
Contents

Acknowledgments	xiii
Introduction	xv
1 Cartesian Coordinate Systems	1
1.1 1D Mathematics	2
1.2 2D Cartesian Space	5
1.3 3D Cartesian Space	12
1.4 Odds and Ends	19
1.5 Exercises	27
2 Vectors	31
2.1 Mathematical Definition of Vector, and Other Boring Stuff	32
2.2 Geometric Definition of Vector	34
2.3 Specifying Vectors with Cartesian Coordinates	36
2.4 Vectors versus Points	39
2.5 Negating a Vector	43
2.6 Vector Multiplication by a Scalar	45
2.7 Vector Addition and Subtraction	47
2.8 Vector Magnitude (Length)	51
2.9 Unit Vectors	53
2.10 The Distance Formula	55
2.11 Vector Dot Product	56
2.12 Vector Cross Product	66
2.13 Linear Algebra Identities	70
2.14 Exercises	71
3 Multiple Coordinate Spaces	79
3.1 Why Bother with Multiple Coordinate Spaces?	80
3.2 Some Useful Coordinate Spaces	81
3.3 Basis Vectors and Coordinate Space Transformations	86
3.4 Nested Coordinate Spaces	106

3.5	In Defense of Upright Space	108
3.6	Exercises	109
4	Introduction to Matrices	113
4.1	Mathematical Definition of Matrix	113
4.2	Geometric Interpretation of Matrix	124
4.3	The Bigger Picture of Linear Algebra	130
4.4	Exercises	132
5	Matrices and Linear Transformations	137
5.1	Rotation	138
5.2	Scale	144
5.3	Orthographic Projection	148
5.4	Reflection	151
5.5	Shearing	152
5.6	Combining Transformations	153
5.7	Classes of Transformations	154
5.8	Exercises	159
6	More on Matrices	161
6.1	Determinant of a Matrix	161
6.2	Inverse of a Matrix	168
6.3	Orthogonal Matrices	171
6.4	4×4 Homogeneous Matrices	176
6.5	4×4 Matrices and Perspective Projection	183
6.6	Exercises	189
7	Polar Coordinate Systems	191
7.1	2D Polar Space	191
7.2	Why Would Anybody Use Polar Coordinates?	201
7.3	3D Polar Space	203
7.4	Using Polar Coordinates to Specify Vectors	213
7.5	Exercises	214
8	Rotation in Three Dimensions	217
8.1	What Exactly is “Orientation”?	218
8.2	Matrix Form	220
8.3	Euler Angles	229
8.4	Axis-Angle and Exponential Map Representations	244
8.5	Quaternions	246
8.6	Comparison of Methods	273
8.7	Converting between Representations	275
8.8	Exercises	291

9	Geometric Primitives	295
9.1	Representation Techniques	295
9.2	Lines and Rays	297
9.3	Spheres and Circles	303
9.4	Bounding Boxes	304
9.5	Planes	311
9.6	Triangles	317
9.7	Polygons	332
9.8	Exercises	339
10	Mathematical Topics from 3D Graphics	343
10.1	How Graphics Works	345
10.2	Viewing in 3D	362
10.3	Coordinate Spaces	369
10.4	Polygon Meshes	381
10.5	Texture Mapping	393
10.6	The Standard Local Lighting Model	396
10.7	Light Sources	414
10.8	Skeletal Animation	424
10.9	Bump Mapping	431
10.10	The Real-Time Graphics Pipeline	438
10.11	Some HLSL Examples	457
10.12	Further Reading	475
10.13	Exercises	476
11	Mechanics 1: Linear Kinematics and Calculus	479
11.1	Overview and Other Expectation-Reducing Remarks	479
11.2	Basic Quantities and Units	483
11.3	Average Velocity	486
11.4	Instantaneous Velocity and the Derivative	490
11.5	Acceleration	513
11.6	Motion under Constant Acceleration	516
11.7	The Integral	530
11.8	Uniform Circular Motion	542
11.9	Exercises	549
12	Mechanics 2: Linear and Rotational Dynamics	553
12.1	Newton's Three Laws	554
12.2	Some Simple Force Laws	562
12.3	Momentum	581
12.4	Impulsive Forces and Collisions	590
12.5	Rotational Dynamics	603
12.6	Real-Time Rigid Body Simulators	621

12.7	Suggested Reading	639
12.8	Exercises	640
13	Curves in 3D	645
13.1	Parametric Polynomial Curves	646
13.2	Polynomial Interpolation	653
13.3	Hermite Curves	665
13.4	Bézier Curves	670
13.5	Subdivision	685
13.6	Splines	690
13.7	Hermite and Bézier Splines	694
13.8	Continuity	697
13.9	Automatic Tangent Control	702
13.10	Exercises	711
14	Afterword	715
14.1	What Next?	715
14.2	Exercises	715
A	Geometric Tests	717
A.1	Closest Point on 2D Implicit Line	717
A.2	Closest Point on a Parametric Ray	718
A.3	Closest Point on a Plane	719
A.4	Closest Point on a Circle or Sphere	719
A.5	Closest Point in an AABB	720
A.6	Intersection Tests	720
A.7	Intersection of Two Implicit Lines in 2D	721
A.8	Intersection of Two Rays in 3D	722
A.9	Intersection of a Ray and Plane	724
A.10	Intersection of an AABB and Plane	725
A.11	Intersection of Three Planes	726
A.12	Intersection of Ray and a Circle or Sphere	727
A.13	Intersection of Two Circles or Spheres	729
A.14	Intersection of a Sphere and AABB	731
A.15	Intersection of a Sphere and a Plane	732
A.16	Intersection of a Ray and a Triangle	734
A.17	Intersection of Two AABBs	737
A.18	Intersection of a Ray and an AABB	740
B	Answers to the Exercises	745
B.1	Chapter 1	745
B.2	Chapter 2	746
B.3	Chapter 3	758

B.4	Chapter 4	759
B.5	Chapter 5	763
B.6	Chapter 6	765
B.7	Chapter 7	767
B.8	Chapter 8	772
B.9	Chapter 9	774
B.10	Chapter 10	779
B.11	Chapter 11	781
B.12	Chapter 12	784
B.13	Chapter 13	792
B.14	Chapter 14	799
	Bibliography	801
	Index	807

So much time, and so little to do!
Strike that, reverse it.
— Willy Wonka



Acknowledgments

Fletcher would like to thank his wife, A'me, who endured the absolute *eternity* that it took to produce this book, and his general tendency to generate lots of interesting ideas for large-scale projects that are initiated and then dropped a quarter of the way through. (No more gigantic projects for at least two or three weeks, I promise!)

Ian would like to thank his wife and children for not whining too loudly, and Fletcher for putting up with his procrastination. He would also like to thank Douglas Adams for the herring sandwich scoop, the bowl of petunias, and countless other references to the *Hitchhiker's Guide to the Galaxy* trilogy that you will find in this book.

Mike Pratcher gets a very huge thanks for his detailed and knowledgeable critique, and for writing a very large portion of the exercises. Matt Carter made the robot and kitchen and agreed to numerous requests to pose the robot one way or another. Thanks to Glenn Gamble for the dead sheep. Eric Huang created the cover illustration and all other 2D artwork that required any artistic talent whatsoever. (The authors made the rest.) Pavel Krajcevski provided helpful criticism.

Gratitude is merely the secret hope of further favors.

— Francois de La Rochefoucauld (1613–1680)

Always look and smell your best.

— Riley Dunn (1945–)



Introduction

First things first, but not necessarily in that order.

— Doctor Who from *Meglos* (1980)

Who Should Read This Book

This book is about 3D math, the geometry and algebra of 3D space. It is designed to teach you how to describe objects and their positions, orientations, and trajectories in 3D using mathematics. This is not a book about computer graphics, simulation, or even computational geometry, although if you plan on studying those subjects, you will definitely need the information here.

This is not just a book for video game programmers. We do assume that a majority of our readers are learning for the purpose of programming video games, but we expect a wider audience and we have designed the book with a diverse audience in mind. If you're a programmer or interested in learning how to make video games, welcome aboard! If you meet neither of these criteria, there's still plenty for you here. We have made every effort to make the book useful to designers and technical artists. Although there are several code snippets in the book, they are (hopefully) easy to read even for nonprogrammers. Most important, even though it is always necessary to understand the surrounding concepts to make sense of the code, the reverse is never true. We use code samples to illustrate how ideas can be implemented on a computer, not to explain the ideas themselves.

The title of this book says it is for “game development,” but a great deal of the material that we cover is applicable outside of video games. Practically anyone who wants to simulate, render, or understand a three-dimensional world will find this book useful. While we do try to provide motivating examples from the world of video game development, since that is our area of expertise and also our primary target audience, you won't be left out if the last game you completed was *Space Quest*.¹ If your interests

¹Well, you may be left out of a few jokes, like that one. Sorry.

lie in more “grown up” things than video games, rest assured that this book is *not* filled with specific examples from video games about head-shots or severed limbs or how to get the blood spurt to look just right.

Why You Should Read This Book

This book has many unique features, including its topic, approach, authors, and writing style.

Unique topic. This book fills a gap that has been left by other books on graphics, linear algebra, simulation, and programming. It’s an introductory book, meaning we have focused our efforts on providing thorough coverage on fundamental 3D concepts—topics that are normally glossed over in a few quick pages or relegated to an appendix in other books (because, after all, you already know all this stuff). We have found that these very topics are often the sticking points for beginners! In a way, this book is the mirror image of gluing together books on graphics, physics, and curves. Whereas that mythical conglomeration would begin with a brief overview of the mathematical fundamentals, followed by in-depth coverage of the application area, we start with a thorough coverage of the math fundamentals, and then give compact, high-level overviews of the application areas.

This book does try to provide a graceful on-ramp for beginners, but that doesn’t mean we’ll be stuck in the slow lane forever. There is plenty of material here that is traditionally considered “advanced” and taught in upper-level or graduate classes. In reality, these topics are specialized more than they are difficult, and they have recently become important prerequisites that need to be taught earlier, which is part of what has driven the demand for a book like this.

Unique approach. All authors think that they strike the perfect balance between being pedantic and being chatty in order to best reach their audience, and we are no exception. We recognize, however, that the people who disagree with this glowing self-assessment will mostly find this book too informal (see the index entry for “stickler alert”). We have focused on perspicuous explanations and intuition, and sometimes we have done this at the expense of rigor. Our aim is to simplify, but not to oversimplify. We lead readers to the goal through a path that avoids the trolls and dragons, so why begin the journey by pointing them all out before we’ve even said what our destination is or why we’re going there? However, since we know readers will be crossing the field on their own eventually, after we reach our goal we will turn around to point out where the dangers lie. But we may sometimes need to leave certain troll-slaying to another source, especially if

we expect that your usual path won't take you near the danger. Those who intend to be on that land frequently should consult with a local for more intimate knowledge. This is not to say that we think rigor is unimportant; we just think it's easier to get rigor after intuition about the big picture has been established, rather than front-loading every discussion with definitions and axioms needed to handle the edge cases. Frankly, nowadays a reader can pursue concise and formal presentations *free* on wikipedia.org or Wolfram MathWorld (mathworld.wolfram.com), so we don't think any book offers much worth paying for by dwelling excessively on definitions, axioms, proofs, and edge cases, especially for introductory material targeted primarily to engineers.

Unique authors. Our combined experience brings together academic authority with in-the-trenches practical advice. Fletcher Dunn has 15 years of professional game programming experience, with around a dozen titles under his belt on a variety of gaming platforms. He worked at Terminal Reality in Dallas, where as principal programmer he was one of the architects of the Infernal engine and lead programmer on *BloodRayne*. He was a technical director for The Walt Disney Company at Wideload Games in Chicago and the lead programmer for *Disney Guilty Party*, IGN's E3 2010 Family Game of the Year. He now works for Valve Software in Bellevue, Washington. But his biggest claim to fame by far is as the namesake of Corporal Dunn from *Call of Duty: Modern Warfare 2*.

Dr. Ian Parberry has more than a quarter century of experience in research and teaching in academia. This is his sixth book, his third on game programming. He is currently a tenured full professor in the Department of Computer Science & Engineering at the University of North Texas. He is nationally known as one of the pioneers of game programming in higher education, and has been teaching game programming classes at the University of North Texas continuously since 1993.

Unique writing style. We hope you will enjoy reading this math book (say *what?*) for two reasons. Most important, we want you to *learn* from this book, and learning something you are interested in is fun. Secondly, we want you to enjoy *reading* this book in the same way that you enjoy reading a work of literature. We have no delusions that we're in the same class as Mark Twain, or that this book is destined to become a classic like, say, *The Hitchhikers Guide to the Galaxy*. But one can always have aspirations. Honestly, we are just silly people. At the same time, no writing style should stand in the way of the first priority: clear communication of mathematical knowledge about video games.²

²Which is why we've put most of the jokes and useless trivia in footnotes like this. Somehow, we felt like we could get away with more that way.

What You Should Know before Reading This Book

We have tried to make the book accessible to as wide an audience as possible; no book, however, can go back all the way to first principles. We expect from the reader the following basic mathematical skills:

- Manipulating algebraic expressions, fractions, and basic algebraic laws such as the associative and distributive laws and the quadratic equation.
- Understanding what variables are, what a function is, how to graph a function, and so on.
- Some very basic 2D Euclidian geometry, such as what a point is, what a line is, what it means for lines to be parallel and perpendicular, and so forth. Some basic formulas for area and circumference are used in a few places. It's OK if you have temporarily forgotten those—you will hopefully recognize them when you see them.
- Some prior exposure to trigonometry is best. We give a brief review of trigonometry in the front of this book, but it is not presented with the same level of paced explanation found most elsewhere in this book.
- Readers with some prior exposure to calculus will have an advantage, but we have restricted our use of calculus in this book to very basic principles, which we will (attempt to) teach in Chapter 11 for those without this training. Only the most high-level concepts and fundamental laws are needed.

Some programming knowledge is helpful, but *not* required. In several places, we give brief code snippets to show how the ideas being discussed get translated into code. (Also certain procedures are just easier to explain in code.) These snippets are extremely basic, well commented, and require only the most rudimentary understanding of C language syntax (which has been copied to several other languages). Most technical artists or level designers should be able to interpret these snippets with ease.

Overview

- *Chapter 1* gets warmed up with some groundwork that it is needed in the rest of the book and which you probably already know. It reviews the Cartesian coordinate system in 2D and 3D and discusses how to use the Cartesian coordinate system to locate points in space. Also included is a very quick refresher on trigonometry and summation notation.

- *Chapter 2* introduces vectors from a mathematical and geometric perspective and investigates the important relationship between points and vectors. It also discusses a number of vector operations, how to do them, what it means geometrically to do them, and situations for which you might find them useful.
- *Chapter 3* discusses examples of coordinate spaces and how they are nested in a hierarchy. It also introduces the central concepts of basis vectors and coordinate-space transformations.
- *Chapter 4* introduces matrices from a mathematical and geometric perspective and shows how matrices are a compact notation for the math behind linear transformations.
- *Chapter 5* surveys different types of linear transformations and their corresponding matrices in detail. It also discusses various ways to classify transformations.
- *Chapter 6* covers a few more interesting and useful properties of matrices, such as affine transforms and perspective projection, and explains the purpose and workings of four-dimensional vectors and matrices within a three-dimensional world.
- *Chapter 7* discusses how to use polar coordinates in 2D and 3D, why it is useful to do so, and how to convert between polar and Cartesian representations.
- *Chapter 8* discusses different techniques for representing orientation and angular displacement in 3D: Euler angles, rotation matrices, exponential maps, and quaternions. For each method, it explains how the method works and presents the advantages and disadvantages of the method and when its use is recommended. It also shows how to convert between different representations.
- *Chapter 9* surveys a number of commonly used geometric primitives and discusses how to represent and manipulate them mathematically.
- *Chapter 10* is a whirlwind lesson on graphics, touching on a few selected theoretical as well as modern practical issues. First, it presents a high-level overview of “how graphics works,” leading up to the rendering equation. The chapter then walks through a few theoretical topics of a mathematical nature. Next it discusses two contemporary topics that are often sources of mathematical difficulty and should be of particular interest to the reader: skeletal animation and bump mapping. Finally, the chapter presents an overview of the real-time graphics pipeline, demonstrating how the theories from the first half

of the chapter are implemented in the context of current rendering hardware.

- *Chapter 11* crams two rather large topics into one chapter. It interleaves the highest-level topics from first-semester calculus with a discussion of rigid body kinematics—how to describe and analyze the motion of a rigid body without necessarily understanding its cause or being concerned with orientation or rotation.
- *Chapter 12* continues the discussion of rigid body mechanics. It starts with a condensed explanation of classical mechanics, including Newton’s laws of motion and basic concepts such as inertia, mass, force, and momentum. It reviews a few basic force laws, such as gravity, springs, and friction. The chapter also considers the rotational analogs of all of the linear ideas discussed up to this point. Due attention is paid to the important topic of collisions. The chapter ends with a discussion of issues that arise when using a computer to simulate rigid bodies.
- *Chapter 13* explains parametric curves in 3D. The first half of the chapter explains how a relatively short curve is represented in some common, important forms: monomial, Bézier, and Hermite. The second half is concerned with fitting together these shorter pieces into a longer curve, called a *spline*. In understanding each system, the chapter considers what controls the system presents to a designer of curves, how to take a description of a curve made by a designer and recreate the curve, and how these controls can be used to construct a curve with specific properties.
- *Chapter 14* inspires the reader to pursue greatness in video games.
- *Appendix A* is an assortment of useful tests that can be performed on geometric primitives. We intend it to be a helpful reference, but it can also make for interesting browsing.
- *Appendix B* has all the answers.³

Find a Bug in This Book?

We calculated the odds that we could write an 800+ page math book free of mistakes. The result was a negative number, which we know can’t be right, but is probably pretty close. If you find a bug in this book, please

³To the exercises, that is.

visit gamemath.com. Most likely, the error is already listed in the errata, in which case you have our profound apologies. Otherwise, send us an email, and you will have (in addition to our profound thanks) everlasting fame via credit in the errata for being the first to find the mistake.

Careful. We don't want to learn from this.
— Bill Watterson (1958–) from *Calvin and Hobbes*