Vedant Puri
veantpu@andrew.cmu.edu

Project 1

ME 24-658
December 16, 2022

Problem 1

In problem 1, we compute the volume of dog brain from a three-dimensional MRI image of the brain. The image is segmented using the program 'Volume Rover'. The distribution of seed points for segmentation is presented in Figure 1 where the cyan colored points correspond to the brain, our area of interest, and the red points correspond to the region we need to remove. The resultant segmented image is presented in Figure 2. The segmented image has the same dimension as the initial image, though the region outside the brain has been zeroed out.

To compute the volume, we first read the code in using the code in § . Then we proceed to compute the volume of a voxel in the image and multiply that with the number of voxels corresponding to the brain, ie the voxels with a nonzero value. From the output in § , we see that the image is of size $219.1386 \times 219.1386 \times 28.0014\,\mathrm{mm}^3$ in x, y, z directions respectively with 256 points in x and y directions and 15 points in the z direction. The volume of a voxel is the product of grid spacing in each direction, and comes out to be $1.4771\,\mathrm{mm}^3$. The number of voxels corresponding to the brain are $39,144$. The volume of the brain is therefore $57820\,\mathrm{mm}^3$.

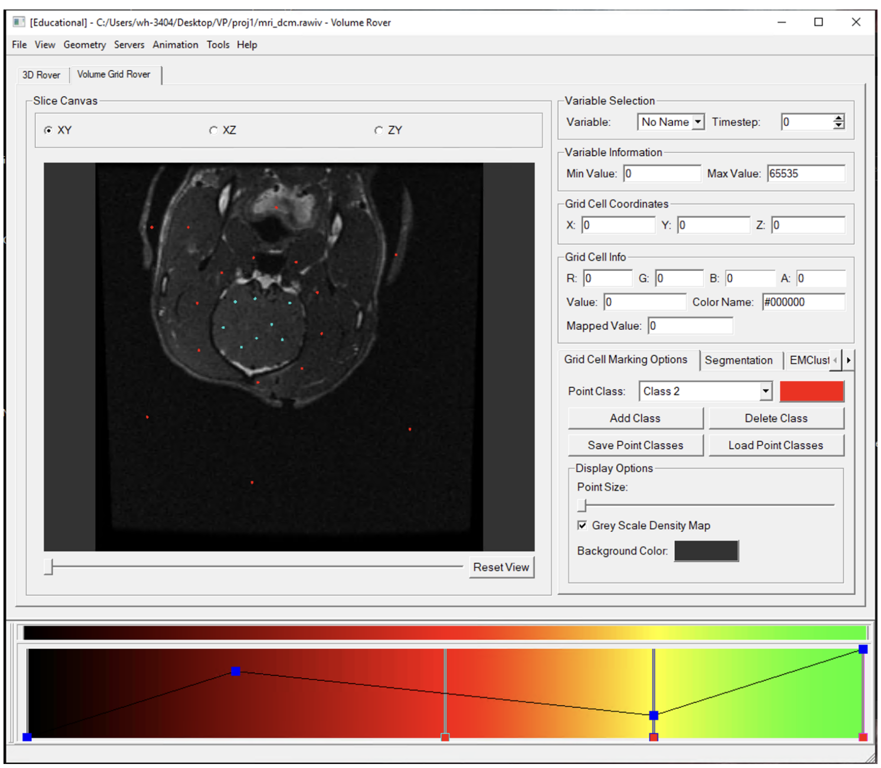

Figure 1: Seed points for segmenting dog brain

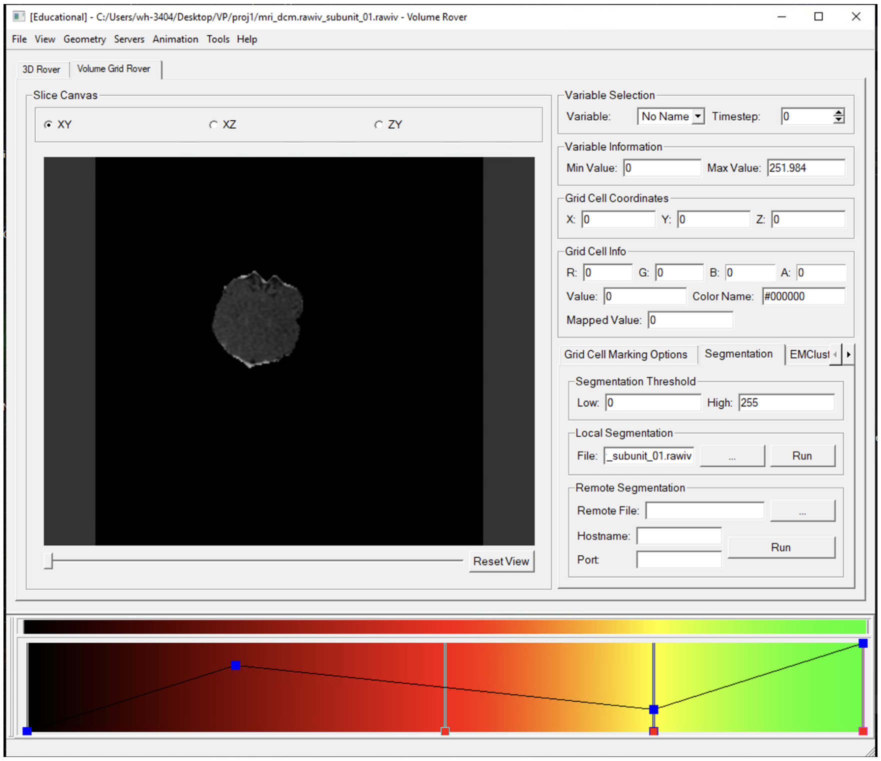

Figure 2: Segmented MRI of dog brain

Code

`readRawiv.m`

```matlab
function rawiv = readRawiv(rawivName)
%   Read rawiv data format into Matlab and saveas a raw file
%   http://ccvweb.csres.utexas.edu/docs/data-formats/rawiv.html
%
%   Useage:
%   rawiv = readRawiv(rawivName)
%
%   Example
%   rawiv = readRawiv('head.rawiv');
%
%   Author:     Sheng Yue
%   Email:      sheng.yue.84@gmail.com
%   Created:    18 March 2011
%   Version:    1.0
fid=fopen(rawivName,'r');
rawiv.minXYZ        =   fread(fid,3,'float','b')';
rawiv.maxXYZ        =   fread(fid,3,'float','b')';
rawiv.numVerts      =   fread(fid,1,'uint32','b')';
rawiv.numCells      =   fread(fid,1,'uint32','b')';
rawiv.dimXYZ        =   fread(fid,3,'uint32','b')';
rawiv.originXYZ     =   fread(fid,3,'float','b')';
rawiv.spanXYZ       =   fread(fid,3,'float','b')';
rawiv.image         =   fread(fid,prod(rawiv.dimXYZ),'*float','b');
rawiv.image         =   reshape(rawiv.image,rawiv.dimXYZ);
fclose(fid);

outName = [rawivName '_' num2str(rawiv.dimXYZ(1)) '_'...
                         num2str(rawiv.dimXYZ(2)) '_'...
                         num2str(rawiv.dimXYZ(3)) '.raw'];
fid=fopen(outName,'wb');
fwrite(fid, rawiv.image, 'float');
fclose(fid);

end
```

`compute_volume.m`

```
%
filename = 'mri_dcm.rawiv_subunit_01.rawiv';

data = readRawiv(filename)
img  = data.image;

vol_voxel = prod(data.spanXYZ);
num_voxel = sum(img ~= 0, 'all');
vol = vol_voxel * num_voxel
```

Output

```
>> compute_volume

data =

  struct with fields:

        minXYZ: [0 0 0]
        maxXYZ: [219.1386 219.1406 28.0014]
      numVerts: 983040
      numCells: 910350
        dimXYZ: [256 256 15]
     originXYZ: [0 0 0]
       spanXYZ: [0.8594 0.8594 2.0001]
         image: [256x256x15 single]


vol =

   5.7820e+04
```

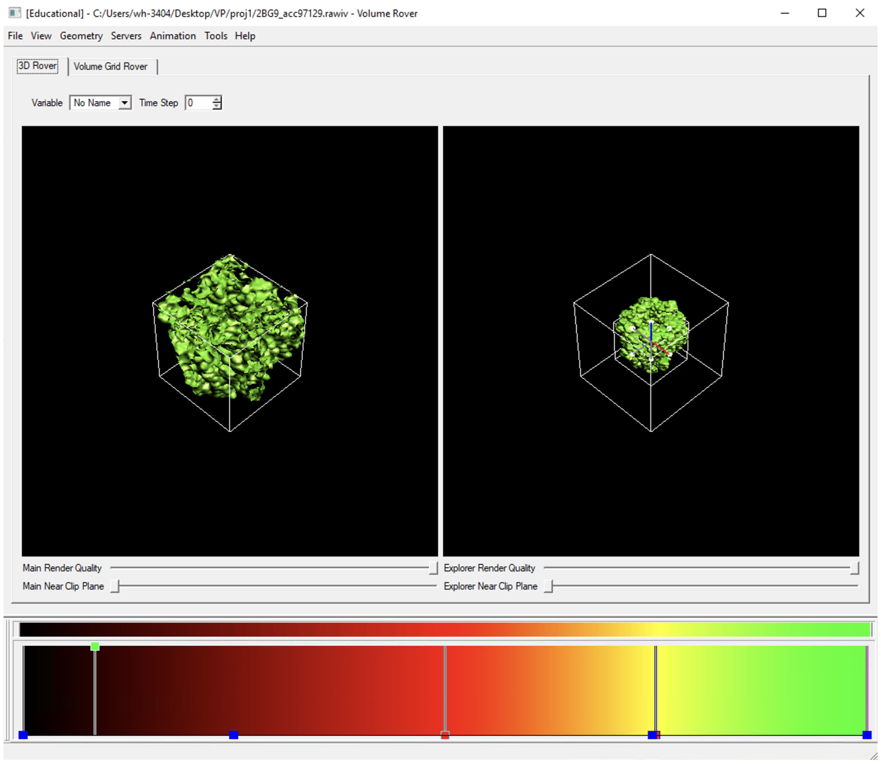

Figure 3: Isosurface of potential field

Problem 2

In problem 2, we are provided with a dataset consisting of an electric potential field over a domain. We select an isosurface of the field as shown in Figure 4 and extract it in the form of a triangle mesh, displayed in **??**. We interpolate the volumetric potential field to the isosurface by means of trilinear interpolation, as illustrated in § . The minimum, maximum, and mean of the potential field on the isosurface are

Quantity	**Value**
Minimum	-334.4745
Maximum	198.5442
Mean	-3.2103

Table 1: Statistics on the isosurface

The isosuface is replotted with coloring such that regions with value less that or greater than ± 0.1 are colored red and blue respectively. The plot is displayed in Figure 5.

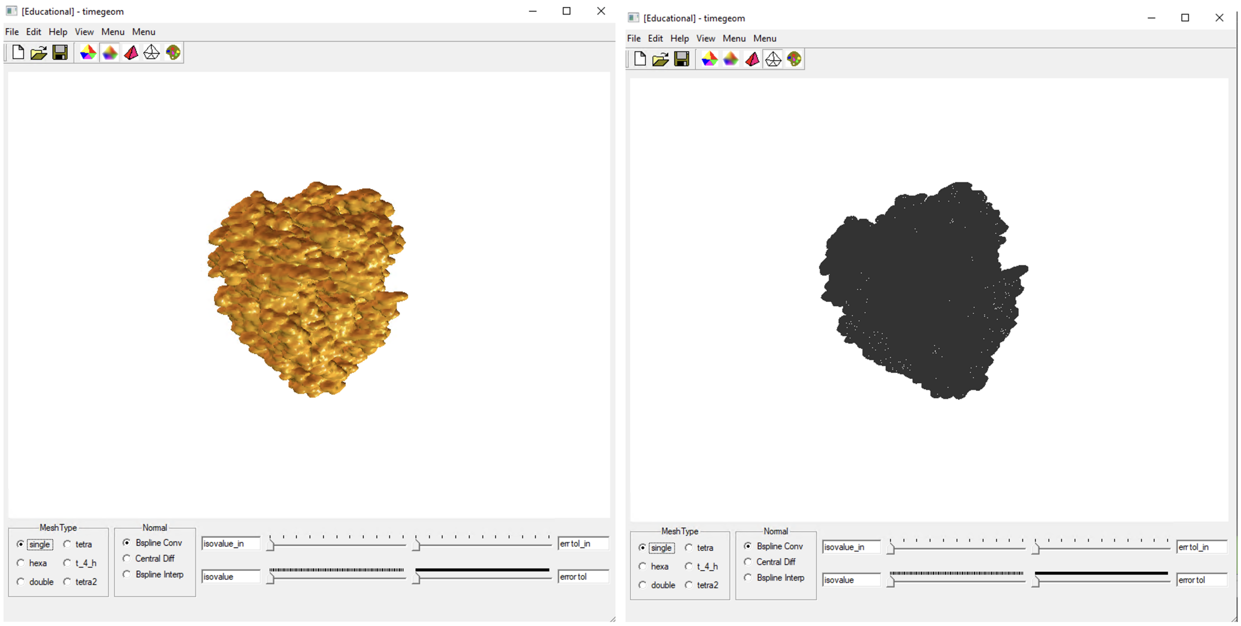

Figure 4: Isosurface of potential field

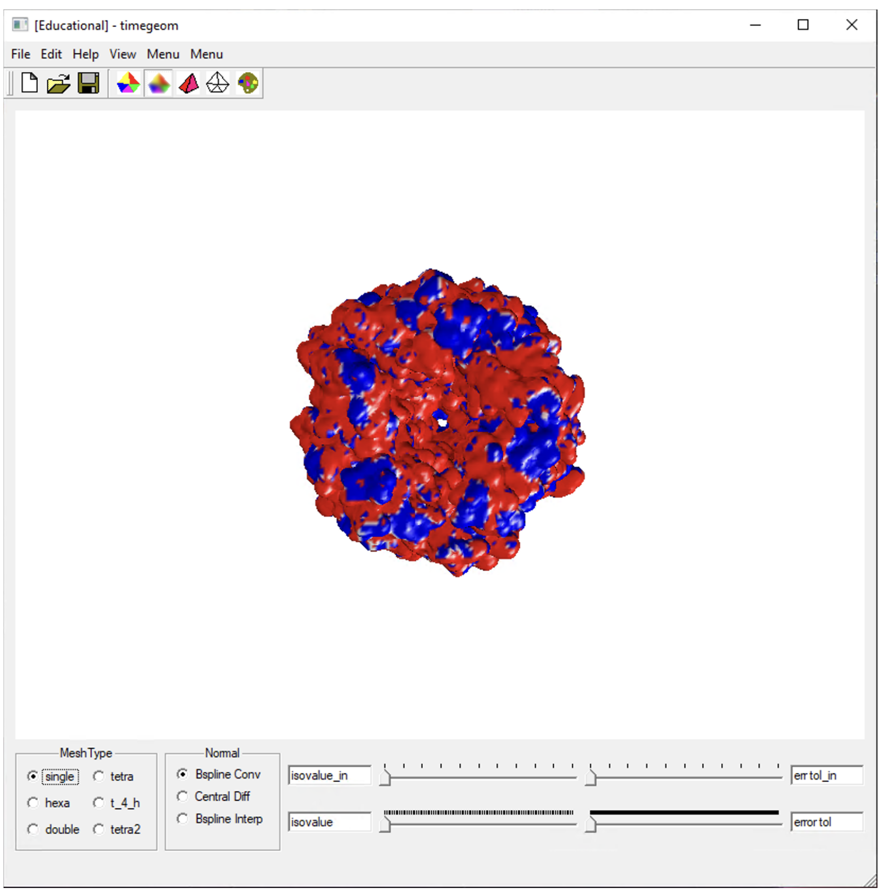

Figure 5: Isosurface of potential field

Code

`prob2.m`

```
%
pfile = '2BG9_pot97129.rawiv';
pfile = readRawiv(pfile);

pot = pfile.image;
xm = linspace(pfile.minXYZ(1), pfile.maxXYZ(1), pfile.dimXYZ(1));
ym = linspace(pfile.minXYZ(2), pfile.maxXYZ(2), pfile.dimXYZ(2));
zm = linspace(pfile.minXYZ(3), pfile.maxXYZ(3), pfile.dimXYZ(3));

tfile = 'iso_tri.raw';
tfile = readTriRaw(tfile);

Nv = tfile.nVerts;
xyz  = horzcat(tfile.x, tfile.y, tfile.z);
pval = zeros(Nv, 1);

for i=1:Nv
    pval(i) = compute_val(pot, xyz(i,:), xm, ym, zm);
end

pmax  = max(pval)
pmin  = min(pval)
pmean = sum(pval) / Nv

rgb = zeros(Nv, 3);
for i=1:Nv
    if pval(i) > 0.1;
        rgb(i,:) = [0, 0, 1];
    elseif pval(i) < -0.1;
        rgb(i,:) = [1, 0, 0];
    else
        rgb(i,:) = [1, 1, 1];
    end
end

Ne = tfile.nElems;

filename = 'iso_tri.rawc';
fid = fopen(filename, 'w');
fprintf(fid, [num2str(Nv), ' ', num2str(Ne), '\n']);
```

```
dlmwrite(filename, horzcat(xyz, rgb), 'delimiter', ' ', '-append')
    ;
dlmwrite(filename, tfile.A           , 'delimiter', ' ', '-append')
    ;
type(filename);

%=============================================================%
% HELPER FUNCTIONS
%=============================================================%
function ixyz = voxel_idx(xyz, xm, ym, zm)
    % returns index to bottom-left corner of voxel

    x = xyz(1);
    y = xyz(2);
    z = xyz(3);

    dx = xm(2) - xm(1);
    dy = ym(2) - ym(1);
    dz = zm(2) - zm(1);

    ix = floor(x / dx) + 1;
    iy = floor(y / dy) + 1;
    iz = floor(z / dz) + 1;

    ixyz = [ix, iy, iz];
end

function uvw = local_coordinates(xyz, ixyz, xm, ym, zm)
    % xyz: point coordinates
    % vi : voxel bottom left corner index

    ix = ixyz(1);
    iy = ixyz(2);
    iz = ixyz(3);

    x0 = xm(ix  );
    x1 = xm(ix + 1);

    y0 = ym(iy  );
    y1 = ym(iy + 1);

    z0 = zm(iz  );
    z1 = zm(iz + 1);

    x = xyz(1);
```

```
    y = xyz(2);
    z = xyz(3);

    u = (x - x0) / (x1 - x0);
    v = (y - y0) / (y1 - y0);
    w = (z - z0) / (z1 - z0);

    assert(-1e-12 < u < 1.0 + 1e-12);
    assert(-1e-12 < v < 1.0 + 1e-12);
    assert(-1e-12 < w < 1.0 + 1e-12);

    uvw = [u, v, w];
end

function wts = intp_wts(u, ixyz)

    ix = ixyz(1);
    iy = ixyz(2);
    iz = ixyz(3);

    wts.u000 = u(ix  , iy  , iz  );

    wts.u100 = u(ix+1, iy  , iz  );
    wts.u010 = u(ix  , iy+1, iz  );
    wts.u001 = u(ix  , iy  , iz+1);

    wts.u110 = u(ix+1, iy+1, iz  );
    wts.u101 = u(ix+1, iy  , iz+1);
    wts.u011 = u(ix  , iy+1, iz+1);

    wts.u111 = u(ix+1, iy+1, iz+1);

end

function val = interpolate(wts, uvw)
    u = uvw(1);
    v = uvw(2);
    w = uvw(3);

    val   =       wts.u000 * (1 - u) * (1 - v) * (1 - w);
%
    val = val +   wts.u100 *      u  * (1 - v) * (1 - w);
    val = val +   wts.u010 * (1 - u) *      v  * (1 - w);
    val = val +   wts.u001 * (1 - u) * (1 - v) *      w ;
%
```

```
    val = val +  wts.u110 *      u   *      v   * (1 - w);
    val = val +  wts.u101 *      u   * (1 - v) *      w ;
    val = val +  wts.u011 * (1 - u) *      v   *      w ;
%
    val = val +  wts.u111 *      u   *      v   *      w ;

end

function val = compute_val(u, xyz, xm, ym, zm)
    % u - array to sample
    % xyz - point coordinates

    ixyz = voxel_idx(xyz, xm, ym, zm);
    uvw  = local_coordinates(xyz, ixyz, xm, ym, zm);

    wts = intp_wts(u, ixyz);
    val = interpolate(wts, uvw);
end
```

```
>> prob2

pmax =

  198.5442


pmin =

 -334.4745


pmean =

   -3.2103
```