Ethan Chen & Kysen Osburn
October 1, 2020
CS10, Fall 2020, Pierson

Region Finder

This region finder implements the algorithm for findRegions() given by Professor Pierson in the problem set description, including the blank canvas to check for all points that have been visited. The code has also been broken down with a helper method called getNeighbors() which returns all of the possible neighbors for any given point. The class also includes instance variable "largestRegions" and a second recolorImage() method that takes parameters Color color, ArrayList<ArrayList<Point>> regions, and BufferedImage image, which paints the given regions onto image with the given color. We have also included an alternate algorithm to the one provided in the problem set description. Instead of screening neighbors before adding them to toVisit, we added all of the neighbors and then skipped them if they failed the colorMatching or visited tests. This algorithm also includes a second helper method addNeighbors() which adds the neighbors of a given point to the toVisit ArrayList.

The parameters we chose were
maxColorDiff = 30;
minRegion = 30;

What we found was that decreasing minRegion gave us new smaller regions with just a few pixels that were speckled all over the place. On the other hand, increasing maxColorDiff expanded on some smaller regions and joined some into the same region, making it look more dense, but also created smaller regions in different areas of the picture. For example, in the picture of Baker library, when we reduced maxColorDiff from the default of 50 to 20, new, small speckled regions appeared, but the rest remained relatively unchanged. When we increased minRegion from 20 to 50, the sizes of all of the regions changed, and many different parts of the grass got picked up as the same color as the brick. Setting each to 30 seemed to create the best combination such that the regions looked dense, but the finder did not pick up the grass.

CamPaint

CamPaint utilizes the Region Finder class on the user's webcam. When the user clicks on a point on the webcam, CamPaint looks at the color of that point and finds regions of the webcam that are close to that color. Once it finds those regions using the Region Finder class, it paints them a given color each frame. It uses the largestRegion method of the Region Finder class to identify the largest region on the webcam and allow that region to leave a trail of paint behind it. It shows the unedited webcam by default. Pressing the 'r' key shows the webcam with the paint enabled, pressing the 'w' key shows the webcam unedited, and pressing the 'p' key shows the paint that has been applied, but on a white background instead of the webcam.

The region growing algorithm is useful for differentiating between regions of the same color, therefore enabling us to find the largest region that will act as the paintbrush. Therefore, keeping track of each of the largest regions in each frame and painting them accordingly is fairly easy to do. However, because we have to run the region growing algorithm on every single frame of the webcam, it is likely very inefficient. A more efficient program would not check the entire picture for new regions, but just the area surrounding the largest one. Alternatively, after finding the largest region, if there were some way to track any additional new pixels in a 100 pixel boundary outside of the last frame, we could eliminate the need to run the region growing algorithm at every step.

Extensions

We have provided two extension methods: RegionFinderExtension and CamPaintExtension. The methods have allowed for the following actions: c to clear the canvas, t to toggle paint mode on and off, 1 to change the brush color to blue, 2 to change the brush color to red, 3 to change the brush color to green, 4 to change the brush color to yellow, and 0 to change the color to the color of the paintbrush (if orange was clicked, it will draw orange). CamPaintExtension has included the functionality within the key clicks, while RegionFinderExtension has implemented a new private class ColorRegion which stores a region (ArrayList of points) as well as its corresponding color. All of the methods and variables that previously used a region have been altered to use a ColoredRegion, such as the recolorImage method.

It also allows the paintbrush to be switched. In order to do this, paint mode must be off, but when it is and the painting stops, you can select a new object to paint with and proceed as usual.