Algorithms for GIS:

Computing visibility on terrains

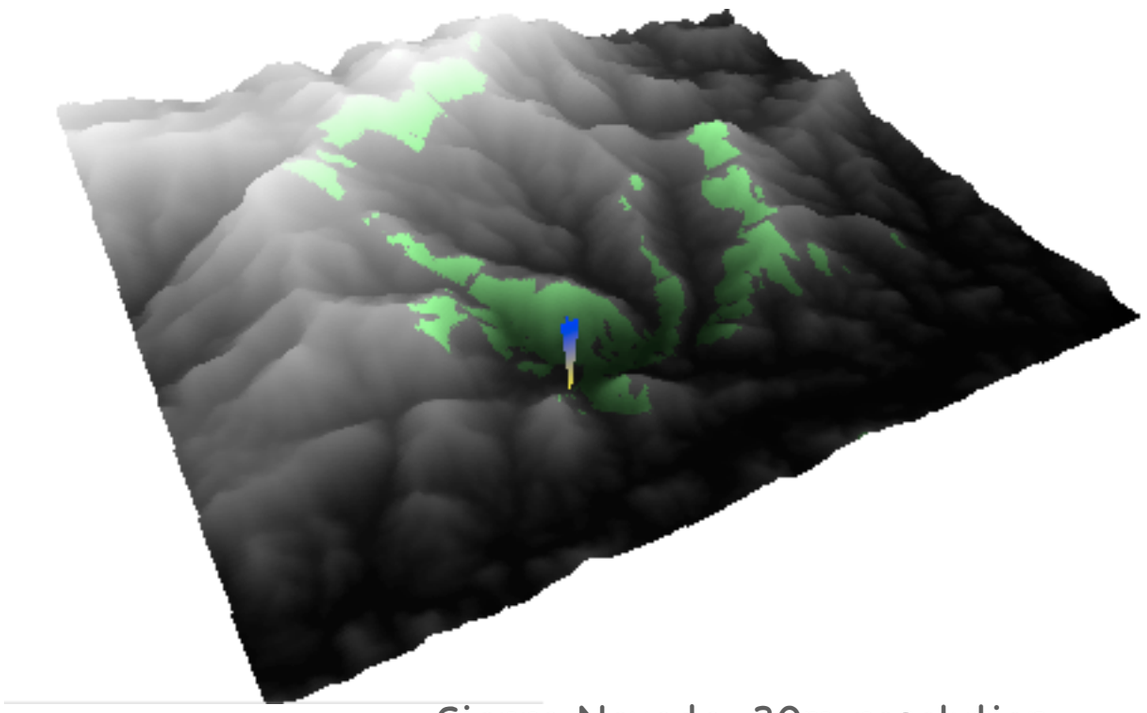
Visibility on terrains

- Are two points (on a terrain) visible to each other?
- What can one see from a given point (on a terrain)?
- How much does the visible area increase if we stand on a 10ft ladder?
- What is the point with largest visibility?
- What is the point with lowest visibility?
- How to place an ugly pipe in a scenic area?
- How to place a scenic highway?
- What is the cumulative visible area from these set of cell towers?
- Find a set of tower locations to cover the terrain
- …

Visibility on terrains

Problem:

- Terrain model T + viewpoint v
- Compute the **viewshed** of v: the set of points in T visible from v



Sierra Nevada, 30m resolution

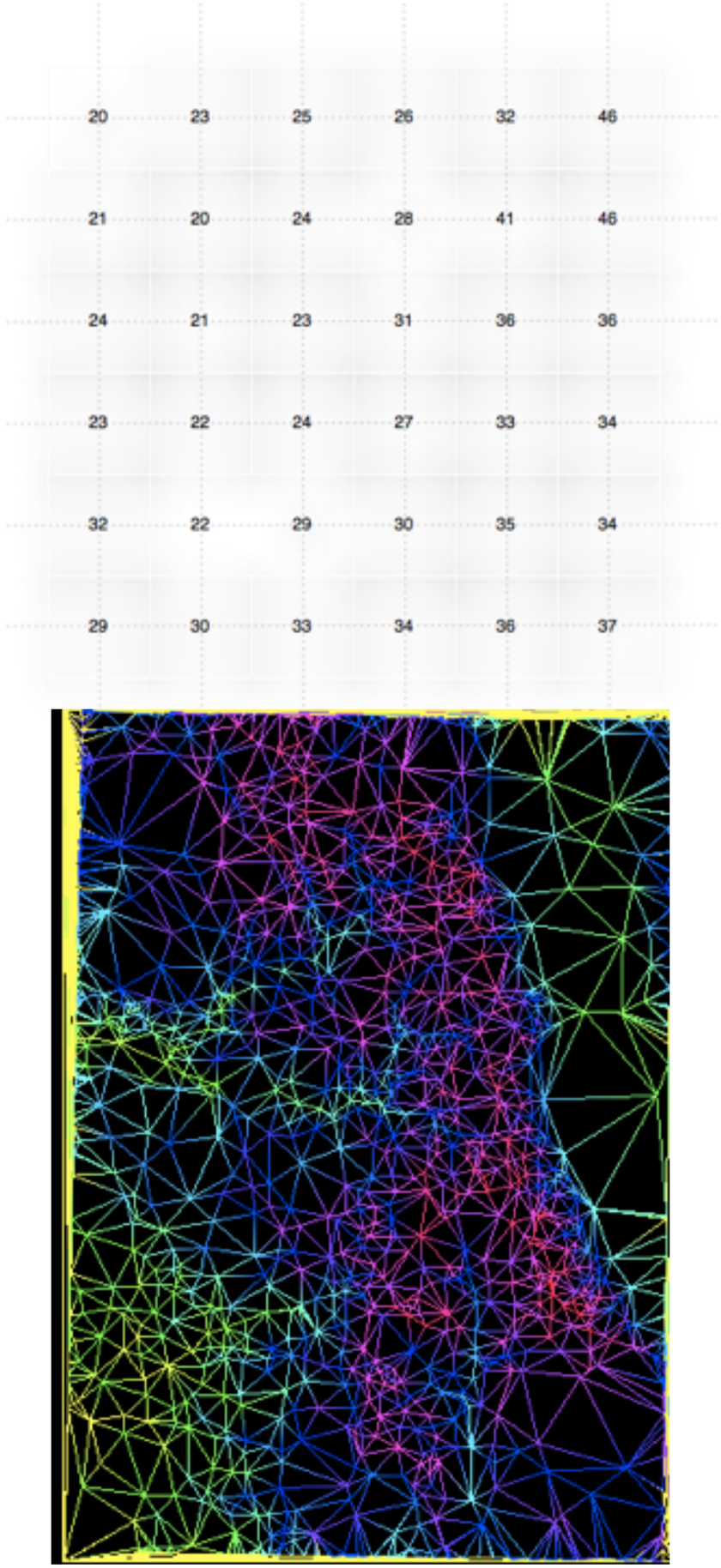
Visibility on terrains

Input: terrain model

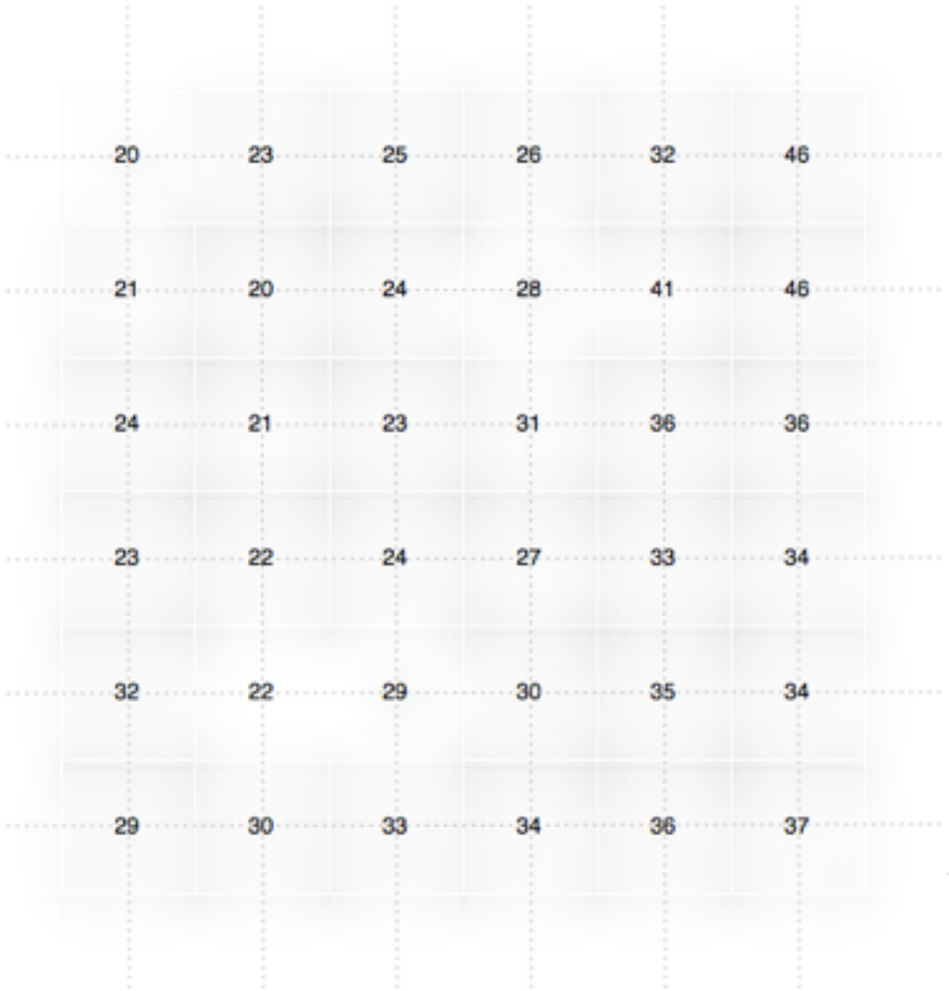
- grid or TIN (triangulation)

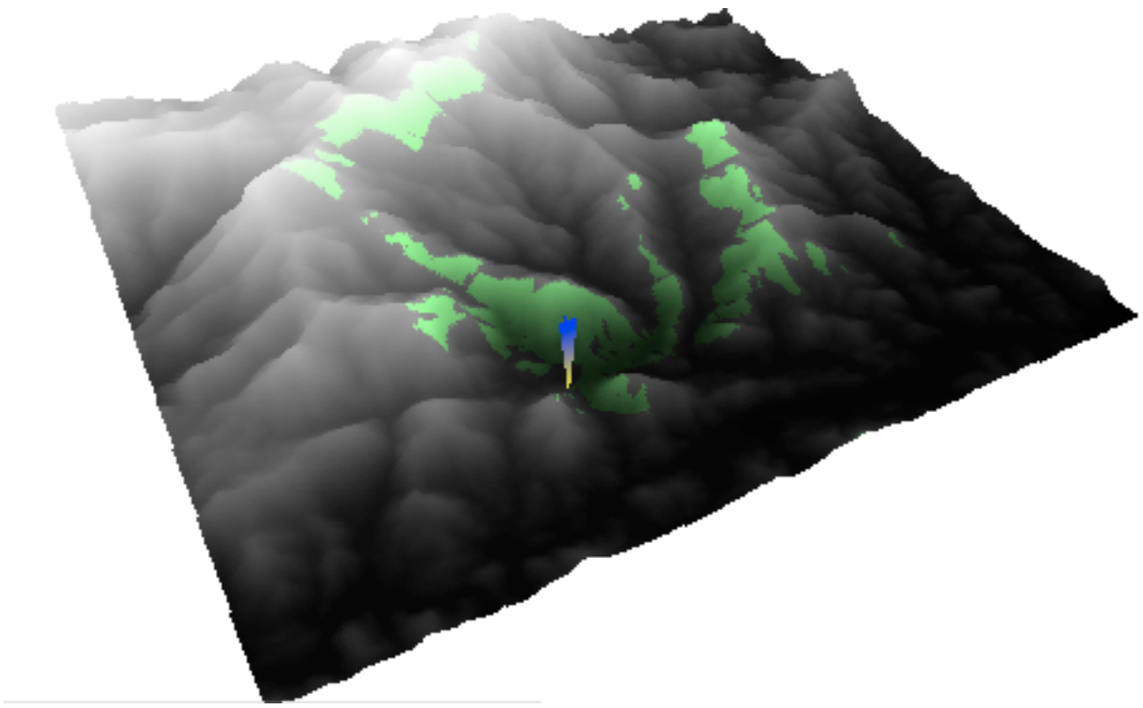
Output: viewshed model

- grid elevation model ==> grid viewshed
- TIN elevation model ==> TIN viewshed
(…more later…)



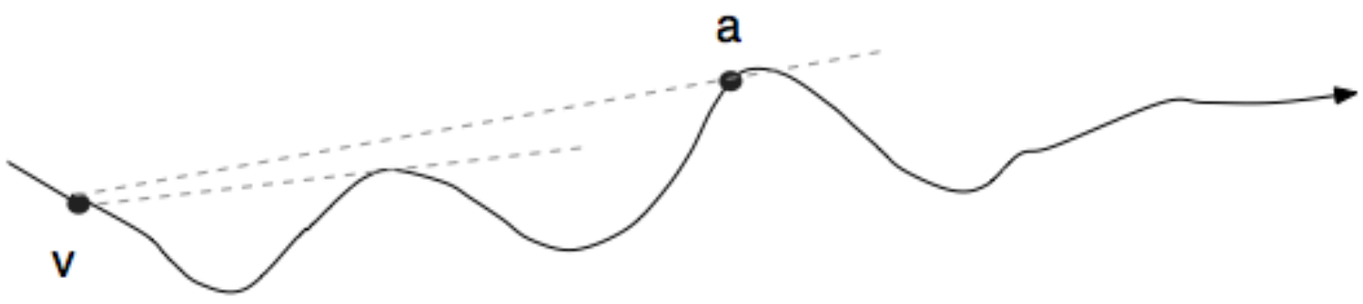
Visibility on grid terrains

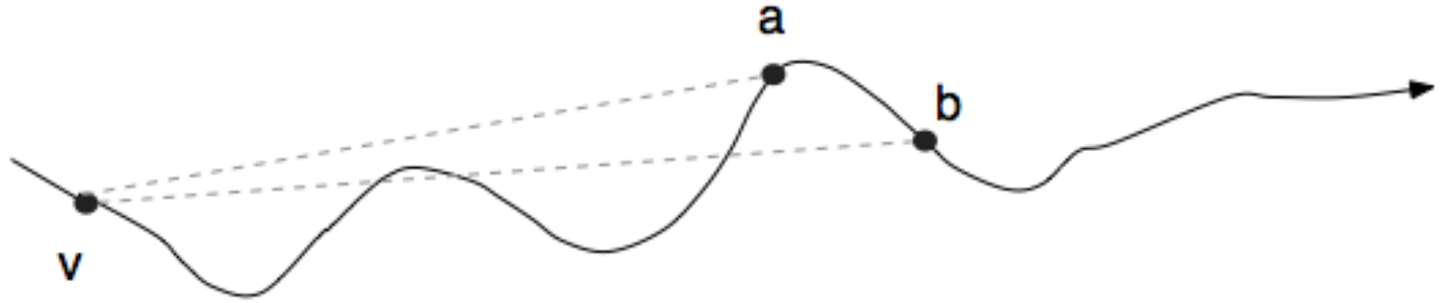




Sierra Nevada, 30m resolution

Visibility





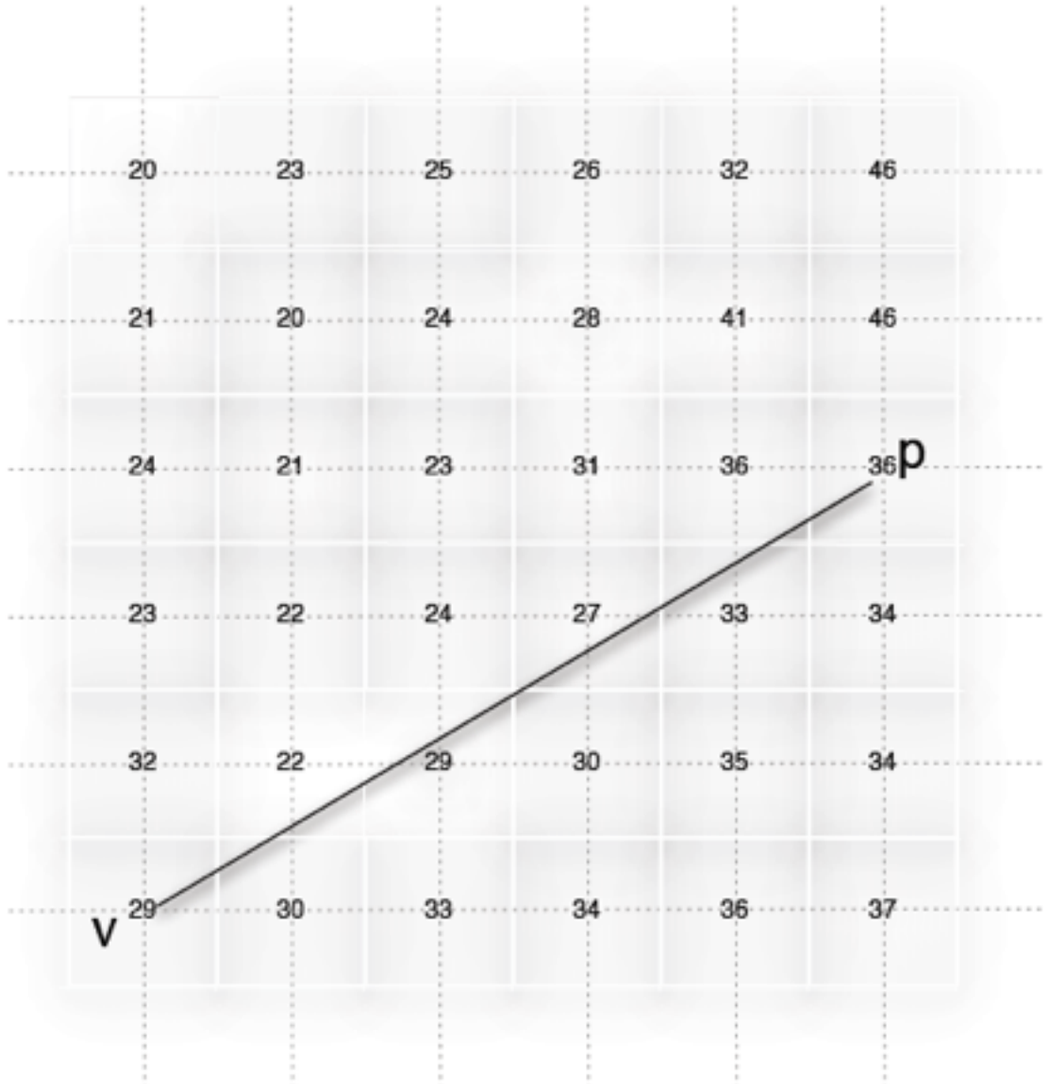
(u,v) visible iff segment uv does not intersect T

uv is called line-of-sight (LOS)

Basic viewshed algorithm

Input: elevation grid
Output: visibility grid, each point marked visible/invisible

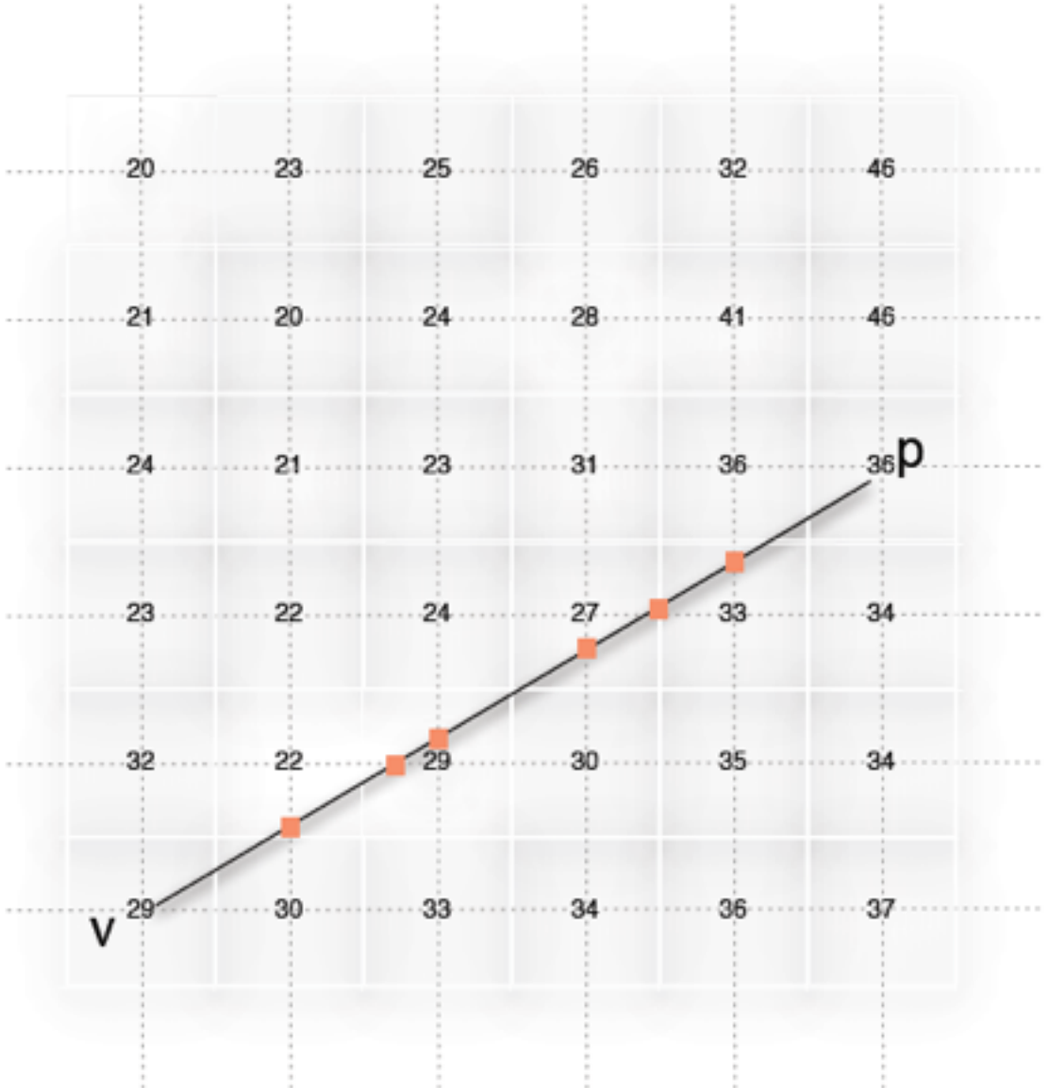
- For each p in grid
  - compute intersections between *vp* and grid lines
  - if all these points are below *vp* then *p* is visible



Basic viewshed algorithm

Input: elevation grid
Output: visibility grid, each point marked visible/invisible

- For each p in grid
  - compute intersections between *vp* and grid lines
  - if all these points are below *vp* then *p* is visible

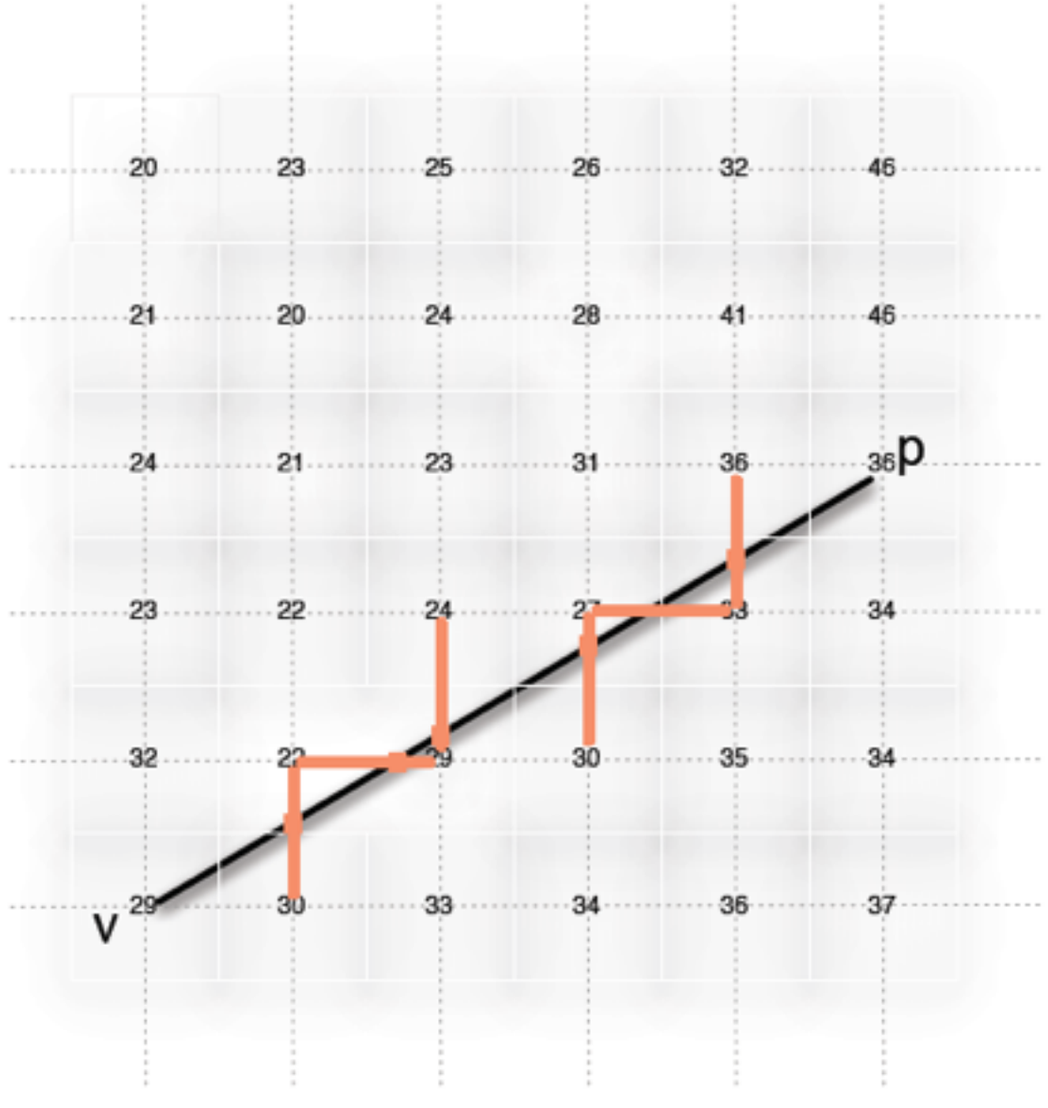


Basic viewshed algorithm

Input: elevation grid
Output: visibility grid, each point marked visible/invisible

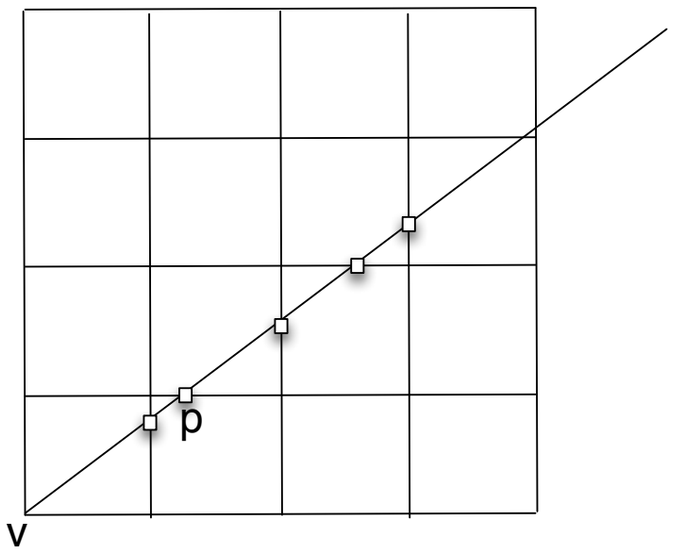
- For each p in grid
  - compute intersections between *vp* and grid lines
  - if all these points are below *vp* then *p* is visible

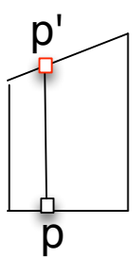
Assume grid of n points
($\sqrt{n}$ x $\sqrt{n}$)
Running time: $O(n\sqrt{n})$

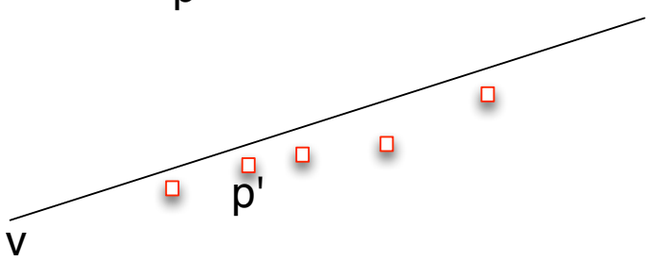


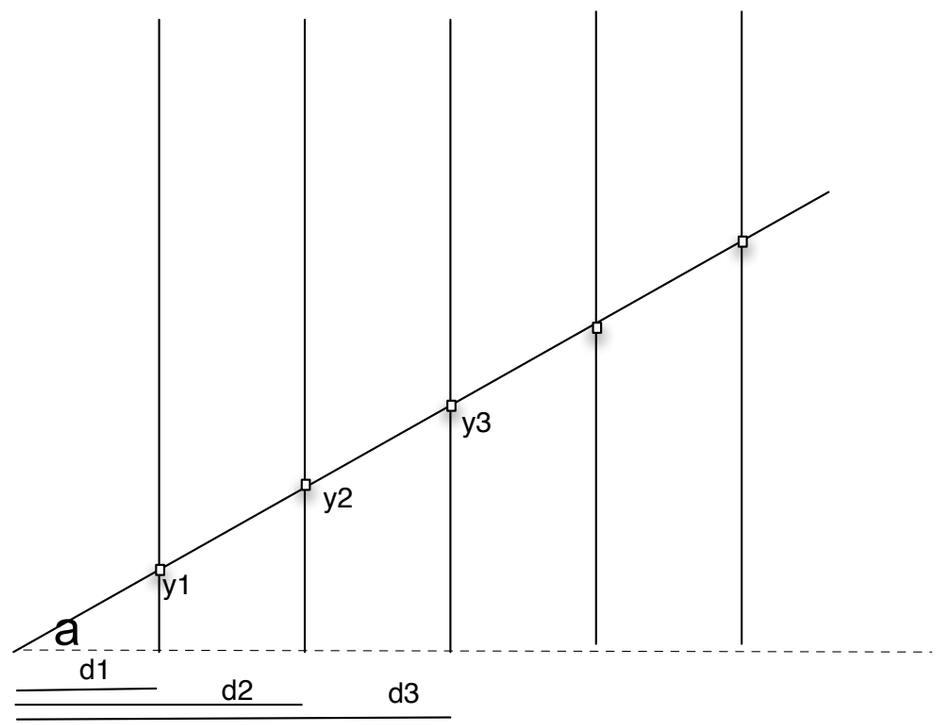
Basic viewshed algorithm

1. Find the 2D intersections between vp and the grid lines
2. Lift to 3D: find the height of p by linear interpolation
3. Check if slope(vp') is below slope (LOS)









y1 = d1 tan a
y2 = d2 tan a

Viewshed on grids

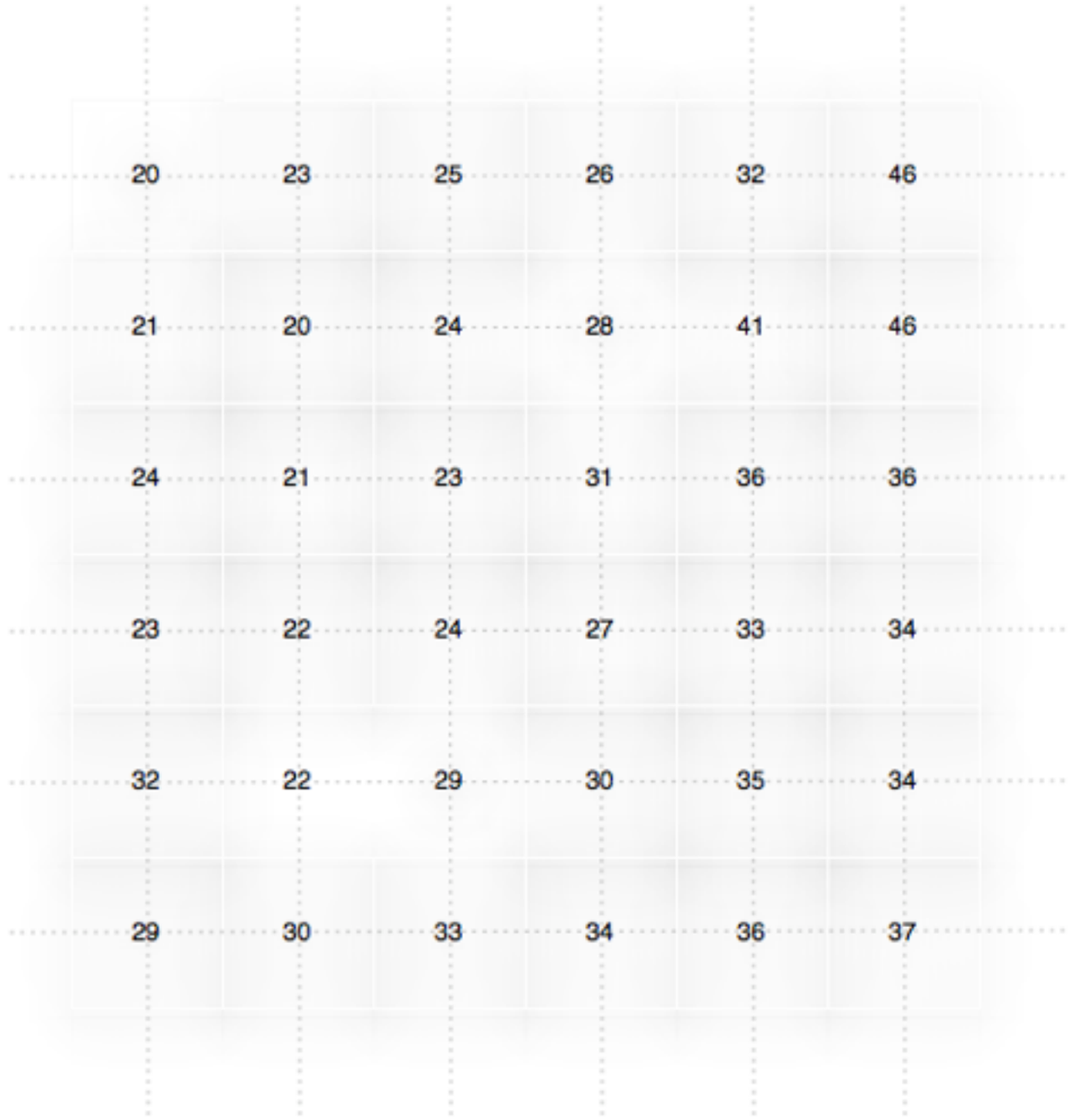
Grid of n points:
$\sqrt{n} \times \sqrt{n}$

- The straightforward O($n\sqrt{n}$) algorithm
  - uses linear interpolation
  - “exact” as much as data allows; uses all data available

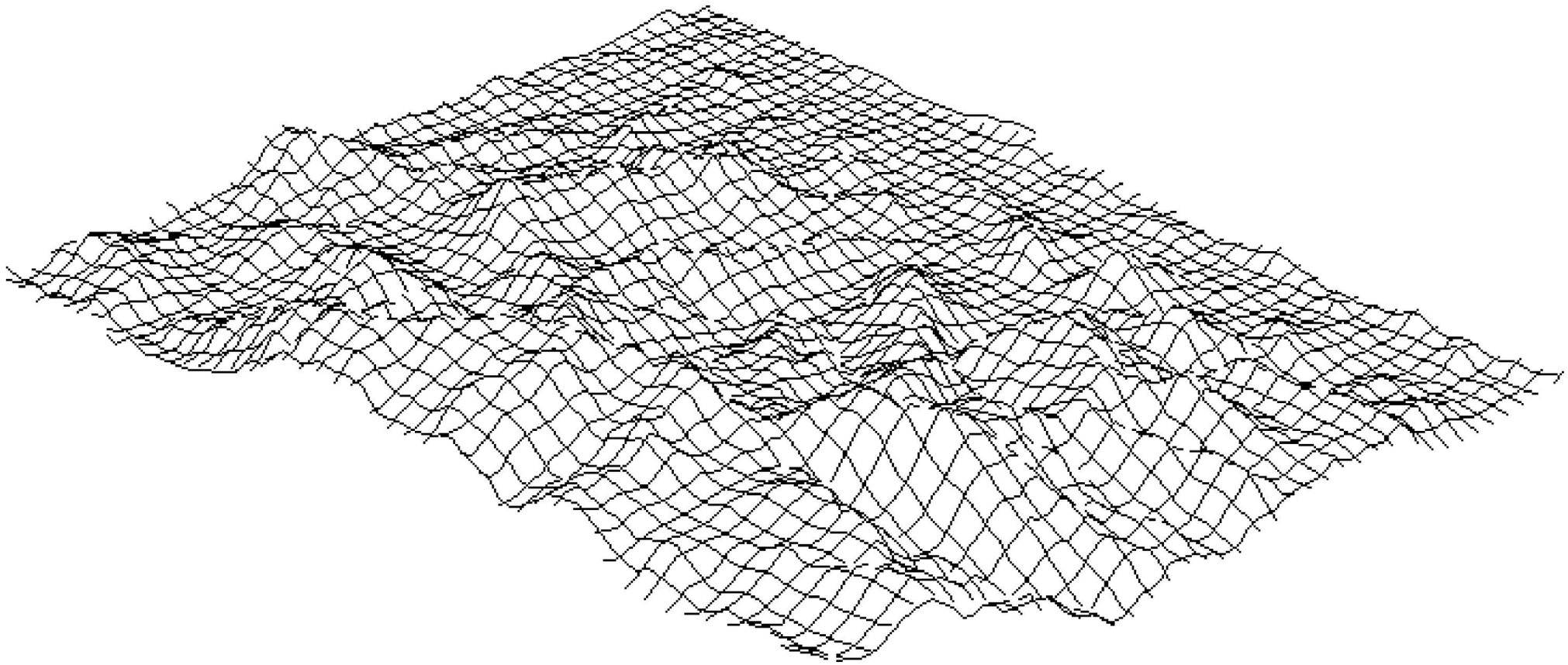
- Can we do better (faster) without introducing any approximation?

- Van Kreveld[vK’96]
  - nearest neighbor interpolation <——- simpler
  - O (n lg n)

Grids

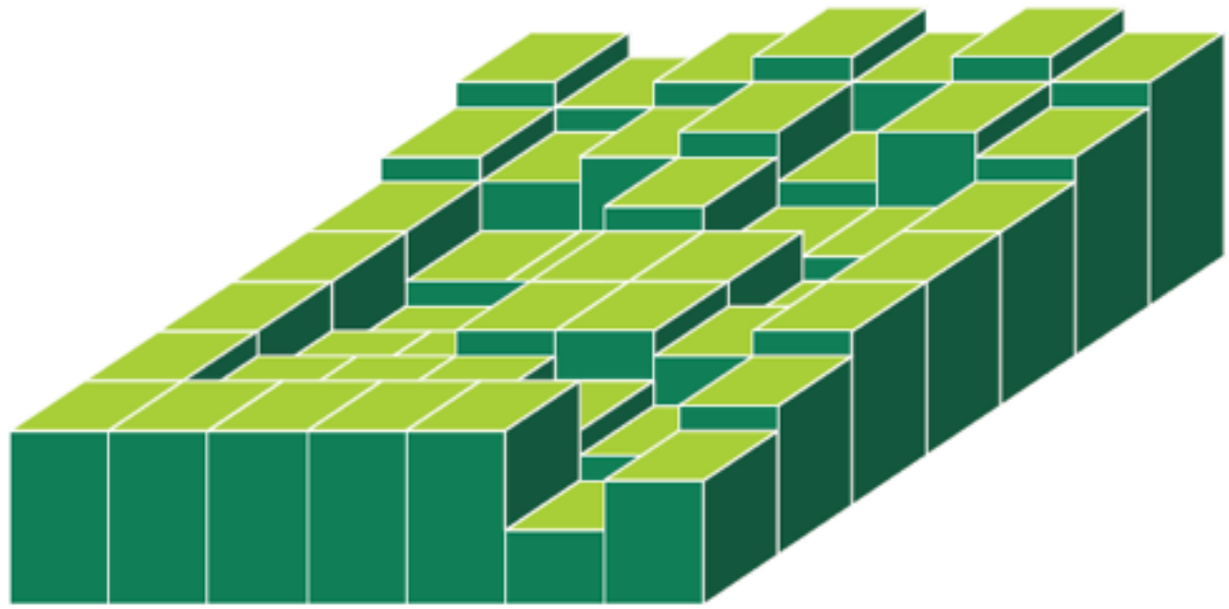

20 23 25 26 32 46
21 20 24 28 41 46
24 21 23 31 36 36
23 22 24 27 33 34
32 22 29 30 35 34
29 30 33 34 36 37

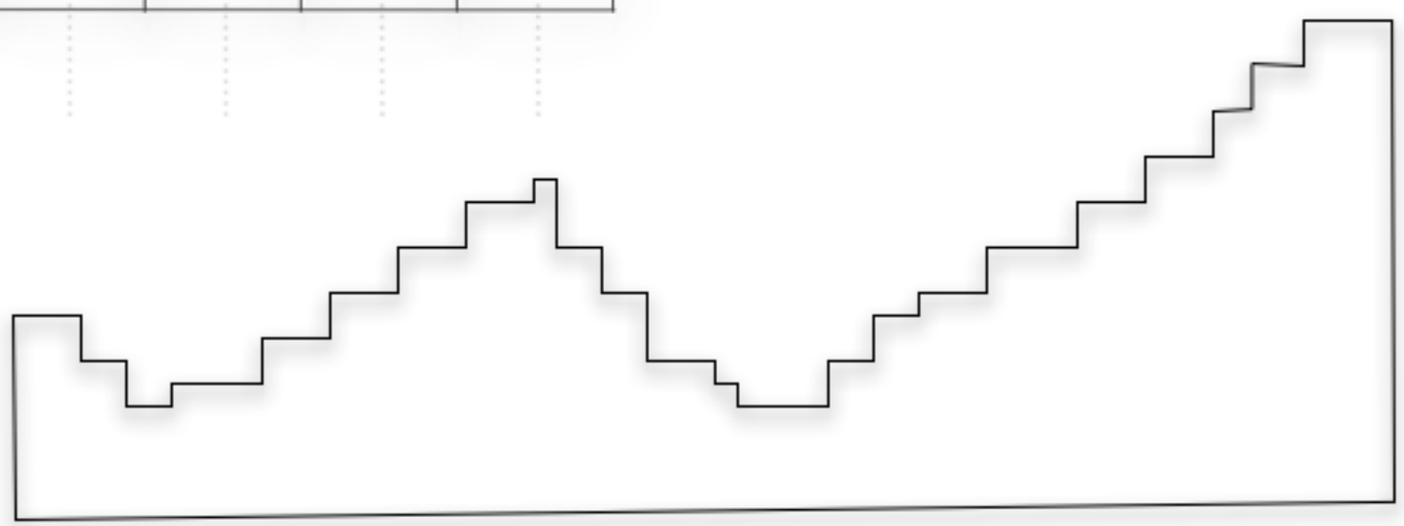
Grids with linear interpolation



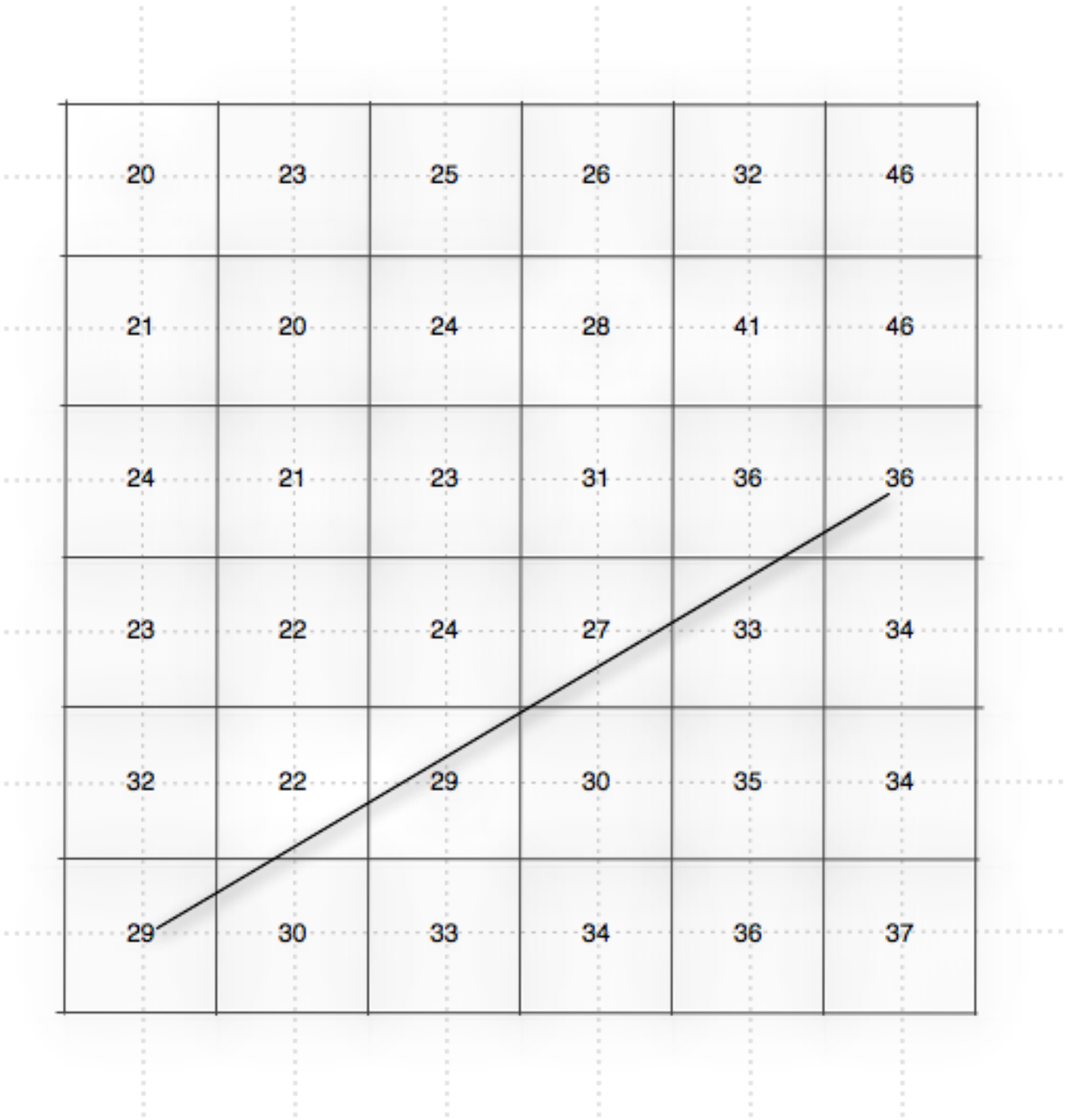
Grids with nearest neighbor interpolation

20	23	25	26	32	46
21	20	24	28	41	46
24	21	23	31	36	36
23	22	24	27	33	34
32	22	29	30	35	34
29	30	33	34	36	37

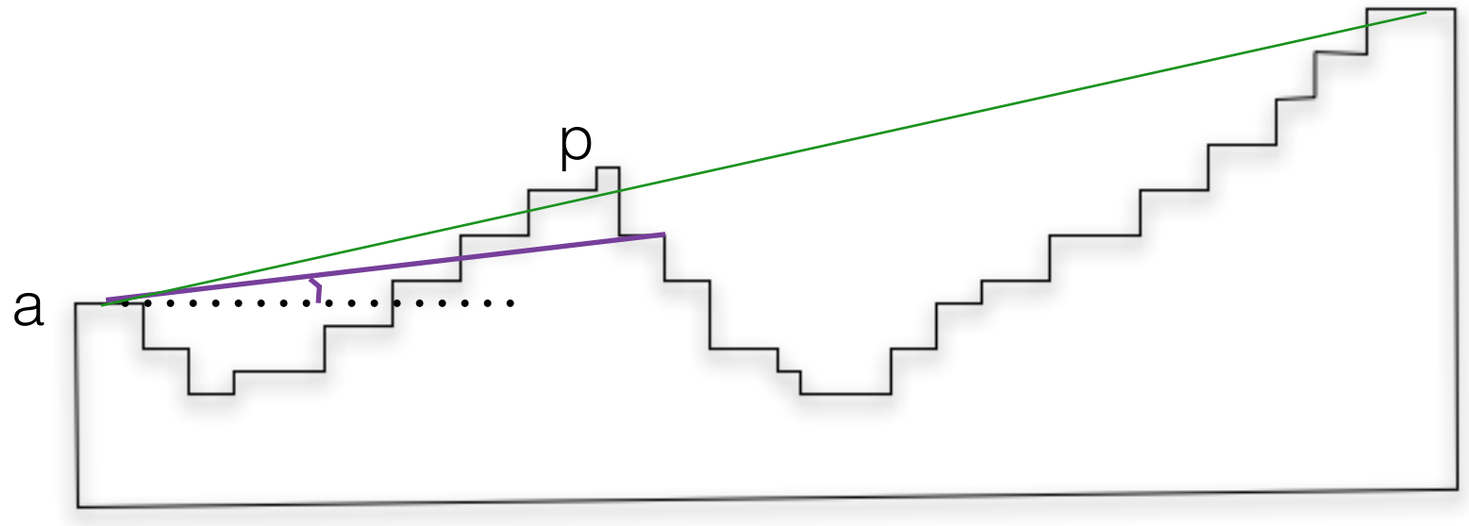




20	23	25	26	32	46
21	20	24	28	41	46
24	21	23	31	36	36
23	22	24	27	33	34
32	22	29	30	35	34
29	30	33	34	36	37


20 23 25 26 32 46
21 20 24 28 41 46
24 21 23 31 36 36
23 22 24 27 33 34
32 22 29 30 35 34
29 30 33 34 36 37

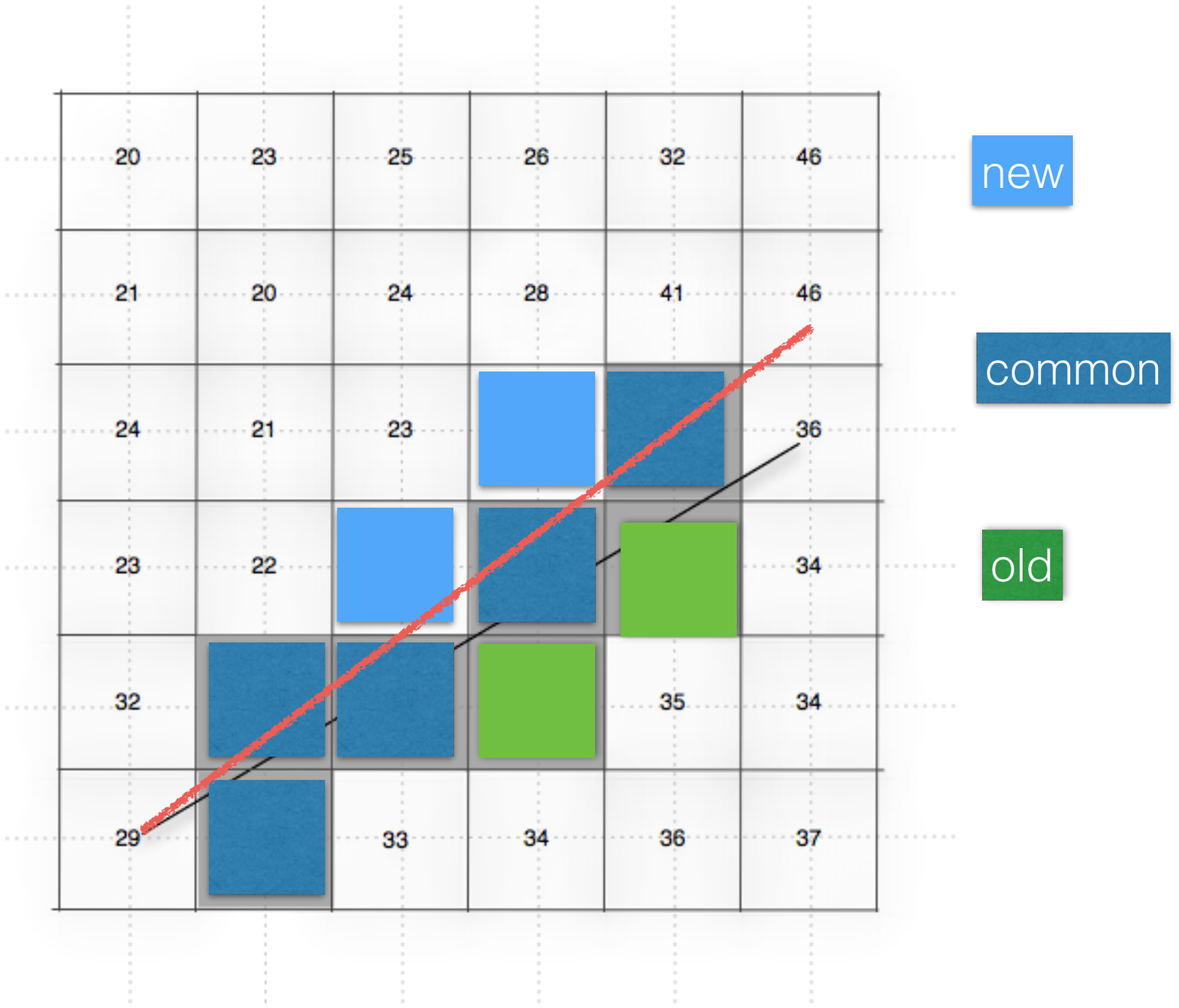
20	23	25	26	32	46
21	20	24	28	41	46
24	21	23	31	36	36
23	22	24	27	33	34
32	22	29	30	35	34
29	30	33	34	36	37

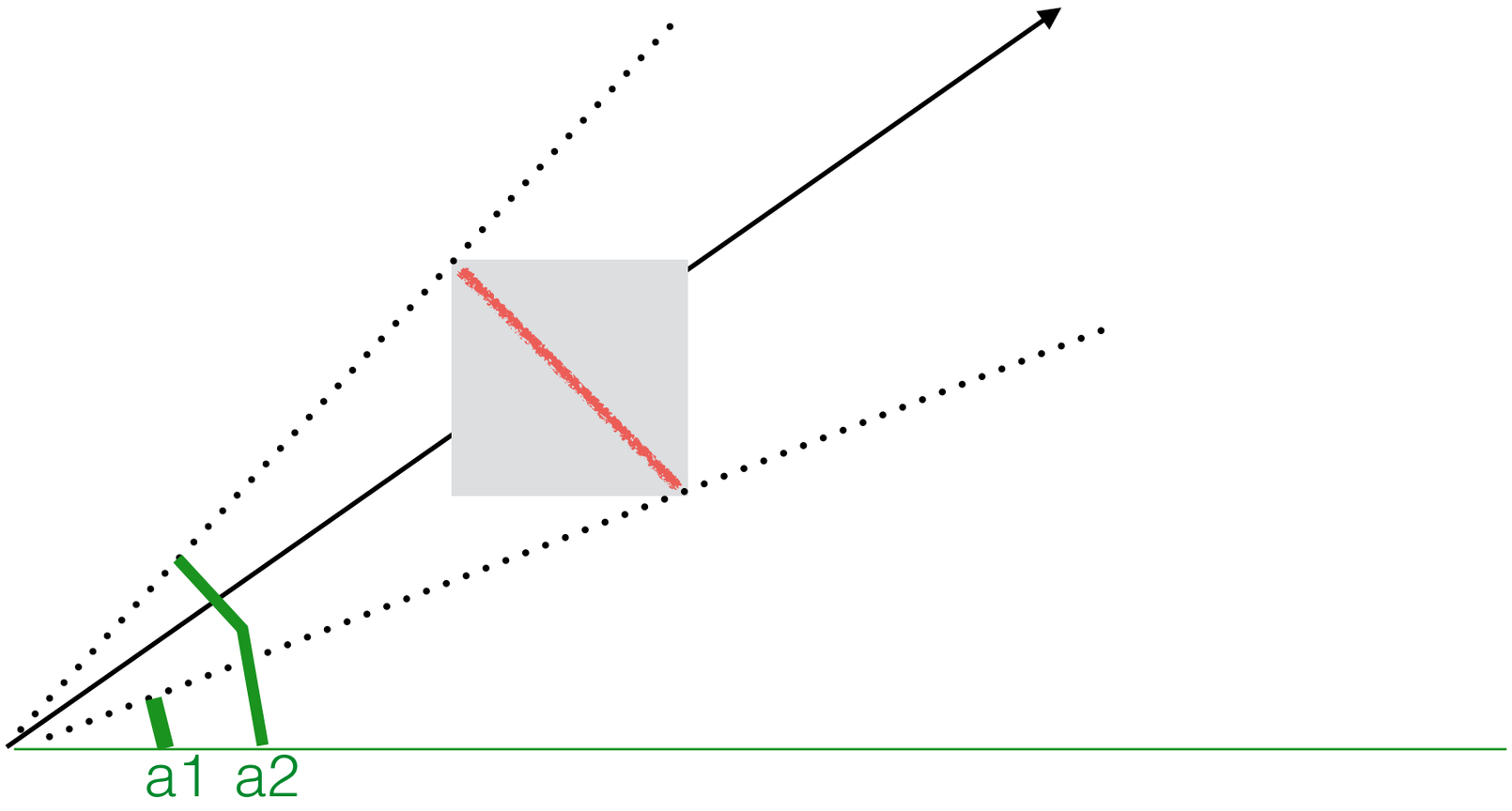


vertical slope(p,a) = (h_p - h_a) / d(a,p)

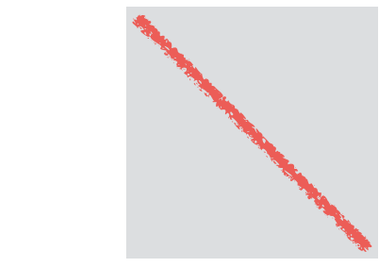
20	23	25	26	32	46
21	20	24	28	41	46
24	21	23	31	36	36
23	22	24	27	33	34
32	22	29	30	35	34
29	30	33	34	36	37

20	23	25	26	32	46
21	20	24	28	41	46
24	21	23	31	36	36
23	22	24	27	33	34
32	22	29	30	35	34
29	30	33	34	36	37

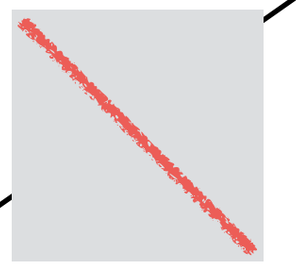

20 23 25 26 32 46
21 20 24 28 41 46
24 21 23 36
23 22 34
32 35 34
29 33 34 36 37
new
common
old

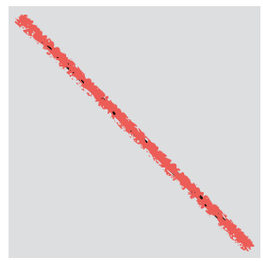

a1
a2

future



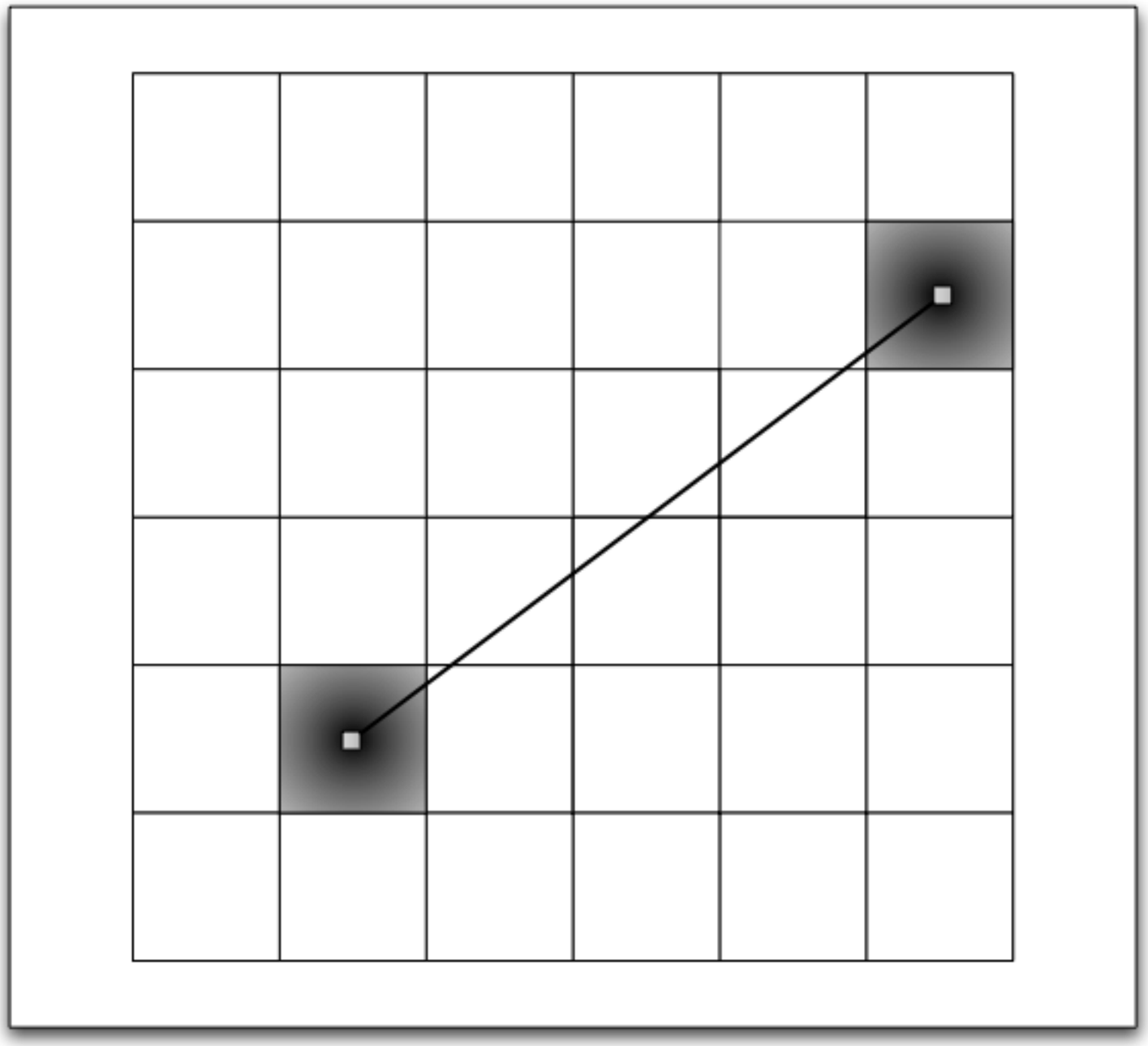
alive



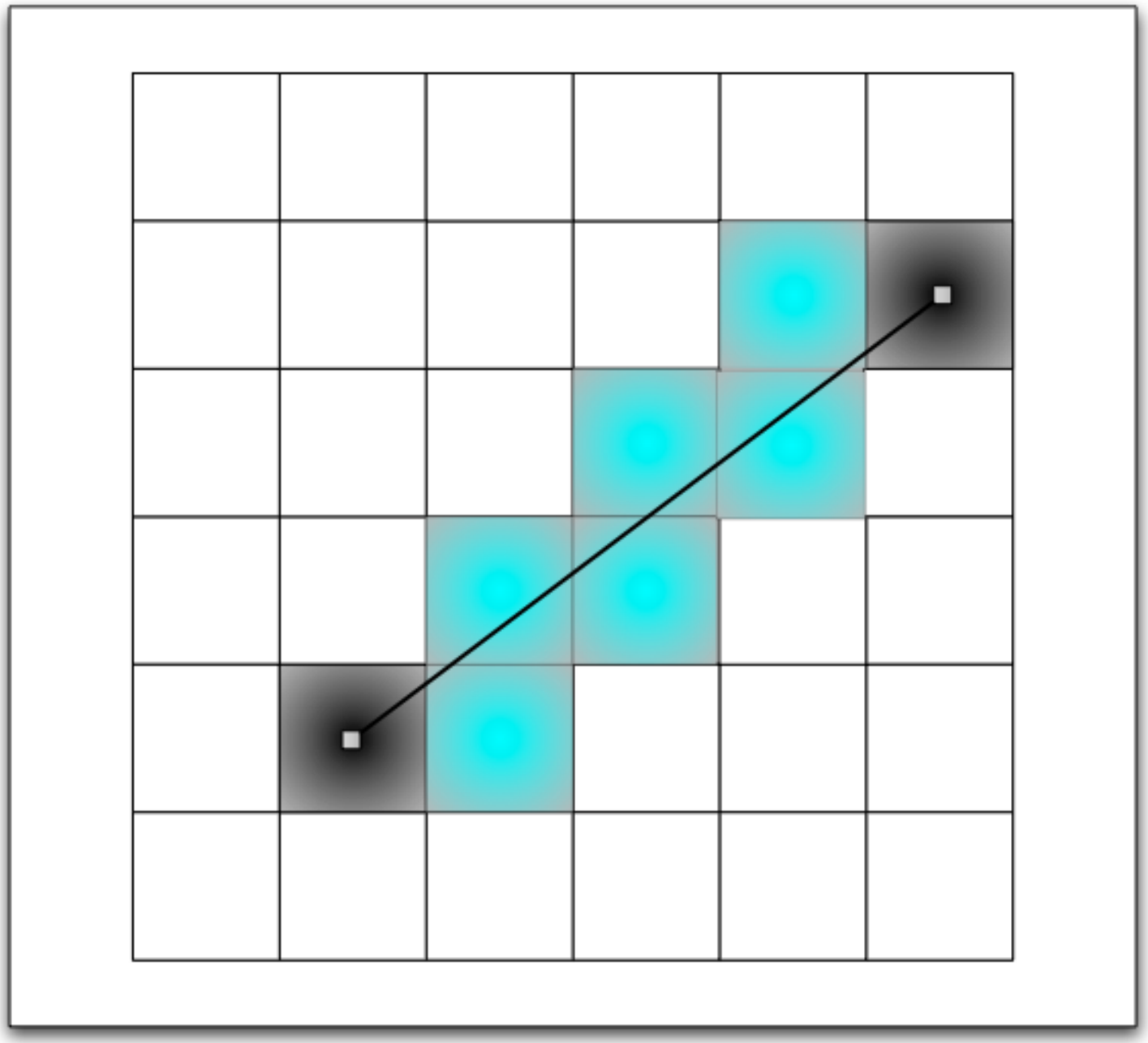


dead

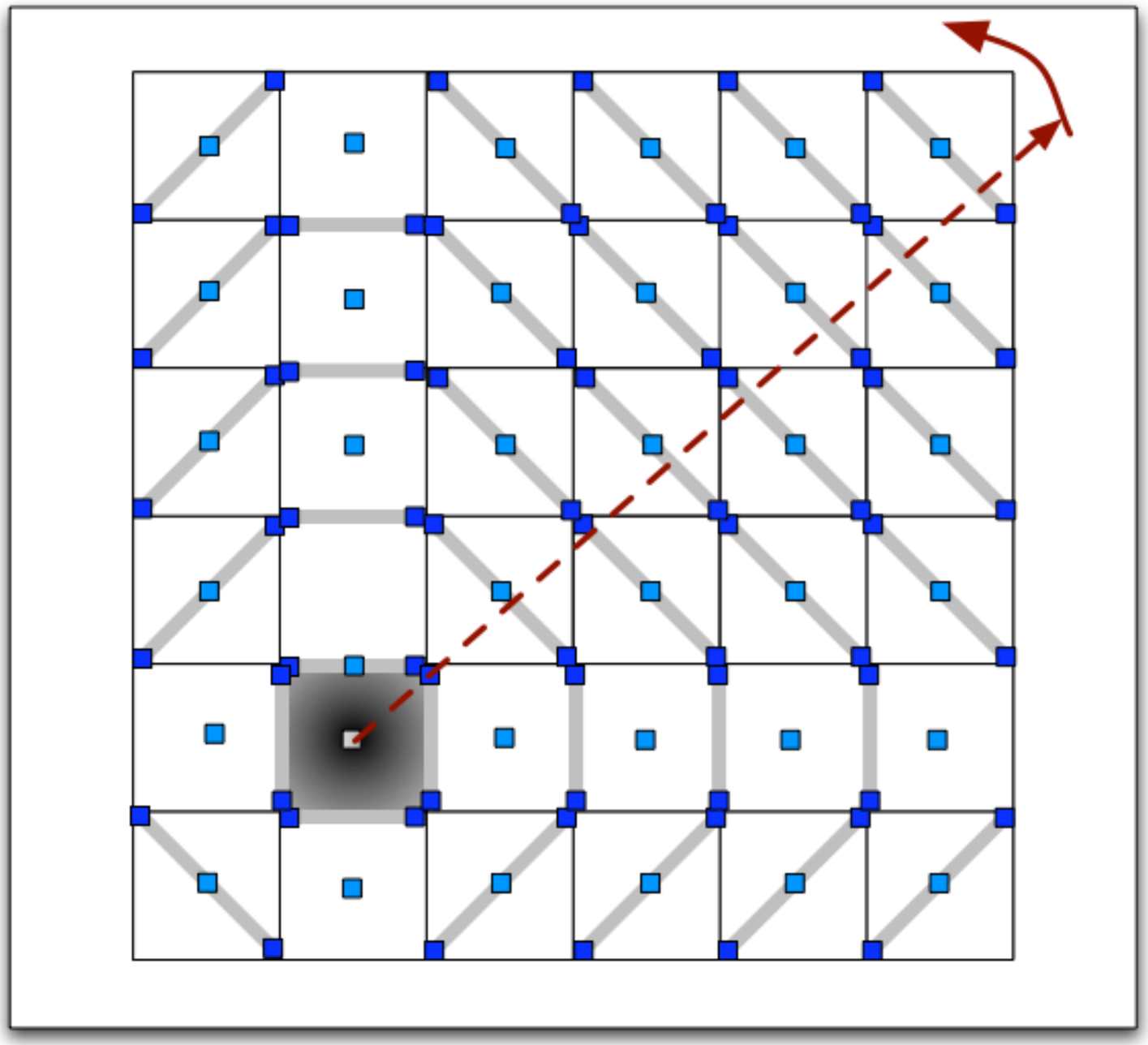
Van Kreveld’s radial sweep algorithm



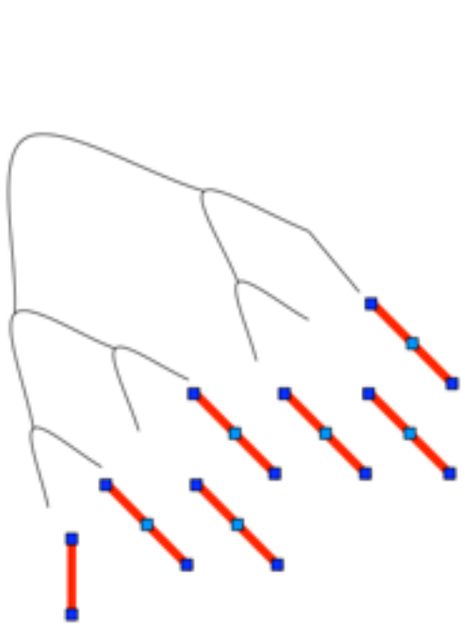
Van Kreveld's radial sweep algorithm

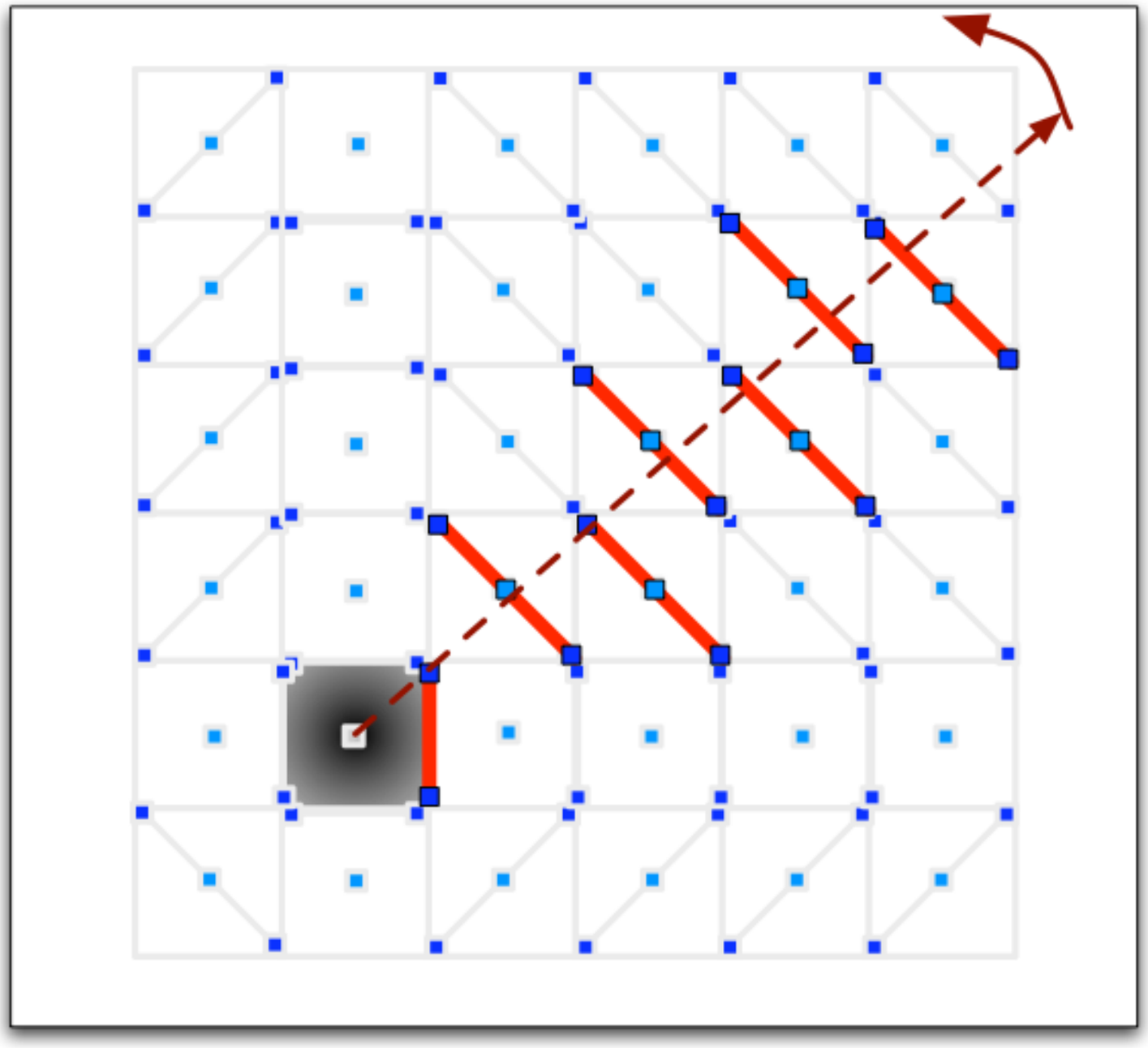


Van Kreveld's radial sweep algorithm



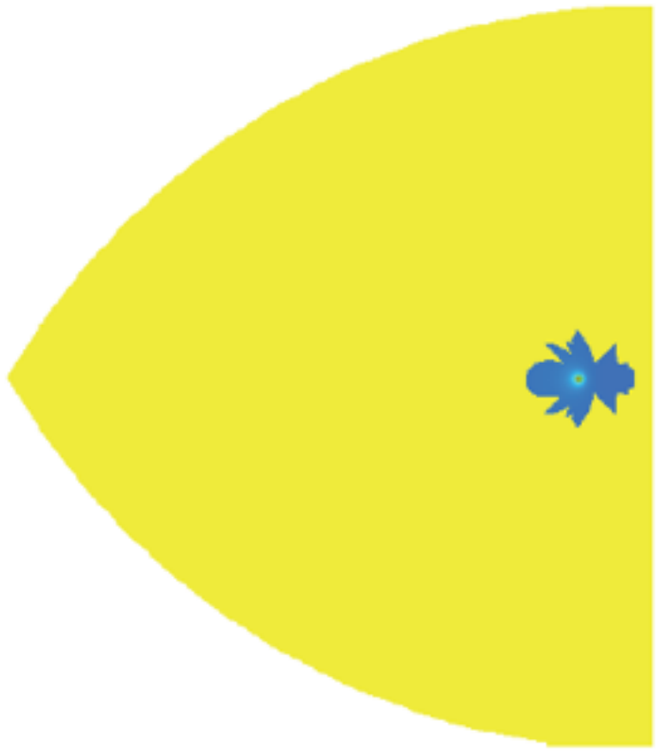
Van Kreveld's radial sweep algorithm



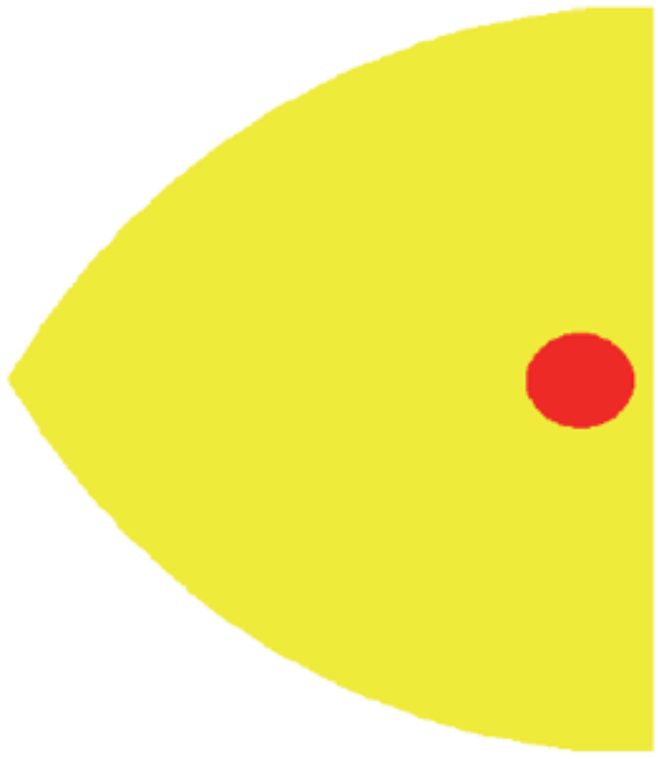


Accuracy!!

test grid: hemisphere



viewshed with NN interpolation



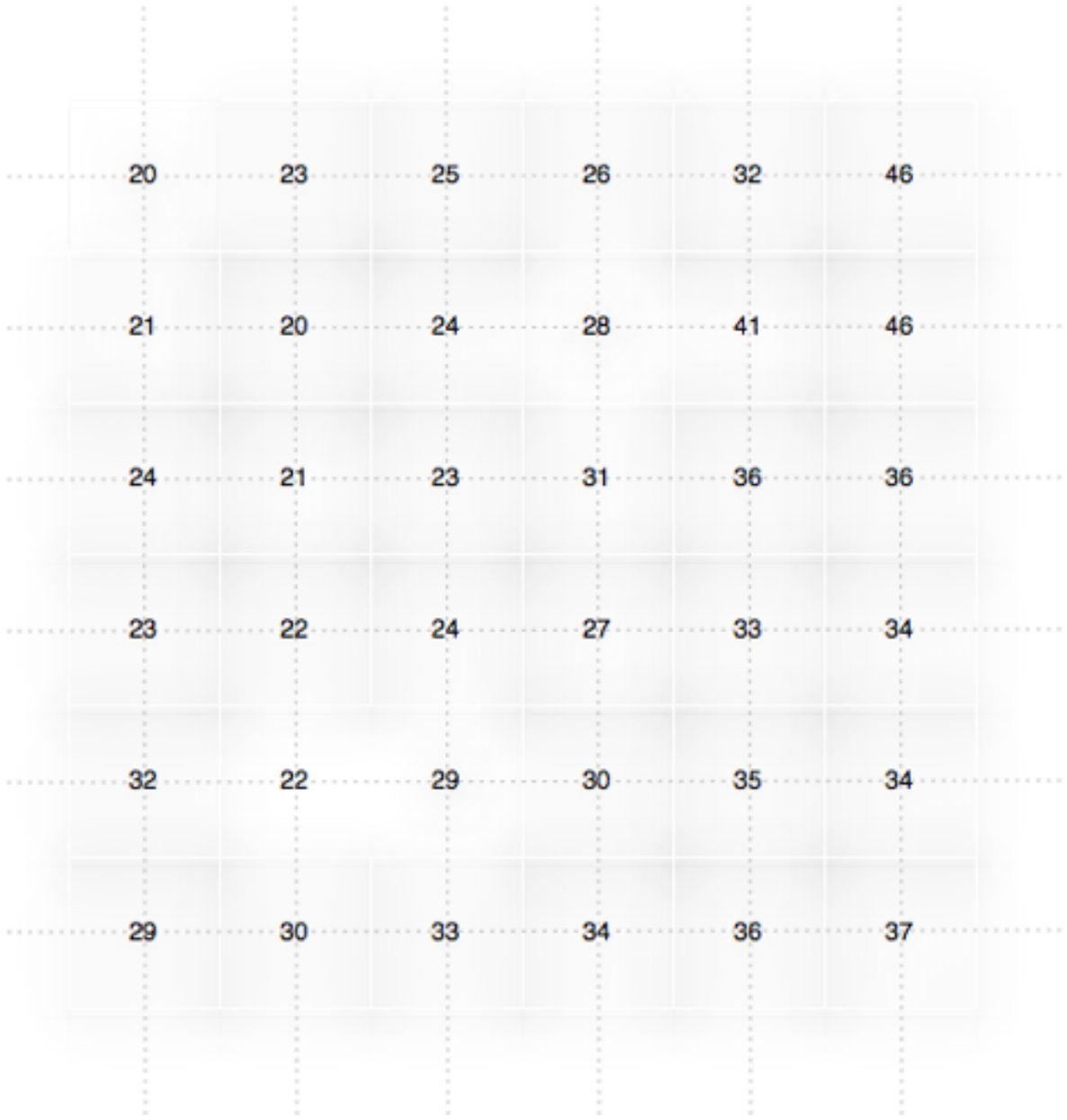
viewshed with linear interpolation

Computing viewsheds

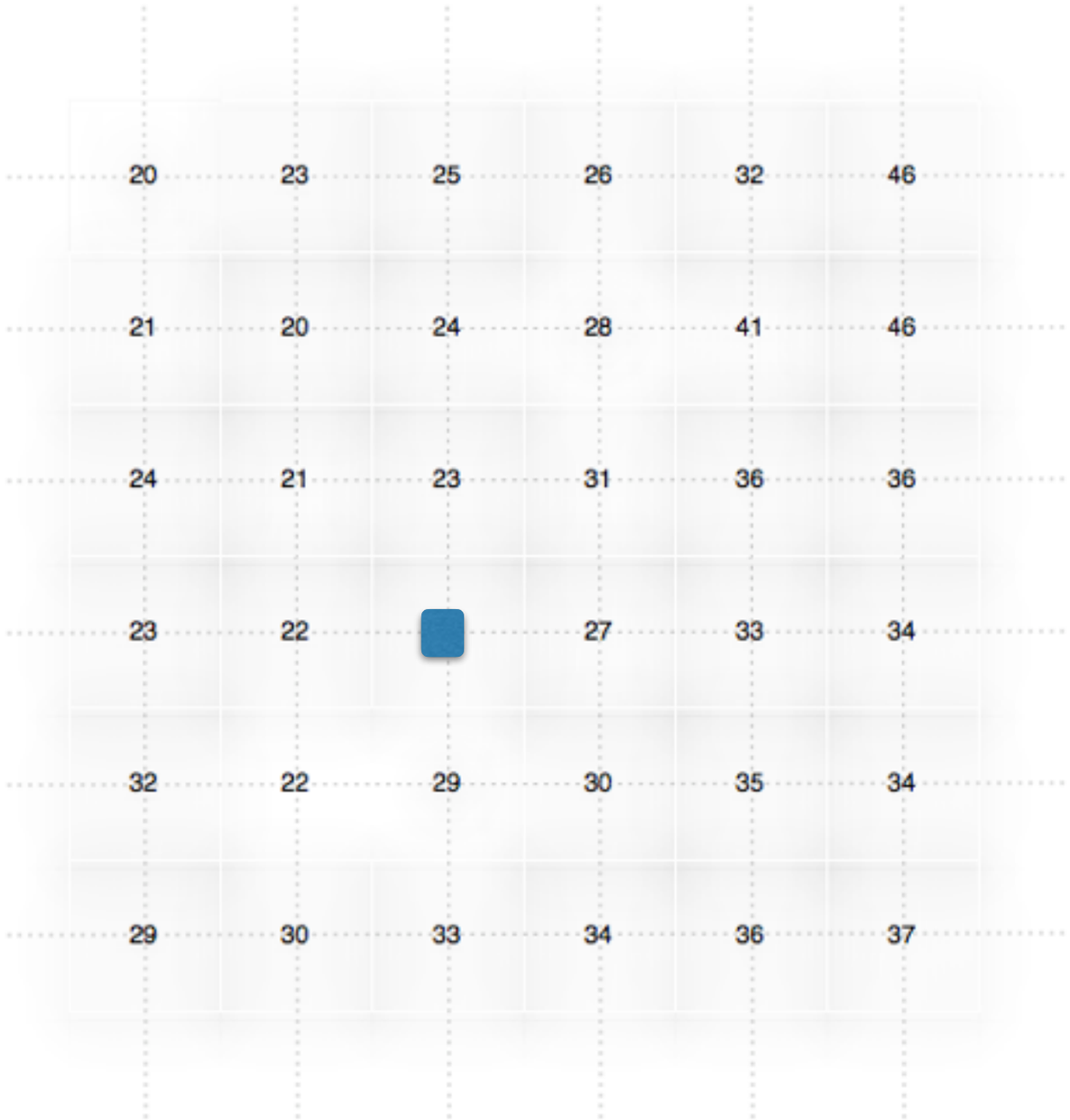
Grid of n points:
$\sqrt{n} \times \sqrt{n}$

1. Straightforward algorithm
   - uses linear interpolation
   - can be adapted to other interpolations
   - $O(n\sqrt{n})$
2. Radial sweep approach
   - uses nearest neighbor interpolation
   - crucially exploits that cells are “flat” obstacles
   - uses radial sweep + augmented RB tree
   - has accuracy issues ==> undesirable
   - O(n lg n)
3. A different approach: concentric sweep using horizons

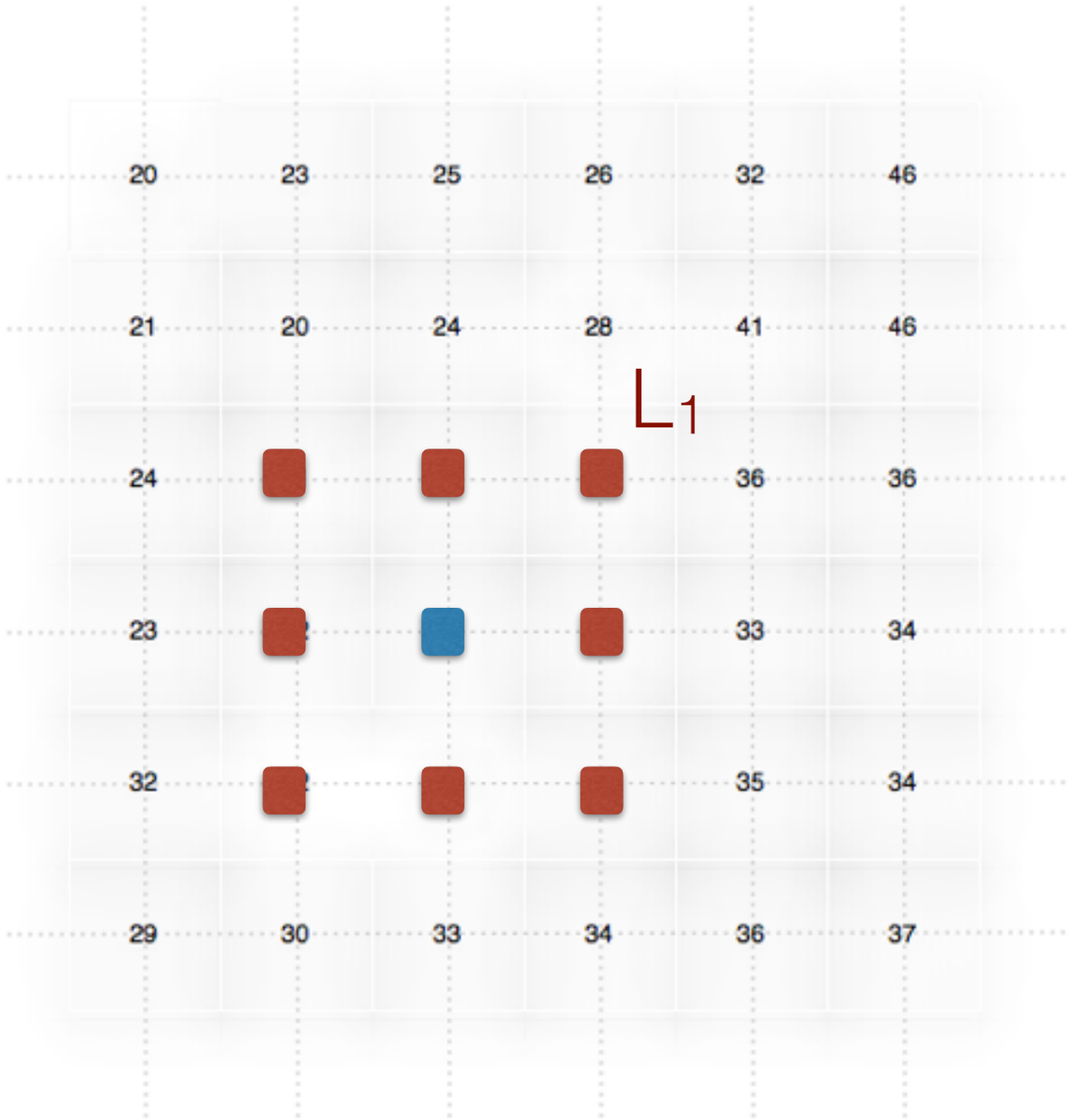
Viewshed on grids using a concentric sweep



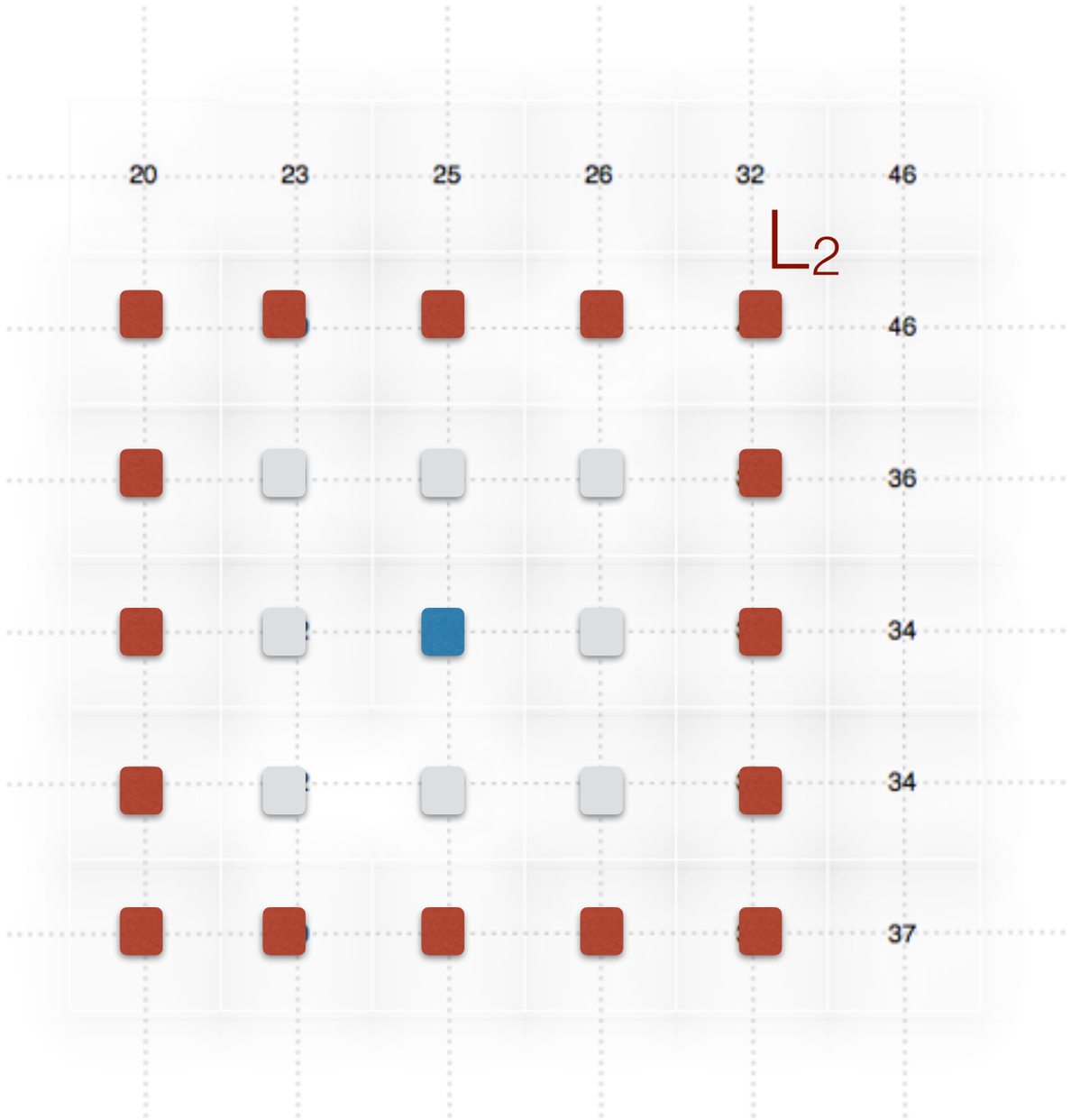
Viewshed on grids using a concentric sweep



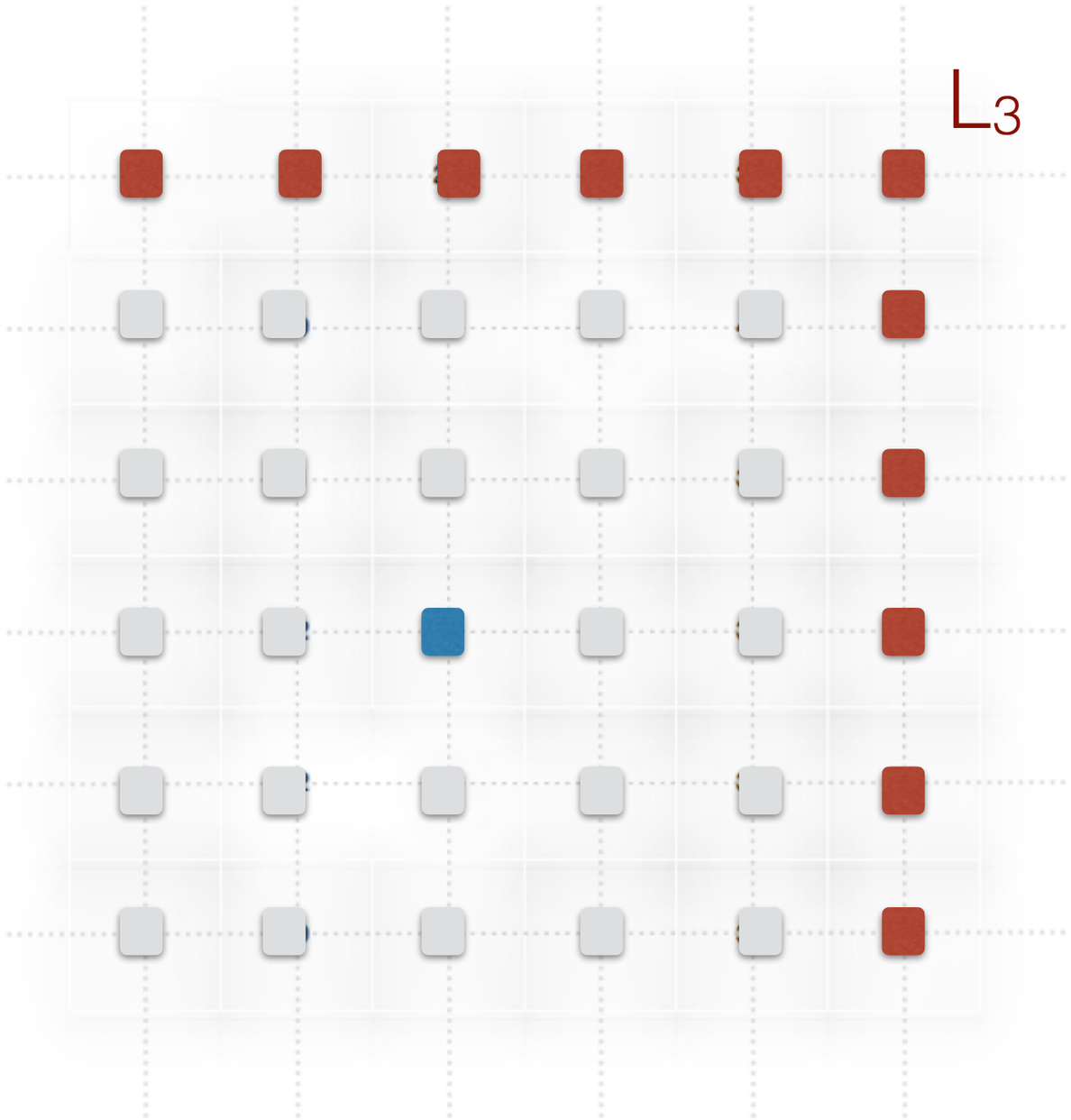
Viewshed on grids using a concentric sweep



Viewshed on grids using a concentric sweep



Viewshed on grids using a concentric sweep



Horizons

- Merriam Webster:
  - the line where the terrain and the sky seem to meet

Horizons

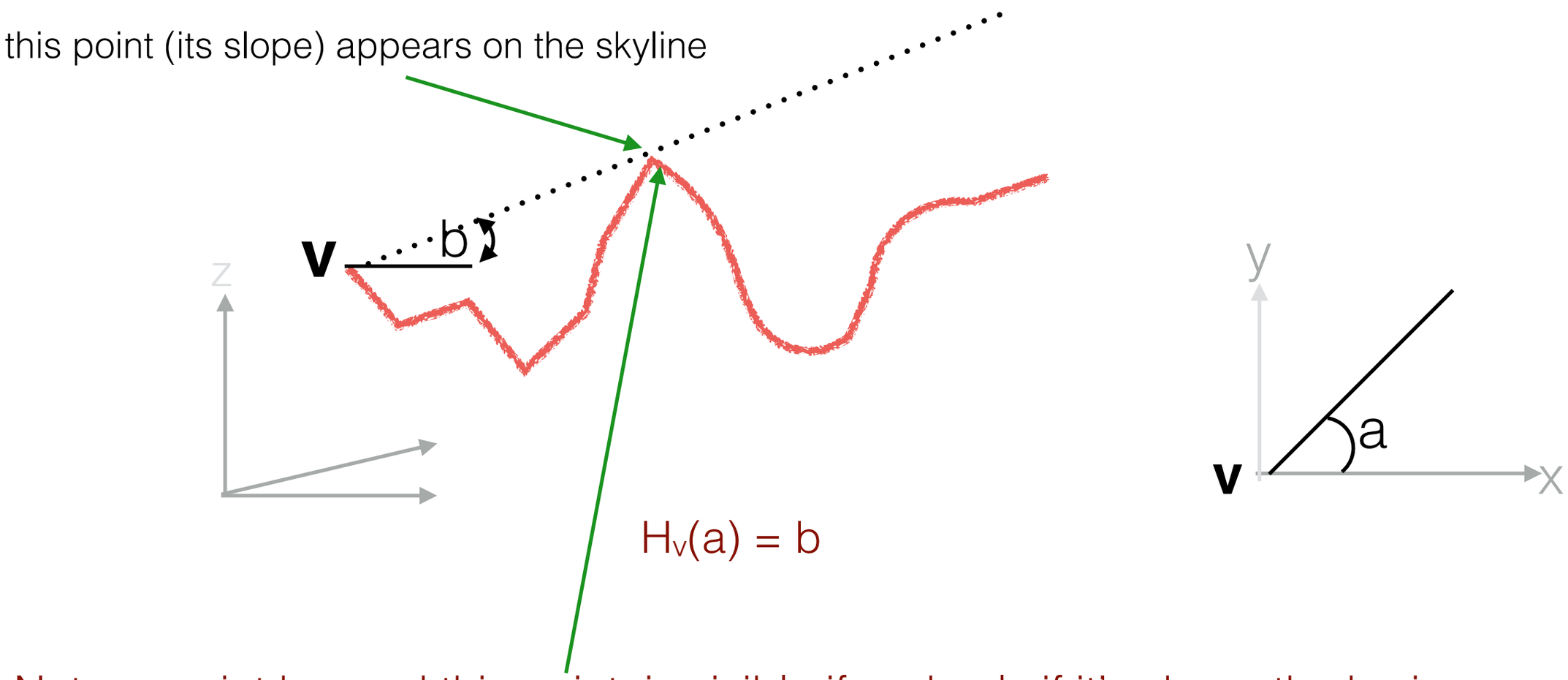
- Merriam Webster:
  - the line where the terrain and the sky seem to meet

Horizon

H_v : [0, 2PI) ——> R

Horizon (with respect to v) in direction a, $H_v(a)$

- cut the terrain with a vertical plane through a ray from v of azimuth a
- $H_v(a)$ is the maximum vertical angle (zenith) of all points intersected by this plane (all the points on T whose projection on the xy-plane has azimuth a)



- Note: a point beyond this point is visible if and only if it's above the horizon

Horizons

- Merriam Webster:
  - the line where the terrain and the sky seem to meet

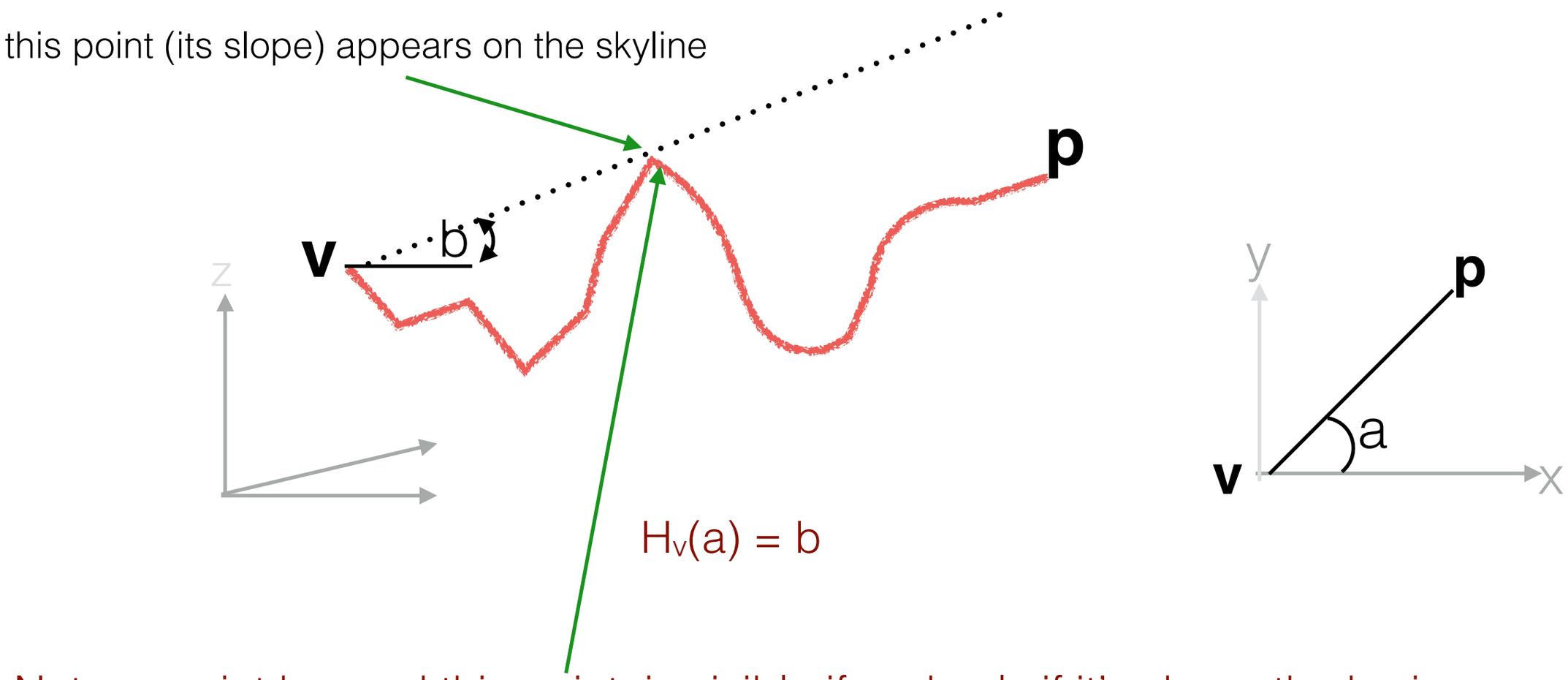
Horizon

Given point p on the terrain

Look at the cross-section of the terrain from v towards p, and consider only the points between v and p

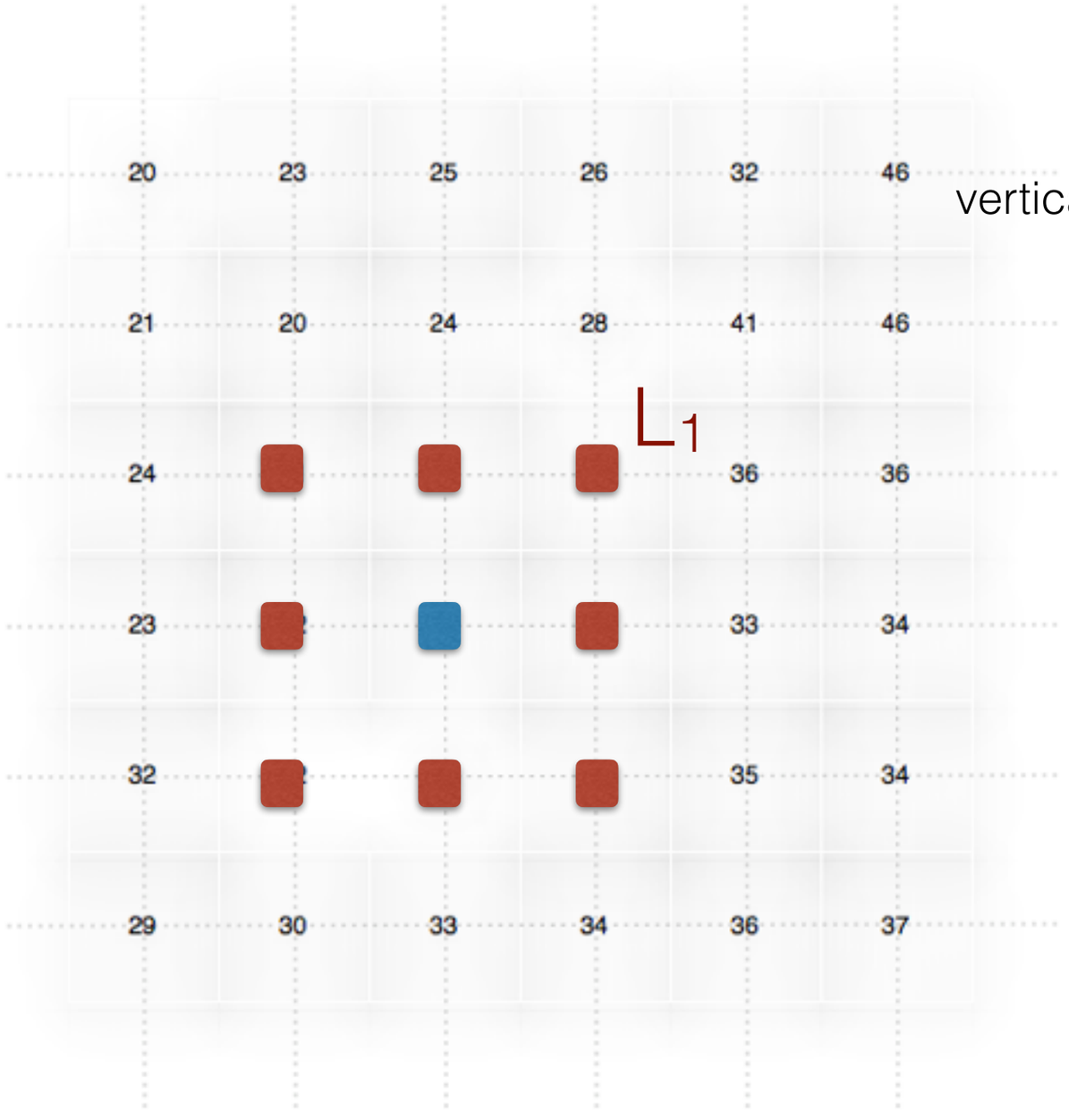
Point p defines the azimuth angle ==> We can define a horizon in the same way

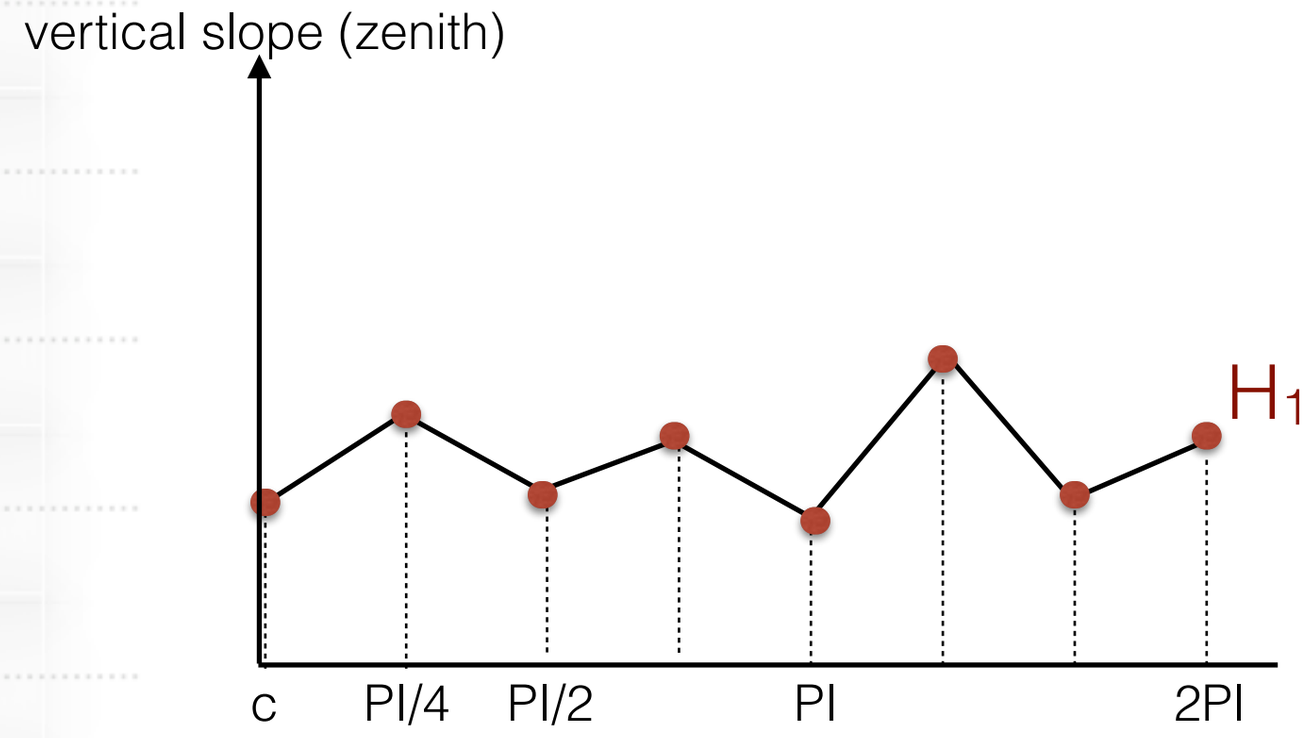
- $H_v(p)$ is the maximum vertical angle (zenith) of all points between v and p whose projection on the xy-plane has azimuth = azimuth(vp)



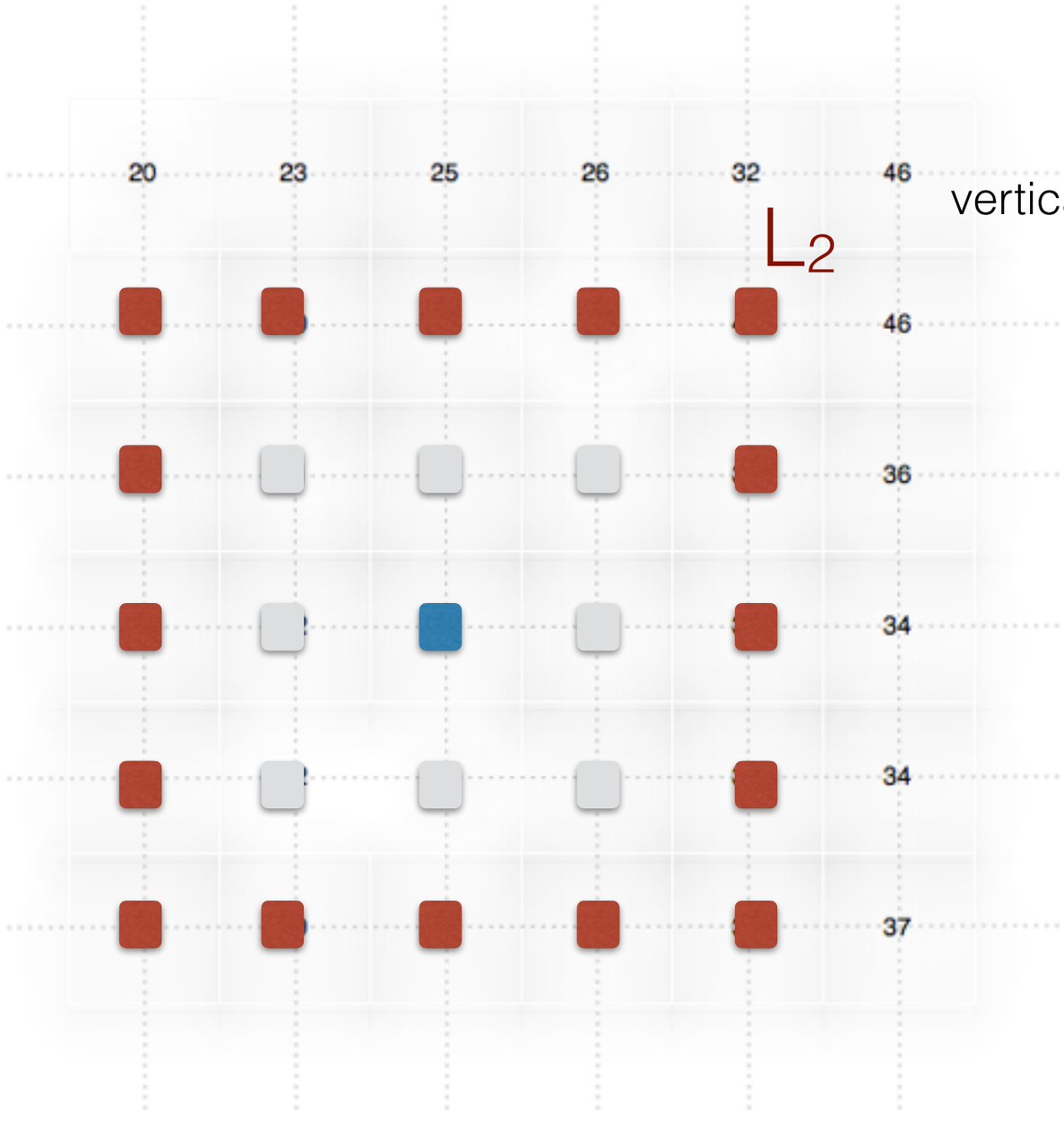
- Note: a point beyond this point is visible if and only if it's above the horizon

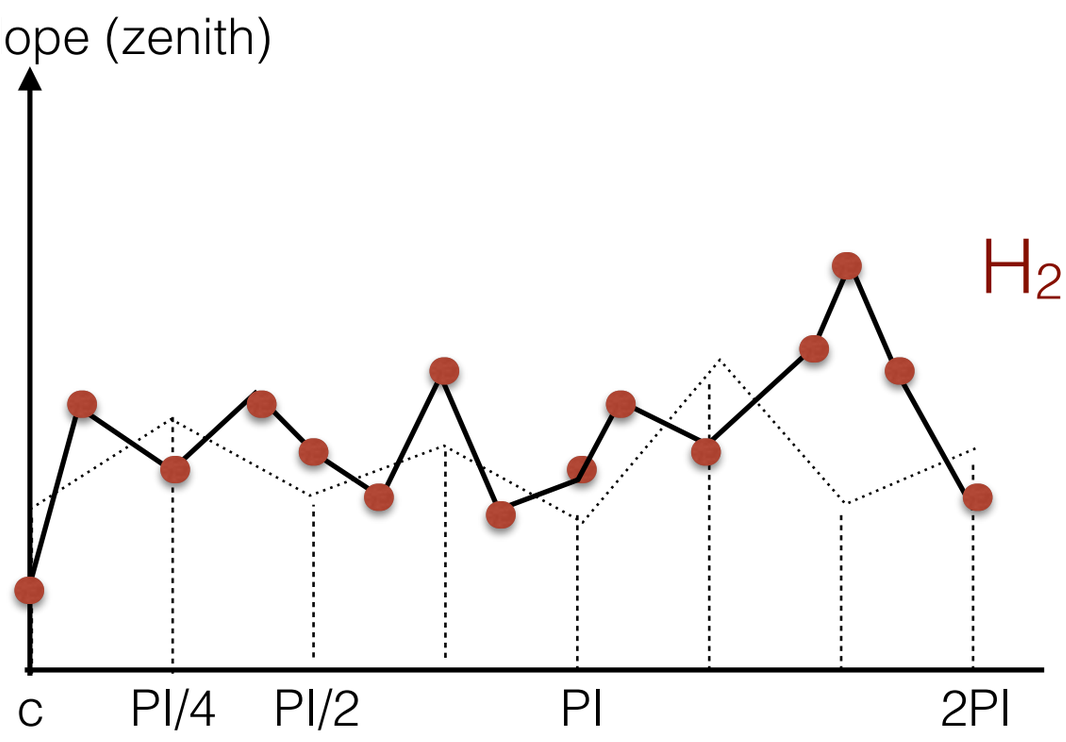
Viewshed on grids using a concentric sweep



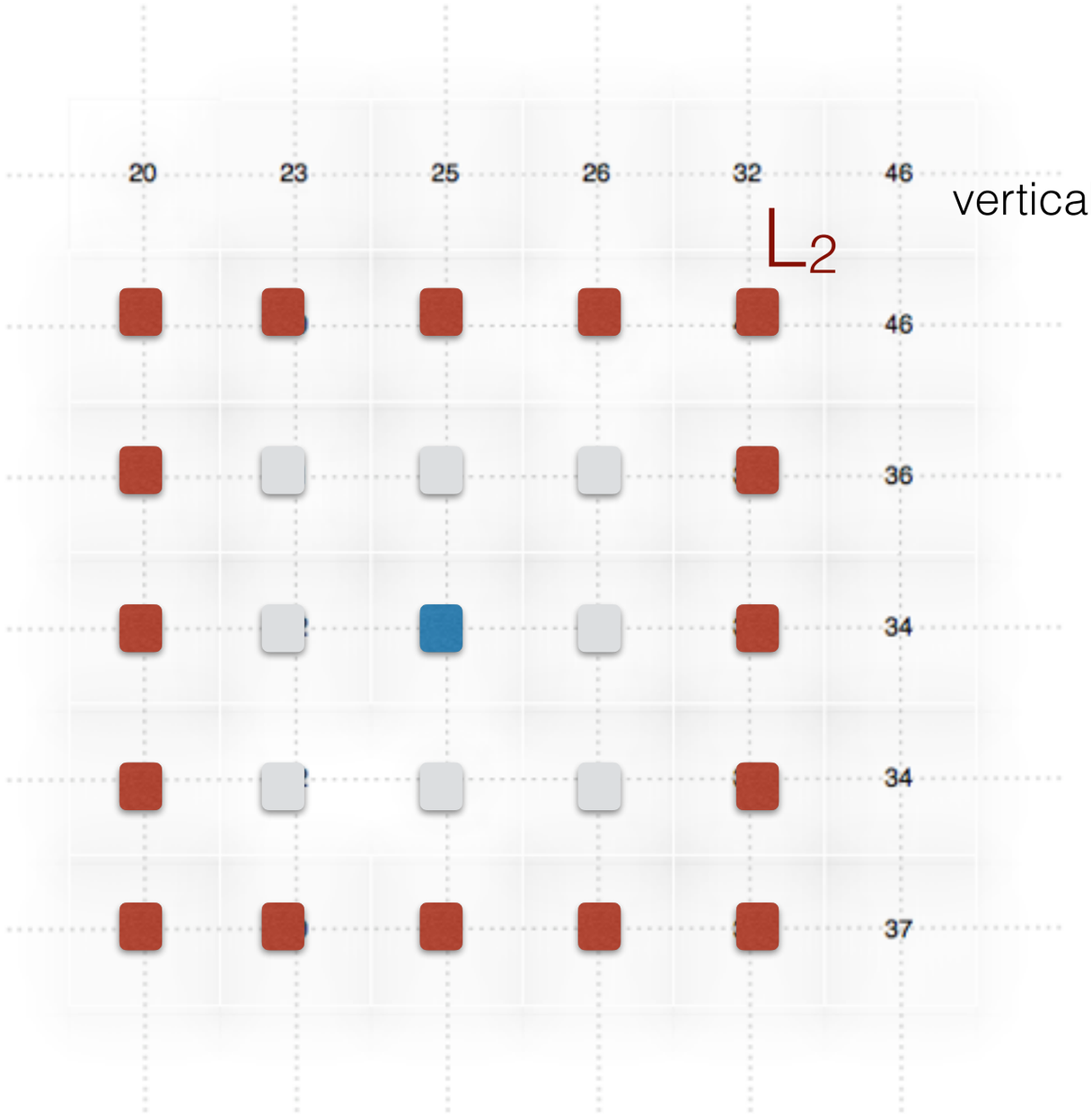


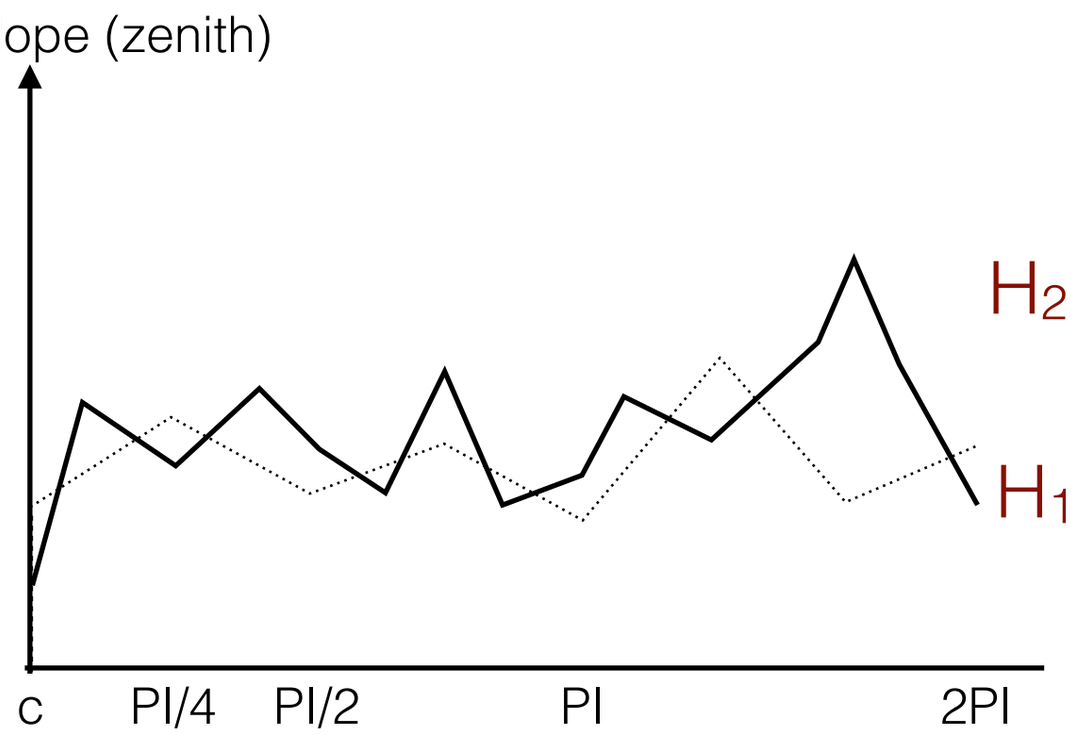
Viewshed on grids using a concentric sweep



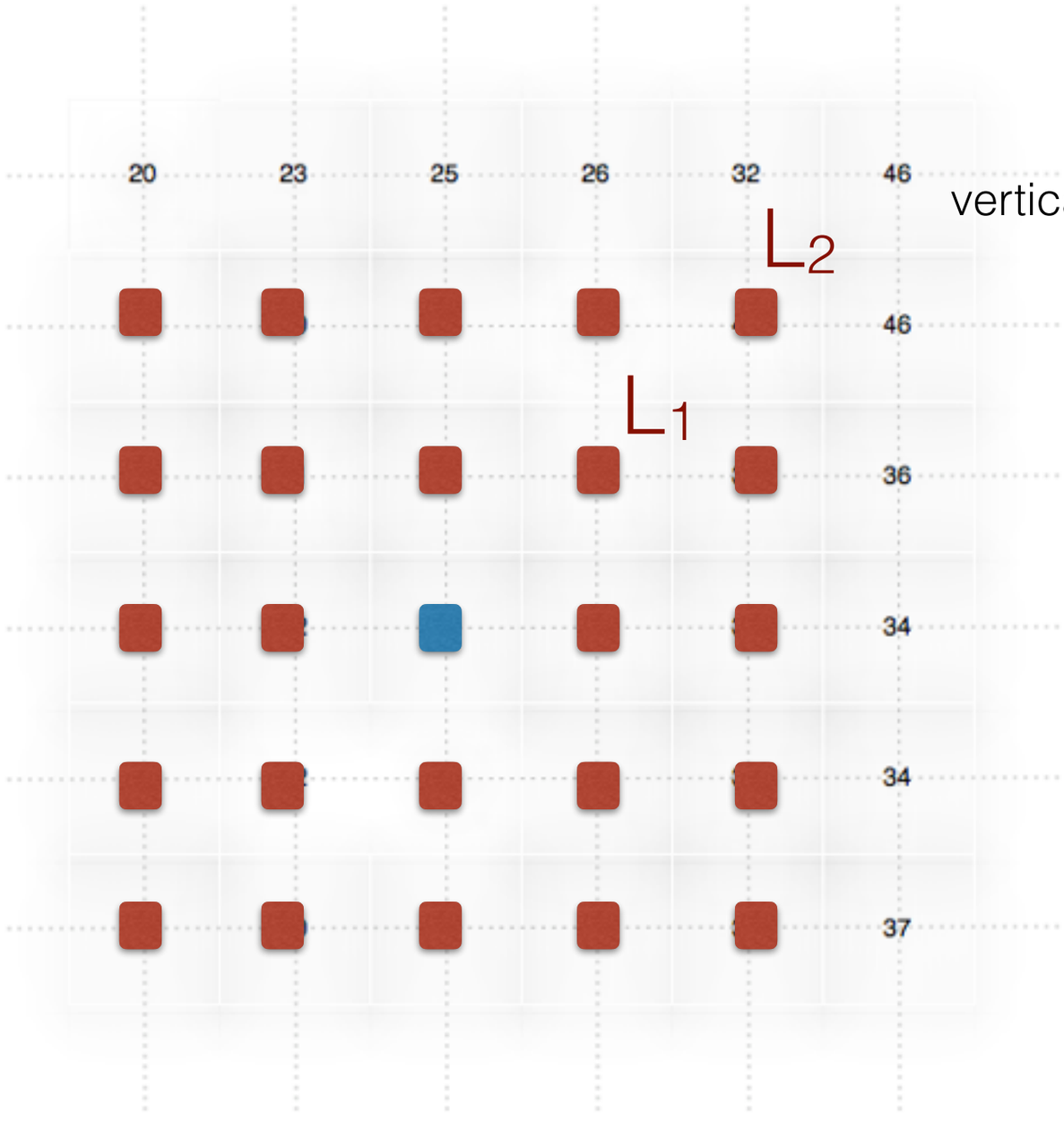


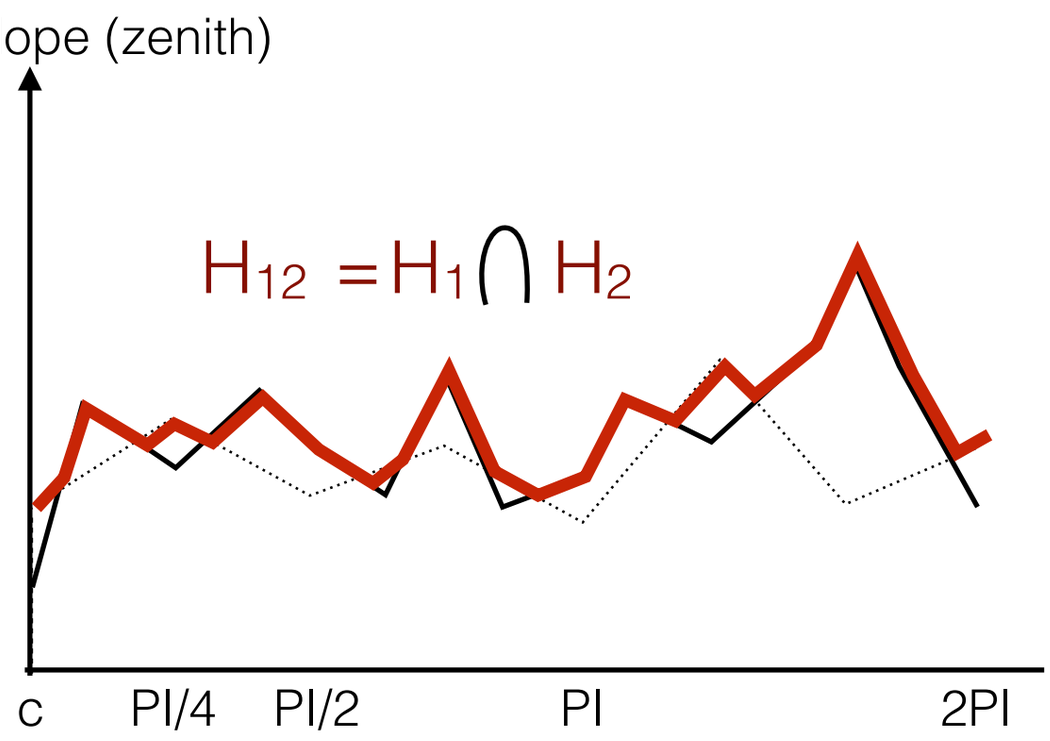
Viewshed on grids using a concentric sweep



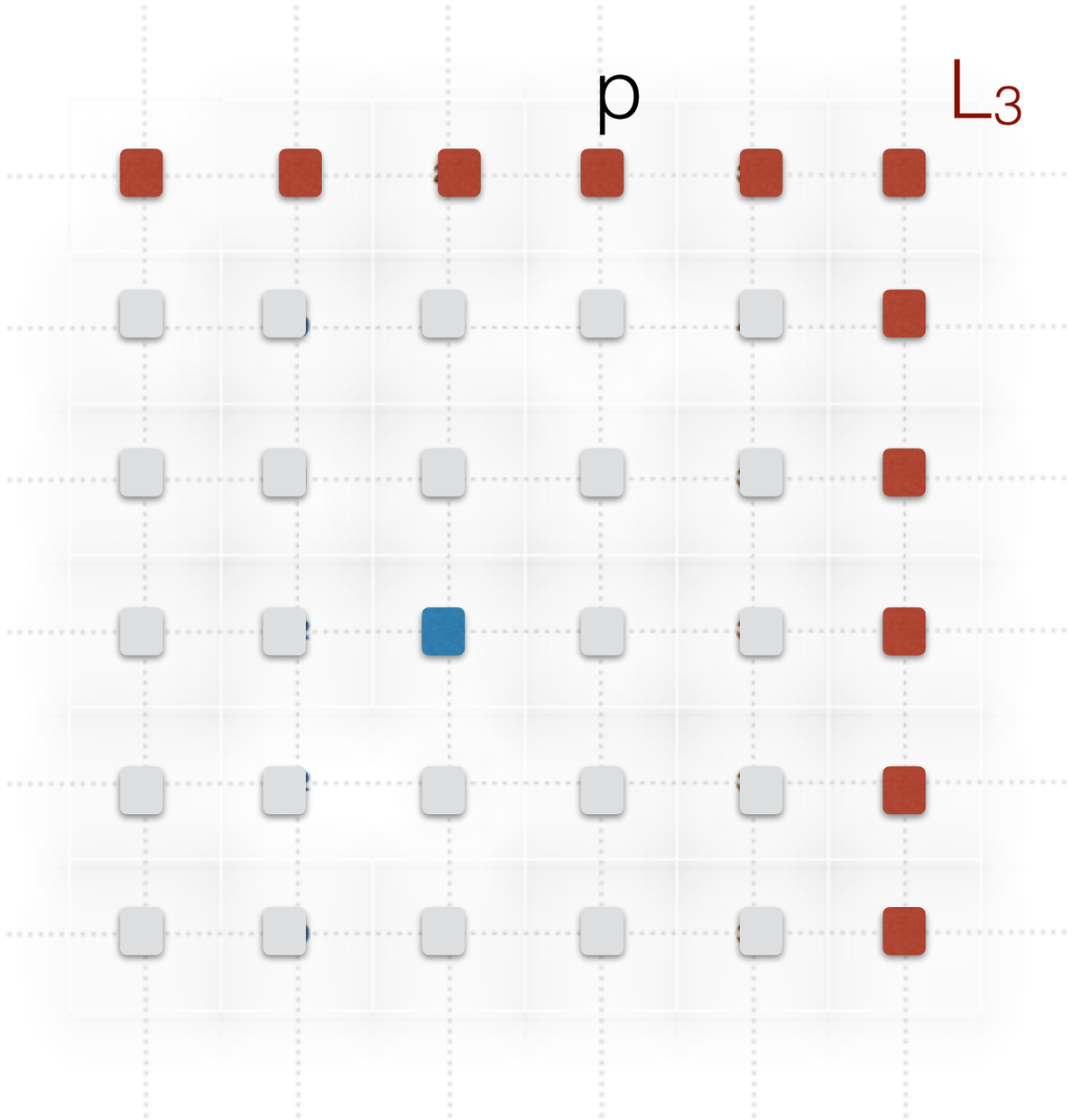


Viewshed on grids using a concentric sweep



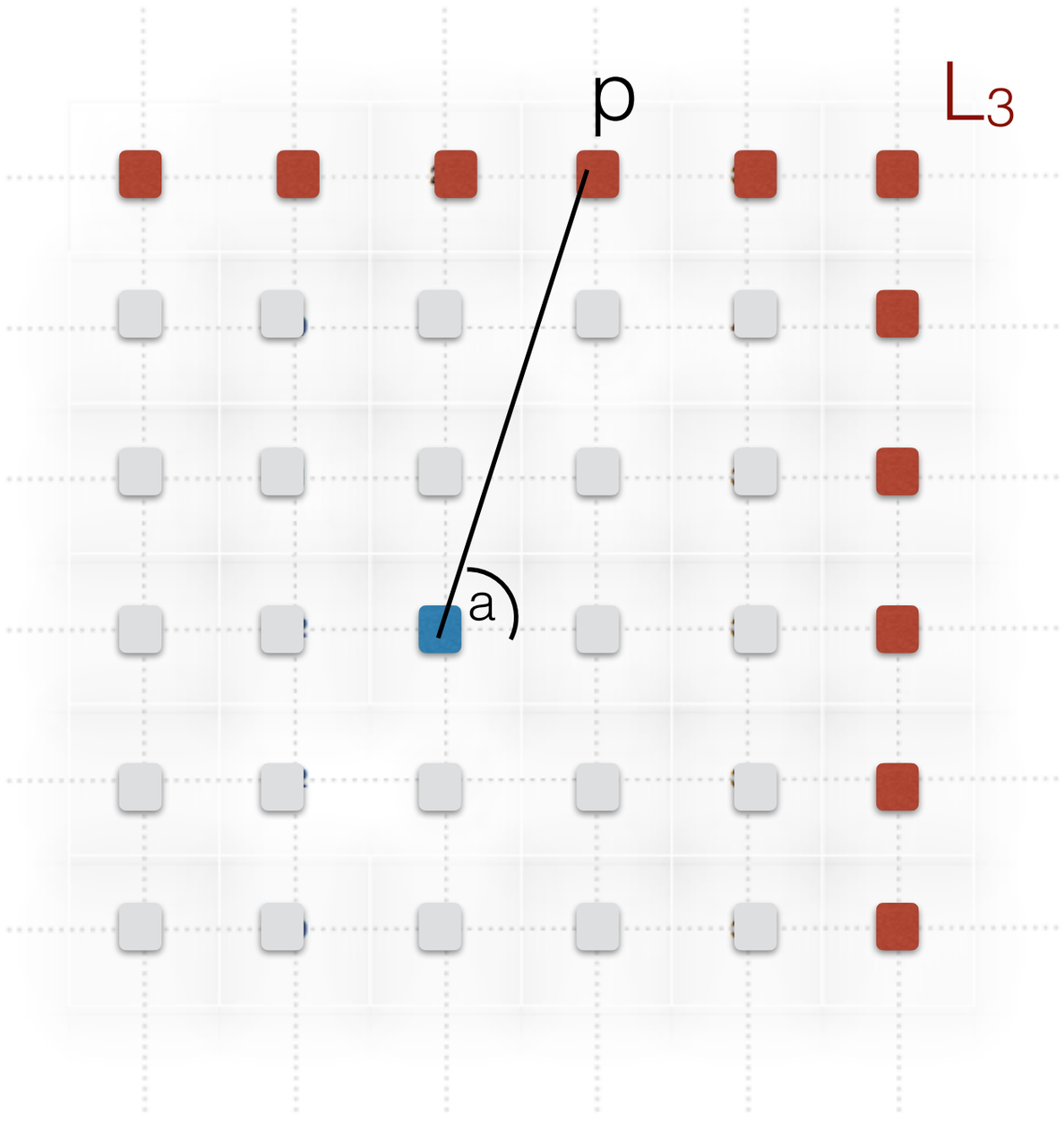


Viewshed on grids using a concentric sweep



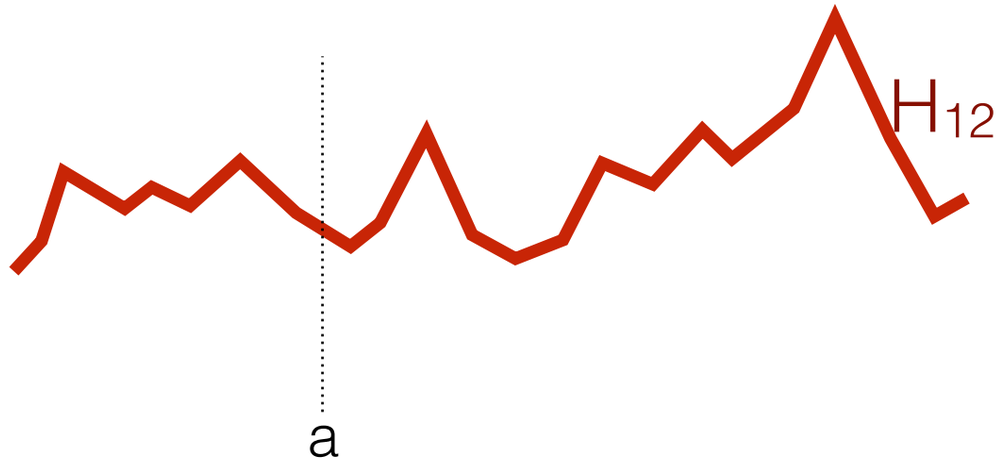
Is point p in L3 visible ?

Viewshed on grids using a concentric sweep



Is point p in L3 visible ?

p is visible if
slope(vp) < H_{12}(a)



Computing viewsheds

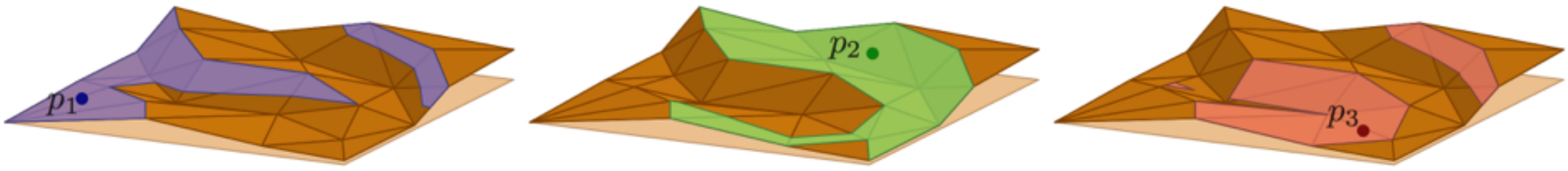
n = nb. of cells in the grid

- Straightforward algorithm: O(n sqrt n) with linear interpolation
- O(n lg n) with nearest neighbor (NN) interpolation
  - uses radial sweep + augmented RB tree
  - crucially exploits NN
  - loses some accuracy

- A different approach: concentric sweep using horizons
  - can be adapted for linear interpolation or nearest neighbor
  - can be used for triangulated terrains
  - fast in practice because horizons stay very small

Viewsheds on triangulated terrains

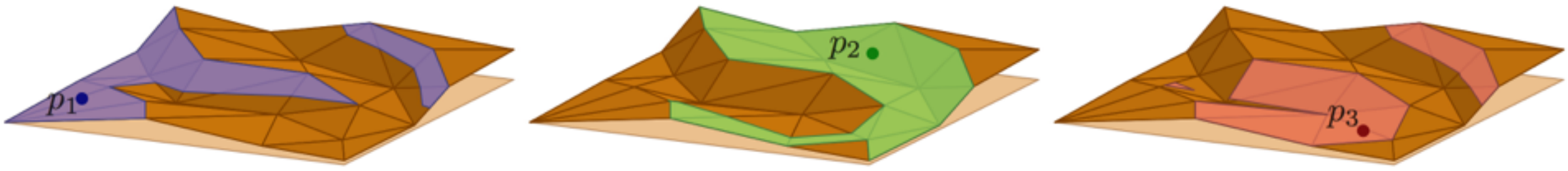
- viewshed(p) contains all points of the terrain that are visible from p



from: http://arxiv.org/pdf/1309.4323.pdf

Viewsheds on triangulated terrains

- viewshed(p) contains all points of the terrain that are visible from p



from: http://arxiv.org/pdf/1309.4323.pdf

- viewshed(p) may intersect a triangle
- viewshed(p) may intersect a triangle multiple times
- Question: what is the complexity of viewshed(p) in the worst case?
- Theorem: The complexity (number of vertices) of a viewshed on a triangulated terrain is $O(n^2)$.

from: HH, MdB, KT 2009

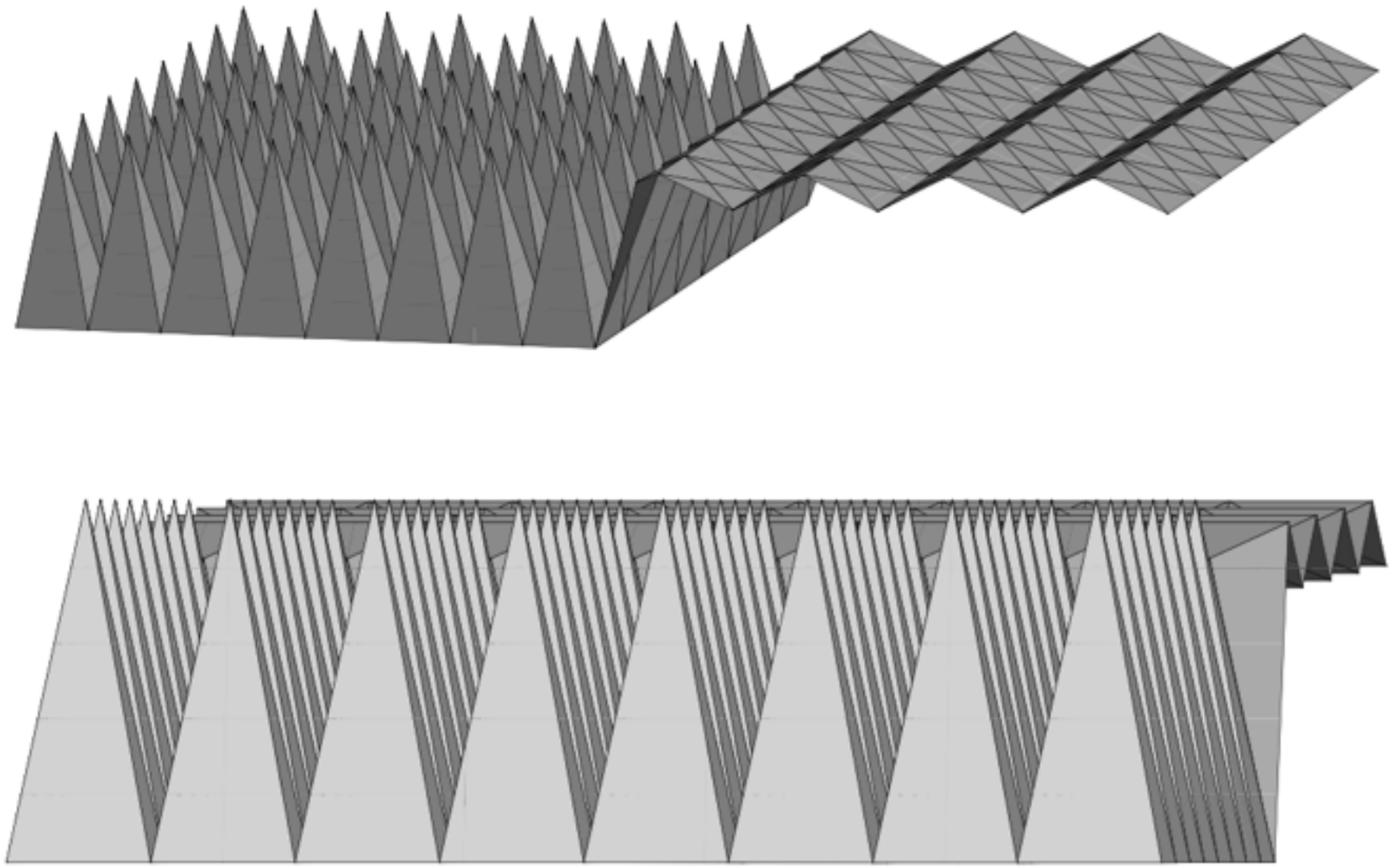


Figure 1: Two views of the same terrain defined by a regular grid. The second view gives a visibility map of complexity $\Theta(n\sqrt{n})$. Note that the terrain can be flattened further without changing the view combinatorially.

Computing viewsheds on triangulated terrains

- Based on horizon computation
  - similar to grids
  - idea: traverse triangles in order of increasing distance form viewpoint, and update horizon.
  - bootstrap with divide and conquer

Summary

- Visibility is a fundamental problem
- Straightforward solution is fairly intuitive/simple/fast
  - works well even for very large terrains
- Refinement of straightforward solution exposes elegant ideas
  - radial sweep
  - augmented RB-trees
  - visibility via horizons
    - starting point for all improved solutions
- Accuracy
  - Interpolation is important

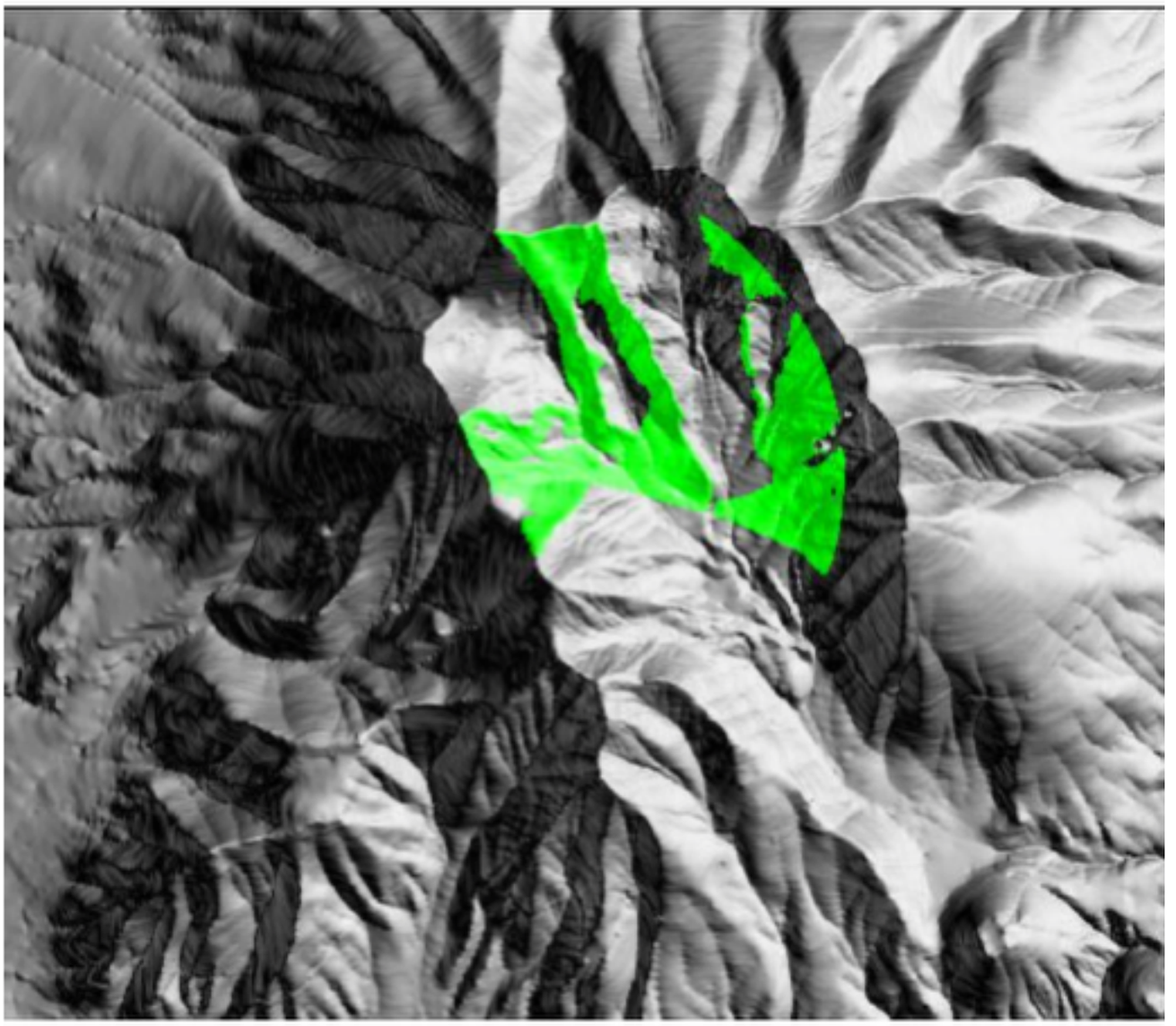
Visibility on terrains

- Viewsheds are starting point for many other problems
  - Given a set of viewpoints, compute their joint visibility
    - aka multiple-source viewshed
  - Find locations of watch towers so that together they can guard the terrain
  - Find point of maximum/minimum visibility
  - Find viewshed count (VC) grid
    - VC(i,j) = nb. visible points in viewshed(i,j)
  - …

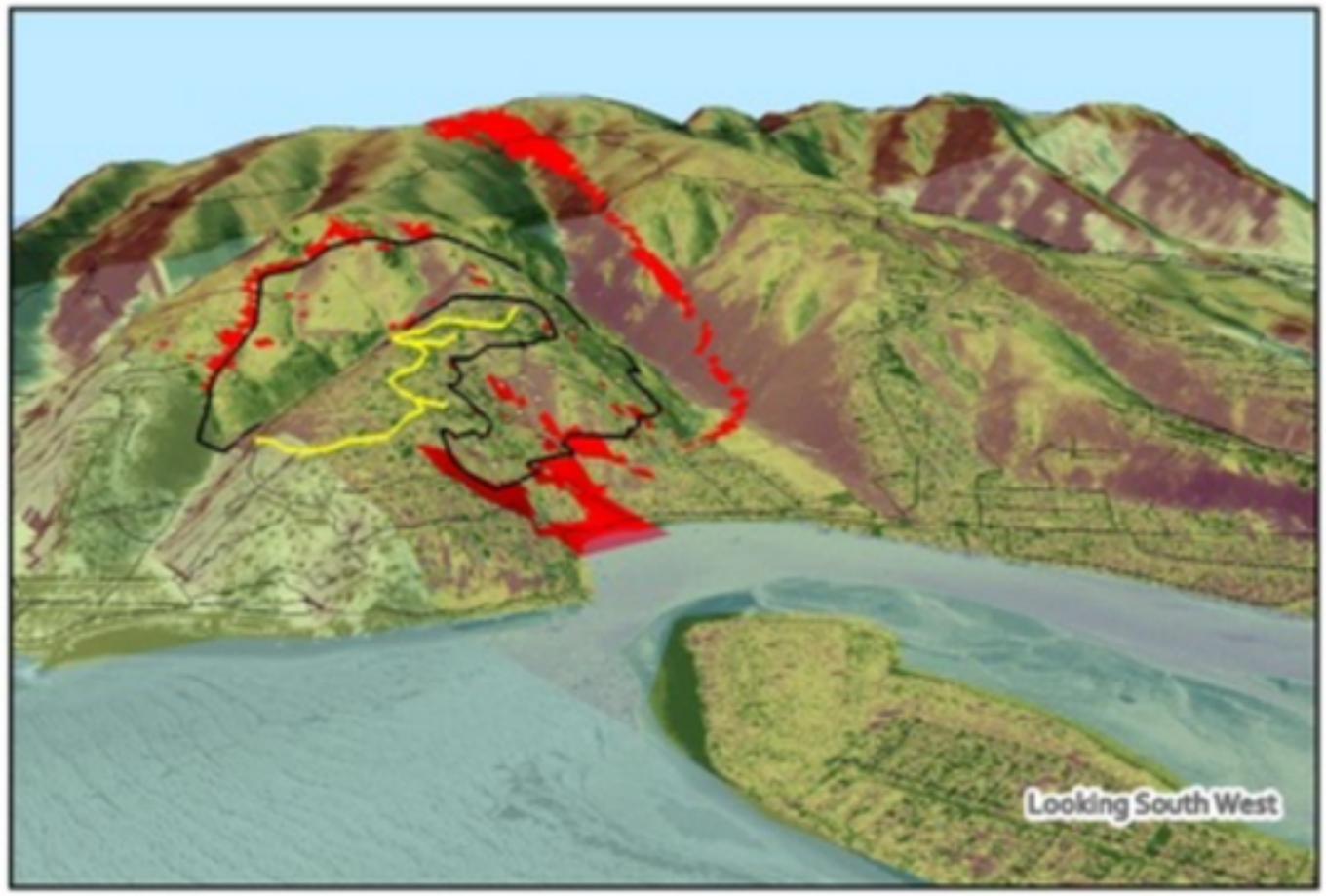
Viewshed count

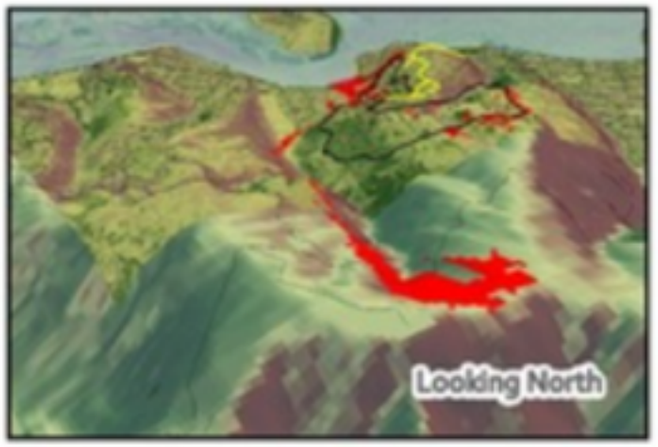
- Input: elevation grid G
- Output: VC grid
    - VC(i,j) = size of viewshed(i,j)

- Sketch an algorithm to compute VC and its running time



Visual Impact of Proposed Subdivision Port Hills, Christchurch





The above images present the proposed new subdivision (outlined black) on the Port Hills. The red area shows potential viewshed loss to residence on Hurst Seager Lane, Panorama Road, Revelation Drive and Starwood Lane (hightlighted yellow) if the proposed subdivision was to go ahead. The impact is based on potential bulidings being up to 10m in height.

GEOG205 Lab 7 - Viewshed Analysis and Visualisation - 28996639 - Bianca de Jong - Lab Stream 4 - 18 September 2012

Areas Visible From Welsh Mountain, Highest Point in Chester County, PA