Agenda

8.1 Shape-from-Shading

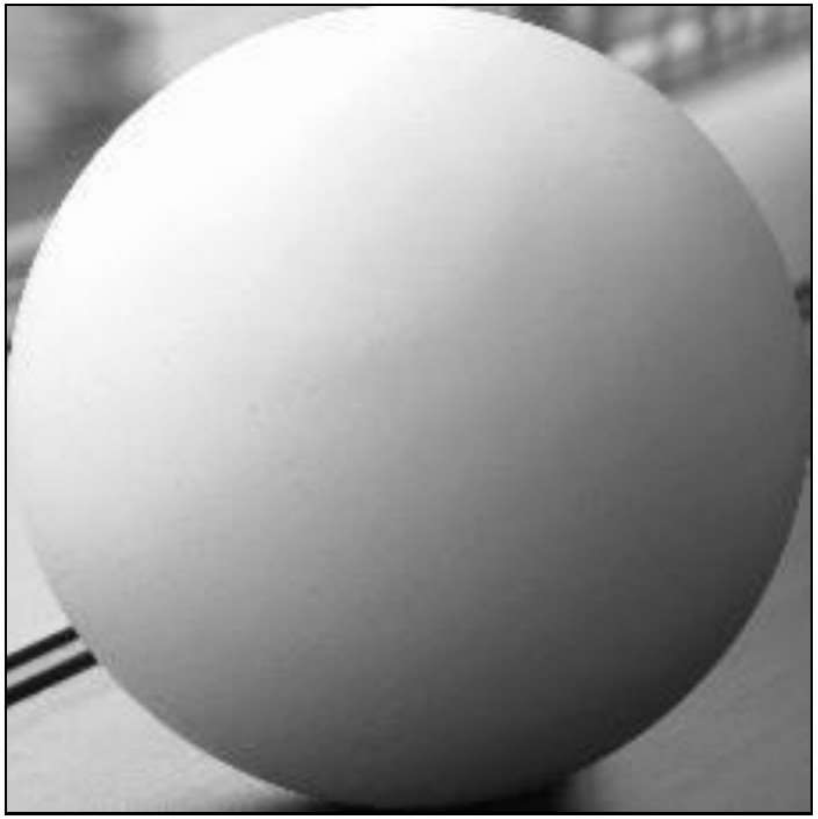
8.2 Photometric Stereo

8.3 Shape-from-X

8.4 Volumetric Fusion

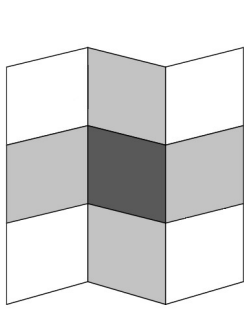
8.1
Shape-from-Shading

Can we recover shape from shading?

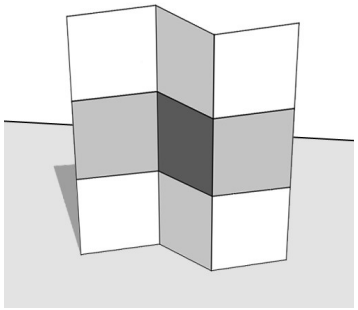


- What is the relation between **intensity** and **shape**? Can we recover shape?

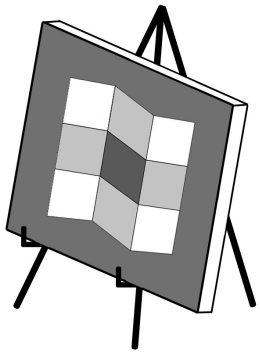
Adelson and Pentland's Workshop Metaphor



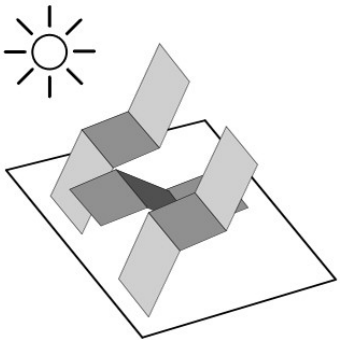
(a) an image



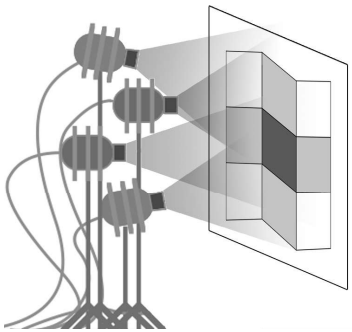
(b) a likely explanation



(c) painter's explanation



(d) sculptor's explanation

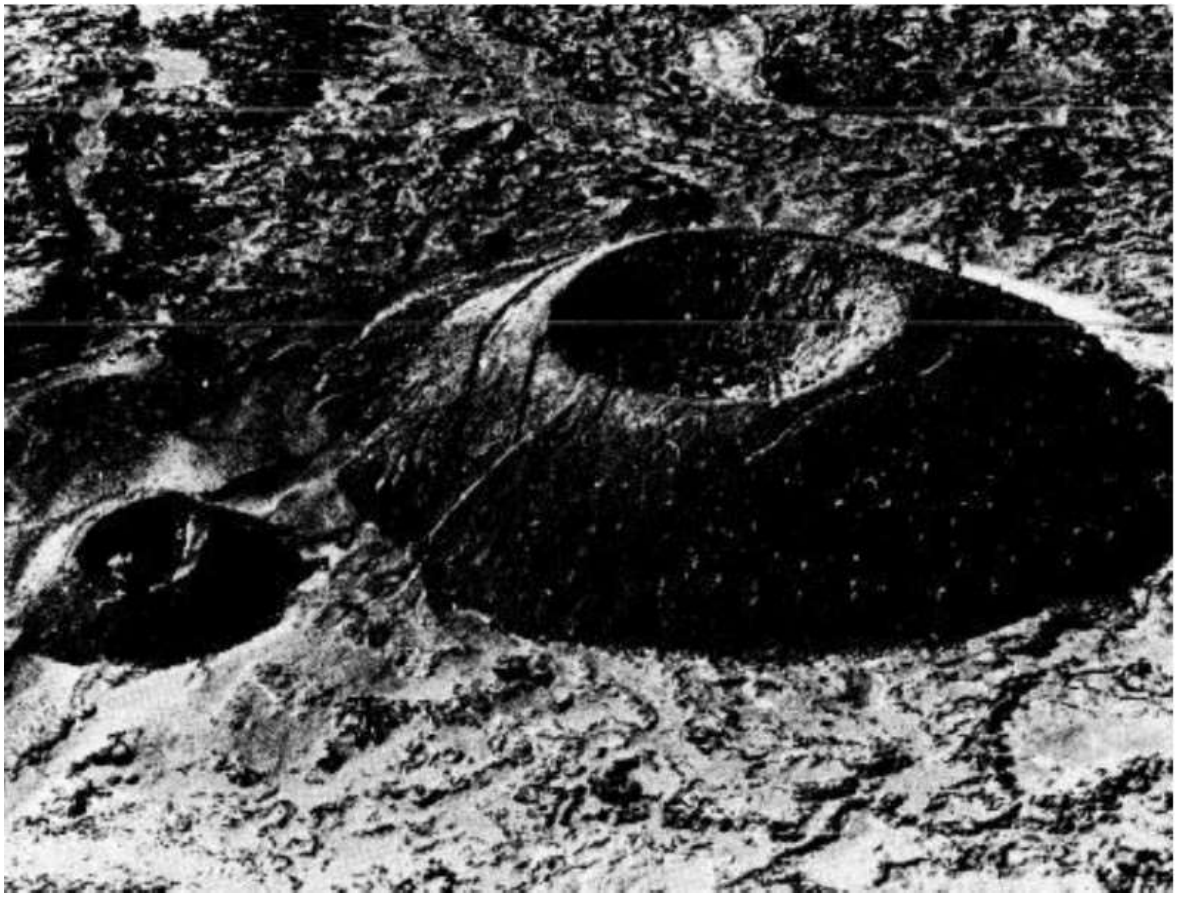


(e) gaffer's explanation

Can we recover **3D from a single image?**

- Inverting the graphics problem is very challenging
- The space of shapes, paint, and light that exactly reproduce an image is vast
- The image in (a) very likely corresponds to (b), but it could be a painting (c), sculpture (d), or an arrangement of lights (e)
- We are looking for a **simple explanation** and must use **prior knowledge**

Human Perception



Human Perception

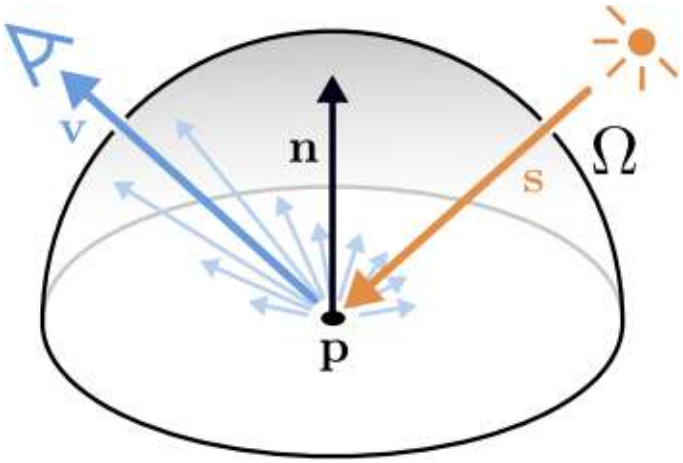
Recap: Rendering Equation

Rendering Equation

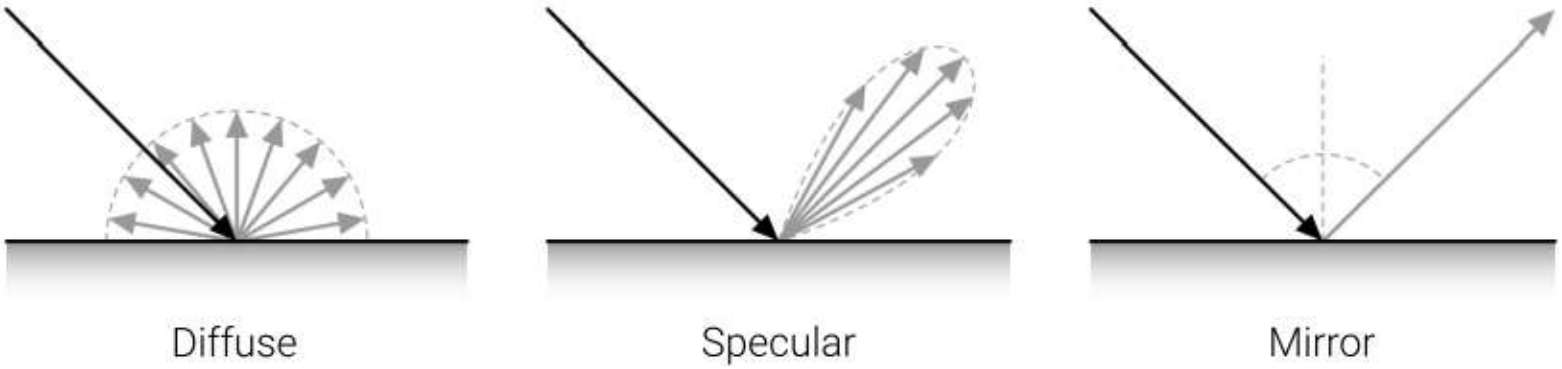
Let $\mathbf{p} \in \mathbb{R}^3$ denote a 3D surface point, $\mathbf{v} \in \mathbb{R}^3$ the viewing direction and $\mathbf{s} \in \mathbb{R}^3$ the incoming light direction. The **rendering equation** describes how much of the light L_{in} with wavelength λ arriving at $\mathbf{p}$ is reflected into the viewing direction $\mathbf{v}$:

$$L_{\text{out}}(\mathbf{p}, \mathbf{v}, \lambda) = L_{\text{emit}}(\mathbf{p}, \mathbf{v}, \lambda) + \int_{\Omega} \text{BRDF}(\mathbf{p}, \mathbf{s}, \mathbf{v}, \lambda) \cdot L_{\text{in}}(\mathbf{p}, \mathbf{s}, \lambda) \cdot (-\mathbf{n}^\top \mathbf{s}) \, d\mathbf{s}$$

- Ω is the unit hemisphere at normal $\mathbf{n}$
- The bidirectional reflectance distribution function $\text{BRDF}(\mathbf{p}, \mathbf{s}, \mathbf{v}, \lambda)$ defines how light is reflected at an opaque surface.
- $L_{\text{emit}} > 0$ only for light emitting surfaces

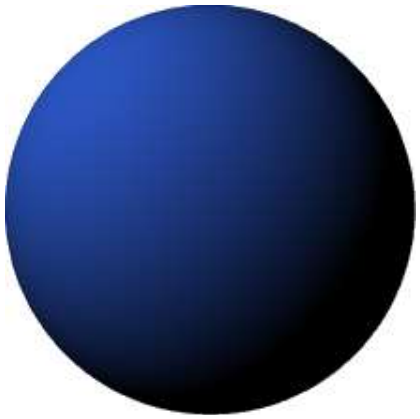


Diffuse and Specular Reflection



- Typical BRDFs have a **diffuse** and a **specular** component
- The diffuse (=constant) component scatters light uniformly in all directions
- This leads to shading, i.e., smooth variation of intensity wrt. surface normal
- The specular component depends strongly on the outgoing light direction

Diffuse and Specular Reflection



Diffuse

Specular

Combined

- Typical BRDFs have a **diffuse** and a **specular** component
- The diffuse (=constant) component scatters light uniformly in all directions
- This leads to shading, i.e., smooth variation of intensity wrt. surface normal
- The specular component depends strongly on the outgoing light direction

Diffuse and Specular Reflection

Clay Pot

Cloud Gate (Kapoor, 2006)

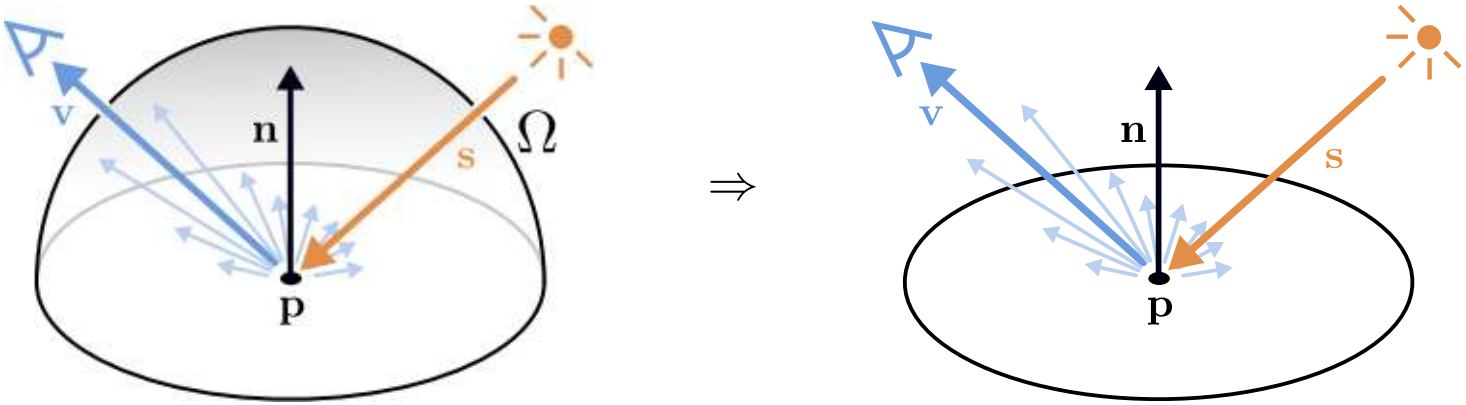
Simplifying the Rendering Equation

Dropping the dependency on λ and $\mathbf{p}$ for notational simplicity, and considering a **single point light source** located in direction $\mathbf{s}$, the rendering equation

$$L_{\text{out}}(\mathbf{p}, \mathbf{v}, \lambda) = L_{\text{emit}}(\mathbf{p}, \mathbf{v}, \lambda) + \int_{\Omega} \text{BRDF}(\mathbf{p}, \mathbf{s}, \mathbf{v}, \lambda) \cdot L_{\text{in}}(\mathbf{p}, \mathbf{s}, \lambda) \cdot (-\mathbf{n}^{\top}\mathbf{s})\, d\mathbf{s}$$

simplifies as follows:

$$L_{\text{out}}(\mathbf{v}) = \text{BRDF}(\mathbf{s}, \mathbf{v}) \cdot L_{\text{in}} \cdot (-\mathbf{n}^{\top}\mathbf{s})$$



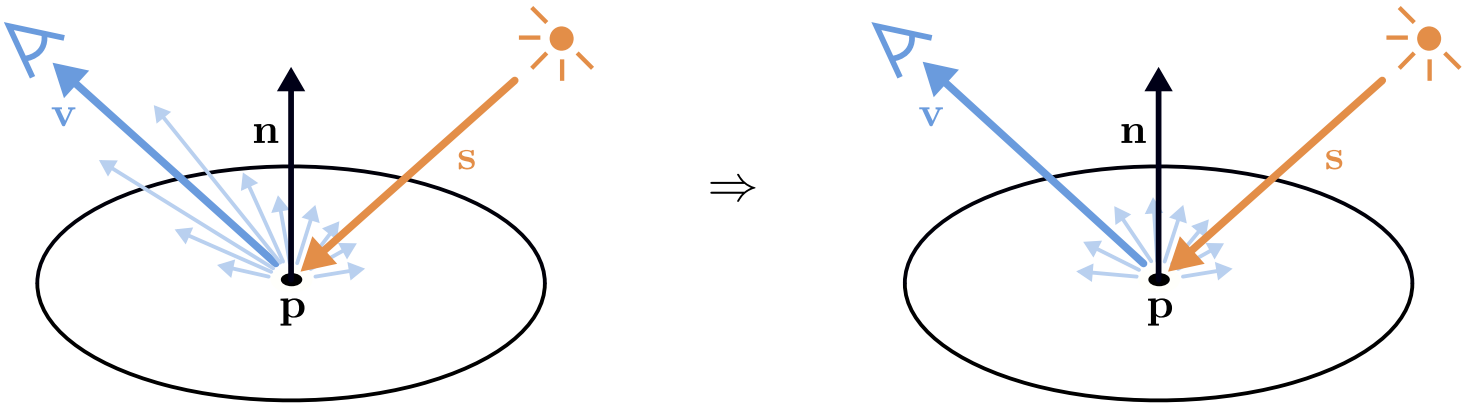
Simplifying the Rendering Equation

Assuming a purely **diffuse material** with albedo (=diffuse reflectance) $\text{BRDF}(\mathbf{s}, \mathbf{v}) = \rho$

$$L_{\text{out}}(\mathbf{v}) = \text{BRDF}(\mathbf{s}, \mathbf{v}) \cdot L_{\text{in}} \cdot (-\mathbf{n}^\top \mathbf{s})$$

further simplifies to the following equation (L_{out} becomes independent of $\mathbf{v}$):

$$L_{\text{out}} = \rho \cdot L_{\text{in}} \cdot (-\mathbf{n}^\top \mathbf{s})$$



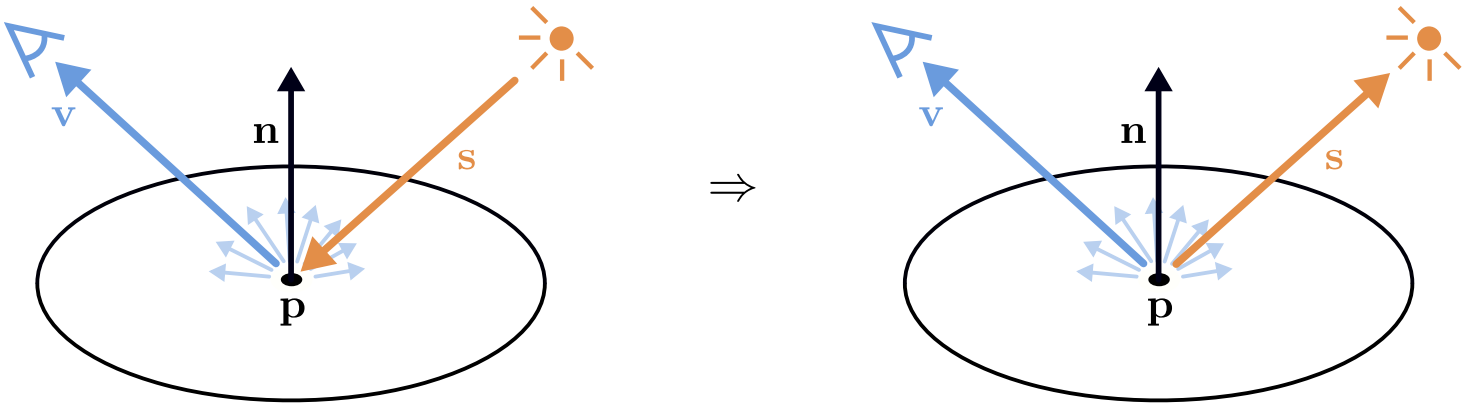
Simplifying the Rendering Equation

For simplicity, we further eliminate the **minus sign** in

$$L_{\text{out}} = \rho \cdot L_{\text{in}} \cdot (-\mathbf{n}^\top \mathbf{s})$$

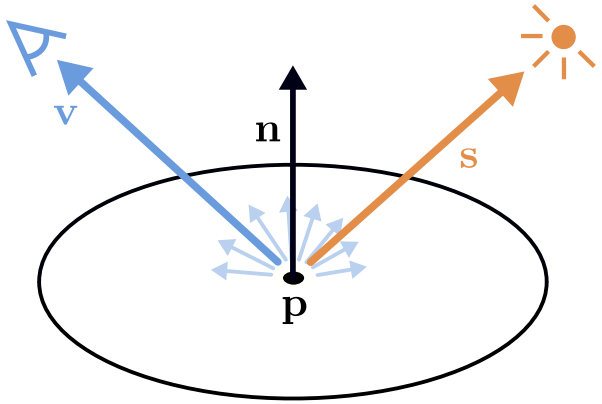
by reversing the orientation (definition) of the light ray $\mathbf{s}$ and obtain:

$$L_{\text{out}} = \rho \cdot L_{\text{in}} \cdot \mathbf{n}^\top \mathbf{s}$$



Simplifying the Rendering Equation

$$L_{\text{out}} = \rho \cdot L_{\text{in}} \cdot \mathbf{n}^\top \mathbf{s} = R(\mathbf{n})$$



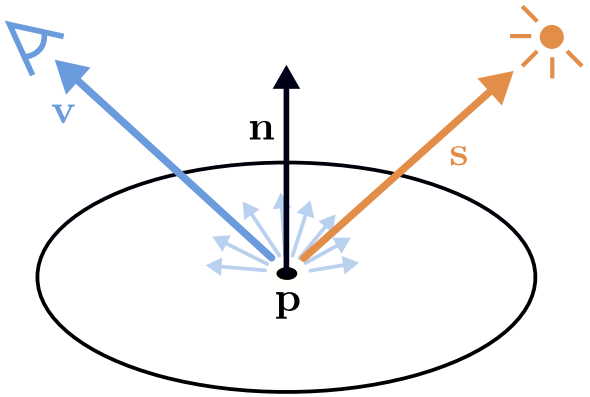
- For a fixed material and light source, the reflected light L_{out} is a function of $\mathbf{n}$
- This function $R(\mathbf{n})$ is called **reflectance map** (we will see examples)
- If we would know $\mathbf{n}$ at each surface point, we can integrate the geometry
- Can we determine $\mathbf{n}$ from the observation L_{out} for every pixel in an image?
- This problem is called **Shape-from-Shading** (Berthold Horn, 1970)

Shape-from-Shading

Shape-from-Shading (SfS)

Shape-from-Shading makes the following assumptions:

- **Diffuse material with spatially constant albedo ρ**
  - Reduces number of material parameters to 1
- **Known point light source at infinity**
  - Keeps light direction $\mathbf{s}$ constant across all pixels
  - Makes $\mathbf{s}$ independent of geometry/depth
- **Known camera at infinity (orthography)**
  - Keeps view direction $\mathbf{v}$ constant across all pixels
  - Makes $\mathbf{v}$ independent of geometry/depth



Horn: Obtaining shape from shading information. Psychology of Computer Vision, 1975.

Gradient Space Representation

How should we **parameterize** the normal $\mathbf{n}$?

- While $\mathbf{n} \in \mathbb{R}^3$, it has only 2 degrees of freedom
- Parameterize $\mathbf{n}$ by **neg. gradients of depth map:**

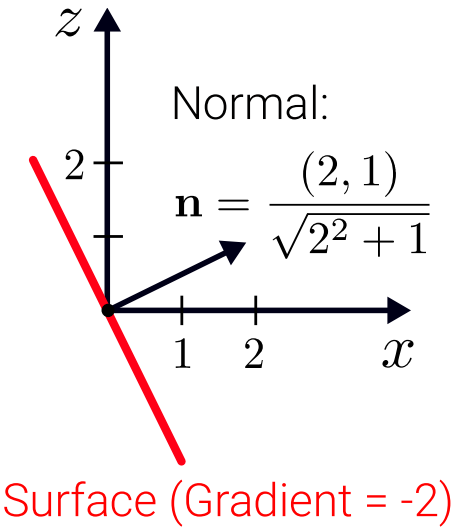
$$(p, q) = \left(-\frac{\partial z}{\partial x}, -\frac{\partial z}{\partial y} \right)$$

- The surface normal $\mathbf{n}$ at pixel (x, y) is given by:

$$\mathbf{n} = \frac{(p, q, 1)^\top}{\sqrt{p^2 + q^2 + 1}}$$

- Assuming $\rho \cdot \mathrm{L}_{in} = 1$, the **reflectance** becomes:

$$R(\mathbf{n}) = \mathbf{n}^\top \mathbf{s} = \frac{p\, s_x + q\, s_y + s_z}{\sqrt{p^2 + q^2 + 1}} = R(p, q)$$

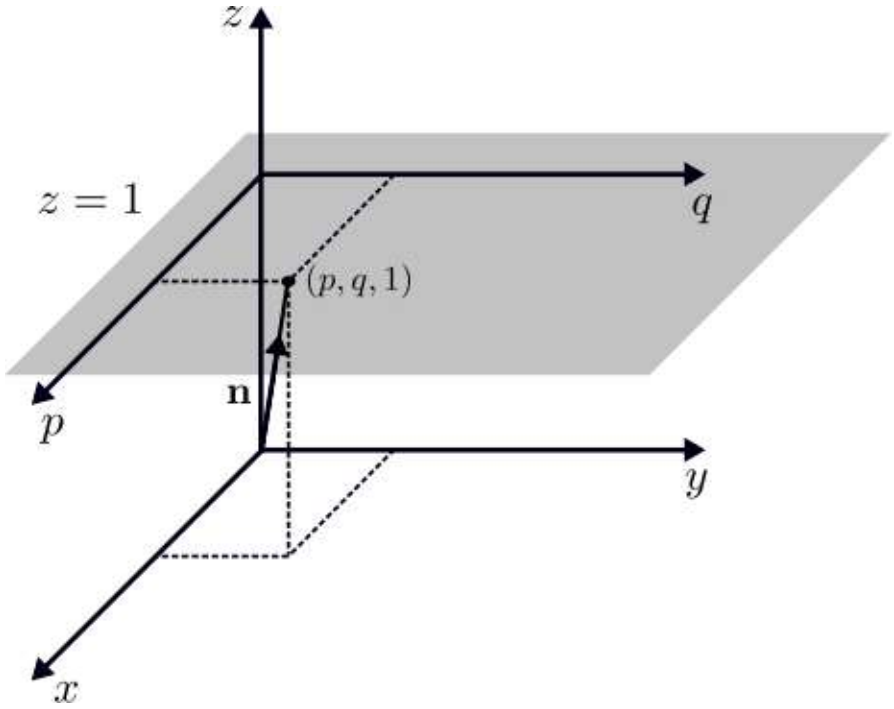


Ikeuchi and Horn: Numerical Shape from Shading and Occluding Boundaries. AI, 1981

Gradient Space Representation

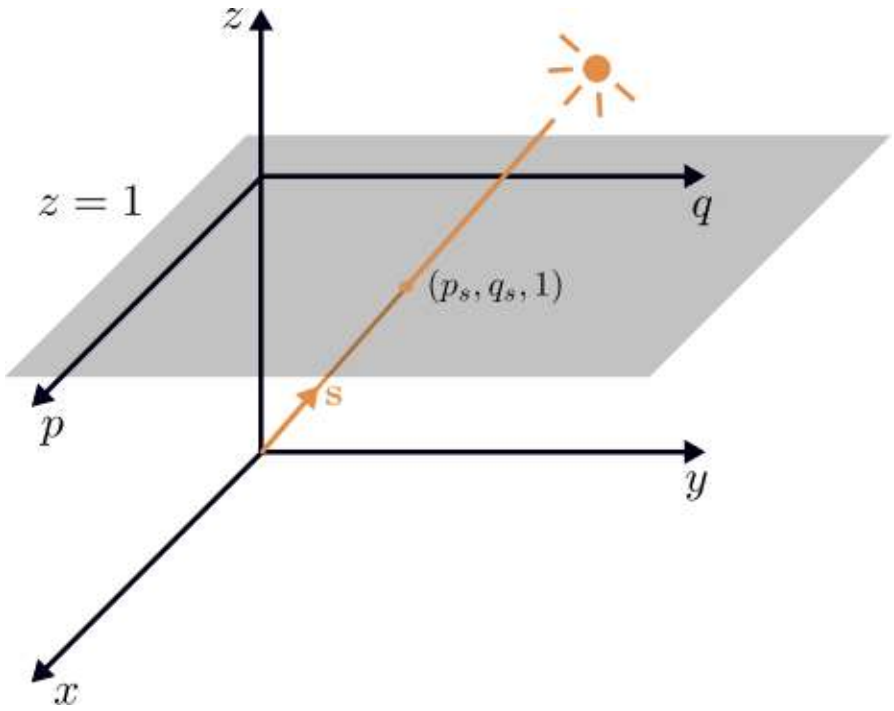
- Visualization of the **gradient space** (gray)
- Normal $\mathbf{n}$ intersects $z = 1$ plane at $(p, q, 1)^\top$
- Unit normal given by:

$$\mathbf{n} = \frac{(p, q, 1)^\top}{\sqrt{p^2 + q^2 + 1}}$$



Gradient Space Representation

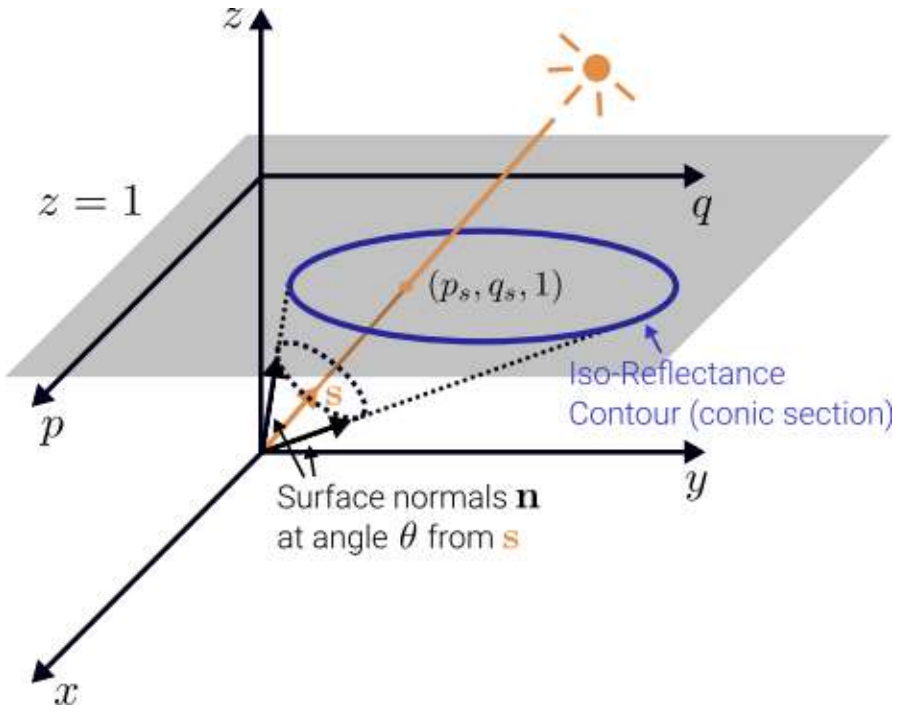
- The **gradient space** can also be used to represent the **light direction** $\mathbf{s}$
- Light ray $\mathbf{s}$ intersects $z = 1$ plane at $(p_s, q_s, 1)^\top$
- For a given light ray $\mathbf{s}$ and observed reflectance R what is the normal $\mathbf{n}$?



Ikeuchi and Horn: Numerical Shape from Shading and Occluding Boundaries. AI, 1981

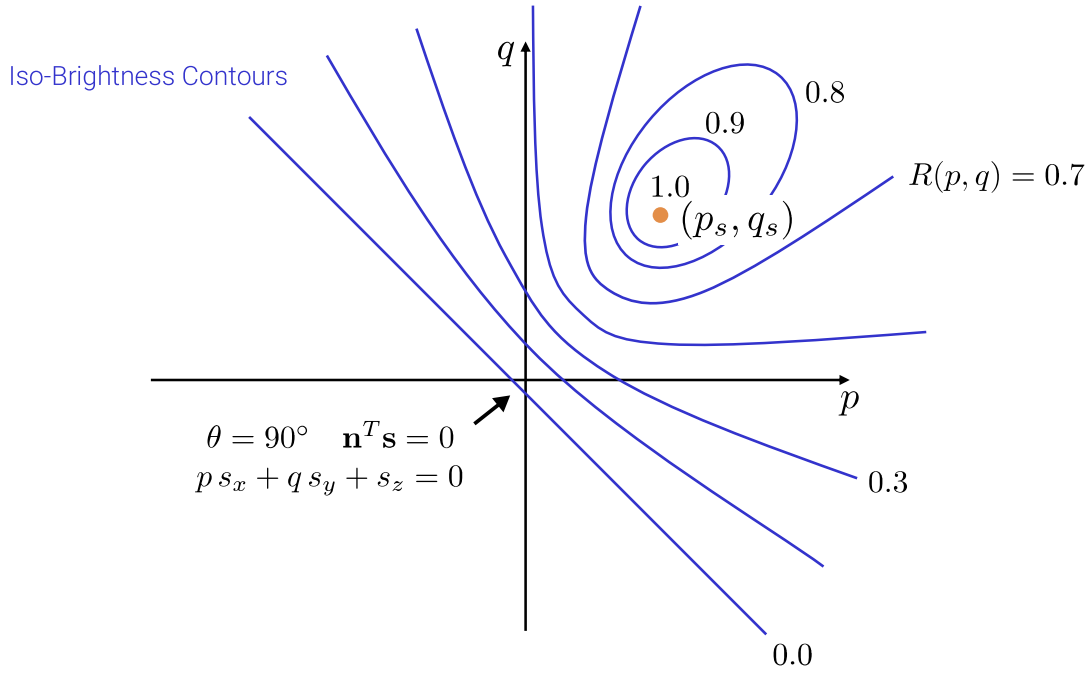
Gradient Space Representation

- For a given light ray $\mathbf{s}$ and observed reflectance R what is the normal $\mathbf{n}$?
- We have $R = \mathbf{n}^\top \mathbf{s} = \cos(\theta)$ with angle θ between $\mathbf{n}$ and $\mathbf{s}$
- The set of solutions $\{\mathbf{n}\}$ is located on a **circle** around $\mathbf{s}$
- Projecting this set to the $z = 1$ plane yields a **conic section**



Ikeuchi and Horn: Numerical Shape from Shading and Occluding Boundaries. AI, 1981

Reflectance Map



- Except for $R(p,q) = 1.0$, the normals consistent with R are **not unique**
- Inferring unique normals from a single image thus requires **further constraints**
- **Solutions:** Regularization (SfS), more observations (photometric stereo)

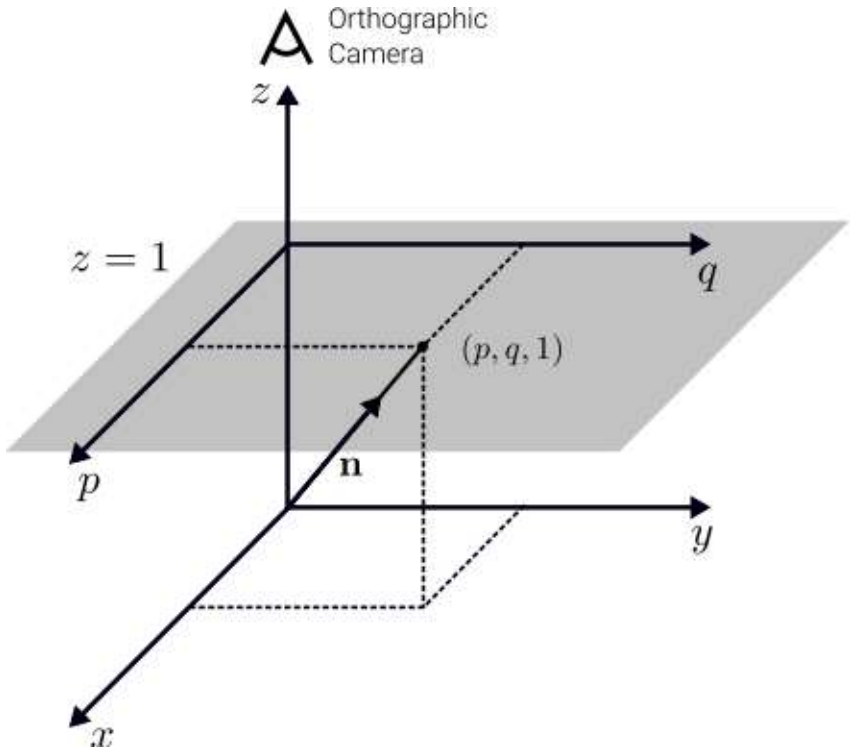
Ikeuchi and Horn: Numerical Shape from Shading and Occluding Boundaries. AI, 1981

Reflectance Map

Under the assumptions of a known distant point light and observer, the variation in reflectance is a function of the local surface orientation:

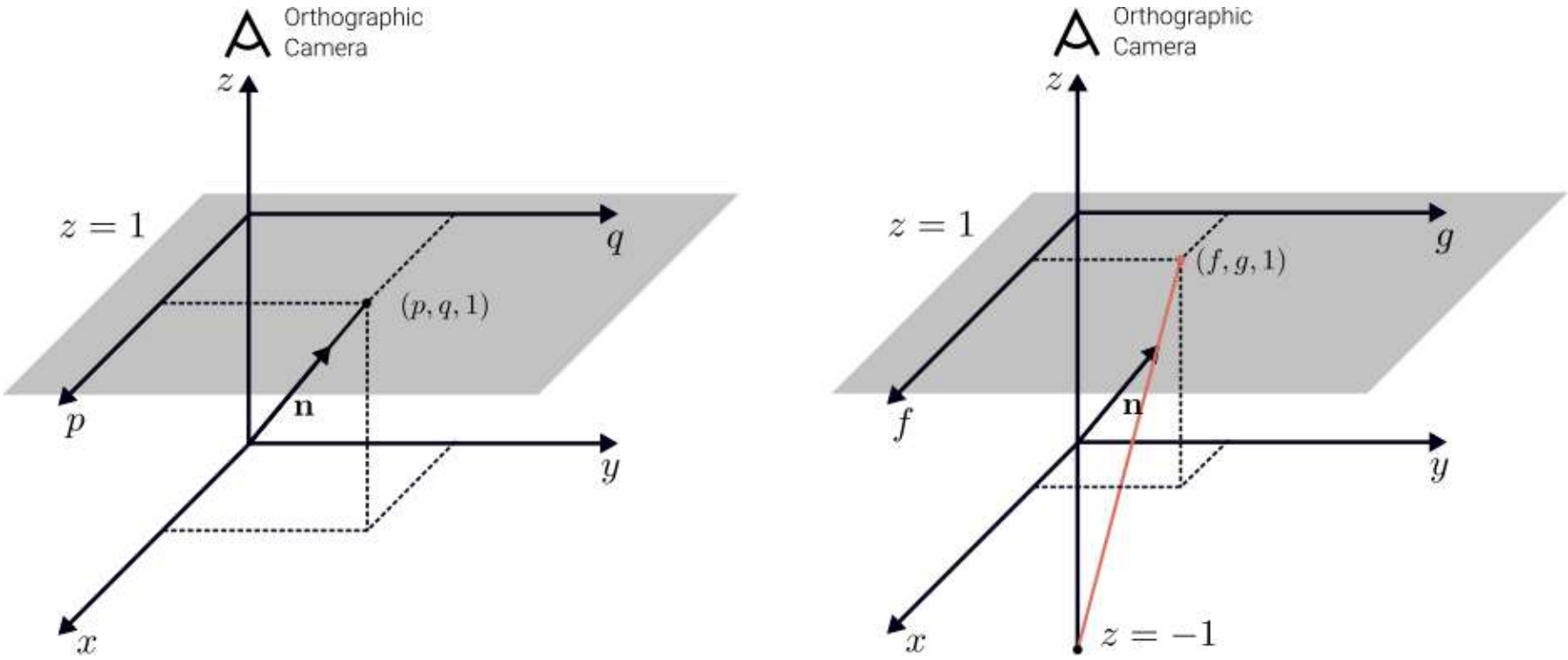
$$R(\mathbf{n}) = \mathbf{n}^\top \mathbf{s} = \frac{p\, s_x + q\, s_y + s_z}{\sqrt{p^2 + q^2 + 1}} = R(p, q)$$

- $(p, q) = \left(-\frac{\partial z}{\partial x}, -\frac{\partial z}{\partial y}\right)$ are the negative **gradients** of the depth map
- $R(p, q)$ is called **reflectance map**
- What happens when $\mathbf{n}$ becomes orthogonal to the viewing direction?



Ikeuchi and Horn: Numerical Shape from Shading and Occluding Boundaries. AI, 1981

Stereographic Mapping



- Solution: **Change of representation** that bounds surface gradients (norm < 2)

Ikeuchi and Horn: Numerical Shape from Shading and Occluding Boundaries. AI, 1981

Stereographic Mapping

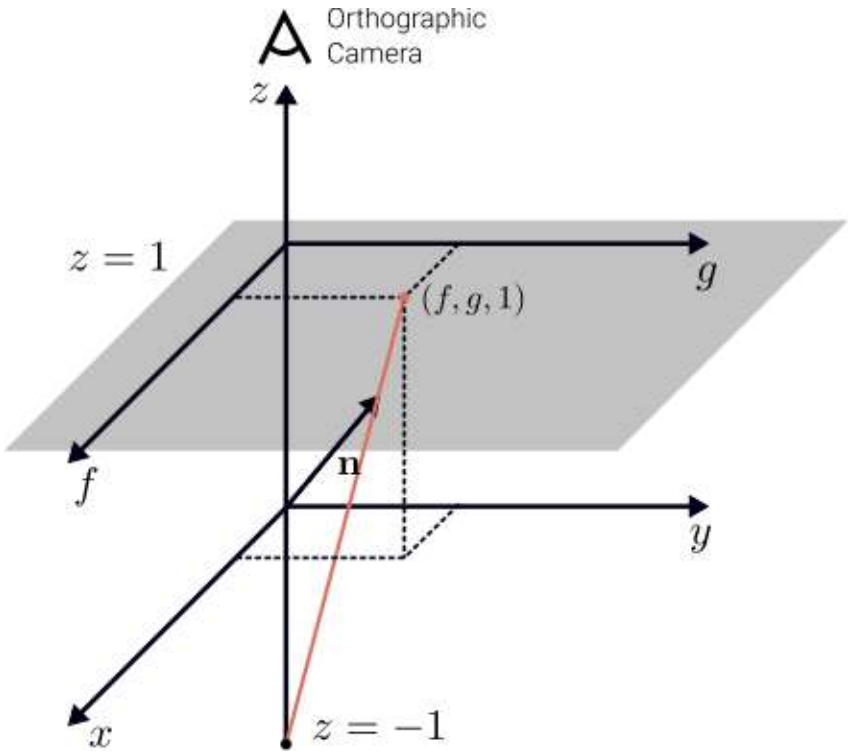
The following relationship holds:

$$f = \frac{2p}{1 + \sqrt{p^2 + q^2 + 1}}$$

$$g = \frac{2q}{1 + \sqrt{p^2 + q^2 + 1}}$$

Now R depends on f and g:

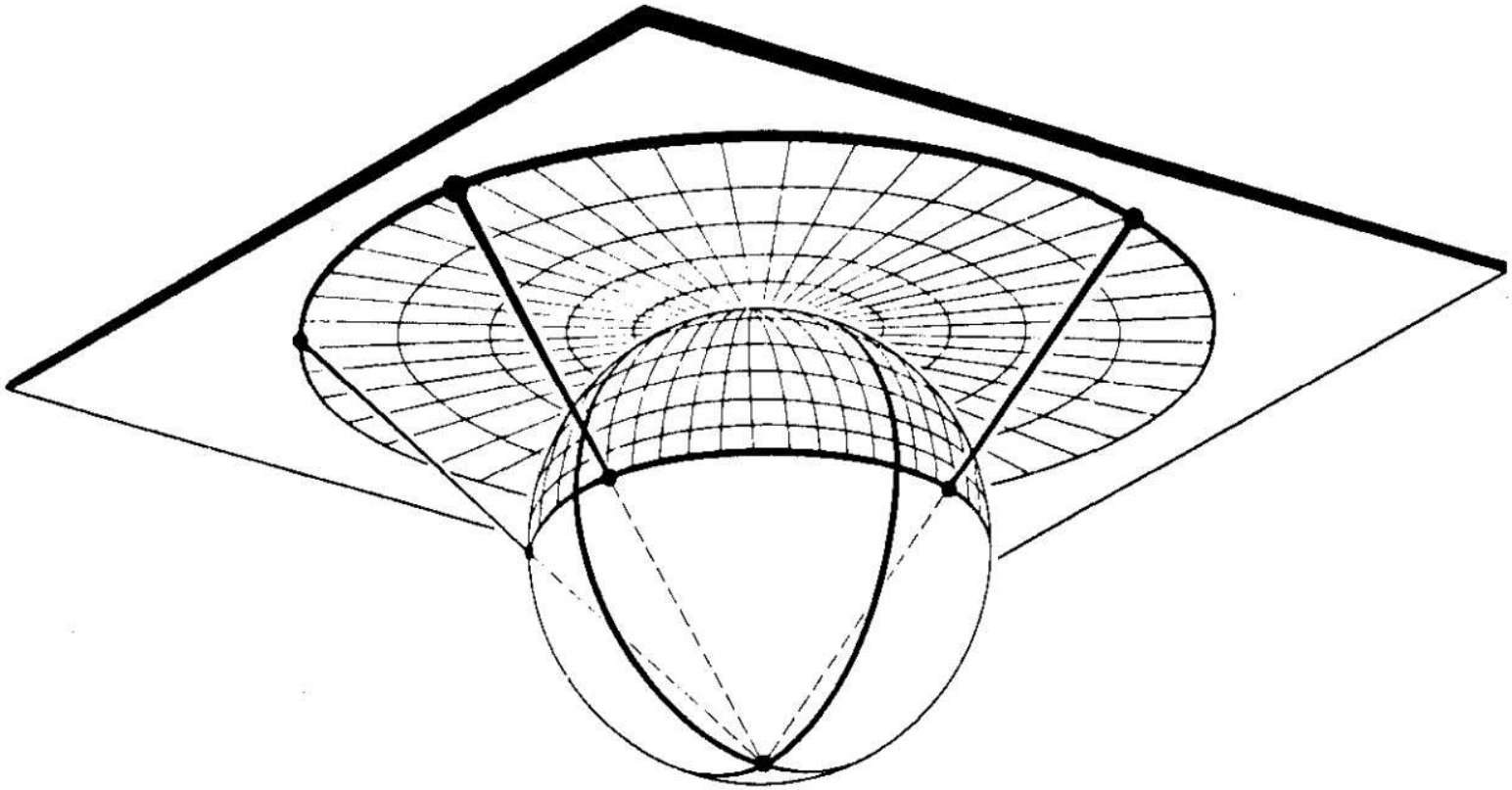
$$R(f(x, y), g(x, y))$$



- Solution: **Change of representation** that bounds surface gradients (norm < 2)

Ikeuchi and Horn: Numerical Shape from Shading and Occluding Boundaries. AI, 1981

Stereographic Mapping



Ikeuchi and Horn: Numerical Shape from Shading and Occluding Boundaries. AI, 1981

Shape-from-Shading Formulation

Assumption: image irradiance (=intensity) should equal the reflectance map:

$$I(x, y) = R(f(x, y), g(x, y))$$

SfS thus minimizes:

$$E_{image}(f, g) = \int \int (I(x, y) - R(f, g))^2 \, dx \, dy$$

Goal: Penalize errors between image irradiance and reflectance map

However, as we have seen, this problem is ill-posed (#unknowns $>$ #observations)

Numerical Shape-from-Shading

To constrain this ill-posed problem, SfS exploits two additional constraints:

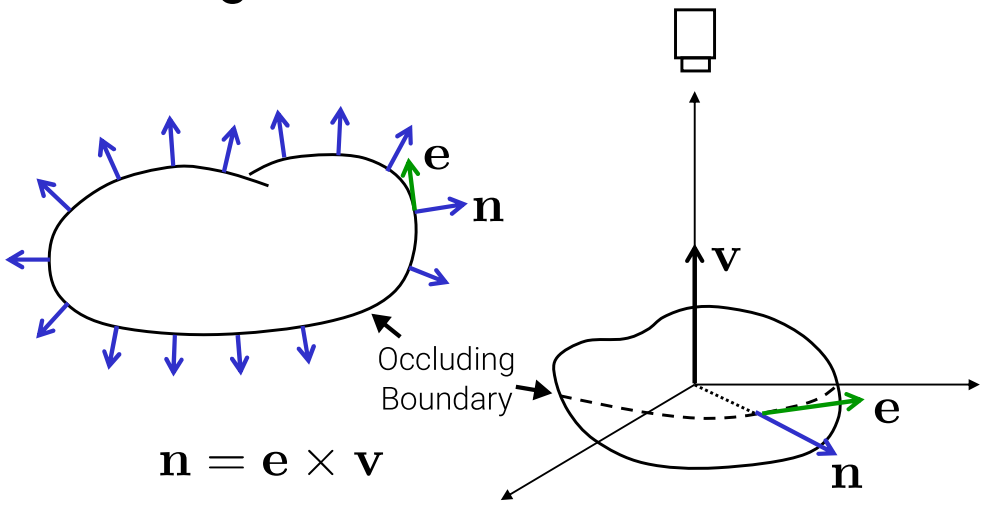
Smoothness:

$$E_{smooth}(f,g) = \int\int f_x^2 + f_y^2 + g_x^2 + g_y^2 \, dx \, dy$$

with gradients $f_x = \frac{\partial f}{\partial x}$, $f_y = \frac{\partial f}{\partial y}$, ..

- Goal: Penalize rapid changes in surface gradients f and g

Occluding Boundaries:



- Goal: Constrain normals at occluding boundaries where they are known

Ikeuchi and Horn: Numerical Shape from Shading and Occluding Boundaries. AI, 1981

Numerical Shape-from-Shading

In summary, we optimize the **variational energy**

$$E(f, g) = E_{image}(f, g) + \lambda E_{smooth}(f, g)$$

wrt. the gradient fields $f(x, y)$ and $g(x, y)$ subject to occluding boundary constraints.

Remarks:

- As images are spatially discrete, this problem is transferred to a spatially discrete problem similar to the Horn-Schunck optical flow algorithm discussed in lecture 6
- Iterative optimization required as R depends non-linearly on f and g
- We can use the known normals to fix f and g on the occluding boundary
- Initialize other variables to 0, see also: `https://fpcv.cs.columbia.edu/`

Surface Integration

Surface Integration

Given the **surface gradients**

$$(p, q) = \left(-\frac{\partial z}{\partial x}, -\frac{\partial z}{\partial y} \right)$$

how can we **recover the 3D surface** / depth map?

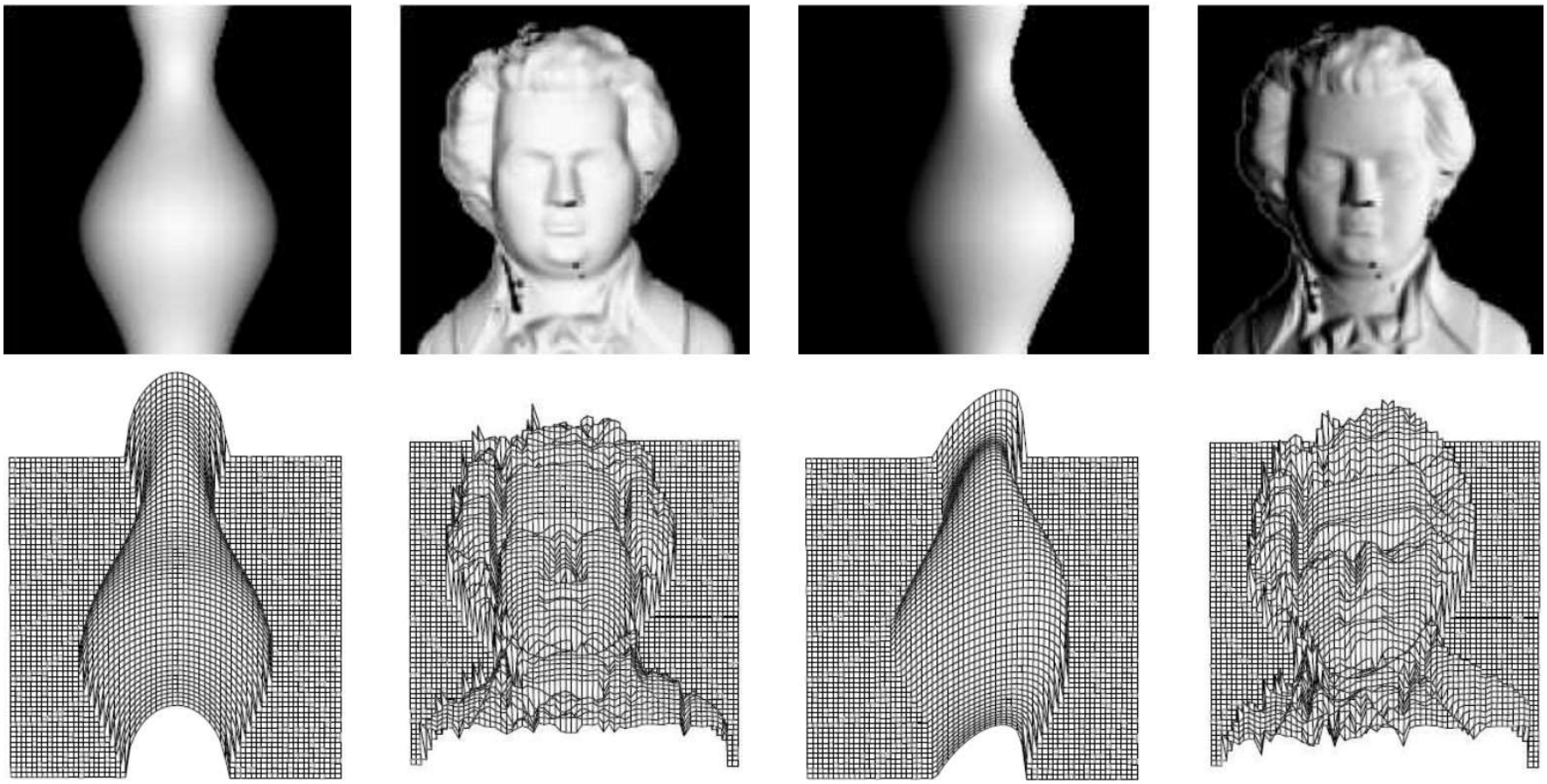
Assuming a smooth surface, we can solve the following **variational problem**

$$E(z) = \int \int \left(\frac{\partial z}{\partial x} + p \right)^2 + \left(\frac{\partial z}{\partial y} + q \right)^2 dx\, dy$$

efficiently using the discrete **Fast Fourier Transform** (Frankot and Chellappa, 1988).

Frankot and Chellappa: A Method for enforcing integrability in shape from shading algorithms. TPAMI, 1988.

Results

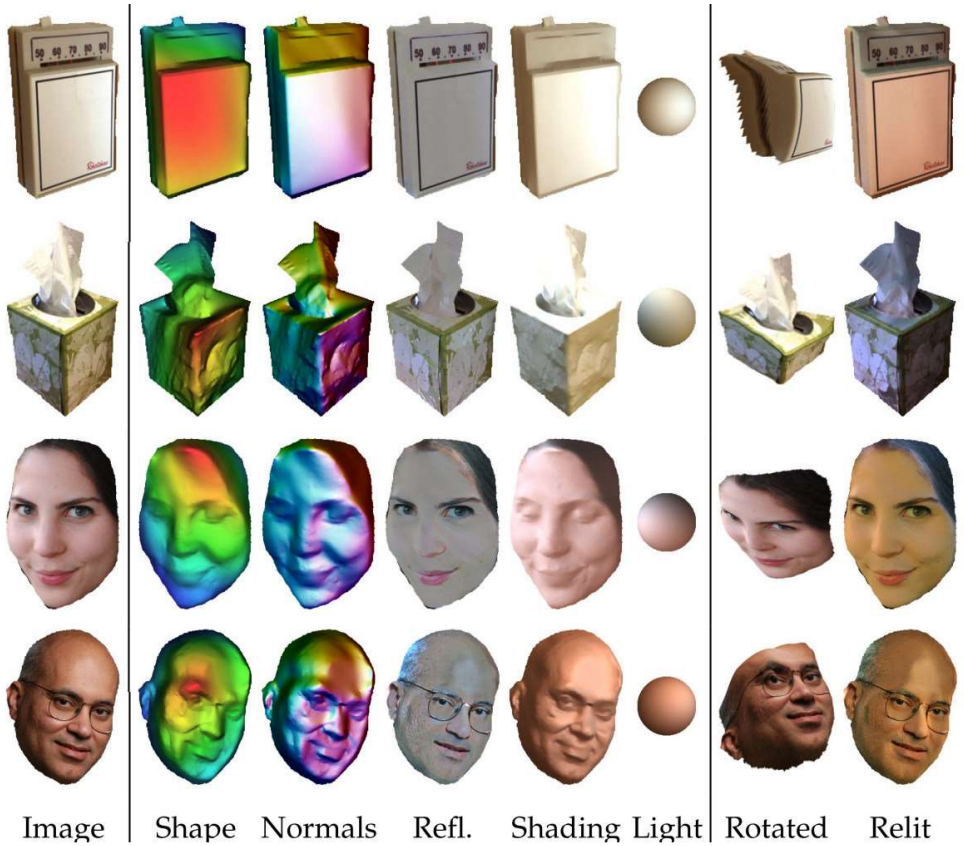


Frankot and Chellappa: A Method for enforcing integrability in shape from shading algorithms. TPAMI, 1988.

SIRFS: Shape, Illumination, and Reflectance from Shading

SIRFS:

- Modern version of SfS
- Input: Single masked RGB image
- Estimates depth, normals, reflectance, shading, lighting
- Doesn't assume known point light
- Doesn't assume constant color
- Much harder problem than SfS

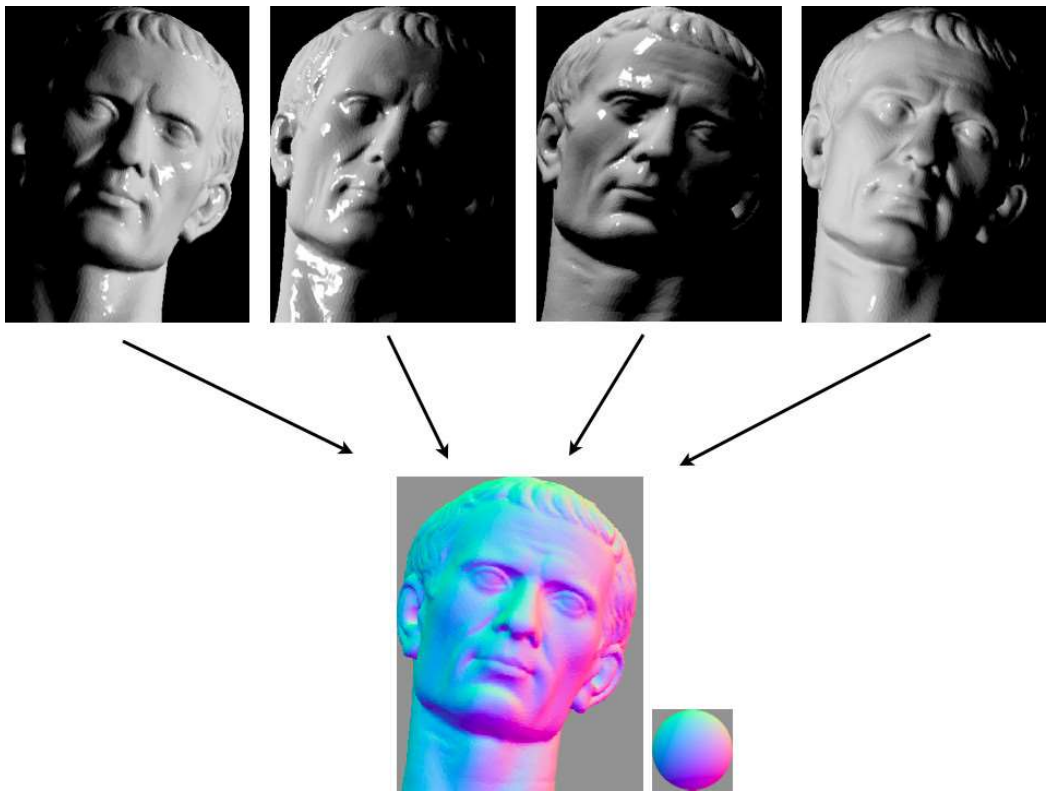


Barron and Malik: Shape, Illumination, and Reflectance from Shading. TPAMI, 2015.

8.2
Photometric Stereo

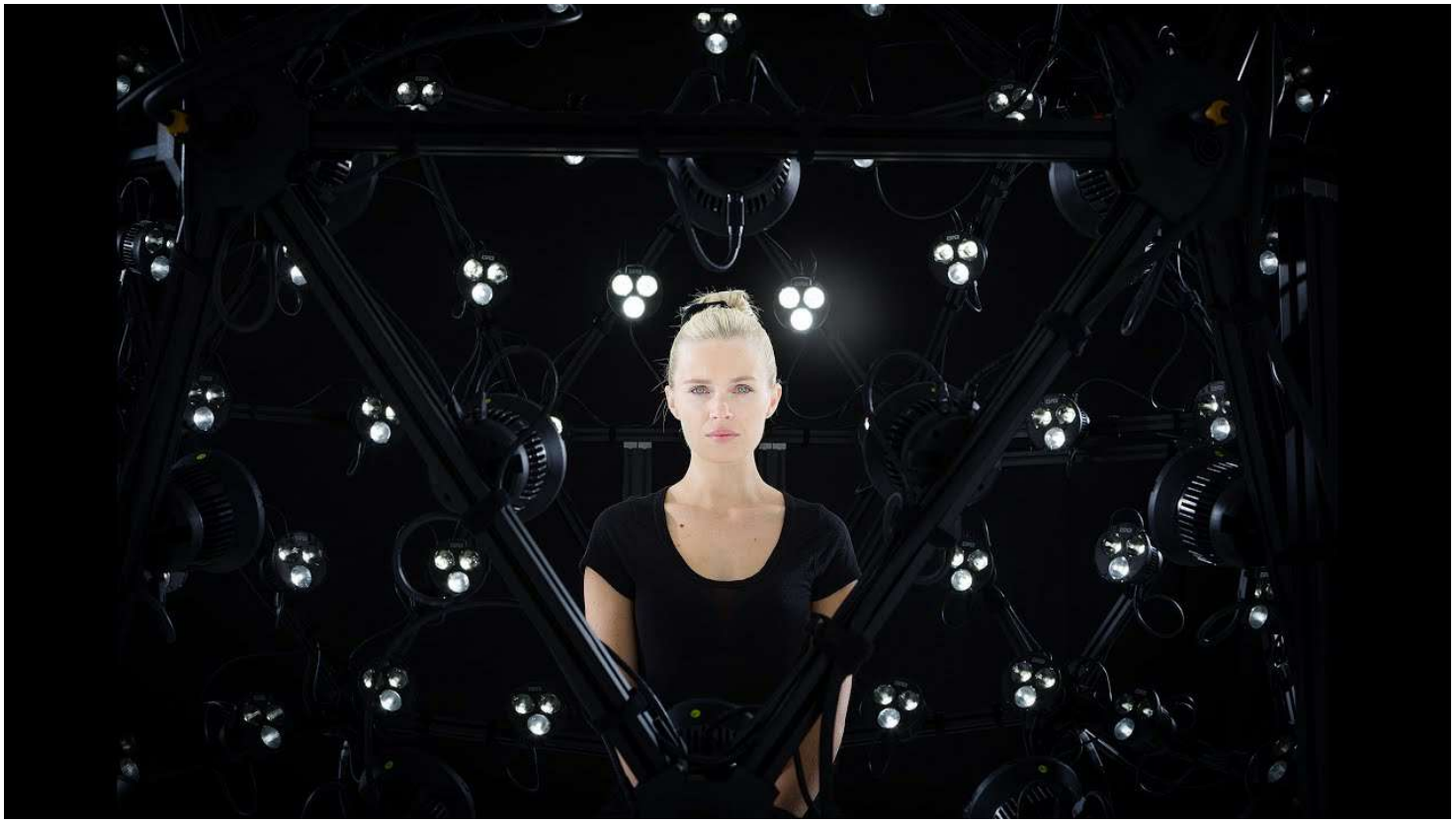
Photometric Stereo

- Instead of smoothness constraints, add **more observations per pixel**
- Take **K images** of the object from **same viewpoint** (e.g., with a tripod) but with different (known) point light source each
- Per-pixel estimation of **normal** and **albedo** or **material**
- Also assumes far camera/light



Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

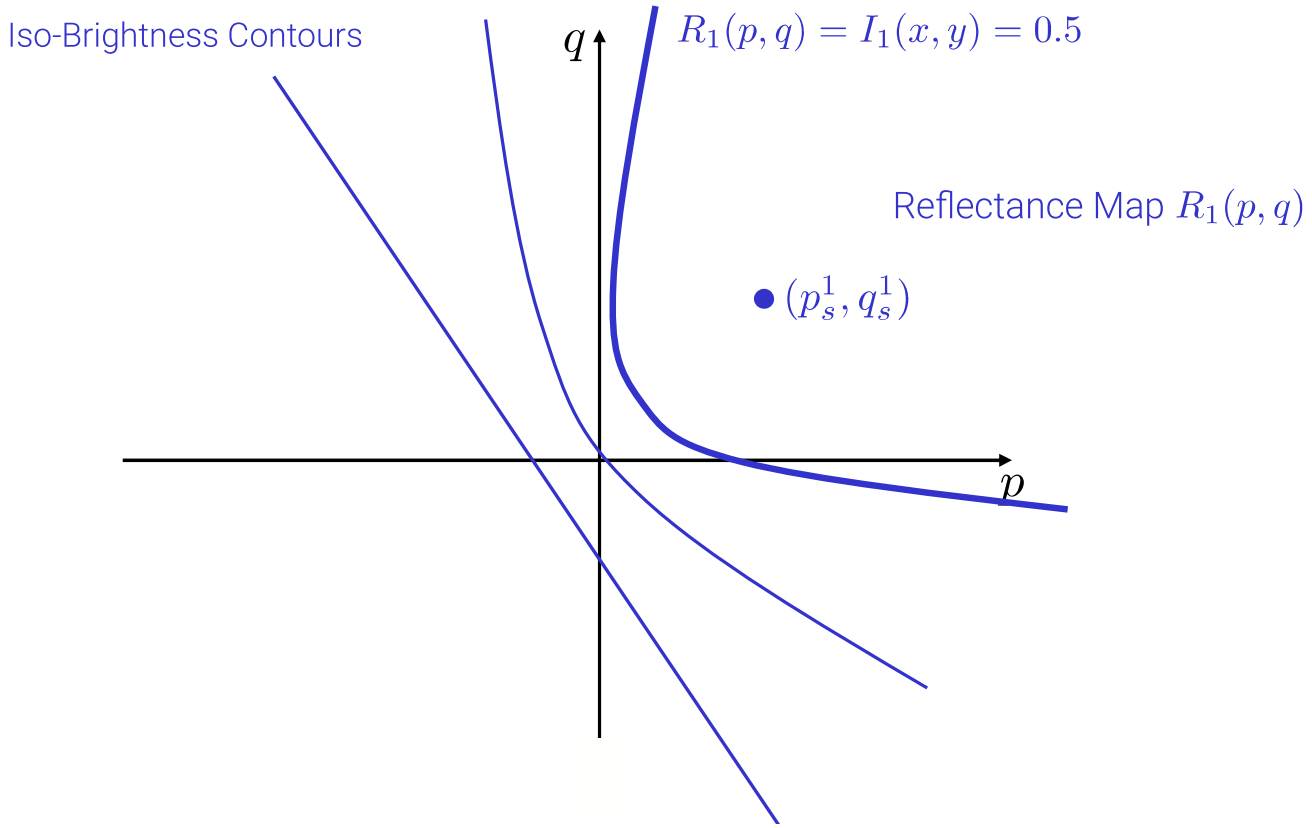
Light Stage



https://www.esperhq.com/product/lightcage-scanning-rig/

Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

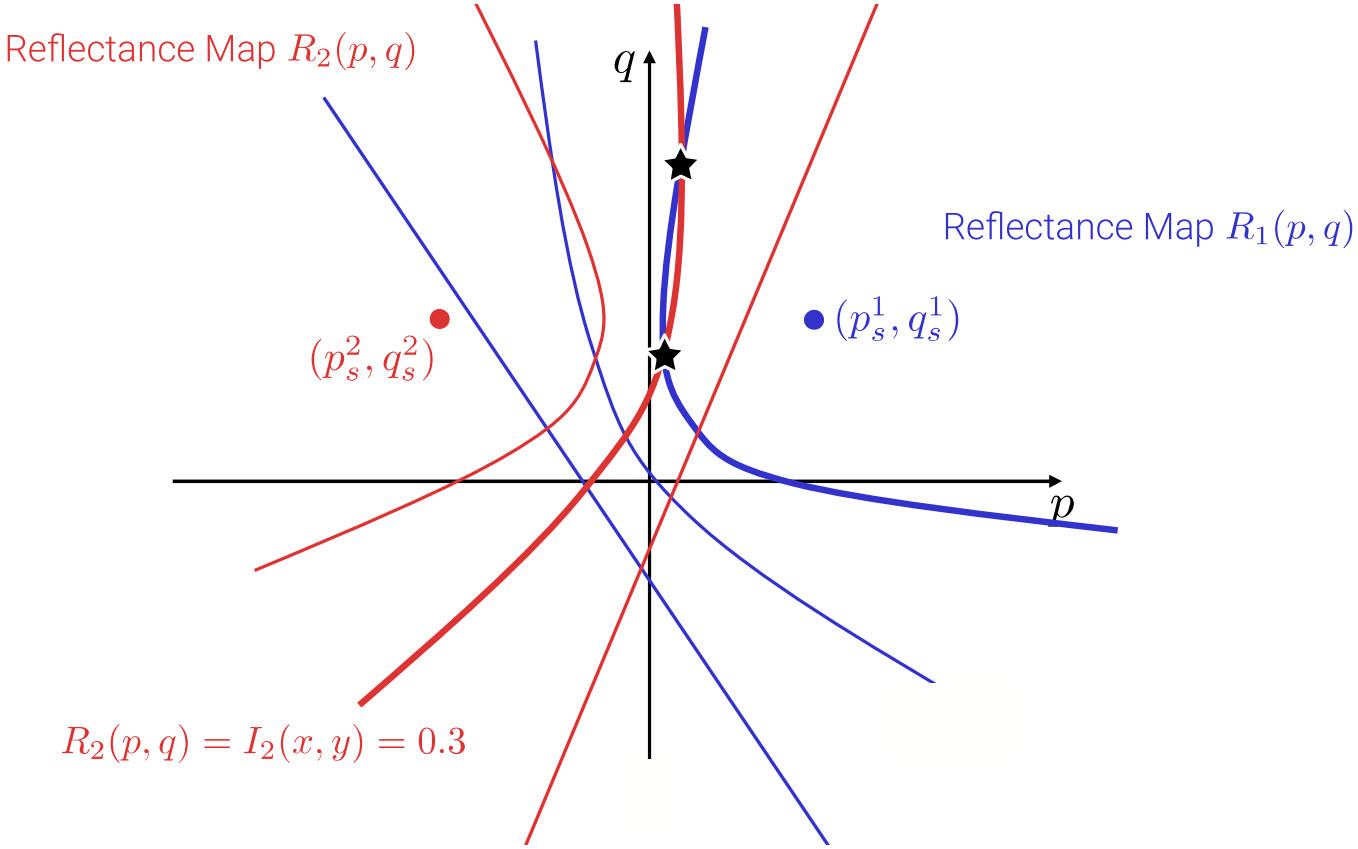
Reflectance Maps



- Capture K images, construct reflectance maps/curves per pixel, find intersection

Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

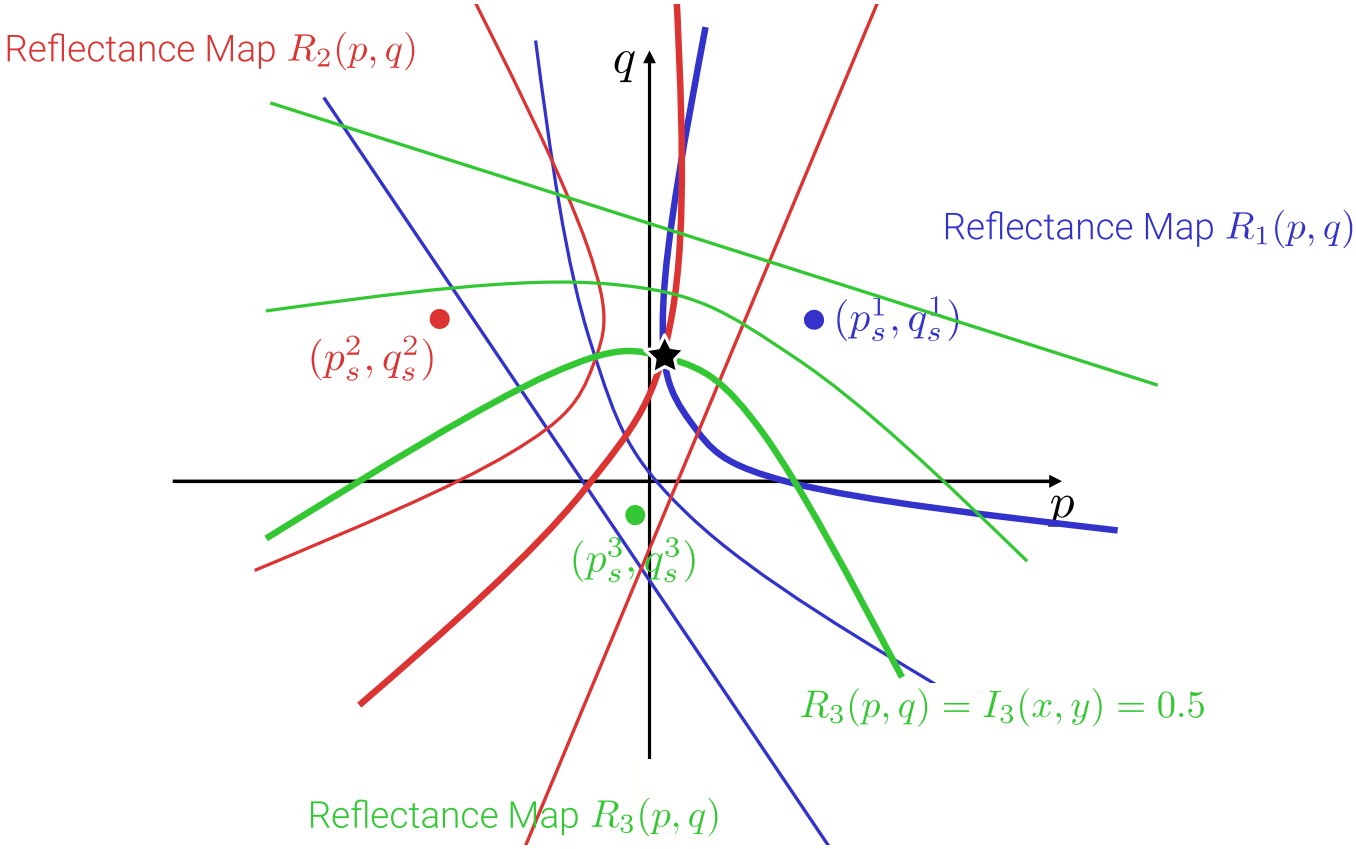
Reflectance Maps



- Capture K images, construct reflectance maps/curves per pixel, find intersection

Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

Reflectance Maps



- Capture K images, construct reflectance maps/curves per pixel, find intersection

Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

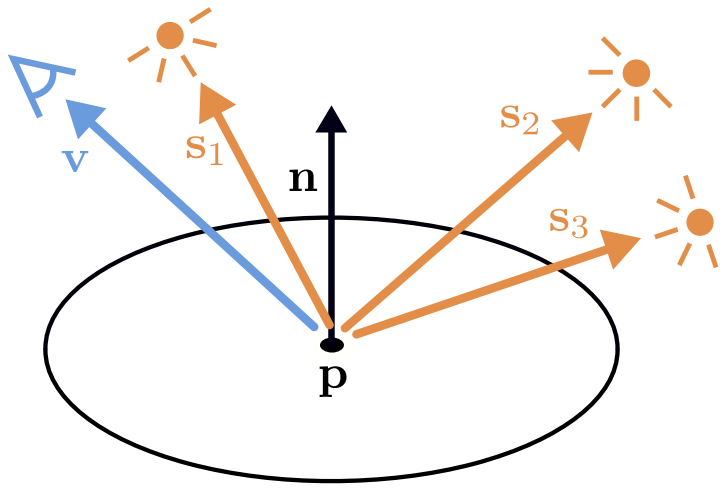
Photometric Stereo Formulation

Assuming **Lambertian reflectance** and $L_{\text{in}} = 1$, the image intensity / reflected light is:

$$I = L_{out} = \rho \cdot \mathbf{n}^\top \mathbf{s} = \rho \cdot \mathbf{s}^\top \mathbf{n}$$

Given **3 observations** (same $\mathbf{v}$, different $\mathbf{s}$),
we can express this in matrix form as follows:

$$\underbrace{\begin{pmatrix} I_1 \\ I_2 \\ I_3 \end{pmatrix}}_{\mathbf{I}} = \underbrace{\begin{pmatrix} \mathbf{s}_1^\top \\ \mathbf{s}_2^\top \\ \mathbf{s}_3^\top \end{pmatrix}}_{\mathbf{S}} \underbrace{\rho\,\mathbf{n}}_{\tilde{\mathbf{n}}}$$



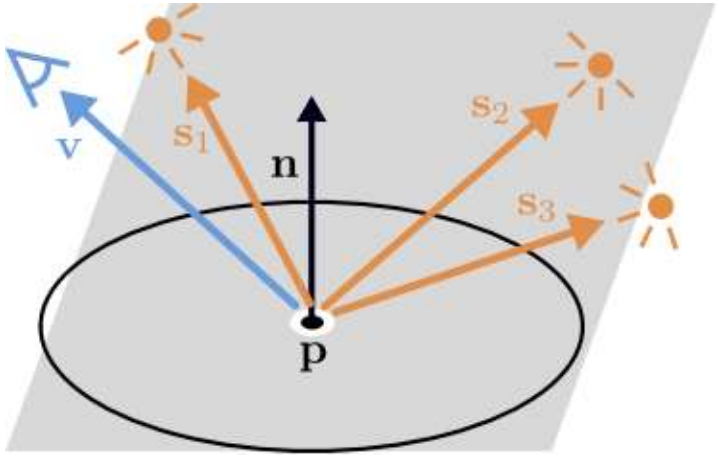
Solution given by $\tilde{\mathbf{n}} = \mathbf{S}^{-1}\mathbf{I}$, $\rho = \|\tilde{\mathbf{n}}\|_2$ and $\mathbf{n} = \tilde{\mathbf{n}}/\rho$.

Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

Photometric Stereo

When does Photometric Stereo **not work**?

$$\underbrace{\begin{pmatrix} I_1 \\ I_2 \\ I_3 \end{pmatrix}}_{\mathbf{I}} = \underbrace{\begin{pmatrix} \mathbf{s}_1^\top \\ \mathbf{s}_2^\top \\ \mathbf{s}_3^\top \end{pmatrix}}_{\mathbf{S}} \underbrace{\rho\,\mathbf{n}}_{\tilde{\mathbf{n}}}$$



If $\mathbf{s}_3 = \alpha\mathbf{s}_1 + \beta\mathbf{s}_2$, i.e., if all three light sources $\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3$ and the origin $\mathbf{p}$ lie on a 3D plane, the linear system becomes rank-deficient and thus there exists no unique solution $\tilde{\mathbf{n}} = \mathbf{S}^{-1}\mathbf{I}$.

Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

Photometric Stereo

Better results can be obtained by using **more images** (by averaging noise):

$$\underbrace{\begin{pmatrix} I_1 \\ \vdots \\ I_K \end{pmatrix}}_{\mathbf{I}} = \underbrace{\begin{pmatrix} \mathbf{s}_1^\top \\ \vdots \\ \mathbf{s}_K^\top \end{pmatrix}}_{\mathbf{S}} \underbrace{\rho\,\mathbf{n}}_{\tilde{\mathbf{n}}}$$

The **least squares solution** is given by (albedo ρ and normal $\mathbf{n}$ as before):

$$\mathbf{S}^\top \mathbf{I} = \mathbf{S}^\top \mathbf{S} \tilde{\mathbf{n}}$$

$$\tilde{\mathbf{n}} = \underbrace{(\mathbf{S}^\top \mathbf{S})^{-1} \mathbf{S}^\top}_{\text{Pseudoinverse}} \mathbf{I}$$

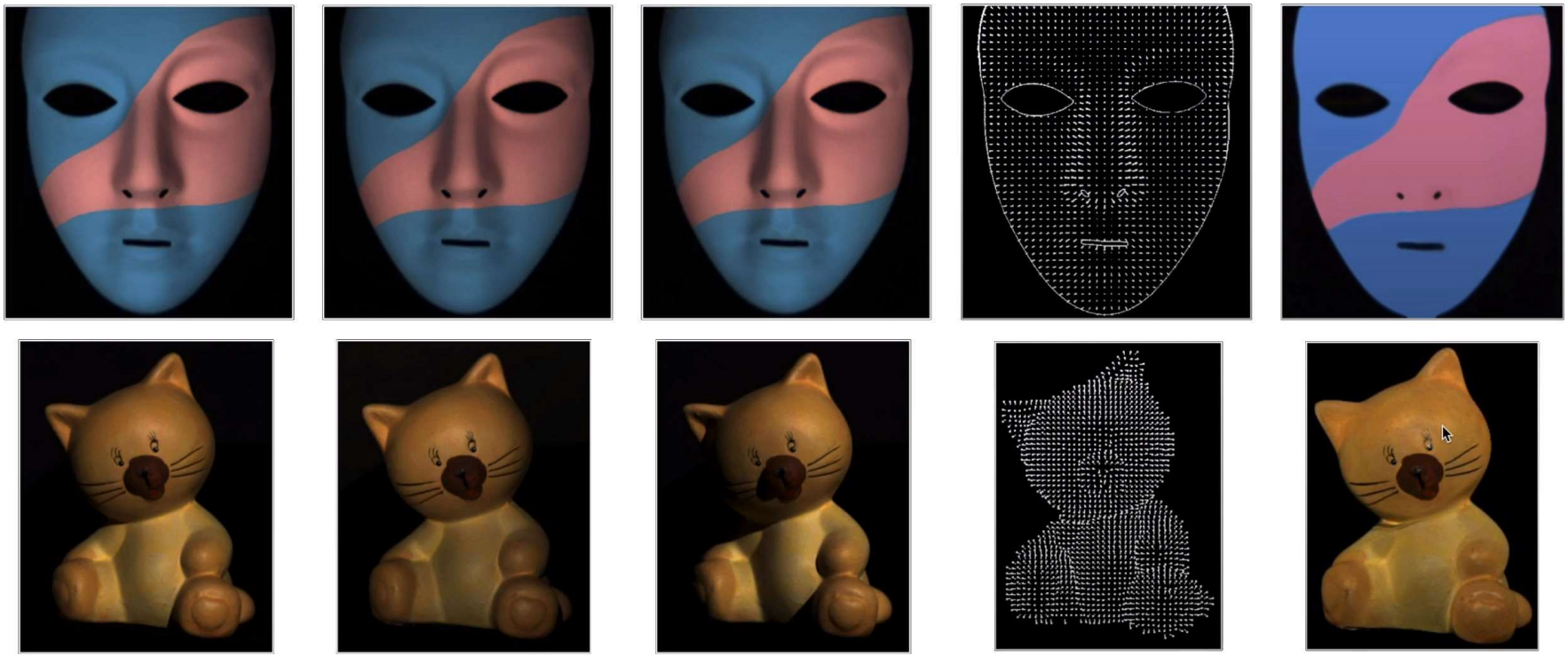
Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

Photometric Stereo Algorithm

1. Compute **surface normals and albedo** values (per pixel)
2. **Integrate depth** from surface normals
3. **Relight** the object (here: with uniform albedo)

Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

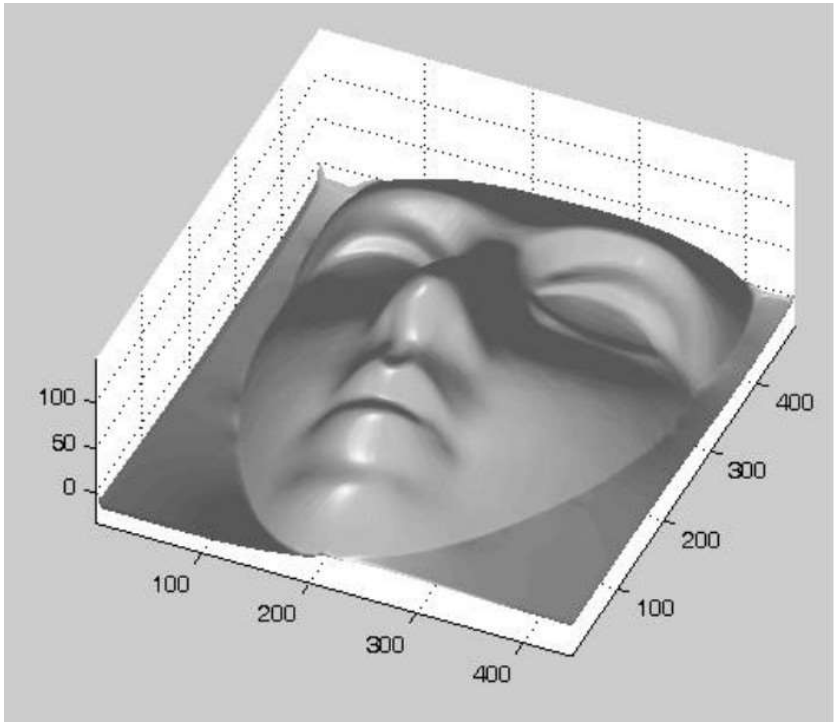
Photometric Stereo Algorithm

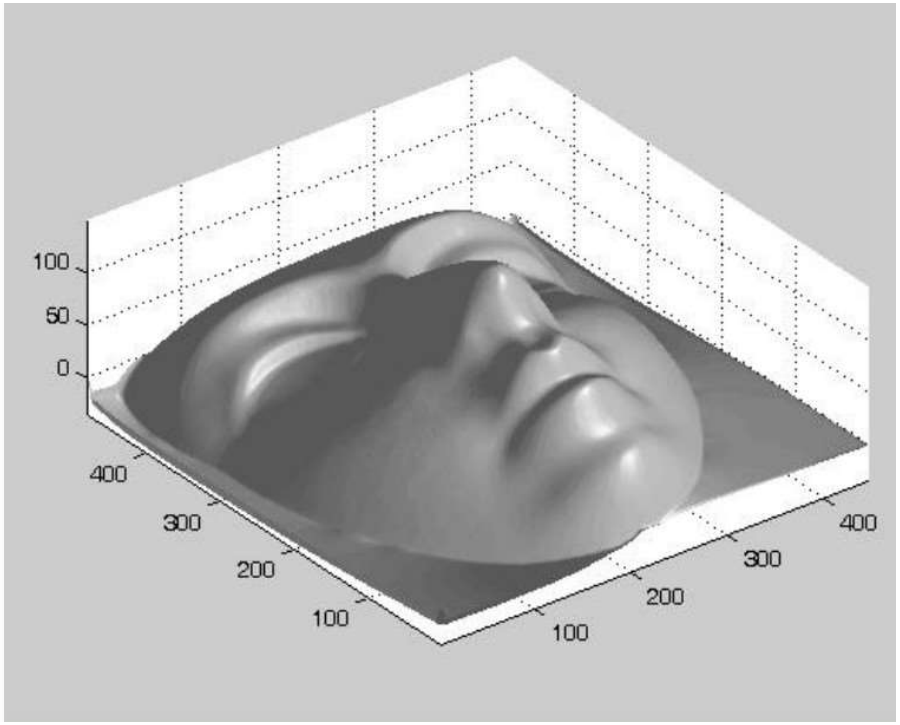


- For color images, apply PS to each channel separately to obtain color albedo
- Deviations from Lambertian assumption and global illumination cause errors

Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

Photometric Stereo Algorithm





- For color images, apply PS to each channel separately to obtain color albedo
- Deviations from Lambertian assumption and global illumination cause errors

Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

Photometric Stereo for Outdoor Webcams

Figure 1. The geographic context of the scene with camera (red marker) and target object (green marker) on the left, and an example image from the camera with sky mask (red region) and object mask (green region) on the right. Sat image © by Google Inc.

Figure 6. One input image, detected shadow regions, selected points for intensity estimation and the recovered object albedo.

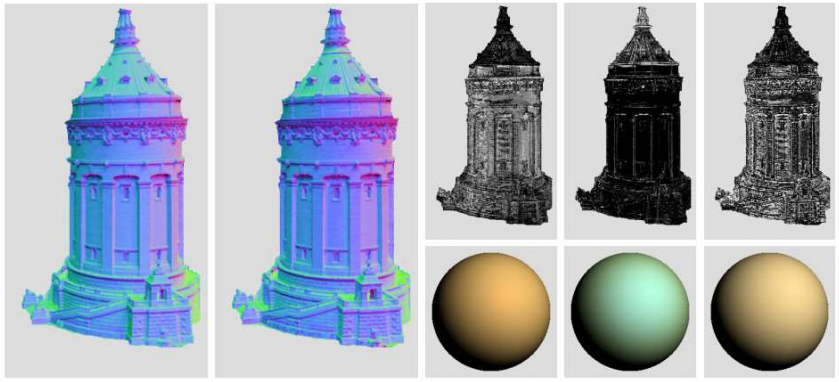


Figure 7. The initial normal map, the final normal map, and the four recovered BRDFs with corresponding material map.

Deep Uncalibrated Photometric Stereo

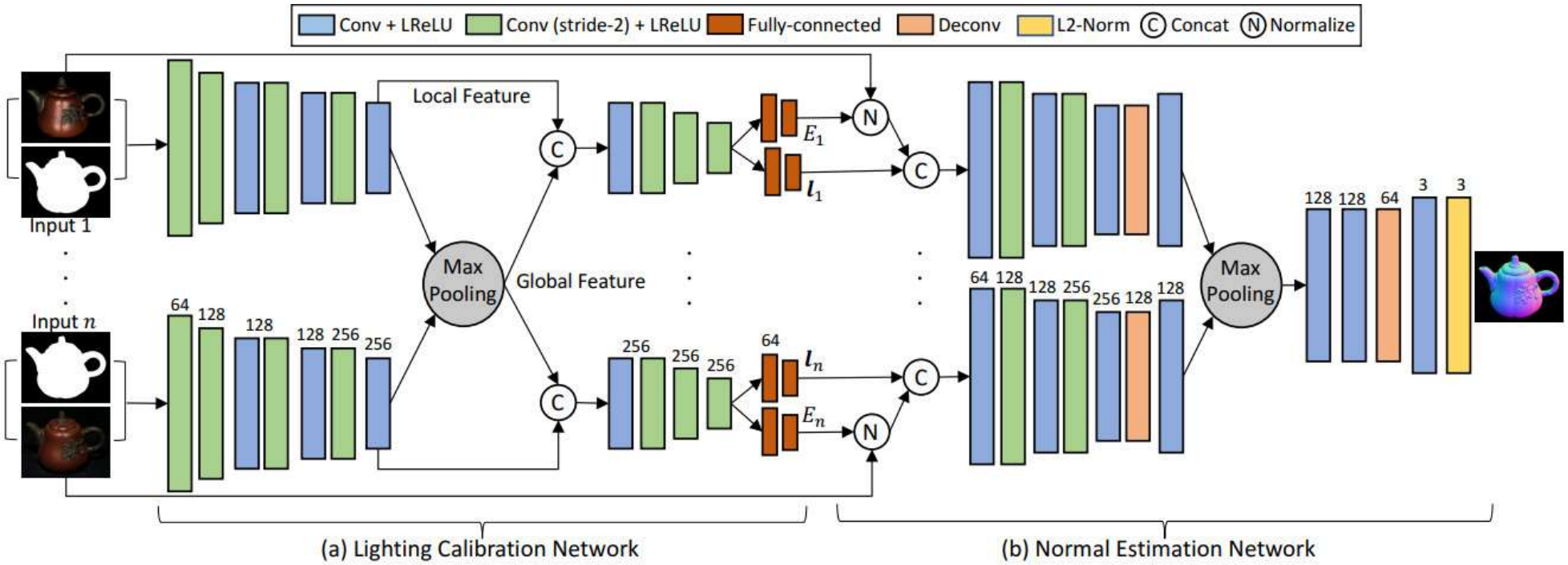
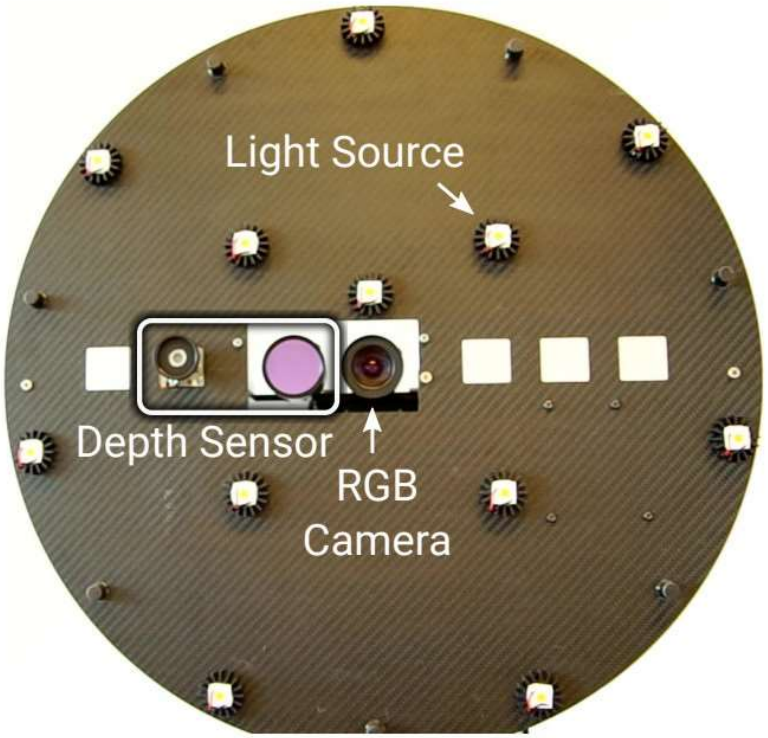


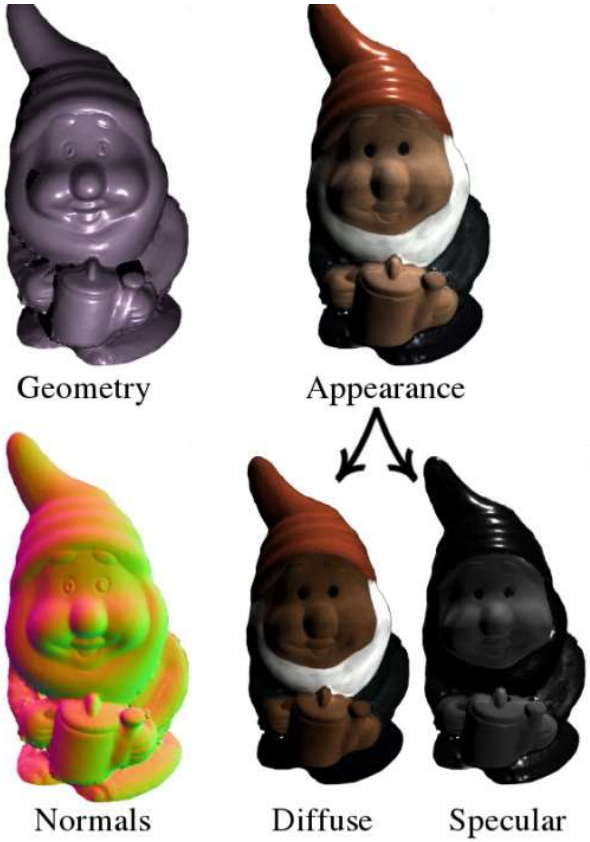
Figure 1. The network architecture of SDPS-Net is composed of (a) Lighting Calibration Network and (b) Normal Estimation Network. Kernel sizes for all convolutional layers are 3×3, and values above the layers indicate the number of feature channels.

Chen, Han, Shi, Matsushita and Wong: Self-Calibrating Deep Photometric Stereo Networks. CVPR, 2019.

Pose, Geometry and svBRDF from a Handheld Scanner

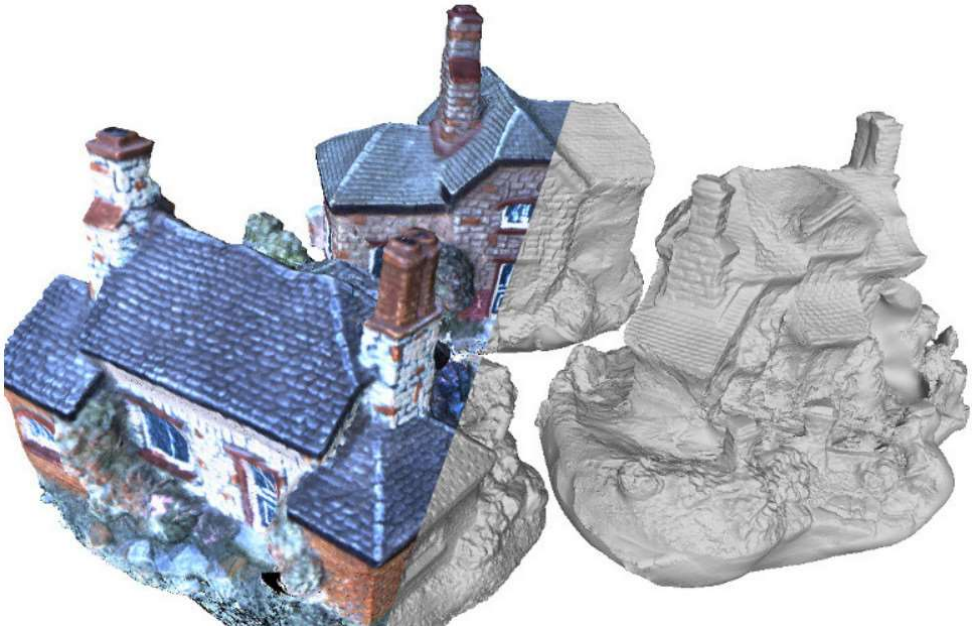


Input observations



Schmitt, Donne, Riegler, Koltun and Geiger: On Joint Estimation of Pose, Geometry and svBRDF from a Handheld Scanner. CVPR, 2020.

Volumetric Multi-View Photometric Stereo



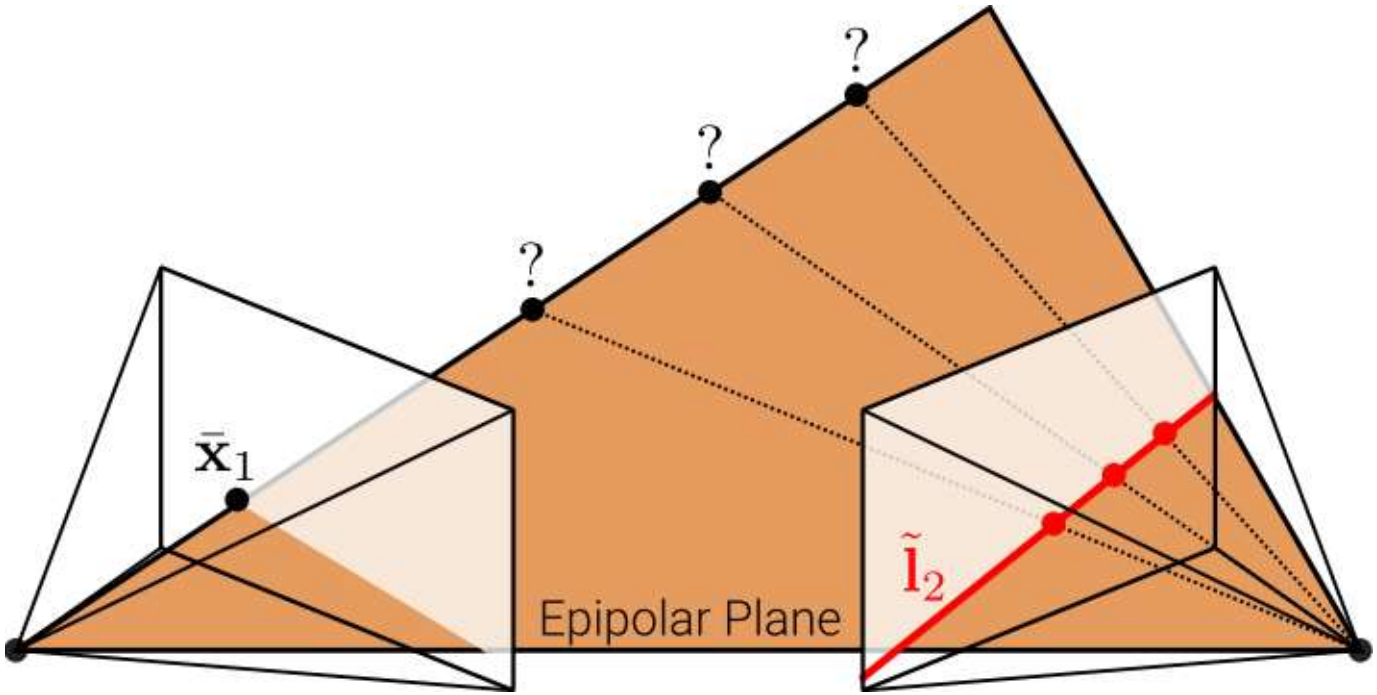
Current Research:

- Perspective PS, Uncalibrated PS, Deep PS, Volumetric PS, Mobile Setups
- Shadows, Estimation of non-lambertian materials (more parameters $\Rightarrow$ larger K)

Logothetis, Mecca and Cipolla: A differential volumetric approach to multi-view photometric stereo. ICCV, 2019

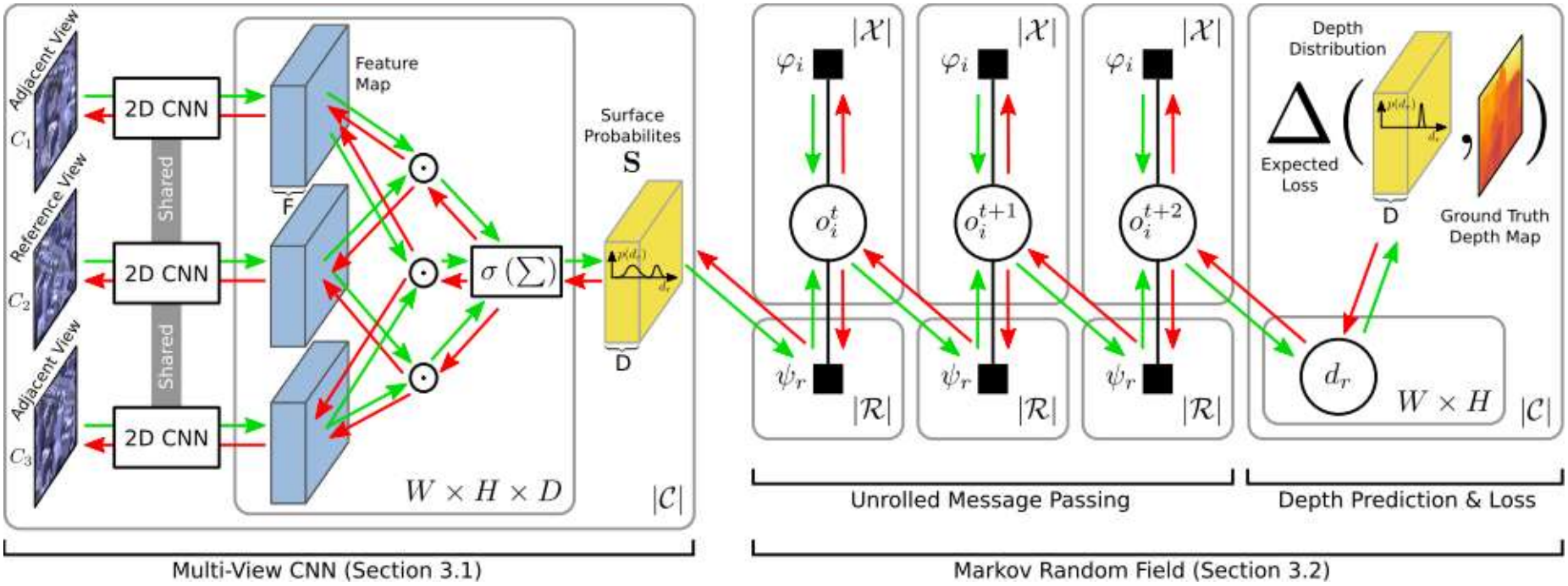
8.3
Shape-from-X

Binocular Stereo



- Find correspondences between two images

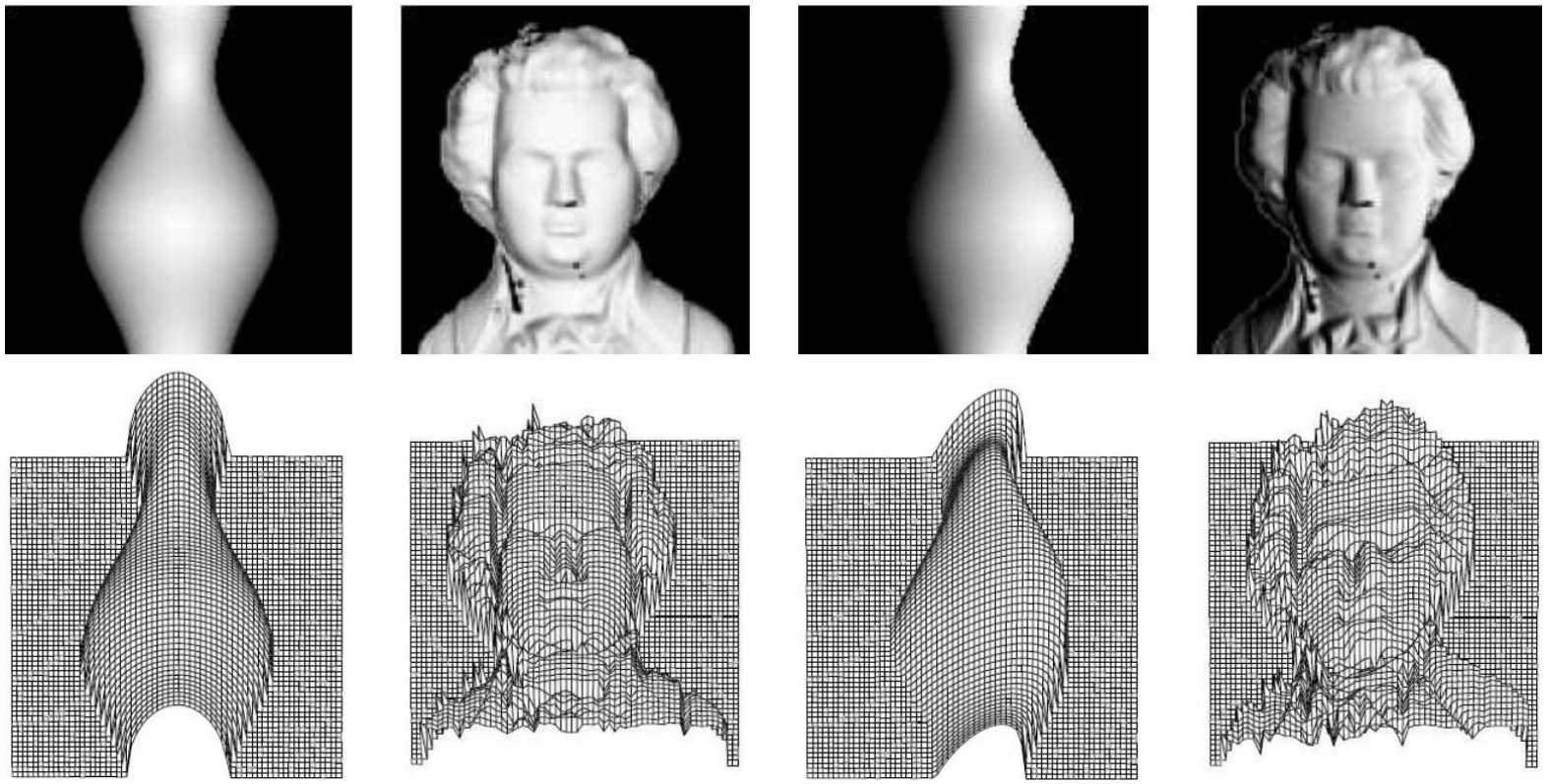
Multi-View Stereo



- Find correspondences between multiple images or estimate volumetric model

Paschalidou, Ulusoy, Schmitt, van Gool and Geiger: RayNet: Learning Volumetric 3D Reconstruction with Ray Potentials. CVPR, 2018.

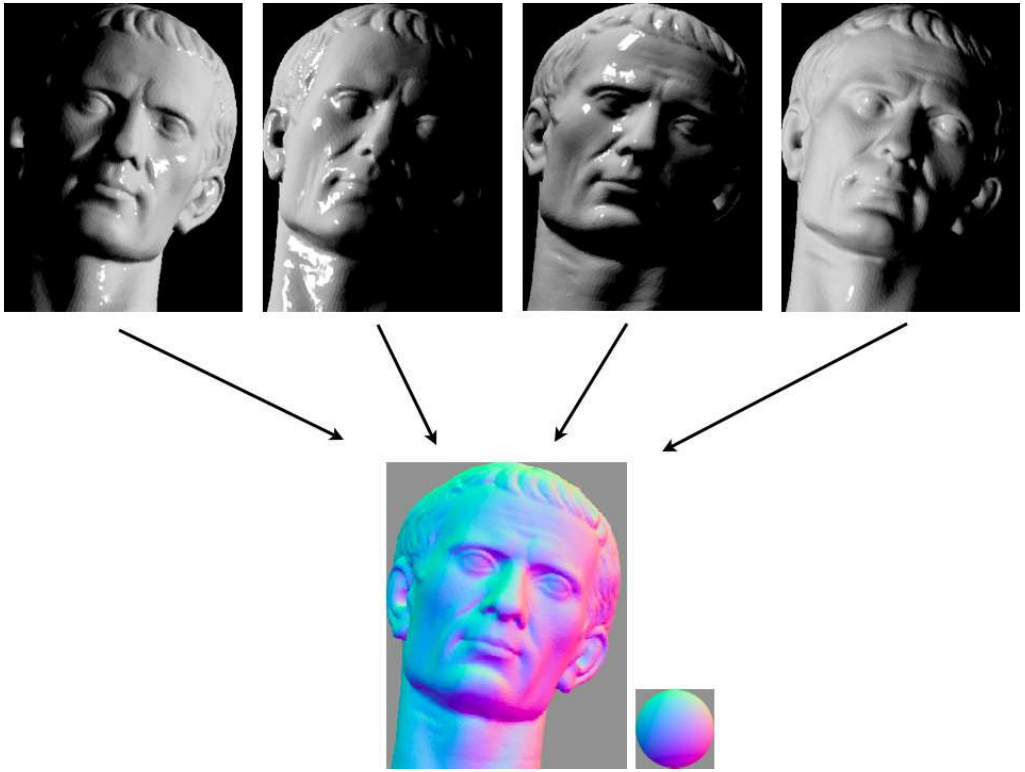
Shape-from-Shading



- Estimate shape from shading of a single image (smooth Lambertian scene)

Frankot and Chellappa: A Method for enforcing integrability in shape from shading algorithms. TPAMI, 1988.

Photometric Stereo



- Estimate shape (and albedo) from multiple images under varying illumination

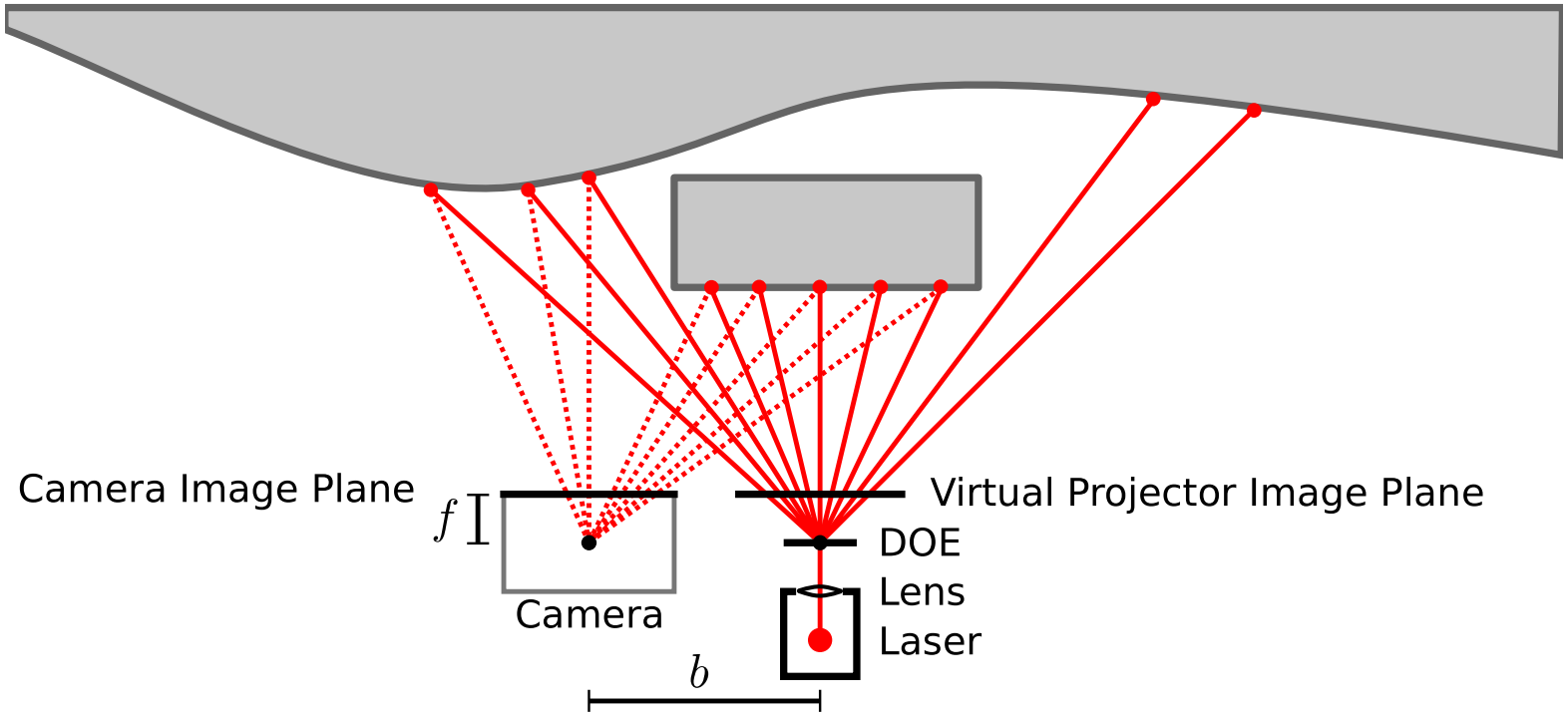
Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981.

Shape-from-Texture

- Estimate shape by exploiting regularity of local textural elements

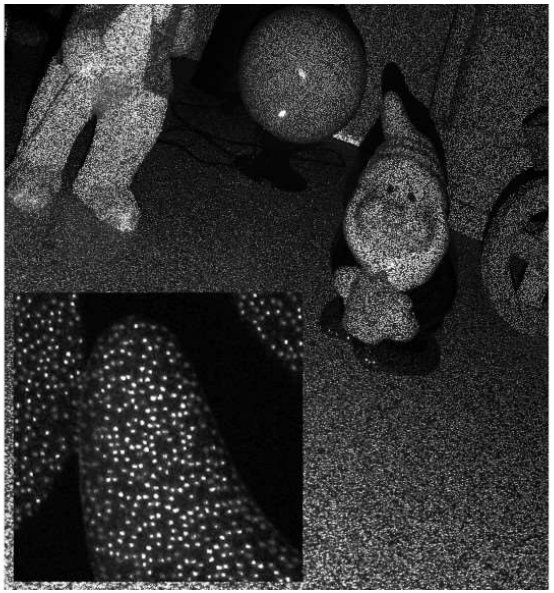
Verbin, E. Zickler: Toward a Universal Model for Shape From Texture. CVPR, 2020.

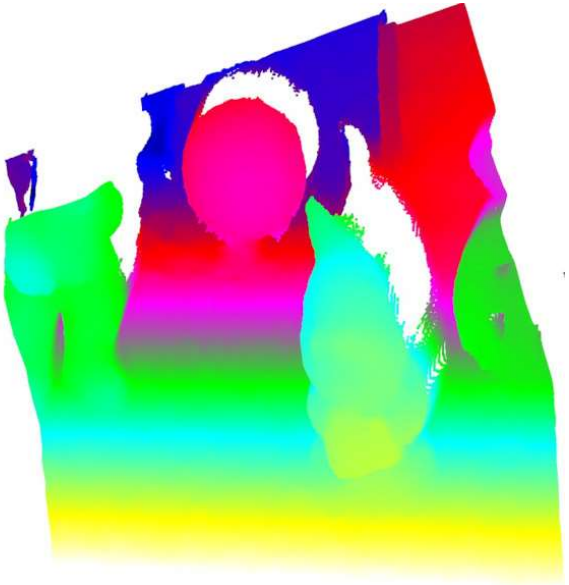
Structured Light



- Use a pattern projector to actively illuminate the scene
- If pattern is calibrated and known, one camera suffices

Structured Light

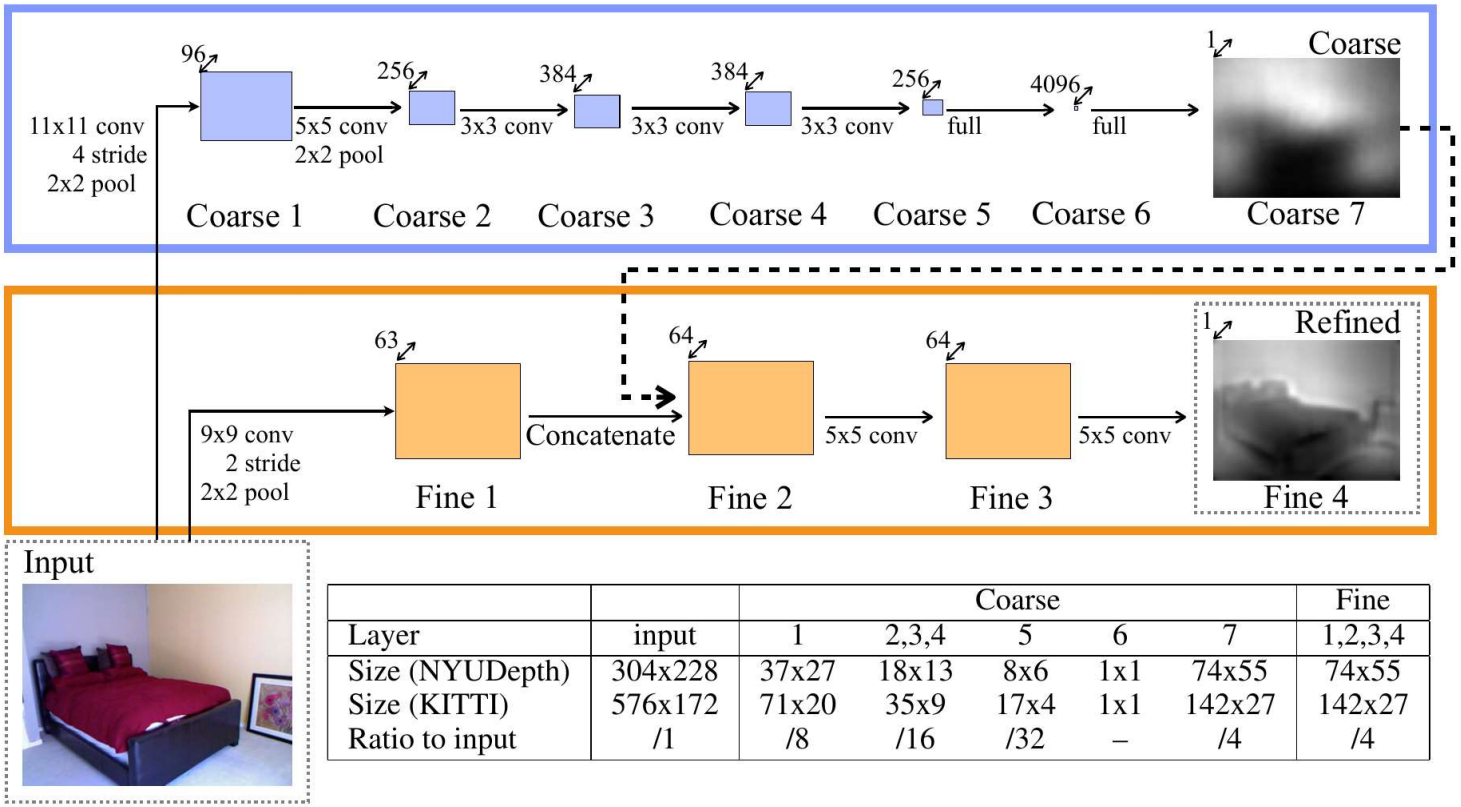




- Use a pattern projector to actively illuminate the scene
- If pattern is calibrated and known, one camera suffices

Tosi, Liao, Schmitt and Geiger: SMD-Nets: Stereo Mixture Density Networks. CVPR, 2021.

Monocular Depth Estimation



		Coarse					Fine
Layer	input	1	2,3,4	5	6	7	1,2,3,4
Size (NYUDepth)	304x228	37x27	18x13	8x6	1x1	74x55	74x55
Size (KITTI)	576x172	71x20	35x9	17x4	1x1	142x27	142x27
Ratio to input	/1	/8	/16	/32	–	/4	/4

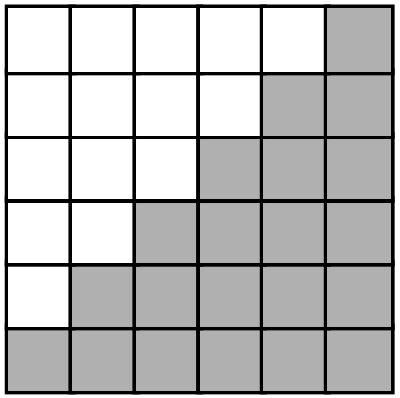
- Learn to predict depth from single RGB image using large annotated datasets

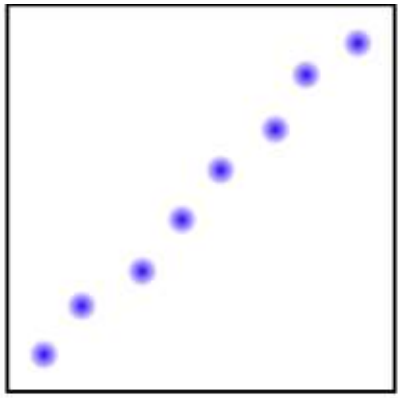
Eigen, Puhrsch and Fergus: Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. NIPS, 2014.

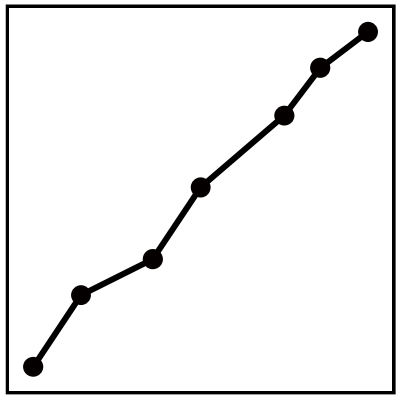
Monocular Depth Estimation

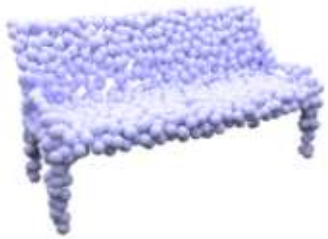
Lasinger, Ranftl, Schindler and Koltun: Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer. Arxiv, 2019.

Monocular Shape Estimation and Completion









Voxels

[Maturana et al., IROS 2015]

Points

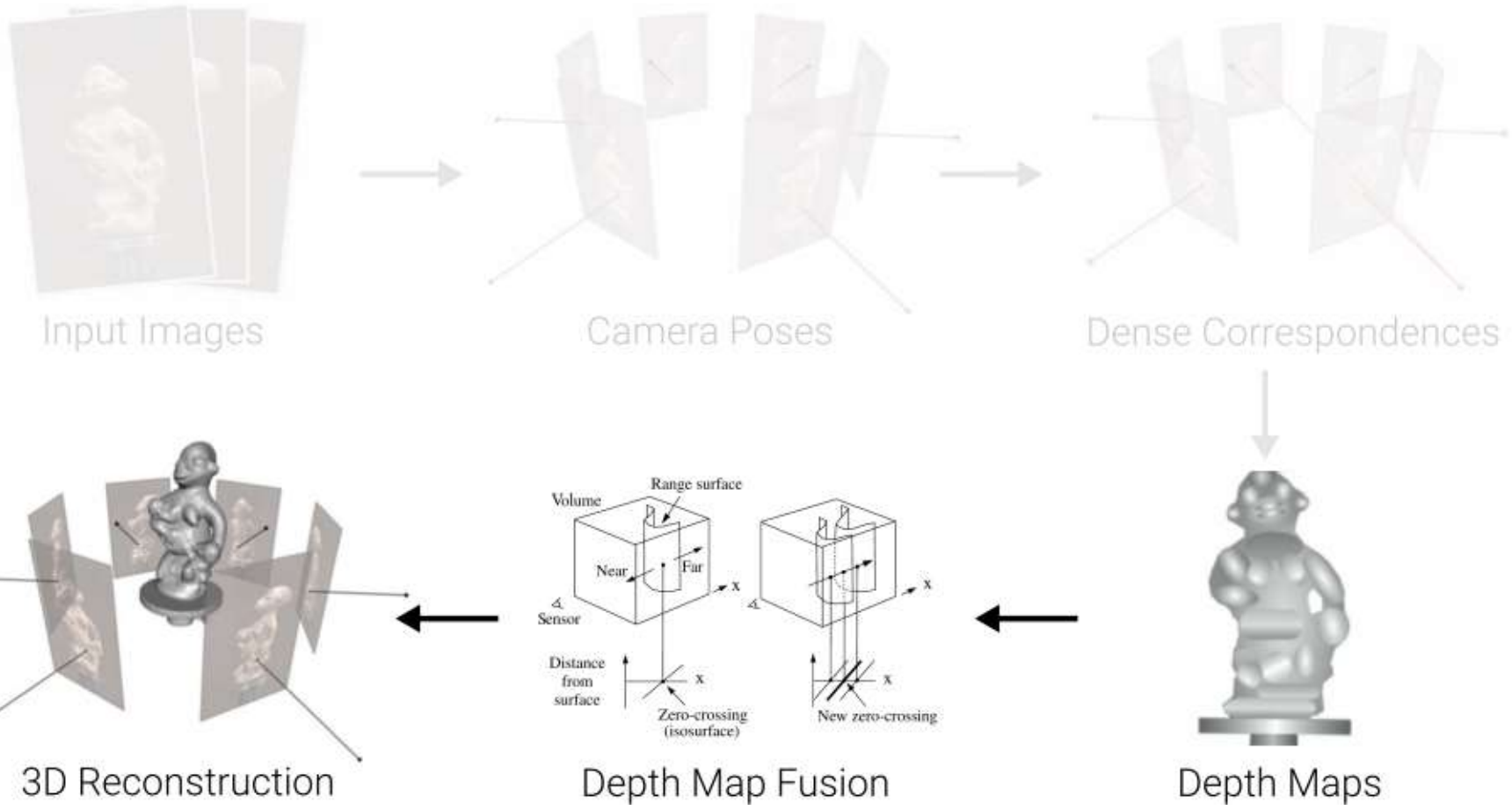
[Fan et al., CVPR 2017]

Meshes

[Groueix et al., CVPR 2018]

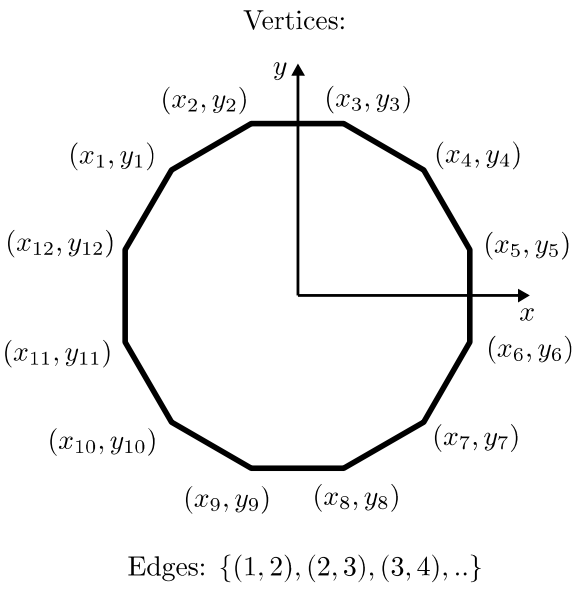
8.4
Volumetric Fusion

Traditional 3D Reconstruction Pipeline



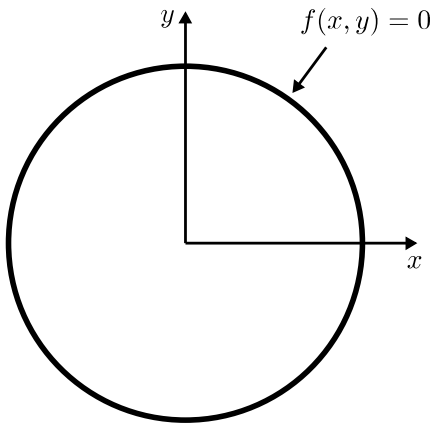
Representations

Mesh

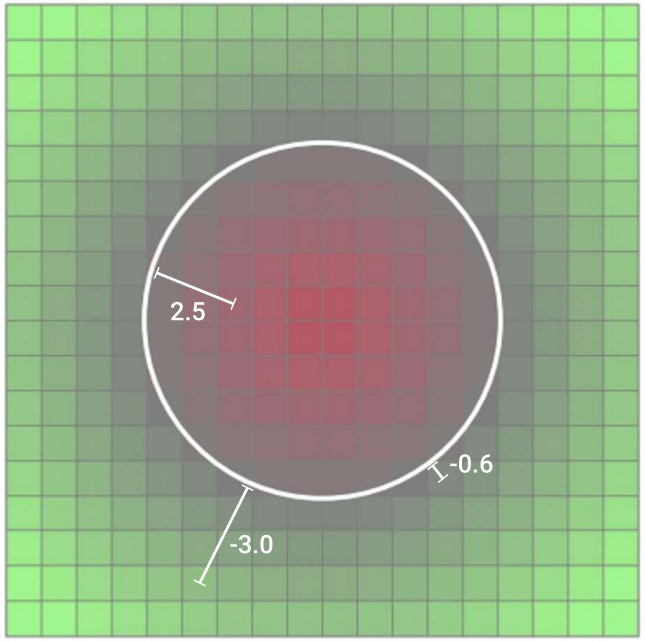


Implicit (Continuous)

$f(x, y) = x^2 + y^2 - r^2$

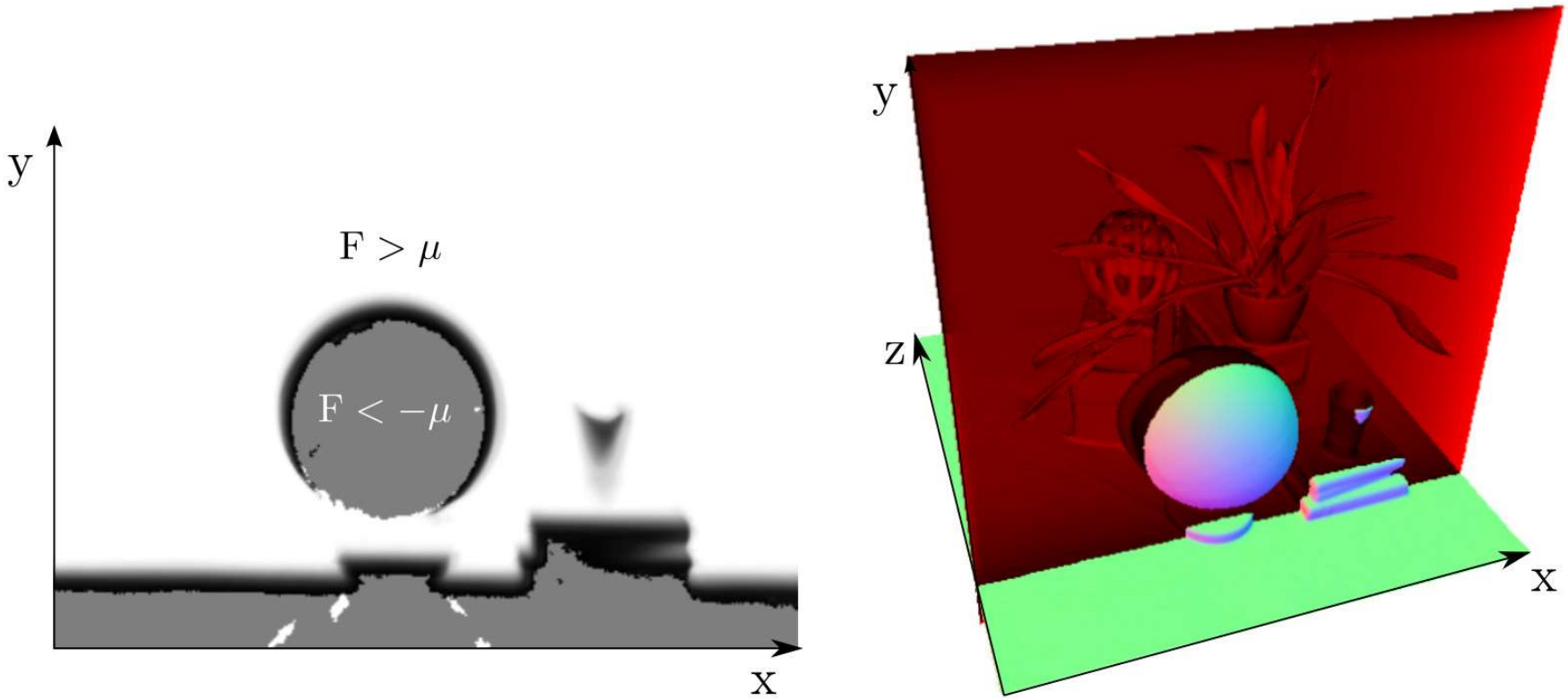


Implicit (Discrete SDF)



- SDF models store the **signed distance to the closest surface** at each voxel
- The **surface** is hence represented implicitly as the **zero level set**

Implicit Representations



- Crucial advantage: Implicit models allow for representing **arbitrary topology**

Implicit Representations

- Crucial advantage: Implicit models allow for representing **arbitrary topology**

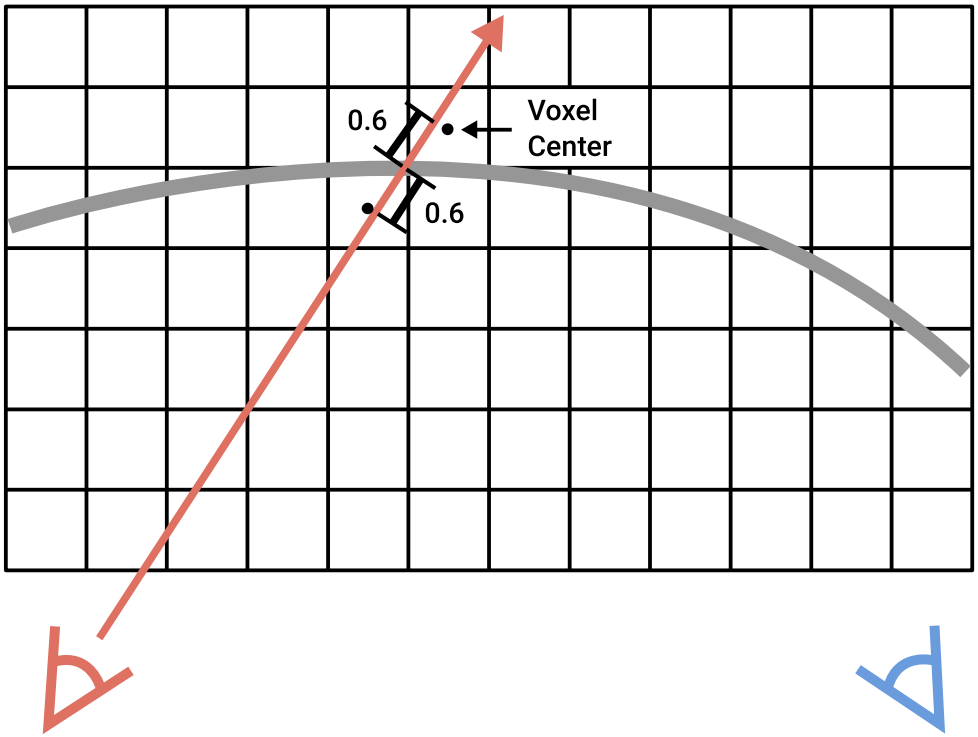
Volumetric Fusion

3 Steps:

- Depth-to-SDF Conversion
- Volumetric Fusion
- Mesh Extraction

Depth-to-SDF Conversion

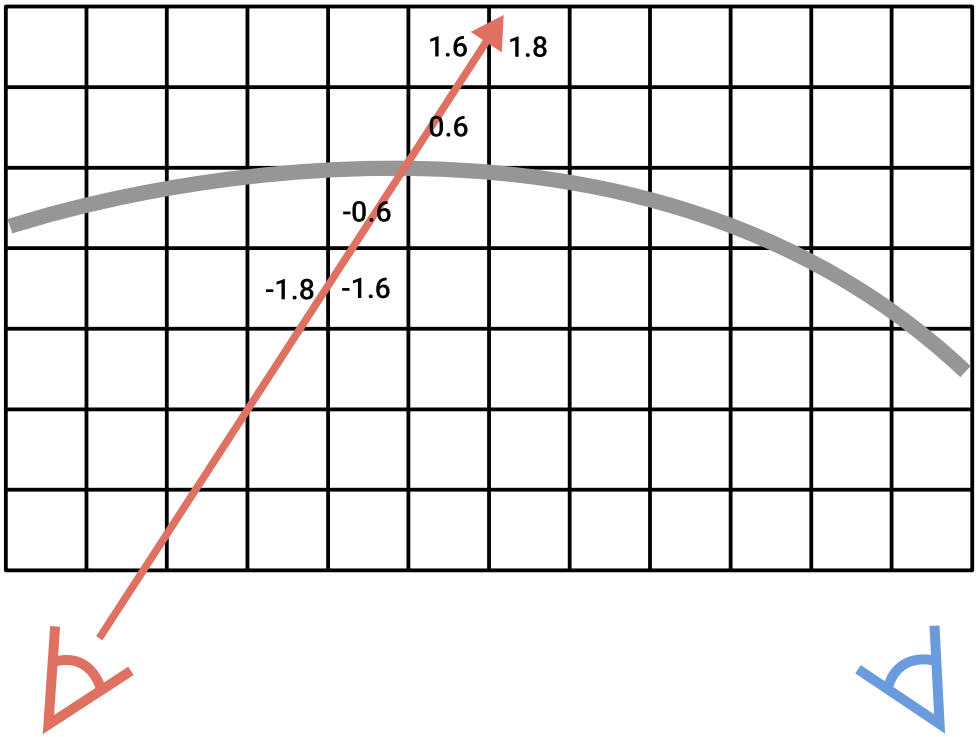
Depth-to-SDF Conversion



- As the distance to surface is unknown, approximate it with **distance along ray**
- This approximation is good only in the vincinity of the surface (often suffices)

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

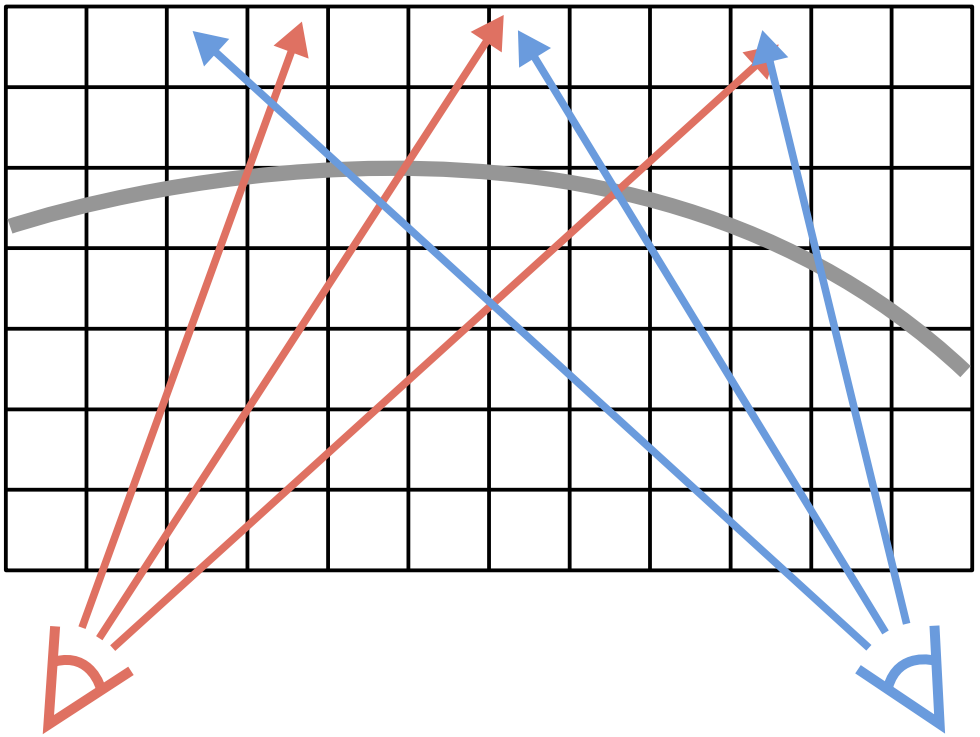
Depth-to-SDF Conversion



- As the distance to surface is unknown, approximate it with **distance along ray**
- This approximation is good only in the vincinity of the surface (often suffices)

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Depth-to-SDF Conversion

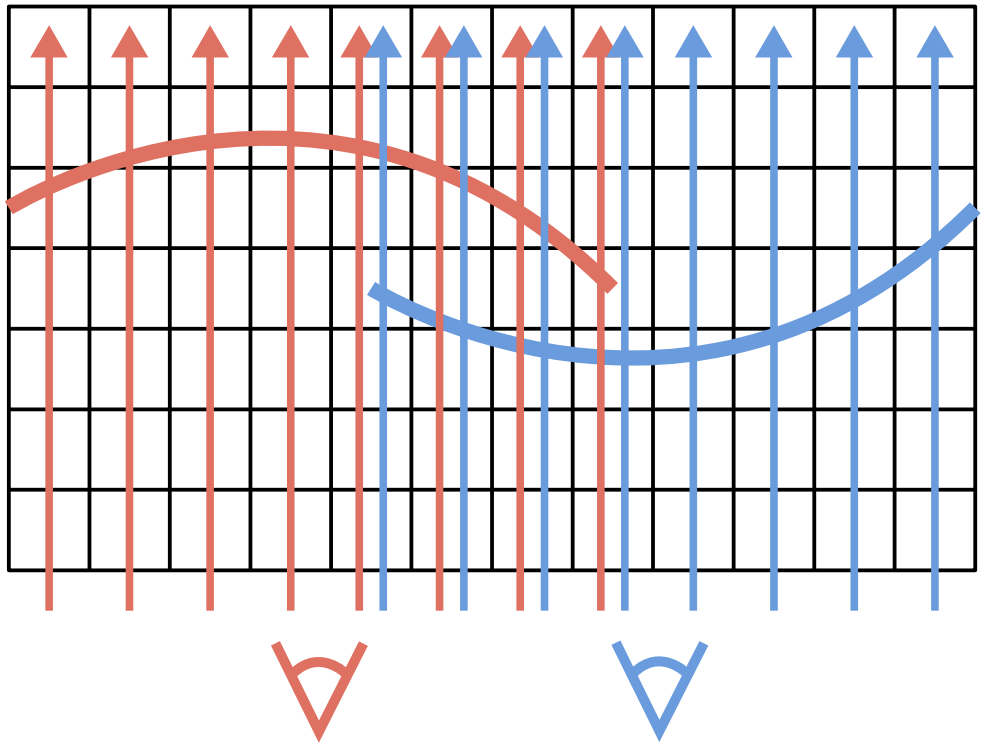


- As the distance to surface is unknown, approximate it with **distance along ray**
- This approximation is good only in the vincinity of the surface (often suffices)

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Volumetric Fusion

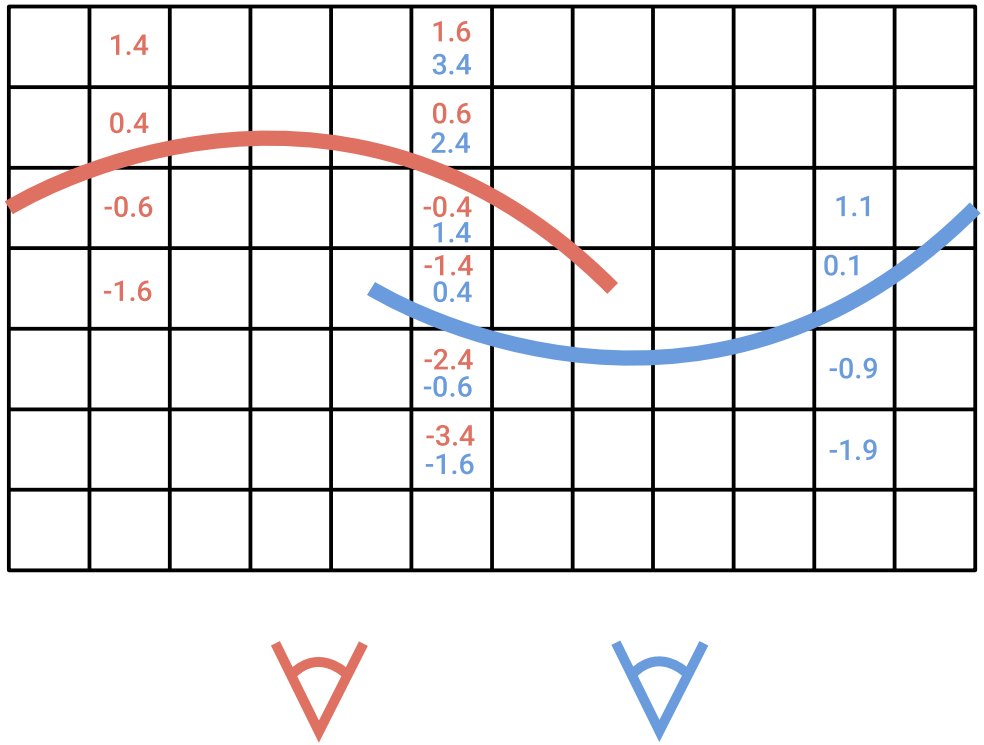
Volumetric Fusion (Orthographic Example)



- After conversion, calculate average of the discrete SDF fields
- The implicit surface will be an average of the two original ones

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

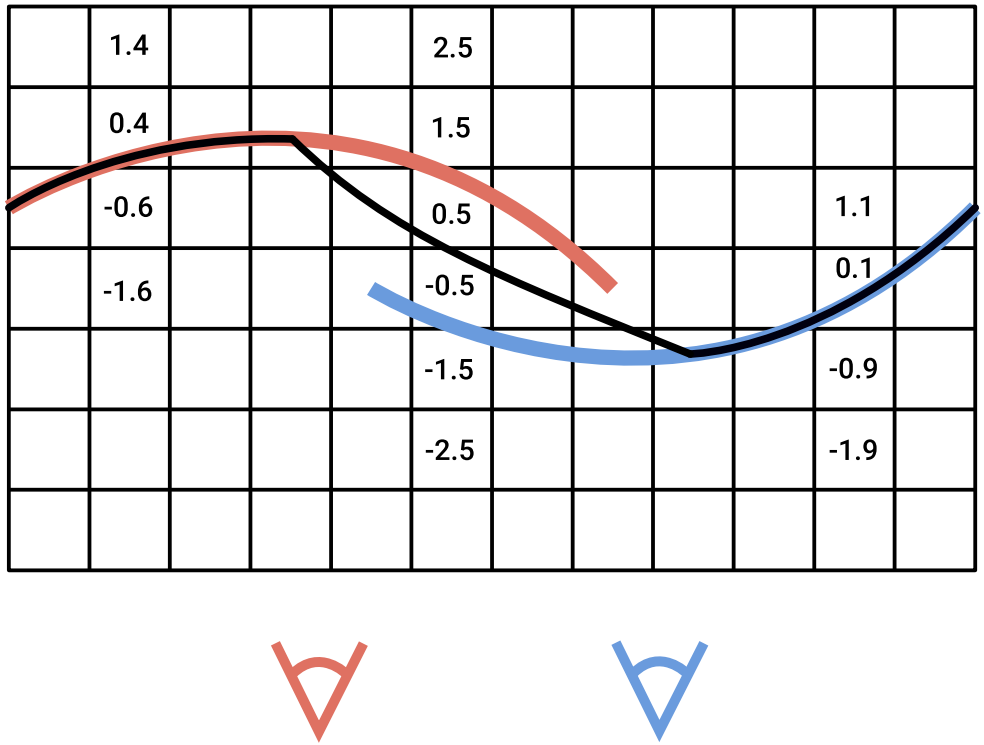
Volumetric Fusion (Orthographic Example)



- After conversion, calculate average of the discrete SDF fields
- The implicit surface will be an average of the two original ones

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Volumetric Fusion (Orthographic Example)



- After conversion, calculate average of the discrete SDF fields
- The implicit surface will be an average of the two original ones

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Volumetric Fusion Formulation

SDF Fusion calculates the **weighted average** per voxel:

$$D(\mathbf{x}) = \frac{\sum_i w_i(\mathbf{x})\, d_i(\mathbf{x})}{\sum_i w_i(\mathbf{x})}$$

$$W(\mathbf{x}) = \sum_i w_i(\mathbf{x})$$

- $w_i(\mathbf{x}), d_i(\mathbf{x})$: weight and distance along ray at voxel $\mathbf{x}$ for camera i
- $W(\mathbf{x}), D(\mathbf{x})$: sum of weights and fused distance at voxel $\mathbf{x}$

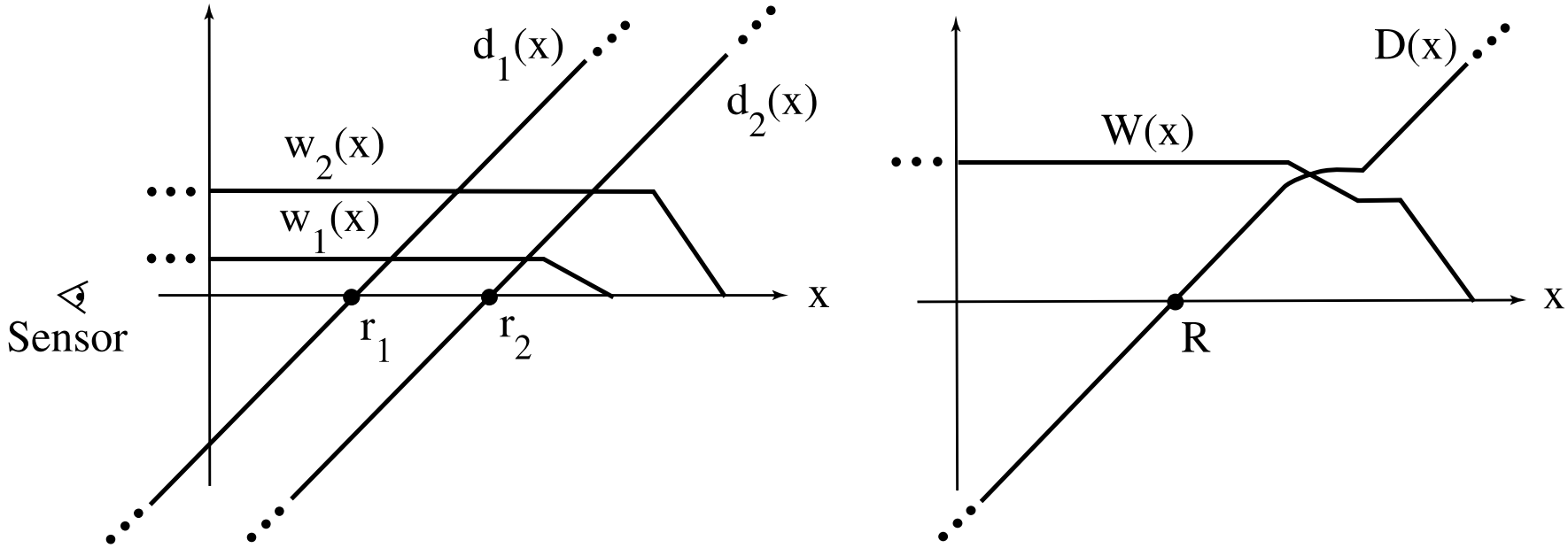
This can be conveniently expressed with an **incremental update rule:** (exercise)

$$D_{i+1}(\mathbf{x}) = \frac{W_i(\mathbf{x})\, D_i(\mathbf{x}) + w_{i+1}(\mathbf{x})\, d_{i+1}(\mathbf{x})}{W_i(\mathbf{x}) + w_{i+1}(\mathbf{x})}$$

$$W_{i+1}(\mathbf{x}) = W_i(\mathbf{x}) + w_{i+1}(\mathbf{x})$$

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Volumetric Fusion Formulation



- Weights useful to downweight SDF values **behind the surface** (no observations)
- Also used to downweight **less certain** rays (distance, orientation, boundaries)

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Volumetric Fusion Formulation

Why is weighted averaging a good idea?

$$D(\mathbf{x}) = \frac{\sum_i w_i(\mathbf{x})\, d_i(\mathbf{x})}{\sum_i w_i(\mathbf{x})}$$

It is the solution to the following **weighted least squares problem:** (exercise)

$$D^* = \underset{D}{\operatorname{argmin}} \sum_i w_i \left(d_i - D\right)^2$$

In other words, D is the optimal distance in least squares sense (at every location $\mathbf{x}$).

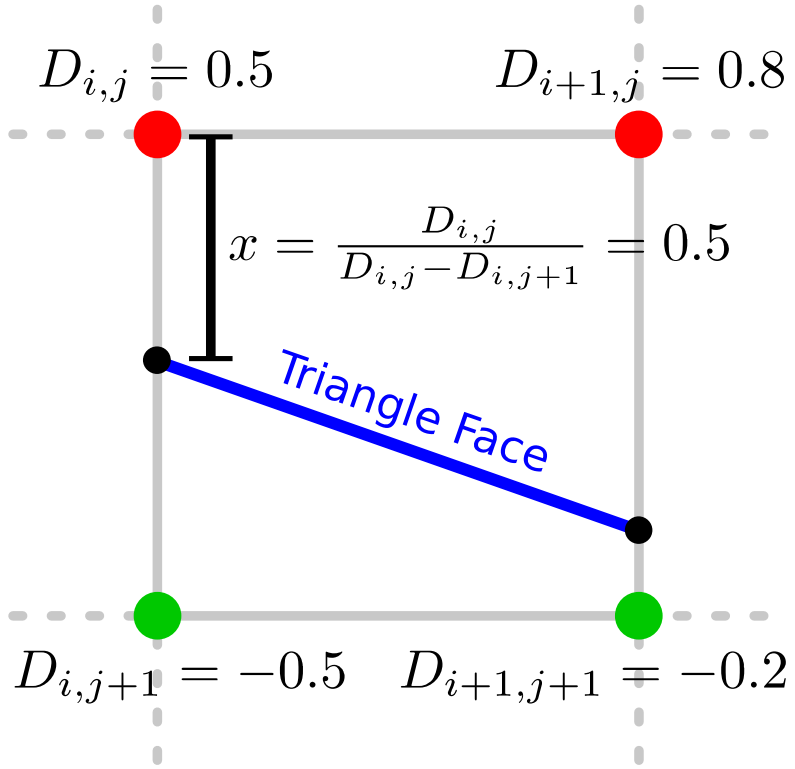
Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Mesh Extraction

Marching Cubes

How to extract a **mesh from the SDF?**

- Zero-level set $\Rightarrow$ triangular mesh
- "March" through all grid cells and insert triangle faces whenever a sign change is detected. 2 Steps:
  1. Estimate **topology**
  2. Predict **vertex location**
- In 3D: $2^8 = 256$ possible topologies
- Vertices are zero crossing of linear interpolant $f(x) = D + x(D' - D)$
- $f(x) \stackrel{!}{=} 0 \Rightarrow x = D/(D - D')$



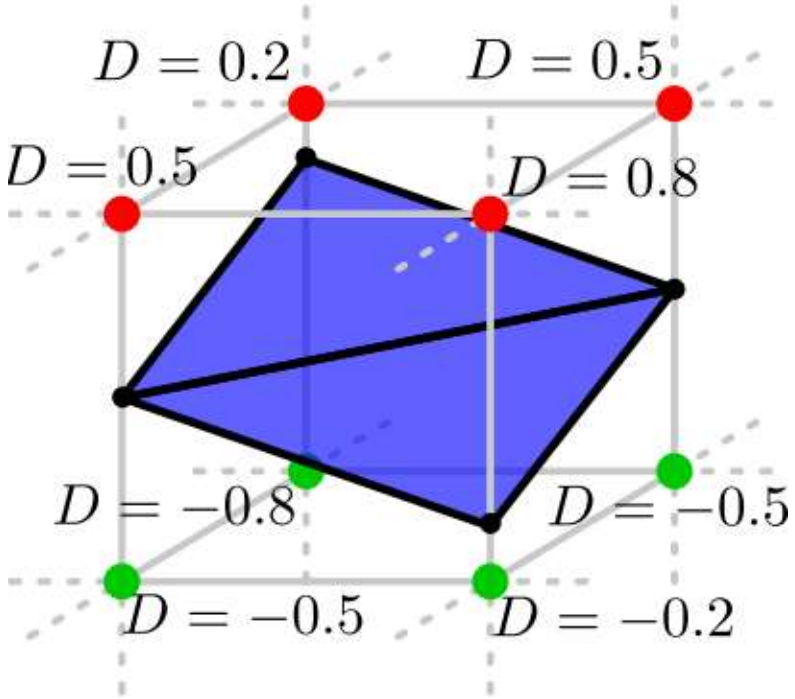
Red/green = **centers** of inside/outside cells

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Marching Cubes

How to extract a **mesh from the SDF?**

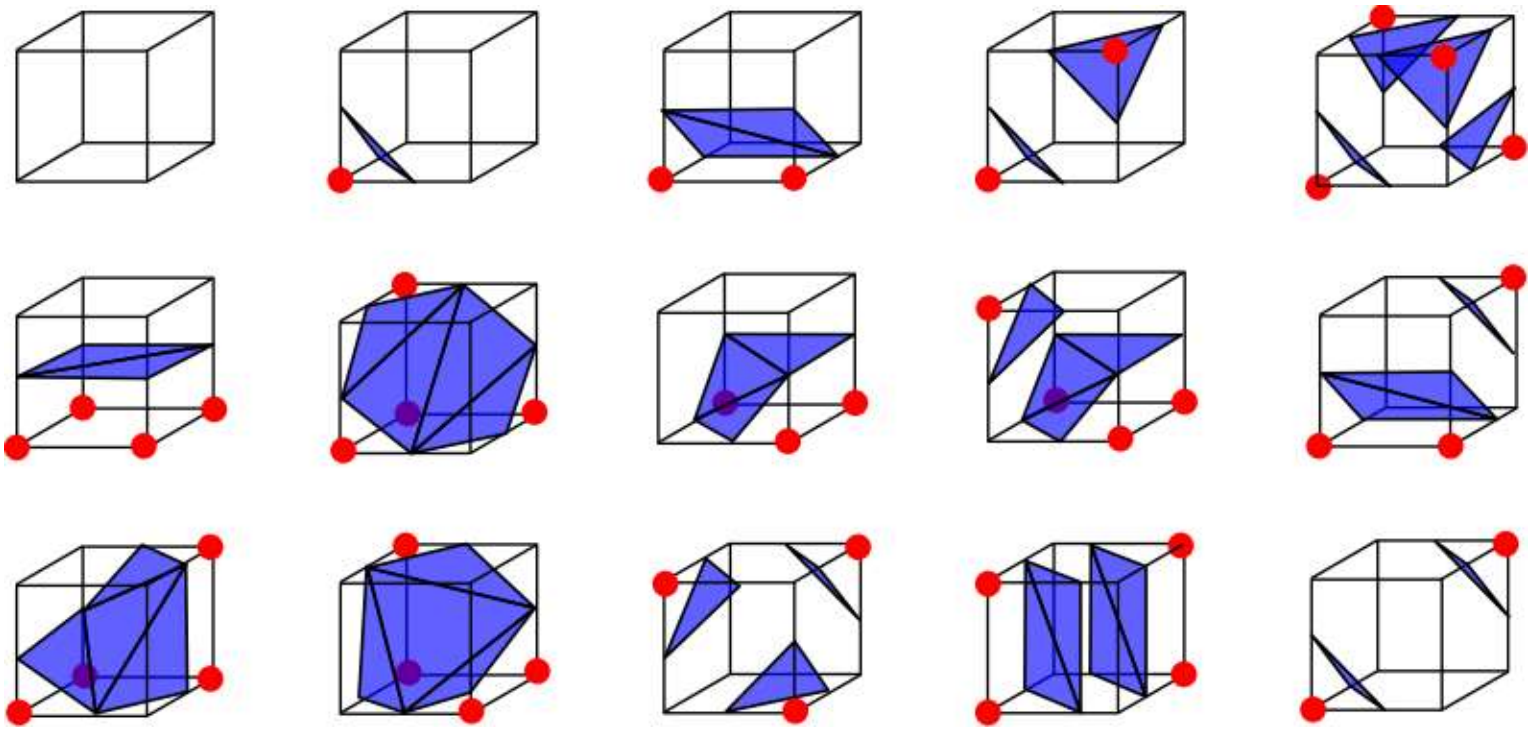
- Zero-level set $\Rightarrow$ triangular mesh
- "March" through all grid cells and insert triangle faces whenever a sign change is detected. 2 Steps:
  1. Estimate **topology**
  2. Predict **vertex location**
- In 3D: $2^8 = 256$ possible topologies
- Vertices are zero crossing of linear interpolant $f(x) = D + x(D' - D)$
- $f(x) \stackrel{!}{=} 0 \Rightarrow x = D/(D - D')$



Red/green = **centers** of inside/outside cells

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Marching Cubes



- The $2^8 = 256$ topologies can be grouped into 15 classes (rotational symmetry)

Curless and Levoy: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH, 1996.

Applications

KinectFusion

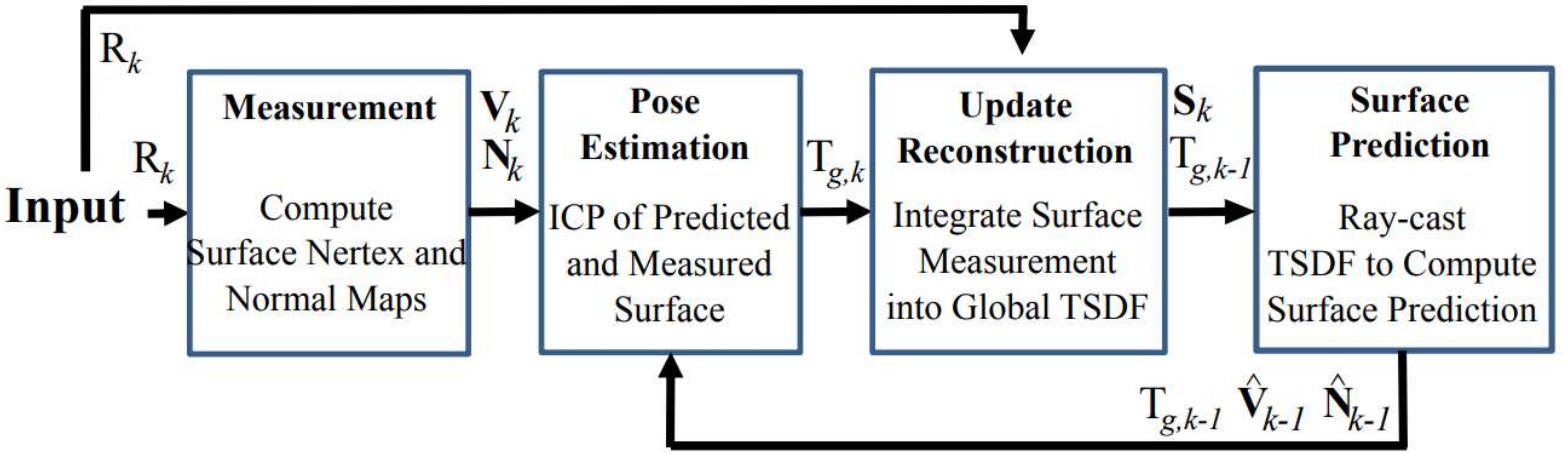
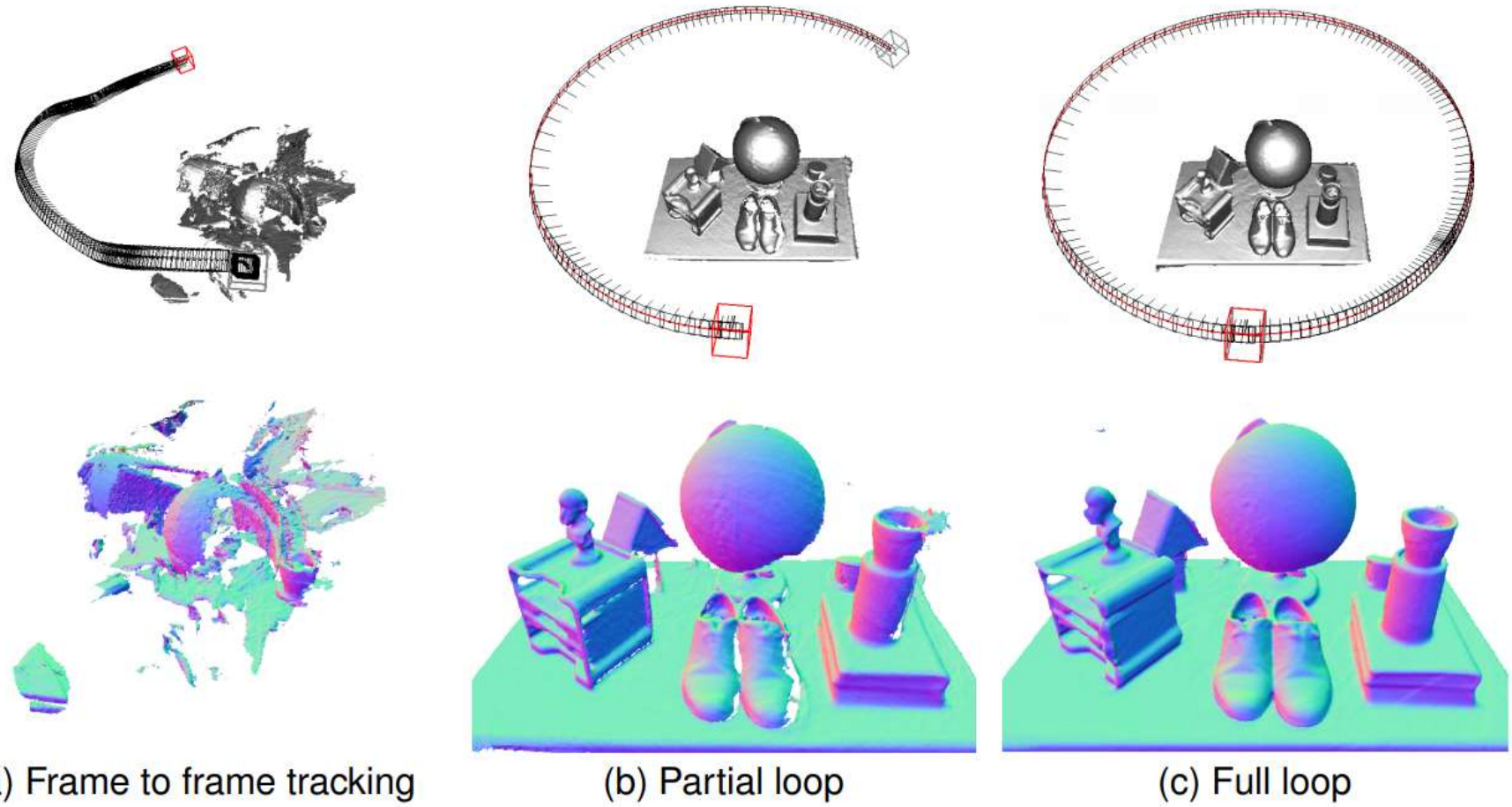


Figure 3: Overall system workflow.

- Real-time pose estimation and volumetric fusion using Kinect depth data as input

Newcombe et al.: KinectFusion: Real-time dense surface mapping and tracking. ISMAR, 2011.

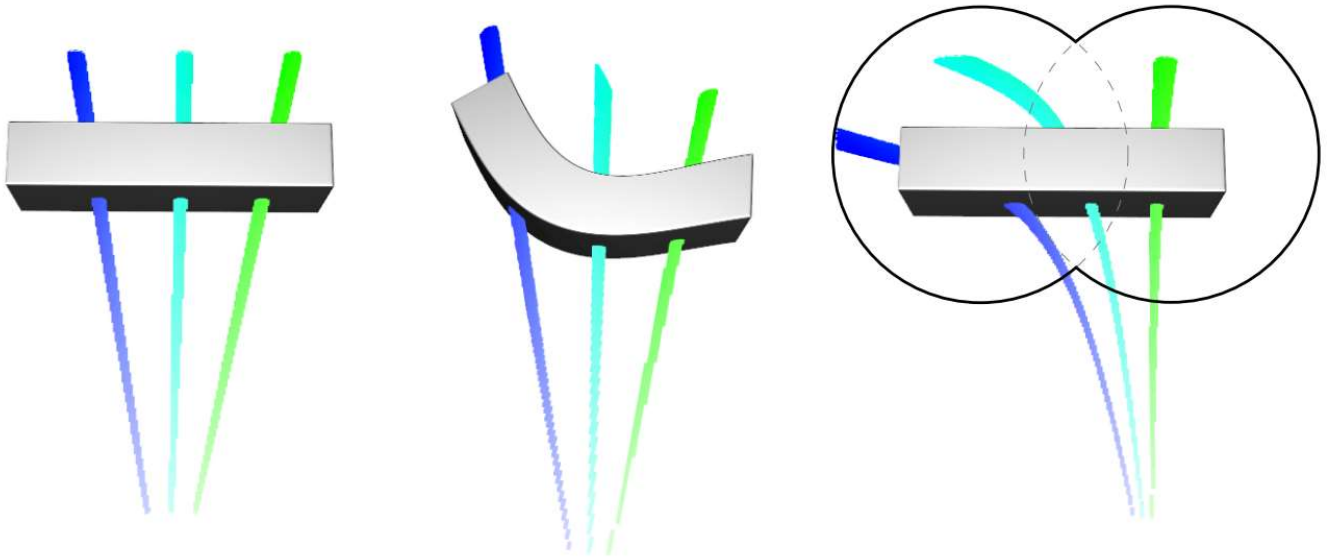
KinectFusion



(a) Frame to frame tracking (b) Partial loop (c) Full loop

Newcombe et al.: KinectFusion: Real-time dense surface mapping and tracking. ISMAR, 2011.

DynamicFusion

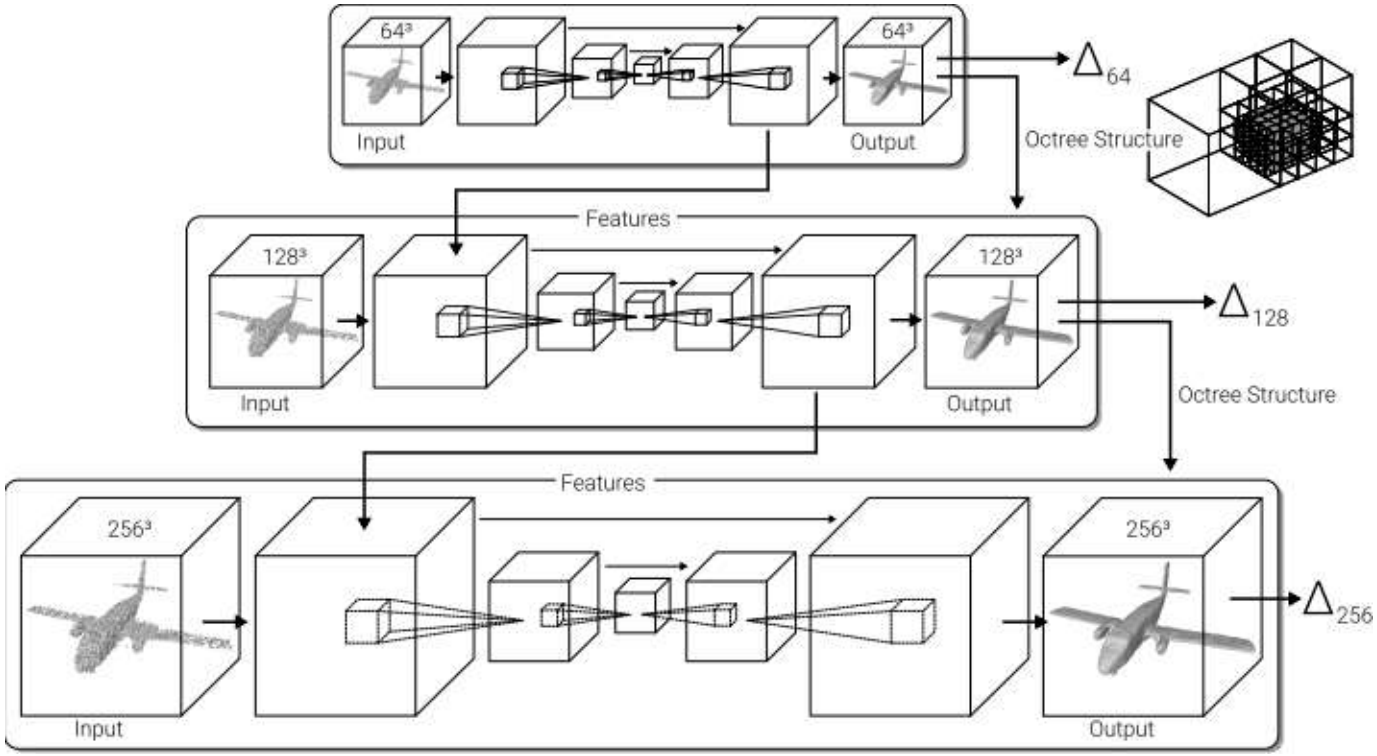


(a) Live frame $t = 0$ (b) Live Frame $t = 1$ (c) Canonical $\mapsto$ Live

- Reconstruct non-rigid scenes by warping the estimated geometry into a live frame

Newcombe, Fox, Seitz: DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. CVPR, 2015.

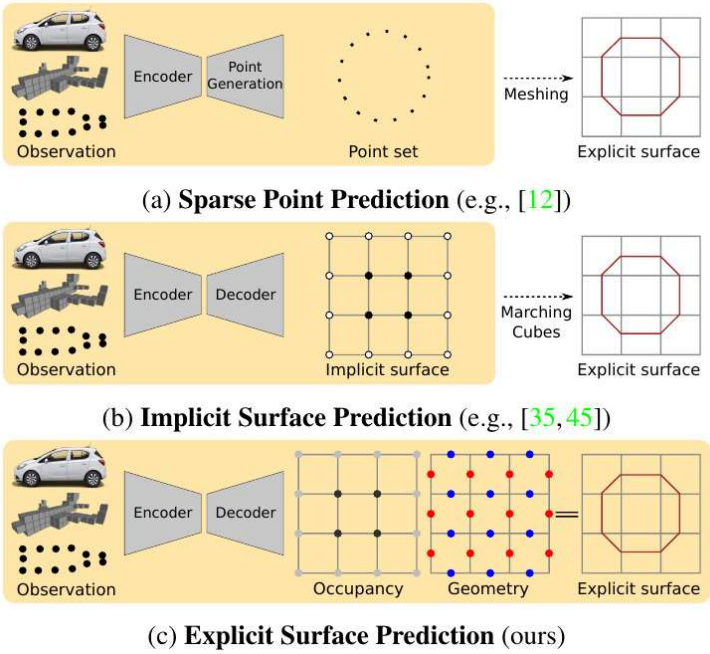
OctNetFusion

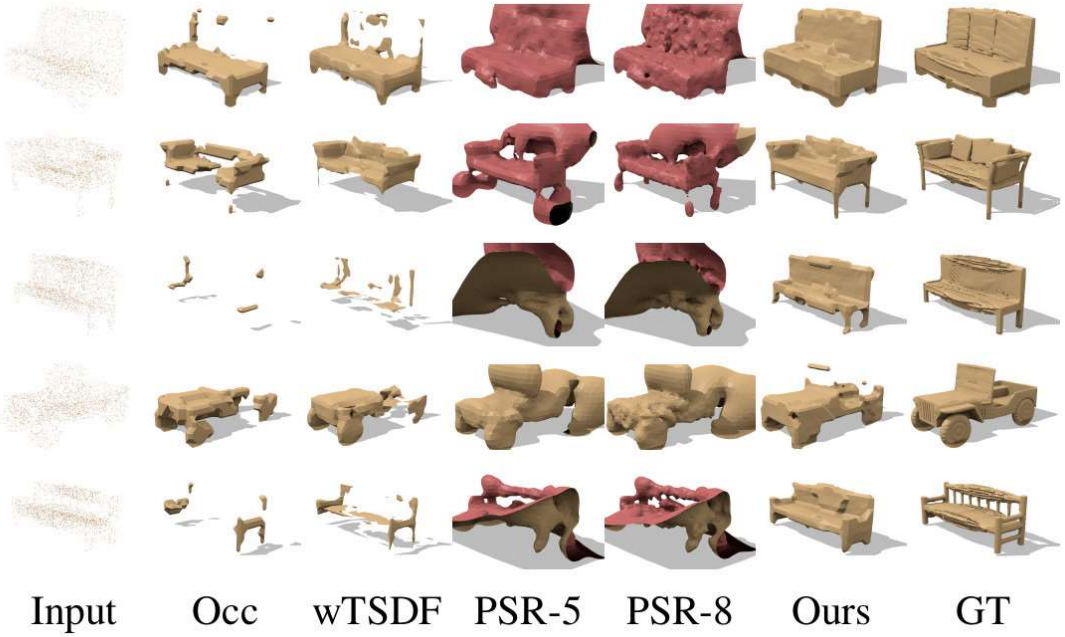


- Use fused geometry as features in a learning framework to complete and denoise

Riegler, Ulusoy, Bischof and Geiger: OctNetFusion: Learning Depth Fusion from Data. 3DV, 2017.

Deep Marching Cubes





- Differentiable variants enable supervision directly via 3D point clouds

Liao, Donné and Geiger: Deep Marching Cubes: Learning Explicit Surface Representations. CVPR, 2018.