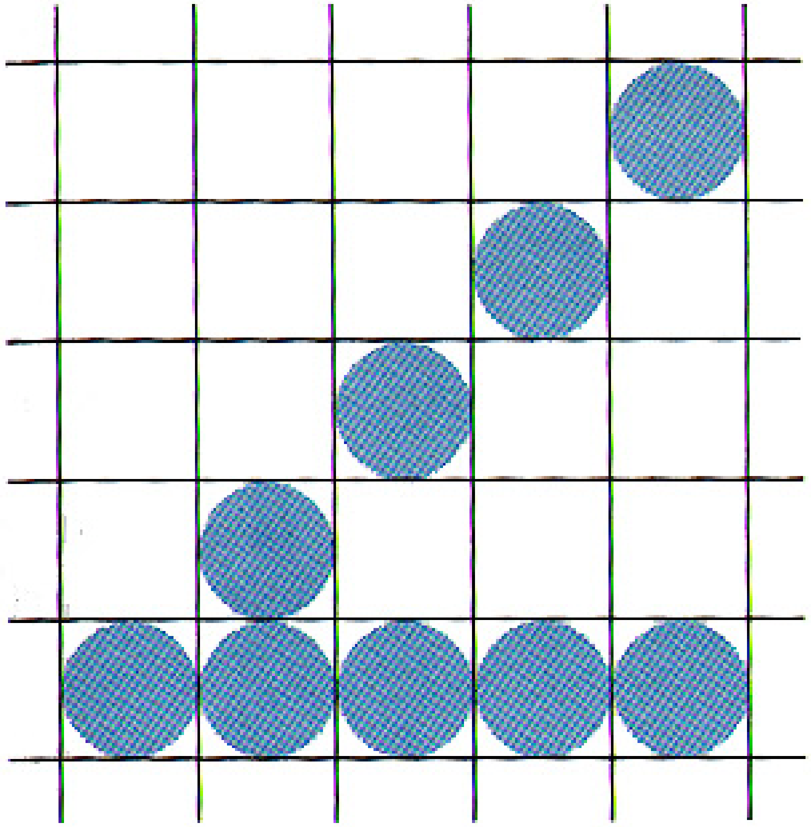


Unequal-length lines displayed with the same number of pixels in each line.

Scan Conversion of Circles

- The standard equation for a circle is

$$(x - x_c)^2 + (y - y_c)^2 = r^2$$

- We could calculate the y value of this equation as x steps from $x_c - r$ to $x_c + r$.

- y is then given by

$$y = y_c \pm \sqrt{r^2 - (x - x_c)^2}$$

- This is costly since it involves floating point multiplication and taking square roots. It has the potential drawback that points it generates are not spaced evenly: the are widely spaced when the gradient is > 1.

- If the slope < 1 we should step through x otherwise step through y.

Scan Conversion of Circles (cont.)

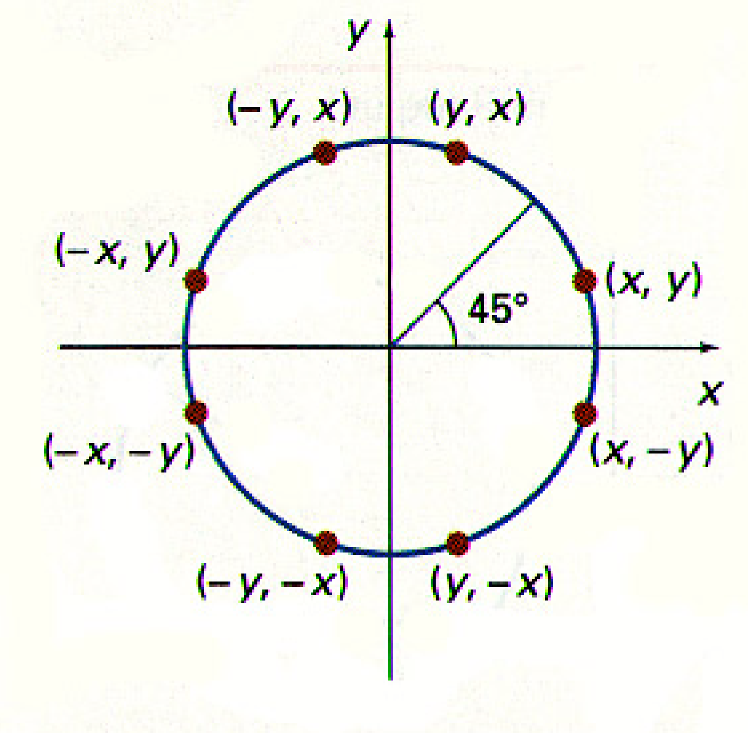
- In generating circles the work can be reduced by taking advantage of symmetry.

- To avoid uneven point spacing we could use the parametric polar form

  $$x = x_c + r\cos\theta \quad y = y_c + r\sin\theta$$

  θ could be set to *1/r*.

- The DDA and Bresenham's algorithm for line drawing can also be generalized for circles.



Symmetry of a circle. Calculation of a circle point (x, y) in one octant yields the circle points shown for the other seven octants.

Scan Conversion of Other Curves

- The standard equation for an ellipse is

$$\left(\frac{x - x_c}{r_1}\right)^2 + \left(\frac{y - y_c}{r_2}\right)^2 = 1$$

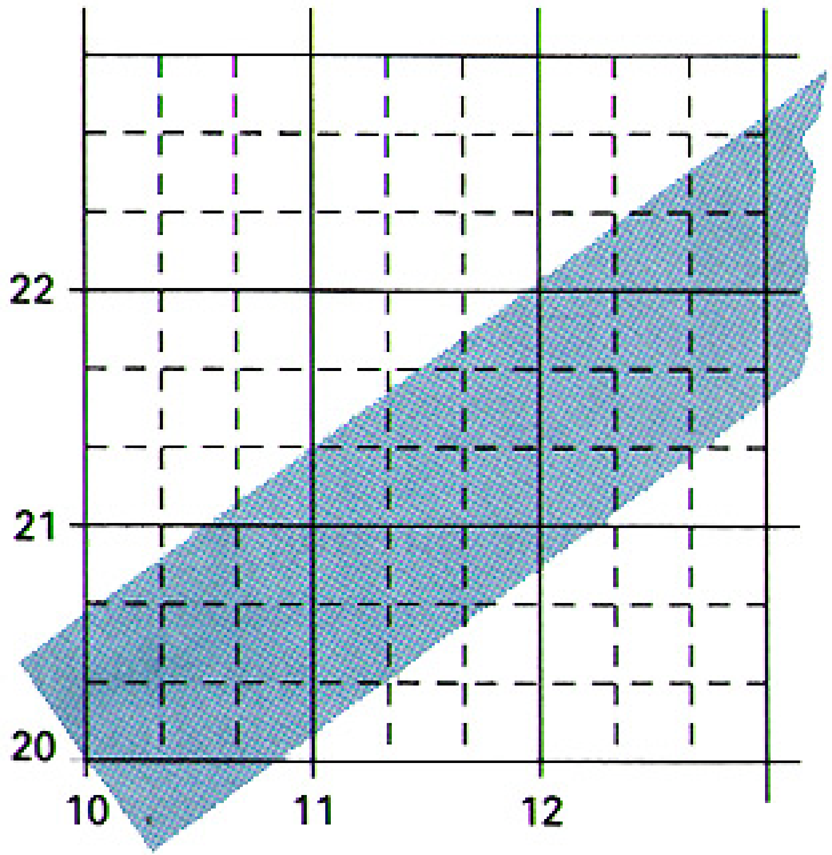
  Bresenham's algorithm for plotting points on a circle can be generalized for ellipses.

- Other curves:
  Given $y = f(x)$, we can step through x.

- In regions where $f'(x) > 1$ we will plot too few points so it would be preferable to vary y and compute x from y via $x = f^{-1}(y)$

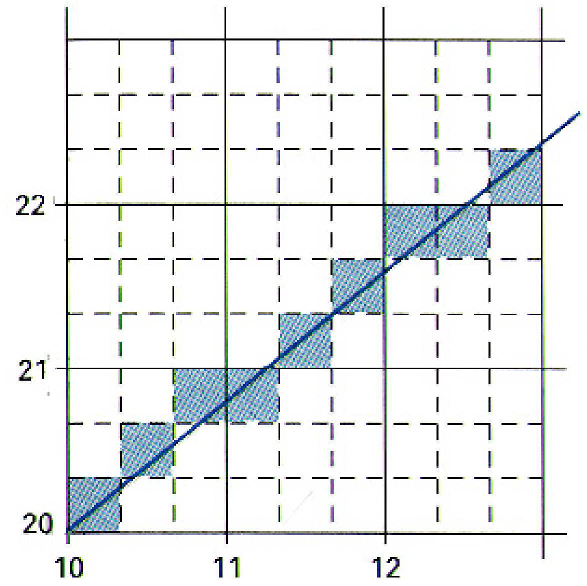
Antialiasing Lines

- **Staircasing** or **aliasing** results because coordinates for points need to be rounded to the nearest integer pixel position.
- Display devices that can display different intensity levels allow us to adjust the pixel intensity according to the amount of the pixel that is covering the line.



Supersampling subpixel positions in relation to the interior of a line of finite width.

- We can also subdivide a pixel and **supersample** the sub-pixels to arrive at a better approximation of the intensity of the pixel.



Supersampling subpixel positions along a straight line segment whose left endpoint is at screen coordinates (10, 20).

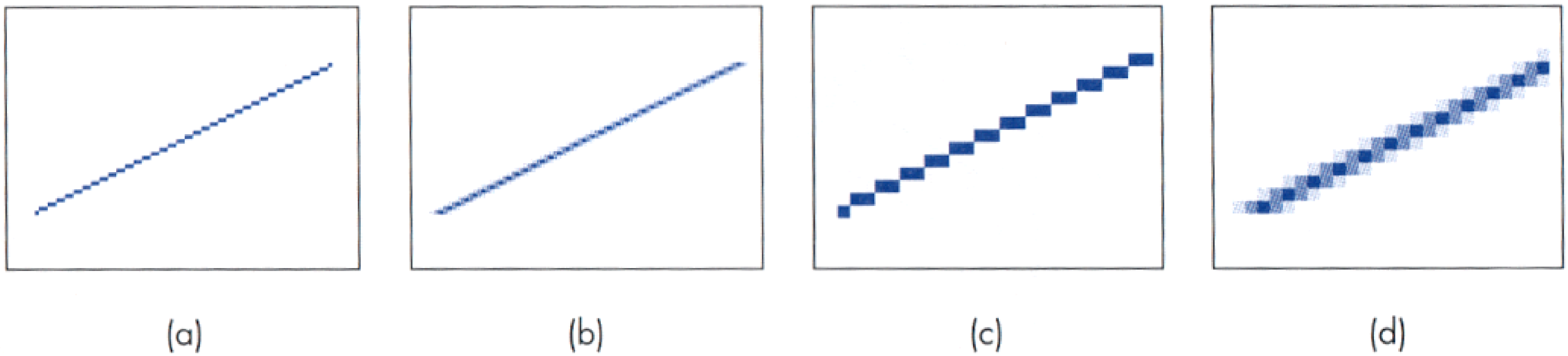
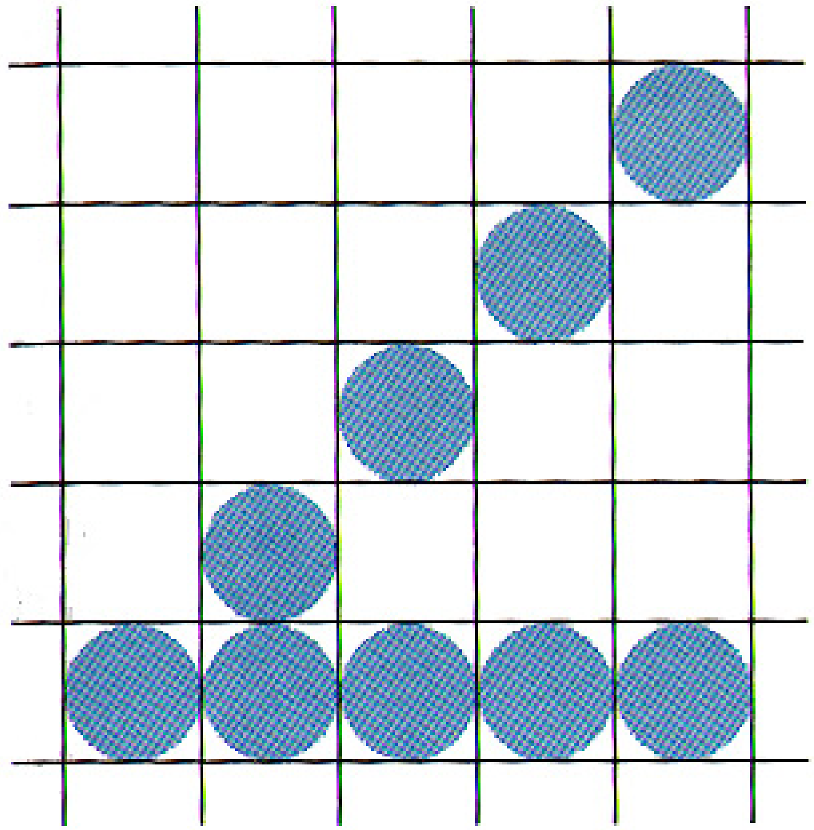


Figure 8.60 Aliased versus antialiased line segments. (a) Aliased line segment. (b) Antialiased line segment. (c) Magnified aliased line segment. (d) Magnified antialiased line segment.

Intensity Compensation

- Similarly we can adjust the intensity of pixels dependent on the gradient (diagonal lines with the same number of pixels as horizontal or vertical lines are longer by a factor which can be as large as $\sqrt{2}$).



Unequal-length lines displayed with the same number of pixels in each line.

Polygon Scan Conversion – Fill Areas

- Polygon scan algorithms tend to process a picture from left to right and top to bottom.
- We need to be able to detect when a scan line intersects with a polygon.
- At each intersection we are either entering or leaving the polygon.
- Each pair of intersections determines a line segment of pixels which are inside the polygon.

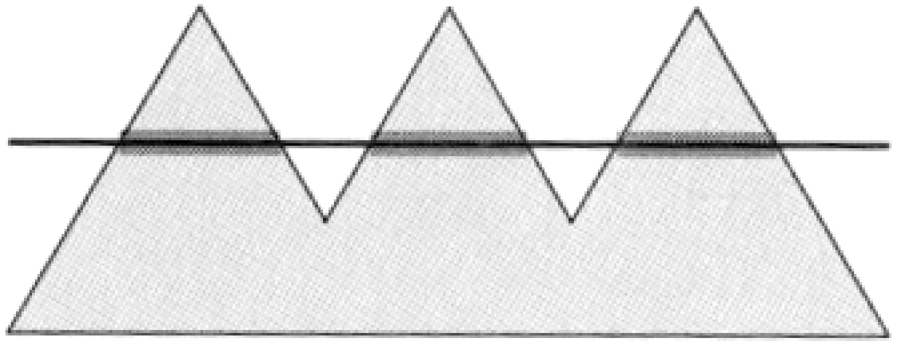


Figure 8.54 Polygon with spans.

- Special difficulties may arise when a vertex intersects with a scan line (called a **singularity**).
- An isolated vertex is the intersection of two edges whose gradients have different signs and can be treated as a pair of intersections with a scan line.
- If the vertex is at the junction of two edges whose gradients have the same sign, then the intersection should only be counted singly.
- Some algorithms perturb singularities so they never occur on a scan-line. This approach sacrifices accuracy for the sake of efficiency.

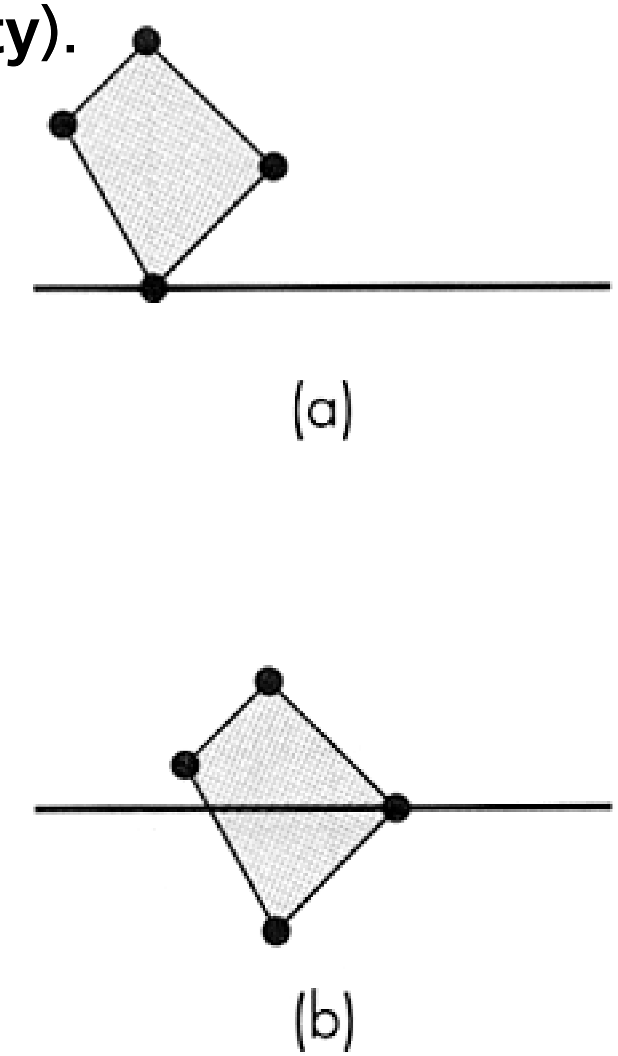


Figure 8.58 Singularities. (a) Zero or two edge crossings. (b) One edge crossing.

Polygon Scan-conversion (cont.)

- Another method to remove singularities is to use a virtual frame buffer, twice the size of the actual frame buffer.
- In the virtual frame buffer, pixels are located only at even values of y and vertices are only located at odd values of y.
- Open GL uses an approach similar to this, placing pixel centres halfway between integers.
- Polygon scan algorithms can take advantage of **coherence** properties of objects to be processed. That is, from scan-line to scan-line we expect only a small number of changes.

Polygon Scan Conversion (cont.)

- For efficient processing, we could keep a list of active edges – edges that cross the current scan line.
- Since the gradient of a line is constant we can predict where the edge will cross the next scan line.
- In addition, all edges can be sorted in order from largest y-value of endpoint to smallest and edges can be discarded once they no longer appear in the picture.

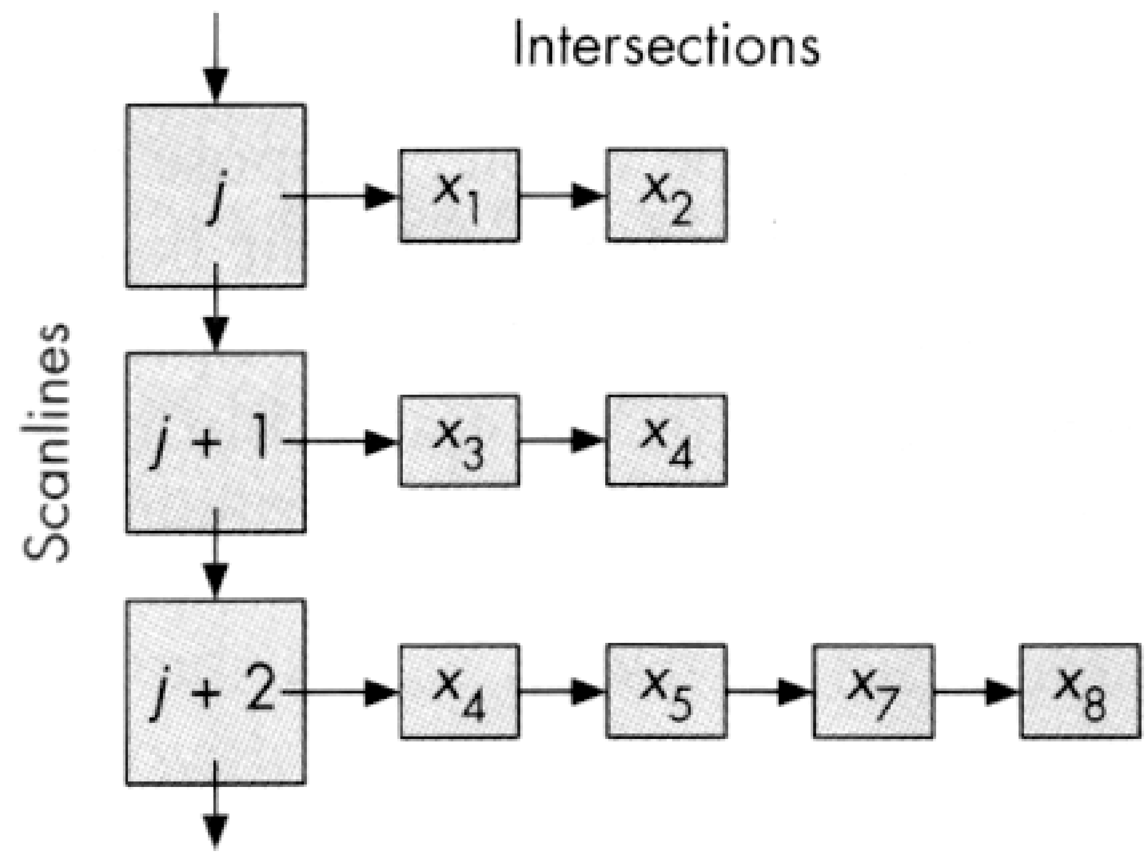


Figure 8.57 Data structure for y–x algorithm.