

1

Vector Visualization



Oregon State University

Mike Bailey
mjb@cs.oregonstate.edu



This work is licensed under a [Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License](https://creativecommons.org/licenses/by-nc-nd/4.0/).



Oregon State University
Computer Graphics

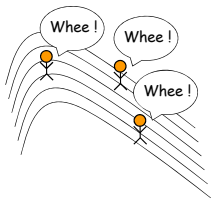
vector.pptx

mjb - March 12, 2019

2

What is a Vector Visualization Problem?

A vector has direction and magnitude. Typically science and engineering problems that work this way are those involving fluid flow through a velocity field.



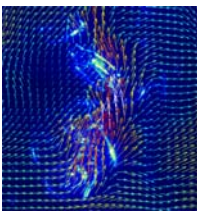


Oregon State University
Computer Graphics

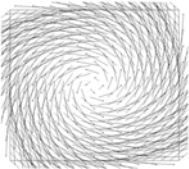
mjb - March 12, 2019

3

Examples



http://mathforum.org/mathimages/index.php/Vector_Fields



http://www.physicsdaily.com/physics/Image/Vectorfield_jaredwf.png



Chuck Evans



Oregon State University
Computer Graphics

mjb - March 12, 2019

4

Example – Hurricane Irma (September 2017)





Computer Graphics

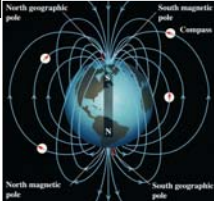
mjb - March 12, 2019

5

It Doesn't Always Have to be *Physically-Moving Flow*:
Things Like Magnetic Fields Count Too



This very cool magnetic wand toy consists of a small bar magnet on a universal joint, and is a good way to give a physical feel to magnetic fields. (You can get these from Edmund Scientific.)





http://www.physics.sjsu.edu/becke/physics51/mag_field.htm

Oregon State University
Computer Graphics

mjb - March 12, 2019

6

Two Types of Vector Visualization

1. What does the field itself look like?
2. What do things placed in the field do?



Oregon State University
Computer Graphics

mjb - March 12, 2019

What Does the Field Look Like? Glyphs

The most straightforward way to see what the field looks like is to place glyphs throughout the field.

The OpenDX AutoGlyph function gives you these type options:

- speedy
- spiffy
- standard
- 0.0
- 1.0
- 0.5
- text
- colored-text
- cube
- square
- arrow2D
- circle
- diamond
- needle
- needle2D
- rocket
- rocket2D
- sphere

msb - March 12, 2019

OpenDX Vector Glyphs

Needle

Arrow

Rocket

msb - March 12, 2019

A Cool Wind Speed and Direction Glyph

Wolfgang Bloem

msb - March 12, 2019

What Does the Field Look Like? Glyphs as Vector Clouds

In the same way that a point cloud was a simple way to visualize a scalar field, a Vector Cloud is a simple way to visualize a vector field. Go to selected points in the data volume, look up the velocity vector there (v_x, v_y, v_z), and draw an arrow centered around that point. The arrow's direction shows the direction of flow. The arrow's length shows the magnitude of the velocity field (i.e., its speed).

Nuance alert: the size of the arrow comes out in whatever units the velocity field is defined in, and might be too small to be seen or so large that it clutters the screen. You typically have to uniformly scale all the arrows to make the display

Bad

Better

msb - March 12, 2019

Drawing an Arrow

It's surprisingly involved to draw a good-looking 3D arrow. So, you've been given a C/C++ function to do it for you. Use it like this:

```
float tail[3], head[3];
...
// Center a 3D arrow at the point (x,y,z) indicating a
// velocity there of (vx,vy,vz);

tail[0] = x - Scale*vx/2.;
tail[1] = y - Scale*vy/2.;
tail[2] = z - Scale*vz/2.;

head[0] = x + Scale*vx/2.;
head[1] = y + Scale*vy/2.;
head[2] = z + Scale*vz/2.;

Arrow( tail, head );
```

Arrow() uses OpenGL lines, so the current line width and current color can be set as usual:

```
glLineWidth( w );           // number of pixels wide (floating point), ≥ 1.
glColor3f( r, g, b );       // red, green, blue in the range 0.-1.
Arrow( tail, head );
```

The arrows also get transformed along with everything else.

msb - March 12, 2019

Turning it Into a Scalar Problem: Magnitude Isosurfaces

msb - March 12, 2019

Particle Advection

Vector Clouds are OK, but we can do more. The next step is to think about what would happen if we released an imaginary massless ping-pong ball somewhere in the velocity field. Where would it go? This is called **Particle Advection**.

Oregon State University
Computer Graphics

mb - March 12, 2019

Taking a First Order Step

If we are at Point A, and the velocity field is as shown, how do we know where we end up next?

Easy, right? We look at the velocity field at Point A, and take a step in that direction, ending up at Point B.

This is called a **First Order Step**. It is also sometimes called **Euler's Method**.

Oregon State University
Computer Graphics

mb - March 12, 2019

Taking a First Order Step

```
void
Advect( float *x, float *y, float *z )
{
    xa = *x; ya = *y; za = *z;

    GetVelocity( xa, ya, za, &vxa, &vya, &vza );

    xb = xa + TimeStep*vxa;
    yb = ya + TimeStep*vya;
    zb = za + TimeStep*vza;

    *x = xb; *y = yb; *z = zb;
}
```

Oregon State University
Computer Graphics

mb - March 12, 2019

All is Not Right: the Spiral Problem

Assume we have a vector field that moves in a circle, that is:

$$v_x = -y$$

$$v_y = x$$

Now, take a First Order step, and you move like this:

Which puts you on a larger radius. The next First Order step puts you on an even larger radius. And so on, and so on. What should be circular motion has now become spiraling-out motion.

Oregon State University
Computer Graphics

mb - March 12, 2019

All is Not Right with First Order

Clearly something is not right. While we were taking that straight-line step, the velocity field was changing underneath us, and we weren't taking it into account.

Obviously, we could simply take smaller time steps, but that wouldn't solve the problem, just make it smaller. And, in the process, it could take lots longer to compute.

Oregon State University
Computer Graphics

mb - March 12, 2019

Taking a Second Order Step

Here's another approach. Let's assume that the field change during the time step is linear so that the average velocity vector during the step is the average of the velocity vector at A and the velocity vector at B. You do this by adding up the individual x, y, and z vector components and dividing by 2:

This is called a **Second Order Step**.

And, of course, you can continue this way even more.

Average the velocity vectors at A and C, take a step in that direction, and call it D.

Average the velocity vectors at A and D, take a step in that direction, and call it E.

Oregon State University
Computer Graphics

mb - March 12, 2019

Taking a Second Order Step

```

void
Advect( float *x, float *y, float *z )
{
    xa = *x;  ya = *y;  za = *z;

    GetVelocity( xa, ya, za, &vxa, &vya, &vza );

    xb = xa + TimeStep*vxa;
    yb = ya + TimeStep*vya;
    zb = za + TimeStep*vza;

    GetVelocity( xb, yb, zb, &vxb, &vyb, &vzb );

    vx = ( vxa + vxb ) / 2.;
    vy = ( vya + vzb ) / 2.;
    vz = ( vza + vzb ) / 2.;

    xc = xa + TimeStep*vx;
    yc = ya + TimeStep*vy;
    zc = za + TimeStep*vz;

    *x = xc;  *y = yc;  *z = zc;
}

```

First Order Code →

```

Advect( float *x, float *y, float *z )
{
    xa = *x;  ya = *y;  za = *z;

    GetVelocity( xa, ya, za, &vxa, &vya, &vza );

    xb = xa + TimeStep*vxa;
    yb = ya + TimeStep*vya;
    zb = za + TimeStep*vza;

    *x = xb;  *y = yb;  *z = zb;
}

```

Oregon State University Computer Graphics

The World's Largest Particle Advection Experiment

THEIR JOURNEY SO FAR

- 1. 29,000 ducks left China in 1992 and fell off boat in Pacific; 10,000 drift north
- 2. 19,000 go south and wash up in Australia, Indonesia and South America
- 3. 15 years and 17,000 miles later the bath toys are now on course for British beaches

2,000 miles

Oregon State University Computer Graphics

The World's Largest Particle Advection Experiment

Oregon State University Computer Graphics

Streamlines

Using particle advection, we could animate little ping-pong balls flying through the field. We can also take the particle advection idea and create other geometrizatons.

In this case, we are going to advect a particle and draw a line between its locations at successive time steps. This is called a **Streamline**. Because of the nature of particle advection, the tangent of the streamline curve always shows the direction of the velocity field there.

Oregon State University Computer Graphics

Streamlines

```

void
Streamline( float x, float y, float z )
{
    glLineWidth( 2. );
    glColor3f( ??, ??, ?? );
    glBegin( GL_LINE_STRIP );

    for( int i = 0; i < MAX_ITERATIONS; i++ )
    {
        if( x < Xmin || x > Xmax ) break;
        if( y < Ymin || y > Ymax ) break;
        if( z < Zmin || z > Zmax ) break;

        glVertex3f( x, y, z );

        GetVelocity( x, y, z, &vx, &vy, &vz );
        if( ||vx,vy,vz|| < SOME_TOLERANCE ) break;

        Advect( &x, &y, &z );
    }
    glEnd();
}

```

Three reasons to stop drawing the streamline

1. Boundary conditions (x, y, z outside the domain)
2. Velocity magnitude (||vx,vy,vz|| < SOME_TOLERANCE)
3. Iteration limit (i < MAX_ITERATIONS)

Oregon State University Computer Graphics

Streamlines and Particle Advection

National Center for Supercomputing Applications

Oregon State University Computer Graphics

Streaklines

25

So far, we have been treating the flow as if it was steady-state, that is, we are advancing the streamline using a snapshot of the vector field information. What if it's not steady-state?

If we follow the same procedure, but use a new time's vector field every time we advance the streamline, then we have what is known as a **streakline**.

The formal definition of a streakline is the *locus of fluid particles that have passed through a specific starting point*. Perhaps a more intuitive way to think about streaklines is thinking about what would happen if some colored dye was continuously injected into a flow field at a given point.

If the flow is steady-state, streamlines and streaklines are the same things.

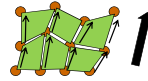


msb - March 12, 2019

Ribbon Traces

26

Envision a series of streamlines created from a row of starting points. But, every time a time step is taken, the corresponding points on the streamlines are connected and colored in. This is called a **Ribbon Trace**.



The big advantage of using a ribbon trace is that it can show *twisting motion* in the field (streamlines can't).

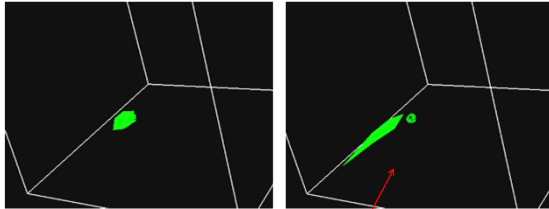


msb - March 12, 2019

Blob Tracing

27

Idea: start with a 3D shape and particle-advect each vertex. Then connect all the vertices with the same topology that was used for the original 3D object.



Stretched object shows that the field is accelerating.

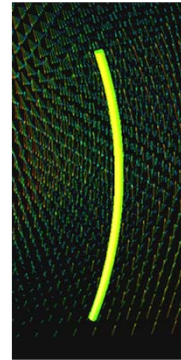
msb - March 12, 2019

Streamtubes

28

A **Streamtube** is like a streamline, but with a finite cross sectional area. (Which doesn't have to be a circle - "tube" is just what it is called.)

This makes your streamlines easier to see, and allows you to plot other information in color on the streamtube.



msb - March 12, 2019

Curl

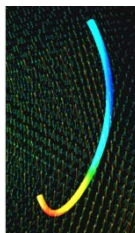
29

Curl and Divergence are referred to as "derived quantities" of a velocity field because they are not, generally, part of the original data, but are computed during the visualization.

The **Curl** tells you how much the field is curving. Think of it as the reciprocal of the radius of curvature of a streamline. The equation of the curl looks like this:

$$\nabla \times \vec{V} = \left(\frac{\partial V_z}{\partial y} - \frac{\partial V_y}{\partial z} \right) \hat{i} + \left(\frac{\partial V_x}{\partial z} - \frac{\partial V_z}{\partial x} \right) \hat{j} + \left(\frac{\partial V_y}{\partial x} - \frac{\partial V_x}{\partial y} \right) \hat{k}$$

This image shows the curl of a velocity field mapped as color to a streamtube.



msb - March 12, 2019

Divergence

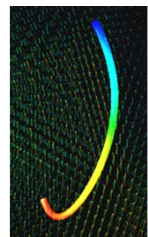
30

The **Divergence** tells you how much the field is spreading out or compressing. The equation of the divergence looks like this:

$$\nabla \cdot \vec{V} = \frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} + \frac{\partial V_z}{\partial z}$$

If the fluid that is flowing is incompressible, then the Conservation of Mass law tells us that the divergence is zero everywhere.

This image shows the divergence of a velocity field mapped as color to a streamtube.



msb - March 12, 2019

Curl Range Sliders

31

Idea: Show where the field has a particular range of curls.

Looking at just the large curls

Curl Range Slider

Oregon State University
Computer Graphics

mb - March 12, 2019

Magnitude Range Sliders

32

Idea: Show where the field has a particular range of magnitudes.

Looking at just the medium speeds

Flow Speed Range Slider

Oregon State University
Computer Graphics

mb - March 12, 2019

Line Integral Convolution

33

Line Integral Convolution (LIC) involves taking a white noise image and smearing it in the directions of the flow, in this case, a circular flow:

Mathematically, we create each pixel in the output image by following a streamline from that point (forwards and backwards) and performing a weighted average of all pixels that the streamline touches in the white noise image:

$$I'(x, y) = \frac{\sum_{i=-L}^L w(i) I(S(i))}{\sum_{i=-L}^L w(i)}$$

Where $S(i)$ is the streamline position " i " pixels away from the starting point, $I(\cdot)$ are the contents of the white noise image, $w(i)$ is the weight used for this pixel, and $I'(\cdot)$ is the resulting image.

Oregon State University
Computer Graphics

mb - March 12, 2019

3D Line Integral Convolution

34

You need to apply some amount of decimation, or you can't see into the volume

Vasu Lakshmanan

Oregon State University
Computer Graphics

mb - March 12, 2019

Peristalsis

35

As long as you're extruding some cross section to make a streamtube, you can also animate a moving bulge through it.

Oregon State University
Computer Graphics

John Datuin

mb - March 12, 2019

How Big Should the Time Step Be?

36

One of the trickiest parts of doing good particle advection for any reason is deciding how large to make the time step, Δt .

You could make it very, very tiny. That would give you good accuracy results, but poor interaction.

You could make it large. That would give you good interactivity, but at a cost of accuracy.

Clearly you need to find some way to *adapt* the time step to the situation.

One way is to think of the divergence and the curl as a way to measure how much the flow at a certain location is deviating from constant-speed straight-line motion. The larger the divergence and the curl, the smaller the time step should be.

Oregon State University
Computer Graphics

mb - March 12, 2019

How Big Should the Time Step Be?

37

Another way to do it is to check what would happen if two half-steps were taken instead of one whole step:

```
void
TakeOneStep( float Δt, float * x0, float * y0, float * z0 )
{
    float xw, yw, zw;
    xw = *x0; yw = *y0; zw = *z0; // one whole step
    Advect(Δt, &xw, &yw, &zw );

    float xh, yh, zh;
    xh = *x0; yh = *y0; zh = *z0; // two half steps
    Advect(Δt/2., &xh, &yh, &zh );
    Advect(Δt/2., &xh, &yh, &zh );

    if( (xh,yh,zh) is "close enough" to (xw,yw,zw) )
    {
        *x0 = xh; *y0 = yh; *z0 = zh;
        return;
    }

    TakeOneStep(Δt, / 2., x0, y0, z0 ); // re-try with a smaller time
step                                     // note: x0,y0,z0 are float pointers
                                         // x0, y0, z0 come back changed
    TakeOneStep(Δt, / 2., x0, y0, z0 );
}
```



March 12, 2019