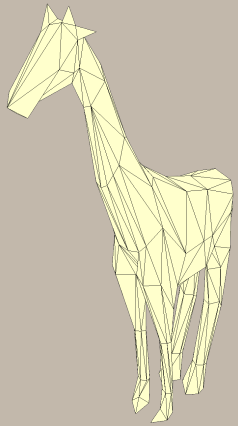


Lecture 11 - Subdivision Surfaces

Philip Caplan

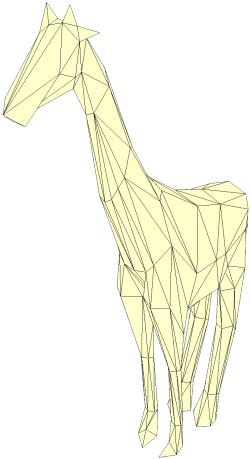
CSCI 0422 - Geometric Modeling (Spring 2022)

Middlebury

Learning objectives

By the end of this lecture you will be able to:

- define a smooth curve or surface as the limit of a sequence of repeated divisions,
- subdivide meshes using Catmull-Clark and Loop subdivision.



Getting started...

Switch Host & Client today!

```
$ git pull
$ make update
$ cmake .
$ make template_class11_subdivision
$ cmake .
```

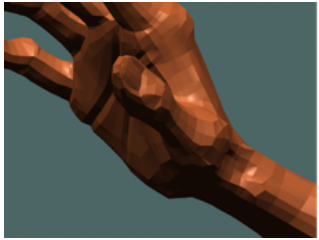
Compiling and running the exercise:

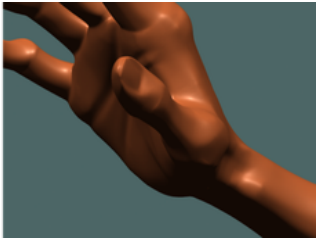
```
$ make class11_subdivision
```

Compiling and running the solution (after class):

```
$ make class11_subdivision_sol
```

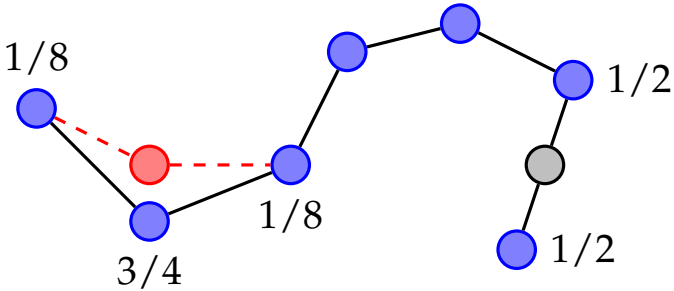
Motivation: render smoother object movement in more detail.





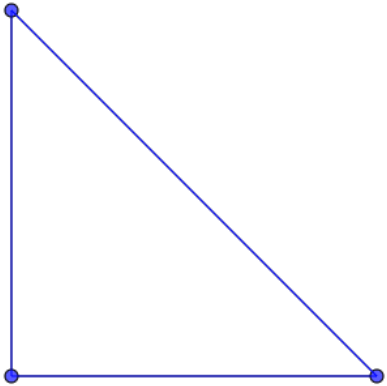
https://www.youtube.com/watch?v=uMVtpCPx8ow

Imagine we have a curve defined by a bunch of line segments.



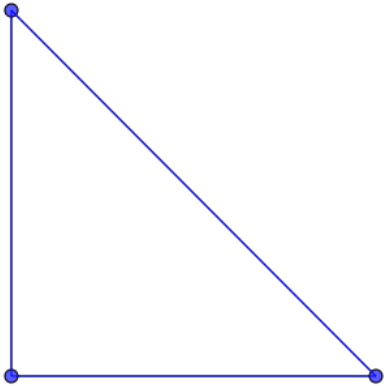
Subdivision is useful if you don't have an analytic geometry description.

Chaikin's corner cutting scheme
(inserts two new points on every edge at 1/4 and 3/4)



level 0

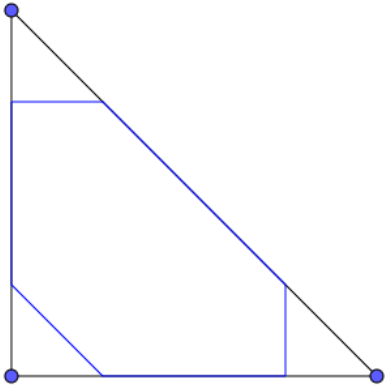
Catmull-Clark scheme
(inserts edge midpoint + smoothes existing vertices)



level 0

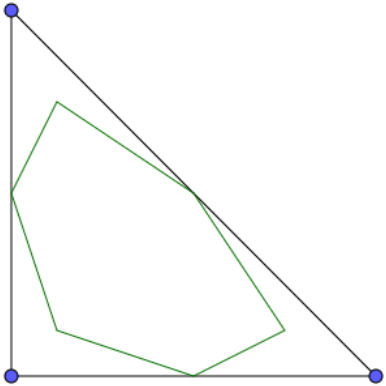
Subdivision is useful if you don't have an analytic geometry description.

Chaikin's corner cutting scheme
(inserts two new points on every edge at 1/4 and 3/4)



level 1

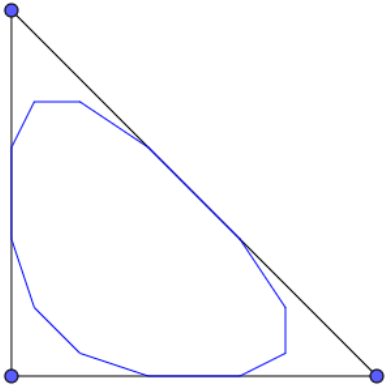
Catmull-Clark scheme
(inserts edge midpoint + smoothes existing vertices)



level 1

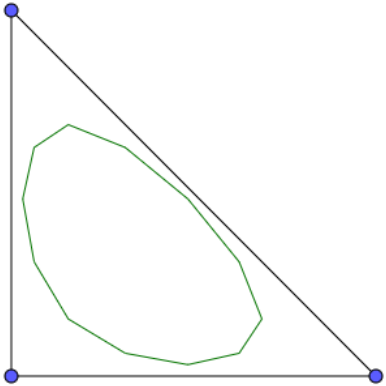
Subdivision is useful if you don't have an analytic geometry description.

Chaikin's corner cutting scheme
(inserts two new points on every edge at 1/4 and 3/4)



level 2

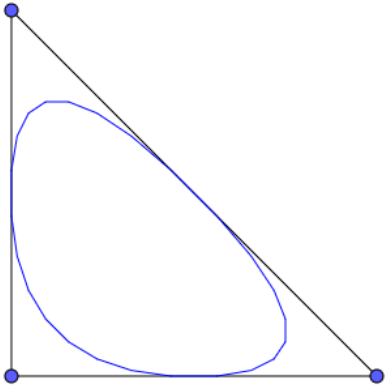
Catmull-Clark scheme
(inserts edge midpoint + smoothes existing vertices)



level 2

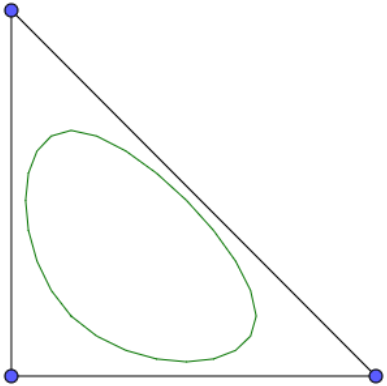
Subdivision is useful if you don't have an analytic geometry description.

Chaikin's corner cutting scheme
(inserts two new points on every edge at 1/4 and 3/4)



level 3

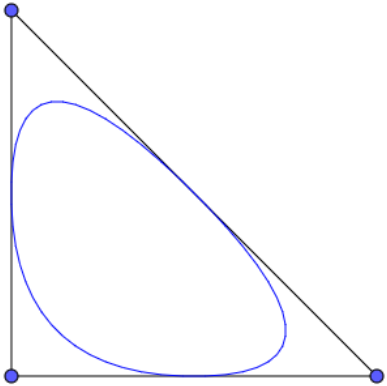
Catmull-Clark scheme
(inserts edge midpoint + smoothes existing vertices)



level 3

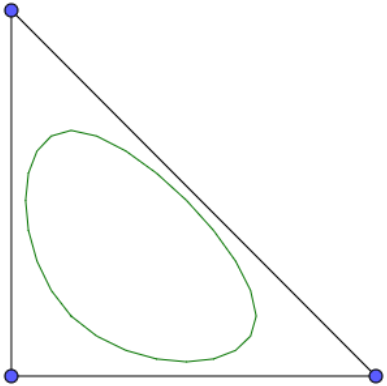
Subdivision is useful if you don't have an analytic geometry description.

Chaikin's corner cutting scheme
(inserts two new points on every edge at 1/4 and 3/4)



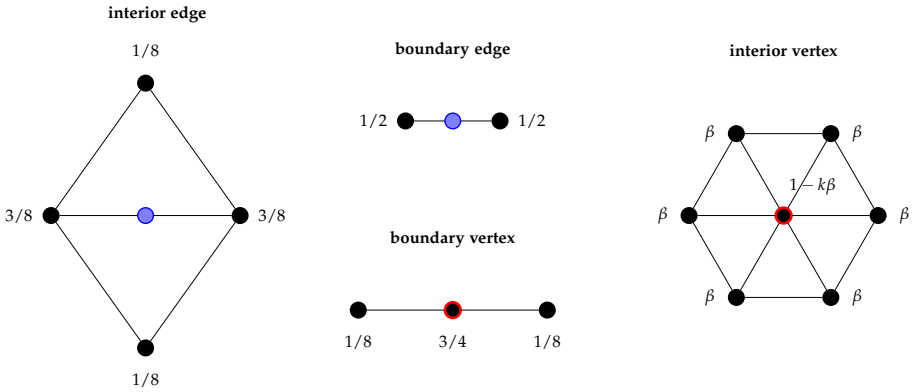
level 4

Catmull-Clark scheme
(inserts edge midpoint + smoothes existing vertices)



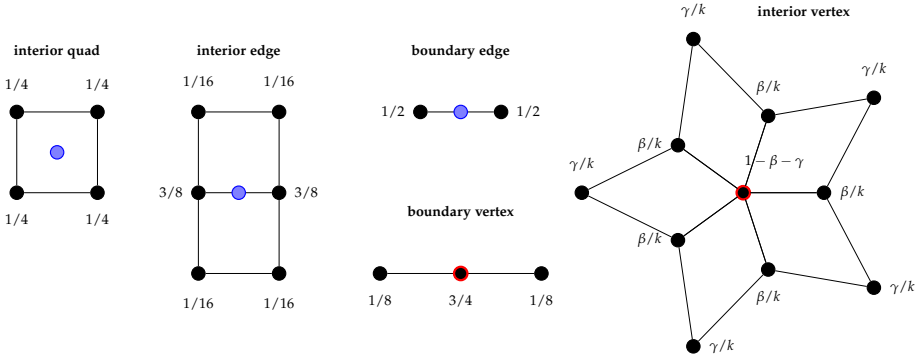
level 4

Loop subdivision scheme can be used for triangle meshes.



subdivide every triangle into four new triangles

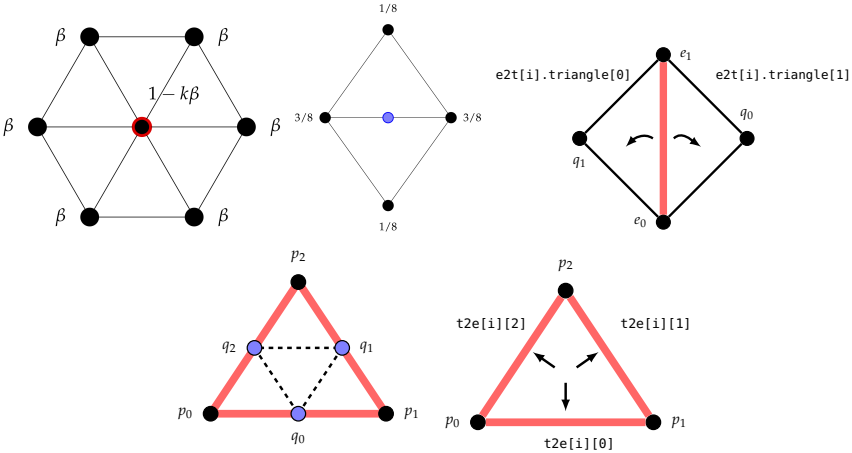
Catmull-Clark subdivision scheme can be used for quad meshes.



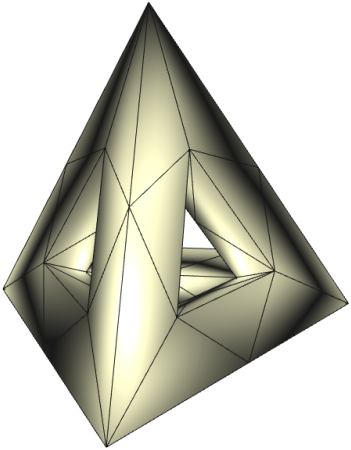
subdivide every quad into four new quads

Data structures for Loop subdivision.

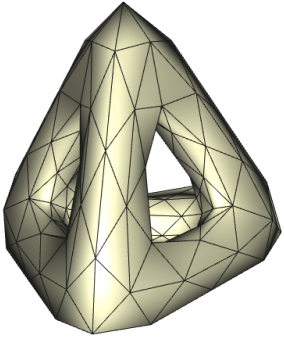
connectivity-based or halfedge-based will work



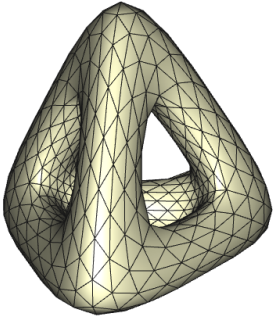
Loop subdivision in action.



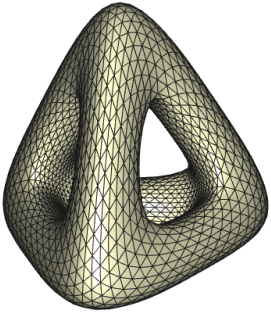
Loop subdivision in action.



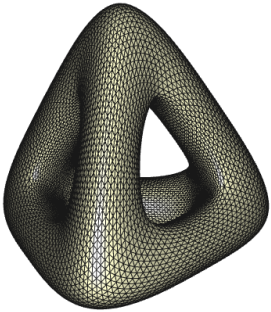
Loop subdivision in action.



Loop subdivision in action.

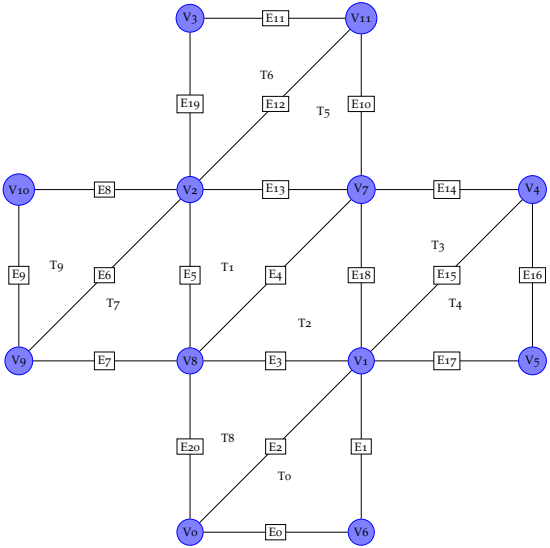


Loop subdivision in action.



Let's do some exercises!

1. implement the `getOppositePoint` function,
2. calculate coordinates of points to insert along edges,
3. divide a triangle into four new triangles.

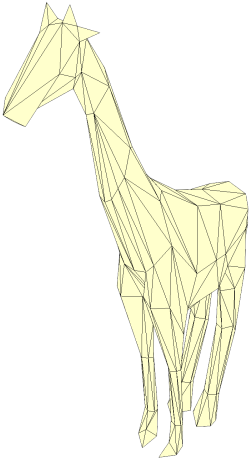


Please follow the exercises in the comments.

```

// example 1: getting the opposite point of edge e0 = (0,6) that is in triangle 0
flux_assert( getOppositePoint(0,6,0) == 1 );

// exercise 1a: get both vertex indices "opposite" edge e6
// and print them out

// exercise 1b: get both vertex indices "opposite" edge e18
// and print them out

// exercise 2: compute the coordinates of the point to insert along edge e4
// using the Loop subdivision scheme, and print the coordinates

// exercise 3a: compute the indices of the four triangles obtained when subdividing triangle t1
// note: assume edge points have been added in order of edges
// print the vertex indices of the four new triangles

// exercise 3b: compute the indices of the four triangles obtained when subdividing triangle t2
// note: assume edge points have been added in order of edges
// print the vertex indices of the four new triangles
```

Your TODO list . . .

- nothing added to `flux-base` that requires testing today,
- work on Project 2!



Don't forget to commit and push your changes!

Host: (assuming you are in top-level `flux` directory)

```
$ git add exercises/class11
$ git commit -a -m "added exercises from lecture 11"
$ git push
```

If you are in a build directory, the first command would be: `$ git add ../../exercises/class11`

Client:

```
$ git pull
```