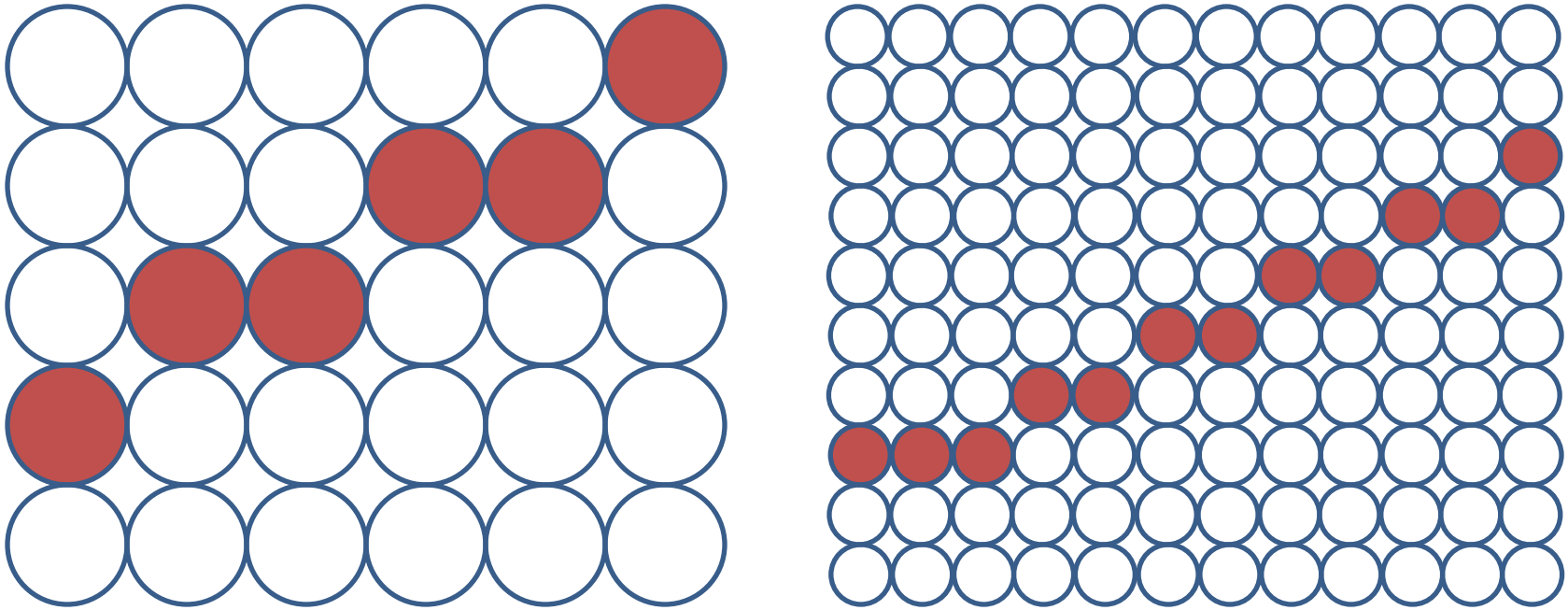
Antialiasing

Same questions:

Why? How?

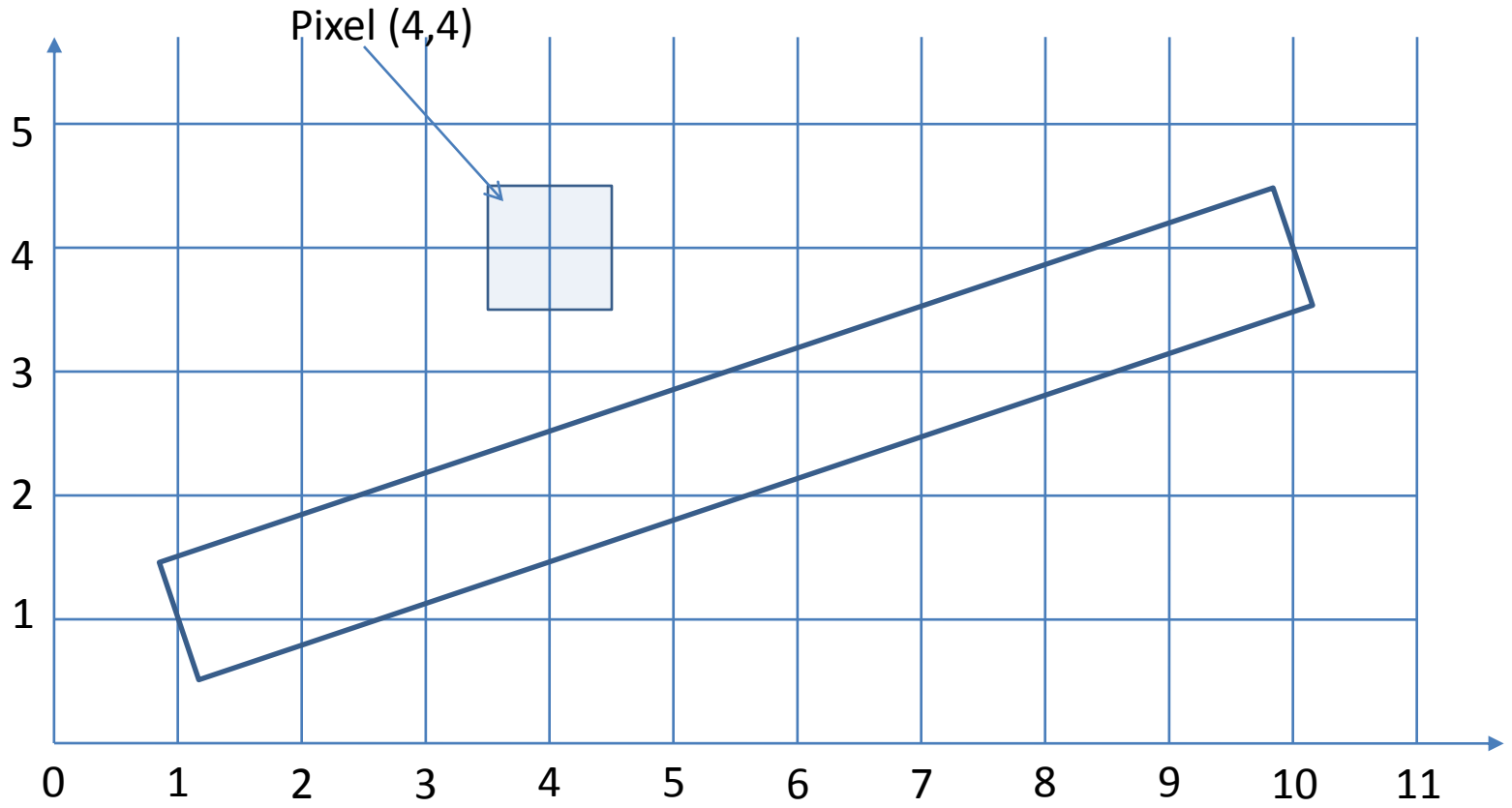
Drawing Line Creates Jaggies



Jaggies and resolutions
(higher resolution, more jaggies but smaller ones)

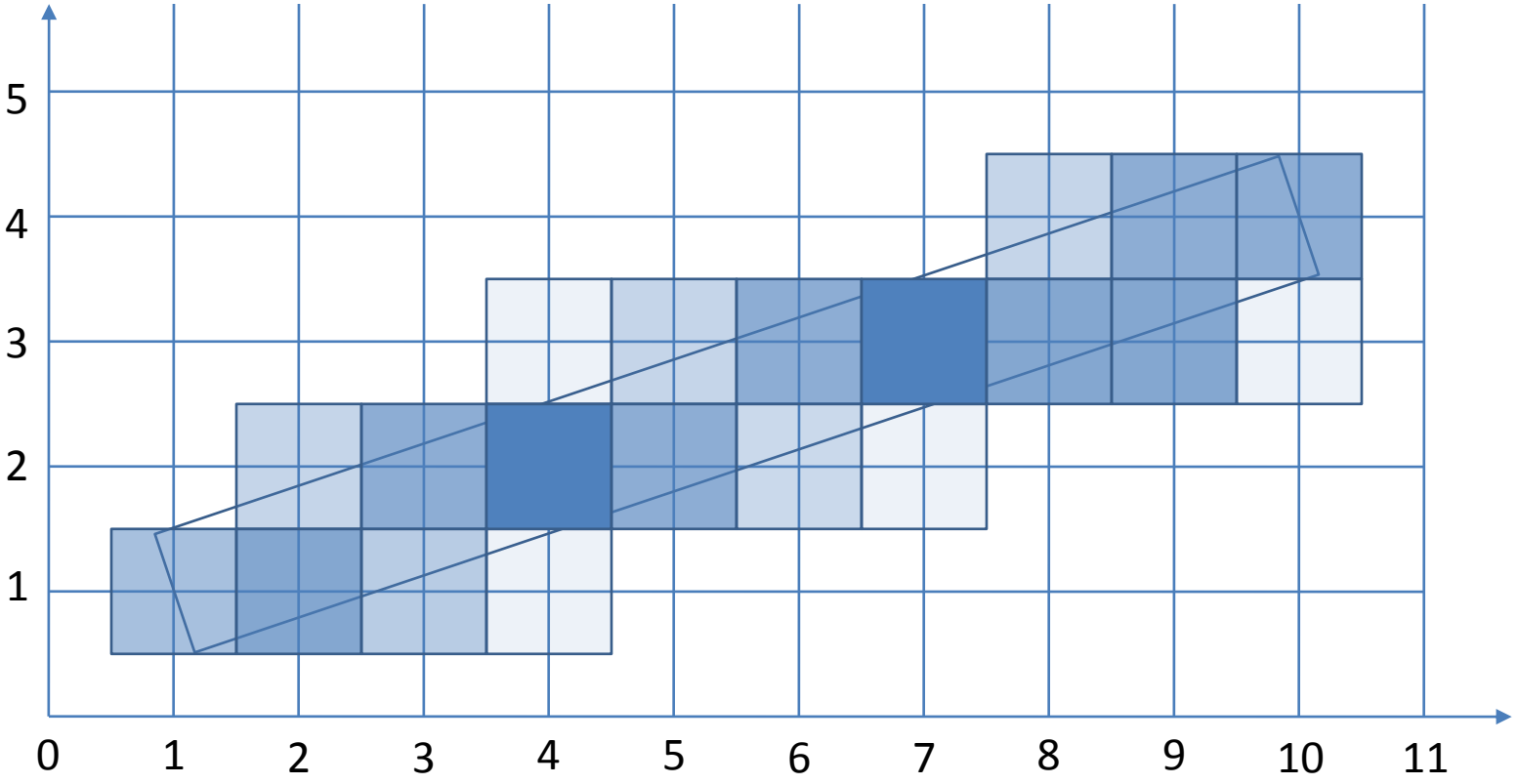
Assumptions

- Line has nonzero width
- Line contribute to the intensity of a pixel (not every pixel should be 100% black)
- Each pixel covers the square of the grid size whose center is the pixel



Unweighted Area Sampling

- The intensity of each pixel is determined proportional to its covered area (1,1) – 50%, (2,1) – 70%, …. (how to count? Exercise!)
- Properties: a) intensity decreases if distance to the line increases; b) no intersection => no influence; c) equal areas contribute equal intensity

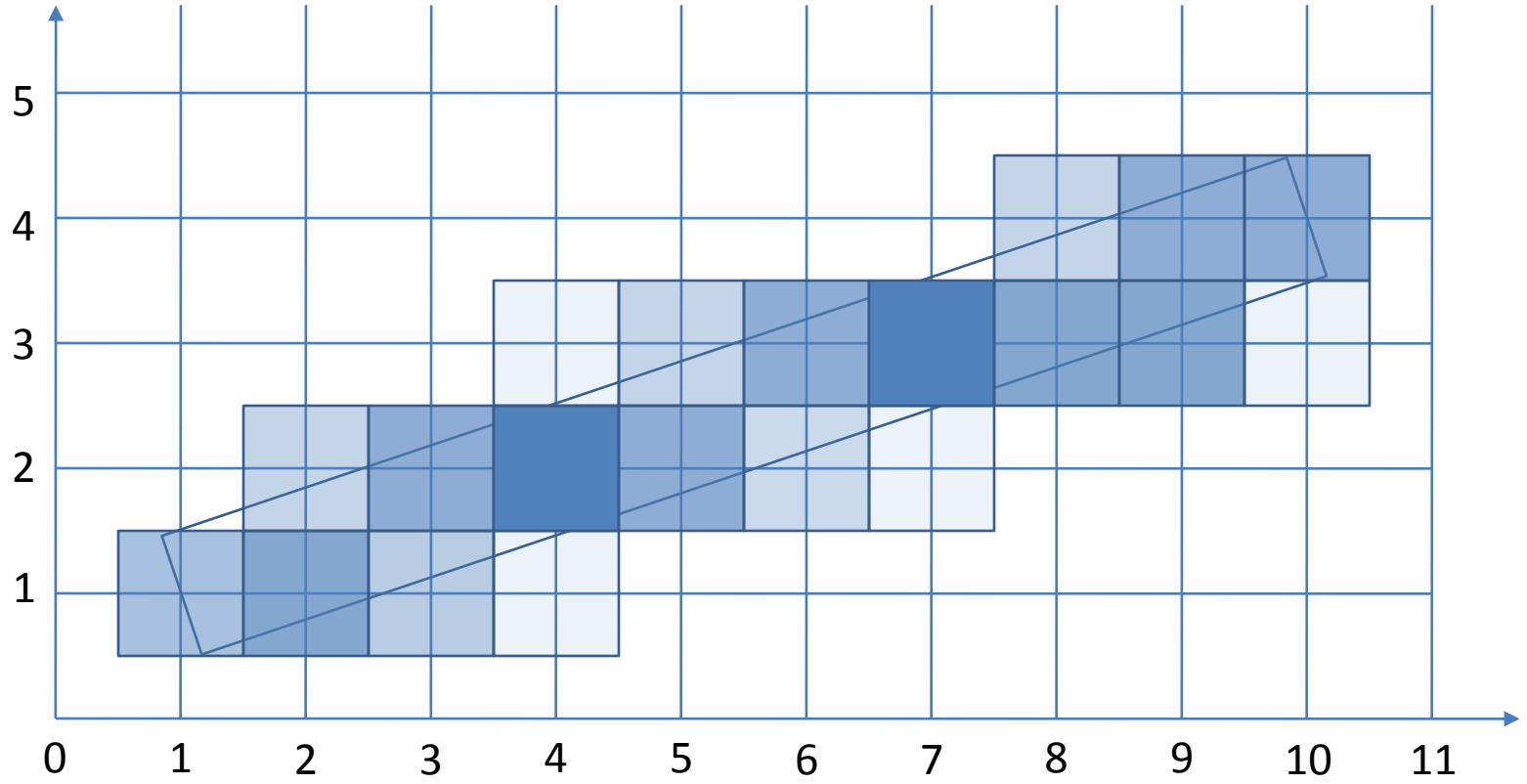


Intensity proportional to area covered

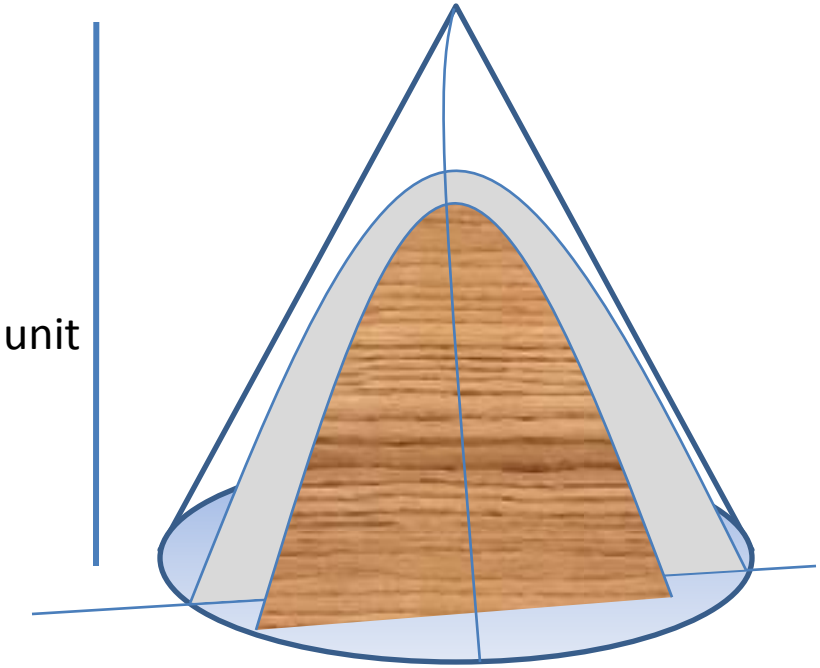
Weighted Area Sampling

- Maintains: a) intensity decreases if distance to the line increases; b) no intersection => no influence;
- changes: c) equal areas contribute equal intensity - Use a weighting function (filter function) to accomplish this (less intensity for area father from the center of the line) and new shape for pixel (circular area, larger than the square tile)

Weighting function for weighted area sampling: circular cone (cone filter!)



Cone weighting function



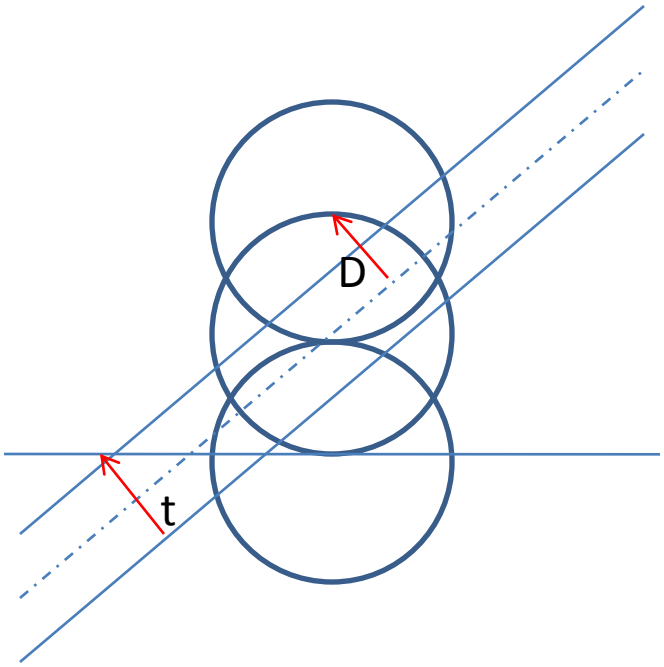
Same area – different volume (subvolume)

➔use to define intensity

Maintianing two properties: a) intensity decreases if distance to the line increases;
b) No intersection => no influence;
but altering c) equal areas contribute equal intensity

Gupta-Sproull Antialiased Lines

- Use a pixel with radius equal to a grid unit
  - ➔Line of unit thickness with slope less than 1 typically intersects three supports in a column, a minimally two, and maximally five
  - ➔Compute a table (of weight) prior to drawing and use table look up to decide the intensity based on a function depends on (D,t)
  - ➔The function is 4 bit display (D is from 1 to 16) between 0 and 1.5 and t = 1



D – distance to the pixel
t – thickness of the line

Modified Midpoint Algorithm

Main idea:

Determining D incrementally

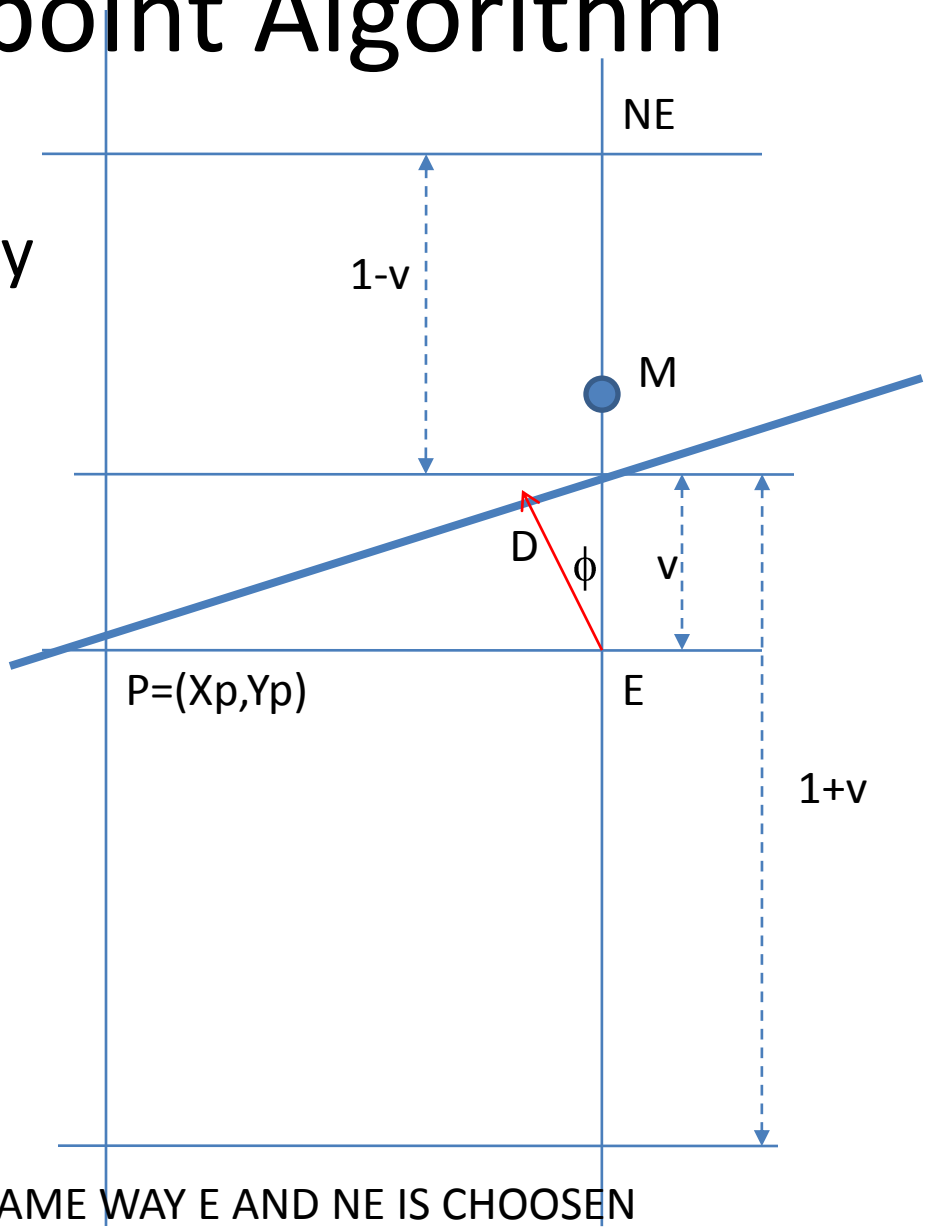
$$D = v\cos\theta = \frac{vdx}{\sqrt{dx^2 + dy^2}}$$

dx - difference in x coordinates

dy - difference in x coordinates

It can be shown:

$vdx = \boldsymbol{F}(Xp+1, yp)/2$



IDEA: COMPUTE *D* INCREMENTALLY IN THE SAME WAY E AND NE IS CHOOSEN

$F(x,y) = 2(ax+by+c)$ (Line equation)

$E: \quad xe = xp+1, ye = yp, v = y - yp$

$So \quad v = ((a(xp+1)+c)/-b) - yp$

$and \quad -bv = a(xp+1)+byp+c = F(xp+1, yp)/2$

$Let \quad d = F(M)$

$For\ E \quad 2vdx = d + dx$

$Thus \quad D = \dfrac{d+dx}{2\sqrt{x^2+y^2}}$

$For\ SE \quad 2vdx = d - dx$

This computation is then used in
the midpoint algorithm (Detail in Book)