

6. If you are drawing an object that looks fine at normal scale but appears to have the wrong lighting effects when drawn after a scale operation you should compute the object at the final size and avoid scaling. For example, a scatter plot draws markers by default with an isotropic scaling factor of 1 (no scaling). If you create a sphere object with radius 1 for use in a scatter plot, you would need to reduce its size somewhere. You can do so by setting the scale in the Scatter Properties dialog or by reducing the radius of the sphere.

Troubleshooting Gizmo Transparency Problems

1. Check the display list to make sure that there is a GL_BLEND enable operation above the transparent objects and that a blend function exists and has the correct parameters. See **Transparency and Translucency** on page II-432 for details.
2. Check the drawing order of objects. Transparent objects must be after opaque objects in the display list.
3. If you have transparent surfaces that have some overlapping facets, you have no choice but to decompose the surface into triangles and order them according to depth. You can find an example of this in the Depth Sorting demo experiment.

Troubleshooting Gizmo Clipping Problems

Try these tips if an object or part of an object is clipped or not visible.

1. Set the axis range to auto for all axes using Gizmo Menu→Axis Range.
2. Make sure that the display list does not contain any clipping planes.
3. Make sure that the object in question does not follow in the display list after translate, rotate or scale operations.
4. Check the zoom level and the main transformation. If you do not have any transformation operations on the display list insert a default ortho operation (+/-2 for all axes) at the top of the display list.

Gizmo Compatibility

Before Igor7 Gizmo was implemented based on the original OpenGL 1.0 specification. Changes in OpenGL, especially since OpenGL 3.0, required that we modify various features of Gizmo. We have made an effort to allow Gizmo windows in old Igor experiments to open without error and to appear as close as possible to their original form. Notable exceptions include the orientation of string objects, axis labels and tick mark labels; see **Changes to Gizmo Text** on page II-471.

We have also marked in the Gizmo reference documentation some features as "obsolete" or "deprecated". "Obsolete" means that the feature is no longer supported. "Deprecated" means that it may be removed from a future version of Igor and you should use an alternative feature if possible.

Gizmo Commands in User-Defined Functions

In Igor7 and later Gizmo commands can be used in user-defined functions. Supporting this required some changes to the command syntax. The main syntax change is the addition of the objectType keyword in ModifyGizmo. Prior to Igor7 you could use:

```
ModifyGizmo modifyObject=quad0, property={calcNormals,1}
```

The Igor7 syntax requires adding the objectType=<type> keyword as in:

```
ModifyGizmo modifyObject=quad0, objectType=Quad, property={calcNormals,1}
```

So that Igor can interpret Gizmo recreation macros created by Igor6, the objectType keyword is required only in user-defined functions, not from the command line or in macros.

For backward compatibility, the Gizmo XOP as of Igor Pro 6.30 accepts the objectType keyword.