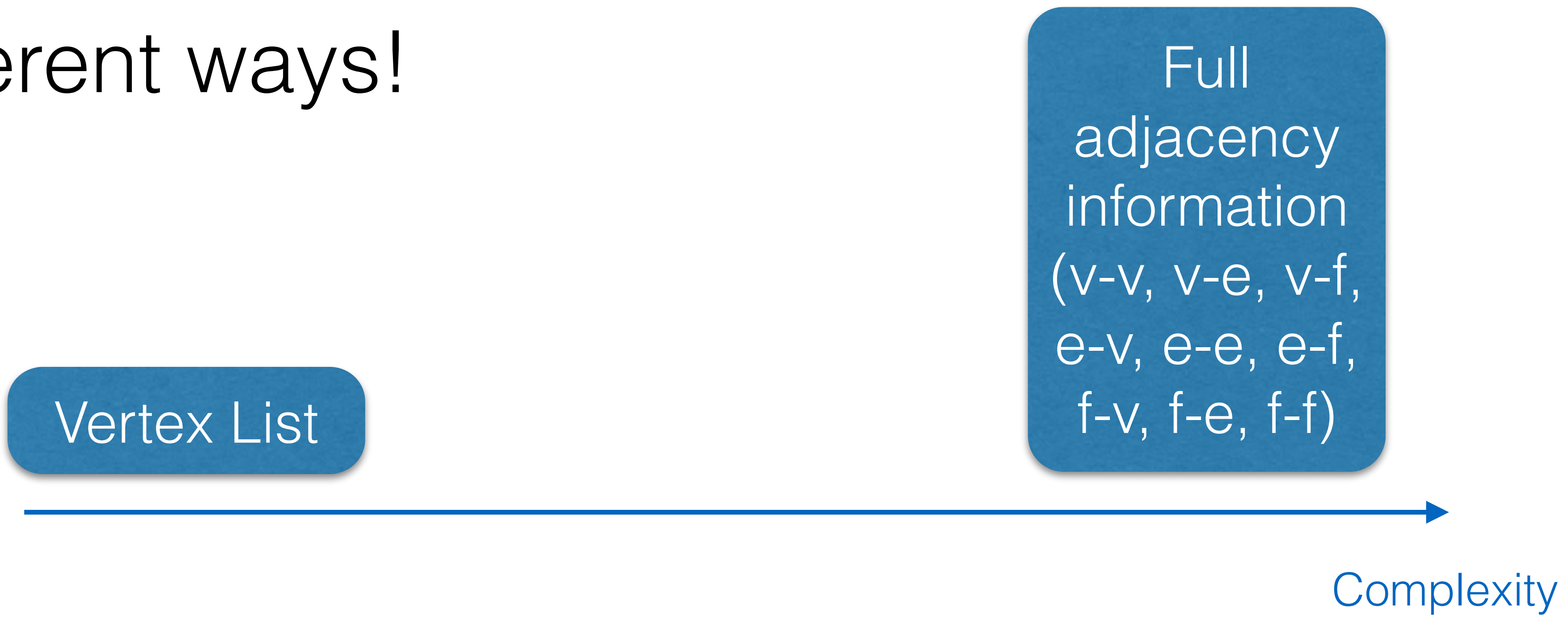
Assignment 2 & Half-Edge

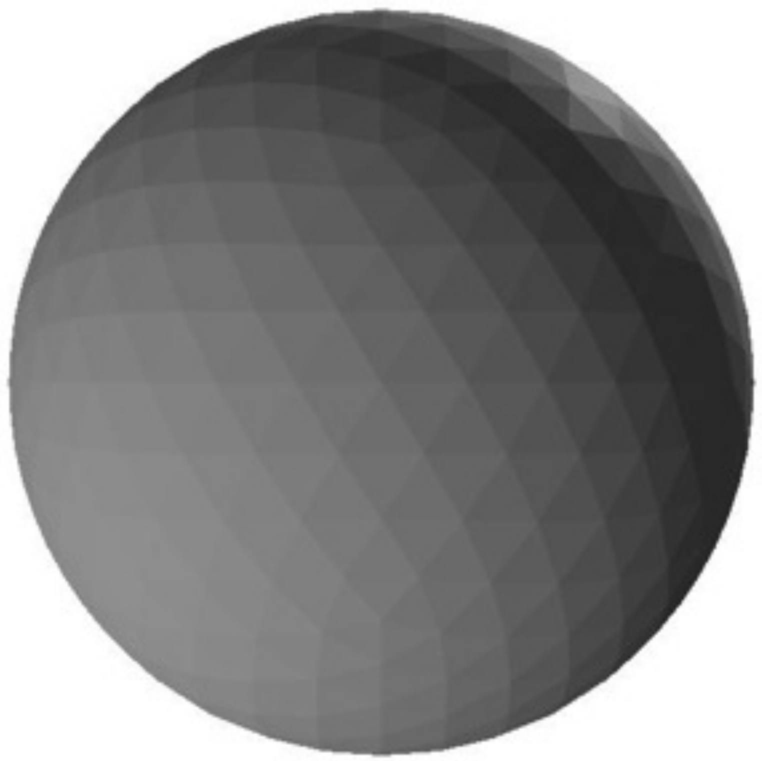
Mesh Representation

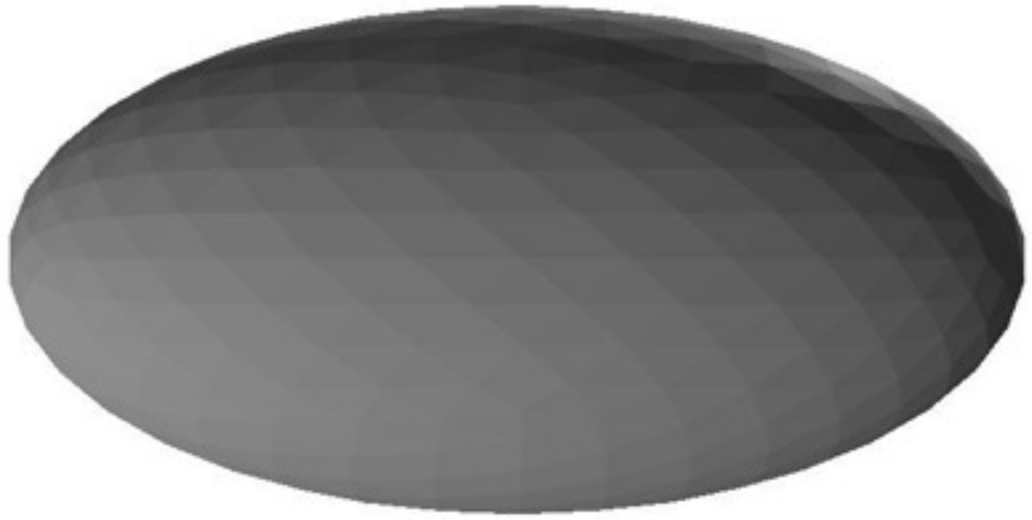
- Many different ways!


Vertex List
Full adjacency information (v-v, v-e, v-f, e-v, e-e, e-f, f-v, f-e, f-f)
Complexity

- Runtime / memory constraints
- Ease of coding
- Usage

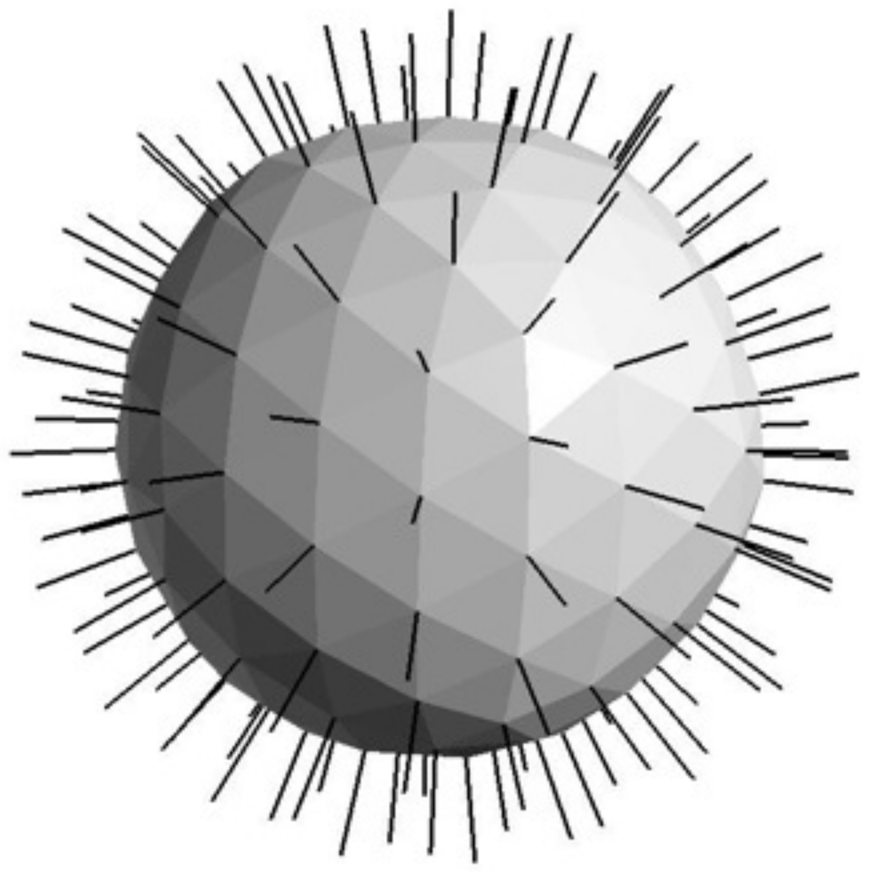
Scale





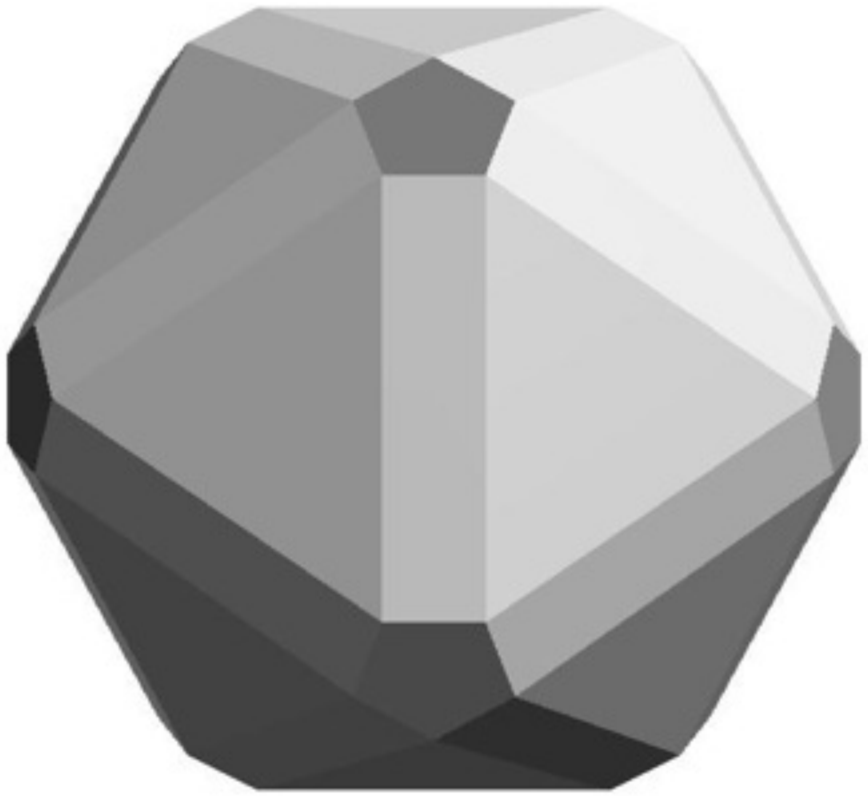
Need: vertex location

Compute Normals

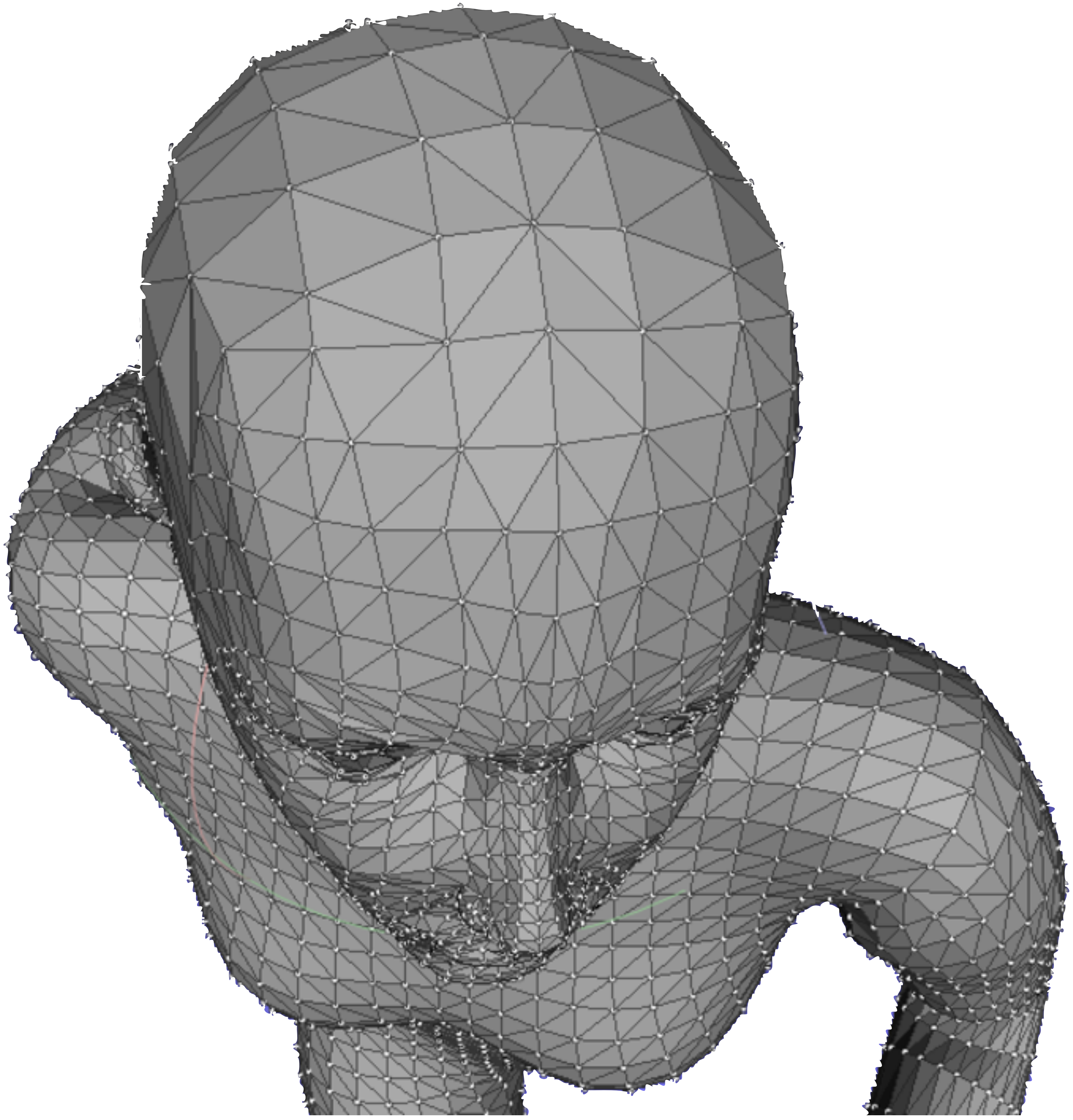


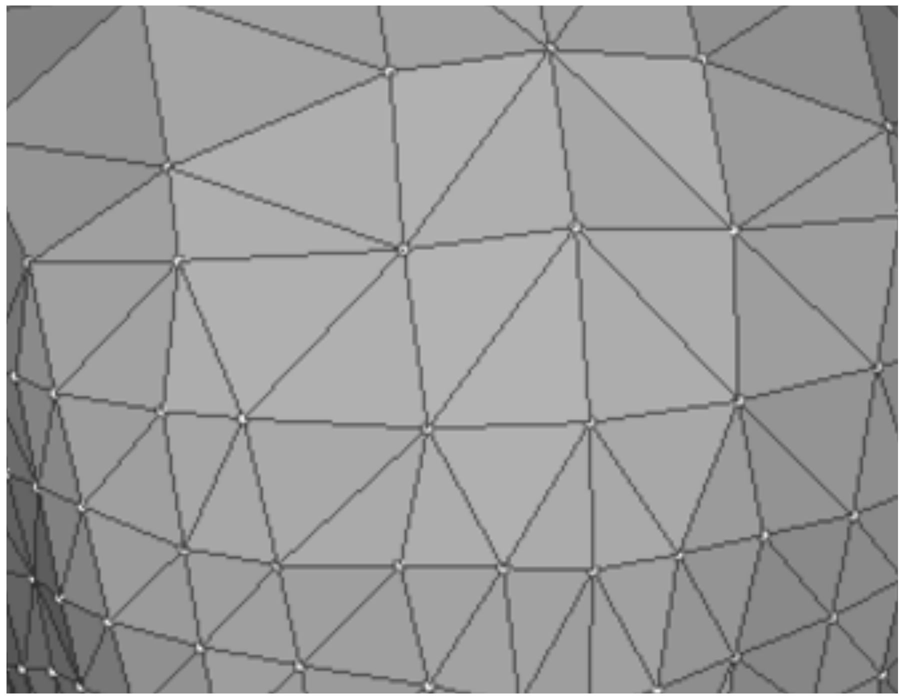
Need: v-f adjacency information

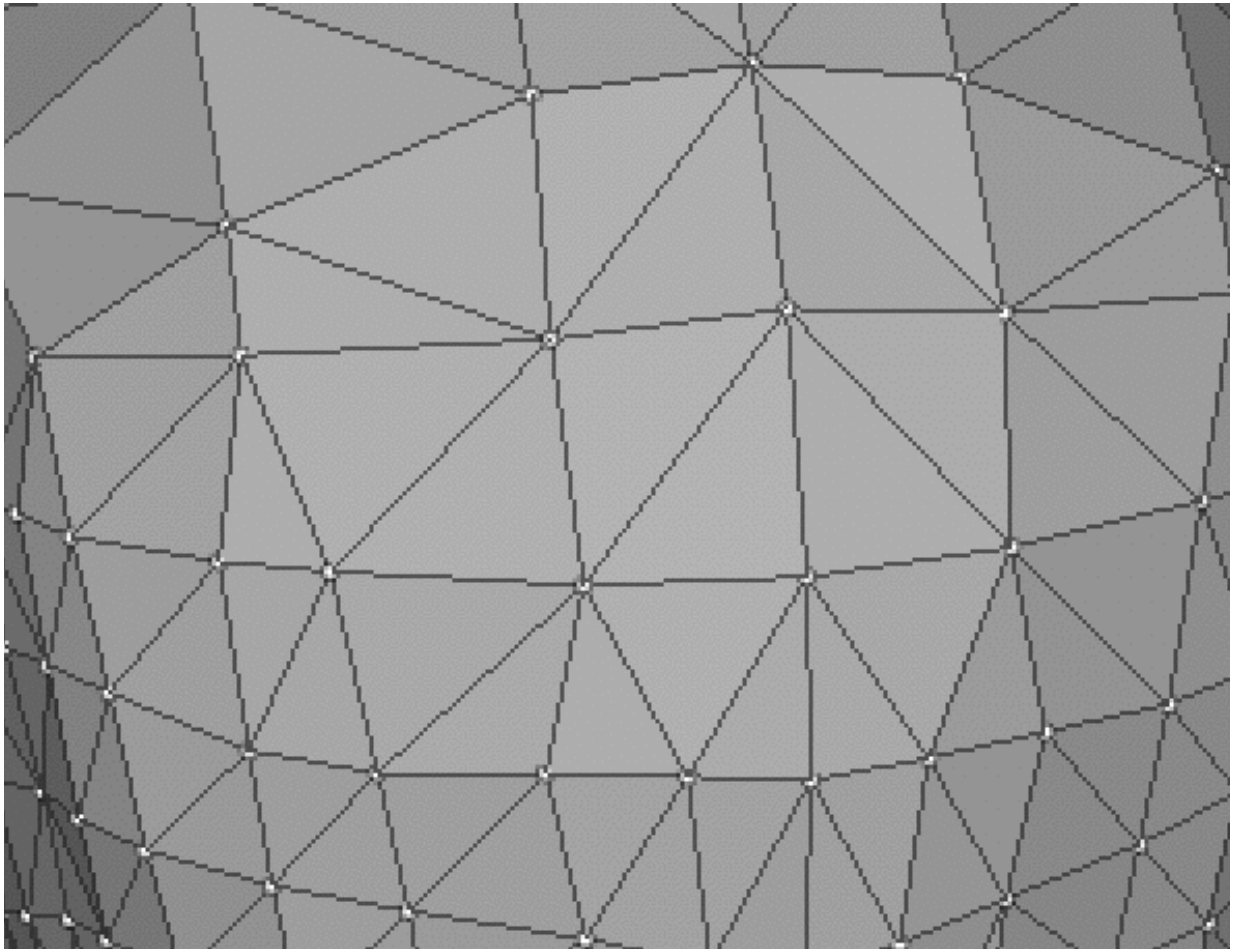
Bevel

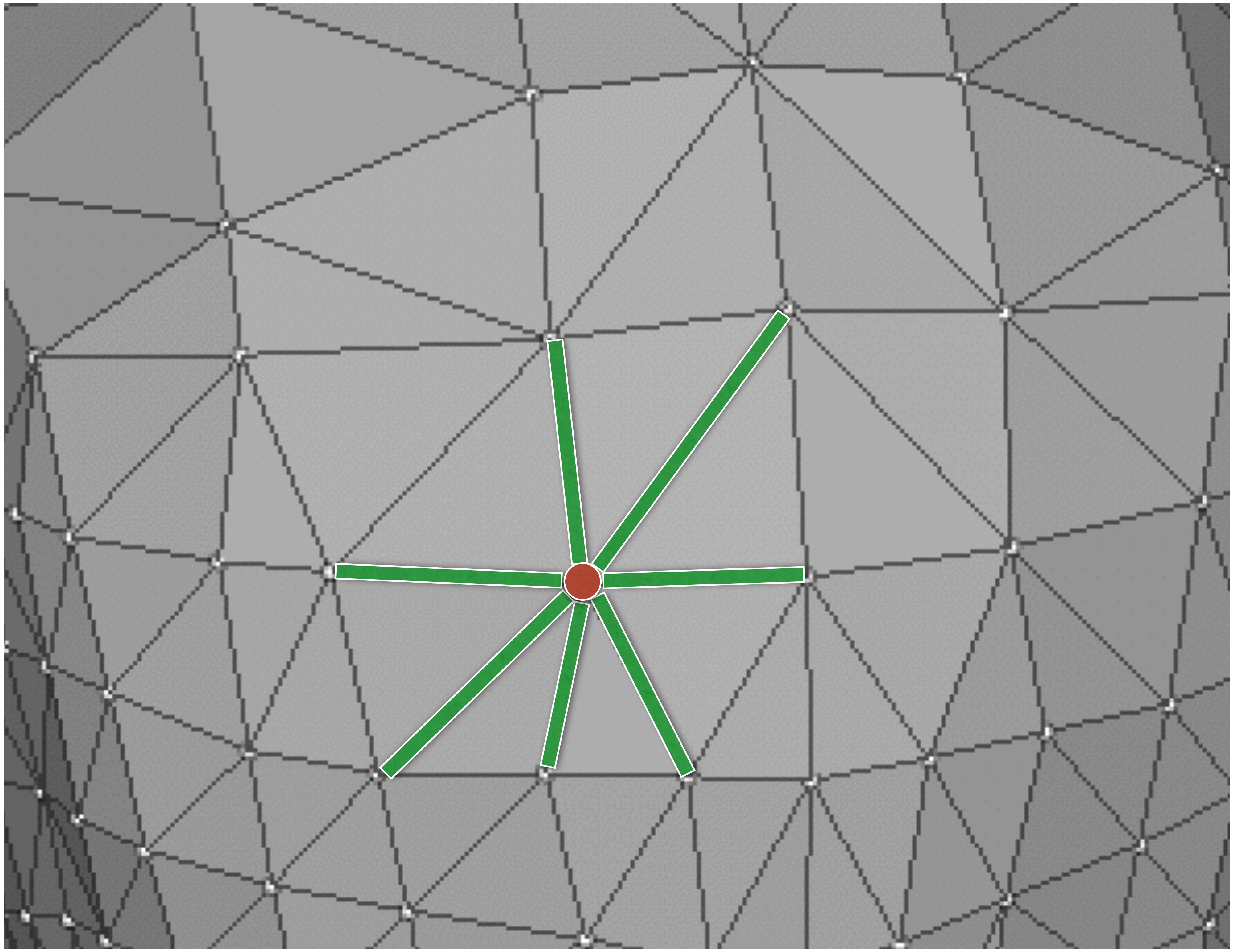


Need: full adjacency information

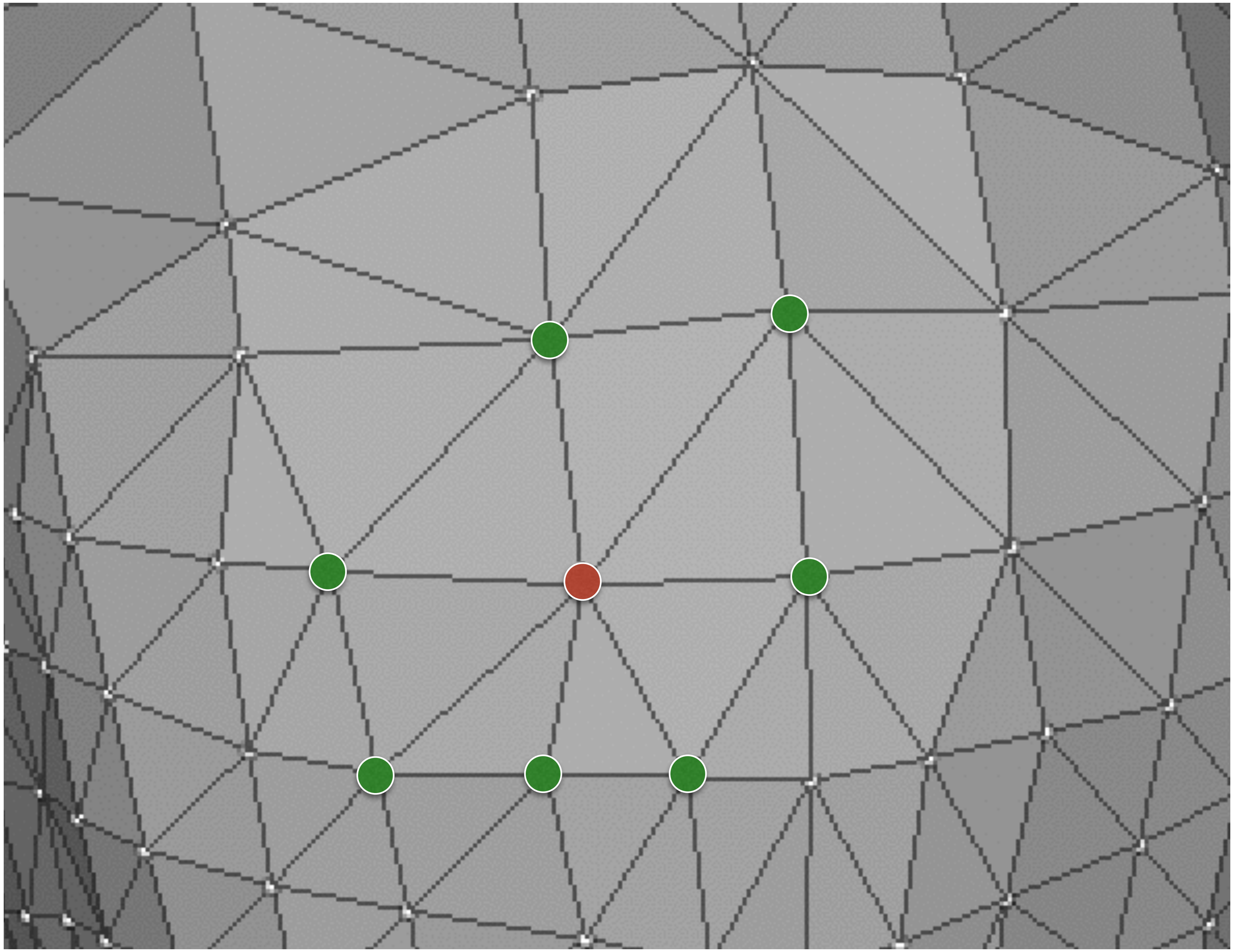








Runtime: O(# adjacent edges)



Runtime: O(# adjacent vertices)

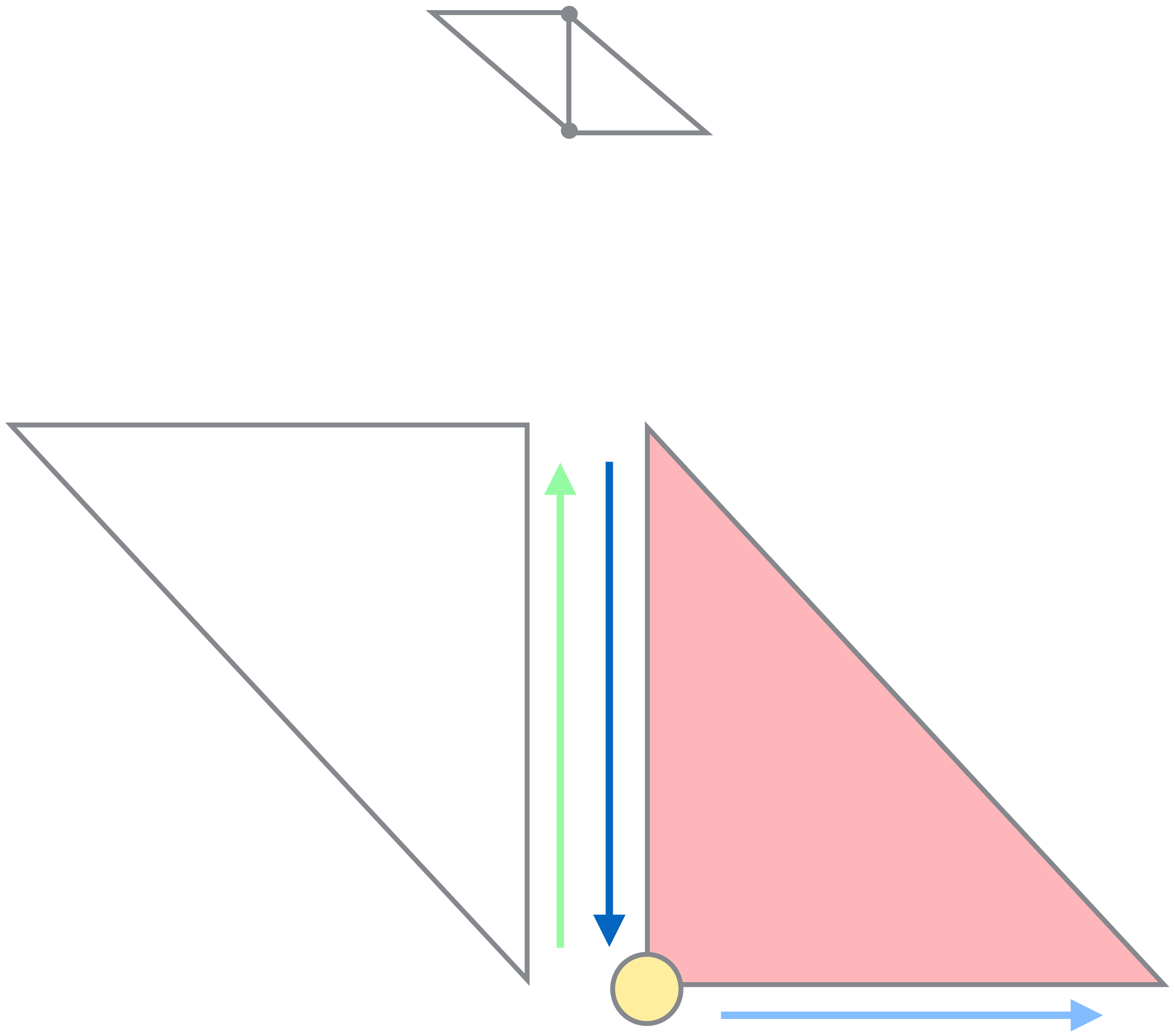
Half-Edge Data Structure

Half Edge	Vertex	Face
End Vertex	Location	(Some) Half Edge
Other Half	(Some) Half Edge starting at Vertex	…
Face	…	
Next Half Edge		
…		

Half-Edge Data Structure

Half Edge

- End Vertex
- Other Half
- Face
- Next Half Edge



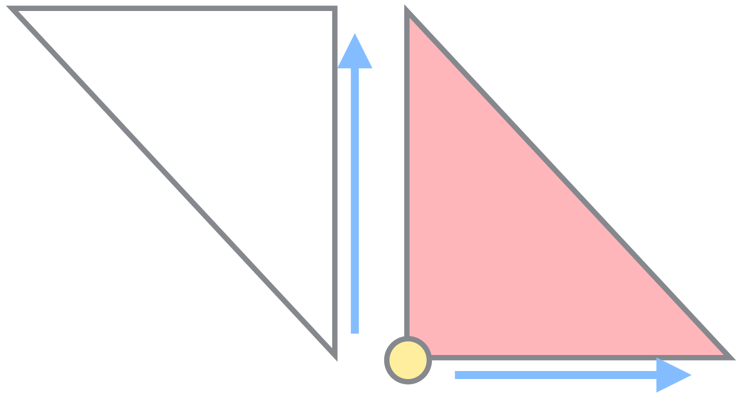
Half-Edge Data Structure

Vertex

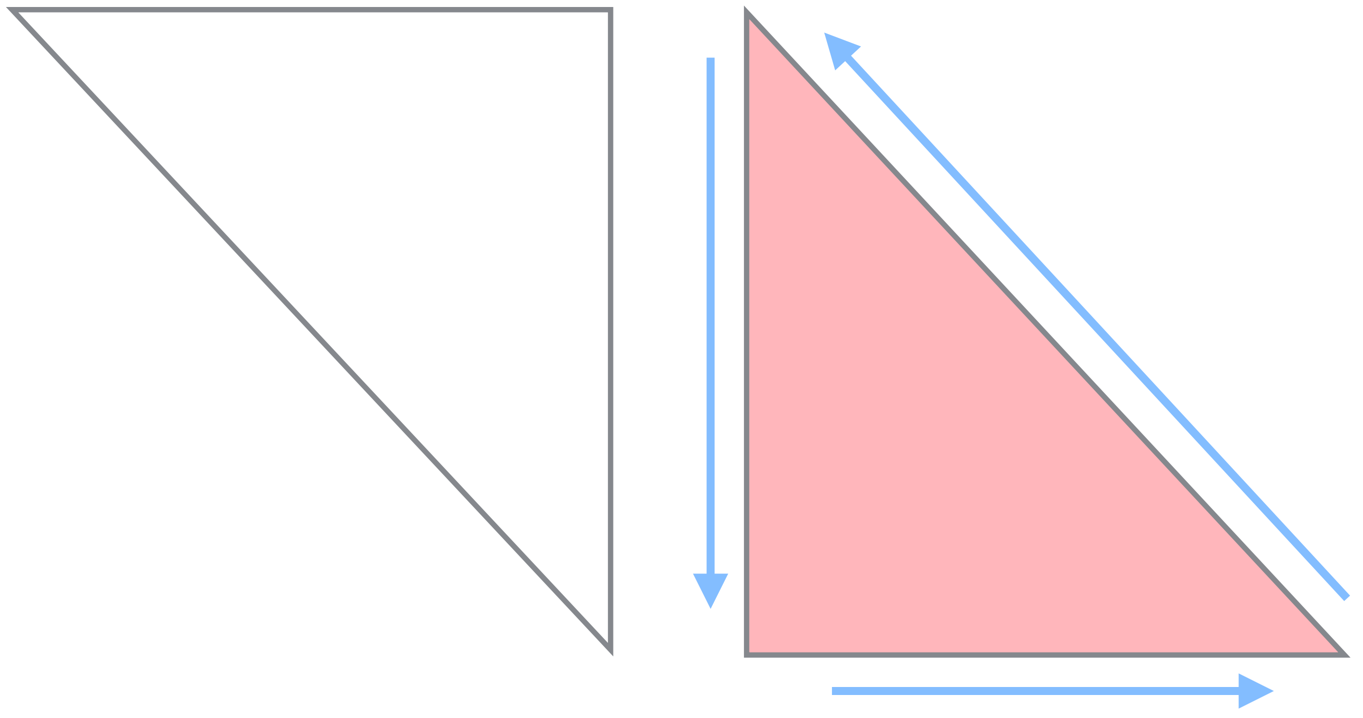
Location

(Some) Half Edge
starting at Vertex

...



Half-Edge Data Structure



Face

(Some) Half Edge

. . .

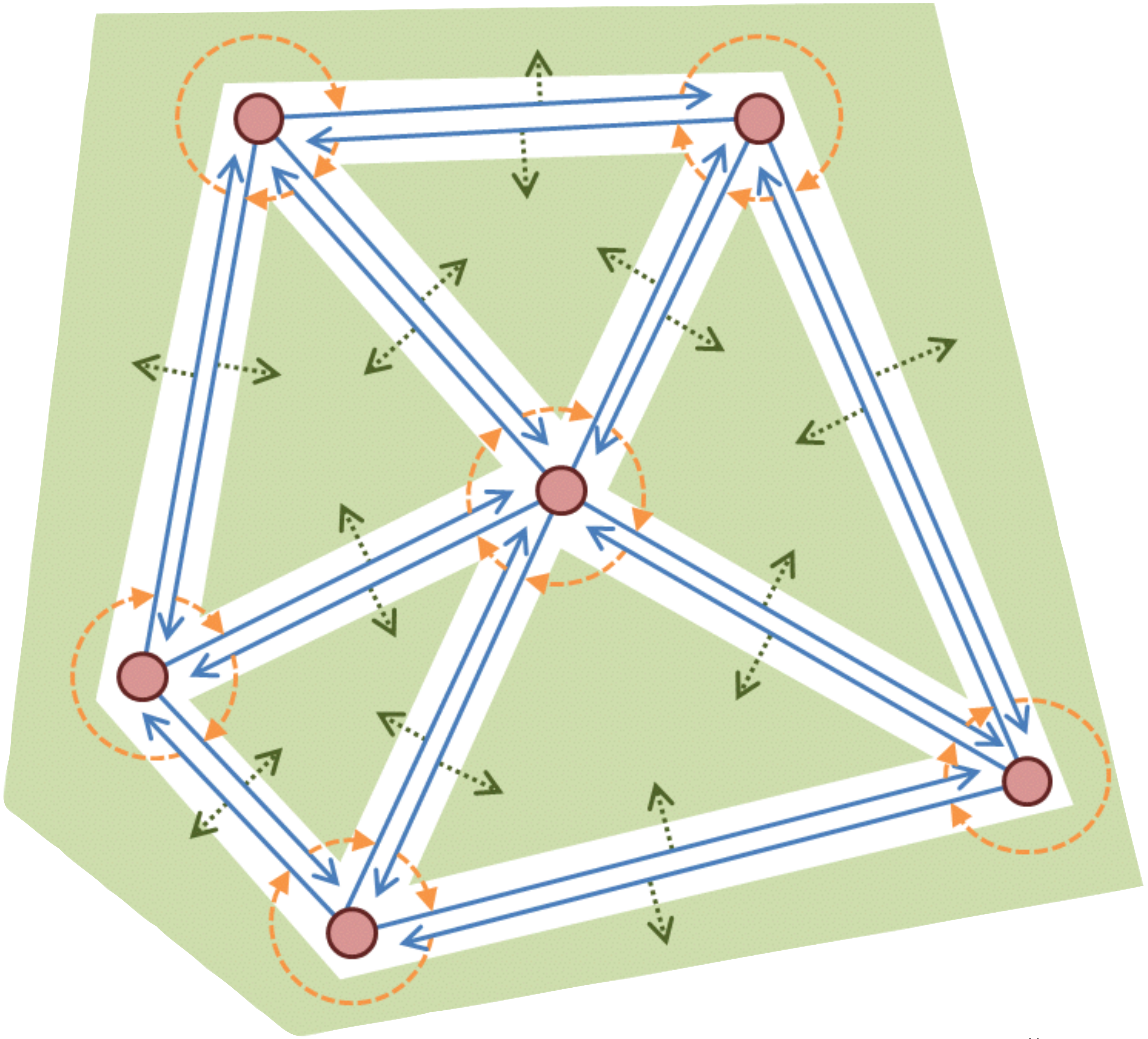
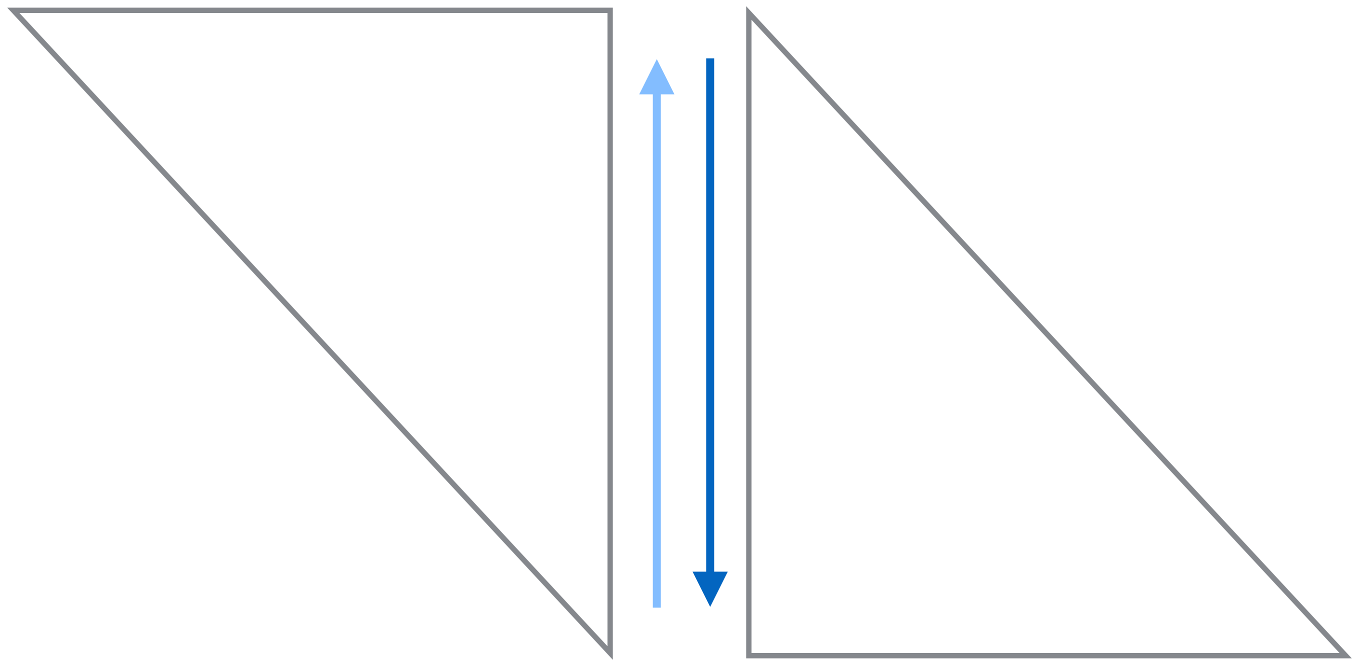


Image credit: http://leonardofischer.com/

Iterating the Data Structure

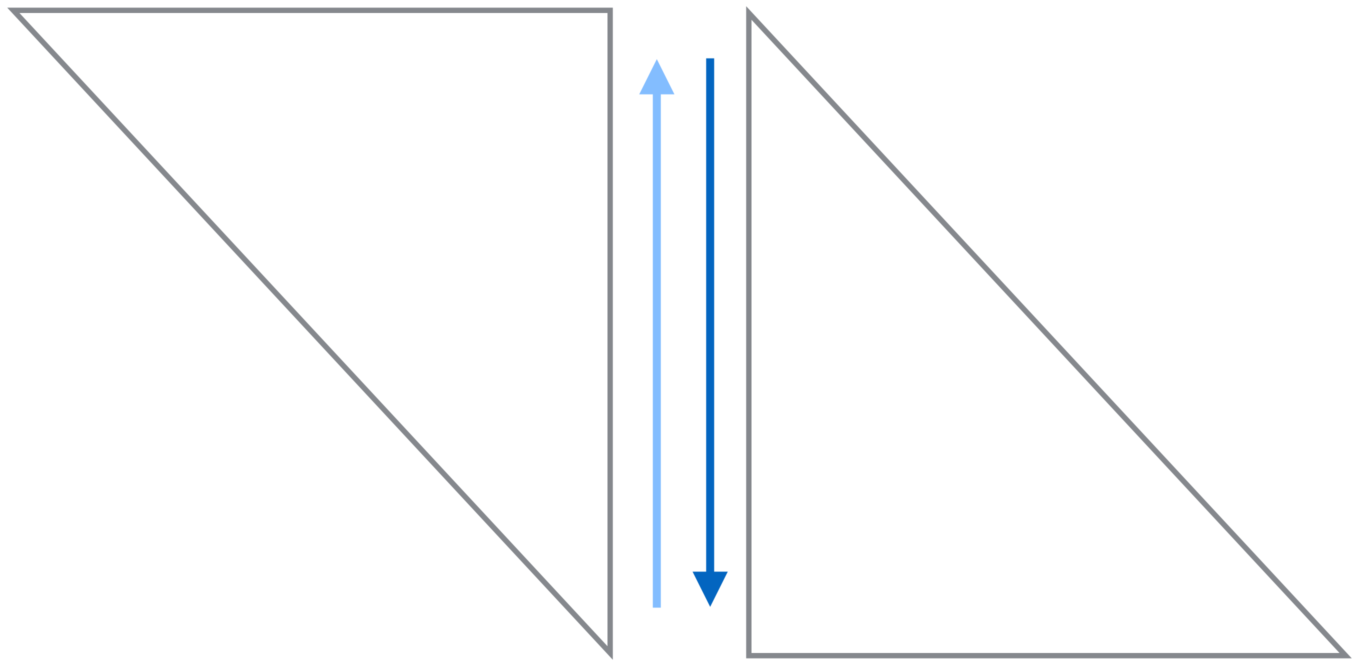
Edge-Vertex Neighbors



```
origHalfEdge->endVertex
origHalfEdge->dual->endVertex
```

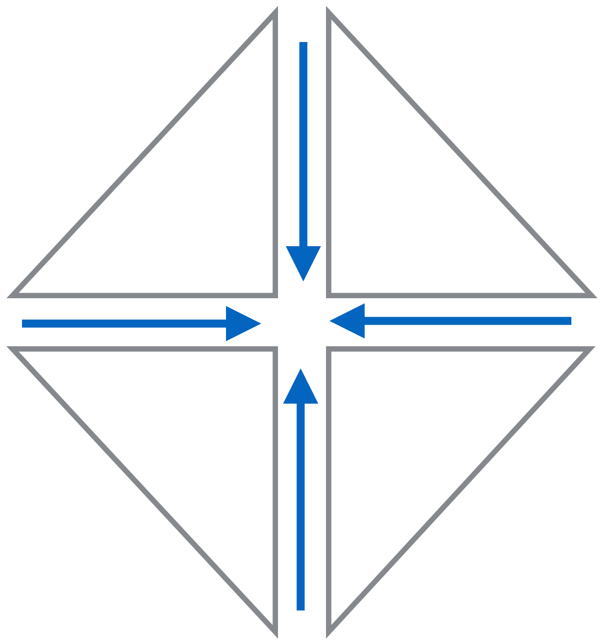
*assuming dual exists

Edge-Face Neighbors



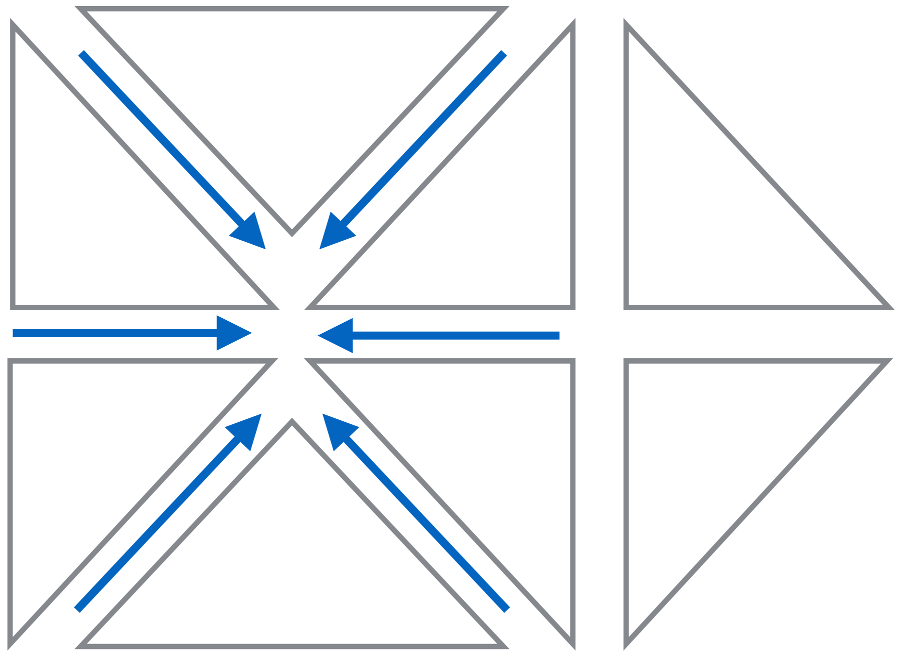
```
origHalfEdge->face
origHalfEdge->dual->face
```

Edge-Edge Neighbors



```
current = origHalfEdge;
do {
  current = current->next->dual;
} while (current != origHalfEdge);
```

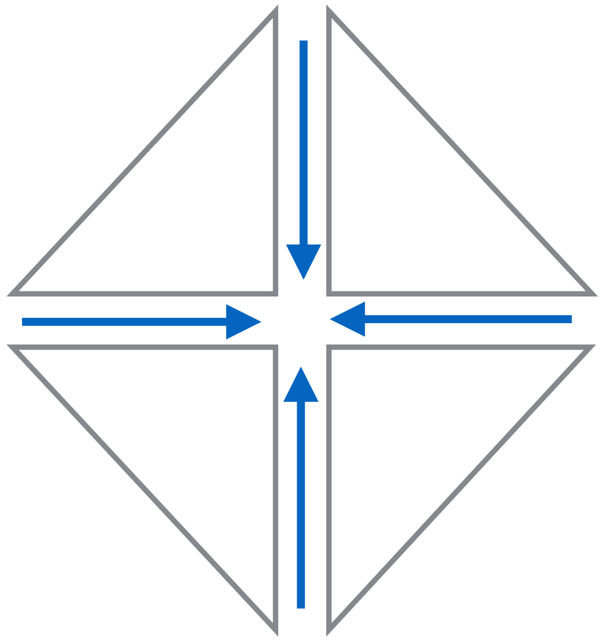
Edge-Edge Neighbors



Same code for the other side

```
current = origHalfEdge;
do {
   current = current->next->dual;
} while (current != origHalfEdge);
```

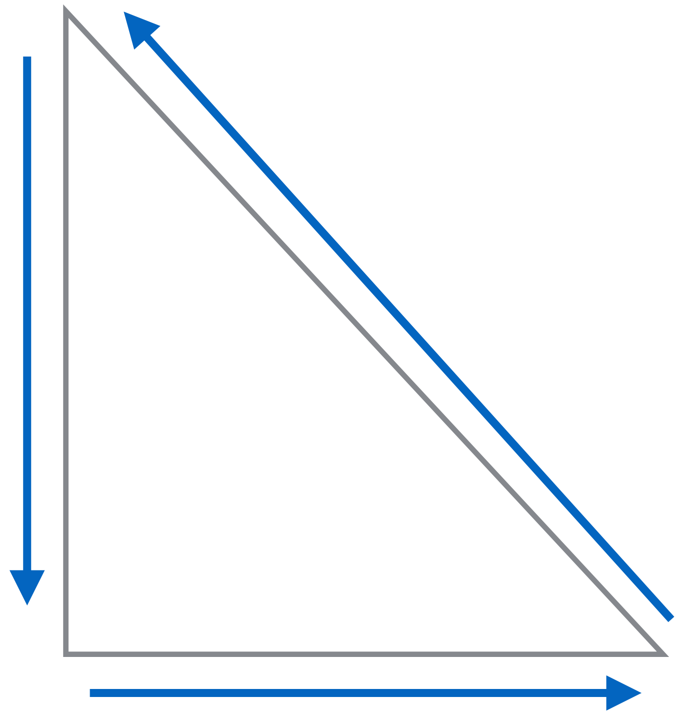
Edge-Edge Neighbors



Also applies to Vertex-Vertex, Vertex-Edge and Vertex-Face

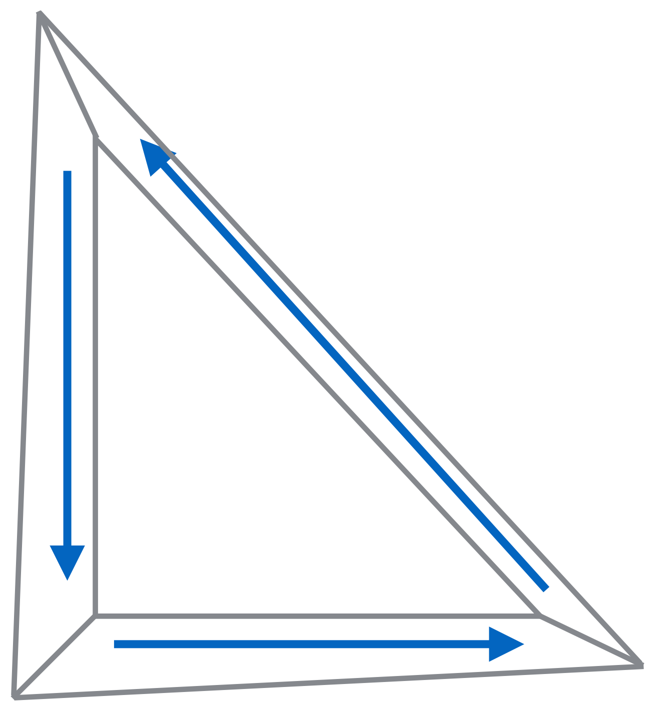
```
current = origHalfEdge;
do {
  current = current->next->dual;
} while (current != origHalfEdge);
```

Face-Edge Neighbors



```
current = origFace->halfEdge;
do {
   current = current->next;
} while (current != origFace->halfEdge);
```

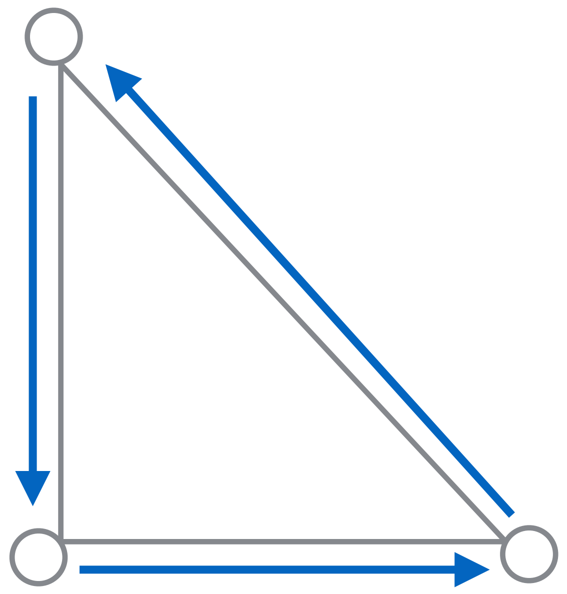
Face-Edge Neighbors



Also applies to Face-Face

```
current = origFace->halfEdge;
do {
   face = current->dual->face;
   current = current->next;
} while (current != origFace->halfEdge);
```

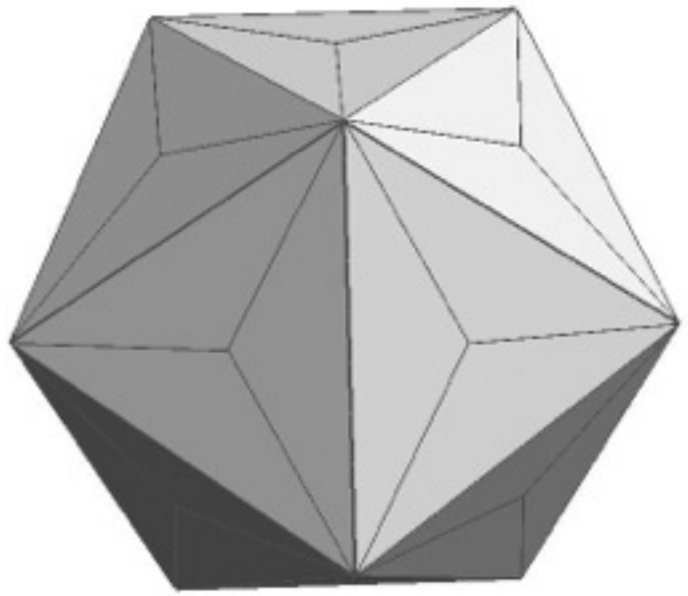
Face-Edge Neighbors

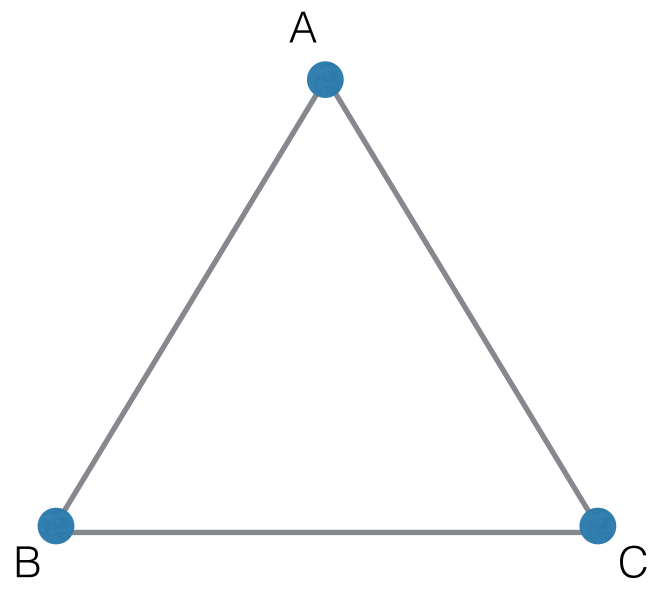


Also applies to Face-Vertex

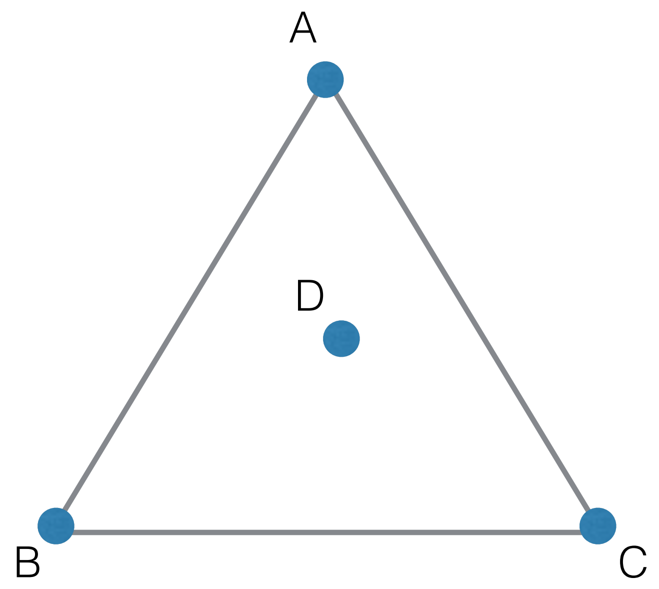
```
current = origFace->halfEdge;
do {
  vertex = current->endVertex;
  current = current->next;
} while (current != origFace->halfEdge);
```

Editing Example: Star Faces

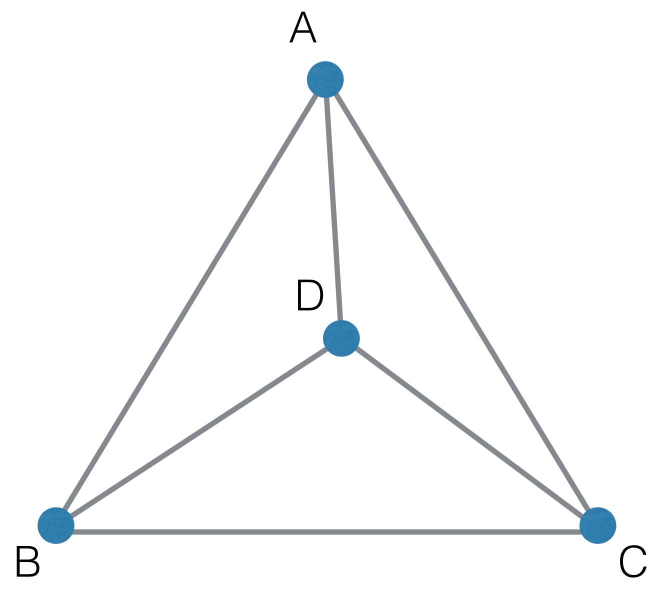




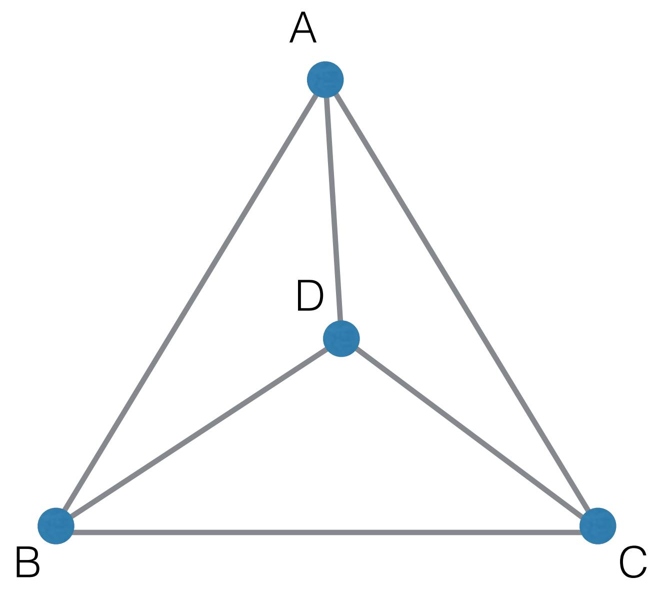
Vertices	Edges	Faces
A	AB	ABC
B	BA	
C	AC	
	CA	
	BC	
	CB	



Vertices	Edges	Faces
A	AB	ABC
B	BA	
C	AC	
D	CA	
	BC	
	CB	

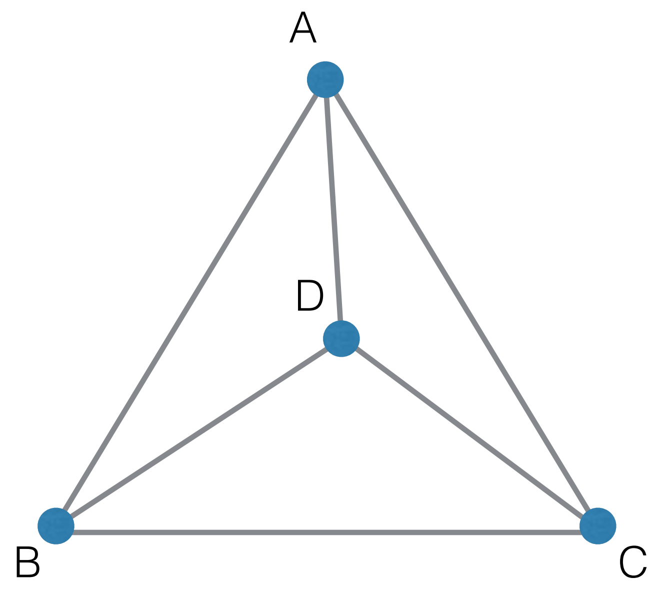


Vertices	Edges	Faces
A	AB	~~ABC~~
B	BA	ABD
C	AC	ACD
D	CA	BCD
	BC	
	CB	



Vertices	Edges	Faces
A	AB	~~ABC~~
B	BA	ABD
C	AC	ACD
D	CA	BCD
	BC	
	CB	
	AD	
	DA	
	BD	
	DB	
	CD	
	DC	

All pointers should be updated!
For example:



Old	New
`CA->next == AB`	`CA->next == AD`

`AD undefined`
`DC undefined`
	`AD->next == DC`

`CA->face == ABC`	`CA->face == ACD`
...	...