Simple Line Drawing Algorithms

Taylor Foxhall

February 18, 2016

1 Introduction

In the field of computer graphics, there are two well-known algorithms used for drawing lines. We will compare Brensenham's algorithm, the midpoint line algorithm, and OpenGL's implementation of line drawing.

2 Brensenham Line Drawing

2.1 Idea

Brensenham's algorithm chooses which pixel to draw based on an accumulated error. Assuming we're trying to draw a line that is in the first octant of a graph, we can draw horizontally unless our error crosses a certain threshold, then we increment y and subtract the accumulated error. Other lines in other octants can be handled by a simple transformation (not shown below, but is included in the source file.)

2.2 Code Exposé

```
void brensenham_line(int x0, int y0, int x1, int y1) {
  int dx = x1 - x0;
  int dy = y1 - y0;
  int y = y0;
  int decider = 2*dy - dx;

  glBegin(GL_POINTS);
  for (int x = x0; x <= x1; x++) {
    glVertex2i(x, y);
    decider += 2*dy;
    if (decider > 0) {
      y++;
      decider -= 2*dx;
    }
  }
  glEnd();
}
```

2.3 Demos

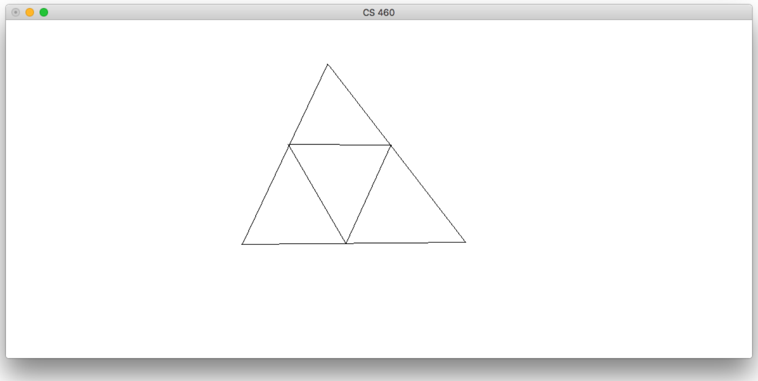

Figure 1: A simple drawing using Brensenham's algorithm

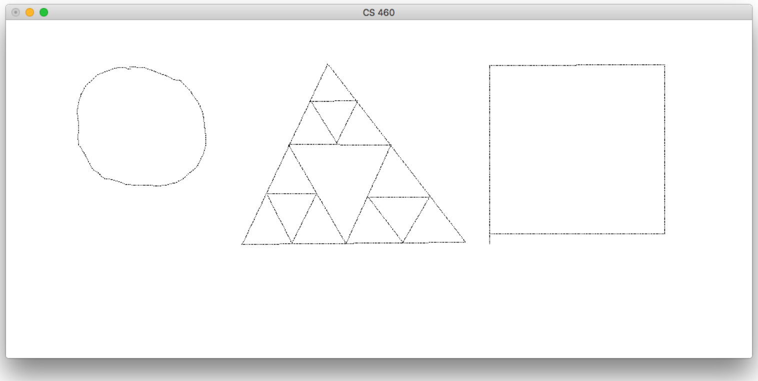

Figure 2: Line stippling with Brensenham's algorithm

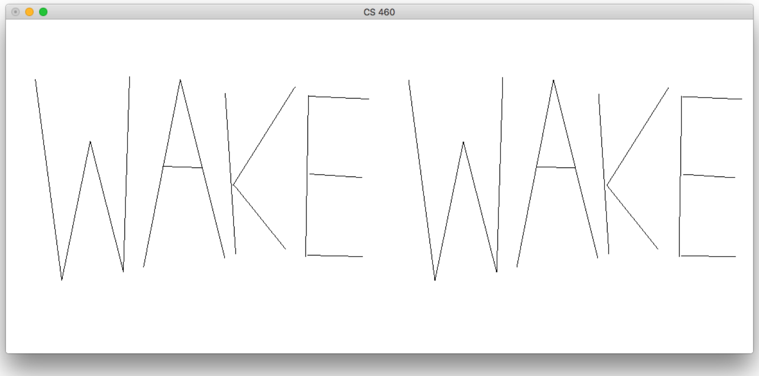

Figure 3: "WAKE" drawn with Brensenham's algorithm (left) vs. OpenGL's lines (right)

3 Midpoint Line Drawing

3.1 Idea

This algorithm tries to improve upon Brensenham by substituting the error accumulator for an approximation of the line function and compares it with its location of the midpoint of the eastern pixel and the northeastern pixel. The algorithm fills in whatever pixel is closer to the approximation. The octant switching in Brensenham's algorithm still applies.

3.2 Code Exposé

```
void midpoint_line(int x0, int y0, int x1, int y1) {
  int dx = x1 - x0;
  int dy = y1 - y0;
  int y = y0;
  int decider = 2*dy - dx;
  int inc_e = 2*dy;
  int inc_ne = 2*(dy - dx);

  glBegin(GL_POINTS);
  for (int x = x0; x <= x1; x++) {
      glVertex2i(x, y);
    if (decider > 0) {
      decider += inc_ne;
      y++;
    }
    else {
      decider += inc_e;
    }
  }
  glEnd();
}
```

3.3 Demos

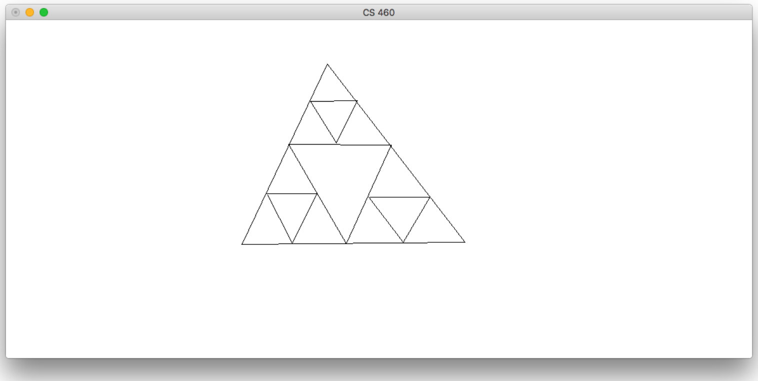

Figure 4: A simple drawing using the Midpoint algorithm

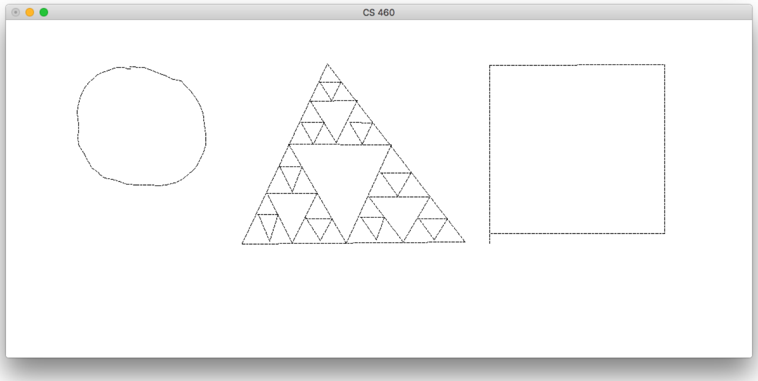

Figure 5: Line stippling with the Midpoint algorithm

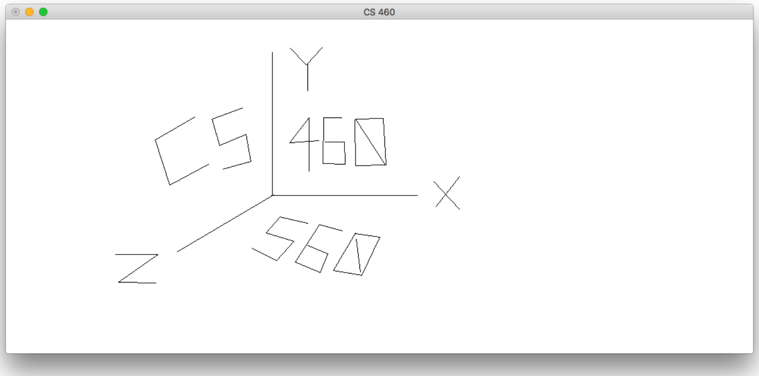

Figure 6: A coordinate system drawn with the Midpoint algorithm