Fundamentals of Computer Science I: Media Computing (CS151.02 2007F)

Laboratory: Transforming RGB Colors

Summary: In this laboratory, you will experiment with a variety of techniques for transforming RGB colors and images built from RGB colors.

Contents:

Reference:

Preparation

In this laboratory, you will be creating a few images and manipulating others. We will also be working with some colors.

Create a new 4x3 image, call it `canvas`, show it, and zoom in to 16x resolution.

Open an existing image of your choice, call it `picture`, and show it. Please choose an image that is not too large (say, not much more than 250x250).

Load your list of favorite colors, `/home/`*`username`*`/Desktop/fave-colors.sct`. If you don't have that file, use `/home/rebelsky/glimmer/samples/fave-colors.sct`.

Exercises

Exercise 1: Lighter and Darker

a. Using `rgb->rgb-list` or `rgb->string`, remind yourself of the three components of `fave1`.

b. Determine what happens to the components when you apply `rgb.darker` to `fave1`. You'll need to apply `rgb.darker` and then find out the components of the new color.

c. Determine what happens when you apply `rgb.lighter` to `fave1`.

d. Draw three pixels in sequence on `canvas`. The first should be the lighter version of `fave1`. The second should be `fave1`. The third should be the darker version of `fave1`. Do you see a difference?

e. What do you expect to happen to the red, green, and blue components if you apply `rgb.lighter` three times to the color 127/20/20, as in the following?

```
(define newcolor (rgb.lighter (rgb.lighter (rgb.lighter (rgb.new 127 20 20)))))
```

f. Check your answer experimentally.

g. What do you expect to happen to the red, green, and blue components if you apply `rgb.darker` three times to the color 127/20/20, as in the following?

```
(define newcolor (rgb.darker (rgb.darker (rgb.darker (rgb.new 127 20 20)))))
```

h. Check your answer experimentally.

Exercise 2: Other Transformations

As you may recall from the reading, there are also color transformations that make more significant changes to colors. For example, `rgb.phaseshift` shifts each component by 128 (adding to small components and subtracting from large components). In contrast, `rgb.complement` computes the complement of a color.

Suppose we've defined the following colors:

```
(define c0 (rgb.new 64 128 196))
(define c1 (rgb.new 32 96 255))
(define c2 (rgb.new 240 0 127))
```

a. What do you expect the complements of `c0`, `c1`, and `c2` to be?

b. Check your answer experimentally.

c. What do you expect the phase shifts of `c0`, `c1`, and `c2` to be?

d. Check your answer experimentally.

Exercise 3: Transforming Pixels

a. Set pixels (0,0) and (1,1) of `canvas` to `fave1`.

b. Make the top-left pixel darker using the more verbose instruction from the reading.

```
(image.set-pixel! canvas 0 0 (rgb.darker (image.get-pixel canvas 0 0)))
```

c. Make the pixel at (1,1) darker using the the more concise `image.transform-pixel!` procedure.

```
(image.transform-pixel! canvas 1 1 rgb.darker)
```

d. Do you see any advantages of using the longer instruction?

Exercise 4: Transforming and Copying Pixels

Of course, rather than computing new pixels from the pixels in one image, we can also compute them from the pixels in another image. For example, the following code should put in `canvas` a darker version of the top-left square of some pixels from `picture`

```
(image.set-pixel! canvas 0 0 (rgb.darker (image.get-pixel picture 100 100)))
(image.set-pixel! canvas 0 1 (rgb.darker (image.get-pixel picture 100 101)))
(image.set-pixel! canvas 1 0 (rgb.darker (image.get-pixel picture 101 100)))
(image.set-pixel! canvas 1 1 (rgb.darker (image.get-pixel picture 101 101)))
```

Confirm experimentally that these instructions work as we suggested.

Exercise 5: Multiple Transformations

Of course, we can get more transformations by combining the basic transformations. For example, we get a different color when we complement and darken a color than when we complement or darken the color alone.

a. Does the order in which we apply transformations matter? In particular, do you get the same or different color when you complement and then darken a color as compared to when you darken and then complement the color? In code, what is the relationship between `newcolor1` and `newcolor2`?

```
(define newcolor1 (rgb.darken (rgb.complement fave1)))
(define newcolor2 (rgb.complement (rgb.darken fave1)))
```

b. Check your answer experimentally.

c. What do you expect to have happen if you complement a color twice, as in this example?

```
(define newcolor3 (rgb.complement (rgb.complement fave2)))
```

d. Check your answer experimentally.

e. At first glance, lightening and darkening an image seem to be complements. Are there ever times in which the sequence of lighten and then darken does not give you back the same color?

```
(define newcolor4 (rgb.darken (rgb.lighten fave3)))
```

f. Check your answer experimentally. In doing so, try colors near the extremes, such as black, white, yellow, 10/255/127, and such.

Exercise 6: Transforming Larger Sections

In the corresponding reading, there is a set of sample code that is intended to transform `canvas` by complementing every pixel.

```
(image.transform-pixel! canvas 0 0 rgb.complement)
(image.transform-pixel! canvas 0 1 rgb.complement)
(image.transform-pixel! canvas 0 2 rgb.complement)
(image.transform-pixel! canvas 0 3 rgb.complement)
(image.transform-pixel! canvas 1 0 rgb.complement)
(image.transform-pixel! canvas 1 1 rgb.complement)
(image.transform-pixel! canvas 1 2 rgb.complement)
(image.transform-pixel! canvas 1 3 rgb.complement)
(image.transform-pixel! canvas 2 0 rgb.complement)
(image.transform-pixel! canvas 2 1 rgb.complement)
(image.transform-pixel! canvas 2 2 rgb.complement)
(image.transform-pixel! canvas 2 3 rgb.complement)
```

a. There is a subtle error in the code. Identify the error and fix it. (If you can't figure out the error, try running the code to see what error messages you get.)

b. Update `canvas` so that it has a variety of colors. Here is one set of simple changes, but you can do what you want.

```
(image.set-pixel! canvas 0 0 (rgb.new 0 0 0))
(image.set-pixel! canvas 1 0 (rgb.new 255 0 0))
(image.set-pixel! canvas 2 0 (rgb.new 0 255 0))
(image.set-pixel! canvas 3 0 (rgb.new 0 0 255))
(image.set-pixel! canvas 0 1 (rgb.new 255 255 255))
(image.set-pixel! canvas 1 1 (rgb.new 255 0 255))
(image.set-pixel! canvas 2 1 (rgb.new 255 255 0))
(image.set-pixel! canvas 3 1 (rgb.new 0 255 255))
(image.set-pixel! canvas 0 2 (rgb.new 63 127 195))
(image.set-pixel! canvas 1 2 (rgb.new 127 195 63))
(image.set-pixel! canvas 2 2 (rgb.new 195 63 127))
```

c. Verify that the repaired instructions do, in fact, complement all of the pixels.

d. What do you expect to have happen if you run this code twice?

e. Check your answer to the previous question experimentally.

For Those With Extra Time

Extra 1: Phase Shifting vs. Complementing

At first glance, some find that `rgb.phaseshift` is a lot like `rgb.complement`. After all, each changes a color by shifting the components, and adding or subtracting 128 may feel like an easier way to get something that sums to 255. However, as we've suggested in the reading, the two operations are quite different.

a. Find two colors whose pseudo-complements are fairly close to their phase-shifted versions. You may find the following code useful as you visually compare the different colors.

```
(define color1 (rgb.new __ __ __))
(define color2 (rgb.new __ __ __))
(define ps1 (rgb.phaseshift color1))
(define ps2 (rgb.phaseshift color2))
(define comp1 (rgb.complement color1))
(define comp2 (rgb.complement color2))
(image.set-pixel! canvas 0 0 color1)
(image.set-pixel! canvas 1 0 ps1)
(image.set-pixel! canvas 2 0 comp1)
(image.set-pixel! canvas 0 2 color2)
(image.set-pixel! canvas 1 2 ps2)
(image.set-pixel! canvas 2 2 comp2)
```

b. Find two colors whose phase-shifted versions are much different than their pseudo-complements.

c. Do you expect there to be more colors like those in a, or more colors like those in b? (That is, is it more likely that the pseudo-complement of a color is close to the phase-shifted color, or that they are different?) Explain your answer.
