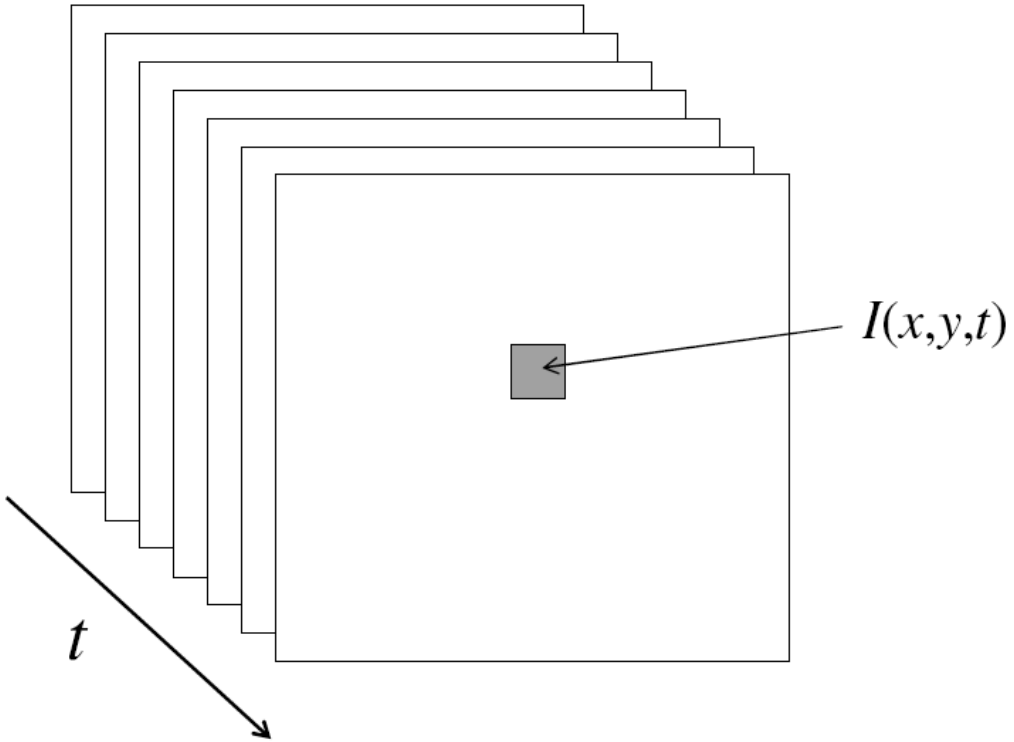
Optical Flow

Video

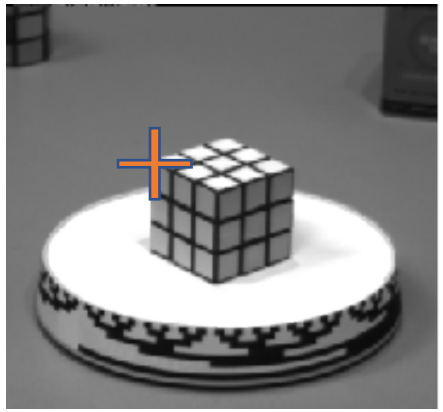
- A video is a sequence of frames captured over time
- Now our image data is a function of space (x, y) and time (t)



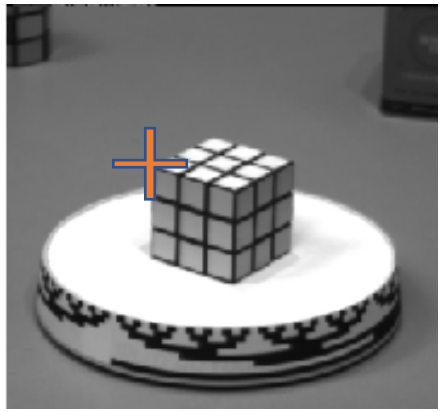
Credit: K. Grauman

Motion field

- The motion field is the projection of the 3D scene motion into the image

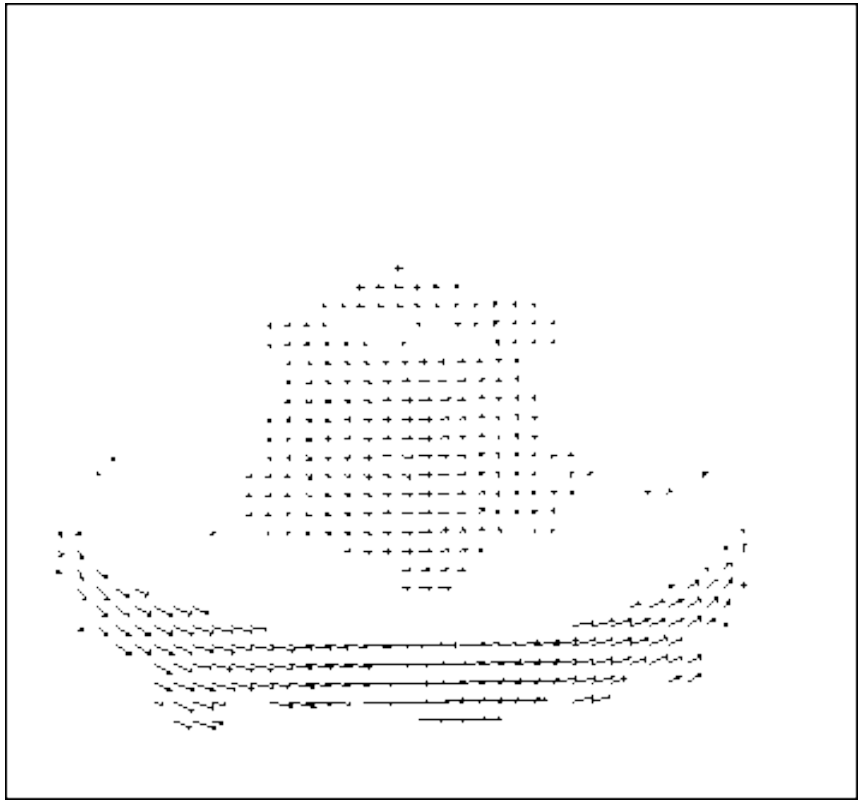


(100, 100)



(101, 102)

(u,v) = (1, 2)



Credit: K. Grauman

Optical flow

- Definition: optical flow is the *apparent* motion of brightness patterns in the image

Credit: K. Grauman

Optical flow

- Definition: optical flow is the *apparent* motion of brightness patterns in the image
- Ideally, optical flow would be the same as the motion field
- We need to be careful: apparent motion can be caused by lighting changes without any actual motion

Credit: K. Grauman

Apparent motion != motion field

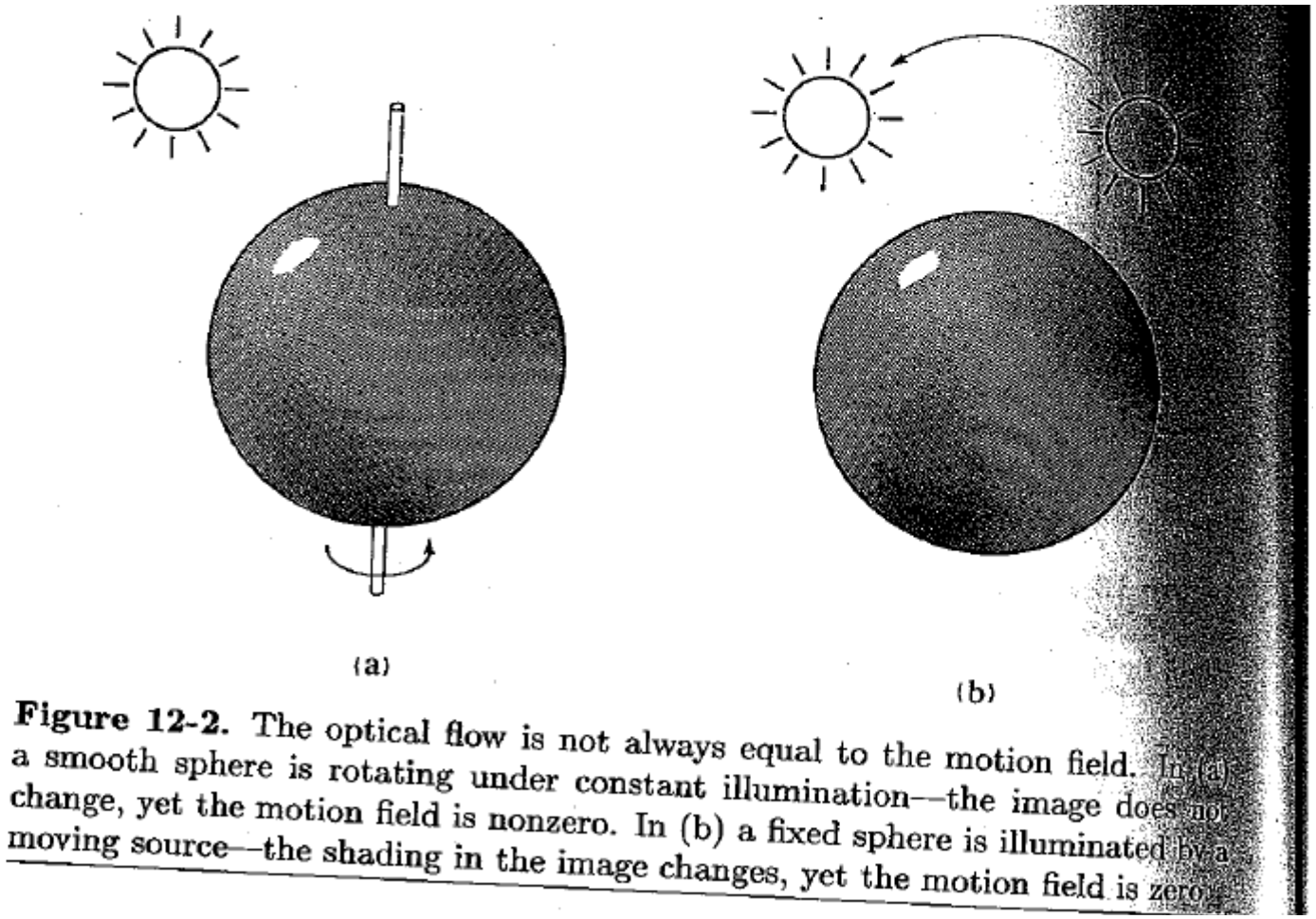
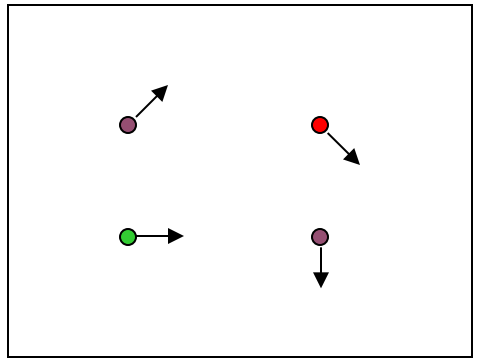


Figure 12-2. The optical flow is not always equal to the motion field. In (a) a smooth sphere is rotating under constant illumination—the image does not change, yet the motion field is nonzero. In (b) a fixed sphere is illuminated by a moving source—the shading in the image changes, yet the motion field is zero.

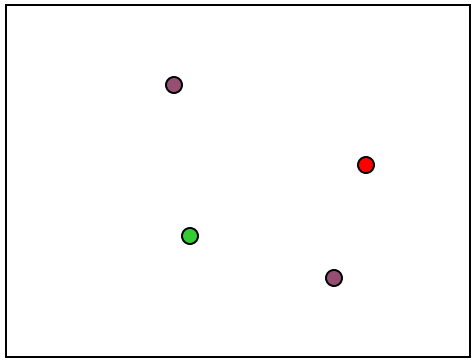
Figure from Horn book

Credit: K. Grauman

Problem definition: optical flow



$H(x, y)$



$I(x, y)$

- How to estimate pixel motion from image H to image I?
  - Solve pixel correspondence problem
    - given a pixel in H, look for nearby pixels of the same color in I

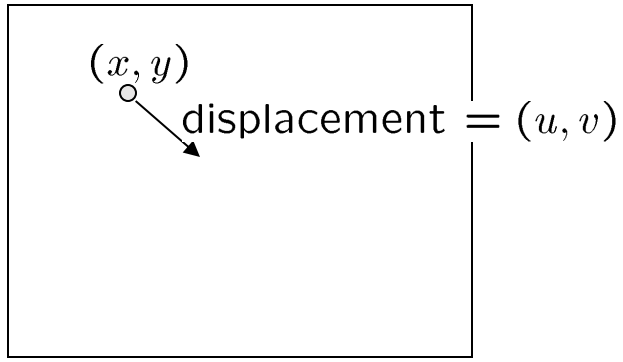
Key assumptions

- **color constancy:** a point in H looks the same in I
  - For grayscale images, this is **brightness constancy**
- **small motion**: points do not move very far

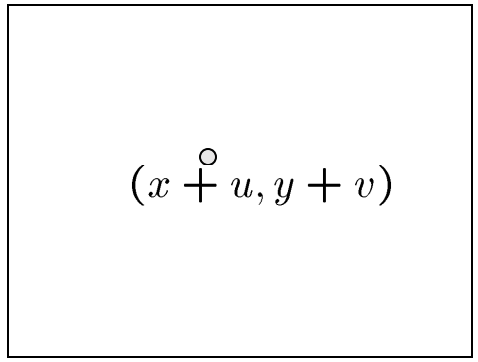
This is called the **optical flow** problem

Credit: K. Grauman

Optical flow constraints



$H(x, y)$



$I(x, y)$

- Let's look at these constraints more closely
  - brightness constancy: Q: what's the equation?

$$H(x, y) = I(x + u, y + v)$$

  - small motion:

$$I(x+u, y+v) = I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

$$\approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

Optical flow equation

shorthand: $I_x = \frac{\partial I}{\partial x}$

- Combining these two equations

$$
\begin{aligned}
0 &= I(x+u, y+v) - H(x,y) \\
&\approx I(x,y) + I_x u + I_y v - H(x,y) \\
&\approx (I(x,y) - H(x,y)) + I_x u + I_y v \\
&\approx I_t + I_x u + I_y v \\
&\approx I_t + \nabla I \cdot [u \ v]
\end{aligned}
$$

Optical flow equation

$$0 = I_t + \nabla I \cdot [u \ v]$$

- Q: how many unknowns and equations per pixel?

2 unknowns, 1 equation

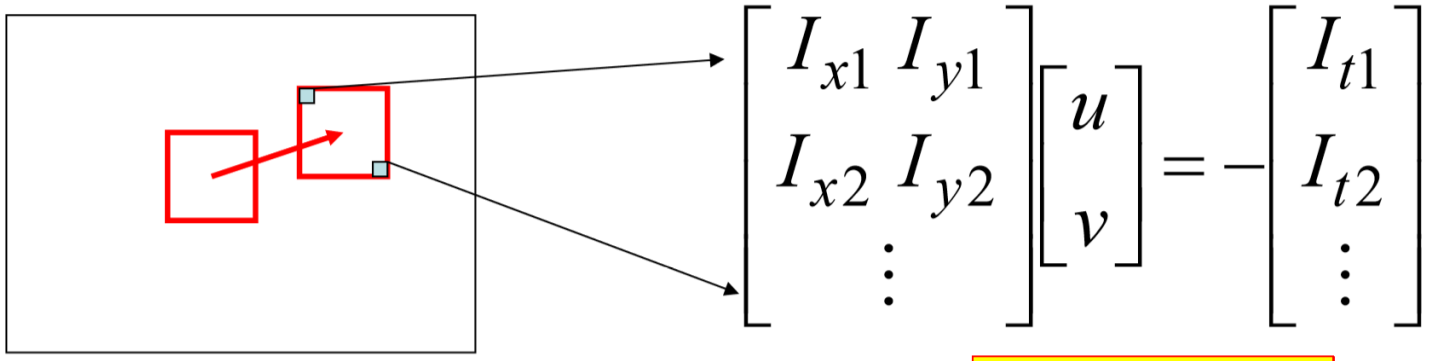
Solving the optical flow problem

- How to get more equations for a pixel?
- **Spatial coherence constraint:** pretend the pixel's neighbors have the same (u,v)
  - If we use a 5x5 window, that gives us 25 equations per pixel $0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u\ v]$

Local Patch Constant Motion: Lucas–Kanade optical flow algorithm

$$I_x u + I_y v = -I_t \quad \Rightarrow \quad \begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = -I_t$$

Assume constant motion (u,v) in small neighborhood



$$\begin{bmatrix} I_{x1} & I_{y1} \\ I_{x2} & I_{y2} \\ \vdots & \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_{t1} \\ I_{t2} \\ \vdots \end{bmatrix}$$

$$A\vec{u} = b$$

5x5的矩阵（Ix1 Iy1 … Ix5 Iy5），
视为共用相同的变换 (u, v)。

Lucas–Kanade Optical Flow Algorithm

Goal: Minimize $\left\| A\vec{\mathrm{u}} - b \right\|^2$

Method: Least-Squares

$$A\vec{\mathrm{u}} = b$$

$$\Downarrow$$

$$\underbrace{A^T A}_{2\mathrm{x}2}\ \underbrace{\vec{\mathrm{u}}}_{2\mathrm{x}1} = \underbrace{A^T b}_{2\mathrm{x}1}$$

$$\Downarrow$$

$$\vec{\mathrm{u}} = \left(A^T A\right)^{-1} A^T b$$

Solving the optical flow problem

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

- The summations are over all pixels in the K x K window
- This technique was first proposed by Lucas & Kanade (1981)

Slide credit: Steve Seitz

Conditions for solvability

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $\qquad$ $A^T b$

When is this solvable?

- $\mathbf{A^T A}$ should be invertible
- $\mathbf{A^T A}$ should not be very small
  - eigenvalues λ_1 and λ_2 of $\mathbf{A^T A}$ should not be very small
- $\mathbf{A^T A}$ should be well-conditioned
  - λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

Slide by Steve Seitz, UW

Edge

– gradients very large or very small

– large λ_1, small λ_2

Low-texture region

– gradients have small magnitude

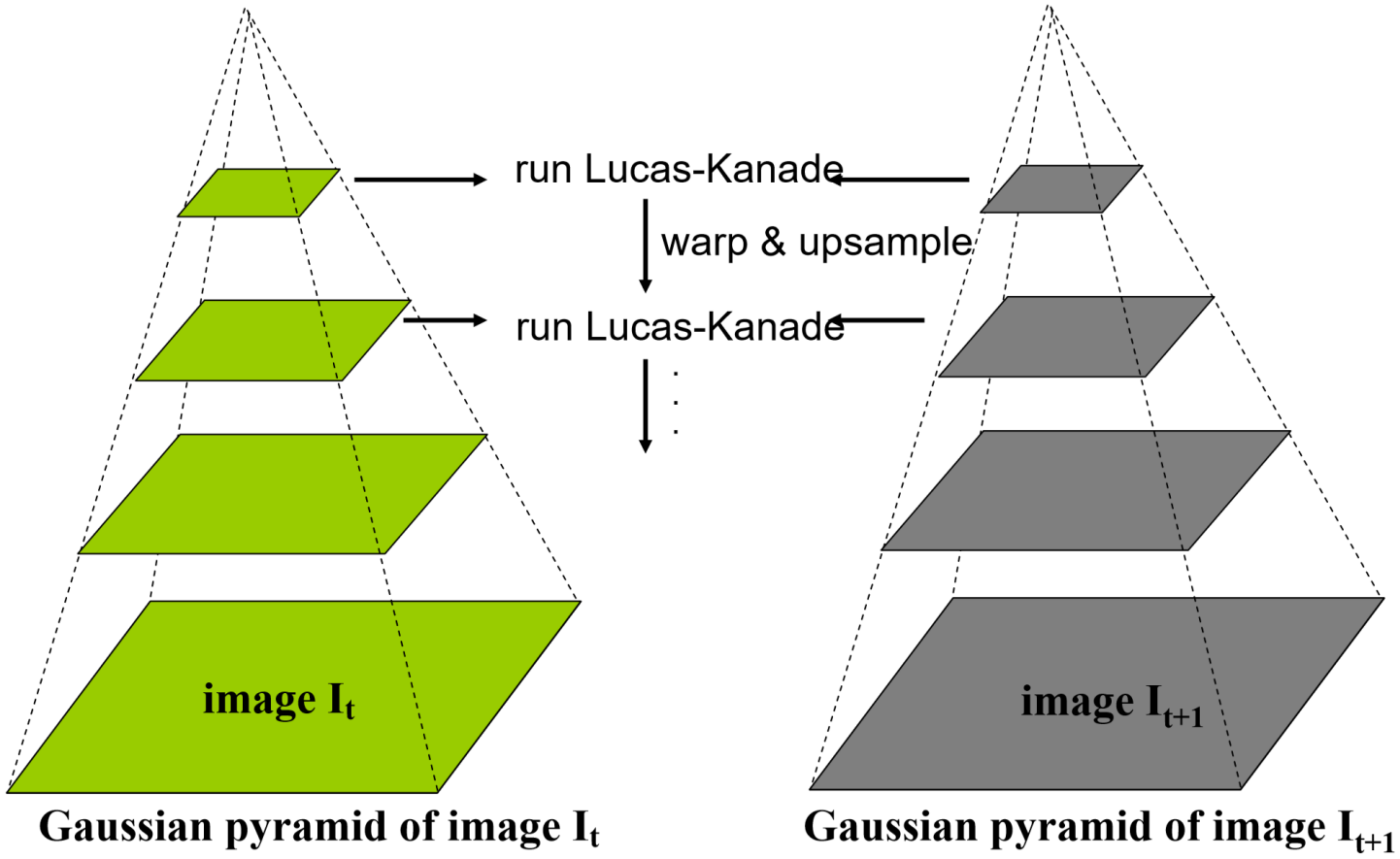
– small λ_1, small λ_2

High-texture region

– gradients are different, large magnitudes

– large λ_1, large λ_2

Multi-Scale Pyramid Estimation



Horn–Schunck Optical Flow Algorithm (1981)

- Regularisation: use a global smoothness term

Smoothness error: $$E_s = \iint_D \left(u_x^2 + u_y^2\right) + \left(v_x^2 + v_y^2\right) dx\, dy$$

Error in brightness constancy equation: $$E_c = \iint_D \left(I_x u + I_y v + I_t\right)^2 dx\, dy$$

Minimize: $E_c + \lambda E_s$

Solve by calculus of variations

Variational Minimization

The Euler-Lagrange equations :

$$F_u - \frac{\partial}{\partial x} F_{u_x} - \frac{\partial}{\partial y} F_{u_y} = 0$$

$$F_v - \frac{\partial}{\partial x} F_{v_x} - \frac{\partial}{\partial y} F_{v_y} = 0$$

In our case ,

$$F = (u_x^2 + u_y^2) + (v_x^2 + v_y^2) + \lambda (I_x u + I_y v + I_t)^2,$$

so the Euler-Lagrange equations are

$$\Delta u = \lambda (I_x u + I_y v + I_t) I_x,$$

$$\Delta v = \lambda (I_x u + I_y v + I_t) I_y,$$

$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ is the Laplacian operator

Horn–Schunck Optical Flow Algorithm

1. Coupled PDEs solved using iterative methods and finite differences

$$\frac{\partial u}{\partial t} = \Delta u - \lambda(I_x u + I_y v + I_t)I_x,$$

$$\frac{\partial v}{\partial t} = \Delta v - \lambda(I_x u + I_y v + I_t)I_y,$$

2. More than two frames allow a better estimation of I_t

3. Information spreads from corner-type patterns

Optical Flow: Summary, readings

- Optical flow definition

- Lucas–Kanade optical flow algorithm (KLT tracker)

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

- Horn–Schunck optical flow algorithm

B.K.P. Horn and B.G. Schunck, "Determining optical flow." *Artificial Intelligence, 1981*

- Computer Vision: Algorithms and Applications (2nd edition), Chapter 9.3

ENGN6528 Shape From Shading, Photometric Stereo

Outline

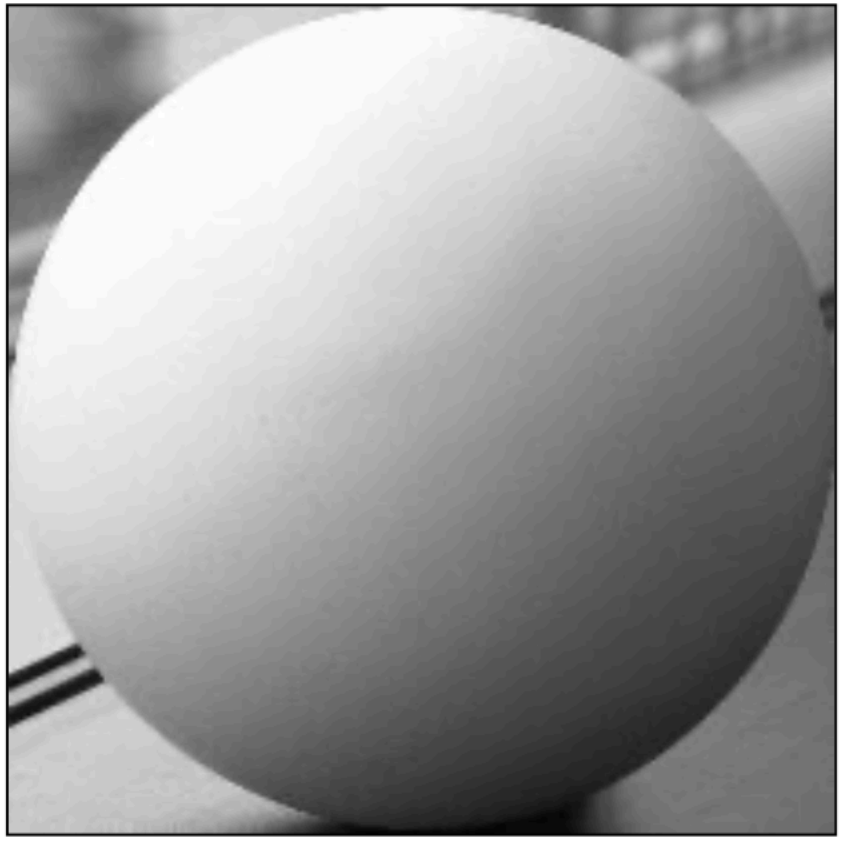
- Shape from Shading
- Photometric Stereo

Shape from Shading

Discussion:

W*hat visual cues can be used to infer 3D depth information?*

What is Shading?



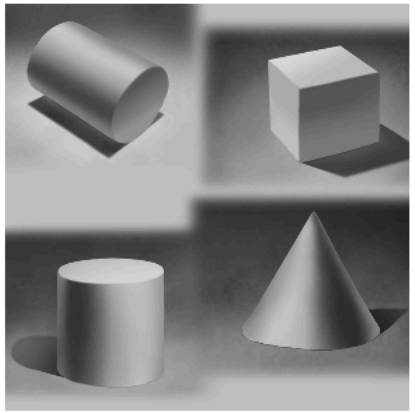
What is shading?

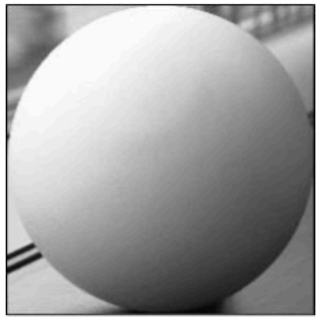
- Well… shading is not shadow…

- We can’t reconstruct shape from one shadow …

What is shading?

- Shading is the smooth (non-shiny) variation of **intensity** with surface normal.
- Shading tries to approximate local behaviour of light on the object's surface .

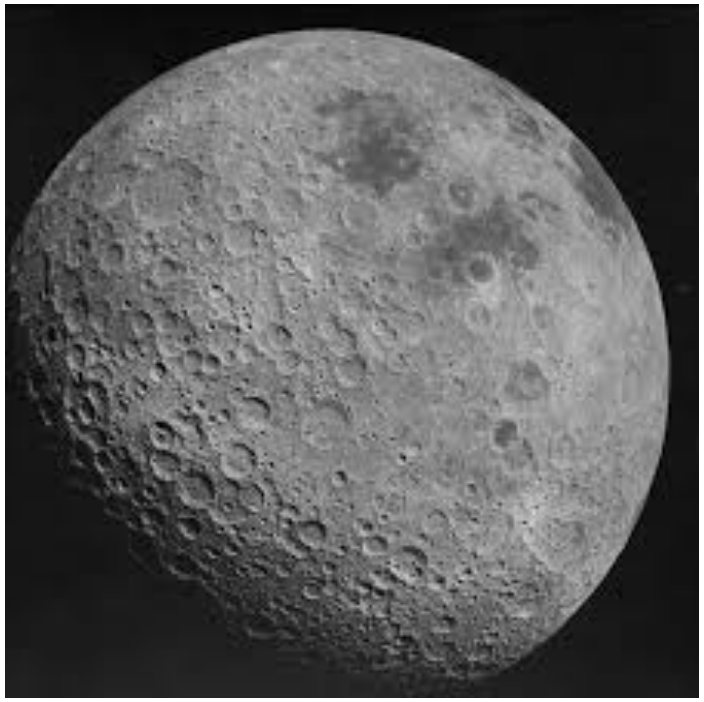




Represent 3D shape by shadings

Infer 3D geometry from Shading

- Shading gives a cue for the actual 3D shape
- There is a relation between intensity (shading) and 3D shape



Review: Photometric Image formation Process

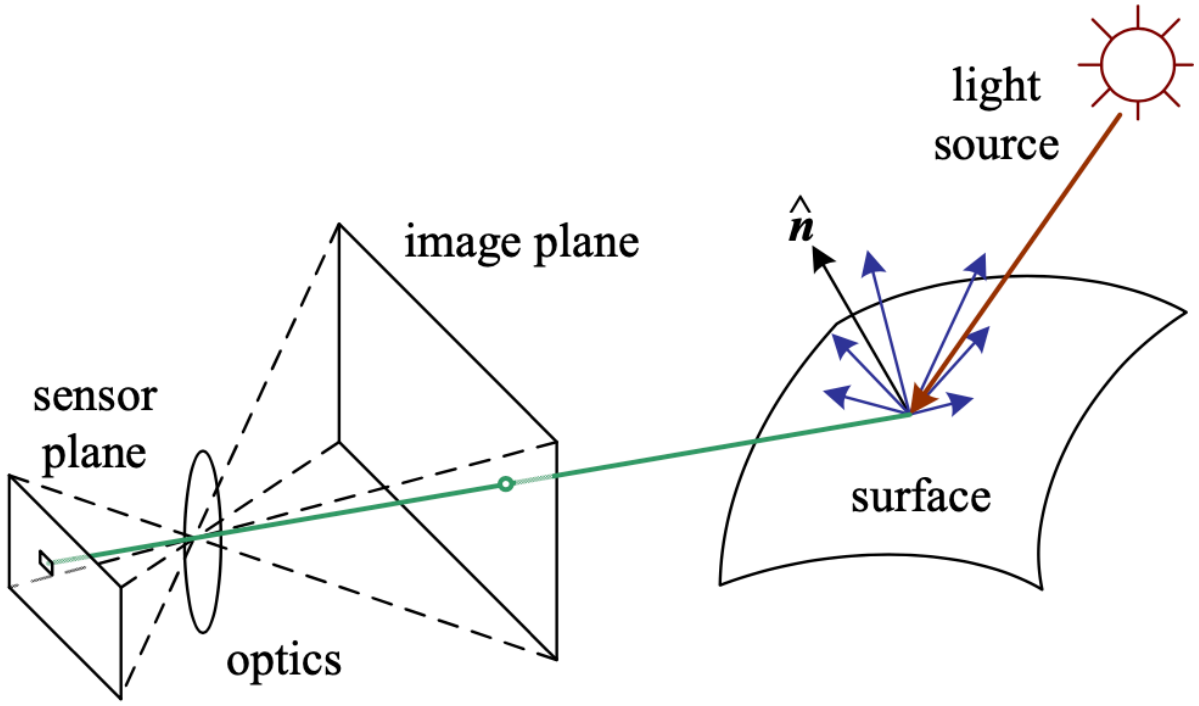
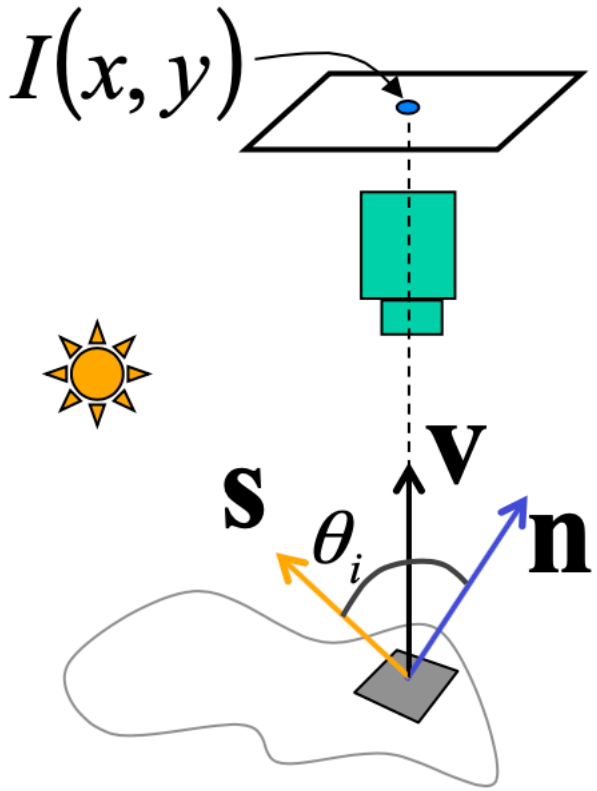


Figure 2.14 A simplified model of photometric image formation. Light is emitted by one or more light sources and is then reflected from an object's surface. A portion of this light is directed towards the camera. This simplified model ignores multiple reflections, which often occur in real-world scenes.

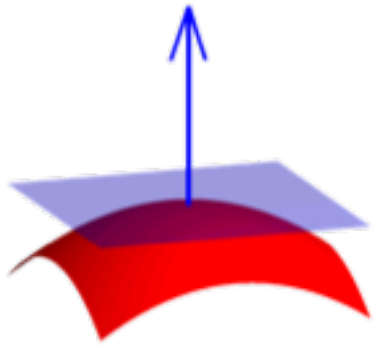
What determines the intensity of an image?

- The amount of light that falls on the surface
- The fraction of light that is reflected (**albedo**)
- Geometry of light reflection
  - Shape of surface
  - Viewpoint

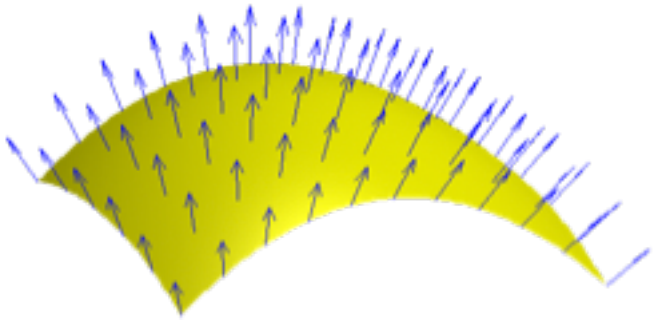


3D Surface Geometry

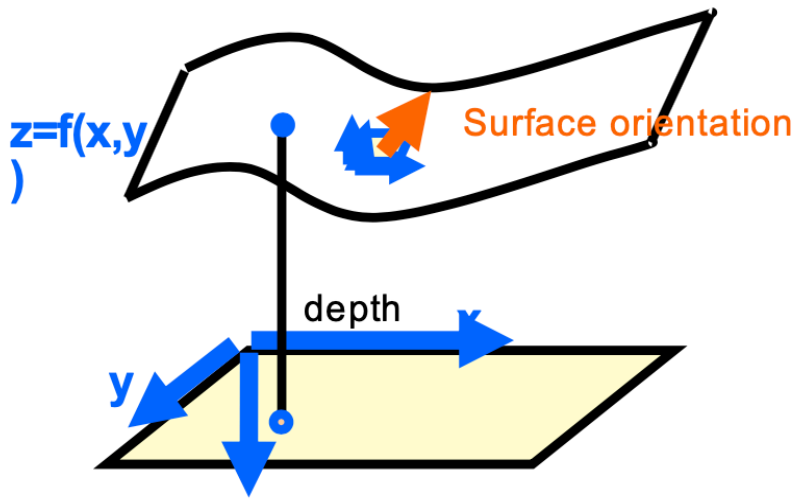
- 3D surface geometry is defined by surface normal at every point.
- A smooth surface has a tangent plane at every point.

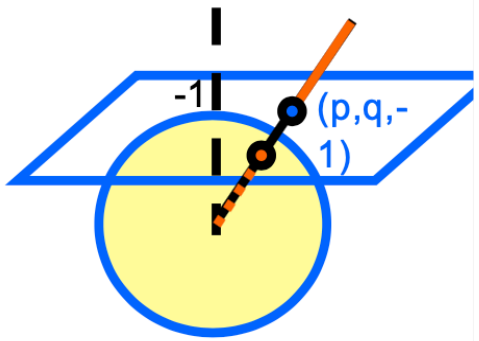


- We can model the surface using the normal at every point



depth-map and surface normal





Surface $s(x,y) = (x, y, f(x,y))$

Tangent plane $\frac{\partial s}{\partial x} = \left(1, 0, \frac{\partial f}{\partial x}\right)^T \quad \frac{\partial s}{\partial y} = \left(0, 1, \frac{\partial f}{\partial y}\right)^T$

Normal vector $\mathbf{n} = \frac{\partial s}{\partial x} \times \frac{\partial s}{\partial y} = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, -1\right)^T$

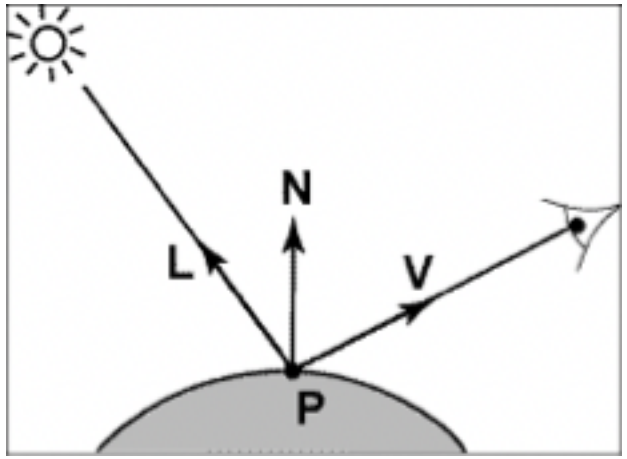
Gradient space

$$p = \frac{\partial f}{\partial x} \quad q = \frac{\partial f}{\partial y}$$

$$\mathbf{n} = (p, q, -1)$$

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

Modelling photometric Image formation

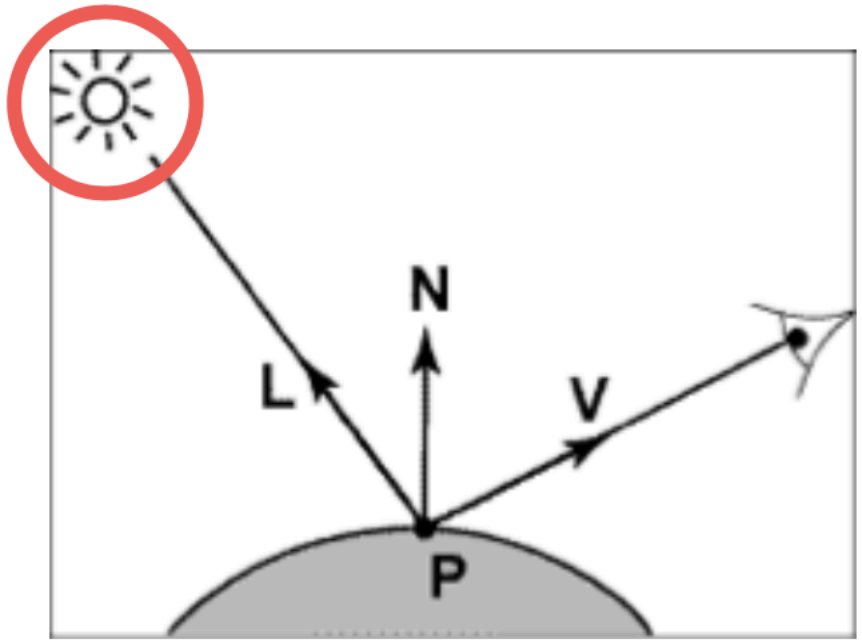


Now we need to reason about:

- How light interacts with the scene
- How a pixel value is related to light energy in the world

- Track a ray of light all the way from light source to the image CMOS/CCD sensor

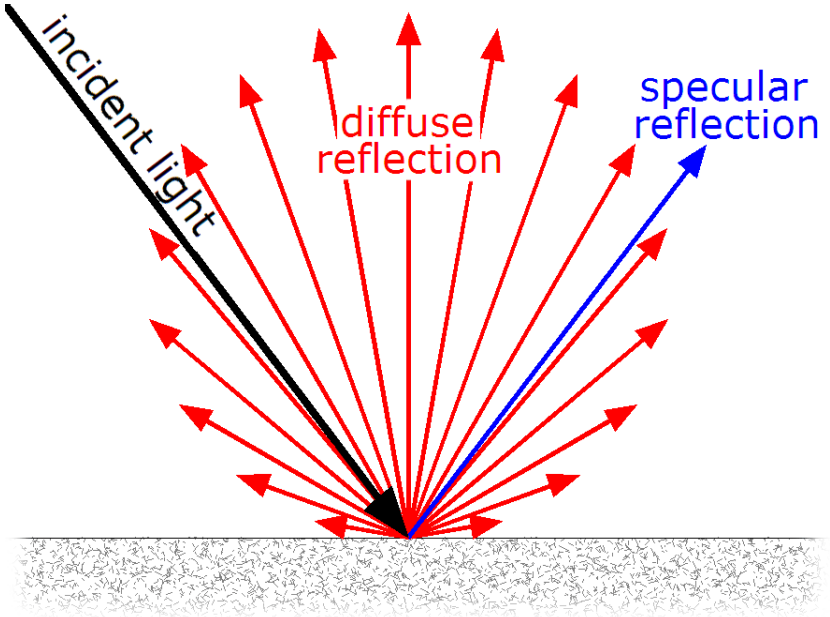
Assume far-field directional lighting



- Key property: all rays are parallel
- Equivalent to an infinitely distant point source

An ideal model: Lambertian Surface

- Appear equally bright from all viewing directions
- Reflects all light without absorbing
- Matte surface, no “shiny” spots
- Brightness of the surface as seen from camera is linearly correlated to the amount of light falling on the surface

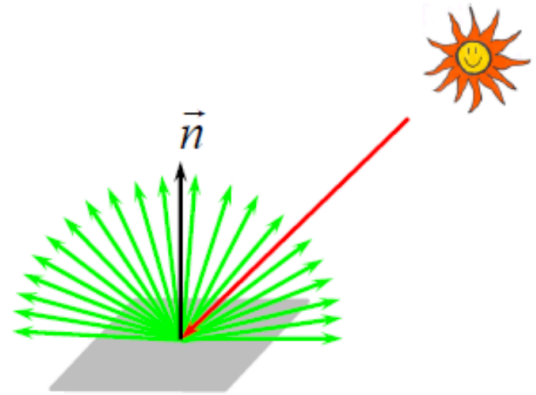


Lambertian surface

- Lambertian (perfectly diffuse) surface

$$f_L(\phi_i, \theta_i; \phi_r, \theta_r) = const = \bar{f} = \rho \frac{1}{\pi}$$

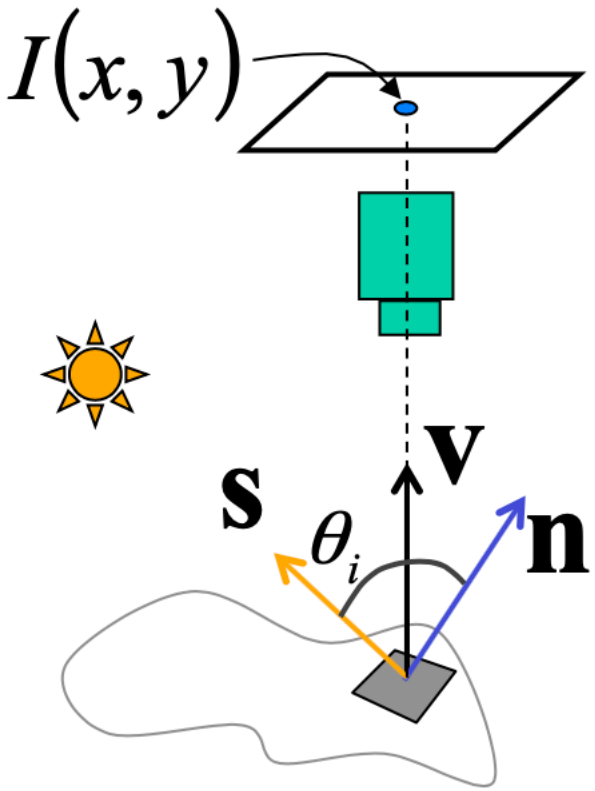
Albedo



Bidirectional reflectance distribution function (**BRDF**) $\boldsymbol{f_L(\varphi_i, \theta_i; \varphi_r, \theta_r)}$**:** defines how light is reflected at an opaque surface.

Albedo is a measure of how much light that hits a surface and is reflected without being absorbed.

Image formation for Lambertian Surface



- Relate image irradiance I(x,y) to surface orientation

(p,q,-1) for a given light source direction (p_s,q_s,-1) and surface reflectance.

- Lambertian case:

k : light source brightness
ρ : surface albedo (reflectance)

c : constant (optical system)

- Image irradiance:

$$I = \frac{\rho}{\pi} kc \cos\theta_i = \frac{\rho}{\pi} kc\mathbf{n} \cdot \mathbf{s}$$

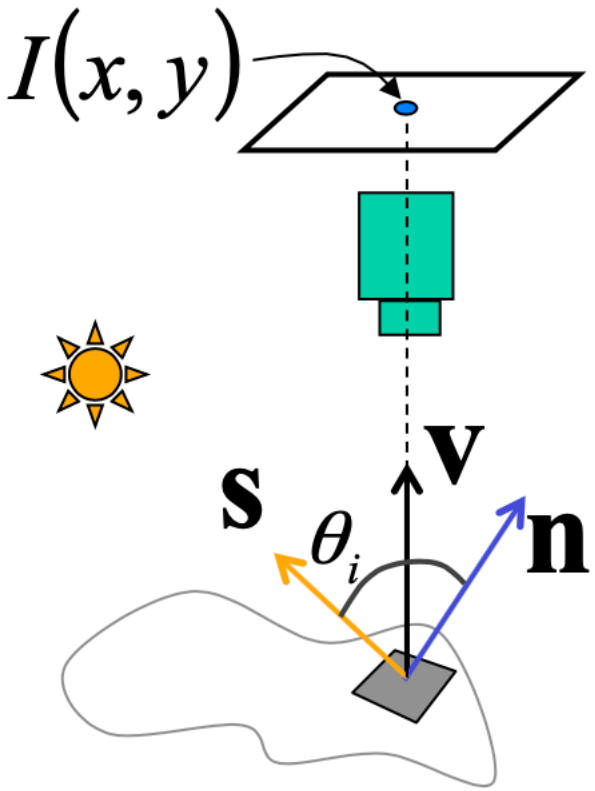
If $\frac{\rho}{\pi} kc = 1, I = n \cdot s = cos\theta_i$

n: surface normal

s: light direction

通过光源信息计算表面反射强度

Image formation for Lambertian Surface



- Relate image irradiance I(x,y) to surface orientation

(p,q,-1) for a given light source direction (p_s,q_s,-1) and surface reflectance.

- Lambertian case:

k : light source brightness
ρ : surface albedo (reflectance)

c : constant (optical system)

- Image irradiance:

$$I = \frac{\rho}{\pi} kc \cos\theta_i = \frac{\rho}{\pi} kc \mathbf{n} \cdot \mathbf{s}$$

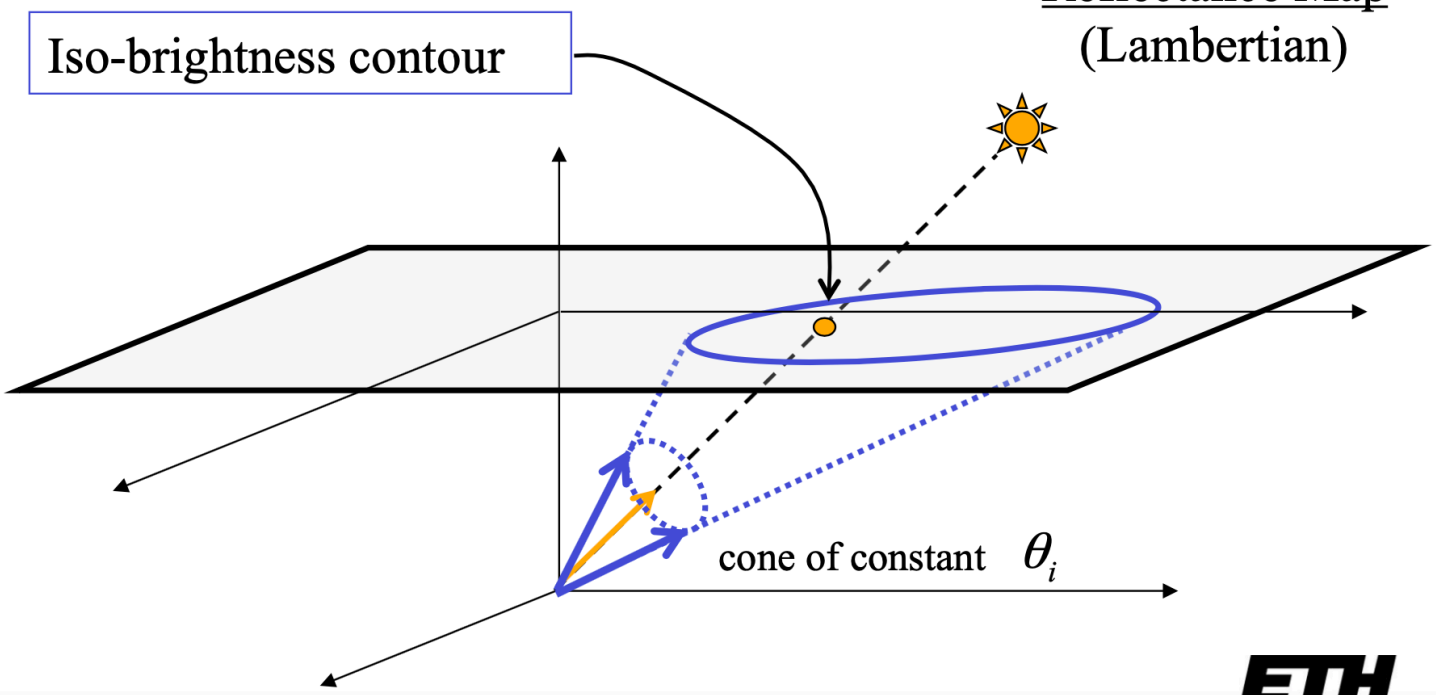
If $\frac{\rho}{\pi} kc = 1, I = n \cdot s = cos\theta_i$

Lambertian case

$$I = \cos\theta_i = \mathbf{n}\cdot\mathbf{s} = \frac{(pp_s + qq_s + 1)}{\sqrt{p^2 + q^2 + 1}\sqrt{p_S^2 + q_S^2 + 1}} = R(p,q)$$

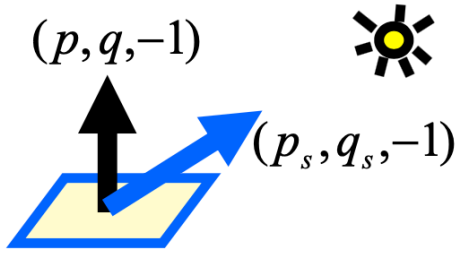
Reflectance Map
(Lambertian)

$$\mathbf{n} = \begin{pmatrix} p \\ q \\ -1 \end{pmatrix} \quad \mathbf{s} = \begin{pmatrix} p_s \\ q_s \\ -1 \end{pmatrix}$$



ETH

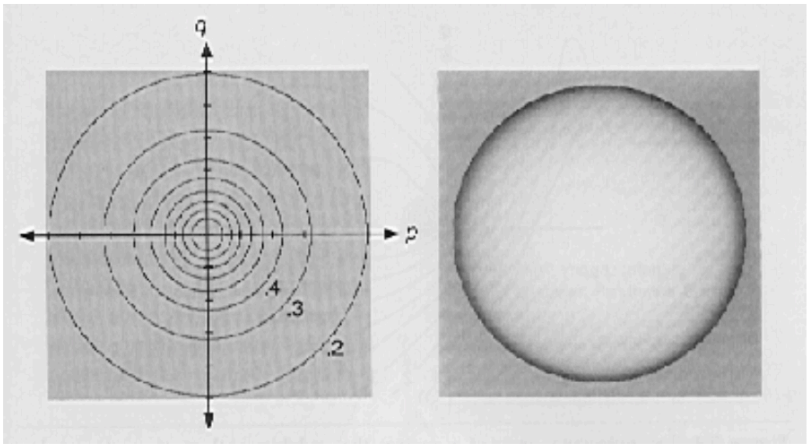
Lambertian Reflectance map



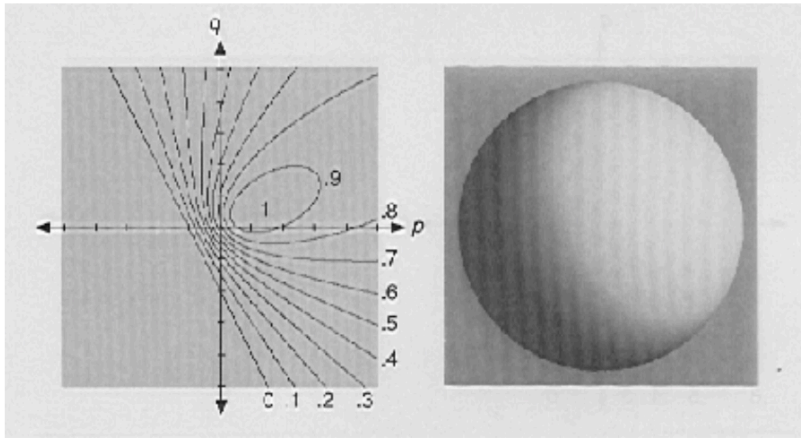
$$R(p,q) = \frac{1 + p_s p + q_s q}{\sqrt{1 + p^2 + q^2}\sqrt{1 + {p_s}^2 + {q_s}^2}}$$

$$\mathbf{n} = \begin{pmatrix} p \\ q \\ -1 \end{pmatrix} \quad \mathbf{s} = \begin{pmatrix} p_s \\ q_s \\ -1 \end{pmatrix}$$

Local surface orientation that produces equivalent intensities are quadratic conic sections contours in gradient space



$p_s=0, q_s=0$



$p_s=-2, q_s=-1$

Lambertian reflectance map

- Relationship between surface orientation and brightness

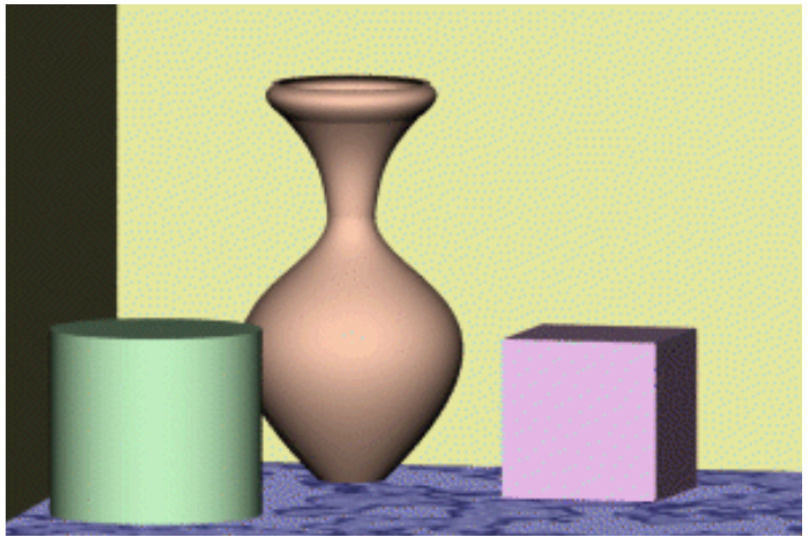
$$R(p,q) = \max(0, \hat{n} \cdot \hat{l})$$

where

$$\hat{n} \cdot \hat{l} = \frac{1 + p_s p + q_s q}{\sqrt{1 + p^2 + q^2}\sqrt{1 + p_s^2 + q_s^2}}$$

- The image irradiance (brightness) is proportional to R

Computer graphics (forward) rendered Lambertian surfaces



Scene

(Oren and Nayar)

Lambertian sphere as the light moves.

(Steve Seitz)

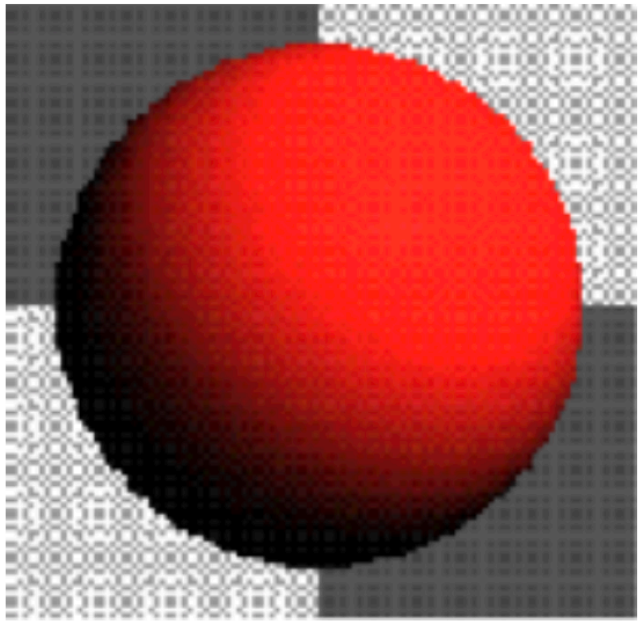
Computer Graphics: **Forward rendering**

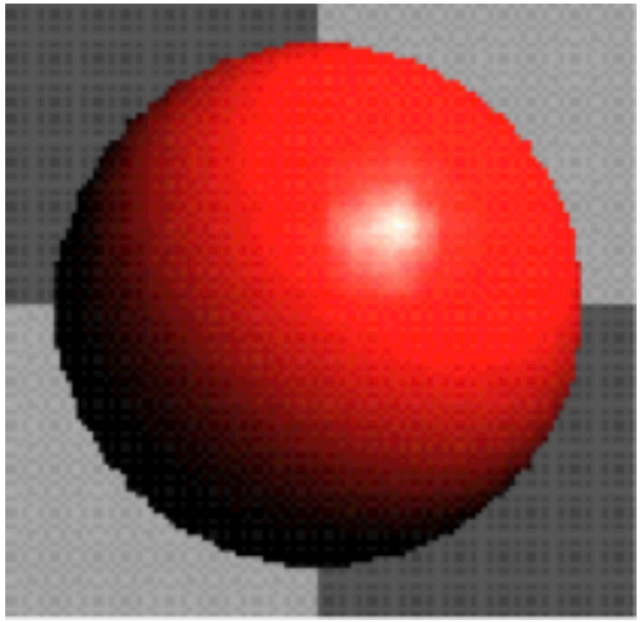
- Given a 3D surface $z(x, y)$, lighting and viewing direction, we can compute the gray scale level of the pixel I(x,y) of the surface
- Find the gradient of the surface (p, q), surface normal becomes

(p, q, -1)

- Use $I(x, y) = R(p, q) = \frac{pp_s + qq_s + 1}{\sqrt{p^2 + q^2 + 1}\sqrt{p_s^2 + q_s^2 + 1}}$ to determine the gray level

More realistic case: Lambertian+Specular

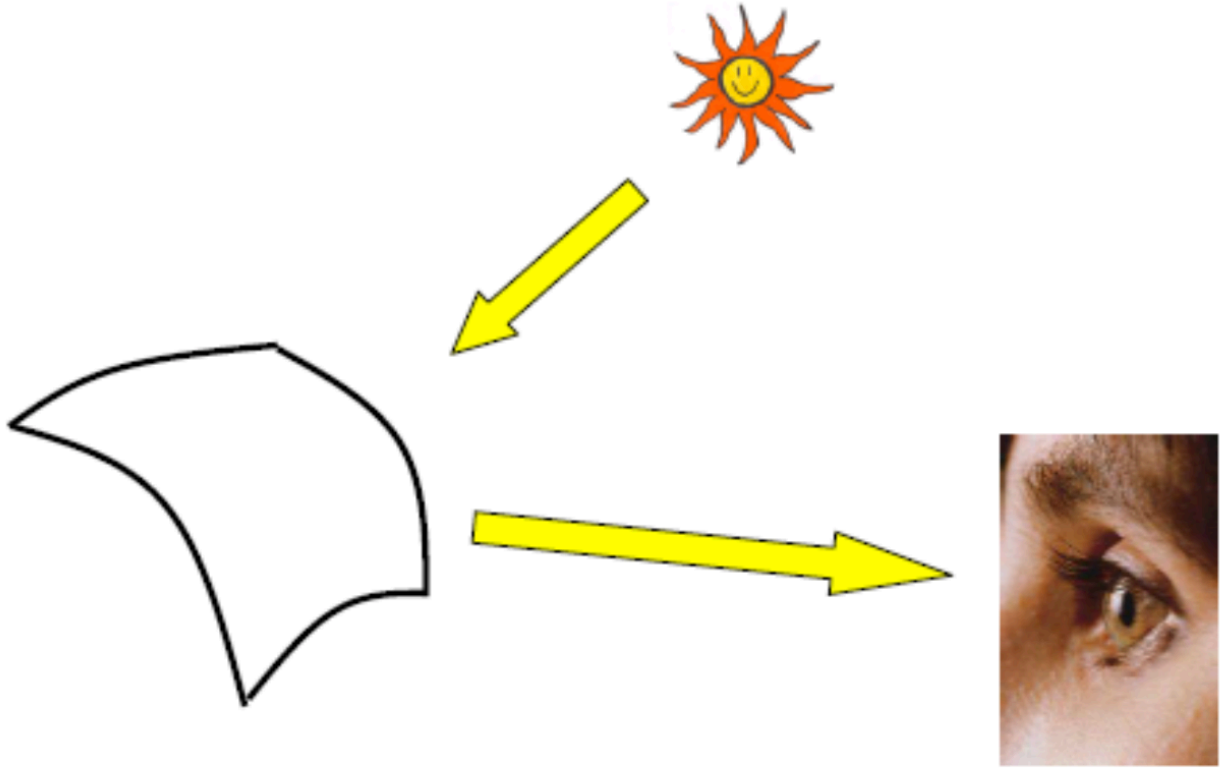




(http://graphics.cs.ucdavis.edu/GraphicsNotes/Shading/Shading.html)

‘Shading from shape’ Vs ‘Shape from Shading’

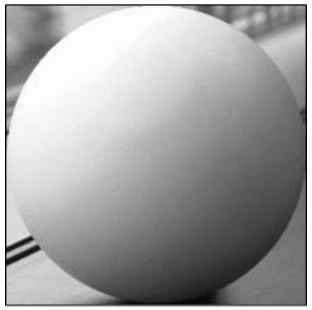
- Image formation = ‘shading from shape’ (and light source)



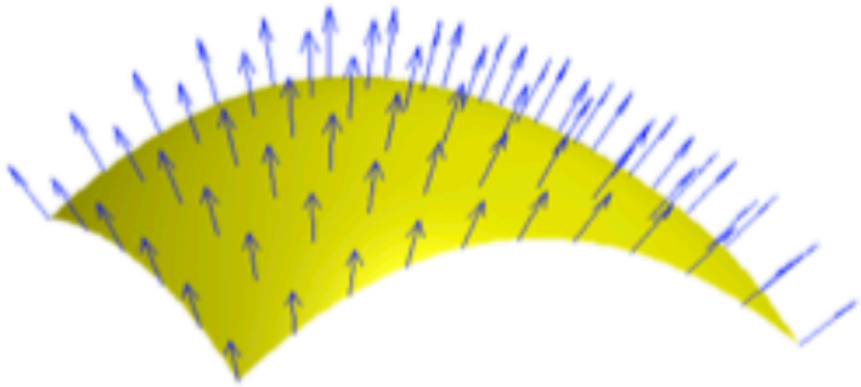
- Invert the image formation process

Shape from Shading: Inverse Rendering

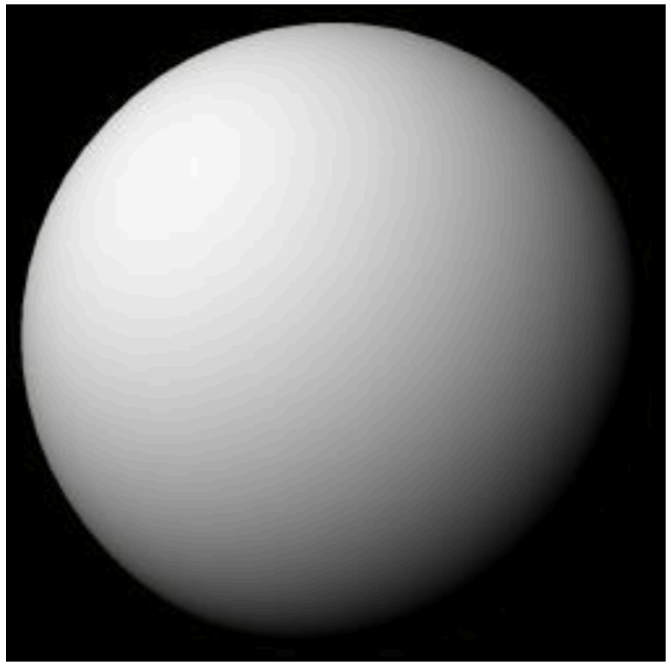
- Given a grayscale image, and albedo, point light source direction



- Shape from shading is to reconstruct scene geometry which can be modeled by *surface normal*.



Solve ’Shape from Shading’



$$I = k_d \mathbf{N} \cdot \mathbf{L}$$

Assume k_d is 1 for now.

What can we measure from one image?

- $\cos^{-1}(I)$ is the angle between N and L

Solve ’Shape from Shading’

- We have

$$R(p,q) = \frac{1 + p_s p + q_s q}{\sqrt{1 + p^2 + q^2}\sqrt{1 + {p_s}^2 + {q_s}^2}}$$

- Discretize: end up with one nonlinear PDE equation per pixel
- For N pixels, → *N* equations in 2*N* unknowns…

Solve ‘Shape from Shading’

- Solution: add smoothness constraint; add initial boundary conditions.
- Form an energy minimization problem:
- Given observed E(x,y), find {(p,q)} that minimize energy

$$\mathcal{E} = \int \left(\left(I(x,y) - R(p,q) \right)^2 + \lambda \left(p_x^{\,2} + p_y^{\,2} + q_x^{\,2} + q_y^{\,2} \right) \right) dx\, dy$$

- Regularization: minimize combination of difference between observed image and ‘rendering by R(p,q)’, surface curvature

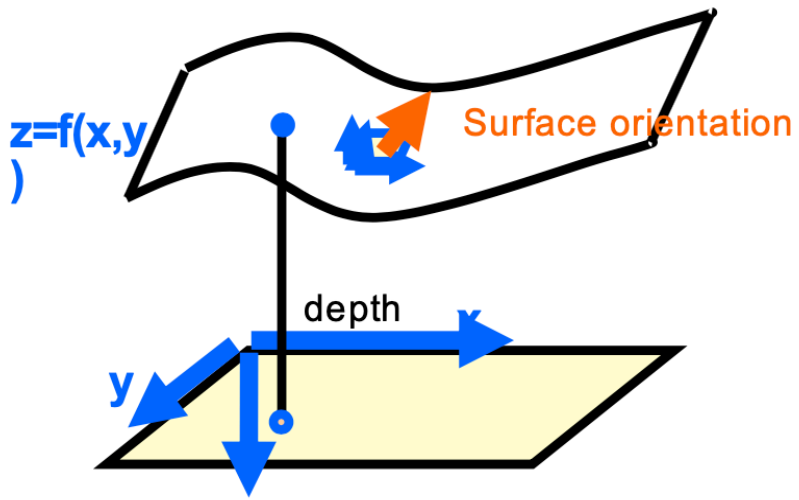
Solve ‘Shape from Shading’

- Solve by techniques from calculus of variations
- Use Euler-Lagrange equations to get a PDE, solve numerically

$$\frac{\partial E}{\partial p} - \frac{d}{dx}\frac{\partial E}{\partial p_x} - \frac{d}{dy}\frac{\partial E}{\partial p_y} = 0$$

$$\frac{\partial E}{\partial q} - \frac{d}{dx}\frac{\partial E}{\partial q_x} - \frac{d}{dy}\frac{\partial E}{\partial q_y} = 0$$

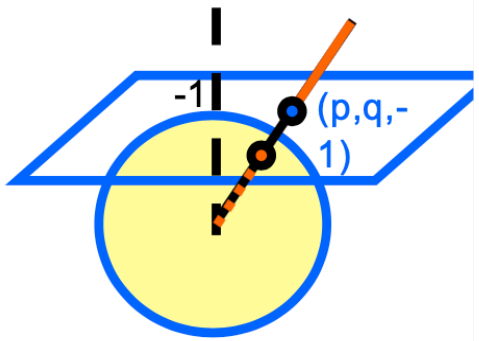
How to recover depth-map from surface normal?



Surface $s(x,y) = (x, y, f(x,y))$

Tangent plane $\frac{\partial s}{\partial x} = \left(1, 0, \frac{\partial f}{\partial x}\right)^T \quad \frac{\partial s}{\partial y} = \left(0, 1, \frac{\partial f}{\partial y}\right)^T$

Normal vector $\mathbf{n} = \frac{\partial s}{\partial x} \times \frac{\partial s}{\partial y} = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, -1\right)^T$



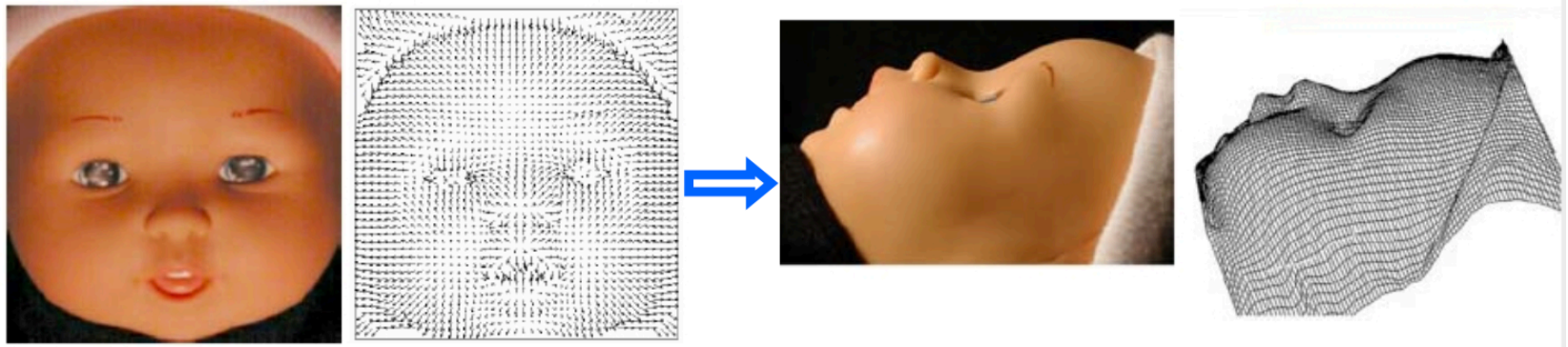
Gradient space

$$p = \frac{\partial f}{\partial x} \quad q = \frac{\partial f}{\partial y}$$

$$\mathbf{n} = (p, q, -1)$$

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

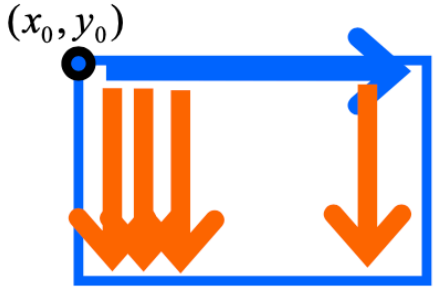
Depth from Surface normal by 2D integration



[D. Kriegman]

Integrate normal (gradients p,q) across the image
Simple approach – integrate along a curve from

(x_0, y_0)

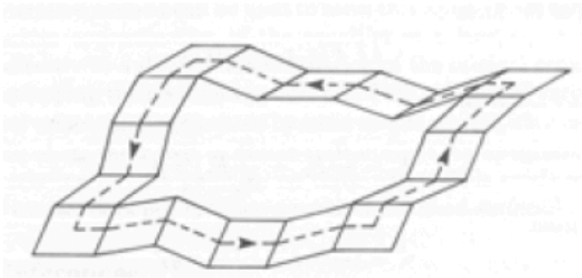


1. From $\mathbf{n} = (n_x, n_y, n_z)$ $p = n_x / n_z$ $q = n_y / n_z$
2. Integrate $p = \partial f / \partial x$ along $(x,0)$ to get $f(x,0)$
3. Integrate $q = \partial f / \partial y$ along each column

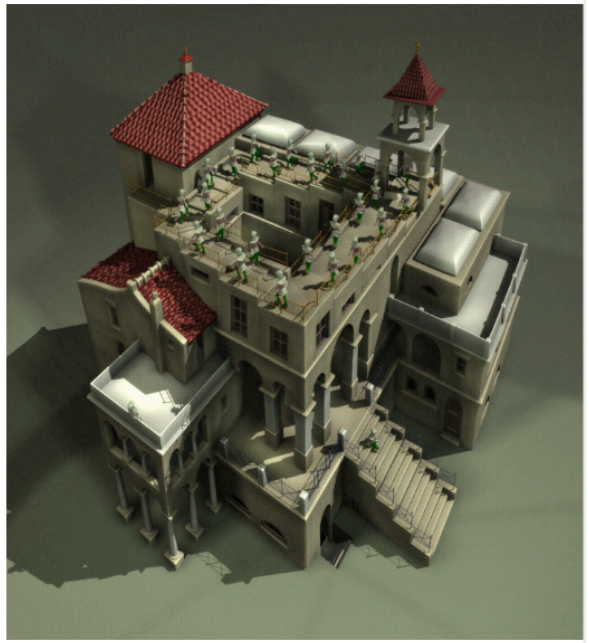
$$f(x,y) = f(x_0,y_0) + \int_{(x_0,y_0)}^{(x,y)} (pdx + qdy)$$

Enforce Integrability

$$f(x,y) = f(x_0,y_0) + \int_{(x_0,y_0)}^{(x,y)} (pdx + qdy)$$



Integrate along a curve from (x_0, y_0)



[Escher] no integrability

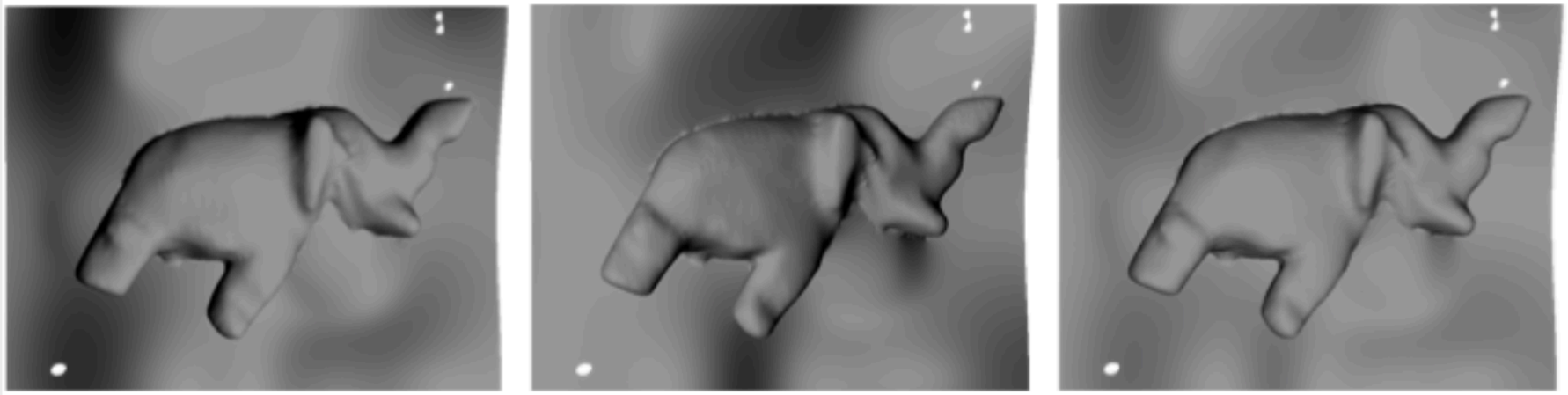
Impose integrability

A normal map that produces a unique depth map is called integrable

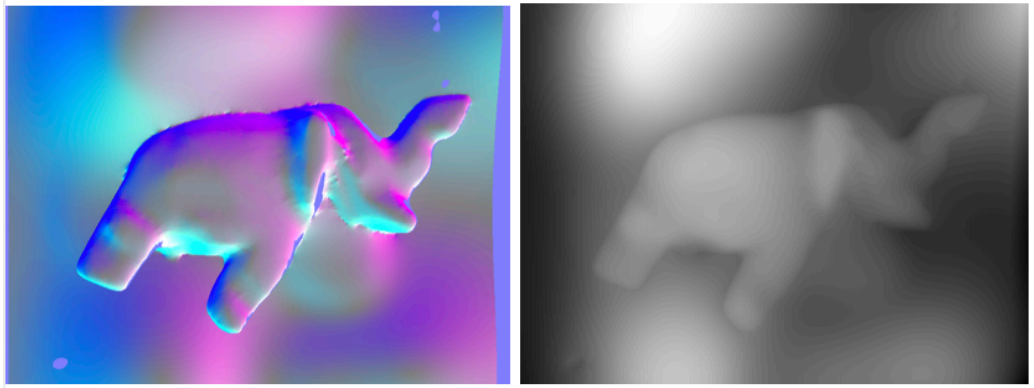
Enforced by $\frac{\partial p}{\partial y} = \frac{\partial q}{\partial x}; \quad \frac{\partial}{\partial y}\frac{\partial f}{\partial x} = \frac{\partial}{\partial x}\frac{\partial f}{\partial y}$

Sample results

[Neil Birkbeck]

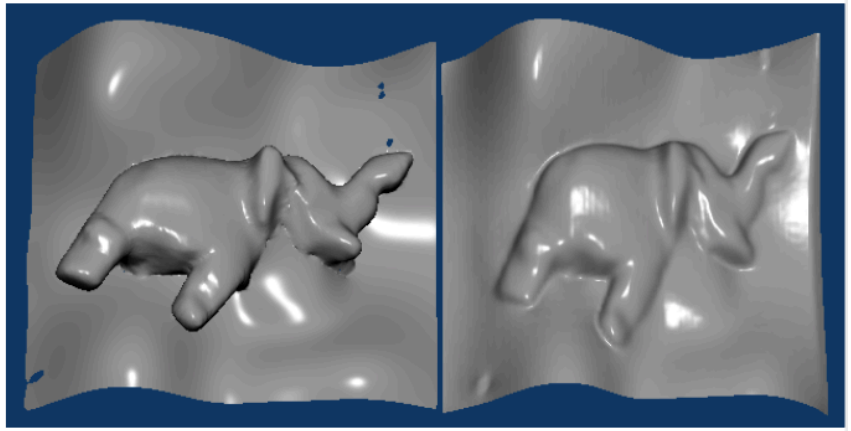


images with different light



normals

Integrated depth

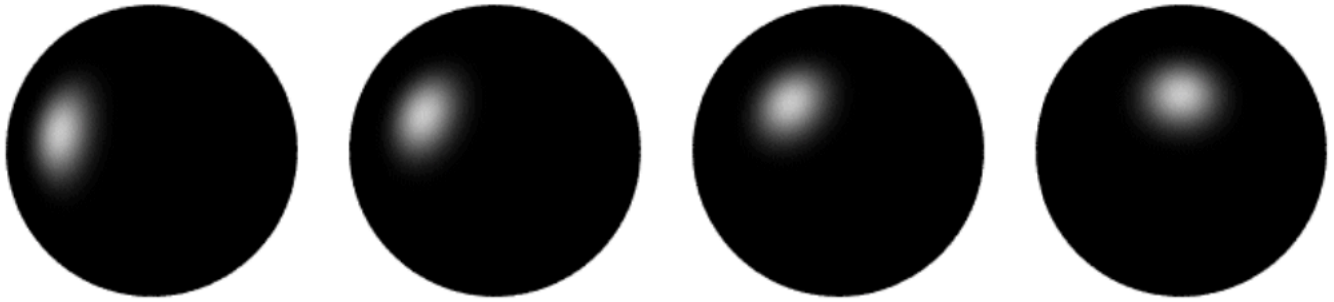


original surface

reconstructed

How to determine the light source?

- Trick: Place a mirror ball in the scene.

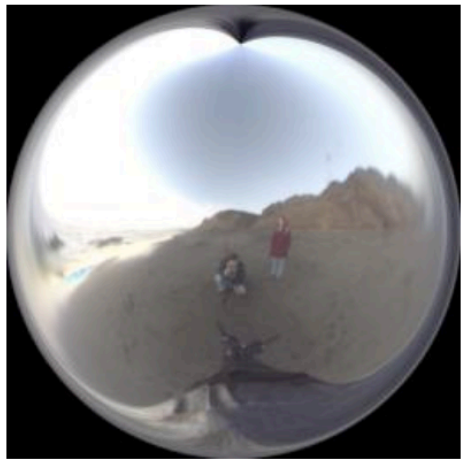


- The location of the highlight is determined by the light source direction.


Canon
Canon

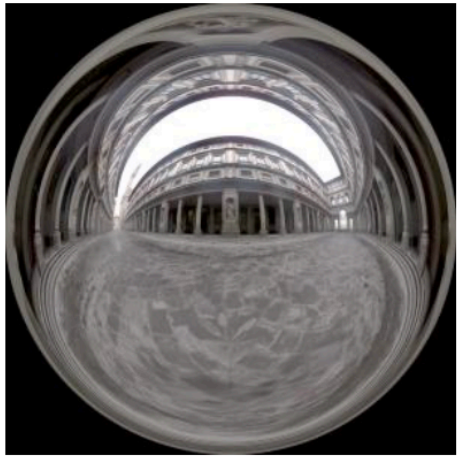
Real world lighting environment

Funston Beach

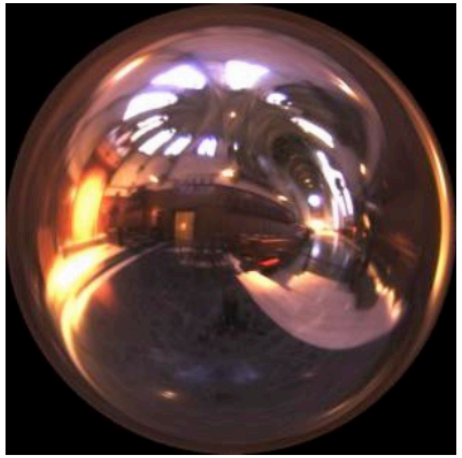


Eucalyptus Grove

Uffizi Gallery



Grace Cathedral



Lighting Environments from the Light Probe Image Gallery:
http://www.debevec.org/Probes/

Example: Shape from Shading

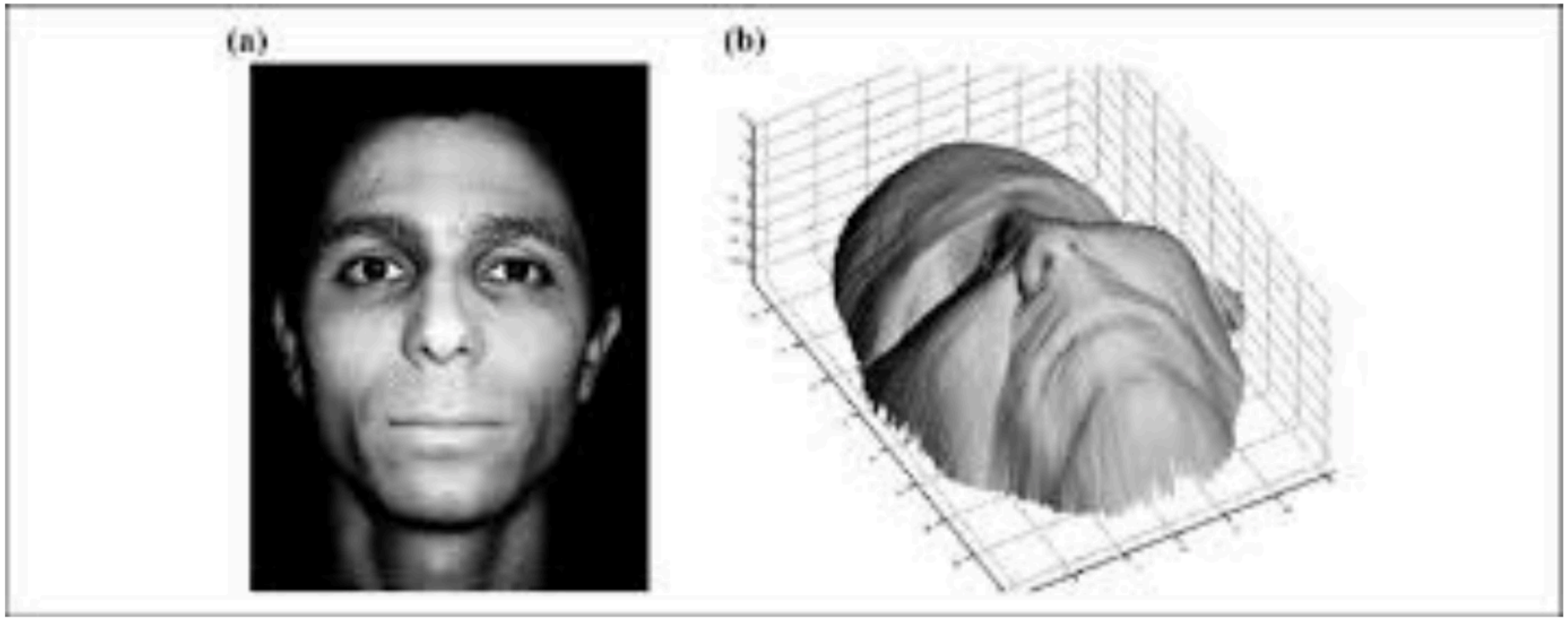


Image (Shading)

Reconstructed Surface

Readings

- Chapter 13.1.1 Computer Vision: Algorithms and Applications (2nd Edition)