# *Making Cool QuickDraw 3D Applications!*

By
Brian Greenstone
Apple Computer, Inc.

# Introduction

QuickDraw 3D is perhaps the best-designed API that Apple Computer has ever created. It provides a way for the average programmer who is not well versed in 3D programming to create complex 3D scenes in very little time with very little effort. The API is so robust that even beginner programmers should have no problem diving into this otherwise convoluted and unstandardized technology we all know as "3D Graphics". A recent survey of 25 QuickDraw 3D developers showed that the average developer gave the QuickDraw 3D API an 8.8 out of 10 where 0 was awful, 5 was average and 10 was great.

At first glance QuickDraw 3D may seem overwhelming (the QuickDraw 3D 1.5 Technical Reference is almost 1400 pages long!), but keep in mind that you'll probably never use 70% of what's in the book. The reality of the situation is that to use QuickDraw 3D and to use it well, you only need to be familiar with a small subset of the functions that QuickDraw 3D provides.

This document is going to focus on that small subset of API functions which are needed to create fast and efficient 3D worlds. We will not be discussing the slow and difficult to use geometries such as NURBS, but rather we will focus mainly on one geometry in particular: the TriMesh. This is the data structure which is the easiest to work with and also provides the maximum rendering performance, especially with 3D accelerator hardware.

There are many QuickDraw 3D applications available today, but very few of the programmers who wrote them knew how to write the code in such a way as to achieve maximum performance. I've seen some 3DMF geometry files created by these applications which render up to 13 times faster after they have been reoptimized with the techniques discussed in this book. It's not that these weren't good programmers, it's just that there is very little information available to developers to teach them what works best in QuickDraw 3D.

Many people have asked me if I really think QuickDraw 3D is “fast.” Well, yes, it’s “fast”, but “fast” is a relative term. QuickDraw 3D will never be as fast as a custom 3D engine which you might write for a specific task such as a game or a modeling application. No general purpose API is ever as fast as an engine built for a specific task, but QuickDraw 3D can come very, very close. I believe that QuickDraw 3D can come to 90-95% the speed of a custom 3D engine in most cases. I’ve thought about writing my own 3D engine for several years now, but as QuickDraw 3D has gotten better and better, I’ve found that it’s simply not worth the expense of writing such an engine when QuickDraw 3D does everything I’d ever need and it only costs me a small percentage of relative speed.

This document is not meant to be a 3D tutorial, nor a tutorial on QuickDraw 3D. I am going to assume that you already have a basic understanding of how 3D and QuickDraw 3D work. This will save hundreds of pages which would only duplicate information found in dozens of other 3D programming books found at any book store. Instead, this book will teach you how to code QuickDraw 3D such that you get the maximum possible performance out of your applications. It will also show you how to perform various 3D tasks such as calculating splines, doing rudimentary collision detection, and creating QuickTime movies with 3D tracks. By the time you finish reading this documentation you will know all there is to know about writing super-fast QuickDraw 3D applications.