

# The Real-Time Graphics Pipeline

Benjamin Brown

Monday, 26th September 2022

# The Key Idea

Basic task in computer graphics is ***render*** 3-dimensional objects:

- ▶ given a scene composed of geometric objects in 3d space;
- ▶ produce a 2d image showing the objects from a specific viewpoint.

Two methods of rendering:

- ▶ ***object-order rendering***: for each **object**, which **pixels** are influenced by it?
  - ▶ Example: rasterisation.
- ▶ ***image-order rendering***: for each **pixel**, which **object** is influenced by it?
  - ▶ Example: ray-tracing.

# Real-Time Rendering

***Real-time rendering*** refers to the process of rendering a scene in less than  $1/30^{\text{th}}$  of a second (i.e., refresh rate  $> 30$  Hz).

- ▶ Real-time rendering refers to rendering that is fast enough to allow for the user's ***real-time interaction***.
- ▶ As little as 15 ms of temporal delay can slow and interfere with the interactivity.

So speed is essential for interactivity:

- ▶ For a 1080p image at 90 Hz, an image-order rendering iteration would need to be performed 186,624,000 times per second.
- ▶ Object-order rendering is (usually) faster in this case.

# The Graphics Pipeline

The main function of the *graphics pipeline* is:

- ▶ Given a virtual camera, 3d objects, light sources, etc., render a 2d image;
- ▶ Object locations and shapes determined by their geometry, environment, camera placement, etc.;
- ▶ Object appearance affected by material, light sources, shading, etc.

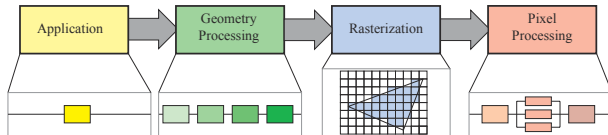
Some key ideas regarding the pipeline are:

- ▶ Consists of several states, with each making up part of a larger task;
- ▶ The states execute in parallel, with each stage dependent on the result from the previous.

# Stages of the Graphics Pipeline

Roughly, the main stages of the pipeline are:

1. *application*;
2. *geometry processing*;
3. *rasterisation*;
4. *pixel processing*.



# The Application Stage

Involves tasks typically implemented by the software running on general-purpose CPUs:

- ▶ collision detection;
- ▶ animation;
- ▶ physics simulation, etc...

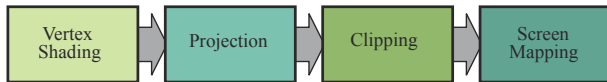
More specifically, the application stage:

- ▶ Reads ***primitive*** data and assembles it into primitives for later states.
  - ▶ Vertex data (position, normal vectors, colour, etc.),
- ▶ Initialises GPU allocated memory buffers.
  - ▶ Vertex and index buffers.

# The Geometry Processing Stage

Responsible for the per-triangle and per-vertex operations:

- ▶ Vertex processing (transforming and shading);
- ▶ Projecting;
- ▶ Clipping;
- ▶ Screen mapping.



# Geometry Processing Stage – Vertex Processing

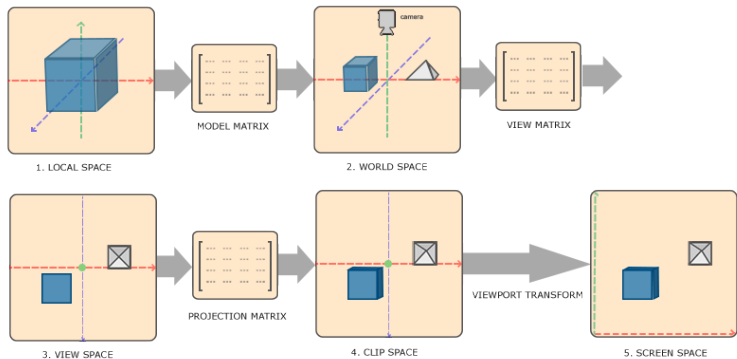
***Vertex shading*** deals with:

- ▶ Determining the effect of a light on a material (***shading***);
- ▶ Projecting from ***world space*** onto ***view space***;
- ▶ ***Clipping*** away the view space primitives which do not lie within the view volume;
- ▶ Mapping the vertices onto ***screen space***;



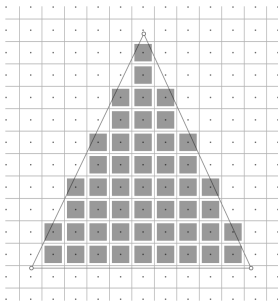
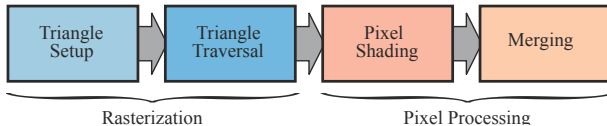


# The Geometry Stage – Transformations



# The Rasterisation Stage

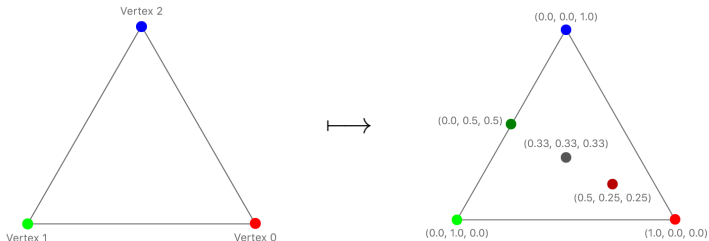
***Rasterisation*** can be split into two sub-pipelines:



Data for each pixel is called a ***fragment***.

# The Rasterisation Stage – Pixel Processing

- Example – fragment data from interpolation.



# The Rasterisation Stage – Pixel Processing

- ▶ Example – texture mapping.



# Tables and Figures

- ▶ Use `tabular` for basic tables — see Table 1, for example.
- ▶ You can upload a figure (JPEG, PNG or PDF) using the files menu.
- ▶ To include it in your document, use the `includegraphics` command (see the comment below in the source code).

## Examples

Some examples of commonly used commands and features are included, to help you get started.

Item	Quantity
Widgets	42
Gadgets	13

Table 1: An example table.

## Readable Mathematics

Let  $X_1, X_2, \dots, X_n$  be a sequence of independent and identically distributed random variables with  $E[X_i] = \mu$  and  $\text{Var}[X_i] = \sigma^2 < \infty$ , and let

$$S_n = \frac{X_1 + X_2 + \dots + X_n}{n} = \frac{1}{n} \sum_i^n X_i$$

denote their mean. Then as  $n$  approaches infinity, the random variables  $\sqrt{n}(S_n - \mu)$  converge in distribution to a normal  $\mathcal{N}(0, \sigma^2)$ .