function was doGray) that checks that the image is loaded, calls the function that modifies the image, and displays the filter on the canvas.

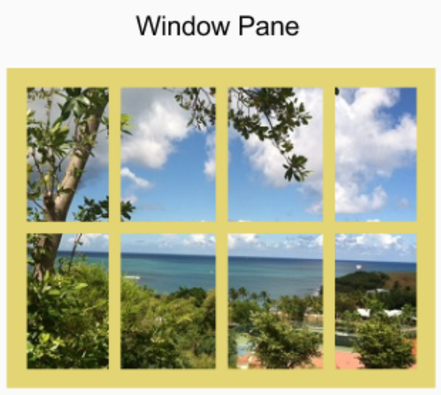
Try your red filter button to see if it works! Try pressing your red filter button several times. What happens to the image? What happens if you now click the grayscale filter button, then the red filter button?

This is a great time to check whether your reset button works properly and resets all the filter images to the original image, so that the next time you run the filters they run on the original image, not on the already filtered image.

Design your own filter

As the final part of your MiniProject, design and program your own image filter. It can do whatever you want, but it should be something original, not one of the filters you have seen in the class. Here are a couple of examples.

Window Pane



Trigonometry Shark

In these examples, pixels are manipulated based on where they are in the image, as well as their RGB values. You can can also include more complex math operations with JavaScript's Math object methods: http://www.w3schools.com/js/js_math.asp.

Be creative!

Once you have decided what you want your filter to do, you should follow the seven step process:

1. Work some small examples (i.e., a few to several pixels) by hand.
2. Write down the steps you took. Be very specific—this will help in translating to code later.
3. Look for patterns and generalize your solution.
4. Check your algorithm by hand on a few small examples to make sure you have not missed anything.
5. Once you are satisfied that your algorithm does what you want it to, translate it to code. You should develop your code in the Duke environment on the course site: http://www.dukelearntoprogram.com/course1/example/index.php (linked in the Resources tab as well). Later you will put this code on your web page.
6. Test your code. Think of what types of images would be good to try your filter on.
7. Debug any problems in your code.

Once your code is working in the Duke environment, add it to your web page as you added the grayscale and red filters and test it by uploading a few different images to your web page.

Personalize and share

The appearance of your web page is up to you! You should think about how you want to structure and style your web page using the HTML and CSS you have learned.

Once you have a version you would like to share, post a link to your pen in the forum "Share Your Success." Check out some of the other learners' projects, and give them your feedback!

Need help? Review the documentation at http://www.dukelearntoprogram.com/course1/doc/ to remind yourself of the SimpleImage methods and common HTML and CSS for laying out and styling a web page.

Mark as completed