Review: Vertex Shaders
- replace model/view transformation, lighting, perspective projection
- a little assembly-style program is executed on every individual vertex independently
  - it sees:
    - vertex attributes that change per vertex:
      - position, color, texture coordinates...
    - registers that are constant for all vertices (changes are expensive):
      - matrices, light position and color, ...
    - temporary registers
    - output registers for position, color, tex coords...

Review: Compositing
- specifically capacite with alpha channel: (r,g,b,a)
  - (a = 0) opaque, (a = 1) transparent, (a = 0) transparent
- A over B
  - C = αA + (1−α)B
- but what if B is also partially transparent?
  - C = αA + (1−α)B + γβA' + (1−γ−β)B'
  - γ + β + (1−α) = 1
    - 3 multiplies, different equations for alpha vs. RGB
- premultiplying by alpha
  - C' = γC + B' = βB + A' + (1−γ−β)B'
  - C = γC + B' = βB + A' + (1−γ−β)B'
  - 1 multiply to find C, same equations for alpha and RGB

Correction/Review: Premultiplying Colors
- so far rendering pipeline as a specific set of stages with fixed functionality
- modern graphics hardware much more flexible
  - programmable "vertex shaders" replace several geometry processing stages
  - programmable "fragment/pixel shaders" replace texture mapping stage
  - hardware with these features now called Graphics Processing Unit (GPU)
- program shading hardware with assembly language analog, or high level shading language

Review: Rendering Pipeline
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: GPGPU Programming
- a spline is a parametric curve defined by control points
  - term "spline" dates from engineering drawing, where a spline was a piece of flexible wood
  - control points are adjusted by the user to control shape of curve

Review: Splines
- project 4 grading slots signup
  - Wed Apr 18 10-12
  - Wed Apr 18 4-6
  - Fri Apr 20 10-1

Old News
- extra TA office hours in lab for hw/project Q&A
  - next week: Thu 4-6, Fri 10-2
- last week of classes:
  - Mon 2-5, Tue 4-6, Wed 2-4, Thu 4-6, Fri 9-6
- final review Q&A session
  - Mon Apr 16 10-12
- reminder: no lecture/labs Fri 4/6, Mon 4/9

Curves
- parametric form for a line:
  - $x = x_0 + t(1-t)x_1$
  - $y = y_0 + t(1-t)y_1$
  - $z = z_0 + t(1-t)z_1$
- x, y and z are each given by an equation that involves:
  - parameter t
- some user specified control points, $x_0$ and $x_1$
- this is an example of a parametric curve

Review: Skinning Vertex Shader
- arm example:
  - M1: matrix for upper arm
  - M2: matrix for lower arm
  - upper arm: weight for M1=1
  - weight for M2=0
  - lower arm: weight for M1=0
  - weight for M2=1

Review: Rendering Pipeline
- so far rendering pipeline as a specific set of stages with fixed functionality
- modern graphics hardware much more flexible
  - programmable "vertex shaders" replace several geometry processing stages
  - programmable "fragment/pixel shaders" replace texture mapping stage
  - hardware with these features now called Graphics Processing Unit (GPU)
- program shading hardware with assembly language analog, or high level shading language

Review: GPGPU Programming
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Splines
- a spline is a parametric curve defined by control points
  - term "spline" dates from engineering drawing, where a spline was a piece of flexible wood
  - control points are adjusted by the user to control shape of curve

Splines
- for any answer involving calculation, although it’s fine to show your work in analytical form, the final answer should be expressed as a number to two decimal places.

Reminder for H4
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Rendering Pipeline
- so far rendering pipeline as a specific set of stages with fixed functionality
- modern graphics hardware much more flexible
  - programmable "vertex shaders" replace several geometry processing stages
  - programmable "fragment/pixel shaders" replace texture mapping stage
  - hardware with these features now called Graphics Processing Unit (GPU)
- program shading hardware with assembly language analog, or high level shading language

Review: GPGPU Programming
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Splines
- a spline is a parametric curve defined by control points
  - term "spline" dates from engineering drawing, where a spline was a piece of flexible wood
  - control points are adjusted by the user to control shape of curve

Splines
- for any answer involving calculation, although it’s fine to show your work in analytical form, the final answer should be expressed as a number to two decimal places.

Reminder for H4
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Rendering Pipeline
- so far rendering pipeline as a specific set of stages with fixed functionality
- modern graphics hardware much more flexible
  - programmable "vertex shaders" replace several geometry processing stages
  - programmable "fragment/pixel shaders" replace texture mapping stage
  - hardware with these features now called Graphics Processing Unit (GPU)
- program shading hardware with assembly language analog, or high level shading language

Review: GPGPU Programming
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Splines
- a spline is a parametric curve defined by control points
  - term "spline" dates from engineering drawing, where a spline was a piece of flexible wood
  - control points are adjusted by the user to control shape of curve

Splines
- for any answer involving calculation, although it’s fine to show your work in analytical form, the final answer should be expressed as a number to two decimal places.

Reminder for H4
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Rendering Pipeline
- so far rendering pipeline as a specific set of stages with fixed functionality
- modern graphics hardware much more flexible
  - programmable "vertex shaders" replace several geometry processing stages
  - programmable "fragment/pixel shaders" replace texture mapping stage
  - hardware with these features now called Graphics Processing Unit (GPU)
- program shading hardware with assembly language analog, or high level shading language

Review: GPGPU Programming
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Splines
- a spline is a parametric curve defined by control points
  - term "spline" dates from engineering drawing, where a spline was a piece of flexible wood
  - control points are adjusted by the user to control shape of curve

Splines
- for any answer involving calculation, although it’s fine to show your work in analytical form, the final answer should be expressed as a number to two decimal places.

Reminder for H4
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Rendering Pipeline
- so far rendering pipeline as a specific set of stages with fixed functionality
- modern graphics hardware much more flexible
  - programmable "vertex shaders" replace several geometry processing stages
  - programmable "fragment/pixel shaders" replace texture mapping stage
  - hardware with these features now called Graphics Processing Unit (GPU)
- program shading hardware with assembly language analog, or high level shading language

Review: GPGPU Programming
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Splines
- a spline is a parametric curve defined by control points
  - term "spline" dates from engineering drawing, where a spline was a piece of flexible wood
  - control points are adjusted by the user to control shape of curve

Splines
- for any answer involving calculation, although it’s fine to show your work in analytical form, the final answer should be expressed as a number to two decimal places.

Reminder for H4
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Rendering Pipeline
- so far rendering pipeline as a specific set of stages with fixed functionality
- modern graphics hardware much more flexible
  - programmable "vertex shaders" replace several geometry processing stages
  - programmable "fragment/pixel shaders" replace texture mapping stage
  - hardware with these features now called Graphics Processing Unit (GPU)
- program shading hardware with assembly language analog, or high level shading language

Review: GPGPU Programming
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Splines
- a spline is a parametric curve defined by control points
  - term "spline" dates from engineering drawing, where a spline was a piece of flexible wood
  - control points are adjusted by the user to control shape of curve

Splines
- for any answer involving calculation, although it’s fine to show your work in analytical form, the final answer should be expressed as a number to two decimal places.

Reminder for H4
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Rendering Pipeline
- so far rendering pipeline as a specific set of stages with fixed functionality
- modern graphics hardware much more flexible
  - programmable "vertex shaders" replace several geometry processing stages
  - programmable "fragment/pixel shaders" replace texture mapping stage
  - hardware with these features now called Graphics Processing Unit (GPU)
- program shading hardware with assembly language analog, or high level shading language

Review: GPGPU Programming
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Splines
- a spline is a parametric curve defined by control points
  - term "spline" dates from engineering drawing, where a spline was a piece of flexible wood
  - control points are adjusted by the user to control shape of curve

Splines
- for any answer involving calculation, although it’s fine to show your work in analytical form, the final answer should be expressed as a number to two decimal places.

Reminder for H4
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Rendering Pipeline
- so far rendering pipeline as a specific set of stages with fixed functionality
- modern graphics hardware much more flexible
  - programmable "vertex shaders" replace several geometry processing stages
  - programmable "fragment/pixel shaders" replace texture mapping stage
  - hardware with these features now called Graphics Processing Unit (GPU)
- program shading hardware with assembly language analog, or high level shading language

Review: GPGPU Programming
- general purpose GPU
- use graphics card as SIMD parallel processor
- textures as arrays
- computation: render large quadrilateral
- multiple rendering passes

Review: Splines
- a spline is a parametric curve defined by control points
  - term "spline" dates from engineering drawing, where a spline was a piece of flexible wood
  - control points are adjusted by the user to control shape of curve

Splines
- for any answer involving calculation, although it’s fine to show your work in analytical form, the final answer should be expressed as a number to two decimal places.

Reminder for H4
Splines - History
- draftsman used 'ducks' and strips of wood (splines) to draw curves
- wood splines have second-order continuity, pass through the control points
- ducks trace out curve

Bézier Curves
- similar to Hermite, but more intuitive definition of endpoint derivatives
- four control points, two of which are knots

Bézier Curves
- curve will always remain within convex hull (bounding region) defined by control points
- interpolate between first, last control points
- 1st point’s tangent along line joining 1st, 2nd pts
- 4th point’s tangent along line joining 3rd, 4th pts

Bézier Curves
- derivative values of Bézier curve at knots dependent on adjacent points
- \( Vp_T = 3(p_2 - p_1) \)
- \( Vp_B = 3(p_4 - p_3) \)

Hermite Spline
- hermite spline is curve for which user provides:
- endpoints of curve
- parametric derivatives of curve at endpoints
- parametric derivatives are \( dx/dt, dy/dt, dz/dt \)
- more derivatives would be required for higher order curves

Bézier Blending Functions
- look at blending functions
- family of polynomials called order-3 Bernstein polynomials
- \( C(3, k) = (1-t)^{3-k} t^k; 0 \leq k \leq 3 \)
- all positive in interval \([0,1]\)
- sum is equal to 1

Comparing Hermite and Bézier
- every point on curve is linear combination of control points
- weights of combination are all positive
- sum of weights is 1
- therefore, curve is a convex combination of the control points

Rendering Bezier Curves: Simple
- evaluate curve at fixed set of parameter values, join points with straight lines
- advantage: very simple
- disadvantages:
  - expensive to evaluate the curve at many points
  - no easy way of knowing how fine to sample points, and maybe sampling rate must be different along curve
  - no easy way to adapt: hard to measure deviation of line segment from exact curve

Rendering Beziers: Subdivision
- a cubic Bézier curve can be broken into two shorter cubic Bézier curves that exactly cover original curve
- suggests a rendering algorithm:
  - keep breaking curve into sub-curves
  - stop when control points of each sub-curve are nearly collinear
  - draw the control polygon: polygon formed by control points

Sub-Dividing Bezier Curves
- step 1: find the midpoints of the lines joining the original control vertices. Call them \( M_0, M_1, M_2, M_3 \)

Sub-Dividing Bezier Curves
- step 2: find the midpoints of the lines joining \( M_{01}, M_{12} \) and \( M_{12}, M_{23} \). Call them \( M_{012}, M_{123} \)

Sub-Dividing Bezier Curves
- step 3: find the midpoint of the line joining \( M_{012}, M_{123} \). Call it \( M_{0123} \)
Sub-Dividing Bezier Curves
- curve $P_0, M_{01}, M_{12}, M_{23}$, exactly follows original from $t=0$ to $t=0.5$
- curve $M_{012}, M_{123}, M_{23}$, exactly follows original from $t=0.5$ to $t=1$

B-Spline Curve
- start with a sequence of control points
- select four from middle of sequence ($P_{i-2}, P_{i-1}, P_i, P_{i+1}$)
  - doesn't interpolate (touch) any of them but approximates the going through $P_{i-1}$ and $P_i$

Longer Curves
- a single cubic Bezier/Hermite curve can only capture a small class of curves
  - at most 2 inflection points
  - one solution is to raise the degree
    - allows more control, at the expense of more control points and higher degree polynomials
    - control is not local, one control point influences entire curve
  - better solution is to join pieces of cubic curve together into piecewise cubic curves
  - entire curve can be broken into pieces, each of which is cubic
  - local control, each control point only influences a limited part of the curve
  - interaction and design is much easier

Piecewise Bezier: Continuity Problems
- when two curves joined, typically want some degree of continuity across knot boundary
  - $C_0$, "C-zero", point-wise continuous, curves share same point where they join
  - $C_1$, "C-one", continuous derivatives
  - $C_2$, "C-two", continuous second derivatives

Continuity
- when two curves joined, typically want some degree of continuity across knot boundary
  - $C_0$, "C-zero", point-wise continuous, curves share same point where they join
  - $C_1$, "C-one", continuous derivatives
  - $C_2$, "C-two", continuous second derivatives
  - when two curves joined, typically want some degree of continuity across knot boundary

Geometric Continuity
- derivative continuity is important for animation
  - if object moves along curve with constant parametric speed, should be no sudden jump at knots
  - for other applications, tangent continuity suffices
    - requires that the tangents point in the same direction
    - referred to as $C^1$ geometric continuity
    - curves could be made $C^2$ with a re-parameterization
  - geometric version of $C^1$ is $G^2$, based on curves having the same radius of curvature across the knot

Sub-Dividing Bezier Curves
- continue process to create smooth curve

Continuity
- when two curves joined, typically want some degree of continuity across knot boundary
  - $C_0$, "C-zero", point-wise continuous, curves share same point where they join
  - $C_1$, "C-one", continuous derivatives
  - $C_2$, "C-two", continuous second derivatives

Geometric Continuity
- derivative continuity is important for animation
  - if object moves along curve with constant parametric speed, should be no sudden jump at knots
  - for other applications, tangent continuity suffices
    - requires that the tangents point in the same direction
    - referred to as $C^1$ geometric continuity
    - curves could be made $C^2$ with a re-parameterization
  - geometric version of $C^1$ is $G^2$, based on curves having the same radius of curvature across the knot

de Casteljau’s Algorithm
- can find the point on a Bezier curve for any parameter value $t$ with similar algorithm
  - for $t=0.25$, instead of taking midpoints take points 0.25 of the way

B-Spline
- by far the most popular spline used
  - $C_0$, $C_1$, and $C_2$ continuous

Achieving Continuity
- Hermite curves
  - user specifies derivatives, so $C^1$ by sharing points and derivatives across knot
- Bezier curves
  - they interpolate endpoints, so $C^2$ by sharing control pts
  - introduce additional constraints to get $C^1$ parametric derivative is a constant multiple of vector joining first/last 2
    - so $C^1$ achieved by setting $P_{i-1}, P_i, P_{i+1}$, and making $P_{i-1}$ and $P_i$ to be collinear, with $P_{i-1}P_i = P_iP_{i+1}$
    - $C^2$ comes from further constraints on $P_{i-1}$ and $P_{i+1}$
  - $C^2$ comes from further constraints on $P_{i-1}$ and $P_{i+1}$