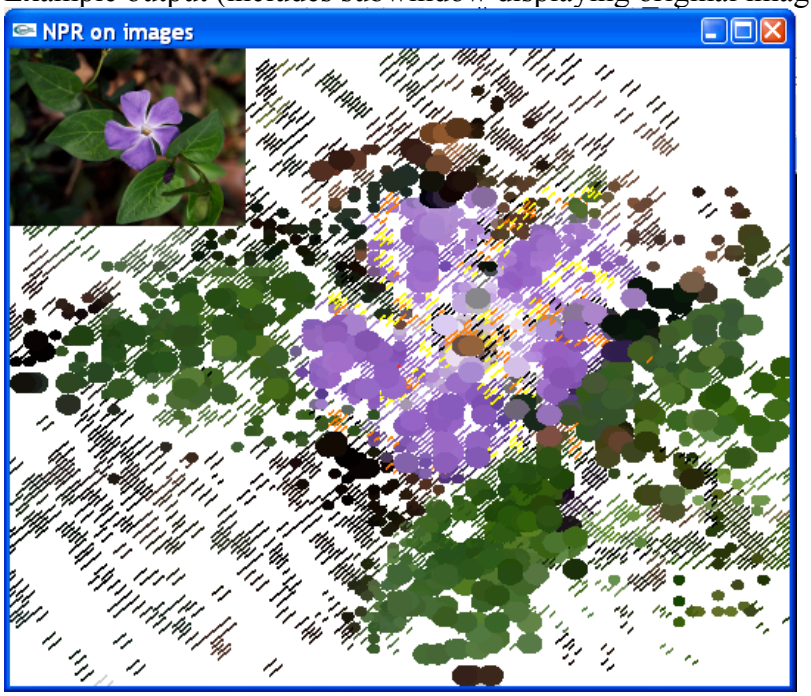
CSC 471: Program I

This program is due **Friday, January 13th** at 11:59pm. The late policy specified in the syllabus applies. As with all programming assignments this program should be done **individually**! You may talk to one another about the program, but you may not look at someone's working code!

Orientation:
You will be writing a simple NPR painterly image renderer using simple OpenGL primitives to generate a self-portrait (thus please select an input image that is representational of you in some way). Your program will allow the user to 'paint' over a starter image (read in from file), using the mouse. Wherever, the user paints, your program will generate 'strokes' of the same color on a new canvas (that starts out white). The strokes will be of various sizes and shapes as chosen by the user.

Example output (includes subwindow displaying original image):



Specifications:
Start with the base code provided to read in bmp images. This starter code includes a main glut window and a glut subwindow window which displays the image (read in from standard input, that means to run the program, you would for example, type: *a.out* < *flower.bmp*). You will write code to allow the user to sample the image and generate strokes in the main window. The user will be able to control the shape of the stroke, its size and its color as described below. As the user draws "strokes" over the original image, strokes represented by openGL primitives should appear in the appropriate place in the main display window. Strokes are drawn by depressing the left mouse button and moving the mouse over the original image. The user is allowed to draw multiple types of strokes (shape, size and color).

1) If the user clicks the right mouse button a menu appears. This menu should contain three different selection criteria: "shape", "size", and "color".

2) The stroke shape selection should include the following three stroke types: “point”, “line stroke”, and your choice (i.e. you must support line and circle strokes but have a choice for the third style and it must be reasonable, for example a triangle). When the user selects a stroke style all strokes drawn after the selection will be of that style. The user can have multiple different style strokes on the screen at a time. Be sure that a default stroke is selected at the launch of the application.
3) Size criteria should contain two sizes: small and large. The size criteria should affect all strokes drawn after the selection. Make the sizes vary with changing image sizes. A default stroke size should be selected at the launch of the application. Feel free to add a third size option based on the mouse stroke if you’d like.
4) The color criteria should default to “sample from image” – that is, when the user moves the mouse over the original image (with the left mouse button depressed) the application samples the color from the image and uses that for the current stroke color. Multiple strokes should be generated as the mouse is moved over the image (with the left mouse button depressed). In addition to sampling color from the image, allow the user to select at least three other stroke colors to provide creative expression. When sampling color from the image is selected, the strokes drawn in the main window should match the color under the mouse. Note that you will need to think about how to map coordinates from the subwindow appropriately. Please use **glReadPixels** to sample the color from the image.
5) Add a menu option to regularly sample the image (that is sample the image every 3-10th pixel) and put down background strokes (see in class demo).

Other then the straight forward openGL and glut code functionality you need to add, you need to think about a good data structure to support the user’s use of the application and about coordinate transforms. You will also need to make sure that when your strokes are rendered they are rendered using a Vertex Buffer Object. Be sure that stoke position and color are correct! Note that you will need to use the `glutSetWindow(int)` to toggle between the two glut windows (the main and subwindow) depending upon where you want functions to act.

Point break-down:

40 correct stroke matching (color and position)
30 stroke functionality (shapes, color, size) working as expected
15 correct interactivity (menus and drawing)
15 general (code style, execution, creativity, self expression)

Submit a README with your program which describes any necessary user instructions and design choices you made and why. Please document which features of the program work or do not work in your README file. A completely functional program is worth 100 points total. Points are awarded for a combination of functioning code and decent code design – i.e. I will look at your code design for this application. We will discuss this assignment in lecture on Tuesday. In addition to handing in your code, you are **required** to submit your favorite screen shot of the best “painterly renderings” that you generate with your program. You may submit up two images all which must be named: yourlogin1.jpg and yourlogin2.jpg (where “yourlogin” is replaced by your actual login name).