Splines

- Smooth piecewise curves
- Mostly cubic polynomials
- Parametric curves

Control points

Knots

Some interpolate (= pass through) the control points,
Others do not.
Hierarchical Bicubic B-Splines
Hermite splines

Define end points and gradients.

Continuity $C^0$ end points match
$C^1$ 1st derivatives match
$C^2$ 2nd derivatives match
Splines are parametric polynomials

\[ y = a + bu + cu^2 + du^3 \]

Splines are arranged to have continuity at the joins (knots)

\( C^0 \): The curves touch at the join point.
\( C^1 \): The curves also share a common tangent direction at the join point.
\( C^2 \): The curves also share a common center of curvature at the join point.
Hermites

- C1 continuous
- Matching end points, matching gradients
  \[ y = a + bu + cu^2 + du^3 \]
  \[ dy/du = b + 2cu + 3du^2 \]

- Example: \[ u_1 = 0, y_1 = 2.0, dy_1/du_1 = 1 \]
  \[ u_2 = 1, y_2 = 3.0, dy_2/du_2 = -2 \]

Find a, b, c, d to define the curve
Write out the equation
General derivation

- For any \((y_1,g_1) \ (y_2,g_2)\)
In C++

```cpp
_ay = y1;
_by = dy1;

_cy = 3.0f*(y2 - y1) - 2.0*dy1 - dy2;
_dy = 2.0f*(y1 - y2) + dy1 + dy2;
```
2D Hermite:

- Just do it twice, y is a function of u.
- x is a function of u.

To get a 3D Hermite, just do it 3 times for x, for y and for z.
- x,y,z all are functions of a common u.
How to calculate normals

For the start just define a normal at right angles to the starting forward vector. E.g. start track horizontally F(0,0,1), S(1,0,0), T(010);

Do this by defining the gradients.

dx/du = 0; dy/dz = 1; dz/du = 1;

For the next segment, of the 100 segments.

Take the cross product of the new forward vector with the old top vector (result a new sideways vector).

Take the cross product of the new sideways vector with the new forward vector to get the new top vector.
Construct box

- Use the vectors.
- Center +/- halfwidth to get the sides.
- Center – thickness to define the bottom.
- Use the normals in the rendering loop.
To make the view follow the object

- Use `gluLookAt(ex, ey, ez, atx, aty, atz, upx, upy, upz);
- The *at* position, should be the center of the ball.
- The *up* direction should use the top vector
- Case 1: Wingman, View *eye* position using the sideways vector
- Case 2: Above and behind. Use the up vector and the forward vector to define the *eye* position.
typedef float Point3f[3];
Point3f *BoxTopL, *BoxTopR, *BoxBotL, *BoxBotR;

BoxTopL = new Point3f[len];
BoxTopR = new Point3f[len];
BoxBotL = new Point3f[len];
BoxBotR = new Point3f[len];

Draw the top, the bottom, the left, the right.
Natural Cubic splines

- If we have $n$ control points and $n$ splines we can make a set of curves.
- Such that end points match (2 dof).
- First derivatives match (1 dof).
- Second derivatives match (2 dof). = angular acceleration

- But we lose local control.
- Solve a system of equations.