Linear Models in Motion and Lighting

Linear Models

- Structure-from-motion with affine projection
  - Scaled orthographic projection, ignoring some non-linear constraints (we can add those later if we want).
  - Simple, linear solution.
- Photometric Stereo
  - Lambertian reflectance, no shadows
  - Everything is linear

The Equation of Weak Perspective (scaled orthographic projection)

$$(x, y, z) \rightarrow s(x, y)$$

- Approximate by assuming depth constant.
- So *s* is constant for all points.
- Parallel lines no longer converge, they remain parallel.

First: Represent motion

- We'll talk about a fixed camera, and moving object.
- Key point:

Points

$$P = \begin{pmatrix} x_1 & x_2 & & & & x_n \\ y_1 & y_2 & . & . & . & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

Some matrix

$$S = \begin{pmatrix} s_{1,1} & s_{1,2} & s_{1,3} & t_x \\ s_{2,1} & s_{2,2} & s_{2,3} & t_y \end{pmatrix}$$

The image

$$I = \begin{pmatrix} u_1 & u_2 & . & . & . & u_n \\ v_1 & v_2 & & & & v_n \end{pmatrix}$$

Then: $$I = SP$$

Remember what this means.

- We are representing moving a set of points, projecting them into the image, and scaling them.
- Matrix multiplication: take inner product between each row of S and each point. First row of S produces X coordinates, while second row produces Y.
- Projection occurs because S has no third row.
- Translation occurs with tx and ty.
- Scaling can be encoded with a scale factor in S.
- The rest of S must be allowing the object to rotate.

Examples:

- S = [s, 0, 0, 0; 0, s, 0, 0]; This is just projection, with scaling by s.
- S = [s, 0, 0, s*tx; 0, s, 0, s*ty]; This is translation by (tx,ty,something), projection, and scaling.

Structure-from-Motion

- S encodes:
  - Projection: only two lines
  - Scaling, since S can have a scale factor.
  - Translation, by tx/s and ty/s.
  - Rotation:

$$I = SP$$

Rotation

$$\begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{pmatrix} P$$

Represents a 3D rotation of the points in P.

First, recall 2D rotation (easier)

Matrix R acts on points by rotating them.

$$R = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$$

$$\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}\begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \end{pmatrix}$$

- Also, RR^T = Identity. R^T is also a rotation matrix, in the opposite direction to R.

Why does multiplying points by R rotate them?

- Think of the rows of R as a new coordinate system. Taking inner products of each points with these expresses that point in that coordinate system.
  - This means rows of R must be orthonormal vectors (orthogonal unit vectors).
- Think of what happens to the points (1,0) and (0,1). They go to (cos theta, -sin theta), and (sin theta, cos theta). They remain orthonormal, and rotate clockwise by theta.
  - Any other point, (a,b) can be thought of as a(1,0) + b(0,1). R(a(1,0)+b(0,1) = Ra(1,0) + Ra(0,1) = aR(1,0) + bR(0,1). So it's in the same position relative to the rotated coordinates that it was in before rotation relative to the x, y coordinates. That is, it's rotated.

Simple 3D Rotation

$$\begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \end{pmatrix}$$

Rotation about z axis.

Rotates x,y coordinates. Leaves z coordinates fixed.

Full 3D Rotation

$$R = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{pmatrix}$$

- Any rotation can be expressed as combination of three rotations about three axes.

$$RR^T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Rows (and columns) of *R* are orthonormal vectors.
- R has determinant 1 (not -1).

- Intuitively, it makes sense that 3D rotations can be expressed as 3 separate rotations about fixed axes. Rotations have 3 degrees of freedom; two describe an axis of rotation, and one the amount.
- Rotations preserve the length of a vector, and the angle between two vectors. Therefore, (1,0,0), (0,1,0), (0,0,1) must be orthonormal after rotation. After rotation, they are the three columns of R. So these columns must be orthonormal vectors for R to be a rotation. Similarly, if they are orthonormal vectors (with determinant 1) R will have the effect of rotating (1,0,0), (0,1,0), (0,0,1). Same reasoning as 2D tells us all other points rotate too.
  - Note if R has determinant -1, then R is a rotation plus a reflection.

Putting it Together

Scale

3D Translation

Projection

$$s\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{pmatrix}\begin{pmatrix}1 & 0 & 0 & t_x\\ 0 & 1 & 0 & t_y\\ 0 & 0 & 1 & t_z\end{pmatrix}\begin{pmatrix}r_{1,1} & r_{1,2} & r_{1,3} & 0\\ r_{2,1} & r_{2,2} & r_{2,3} & 0\\ r_{3,1} & r_{3,2} & r_{3,3} & 0\\ 0 & 0 & 0 & 1\end{pmatrix}P$$

3D Rotation

$$\equiv\begin{pmatrix}s_{1,1} & s_{1,2} & s_{1,3} & st_x\\ s_{2,1} & s_{2,2} & s_{2,3} & st_y\end{pmatrix}P$$

where

$$(s_{1,1}, s_{1,2}, s_{1,3}) \bullet (s_{2,1}, s_{2,2}, s_{2,3}) = 0$$

$$\|(s_{1,1}, s_{1,2}, s_{1,3})\| = \|(s_{2,1}, s_{2,2}, s_{2,3})\|$$

We can just write st_x as t_x and st_y as t_y.

Affine Structure from Motion

$$\begin{pmatrix} s_{1,1} & s_{1,2} & s_{1,3} & t_x \\ s_{2,1} & s_{2,2} & s_{2,3} & t_y \end{pmatrix} P$$

where

$$(s_{1,1}, s_{1,2}, s_{1,3}) \bullet (s_{2,1}, s_{2,2}, s_{2,3}) = 0$$

$$\|(s_{1,1}, s_{1,2}, s_{1,3})\| = \|(s_{2,1}, s_{2,2}, s_{2,3})\|$$

Affine Structure-from-Motion: Two Frames (1)

$$\begin{pmatrix} u_1^1 & u_2^1 & . & . & . & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

Affine Structure-from-Motion: Two Frames (2)

To make things easy, suppose:

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Affine Structure-from-Motion: Two Frames (3)

$$\begin{pmatrix} u_1^1 & u_2^1 & . & . & . & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

Looking at the first four points, we get:

$$\begin{pmatrix} u_1^1 & u_2^1 & u_3^1 & u_4^1 \\ v_1^1 & v_2^1 & v_3^1 & v_4^1 \\ u_1^2 & u_2^2 & u_3^2 & u_4^2 \\ v_1^2 & v_2^2 & v_3^2 & v_4^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Affine Structure-from-Motion: Two Frames (4)

$$\begin{pmatrix} u_1^1 & u_2^1 & u_3^1 & u_4^1 \\ v_1^1 & v_2^1 & v_3^1 & v_4^1 \\ u_1^2 & u_2^2 & u_3^2 & u_4^2 \\ v_1^2 & v_2^2 & v_3^2 & v_4^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

We can solve for motion by inverting matrix of points.

Or, explicitly, we see that first column on left (images of first point) give the translations. After solving for these, we can solve for the each column of the *s* components of the motion using the images of each point, in turn.

Affine Structure-from-Motion: Two Frames (5)

$$\begin{pmatrix} u_k^1 \\ v_k^1 \\ u_k^2 \\ v_k^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} a_k \\ b_k \\ c_k \\ 1 \end{pmatrix}$$

Once we know the motion, we can use the images of another point to solve for the structure. We have four linear equations, with three unknowns.

Affine Structure-from-Motion: Two Frames (6)

Suppose we just know where the *k*' th point is in image 1.

$$\begin{pmatrix} u_k^1 \\ v_k^1 \\ ? \\ ? \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} a_k \\ b_k \\ c_k \\ 1 \end{pmatrix}$$

Then, we can use the first two equations to write a_k and b_k as linear in c_k. The final two equations lead to two linear equations in the missing values and c_k. If we eliminate c_k we get one linear equation in the missing values. This means the unknown point lies on a known line. That is, we recover the epipolar constraint. Furthermore, these lines are all parallel.

Affine Structure-from-Motion: Two Frames (7)

But, what if the first four points aren' t so simple?

Then we define *A* so that:

$$A\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

This is always possible as long as the points aren' t coplanar.

Affine Structure-from-Motion: Two Frames (8)

Then, given:

$$\begin{pmatrix} u^1_1 & u^1_2 & . & . & . & u^1_n \\ v^1_1 & v^1_2 & & & & v^1_n \\ u^2_1 & u^2_2 & & & & u^2_n \\ v^2_1 & v^2_2 & & & & v^2_n \end{pmatrix} = \begin{pmatrix} s^1_{1,1} & s^1_{1,2} & s^1_{1,3} & t^1_x \\ s^1_{2,1} & s^1_{2,2} & s^1_{2,3} & t^1_y \\ s^2_{1,1} & s^2_{1,2} & s^2_{1,3} & t^2_x \\ s^2_{2,1} & s^2_{2,2} & s^2_{2,3} & t^2_y \end{pmatrix} \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

We have:

$$\begin{pmatrix} u^1_1 & u^1_2 & . & . & . & u^1_n \\ v^1_1 & v^1_2 & & & & v^1_n \\ u^2_1 & u^2_2 & & & & u^2_n \\ v^2_1 & v^2_2 & & & & v^2_n \end{pmatrix} = \begin{pmatrix} s^1_{1,1} & s^1_{1,2} & s^1_{1,3} & t^1_x \\ s^1_{2,1} & s^1_{2,2} & s^1_{2,3} & t^1_y \\ s^2_{1,1} & s^2_{1,2} & s^2_{1,3} & t^2_x \\ s^2_{2,1} & s^2_{2,2} & s^2_{2,3} & t^2_y \end{pmatrix} A^{-1} A \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

And:

$$\begin{pmatrix} u^1_1 & u^1_2 & . & . & . & u^1_n \\ v^1_1 & v^1_2 & & & & v^1_n \\ u^2_1 & u^2_2 & & & & u^2_n \\ v^2_1 & v^2_2 & & & & v^2_n \end{pmatrix} = \begin{pmatrix} s^1_{1,1} & s^1_{1,2} & s^1_{1,3} & t^1_x \\ s^1_{2,1} & s^1_{2,2} & s^1_{2,3} & t^1_y \\ s^2_{1,1} & s^2_{1,2} & s^2_{1,3} & t^2_x \\ s^2_{2,1} & s^2_{2,2} & s^2_{2,3} & t^2_y \end{pmatrix} A^{-1} \begin{pmatrix} 0 & 1 & 0 & 0 & . & . & . & x_n \\ 0 & 0 & 1 & 0 & & & & y_n \\ 0 & 0 & 0 & 1 & & & & z_n \\ 1 & 1 & 1 & 1 & & & & 1 \end{pmatrix}$$

Affine Structure-from-Motion: Two Frames (9)

Given:

$$\begin{pmatrix} u^1_1 & u^1_2 & . & . & . & u^1_n \\ v^1_1 & v^1_2 & & & & v^1_n \\ u^2_1 & u^2_2 & & & & u^2_n \\ v^2_1 & v^2_2 & & & & v^2_n \end{pmatrix} = \begin{pmatrix} s^1_{1,1} & s^1_{1,2} & s^1_{1,3} & t^1_x \\ s^1_{2,1} & s^1_{2,2} & s^1_{2,3} & t^1_y \\ s^2_{1,1} & s^2_{1,2} & s^2_{1,3} & t^2_x \\ s^2_{2,1} & s^2_{2,2} & s^2_{2,3} & t^2_y \end{pmatrix} A^{-1} \begin{pmatrix} 0 & 1 & 0 & 0 & . & . & . & x_n \\ 0 & 0 & 1 & 0 & & & & y_n \\ 0 & 0 & 0 & 1 & & & & z_n \\ 1 & 1 & 1 & 1 & & & & 1 \end{pmatrix}$$

Then we just pretend that:

$$\begin{pmatrix} s^1_{1,1} & s^1_{1,2} & s^1_{1,3} & t^1_x \\ s^1_{2,1} & s^1_{2,2} & s^1_{2,3} & t^1_y \\ s^2_{1,1} & s^2_{1,2} & s^2_{1,3} & t^2_x \\ s^2_{2,1} & s^2_{2,2} & s^2_{2,3} & t^2_y \end{pmatrix} A^{-1}$$

is our motion, and solve as before.

Affine Structure-from-Motion: Two Frames (10)

This means that we can never determine the exact 3D structure of the scene. We can only determine it up to some transformation, *A*. Since if a structure and motion explains the points:

$$\begin{pmatrix} u_1^1 & u_2^1 & . & . & . & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

So does another of the form:

$$\begin{pmatrix} u_1^1 & u_2^1 & . & . & . & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \left(\begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} A^{-1} \right) \left(A \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix} \right)$$

Affine Structure-from-Motion: Two Frames (11)

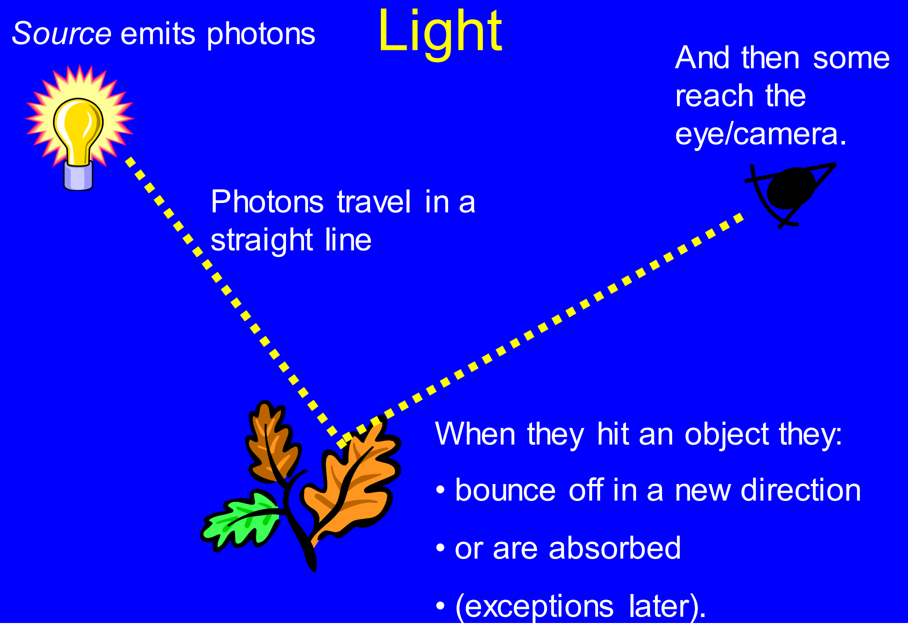
$$\begin{pmatrix} u_1^1 & u_2^1 & . & . & . & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \left(\begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} A^{-1} \right) \left(A \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix} \right)$$

Note that *A* has the form:

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

A corresponds to translation of the points, plus a linear transformation.

Light



Lambertian + Point Source

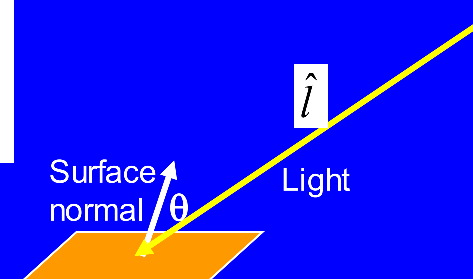
$$\vec{l} = l \bullet \hat{l} \quad \begin{cases} \hat{l} \text{ is direction of light} \\ l \text{ is intensity of light} \end{cases}$$

$$i = \max(0, \lambda(\vec{l} \bullet \hat{n})$$

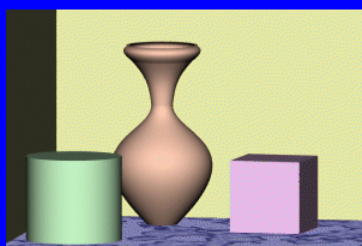
i is radiance

λ is *albedo*

$\hat{n}$ is surface normal

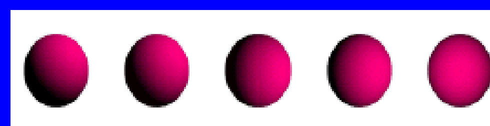


Lambertian Examples



Scene

(Oren and Nayar)



Lambertian sphere as the light moves.

(Steve Seitz)

Lambertian, point sources, no shadows. (Shashua, Moses)

- *Whiteboard*
- Solution linear
- Linear ambiguity in recovering scaled normals
- Lighting not known.
- Recognition by linear combinations.

With no shadows, Lambertian reflectance
If we let *n*=lambda*\hat(n) (albedo times surface normal) and *l* denote the lighting intensity times the direction, *i=<n,l>,* the inner product between two 3D vectors, one representing the normal scaled by albedo, the other representing lighting direction scaled by intensity. If we have an image with many pixels:

$$\begin{pmatrix} i_1 & i_2 & \dots i_m \end{pmatrix} = \begin{pmatrix} l_x & l_y & l_z \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \dots \\ y_1 & y_2 & \dots \\ z_1 & z_2 & \dots \end{pmatrix}$$

If we have several images, we can stack these:

$$\begin{pmatrix} i_1 & i_2 & \dots i_m \\ i_1 & i_2 & \dots i_m \\ i_1 & i_2 & \dots i_m \end{pmatrix} = \begin{pmatrix} l_x & l_y & l_z \\ l_x & l_y & l_z \\ l_x & l_y & l_z \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \dots \\ y_1 & y_2 & \dots \\ z_1 & z_2 & \dots \end{pmatrix}$$

$$\begin{pmatrix} i_1 & i_2 & \dots i_m \\ i_1 & i_2 & \dots i_m \\ i_1 & i_2 & \dots i_m \end{pmatrix} = \begin{pmatrix} l_x & l_y & l_z \\ l_x & l_y & l_z \\ l_x & l_y & l_z \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \dots \\ y_1 & y_2 & \dots \\ z_1 & z_2 & \dots \end{pmatrix}$$

Each row on the left is a different image, and each row on the first matrix on the right is a different light, but these are all images of the same scene. This equation looks exactly like structure-from-motion, and can be solved in the same way. With many pixels we can solve for the scaled surface normals, up to a linear transformation.