Program #6
March 14, 2018

CS5720: Data Visualization
R. M. Parry

---

1. Using the MRI dataset, write a program that produces an isosurface using the basic marching cubes algorithm:
   `https://en.wikipedia.org/wiki/Marching_cubes`
2. You can download the data here:
   `http://cs.appstate.edu/~rmp/cs5720/mri.zip`
3. To help with debugging, I created 256 image cubes containing each $2 \times 2 \times 2$ case available here:
   `http://cs.appstate.edu/~rmp/cs5720/cubes.zip`
4. The ZIP files contain PGM files, one for each slice. The PGM images contain a simple text format that you can read in any language:
   `https://en.wikipedia.org/wiki/Netpbm_format#PGM_example`
5. To determine which of the 15 cases to use for each $2 \times 2 \times 2$ voxel neighborhood, you can rotate it 24 different ways:
   `http://www.euclideanspace.com/maths/discrete/groups/categorise/finite/cube/index.htm`

   Start with vertices for a $2 \times 2 \times 2$ cube centered at the origin. Compute it's numerical representation. Check if it matches any of the 15 base cases. If not, try a rotation, compute the new numerical representation and check again. Try all of the 24 rotations until you find a match. If you don't find a match, something is wrong.

   Once you find a match to one of the 15 base cases, you can define the triangles and normals to draw in the centered cube, then invert the rotation using its transpose. Finally translate and scale the points to where they belong in the 3D image.
6. The framework code provides a function to produce a rotation matrix around an arbitrary axis `rot(a, x, y, z)` and a list of the rotations to apply, taken from here:
   `http://www.euclideanspace.com/maths/geometry/rotations/axisAngle/examples/index.htm`
   Framework is here:
   `http://cs.appstate.edu/~rmp/cs5720/marching_cubes_framework.py`
7. As before, provide a bash script that executes it with the path to the CSV file provided on the command line. Usage should be like this:

   ```
   $ bash fig_mri_contour.sh /home/user/mri/14.pgm 40 fig_mri_contour.png
   ```

   The bash script runs your program using the provided file. For example, if you write a Python program the bash script might look like this:

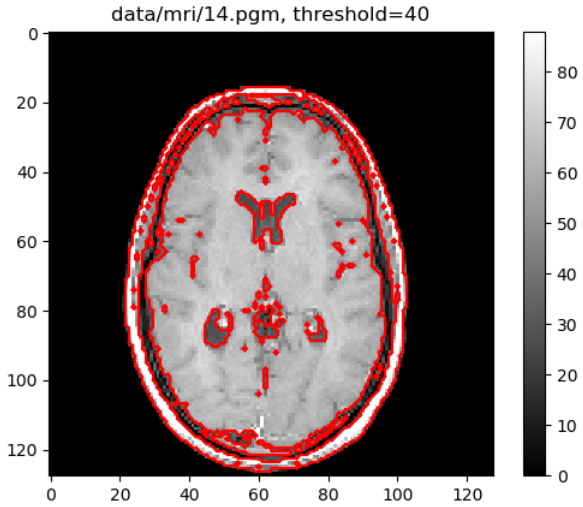
   ```
   python fig_mri_contour.py "$@"
   ```

   If you decide to use R, it might look like this:

   ```
   Rscript fig_mri_contour.R "$@"
   ```
8. The program should produce a PNG with the name of the second argument: `fig_mri_contour.png` in the example above.

   Zip your program and submit it to asulearn.

   It should look a lot like this:


data/mri/14.pgm, threshold=40
0
20
40
60
80
100
120
0
20
40
60
80
100
120
80
70
60
50
40
30
20
10
0