# Computer Graphics

Dr. rer. nat. Martin Möddel

April 6, 2021

Institut für Biomedizinische Bildgebung

# Rendering

# Definition

Rendering refers to the process of creating 2D raster images from 2D or 3D models using computer programs.

Rendering is grouped into two different algorithmic families

**Object ordered rendering**
Each object is considered, one at a time, and the pixels in the image influenced by the object are found and updated.

**Image ordered rendering**
Each pixel is considered, one at a time, and the objects influencing this pixel are found. The pixels value is updated accordingly.

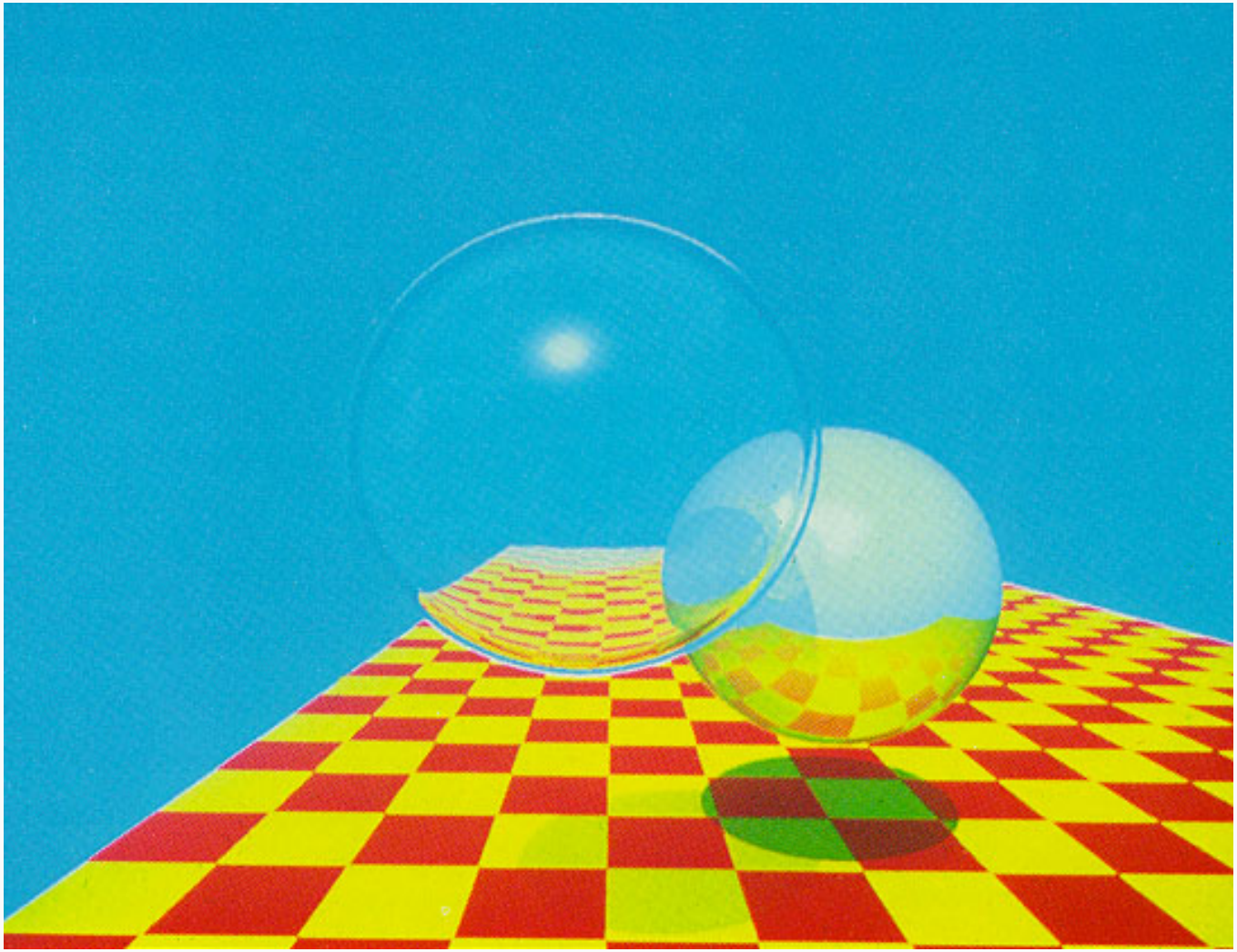
# Remarks

Algorithmically the difference between the two can be described by the order of the nested loops. In OOR the **object loop** is outer and the **pixel loop** is inner, while in IOR the order is reversed.

- In theory both algorithmic families compute the same raster image
- In practice that is not true, due to simplifications made within each family
- Sophisticated rendering models that are complicated to implement in OOR are most often quite simple in IOR
- IOR is thus algorithmically simpler but requires much more execution time

# Ray Tracing

# Ray Tracing



# Ray Tracing

# Definition

A ray tracer is an IOR algorithm, where a ray is traced into a scene until it hits a scene object. Information on hit point and object are used to color image pixels. A basic ray tracer has three parts

1. Ray generation: compute origin and direction of pixels viewing ray based on camera geometry and orientation
2. Ray intersection: Find closest object intersecting the viewing ray
3. Shading: Compute pixel color based on the result of the ray intersection

# Algorithm

```
1: A: a scene (collection of objects)
2: Shader: program to calculate pixel values
3: procedure traceRays(A,Shader)
4:     for each pixel do
5:         compute viewing ray
6:         find nearest object hit by ray in scene A
7:         calculate pixel color using the Shader
8:         set pixel color
9:     end for
10: end procedure
```

# Rays

**Definition**
A ray is a 3D parametric line

$$\mathrm{p}(t) = \mathrm{o} + t(\mathrm{s} - \mathrm{o}). \tag{1}$$

Here o is the origin of the ray, $\mathrm{d} = \mathrm{s} - \mathrm{o}$ is the direction and $t \in \mathbb{R}$ is the fractional distance of $\mathrm{p}(t)$ to o.

- $p(0) = \mathrm{o}$ and $\mathrm{p}(1) = \mathrm{s}$
- If $0 < t_1 < t_2$ then $p(t_1)$ is closer to the origin o than $\mathrm{p}(t_2)$
- If $t < 0$ then $\mathrm{p}(t)$ is behind o

# Camera

Ray generation starts within the camera coordinate system, which is defined by

- e view point (eye point)
- u rightwards direction of the camera
- v upwards direction of the camera
- w backwards direction of the camera

$\{u, v, w\}$ forms a right-handed coordinate system. Usually only the upward and backward directions are provided, since $u = v \times w$.

# Orthographic View

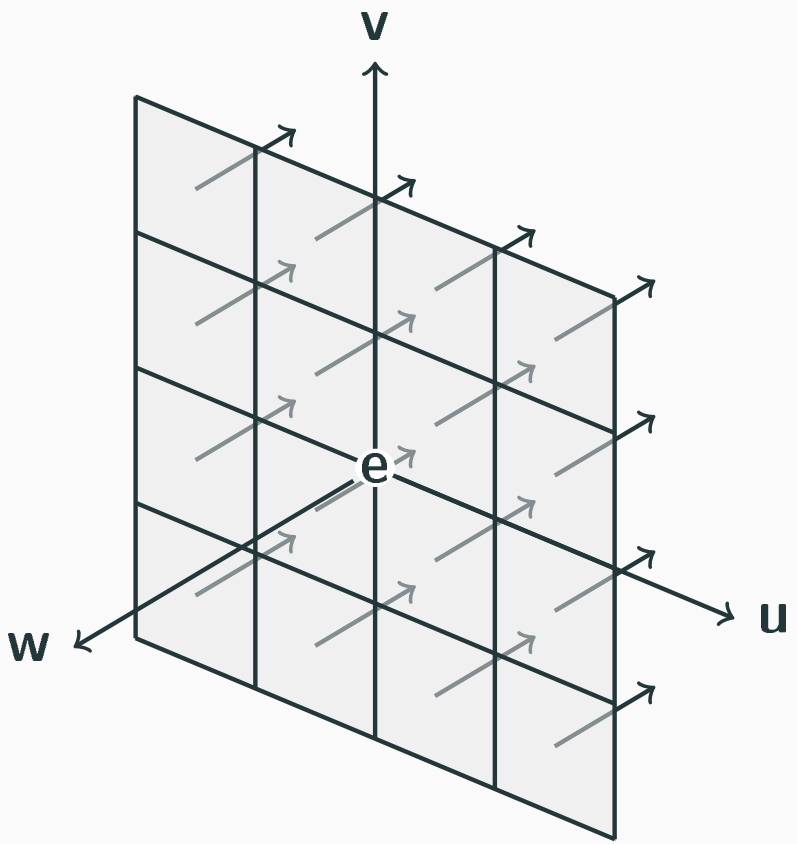
- each ray points towards $\mathrm{d} = -\mathrm{w}$
- each ray should starts on the plane spanned by u and v containing e
- The image size is defined by the left, right, bottom and top edges of the image $l < 0 < r$, $b < 0 < t$
- A canonical choice of the edges is given by $r = t = 1$ and $l = b = -1$
- With $n_x \times n_y$ pixels the ray (i,j) has origin $\mathrm{o} = \mathrm{e} + u\mathrm{u} + v\mathrm{v}$, where

$$u = l + (r - l)(i + 0.5)/n_x \tag{2}$$

$$v = t + (b - t)(j + 0.5)/n_y. \tag{3}$$

**Remark**
Both o and $(\mathrm{s} - \mathrm{o})$ are then given in the global coordinate system

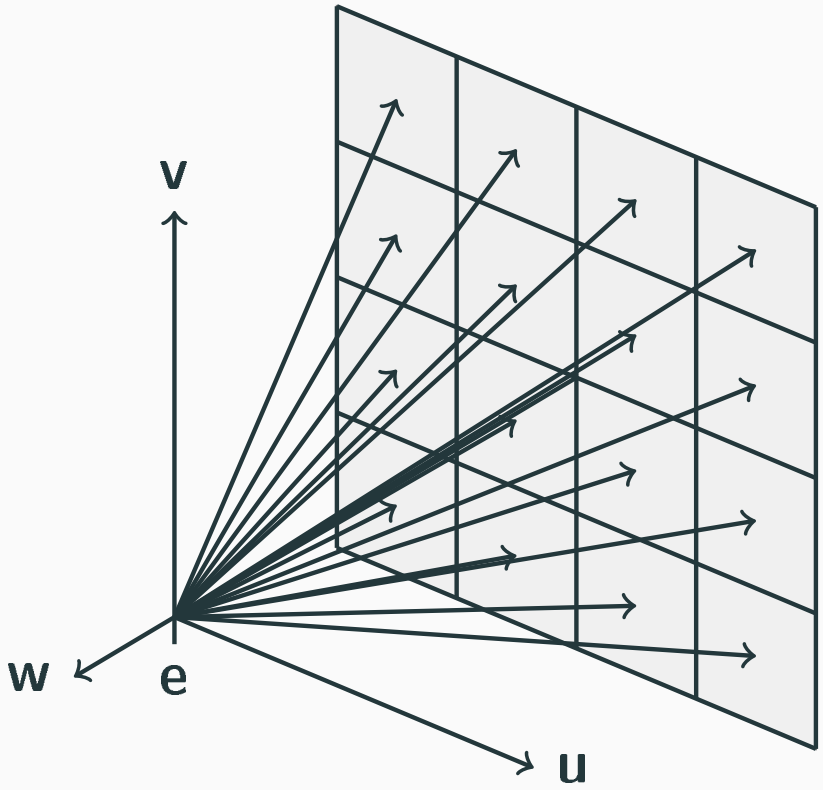
# Orthographic View



# Perspective View

- The ray origin is $\mathrm{o} = \mathrm{e}$ for all rays
- The image plane is positioned at $-d\mathrm{w}$, i.e. some distance d in front of e
- One sometimes refers to $d$ as focal length
- The viewing direction is given by vector pointing from e to the pixel on the image plane $\mathrm{d} = -d\mathrm{w} + u\mathrm{u} + v\mathrm{v}$ with $u$ and $v$ defined as in the orthographic view
- Usually the ray origin is shifted by d, such that the ray origin is on the image plane and the ray direction is normalized, such that the parameter $t$ denotes the Euclidean distance from the image plane

# Perspective View



# Ray Intersection

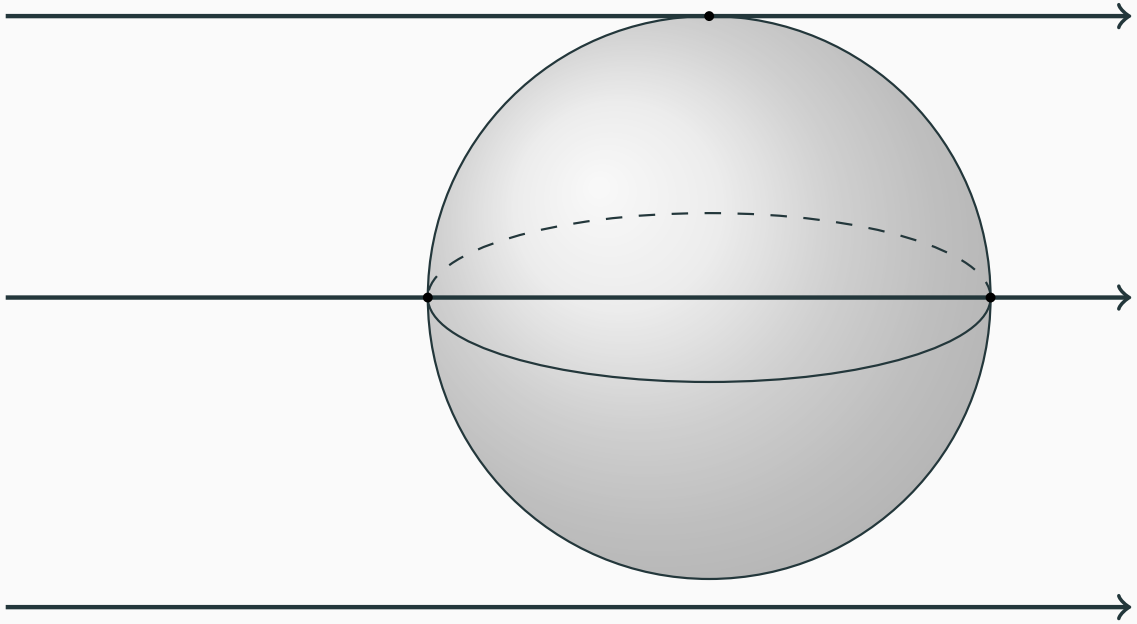
Apart from ray generation, intersection methods are an integral part of ray tracing, though they find their use in many other applications in computer graphics. Intersection methods should answer the following questions:

- Is my ray hitting any object?
- Are the objects in front of my camera, on its back or surrounding the camera?
- Which object is closest in front of the camera?
- Where does the object get hit?
- Where does the normal of the surface points towards?

# Ray Intersection

If one tries to find the intersection between an object and a ray it is often simpler to not consider the object as a whole, but only its surface. To mathematically describe surfaces there are essentially two possible ways, both of which are used in computer graphics:

- Implicit surfaces
- Explicit surfaces



# Implicit Surfaces

Let $f : \mathbb{R}^3 \to \mathbb{R}$ be a function mapping points to the real numbers, then the kernel of $f$ defines an implicit surface

$$\left\{ \mathrm{x} \in \mathbb{R}^3 | f(\mathrm{x}) = 0 \right\}. \tag{4}$$

An example is the sphere of radius $r$, which is implicitly given by the function $f(\mathrm{x}) = \|\mathrm{x}\|_2^2 - r^2$.

# Ray Intersection with Implicit Surfaces

Let $\mathsf{p}(t) = \mathsf{o} + t\mathsf{d}$ be a ray and $f : \mathbb{R}^3 \to \mathbb{R}$ be a function describing an implicit surface. The set of hit points between ray and surface is then given by

$$\{\mathsf{p}(t) | t \in \mathbb{R} \text{ such that } f(\mathsf{o} + t\mathsf{d}) = 0\}. \tag{5}$$

**Note**
The intersection problem boils down to a one dimensional root-finding problem, which can be solved numerically, though that might be quite demanding.

# Normal Vectors of Implicit Surfaces

Let $f : \mathbb{R}^3 \to \mathbb{R}$ be a smooth function defining an implicit surface and x be a point on the surface. The normal vector of the surface at this point is given by the gradient of the function at this point

$$\mathrm{n} = \nabla f(\mathrm{x}). \tag{6}$$

**Example**
For $f(\mathrm{x}) = \|\mathrm{x}\|_2^2 - r^2$ we have

$$\nabla f(\mathrm{x}) = (2x, 2y, 2z). \tag{7}$$

# Explicit Surfaces

An explicit surface on the other hand is given by the range of a parameter function $\mathrm{f} : P \to \mathbb{R}^3$

$$\{\mathrm{f}(\mathrm{x}) | \mathrm{x} \in P\}\,. \tag{8}$$

An example is again the sphere of radius $r$, where

$$\mathrm{f}(\varrho, \varphi) = (r \sin\varrho \cos\varphi, r \sin\varrho \sin\varphi, r \cos\varrho), \tag{9}$$

$(\varrho, \varphi) \in P = [0, \pi) \times [0, 2\pi)$.

# Ray Intersection with Explicit Surfaces

Let $\mathsf{p}(t) = \mathsf{o} + t\mathsf{d}$ be a ray and $\mathsf{f} : P \to \mathbb{R}^3$ be a function describing an explicit surface. The set of hit points between ray and surface is then given by

$$\{\mathsf{f}(\mathsf{x}) | \exists (t, x) \in \mathbb{R} \times P \text{ such that } \mathsf{p}(t) = \mathsf{f}(\mathsf{x})\}. \tag{10}$$

**Note**

- The intersection problem can be reformulated into a multi dimensional root finding problem

$$\{(t, \mathsf{x}) | \mathsf{F}(t, \mathsf{x}) = \mathsf{p}(t) - \mathsf{f}(\mathsf{x}) = 0\}. \tag{11}$$

- The hit points are then simply given by $\mathsf{f}(\mathsf{x})$ for all $(t, \mathsf{x}) \in \{(t, \mathsf{x}) | \mathsf{F}(t, \mathsf{x}) = 0\}$

# Normal Vectors of Explicit Surfaces

Let $\mathsf{f} : P \subseteq \mathbb{R}^2 \to \mathbb{R}^3$ be a smooth function defining an explicit surface over the open subset $P$ and $\mathsf{p} = (p_1, p_2) \in P$ be the parameter mapping to the surface location $\mathsf{x} = \mathsf{f}(\mathsf{p})$ then the surface normal is orthogonal to the partial derivatives spanning the tangent space

$$\mathsf{n} = \partial_{p_1}\mathsf{f}(\mathsf{p}) \times \partial_{p_2}\mathsf{f}(\mathsf{p}). \tag{12}$$

**Example**

$$\partial_\varrho \mathsf{f}(\varrho, \varphi) = r(\cos\varrho\cos\varphi, \cos\varrho\sin\varphi, -\sin\varrho) \tag{13}$$

$$\partial_\varphi \mathsf{f}(\varrho, \varphi) = r(-\sin\varrho\sin\varphi, \sin\varrho\cos\varphi, 0) \tag{14}$$

$$\partial_\varrho \mathsf{f}(\varrho, \varphi) \times \partial_\varphi \mathsf{f}(\varrho, \varphi) = r^2 \sin\varrho(\sin\varrho\cos\varphi, \sin\varrho\sin\varphi, \cos\varrho) \tag{15}$$