Contents

List of Examples xvii

List of Figures xix

List of Tables xxi

About This Guide xxiii
What This Guide Contains xxiii
What You Should Know Before Reading This Guide xxv
Background Reading xxv
OpenGL and Associated Tools and Libraries xxv
X Window System: Xlib, X Toolkit, and OSF/Motif xxv
Other Sources xxvi
Conventions Used in This Guide xxvi
Typographical Conventions xxvi
Function Naming Conventions xxvii

1. OpenGL on Silicon Graphics Systems 1
Using OpenGL With the X Window System 1
The GLX Extension to the X Window System 2
Libraries, Tools, Toolkits, and Widget Sets 2
RapidApp 3
Open Inventor 4
IRIS ViewKit 5
The IRIS IM Widget Set 5
The Xlib Library 5
A Note to IRIS GL Users 5
Extensions to OpenGL 6

Debugging and Performance Optimization 7
Debugging Your Program 7
Tuning Your OpenGL Application 7
Maximizing Performance With IRIS Performer 8
Location of Example Source Code 8

2. OpenGL and X: Getting Started 9
Background and Terminology 9
The X Window System on Silicon Graphics Systems 9
The Silicon Graphics X Server 10
The GLX Extension to X 10
Compiling With the GLX Extension 10
X Window System Concepts 11
Visuals 11
Drawables 12
Rendering Contexts 12
Resources 13
Colormaps 13
Libraries, Toolkits, and Tools 14
Widgets and the Xt Library 14
The Xt Library 15
For More Information 15
Other Toolkits and Tools 15
Integrating Your OpenGL Program With IRIS IM 15
Simple Motif Example Program 16
Looking at the Example Program 19
Opening the X Display 19
Selecting a Visual 20
Creating a Rendering Context 22
Creating the Window 22
Binding the Context to the Window 22
Mapping the Window 23
Summary 24

Compiling With OpenGL and Related Libraries 24
Link Lines for Individual Libraries 24
Link Lines for Groups of Libraries 25

3. OpenGL and X: Examples 27
Using Widgets 27
What are OpenGL Drawing-Area Widgets? 28
Drawing-Area Widget Setup and Creation 29
Setting Up Fallback Resources 29
Creating the Widgets 30
Choosing the Visual for the Drawing-Area Widget 31
Creating Multiple Widgets With Identical Characteristics 31
Using Drawing-Area Widget Callbacks 32
Input Handling With Widgets and Xt 34
Background Information 34
Using the Input Callback 34
Using Actions and Translations 36
Creating Colormaps 37
Widget Troubleshooting 37
Keyboard Input Disappears 38
Inheritance Issues 38
Using Xlib 39
Simple Xlib Example Program 39
Creating a Colormap and a Window 42
Installing the Colormap 44
Xlib Event Handling 45
Handling Mouse Events 45
Exposing a Window 47
Using Fonts and Strings 47

4. OpenGL and X: Advanced Topics 51
Using Animations 51
Swapping Buffers 52
Controlling an Animation With Workprocs 53
General Workproc Information 53
Workproc Example 54
Controlling an Animation With Timeouts 56
Using Overlays 58
Introduction to Overlays 58
A Note for IRIS GL Users 60
How to Create Overlays 60
Overlay Troubleshooting 62
Rubber Banding 63
Using Popup Menus With the GLwMDrawingArea Widget 64
Using Visuals 66
Some Background on Visuals 66
Running OpenGL Applications Using a Single Visual 68
Using Colormaps 69
Background Information About Colormaps 70
Color Variation Across Colormaps 70
Multiple Colormap Issues 71
Choosing Which Colormap to Use 72
Colormap Example 74
Stereo Rendering 74
Stereo Rendering Background Information 75
Quad Buffer Stereo 75
Divided-Screen Stereo 75
For More Information 75
How to Do Stereo Rendering 76
Performing Stereo Rendering on High-End Systems 76
Performing Stereo Rendering on Low-End and Mid-Range Systems 76

Using Pixmaps 77
Creating and Using Pixmaps 78
Direct and Indirect Rendering 79
Performance Considerations for X and OpenGL 79
Portability 80

5. Introduction to OpenGL Extensions 81
Determining Extension Availability 81
How to Check for OpenGL Extension Availability 82
Example Program: Checking for Extension Availability 82
Checking for GLX Extension Availability 84
Finding Information About Extensions 84

6. Texturing Extensions 85
The Texture Extension 85
Why Use the Texture Extension? 86
Using the Texture Extension 86
The Texture Extension Proxy Mechanism 89
Extended Functions 93
The Texture Object Extension 93
How Texture Objects Work 93
Using Texture Objects 94
Texture Object Names 94
Editing and Querying Texture Objects 95
Texture Priorities and Residency 95
Default Textures 95
New and Extended Functions 96
The Subtexture Extension 96
Why Use the Subtexture Extension? 96
Using Subtextures 97
Using Null Images With Subtextures 97
New Functions 98

The Copy Texture Extension 98
Why Use the Copy Texture Extension? 98
Copying Texture Images 98
New Functions 99
The 3D Texture Extension 100
Why Use the 3D Texture Extension? 100
Using 3D Textures 101
3D Texture Example Program 102
New Functions 105
The Texture Color Table Extension 105
Why Use a Texture Color Table? 105
Using Texture Color Tables 106
The Sharpen Texture Extension 107
Using the Sharpen Texture Extension 107
How to Use the Sharpen Texture Extension 108
How Sharpen Texture Works 108
Customizing the LOD Extrapolation Function 109
Using Sharpen Texture and Texture Object 110
Sharpen Texture Example Program 110
New Functions 112
The Detail Texture Extension 112
Using the Detail Texture Extension 113
Creating a Detail Texture and a Low-Resolution Texture 113
How Detail Texture Is Computed 115
Customizing the Detail Function 116
Using Detail Texture and Texture Object 117
Detail Texture Example Program 117
New Functions 119
Texture Clamp Extensions 119
Texture Clamping Background Information 119
Why Use the Texture Clamp Extensions? 120
Using the Texture Clamp Extensions 120

The Texture LOD Extension 121
Specifying a Minimum or Maximum Level of Detail 121
Specifying Image Array Availability 121

7. Imaging and Blending Extensions 123
Introduction to Imaging and Blending Extensions 123
Where Extensions Are in the Imaging Pipeline 123
Functions Affected by Imaging Extensions 125
The ABGR Extension 125
The Packed Pixels Extension 125
Why Use the Packed Pixels Extension? 126
Using Packed Pixels 126
What the Different Pixel Types Mean 127
The Color Matrix Extension 128
The Convolution Extension 129
Performing Convolution 129
Separable and General Convolution Filters 131
New Functions 131
The Histogram and Minmax Extensions 132
Using the Histogram Extension 133
Using the Minmax Part of the Histogram Extension 134
Using Proxy Histograms 135
New Functions 136
The Color Table Extension 136
Why Use the Color Table Extension? 136
Specifying a Color Table 136
Using Framebuffer Image Data for Color Tables 138
Where Are the Lookup Tables in the Image Pipeline? 138
New Functions 139
The Interlace Extension 139
Using the Interlace Extension 139
How to Use Interlace and Copy Texture Together 140

Blending Extensions 140
The Constant Color Blending Extension 141
Using Constant Colors for Blending 141
New Functions 142
The Minmax Blending Extension 142
Using a Blend Equation 143
New Functions 143
The Logical Operation Blending Extension 143
The Blend Subtract Extension 144

8. Miscellaneous OpenGL Extensions 145
The Polygon Offset Extension 145
Polygon Offset Example Program 146
New Functions 150
The Vertex Array Extension 151
Why and When to Use the Vertex Array Extension 151
How the Vertex Array Extension Works 151
Specifying Arrays for Different Kinds of Data 152
Rendering the Arrays 153
New Functions 154
The Multisample Extension 154
Introduction to Multisampling 155
When to Use Multisampling 155
Using the Multisample Extension 155
Using Advanced Multisampling Options 156
Color Blending and Screen-Door Transparency 157
Using a Multisample Mask to Fade Levels of Detail 158
Accumulating Multisampled Images 159
How Multisampling Affects Different Primitives 160
Multisampled Points 160
Multisampled Lines 160
Multisampled Polygons 161
Multisample Rasterization of Pixels and Bitmaps 161
New Functions 161

9. Extensions to GLX 163
The Make Current Read Extension 164
Possible Match Errors 165
New Functions 165
The Visual Info Extension 165
Using the Visual Info Extension 165
Using Transparent Pixels 166
The Visual Rating Extension 167
Using the Visual Rating Extension 167
The Video Synchronization Extension 167
New Functions 168
The Swap Control Extension 168
New Functions 168
The Import Context Extension 169
Importing a Context 169
Retrieving Display and Context Information 170
New Functions 170
The Framebuffer Configuration Extension 171
Why Use the Framebuffer Configuration Extension? 171
Describing a Drawable With a GLXFBConfig Construct 171
Less-Rigid Similarity Requirements When Matching Context and Drawable 172
Less-Rigid Match of GLX Visual and X Visual 172
The GLXFBConfig Construct 173
Choosing a GLXFBConfig Construct 173
Retrieving GLXFBConfig Attribute Values 177
How a GLXFBConfig Is Selected 178
New Functions 178
The Pixel Buffer Extension 179
Rendering to a GLXPbuffer 181
Directing the Buffer Clobber Event 181
New Functions 183

The Video Source Extension 183
New Functions 194

10. Debugging OpenGL Programs 195
Ogldebug Overview 195
How ogldebug Operates 196
Setting Up ogldebug 196
Using a Configuration File 196
Basic Debugging With ogldebug 197
Starting ogldebug 197
Creating a Trace File to Discover OpenGL Problems 198
Interacting With ogldebug 199
Commands for Basic Interaction 199
Understanding Display Areas 199
Using Checkboxes 200
Control-Flow Checkboxes 200
Trace Checkboxes 201
Fast Mode Checkboxes 201
Using Menus to Interact With ogldebug 202
Using the File Menu to Interact With ogldebug 202
Using the Commands Menu to Interact With Your Program 203
Using the Options Menu to Access Information 203
Using the References Menu for Background Information 204
Tips for Debugging Graphics Programs 205
General Debugging Tips 205
Specific Problems and Troubleshooting 206
Blank Window 206
Rotation and Translation Problems 206
Depth Buffering Problems 207
Animation Problems 207
Lighting Problems 207
X Window System Problems 208
Pixel and Texture Write Problems 208
System-Specific Problems 208

11. Tuning Graphics Applications: Fundamentals 209
Why Is Tuning Useful? 210
What Is Pipeline Tuning? 210
Three-Stage Model of the Graphics Pipeline 210
Isolating Bottlenecks in the Pipeline: Overview 212
Factors Influencing Performance 213
Tuning Animation 214
Factors Contributing to Animation Speed 214
Optimizing Frame Rate Performance 215
Optimizing Cache and Memory Use 215
How Memory Is Organized 215
How to Minimize Memory Paging 216
Minimizing Lookup 216
Minimizing Cache Misses 217
Measuring Cache-Miss and Page-Fault Overhead 218
Taking Timing Measurements 218
Benchmarking Basics 219
Achieving Accurate Timing Measurements 219
Achieving Accurate Benchmarking Results 221

12. Tuning the Pipeline 223
CPU Tuning: Basics 223
Immediate Mode Drawing Versus Display Lists 224
CPU Tuning: Immediate Mode Drawing 225
Optimizing the Data Organization 226
Optimizing Database Rendering Code 227
Examples for Optimizing Data Structures for Drawing 227
Examples for Optimizing Program Structure 228
Using Specialized Drawing Subroutines and Macros 231
Preprocessing Drawing Data: Introduction 232
Preprocessing Meshes Into Fixed-Length Strips 232
Preprocessing Vertex Loops 233
CPU Tuning: Display Lists 234

CPU Tuning: Advanced Techniques 236
Mixing Computation With Graphics 236
Examining Assembly Code 237
Using Additional Processors for Complex Scene Management 237
Modeling to the Graphics Pipeline 237
Tuning the Geometry Subsystem 238
Using Fast Drawing Modes 239
Using Peak Performance Primitives for Drawing 239
Optimizing Lighting Performance 240
Lighting Operations With Noticeable Performance Costs 240
Lighting Operations With Significant Performance Costs 241
Practices to Avoid 242
Using Expensive Modes Efficiently 242
Advanced Transform-Limited Tuning Techniques 243
Tuning the Raster Subsystem 243
Using Backface/Frontface Removal 244
Using Expensive Per-Fragment Operations Efficiently 244
Using Depth-Buffering Efficiently 245
Other Considerations 245
Balancing Polygon Size and Pixel Operations 246
Clearing the Color and Depth Buffers Simultaneously 246
Tuning the Imaging Pipeline 246

13. Tuning Graphics Applications: Examples 249
Drawing Pixels Fast 249
Tuning Example 251
Testing for CPU Limitation 260
Using the Profiler 261
Testing for Fill Limitation 263
Working on a Geometry-Limited Program 264
Smooth Shading Versus Flat Shading 264
Reducing the Number of Polygons 264
Testing Again for Fill Limitation 265

14. System-Specific Tuning 267
Optimizing Performance on Low-End Graphics Systems 267
Choosing Features for Optimum Performance 268
Using the Pipeline Effectively 268
Using Geometry Operations Effectively 269
Optimizing Line-Drawing 269
Optimizing Triangles and Polygons 270
Using Per-Fragment Operations Effectively 270
Getting the Optimum Fill Rates 270
Using Per-Fragment Operations Effectively 272
Low-End Specific Extensions 273
Optimizing Performance on Mid-Range Systems 274
General Performance Tips 274
Optimizing Geometry Operations on Mid-Range Systems 274
Optimizing Per-Fragment Operations on Mid-Range Systems 275
Optimizing Performance on Indigo2 IMPACT Systems 276
General Tips for Performance Improvement 276
Achieving Peak Geometry Performance 277
Using Textures 278
Using Images 279
Accelerating Color Space Conversion 280
Using Display Lists Effectively 280
Display List Compilation on Indigo2 IMPACT Hardware 281
DMA Display Lists on Indigo2 IMPACT Systems 281
Offscreen Rendering Capabilities 282
Optimizing Performance on RealityEngine Systems 282
Optimizing Geometry Performance 282
Optimizing Rasterization 283
Optimizing Use of the Vertex Array Extension 284
Optimizing Multisampling and Transparency 284
Optimizing the Imaging Pipeline 285
Using the Color Matrix and the Color Writemask 285

A. OpenGL and IRIS GL 287
Some IRIS GL Functionality and OpenGL Equivalents 287

B. Benchmarks 297

C. Benchmarking Libraries: libpdb and libisfast 305
Libraries for Benchmarking 306
Using libpdb 306
Example for pdbRead 309
Example for pdbMeasureRate() 310
Example for pdbWriteRate() 311
Using libisfast 311

Index 313