Newsgroups: comp.sources.unix
From: bobp@hal.com (Bob Pendelton)
Subject: v26i048: line3d - 3D Bresenham's (a 3D line drawing algorithm)
Sender: unix-sources-moderator@pa.dec.com
Approved: vixie@pa.dec.com

Submitted-By: bobp@hal.com (Bob Pendelton)
Posting-Number: Volume 26, Issue 48
Archive-Name: line3d

[For the mathematically challenged (meaning me, until I looked it up),
  Bresenham's algorithm is a neato and efficient way to enumerate all of
  (actually "most of") the points between two endpoints. Most such
  algorythms take as an input parameter the resolution wanted -- since
  it would be a waste to calculate at a higher resolution than you use
  (you'd just have a lot of points that would "go in the same place").
  This algorythm uses only integer arithmetic and assumes that you want
  all the contiguous points which can be represented with integers.

  I expect that there's something like this in the X11 Phigs code. But
  since this submission is short and clean and relatively easy to under-
  stand, I think it's useful and am therefore posting it. Note that it
  has no man page or README or Makefile, and in the average case I would
  reject a submission absent of any of those three. If I get a flood of
  small C functions in response to this submission, I'll have to make up
  some rules about what they have to include. Heck, perhaps I'll start
  a library and post updates to it. (That's not a promise!)

  --vix]

This is something I've hoped to see on the net for years. And I can't
guess how many times I've seen requests for something like this. It
turned out to be trivial to build, once I spent a couple of weeks of
my spare time researching it... (I don't have a lot of spare time. :-)
So here it is, a simple, fast, 3D version of Bresenham's line drawing
algorithm.

```
------------------------------------------------------------------------
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#               "End of shell archive."
# Contents:  dl.c
# Wrapped by bobp@oscar on Tue Feb 25 14:27:48 1992
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f dl.c -a "${1}" != "-c" ; then
  echo shar: Will not over-write existing file \"dl.c\"
else
echo shar: Extracting \"dl.c\" \(2795 characters\)
sed "s/^X//" >dl.c <<'END_OF_dl.c'
X/*
X * line3d was dervied from DigitalLine.c published as "Digital Line Drawing"
X * by Paul Heckbert from "Graphics Gems", Academic Press, 1990
X *
X * 3D modifications by Bob Pendleton. The original source code was in the public
X * domain, the author of the 3D version places his modifications in the
X * public domain as well.
X *
X * line3d uses Bresenham's algorithm to generate the 3 dimensional points on a
```

```
X * line from (x1, y1, z1) to (x2, y2, z2)
X *
X */
X
X/* find maximum of a and b */
X#define MAX(a,b) (((a)>(b))?(a):(b))
X
X/* absolute value of a */
X#define ABS(a) (((a)<0) ? -(a) : (a))
X
X/* take sign of a, either -1, 0, or 1 */
X#define ZSGN(a) (((a)<0) ? -1 : (a)>0 ? 1 : 0)
X
Xpoint3d(x, y, z)
X    int x, y, z;
X{
X
X    /* output the point as you see fit */
X
X}
X
Xline3d(x1, y1, x2, y2, z1, z2)
X    int x1, y1, x2, y2, z1, z2;
X{
X    int xd, yd, zd;
X    int x, y, z;
X    int ax, ay, az;
X    int sx, sy, sz;
X    int dx, dy, dz;
X
X    dx = x2 - x1;
X    dy = y2 - y1;
X    dz = z2 - z1;
X
X    ax = ABS(dx) << 1;
X    ay = ABS(dy) << 1;
X    az = ABS(dz) << 1;
X
X    sx = ZSGN(dx);
X    sy = ZSGN(dy);
X    sz = ZSGN(dz);
X
X    x = x1;
X    y = y1;
X    z = z1;
X
X    if (ax >= MAX(ay, az))            /* x dominant */
X    {
X        yd = ay - (ax >> 1);
X        zd = az - (ax >> 1);
X        for (;;)
X        {
X            point3d(x, y, z);
X            if (x == x2)
X            {
X                return;
X            }
X
X            if (yd >= 0)
X            {
X                y += sy;
X                yd -= ax;
```

```
X           }
X
X           if (zd >= 0)
X           {
X               z += sz;
X               zd -= ax;
X           }
X
X           x += sx;
X           yd += ay;
X           zd += az;
X       }
X   }
X   else if (ay >= MAX(ax, az))           /* y dominant */
X   {
X       xd = ax - (ay >> 1);
X       zd = az - (ay >> 1);
X       for (;;)
X       {
X           point3d(x, y, z);
X           if (y == y2)
X           {
X               return;
X           }
X
X           if (xd >= 0)
X           {
X               x += sx;
X               xd -= ay;
X           }
X
X           if (zd >= 0)
X           {
X               z += sz;
X               zd -= ay;
X           }
X
X           y += sy;
X           xd += ax;
X           zd += az;
X       }
X   }
X   else if (az >= MAX(ax, ay))           /* z dominant */
X   {
X       xd = ax - (az >> 1);
X       yd = ay - (az >> 1);
X       for (;;)
X       {
X           point3d(x, y, z);
X           if (z == z2)
X           {
X               return;
X           }
X
X           if (xd >= 0)
X           {
X               x += sx;
X               xd -= az;
X           }
X
X           if (yd >= 0)
X           {
```

```
X                y += sy;
X                yd -= az;
X            }
X
X            z += sz;
X            xd += ax;
X            yd += ay;
X        }
X    }
X}
END_OF_dl.c
if test 2795 -ne `wc -c <dl.c`; then
    echo shar: \"dl.c\" unpacked with wrong size!
fi
# end of overwriting check
fi
echo shar: End of shell archive.
exit 0
```