Image Processing and Computer Graphics

Summary

Matthias Teschner

Computer Science Department
University of Freiburg

Outline

- rendering pipeline
- homogeneous coordinates and transformations
- lighting models
- shading models
- rasterization
- texture mapping
- shadows
- transparency
- reflections
- outlook

Rendering Pipeline

- 3D input
  - a virtual camera
    - position, orientation, focal length
  - objects
    - points (vertex / vertices), lines, polygons
    - geometry and material properties
      (position, normal, color, texture coordinates)
  - light sources
    - direction, position, color, intensity
  - textures (images)
- 2D output
  - per-pixel color values in the framebuffer

Main Stages

- vertex processing / geometry stage / vertex shader
  - processes all vertices independently in the same way
  - performs transforms per vertex, computes lighting per vertex
- primitive assembly and rasterization / rasterization stage
  - assembles primitives such as points, lines, triangles
  - converts primitives into a raster image
  - generates fragments / pixel candidates
  - fragment attributes are interpolated from vertices of a primitive
- fragment processing / fragment shader
  - processes all fragments independently in the same way
  - fragments are processed, discarded or stored in the framebuffer

Vertex Processing / Geometry Stage

object space

$\Downarrow$ modelview transform

eye space / camera space

$\Downarrow$ lighting, projection

clip space / normalized device coordinates

$\Downarrow$ clipping, viewport transform

window space

Primitive Assembly / Rasterization

- primitive assembly
  - vertex and primitive information are combined for further processing of points, lines and triangles
- geometry shader
  - change, delete, generate primitives
- rasterization
  - converts primitives into fragments
  - computes positions of screen pixels that are affected by a primitive
  - generates a fragment for each affected pixel position
  - interpolates attributes from vertices to fragments

Fragment Processing

- fragment attributes are processed and tests are performed
  - fragment attributes are processed
  - fragments are discarded or
  - fragments pass a test and finally update the framebuffer
- processing and testing use
  - fragment attributes
  - textures
  - additional data that is available for each pixel position
    - depth, color, stencil, accumulation buffer

Outline

- rendering pipeline
- homogeneous coordinates and transformations
- lighting models
- shading models
- rasterization
- texture mapping
- shadows
- transparency
- reflections
- outlook

Homogeneous Coordinates

- $(x, y, z, w)^T$ with $w \neq 0$ are the homogeneous coordinates of the 3D point $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^T$
- $(x, y, z, 0)^T$ is a point at infinity in the direction of $(x, y, z)^T$
- $(x, y, z, 0)^T$ is a vector in the direction of $(x, y, z)^T$
- $\begin{pmatrix} m_{00} & m_{01} & m_{02} & t_0 \\ m_{10} & m_{11} & m_{12} & t_1 \\ m_{20} & m_{21} & m_{22} & t_2 \\ p_0 & p_1 & p_2 & w \end{pmatrix}$ is a transformation, representing rotation, scale, translation, projection
- if a surface is transformed by **A**, its normal is transformed by $(\mathbf{A}^{-1})^T$

UNI FREIBURG

Motivation

- uniform handling of
  - affine transformations and perspective projections
  - modelview transformation
  - projection transformation
  - viewport transformation
  - combinations of transformations
- uniform handling of
  - points
  - vectors / points at infinity

Outline

- rendering pipeline
- homogeneous coordinates and transformations
- lighting models
- shading models
- rasterization
- texture mapping
- shadows
- transparency
- reflections
- outlook

Radiometry

- radiant energy
- radiant flux / radiant power
- irradiance / radiant exitance / flux density
- radiant intensity
- radiance
- inverse square law
- conservation of radiance
- Lambert's cosine law

Lighting Models

- Phong illumination model
  - Phong / Blinn-Phong / normalized
  - computation only uses local information
  - diffuse, specular, ambient components
- attenuation
- fog
- light sources

Outline

- rendering pipeline
- homogeneous coordinates and transformations
- lighting models
- shading models
- rasterization
- texture mapping
- shadows
- transparency
- reflections
- outlook

Shading Models

- flat shading / constant shading
  - evaluation of the lighting model per vertex
  - primitives are colored with the color of one specific vertex
- Gouraud shading
  - evaluation of the lighting model per vertex
  - primitives are colored by bilinear interpolation of vertex colors
- Phong shading
  - bilinear interpolation of vertex normals during rasterization
  - evaluation of the lighting model per fragment

Shading Models

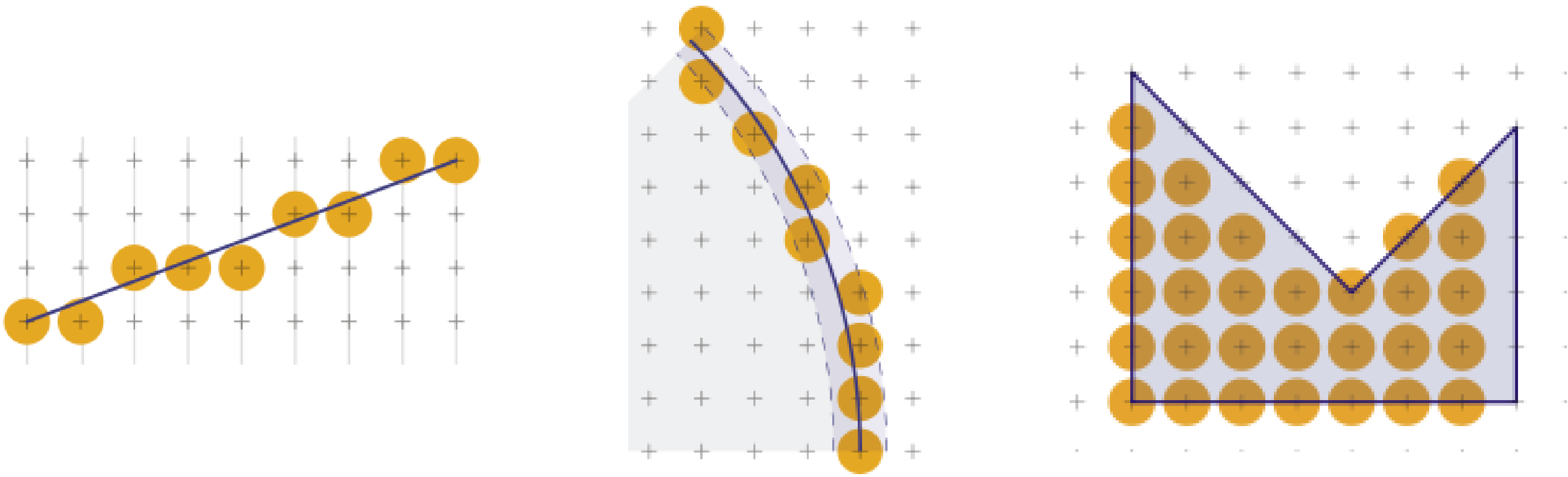
- flat shading / constant shading
  - simplest, fastest
- Gouraud shading
  - more realistic than flat shading for the same tessellation
  - suffers from Mach band effect
  - local highlights are not resolved, if the highlight is not captured by a vertex
- Phong shading
  - highest quality, most expensive

Outline

- rendering pipeline
- homogeneous coordinates and transformations
- lighting models
- shading models
- rasterization
- texture mapping
- shadows
- transparency
- reflections
- outlook

Rasterization

- computation of pixel positions that represent a primitive



[Wikipedia: Rasterung von Linien, Rasterung von Polygonen, Rasterung von Kreisen]

Lines / Circles / Polygons

- line / circle rasterization algorithms are usually described for a subset of lines / circles and generalized using symmetries
- incremental updates are often employed
- Bresenham avoids floating-point arithmetic
- polygon rasterization
  - estimates pixel positions inside a polygon
  - works for closed polygons
  - pixels on shared edges should be rasterized exactly once

Outline

- rendering pipeline
- homogeneous coordinates and transformations
- lighting models
- shading models
- rasterization
- texture mapping
- shadows
- transparency
- reflections
- outlook

Texturing

- adding per-fragment surface details without raising the geometric complexity of a scene
- textures can influence a variety of properties
- textures can be 1D, 2D, 3D, ..., or procedural
- texture coordinates at vertices or fragments are usedto lookup texels
- quality of applied textures can be improved by
  - perspective-correct interpolation
  - considering magnification and minification
- examples
  - color, alpha, environment, light, bump, parallax

UNI FREIBURG

Outline

- rendering pipeline
- homogeneous coordinates and transformations
- lighting models
- shading models
- rasterization
- texture mapping
- shadows
- transparency
- reflections
- outlook

Shadow Algorithms

- determine shadowed surfaces in a scene
  - projection shadows
  - shadow mapping
  - shadow volumes
- concepts
- implementations / rendering passes
- limitations

Shadow Algorithms

- projection shadows
  - restricted to planar receivers
  - no self-shadowing
- shadow maps
  - image-space technique, two rendering passes
  - works correct, if all relevant objects are "seen" by the light
  - sampling issues
- shadow volumes
  - requires a polygonal representation of the shadow volume
  - multiple rendering passes
  - clipping of shadow volume polygons has to be addressed

Outline

- rendering pipeline
- homogeneous coordinates and transformations
- lighting models
- shading models
- rasterization
- texture mapping
- shadows
- transparency
- reflections
- outlook

Transparency

- concepts
  - stipple patterns (screen-door transparency)
  - color blending (depth peeling, BSP trees)
- depth ordering
  - convex objects
  - arbitrary objects
    - object space (BSP trees)
    - screen space (depth peeling)
- BSP trees
  - generation and query
- depth peeling
  - shadow-mapping implementation

Reflection

- planar surfaces
  - implementation with / without stenciling
- arbitrary surfaces
  - environment mapping
  - representation of the environment texture
  - properties / limitations

Outline

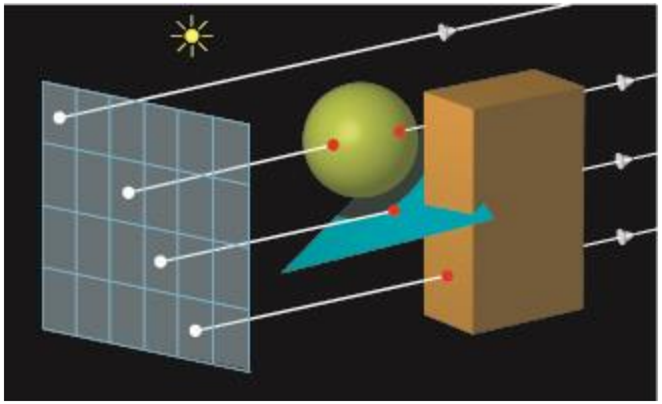
- rendering pipeline
- homogeneous coordinates and transformations
- lighting models
- shading models
- rasterization
- texture mapping
- shadows
- transparency
- reflections
- outlook

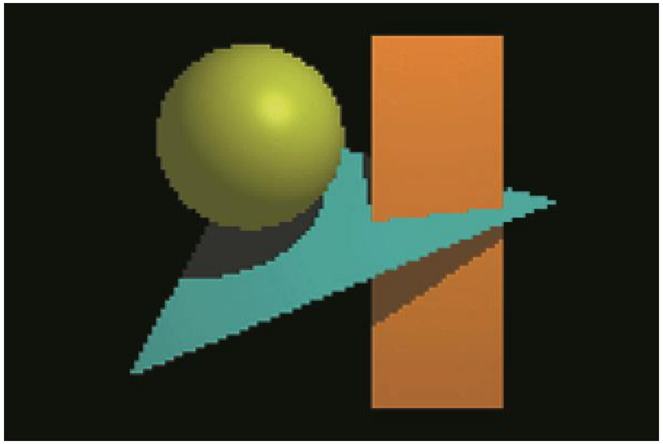
Specialization Course Advanced Computer Graphics

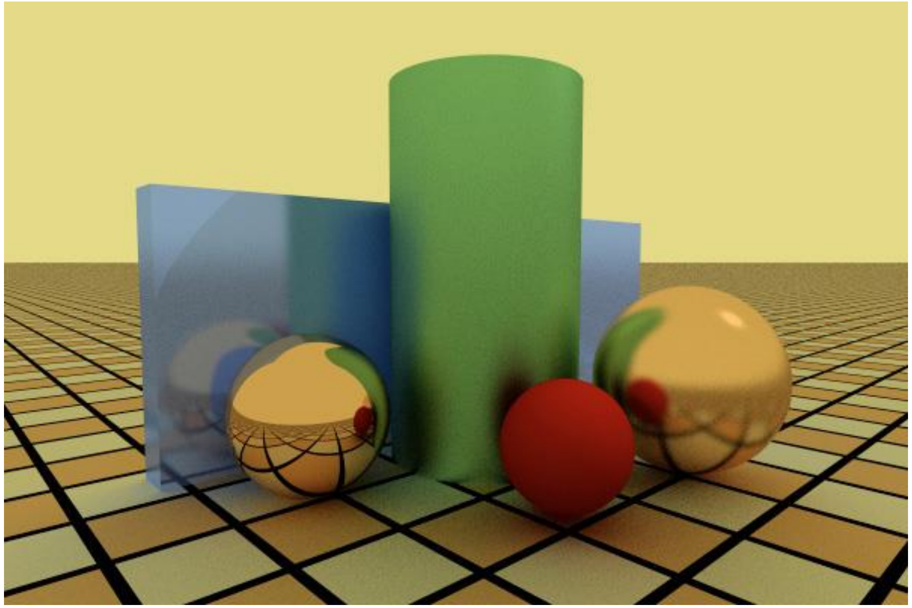
- ray tracing
  - global illumination model
  - trace rays from a camera
- ray-object intersection
  - spatial data structures
- light distribution
  - direct and indirect illumination
- surface scattering
  - interaction of light with surfaces
- recursive ray tracing
  - trace additional rays for surfaces

Specialization Course
Advanced Computer Graphics

- ray tracing







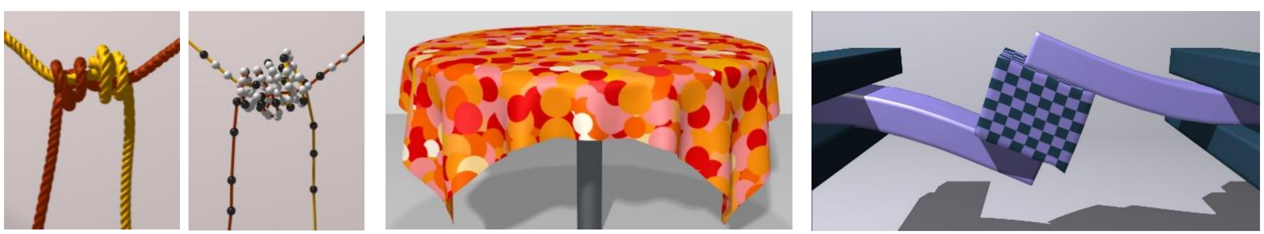
[Kevin Suffern: Ray Tracing from the Ground Up]

Specialization Course
Simulation in Computer Graphics

- physically-based animation of the dynamics of
  - rigid bodies, deformable objects, fluids
- collision handling

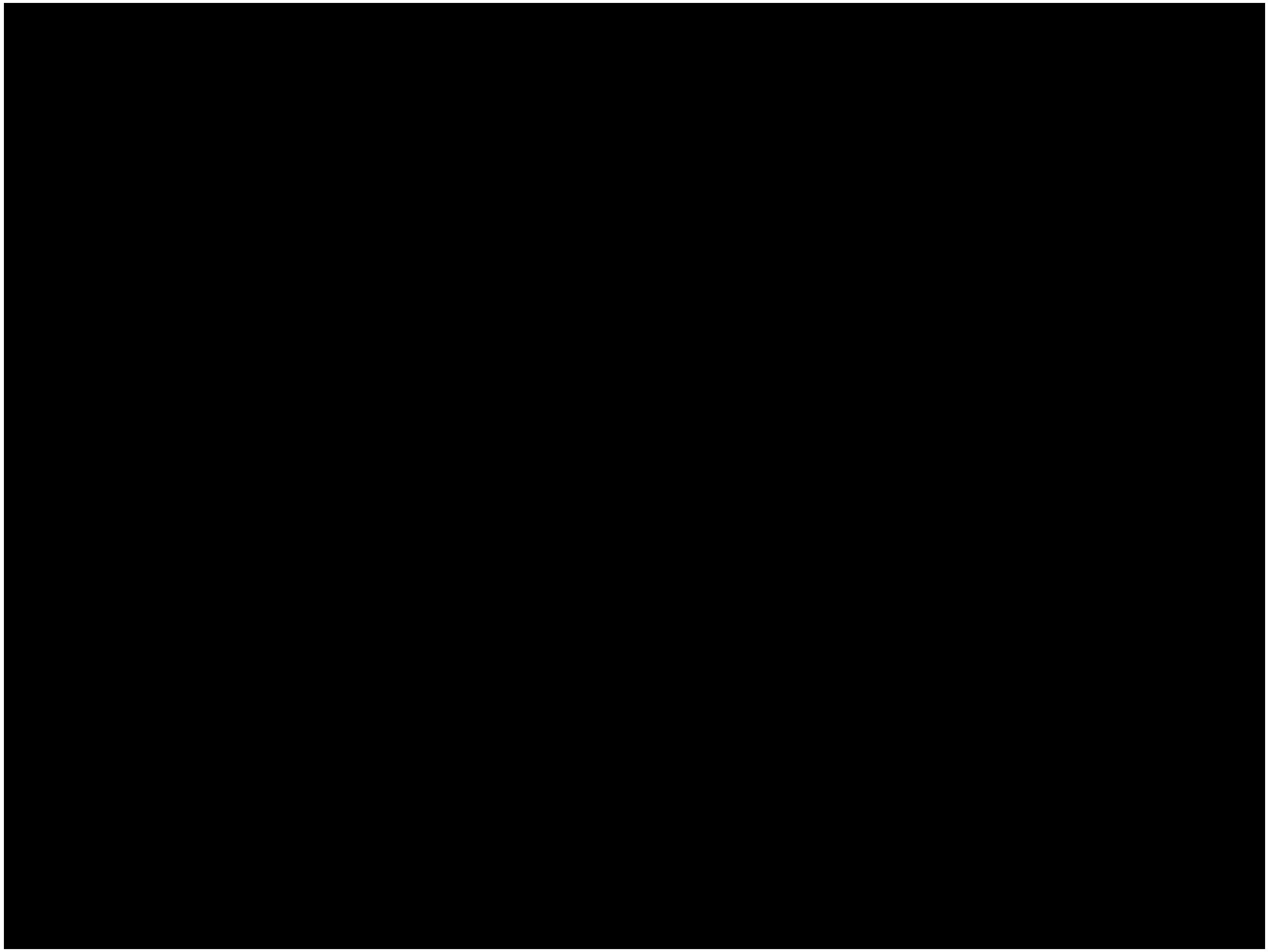
rigid bodies

fluids



deformable objects – 1D, 2D, 3D

Specialization Course
Simulation in Computer Graphics



www.youtube.com / cgfreiburg