A vector field: \( F(U) = V \)

- \( U \): field domain \((x,y)\) in 2D \((x,y,z)\) in 3D
- \( V \): vector \((u,v)\) or \((u,v,w)\)

- Like scalar fields, vectors are defined at discrete points

Vector Field Visualization Challenges

General Goal: Display the field’s directional information

Domain Specific: Detect certain features
- Vortex cores, Swirl

A good vector field visualization is difficult to get:
- Displaying a vector requires more visual attributes
  \((u,v,w)\): direction and magnitude
- Displaying a vector requires more screen space
  more than one pixel is required to display an arrow

\[ \Rightarrow \text{It becomes more challenging to display a dense vector field} \]

Vector Field Visualization Techniques

Local technique: Particle Advection -
- Display the trajectory of a particle starting from a particular location

Global technique: Hedgehogs, Line Integral Convolution,
- Texture Splats etc -
  Display the flow direction everywhere in the field

Local technique - Particle Tracing

Visualizing the flow directions by releasing particles and calculating a series of particle positions based on the vector field

The motion of particle: \( \frac{dx}{dt} = v(x) \)

\( x \): particle position \((x,y,z)\)

\( v(x) \): the vector (velocity) field

Use numerical integration to compute a new particle position

\[ x(t+dt) = x(t) + \int \text{Integration}( v(x(t)) \, dt ) \]
Numerical Integration

First Order Euler method:

\[ x(t+dt) = x(t) + v(x(t)) \cdot dt \]

- Not very accurate, but fast
- Other higher order methods are available: Runge-Kutta second and fourth order integration methods (more popular due to their accuracy)

Numerical Integration (2)

Second Runge-Kutta Method

\[
\begin{align*}
x(t+dt) &= x(t) + \frac{1}{2} \cdot (k_1 + k_2) \\
\frac{1}{2} &= \left[ v(x(t)) + v(x(t) + dt \cdot v(x(t)) \right]
\end{align*}
\]

Numerical Integration (3)

Standard Method: Runge-Kutta fourth order

\[
x(t+dt) = x(t) + \frac{1}{6} \left( k_1 + 2k_2 + 2k_3 + k_4 \right)
\]

-\[ k_1 = dt \cdot v(t); \]
-\[ k_2 = dt \cdot v(x(t) + k_1/2); \]
-\[ k_3 = dt \cdot v(x(t) + k_2/2); \]
-\[ k_4 = dt \cdot v(x(t) + k_3) \]

Streamlines

Curves that connect all the particle positions

Streamlines (cont’d)

- Displaying streamlines is a local technique because you can only visualize the flow directions initiated from one or a few particles
- When the number of streamlines is increased, the scene becomes cluttered
- You need to know where to drop the particle seeds
- Streamline computation is expensive

Pathlines, Timelines, and Streaklines

- Extension of streamlines for time-varying data

- Pathlines:

- Timelines:
Streaklines
- Continuously injecting a new particle at each time step, advecting all the existing particles and connect them together into a streakline

Global techniques
- Display the entire flow field in a single picture
- Minimum user intervention
- Example: Hedgehogs (global arrow plots)

Hedgehogs
- Dense 2D fields or 3D fields are difficult

Line Integral Convolution (LIC)

LIC Example

3D LIC
Comparison (LIC and Streamlines)

More global techniques
Texture Splits

Line bundles

LIC on 3D surfaces and volumes

Spot Noise