CS535: Assignment #2 – Splat and Squash!

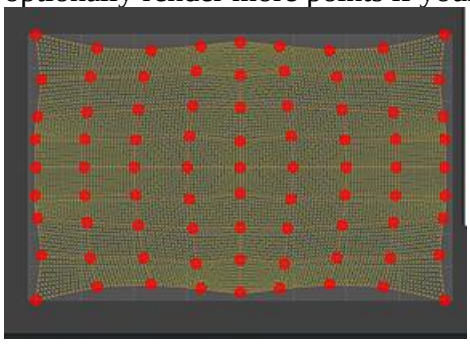
Out: September 17, 2024
Back/Due: October 4, 2024

Objective:
This objective of this assignment is a play with point-based rendering and image transformations. Inpsired by https://repo-sam.inria.fr/fungraph/differentiable-multi-view you can see that good quality imagery is possible. Within the time frame of this assignment, we will make a considerable simplification. Further, to make it fun the assignment is in the context of a game. You will render a set of points resembling a sheet. Each point corresponds to a sample from a given image (or texture). Using the mouse, you can then "deform" the points/sheet with a mouse press. The deformation will either be dynamic (e.g., occur over a small number of frames) or static (e.g., it will remain deformed). You can then press other points in the sheet. Your program should support at least 2 input images, 2 splat sizes, and at least one deformation technique (with one being a simple push out by a decaying inverse distance function).

You will use OpenGL with FreeGLUT (or similar). This program is being assigned as a CPU centered program (e.g. you can compute the vertex displacements using the CPU). If you wish to venture into GPU programming, that is permitted so long as the same functionality occurs. The next assignment will be GPU programming and thus there you will definitely see the contrast.

Specifics:

(0) (25%) Points: you should define and render a grid of least 64x64 points, most likely in the Z=0 plane and centered at the origin. The color of each point shall be extracted from the image. For example, if the image is 64x64, then each point color is essentially a pixel color. If the image is 640x640 then every block of 10x10 pixels should be averaged to form one point color. You can optionally render more points if your computer can handle it.

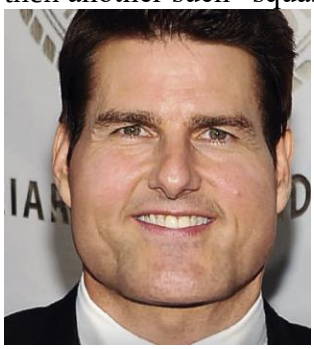


(this is a 9x9 grid example)

(1) (15%+15%) Splatting: you should support two splatting mechanisms. One method should be a fixed splat size (i.e., render each point as an opaque disc or box of diameter D). Then, you should support a second splatting mechanism. For example, render each disc as a Gaussian; alternatively, have the splat size change with distance from camera to the sheet.

(2) (25%) Squashing: you should enable the user clicking on a point P on this image. Then centered at the clicked point P, you should "push" the points p_i away from P using $d_{ij} = \alpha/\|p_i - P\|^2$; in other words, by the inverse squared distance between clicked point P and point p_i. The constant α is a user defined scaling factor. Lets say the sheet points occupy the space $[-32, +32]$. Thus the distance between P and a neighboring p_i might be 1. If $\alpha = 0.1$ then p_i

will be moved along the vector $(p_i - P)$ by 0.1 further away from P. If another point is clicked, then another such "squash displacement" will occur. An example and nice result is:



(This is possible)

(Can you get this result? Not really -- unless you plan very carefully)

(3) (20%) GUI: your program GUI should enable choosing "imageA.png" or "imageB.png". Specifying a file name is fine (and preferred) but not required. The GUI should also enable choosing the splatting technique. The mouse should be used to specify the deformation center point. Use the 3D camera control of A0. However, the sheet should always appear reasonable large/centered in the window. You could specify some camera controls via mouse or keyboard or menu usage.

(4) EXTRA CREDIT (10%)
   a. Deformations: make additional deformations possible!
   b. Enable saving the output image to disk, and then loading it back in for further deformations.

Turn-in:
To give in the assignment, please use Brightspace. Give in a zip file with your complete project (project files, source code, and precompiled executable, but no debug output). The assignment is due BEFORE class on the due date. It is your responsibility to make sure the assignment is delivered/dated before it is due.
Don't wait until the last moment to hand in the assignment! For grading, the program will be run with no command line parameters and the code will be inspected. If the program does not compile, zero points will be given.

If you have more questions, please ask!