

\$SPAD/src/lib xdither.c

The Axiom Team

July 29, 2014

**Abstract**

# Contents

1	License	3
---	---------	---

# 1 License

```
/*
Copyright (c) 1991-2002, The Numerical Algorithms Group Ltd.
All rights reserved.
```

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of The Numerical Algorithms Group Ltd. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

```
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER
OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*/
```

— \* —

```
#ifndef MSYSplatform

#include <stdio.h>
#include <stdlib.h>
#if !defined(BSDplatform)
#include <malloc.h>
#endif

#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include <X11/Xos.h>
```

```

#include <X11/Intrinsic.h>
#include <X11/StringDefs.h>
#include <X11/cursorfont.h>

#define XDitherWidth 3
#define XDitherMax 10

char XDitherBits[] = {
    0x00, 0x00, 0x00,
    0x00, 0x02, 0x00,
    0x00, 0x03, 0x00,
    0x00, 0x03, 0x02,
    0x00, 0x07, 0x02,
    0x04, 0x07, 0x02,
    0x04, 0x07, 0x03,
    0x05, 0x07, 0x03,
    0x05, 0x07, 0x07,
    0x07, 0x07, 0x07 };

#include "xdither.h1"

Pixmap XDither[XDitherMax];
unsigned int DITHERINIT = 0;

/*
 * This routine has the function of returning the number of characters needed
 * to store a bitmap. It first calculates the number of bits needed per line.
 * Then it finds the closest multiple of 8 which is bigger than the number of
 * bits. Once that is done, it multiplies this number by the number of bits
 * high the bitmap is.
 */
int
dither_char_bitmap(void)
{
    int bits_line;
    int total_chars;

    for (bits_line = 8, total_chars = 1; bits_line < XDitherWidth; total_chars++)
        bits_line += 8;

    total_chars = total_chars * XDitherWidth;

    return total_chars;
}

int
XInitDither(Display *display, int screen, GC gc, unsigned long fg,
            unsigned long bg)

```

```

{

    char *bits;
    int count;
    int chars_bitmap = dither_char_bitmap();
    int bit;
    XGCValues xgcv;

    DITHERINIT = 1;

    /*
     * First thing I should do is load in the Pixmaps
     */
    bits = (char *) malloc(chars_bitmap * sizeof(char));

    for (count = 0; count < XDitherMax; count++) {

        /*
         * Load in the next bitmap
         */
        for (bit = 0; bit < chars_bitmap; bit++)
            bits[bit] = XDitherBits[count * chars_bitmap + bit];

        /*
         * Create it and put it into the Pixmap array
         */
        XDither[count] = XCreatePixmapFromBitmapData(display,
                                                    RootWindow(display, screen),
                                                    bits,
                                                    XDitherWidth, XDitherWidth,
                                                    BlackPixel(display, screen),
                                                    WhitePixel(display, screen),
                                                    1);
    }

    /*
     * Now reset the gc values to be as I need them
     */
    xgcv.background = bg;
    xgcv.foreground = fg;
    xgcv.fill_style = FillOpaqueStippled;
    xgcv.stipple = XDither[4];

    XChangeGC(display, gc,
              GCForeground | GCBackground | GCFillStyle | GCStipple, &xgcv);

    return (XDitherMax);
}

```

```

int
XChangeDither(Display *display, GC gc, int dither)
{
    if (!DITHERINIT) {
        fprintf(stderr, "XChange Error: Init Not Called\n");
        exit(-1);
    }
    if (dither >= XDitherMax || dither < 0) {
        fprintf(stderr, "Dither %d, out of range\n", dither);
        return (-1);
    }
    XSetStipple(display, gc, XDither[dither]);
    return (1);
}

void
XDitherRectangle(Display *display, Drawable drawable, GC gc, int x,
                 int y, unsigned int width, unsigned int height)
{
    if (!DITHERINIT) {
        fprintf(stderr, "xdither Error: Tried to fill before INIT called\n");
        exit(-1);
    }
    XFillRectangle(display, drawable, gc, x, y, width, height);
}

void
XDitherRectangles(Display *display, Drawable drawable, GC gc,
                  XRectangle *rectangles, int nrectangles)
{
    if (!DITHERINIT) {
        fprintf(stderr, "xdither Error: Tried to fill before INIT called\n");
        exit(-1);
    }
    XFillRectangles(display, drawable, gc,
                    rectangles, nrectangles);
}

void
XDitherPolygon(Display * display, Drawable drawable, GC gc,

```

```

        XPoint *points, int npoints, int shape, int mode)
{
    if (!DITHERINIT) {
        fprintf(stderr, "xdither Error: Tried to fill before INIT called\n");
        exit(-1);
    }

    XFillPolygon(display, drawable, gc,
                  points, npoints, shape, mode);
}

void
XDitherArc(Display *display, Drawable drawable, GC gc, int x,int y,
            unsigned int width, unsigned int height, int angle1, int angle2)
{
    if (!DITHERINIT) {
        fprintf(stderr, "xdither Error: Tried to fill before INIT called\n");
        exit(-1);
    }
    XFillArc(display, drawable, gc, x, y, width,
              height, angle1, angle2);
}

void
XDitherArcs(Display *display,Drawable  drawable, GC gc, XArc *arcs,int narcs)
{
    if (!DITHERINIT) {
        fprintf(stderr, "xdither Error: Tried to fill before INIT called\n");
        exit(-1);
    }
    XFillArcs(display, drawable, gc, arcs, narcs);
}
#endif /* MSYSplatform */

```

---

## References

- [1] nothing