

HANSER

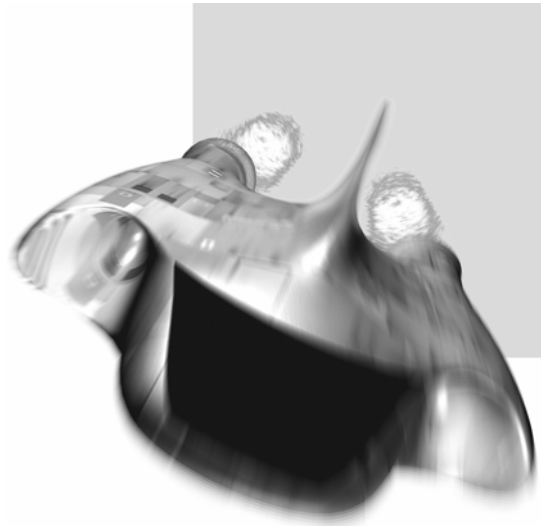
3D-Spieleprogrammierung mit DirectX 9 und C++

David Scherfgen

ISBN 3-446-40596-8

Inhaltsverzeichnis

Weitere Informationen oder Bestellungen unter
<http://www.hanser.de/3-446-40596-8> sowie im Buchhandel



Inhalt

| | | |
|----------|--|----------|
| 1 | Einleitung..... | 1 |
| 1.1 | Ein paar Dinge im Voraus..... | 2 |
| 1.1.1 | Was Sie erwartet..... | 2 |
| 1.1.2 | Was Sie nicht erwartet..... | 2 |
| 1.1.3 | Voraussetzungen..... | 3 |
| 1.1.4 | Die Schriftformate in diesem Buch..... | 3 |
| 1.1.5 | Was tun bei Problemen?..... | 3 |
| 1.1.6 | Zu DirectX 9.0 und DirectX 9.0c..... | 4 |
| 1.1.7 | Die ungarische Notation..... | 4 |
| 1.2 | Einführung in die Spieleprogrammierung..... | 5 |
| 1.2.1 | Der kleine Unterschied..... | 5 |
| 1.2.2 | Was macht ein Spiel eigentlich?..... | 6 |
| 1.2.3 | Eingliederung in die Windows-Architektur..... | 8 |
| 1.2.4 | Das Problem mit der Zeit..... | 9 |
| 1.2.5 | Die verschiedenen Seiten eines Spiels..... | 13 |
| 1.2.6 | Rückblick..... | 14 |
| 1.3 | DirectX und C++..... | 14 |
| 1.3.1 | Was ist DirectX?..... | 14 |
| 1.3.2 | Die perfekte Kombination mit C++..... | 16 |
| 1.3.3 | Das COM – Grundlage von DirectX..... | 18 |
| 1.3.4 | Rückblick..... | 21 |
| 1.4 | Wir bauen uns eine eigene Engine!..... | 21 |
| 1.4.1 | Was versteht man unter einer Engine?..... | 21 |
| 1.4.2 | Verschiedene Entwicklungsansätze..... | 21 |
| 1.4.3 | Konkrete Planung..... | 22 |
| 1.4.4 | Installation der Engine und Einrichten eines Projekts..... | 25 |

| | | |
|----------|---|-----------|
| 1.4.5 | Vorgabefunktionen, Klassen und Makros..... | 26 |
| 1.4.6 | Rückblick | 36 |
| 1.4.7 | Übungsaufgaben..... | 36 |
| 1.5 | Tipps zum Debuggen | 37 |
| 1.6 | Ausblick | 40 |
| 2 | 3D-Grafik | 41 |
| 2.1 | Was Sie in diesem Kapitel erwartet..... | 42 |
| 2.2 | Einführung in die 3D-Grafik | 42 |
| 2.2.1 | Ein neues Koordinatensystem..... | 42 |
| 2.2.2 | Theorie der 3D-Grafik | 44 |
| 2.2.3 | Vektoren | 46 |
| 2.2.4 | Matrizen | 65 |
| 2.2.5 | Ebenen..... | 88 |
| 2.2.6 | Das RGB-Farbsystem | 94 |
| 2.2.7 | Rückblick | 97 |
| 2.2.8 | Übungsaufgaben | 98 |
| 2.3 | Direct3D-Grundlagen..... | 98 |
| 2.3.1 | Was ist Direct3D? | 98 |
| 2.3.2 | Die Transformationspipeline | 99 |
| 2.3.3 | Der Rasterizer..... | 100 |
| 2.3.4 | Die wichtigsten Schnittstellen | 101 |
| 2.3.5 | Ressourcen | 101 |
| 2.3.6 | Oberflächen | 102 |
| 2.3.7 | Direct3D im C++-Programm ansprechen | 106 |
| 2.3.8 | Rückblick | 106 |
| 2.4 | Initialisierung von Direct3D..... | 107 |
| 2.4.1 | Erstellen der <i>IDirect3D9</i> -Schnittstelle..... | 107 |
| 2.4.2 | Adapterinformationen..... | 108 |
| 2.4.3 | Caps – die Fähigkeiten eines Geräts | 112 |
| 2.4.4 | Erstellen des Fensters | 113 |
| 2.4.5 | Erstellen der <i>IDirect3DDevice9</i> -Schnittstelle..... | 116 |
| 2.4.6 | Direct3D herunterfahren..... | 128 |
| 2.4.7 | Beispielprogramm: eine komplette Direct3D-Anwendung..... | 128 |
| 2.4.8 | Rückblick | 130 |
| 2.4.9 | Übungsaufgaben | 131 |
| 2.5 | Das erste Dreieck | 131 |
| 2.5.1 | Vertizes | 131 |
| 2.5.2 | Erste Render-States | 133 |
| 2.5.3 | Setup der Transformationspipeline | 137 |
| 2.5.4 | Der Zeichenvorgang | 139 |
| 2.5.5 | Rückblick | 145 |
| 2.5.6 | Übungsaufgaben | 146 |
| 2.6 | Texturen | 146 |
| 2.6.1 | Was Texturen sind..... | 146 |
| 2.6.2 | Grundlegende Dinge..... | 149 |
| 2.6.3 | Der bilineare Filter | 151 |

| | | |
|--------|---|-----|
| 2.6.4 | MIP-Mapping – schnell und schön | 153 |
| 2.6.5 | Weitere Spielereien | 155 |
| 2.6.6 | Texturen mit D3DX laden | 156 |
| 2.6.7 | Texturinformationen abfragen | 159 |
| 2.6.8 | Das Beispielprogramm | 160 |
| 2.6.9 | Kachelfähige Texturen erzeugen | 165 |
| 2.6.10 | Rückblick | 166 |
| 2.6.11 | Übungsaufgaben | 167 |
| 2.7 | Vertex- und Index-Buffer | 167 |
| 2.7.1 | Zweck von Vertex- und Index-Buffern | 167 |
| 2.7.2 | Der Vertex-Buffer im Detail | 168 |
| 2.7.3 | Der Index-Buffer im Detail | 172 |
| 2.7.4 | Das Beispielprogramm | 176 |
| 2.7.5 | Rückblick | 181 |
| 2.7.6 | Übungsaufgaben | 181 |
| 2.8 | Nebel | 182 |
| 2.8.1 | Die Theorie | 182 |
| 2.8.2 | Nebel mit Direct3D | 184 |
| 2.8.3 | Das Beispielprogramm | 186 |
| 2.8.4 | Rückblick | 187 |
| 2.8.5 | Übungsaufgaben | 188 |
| 2.9 | Beleuchtung | 188 |
| 2.9.1 | Ein einfaches Beleuchtungssystem | 188 |
| 2.9.2 | Die Praxis – Beleuchtung mit Direct3D | 195 |
| 2.9.3 | Das Beispielprogramm | 203 |
| 2.9.4 | Rückblick | 204 |
| 2.9.5 | Übungsaufgaben | 205 |
| 2.10 | Alpha-Blending | 206 |
| 2.10.1 | Die Theorie | 206 |
| 2.10.2 | Alpha-Blending mit Direct3D | 208 |
| 2.10.3 | Das Beispielprogramm | 211 |
| 2.10.4 | Rückblick | 212 |
| 2.10.5 | Übungsaufgaben | 213 |
| 2.11 | Multi-Texturing | 213 |
| 2.11.1 | Der theoretische Teil | 213 |
| 2.11.2 | Multi-Texturing anwenden | 217 |
| 2.11.3 | Mehr über Texturkoordinaten | 220 |
| 2.11.4 | Das Beispielprogramm | 223 |
| 2.11.5 | Rückblick | 225 |
| 2.11.6 | Übungsaufgaben | 225 |
| 2.12 | Exotische Texturformen | 226 |
| 2.12.1 | Volumentexturen | 226 |
| 2.12.2 | Umgebungstexturen | 230 |
| 2.12.3 | Bump-Mapping | 240 |
| 2.12.4 | Rückblick | 243 |
| 2.12.5 | Übungsaufgaben | 244 |

| | | |
|----------|---|------------|
| 2.13 | Der Stencil-Buffer..... | 245 |
| 2.13.1 | Was war das noch gleich? | 245 |
| 2.13.2 | Die Details..... | 245 |
| 2.13.3 | Das Beispielprogramm | 249 |
| 2.13.4 | Rückblick | 252 |
| 2.13.5 | Übungsaufgaben..... | 252 |
| 2.14 | D3DX-Effekte..... | 252 |
| 2.14.1 | Probleme beim Verwalten von Modellen | 252 |
| 2.14.2 | „Effekte“..... | 253 |
| 2.14.3 | Laden eines Effekts | 256 |
| 2.14.4 | Mit Effekten rendern | 257 |
| 2.14.5 | Variablen von außen setzen und abfragen | 259 |
| 2.14.6 | Das Beispielprogramm | 261 |
| 2.14.7 | Rückblick | 263 |
| 2.14.8 | Übungsaufgaben..... | 263 |
| 2.15 | Transformierte Vertizes für 2D-Grafik..... | 264 |
| 2.15.1 | Wozu denn noch 2D? | 264 |
| 2.15.2 | Die Transformation umgehen..... | 264 |
| 2.15.3 | Ein anderes Vertexformat..... | 265 |
| 2.15.4 | DirectDraw imitieren..... | 266 |
| 2.15.5 | Eine andere Methode für 2D-Grafik..... | 268 |
| 2.15.6 | Rückblick | 269 |
| 2.15.7 | Übungsaufgaben..... | 269 |
| 2.16 | In Texturen rendern..... | 269 |
| 2.16.1 | Schritt 1: Erstellen einer Textur und eines Z-Buffers | 269 |
| 2.16.2 | Schritt 2: Das neue Render-Target setzen..... | 270 |
| 2.16.3 | Schritt 3: Rendern! | 270 |
| 2.16.4 | Einfacher mit D3DX..... | 270 |
| 2.16.5 | Wozu braucht man das?..... | 271 |
| 2.17 | Ausblick | 272 |
| 3 | 3D-Grafik mit der TriBase-Engine..... | 273 |
| 3.1 | Was Sie in diesem Kapitel erwartet..... | 274 |
| 3.2 | Direct3D mit der TriBase-Engine | 274 |
| 3.2.1 | Was uns das Leben leichter machen kann | 274 |
| 3.2.2 | Die Klasse <i>tbDirect3D</i> | 276 |
| 3.2.3 | Der Texturmanager – <i>tbTextureManager</i> | 287 |
| 3.2.4 | <i>tbVertexBuffer</i> und <i>tbIndexBuffer</i> | 301 |
| 3.2.5 | Die Effektklasse <i>tbEffect</i> | 312 |
| 3.2.6 | Ein allumfassendes Beispielprogramm..... | 316 |
| 3.2.7 | Rückblick | 325 |
| 3.2.8 | Ausblick | 325 |
| 3.3 | Modelldateien..... | 325 |
| 3.3.1 | Die Vorarbeit..... | 326 |
| 3.3.2 | Der Konverter..... | 331 |
| 3.3.3 | Eine Modellklasse | 333 |
| 3.3.4 | Das Beispielprogramm | 345 |

| | | |
|----------|--|------------|
| 3.3.5 | Rückblick | 348 |
| 3.4 | Texte zeichnen | 349 |
| 3.4.1 | Speicherung der Zeichen | 349 |
| 3.4.2 | Das Format der Textur..... | 350 |
| 3.4.3 | Transformierte Vertizes für Texte | 350 |
| 3.4.4 | Der Weg von TrueType zur Bitmap-Font..... | 351 |
| 3.4.5 | Inhalt der TBF-Dateien..... | 351 |
| 3.4.6 | Programmierung einer Schriftartklasse | 352 |
| 3.4.7 | Das Beispielprogramm | 362 |
| 3.4.8 | Rückblick | 364 |
| 3.5 | Ausblick | 364 |
| 4 | Eingabe | 365 |
| 4.1 | Was uns in diesem Kapitel erwartet | 366 |
| 4.2 | DirectInput kurz vorgestellt | 366 |
| 4.2.1 | Was kann DirectInput besser als Windows?..... | 366 |
| 4.2.2 | Geräte und Geräteklassen | 367 |
| 4.2.3 | GUIDs | 367 |
| 4.2.4 | Achsen und Knöpfe | 367 |
| 4.2.5 | Die Funktionsweise von DirectInput | 368 |
| 4.2.6 | Ein paar Worte zum Debuggen..... | 369 |
| 4.3 | Der Startschuss fällt | 369 |
| 4.3.1 | Erstellen des <i>IDirectInput8</i> -Objekts | 369 |
| 4.3.2 | Eingabegeräte abzählen | 371 |
| 4.3.3 | Rückblick | 372 |
| 4.4 | Initialisierung eines Geräts und Datenabfrage..... | 373 |
| 4.4.1 | Keine Angst vor <i>CreateDevice</i> !..... | 373 |
| 4.4.2 | Vorbereitungen treffen | 373 |
| 4.4.3 | Auf verlorene Eingabe achten! | 375 |
| 4.4.4 | Hinterlassen Sie Ihren Platz | 375 |
| 4.4.5 | Einmal Daten, bitte! | 375 |
| 4.4.6 | Rückblick | 376 |
| 4.5 | Die Tastatur..... | 376 |
| 4.5.1 | Das Datenformat der Tastatur..... | 376 |
| 4.5.2 | Tastencodes | 377 |
| 4.5.3 | Das Beispielprogramm | 379 |
| 4.5.4 | Begrenzungen der Tastatur..... | 380 |
| 4.5.5 | Rückblick | 381 |
| 4.6 | Die Maus..... | 381 |
| 4.6.1 | Das Datenformat der Maus..... | 381 |
| 4.6.2 | Relative Achsen..... | 381 |
| 4.6.3 | Die Mausknöpfe | 382 |
| 4.6.4 | Der exklusive Modus..... | 382 |
| 4.6.5 | Das Beispielprogramm | 383 |
| 4.6.6 | Rückblick | 384 |
| 4.7 | Der Joystick | 384 |
| 4.7.1 | Achsen, Knöpfe, POVs und Schieberegler | 384 |

| | | |
|----------|---|------------|
| 4.7.2 | Das Joystick-Datenformat | 385 |
| 4.7.3 | Das Beispielprogramm | 386 |
| 4.7.4 | Rückblick | 387 |
| 4.8 | Objekte abzählen und kalibrieren | 388 |
| 4.8.1 | Objekte abzählen | 388 |
| 4.8.2 | Eigenschaften festlegen | 389 |
| 4.8.3 | Achsenmodus | 390 |
| 4.8.4 | Achsenkalibrierung | 390 |
| 4.8.5 | Die tote Zone | 391 |
| 4.8.6 | Sättigung | 392 |
| 4.8.7 | Das Beispielprogramm | 392 |
| 4.8.8 | Gepufferte Daten und direkte Daten | 392 |
| 4.8.9 | Rückblick | 393 |
| 4.9 | Übungsaufgaben | 393 |
| 4.10 | Eine Eingabeklasse für die Engine | 393 |
| 4.10.1 | Probleme mit <code>DirectInput</code> | 393 |
| 4.10.2 | Das Prinzip der analogen Knöpfe | 394 |
| 4.10.3 | Die <code>tbDirectInput</code> -Klasse | 396 |
| 4.10.4 | Das Beispielprogramm | 414 |
| 4.10.5 | Rückblick | 416 |
| 4.11 | Ausblick | 416 |
| 5 | Sound und Musik | 417 |
| 5.1 | DirectSound kurz vorgestellt | 418 |
| 5.1.1 | Was kann DirectSound besser als Windows? | 418 |
| 5.1.2 | Soundpuffer und Mixer | 418 |
| 5.1.3 | Die Schnittstellen | 420 |
| 5.2 | Initialisierung von DirectSound | 420 |
| 5.2.1 | Formale Dinge | 420 |
| 5.2.2 | Abzählen der DirectSound-Geräte | 420 |
| 5.2.3 | Erstellung der <code>IDirectSound8</code> -Schnittstelle | 421 |
| 5.2.4 | Die Kooperationsebene wählen | 422 |
| 5.2.5 | Rückblick | 422 |
| 5.3 | Erstellen von Soundpuffern | 422 |
| 5.3.1 | Eigenschaften der Soundpuffer | 423 |
| 5.3.2 | Das Format eines Soundpuffers | 425 |
| 5.3.3 | Anfordern der <code>IDirectSoundBuffer8</code> -Schnittstelle | 426 |
| 5.3.4 | Der primäre Soundpuffer | 427 |
| 5.3.5 | Rückblick | 428 |
| 5.4 | Füllen eines sekundären Soundpuffers | 428 |
| 5.4.1 | Eine kleine Einführung in die Akustik | 428 |
| 5.4.2 | Wir sperren den Soundpuffer | 433 |
| 5.4.3 | Entsperren | 435 |
| 5.4.4 | Hinein mit den Daten! | 435 |
| 5.4.5 | Rückblick | 437 |
| 5.5 | Kontrolle eines Sounds | 438 |
| 5.5.1 | Die <code>Play</code> -Methode | 438 |

| | | |
|--------|---|-----|
| 5.5.2 | Festlegen der Lautstärke..... | 439 |
| 5.5.3 | Festlegen der Balance..... | 440 |
| 5.5.4 | Festlegen der Abspielfrequenz | 440 |
| 5.5.5 | Das Beispielprogramm | 441 |
| 5.5.6 | Rückblick | 441 |
| 5.6 | WAV-Dateien laden..... | 442 |
| 5.6.1 | Der RIFF-Header..... | 442 |
| 5.6.2 | Die WAV-Chunks | 443 |
| 5.6.3 | Die Funktion <i>LoadWAVFile</i> | 443 |
| 5.7 | 3D-Sound | 446 |
| 5.7.1 | Theorie des 3D-Sounds | 446 |
| 5.7.2 | Die <i>IDirectSound3DBuffer8</i> -Schnittstelle | 446 |
| 5.7.3 | Die <i>IDirectSound3DListener8</i> -Schnittstelle | 448 |
| 5.7.4 | Das Beispielprogramm | 450 |
| 5.7.5 | Rückblick | 451 |
| 5.8 | Echtzeiteffekte | 451 |
| 5.8.1 | Effekte – vorberechnet und in Echtzeit..... | 451 |
| 5.8.2 | Verschiedene Effektschnittstellen | 452 |
| 5.8.3 | Vorwarnung erforderlich!..... | 453 |
| 5.8.4 | Effekte mit <i>SetFX</i> anfordern..... | 453 |
| 5.8.5 | Die Effektschnittstelle abfragen | 454 |
| 5.8.6 | Effektparameter am Beispiel des Echos | 455 |
| 5.8.7 | Experimentieren ist angesagt!..... | 456 |
| 5.8.8 | Rückblick | 456 |
| 5.9 | Ergänzende Informationen | 457 |
| 5.9.1 | Die verschiedenen Schnittstellen..... | 457 |
| 5.9.2 | Klonen von Sounds | 458 |
| 5.9.3 | Status eines Soundpuffers..... | 458 |
| 5.10 | Die Klasse <i>tbDirectSound</i> | 459 |
| 5.10.1 | Erweiterung des Konfigurationsdialogs..... | 459 |
| 5.10.2 | Was <i>tbDirectSound</i> können soll | 460 |
| 5.10.3 | Die Klassendefinition | 460 |
| 5.10.4 | Die Initialisierungsmethode <i>Init</i> | 462 |
| 5.10.5 | Der Umgang mit dem Hörer | 463 |
| 5.11 | Die <i>tbSound</i> -Klasse | 464 |
| 5.11.1 | Fähigkeiten der Klasse | 464 |
| 5.11.2 | Das Prinzip der Soundpufferliste..... | 465 |
| 5.11.3 | Die Klassendefinition | 465 |
| 5.11.4 | Laden des Sounds | 467 |
| 5.11.5 | Die <i>Exit</i> -Methode | 469 |
| 5.11.6 | Die <i>SetPosition</i> -Methode..... | 470 |
| 5.11.7 | Einen Sound abspielen | 471 |
| 5.11.8 | Abspielen des nächsten Soundpuffers | 471 |
| 5.11.9 | Die restlichen Methoden | 472 |
| 5.12 | Musik ins Spiel bringen | 472 |
| 5.12.1 | Was unterscheidet Musik von Soundeffekten?..... | 472 |
| 5.12.2 | DirectShow-Grundlagen..... | 473 |

| | | |
|----------|---|------------|
| 5.12.3 | Kontrolle über den Filtergraphen..... | 475 |
| 5.12.4 | Die Klasse <i>tbMusic</i> | 476 |
| 5.12.5 | Das Beispielprogramm | 479 |
| 5.13 | Ausblick | 480 |
| 6 | Theorie der Spieleprogrammierung | 481 |
| 6.1 | Was Sie in diesem Kapitel erwartet..... | 482 |
| 6.2 | Warum Planung wichtig ist | 482 |
| 6.3 | Am Anfang steht die Idee..... | 482 |
| 6.3.1 | Inspiration..... | 483 |
| 6.3.2 | Auf Ideen vorbereitet sein | 484 |
| 6.3.3 | Aussortieren | 484 |
| 6.3.4 | Storydesign..... | 484 |
| 6.3.5 | Entwicklung eines Ablaufschemas | 486 |
| 6.4 | Suche nach Teammitgliedern | 486 |
| 6.5 | Vermitteln des Spiels und gemeinsame Analyse..... | 486 |
| 6.5.1 | Die Absichten klarmachen..... | 487 |
| 6.5.2 | Machbarkeitsanalyse | 487 |
| 6.5.3 | Tipps..... | 488 |
| 6.6 | Ausarbeitung der Details..... | 488 |
| 6.7 | Einteilung in Module..... | 489 |
| 6.8 | Level-Design und Atmosphäre..... | 489 |
| 6.8.1 | Abenteuer-, Action- und Rollenspiele | 489 |
| 6.8.2 | Puzzlespiele | 490 |
| 6.8.3 | Simulatoren | 490 |
| 6.8.4 | Wann eine Aufgabe zu schwer ist | 490 |
| 6.8.5 | Tipps für das Level-Design | 491 |
| 6.8.6 | Allgemeine Tipps für eine bessere Spielatmosphäre | 493 |
| 6.9 | Tipps zum Programmieren | 495 |
| 6.9.1 | Planung und Standard..... | 495 |
| 6.9.2 | Implementierung neuer Features | 495 |
| 6.9.3 | Die Liebe zum Detail..... | 497 |
| 6.10 | Testen Ihres Spiels | 497 |
| 6.10.1 | Testen während des Entwicklungsprozesses | 497 |
| 6.10.2 | Testen des fertigen Spiels | 497 |
| 6.11 | Ausblick | 499 |
| 7 | Das erste Spiel | 501 |
| 7.1 | Was Sie in diesem Kapitel erwartet..... | 502 |
| 7.2 | Planung | 502 |
| 7.2.1 | Das Spielprinzip und der Name des Spiels | 502 |
| 7.2.2 | Die Darstellung..... | 503 |
| 7.2.3 | Die Spielzustände | 503 |
| 7.2.4 | Das Spielgerüst..... | 504 |
| 7.3 | Die Grundklasse <i>CBreakanoid</i> | 505 |
| 7.3.1 | Variablen | 505 |
| 7.3.2 | Methoden..... | 506 |

| | | |
|----------|--|------------|
| 7.3.3 | Die <i>WinMain</i> -Funktion für Breakanoid..... | 511 |
| 7.4 | Das Titelbild..... | 511 |
| 7.4.1 | Planung des Titelbilds | 512 |
| 7.4.2 | Die Schriftarten | 512 |
| 7.4.3 | Initialisieren, Laden und Entladen des Titelbilds..... | 513 |
| 7.4.4 | Rendern des Titelbilds..... | 514 |
| 7.4.5 | Bewegung des Titelbilds | 516 |
| 7.5 | Das Hauptmenü..... | 516 |
| 7.5.1 | Planung des Hauptmenüs..... | 516 |
| 7.5.2 | Laden, Entladen, Betreten und Verlassen | 517 |
| 7.5.3 | Rendern | 517 |
| 7.5.4 | Bewegen des Hauptmenüs..... | 519 |
| 7.5.5 | Sound für das Hauptmenü!..... | 520 |
| 7.6 | Das Spiel | 521 |
| 7.6.1 | Planung des Spiels..... | 522 |
| 7.6.2 | Schritt 1: die <i>CGame</i> -Klasse | 523 |
| 7.6.3 | Schritt 2: Anzeigen des Levelmodells | 524 |
| 7.6.4 | Schritt 3: Her mit dem Schläger!..... | 526 |
| 7.6.5 | Schritt 4: Ein Levelsystem..... | 530 |
| 7.6.6 | Schritt 5: Bälle hinzufügen..... | 531 |
| 7.6.7 | Schritt 6: Die Blöcke | 537 |
| 7.6.8 | Schritt 7: Versuche | 543 |
| 7.6.9 | Schritt 8: Punkte | 545 |
| 7.6.10 | Schritt 9: Sound für das Spiel | 546 |
| 7.6.11 | Schritt 10: Hier spielt die Musik! | 547 |
| 7.7 | Minimieren im Vollbildmodus..... | 548 |
| 7.7.1 | Das Problem | 548 |
| 7.7.2 | Die Lösung | 548 |
| 7.8 | Motion-Blurring | 549 |
| 7.9 | Erweiterungsvorschläge | 552 |
| 7.10 | Ausblick | 552 |
| 8 | Das zweite Spiel..... | 553 |
| 8.1 | Was Sie in diesem Kapitel erwartet..... | 554 |
| 8.2 | Planung | 554 |
| 8.2.1 | Das Spielprinzip und der Name des Spiels | 554 |
| 8.2.2 | Die Spielzustände | 555 |
| 8.2.3 | Die Schiffe..... | 555 |
| 8.2.4 | Die Waffensysteme | 556 |
| 8.2.5 | Speicherung der Informationen | 557 |
| 8.2.6 | Die Schiffssysteme | 558 |
| 8.3 | Schiffs- und Waffentypen | 559 |
| 8.3.1 | Die Struktur <i>SShipType</i> | 559 |
| 8.3.2 | Die Struktur <i>SWeaponType</i> | 560 |
| 8.3.3 | Laden aus der INI-Datei | 561 |
| 8.4 | Die Klasse <i>tObject</i> | 563 |
| 8.4.1 | Unser bisheriger Ansatz | 563 |

| | | |
|--------|--|-----|
| 8.4.2 | Das neue Prinzip..... | 563 |
| 8.4.3 | Position und Skalierung..... | 564 |
| 8.4.4 | Ein Fall für die Matrix!..... | 564 |
| 8.4.5 | Relativ zu absolut – und zurück..... | 565 |
| 8.4.6 | Die Physik kommt hinzu | 566 |
| 8.4.7 | Implementierung von <i>tbObject</i> | 567 |
| 8.5 | Der Umgang mit Schiffen | 573 |
| 8.5.1 | Die <i>CShip</i> -Klasse | 573 |
| 8.5.2 | Integrierung in <i>CGame</i> | 575 |
| 8.5.3 | Bewegen der Schiffe..... | 576 |
| 8.5.4 | Kontrolle eines Schiffs | 581 |
| 8.5.5 | Rendern der Schiffe..... | 583 |
| 8.5.6 | Aufschalten von Zielen..... | 584 |
| 8.6 | Alle Waffen abfeuern!..... | 586 |
| 8.6.1 | Die <i>CProjectile</i> -Klasse | 586 |
| 8.6.2 | Feuern..... | 587 |
| 8.6.3 | Bewegen | 589 |
| 8.6.4 | Rendern | 591 |
| 8.7 | Sprites | 591 |
| 8.7.1 | Was sind Sprites? | 591 |
| 8.7.2 | Missbrauch der Kameraachsen | 592 |
| 8.7.3 | Die TriBase-Sprite-Engine | 592 |
| 8.7.4 | Zurück zum Spiel: Rendern der Laser | 598 |
| 8.8 | Kollisionserkennung | 602 |
| 8.8.1 | Rückblick: Umgebungskugel und Umgebungsquader..... | 602 |
| 8.8.2 | Das Prinzip der Kollisionserkennung | 602 |
| 8.8.3 | Kugel – Kugel | 602 |
| 8.8.4 | Linie – Kugel..... | 604 |
| 8.8.5 | Linie – Dreieck..... | 609 |
| 8.8.6 | Dreieck – Dreieck..... | 616 |
| 8.8.7 | Linie – Quader..... | 621 |
| 8.8.8 | Quader – Quader | 626 |
| 8.8.9 | Wie wir mit Modellen umgehen..... | 628 |
| 8.8.10 | Vorberechnungen | 630 |
| 8.8.11 | Linien und Modelle | 639 |
| 8.8.12 | Kollision zwischen zwei Modellen..... | 643 |
| 8.8.13 | Hardcore-Kollisionserkennung..... | 647 |
| 8.8.14 | Volltreffer! | 648 |
| 8.8.15 | Zusammenstoß zweier Schiffe..... | 651 |
| 8.9 | Auto-Aiming | 654 |
| 8.9.1 | Definition | 654 |
| 8.9.2 | Der mathematische Hintergrund..... | 654 |
| 8.9.3 | Die neue <i>CShip::Fire</i> -Methode | 655 |
| 8.10 | „Künstliche Intelligenz“ | 656 |
| 8.10.1 | Das Verhalten eines Schiffs..... | 657 |
| 8.10.2 | Schritt 1: Kurs auf das Ziel nehmen | 657 |
| 8.10.3 | Schritt 2: Feuern | 659 |

| | | |
|----------|--|------------|
| 8.10.4 | Schritt 3: Ausweichmanöver bei Treffer | 660 |
| 8.10.5 | Schritt 4: Ausweichmanöver bei drohender Kollision | 661 |
| 8.10.6 | Schritt 5: Wechseln des Ziels | 662 |
| 8.11 | Partikel | 662 |
| 8.11.1 | Was Partikel sind | 662 |
| 8.11.2 | Das Partikelsystem der TriBase-Engine | 663 |
| 8.11.3 | Antriebs- und Raketenflammen | 664 |
| 8.11.4 | Explosionen | 665 |
| 8.11.5 | Aufleuchten des Schutzschildes | 668 |
| 8.11.6 | Trümmer | 670 |
| 8.12 | Weitere optische Verfeinerungen | 671 |
| 8.12.1 | Sky-Box, Licht und Nebel | 671 |
| 8.12.2 | Ein „Sternfeld“ | 674 |
| 8.12.3 | Glühen von Projektilen | 676 |
| 8.13 | Die Kamera | 676 |
| 8.14 | Das Cockpit | 679 |
| 8.14.1 | Das Cockpitmodell | 679 |
| 8.14.2 | Die Anzeigen | 680 |
| 8.14.3 | Das HUD | 683 |
| 8.14.4 | Radar | 684 |
| 8.15 | Der Sound | 690 |
| 8.15.1 | Schüsse | 690 |
| 8.15.2 | Antriebssounds | 691 |
| 8.15.3 | Der Hörer | 692 |
| 8.16 | Die Benutzeroberfläche | 692 |
| 8.16.1 | Die TriBase-Benutzeroberfläche | 693 |
| 8.16.2 | Erstellung des Hauptmenüs | 702 |
| 8.17 | Optimierungen und der Feinschliff | 709 |
| 8.17.1 | Sichtbarkeit eines Objekts | 709 |
| 8.17.2 | Render-Modell und Kollisionsmodell | 713 |
| 8.17.3 | Musik | 714 |
| 8.17.4 | Wackelndes Cockpit | 714 |
| 8.17.5 | Screenshots schießen | 714 |
| 8.18 | Erweiterungsvorschläge | 715 |
| 8.19 | Ausblick | 716 |
| 9 | Fortgeschrittene Techniken | 717 |
| 9.1 | Was Sie in diesem Kapitel erwartet | 718 |
| 9.2 | Schatten mit dem Stencil-Buffer | 718 |
| 9.2.1 | Schatten in der 3D-Grafik | 718 |
| 9.2.2 | Ansätze | 718 |
| 9.2.3 | Das Prinzip | 719 |
| 9.2.4 | Die Klasse <i>tbShadowVolume</i> | 724 |
| 9.2.5 | Das Beispielprogramm | 733 |
| 9.3 | Videos abspielen | 733 |
| 9.3.1 | Zielsetzung | 733 |
| 9.3.2 | Schreiben eines eigenen Filters | 733 |

| | | |
|--------|---|-----|
| 9.3.3 | Verwenden des Filters | 743 |
| 9.3.4 | Der Back-Buffer-Mechanismus | 746 |
| 9.3.5 | Videos in den Speicher laden | 746 |
| 9.4 | Stereo-3D-Grafik | 747 |
| 9.4.1 | Das räumliche Sehen | 747 |
| 9.4.2 | Trennen der Bilder | 748 |
| 9.4.3 | Implementierung | 750 |
| 9.4.4 | Beispielprogramme | 751 |
| 9.5 | Raumaufteilung | 751 |
| 9.5.1 | Rekursives Rendern | 751 |
| 9.5.2 | PVS und Portale | 752 |
| 9.5.3 | Light-Mapping | 753 |
| 9.5.4 | Die TriBase-Klasse <i>tbOctree</i> | 755 |
| 9.6 | Terrain-Rendering | 755 |
| 9.6.1 | Repräsentierung eines Terrains | 756 |
| 9.6.2 | Unterteilung des Terrains | 756 |
| 9.6.3 | Erzeugen der Dreiecke | 757 |
| 9.6.4 | Terrain-Rückruffunktion | 758 |
| 9.6.5 | Geo-MIP-Mapping | 758 |
| 9.6.6 | Texturierung | 758 |
| 9.6.7 | Beleuchtung | 760 |
| 9.6.8 | Das TriBase-Tool <i>TerrainEditor</i> | 761 |
| 9.7 | Die Welt der Shader | 762 |
| 9.7.1 | Was ein Shader ist | 762 |
| 9.7.2 | Einsatzgebiete | 762 |
| 9.7.3 | Die fixe und die programmierbare Rendering-Pipeline | 763 |
| 9.7.4 | Ein einfacher Vertex-Shader | 763 |
| 9.7.5 | Ein einfacher Pixel-Shader | 766 |
| 9.7.6 | Praktischer Einsatz von Shadern | 768 |
| 9.7.7 | Weiterführende Quellen und Referenz | 778 |
| 9.8 | Charakteranimation | 779 |
| 9.8.1 | Das Grundprinzip | 779 |
| 9.8.2 | Skinning in Hardware | 780 |
| 9.8.3 | Skinning mit D3DX | 781 |
| 9.8.4 | Weitere Informationen | 781 |
| 9.9 | PlugIns schreiben und laden | 781 |
| 9.9.1 | DLL-Dateien explizit laden | 781 |
| 9.9.2 | Adresse einer Funktion abfragen | 782 |
| 9.9.3 | DLL-Dateien erzeugen | 783 |
| 9.9.4 | Die Kommunikation zwischen Anwendung und PlugIn | 785 |
| 9.9.5 | Das Beispielprogramm | 785 |
| 9.10 | Arbeiten mit Threads | 785 |
| 9.10.1 | Prozesse im Betriebssystem | 785 |
| 9.10.2 | Was ist ein Thread? | 786 |
| 9.10.3 | Die Thread-Funktion | 787 |
| 9.10.4 | Erzeugen eines Threads | 787 |
| 9.10.5 | Verwaltungsfunktionen | 788 |

| | | |
|-----------|--|------------|
| 9.10.6 | Thread-Synchronisierung | 789 |
| 9.10.7 | Zusammenfassung | 794 |
| 9.11 | Eine einfache Skriptsprache | 794 |
| 9.11.1 | Einsatzgebiet von Skriptsprachen | 794 |
| 9.11.2 | Ein Skript als Liste von Befehlen | 795 |
| 9.11.3 | Verwendung von Funktionszeigern | 795 |
| 9.11.4 | Übergabe von Parametern | 796 |
| 9.11.5 | Einen Thread verwenden | 797 |
| 9.11.6 | Die Klasse <i>CScript</i> | 797 |
| 9.11.7 | Das Beispielprogramm | 803 |
| 9.11.8 | Fortgeschrittene Skriptsprachen | 804 |
| 9.12 | Interpolationen | 804 |
| 9.12.1 | Nachteil linearer Interpolationen | 804 |
| 9.12.2 | Hermite-Interpolation mit Tangenten | 805 |
| 9.12.3 | In zwei Richtungen – die bilineare Interpolation | 809 |
| 9.13 | Abstrakte Spiel- und Spielzustandsklassen | 810 |
| 9.13.1 | Die Spielzustandsklasse | 810 |
| 9.13.2 | Die Spielklasse | 812 |
| 9.13.3 | Die Anwendung | 814 |
| 9.14 | Fixed-Step-Spiellogik | 814 |
| 9.14.1 | Die Problematik | 814 |
| 9.14.2 | Ein Lösungsansatz | 815 |
| 9.15 | Online-Highscores und Versionskontrolle | 819 |
| 9.15.1 | Die Möglichkeiten | 819 |
| 9.15.2 | Die Realisierung | 819 |
| 9.15.3 | Internetseiten in einem C++-Programm abrufen | 820 |
| 9.16 | Ausblick | 821 |
| 10 | FAQ, Internetseiten und CD-ROM | 823 |
| 10.1 | FAQ | 824 |
| 10.2 | Interessante Internetseiten | 825 |
| 10.3 | Die Programme auf der Begleit-CD-ROM | 827 |
| 10.3.1 | 3Dografe | 827 |
| 10.3.2 | AC3D v5.021 Demo | 828 |
| 10.3.3 | MilkShape 3D v1.76 Demo | 829 |
| 10.3.4 | POV-Ray v3.6 | 830 |
| 10.3.5 | Sound Forge 7 Demo | 831 |
| 10.3.6 | Terragen 0.943 | 832 |
| 10.3.7 | Texture Maker v2.81 Demo | 833 |
| 10.3.8 | Visual Assist X Demo | 834 |
| 10.3.9 | 2D-Game-Framework | 834 |
| 10.4 | Das Ende | 834 |
| | Index | 835 |