### Chap 1. Introduction

#### Notations

<table>
<thead>
<tr>
<th>Type</th>
<th>Notation</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle</td>
<td>lowercase Greek</td>
<td>$\alpha, \phi, \rho, \eta, \gamma_{242}, \theta$</td>
</tr>
<tr>
<td>scalar</td>
<td>lowercase italic</td>
<td>$a, b, t, u_k, v, w_{ij}$</td>
</tr>
<tr>
<td>vector or point</td>
<td>lowercase bold</td>
<td>$a_u, v_s, h(\rho), h_z$</td>
</tr>
<tr>
<td>matrix</td>
<td>capital bold</td>
<td>$T(t), X, R_x(\rho)$</td>
</tr>
<tr>
<td>plane</td>
<td>$\pi$: a vector +</td>
<td>$\pi: n \cdot x + d$,</td>
</tr>
<tr>
<td></td>
<td>a scalar</td>
<td>$\pi_1: n_1 \cdot x + d_1$</td>
</tr>
<tr>
<td>triangle</td>
<td>$\triangle 3$ points</td>
<td>$\triangle v_0 v_1 v_2, \triangle cba$</td>
</tr>
<tr>
<td>line segment</td>
<td>two points</td>
<td>$uv, a_i b_j$</td>
</tr>
<tr>
<td>geometric entity</td>
<td>capital italic</td>
<td>$A_{OBB}, T, B_{AABB}$</td>
</tr>
</tbody>
</table>

Table 1.1. Summary of the notation used in this book.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:</td>
<td>dot product</td>
</tr>
<tr>
<td>2:</td>
<td>cross product</td>
</tr>
<tr>
<td>3:</td>
<td>transpose of the vector $v$</td>
</tr>
<tr>
<td>4:</td>
<td>piecewise vector multiplication</td>
</tr>
<tr>
<td>5:</td>
<td>the unary, perp dot product operator</td>
</tr>
<tr>
<td>6:</td>
<td>determinant of a matrix</td>
</tr>
<tr>
<td>7:</td>
<td>absolute value of a scalar</td>
</tr>
<tr>
<td>8:</td>
<td>length (or norm) of argument</td>
</tr>
<tr>
<td>9:</td>
<td>factorial</td>
</tr>
<tr>
<td>10:</td>
<td>binomial coefficients</td>
</tr>
</tbody>
</table>

Table 1.2. Notation for some mathematical operators.

\[
M = \begin{pmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{pmatrix}
\]  
(1.1)

\[
M = \begin{pmatrix} m_{0} & m_{1} & m_{2} \end{pmatrix} = \begin{pmatrix} m_x & m_y & m_z \end{pmatrix} = \begin{pmatrix} m_{0}^T \\ m_{1}^T \\ m_{2}^T \end{pmatrix}
\]  
(1.2)
Figure 2.2. The basic construction of the rendering pipeline, consisting of three stages: application, geometry, and the rasterizer. Each of these stages may be a pipeline in itself, as illustrated below the geometry stage, or a stage may be (partly) parallelized as shown below the rasterizer stage. In this illustration, the application stage is a single process, but this stage could also be pipelined or parallelized.

The Application Stage

- It executes in software.
- It can affect the time consumed by the geometry and the rasterizer stages by decreasing the number of triangles to be rendered.
- This stage includes collision detection, animation, acceleration algorithms (such as hierarchical view frustum culling), etc.

The Geometry Stage

- It is responsible for per-polygon or per-vertex operations.
- Five major functional stages:
  1. Model and View Transform
  2. Lighting and Shading
  3. Projection
  4. Clipping
  5. Screen Mapping
0. Model and View Transform
   - Model Space $\rightarrow$ Model Transform
   - World Space $\rightarrow$ View Transform
   - Eye Space $\rightarrow$

   For efficiency reasons, these are usually concatenated into one matrix before transforming the models. (Thus, the world space is not available.)

   - More elaborate transforms can happen at this stage (e.g., vertex blending and transforms performed by a vertex shader).

0. Lighting and Shading
   - Light sources
   - Texture, color
   - Gouraud shading
   - Pixel shading techniques are employed during rasterization

0. Projection
   - Canonical view volume
   - Orthographic Projection
   - Perspective Projection
   - Normalized device coordinates (after the transform, the models are in.)
0. **Clipping**
   - After the projection transformation, the transformed primitives are clipped against the unit cube, and the canonical view volume.

0. **Screen Mapping**
   - The x- and y-coordinates of each primitive (inside the view volume) are transformed to form screen coordinates.
   - Window coordinate = screen coordinates + the z-coordinate
     (The z-coordinate is not affected by this mapping.)

**The Rasterizer Stage**

0. To assign correct colors to the pixels from the transformed vertices, colors, and texture coordinates.

0. **Rasterization or Scan conversion**

0. It handles per-pixel operations.

0. **Color buffer**, depth buffer, stencil buffer, accumulation buffer, (part of OpenGL)

0. **Frame buffer consists of all the buffers on a system.**

0. **Double buffering**
   - Front buffer
   - Back buffer

0. **Alpha channel** is associated with the color buffer and stores a related opacity value for each pixel.

0. **Accumulation buffer** is used for effects such as motion blur, depth of field, antialiasing,