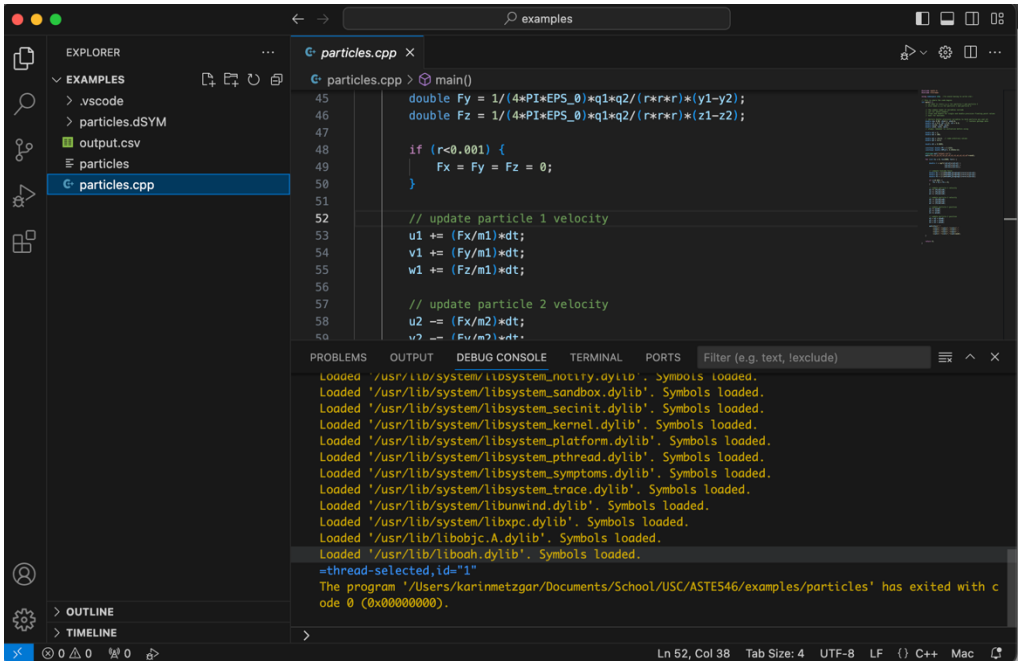
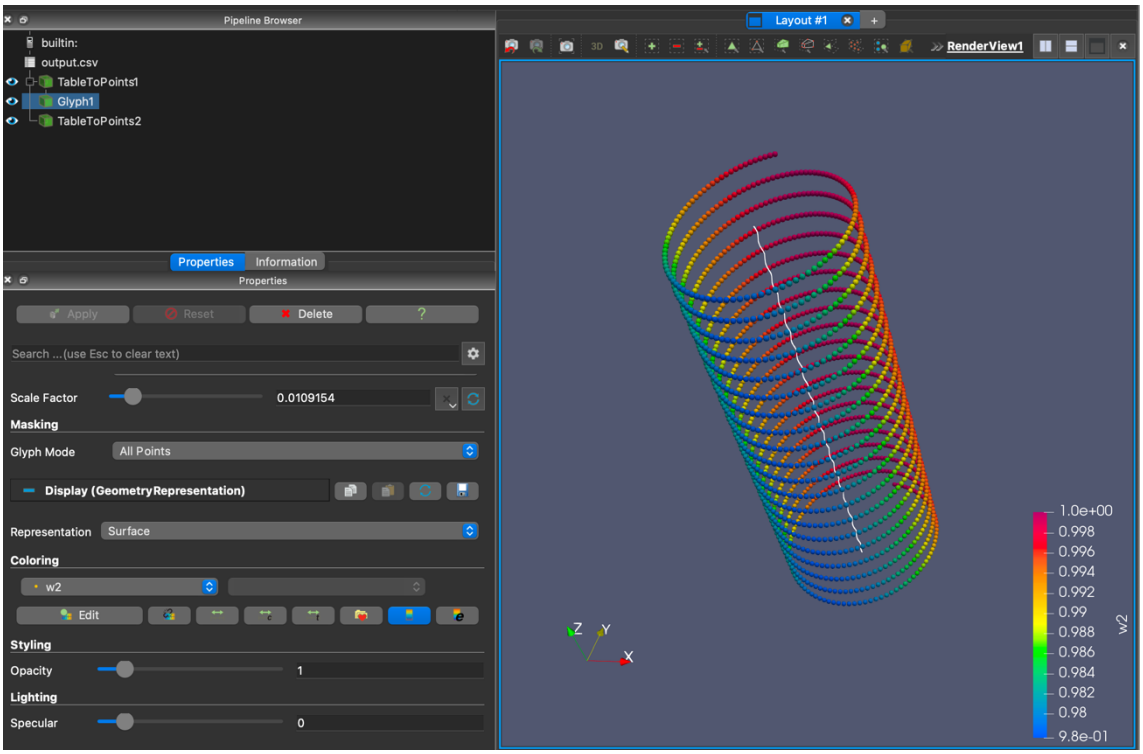
Karin Metzgar
ASTE 546
HW #1

Part 1: Compile particles.cpp and make visualization in ParaView.

I was able to setup VSCode for Mac and compile the code, and then create the visual of output.csv in ParaView following instructions from the lecture slides. Cool!





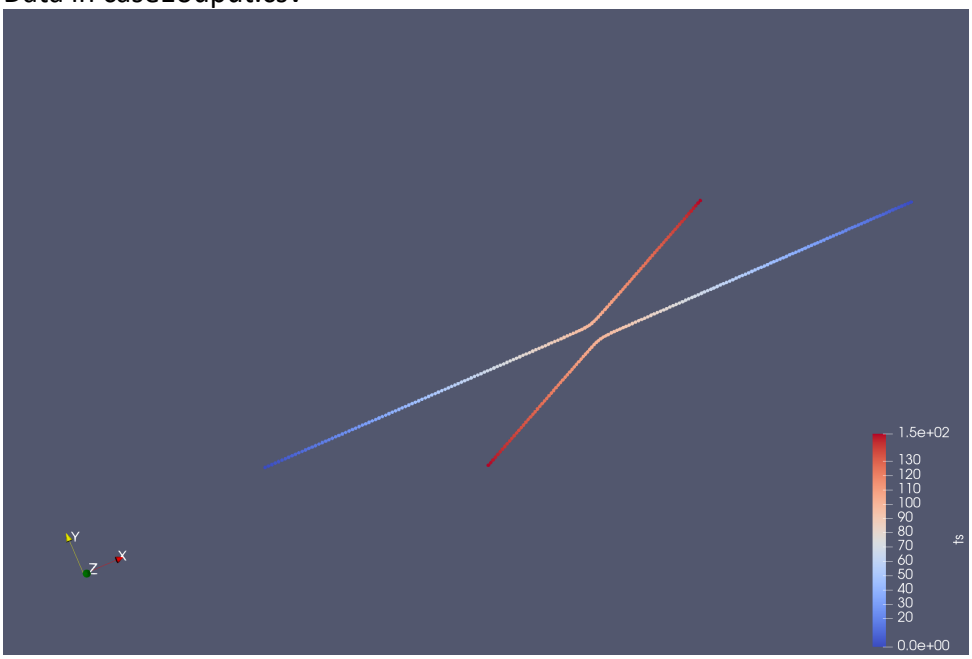
Part2:

For part 2 I edited the initial conditions to match those given in the assignment.

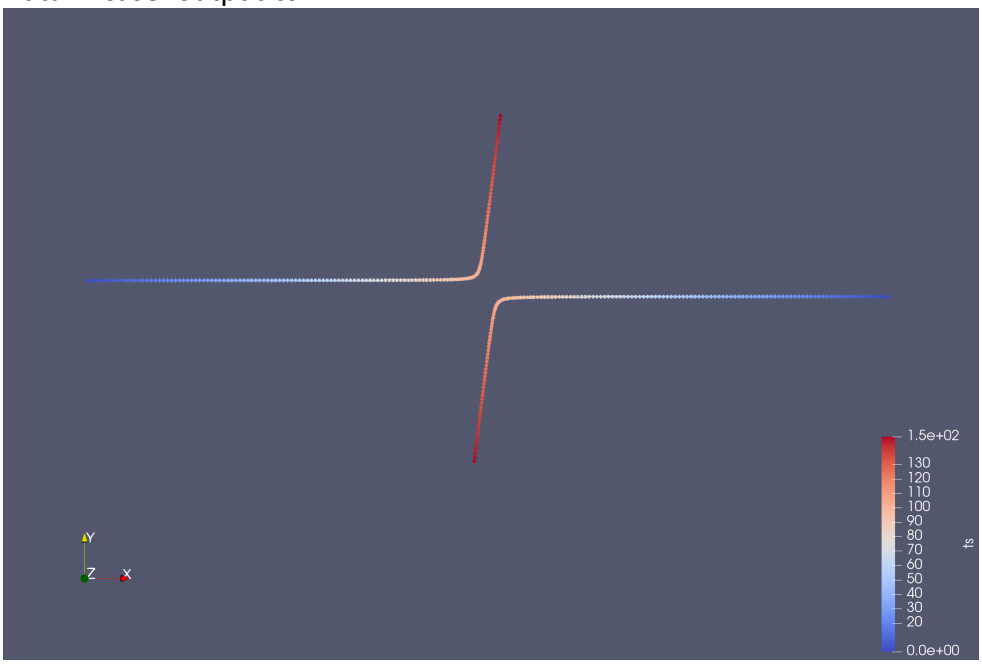
```cpp
// this is where the code begins
int main() {
    // we need to store x,y,z for particle 1 and particle 2
    // also need u,v,w for particle 1 and particle 2

    // the common types of variables include
    // int for whole numbers
    // float and double for single and double precision floating point values
    // bool for booleans

    // declare double precision variables to hold particle pos and vel
    double x1= 0.00, y1=0.2, z1=0.0;           // contain garbage data
    double x2 = 0.0, y2 = 0.0, z2 = 0.0;
    double u1=20, v1=0, w1=0;
    double u2=0, v2=0, w2=1;
    // always remember to initialize before using

    double m1 = 1;
    double m2 = 100;

    double q1 = -1e-4;  // some arbitrary values
    double q2 = 1e-4;

    double dt = 0.0005;

    constexpr double PI = 3.1415;
    constexpr double EPS_0 = 8.8542e-12;

    ofstream out("output.csv");
    out<<"ts,x1,y1,z1,x2,y2,z2,u1,v1,w1,u2,v2,w2"<<endl;
```

Case1:

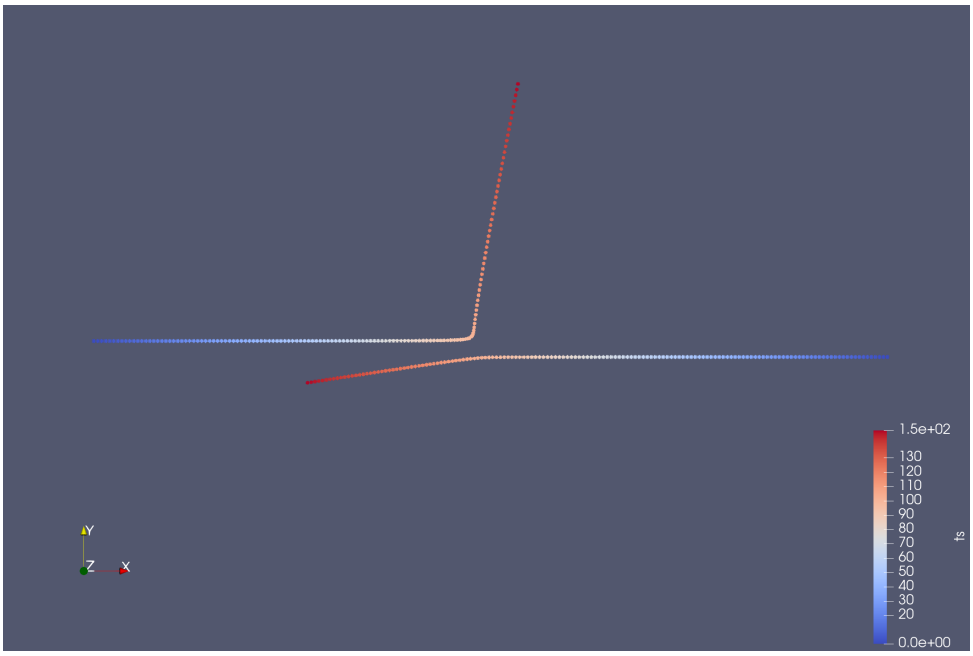
Data in case1ouput.csv



Case 2:
Data in case2output.csv



Case 3:
Data in case3ouput.csv



Part 3:
I did not find this Part to be easy ☺
Code found in particlesPart3.cpp
First I didn't realize that when defining the array you can't use expressions? Or maybe you can and I wasn't doing it right but putting in –(1/3) gave a value of 0? And that took me a WHILE to notice. Also ended up putting the output code in the wrong place so it was iterating the output as i++ also..
I didn't use a for loop to define the arrays as suggested in the assignment, I wasn't sure how that would be easier? Hope this is okay.

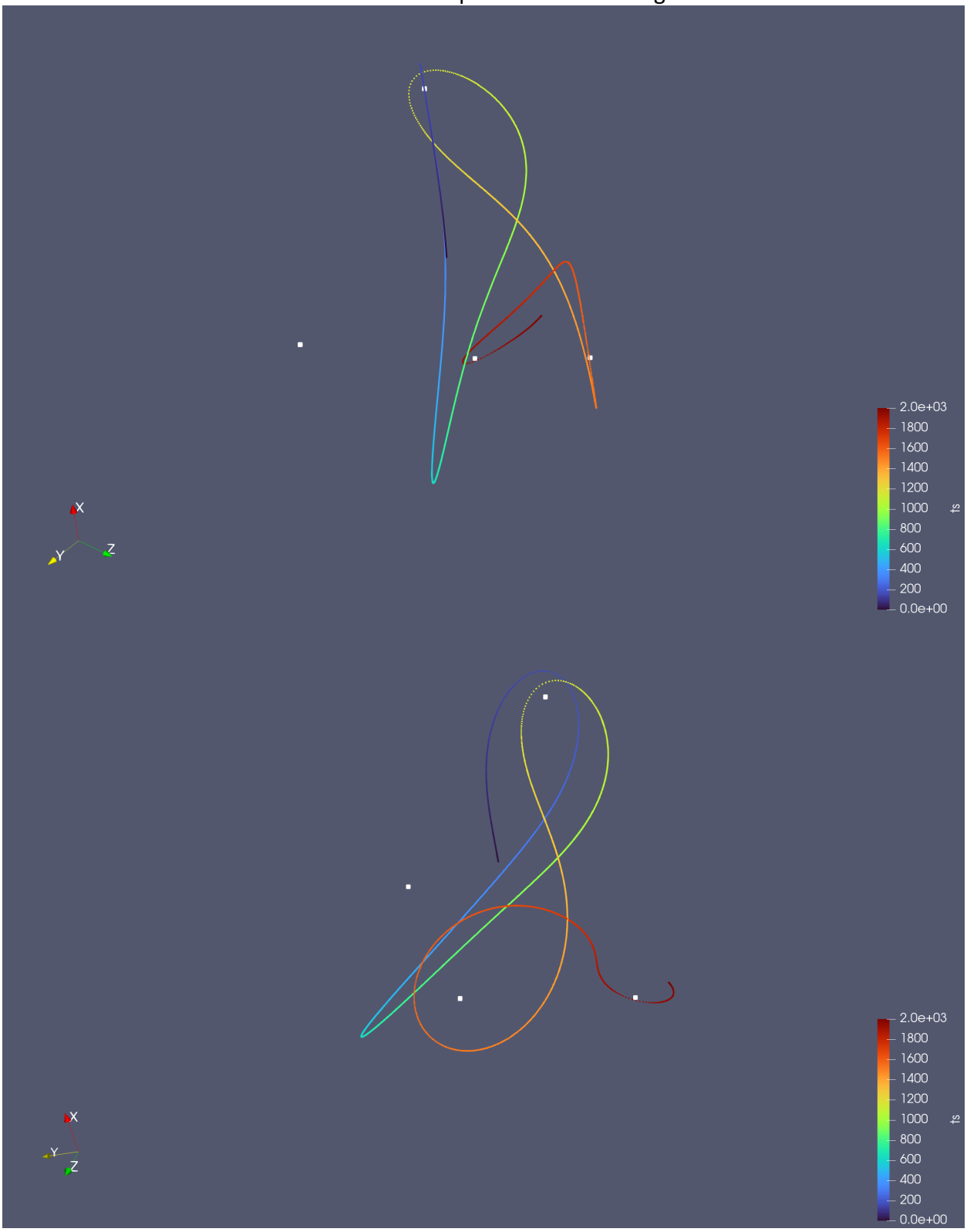
```
//initialize stuff
    double dt = 0.00005;

    //Part 3:
    constexpr int N = 5;
    double x[N] = {0,0.816,-0.816,0,0.2};
    double y[N] = {0,-0.333,-0.333,0.666,0};
    double z[N] = {0.816,-0.333,-0.333,-0.333,0};
    double u[N] = {0,0,0,0,100};
    double v[N] = {0,0,0,0,-15};
    double w[N] = {0,0,0,0,10};
    double m[N] = {1e6,1e6,1e6,1e6,1};
    double q[N] = {5e-3,5e-3,5e-3,5e-3,-1e-4};


    constexpr double PI = 3.1415;
    constexpr double EPS_0 = 8.8542e-12;

    ofstream out("part3output1.csv");
    out<<"ts";
    for (int i=0; i<N; i++){
        out<<",x"<<i<<",y"<<i<<",z"<<i;
    }
    out<<"\n";
```

```
for (int ts = 0; ts<2000; ts++) {

    for (int i=0; i<N; i++)
    {
        double Fx = 0;
        double Fy = 0;
        double Fz = 0;

        for(int j=0; j<5; j++)  {

            double r = sqrt((x[j]-x[i])*(x[j]-x[i]) +
                            (y[j]-y[i])*(y[j]-y[i]) +
                            (z[j]-z[i])*(z[j]-z[i]));

            // compute Coulomb force


            if (i!=j && r>0.001) {
                Fx += 1/(4*PI*EPS_0)*q[i]*q[j]/(r*r*r)*(x[i]-x[j]);
                Fy += 1/(4*PI*EPS_0)*q[i]*q[j]/(r*r*r)*(y[i]-y[j]);
                Fz += 1/(4*PI*EPS_0)*q[i]*q[j]/(r*r*r)*(z[i]-z[j]);
            }
```

I originally had
<<x[i]<<",”<<y[i]<<",”<<z[i]<< .. at the bottom but I was only getting data for columns 1-4 in the output file, so I’m not sure if there is a way to do that within the for loops that is more elegant? But this is how I got data for all the particles to output properly.

```
        }
        // update particle 1 velocity
            u[i] += (Fx/m[i])*dt;
            v[i] += (Fy/m[i])*dt;
            w[i] += (Fz/m[i])*dt;

            // update particle 2 velocity
            //u[j] -= (Fx/m[j])*dt;
            //v[j] -= (Fy/m[j])*dt;
            //w[j] -= (Fz/m[j])*dt;

            // update particle 1 position
            x[i] += u[i]*dt;
            y[i] += v[i]*dt;
            z[i] += w[i]*dt;

            // update particle 2 position
            //x[j] = x[j] + u[j]*dt;
            //y[j] = y[j] + v[j]*dt;
            //z[j] = z[j] + w[j]*dt;

    }
    out<<ts<<","
            <<x[0]<<","<<y[0]<<","<<z[0]<<","
            <<x[1]<<","<<y[1]<<","<<z[1]<<","
            <<x[2]<<","<<y[2]<<","<<z[2]<<","
            <<x[3]<<","<<y[3]<<","<<z[3]<<","
            <<x[4]<<","<<y[4]<<","<<z[4]<<endl;

}


return 0;
```

Pictures below are from the initial conditions provided in the assignment.



I then added a positive z component to the electron's initial velocity in part3output1.csv