Computer graphics

Linus Falk

January 17, 2023

Contents

1 Introduction to computer graphics 2

2 Digital images 2

2.1 Spatial resolution 3

2.2 Color images 3

3 High dynamic range HDR images 3

4 Computer graphics history 3

5 Quick look at what we will learn 4

6 Pipeline 4

7 Conclusion for this lecture 4

8 Graphics programming 4

Lecture 1: Introduction

monday 16 jan 10:15

1 Introduction to computer graphics

What can it be used for

- Entertainment
  - Games, pushing graphic card development forward..
- User interfaces
- Applications
- Data visualization
  - Information Visualixation
    - Charts diagrams etc
    - Plots of N-d data after dimensonality
  - Data visulization
    - ...
- Image procsessing
  - Extracts information
  - producess outputs like: calssifications and segmented objects
- Computer vision
  - feature tracking
  - Tracking
  - **3D reconstruction**
- Photogrammetry?
  - Reconstruct 3d models from photos taken from different viewpoints. How do the different points align. Use **CG** to view result.
- Cultural heritage?
  - preserve art works, 3D scanning and other techniuques.
  - Display the result for the audience with **CG**
- CAD
  - 3d modeling
- much much more..

2 Digital images

Grayscale image a (raster graphics) is stored as a matrix with intensity representing each pixel.

2.1 Spatial resolution

Modern computer 2660x1600 pixels, picture element

Color depth

Bit depth, how many bits are used to represent the intensity? True color 24 bit, 8bit per color channel

2.2 Color images

Three color images RGB, some times an additional channel **alpha** for storing opacity

3 High dynamic range HDR images

8 bits per collor channnel often sufficient. In many computer graphics applications a higher dynamic range is needed. It is therefore common to use mere bits per color channel, e.g. 16 or 32 bit floating point values

- When images are manipulated,

more from slides

Image based lightning with high dynamic range images of real world enviroment.

useful to create a realistic illuminations.

To capturing a the great contrast between sky and the ground, 8 bit depth is not usually enough.

4 Computer graphics history

University of Utah prominent sections that did a lot of work in this are during 70s. Many methods invented by people from this sections, Sutherland, Blinn, Phong, Guoraud, Catmull (Turing awards winner), Newell and others. During the 80s we got Graphical user interfaces, Mac, Amiga and graphics in movies.

- Utah teapot
- Stanford bunny

In the 90s fully animated CG movies and special effects. 00s Rise of the GPU's, start to be able to program and do custom things on GPUs. A move toward physically based rendering, start to use more correct models of illumination thanks to better computational recourses. 10s saw fusion with Image processing and computer vision.. Film rendering to **path tracing**. Real-time **ray tracing**. 20s machine learning?? this would mean new principles. Neuro .... read more of own intrest

5 Quick look at what we will learn

- Transformation
  - Affine transformations in homogenous coordinates
  - Orthographic and perpspective projecteions for cameras.
- Shading
  - Going from geometry to what color to put
  - Gouraud shading (computed per-vertex)
  - Phong cont...
- Illumination
  - The Phon refleciton model
  - BLinn-Phong
- Texure mapping
  - Texture mapping
  - Bump mapping
  - Environment mappong
  - Geometry
  - Normal
- The programmable **Graphics pipeline**

6 Pipeline

- Input are the objects and their vertices
- The output are the pixels on the screen
- Performs:
  - transformations
  - clippin and assembly
  - shadding, textureing and illumination

7 Conclusion for this lecture

Study the slides but also the other recoursces. Prepare for tbe practicals (assignment and prjoject) and start to play around with graphics programming.

Lecture 2: Graphics programming

tuesday 17 jan 10:15

8 Graphics programming

Typicalli deals with how to define 3d scene, with virtual camera, how to create a 2d projection of the 3d scene

Real time vs ofline

real time used in games, scene must be updated 30-60 fps speed > im quality, but today hardware can do pretty much both

offline is used for animatied movies and visual effects. not meant to be used interactivly rendering a single frame can take several hours image qualite > speed. Movie production uses clusters to calculate the renders in pareallell

how do we draw

version 1 for every pixel on the screen query all objects to be drawn there is a outher loop - ray tracing

version 2 iterate over objects for every object draw the objects to the screen-rasterization dont need to keep the whole scen in memory

Representation

typically used for represent 3d models, vertices, edges and faces. faces can be arbitrary polygons. Why do we split them in triangels? The implementation gets simpler

vertex data

polygonal mesh has one or several attributes

- Position
- Color
- Normal vector
- texture coordinate

vertex data is loaded on the cpu uploaded to the gpu correct

transform

Use transforms to maniuplate positions orentation use transforms to define our virtual camera basic transforms:

- scale ... cont..

Cooerdinate systms

THe OpenGL pipeline has 6 differnte systems

- Object
- World
- Eye (or camera)
- Clip
- Nomalized devices
- Window (or screen)

Surface normals

a normal vector that points outwards from the surface.. descrives a local orientation. super important to lighting. Useful to know if the surface is pointing towards me or not

Can visualize it with RGB vector for example.

Shaderds

shader is a small program that is commpiled on the cpu and exectuted on the GPU. Common use is to apply fransfromations on vertices and compute cont..

pipeline

vertives -> sent to gpu -> assign attributes -> sent to clipper (remove what is not seen) and primitive assembler (takes vertices and conncect them to triangles) -> rasterized (converted into pixels or fragments) -> fragmen shader

We have controll of the vertices and the fragment shader from the cpu

OpenGL

statebased maintain a currents state of things. what shader is being used etc. openGL for desktop openGL ES for embedded WebGl for

it is a state machine

OpenGL functions

executed in the host CPU application. It is responsible for creating and init buffers, shaders, tecture etc. uploading data to gpu

The CPU and GPU

sending objects complicated can become the bottleneck between the cpu and gpu