Summary Image Processing

Abderrahmen Rakez

1 course goals

* introduction to the fundamentals of rasterization-based image generation: rasterization is the task of taking an image described in a vector graphics format and converting it into a raster image(pixels or dots) for output on a video display or printer, or for storage in a bitmap file format.

* functionality of the graphics rendering pipline: Rendering is the automatic porcess of generating a photorealistic or non-photorealistic image from a 2D or 3D model by means of computer graph

* advanced rendering effects

* introduction to the openGL graphics API

2 OpenGL

2.1 Intoduction

OpenGL (Open Graphics Library) is a cross-plarform API[1] for rendering 2D and 3D vertor graphics[2]...

* display of geometric representations and attributes

* independent from operating system and window system

[1] Application Programming Interface

[2] is the use of polygons(are used in Com. Graphics to compose images that are 3-dim in appearence) to represent images in computer graphics

2.2 GPU Data Flow

* data tramsfer to GPU: vertices with attribures and connectivity
* vertex shader[3]: a program that is executed for each vertex, with input and output is a vertex
* rasterizer
* fragment shader: also known pixel shader, a program that is executed for each fragement, with input and output is a fragment. compute color and other attribures of each fragment[4]
* framebuffer update: is portion of RAM containing a bitmap that drives a video display. It is a memory buffer containing a complete frame of data

3 Rendering

3.1 Rasterization

* rendering algorithm for generating 2D images form 3D scenes
* transforming geometric primitives such as lines and polygons into raster image representations, i.e pixels
* 3D objects are approximitely represented by vetices(points), lines, polygons
* these primitives are processed to obtain a 2D image

3.2 Redenting Pipeline

known also as graphics pipeline is a concenptual model in computer graphics that describe what step a graphics system needs to perform to render a 3D scene to a 2D screen, plainly speaking, the graphics pipeline is the process of turning that 3D model into what the computer displays.
Characteristics of the 3D input:

1. a virtual camera: positon, orientation, focal lenght

[3] are HW or SW modules, that implement spezific redering
[4] a technical term usally meaning a single pixel

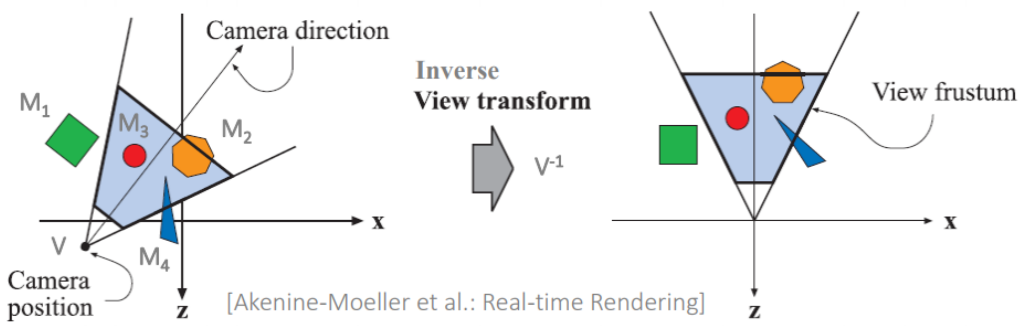
2. objects: points(vertex/vertices), lines, polygons, geometry and merial properties
3. light sources: direction, postion, color, intensity
4. textures(images)

Characteristics of the 2D output: per-pixel color value in the framebuffer

4 Transformations

* is used to: position, reshape, and animate objects, lights, and the virtual camera
* are respresented with 4x4 matrices
* are applied to vertices and normals
* vertices(position) and normals(directions) are represented with 4D vectors

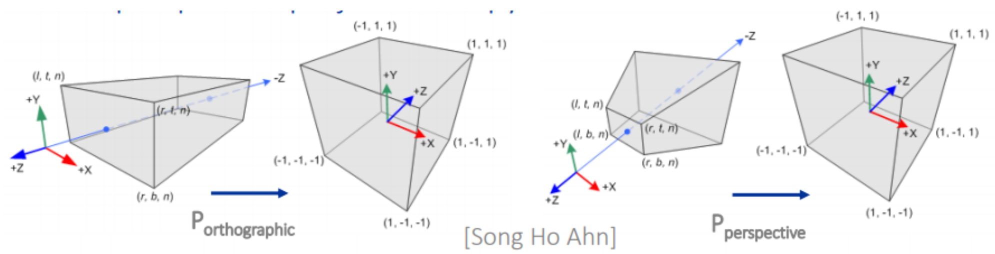
4.1 Modelview Transform



* $M_{1..4}$ and V are matrices representing transformations
* $M_{1..4}$ are model transform to place the objects in the scene
* V places and orientates the camera in space, V^-1 transforms the camera to the origin looking along the negative z-axis
* model and view transforms are combined in the modelview transform
* the modelview tranform $V^-1M_{1..4}$ is applied to the objects

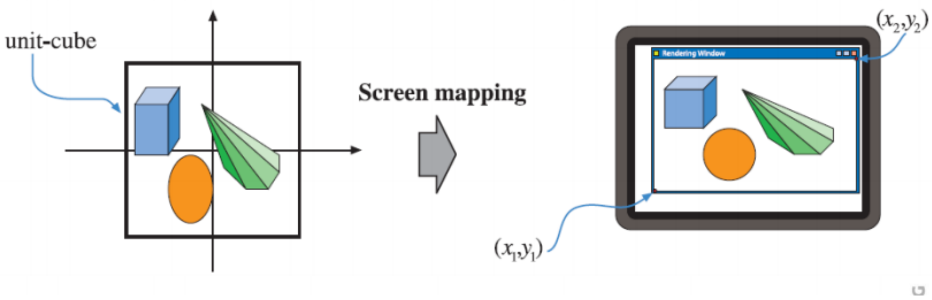
4.2 Prjection Trasform

* P transforms the view volume to the canonical view volume
* the view volume depends on the camera properties :
  1. orthographic projection ==> cuboid
  2. perspective projection ==> pyramidal frustum
* canonical view volume is a cube from (-1, -1, -1) to (1, 1, 1)
* view volume is specified by near, far, left, right, bottom, top



4.3 Viewport Transform / Screen Mapping

* prjected primitive coordinates (x_p, y_p, z_p) are transformed to screen coordinates (x_s, y_s)
* screen coordinates together with depth value are window coordinates (x_s, y_s, z_w)



4.4 Vertex Transforms

Vertex is a data structure that describes certain attribute, like the position of a point in 2D or 3D space ...
the vertex tranform went through some transformation before achieving the window space, like model trans., inverse view trans., projection trans., and viewport trans.

4.5 Other Transformation

* congurent transformations (Euclidean transformations)

- preserve shape and size
- translation, rotation, reflection

* similarity transformations

- preseve shape
- translation, rotation, reflection, scale

4.6 Affine Transformations

Folie Seite 12.