

Introduction to 3D Game Programming with DirectX 9.0c: A Shader Approach

Introduction

Part I Mathematical Prerequisites

Chapter 1 Vector Algebra

1.1 Vectors

1.1.1 Vectors and Coordinate Systems

1.1.2 Left-Handed Versus Right-Handed Coordinate Systems

1.1.3 Basic Vector Operations

1.2 Length and Unit Vectors

1.3 The Dot Product

1.4 The Cross Product

1.5 Change of Frame

1.5.1 Vectors

1.5.2 Points

1.6 Rays, Lines, and Segments

1.7 D3DX Vectors

1.8 Summary

1.9 Exercises

Chapter 2 Matrix Algebra

2.1 Definition

2.2 Matrix Multiplication

2.2.1 Vector-Matrix Multiplication

2.2.2 Matrix-Matrix Multiplication and Associativity

2.3 The Transpose of a Matrix

2.4 The Identity Matrix

2.5 The Inverse of a Matrix

2.6 D3DX Matrices

2.7 Summary

2.8 Exercises

Chapter 3 Transformations; Planes

3.1 Linear Transformations

3.1.1 Definition

3.1.2 Matrix Representation

3.1.3 Scaling

3.1.4 Rotation about the Coordinate Axes

3.1.5 Rotation About an Arbitrary Axis

3.2 Affine Transformations

3.2.1 Definition and Matrix Representation

3.2.2 Translation

- 3.3.3 Affine Matrices for Scaling and Rotation
- 3.3 D3DX Transformation Functions
- 3.4 Composition of Affine Transformations
- 3.5 Planes
 - 3.5.1 D3DXPlane
 - 3.5.2 Point and Plane Spatial Relation
 - 3.5.3 Construction
 - 3.5.4 Normalizing a Plane
 - 3.5.5 Transforming a Plane
 - 3.5.6 Nearest Point on a Plane to a Particular Point
 - 3.5.7 Ray/Plane Intersection
- 3.6 Summary
- 3.7 Exercises

Part II Direct3D Foundations

Chapter 4 Direct3D Initialization

- 4.1 Direct3D Overview
 - 4.1.1 Devices
 - 4.1.2 COM
- 4.2 Some Preliminaries
 - 4.2.1 Surfaces
 - 4.2.2 The Swap Chain and Page Flipping
 - 4.2.3 Pixel Formats
 - 4.2.4 Display Adapters
 - 4.2.5 Depth Buffers
 - 4.2.6 Multisampling
 - 4.2.7 Memory Pools
 - 4.2.8 Vertex Processing and Pure Devices
 - 4.2.9 Device Capabilities
- 4.3 Initializing Direct3D
 - 4.3.1 Acquiring an IDirect3D9 Interface
 - 4.3.2 Verifying HAL Support
 - 4.3.3 Checking for Hardware Vertex Processing
 - 4.3.4 D3DPRESENT_PARAMETERS
 - 4.3.5 Creating the IDirect3DDevice9 Interface
- 4.4 Lost Devices
- 4.5 The Demo Application Framework
 - 4.5.1 D3DApp
 - 4.5.2 Non-Framework Methods
 - 4.5.3 Framework Methods
 - 4.5.4 The Message Handler: `msgProc`
 - 4.5.5 Switching to Full Screen Mode and Back
- 4.6 Demo Application: Hello Direct3D
- 4.7 Debugging Direct3D Application Tips
- 4.8 Summary

4.9 Exercises

Chapter 5 Timing; Direct Input; Animation and Sprites

5.1 The Performance Timer

5.1.1 Time Differential Between Frames

5.1.2 Frames Per Second Calculation

5.1.3 Graphics Stats Demo

5.2 Direct Input Primer

5.2.1 Interfaces

5.2.2 Initializing the Keyboard and Mouse

5.2.3 Cleanup

5.2.4 Polling the Keyboard and Mouse

5.3 Sprites and Animation

5.3.1 ID3DXSprite

5.3.2 The Sprite Demo

5.3.2.1 Bullet Structure

5.3.2.2 Demo Application Class Data Members

5.3.2.3 SpriteDemo

5.3.2.4 ~SpriteDemo

5.3.2.5 checkDeviceCaps

5.3.2.6 onLostDevice

5.3.2.7 onResetDevice

5.3.2.8 updateScene

5.3.2.9 updateShip

5.3.2.10 updateBullets

5.3.2.11 drawScene

5.3.2.12 drawBkgd

5.3.2.13 drawShip

5.3.2.14 drawBullets

5.3.3 Page Flipping Animation

5.4 Summary

5.5 Exercises

Chapter 6 The Rendering Pipeline

6.1 The 3D Illusion

6.2 Model Representation

6.2.1 Vertex Declarations

6.2.2 Triangles

6.2.3 Indices

6.3 The Virtual Camera

6.3.1 The Frustum

6.3.2 Perspective Projection

6.4 The Rendering Pipeline

6.4.1 Local Space and World Space

6.4.2 View Space

6.4.3 Lighting

- 6.4.4 Projection Transformation
 - 6.4.4.1 Defining a Frustum; Aspect Ratio
 - 6.4.4.2 Projecting Vertices
 - 6.4.4.3 Normalizing the Coordinates
 - 6.4.4.4 Transforming the z-Coordinate
 - 6.4.4.5 Writing the Projection Equations with a Matrix
 - 6.4.4.6 `D3DXMatrixPerspectiveFovLH`
- 6.4.5 Backface Culling
- 6.4.6 Clipping
- 6.4.7 Viewport Transform
- 6.4.8 Rasterization
- 6.5 Summary
- 6.6 Exercises

Chapter 7 Drawing in Direct3D Part I

- 7.1 Vertex/Index Buffers
 - 7.1.1 Creating a Vertex and Index Buffer
 - 7.1.2 Accessing a Buffer's Memory
 - 7.1.3 Vertex and Index Buffer Info
- 7.2 Drawing Methods
 - 7.2.1 `DrawPrimitive`
 - 7.2.2 `DrawIndexedPrimitive`
- 7.3 Drawing Preparations
 - 7.3.1 Vertex Streams
 - 7.3.2 Setting Indices
 - 7.3.3 Setting the Vertex Declaration
- 7.4 Cube Demo
 - 7.4.1 Vertex Structure
 - 7.4.2 `CubeDemo` Class Data Members
 - 7.4.3 Construction
 - 7.4.4 Destruction
 - 7.4.5 `onLostDevice` / `onResetDevice`
 - 7.4.6 `updateScene`
 - 7.4.7 `drawScene`
- 7.5 Summary
- 7.6 Exercises

Chapter 8 Drawing in Direct3D Part II

- 8.1 Checking for Shader Support
- 8.2 Shaders; the FX Framework
 - 8.2.1 A Simple Vertex Shader
 - 8.2.2 A Simple Pixel Shader
 - 8.2.3 A Simple FX File
 - 8.2.4 Creating an Effect
 - 8.2.5 Setting Effect Parameters
- 8.3 Applying the Effect

- 8.3.1 Obtaining a Handle to an Effect
- 8.3.2 Activating an Effect and Setting Effect Parameters
- 8.3.3 Beginning an Effect
- 8.3.4 Setting the Current Rendering Pass
- 8.3.5 Ending an Effect
- 8.3.6 Example Code
- 8.4 Triangle Grid Demo
 - 8.4.1 Vertex Generation
 - 8.4.2 Index Generation
 - 8.4.3 Extracting the Grid Geometry
- 8.5 D3DX Geometric Objects
- 8.6 Summary
- 8.7 Exercises

Chapter 9 Color

- 9.1 Color Representation
 - 9.1.1 D3DCOLOR
 - 9.1.2 D3DCOLORVALUE
 - 9.1.3 D3DXCOLOR
- 9.2 Vertex Colors
- 9.3 Colored Cube Demo
- 9.4 Digression: Traveling Sine Waves
 - 9.4.1 Summing Waves
 - 9.4.2 Circular Waves
 - 9.4.3 Directional Waves
- 9.5 Colored Waves Demo
- 9.6 Summary
- 9.7 Exercises

Chapter 10 Lighting

- 10.1 Light and Material Interaction
- 10.2 Diffuse Lighting
 - 10.2.1 Normal Vectors
 - 10.2.1.1 Transforming Normal Vectors
 - 10.2.2 Lambert's Cosine Law
 - 10.2.3 Diffuse Lighting
 - 10.2.4 Diffuse Demo
 - 10.2.4.1 Vertex Structure
 - 10.2.4.2 Effect Parameters
 - 10.2.4.3 The Vertex Shader
 - 10.2.4.4 The Pixel Shader and Technique
- 10.3 Ambient Lighting
- 10.4 Specular Lighting
- 10.5 Point Lights
- 10.6 Spotlights
- 10.7 Attenuation

- 10.8 The Point Light Demo
 - 10.8.1 Grid Normals
 - 10.8.2 Animated Light
 - 10.8.3 Different Materials
 - 10.8.4 The Vertex Shader
- 10.9 The Spotlight Demo
- 10.10 Phong Shading
- 10.11 Summary
- 10.12 Exercises

Chapter 11 Texturing

- 11.1 Texture Coordinates
- 11.2 Creating and Enabling a Texture
- 11.3 Filters
- 11.4 Mipmaps
 - 11.4.1 Mipmap Filter
 - 11.4.2 Using Mipmaps with Direct3D
 - 11.4.3 Hardware Generated Mipmaps
- 11.5 Textured Cube Demo
 - 11.5.1 Specifying the Texture Coordinates
 - 11.5.2 Creating the Texture
 - 11.5.3 Setting and Sampling the Texture
- 11.6 Address Modes
- 11.7 Tiled Ground Demo
- 11.8 Multi-Texturing
 - 11.8.1 Generating Texture Coordinates
 - 11.8.2 Creating and Enabling the Textures
 - 11.8.3 Sampler Objects
 - 11.8.4 The Vertex and Pixel Shader
- 11.9 Spherical and Cylindrical Texturing
 - 11.9.1 Sphere Mapping
 - 11.9.2 Cylindrical Mapping
 - 11.9.3 Texturing Wrapping
 - 11.9.4 Spherical and Cylindrical Texturing Demo
- 11.10 Texture Animation
- 11.11 Compressed Textures and the *DXTex* Tool
- 11.12 Summary
- 11.13 Exercises

Chapter 12 Blending

- 12.1 The Blending Equation
- 12.2 Blend Factors
 - 12.2.1 Blend Factor Example 1
 - 12.2.2 Blend Factor Example 2
 - 12.2.3 Blend Factor Example 3
 - 12.2.4 Blend Factor Example 4

- 12.3 Transparent Teapot Demo
- 12.4 Transparent Teapot Demo with Texture Alpha Channel
- 12.5 The Alpha Test
- 12.6 Summary
- 12.7 Exercises

Chapter 13 Stenciling

- 13.1 Using the Stencil Buffer
 - 13.1.1 Requesting a Stencil Buffer
 - 13.1.2 The Stencil Test
 - 13.1.3 Controlling the Stencil Test
 - 13.1.3.1 Stencil Reference Value
 - 13.1.3.2 Stencil Mask
 - 13.1.3.3 Stencil Value
 - 13.1.3.4 Comparison Operation
 - 13.1.4 Updating the Stencil Buffer
 - 13.1.5 Stencil Write Mask
- 13.2 Mirror Demo
 - 13.2.1 The Mathematics of Reflection
 - 13.2.2 Mirror Implementation Overview
 - 13.2.3 Code and Explanation
 - 13.2.3.1 Part I
 - 13.2.3.2 Part II
 - 13.2.3.3 Part III
 - 13.2.3.4 Part IV
 - 13.2.3.5 Part V
- 13.3 Sample Application: Planar Shadows
 - 13.3.1 Parallel Light Shadows
 - 13.3.2 Point Light Shadows
 - 13.3.3 The Shadow Matrix
 - 13.3.4 Using the Stencil Buffer to Prevent Double Blending
 - 13.3.5 Code and Explanation
- 13.4 Summary
- 13.5 Exercises

Part III Applied Direct3D and the D3DX Library

Chapter 14 Meshes

- 14.1 Geometry Info
- 14.2 Subsets and the Attribute Buffer
- 14.3 Drawing
- 14.4 Adjacency Info
- 14.5 Optimizing
- 14.6 The Attribute Table
- 14.7 Cloning
- 14.8 Creating a Mesh (D3DXCreateMesh)

- 14.9 .X Files
 - 14.9.1 Loading a .X File
 - 14.9.2 Testing for Vertex Normals
 - 14.9.3 Changing the Vertex Format
 - 14.9.4 .X File Materials
 - 14.9.5 Optimizing
 - 14.9.6 .X File Materials
 - 14.9.7 The .X File Demo
- 14.10 Bounding Volumes
 - 14.10.1 Some New Special Constants
 - 14.10.2 Bounding Volume Types
 - 14.10.3 Bounding Box Demo
- 14.11 Survey of Other D3DX Mesh Functions
 - 14.11.1 D3DXSplitMesh
 - 14.11.2 D3DXConcatenateMeshes
 - 14.11.3 D3DXValidMesh
 - 14.11.4 D3DXCleanMesh
 - 14.11.5 D3DXWeldVertices
 - 14.11.6 D3DXSimplifyMesh
 - 14.11.7 D3DXGeneratePMesh
- 14.12 Summary
- 14.13 Exercises

Chapter 15 Mesh Hierarchy Animation Part I: Rigid Meshes

- 15.1 Robot Arm Demo
 - 15.1.1 Mathematical Formulation
 - 15.1.2 Implementation
 - 15.1.2.1 Bone Mesh
 - 15.1.2.2 Bone Data Structure
 - 15.1.2.3 Building the Bone World Matrices
 - 15.1.2.4 Animating and Rendering the Bones
- 15.2 Solar System Demo
 - 15.2.1 Solar Object Data Structure
 - 15.2.2 Building the Solar Object World Matrices
 - 15.2.3 Animating the Solar System
- 15.3 Keyframes and Animation
- 15.4 Summary
- 15.5 Exercises

Chapter 16 Mesh Hierarchy Animation Part II: Skinned Meshes

- 16.1 Overview of Skinned Meshes
 - 16.1.1 Definitions
 - 16.1.2 Reformulating a Bones To-Root Transform
 - 16.1.3 The Offset Transform
 - 16.1.4 Vertex Blending
 - 16.1.5 D3DXFRAME

- 16.2 Skinned Mesh Demo
 - 16.2.1 SkinnedMesh Overview
 - 16.2.2 D3DXMESHCONTAINER
 - 16.2.3 ID3DXAnimationController
 - 16.2.4 ID3DXAllocateHierarchy
 - 16.2.5 D3DXLoadMeshHierarchyFromX and
D3DXFrameDestroy
 - 16.2.6 Finding the One and Only Mesh
 - 16.2.7 Converting to a Skinned Mesh
 - 16.2.8 Building the To-Root Transform Matrix Array
 - 16.2.9 Initialization Summarized
 - 16.2.10 Animating the Skinned Mesh
- 16.3 Summary
- 16.4 Exercises

Chapter 17 Terrain Rendering Part I

- 17.1 Heightmaps
 - 17.1.1 Creating a Heightmap
 - 17.1.2 Heightmap Class Overview
 - 17.1.3 Loading a RAW File
 - 17.1.4 Filtering
- 17.2 Basic Terrain Demo
 - 17.2.1 Building the Terrain Geometry
 - 17.2.2 Lighting and Texturing the Terrain
 - 17.2.3 The Vertex and Pixel Shaders
- 17.3 Multi-Sub-Grid Terrain
- 17.4 Building a Flexible Camera
 - 17.4.1 View Transformation Recapitulation
 - 17.4.2 Camera Functionality
 - 17.4.3 The Camera Class
 - 17.4.4 Updating the Camera
 - 17.4.5 Building the View Matrix
 - 17.4.6 Camera Demo Comments
- 17.5 “Walking” on the Terrain
 - 16.5.1 Getting the Terrain Height
 - 16.5.2 Moving Tangent to the Terrain
- 17.6 Summary
- 17.7 Exercises

Chapter 18 Terrain Rendering Part II

- 18.1 Sub-Grid Culling and Sorting
 - 18.1.1 The SubGrid Structure
 - 18.1.2 Extracting Frustum Planes
 - 18.1.3 Frustum/AABB Intersection Test
 - 18.1.4 Experiments
- 18.2 Trees; Castle

- 18.3 Fog
- 18.4 Grass
 - 18.4.1 The Billboard Matrix
 - 18.4.2 Animating the Grass
 - 18.4.3 The Grass Vertex Structure
 - 18.4.4 Building a Grass Fin
 - 18.4.5 Grass Effect
- 18.5 Water
- 18.6 Summary
- 18.7 Exercises

Chapter 19 Particle Systems

- 19.1 Particles and Point Sprites
 - 19.1.1 Using Point Sprites
 - 19.1.2 Particle Motion
 - 19.1.3 Randomness
 - 19.1.4 Structure Format
 - 19.1.5 Render States
- 19.2 Particle System Framework
 - 19.2.1 Selected `PSystem` Data Members
 - 19.2.2 Selected `PSystem` Methods
- 19.3 Example 1: Fire Ring
 - 19.3.1 Initializing the Particles
 - 19.3.2 The Fire Ring Effect
- 19.4 Example 2: Rain
 - 19.4.1 Initializing the Particles
 - 19.4.2 The Rain Effect
- 19.5 Example 3: Sprinkler
 - 19.5.1 Initializing the Particles
 - 19.5.2 The Sprinkler Effect
- 19.6 Example 4: Bolt Gun
 - 19.6.1 Initializing the Particles
 - 19.6.2 The Gun Effect
- 19.7 Summary
- 19.8 Exercises

Chapter 20 Picking

- 20.1 Screen to Projection Window Transform
- 20.2 World Space Picking Ray
- 20.3 Ray/Object Intersection Tests
- 20.4 Tri-Pick Demo
- 20.5 Asteroids Demo
- 20.6 Summary
- 20.7 Exercises

Chapter 21 Advanced Texturing Part I

- 21.1 Cube Mapping
 - 21.1.1 Environment Maps
 - 21.1.2 Loading and Using Cube Maps in Direct3D
 - 21.1.3 Environment Map Demo
 - 21.1.3.1 Sky Sphere
 - 21.1.3.2 Reflections
- 21.2 Normal Mapping
 - 21.2.1 Storing Normal Maps in Textures
 - 21.2.2 Generating Normal Maps
 - 21.2.3 Using Normal Maps
 - 21.2.4 Implementation Details
 - 21.2.4.1 Computing the TBN-Frame Per Vertex
 - 21.2.4.2 Effect Parameters
 - 21.2.4.3 The Vertex Shader
 - 21.2.4.4 The Pixel Shader
 - 21.2.4.5 The Brick Demo
 - 21.2.5 Normal Mapping Water
- 21.3 Render to Texture
 - 21.3.1 D3DXCreateRenderToSurface
 - 21.3.2 D3DXCreateTexture
 - 21.3.3 IDirect3DTexture9::GetSurfaceLevel
 - 21.3.4 Drawing to the Surface/Texture
 - 21.3.5 DrawableTex2D
 - 21.3.6 Render To Texture Demo
- 21.4 Summary
- 21.5 Exercises

Chapter 22 Advanced Texturing Part II

- 22.1 Projective Texturing
 - 22.1.1 Generating Projective Texture Coordinates
 - 22.1.2 Projective Texture Coordinates Outside [0, 1]
 - 22.1.3 Sample Code
- 22.2 Shadow Mapping
 - 22.2.1 Checking for D3DFMT_R32F Support
 - 22.2.2 Building the Shadow Map
 - 22.2.3 The Shadow Map Test
 - 22.2.4 Filtering
- 22.3 Displacement Mapping
 - 22.3.1 Checking Device Capabilities
 - 22.3.2 Demo Overview
 - 22.3.3 tex2Dlod
 - 22.3.4 Filtering
 - 22.3.5 The Vertex Shader
- 22.4 Summary
- 22.5 Exercises

Appendix A Windows Programming
Appendix B HLSL Reference

Index