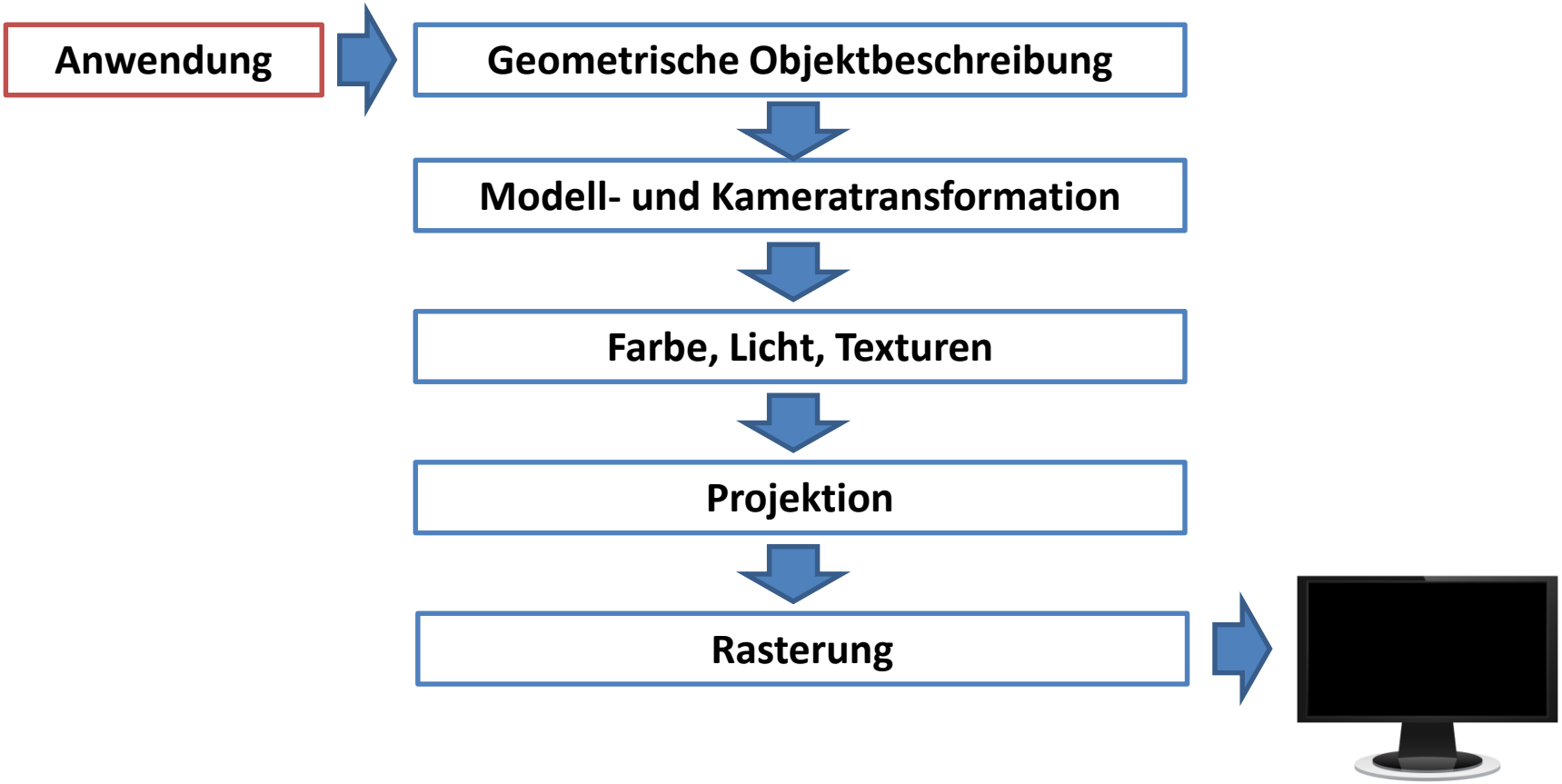
Einordung der Computergrafik

- Wir befassen uns mit generativer Computergrafik
- Breites Anwendungsfeld: Ausbildungssimulation, CAD/CAM, Visualisierung, Unterhaltung,…
- Virtual Reality: Darstellung + Wahrnehmung vollständig computergenerierter virtueller Umgebungen
- Augmented Reality: Computergestützte Erweiterung der Realitätswahrnehmung
- Unterscheidung in der Computergrafik:
  - Interaktive/Echtzeit-CG ⇔ Nicht-Echtzeit-CG

Verarbeitungsschritte

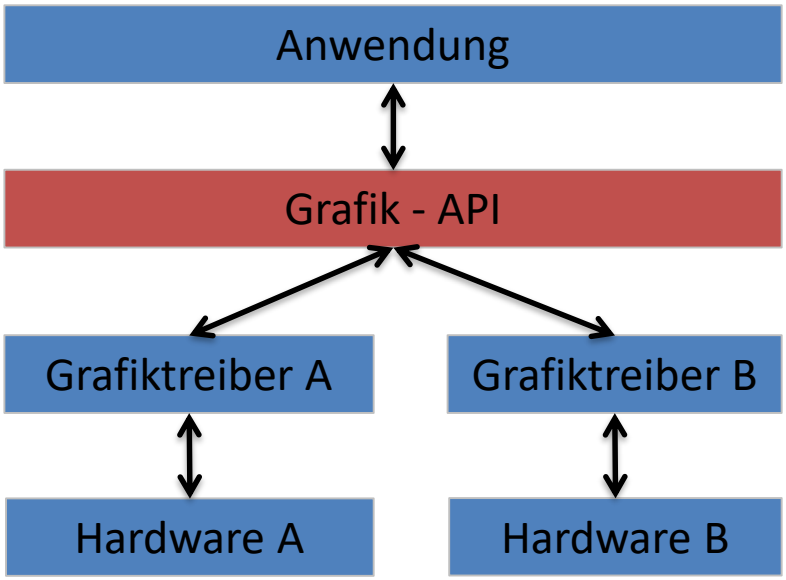
- Vereinfachte Grafikpipeline



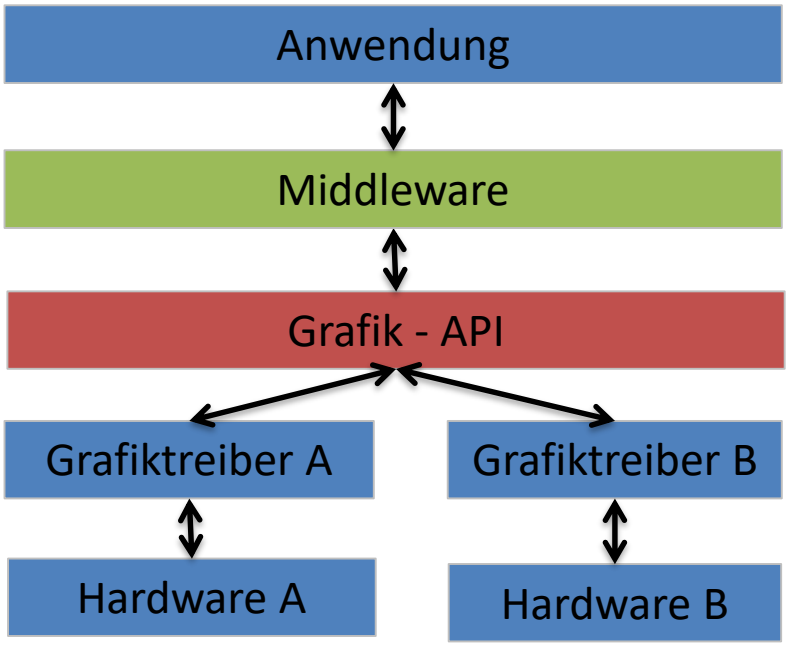
Architekturmodelle

Programmierbibliotheken

Immediate-mode (IM)



Retained-mode (RM)



Programmierbibliotheken

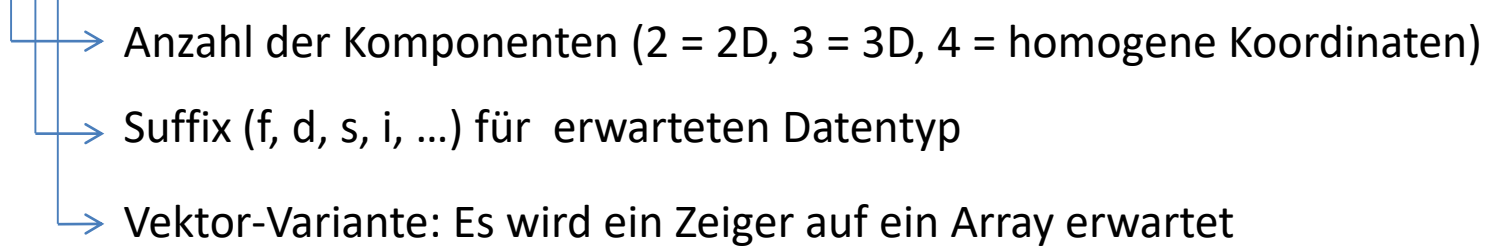
- Fixed Function
  - Konfigurierbare Module für z.B. Beleuchtung, Puffer etc.
- Shaders
  - Teile der Grafik-Pipeline können frei definiert werden
  - Z.B. Implementierung eines eigenen Beleuchtungsmodells
- Programmierbare Pipeline
  - Volle Freiheit in der Programmierung von GPUs mit Nutzung der vollen Beschleunigungsmöglichkeiten

Beispiel: Datentypen und Namenskonventionen

- OpenGL-Aufrufe zur Definition eines Punktes mit Koordinate $x = 1, y = 2$

```
glVertex2i(1, 2);
glVertex2f(1.0, 2.0);

GLfloat v[] = {1.0, 2.0};
glVertex2fv(v);
```



State Machine

- Periodisches Neuzeichnen der Szene
- API-Zugriff setzt eine Zustandsvariable, z.B. aktuelle Farbe
- Zustand bleibt so lange erhalten bis er explizit geändert wird
- Default-Werte für alle Variablen, z.B.
  - Ansicht- und Projektionstransformationen
  - Polygon Drawing Modes
  - Positionen und Eigenschaften des Lichtes, Materialeigenschaften
- Zustandsänderungen und -abfrage

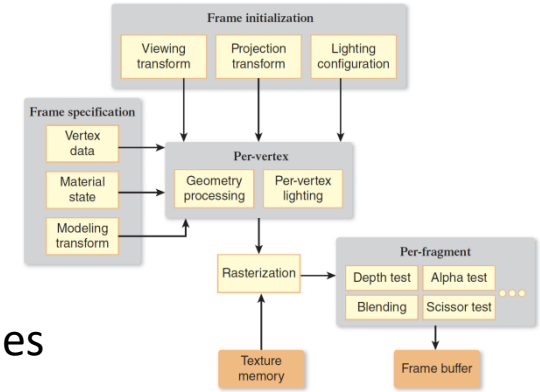
```
glColor3f(1.0,0.0,0.0);             // current color set to red

glEnable(GL_DEPTH_TEST);            // Enable depth buffer testing
glDisable(GL_DEPTH_TEST);           // Disable depth buffer testing

GLboolean state;
state = glIsEnabled(GL_DEPTH_TEST);         // false
state = glIsDisabled(GL_DEPTH_TEST);        // true
glGetBooleanv(GL_DEPTH_TEST, &state);       // false
```

OpenGL Rendering Pipeline

1) Frame Initialization
   - Z.B. Definition der Kameraposition, Blickrichtung, Projektionsart
   - Z.B: Setzen von Statusvariablen für Licht



2) Frame Specification: Definition der Szene
   - Geometrische Daten, z.B. geometrische Körper, Meshes definiert durch Vertices (Punktkoordinaten)
   - Festlegung von Materialeigenschaften (Farben, Verhalten bei Licht)
   - Transformationen des Modells (Verschiebung, Rotation, …)

3) Per-Vertex-Operationen
   - Konvertierung der geometrischen Daten aus Modellkoordinaten zu Fensterkoordinaten auf dem Bildschirm
   - Berechnung der beleuchtungs- und Materialabhängigen Farbe eines geometrischen Objektes

OpenGL Rendering Pipeline

4) Rasterisierung
   - Umwandlung der „kontinuierlichen“ Geometrie in diskrete Pixel
   - Festlegung von Pixel-Farben basierend auf Licht, Texturen, etc.

5) Per-Fragment Operationen
   - Festlegung welche/wie die Pixel auf dem Bildschirm dargestellt werden, z.B. Verdeckung von Objekten, Transparenz

6) Frame-Buffer
   - Zwischenpuffer für die Anzeige der gerenderten Szene