# Mesh adjacency structures

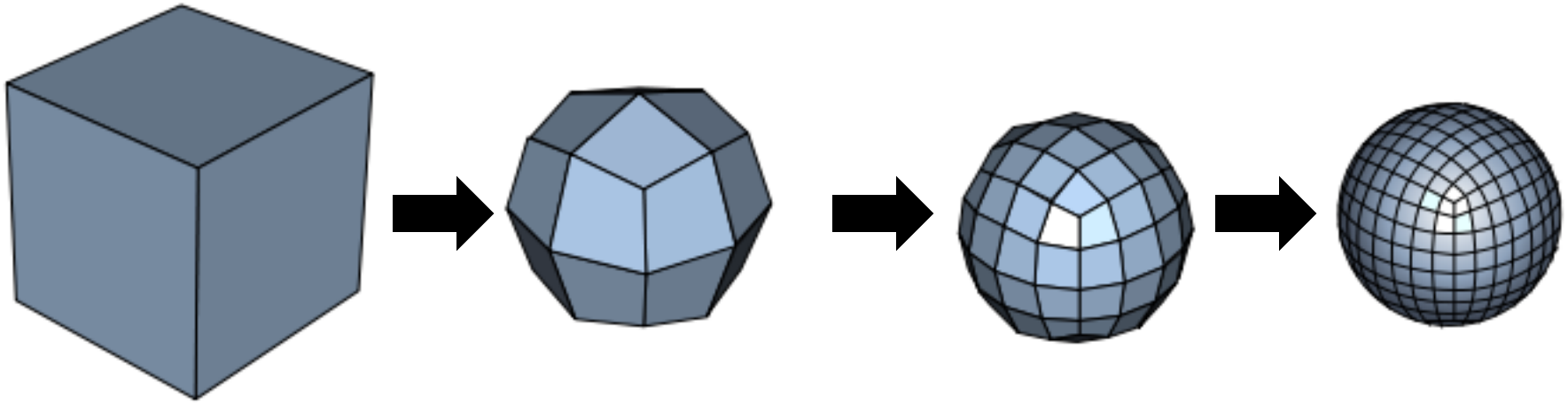
# Today

- Mesh data structures

Acknowledgement: With figures from
http://www.cs.princeton.edu/courses/archive/spring10/cos426/lectures/06-mesh.pdf

# Subdivision surfaces

http://en.wikipedia.org/wiki/Catmull%E2%80%93Clark_subdivision_surface



Input mesh

Subdivision & smoothing

Subdivision & smoothing

Subdivision & smoothing

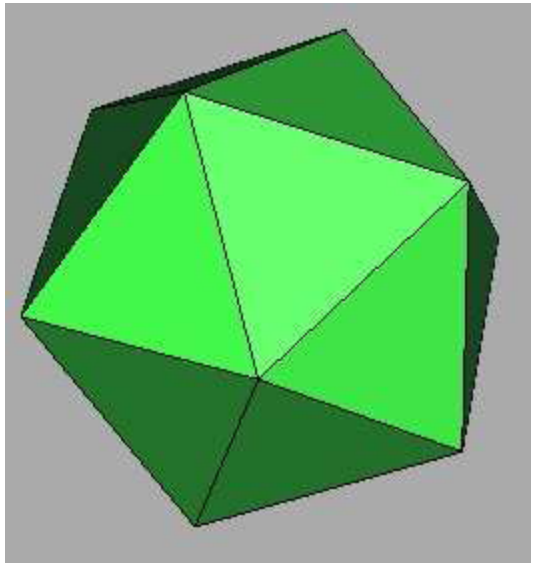
Limit surface

# Subdivision surfaces

- Smoothing step
  - Local average of neighboring vertices
  - Various schemes
- Need access to adjacency information of mesh
  - Given edge, what are adjacent faces?
  - Given vertex, what are adjacent edges?
  - Etc.

# Mesh data structures

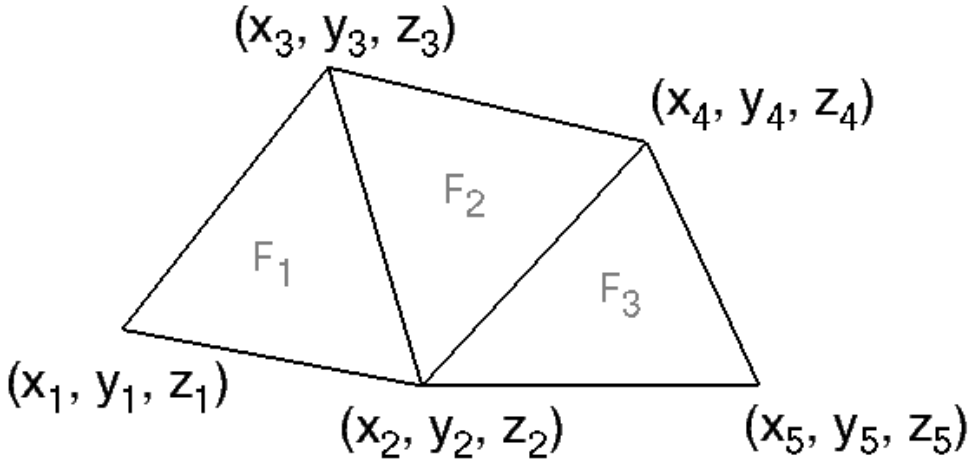
1. Independent faces
2. Vertex and face tables
3. Adjacency lists
4. Winged-edge data structure



# Independent faces

- Redundant vertices
  - Each vertex position stored for each face
- No adjacency information
- „Polygon soup“ http://en.wikipedia.org/wiki/Polygon_soup

| FACE TABLE | |
|---|---|
| $F_1$ | $(x_1, y_1, z_1)$ $(x_2, y_2, z_2)$ $(x_3, y_3, z_3)$ |
| $F_2$ | $(x_2, y_2, z_2)$ $(x_4, y_4, z_4)$ $(x_3, y_3, z_3)$ |
| $F_3$ | $(x_2, y_2, z_2)$ $(x_5, y_5, z_5)$ $(x_4, y_4, z_4)$ |

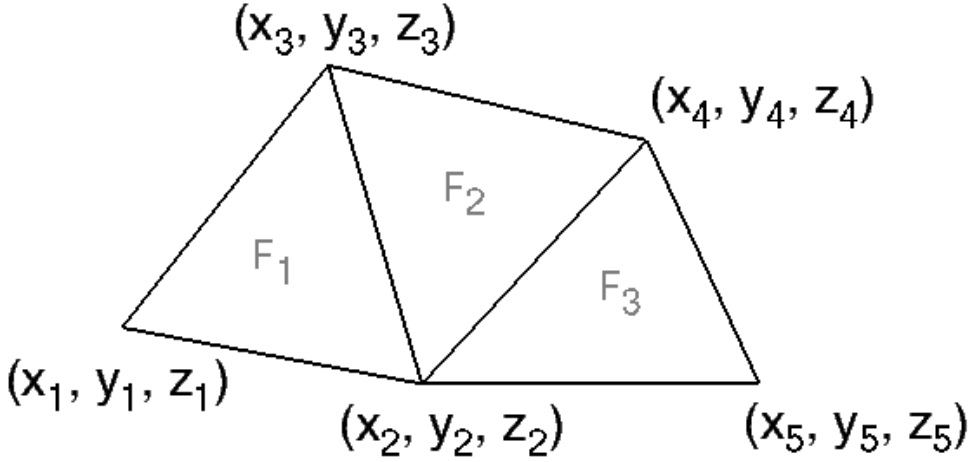


# Vertex and face tables

- Each face lists vertex references
- Shared vertices, each vertex position stored only once
- No adjacency information
- Current data structure in jrtr Java code

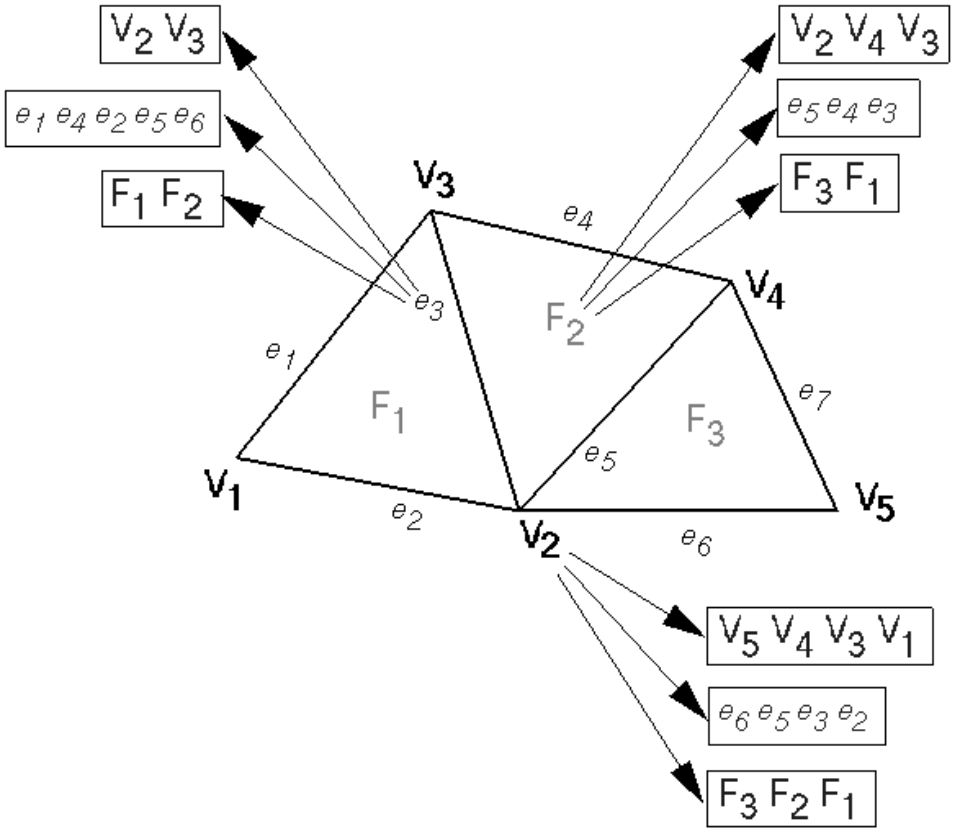
| VERTEX TABLE | | | |
|---|---|---|---|
| $v_1$ | $x_1$ | $y_1$ | $z_1$ |
| $v_2$ | $x_2$ | $y_2$ | $z_2$ |
| $v_3$ | $x_3$ | $y_3$ | $z_3$ |
| $v_4$ | $x_4$ | $y_4$ | $z_4$ |
| $v_5$ | $x_5$ | $y_5$ | $z_5$ |

| FACE TABLE | | | |
|---|---|---|---|
| $F_1$ | $v_1$ | $v_2$ | $v_3$ |
| $F_2$ | $v_2$ | $v_4$ | $v_3$ |
| $F_3$ | $v_2$ | $v_5$ | $v_4$ |



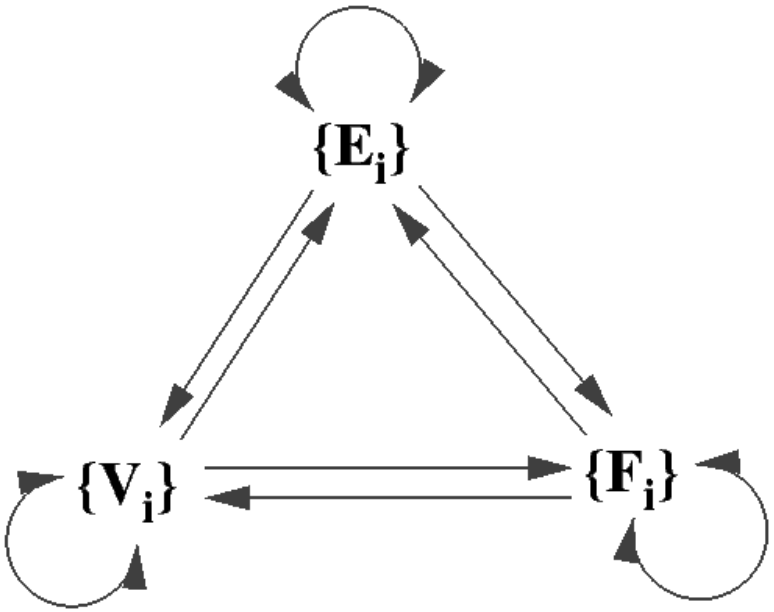
# Adjacency lists

- Store all vertex, edge, face adjacencies
- Trivial retrieval of adjacency information
- Extra storage

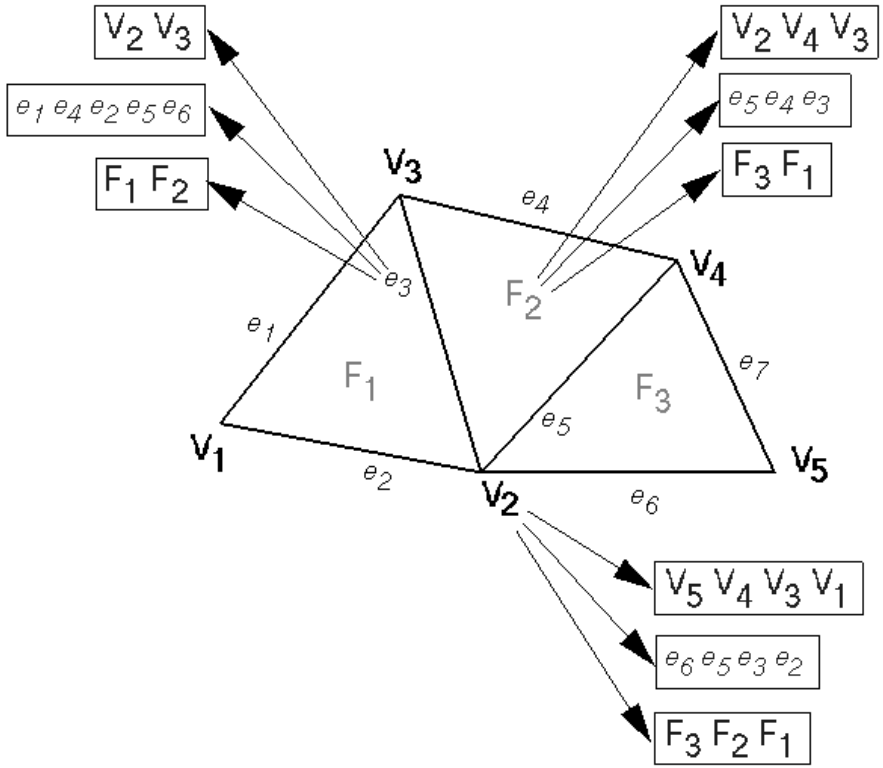


# Partial adjacency lists

- Store only some adjacency relations and derive others?

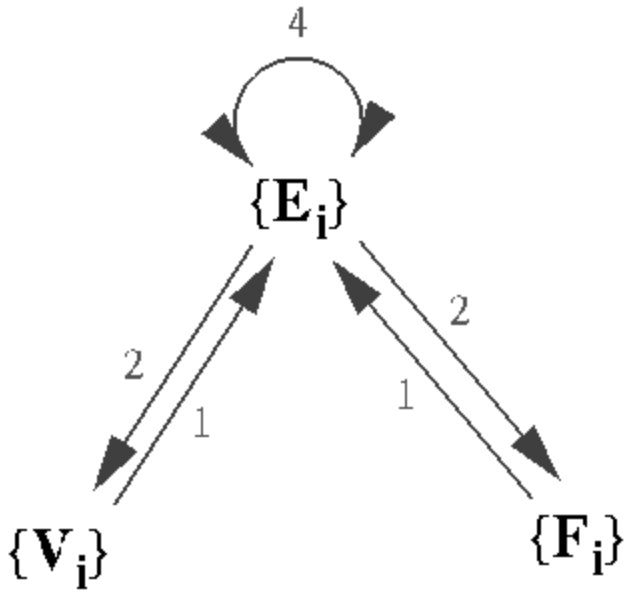



All adjacency relations

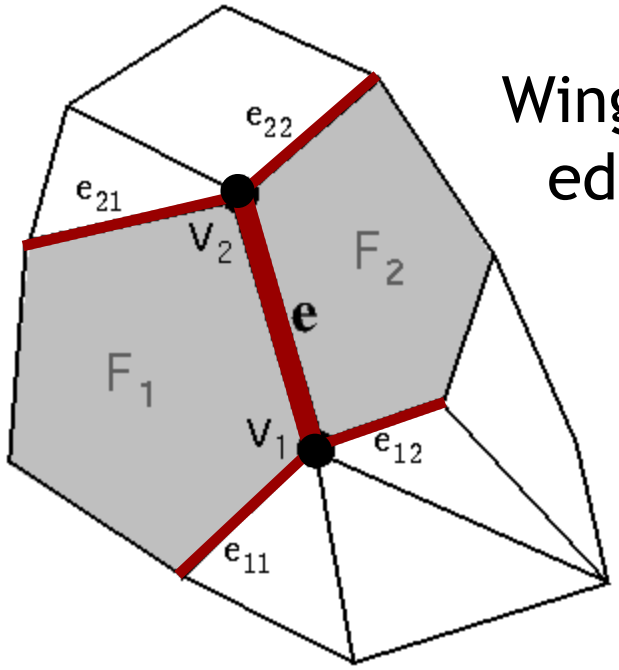


# Winged edge data structure

- Adjacency encoded in edges
- Retrieve all adjacencies in constant time
- Little extra storage
- Arbitrary polygons (not only triangles)

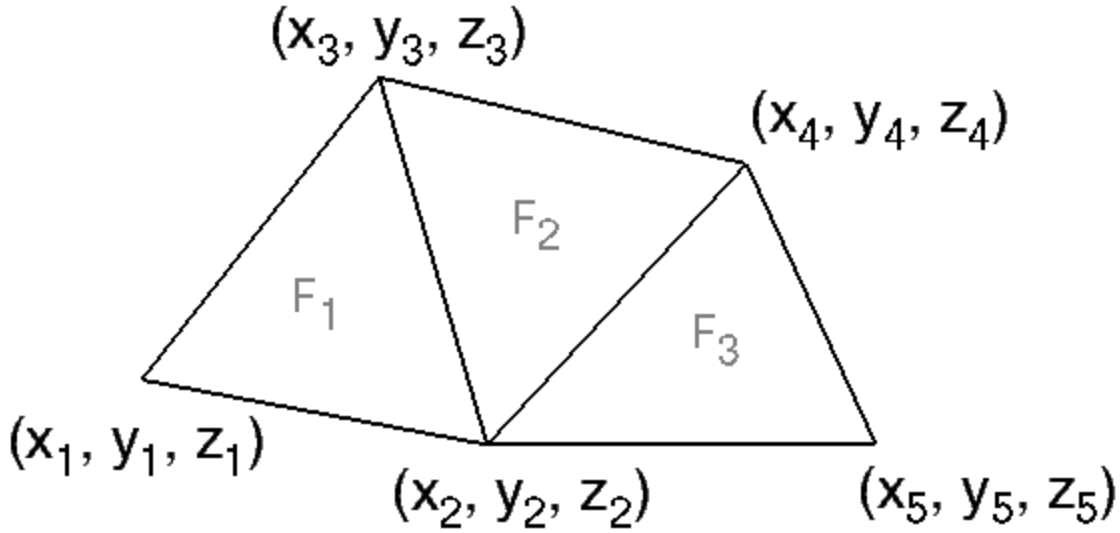



Adjacency relations
in winged edge structure




Winged
edge

# Example




**VERTEX TABLE**

| | | | | |
|---|---|---|---|---|
| $V_1$ | $X_1$ | $Y_1$ | $Z_1$ | $e_1$ |
| $V_2$ | $X_2$ | $Y_2$ | $Z_2$ | $e_6$ |
| $V_3$ | $X_3$ | $Y_3$ | $Z_3$ | $e_3$ |
| $V_4$ | $X_4$ | $Y_4$ | $Z_4$ | $e_5$ |
| $V_5$ | $X_5$ | $Y_5$ | $Z_5$ | $e_6$ |

**EDGE TABLE**

| | | | | | 11 | 12 | 21 | 22 |
|---|---|---|---|---|---|---|---|---|
| $e_1$ | $V_1$ | $V_3$ | | $F_1$ | $e_2$ | $e_2$ | $e_4$ | $e_3$ |
| $e_2$ | $V_1$ | $V_2$ | $F_1$ | | $e_1$ | $e_1$ | $e_3$ | $e_6$ |
| $e_3$ | $V_2$ | $V_3$ | $F_1$ | $F_2$ | $e_2$ | $e_5$ | $e_1$ | $e_4$ |
| $e_4$ | $V_3$ | $V_4$ | | $F_2$ | $e_1$ | $e_3$ | $e_7$ | $e_5$ |
| $e_5$ | $V_2$ | $V_4$ | $F_2$ | $F_3$ | $e_3$ | $e_6$ | $e_4$ | $e_7$ |
| $e_6$ | $V_2$ | $V_5$ | $F_3$ | | $e_5$ | $e_2$ | $e_7$ | $e_7$ |
| $e_7$ | $V_4$ | $V_5$ | | $F_3$ | $e_4$ | $e_5$ | $e_6$ | $e_6$ |

**FACE TABLE**

| | |
|---|---|
| $F_1$ | $e_1$ |
| $F_2$ | $e_3$ |
| $F_3$ | $e_5$ |

# Summary

- Need adjacency information for many mesh processing algorithms
  - E.g., subdivision
- Want efficient data structure
  - Adjacency queries in constant time
  - Little storage overhead
- Winged-edge
  - Other options exist, e.g., half-edge
    http://www.flipcode.com/archives/The_Half-Edge_Data_Structure.shtml

# Next time

- More shaders