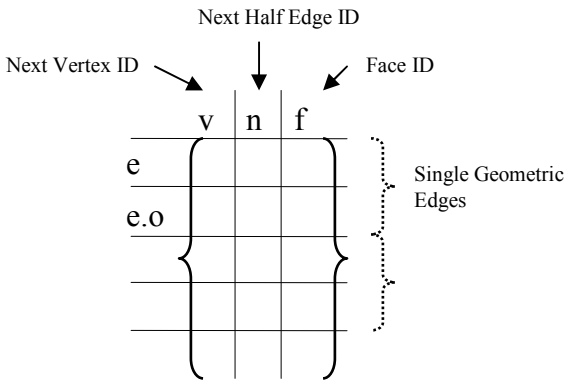
CS7491 Class Notes: 03/09/06

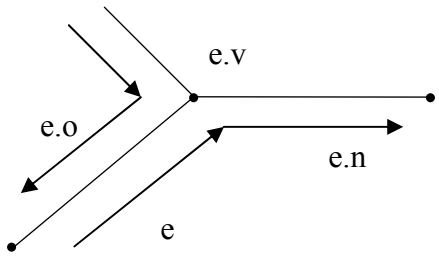
Overview of Project 2:

Connecting Half Edges:

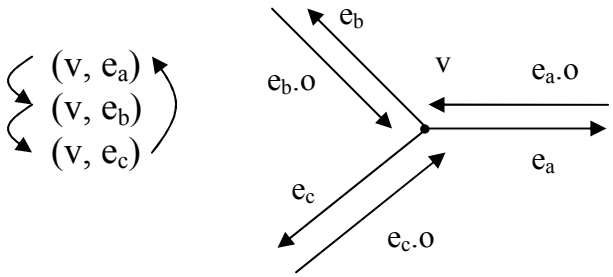
- Input: Edges(V_A, V_B), computed previously based on input
- Data Set: Half edges, every half edge has an opposite half edge
  - Put into an array to hold needed data



  - Metaphor => A half edge is a sidewalk, a sidewalk is on each side of the road



- How to compute e.n:
  - Create a list of associated vertices and leaving edges, (e.o.v, e)
  - Sort the list by the vertices and then sort the edges for a common vertex in counter clockwise order

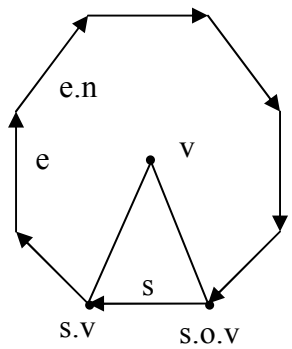


  - Set next values accordingly: e_a.o.n = e_b, e_b.o.n = e_c, etc.

Triangulating Connected Half Edges:

- Triangulate_Faces() {

```
    Set all e.f = -1;                   //Set as un-visited
    Fcount = 0;                         //counter for polygon face
    Vnext = n – 1;                      //counter for new virtual vertices
    For all e {
            If(e.f == -1) {
                    Walk(e, Fcount++, Vnext++); //walk the loop
            }
    }
}
```



- Walk(s, f, v) { //s: 1st half edge, f: face ID, v: virtual virtex

```
    e = s;                              //mark the starting edge
    do() {                              //loop through path
            Tr(e.o.v, e.v, v);          //create a triangle associated with this edge
            e.f = f;                    //mark the face in the table
            e = e.n;                    //move to the next edge in the path
    } while(e != s);                    //continue until back at the first edge
}
```

Notes on Multi-Resolution Slides:

- *Triangle Mesh Idea:*
    - Server holds full original triangle mesh
    - Client needs varying resolutions of this data set (for display/analysis/etc) and must transition cheaply (fast)
    - Resolution decrease of a local region can be sent as a series of edge collapses around a common vertex
    - Resolution increase is a series of splits around that vertex
- Decrease Resolution: Edge Collapse
    - Must send: vertex ID to keep (collapses to other) and edge ID to collapse
- Increase Resolution: Vertex Split
    - Must send: vertex ID to split plus new edge data (multiple options)
- Two proposed methods to send vertex ID's (once vertex ID is known a common method can be used to determine the additional edge information)
- Hoppe method:
    - Sends vertex ID, log(Vn), + edge info
- Batch method:
    - Sends 1 bpv per batch, 0 for no change, 1 for change, + edge info
    - Sent in a given order (E.B. or other)
    - Newer vertices cost less because their bits are not sent previously
- At some level both use equal # of vertex bits. *For large Vn, Batch is better.*
- (Batch modification: Don't split edges with vertices that have already split edges? For accuracy improvement, decoding restriction, cheaper,…?)
- (Compare to multi-depth transitioning of a tetrahedron mesh, discussed later.)

- *Many triangles at low resolution:*
    - Use Points, one per triangle
    - At a distance points display the original pattern, not close up
    - Triangle Mesh: Connectivity is cheap to encode and helps with the geometry prediction.
    - Points: No connectivity. May make geometry encoding more expensive than with it (including its cost).
    - How to get a regular pattern with a point set? (ex. Converted by X-ray casting through facing triangles in a mesh.)