

Drawing Three Points: OpenGL

```
#include<GL/gl.h>
#include<GL/glu.h>
#include<GL/glut.h>

void myDisplay(void)
{
    glClear(GL_COLOR_BUFFER_BIT); // clear the screen
    glBegin(GL_POINTS);
        glVertex2i(100, 50);
        glVertex2i(100, 130);
        glVertex2i(150,130);
    glEnd();
    glFlush();
}

void myInit(void)
{
    glClearColor(1.0,1.0,1.0,0.0); // white background
    glColor3f(0.0f, 0.0f, 0.0f); // drawing color
    glPointSize(4.0); // 4x4 pixel dot
    glMatrixMode(GL_PROJECTION); // random necessary wierdness
    glLoadIdentity();
    gluOrtho2D(0.0, 640.0, 0.0, 480.0);
}

void main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(640, 480);
    glutInitWindowPosition(100, 150);
    glutCreateWindow("three dots");
    glutDisplayFunc(myDisplay);
    myInit();
    glutMainLoop();
}
```

Drawing Three Points: Scheme

```
(define *w1* '())

(define (draw-point x y)
  (position-pen x y)
  (graphics-draw-point *w1* x y))

(set! *w1*
      (make-graphics-device x-graphics-device-type #f #f))
(graphics-set-coordinate-limits *w1* 0 640 0 480)
(draw-point 100 50)
(draw-point 100 130)
(draw-point 150 130)
```

Drawing Three Points: X11

```
/*
 * xdots.c
 *
 * Very simple dot-drawing X11/Xt program.
 */
#include <stdio.h>
#include <unistd.h>

/*
 * Toolkit includes
 */
#include <X11/Intrinsic.h>
#include <X11/StringDefs.h>

/*
 * Widget includes
 */
#include <X11/Xaw/Box.h>
#include <X11/Xaw/Form.h>
#include <X11/Xaw/Label.h>
#include <X11/Xaw/MenuButton.h>
#include <X11/Xaw/Scrollbar.h>
#include <X11/Xaw/SimpleMenu.h>
#include <X11/Xaw/SmeBSB.h>
#include <X11/Xaw/SmeLine.h>
#include <X11/Xaw/Viewport.h>

/*
 * Random useful definitions
 */
#define DEFAULT_WIDTH 640 /* width of frame */
#define DEFAULT_HEIGHT 480 /* height of frame */

/*
 * X11 variables.  Because of the usual X11 pointer madness, these work be
 * when they're globals.
 */
Widget topLevel, /* Toplevel window */
form, /* Overall form */
```

```
dotframe; /* Frame to hold dots */
```

```
Display *display;
```

```
GC gc_bg, /* background context */  
gc_fg; /* dot-color context */
```

```
Window window;  
XtAppContext app_context;  
Arg args[10];
```

```
unsigned int color_bg,  
            color_fg;  
static XtResource resources[] = {  
{"bgColor", "bgColor", XtRPixel, sizeof(Pixel),  
 (Cardinal)&color_bg, XtRString, "Black"},  
{"fgColor", "fgColor", XtRPixel, sizeof(Pixel),  
 (Cardinal)&color_fg, XtRString, "White"}  
};
```

```
inline void PROCESS_Xt_EVENT()  
{  
XEvent event;
```

```
while (XtAppPending(app_context) != 0) {  
XtAppNextEvent(app_context, &event);  
XtDispatchEvent(&event);  
}  
XFlush(display);  
}
```

```
int main(argc, argv, envp)  
int argc;  
char **argv;  
char **envp;  
{  
Cardinal n;  
XSetWindowAttributes atr;
```

```
/*  
 * Set up the X11 world
```

```

*/
display = XOpenDisplay(NULL);
/* Everyone's parent */
topLevel= XtAppInitialize(&app_context, "XDots", NULL, 0, &argc, argv,
NULL, NULL, (Cardinal)0);
XtGetApplicationResources(topLevel, (caddr_t) NULL,
resources, XtNumber(resources), NULL, 0);
/* Form holding all widgets */
form = XtCreateManagedWidget("dot", formWidgetClass, topLevel,
NULL, (Cardinal)0);
/* Frame for the dots */
n = 0;
XtSetArg(args[n], XtNwidth, (Cardinal) DEFAULT_WIDTH); n++;
XtSetArg(args[n], XtNheight, (Cardinal) DEFAULT_HEIGHT); n++;
dotframe = XtCreateManagedWidget("dotframe", simpleWidgetClass,
form, args, n);
XtRealizeWidget(topLevel);

/*
 * Graphics contexts, aka colors shapes and sizes.
 */
window = XtWindow(dotframe);
/* basic black background */
gc_bg = XCreateGC(display, window, 0, NULL);
XSetFillStyle(display, gc_bg, FillSolid);
XSetForeground(display, gc_bg, color_bg);
/* dots */
gc_fg = XCreateGC(display, window, 0, NULL);
XSetFillStyle(display, gc_fg, FillSolid);
XSetForeground(display, gc_fg, color_fg);

/* Random useful stuff */
atr.backing_store = Always;
XChangeWindowAttributes(display, window, CWBackingStore, &atr);

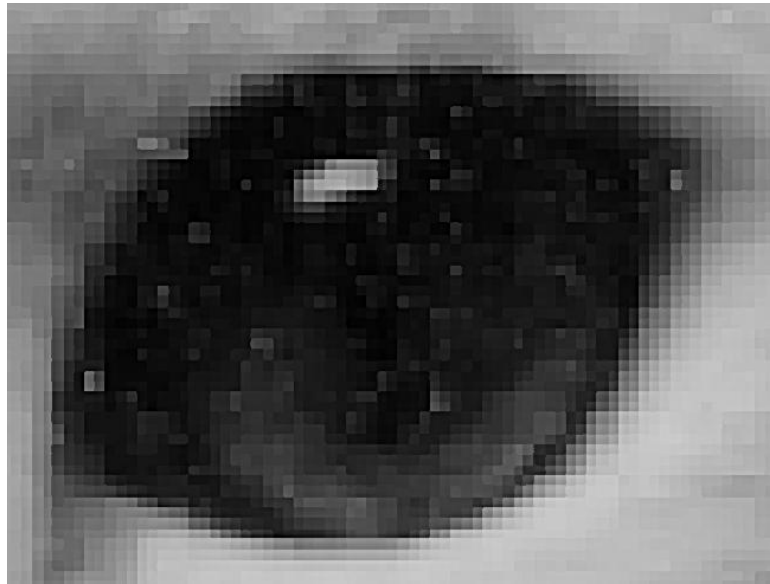
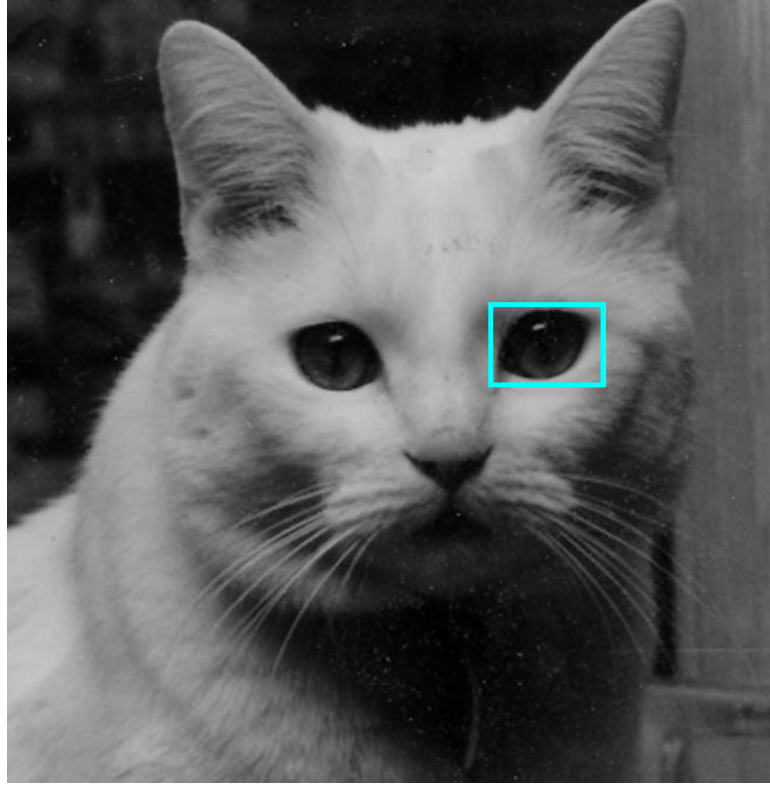
while (1) {
PROCESS_Xt_EVENT();

/*
 * Draw the dots. Since this is all we're doing, we just
 * redraw them after every event.
 */

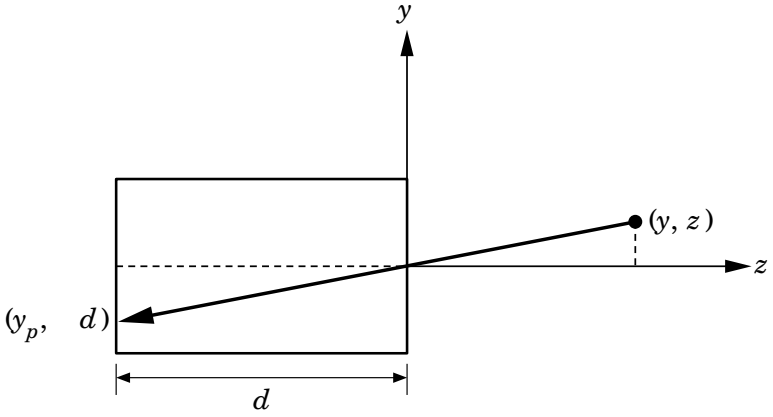
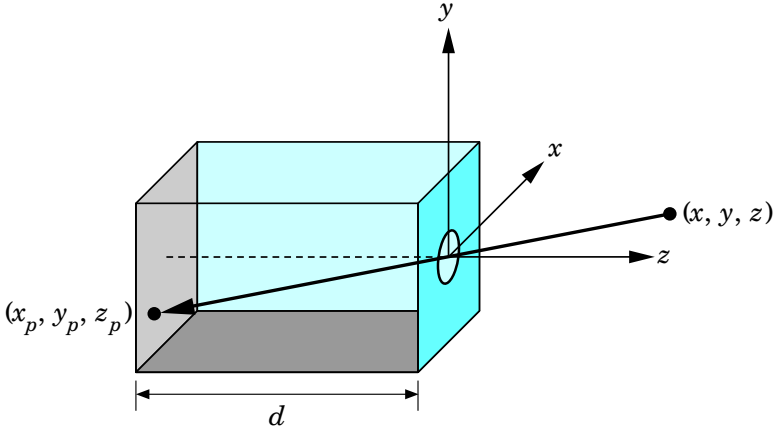
```

```
XDrawPoint(display, window, gc_fg, 100, 130);  
XDrawPoint(display, window, gc_fg, 150, 130);  
XDrawPoint(display, window, gc_fg, 100, 50);  
usleep(20000);  
}
```

Pixellation:



Pinhole Camera:



Pinhole Camera: Angle/Field of View

