CS 465 Homework 7

(revised November 5, 2005)

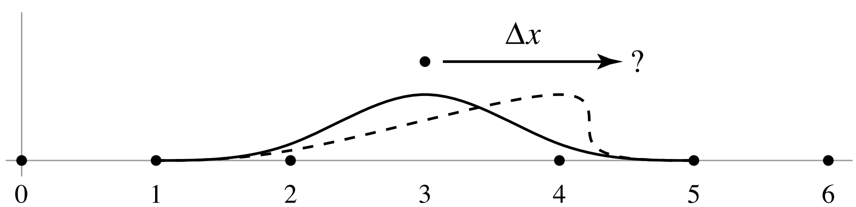
out: Friday 4 November 2005
due: Friday 11 November 2005

Problem 1: B-spline Surfaces

Consider a B-spline surface patch that is defined on a 7 by 7 grid of control points. The points are laid out on in an integer lattice on the x-y plane, running from $(0, 0, 0)$ to $(6, 6, 0)$, except for the center point, which is at $(3, 3, 1)$. In this configuration the spline surface is a graph of the basis function associated with the center patch.

Now suppose we begin moving the center control point in the $+x$ direction. This makes the bump in the surface lean over, and eventually the surface ceases to be a height field. That is, the side of the bump becomes a vertical cliff and then an overhang. The question is, how far has the point moved when this happens?

Let's begin with a B-spline curve instead, in this configuration:



1. How can you tell by looking at the coordinate functions that the curve has a vertical tangent?

2. Show that the first and second derivatives of a B-spline curve are defined by basis functions, in the same way that the position is, and sketch graphs of the basis functions. Note that symmetry can save you some work in coming up with these functions.

3. Sketch plots of the x coordinate and its derivative for the initial configuration and for a configuration with the center point moved a bit to the right. (The point here is qualitative; great precision is not needed.)

4. At what parameter value will the cliff form?

5. Write the relevant derivative of the curve at that parameter value as functions of Δx.

6. At what value of Δx does the cliff form?

Now we return to the case of the spline patch.

7. What are the basis functions for the two tangents of the patch? It's OK to write them in terms of the basis functions you worked out for the curve.

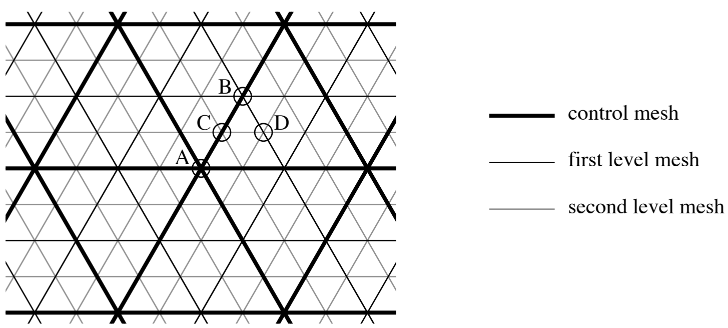
8. At what value of Δx does the cliff form in the spline patch?

Problem 2: Subdivision surfaces

The subdivision masks presented in the lecture (and also available in the course notes linked from the schedule) tell how to compute the positions of vertices at the first level of subdivision from the vertices of the control mesh. The same masks are used to compute vertices at the second level of subdivision from those at the first.

Since the vertices at the second level of subdivision are still linear combinations of the control vertices, it's possible to write down "double subdivision" masks that let you compute the second level vertex positions directly from the control vertices (without computing the level 1 vertices first).

For the Loop scheme, there are four kinds of double subdivision masks for vertices in regular areas (where all involved vertices will always have valence 6). They are exemplified by the vertices labeled A, B, C, and D here:



For example, Figure 1 shows how a vertex of type A (one that was in the control mesh to start with) has a double subdivision mask that is the sum of the subdivision masks of the vertex and all its level-one neighbors.

Give the double-subdivision masks for vertex types B, C, and D.

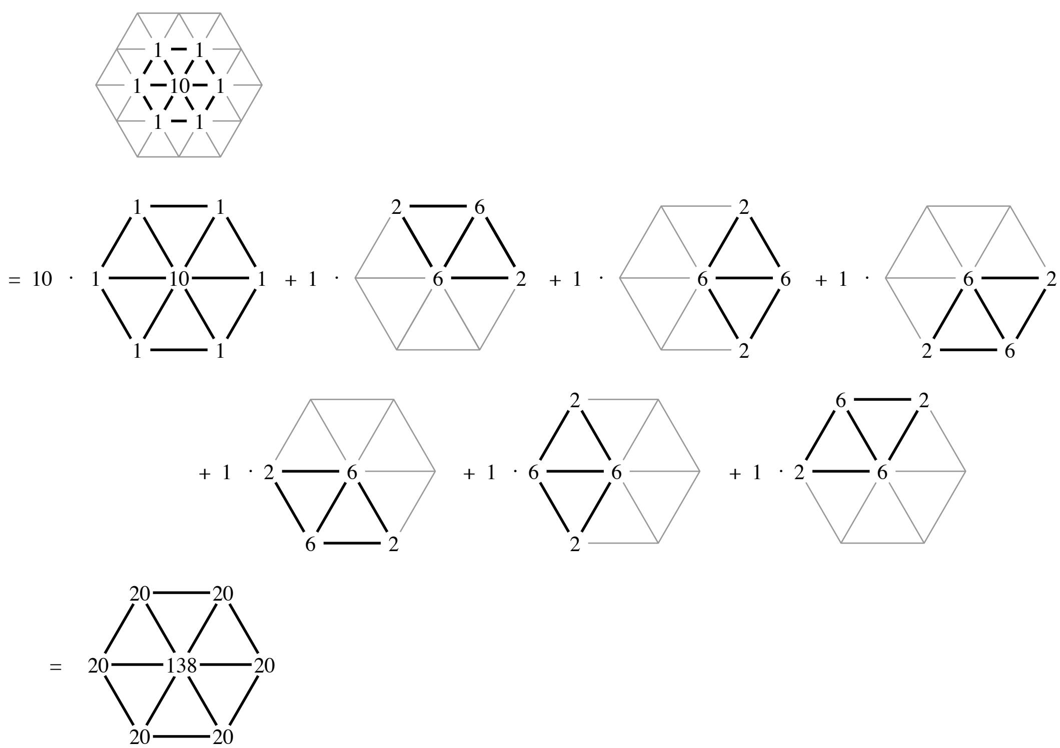


Figure 1: Combining subdivision masks to create a two-level mask. All weights have an assumed denominator of 256 (16 for each of the two masks being multiplied).