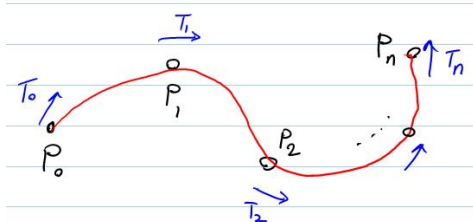


Hermite Curves



Hermite curves

- ❖ Interpolation versus approximation
- ❖ Hermite curve interpolates the control points
- ❖ Piecewise cubic polynomials



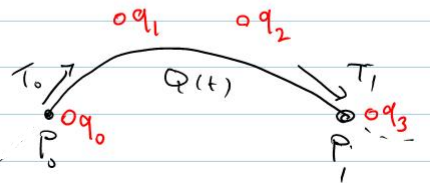
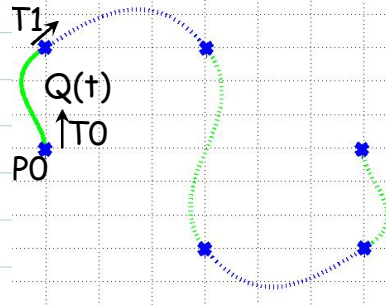
P_0, P_1, \dots, P_n
 T_0, T_1, \dots, T_n given

Find the curve (Red)

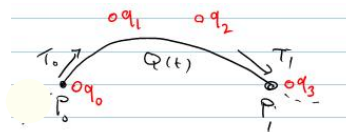
Focus on one segment

We find a piecewise cubic polynomial.

To solve it, first we focus only on one of the segments:



Control points of Bezier curve



$Q(t)$: a cubic Bezier

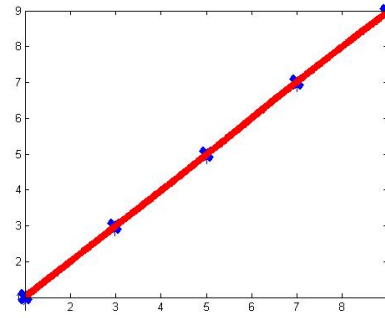
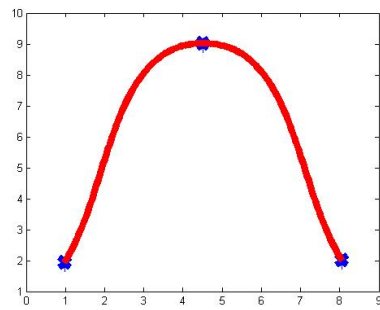
$$Q(t) = B_{0,3}(t)q_0 + B_{1,3}(t)q_1 + B_{2,3}(t)q_2 + B_{3,3}(t)q_3$$

⇓

$$q_0 = p_0 \quad q_1 = p_0 + \frac{1}{3}T_0$$

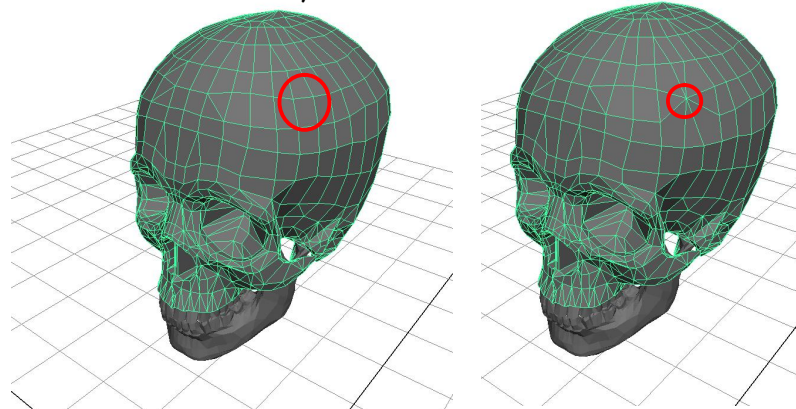
$$q_3 = p_1 \quad q_2 = p_1 - \frac{1}{3}T_1$$

Examples



Mesh processing

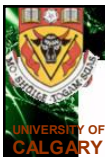
Collapse: edge, vertex, face
Split: edge, vertex, face
Add: new vertices, faces





Queries

- ❖ Given a vertex, Which edges are share it?
- ❖ Given a vertex, Which vertices are adjacent to it?
- ❖ Given a vertex, which faces are adjacent to it?
- ❖ ...



Inefficiencies for mesh processing

- ❖ Hard to add a new point/vertex/face
- ❖ find the vertex neighborhood which is important for any mesh operation is very hard
- ❖ Better methods ??

Example

Vertex	half-edge	x	y	z
V1	Ex (or Hy)			
V2	Gx			
...	...			

Vertex List

face	half-edge
F1	Ex or Gx
F2	Hy
...

Face List

half_edge	vertex (source)	face (left)	next (half-edge)	pair (half-edge)
Ex	V1	F1	Gx	Ey
Ey	V2	F2	Hy	Ex
...

Half-edge

The diagram shows a 3D object with two vertices, V1 and V2. Two faces, F1 and F2, meet at the edge between V1 and V2. Half-edges are labeled: Ex and Gx are on face F1, while Ey and Hy are on face F2. Arrows indicate the direction of the half-edges.

Half-edge

- ❖ Split the edge into two records (half-edge)
- ❖ Face and vertex list
- ❖ Traversing of the edges, vertices !?

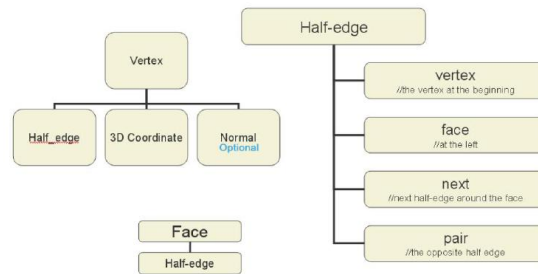
The diagram shows a 3D object with two vertices, V1 and V2. Two half-edges, El and Er, are shown originating from V1 and terminating at V2. El is on the left face and Er is on the right face.

```

graph TD
    HalfEdge[Half-edge] --- Vertex[//the vertex at the beginning]
    HalfEdge --- Face[//at the left]
    HalfEdge --- Next[//next half-edge around the face]
    HalfEdge --- Pair[//the opposite half edge]
    
    Face --- HalfEdgeFace[Half-edge]
    Vertex --- HalfEdgeVertex[Half-edge]
    Vertex --- Coord[3D Coordinate]
    Vertex --- Normal[Normal]
  
```

The diagram illustrates the structure of a half-edge. A central 'Half-edge' box is connected to four descriptive boxes: 'vertex //the vertex at the beginning', 'face //at the left', 'next //next half-edge around the face', and 'pair //the opposite half edge'. Below this, a 'Face' box is connected to a 'Half-edge' box. A 'Vertex' box is connected to three boxes: 'Half-edge', '3D Coordinate', and 'Normal'.

1. A simple mesh with no boundary vertex is saved in a Half-edge data structure. Write an $O(1)$ (constant time respect to the number of vertices) algorithm that replaces a given vertex with the average of all its adjacent vertices. (5 marks) *(This topic has not been covered in this semester yet).*



Back Faces and Hidden Surfaces

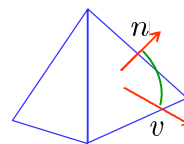
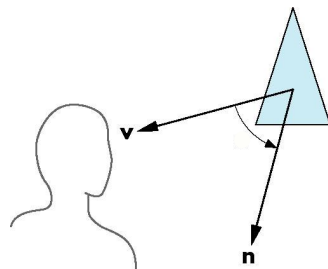
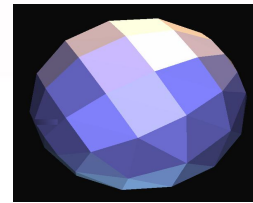
Hidden and Visible Surface

- ❖ Visible surface
 - o Parts of scene that are visible from a chosen viewpoint
- ❖ Hidden surface
 - o Parts of scene that are not visible from a chosen viewpoint



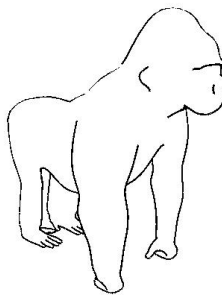
Back-Face Removal

- ❖ Or back-face culling
- ❖ We see a polygon if its normal is pointed toward the viewer
- ❖ Condition: $\cos \theta \geq 0$ or $n \cdot v \geq 0$
- ❖ Is it necessary to send a back-face



An application: Silhouette extraction

- ❖ Silhouette lines are very important for visualizing objects(very useful in the traditional art)
 - ❖ Artistic Rendering
 - ❖ How to find silhouette edges?
- ❖ any edge shared by one front-facing polygon and one back-facing polygon.

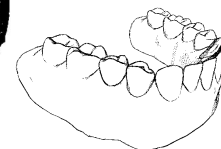
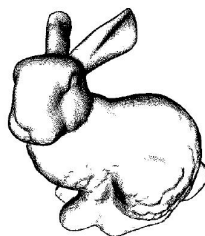
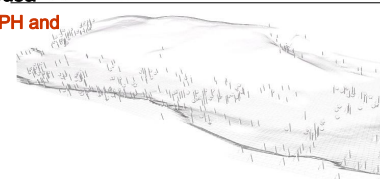
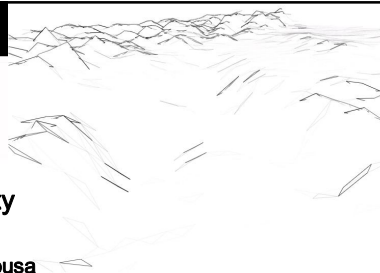


Another work

Silhouette Rendering Based On Stability Measurement

John Brosz, Faramarz Samavati, Mario Costa Sousa

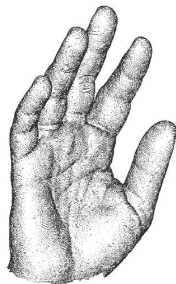
SCCG '04, organized in cooperation with ACM SIGGRAPH and Eurographic.



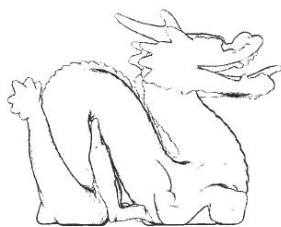
Ribbons

- ❖ Conventional: Extracting mesh from Point clouds
- ❖ Extract Ribbons
- ❖ Use them directly for representations

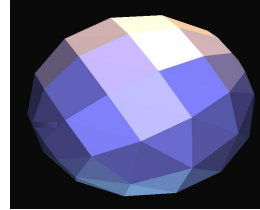
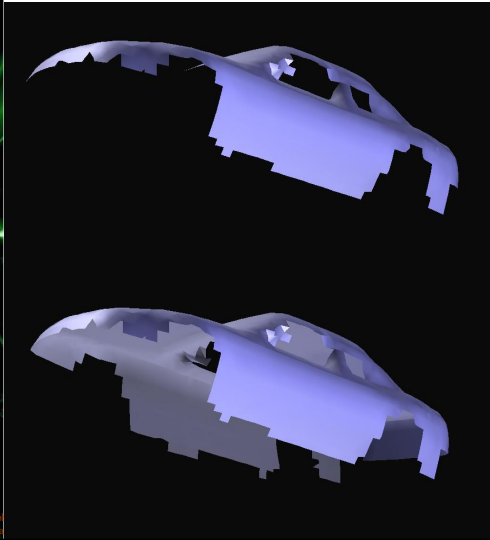
- ❖ Ribbons: A representation for point clouds", Runions, A., Samavati, F.F. and Prusinkiewicz, P., The Visual Computer, Vol. 23, No. 9-11, 2007.



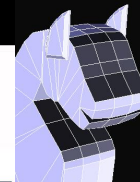
(b)



Enclosed surfaces

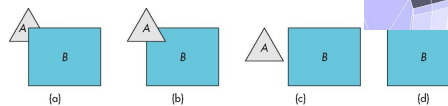


Hidden Surface Algorithms



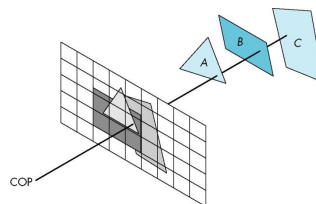
❖ Object-space

- o Working in 3D (ray tracing)



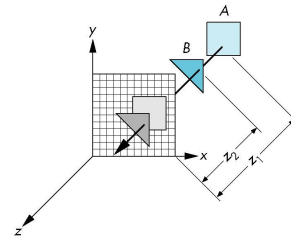
❖ Image-space

- o Decide on visibility at each pixel position



Z-Buffer Method

- ❖ A commonly used image-space approach to hidden-surface removal
- ❖ It is also referred as Depth-Buffer method
- ❖ Use the intensity color of the nearest 3D point for each pixel
- ❖ What is an efficient way for it?
- ❖ A buffer, the z-buffer with the same resolution
- ❖ Each location in the z-buffer contains the distance of the closest 3D point.



Z-Buffer Algorithm

for all positions (x,y) in the view screen

frame(x,y)=
depth(x,y)=

end

for each polygon in the mesh

for each point(x,y) in the polygon-fill algorithm

★ compute, z, the distance of corresponding 3D-point from COP

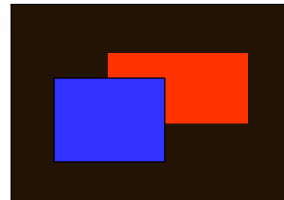
if depth(x,y) > z // a closer point

depth(x,y)=
frame(x,y)=

endif

endfor

endfor



Some Facts About the Algorithm

- ❖ After the algorithm
 - o Frame buffer contains intensity values of the visible surface
 - o z-buffer contains depth values for all visible points
- ❖ For the step \star in algorithm
 - o We know d_1 , d_2 , d_3 and d_4 from vertices of the mesh
 - o Use linear interpolation for other points

