## LIC algorithm

Line integral convolution pseudocode:

1. Define stream line seed point at the center of pixel (x, y)
   1. Seed point P0 = (x + 0.5, y + 0.5) where x and y are coordinates of the pixel
2. (Advection) Compute local stream lines from seed point, forward and backward
   1. Iteratively compute the sequence of pixels a stream line travels through
   2. The next point is given by a vector defined as follows, added or subtracted relative to the current point.
      1. direction = the vector field value at the current pixel (measured at the integer floor of the current point)
      2. magnitude = the minimum needed to cross a pixel boundary in this direction from the current point
      3. Add this vector to the current point for forward advection
      4. Subtract it for backward advection
   3. Save the magnitude values and point values of all the advection steps.
   4. Keep computing for n many steps forward and n many steps backward.
3. Define a convolution kernel, such as a Hanning function.
4. Compute a finite integral of the kernel 2 \* n many times over different segments of the domain.
   1. The widths of the segments are the magnitude values from advection.
   2. Iterate, starting from 0 on each of two passes, once for the forward advection magnitudes and once for the backward advection magnitudes
   3. Start the next segment where the previous segment ended. Move in order along the lists of advection magnitudes.
   4. Compute the integral of each segment and store in two lists, one for the forward advection and one for the backward advection.
5. Use the point values from the advection steps to look up image brightness values in the source image data (typically an image of white noise) and compute a weighted average (convolution) of brightness values over the variable-stepped pixel traversal path
   1. Use the integral values as weights in weighted sums of the pixel brightness values
   2. Divide by the sum of all the integral values
6. Repeat 1-5 for every pixel in the image.

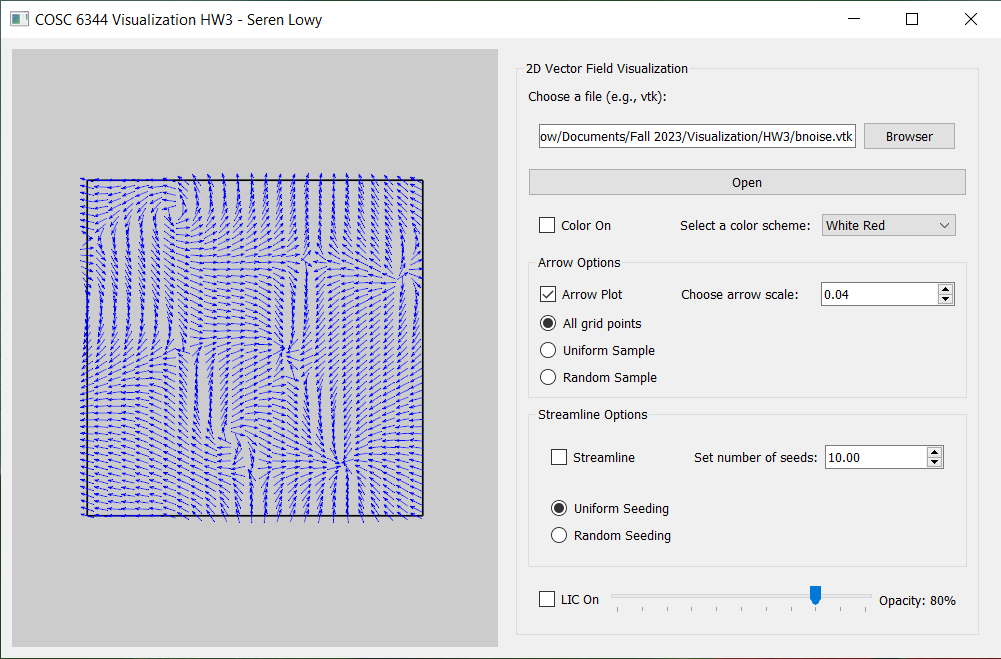
Source: Cabral, B. and Leedom, C. (1993). “Imaging Vector Fields Using Line Integral Convolution.” Lawrence Livermore National Laboratory.

## Placing streamlines

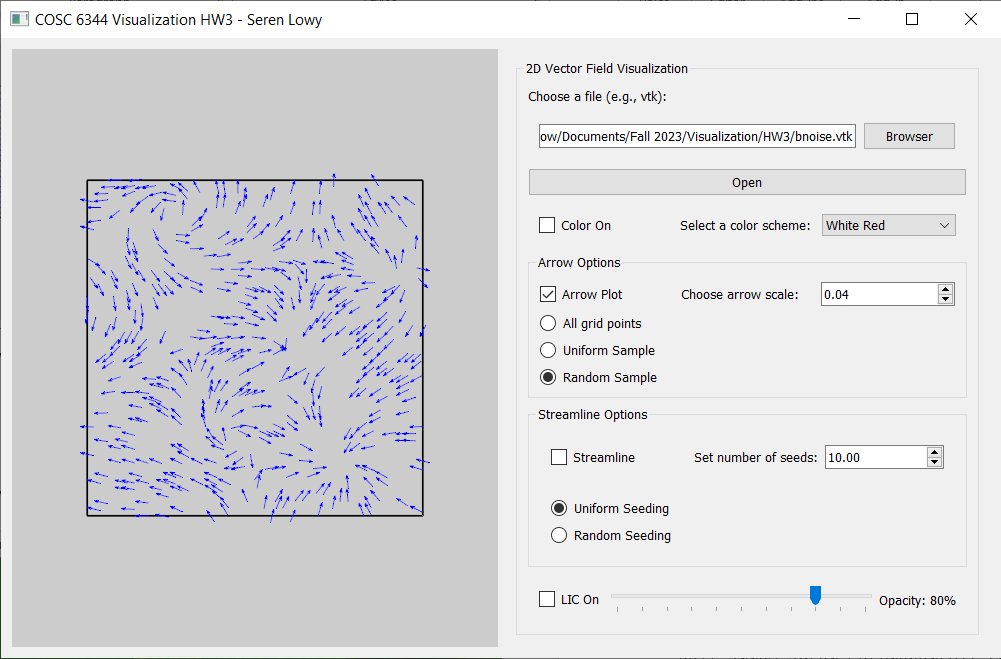
…

## 2. Arrow plots

### bnoise.vtk

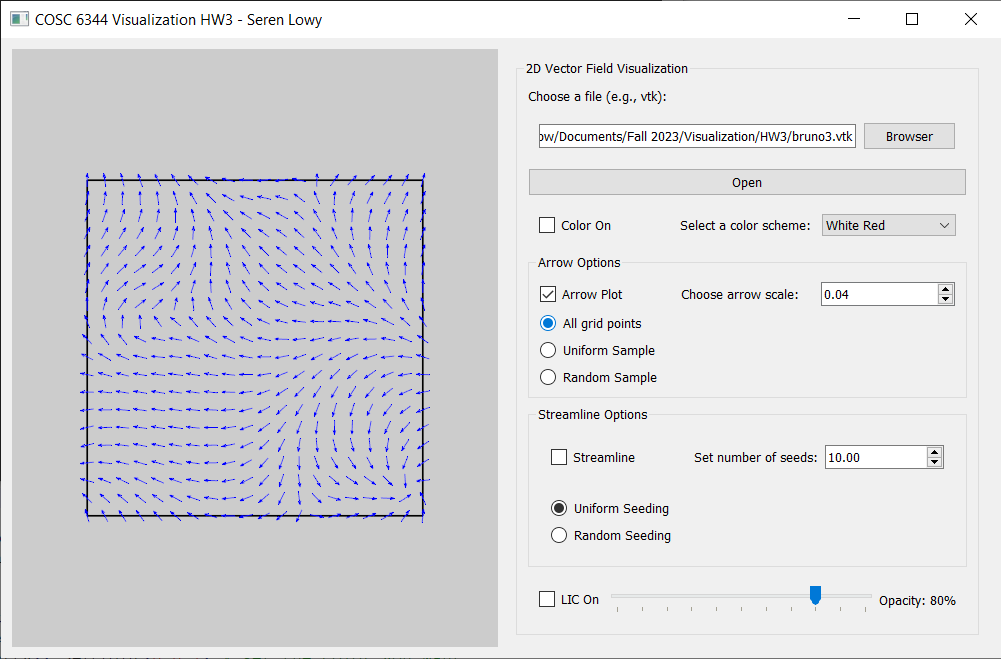


All arrows

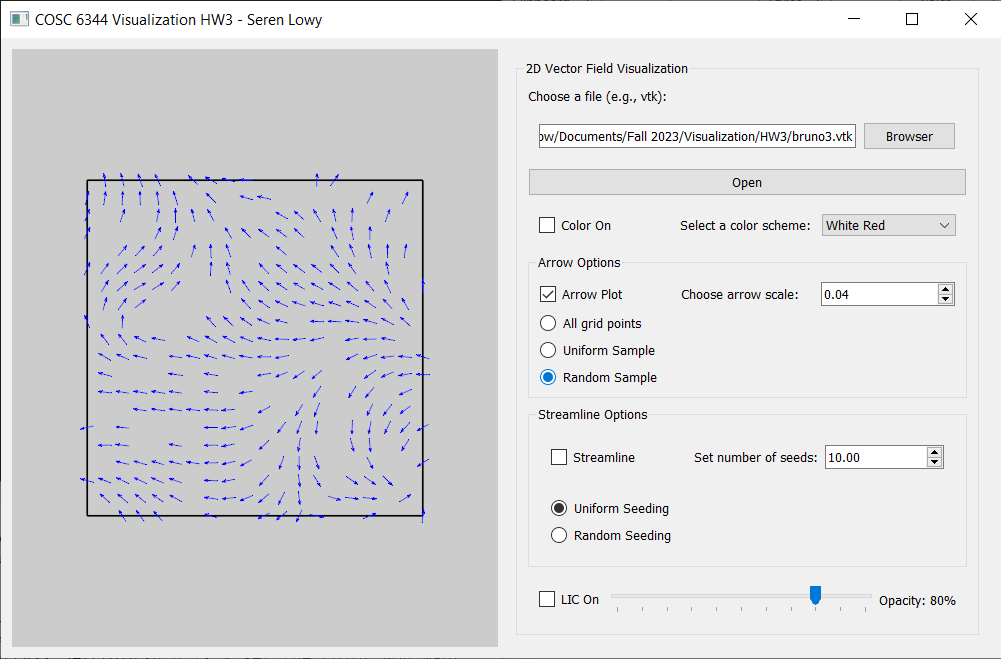


Random down sample (500 arrows)

### bruno3.vtk

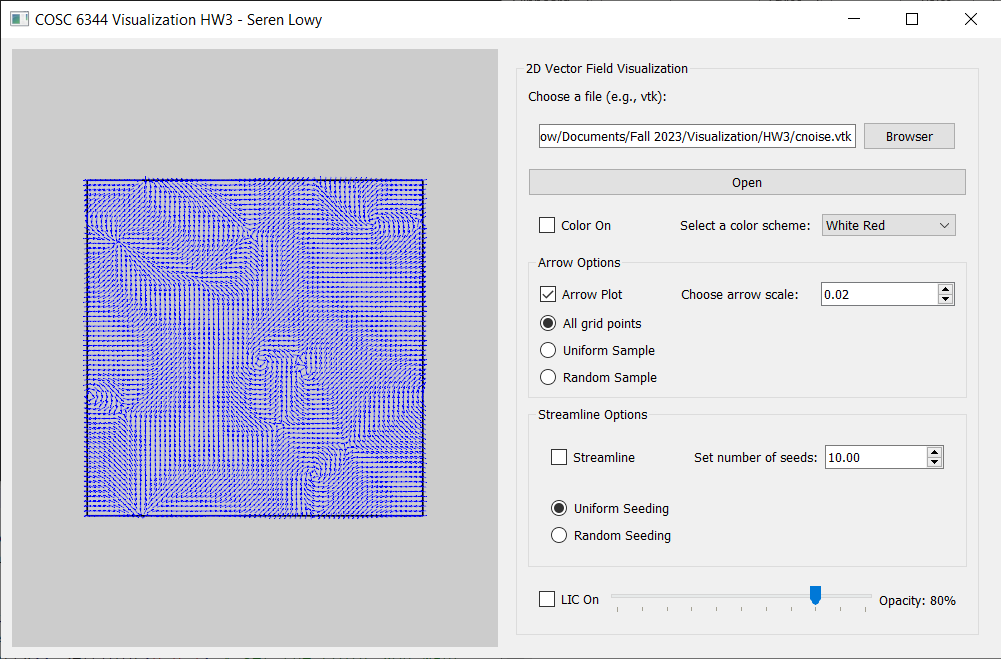


All arrows

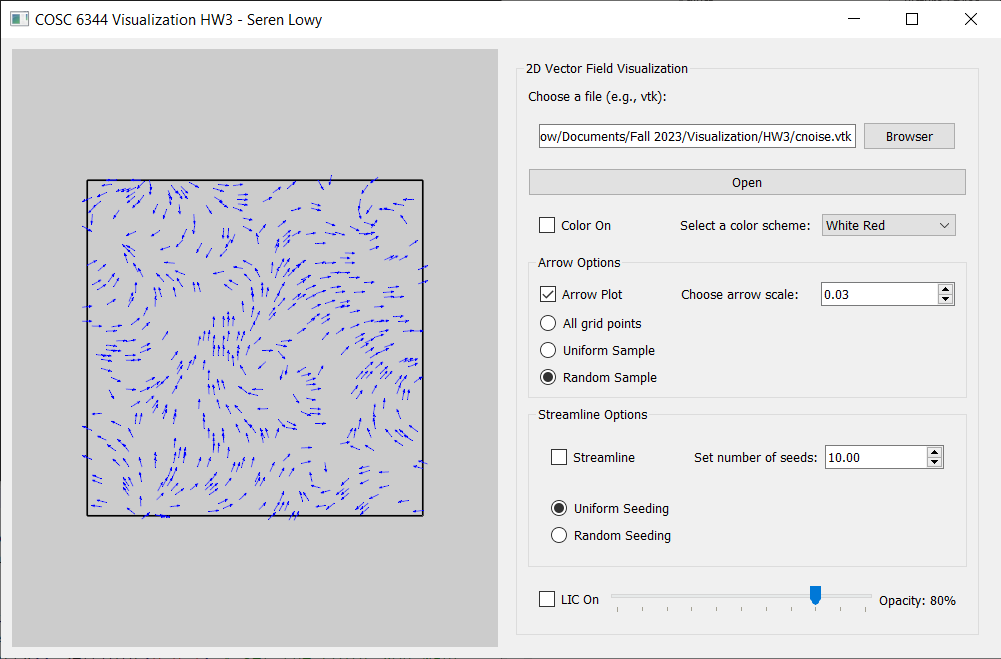


Random down sample (500 arrows)

### cnoise.vtk

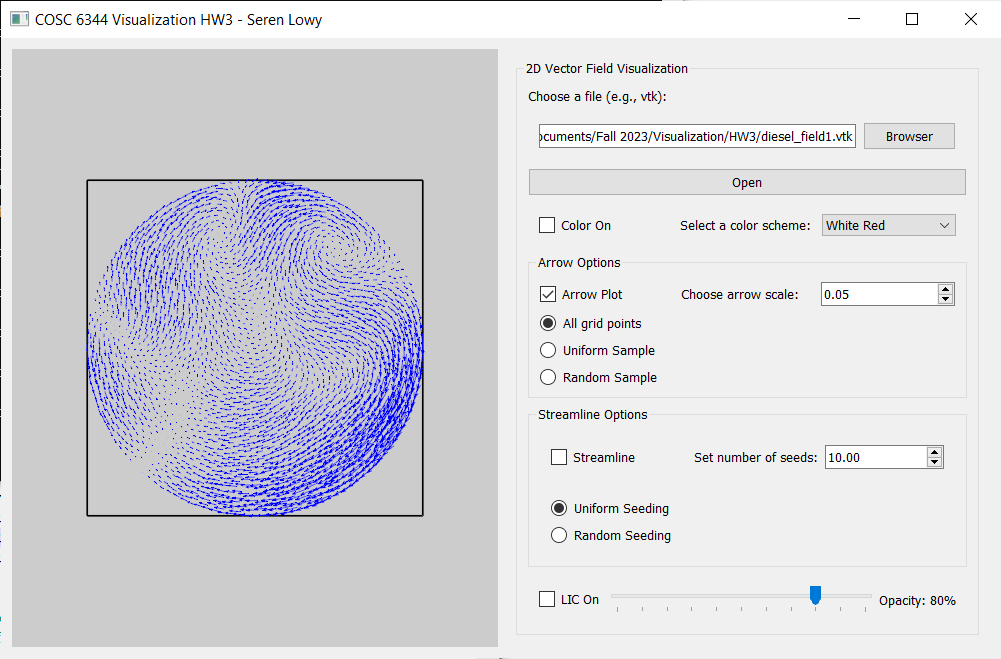


All arrows

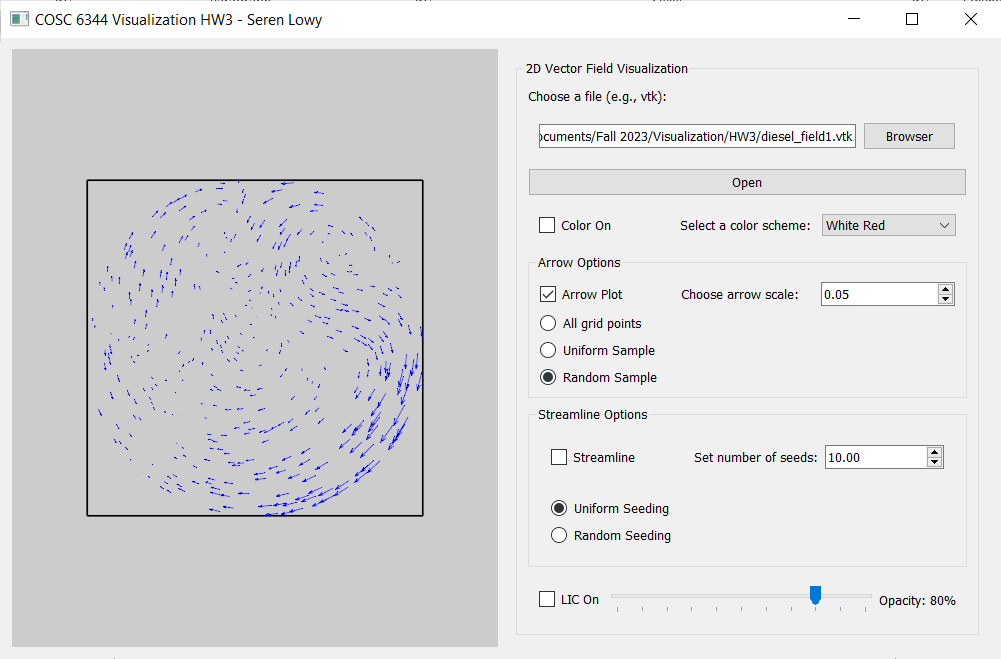


Random down sample (500 arrows)

### diesel\_field1.vtk

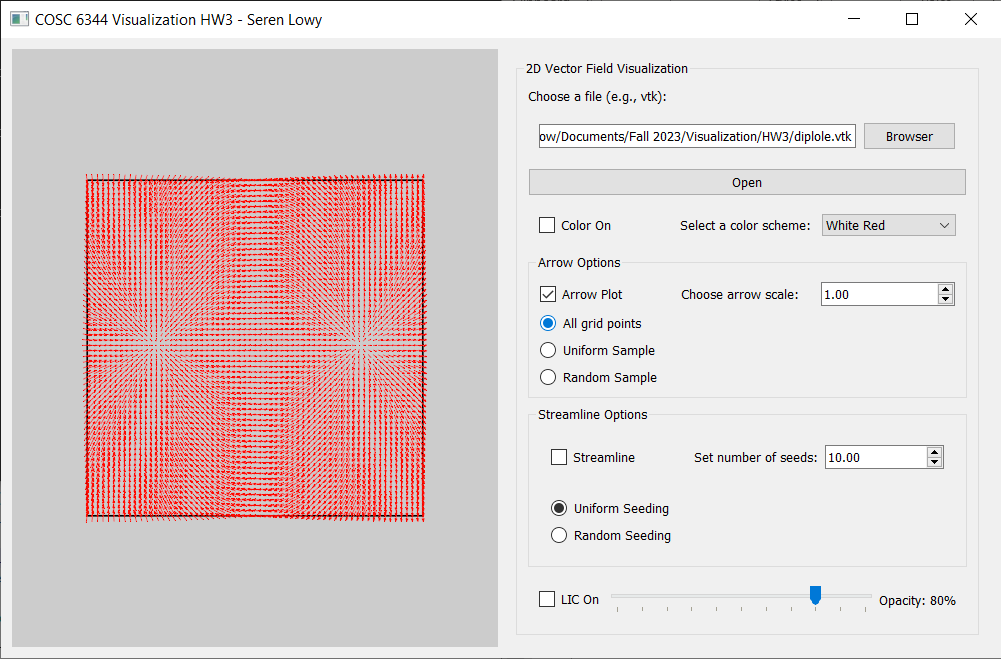


All arrows. The diesel\_field1 dataset appears to have vectors with different magnitudes.

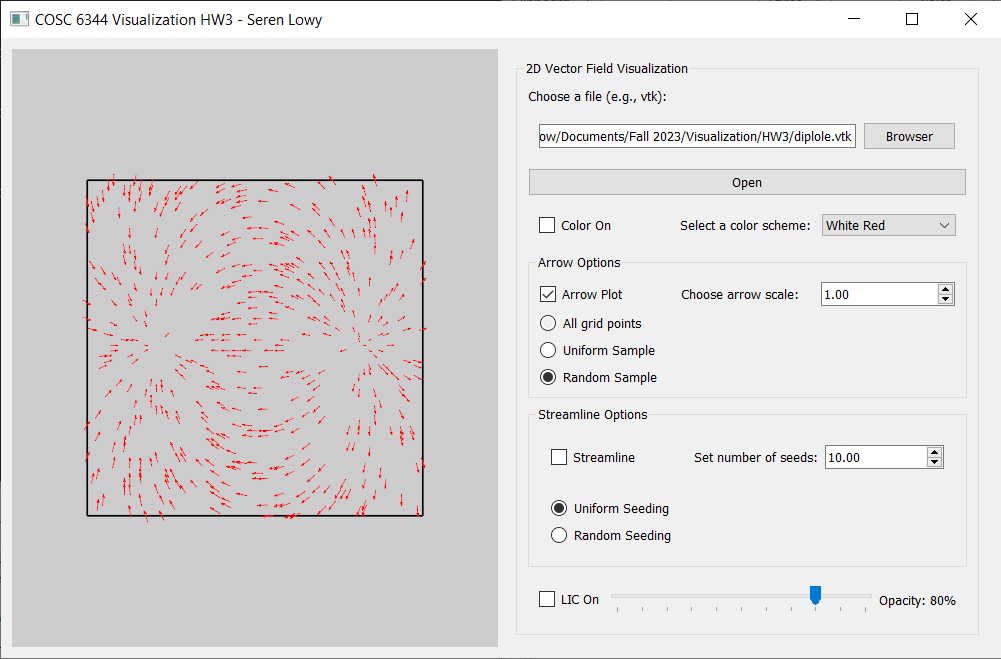


Random down sample (500 arrows)

### diplole.vtk

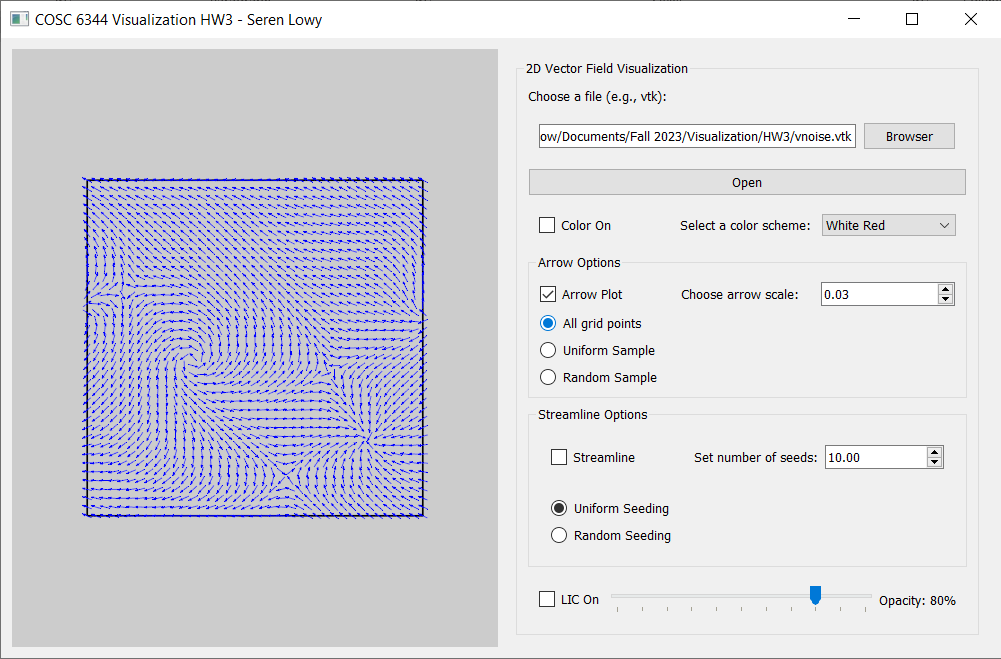


All arrows. The “diplole.vtk” dataset appears to have vectors with very low magnitudes. The arrow scale factor was set to 1.00 (much higher than other datasets) to make the arrows visible.

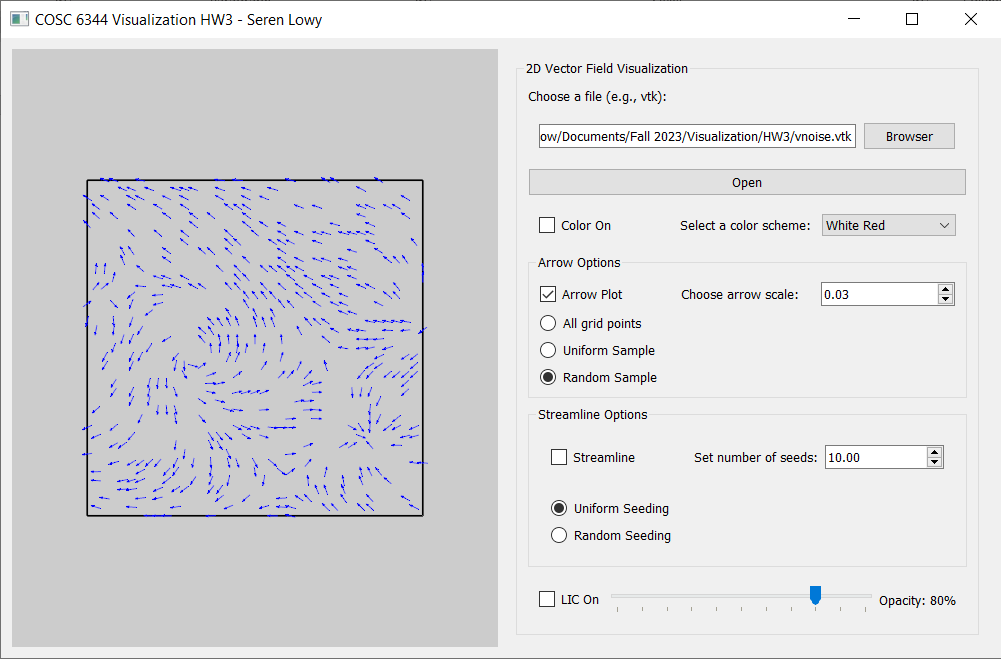


Random down sample (500 arrows)

### vnoise.vtk



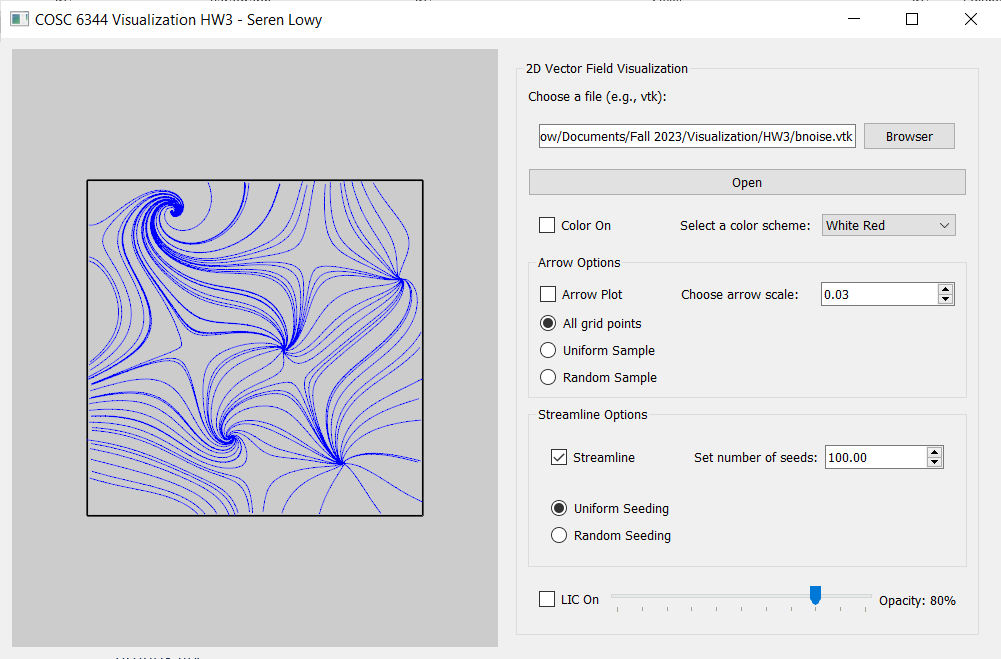
All arrows



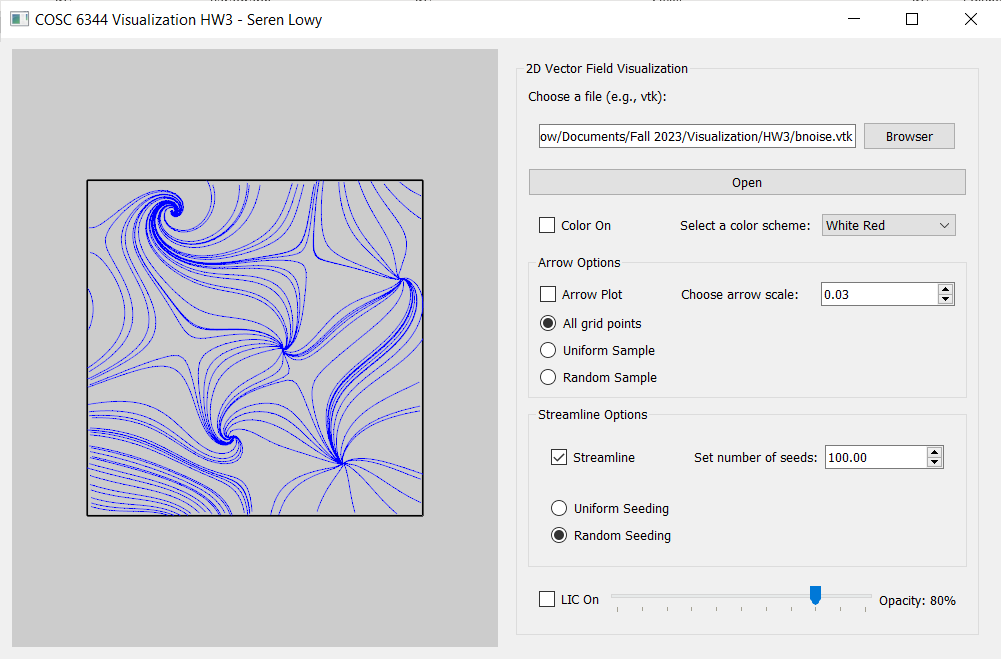
Random down sample (500 arrows)

## 3. Streamline plots

### bnoise.vtk

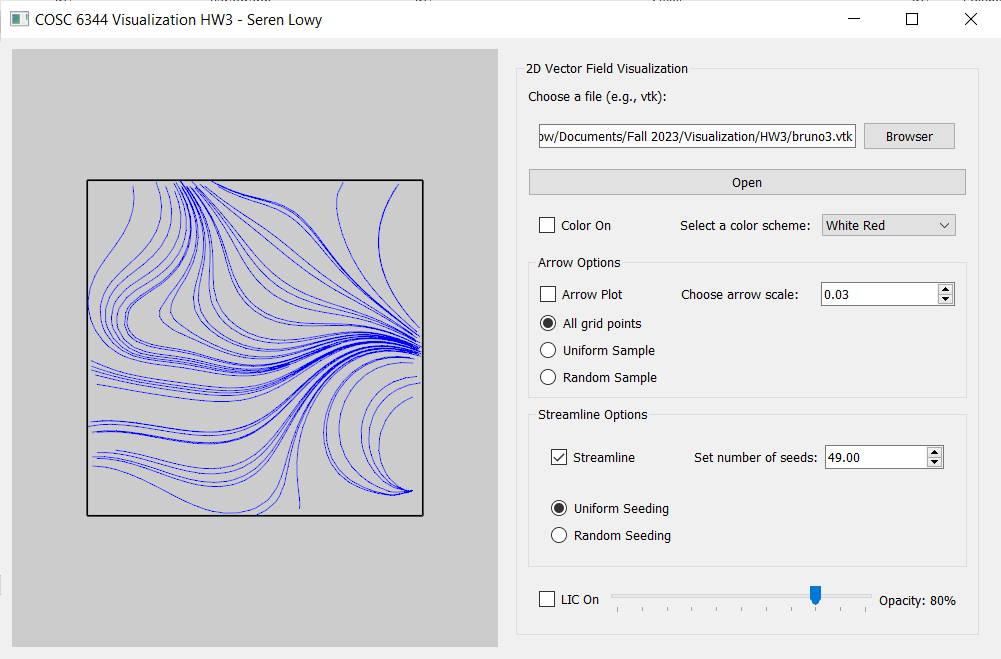


Uniform seed placement (100 seeds)

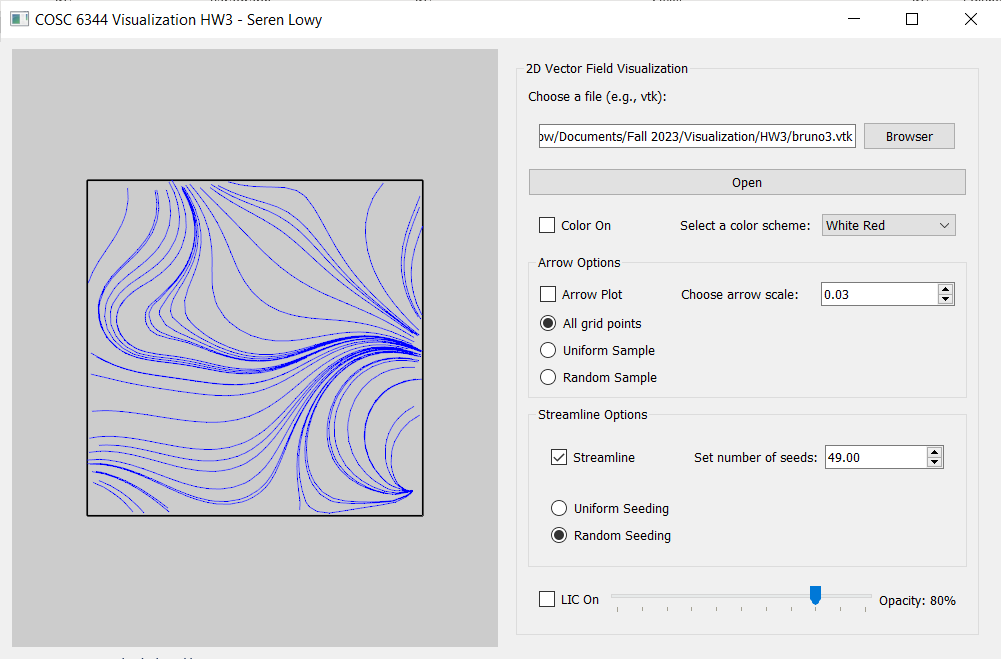


Random seed placement (100 seeds)

### bruno3.vtk

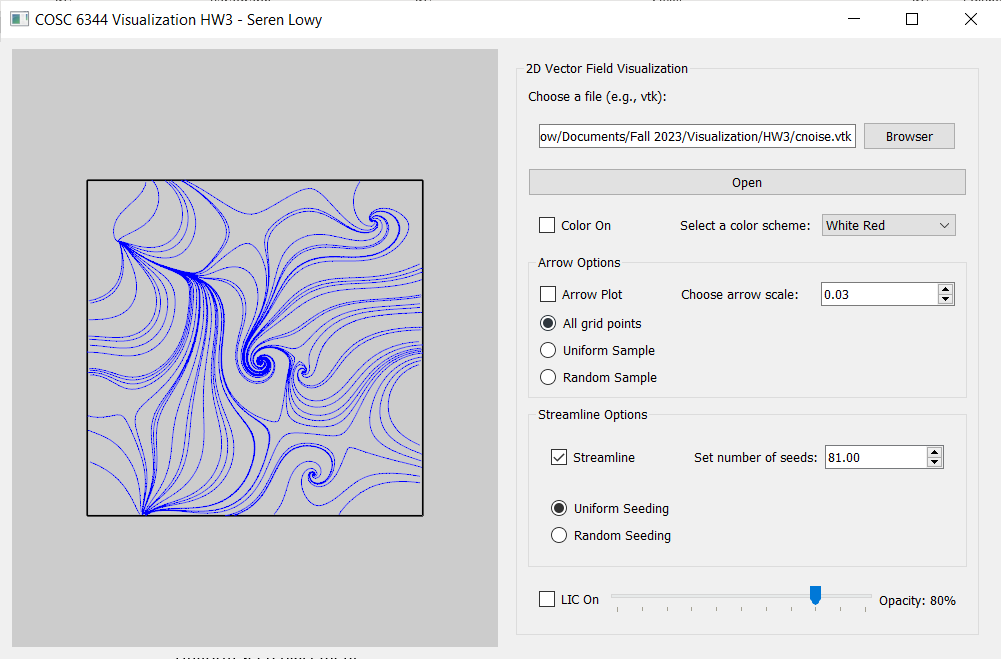


Uniform seed placement (49 seeds)

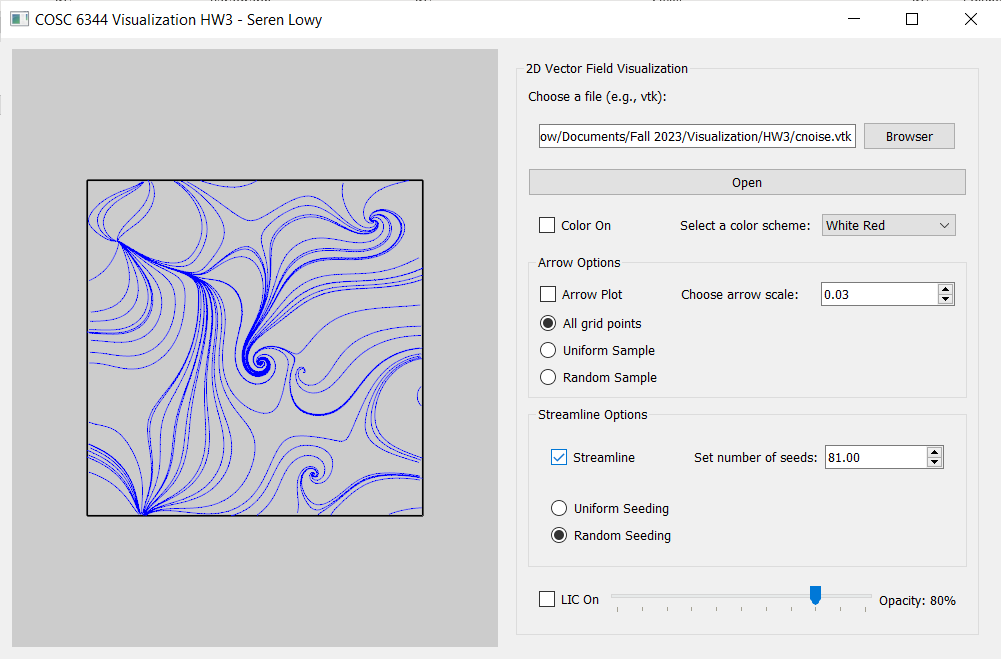


Random seed placement (49 seeds)

### cnoise.vtk

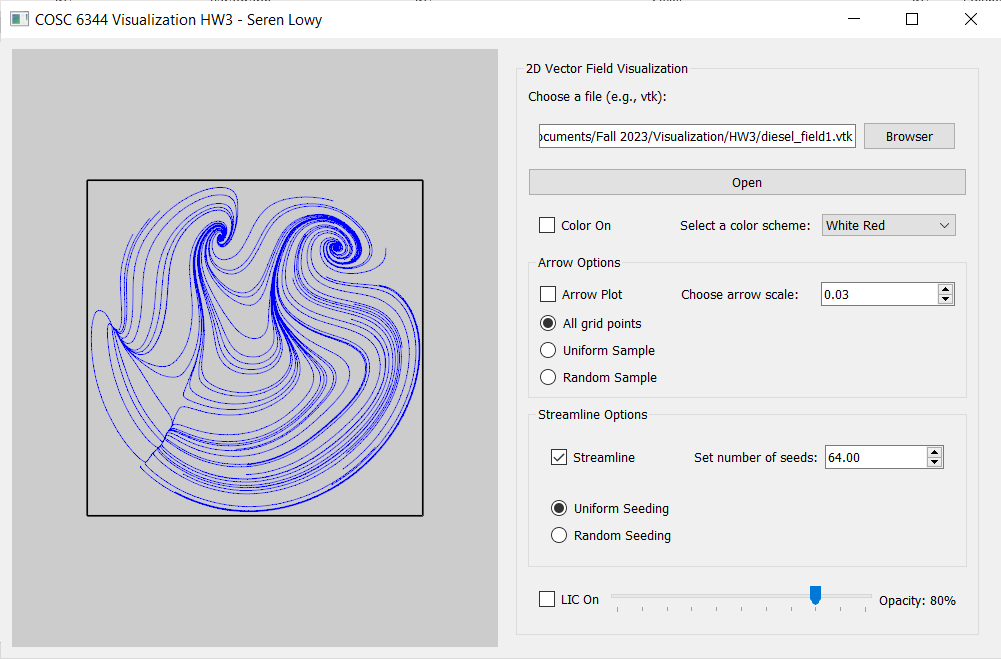


Uniform seed placement (81 seeds)

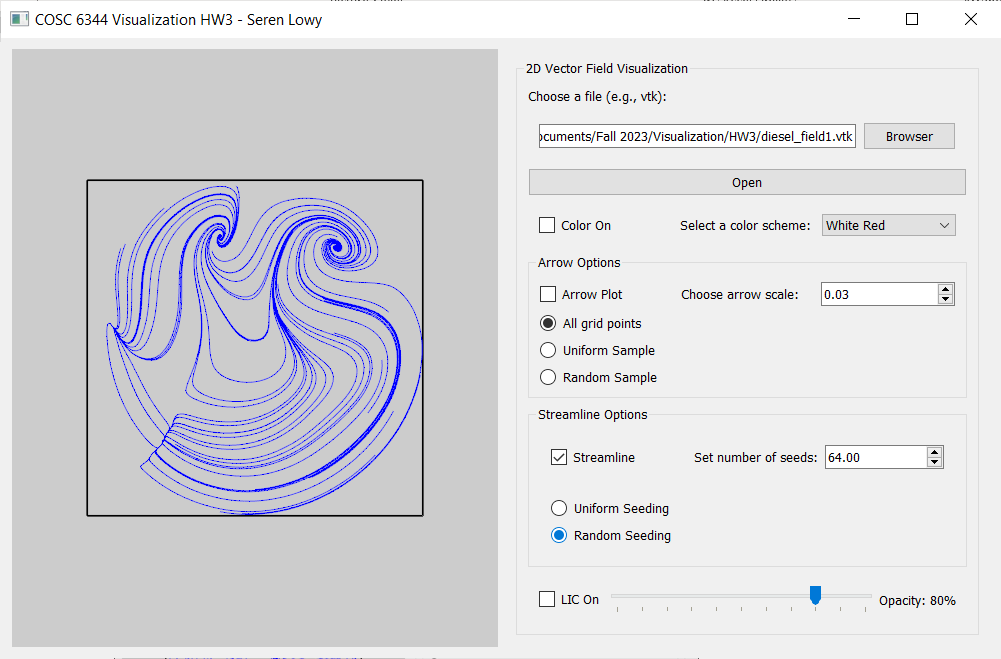


Random seed placement (81 seeds)

### diesel\_field1.vtk

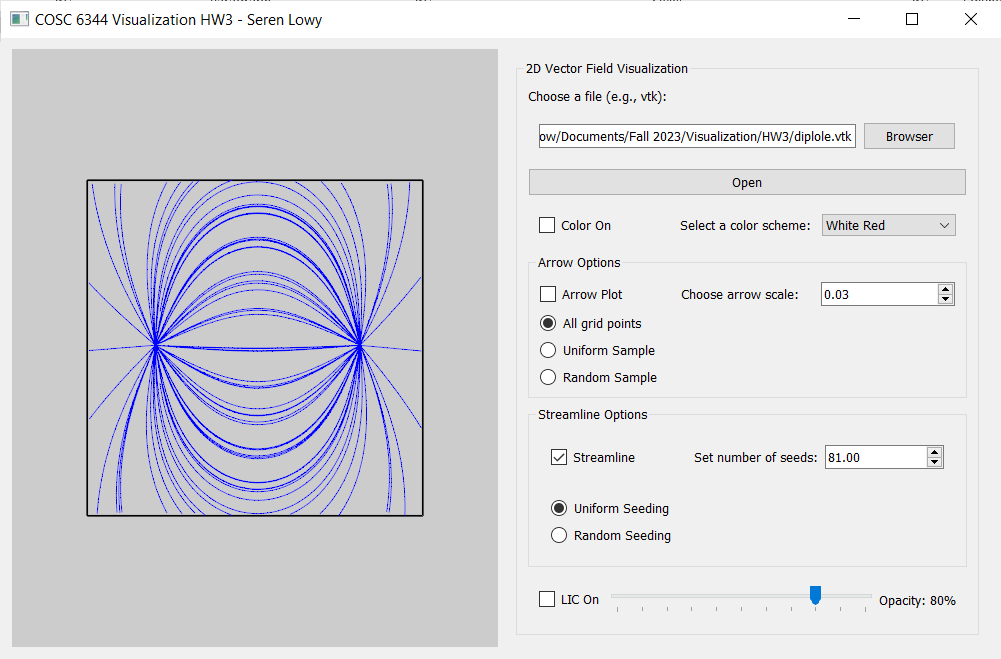


Uniform seed placement



Random seed placement (64 seeds)

### diplole.vtk

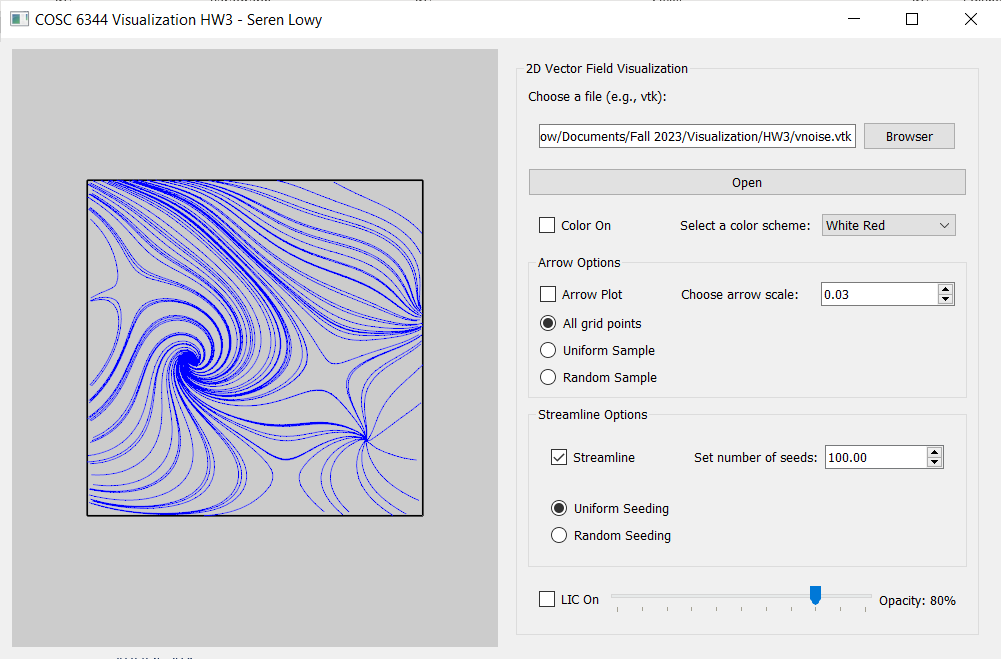


Uniform seed placement (81 seeds)

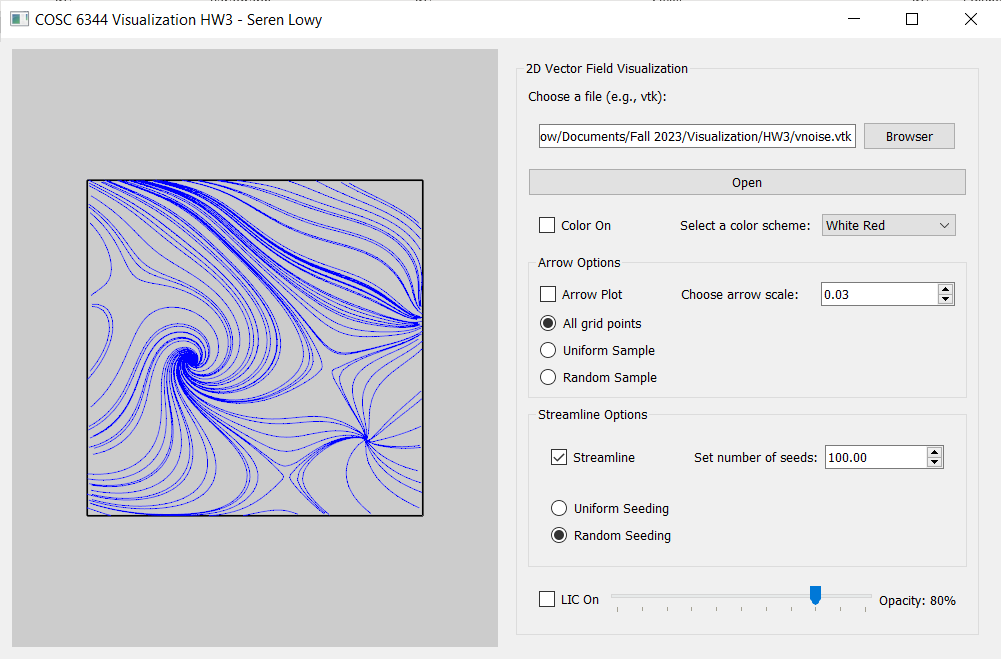


Random seed placement (81 seeds)

### vnoise.vtk



Uniform seed placement (100 seeds)



Random seed placement (100 seeds)

## 4. LIC textures

### bnoise.vtk

Parameters:

### bruno3.vtk

Parameters:

### cnoise.vtk

Parameters:

### diesel\_field1.vtk

Parameters:

### diplole.vtk

Parameters:

### vnoise.vtk

Parameters: