CIS 410/510: Advection
Project 3

• Assigned Thursday
• Due Jan 22, midnight (→ January 23rd, 6am)
• Worth 7% of your grade
• Provide:
  – Code skeleton online
  – Correct answers provided
• You upload
  – source code
• Wrong answers?: <1/2 credit
  – Differencer program available
Project 3 in a nutshell

- I give you a 2D data set.
- You will produce 3 images that are 500x500 pixels.
- The 2D data set will be mapped onto the pixels.
- Pixel (0,0): X=-9, Y=-9
- Pixel (499, 0): X=+9, Y=-9, pixel (0,499): X=-9, Y=+9
- Pixel (499,499): X=+9, Y=+9
Project 3 in a nutshell

- For each of the 250,000 pixels (500x500), you will apply 3 color maps to a scalar field
Advection
Advection

ad·vec·tion  (əd-vĕk′shən)

n.
1. The transfer of a property of the atmosphere, such as heat, cold, or humidity, by the horizontal movement of an air mass: Today's temperatures were higher due to the advection of warm air into the region.
2. The rate of change of an atmospheric property caused by the horizontal movement of air.
3. The horizontal movement of water, as in an ocean current.

[Latin advectiō, advectiōn-, act of conveying, from advectus, past participle of advehere, to carry to: ad-, ad- + vehere, to carry; see wegh- in Indo-European roots.]
Particle advection is the foundation to many visualization algorithms.

Advection will require us to evaluate velocity at arbitrary locations.
LERPing vectors

- LERP = Linear Interpolate
- Goal: interpolate vector between A and B.
- Consider vector X, where X = B - A
- A + t*(B-A) = A+t*X
- Takeaway:
  - you can LERP the components individually
  - the above provides motivation for why this works
Quiz Time: LERPing vectors

What is value of $F(10.3, 12)$?

Answer: $(-0.4, 2.9)$
Particle advection is the foundation to many visualization algorithms.
Overview of advection process

- Place a massless particle at a seed location
- Displace the particle according to the vector field
- Result is an “integral curve” corresponding to the trajectory the particle travels
- Math gets tricky

What would be the difference between a massless and “mass-ful” particle?
Formal definition for particle advection

• Output is an integral curve, $S$, which follows trajectory of the advection
• $S(t) = \text{position of curve at time } t$
  – $S(t_0) = p_0$
    • $t_0$: initial time
    • $p_0$: initial position
  – $S'(t) = v(t, S(t))$
    • $v(t, p)$: velocity at time $t$ and position $p$
    • $S'(t)$: derivative of the integral curve at time $t$

This is an ordinary differential equation (ODE).
The integral curve for particle advection is calculated iteratively

\[ S(t_0) = p_0 \]

while (ShouldContinue())
{
\[ S(t_i) = \text{AdvanceStep}(S(t_{i-1})) \]
}
Integral curve calculation with a fixed number of steps

\[ S_0 = P_0 \]

for (int i = 1 ; i < numSteps ; i++)
{
    \[ S_i = \text{AdvanceStep}(S_{(i-1)}) \]
}

How to do an advance?
AdvanceStep goal:
to calculate $S_i$ from $S_{(i-1)}$

$S_1 = \text{AdvanceStep}(S_0)$
$S_2 = \text{AdvanceStep}(S_1)$
$S_3 = \text{AdvanceStep}(S_2)$
$S_4 = \text{AdvanceStep}(S_3)$
$S_5 = \text{AdvanceStep}(S_4)$

This picture is misleading:
steps are typically much smaller.
AdvanceStep Overview

• Think of AdvanceStep as a function:
  – Input arguments:
    • $S_{(i-1)}$: Position, time
  – Output arguments:
    • $S_i$: New position, new time (later than input time)
  – Optional input arguments:
    • More parameters to control the stepping process.
AdvanceStep Overview

• Different numerical methods for implementing AdvanceStep:
  – Simplest version: Euler step
  – Most common: Runge-Kutta-4 (RK4)
  – Several others as well
Euler Method

• Most basic method for solving an ODE

• Idea:
  – First, choose step size: $h$.
  – Second, $\text{AdvanceStep}(p_i, t_i)$ returns:
    • New position: $p_{i+1} = p_i + h \cdot v(t_i, p_i)$
    • New time: $t_{i+1} = t_i + h$
Quiz Time: Euler Method

• Euler Method:
  – New position: \( p_{i+1} = p_i + h \cdot v(t_i, p_i) \)
  – New time: \( t_{i+1} = t_i + h \)

• Let \( h = 0.01 \text{s} \)
• Let \( p_0 = (1,2,1) \)
• Let \( t_0 = 0 \text{s} \)
• Let \( v(p_0, t_0) = (-1, -2, -1) \)

• What is \( (p_1, t_1) \) if you are using the Euler method?

Answer: ((0.99, 1.98, 0.99), 0.01)
Quiz Time #2: Euler Method

• Euler Method:
  – New position: $p_{i+1} = p_i + h \cdot v(t_i, p_i)$
  – New time: $t_{i+1} = t_i + h$

• Let $h = 0.01s$

• Let $p_1 = (0.99, 1.98, 0.99)$

• Let $t_1 = 0.01s$

• Let $v(p_1, t_1) = (1, 2, 1)$

• What is $(p_2, t_2)$ if you are using the Euler method?

Answer: $((1, 2, 1), 0.02)$
Quiz Time #3: Euler Method

- Euler Method:
  - New position: \( p_{i+1} = p_i + h \cdot v(t_i, p_i) \)
  - New time: \( t_{i+1} = t_i + h \)

- Let \( h = 0.01 \) s
- Let \( p_2 = (1, 2, 1) \)
- Let \( t_2 = 0.02 \) s
- Let \( v(p_2, t_2) = (1, 0, 0) \)
- What is \( (p_3, t_3) \) if you are using the Euler method?

Answer: \( ((1.01, 2, 1), 0.03) \)
Euler Method: Pros and Cons

• Pros:
  – Simple to implement
  – Computationally very efficient

• (quiz) Cons:
  – Prone to inaccuracy

• Above statements are an oversimplification:
  – Can be very accurate with small steps size, but then also very inefficient.
  – Can be very fast, but then also inaccurate.
Quiz Time

• You want to perform a particle advection.
• What inputs do you need?
  – Velocity field
  – Step size
  – Termination criteria / # of steps
  – Initial seed position / time
Quiz Time

• Write down pseudo-code to do advection with an Euler step:
  – Initial seed location: (0,0,0)
  – Initial seed time: 0s
  – Step size = 0.01s
  – Velocity field: v
  – Termination criteria: advance 0.1s (take 10 steps)
  – Function to evaluate velocity:
    • EvaluateVelocity(position, time)
Quiz Time (answer)

\[
S[0] = (0,0,0);
\]
\[
time = 0;
\]
\[
\text{for (int } i = 0 ; i < 10 ; i++)
\]
\[
\{
\text{  } \quad S[i+1] = S[i]+h*EvaluateVelocity(S[i], time);
\text{  } \quad time += h;
\}
\]
Runge-Kutta Method (RK4)

• Most common method for solving an ODE
• Definition:
  – First, choose step size, h.
  – Second, AdvanceStep\((p_i, t_i)\) returns:
    • New position: \(p_{i+1} = p_i + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4)\)
      – \(k_1 = v(t_i, p_i)\)
      – \(k_2 = v(t_i + \frac{h}{2}, \ p_i + \frac{h}{2} * k_1)\)
      – \(k_3 = v(t_i + \frac{h}{2}, \ p_i + \frac{h}{2} * k_2)\)
      – \(k_4 = v(t_i + h, \ p_i + h * k_3)\)
    • New time: \(t_{i+1} = t_i + h\)
Physical interpretation of RK4

• New position: $p_{i+1} = p_i + (1/6)h(k_1 + 2k_2 + 2k_3 + k_4)$
  
  $k_1 = v(t_i, p_i)$
  $k_2 = v(t_i + h/2, p_i + h/2*k_1)$
  $k_3 = v(t_i + h/2, p_i + h/2*k_2)$
  $k_4 = v(t_i + h, p_i + h*k_3)$

Evaluate 4 velocities, use combination to calculate $p_{i+1}$.
Quiz time: Runge-Kutta 4

• Just kidding
Runge-Kutta vs Euler

• Euler Method:
  – “1st order numerical method for solving ordinary differential equations (ODEs)”
    \( \rightarrow \) error per step is \( O(h^2) \), total error is \( O(h) \)

• Runge-Kutta:
  – “4th order numerical method for solving ODEs”
    \( \rightarrow \) error per step is \( O(h^5) \), total error is \( O(h^4) \)

• Oversimplification: all of RK4’s “look-aheads” prevents you from stepping too far into the wrong place.

\[ h \text{ is small, so } h^x \text{ is smaller still} \]
Quiz Time: RK4 vs Euler

• Let $h=0.01\text{s}$
  – How many velocity field evaluations for RK4 to advance 1s?
  – How many velocity field evaluations for Euler to advance 1s?

400 for RK4, 100 for Euler
Quiz Time: RK4 vs Euler

• Let $h=0.01$ for an RK4
• What $h$ for Euler to achieve similar accuracy?

• Error for RK4 is: $O(1e^{-2^4}) = O(1e^{-8})$
• Error for Euler is: $O(h^2)$ $\implies h=1e^{-4}$

• What is the ratio of velocity evaluations for Euler to achieve same accuracy?

25X more for Euler
Other ODE solvers

• Adams/Bashforth, Dormand/Prince are also used

• Idea: adaptive step size.
  – If the field is homogeneous, then take a bigger step size (bigger h)
  – If the field is heterogeneous, then take a smaller step size

• Quiz: why would you want to do this?

  Answer: great accuracy at reduced computational cost
Termination criteria

- Time
Termination criteria

- Time
- Distance
Termination criteria

• Time
• Distance
• Number of steps
  – Same as time?
• Other advanced criteria, based on particle advection purpose
Other reasons for advection termination

- Exit the volume
Other reasons for advection termination

• Advect into a sink
Steady versus Unsteady State

- Unsteady state: the velocity field evolves over time
- Steady state: the velocity field has reached steady state and remains unchanged as time evolves
Most common particle advection technique: streamlines and pathlines

• Idea: plot the entire trajectory of the particle all at one time.

Streamlines in the “fish tank”
Streamline vs Pathlines

- Streamlines: plot trajectory of a particle from a steady state field
- Pathlines: plot trajectory of a particle from an unsteady state field

Quiz: most common configuration?
- Neither!!
  - Pretend an unsteady state field is actually steady state and plot streamlines from one moment in time.

Quiz: why would anyone want to do this?
- (answer: performance)
Pathlines in the real world
Lots more to talk about

• How do we pragmatically deal with unsteady state flow (velocities that change over time)?
• More operations based on particle advection
• Stability of results
Dealing with steady state velocities

- **Euler Method (unsteady):**
  - New position: \( p_{i+1} = p_i + h \cdot v(t_i, p_i) \)
  - New time: \( t_{i+1} = t_i + h \)

- **Euler Method (steady):**
  - New position: \( p_{i+1} = p_i + h \cdot v(t_0, p_i) \)
  - New time: \( t_{i+1} = t_i + h \)
Unsteady vs Steady: RK4

- Unsteady:
  - New position: \( p_{i+1} = p_i + (1/6)h(k_1+2k_2+2k_3+k_4) \)
    - \( k_1 = v(t_i, p_i) \)
    - \( k_2 = v(t_i + h/2, p_i + h/2*k_1) \)
    - \( k_3 = v(t_i + h/2, p_i + h/2*k_2) \)
    - \( k_4 = v(t_i + h, p_i + h*k_3) \)
    - New time: \( t_{i+1} = t_i + h \)
Unsteady vs Steady: RK4

• Steady:
  – New position: \( p_{i+1} = p_i + (1/6)*h*(k_1 + 2k_2 + 2k_3 + k_4) \)
  • \( k_1 = v(t_0, p_i) \)
  • \( k_2 = v(t_0, p_i + h/2*k_1) \)
  • \( k_3 = v(t_0, p_i + h/2*k_2) \)
  • \( k_4 = v(t_0, p_i + h*k_3) \)
  • New time: \( t_{i+1} = t_i + h \)
Project 4

- Assigned today, due Jan 30\textsuperscript{th} (→ 6am Jan 31\textsuperscript{st})
- Worth 7% of your grade
- Provided for you:
  - Code skeleton online
  - Correct answers provided
- What to upload to Canvas:
  - source code
  - screenshot with:
    - text output on screen
    - image of results
Project 4 in a nutshell

- Do some vector LERPing
- Do particle advection with Euler steps
- Examine results
Project 4 in a nutshell

• Implement 3 methods:
  – EvaluateVectorFieldAtLocation
    • LERP vector field. (Reuse code from before, but now multiple components)
  – AdvectWithEulerStep
    • You know how to do this
  – CalculateArcLength
    • What is the total length of the resulting trajectory?
Project 4G (510 only!!) in a nutshell

• Implement Runge-Kutta 4
• Assess the quality of RK4 vs Euler
• Open ended project
  – I don’t tell you how to do this assessment
    • You will need to figure it out
    • Multiple right answers
• Deliverable: short report (~1 page) describing your conclusions and methodology
  – Pretend that your boss wants to know which method to use and you have to convince them which one is the best and why
• Not everyone will receive full credit
Additional Particle Advection Techniques

• This content courtesy of Christoph Garth, Kaiserslautern University.
Streamline, Pathline, Timeline, Streakline,

• Streamline: steady state velocity, plot trajectory of a curve
• Pathline: unsteady state velocity, plot trajectory of a curve
• Timeline: start with a line, advect that line and plot the line’s position at some future time
Advect a surface and see where it goes – “A sheet blowing in the wind”
Streamline, Pathline, Timeline, Streakline

- **Streamline**: steady state velocity, plot trajectory of a curve
- **Pathline**: unsteady state velocity, plot trajectory of a curve
- **Timeline**: start with a line, advect that line and plot the line’s position at some future time
- **Streakline**: unsteady state, introduce new particles at a location continuously
Streaklines

Will do an example on the board
Streaklines in real life
• Stream surface:
  – Start with a seeding curve
  – Advect the curve to form a surface

\[ \frac{d}{dt} S(s, t) = \vec{v}(t, S(s, t)) \]

\[ S(s, 0) := C(s) \]
• Stream surface:
  – Start with a seeding curve
  – Advect the curve to form a surface
Stream Surface Computation

- Skeleton from Integral Curves + Timelines
Stream Surface Computation

- Skeleton from Integral Curves + Timelines
- Triangulation

Stream Surface Example
Stream Surface Example
Stream Surface Example #2

Vortex system behind ellipsoid
Lagrangian Methods

• Visualize manifolds of maximal stretching in a flow, as indicated by dense particles

• Finite-Time Lyapunov Exponent (FTLE)

\[ \sigma_{\Delta t}(t, x) := \frac{1}{\Delta t} \ln \sqrt{\lambda_{\text{max}} (D_x \phi_{\Delta t}(t, x))} \]
Lagrangian Methods

• Visualize manifolds of maximal stretching in a flow, as indicated by dense particles

  – forward in time: \textbf{FTLE}^+ indicates divergence
  – Backward in time: \textbf{FTLE}^+ indicates convergence