
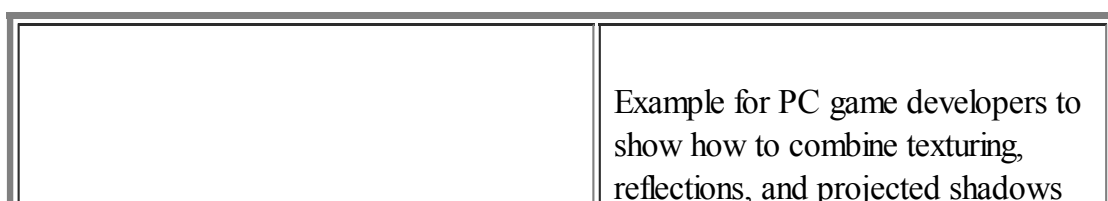
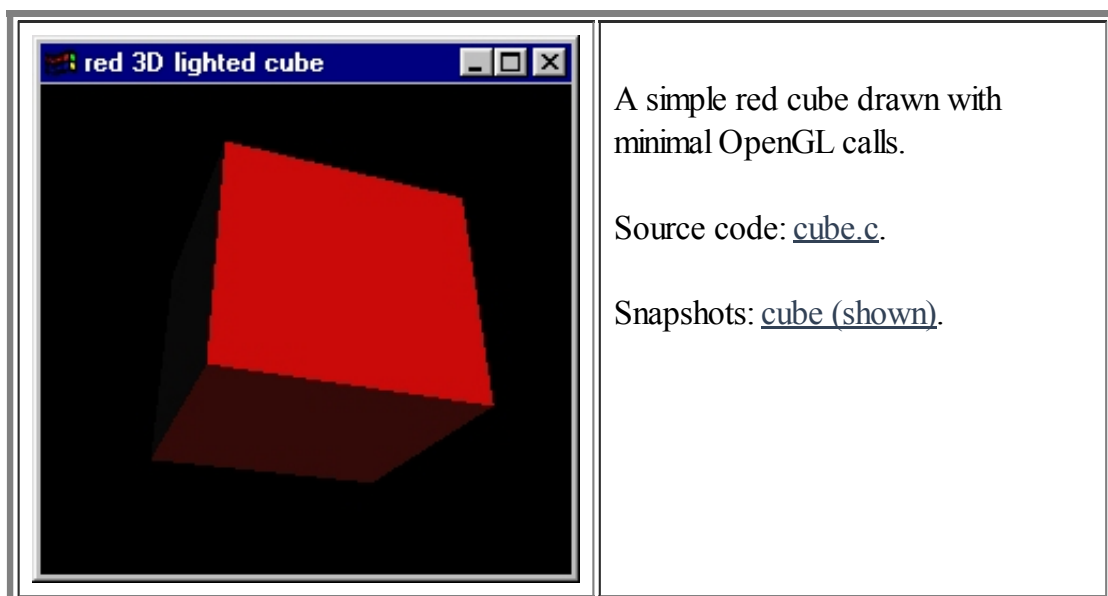
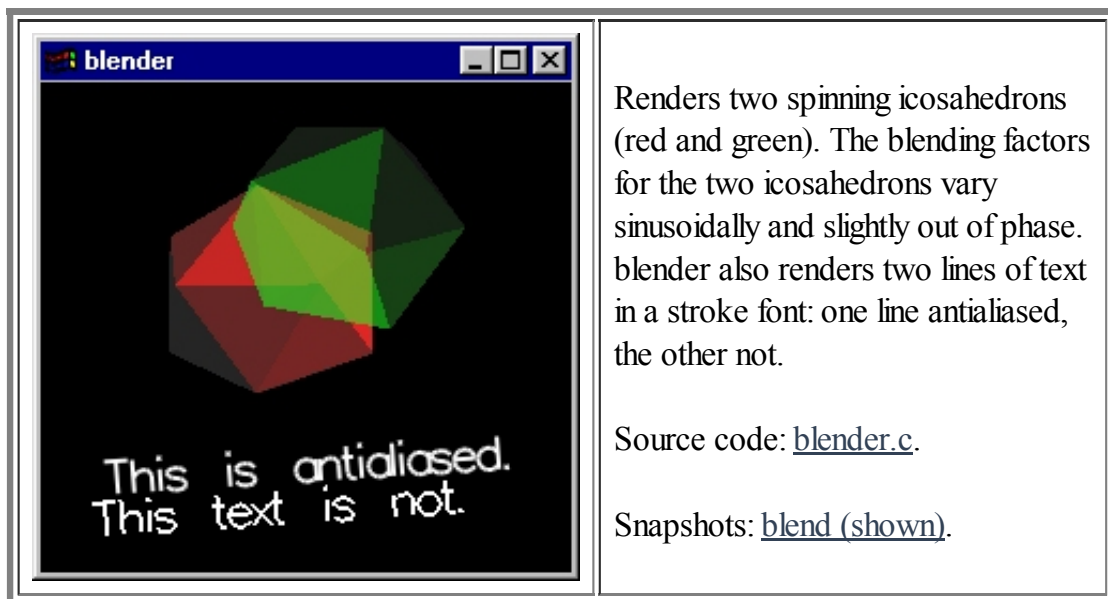
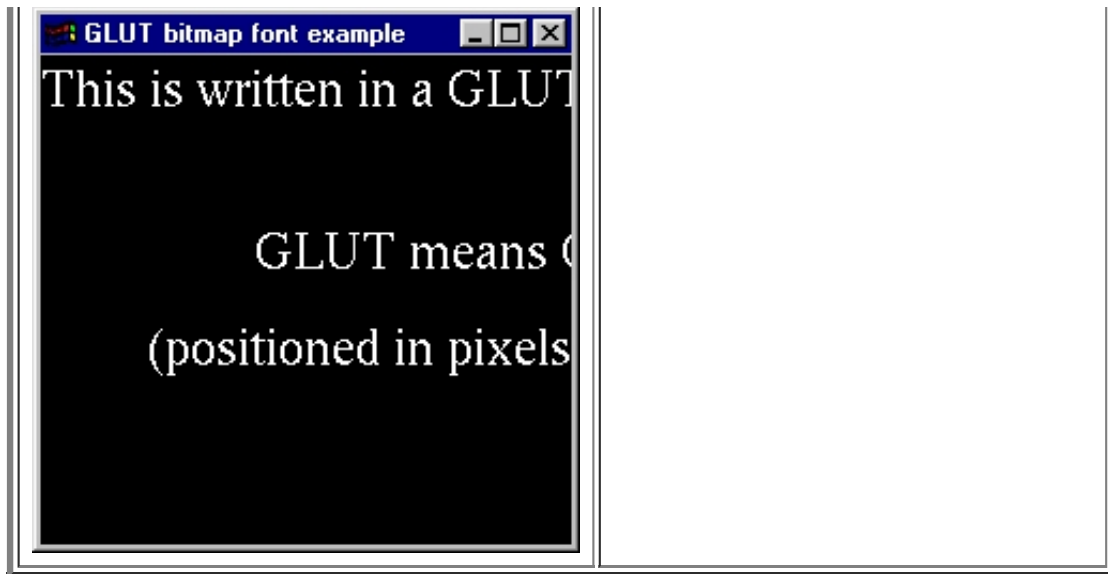


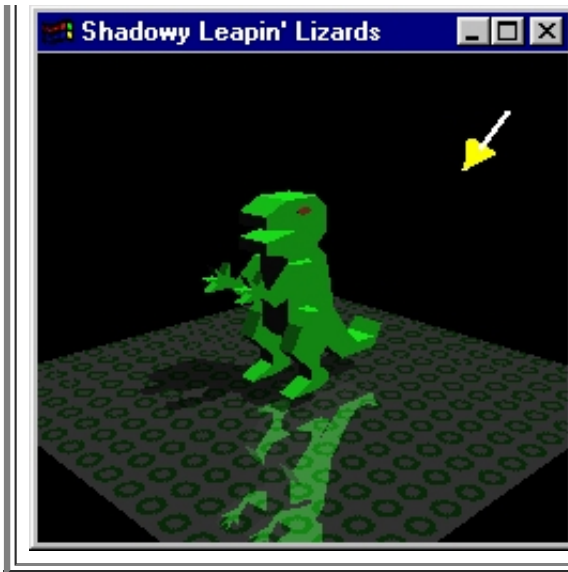
examples.zip



 A screenshot of a window titled "ABGR extension". It displays a 3D scene with a central vertical gap. On the left side, there are horizontal bars of red, green, blue, and magenta. On the right side, there are horizontal bars of red, magenta, yellow, and green. At the top, two triangular shapes meet at a point in the center, with colors transitioning from magenta to green to yellow. The scene is rendered with a black background.	<p>Demonstrates the use of the extension EXT_abgr. The same image data is used for both ABGR and RGBA formats in <code>glDrawPixels</code> and <code>glTexImage2D</code>. The left side uses ABGR, the right side RGBA. The top polygon demonstrates use of texture, and the bottom image is drawn with <code>glDrawPixels</code>. Note that the textures are defined as 3 component, so the alpha value is not used in applying the DECAL environment.</p> <p>Source code: abgr.c.</p> <p>Snapshots: scene (shown).</p>
--	--

	<p>Example of using GLUT bitmap fonts.</p> <p>Source code: bitfont.c.</p> <p>Snapshots: text (shown).</p>
--	---





all in real-time with OpenGL. Robust reflections use stencil. Robust projected shadows use both stencil and polygon offset.

Source code: [dinoshade.c](#).

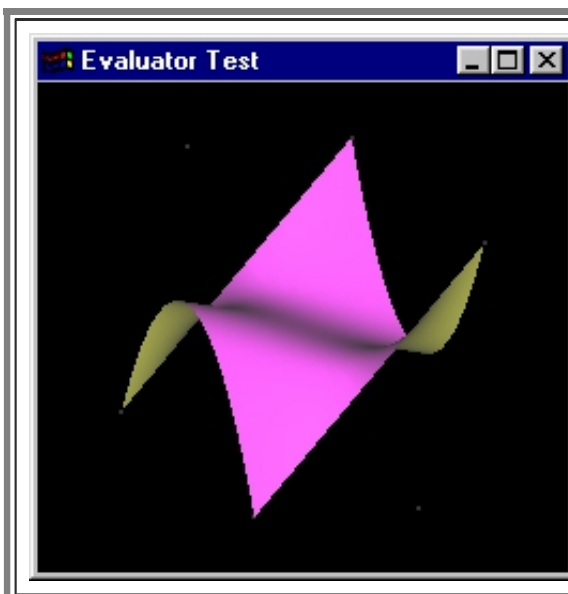
Snapshots: [scene \(shown\)](#).



Arcball like rotation of a chunky dinosaur.

Source code: [dinospin.c](#).

Snapshots: [spinning \(shown\)](#).

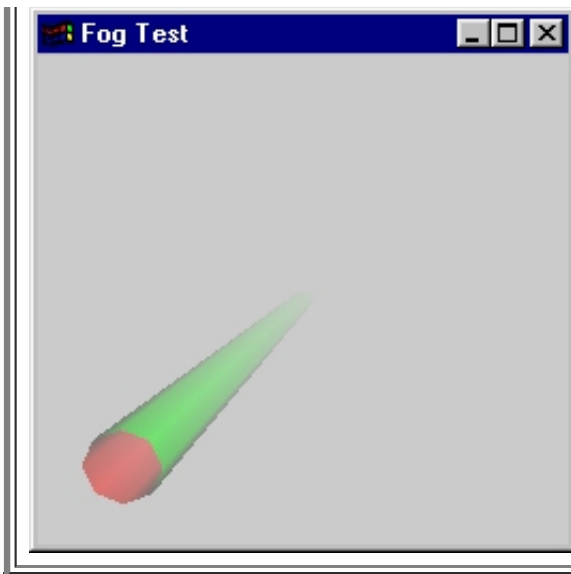


OpenGL evaluators simple example with lots of options.

Source code: [evaltest.c](#).

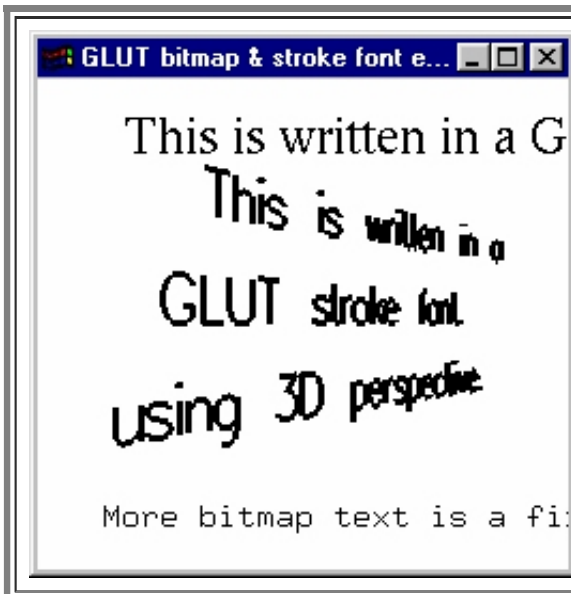
Snapshots: [rotated 3D \(shown\)](#).

Demonstration program exhibiting fog techniques.



Source code: [fogtst.c](#).

Snapshots: [dense fog \(shown\)](#).



Bitmap and stroke fonts demonstration program.

Source code: [fontdemo.c](#).

Snapshots: [fonts \(shown\)](#).



3D puzzle that can solve itself automatically.

Source code: [glpuzzle.c](#).

Snapshots: [mid-game \(shown\)](#).

OpenGL planes a plenty - add and subtract them.



Source code: [glutplane.c](#).

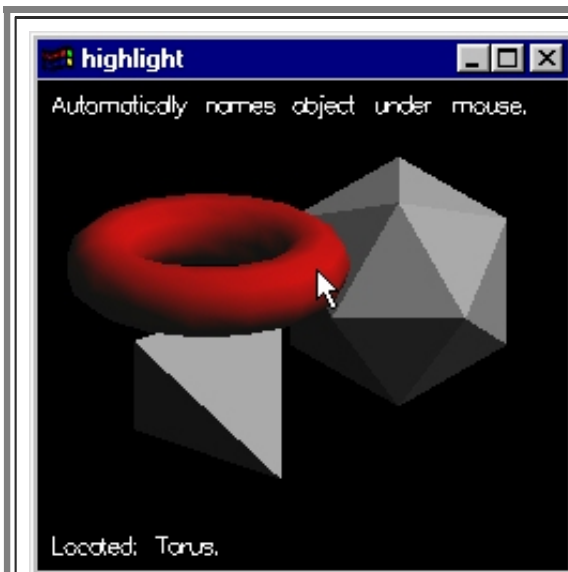
Snapshots: [flying high \(shown\)](#).



Neat haloing effect using the stencil buffer.

Source code: [halomagic.c](#).

Snapshots: [icosahedron \(shown\)](#),
[dinosaur](#).

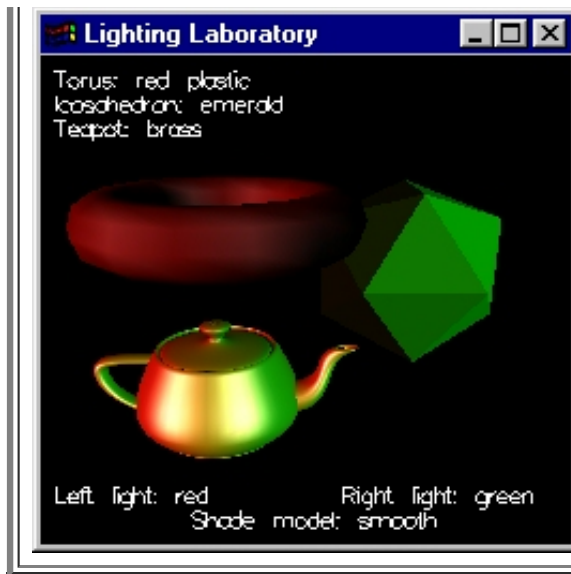


This program demonstrates the use of the GL lighting model. Objects are drawn using a grey material characteristic. A single light source illuminates the objects.

Source code: [highlight.c](#).

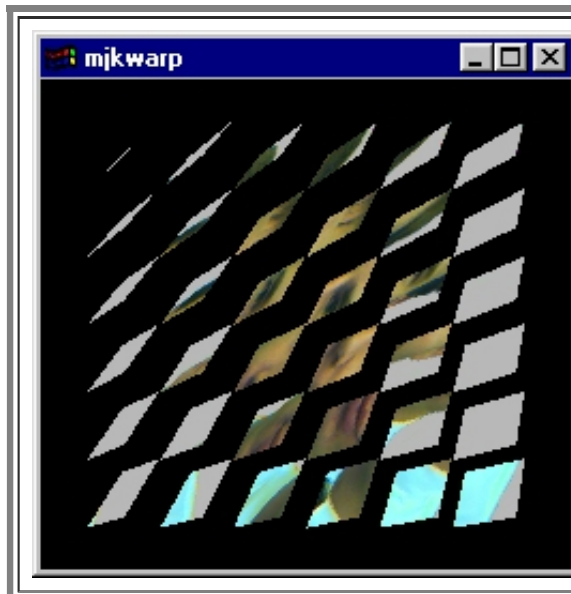
Snapshots: [red torus \(shown\)](#).

Lighting laboratory to experiment with different material properties and lights.



Source code: [lightlab.c](#).

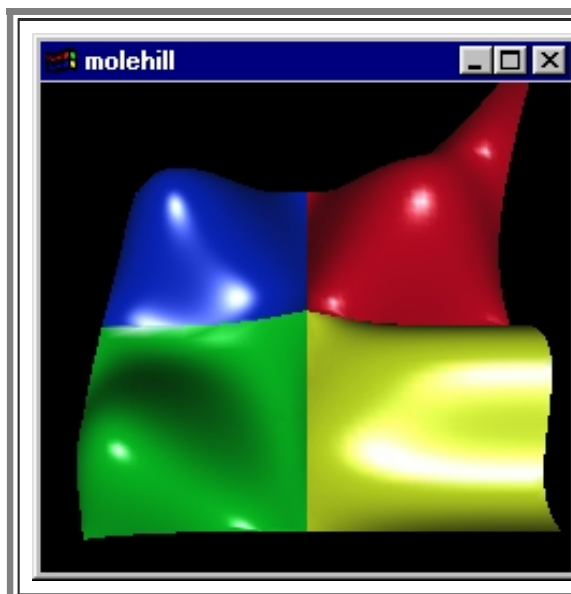
Snapshots: [default \(shown\)](#).



Texture warping example to show many texturing options of OpenGL.

Source code: [mjk warp.c](#).

Snapshots: [breakup \(shown\)](#).

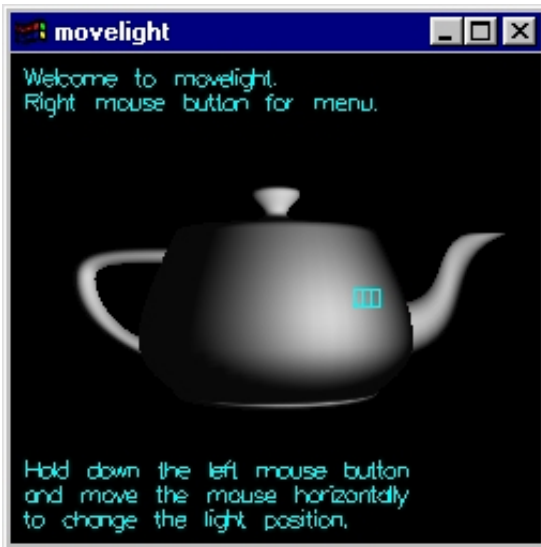


Really, really shiny nurbs/evaluators example.

Source code: [molehill.c](#).

Snapshots: [scene \(shown\)](#).

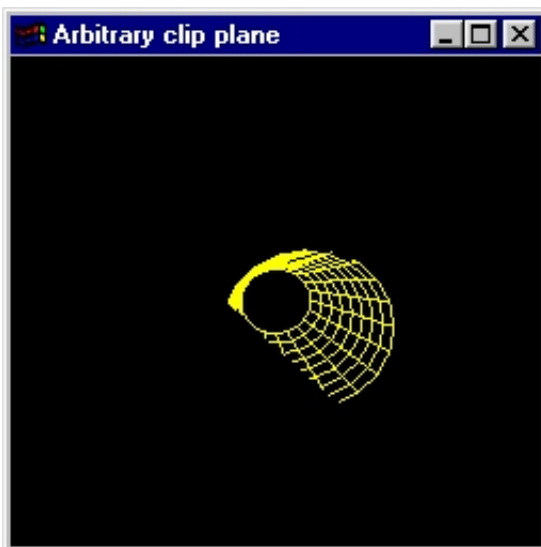
This program demonstrates when to issue lighting and transformation commands to render a model with a



light which is moved by a modeling transformation (rotate or translate). The light position is reset after the modeling transformation is called. The eye position does not change. A sphere is drawn using a grey material characteristic. A single light source illuminates the object. Interaction: pressing the left or middle mouse button alters the modeling transformation (x rotation) by 30 degrees. The scene is then redrawn with the light in a new position.

Source code: [movelight.c](#).

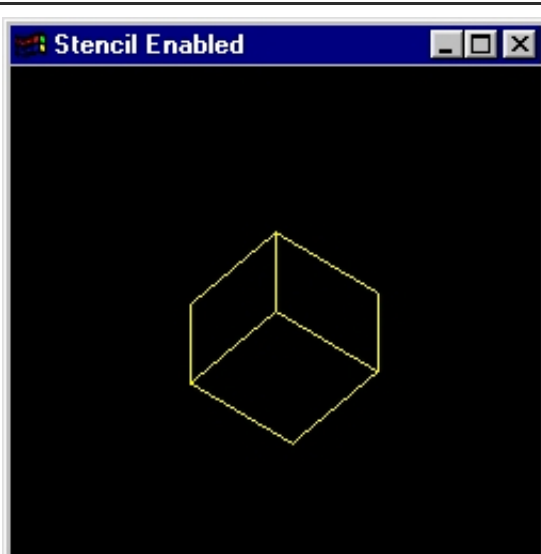
Snapshots: [teapot \(shown\)](#).



The main intent of this program is to demo the arbitrary clipping functionality, hence the rendering is kept simple (wireframe) and only one clipping plane is used.

Source code: [oclip.c](#).

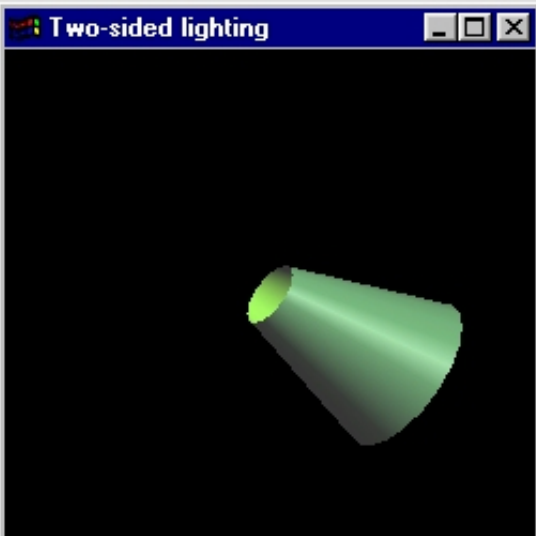
Snapshots: [clip \(shown\)](#).

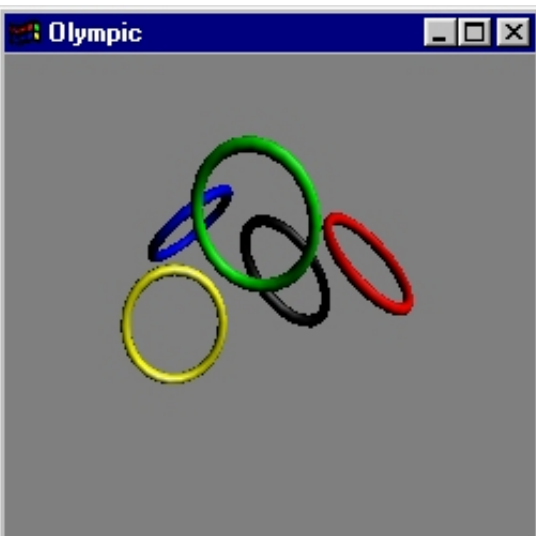


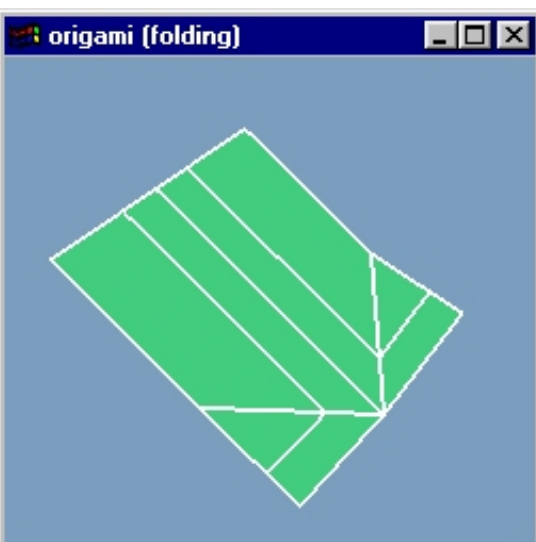
The main intent of this program is to demo the stencil plane functionality, hence the rendering is kept simple (wireframe).

Source code: [ohidden.c](#).

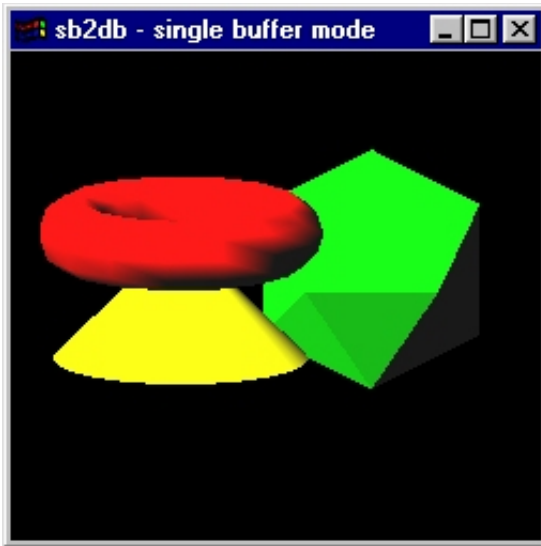
Snapshots: [lines away \(shown\)](#).

	<p>OpenGL example showing how to do hardware lighting including two-sided lighting.</p> <p>Source code: olight.c.</p> <p>Snapshots: two-sided (shown), single-sided.</p>
---	--

	<p>Flying olympic logo. Press spacebar to animate.</p> <p>Source code: olympic.c.</p> <p>Snapshots: motion (shown), final image.</p>
--	--

	<p>Origami folding example (paper airplane).</p> <p>Source code: origami.c.</p> <p>Snapshots: first fold (shown).</p>
---	---

	<p>This program demonstrates switching</p>
--	--



between single buffered and double buffered windows when using GLUT. Use the pop-up menu to change the buffering style used. On machine that split the screen's color resolution in half when double buffering, you should notice better coloration (less or no dithering) in single buffer mode (but flicker on redraws, particularly when rotation is toggled on).

Source code: [sb2db.c](#).

Snapshots: [scene \(shown\)](#).

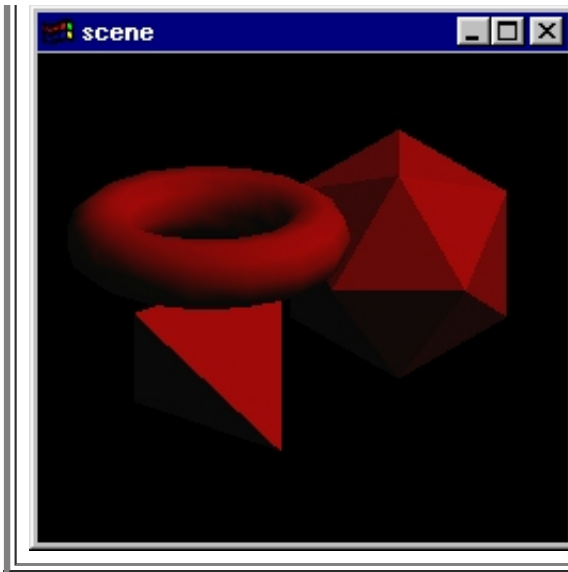


Very simple example of how to achieve reflections on a flat surface using OpenGL blending. The example has a mode using OpenGL stenciling to avoid drawing the reflection not on the top of the floor. Initially, stenciling is not used so if you look (by holding down the left mouse button and moving) at the dinosaur from "below" the floor, you'll see a bogus dinosaur and appreciate how the basic technique works. Enable stenciling with the popup menu and the bogus dinosaur goes away! Also, notice that OpenGL lighting works correctly with reflections.

Source code: [reflectdino.c](#).

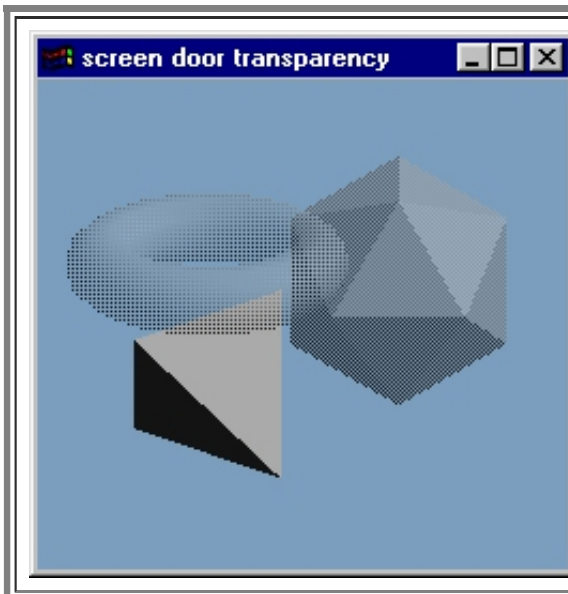
Snapshots: [reflectdino \(shown\)](#).

This program demonstrates the use of the GL lighting model. Objects are drawn using a grey material characteristic. A single light source illuminates the objects.



Source code: [scene.c](#).

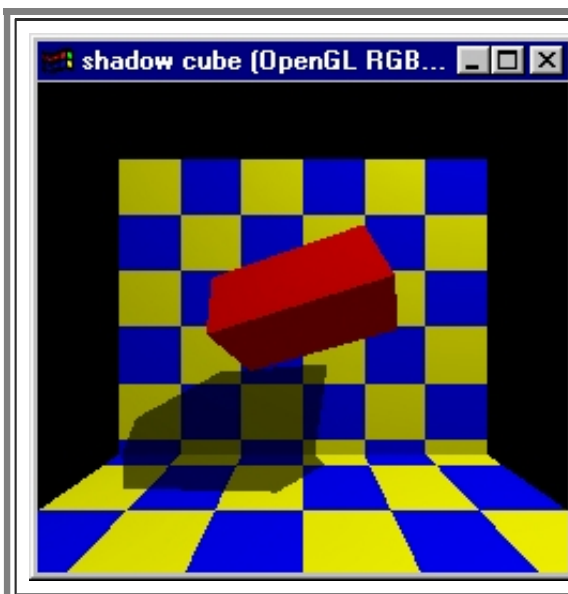
Snapshots: [scene \(shown\)](#).



Demonstrates "screen door" transparency using OpenGL's polygon stipple feature.

Source code: [screendoor.c](#).

Snapshots: [default \(shown\)](#).

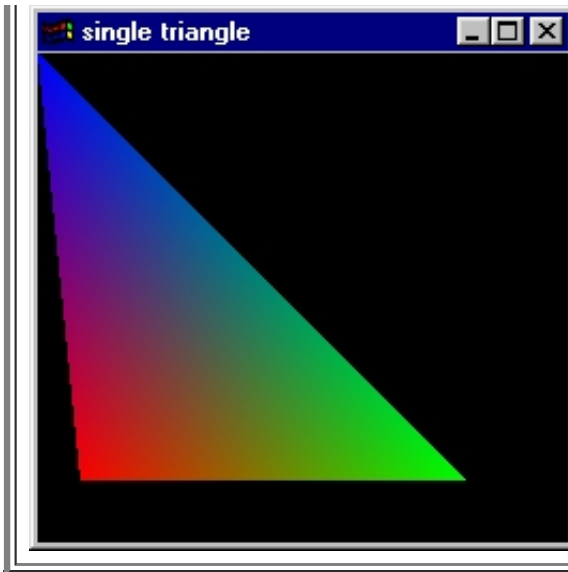


Spinning cube with dynamic shadow (runs in color index mode too!).

Source code: [scube.c](#).

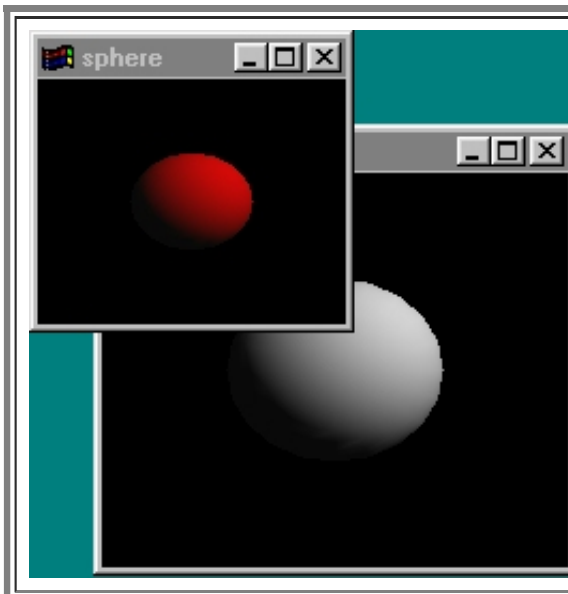
Snapshots: [rgb \(shown\)](#), [color index](#).

Very simple example of how to draw a single triangle with OpenGL.



Source code: [simple.c](#).

Snapshots: [scene \(shown\)](#).



Multiple windows example with GLUT.

Source code: [sphere.c](#).

Snapshots: [staged \(shown\)](#).

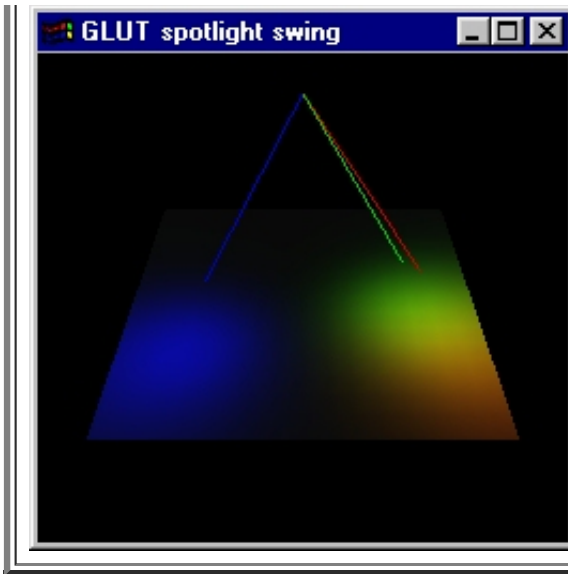


Fun with bitmaps (and pixel transfer maps) in OpenGL.

Source code: [splatlogo.c](#).

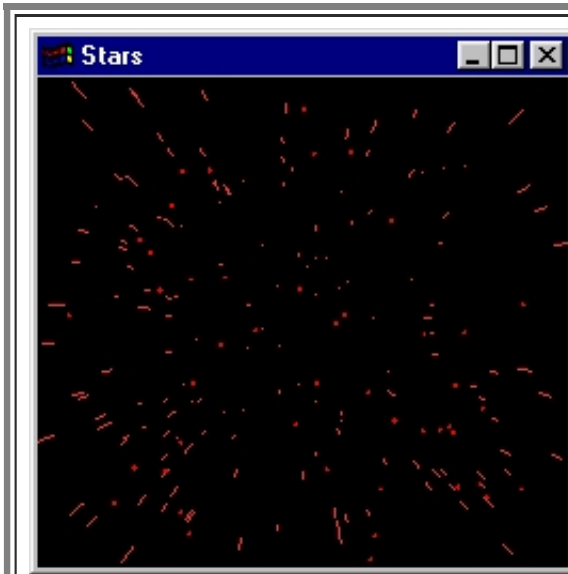
Snapshots: [messy \(shown\)](#).

Excellent spotlight example showing direction of lights too.



Source code: [spots.c](#).

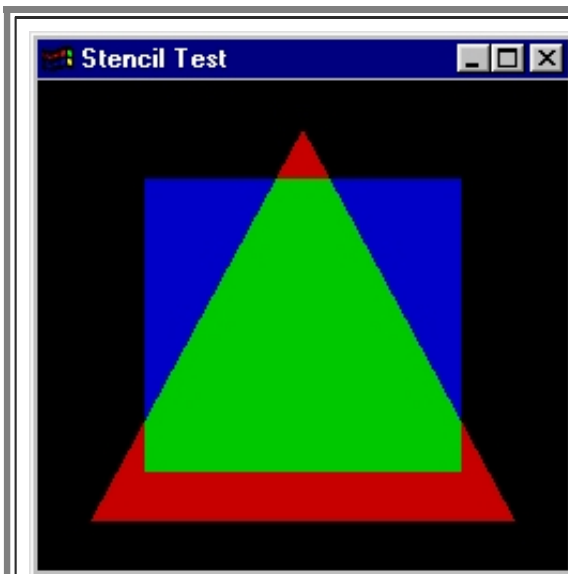
Snapshots: [spots \(shown\)](#).



Stars example program with turbo and crazy mode.

Source code: [stars.c](#).

Snapshots: [warp \(shown\)](#).



An example of how to use the stencil test.

Source code: [stencilst.c](#).

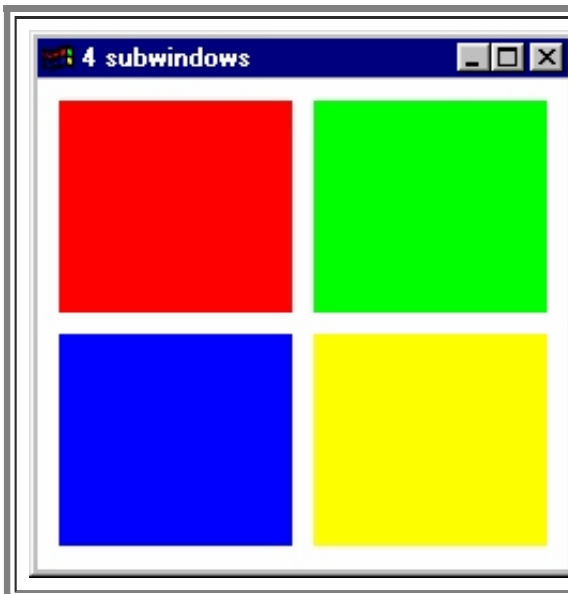
Snapshots: [scene \(shown\)](#).

Stroke fonts using GLUT
(antialiased).



Source code: [stroke.c](#).

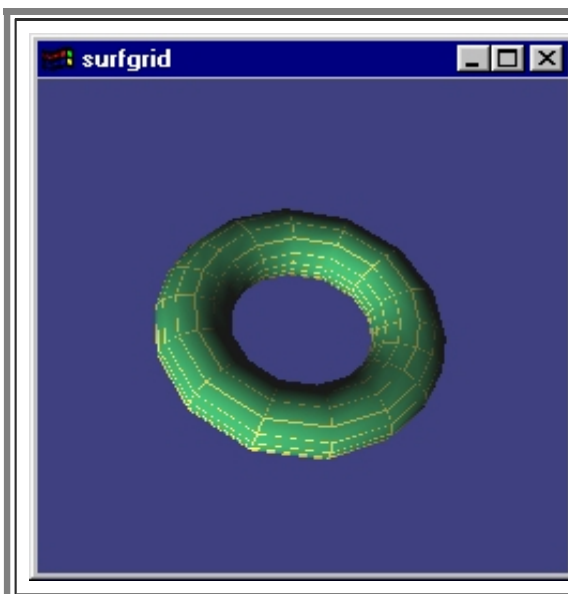
Snapshots: [spin \(shown\)](#).



Subwindows with GLUT - each gets its own color.

Source code: [subwin.c](#).

Snapshots: [default \(shown\)](#).

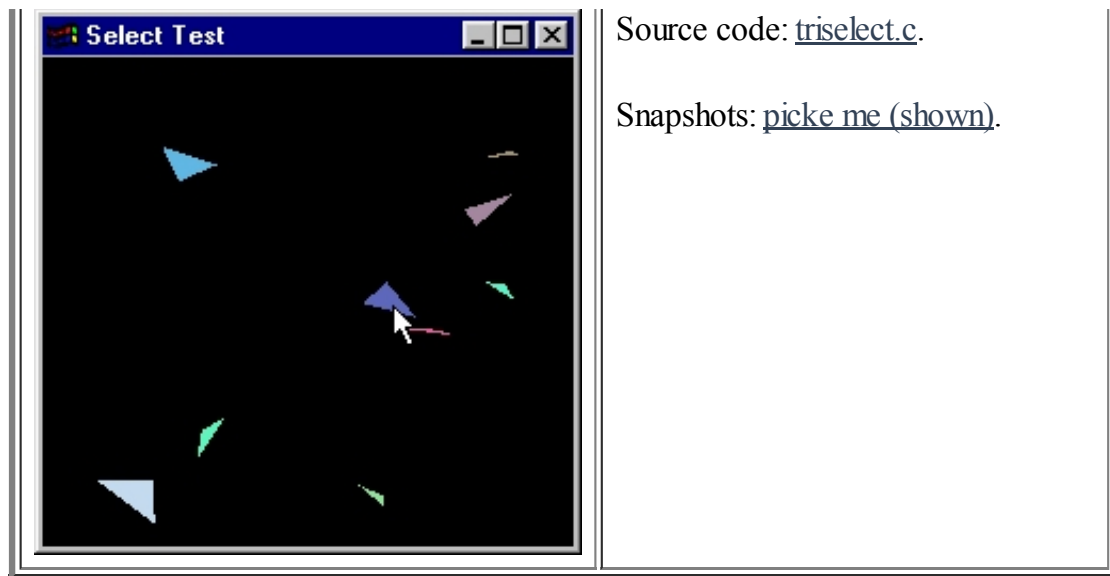


A surface with the grid that it is defined by highlighted.

Source code: [surfgrid.c](#).

Snapshots: [torus \(shown\)](#).

Triangle selection program. Click on the triangles to change their color.



Copyright © 1997 Silicon Graphics Incorporated.