

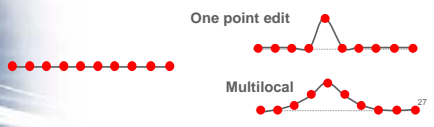
Refinement

- We can now move the inner (refined) points, while all derivatives at the outer points remain unchanged → continuity is guaranteed
- Refinement also solves the scale problem, if small scale deformations are required

26

The scale problem

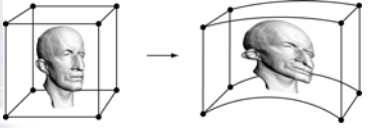
- If we want a control point edit to affect multiple neighboring control points, we can let these control points move along, but scaled down – using a factor decreasing with distance



27

The scale problem: Global shape deformation

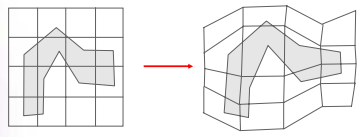
- Global shape changes can be implemented using *free-form deformation*
- ‘Rubber-like’ deformation of space



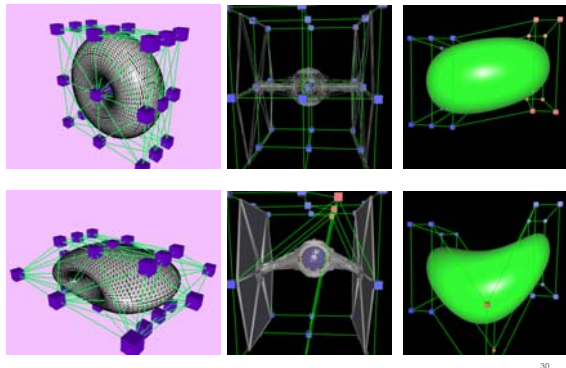
28

Free-form deformation

- Deformation can be modelled by deforming a regular grid



29

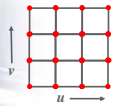


Free-form deformation

- In 2D, if we regard a 3x3 grid as a parametric (u, v) surface, we can write it as

$$(u, v) = \sum_{i=0}^3 \sum_{j=0}^3 \mathbf{R}_{ij} B_i(u) B_j(v)$$

• = \mathbf{R}_{ij}



31

Free-form deformation

- A planar deformation by shifting the control points \mathbf{R}_{ij} to \mathbf{R}'_{ij} can be written as

$$(u', v') = \sum_{i=0}^3 \sum_{j=0}^3 \mathbf{R}'_{ij} B_i(u) B_j(v)$$

32

Free-form deformation

$$(u', v') = \sum_{i=0}^3 \sum_{j=0}^3 \mathbf{R}'_{ij} B_i(u) B_j(v)$$

- The same formula can be used to transform the control points of curves on the grid – by filling in their (u, v) values

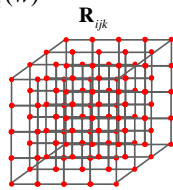
33

Free-form deformation

- We need a 2D grid to specify the deformation of a curve →
- We need a 3D grid to specify a surface deformation

$$(u', v', w') = \sum_{i=0}^3 \sum_{j=0}^3 \sum_{k=0}^3 \mathbf{R}'_{ijk} B_i(u) B_j(v) B_k(w)$$

- Control points \mathbf{P}_{ij} are transformed by filling in their (u, v, w) values
- Free-form deformation can be used on *any* parametric model



Modelling with patches

- Sweeping**
- Interaction**
Editing control points, usually of primitive geometric shapes
- Creating a mesh**
from a set of 3D points

35

Fitting a surface

- Turning a cloud of 3D points to a patch surface




Source of points

- E.g. range scanner:

37

Curve fitting

- E.g. fitting a smooth curve through sampled location points of an object through time

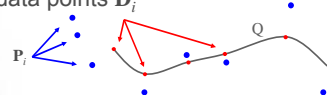


- We already have a solution: interpolating splines (Natural, Hermite, Cardinal...)
- Better: Interpolate a B-spline
 - faster to compute than Natural
 - less constraints than Hermite/Cardinal
 - C² continuity

38

B-spline curve fitting

- Problem: find a set of control points \mathbf{P}_i for a B-spline $\mathbf{Q}(u)$ that interpolates all the data points \mathbf{D}_i



- Assume data points are interpolated at knot values of u

39

Cubic B-spline curve fitting

$$\mathbf{Q}(u_p) = \sum_{i=0}^m \mathbf{P}_i B_i(u_p) = \mathbf{D}_p \quad p = 3, \dots, m+1$$

- Curve equals data point \mathbf{D}_p at knot value u_p
- Easiest solution: set $u_p = p$ (uniform parametrisation)
- Better: space knot values according to *real* data point distances

40

Cubic B-spline curve fitting

$$\mathbf{Q}(u_p) = \sum_{i=0}^m \mathbf{P}_i B_i(u_p) = \mathbf{D}_p \quad p = 3, \dots, m+1$$

For one component: $x(u_p) = \sum_{i=0}^m \mathbf{P}_{x,i} B_i(u_p) = \mathbf{D}_{x,p}$

$$\begin{bmatrix} B_0(u_3) & \dots & B_m(u_3) \\ \vdots & \ddots & \vdots \\ B_0(u_{m+1}) & \dots & B_m(u_{m+1}) \end{bmatrix} \begin{bmatrix} \mathbf{P}_{x,0} \\ \vdots \\ \mathbf{P}_{x,m} \end{bmatrix} = \begin{bmatrix} \mathbf{D}_{x,3} \\ \vdots \\ \mathbf{D}_{x,m+1} \end{bmatrix}$$

m-1 equations, m+1 unknowns: 2 more equations needed → we can interpolate \mathbf{D}_2 and \mathbf{D}_{m+2}

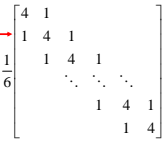
$$\begin{bmatrix} B_0(u_2) & \dots & B_m(u_2) \\ \vdots & \ddots & \vdots \\ B_0(u_{m+2}) & \dots & B_m(u_{m+2}) \end{bmatrix} \begin{bmatrix} \mathbf{P}_{x,0} \\ \vdots \\ \mathbf{P}_{x,m} \end{bmatrix} = \begin{bmatrix} \mathbf{D}_{x,2} \\ \vdots \\ \mathbf{D}_{x,m+2} \end{bmatrix}$$

New first and last rows

41

Cubic B-spline curve fitting

- The B matrix is tridiagonal → easy system to solve
- Name the $m+1$ data points $\{\mathbf{D}_2, \dots, \mathbf{D}_{m+2}\}$ to directly use them in the formula
- Note that with $m+1$ control points you can 'interpolate' $m+1$ data points, but the first and last aren't truly reached



For uniform B-spline

42

Exercise

- Given the set of data points: $\{(0,0), (1,0), (2,0), (2,1)\}$, give the system that provides the y -coordinates of the control points of the interpolating B-spline, and solve it

$$\begin{bmatrix} B_0(u_2) & \dots & B_m(u_2) \\ \vdots & \ddots & \vdots \\ B_0(u_{m+2}) & \dots & B_m(u_{m+2}) \end{bmatrix} \begin{bmatrix} \mathbf{P}_{x,0} \\ \vdots \\ \mathbf{P}_{x,m} \end{bmatrix} = \begin{bmatrix} \mathbf{D}_{x,2} \\ \vdots \\ \mathbf{D}_{x,m+2} \end{bmatrix}$$

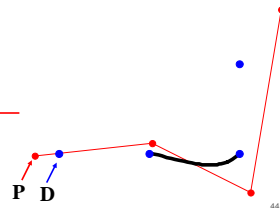
43

Solution

$$\frac{1}{6} \begin{bmatrix} 4 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \rightarrow \begin{cases} (4y_0 + y_1) = 0 \\ (y_0 + 4y_1 + y_2) = 0 \\ (y_1 + 4y_2 + y_3) = 0 \\ (y_2 + 4y_3) = 6 \end{cases}$$

$$\begin{aligned} y_1 &= -4y_0 \\ -15y_0 + y_2 &= 0 & y_2 &= 15y_0 \\ 56y_0 + y_3 &= 0 & y_3 &= -56y_0 \\ 15y_0 - 224y_0 &= 6 & y_0 &= -\frac{6}{209} \end{aligned}$$

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \frac{1}{209} \begin{bmatrix} -6 \\ 24 \\ -90 \\ 336 \end{bmatrix}$$



Surface fitting

- General approach:
 - Fit a network of u, v curves through the data points. Use the B-spline curve fitting technique
 - Convert each B-spline to Bézier segments
 - Each Bézier curve forms the edge of a surface patch

1. Curve fitting

- The fitting of the u, v curves to the data cloud is generally non-trivial! → requires some knowledge of the topology of the surface
- We consider examples where this is known, e.g.
 - Manual digitisation of points
 - Range scanner

2. Conversion to Bézier segments

- The fitted u, v curves are transformed to Bézier curves: Each segment Q_i converts to a four-point Bézier curve

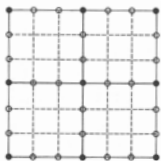
$$[P_0 \ P_1 \ P_2 \ P_3] = \mathbf{B}_z^{-1} \mathbf{B}_i [R_0 \ R_1 \ R_2 \ R_3]^T$$

$$= \frac{1}{6} \begin{bmatrix} 1 & 4 & 1 & 0 \\ 0 & 4 & 2 & 0 \\ 0 & 2 & 4 & 1 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} R_0 \\ R_1 \\ R_2 \\ R_3 \end{bmatrix}$$

R: B-spline control points
P: Bézier control points

2. Conversion to Bézier segments

- First split all the u B-spline curves into Bézier curves, then all the v curves
- This gives a network of Bézier control points:



- Which can be regarded as forming 4x4 patches, but with interior points missing

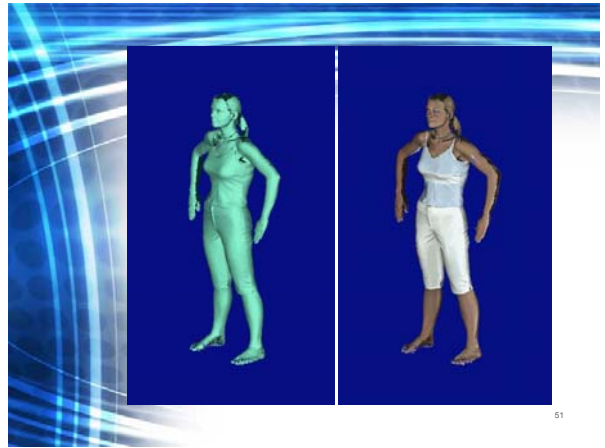
3. Conversion to Bézier patches

- To convert the grid to full patches, we need to estimate the interior points
- This is often done by assuming zero-twist ($Q_{uv}=0$) at the corner control points: implies all subquadrilaterals of the control polyhedron are identical parallelograms and the polyhedron is flat:
 - The interior points are then easily estimated
 - This is only realistic if opposite edges of a patch are similar (only translates). In practice, requires a sufficiently dense u, v sampling





50



51