Topic 6:

3D Curves

- Intro to curve interpolation & approximation
- Polynomial interpolation
- Bézier curves
- Cardinal splines

Interactive Design of Curves

Goal: Expand the capabilities of shapes beyond lines and conics, simple analytic functions and to allow design constraints.

Design Issues:

- Continuity (smoothness)
- Control (local vs. global)
- Interpolation vs. approximation of constraints
- Other geometric properties (planarity, tangent/curvature control)
- Efficient analytic representation

Parametric Polynomial Curves

Recall a linear curve (line) is:

$p(t) = a_1t+a_0$

A cubic curve is similarly:

$$x(t) = a_3t^3 + a_2t^2 + a_1t + a_0$$
$$y(t) = b_3t^3 + b_2t^2 + b_1t + b_0$$
$$z(t) = c_3t^3 + c_2t^2 + c_1t + c_0$$

…or $p(t) = d_3t^3+d_2t^2+d_1t+d_0$, where $d_i = [a_i, b_i, c_i]^T$

Cubics are commonly used in graphics because curves of lower order commonly have too little flexibility (only planar, no curvature control), while curves of higher order are unnecessarily complex and make it easy to introduce undesired wiggles.

Designing Polynomial Curves from constraints

$p(t) = TA$, where T is powers of t. for a cubic $T=[t^3\ t^2\ t^1\ 1]$.

Written with geometric constraints $p(t) = TMG$, where M is the **Basis matrix** of a design curve and G the specific design constraints.

An example of constraints for a cubic Hermite for eg. are end points and end tangents. i.e. P_1,R_1 at t=0 and P_4,R_4 at t=1. Plugging these constraints into $p(t) = TA$ we get.

B

$p(0) = P_1 = [\ 0\ 0\ 0\ 1\ ]\ A_h$
$p(1) = P_4 = [\ 1\ 1\ 1\ 1\ ]\ A_h$
$p'(0)= R_1 = [\ 0\ 0\ 1\ 0\ ]\ A_h$ => $G=BA$, $A=MG$ => $M=B^{-1}$
$p'(1)= R_4 = [\ 3\ 2\ 1\ 0\ ]\ A_h$

Hermite Basis Matrix

An example of constraints for a cubic Hermite for eg. are end points and end tangents. i.e. P_1,R_1 at t=0 and P_4,R_4 at t=1. Plugging these constraints into $p(t) = TA$ we get.

B

$p(0) = P_1 = [\ 0\ 0\ 0\ 1\ ]\ A_h$
$p(1) = P_4 = [\ 1\ 1\ 1\ 1\ ]\ A_h$
$p'(0)= R_1 = [\ 0\ 0\ 1\ 0\ ]\ A_h$ => $G=BA$, $A=MG$ => $M=B^{-1}$
$p'(1)= R_4 = [\ 3\ 2\ 1\ 0\ ]\ A_h$

Hermite Basis Matrix

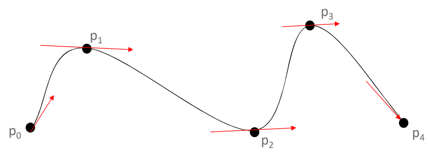
$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1}$$

$= M_{hermite}$

$$\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Catmull-Romm splines using Hermite curves

Catmull-Romm Interpolation



Pick tangents based on a factor k (1/2 for eg.) of the vector between neighbor points.

$p'_i = k*(p_{i+1} - p_{i-1})$.

For the end-points there is only one neighbor:

$p'_0 = k*(p_1 - p_0)$.
$p'_n = k*(p_n - p_{n-1})$.

Bezier Basis Functions

[-1 3 -3 1]
[3 -6 3 0]
[-3 3 0 0]
[1 0 0 0]

The columns of the Basis Matrix form Basis Functions such that:
$p(t)= f_1(t)P_1 + f_2(t)P_2 + f_3(t)P_3 + f_4(t)P_4$.

From the matrix:

$f_i(t) = \binom{n}{i} * (1-t)^{(n-i)} * t^i$

These are also called Bernstein polynomials.

Bezier Basis Matrix

A cubic Bezier can be defined with four points where:
P_1,R_1 at t=0 and P_4,R_4 at t=1 for a Hermite.
$R_1 = 3(P_2-P_1)$ and $R_4 = 3(P_4-P_3)$.

We can thus compute the Bezier Basis Matrix by finding the matrix that transforms $[P_1\ P_2\ P_3\ P_4]^T$ into $[P_1\ P_4\ R_1\ R_4]^T$ i.e.

B_H =[1 0 0 0]
[0 0 0 1]
[-3 3 0 0]
[0 0 -3 3]

$M_{bezier}=M_{hermite} * B\_H$

Basis Functions

Basis functions can be thought of as interpolating functions.
Note: actual interpolation of any point only happens if its Basis function is 1 and all others are zero at some t.

Often Basis functions for design curves sum to 1 for all t.
This gives the curve some nice properties like affine invariance and the convex hull property when the function are additionally non-negative.

Bezier Basis Functions

[-1 3 -3 1]
[3 -6 3 0]
[-3 3 0 0]
[1 0 0 0]

The columns of the Basis Matrix form Basis Functions such that:
$p(t)= f_1(t)P_1 + f_2(t)P_2 + f_3(t)P_3 + f_4(t)P_4$.

From the matrix:

$f_i(t) = \binom{n}{i} * (1-t)^{(n-i)} * t^i$

These are also called Bernstein polynomials.

Geometric continuity at a joint of two curves

Geometric Continuity
G_0: curves are joined
G_1: first derivatives are proportional at the join point
The curve tangents thus have the same direction,
but not necessarily the same magnitude.
i.e., $C_1'(1) = (a,b,c)$ and $C_2'(0) = (k*a, k*b, k*c)$.

G_2: first and second derivatives are proportional at join point
Parametric Continuity
C_0: curves are joined
C_1: first derivatives equal
C_2: first and second derivatives are equal
If t is taken to be time, the acceleration is continuous.
C_n: nth derivatives are equal

Local vs. Global control

Changing a point on the Bezier changes the curve mostly near the point, but a little bit everywhere…

Precise local control can be handled by splines where the Basis functions of points symmetrically increase from 0 to a maximum value over a window and then decrease to 0.

The curve is strictly affected by the point over this parameter range or window.