Meshes

Lecture 16
Sections 3.1.9, 4.9

Robb T. Koether

Hampden-Sydney College

Wed, Oct 5, 2011

Outline

1. Meshes
2. Cross Products
3. Finding Surface Normals
   - Using Neighboring Vertices
   - Newell's Algorithm
4. Assignment

Outline

1. Meshes
2. Cross Products
3. Finding Surface Normals
   - Using Neighboring Vertices
   - Newell's Algorithm
4. Assignment

Meshes

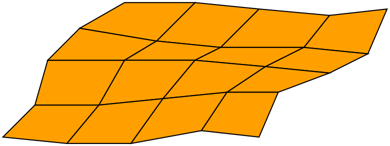
Definition (Mesh)

A mesh is an array of polygons, usually triangles or quadrilaterals, joined together to approximate a smooth surface.

- We store the vertices of the mesh in a two-dimensional array.
- We render the mesh as a `GL_QUAD_STRIP` (deprecated) or a `GL_TRIANGLE_STRIP`.

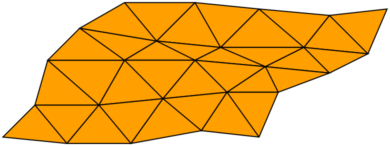
Meshes

- For lighting effects, each vertex of the mesh must have a normal vector.
- Depending on the circumstances, the normals are
  - Computed analytically from the equation of the surface, or
  - Approximated from the vertices themselves.
- Either way, they can be stored in a separate array or computed in real time.



GL_QUAD_STRIP

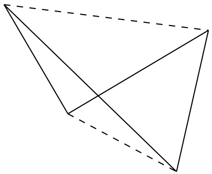
Meshes



GL_TRIANGLE_STRIP

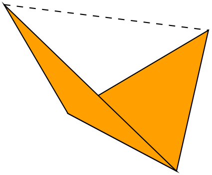
Quad vs. Triangle

- Which is better, quads or triangles?
- OpenGL will tessellate quads into triangles anyway.
- We have no control over which diagonal is used.
- Sometimes, OpenGL with jump back and forth between the two choices as the viewpoint changes.



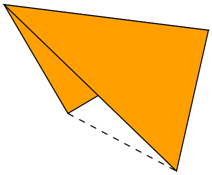
Quad vs. Triangle

- Which is better, quads or triangles?
- OpenGL will tessellate quads into triangles anyway.
- We have no control over which diagonal is used.
- Sometimes, OpenGL with jump back and forth between the two choices as the viewpoint changes.



Quad vs. Triangle

- Which is better, quads or triangles?
- OpenGL will tessellate quads into triangles anyway.
- We have no control over which diagonal is used.
- Sometimes, OpenGL with jump back and forth between the two choices as the viewpoint changes.



Outline

The Cross Product

- To find normal vectors, we need the cross product.

Definition (Cross Product)

The cross product of vectors $\mathbf{u} = (u_1, u_2, u_3)$ and $\mathbf{v} = (v_1, v_2, v_3)$ is defined to be the vector

$$\mathbf{u} \times \mathbf{v} = (u_2 v_3 - u_3 v_2, u_3 v_1 - u_1 v_3, u_1 v_2 - u_2 v_1).$$

- Note that the cross product of vectors is a vector, not a scalar.

The Cross Product

u_1	u_2	u_3
v_1	v_2	v_3

An easy way to remember the cross product.

The Cross Product

u_1	u_2	u_3	u_1	u_2
v_1	v_2	v_3	v_1	v_2

Duplicate the first and second columns.

The Cross Product

u_1	u_2	u_3	u_1	u_2
v_1	v_2	v_3	v_1	v_2

Find this 2×2 determinant for the first component.

The Cross Product

u_1	u_2	u_3	u_1	u_2
v_1	v_2	v_3	v_1	v_2

Find the next 2×2 determinant for the second component.

The Cross Product

u_1	u_2	u_3	u_1	u_2
v_1	v_2	v_3	v_1	v_2

Find the last 2×2 determinant for the third component.

Algebraic Properties of the Cross Product

- Let **u**, **v**, and **w** be vectors and let c be a real number and let θ be the angle between **u** and **v**.

$$\mathbf{u} \times \mathbf{v} = -(\mathbf{v} \times \mathbf{u})$$

$$(c\mathbf{u}) \times \mathbf{v} = \mathbf{u} \times (c\mathbf{v}) = c(\mathbf{u} \times \mathbf{v})$$

$$\mathbf{v} \times \mathbf{v} = \mathbf{0}$$

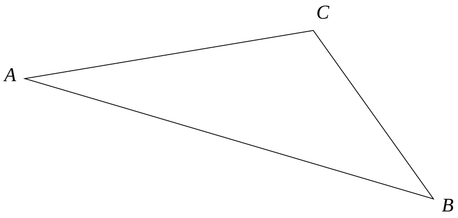
$$(\mathbf{u} \times \mathbf{v}) \cdot \mathbf{u} = (\mathbf{u} \times \mathbf{v}) \cdot \mathbf{v} = 0$$

$$|\mathbf{u} \times \mathbf{v}| = |\mathbf{u}||\mathbf{v}| \sin \theta$$

The Right-hand Rule

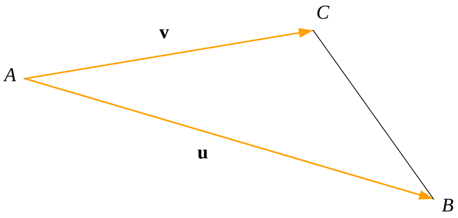
- The right-hand rule helps us remember which way $\mathbf{u} \times \mathbf{v}$ points.
- Arrange the thumb, index finger, and middle finger so that they are mutually orthogonal.
- Let the thumb represent $\mathbf{u}$ and the index finger represent $\mathbf{v}$.
- Then the middle finger represents $\mathbf{u} \times \mathbf{v}$.

Finding Surface Normals



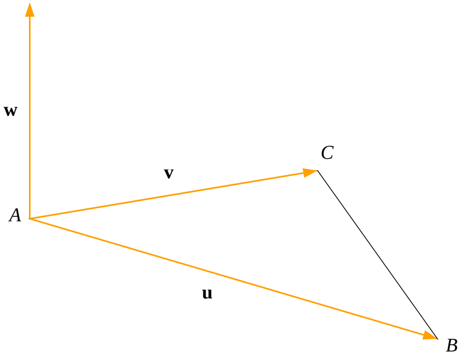
Given a triangle *ABC*, find a unit normal to the surface.

Finding Surface Normals



Form the vectors $\mathbf{u} = B - A$ and $\mathbf{v} = C - A$.

Finding Surface Normals



Compute $\mathbf{w} = \mathbf{u} \times \mathbf{v}$.

Example

Example (Cross Products)

- Let

$$A = (1, 1, 2)$$
$$B = (3, 1, 5)$$
$$C = (1, 0, 4)$$

- Then

$$\mathbf{u} = B - A = (2, 0, 3)$$
$$\mathbf{v} = C - A = (0, -1, 2)$$

- So $\mathbf{w} = \mathbf{u} \times \mathbf{v} = (3, -4, -2)$.

Outline

Outline

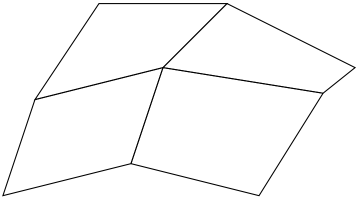
Finding Surface Normals

- If we have an analytical description of the surface (i.e., a mathematical function), then we can use calculus to compute the normal vectors.
- We will do that later.
- On the other hand, if we have only data points, then we must approximate the normal at a vertex by using nearby vertices.

Approximating Surface Normals

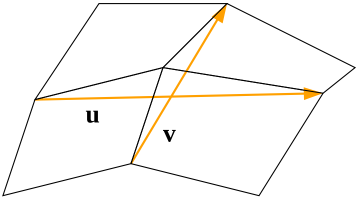
- Assuming a rectangular mesh, a simple method to approximate a normal vector is to
  - Let **u** be the vector from the next vertex on the left to the next vertex on the right.
  - Let **v** be the vector from the next vertex below to the next vertex above.
  - Compute the normal vector as $\mathbf{u} \times \mathbf{v}$.

Approximating Surface Normals



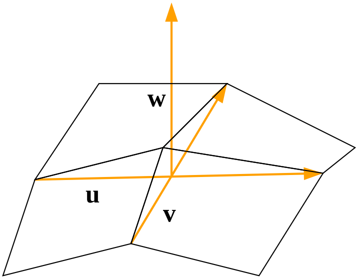
Begin with a vertex in the mesh.

Approximating Surface Normals



Find the neighboring vectors **u** and **v**.

Approximating Surface Normals



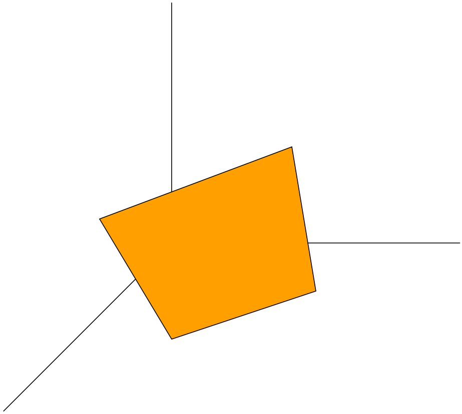
Compute their cross product **w**.

Outline

Newell's Algorithm

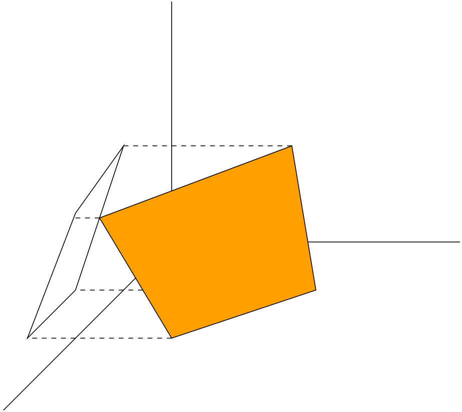
- Newell's algorithm is somewhat more sophisticated.
- Given a polygon in space,
  - Project it onto the three coordinate planes: *yz*, *xz*, and *xy*.
  - Compute the areas of the projections: A_{yz}, A_{xz}, and A_{xy}.
  - Form the normal vector $\mathbf{n} = (A_{yz}, A_{xz}, A_{xy})$.
- The idea is that the more the facet is tilted towards a plane, the greater the area of its projection onto that plane.

Newell's Algorithm

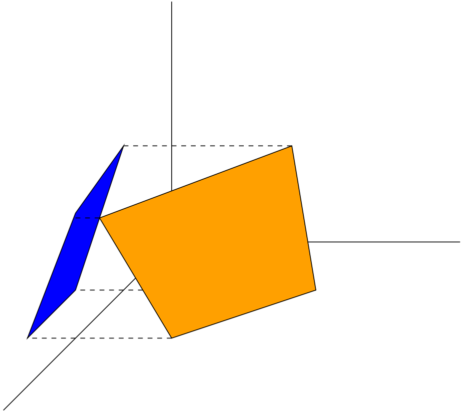


Begin with a quadrilateral in space.

Newell's Algorithm

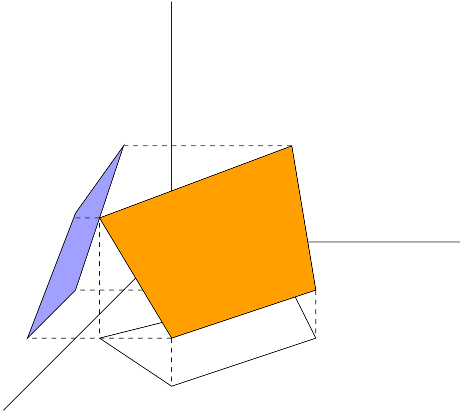


Project it onto the *yz*-plane.



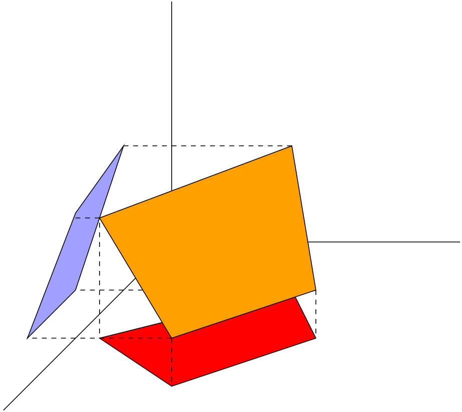
Find the area of the projection.

Newell's Algorithm



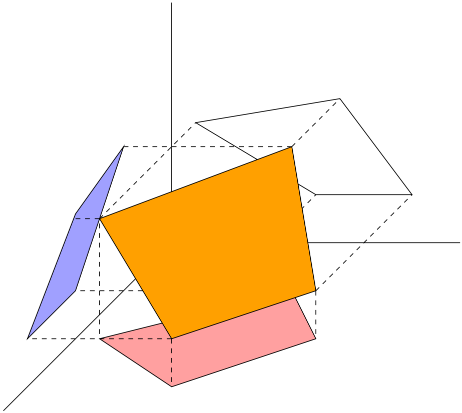
Project it onto the *xz*-plane.

Newell's Algorithm



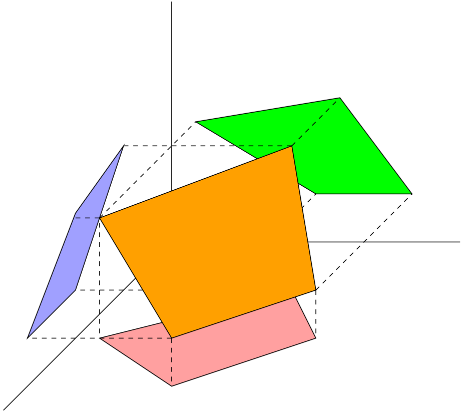
Find the area of the projection.

Newell's Algorithm



Project it onto the xy-plane.

Newell's Algorithm



Find the area of the projection.

Newell's Algorithm

Theorem

If the vertices of a polygon are

$$(x_0, y_0), (x_1, y_1), (x_2, y_2), \ldots, (x_{n-1}, y_{n-1}),$$

then the area of the polygon is

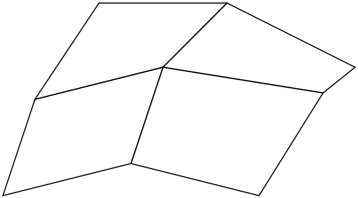
$$A = \frac{1}{2}\left[(x_0 y_1 - x_1 y_0) + (x_1 y_2 - x_2 y_1) + \cdots + (x_{n-1} y_0 - x_0 y_{n-1})\right].$$

Newell's Algorithm

- Applying Newell's Algorithm gives

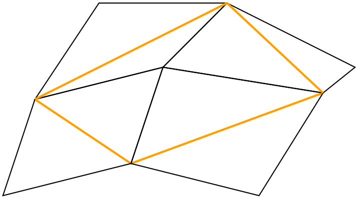
$$A_{yz} = \frac{1}{2}[(y_0z_1 - y_1z_0) + (y_1z_2 - y_2z_1) + \cdots + (y_{n-1}z_0 - y_0z_{n-1})]$$
$$A_{xz} = \frac{1}{2}[(z_0x_1 - z_1x_0) + (z_1x_2 - z_2x_1) + \cdots + (z_{n-1}x_0 - z_0x_{n-1})]$$
$$A_{xy} = \frac{1}{2}[(x_0y_1 - x_1y_0) + (x_1y_2 - x_2y_1) + \cdots + (x_{n-1}y_0 - x_0y_{n-1})]$$

Newell's Algorithm



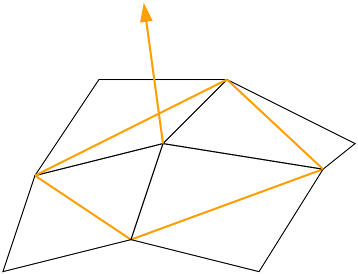
Begin with a vertex in the mesh.

Newell's Algorithm



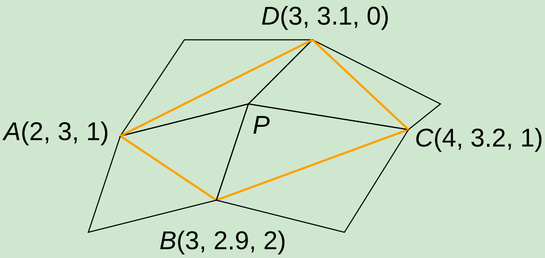
Form the quadrilateral from its neighboring vertices.

Newell's Algorithm



Apply Newell's Algorithm.

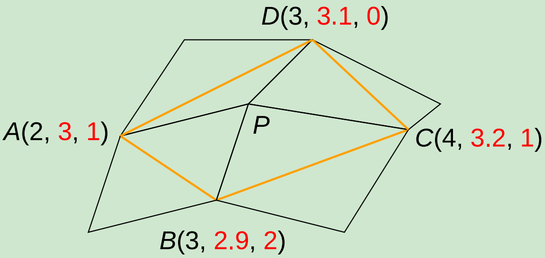
Example (Newell's Algorithm)



Find a normal at P.

Newell's Algorithm

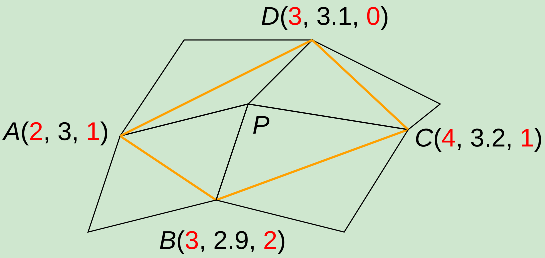
Example (Newell's Algorithm)



$$A_{yz} = -0.2$$

Newell's Algorithm

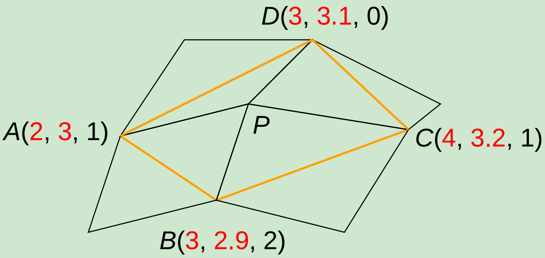
Example (Newell's Algorithm)



$$A_{xz} = 2.0$$

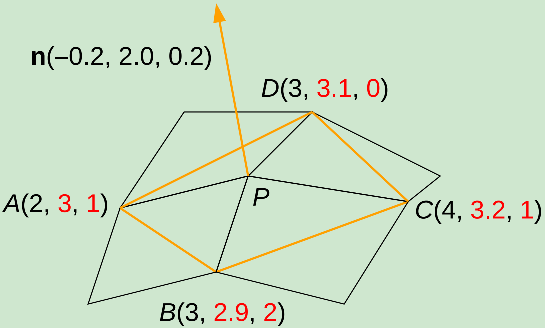
Newell's Algorithm

Example (Newell's Algorithm)



$$A_{xy} = 0.2$$

Example (Newell's Algorithm)



$$\mathbf{n} = (-0.2, 2.0, 0.2)$$

Example (Newell's Algorithm)

- What does the simpler method produce, where we let **u** go from the left neighbor to the right neighbor and let **v** go from the lower neighbor to the upper neighbor?

Example (Newell's Algorithm)

- What does the simpler method produce, where we let **u** go from the left neighbor to the right neighbor and let **v** go from the lower neighbor to the upper neighbor?
- $\mathbf{u} = (4, 3.2, 1) - (2, 3, 1) = (2, 0.2, 0)$.

Newell's Algorithm

Example (Newell's Algorithm)

- What does the simpler method produce, where we let **u** go from the left neighbor to the right neighbor and let **v** go from the lower neighbor to the upper neighbor?
- $\mathbf{u} = (4, 3.2, 1) - (2, 3, 1) = (2, 0.2, 0)$.
- $\mathbf{v} = (3, 3.1, 0) - (3, 2.9, 2) = (0, 0.2, -2)$.

Newell's Algorithm

Example (Newell's Algorithm)

- What does the simpler method produce, where we let **u** go from the left neighbor to the right neighbor and let **v** go from the lower neighbor to the upper neighbor?
- $\mathbf{u} = (4, 3.2, 1) - (2, 3, 1) = (2, 0.2, 0)$.
- $\mathbf{v} = (3, 3.1, 0) - (3, 2.9, 2) = (0, 0.2, -2)$.
- $\mathbf{u} \times \mathbf{v} = (-0.4, 4, 0.4)$.

Outline

Homework

Homework

- Read Subsection 3.1.9 – cross products.
- Read Section 4.9 – meshes.