Mesh Data Structures

Data Structures

- What should be stored?
  - Geometry: 3D coordinates
  - Attributes
    - e.g. normal, color, texture coordinate
    - Per vertex, per face, per edge
  - Connectivity
    - Adjacency relationships

Data Structures

- What should it support?
  - Rendering
  - Geometry queries
    - What are the vertices of face #2?
    - Is vertex A adjacent to vertex H ?
    - Which faces are adjacent to face #1?
  - Modifications
    - Remove/add a vertex/face
    - Vertex split, edge collapse

Data Structures

- How good is a data structure?
  - Time to construct (preprocessing)
  - Time to answer a query
  - Time to perform an operation
  - Space complexity
  - Redundancy

Mesh Data Structures

- Face Set
- Shared Vertex
- Half Edge
- Face Based Connectivity
- Edge Based Connectivity
- Adjacency Matrix
- Corner Table

Face Set

TRIANGLES		
Vertex coord.	Vertex coord.	Vertex coord.
[10 20 30]	[40 5 20]	[10 4 3]
	. . .	

- Simple
- STL File
- No connectivity
- Redundancy

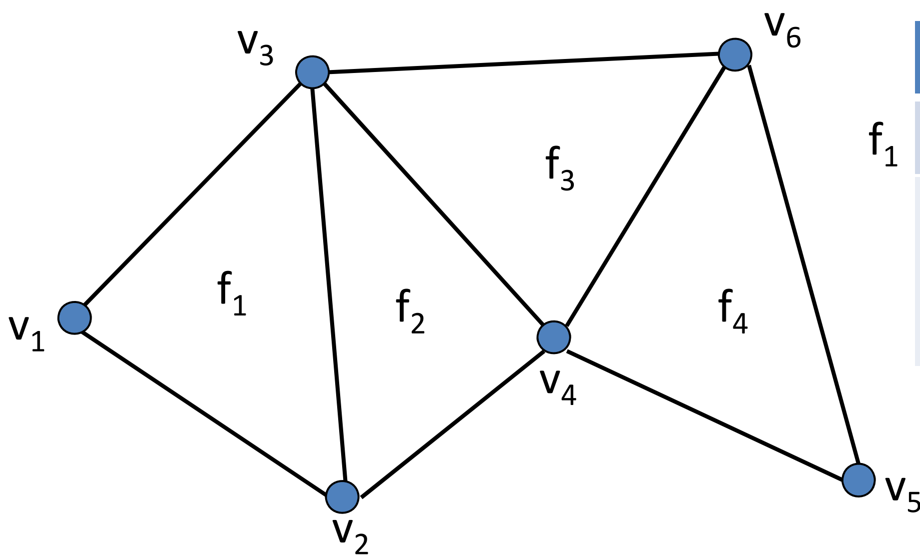
Shared Vertex

TRIANGLES		
Vertex Index	Vertex Index	Vertex Index
2	1	3
	.	
	.	
	.	

VERTICES
Vertex Coord.
[40 5 20]
[10 20 30]
[10 4 3]
.
.
.

- Connectivity
- No neighborhood

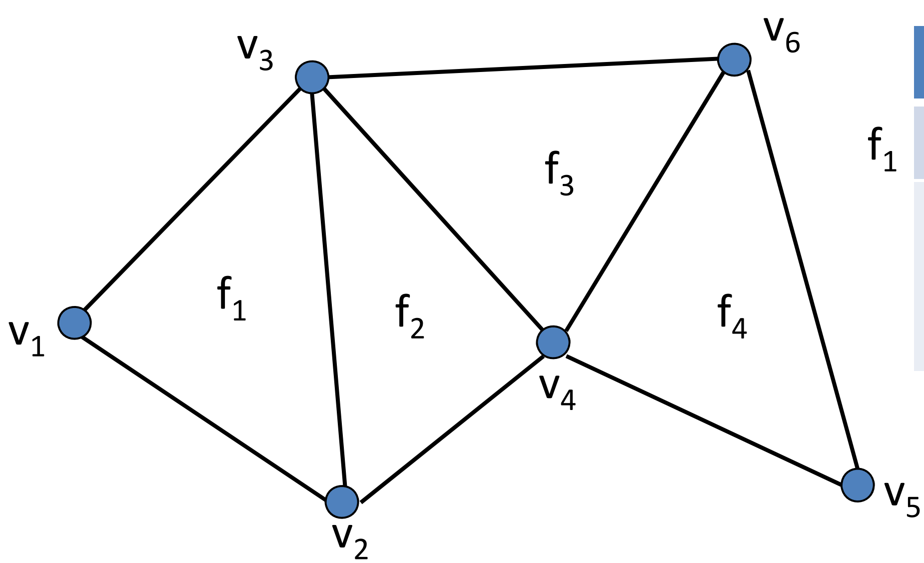
Shared Vertex



	TRIANGLES		
f_1	2	3	1
	.		
	.		
	.		

	VERTICES
v_1	[20 10 0]
v_2	[19 20 0]
v_3	[14 15 0]
	.
	.
	.

Shared Vertex

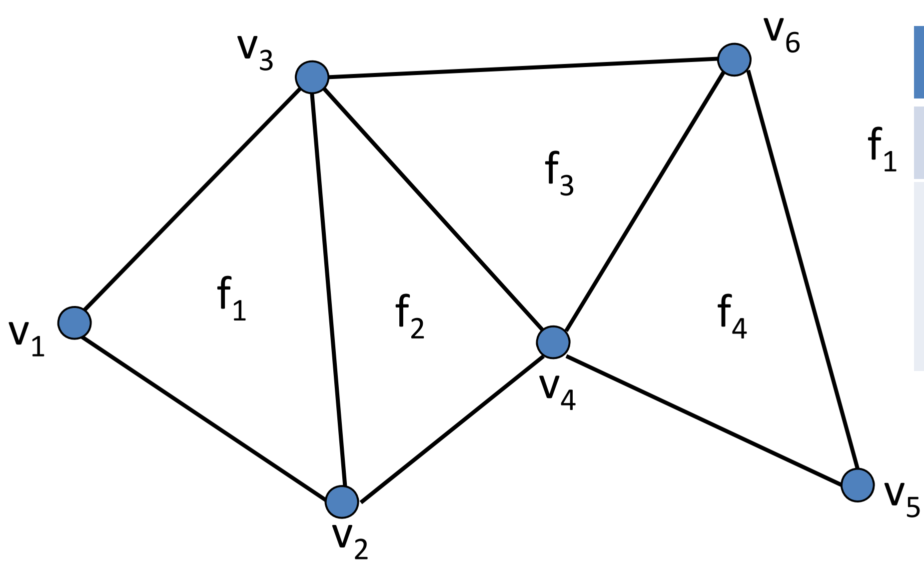


	TRIANGLES		
f_1	2	3	1
	.		
	.		
	.		

	VERTICES
v_1	[20 10 0]
v_2	[19 20 0]
v_3	[14 15 0]
	.
	.
	.

- **What are the vertices of face f_1?**
  - ***O*(1) – first triplet from face list**

Shared Vertex

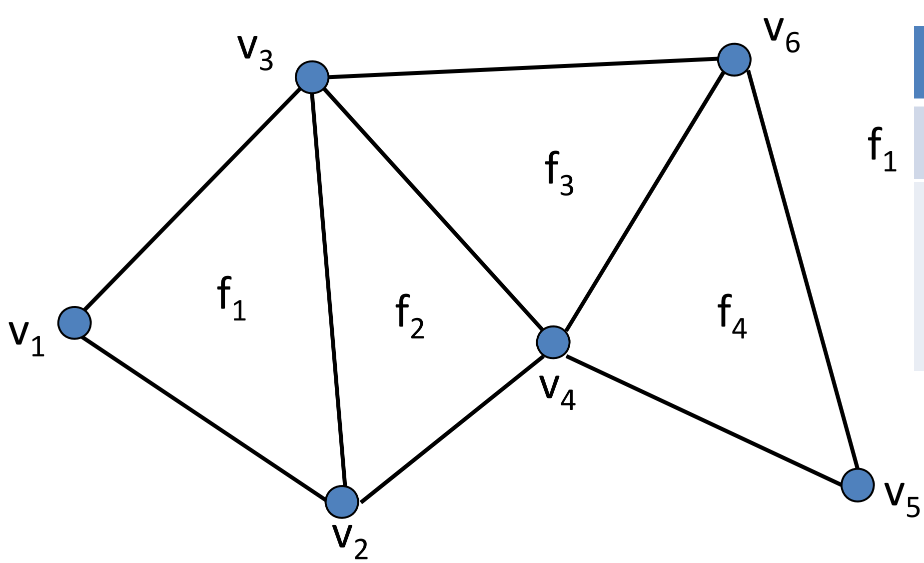


	TRIANGLES		
f_1	2	3	1
	.		
	.		
	.		

	VERTICES
v_1	[20 10 0]
v_2	[19 20 0]
v_3	[14 15 0]
	.
	.
	.

- **What are the one-ring neighbors of v_3?**
  - **Requires a full pass over all vertices**

Shared Vertex



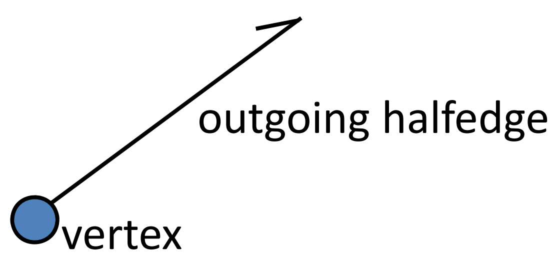
	TRIANGLES		
f_1	2	3	1
		.	
		.	
		.	

	VERTICES
v_1	[20 10 0]
v_2	[19 20 0]
v_3	[14 15 0]
	.
	.
	.

- **Are vertices v_1 and v_5 adjacent?**
  - **Requires a full pass over all faces**

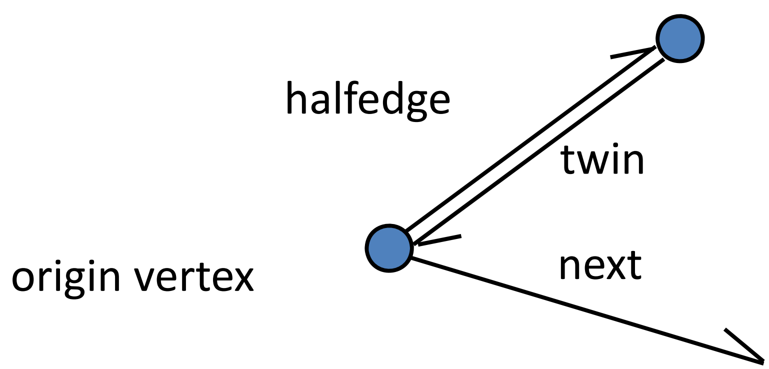
Half Edge Data Structure

- Vertex stores
  - Position
  - 1 outgoing halfedge



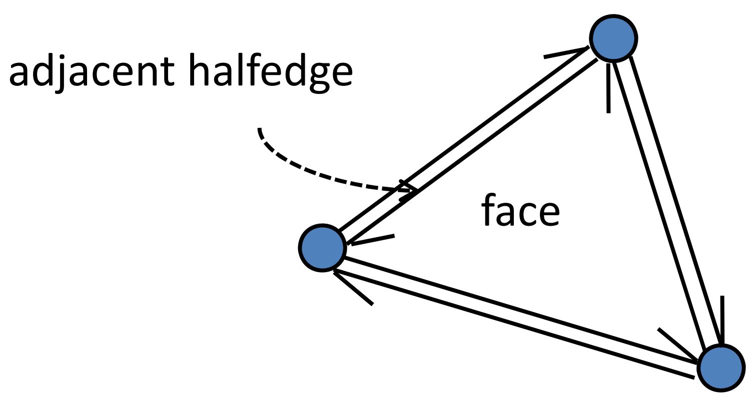
Half Edge Data Structure

- Halfedge stores
  - 1 origin vertex index
  - 1 incident face index
  - next, prev, twin halfedge indices



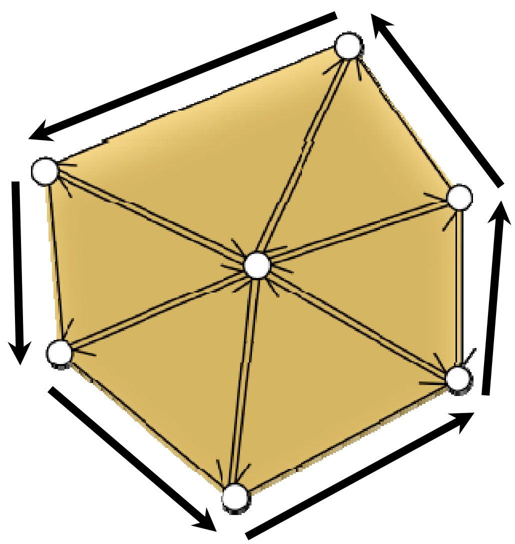
Half Edge Data Structure

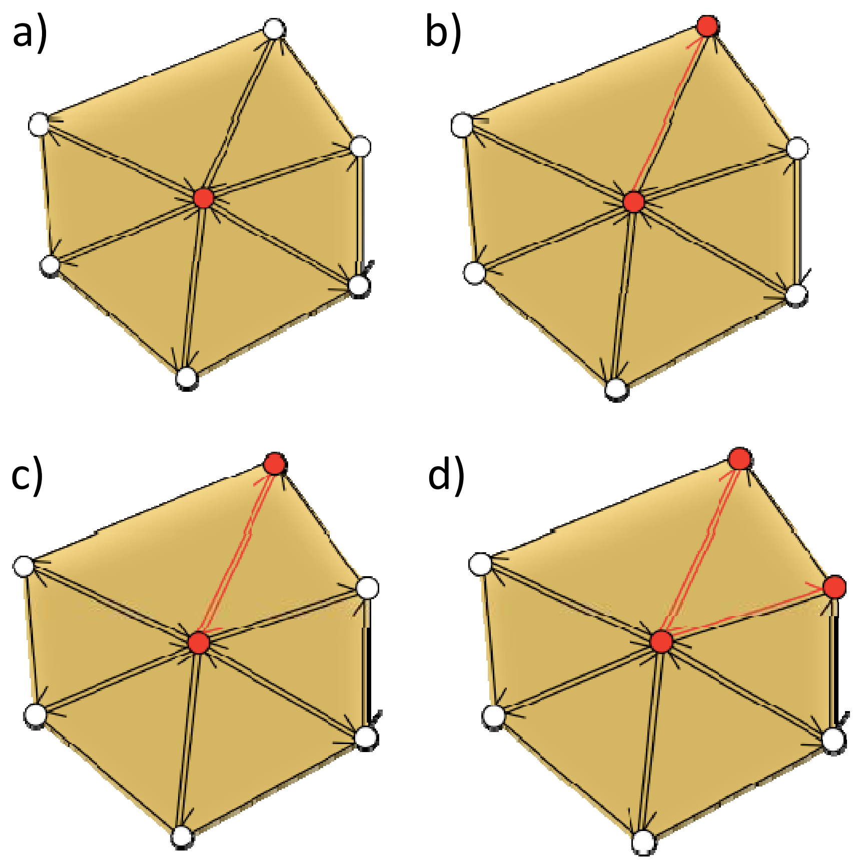
- Face stores
  - 1 adjacent halfedge index



Half Edge Data Structure

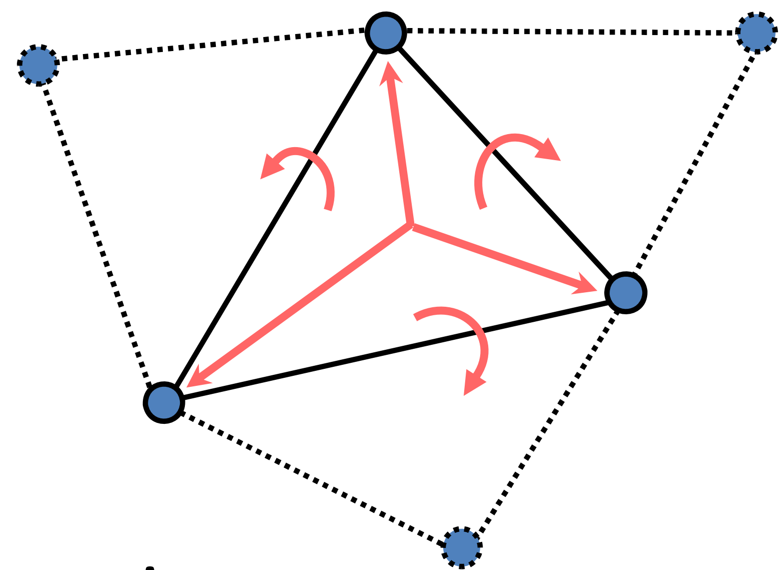
- Neighborhood Traversal





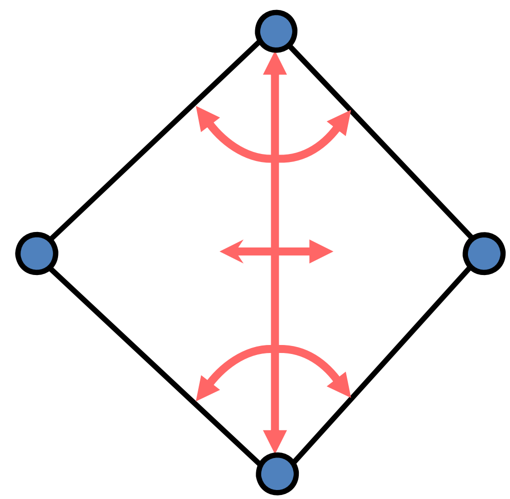
Face Based Connectivity

- Vertex:
  - position
  - 1 adjacent face index
- Face:
  - 3 vertex indices
  - 3 neighboring face indices
- No (explicit) edge information

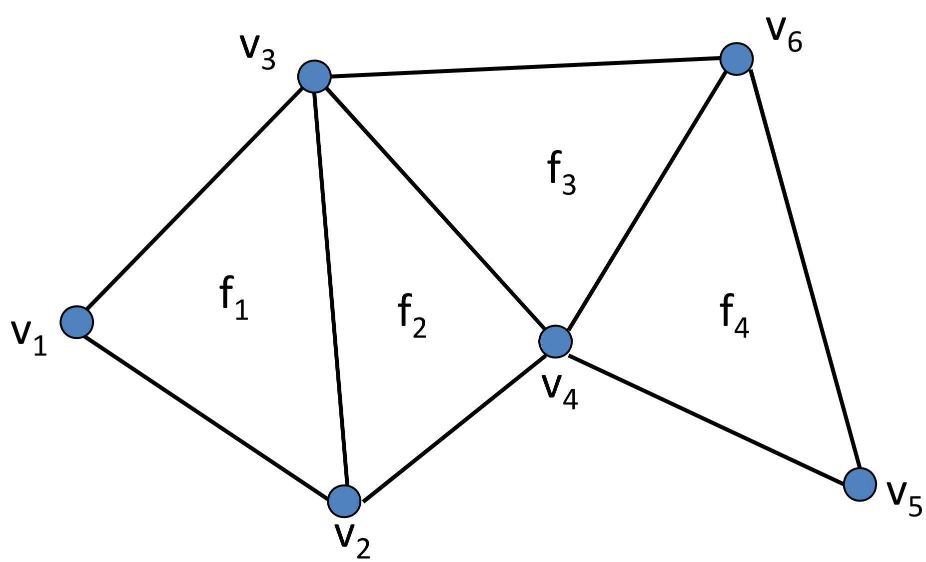


Edge Based Connectivity

- Vertex
  - position
  - 1 adjacent edge index
- Edge
  - 2 vertex indices
  - 2 neighboring face indices
  - 4 edges
- Face
  - 1 edge index
- No edge orientation information



Adjacency Matrix



A =

	v_1	v_2	v_3	v_4	v_5	v_6
v_1		1	1			
v_2	1		1	1		
v_3	1	1		1		1
v_4		1	1		1	1
v_5				1		1
v_6			1	1	1	

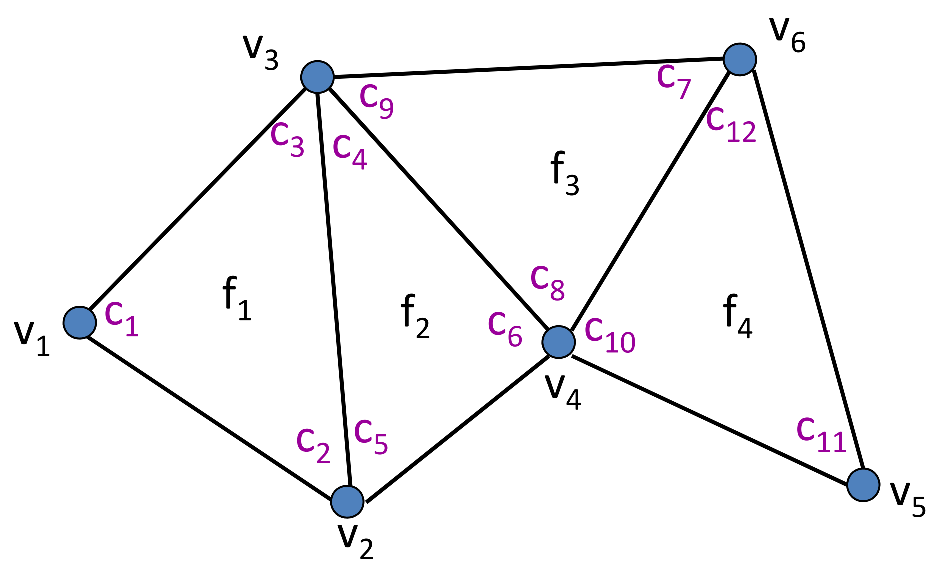
- Adjacency Matrix "A"
- If there is an edge between v_i & v_j then A_{ij}= 1

Adjacency Matrix

- Symmetric for undirected simple graphs
- $(A^n)_{ij}$= # paths of length n from v_i to v_j
- Pros:
  - Can represent non-manifold meshes
- Cons:
  - No connection between a vertex and its adjacent faces

Corner Table

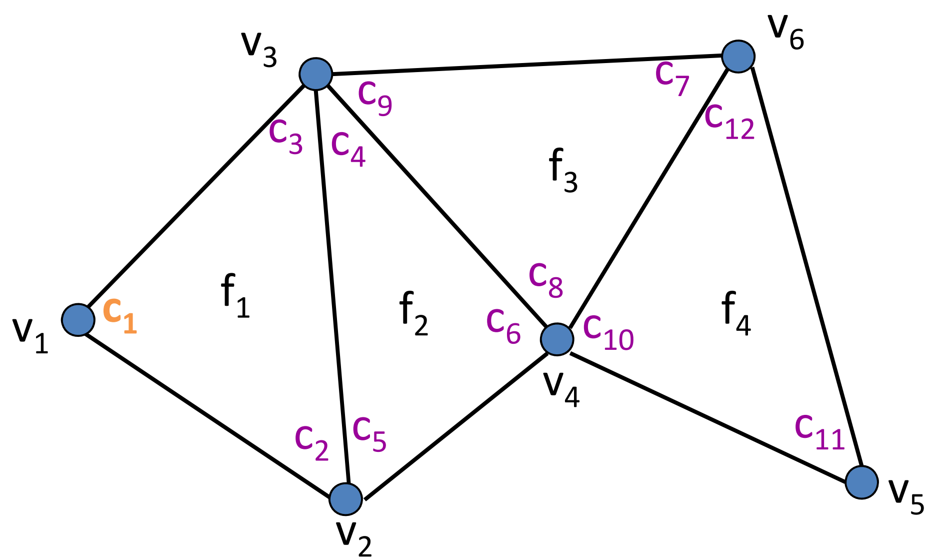
- Corner is a vertex with one of its indicent triangles



Corner Table

- Corner is a vertex with one of its indicent triangles

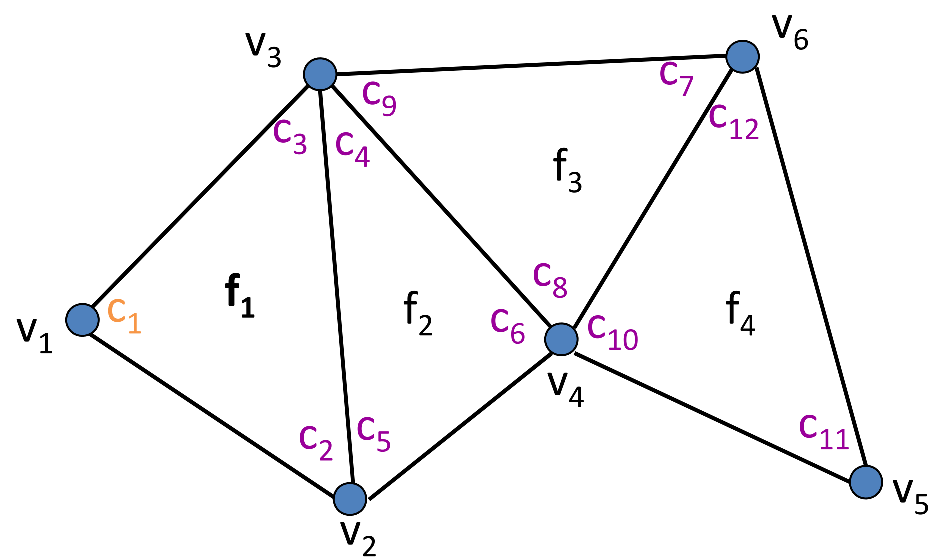
Corner – c



Corner Table

- Corner is a vertex with one of its indicent triangles

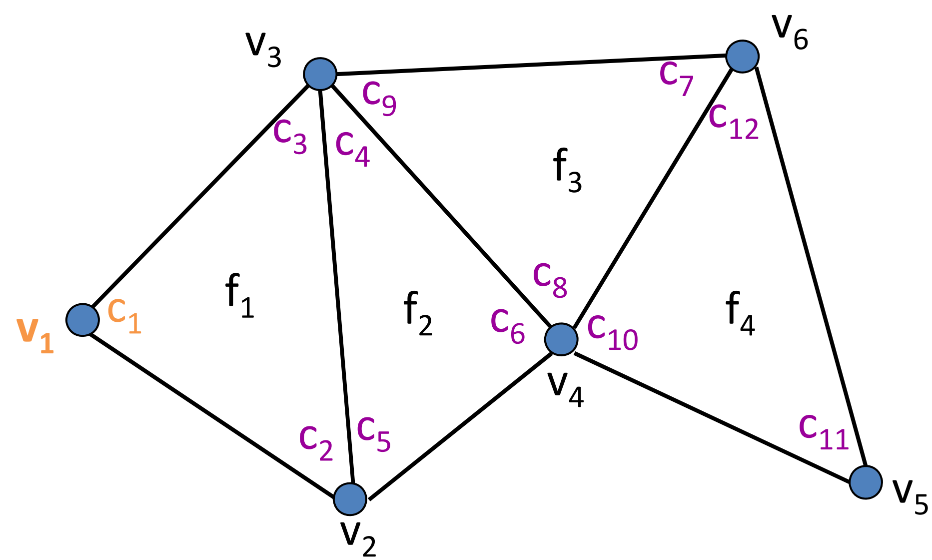
Corner – c
Triangle – c.t



Corner Table

- Corner is a vertex with one of its indicent triangles

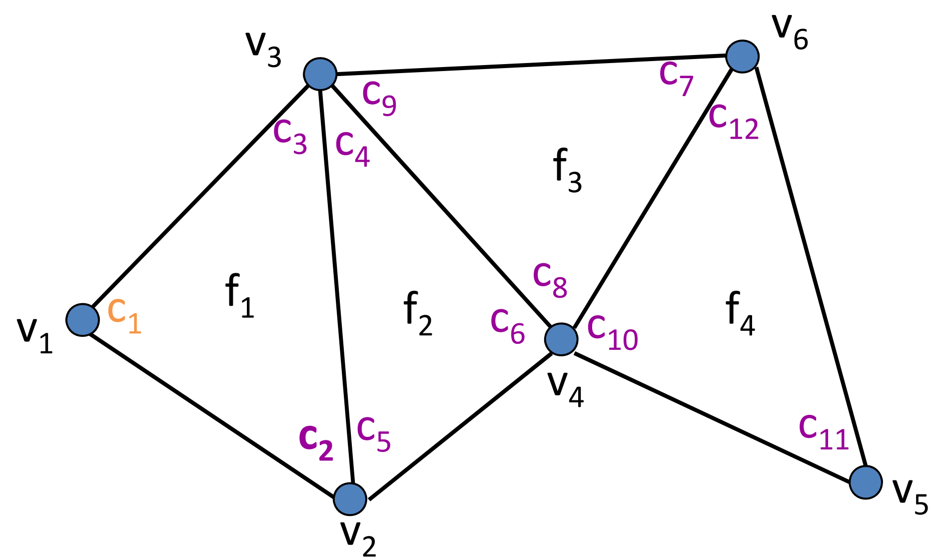
Corner – c
Triangle – c.t
Vertex – c.v



Corner Table

- Corner is a vertex with one of its indicent triangles

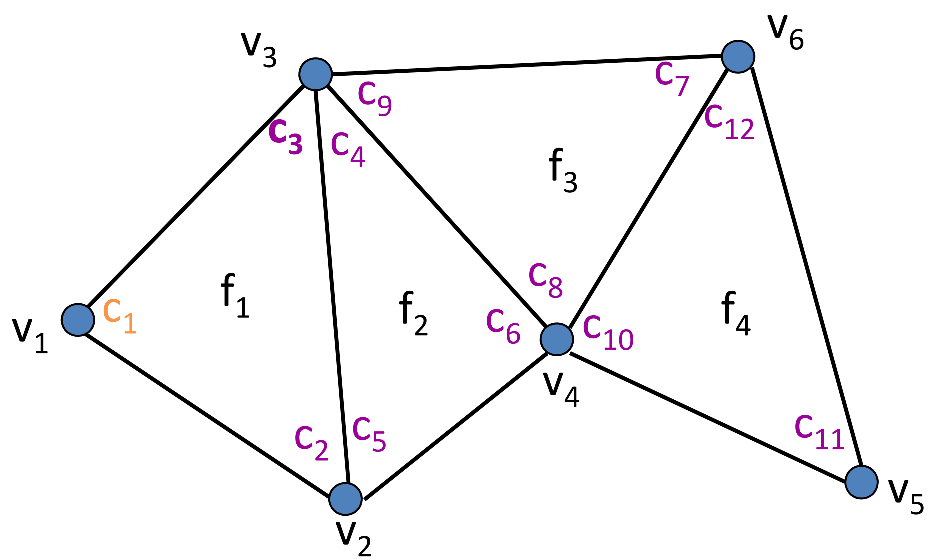
Corner – c
Triangle – c.t
Vertex – c.v
Next corner in c.t (ccw) – c.n



Corner Table

- Corner is a vertex with one of its indicent triangles

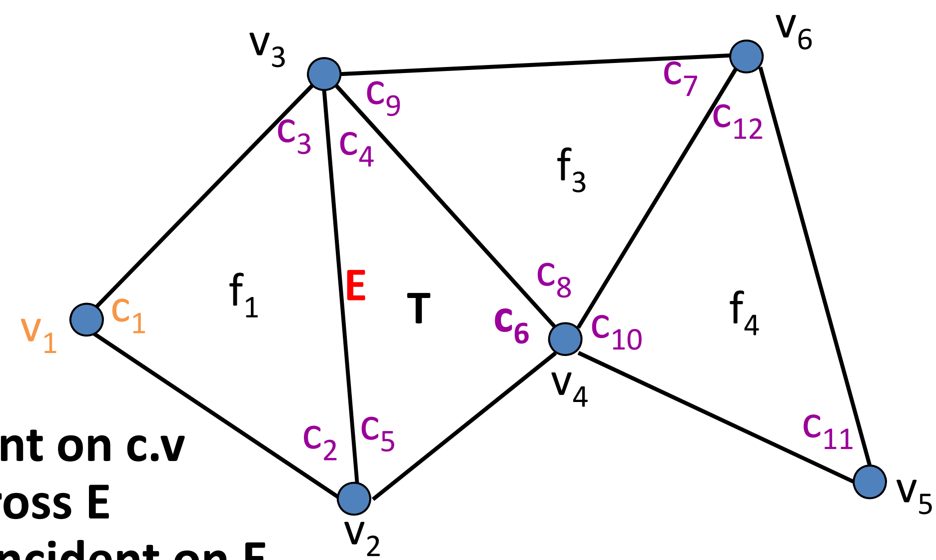
Corner – c
Triangle – c.t
Vertex – c.v
Next corner in c.t (ccw) – c.n
Previous corner – c.p (== c.n.n)



Corner Table

- Corner is a vertex with one of its indicent triangles

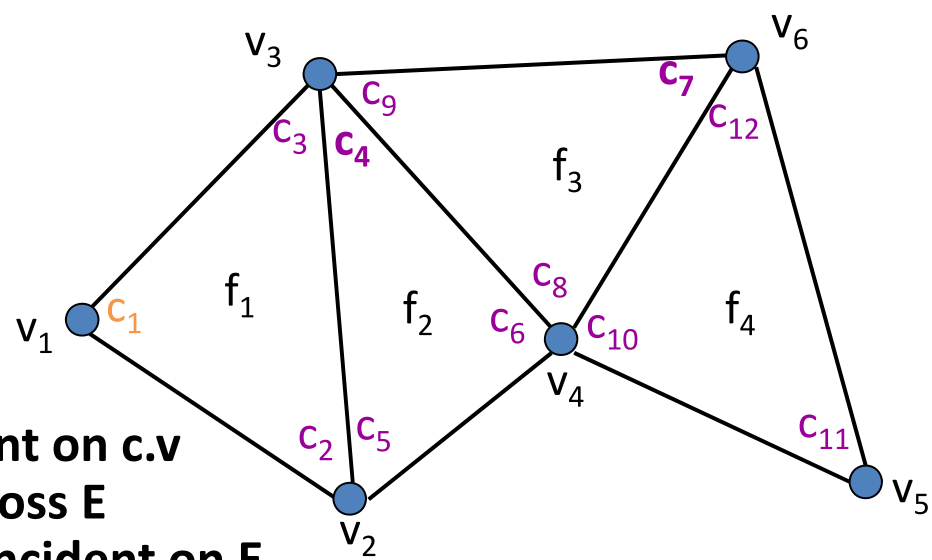
Corner – c
Triangle – c.t
Vertex – c.v
Next corner in c.t (ccw) – c.n
Previous corner – c.p (== c.n.n)
Corner opposite c – c.o
  Edge E opposite c not incident on c.v
  Triangle T adjacent to c.t across E
  c.o.v vertex of T that is not incident on E



Corner Table

- Corner is a vertex with one of its indicent triangles

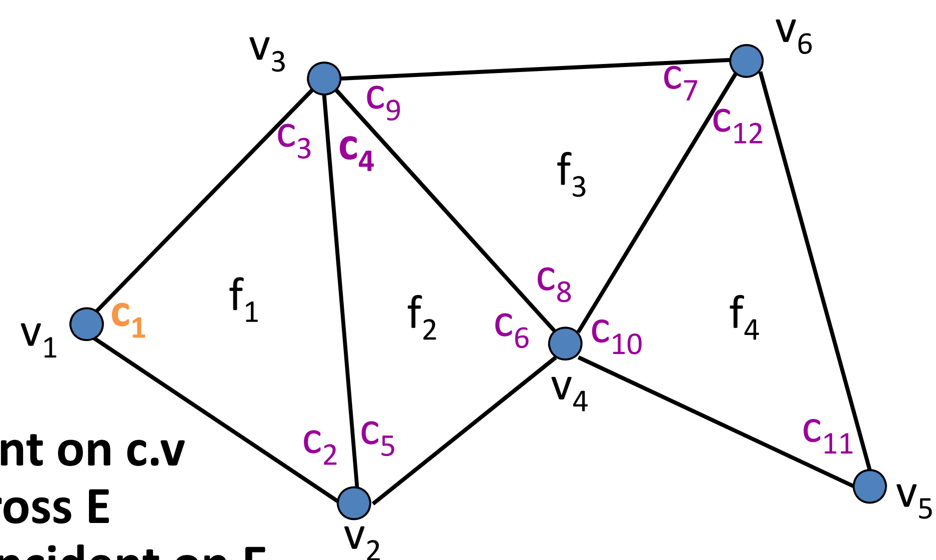
Corner – c
Triangle – c.t
Vertex – c.v
Next corner in c.t (ccw) – c.n
Previous corner – c.p (== c.n.n)
Corner opposite c – c.o
  Edge E opposite c not incident on c.v
  Triangle T adjacent to c.t across E
  c.o.v vertex of T that is not incident on E
Right corner – c.r – corner opposite c.n (== c.n.o)



Corner Table

- Corner is a vertex with one of its indicent triangles

Corner – c
Triangle – c.t
Vertex – c.v
Next corner in c.t (ccw) – c.n
Previous corner – c.p (== c.n.n)
Corner opposite c – c.o
  Edge E opposite c not incident on c.v
  Triangle T adjacent to c.t across E
  c.o.v vertex of T that is not incident on E
Right corner – c.r – corner opposite c.n (== c.n.o)
Left corner – c.l (== c.p.o == c.n.n.o)



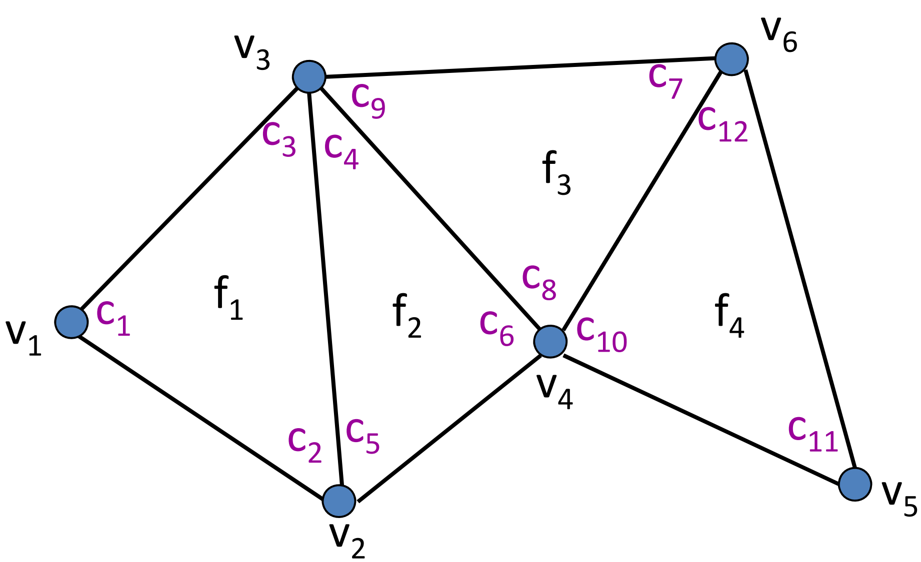
Corner Table

- Corner is a vertex with one of its indicent triangles

Corner – c
Triangle – c.t
Vertex – c.v
Next corner in c.t (ccw) – c.n
Previous corner – c.p (== c.n.n)
Corner opposite c – c.o
  Edge E opposite c not incident on c.v
  Triangle T adjacent to c.t across E
  c.o.v vertex of T that is not incident on E
Right corner – c.r – corner opposite c.n (== c.n.o)
Left corner – c.l (== c.p.o == c.n.n.o)

Corner Table

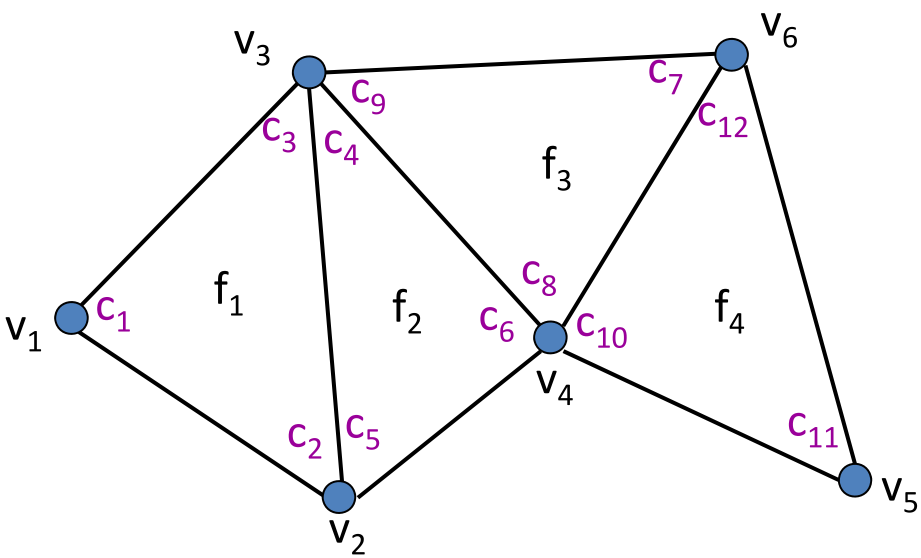
- Corner is a vertex with one of its indicent triangles



corner	c.v	c.t	c.n	c.p	c.o	c.r	c.l
c_1	v_1	f_1	c_2	c_3	c_6		

Corner Table

- Corner is a vertex with one of its indicent triangles



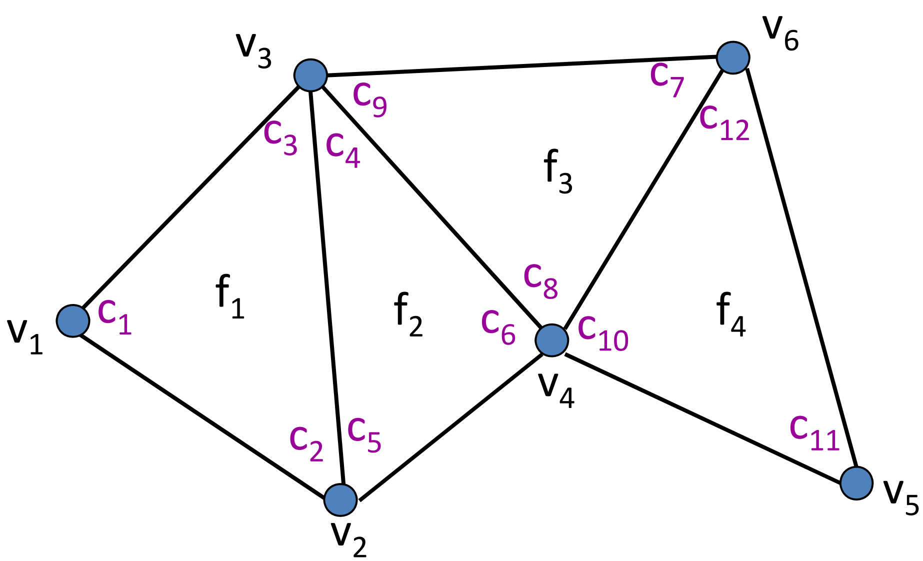
corner	c.v	c.t	c.n	c.p	c.o	c.r	c.l
c_1	v_1	f_1	c_2	c_3	c_6		
c_2	v_2	f_1	c_3	c_1			c_6
c_3	v_3	f_1	c_1	c_2		c_6	
c_4	v_3	f_2	c_5	c_6		c_7	c_1
c_5	v_2	f_2	c_6	c_4	c_7	c_1	
c_6	v_4	f_2	c_4	c_5	c_1		c_7

Corner Table

- Store:
  - Corner table
  - For each vertex – a list of all its corners
- Corner number j*3-2, j*3-1 and j*3 match face number j

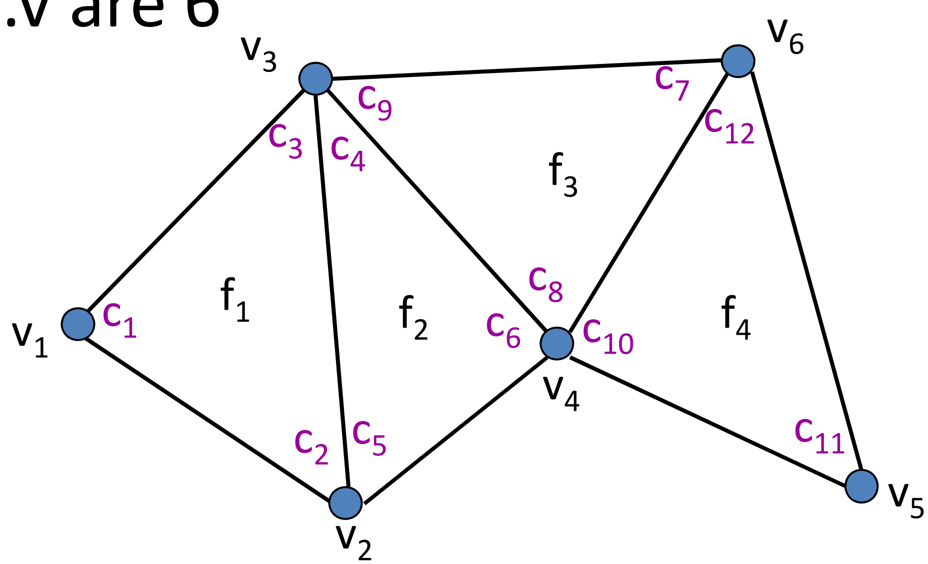
Corner Table

- What are the vertices of face #3?
  - Check c.v of corners 9, 8, 7



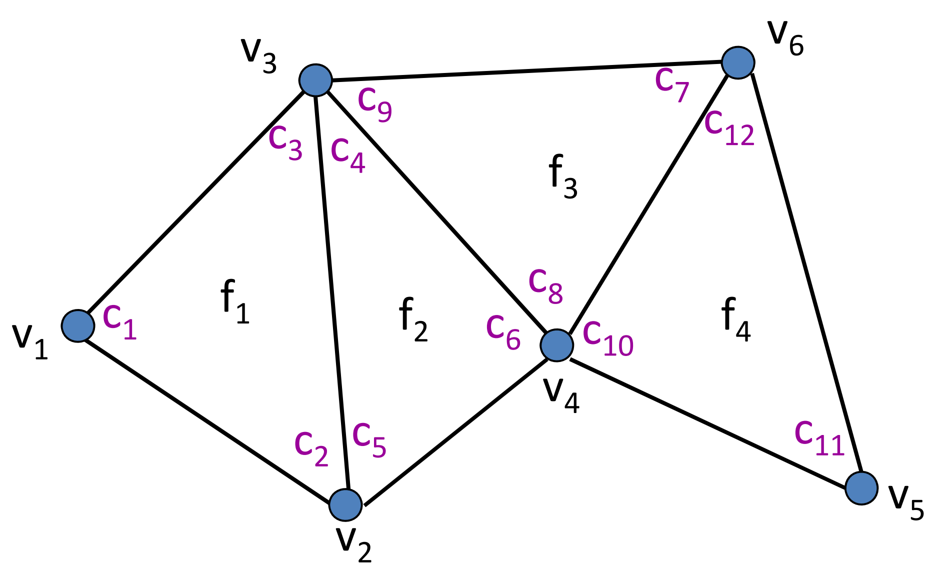
Corner Table

- Are vertices 2 and 6 adjacent?
  - Scan all corners of vertex 2, check if c.p.v or c.n.v are 6



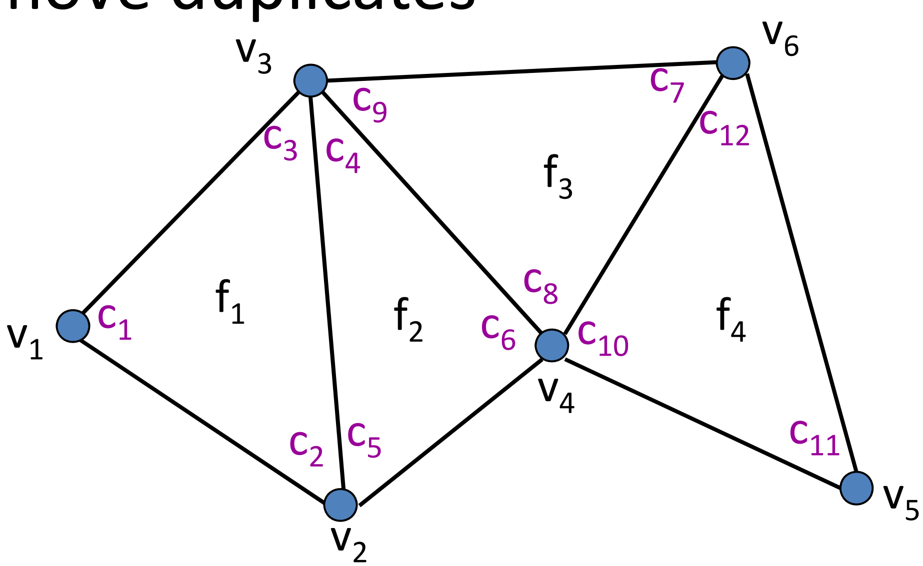
Corner Table

- Which faces are adjacent to vertex 3?
  - Check c.t of all corners of vertex 3



Corner Table

- One ring neighbors of vertex v_4?
  - Get the corners c_6 c_8 c_{10} of this vertex
  - Go to c_i.n.v and c_i.p.v for i = 6, 8, 10.
  - Remove duplicates



Corner Table

- Pros:
  - All queries in O(1) time
  - Most operations are O(1)
  - Convenient for rendering
- Cons:
  - Only triangular, manifold meshes
  - Redundancy