In CAD design, for e.g. designing an airplane or a flange, tank, you encounter several surfaces, not many planar regions. The question is how do you represent, process & render these surfaces efficiently.

I. Piecewise Linear Representation.

E.g. Curve represented by lines (Surfaces by triangles)

![Curves](image)

a. Simplest representation.

Problems:

a. APPROXIMATION: Can never represent the curve accurately, unless no. of linear segments is close to $\Delta$.

b. NUMBER OF PRIMITIVES: A very large number of primitives even to represent something reasonably.
C. PROCESSING is slow: Due to the huge number, processing is slow.

D. NO CONTINUITY AT BOUNDARIES: Always some discontinuity at the boundaries.

II. Use one parametric, implicit or explicit representation for the whole surface.

Problems

a. May not be mathematically feasible.

\[ eq: \begin{align*}
\end{align*} \]

b. No local control: Changing one part will change the entire surface cure.

\[ eq: \begin{align*}
\end{align*} \] Just this change is difficult to make
so we make a tradeoff between the two.

III. Piecewise polynomial curves/surfaces.

Each is a curve/surface

Advantage

a) Accurate representation
b) Local control is available to change some of the pieces.
c) We can control the continuity at the boundary.
d) Since curves we do not need to have a huge numbers as lines.
e) Processing is faster PIECES Parametric form to define the curves.

\[ C(t) = (x(t), y(t), z(t)) \]

Just as in straight line,

\[ x(t) = x_1 + t(x_2-x_1) \quad y(t) = y_1 + t(y_2-y_1) \quad z(t) = z_1 + t(z_2-z_1) \Rightarrow p = p_1 + t(p_2-p_1) \]
Controlled by one parameter, plug in different values of $t$ to get points on the curve.

For surfaces,

$$S(u,v) = (x(u,v), y(u,v), z(u,v))$$

Now, what kind of functions are these $x$, $y$, $z$?

Usually they are single or multivariate cubic polynomials.

$$x(t) = a t^3 + b t^2 + c t + d$$

$$x(u,v) = a_0 + (\text{degree 0 terms})$$

$$+ a_1 u + a_2 v$$

$$+ a_3 u^2 + a_4 uv + a_5 v^2$$

$$+ a_6 u^3 + a_7 uv^2 + a_8 u^2 v + a_9 uv^3$$

why?

a) Any thing of lower degree does not give enough flexibility.

b) Higher degree curves result in too many wiggles.

c) Provide minimum curvature interpolant $N+3$ pts. smoothest curve is a cubic from $x$. 
(3) No lower degree curve can maintain continuity at the boundary.

\[ Q(t) = [x(t), y(t), z(t)] \]

\[ x(t) = ax^3 + bx^2 + cx + dx \]
\[ y(t) = ay^3 + by^2 + cy + dy \]
\[ z(t) = az^3 + bz^2 + cz + dz \]

Represented as a matrix in a compact manner:

\[
\begin{bmatrix}
\begin{bmatrix}
  n \\
  y \\
  z
\end{bmatrix}
\end{bmatrix} =
\begin{bmatrix}
  a_1 & a_2 & a_3 & a_4 \\
  b_1 & b_2 & b_3 & b_4 \\
  c_1 & c_2 & c_3 & c_4 \\
\end{bmatrix}
\begin{bmatrix}
  x^3 \\
  x^2 \\
  x \\
\end{bmatrix}
\]

3 x 1 3 x 4 4 x 1

\[ y \text{ Coefficient Matrix } (C) \]

Now what is \( Q'(t) = \frac{d}{dt} Q(t) \).

\[ Q'(t) = [n'(t), y'(t), z'(t)] \]

\[ n'(t) = 3ax^2 + 2bx + cx \]
\[ y'(t) = 3ay^2 + 2by + cy \]
\[ z'(t) = 3az^2 + 2bz + cz \]
can be again written as a matrix as

\[
\begin{bmatrix}
    x' \\
    y' \\
    z'
\end{bmatrix} = 
\begin{bmatrix}
    a_{11} & b_{11} & c_{11} & d_{11} \\
    a_{12} & b_{12} & c_{12} & d_{12} \\
    a_{13} & b_{13} & c_{13} & d_{13} \\
    a_{14} & b_{14} & c_{14} & d_{14}
\end{bmatrix} \begin{bmatrix}
    3t^2 \\
    2t \\
    1 \\
    0
\end{bmatrix}
\]

\[\text{C remains unchanged. Derivative of } T = \begin{bmatrix}
    f_{x1} \\
    f_{y2} \\
    f_{z3}
\end{bmatrix} \]

\[= \begin{bmatrix}
    \frac{d}{dt} \frac{n'}{n''} \\
    \frac{d}{dt} y' \\
    \frac{d}{dt} z'
\end{bmatrix} = C \frac{d}{dt}
\]

**CONTINUITY**

When designing surfaces it is important to maintain continuity at junctions of two surfaces.

We can define different types of continuities depending on behavior of
If \( Q_1(t) = Q_2(0) \), just the value of the curve is same at \( J \), we call it geometric continuity of order \( O^0 \).

If the direction of the derivatives (defines the tangent vector at that point on the curve) has same direction,

\[
\frac{Q_1'(1)}{|Q_1'(1)|} = \frac{Q_2'(0)}{|Q_2'(0)|} \quad \text{[defined by normalizing to get unit vectors]}
\]

Then called geometric continuity of order \( 1 \), \( G^1 \)

If an additional criteria is added to this, magnitude of tangent vectors should also be same, i.e.

\[
|Q_1'(1)| = |Q_2'(0)|
\]

Then called parametric continuity of order \( 1 \), \( C^1 \). Thus \( C^1 \) is a stricter measure than \( G^1 \).
Note that since there is no directim involved with 0th order, \( C^0 = G^0 \).

The same concept can be extended to higher orders. For e.g. Order 2 assumes continuity in curvature. It will lead to smoother & smoother curves as order is increased.

E.g. yet.

In general \( C^n \) continuity \( \Rightarrow \) \( G^n \) continuity, but not the other way round.

E.g.
If tangent vector is \((0, 0, 0)\), \(C^1 \not\rightarrow G^1\).
Since nothing cannot be said for a well vector.

How to find the coefficient matrix?
Given two endpts of a curve & the tangent at those end pts.

Say \(Q\), starts at \((1, 2, 1) \rightarrow Q(0)\) \(\circ\)
ends at \((4, 5, 4) \rightarrow Q(1)\) \(\circ\)

Derivative at start is \((1, 1, 1) \rightarrow Q'(0)\) \(\circ\)
at end is \((6, 6, 6) \rightarrow Q'(1)\) \(\circ\)

From \((0)\)

\[
\begin{bmatrix}
-1 \\
2 \\
1 \\
\end{bmatrix} = C \begin{bmatrix}
1 \\
2^2 \\
4 \\
\end{bmatrix} = C \begin{bmatrix}
0 \\
0 \\
1 \\
\end{bmatrix}
\]

From \((2)\)

\[
\begin{bmatrix}
4 \\
5 \\
4 \\
\end{bmatrix} = C \begin{bmatrix}
1 \\
1/2 \\
1 \\
\end{bmatrix}
\]
\[\begin{bmatrix} 3 \\ 3 \\ 2 \\ 1 \end{bmatrix} = C \begin{bmatrix} 3x^2 \\ 2x \\ 1 \end{bmatrix} = C \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}\]

\[\begin{bmatrix} 2 \\ 6 \\ 6 \\ 1 \end{bmatrix} = C \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}\]

\[\begin{bmatrix} 1 \\ 4 \\ 1 \\ 6 \end{bmatrix} = C \begin{bmatrix} 0 \\ 1 \\ 0 \\ 3 \end{bmatrix}\]

\[\begin{bmatrix} 2 \\ 5 \\ 1 \\ 6 \end{bmatrix} = C \begin{bmatrix} 0 \\ 1 \\ 2 \\ 1 \end{bmatrix}\]

Basis Matrix (B)

(remains same as long as the way to define curve is same)

\[4 \times 4\]

\[C = \begin{bmatrix} 2 \\ 5 \\ 1 \\ 6 \end{bmatrix} B^{-1}\]

Inverse exists since Square matrix
For a particular class of curves, Bari's matrix is unique so B⁻¹ can be precomputed.

For example, curves defined by \( Q(0), Q(1), Q'(0) \) & \( Q'(1) \) are called Hermite curves.

\( B \) for Hermite curves is:

\[
\begin{bmatrix}
0 & 1 & 0 & 3 & 7 \\
0 & 0 & 1 & 0 & 2 \\
0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
B^{-1} = \begin{bmatrix}
0 & 2 & -3 & 0 & 1 \\
-2 & 3 & 0 & 0 & 0 \\
1 & -2 & 1 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

Now \( Q = C \cdot T = \begin{bmatrix} P_1 & P_2 & T_1 & T_2 \end{bmatrix} B^{-1} T \).

where

\[
\begin{align*}
P_1 &= Q(0) \\
P_2 &= Q(1) \\
T_1 &= Q'(0) \\
T_2 &= Q'(1)
\end{align*}
\]

This is called the Geometry Matrix. This changes from one Hermite curve to another and is responsible for defining the geometry of one particular curve.
\[- \begin{pmatrix} Q \end{pmatrix} = C^T \begin{bmatrix} p_1 & p_2 & p_1 & p_2 \end{bmatrix} \begin{bmatrix} 2 & -3 & 0 & 1 \\ -2 & 3 & 0 & 0 \\ 1 & -2 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} t^3 \\ t^2 \\ t^1 \\ t^0 \end{bmatrix} \]

\[= \begin{bmatrix} p_1 & p_2 & p_1 & p_2 \end{bmatrix} \begin{bmatrix} 2t^3 - 3t^2 + 1 \\ -2t^3 + 3t^2 \\ t^3 - 2t^2 + t \\ t^3 - t^2 \end{bmatrix} \begin{bmatrix} t^1 \\ t^2 \\ t^3 \end{bmatrix} \]

\[= p_1 (2t^3 - 3t^2 + 1) + (-2t^3 + 3t^2)p_2 + (t^3 - 2t^2 + t)p_1 + (t^3 - t^2)p_2 \]

The curve \( Q(t) \) can also be represented as a weighted sum of its geometric parameters \( p_1, p_2, p_1, p_2 \).

These weights expressed as functions of the parameter \( t \), \( W(t) \), are called the blending functions.

Hermite curves are curves that are blended from their endpoints & tangents at the end pts.
Plot of the Blending fn. (called Interpolating Curves)

Note 1. At \( t = 0 \),
\( P_2 \) has 0 contribution.

2. At \( t = 1 \), \( P_1 \) has 0 contribution.

3. \( T_1 \) & \( T_2 \) contribute only in middle.

4. \( T_2 \) has negative contribution also.

Helps us to decide which pts to manipulate to change the shape of the curve when modeling a curve.

E is changed all to mathematically user friendly. Do not need to think about E when modeling.
Bezier Curves

A different type of curve defined by 4 points.

\[ P_1 \rightarrow P_2 \rightarrow P_3 \rightarrow P_4 \]

Four pts to control the shape of the curve.

And the constraint is

\[ T_1 = 3(p_2 - p_1) \]
\[ T_4 = 3(p_4 - p_3) \]

i. Geometry Matrix

\[
\begin{bmatrix}
  p_1 & p_2 & T_1 & T_4 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
  p_1 & p_2 & p_3 & p_4 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
  1 & 0 & -3 & 0 \\
  0 & 0 & 3 & -3 \\
  0 & 1 & 0 & 3 \\
\end{bmatrix}
\]

Find the Basis Matrix

Same as Hermite Curves since still defining \( \dot{Q}(0), \dot{Q}(1), Q'(0), Q'(1) \), only that now a special relationship holds.
\[
B^{-1} = \begin{bmatrix}
-2 & -3 & 0 & 1 \\
1 & -2 & 0 & 0 \\
1 & -1 & 0 & 0 \\
\end{bmatrix}
\]

\[
C = GB^{-1}
\]

\[
= \begin{bmatrix}
P_1 & P_2 & T_1 & T_4 \\
\end{bmatrix} B^{-1}
\]

\[
= \begin{bmatrix}
P_1 & P_2 & P_3 & P_4 \\
\end{bmatrix} \begin{bmatrix}
1 & 6 & -3 & 0 \\
0 & 0 & 3 & -3 \\
0 & 1 & 0 & 3 \\
\end{bmatrix} B^{-1}
\]

Called Control Pts.

\[
= \begin{bmatrix}
P_1 & P_2 & P_3 & P_4 \\
\end{bmatrix} \begin{bmatrix}
-1 & 3 & -3 & 1 \\
-3 & -6 & 3 & 0 \\
-3 & 3 & 0 & 0 \\
1 & 0 & 0 & 0 \\
\end{bmatrix}
\]

\[
\therefore \text{Blending fun. are}
\begin{bmatrix}
P_1 & P_2 & P_3 & P_4 \\
\end{bmatrix} \begin{bmatrix}
-1 & 3 & -3 & 1 \\
-3 & -6 & 3 & 0 \\
-3 & 3 & 0 & 0 \\
1 & 0 & 0 & 0 \\
\end{bmatrix} \begin{bmatrix}
k^3 \\
k^2 \\
k^1 \\
k^0 \\
\end{bmatrix}
\]

\[
P_1 \cdot (k^3 + 3k^2 - 3k + 1) + P_2 \cdot 3(k^3 - 2k^2 + 1) + P_3 \cdot (-3k^3 + 3k^2) + P_4 \cdot k^3
\]
\[ = p_1 \cdot (1-t)^3 + 3t(1-t)^2p_2 + 3t^2(1-t)p_3 + t^3p_4 \]

These set of blending functions are called Bernstein polynomials.

See that at no point in \( t \), \( p_2 \) and \( p_3 \) are the sole contributor. For eg. \( p_1 \) is the sole contributor at \( t=0 \). \( \therefore \) Curve passes through \( p_1 \) since \( Q(0) = p_1 \). Similarly at \( t=1 \), \( Q(1) = p_4 \) \( \therefore \) Curve passes through \( p_4 \) but cannot pass through \( p_3 \) & \( p_4 \).

\( \therefore \) This is a non-interpolating curve since does not pass through all the control \( p_i \)s.
These are cubic curves. Since cubic polynomials for defining the curve, blending functions are also cubic.

**Properties**

a) Curve bounded by the convex hull of control pts. - Called Control Polygon

b) Variation Diminishing Property: - A line intersects the curve no more than its intersection with the control polygon. - No wiggles in control polygon signifies no wiggles in the curve.

c) Symmetric - Reversing control pts order yields same curve with reverse parameterization.

d) Affine Invariant: - Affine transformations of control pts yields affine transformation of curve itself.

e) Can be easily subdivided for rendering.
Mark general

Can be of any degree. Control
in eq. If linear—Need two pts in
Geometry Matrix. Control
If quadratic—Need three pts
in geometry Matrix
If quadric—Need 5 pts.

Bezier is most popular for surfaces.
Only positive blending for is easy to
understand.
2) Non-interpolating gives better control.
3) Only a few control pts & easy to use
but has large flexibility to design
a varied types of curves.

Higher Order Bezier
bernstein Polynomials of degree 3,

\[ b_0, 3 = (1-u)^3 \]
\[ b_1, 3 = 3u(1-u)^2 \]
\[ b_2, 3 = 3u^2(1-u) \]
\[ b_3, 3 = u^3 \]

They have a more general form for
degree n.
\[ b_{k,n} = \binom{n}{k} u^k (1-u)^{n-k} \]
\[ = \frac{n!}{k! (n-k)!} u^k (1-u)^{n-k} \]

\[ \text{with } n \text{ control points,} \]
\[ p(u) = \sum_{k=0}^{n} p_k \binom{n}{k} u^k (1-u)^{n-k}. \]
How do we render?

Subdivide the curve to smaller & smaller segments until each segment is close enough to a line.

Then render the line.

Need an easy subdivision technique.

Say you want to divide the curve at \( t = u \), i.e. at \( Q(u) \),

De Casteljau Construction

1) Divide each \( P_i P_{i+1} \) line segment in \( u \) & \( 1-u \) ratio.
2) This will generate \( N-1 \) lines.
3) Join the pts in sequence to get \( N-2 \) lines.
4) Again subdivide these lines in same way. Will give from \( N-2 \) pts & \( N-3 \) lines.
5) Continue until you get one pt.
6) This pt will be \( Q(u) \).
7) Follow pts of division back to $P_i$ on left & $P_i$ on right.

Eq. $R \ k_1 \ L \ P_i \rightarrow$ on left toward $P_i$

$R \ k_2 \ L \ P_i \rightarrow$ on right toward $P_i$

8. These will generate the control pts for the subdivided curves.