Lecture 2: Introduction to Computer Graphics

Representing geometry, cameras, reflectance, lighting, and rendering images

Respond at **PollEv.com/ronisen**

Text **RONISEN** to **22333** once to join, then text your message

Feel free to share your questions...

Few reminders

- 590: Assignment 1 is out, due date next Thursday Aug 25!
- 790: Starting planning your project proposal and forming your group. If you want to work on your own project, send an email, explain why, and have my written approval.
- 590/790: Please indicate your paper presentation preference by filling out the google form (See course website, under presentation).
- Change in grading plans:
  - Paper presentation: only 790
  - Paper review: only 590
  - 590: 5 assignments instead of 4, but significantly easier, 4 assignment can be done on google colab.

How does Computer Vision & Graphics work together?

Current Image

Vision

Edit Components

Graphics

New Image under different conditions

3D Intrinsic Components

Explicit: Reconstruct 3D
(Introduction to Graphics Lectures)

Implicit: Neural Representation
(Generative Models Lectures

Change:

- Viewpoint
- Lighting
- Reflectance
- Background
- Attributes
- Many others…

Agenda

- How do we define geometry/shape of an object?
- How do we define a camera model? – 3D object to 2D image
- How do we define material property? – glossy, metallic

Slide Credits

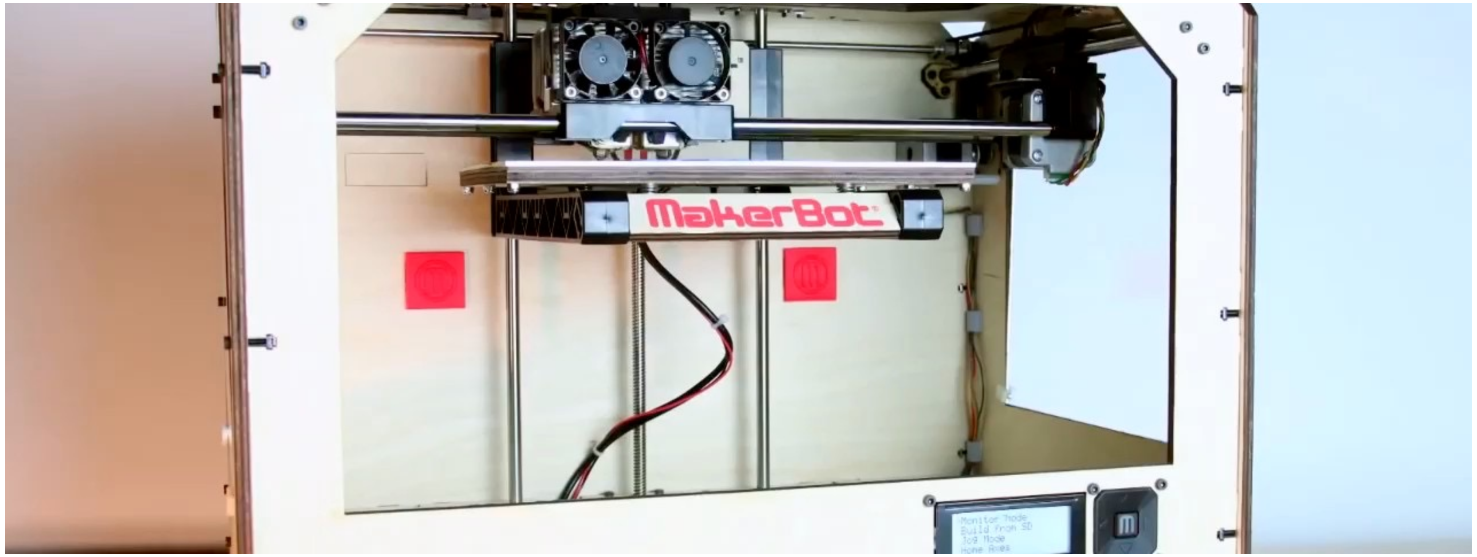
- UC Berkeley CS 184/284a – Spring 2021 (Ren Ng, Angjoo Kanazawa)
- CMU 16-385 Computer Vision – Spring 2017 (Kris Kitani)
- Many amazing research papers!

Agenda

- How do we define geometry/shape of an object?
- How do we define a camera model? – 3D object to 2D image
- How do we define material property? – glossy, metallic

Digital Geometry Processing

3D Scanning



3D Printing

Geometry Processing Pipeline

Scan → **Process** → **Print**

How do we represent geometry?

Geometry: How do we represent shape of an object?

2.5D representation:

1) Depth & Normal map

Explicit representation:

2) Mesh
3) Voxels
4) Point Cloud

Implicit representation:

5) Surface Representation (SDF)

Geometry: How do we represent shape of an object?

2.5D representation:

1) Depth & Normal map

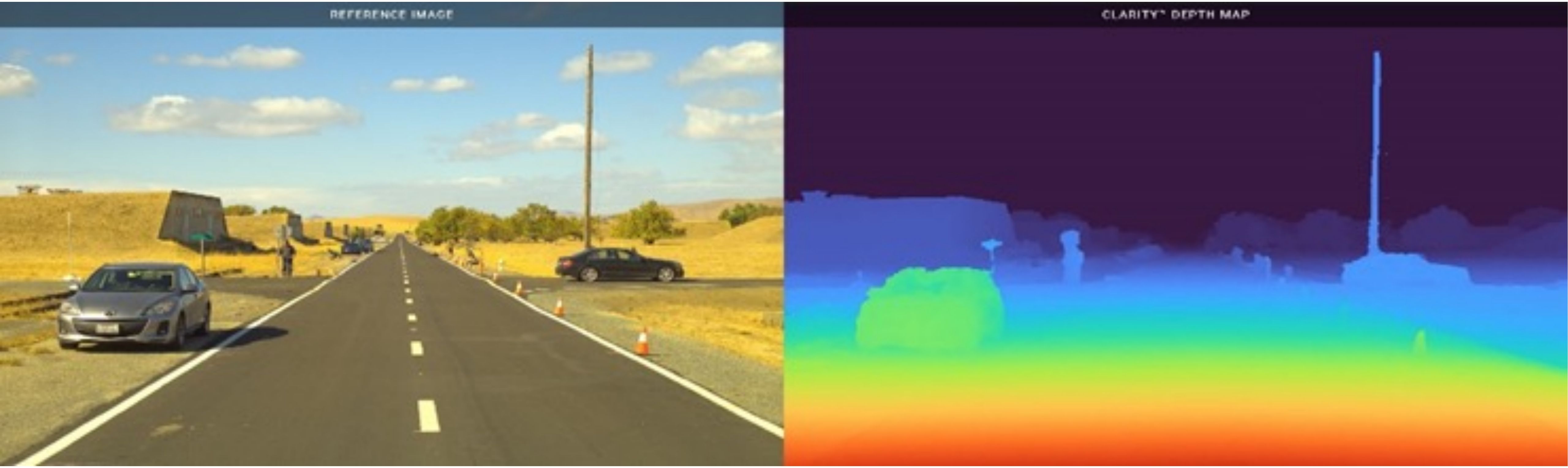
Explicit representation:

2) Mesh
3) Voxels
4) Point Cloud

Implicit representation:

5) Surface Representation (SDF)

Depth Map

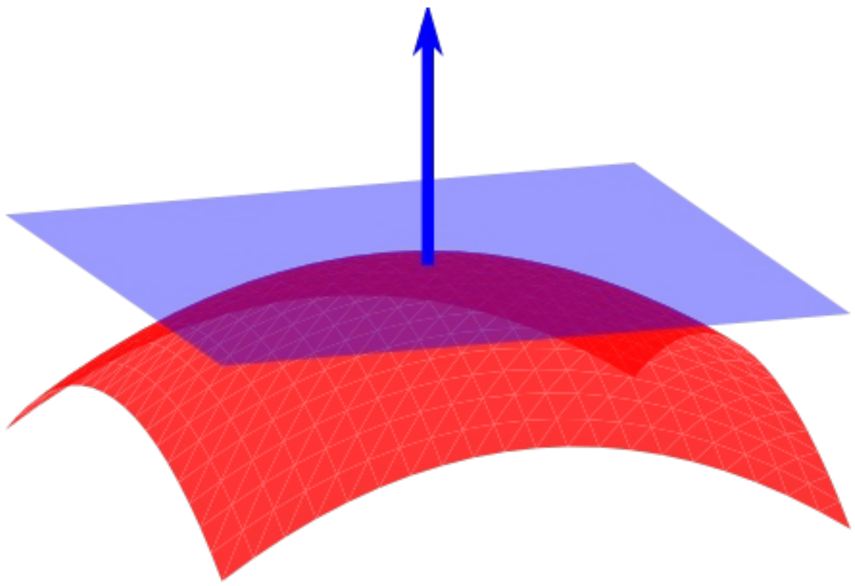


Depth Map D(u,v): Distance of any pixel (u,v) from the camera (usually image plane)

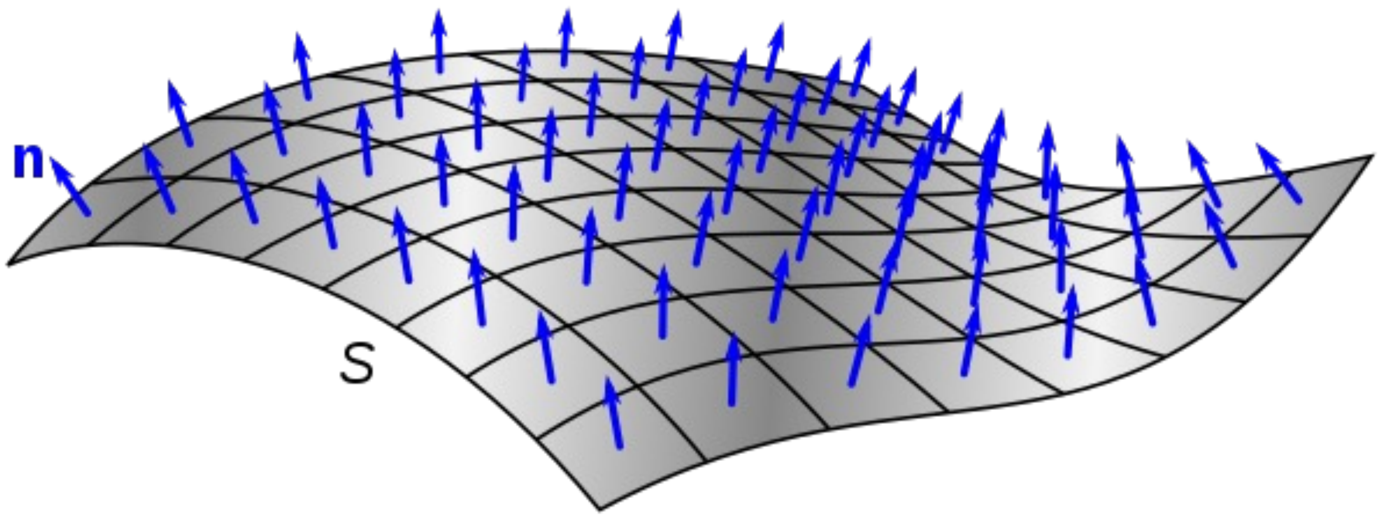
Red-> nearer; blue-> further

For an image HxWx3, a depth map is HxWx1 (scalar value for every pixel)

Surface Normal



Surface Normal (in blue) of a point P is a vector perpendicular to the tanget plane at P.



Surface normal (in blue) of a surface

Surface normal indicate orientation of the surface.

Normal Map

Normal Map N(u,v): [Nx,Ny,Nz] is a unit vector indicating the orientation of the surface.

Pink-> towards left; blue-> towards right

For an image HxWx3, a normal map is HxWx3.

What is a unit vector?

- L2 norm (magnitude) of the vector is 1.
- $Nx^2 + Ny^2 + Nz^2 = 1$

Credits: ibug imperial college london

Relationship between Depth & Normal Map

$$\tilde{N} = [\frac{\partial D}{\partial x}, \frac{\partial D}{\partial y}, -1]$$

$$N = \frac{\tilde{N}}{\|\tilde{N}\|_2}$$

Normalizing to unit vector.

- Differentiation of depth map leads to normal map
- Integration of normal map leads to depth map

Further reading: Normal Integration: A Survey

Geometry: How do we represent shape of an object?

2.5D representation:

1) Depth & Normal map

Explicit representation:

2) Mesh

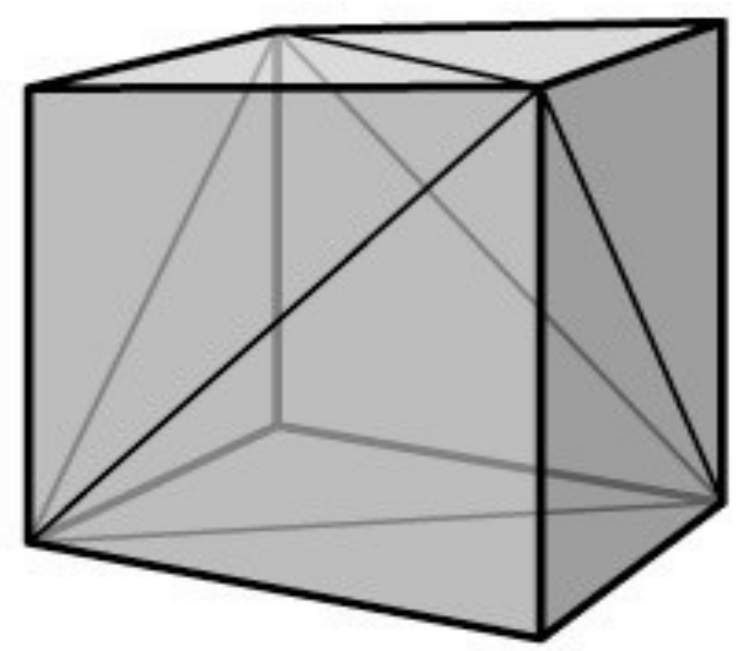
3) Voxels

4) Point Cloud

Implicit representation:

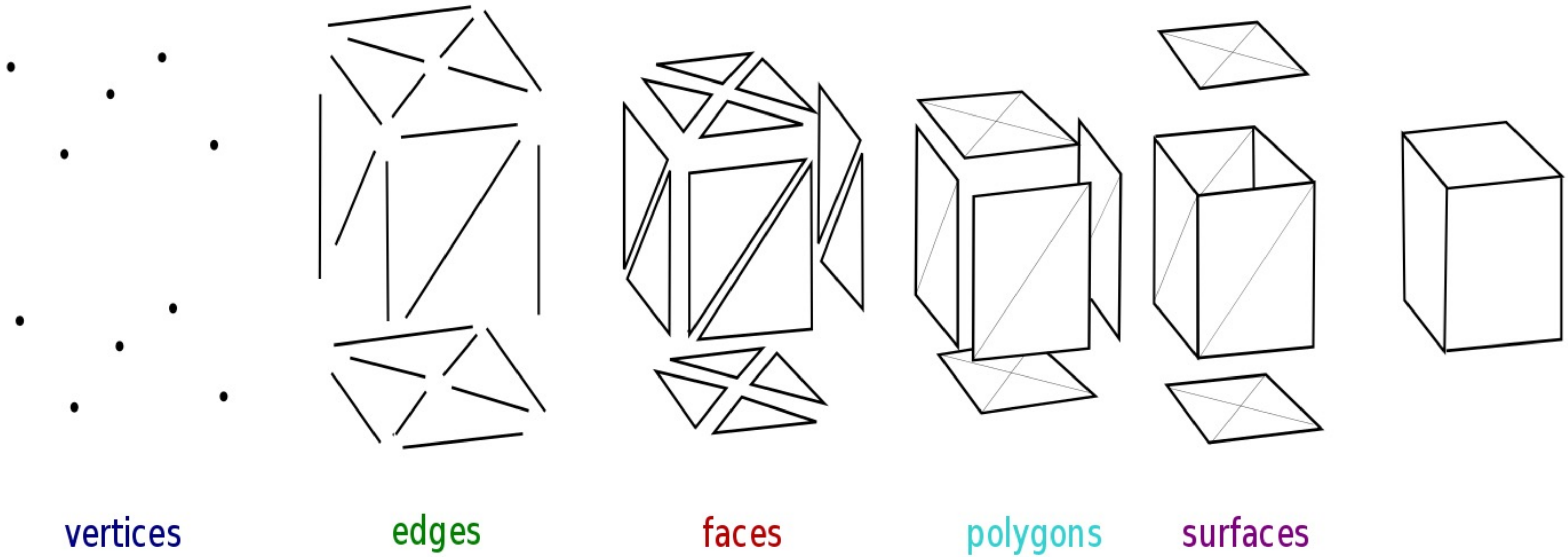
5) Surface Representation (SDF)

A Small Triangle Mesh



8 vertices, 12 triangles

Mesh



How do you represent a mesh file?

Wavefront .obj file

A list of vertices defined by their 3D x,y,z coordinates

```
v -0.238769205544999864 1.310379727060l687 0.130012607000009193
v -0.27582915374543276 1.258256333l865875 0.12364597630502337
v -0.26748883360l6338 1.347437322575l202 0.159127474597429766
v -0.31286627569804O7 1.22271321683485 2 0.146235655433O1947
```

A list of faces that defines which vertices will combine to produce a triangle on the mesh

```
f 1 2 3
f 4 5 3
f 4 3 2
f 9 10 7
f 10 8 7
f 11 12 9
```

Note: This is the most naïve way of defining a mesh. You can add vertex normal, vertex texture, separate material model, and many other things with the .obj format.

MeshLab: a great software to load and visualize a mesh in 3D!

Show demo of using MeshLab to view an object in 3D

Texture Mapping: How do you add color/texture on a mesh?

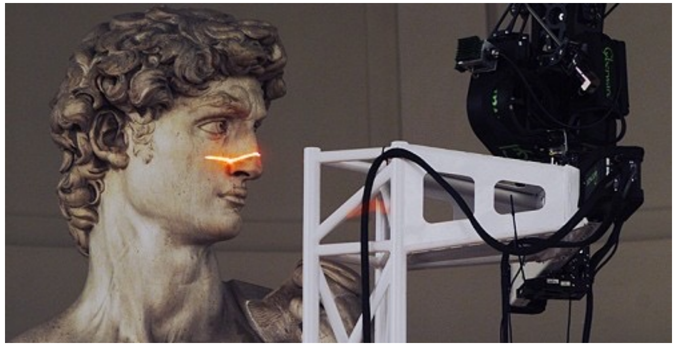
- Texture map is a 2D image
- Texture mapping takes the RGB color of each pixel (u,v) from the texture map and colors a vertex V (x,y,z) of the mesh.
- Color of each faces (triangles of the mesh) are often interpolated between the 3 vertex colors
- Note: Many variation of the above algorithm exist.

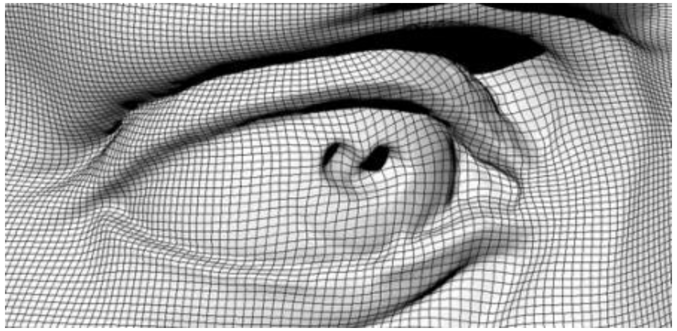
Credits: wiki

Few important/cool research works on meshes

A Large Triangle Mesh

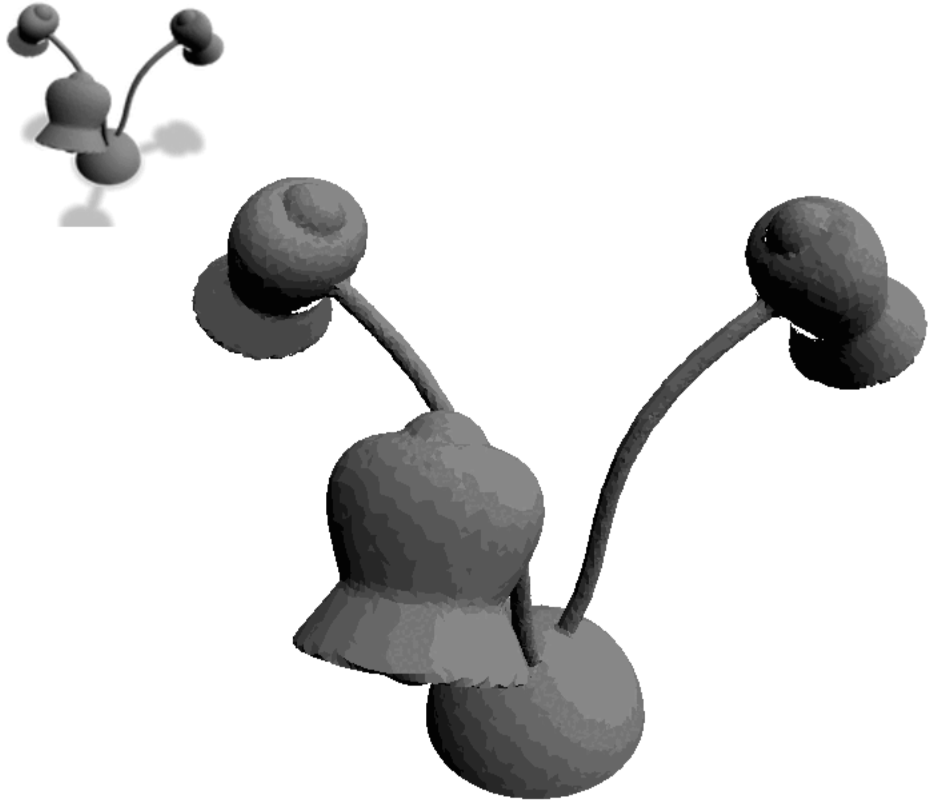
David
Digital Michelangelo Project
28,184,526 vertices
56,230,343 triangles



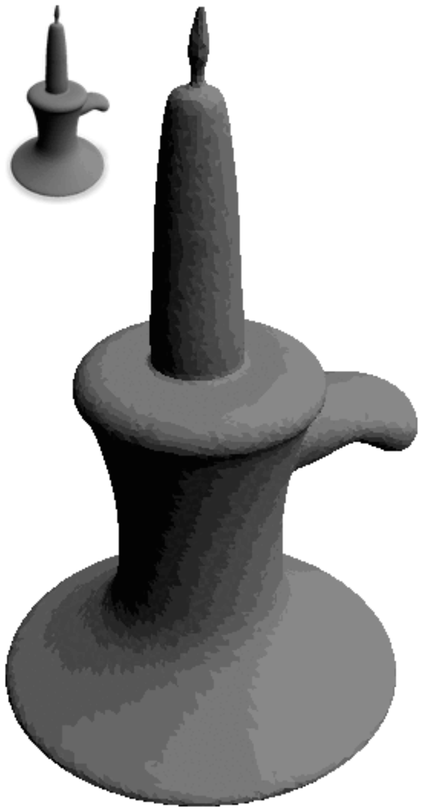


Marc Levoy at Stanford (https://accademia.stanford.edu/mich/)

Text2Mesh, CVPR'22

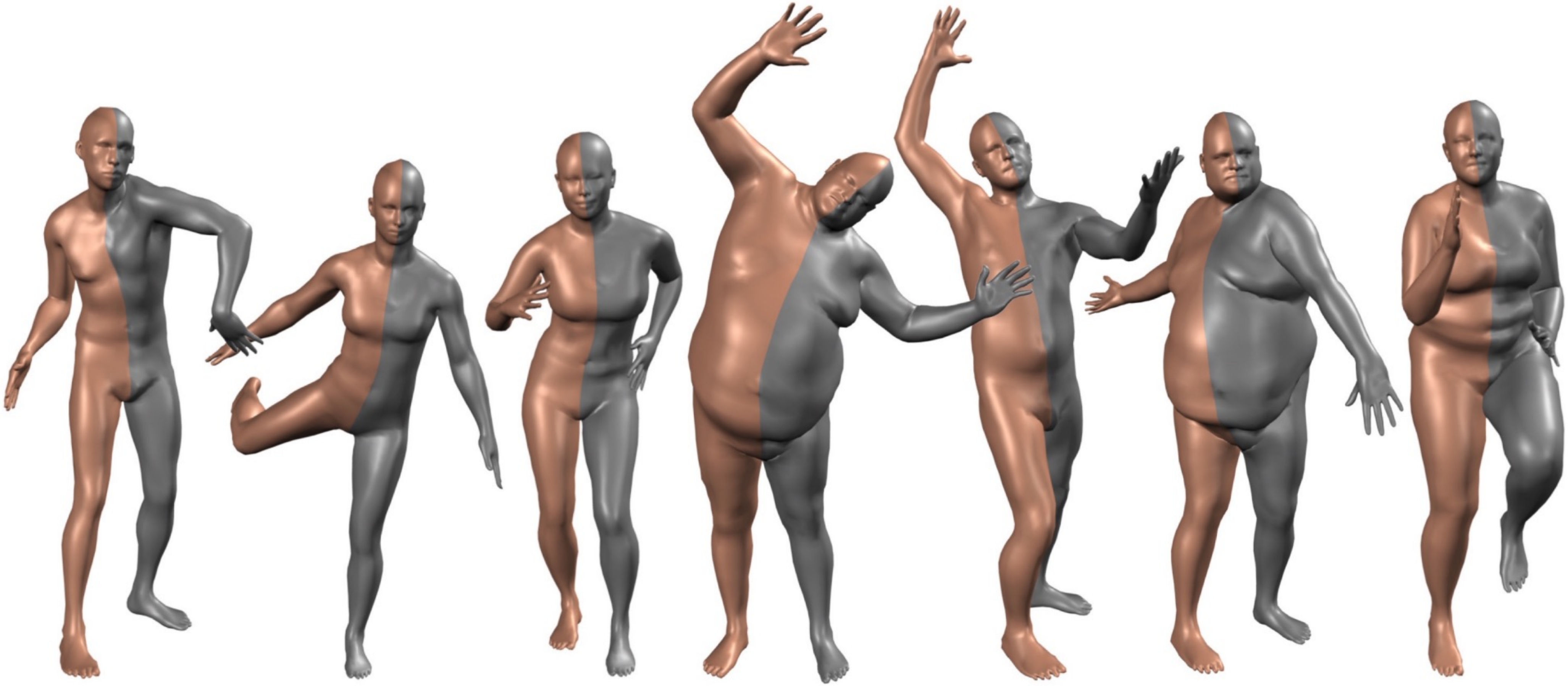


A Brick Lamp



Colorful Crochet Candle

SMPL model, MPI



Geometry: How do we represent shape of an object?

2.5D representation:

1) Depth & Normal map

Explicit representation:

2) Mesh

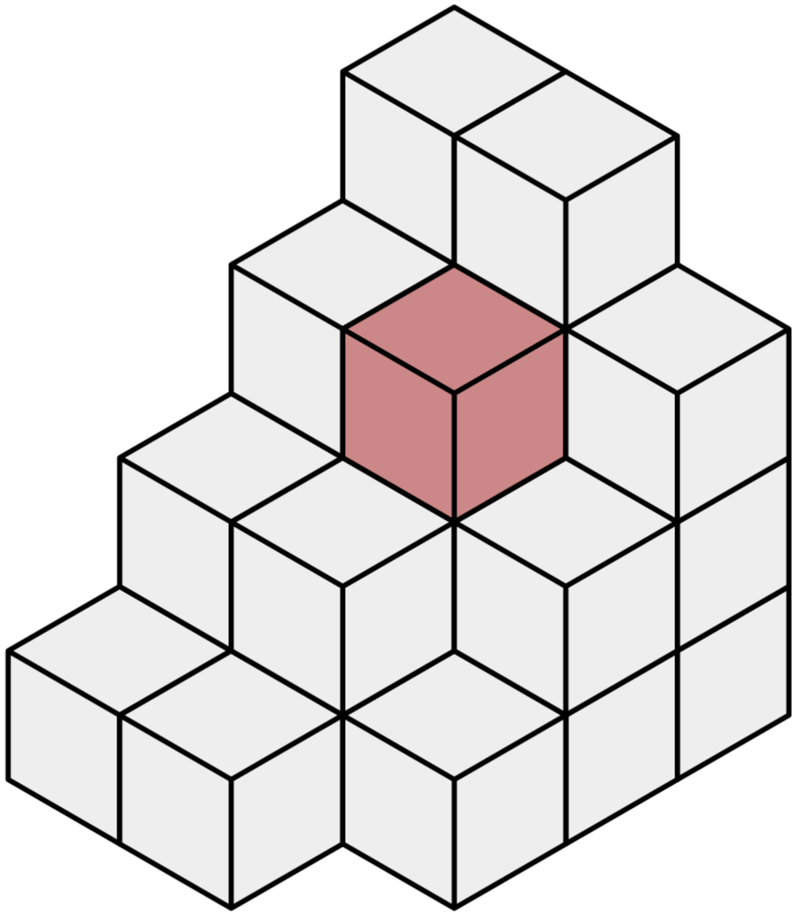
3) Voxels

4) Point Cloud

Implicit representation:

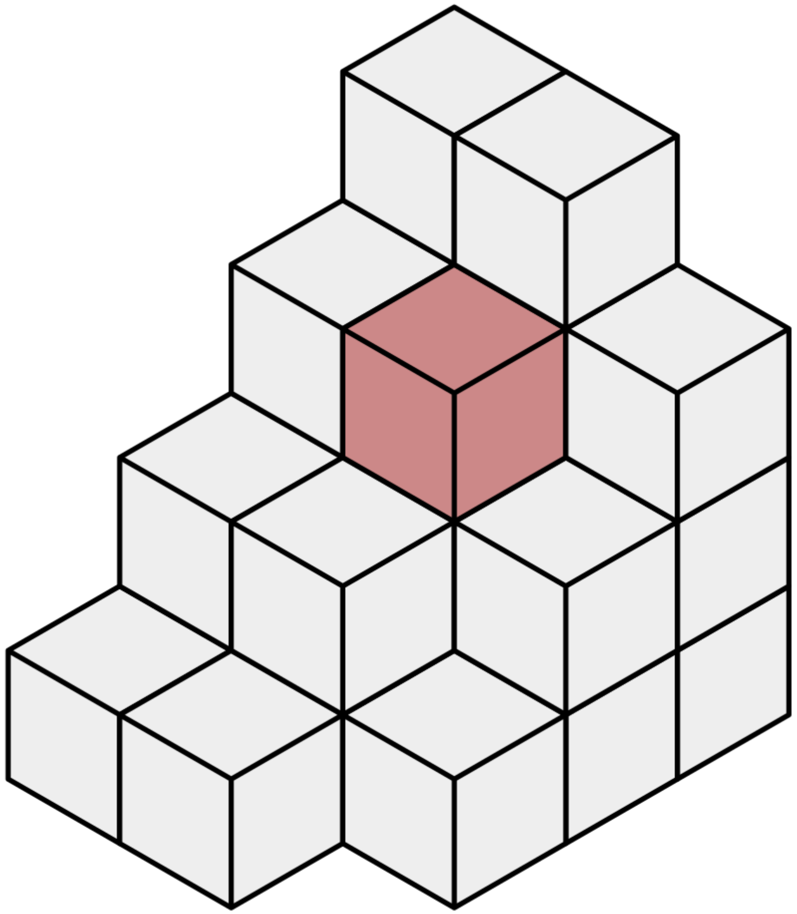
5) Surface Representation (SDF) – implicit

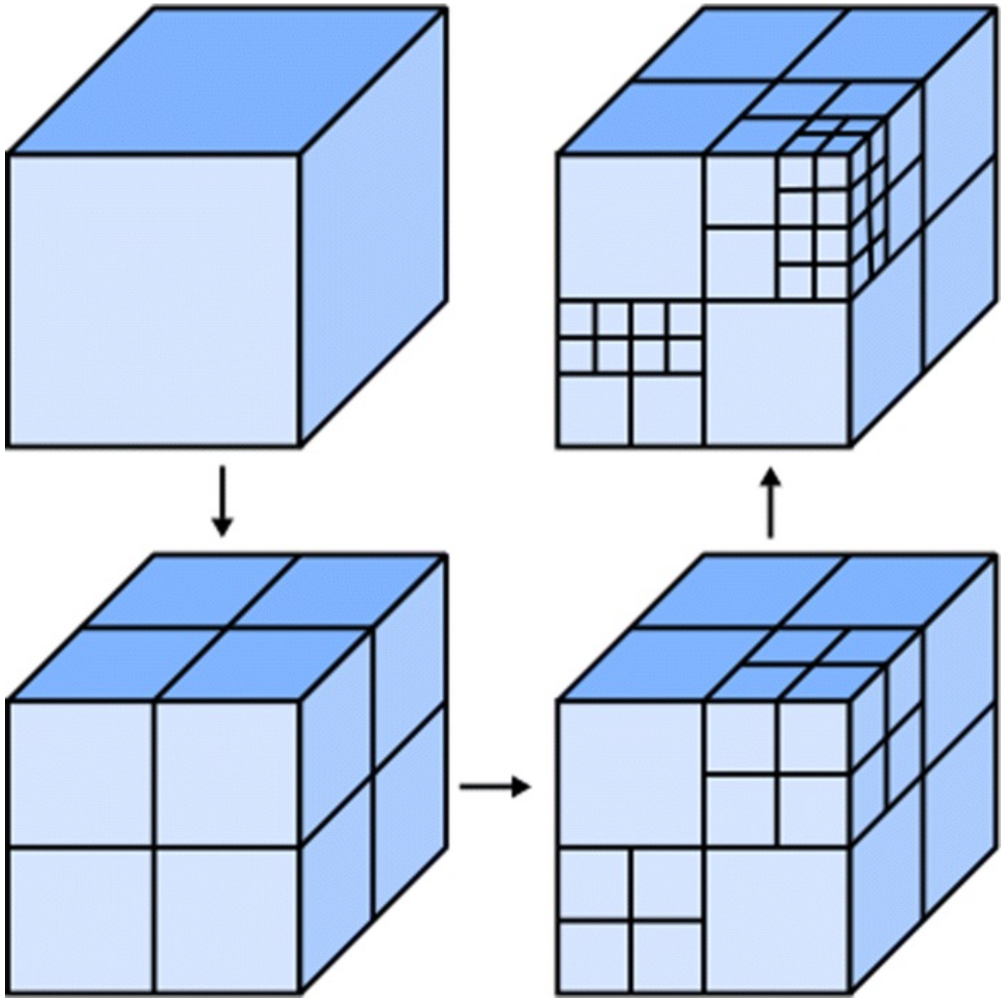
Voxel Representation



It’s like playing with Lego!

Voxel Representation

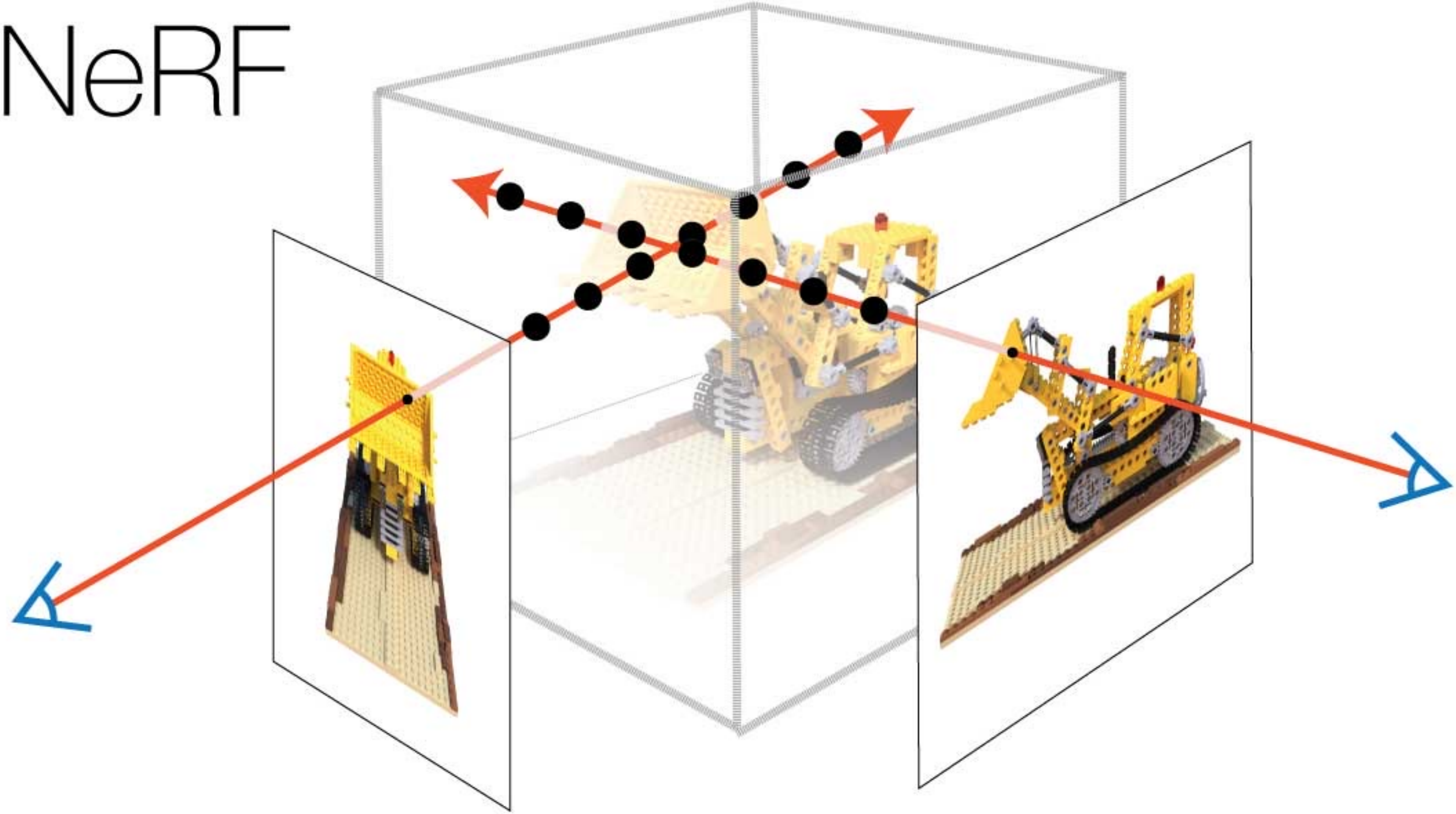




Voxel with octree

Few important/cool research works on voxels.

NeRF



We will learn a lot about voxels in 2nd half of the class when we discuss NeRF.

Geometry: How do we represent shape of an object?

2.5D representation:

1) Depth & Normal map

Explicit representation:

2) Mesh

3) Voxels

4) Point Cloud

Implicit representation:

5) Surface Representation (SDF) – implicit

LiDAR and many other range sensors produces point cloud.

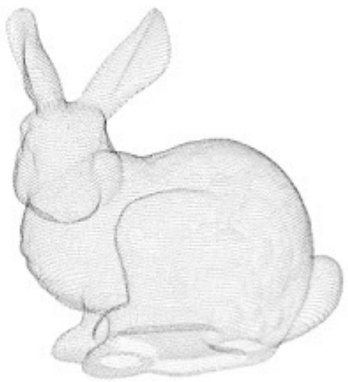
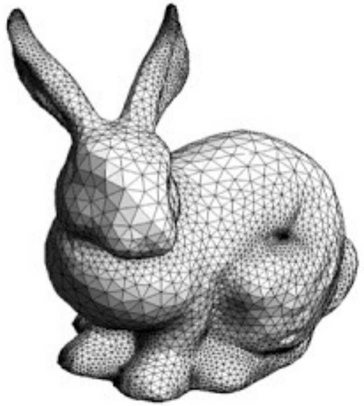
Point Clouds

Sparse model of central Rome using 21K photos produced by COLMAP's SfM pipeline.

Dense models of several landmarks produced by COLMAP's MVS pipeline.

Started at UNC!! – Jan Michael Frahm's group

3D Representations (Explicit)

	Voxel	**Point cloud**	**Polygon mesh**
Memory efficiency	Poor	Not good	Good
Textures	Not good	No	Yes
For neural networks	Easy	Not easy	Not easy

We adopt **polygon mesh** for its high potential

Images are from
http://cse.iitkgp.ac.in/~pb/research/3dpoly/3dpoly.html
http://waldyrious.net/learning-holography/pb-cgh-formulas.xhtm
http://www.cs.mun.ca/~omeruvia/philosophy/images/BunnyWire.gif

Geometry: How do we represent shape of an object?

2.5D representation:

1) Depth & Normal map

Explicit representation:

2) Mesh
3) Voxels
4) Point Cloud

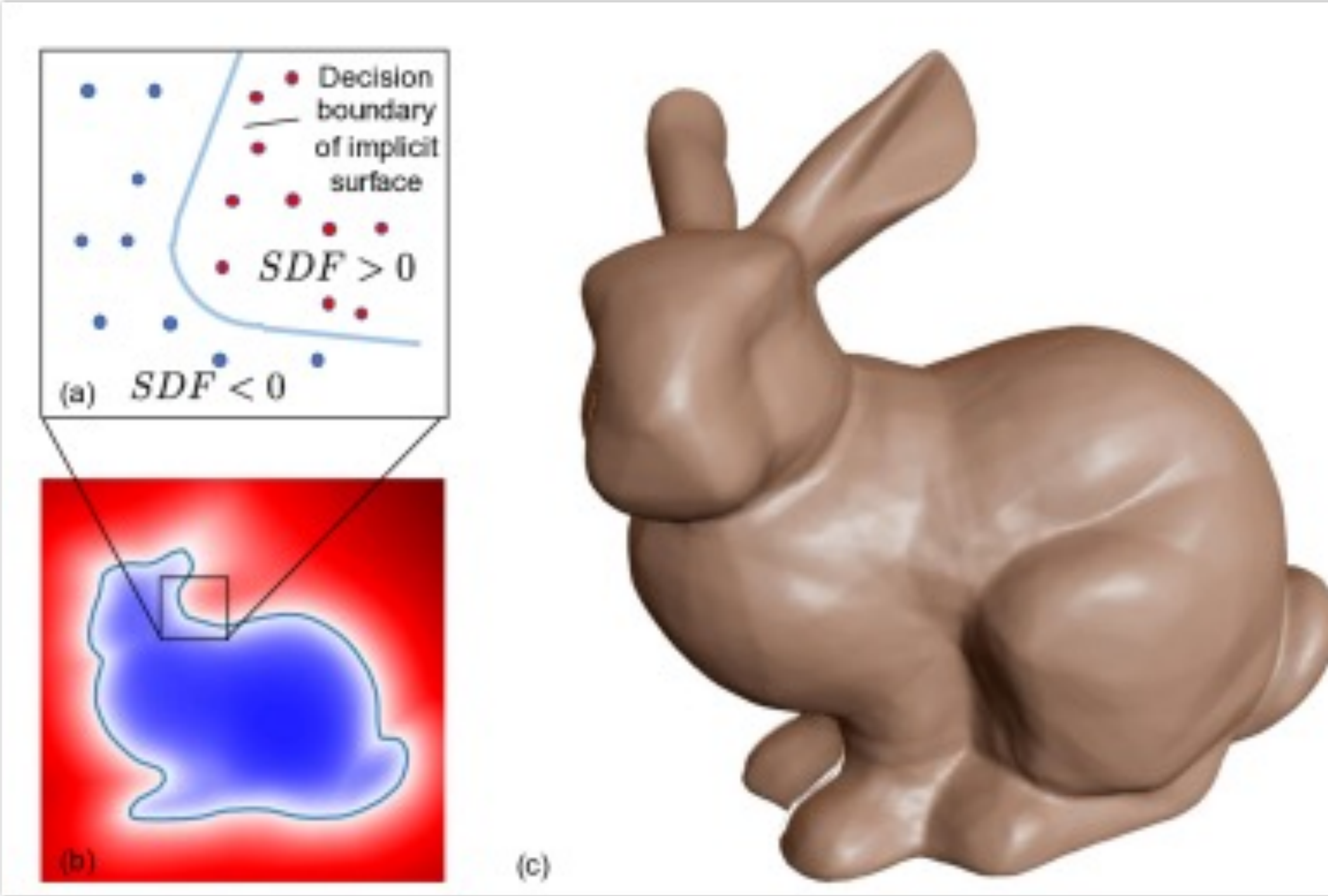
Implicit representation:

5) Surface Representation (SDF)

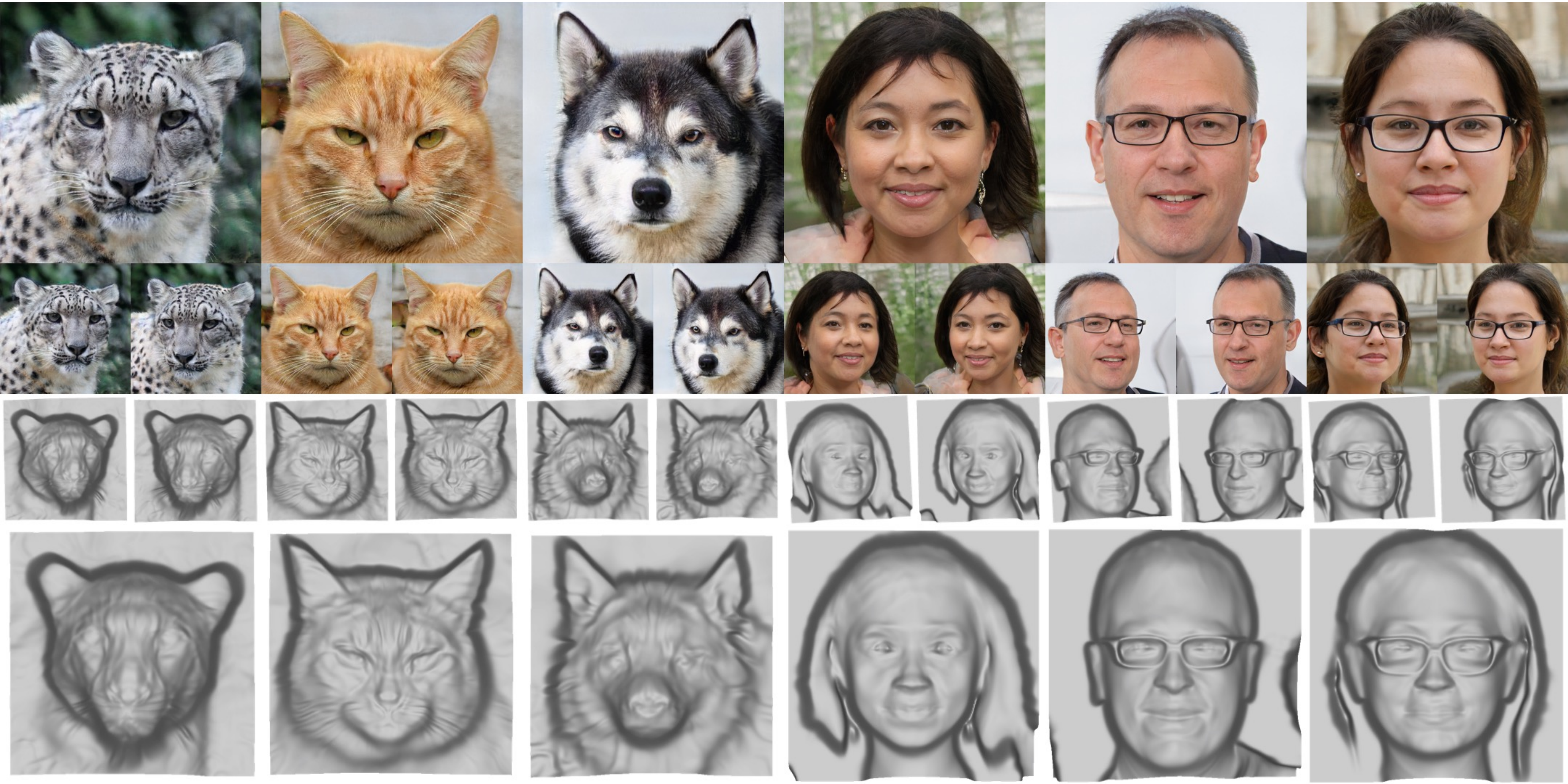
Surface Representation: Signed Distance Function (SDF) - implicit representation via level set

SDF(X) = 0, when X is on the surface.
SDF(X) > 0, when X is outside the surface
SDF(X) < 0, when X is inside the surface

Note: SDF is an implicit representation! Suitable for neural networks but hard to import inside existing graphics software.



Deep SDF: Use a neural network (co-ordinate based MLP) to represent the SDF function.



StyleSDF, Or-El et. al

Neural RGB-D Scene Reconstruction, Azinovic et. al.

Respond at **PollEv.com/ronisen**

Text **RONISEN** to **22333** once to join, then text your message

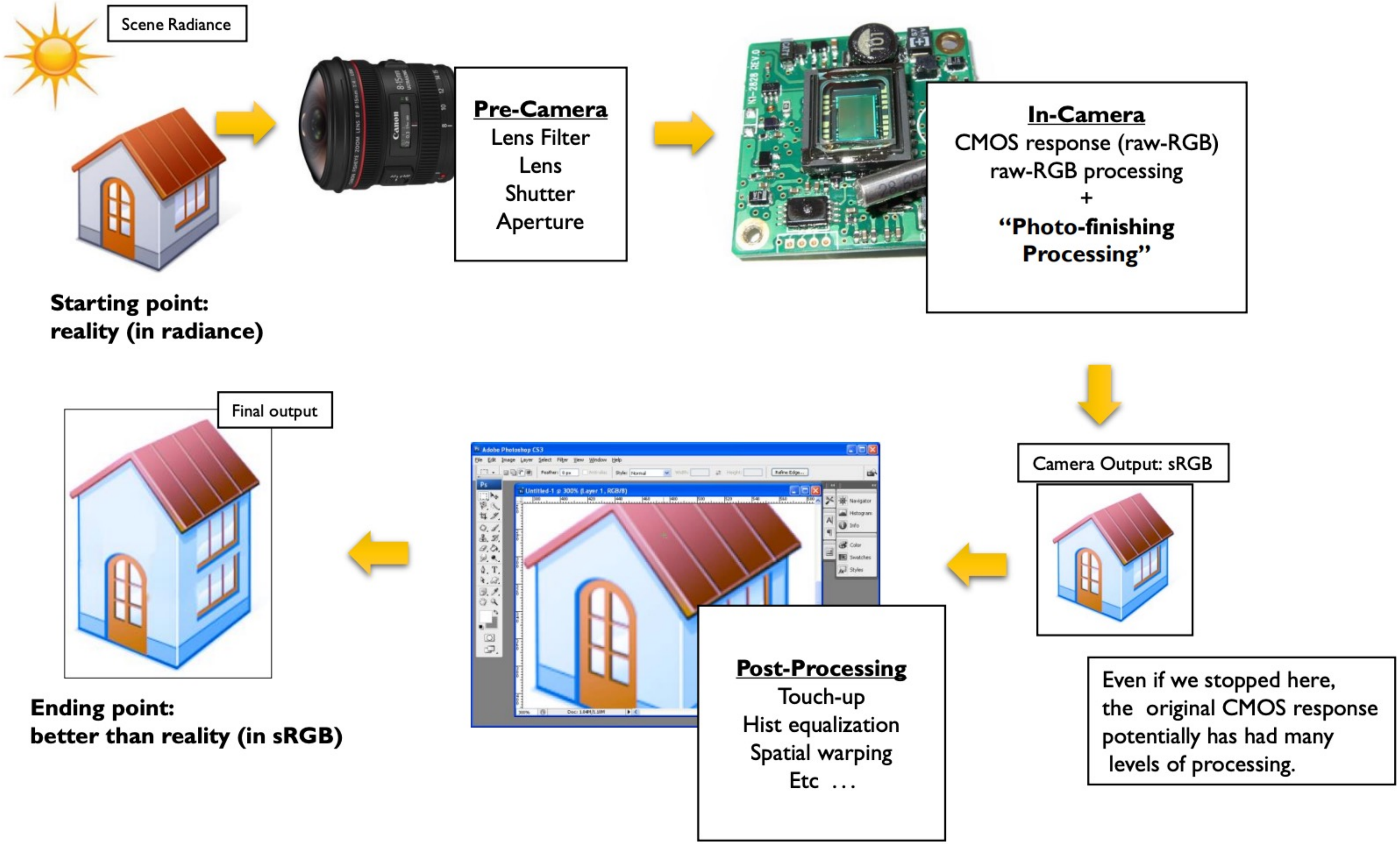
Feel free to share your questions...

Agenda

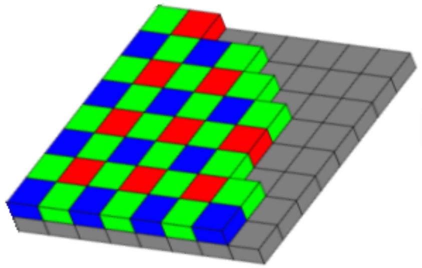
- How do we define geometry/shape of an object?
- How do we define a camera model? – 3D object to 2D image
- How do we define material property? – glossy, metallic

Further reading: [Understanding Color and the In-Camera Image Processing Pipeline for Computer Vision]

Modern photography pipeline



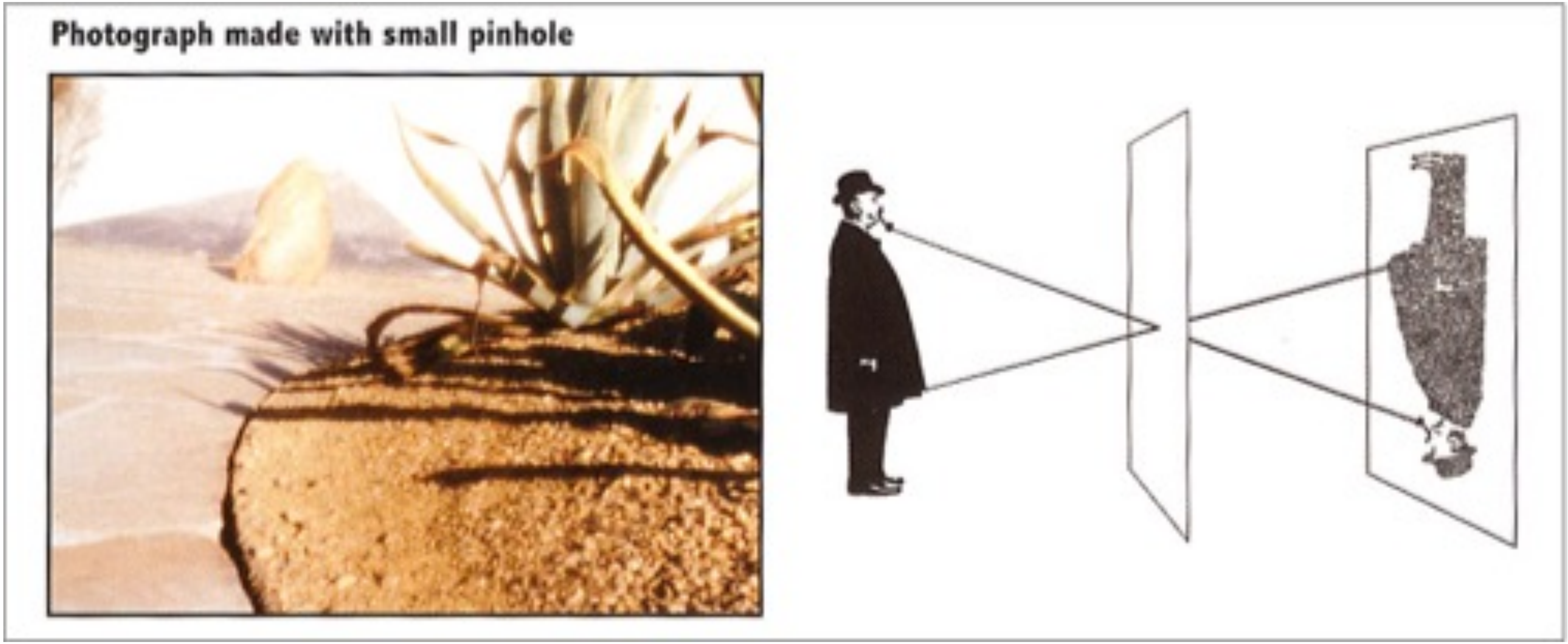
A typical color imaging pipeline



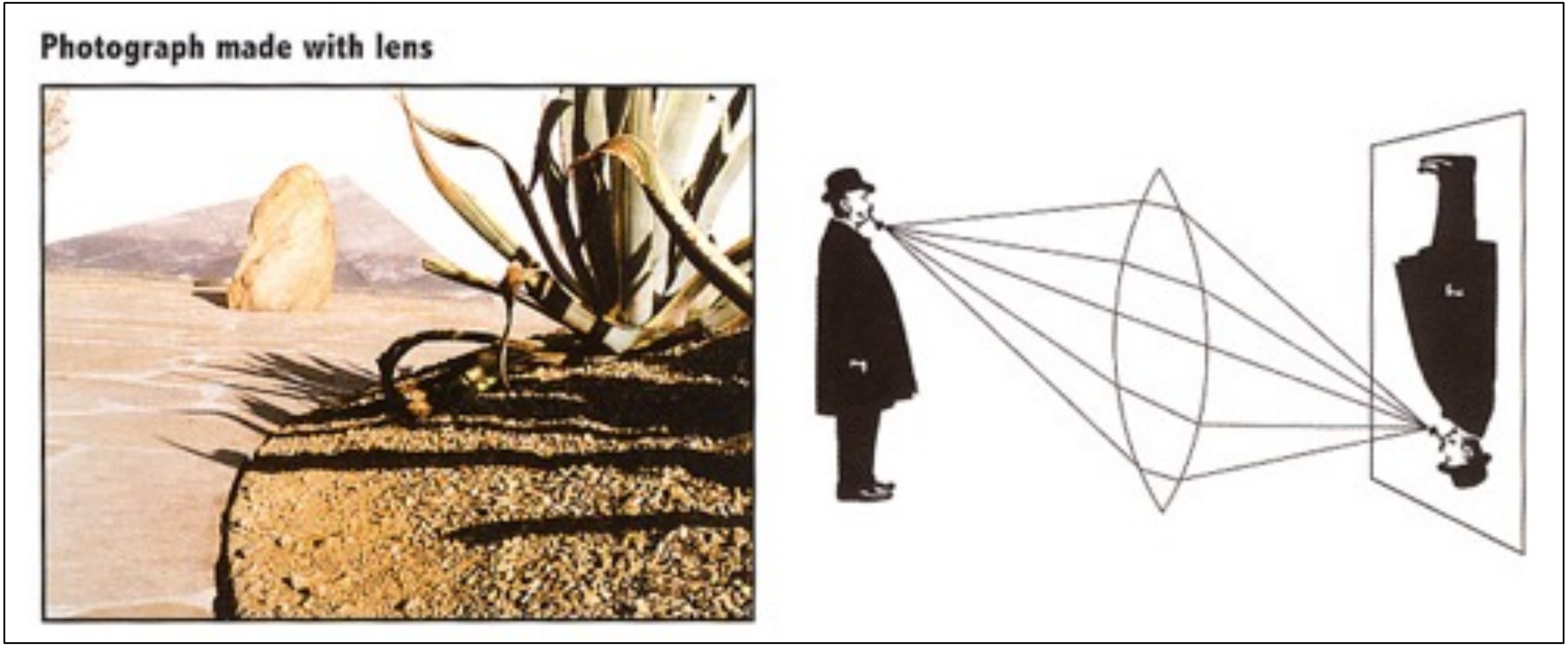
Sensor with color filter array (CFA)
(CCD/CMOS)

→ ISO gain and raw-image processing → RGB Demoasicing → Noise Reduction → White-Balance & Color Space Transform (CIE XYZ) → Color Manipulation (Photo-finishing) → Mapping to sRGB output → JPEG Compression → Save to file

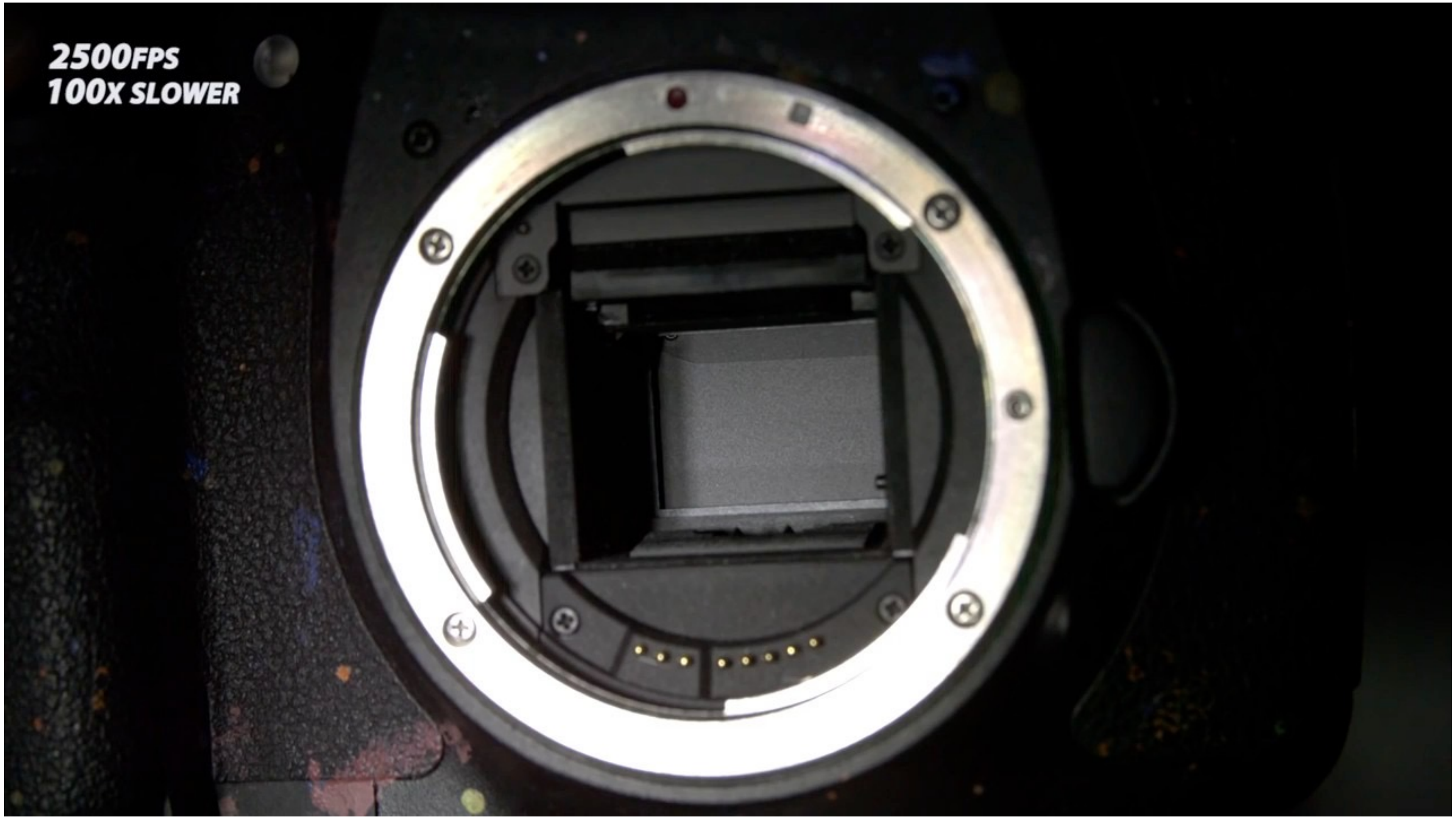
Pinholes & Lenses Form Image on Sensor



Pinholes & Lenses Form Image on Sensor



Shutter Exposes Sensor For Precise Duration



The Slow Mo Guys, https://youtu.be/CmjeCchGRQo

Sensor Accumulates Irradiance During Exposure

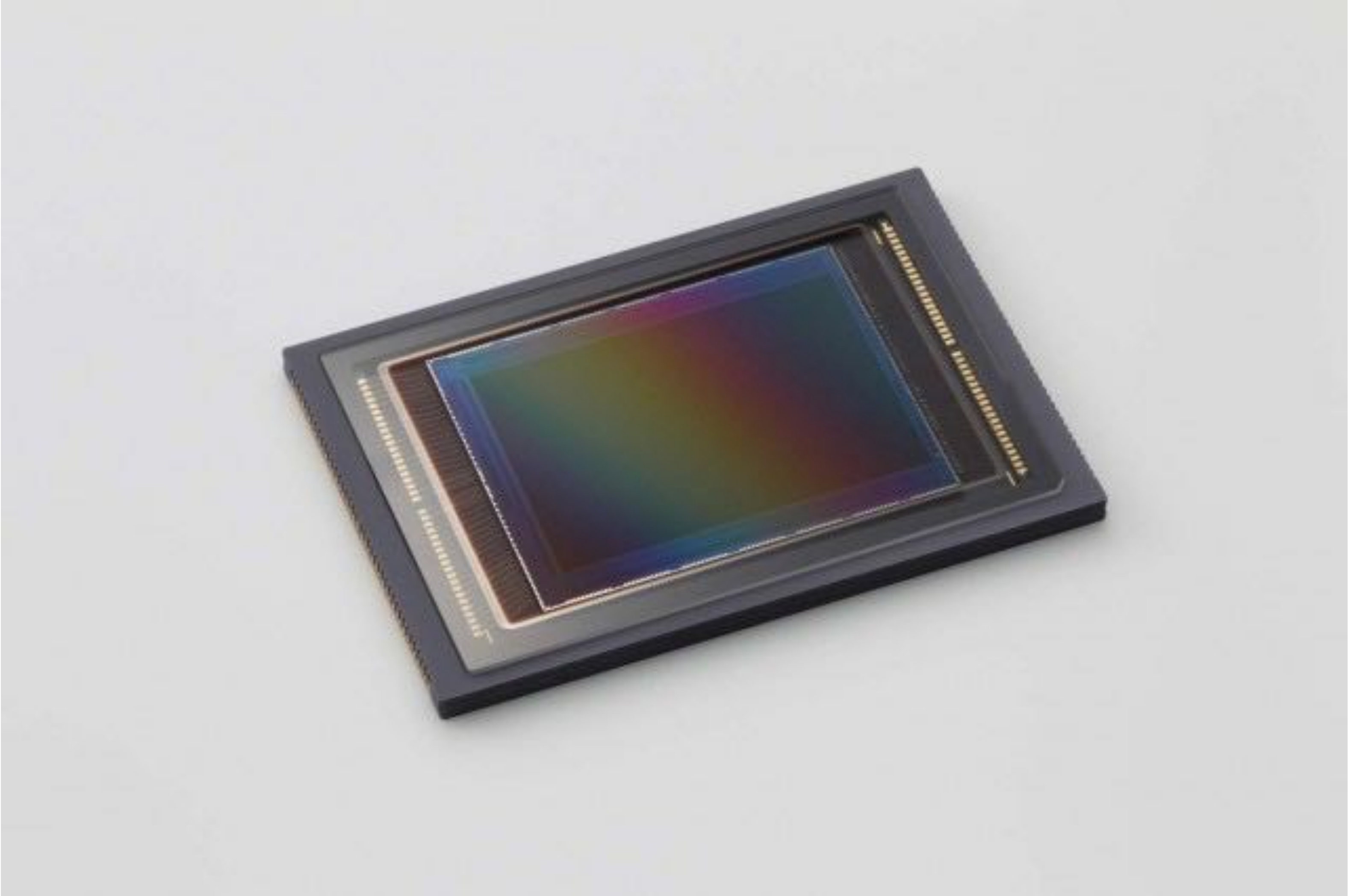
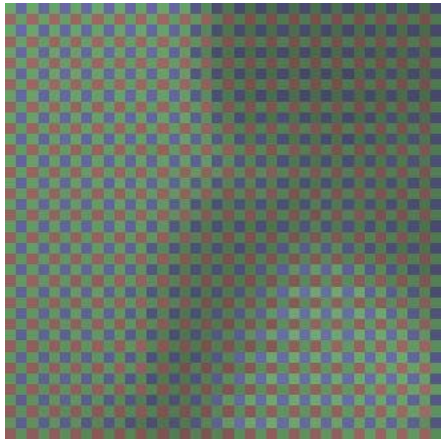
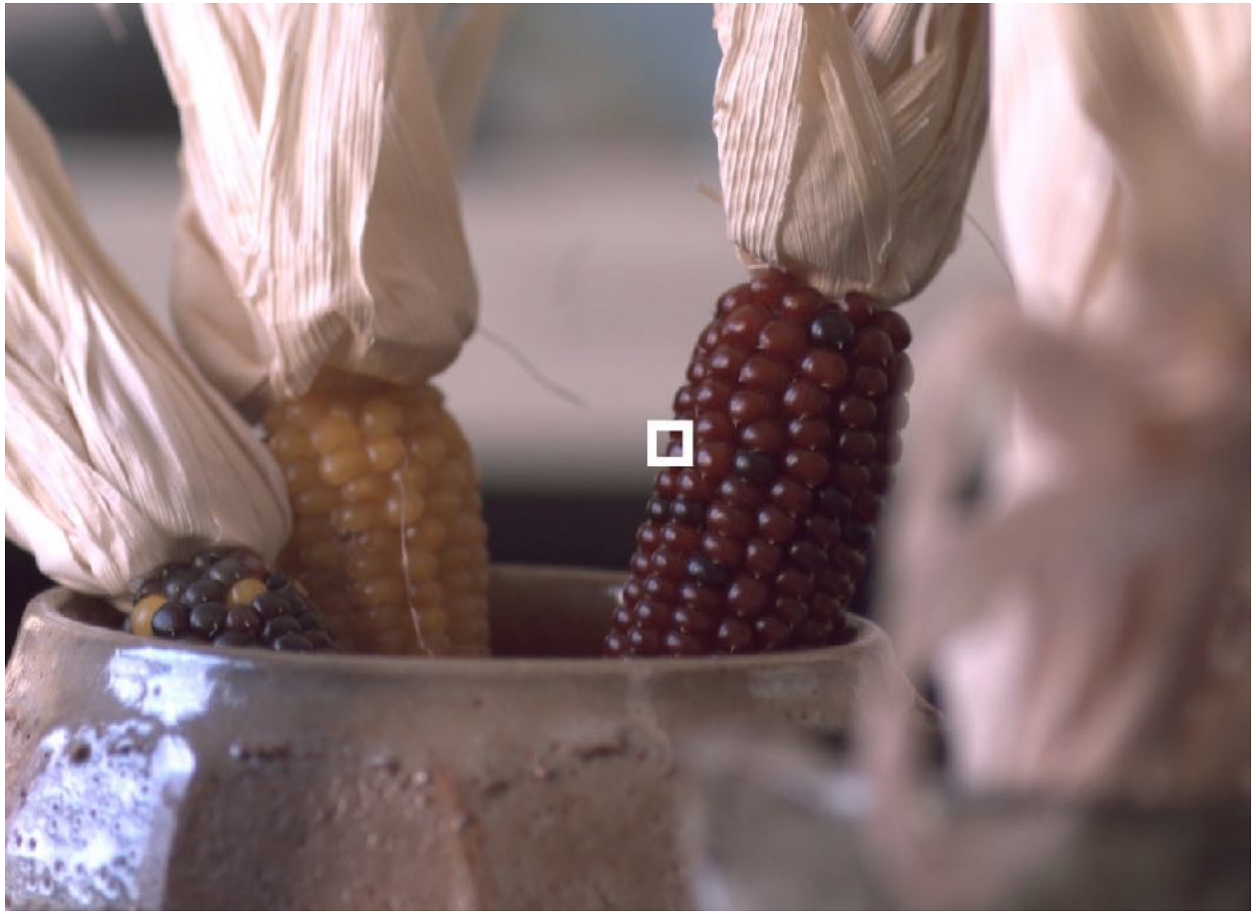


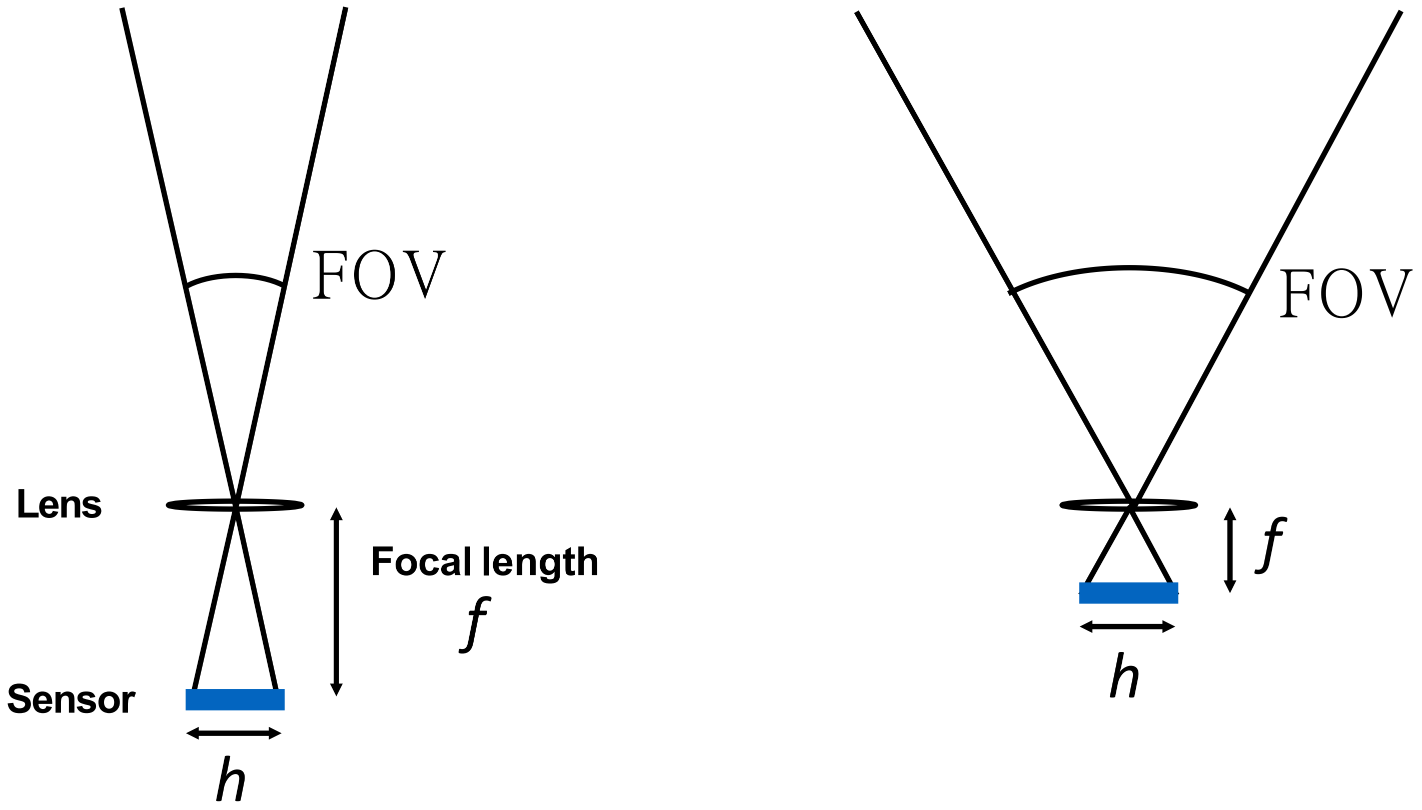
Image Processing: From Sensor Values to Image





Optics of Image Formation:
Field of View

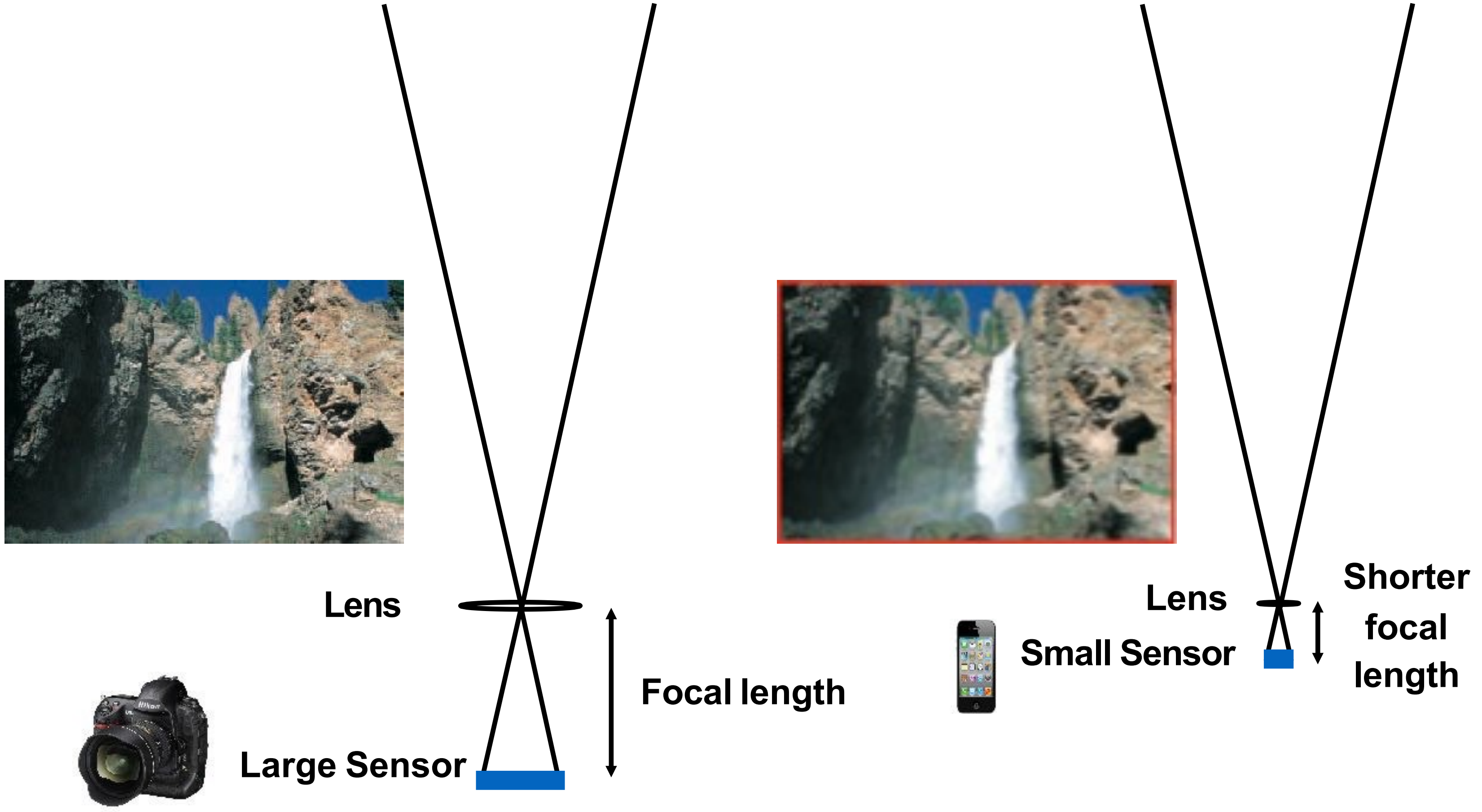
Effect of Focal Length on FOV



For a fixed sensor size, decreasing the focal length increases the field of view.

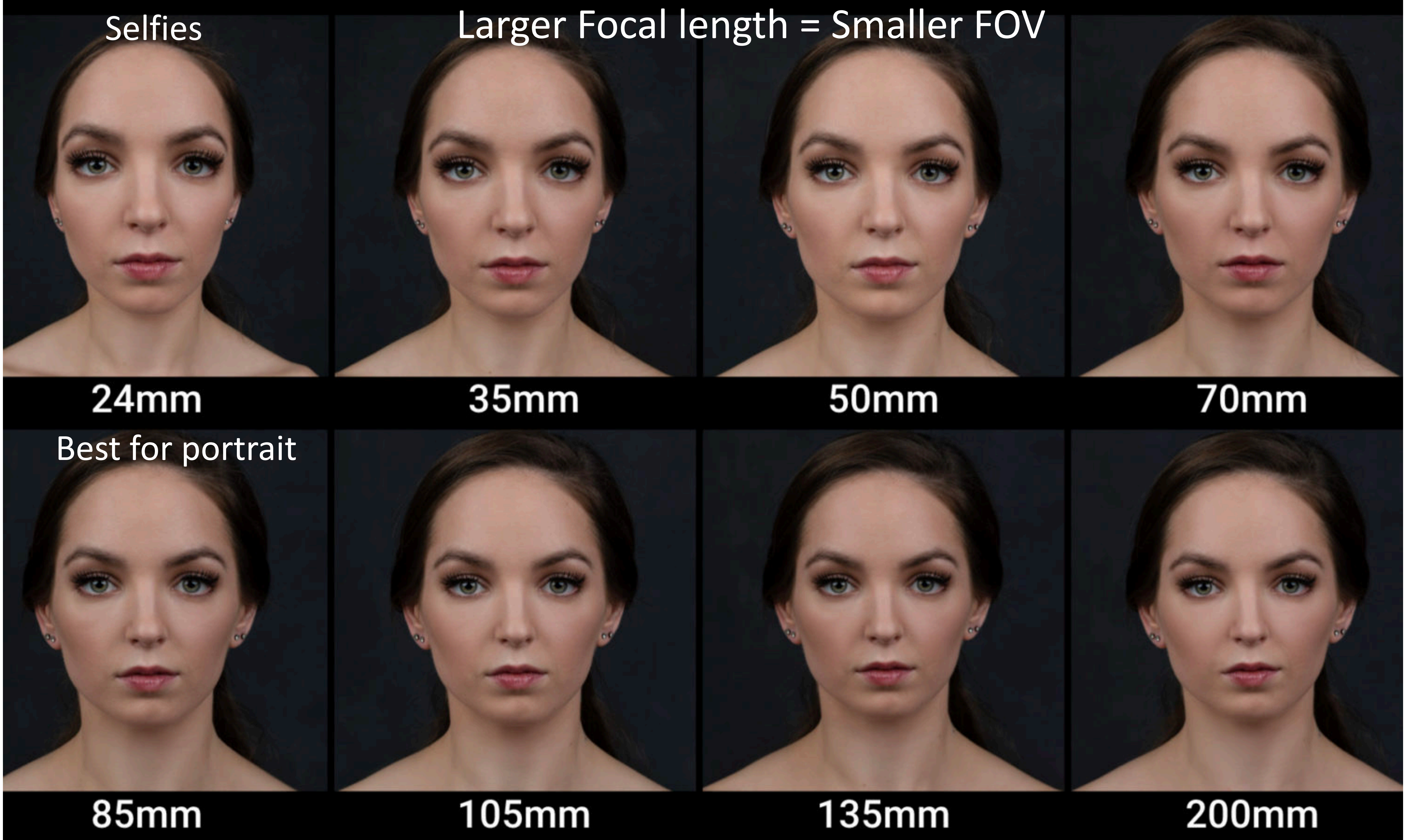
$$\mathrm{FOV} = 2 \arctan \frac{h}{2f}$$

Maintain FOV on Smaller Sensor?



To maintain FOV, decrease focal length of lens in proportion to width/height of sensor

Larger Focal length =
Smaller FOV

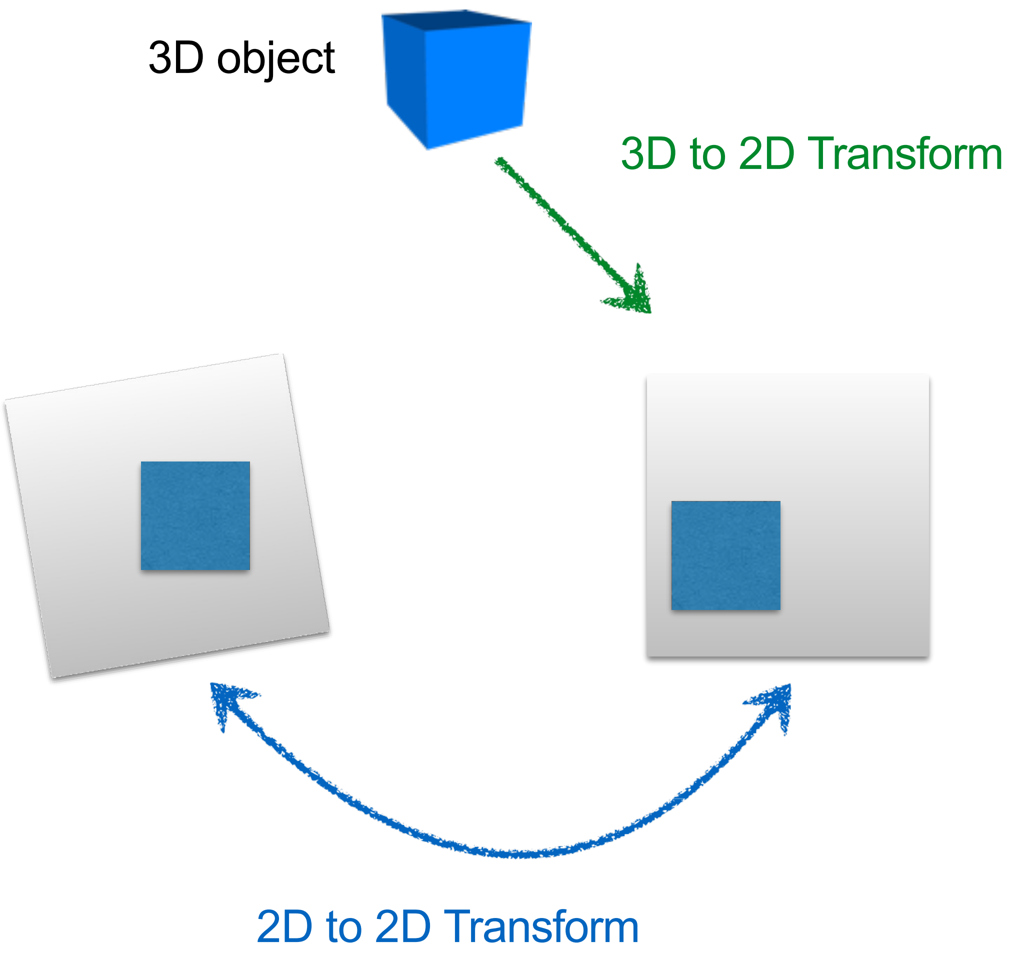

Larger Focal length = Smaller FOV
Selfies
24mm
35mm
50mm
70mm
Best for portrait
85mm
105mm
135mm
200mm

A camera is a mapping between

the **3D world**

and

a **2D image**


3D object
3D to 2D Transform
2D to 2D Transform

A camera is a mapping between the 3D world and a 2D image

$$\boldsymbol{x} = \mathbf{P}\,\mathbf{X}$$

2D image point

camera matrix

3D world point

$$\boldsymbol{x} = \mathbf{PX}$$

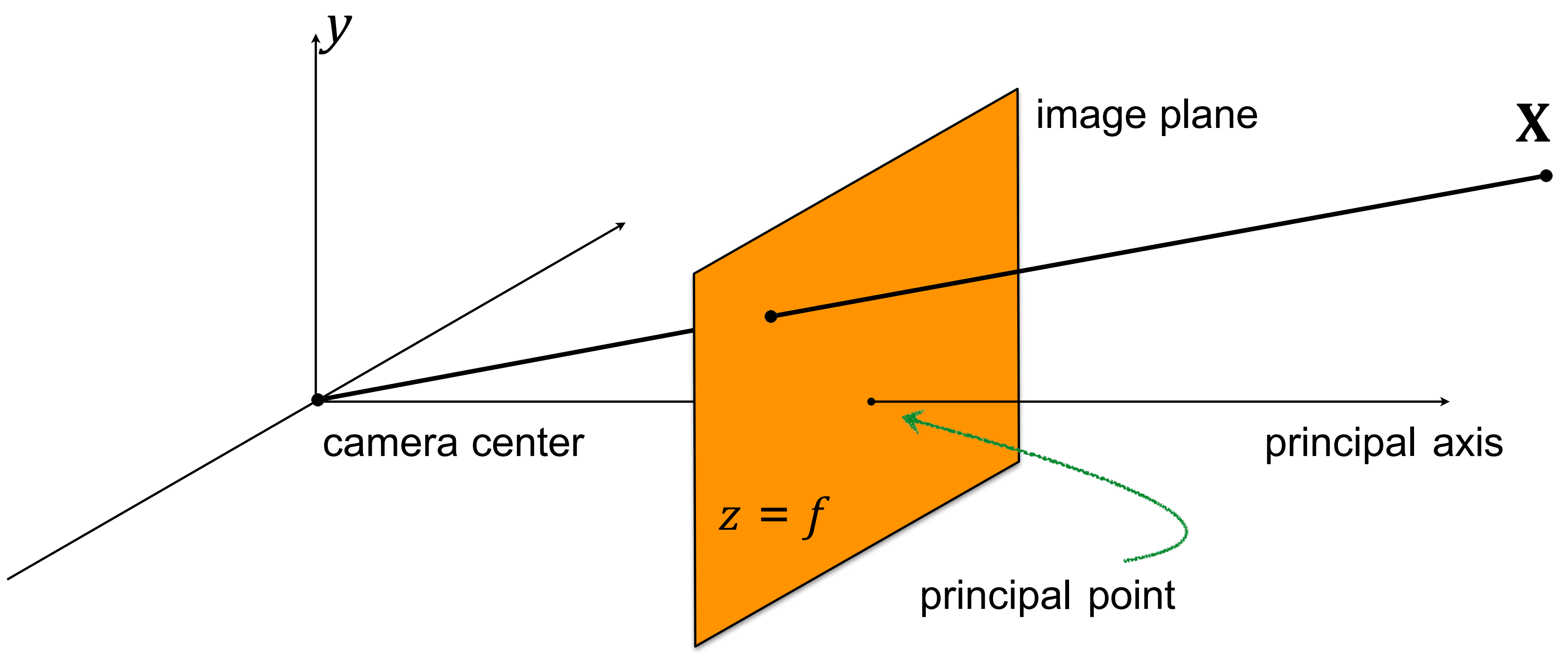
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

homogeneous
image
3 x 1

Camera
matrix
3 x 4

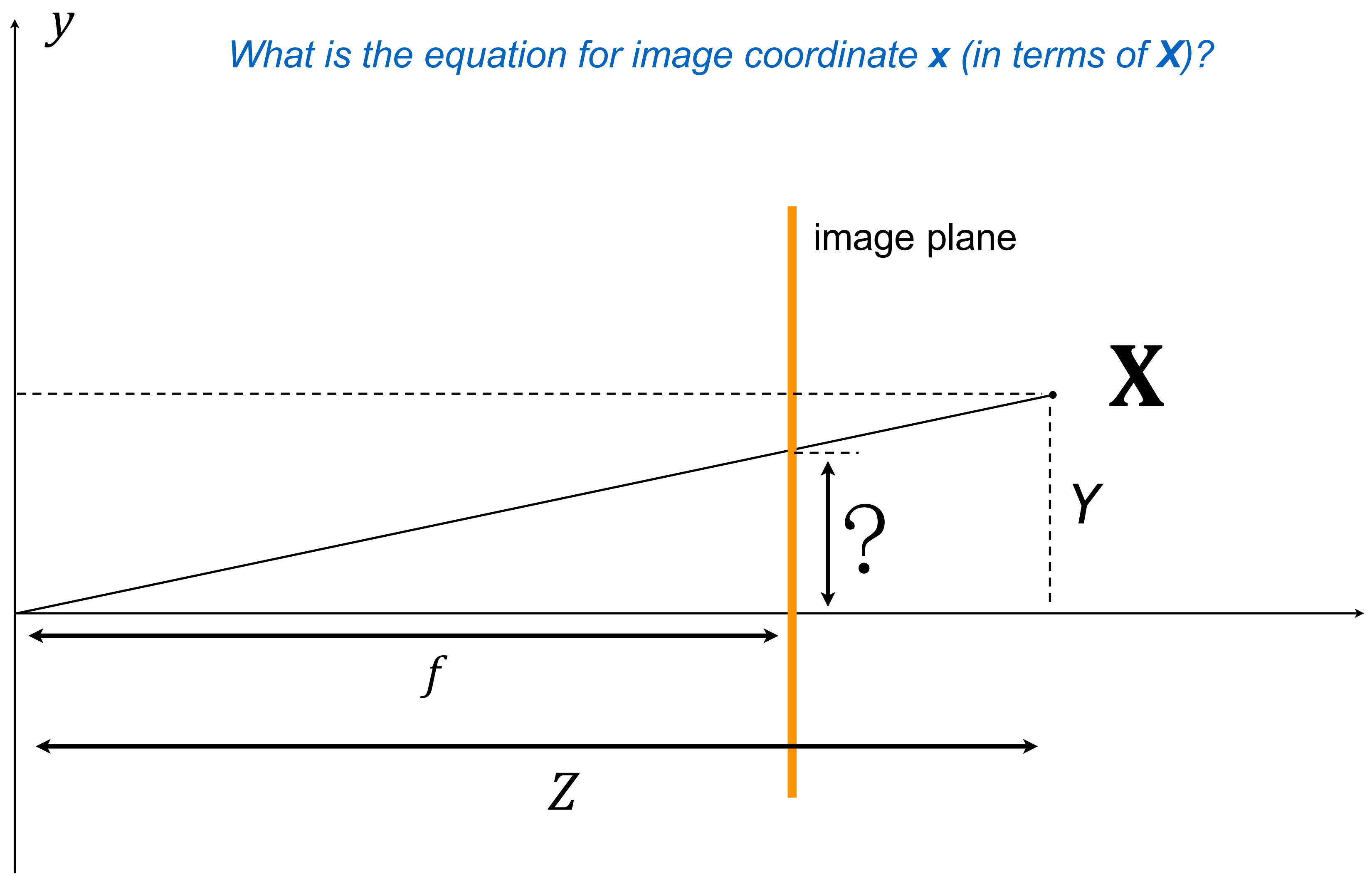
homogeneous
world point
4 x 1

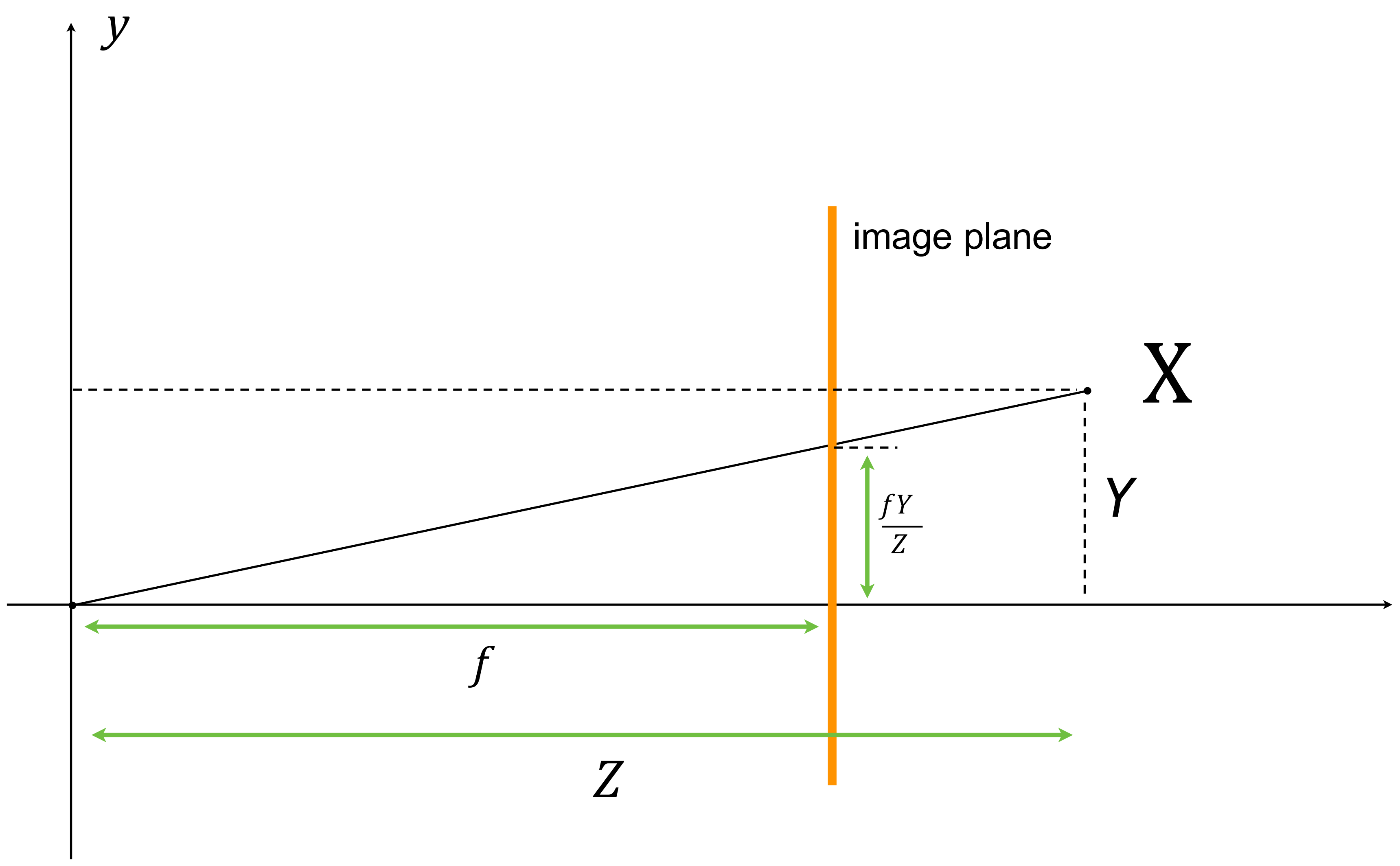
The pinhole camera



*What is the equation for image coordinate **x** (in terms of **X**)?*

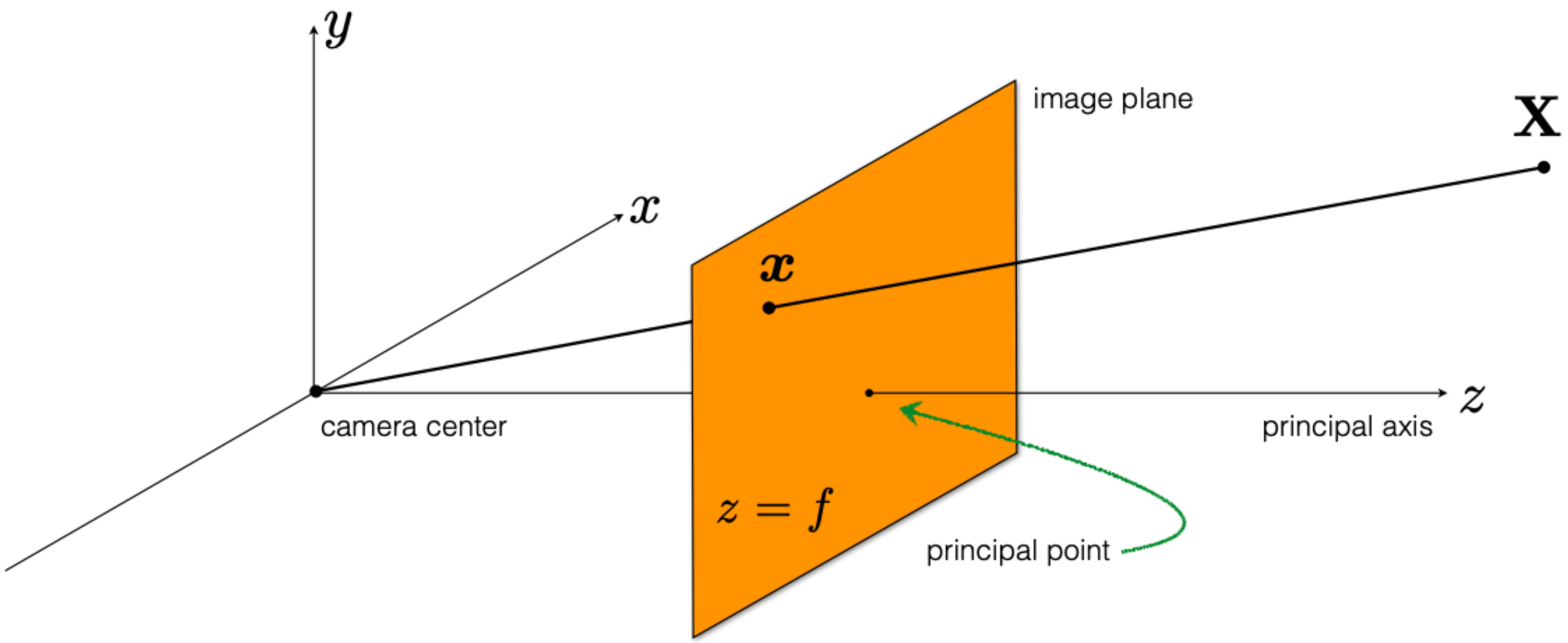
*What is the equation for image coordinate **x** (in terms of **X**)?*





$$[X \quad Y \quad Z]^\top \mapsto [fX/Z \quad fY/Z]^\top$$

Pinhole camera geometry



What is the camera matrix **P** *for a pinhole camera model?*

$$\boldsymbol{x} = \mathbf{P}\mathbf{X}$$

Relationship from similar triangles…

$$[X \quad Y \quad Z]^\top \mapsto [fX/Z \quad fY/Z]^\top$$

generic camera model

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

What does the pinhole camera model look like?

$$\mathbf{P} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$$

Relationship from similar triangles…

$$[X \quad Y \quad Z]^\top \mapsto [fX/Z \quad fY/Z]^\top$$

generic camera model

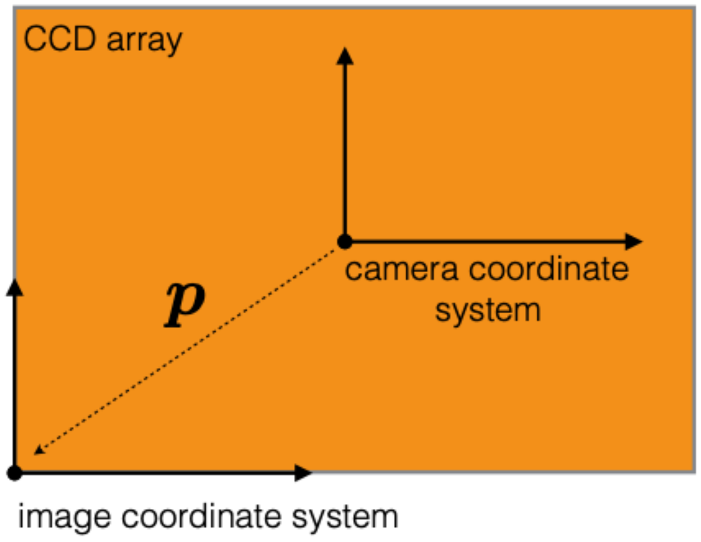
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

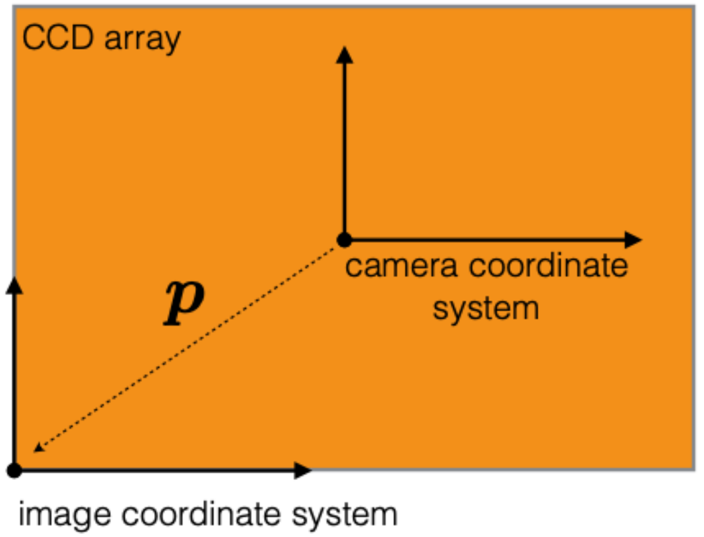
What does the pinhole camera model look like?

$$\mathbf{P} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{P} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Camera origin and image origin might be different

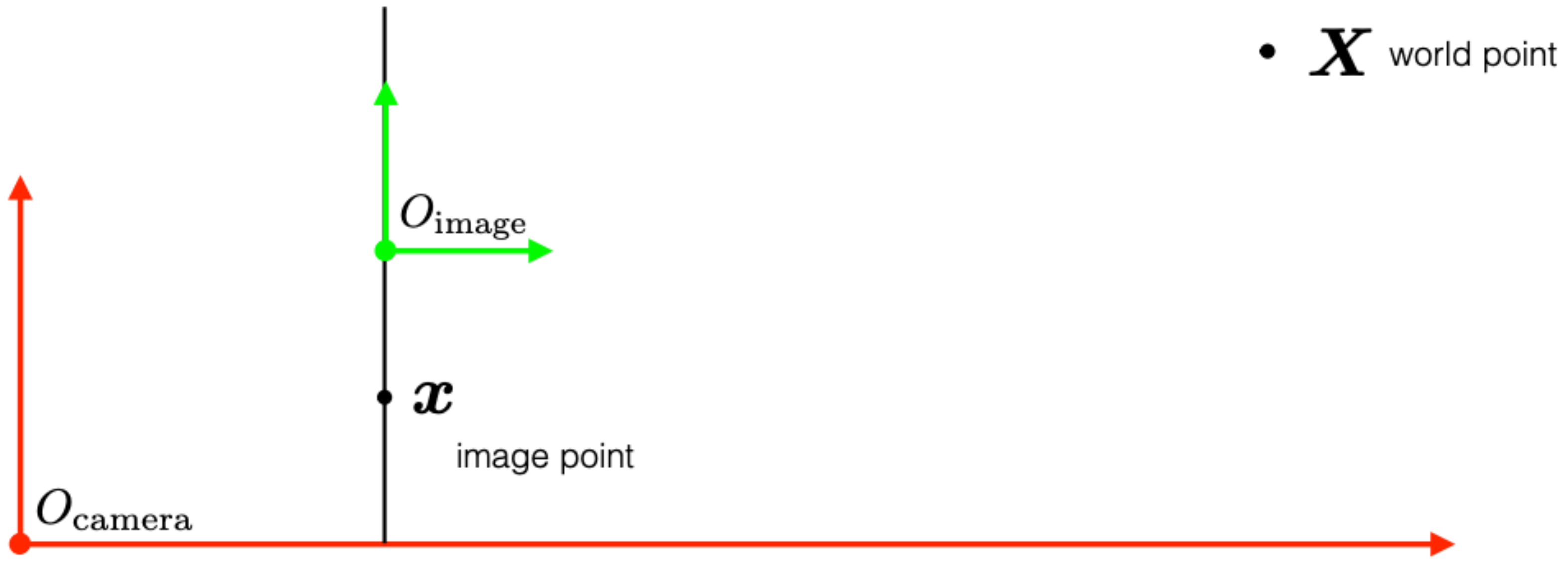




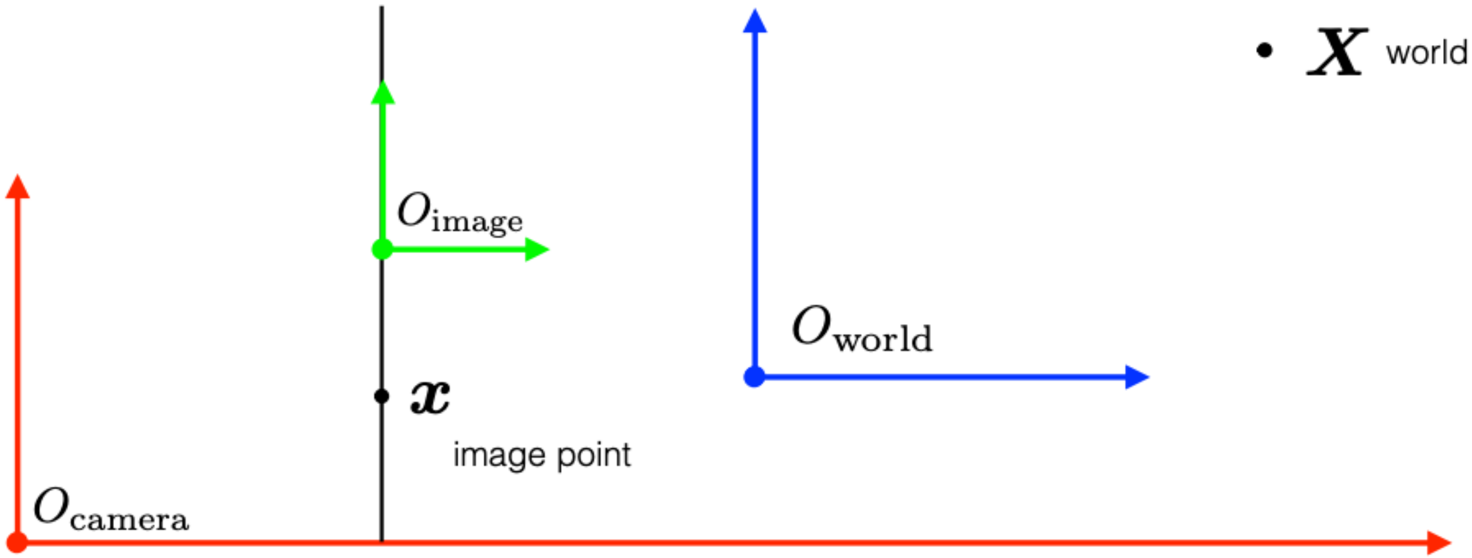
$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Accounts for different origins

In general, the camera and image sensor have **different** coordinate systems



In general, there are **three different** coordinate systems…



so you need the know the transformations between them

Can be decomposed into two matrices

- Relationship between image & camera coord. Systems.
- Camera Calibration matrix
- Camera Extrinsic
- Can be obtained from image meta data.

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array} \right]$$

(3 x 3) (3 x 4)

- Relationship between world & camera coord. Systems.
- Camera Intrinsic
- Often known as 'Camera Pose Estimation/ Camera Localization problem'.

$$\mathbf{P} = \mathbf{K}[\mathbf{I}|\mathbf{0}]$$

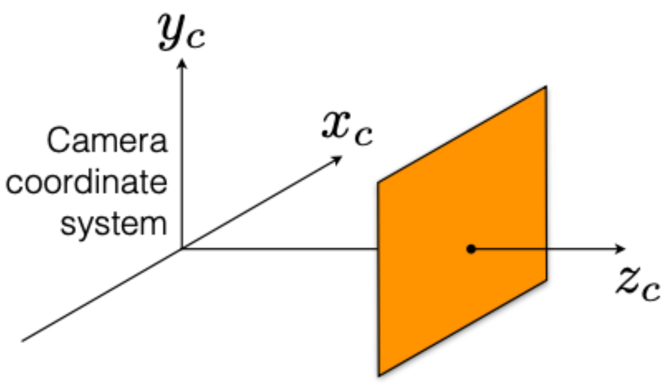
$$\mathbf{K} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

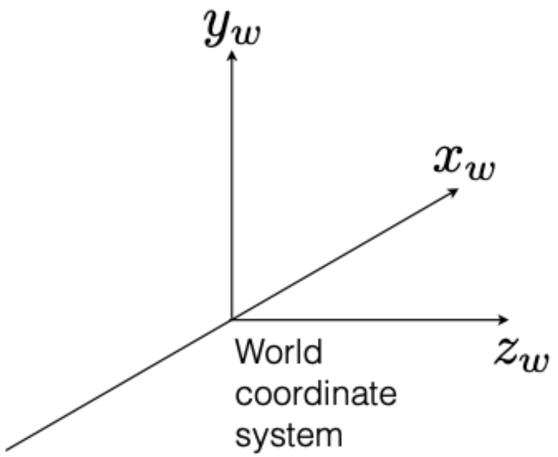
calibration matrix

Assumes that the **camera** and **world** share the same coordinate system

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

What if they are different?
How do we align them?

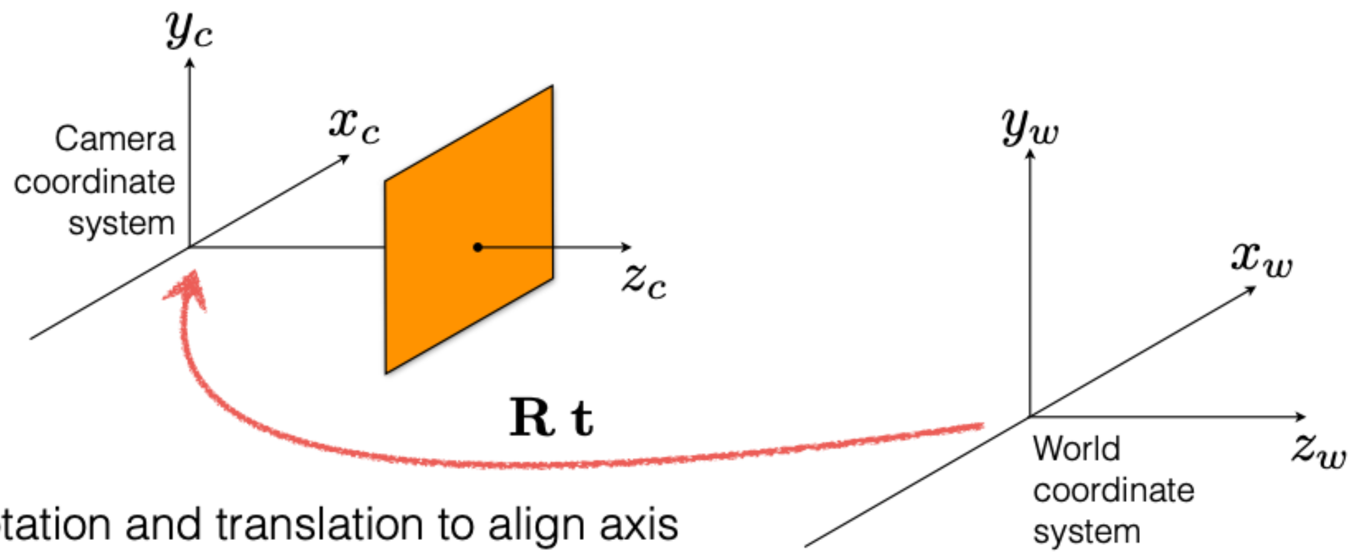




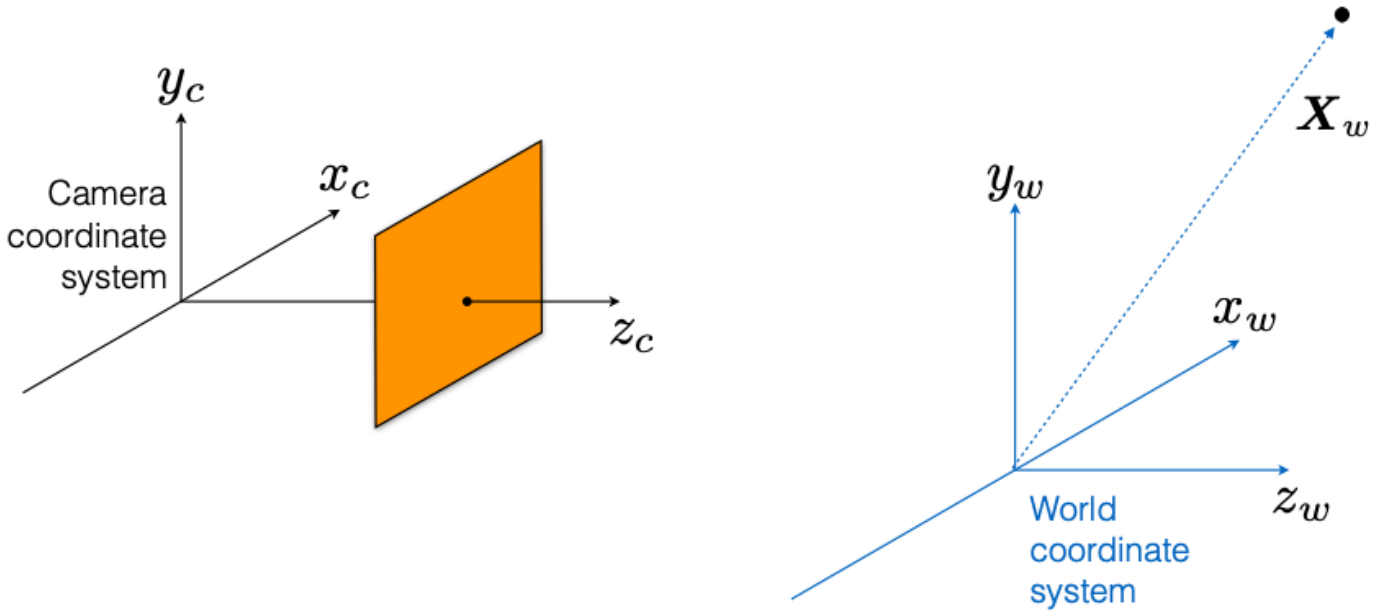
Assumes that the camera and world share the same coordinate system

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

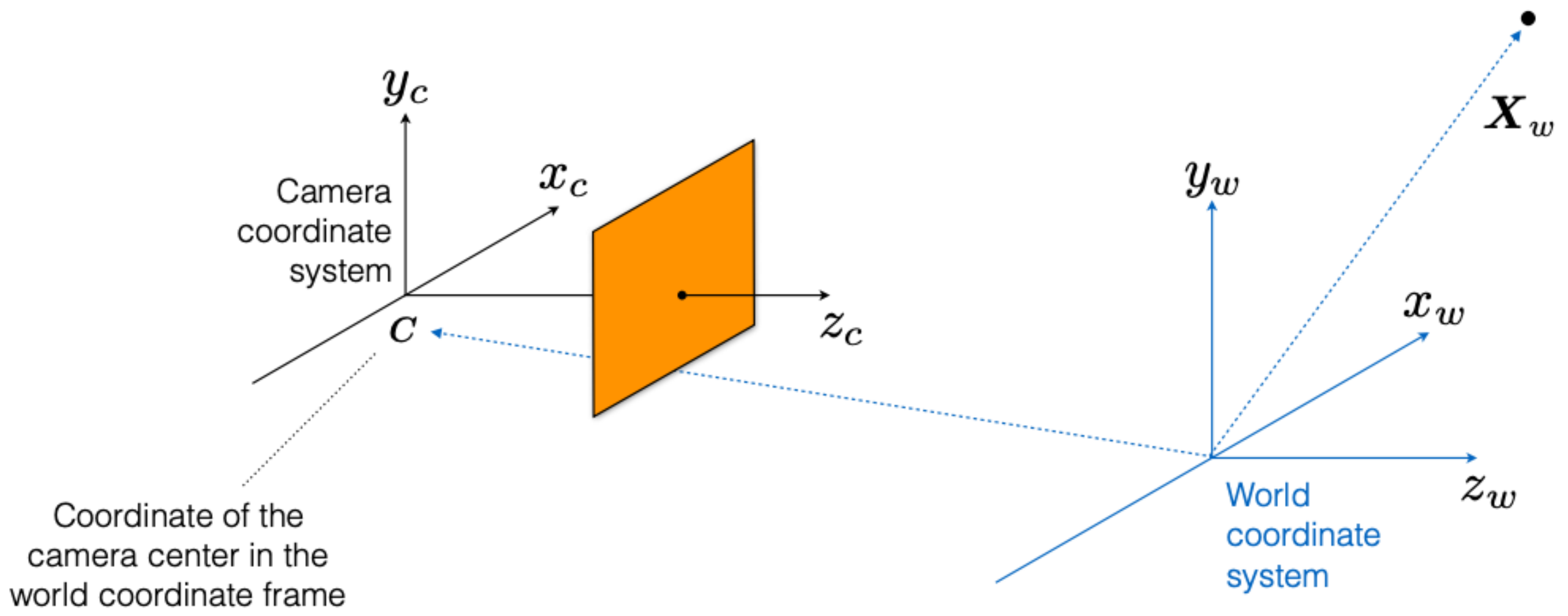
What if they are different?
How do we align them?



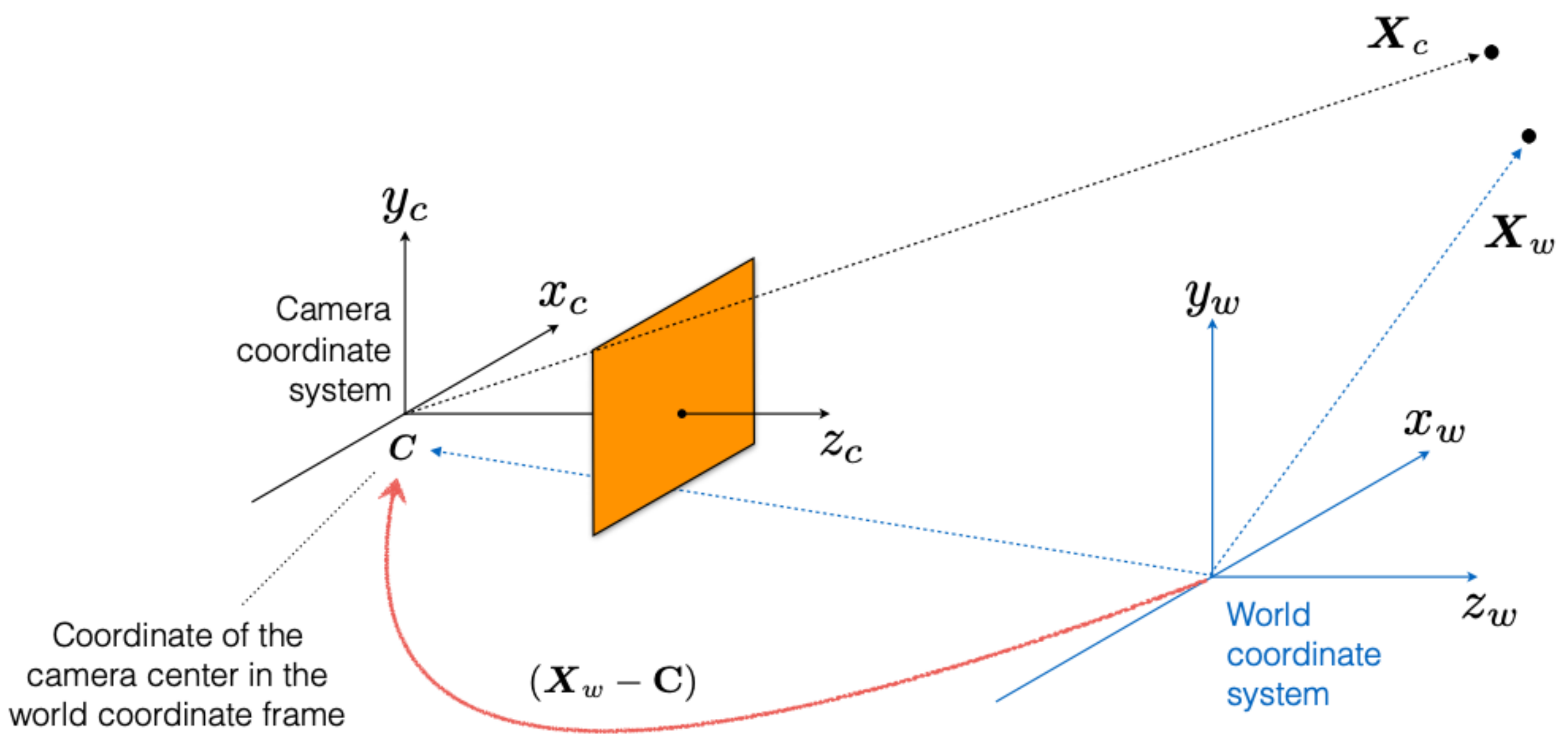
3R rotation and translation to align axis



y_c
x_c
Camera coordinate system
z_c
$\boldsymbol{X}_w$
y_w
x_w
z_w
World coordinate system

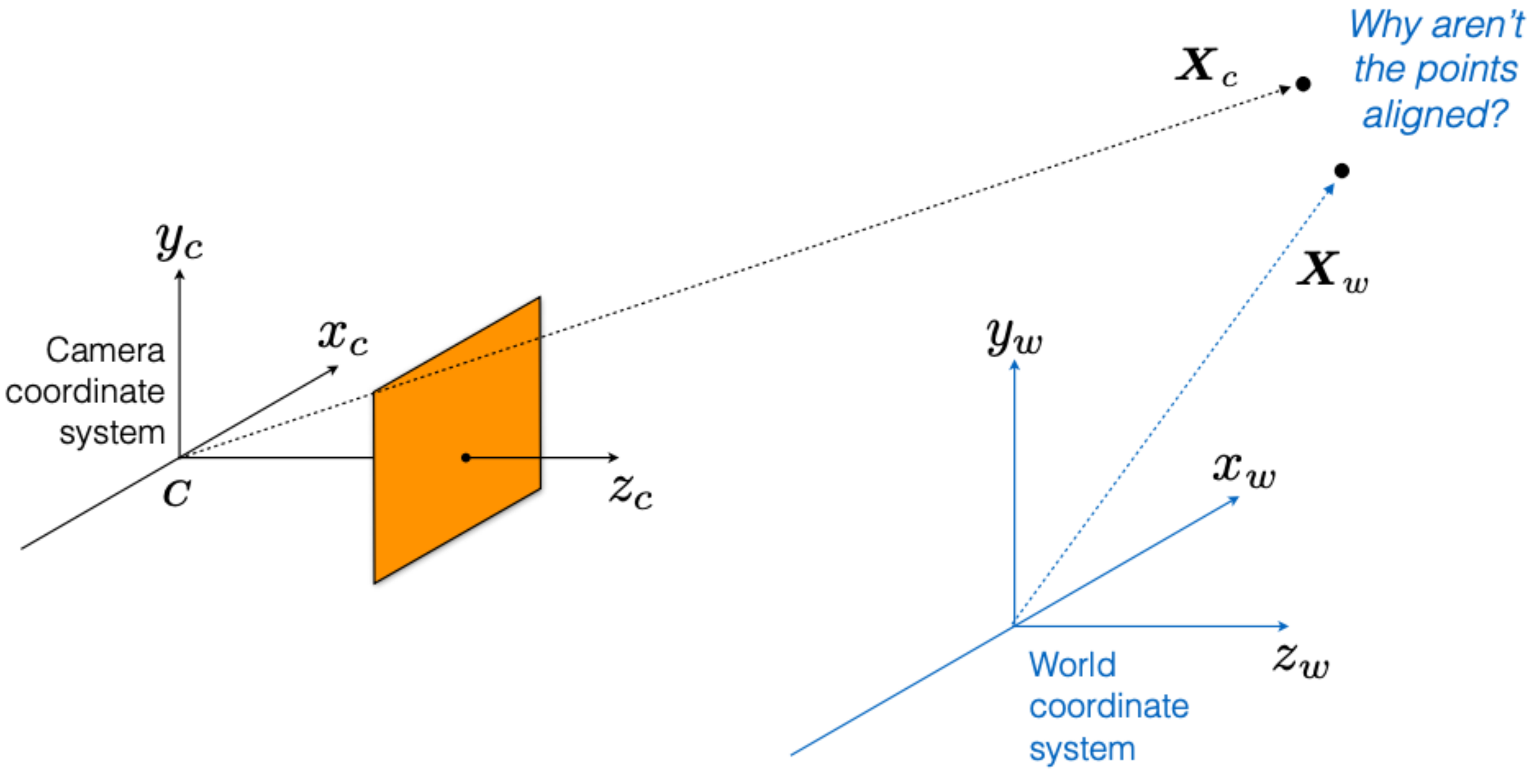


y_c
x_c
Camera coordinate system
z_c
C
Coordinate of the camera center in the world coordinate frame
X_w
y_w
x_w
z_w
World coordinate system



$$(\boldsymbol{X}_w - \mathbf{C})$$

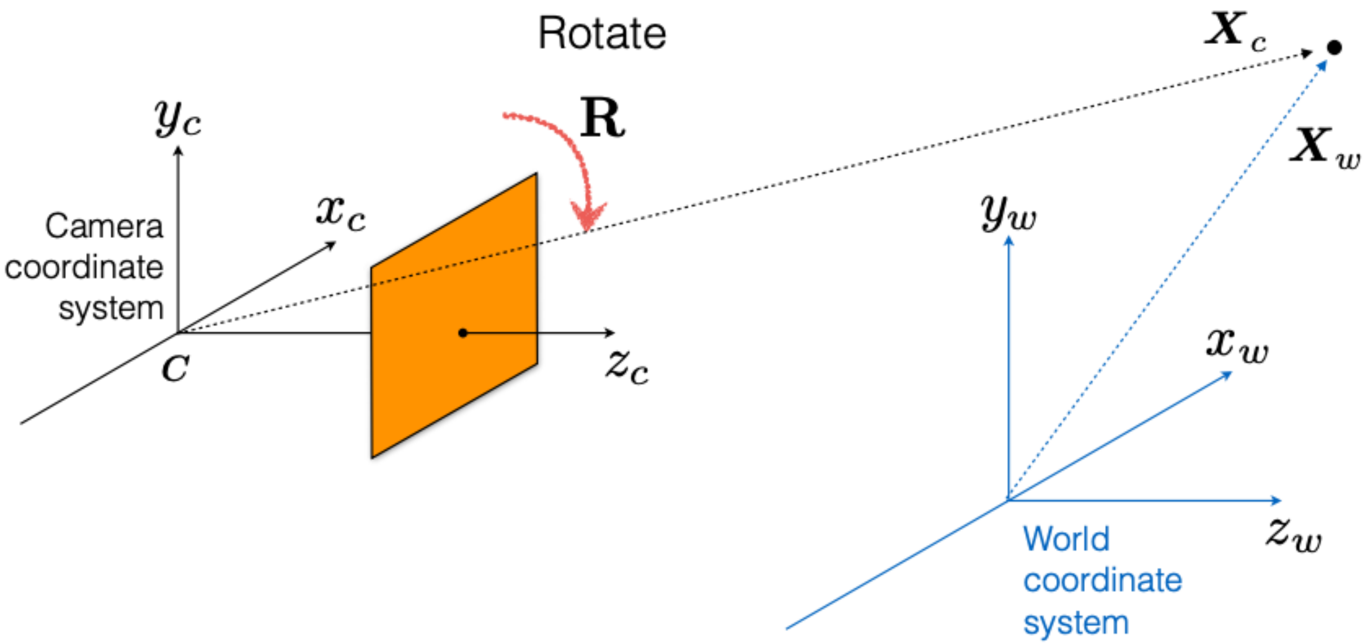
Translate



$$(\boldsymbol{X}_w - \mathbf{C})$$

Translate

What happens to points after alignment?



$$\mathbf{R}(\boldsymbol{X}_w - \mathbf{C})$$

Rotate   Translate

In inhomogeneous coordinates:

$$\boldsymbol{X}_c = \mathbf{R}(\boldsymbol{X}_w - \mathbf{C})$$

Optionally in homogeneous coordinates:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \underset{(4\times 4)}{\begin{bmatrix} \mathbf{R} & -\mathbf{RC} \\ \mathbf{0} & 1 \end{bmatrix}} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

General mapping of a pinhole camera

$$\mathbf{P} = \mathbf{KR}[\mathbf{I} | -\mathbf{C}]$$

General mapping of a pinhole camera

$$\mathbf{P} = \mathbf{KR}[\mathbf{I}| - \mathbf{C}]$$

(translate first then rotate)

Another way to write the mapping

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$$

where

$$\mathbf{t} = -\mathbf{RC}$$

(rotate first then translate)

Respond at **PollEv.com/ronisen**

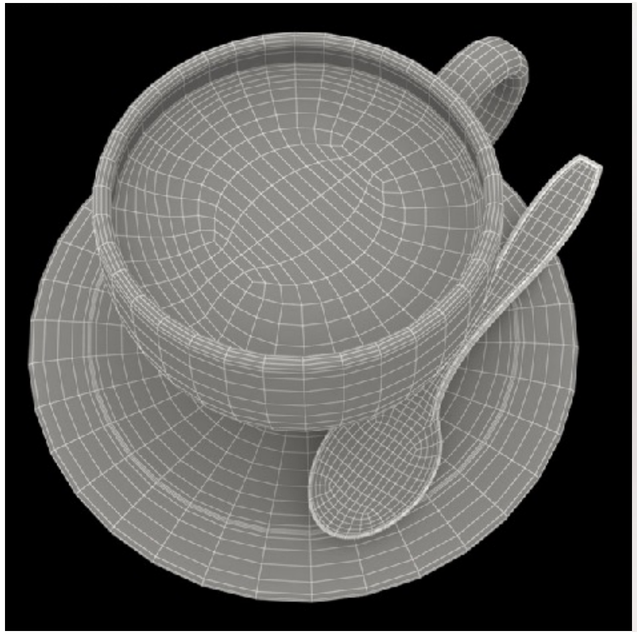
Text **RONISEN** to **22333** once to join, then text your message

Feel free to share your questions...

Agenda

- How do we define geometry/shape of an object?
- How do we define a camera model? – 3D object to 2D image
- How do we define material property? – glossy, metallic

What is Material in Computer Graphics?



3D coffee mug model

Rendered

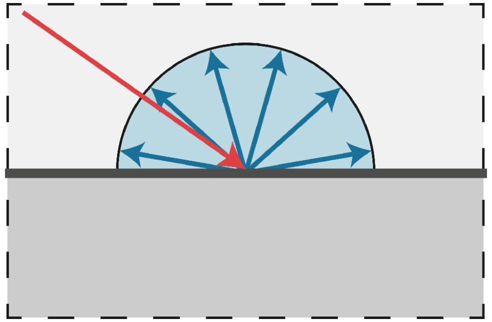
Rendered

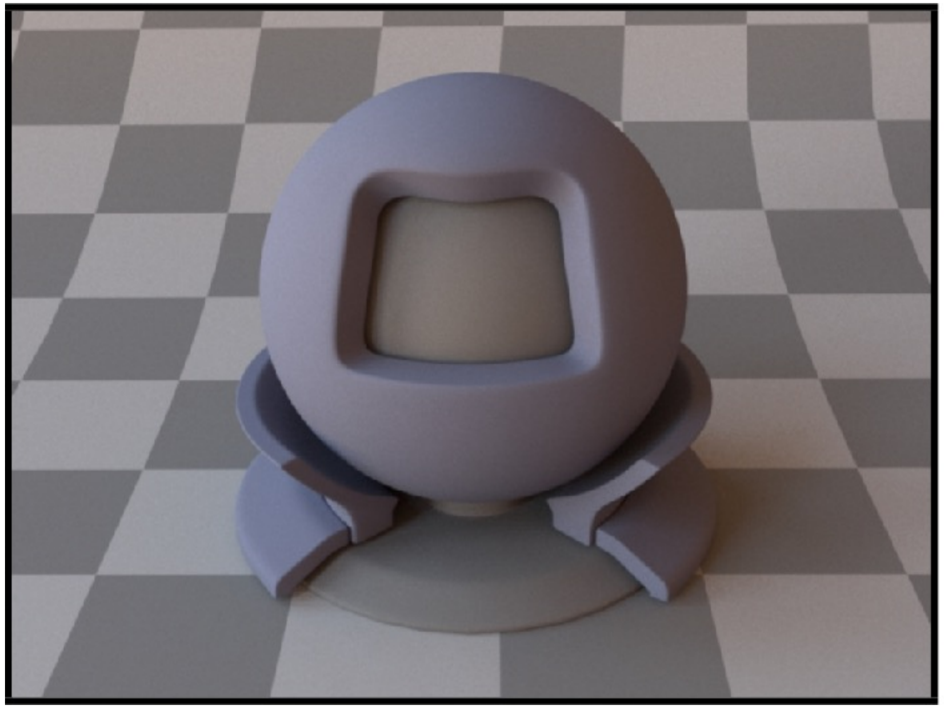
[From TurboSquid, created by artist 3dror]

Material == BRDF

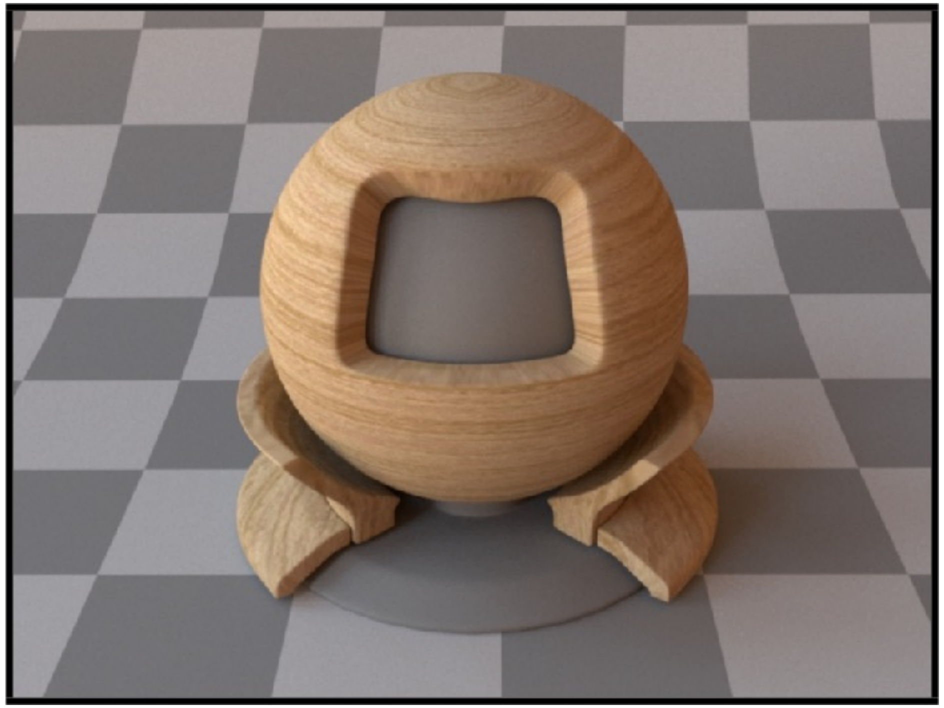
Further reading: https://en.wikipedia.org/wiki/Bidirectional_reflectance_distribution_function

Diffuse / Lambertian Material (BRDF)





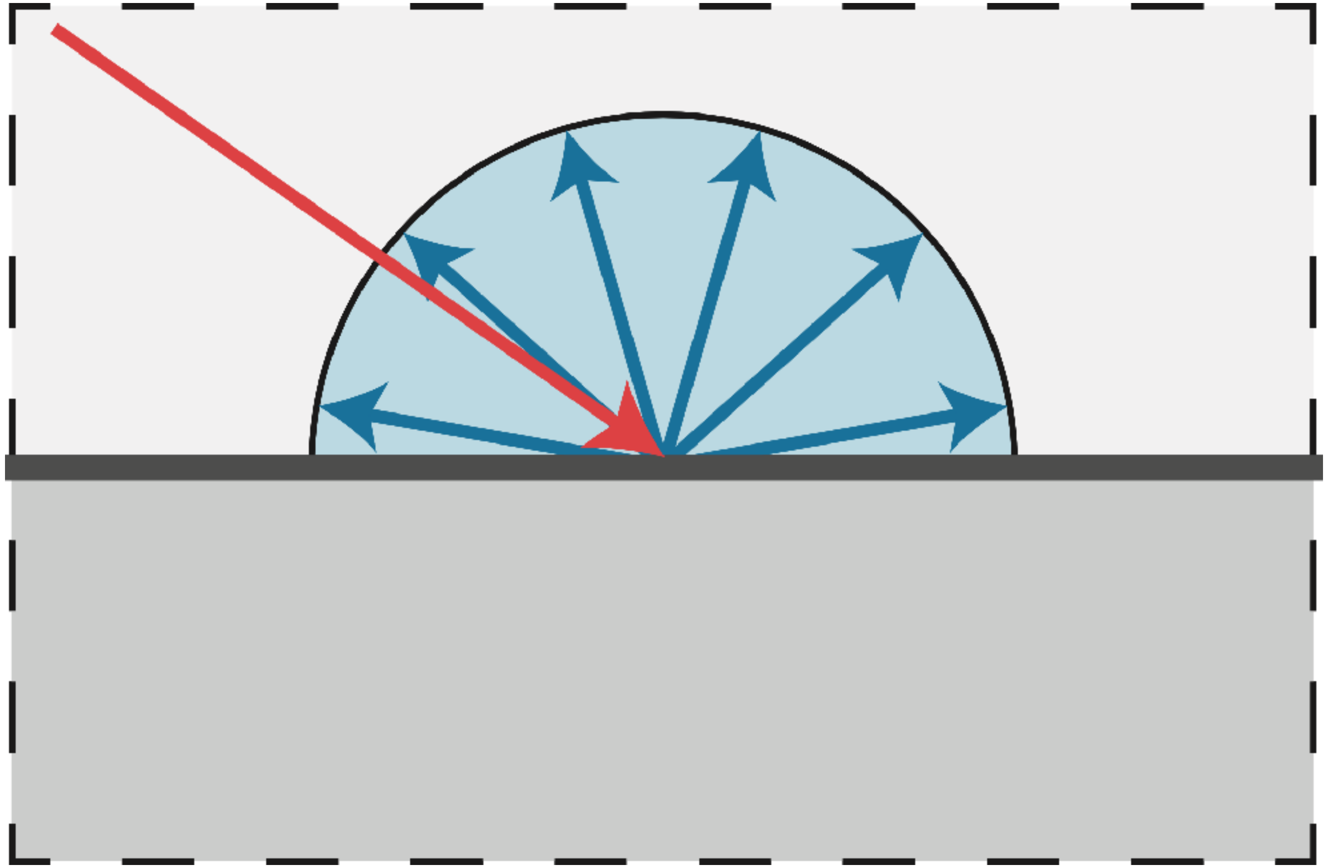
Uniform colored diffuse BRDF



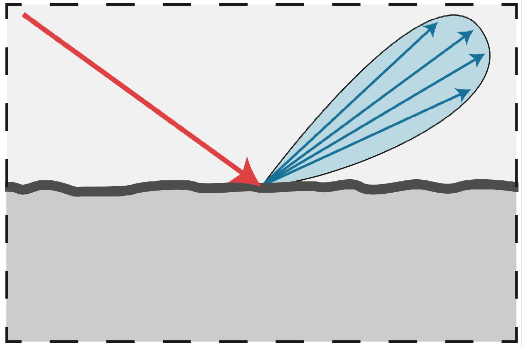
Textured diffuse BRDF

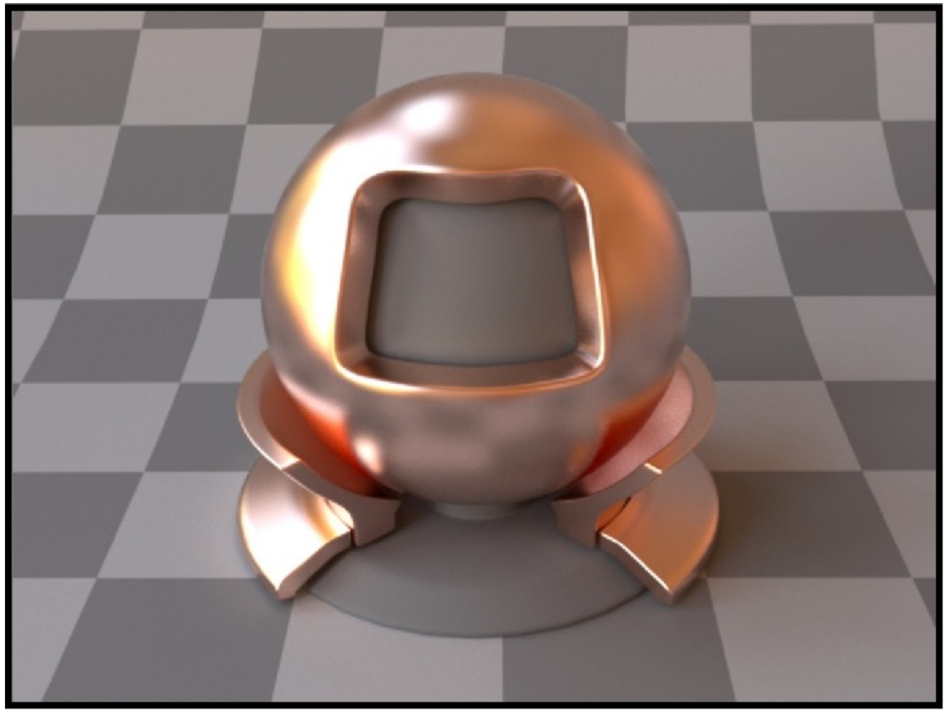
 [Mitsuba renderer, Wenzel Jakob, 2010]

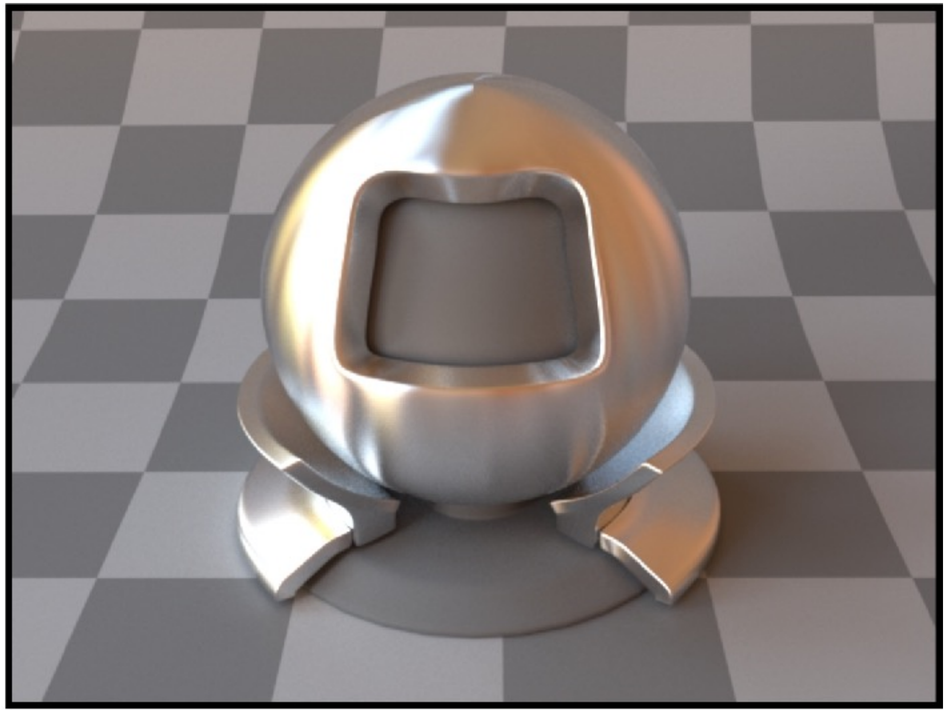
Diffuse/ Lambertian



Glossy material (BRDF)



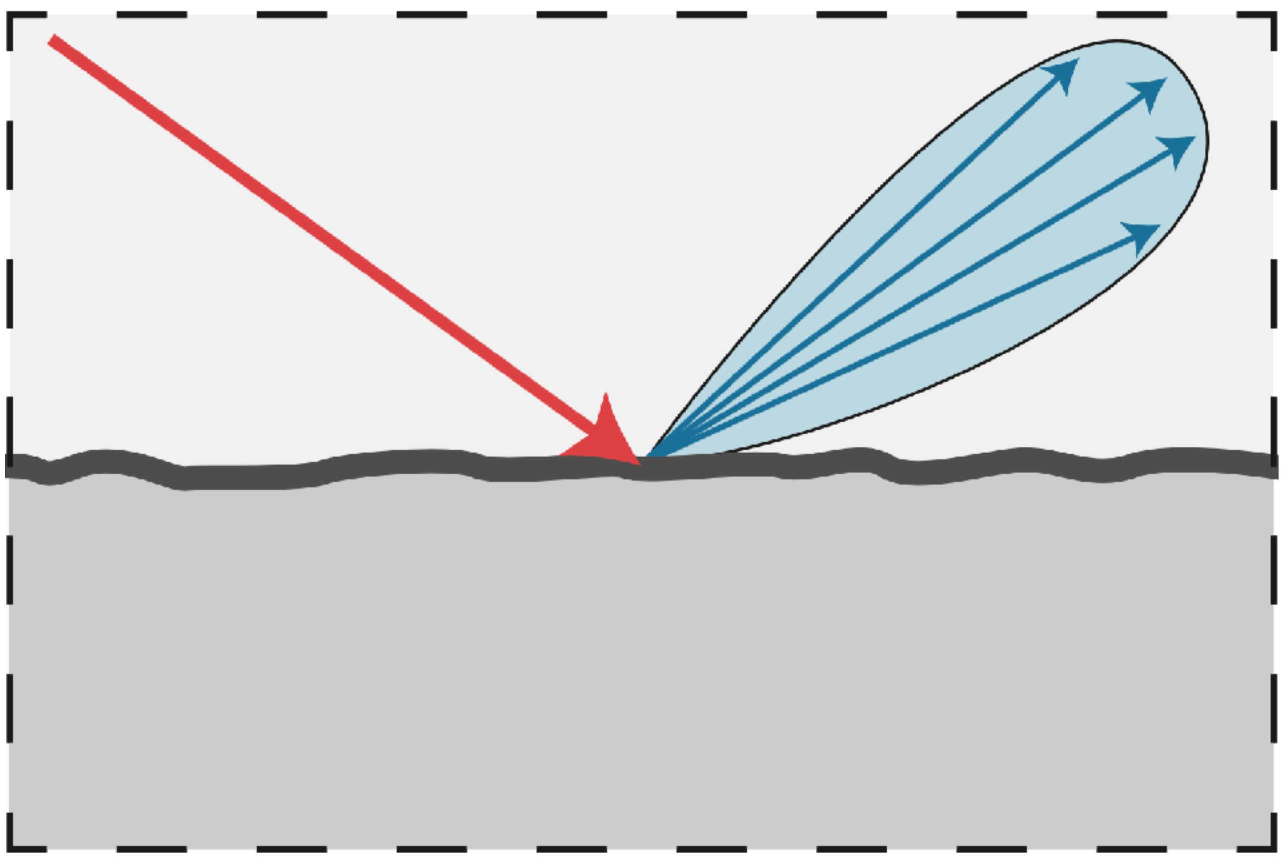




Copper **Aluminum**

[Mitsuba renderer, Wenzel Jakob, 2010]

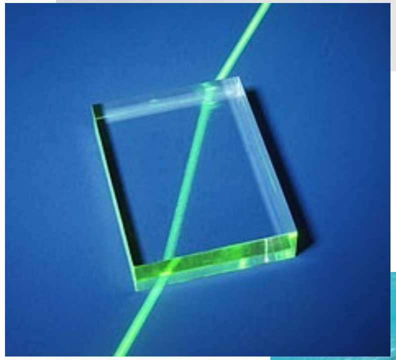
Glossy



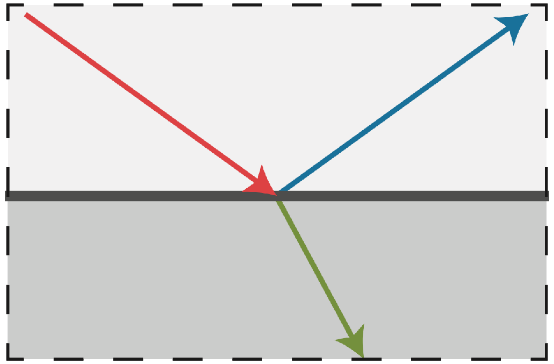
Refraction

In addition to reflecting off surface, light may be transmitted through surface.

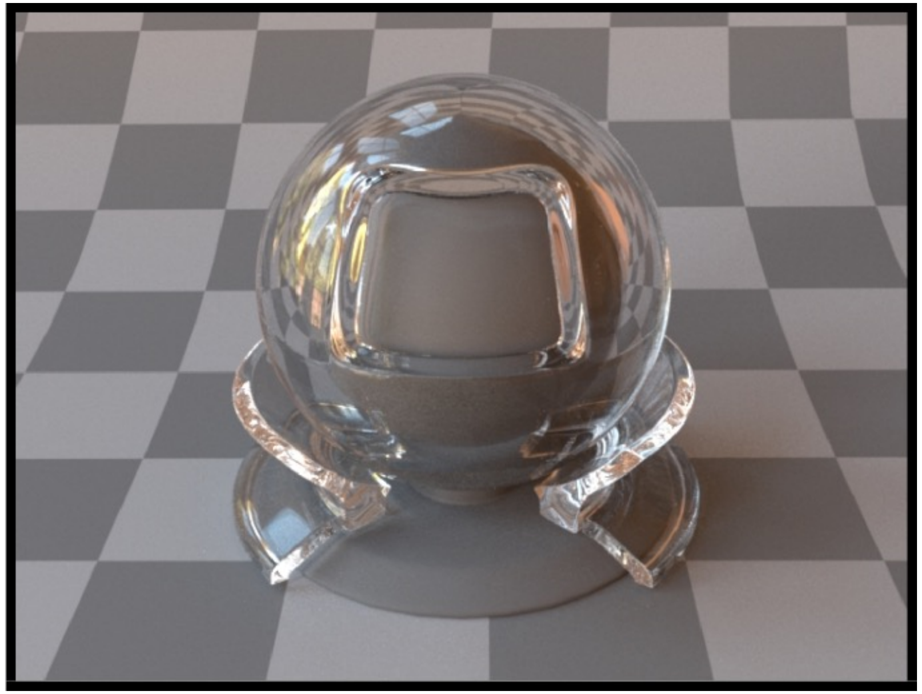
Light refracts when it enters a new medium.



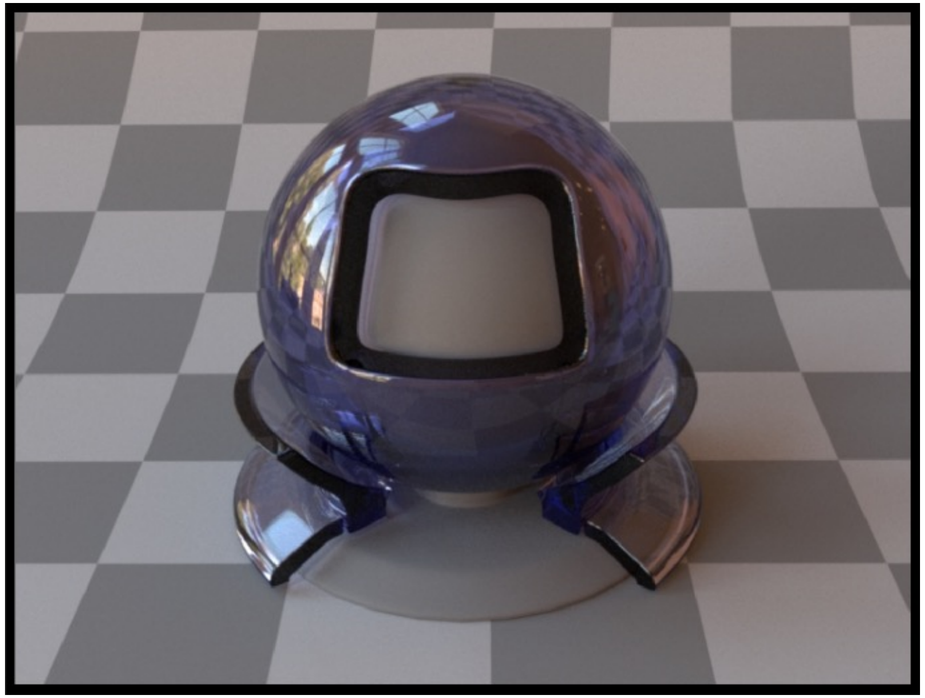
Ideal reflective / refractive material (BSDF*)



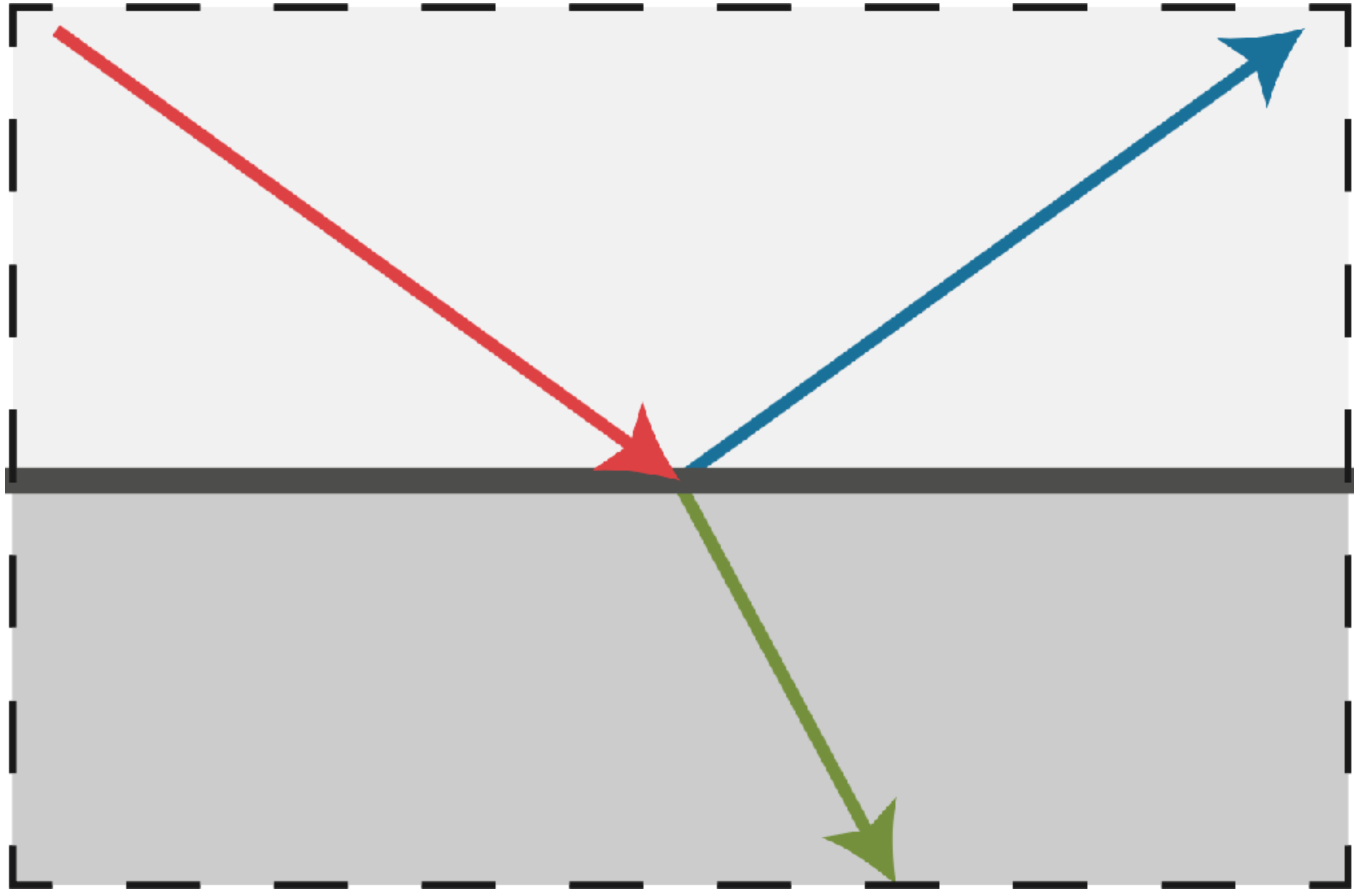
[Mitsuba renderer, Wenzel Jakob, 2010]



Air <-> plastic interface



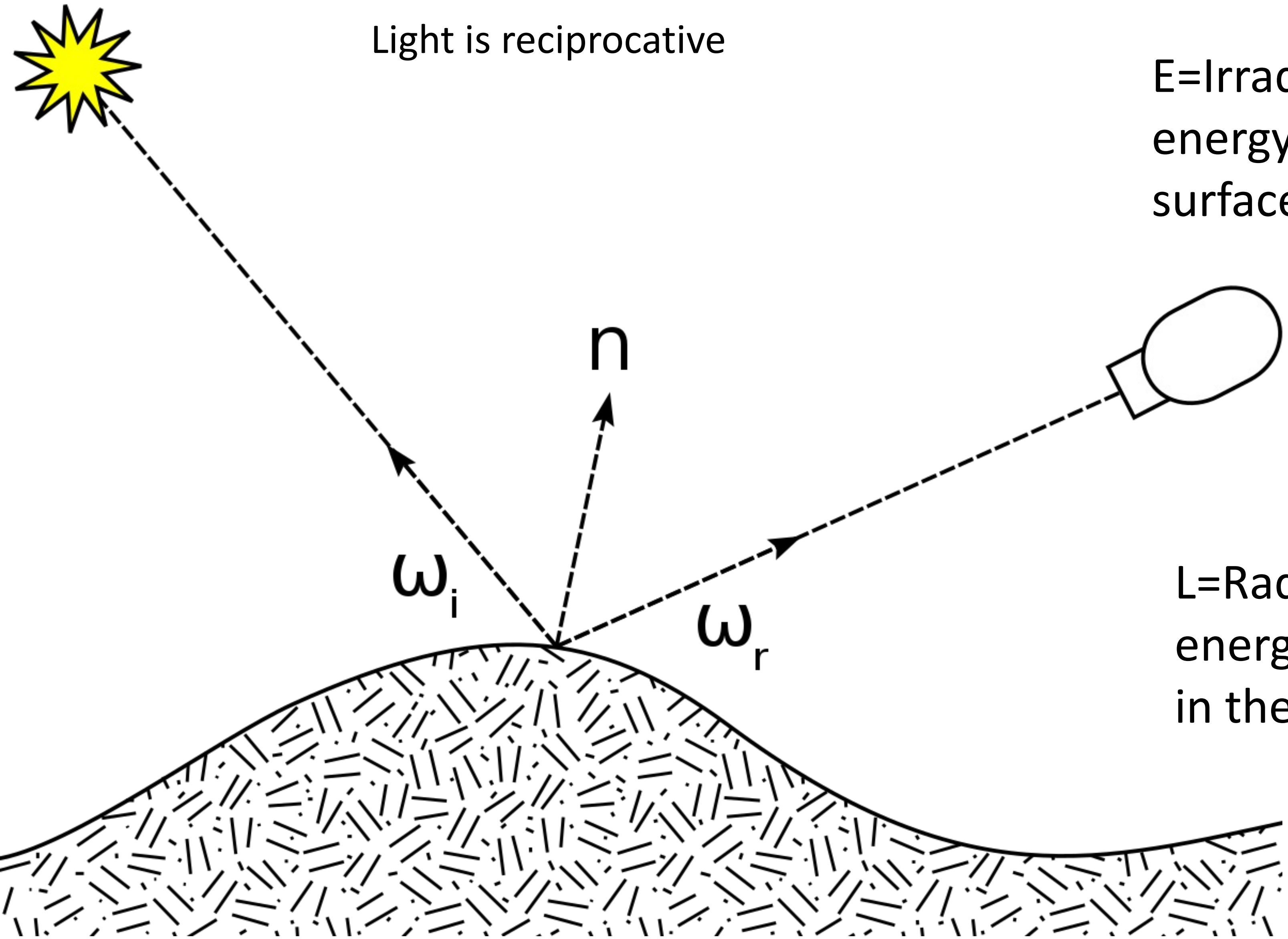
Air <-> glass interface (with absorption)



Bi-directional Radiance Distribution Function (BRDF)

Light is reciprocative

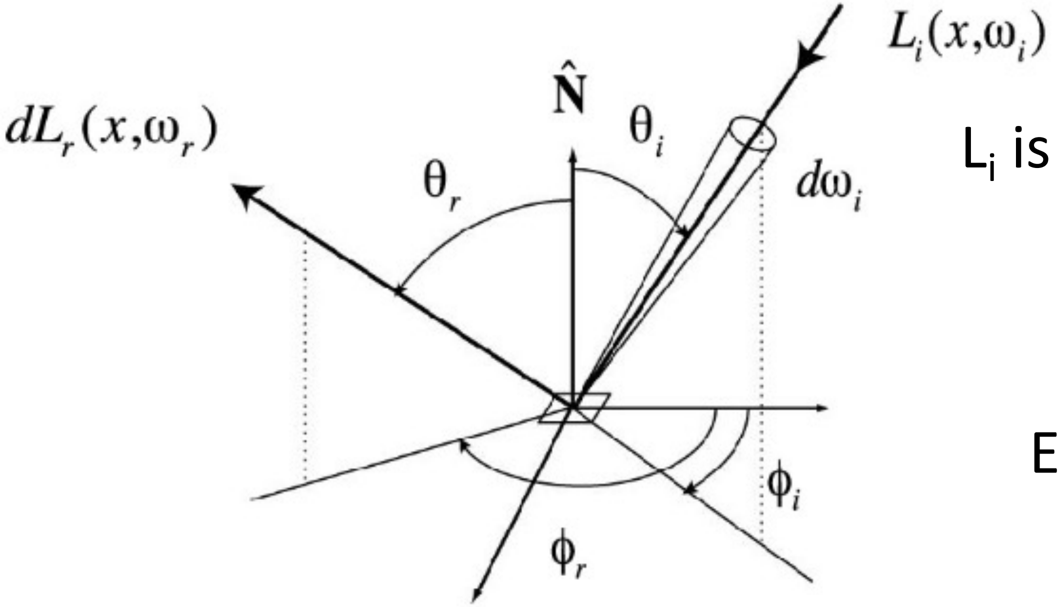
E=Irradiance.
energy per unit area received on the surface in the incoming lighting direction.



L=Radiance
energy per unit area exiting the surface. in the outgoing lighting direction.

BRDF

Definition: The bidirectional reflectance distribution function (BRDF) represents how much light is reflected into each outgoing direction ω_r from each incoming direction



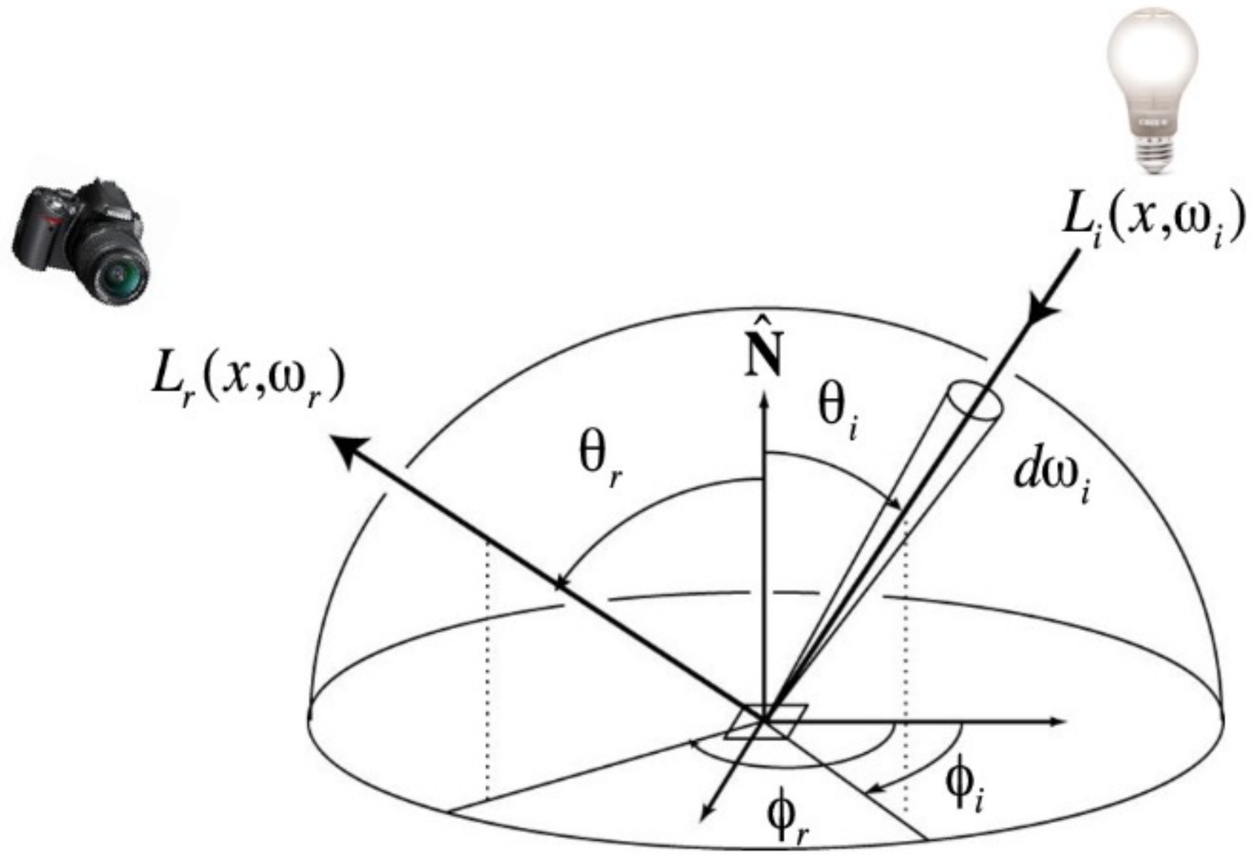
L_i is intensity of the light source.

Energy received by the surface is:

$$E_i = L_i cos\theta_i$$

$$f_r(\omega_i \to \omega_r) = \frac{\mathrm{d}L_r(\omega_r)}{\mathrm{d}E_i(\omega_i)} = \frac{\mathrm{d}L_r(\omega_r)}{L_i(\omega_i)\cos\theta_i\,\mathrm{d}\omega_i} \left[\frac{1}{\mathrm{sr}}\right]$$

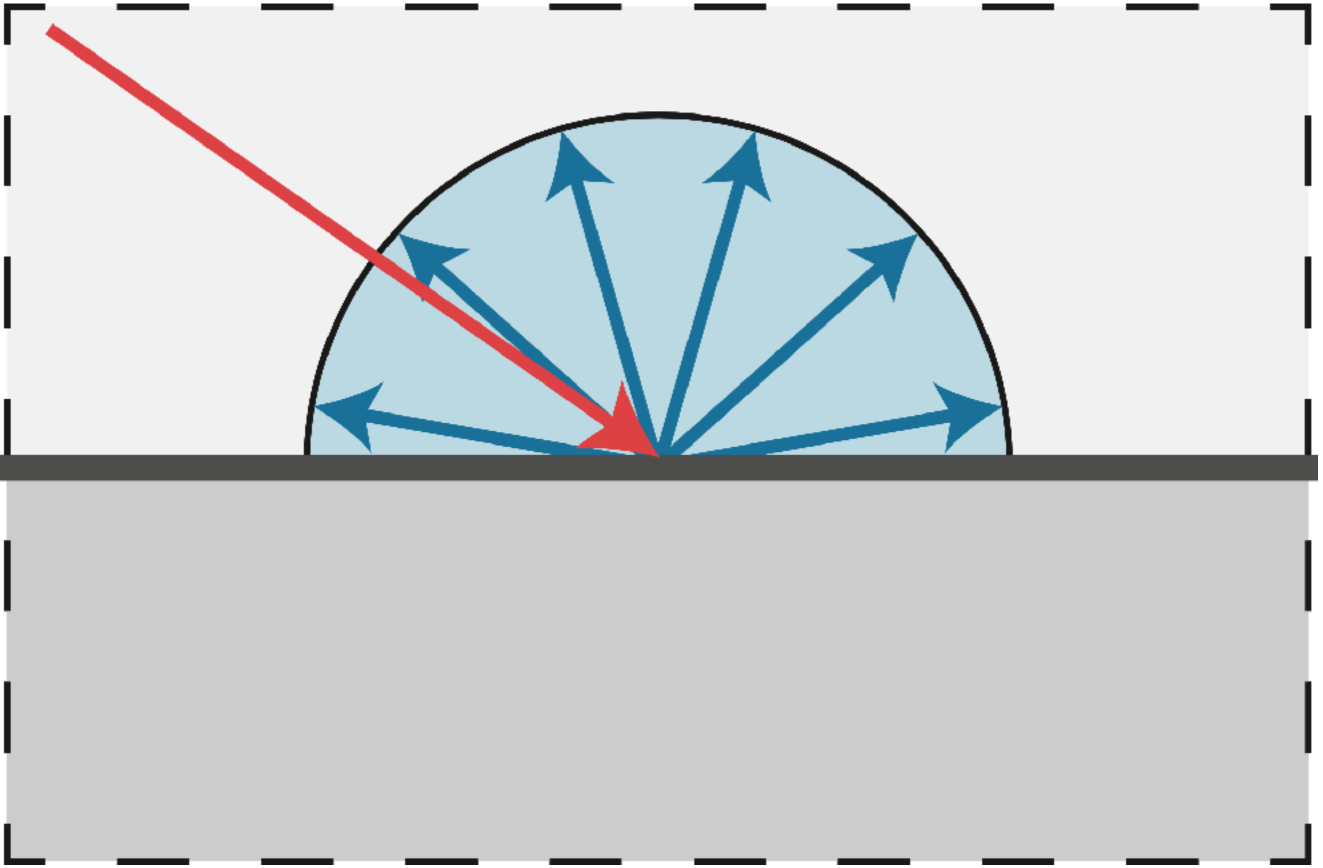
The Reflection Equation



BRDF at a point p on the surface is a 4D function of 4 angles related to incoming and outgoing lighting direction.

$$L_r(\mathrm{p}, \omega_r) = \int_{H^2} f_r(\mathrm{p}, \omega_i \rightarrow \omega_r)\, L_i(\mathrm{p}, \omega_i)\, \cos\theta_i\, \mathrm{d}\omega_i$$

Diffuse/ Lambertian



$$f(p, w_i \to w_r) = a(p)$$

a(p) is termed as Albedo.

Albedo for a HxWx3 RGB image is HxWx3

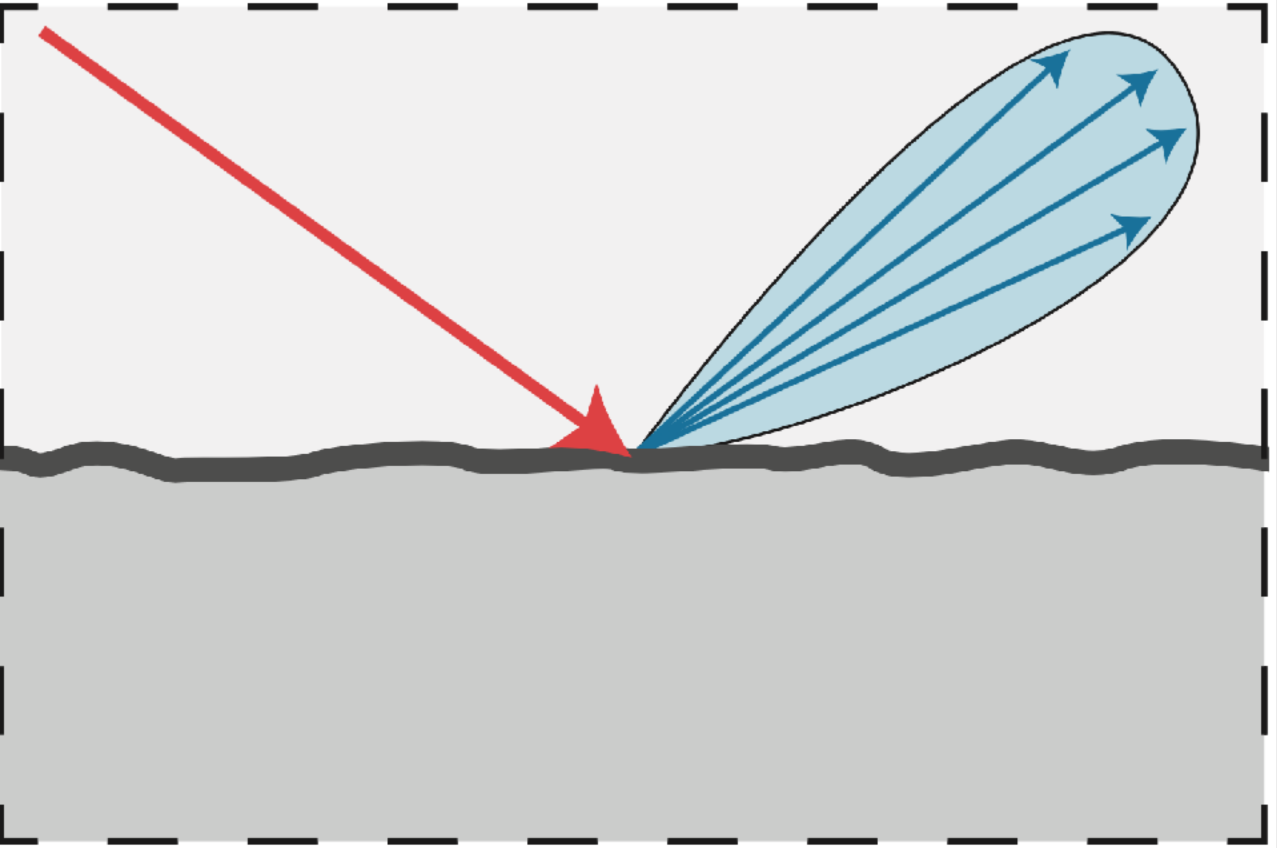
Given a point light source L_i, image intensity at pixel p can be then written as:

$$I(p) = A(p)\langle Li, N(p)\rangle = A(p)L_i cos\theta_i$$

Note: θ is the angle between surface normal N(p) and incident lighting direction L_i.

Glossy/Specular (Phong Reflectance Model)

$$f(p, w_i \rightarrow w_r) = \frac{a(p)}{\pi} + k_s \frac{\alpha + 2}{2\pi} cos^{\alpha} \theta_i$$

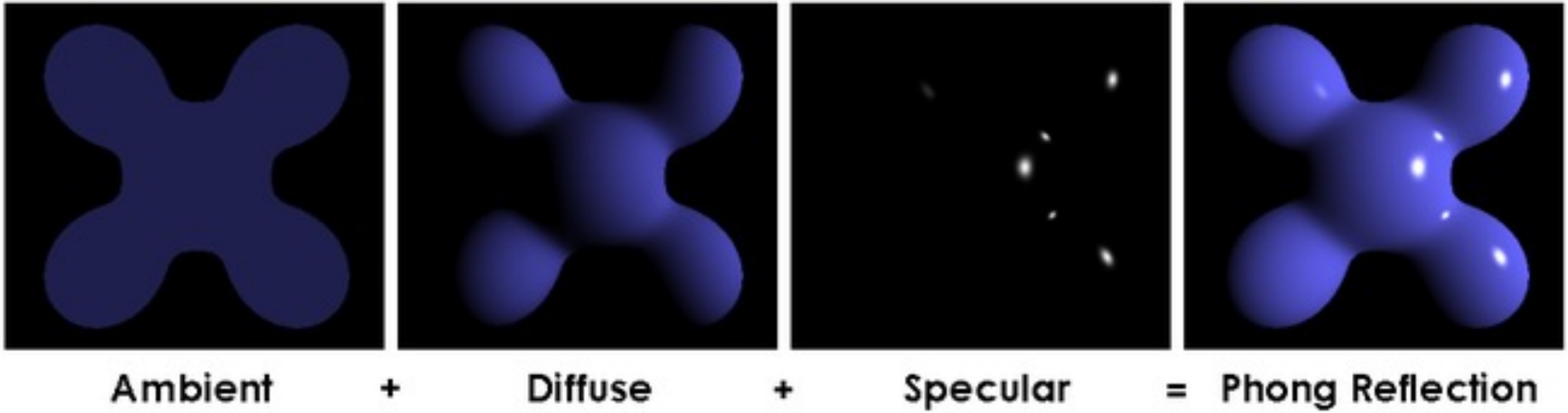


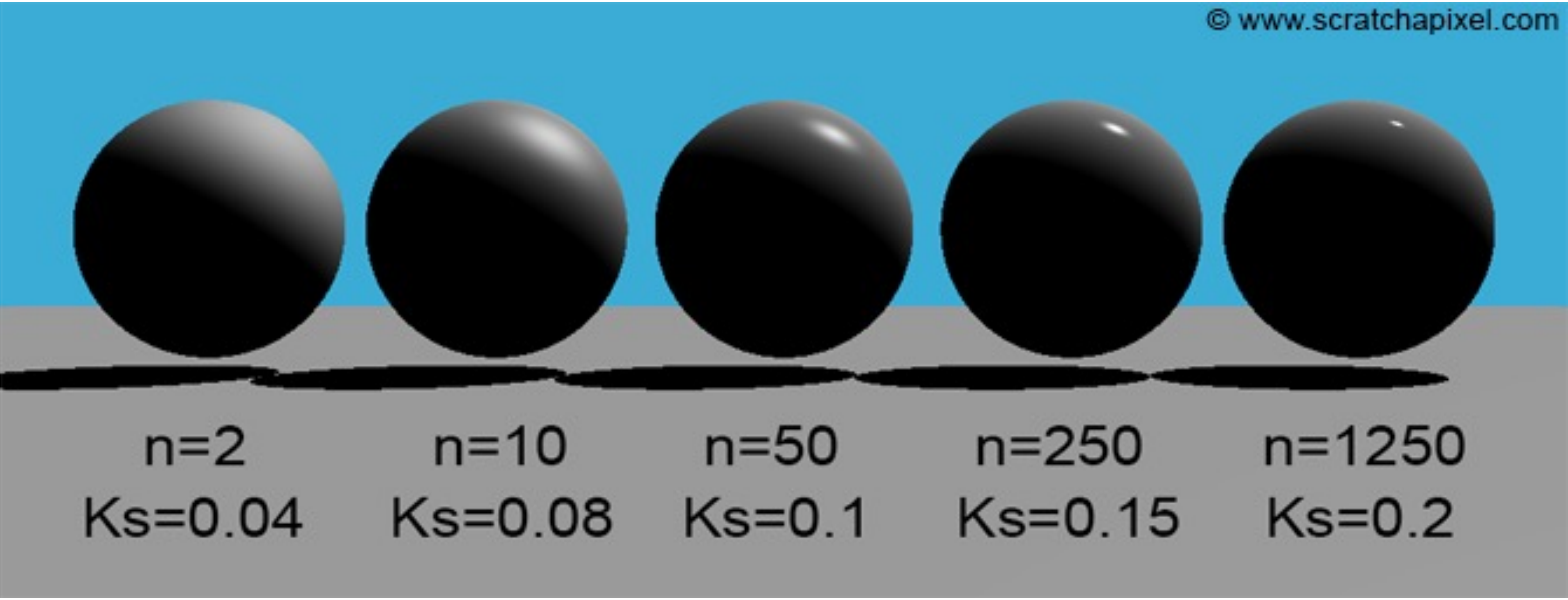
a(p) = albedo at pixel p.

K_s = specular reflectivity, controls how specular the object is.

α = specular exponent, higher value indicates sharper reflections.

Usually specular reflectivity and exponent are pixel invariant, i.e. we assume that the BRDF is spatially in-variant!


Ambient
+
Diffuse
+
Specular
=
Phong Reflection



K_s = specular reflectivity, controls how specular the object is.

α (written as n in the picture)= specular exponent, higher value indicates sharper reflections.

Microfacet Material Model

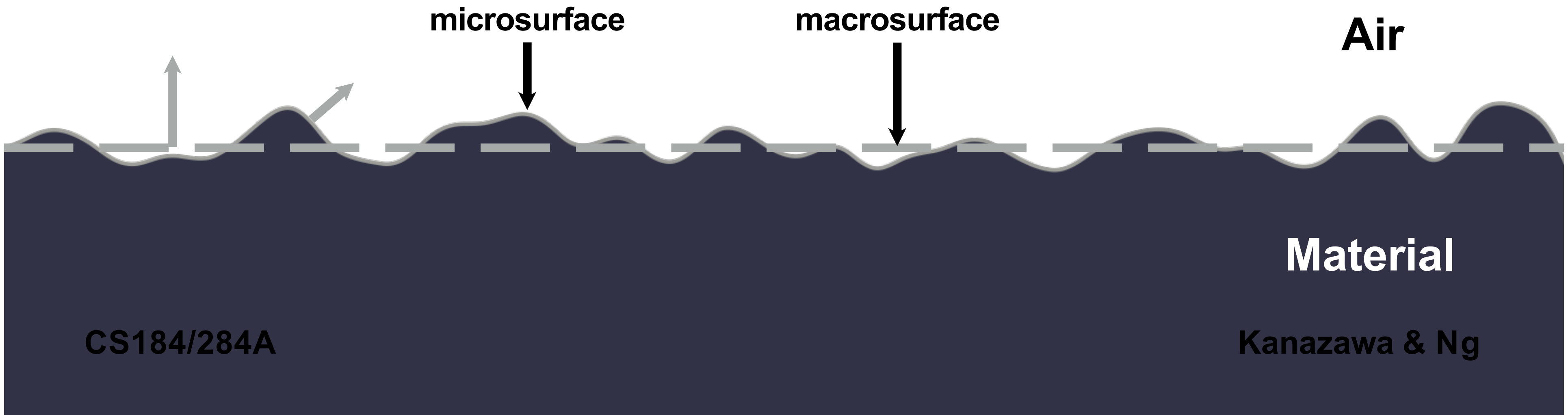
Microfacet Theory

Rough surface

- **Macroscale: flat & rough**
- **Microscale: bumpy & specular**

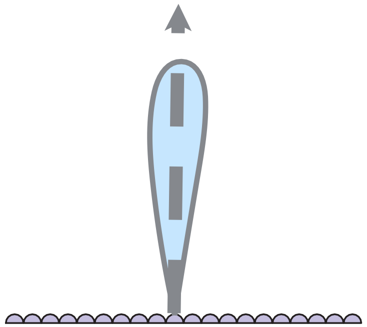
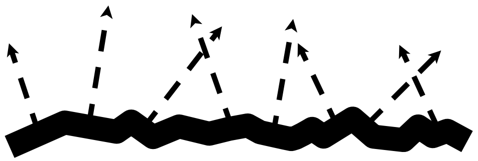
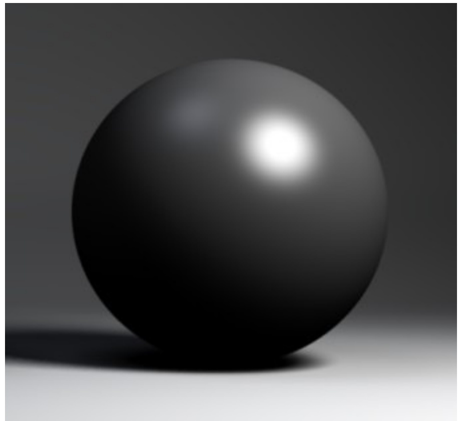
Individual elements of surface act like mirrors

- **Known as "microfacets"**
- **Each microfacet has its own normal vector (photometric normal)**

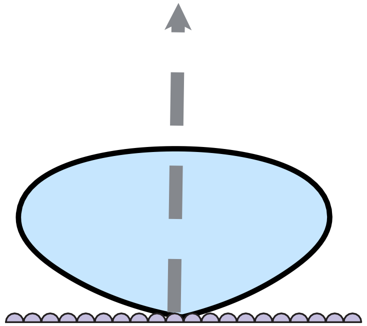


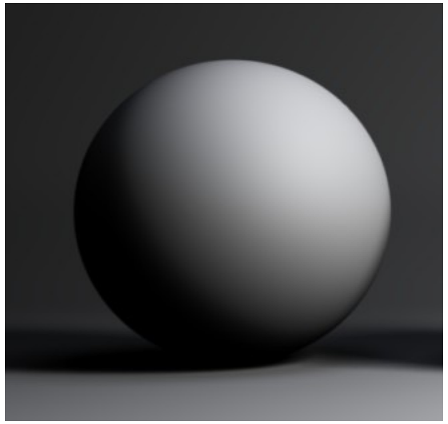
Microfacet BRDF

- **Key: the distribution of microfacets' normals**
  - **Concentrated <==> glossy**

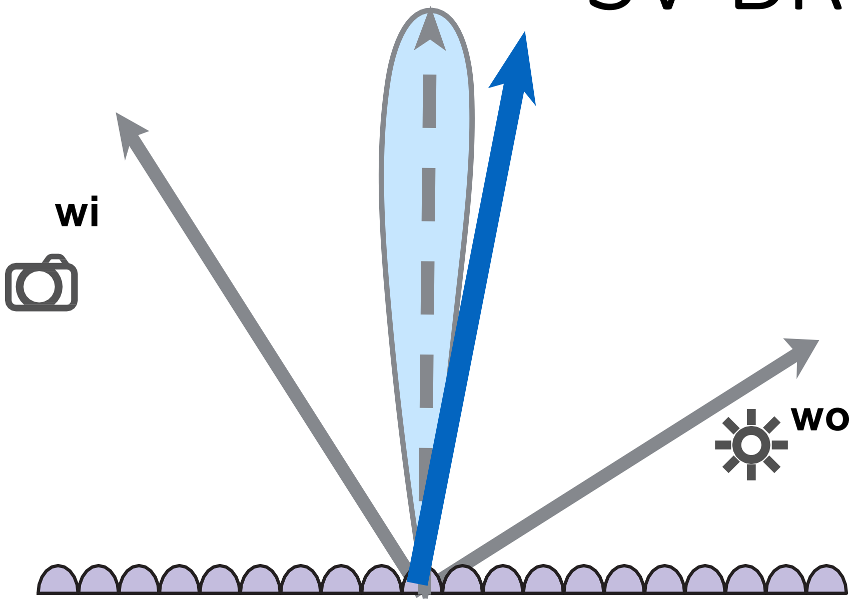




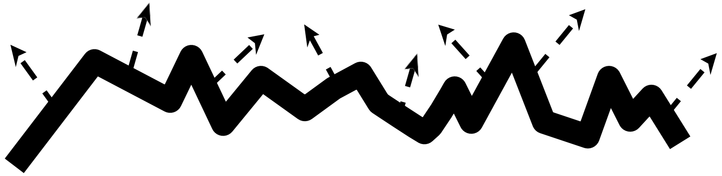
  - **Spread out <==> diffuse**





Microfacet BRDF
SV-BRDF (Spatially Varying BRDF)





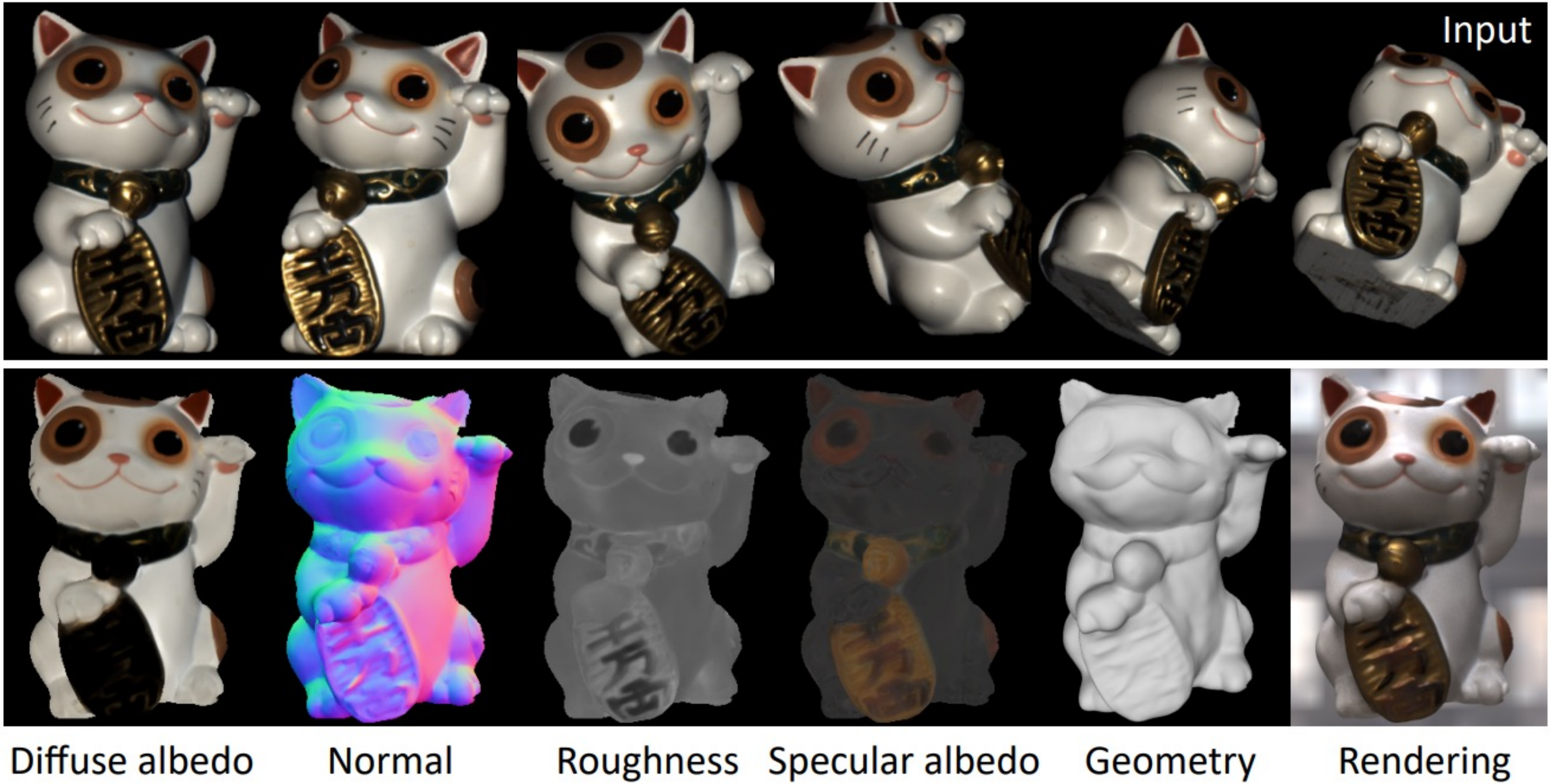
In practice we only have to define albedo per pixel A(p) and roughness R(p), a total HxWx4.

Fresnel term | masking term | Normal distribution function

$$f(\mathbf{i}, \mathbf{o}) = \frac{\mathbf{F}(\mathbf{i}, \mathbf{h})\,\mathbf{G}(\mathbf{i}, \mathbf{o}, \mathbf{h})\,\mathbf{D}(\mathbf{h})}{4(\mathbf{n}, \mathbf{i})(\mathbf{n}, \mathbf{o})}$$

Microfacet BRDF: Examples

[Autodesk Fusion 360]



Deep 3D Capture: Geometry and Reflectance from Sparse Multi-View Images, Bi et. al.

Isotropic vs Anisotropic Reflection

- **So far, Point light + Metal = Round / Elliptical highlight**
- **What can we see inside many metal elevators?**

Inside an elevator

Isotropic vs Anisotropic Reflection

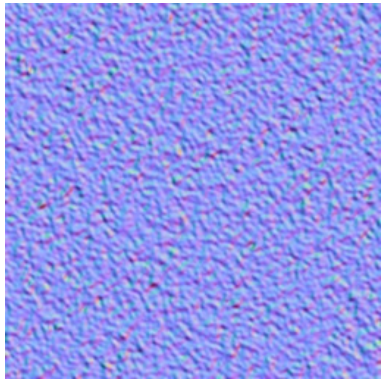
Isotropic

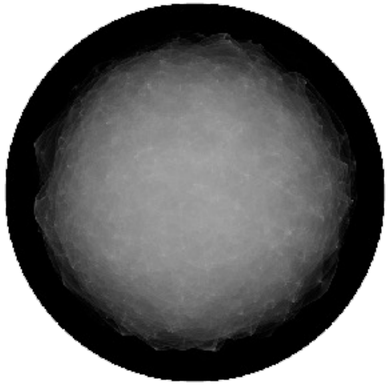
Anisotropic

Isotropic / Anisotropic Materials (BRDFs)

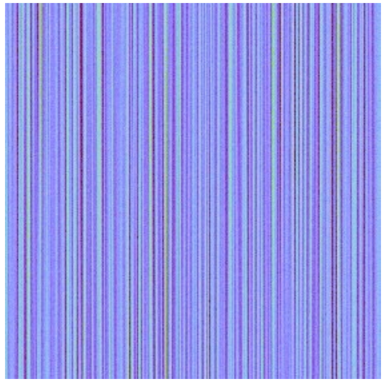
- **Key: directionality of underlying surface**

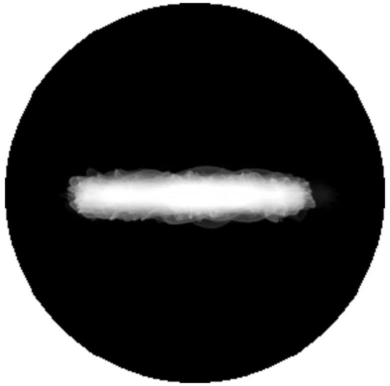
Isotropic





Anisotropic



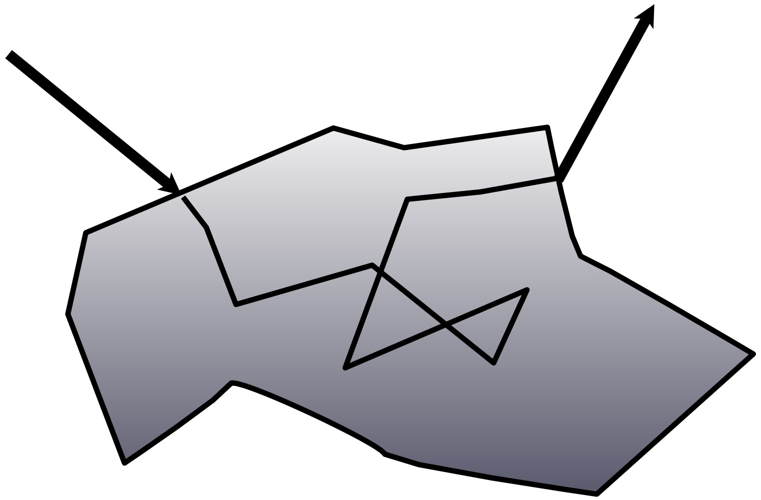


Surface (normals) **BRDF (fix wi, vary wo)**

Subsurface Scattering

Visual characteristics of many surfaces caused by light exiting at different points than it enters

- Violates a fundamental assumption of the BRDF



- Different from transparent

[Jensen et al 2001]

[Donner et al 2008]

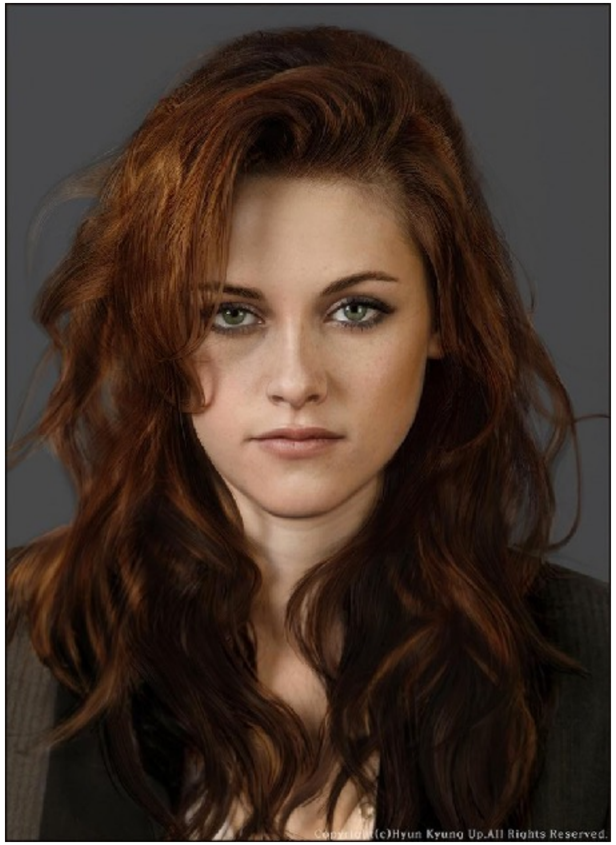
BRDF vs BSSRDF (models sub-surface scattering)

BRDF BSSRDF

[Jensen et al. 2001]

BSSRDF: Application

[Artist: Teruyuki and Yuka]



[Artist: Hyun Kyung]

[Artist: Dan Roarty]

https://cgelves.com/10-most-realistic-human-3d-models-that-will-wow-you/

Respond at **PollEv.com/ronisen**

Text **RONISEN** to **22333** once to join, then text your message

Feel free to share your questions...

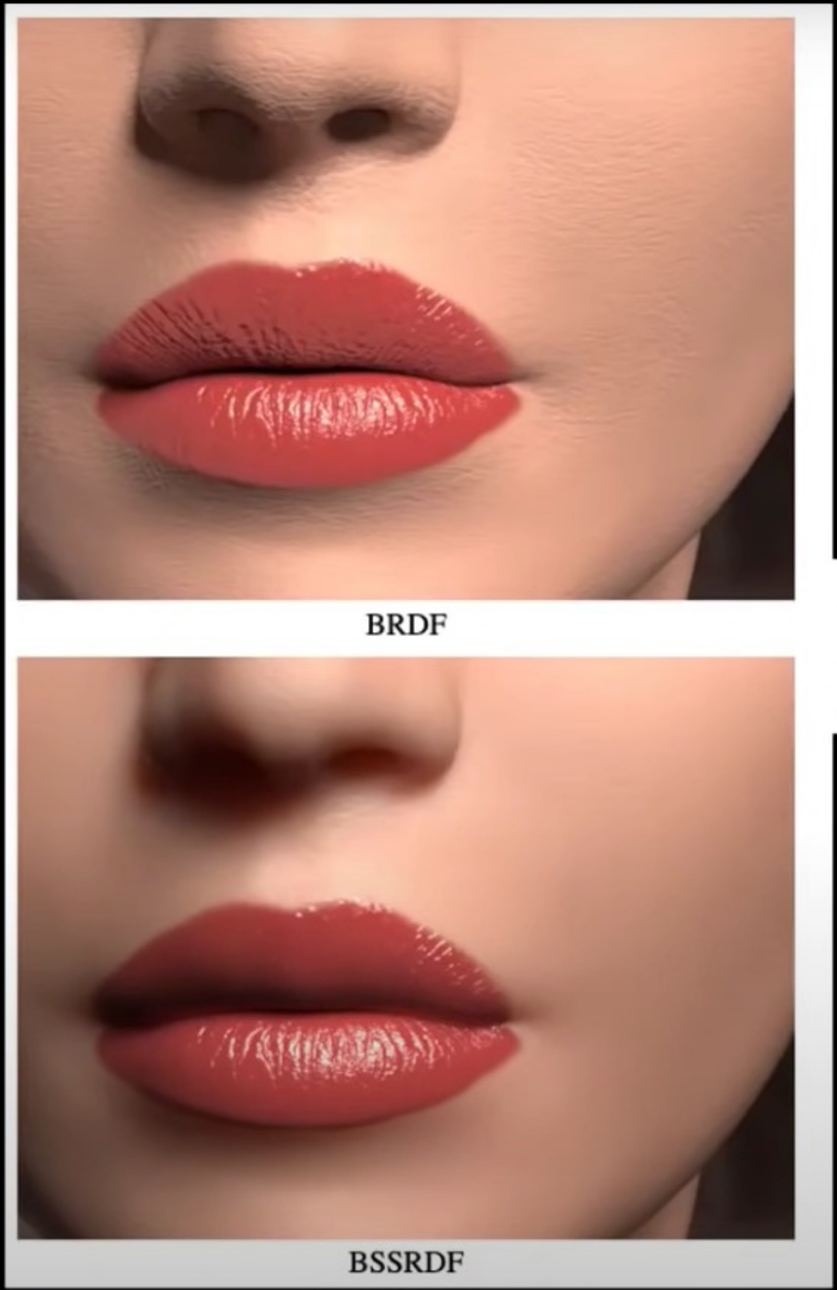
To be continued …

Anti Racist Computer Graphics

- by Theodore Kim, Yale Univ.

Computer Graphics has a race problem!

Listen to this 1hour presentation by Prof. Kim: https://www.youtube.com/watch?v=ROuE8xYLpX8

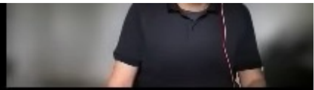


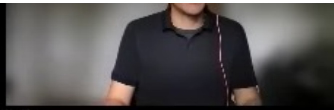
“Skin” = Subsurface Scattering

Figure 11: A face rendered using the BRDF model (top) and the BSSRDF model (bottom). We used our measured values for skin (skin1) and the same lighting conditions in both images (the BRDF

“Skin” = White Skin

Jensen, Marschner, Levoy and Hanrahan, *A Practical Model for Subsurface Light Transport*, Proceedings of SIGGRAPH (2001).





Jensen, Marschner, Levoy and Hanrahan, *A Practical Model for Subsurface Light Transport*, Proceedings of SIGGRAPH (2001).
Stam, *An Illumination Model for a Skin Layer Bounded by Rough Surfaces*, Rendering Techniques (2001).
Jensen and Buhler, *A Rapid Hierarchical Rendering Technique for Translucent Materials*, ACM Transactions on Graphics (2002).
Xie, Olano, Karis, Narkowicz, *Real-time Subsurface Scattering with Single Pass Variance-guided Adaptive Importance Sampling*, Proceedings of the ACM on Computer Graphics and Interactive Techniques (2020).
D'Eon, Luebke, Enderton , *Efficient Rendering of Human* Skin, Rendering Techniques (2007).

Jimenez, Zsolnai, Jarabo, Freude, Auzinger, We, von der Pahlen, Wimmer, Gutierrez, *Separable Subsurface Scattering*, Computer Graphics Forum (2015).
d'Eon, Irving, *A Quantized-Diffusion Model for Rendering Translucent Materials*, ACM Transactions on Graphics (2011).
Frederickx, Dutre, *A Forward Scattering Dipole Model from a Functional Integral Approximation*, ACM Transactions on Graphics (2017).
Donner and Jensen, *Light Diffusion in Multi-Layered Translucent Materials*, ACM Transactions on Graphics (2005).
Jimenez, Scully, Barbosa, Donner, Alvarez, Vieira, Matts, Orvalho, Gutierrez, Weyrich, *A Practical Appearance Model for Dynamic Facial Color*, ACM Transactions on Graphics (2010).
Habel, Christensen, Jarosz, *Photon Beam Diffusion: A Hybrid Monte Carlo Method for Subsurface Scattering*, Eurographics Symposium on Rendering (2013).

Whiter skin has more subsurface scattering, leading to more smoothing effect.

Darker skin has more specular reflection and less subsurface scattering.

Clockers, (1995).

"MetaHuman Creator Documentation." *Unreal Engine*. Accessed August 4, 2021. https://docs.metahuman.unrealengine.com/.

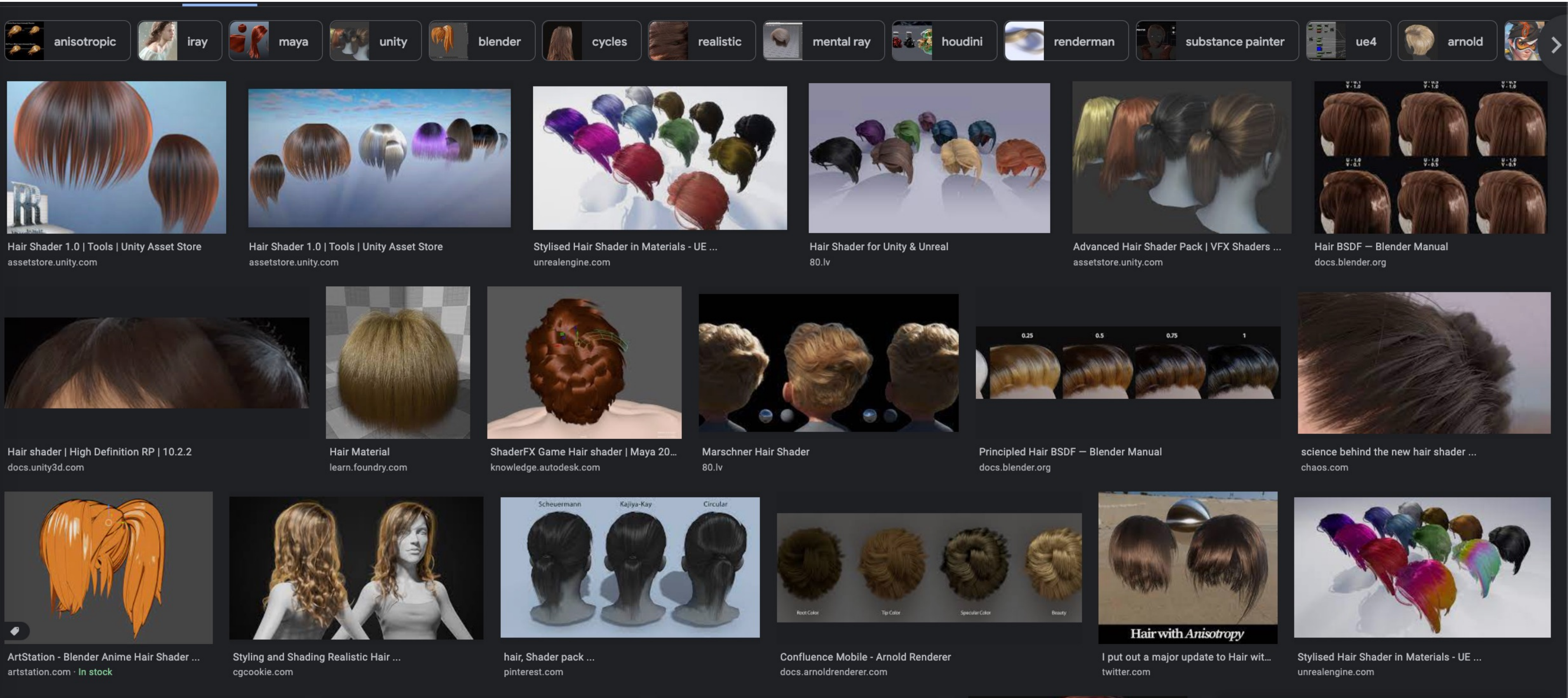
Clockers, (1995).

"MetaHuman Creator Documentation." *Unreal Engine*. Accessed August 4, 2021. https://docs.metahuman.unrealengine.com/.

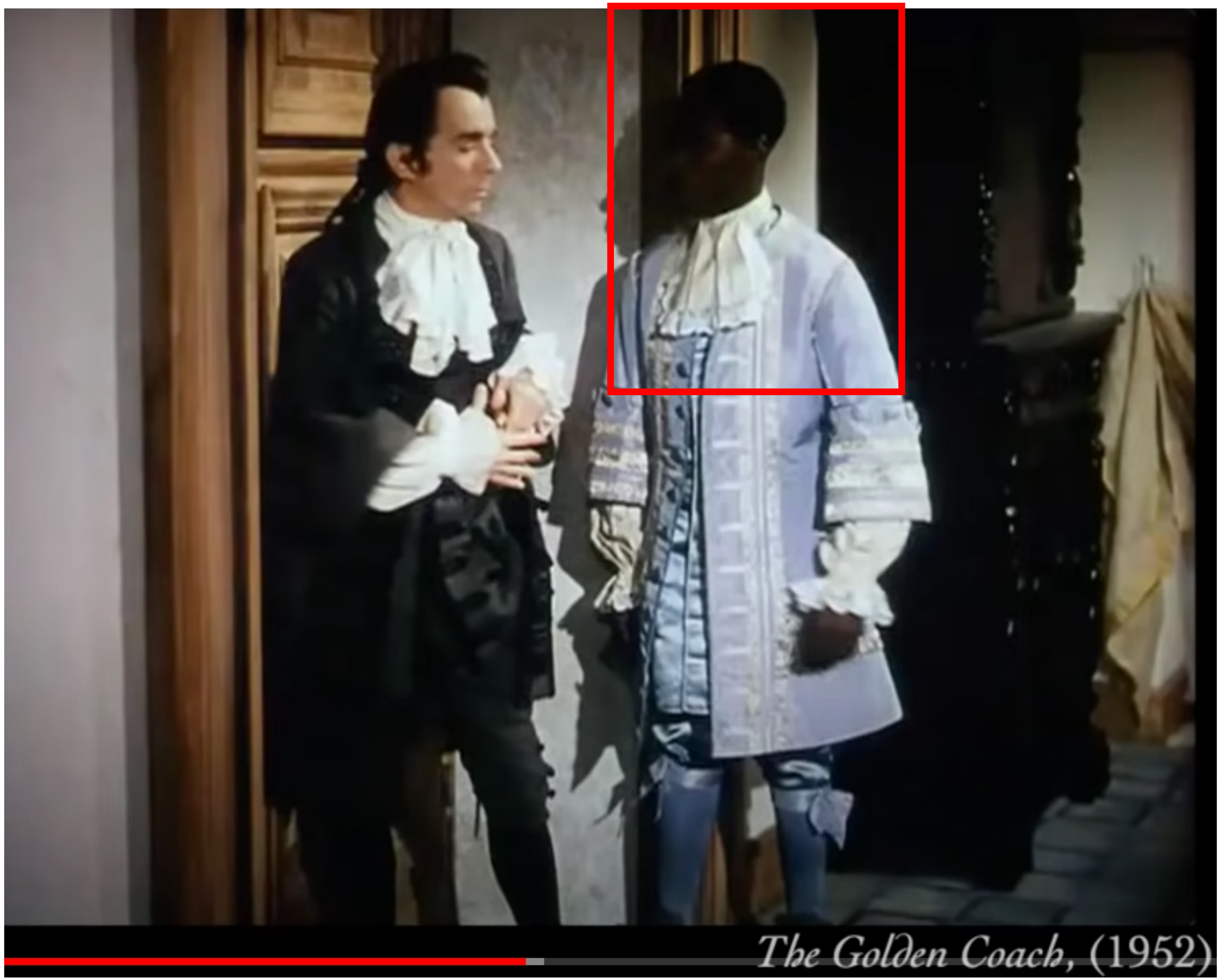
Black Panther, (2018).

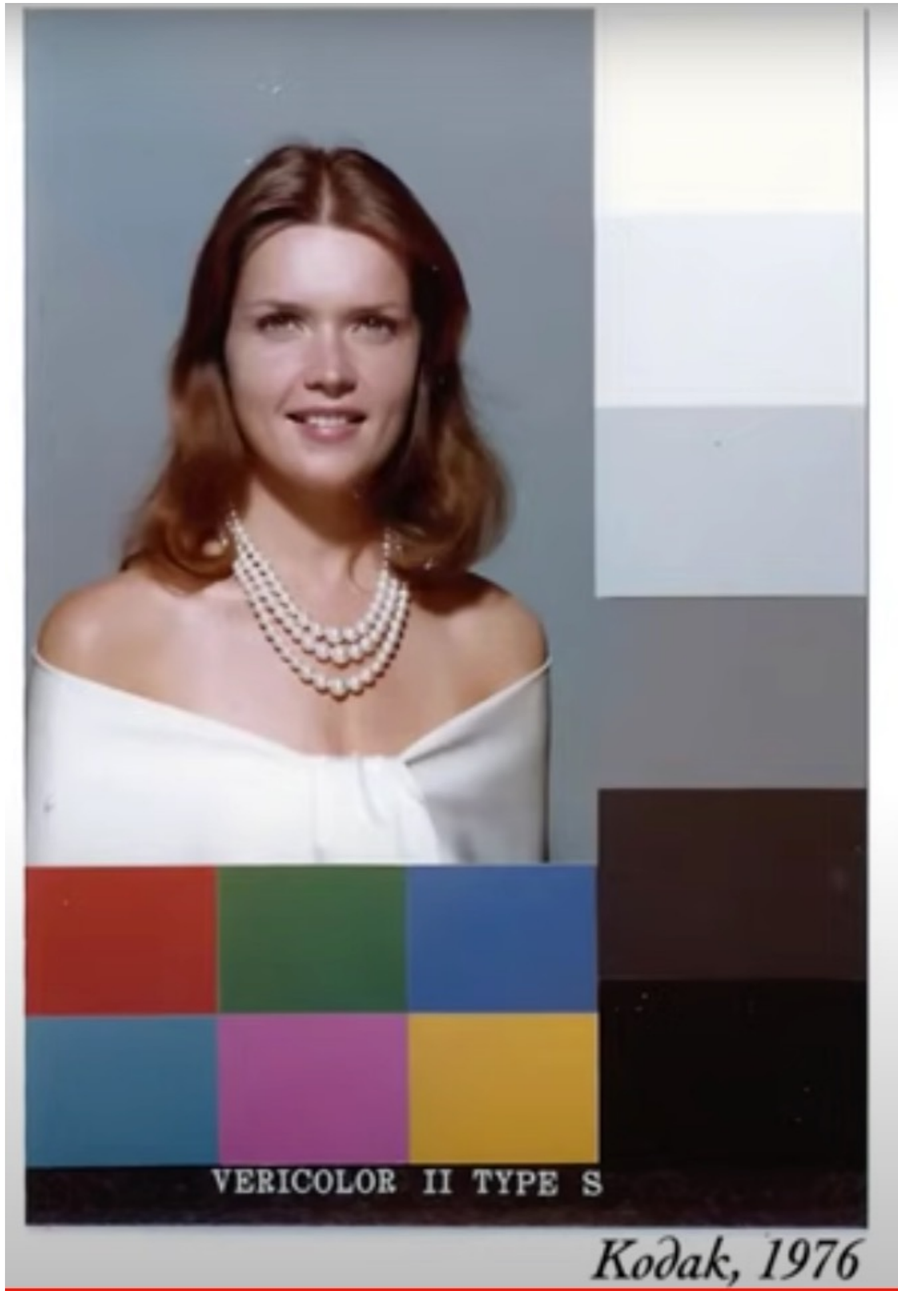
Other racial bias in Computer Graphics?

- Hair = ‘Straight hair’



Camera color tone bias





So what do we do now?

- Incentivize creating good dataset and benchmarks that is diverse and inclusive of all race, ethnicity, genders, disability status etc.
- Discourage working on research problems that are going to potentially cause harm to marginalized community, e.g. detecting sexual orientation from images.
- If working on specific subpopulation, make sure to clarify that in the paper, e.g. write ‘whiter skin tone’ instead of ‘human skin tone’.