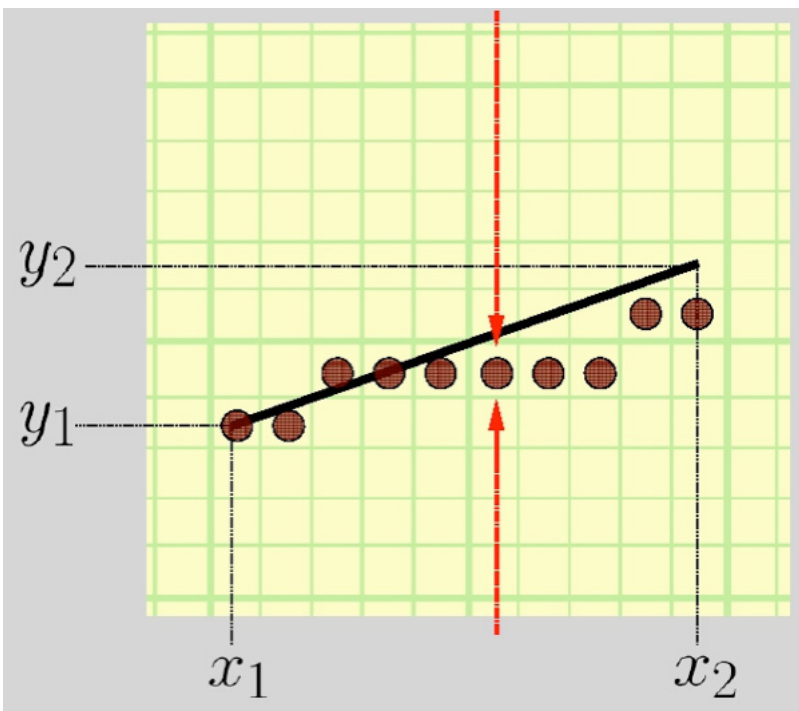
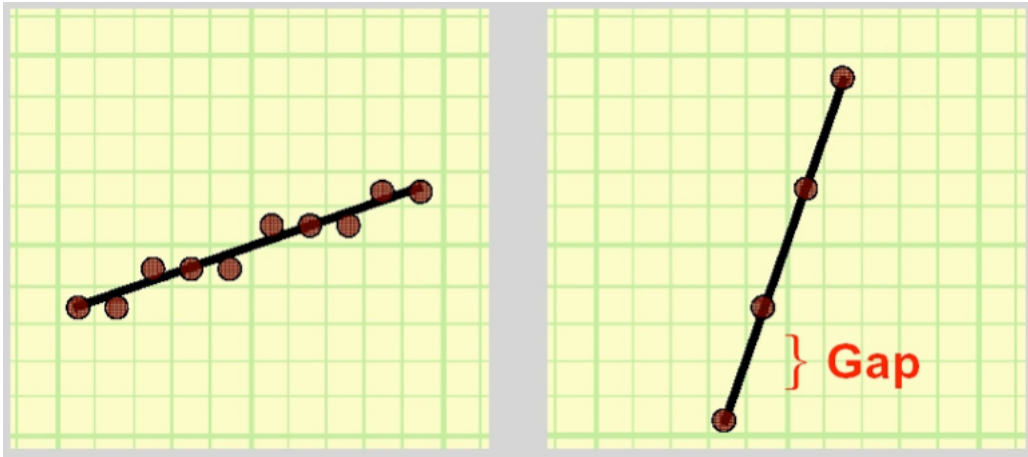


$\Delta x = 1$
 $\Delta y = m \cdot \Delta x$
 After rounding



$\Delta x = 1$
 $\Delta y = m \cdot \Delta x$
 $y += \Delta y$
 Accumulates Error



$|m| > 1$

$|m| \leq 1$

cases:

start at left $x_0 < x_e$

start at right $x_0 > x_e$

m positive > 1

m positive < 1

m negative $\text{abs}(m) > 1$

m negative $\text{abs}(m) < 1$

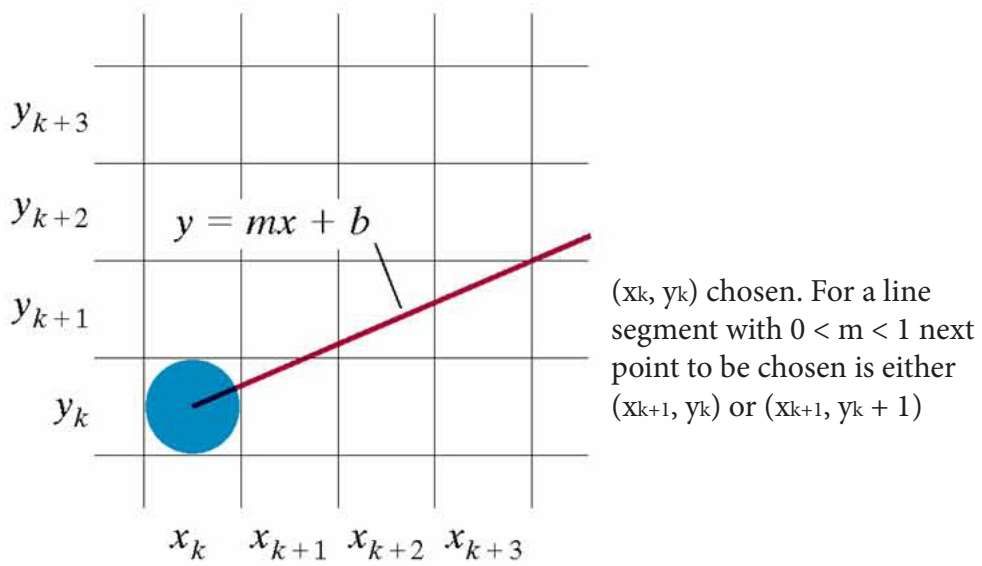
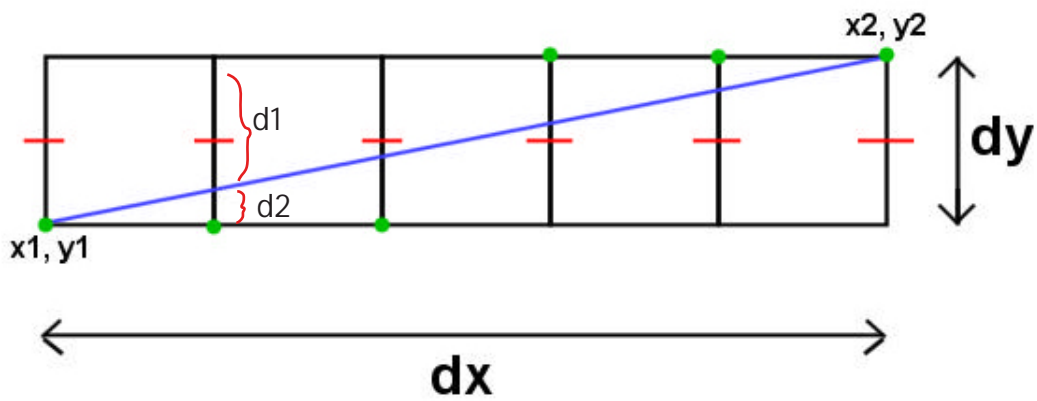
```

public void lineDDA(int x0, int y0, int x1, int y1, Color
color)
{
    int pix = color.getRGB();
    int dy = y1 - y0;
    int dx = x1 - x0;
    float t = (float) 0.5;        // offset for rounding

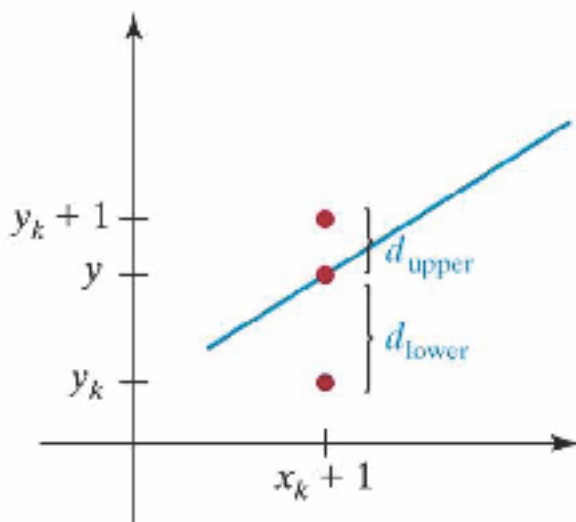
    raster.setPixel(pix, x0, y0);
    if (Math.abs(dx) > Math.abs(dy)) { // slope < 1
        float m = (float) dy / (float) dx;
        // compute slope
        t += y0;
        dx = (dx < 0) ? -1 : 1;
        m *= dx;

        while (x0 != x1) {
            x0 += dx;        // step to next x value
            t += m;          // add slope to y value
            raster.setPixel(pix, x0, (int) t);
        }
    } else { // slope >= 1
        float m = (float) dx / (float) dy;
        // compute slope
        t += x0;
        dy = (dy < 0) ? -1 : 1;
        m *= dy;
        while (y0 != y1) {
            y0 += dy;        // step to next y value
            t += m;          // add slope to x value
            raster.setPixel(pix, (int) t, y0);
        }
    }
}
}

```



Decision Criterion



```

public void lineBresenham(int x0, int y0, int x1, int y1, Color color)
{
    int pix = color.getRGB();
    int dy = y1 - y0;
    int dx = x1 - x0;
    int stepx, stepy;

    if (dy < 0) { dy = -dy; stepy = -1; } else { stepy = 1; }
    if (dx < 0) { dx = -dx; stepx = -1; } else { stepx = 1; }
    dy <<= 1; // dy is now 2*dy
    dx <<= 1; // dx is now 2*dx

    raster.setPixel(pix, x0, y0);
    if (dx > dy) {
        int fraction = dy - (dx >> 1); // same as 2*dy - dx
        while (x0 != x1) {
            if (fraction >= 0) {
                y0 += stepy;
                fraction -= dx; // same as fraction -= 2*dx
            }
            x0 += stepx;
            fraction += dy; // same as fraction -= 2*dy
            raster.setPixel(pix, x0, y0);
        }
    } else {
        int fraction = dx - (dy >> 1);
        while (y0 != y1) {
            if (fraction >= 0) {
                x0 += stepx;
                fraction -= dy;
            }
            y0 += stepy;
            fraction += dx;
            raster.setPixel(pix, x0, y0);
        }
    }
}
}

```