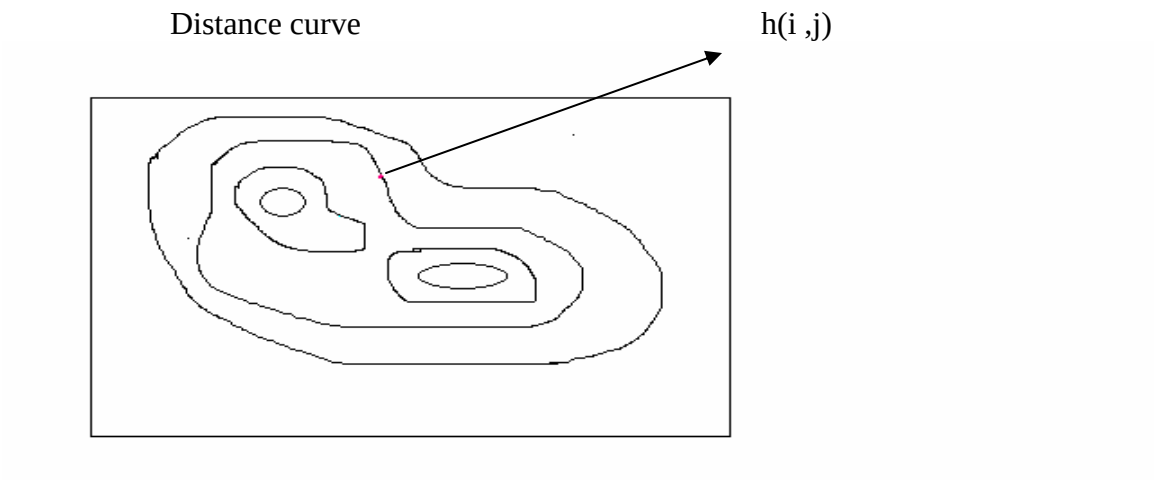
1/31/06

Lecture 07



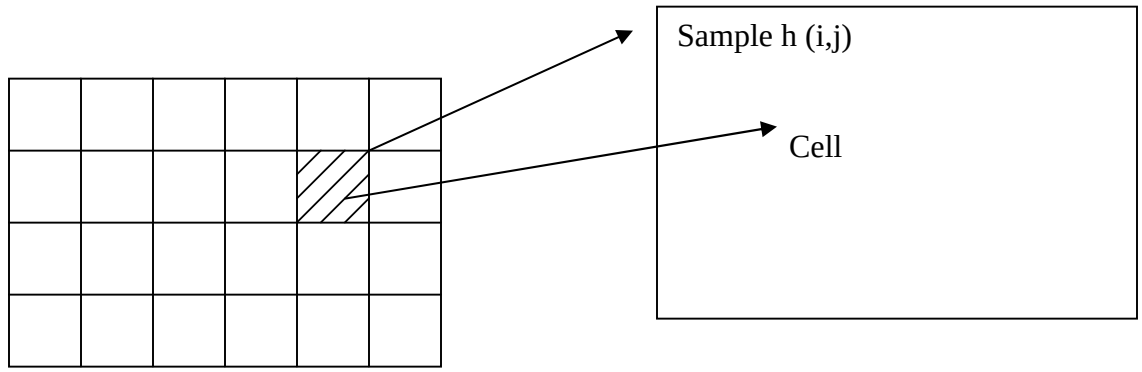
In the above curve has pixels and every pixel has a value associated with it which can be represented using some no. of bits like 16 bits. It can be a terrain where each pixel has some height .

Aim – To transmit a level curve

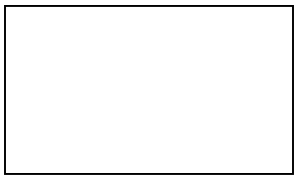
To find a level Curve C(z) = { p : h(p) = z}

Where p = (i , j) is a continuous set of points
z = height

Here we are going to assume that client has very less memory and can store only one level curve .



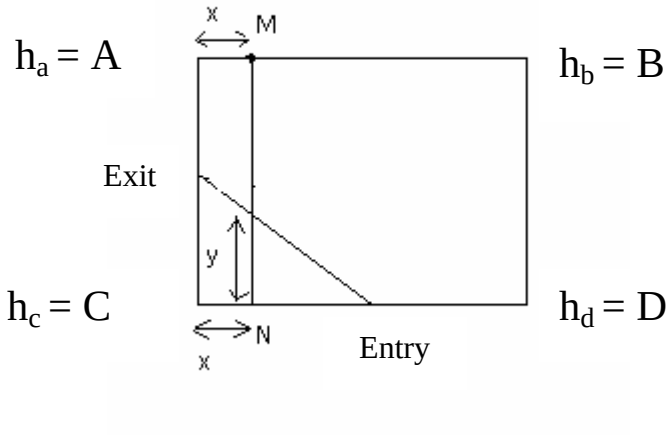
h_a h_b



h_c h_d

For each cell we find whether curve intersects it or not .

We interpolate height in the cell .For this we can use bilinear interpolation



$$M = A + x(B - A)$$

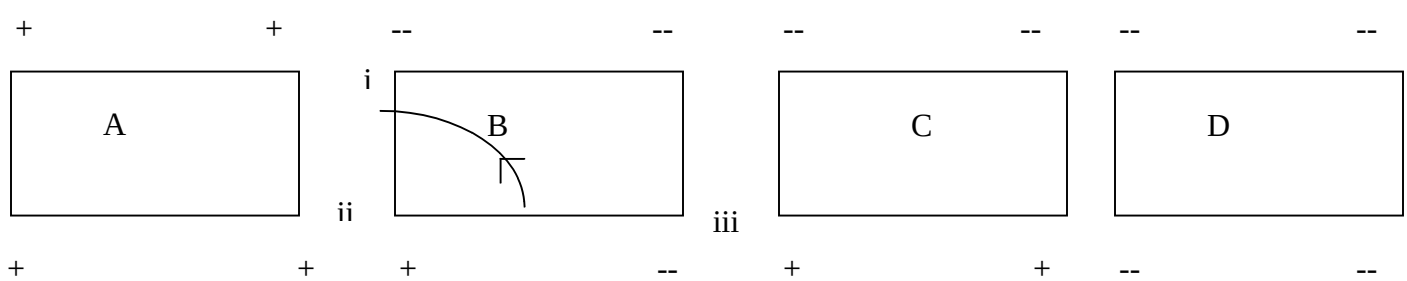
$$N = D + x(C - D)$$

$$h(x, y) = N + y(M - N)$$

Algorithms to find C(z) →

<u>Algo 1</u> (Not very efficient) : For each cell C compute its contribution to C(z) and send it .

Different configurations in which a level curve can intersect a cell in :



+ : the point is at height more than the level curve
- : the point is at height lower than the level curve

Fig A : no point needs to be sent here
Fig B : here just 3 point need to be sent (i , ii & iii)
Fig C : Here all 4 point need to be sent
Fig D : No point needs to be sent

We use linear interpolation to find entry and exit points.

This algorithm sends the whole curve . But we don't want to sent the whole curve because of the limited amount of resources available to the client

So the foll algorithm is more efficient than this one.

Algorithm 2 :

Here we don't send all the points . Here after finding the initial point on curve depending on t he height of the corners of the cell (+ / --) we only transmit L (for goin left) , R (for goin right) or S (for going straight) .

We always move by keeping the the point at more height on the left .