

1 Stereo and structure from motion

1.1 Perspective projection

$$x = \frac{fX}{Z}, \quad y = \frac{fY}{Z}$$

1.2 Orthographic projection

$$x = sX, \quad y = sY$$

1.3 Epipolar lines

Given two images of the same scene, the line connecting the two camera centers is the base line. A world point + the base line determine an epipolar plane. The intersection of this plane with the images determine epipolar lines.

$$\overline{OP}^T (\overline{OO'} \times \overline{O'P}) = 0 \Rightarrow \vec{p}^T (\vec{t} \times R\vec{q}) = 0 \Rightarrow \vec{p}^T [t]_{\times} R\vec{q} = 0$$

$$[t]_{\times} = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$$

$E = [t]_{\times} R$ is the Essential matrix.

All epipolar lines intersect at the epipole. The Epipole is the projection of the other camera's center, i.e., where the other camera is seen in the image.

The left epipole is located at αt , and the right epipole at $R^T t$.

Since the epipole is the null space of E , $\text{rank} E = 2$. This follows also from the fact that $\text{rank}[t]_{\times} = 2$ and $\text{rank} R = 3$.

1.4 Calibration

Let (a_x, a_y) be the dimension of each pixel, b determine the skew (the fact that the pixels are not exactly square), and (c_x, c_y) be the center of the image.

$$K = \begin{bmatrix} a_x & b & c_x \\ 0 & a_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

For an uncalibrated point p' , $p = K^{-1}p'$ is the calibrated point.

$$p'^T \underbrace{K^{-T}EK^{-1}}_F q' = 0$$

F is the Fundamental matrix. Rank $F=2$.

1.5 Finding E/F

Given two images, and a few point correspondences, we wish to find E (or F).

First of all, let us determine how many point correspondences we need:

- For every point correspondence, we get 1 equation ($p^T E q = 0$).
- E is $3 \times 3 \Rightarrow$ it has 9 unknowns \Rightarrow we need 8 correspondences.
- E is rank 2 \Rightarrow 8 unknowns (6 elements plus 2 coefficients for the third row) \Rightarrow need 7 correspondences.
- E was created from a translation and rotation, each has 3 DOF \Rightarrow 6 unknowns \Rightarrow need 5 correspondences.

Given correspondences, how can one calculate E?

$$\text{Let } A = \begin{bmatrix} p_{1x}q_{1x} & p_{1x}q_{1y} & p_{1x}f & p_{1y}q_{1x} & p_{1y}q_{1y} & p_{1y}f & f q_{1x} & f q_{1y} & ff \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ p_{nx}q_{nx} & p_{nx}q_{ny} & p_{nx}f & p_{ny}q_{nx} & p_{ny}q_{ny} & p_{ny}f & f q_{nx} & f q_{ny} & ff \end{bmatrix},$$

and $e = [e_{11}, e_{12}, e_{13}, e_{21}, e_{22}, e_{23}, e_{31}, e_{32}, e_{33}]^T \Rightarrow$ We want to find $\min \|Ae\|^2$
 s.t. $\|e\| = 1$ (Solution is singular vector of A with least singular value)

1.6 Projective geometry

Two lines always intersect. Since two lines intersect exactly once, there are many points at infinity.

Attaching a coordinate system: $P^2 = R^3 - \{(0,0,0)\}$. Consider the center to be at $(0,0,0)$. Each point in the projective geometry is represented by a ray from the center through this point. Since a point is represented by a ray in 3D, a line is represented by a plane.

- Points at infinity are $(x, y, 0)$

- $ax + by + c = 0$ (a euclidean line) $\rightarrow ax + by + cz = 0$ (in projective geom)

\Rightarrow A line (a, b, c) is represented as $(a, b, c) \begin{pmatrix} x \\ y \\ z \end{pmatrix} = 0 \Rightarrow \vec{l}^T \vec{x} = 0$

- Intersection of two lines: $p = l_1 \times l_2$

- A line between two points: $l = p_1 \times p_2$

- Circles, ellipses, hyperbules etc. are: $p^T C p = 0$ for some matrix C.

1.7 Homographies

A finite sequence of perspectivities is a Projective transformation or a Homography.

By definition, a homography maps a line to a line.

Mathematically, homographies can be written as non-singular matrices:

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = H \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

We have seen three cases in which homography applies:

(1) Pure rotation (no translation)

$$Q = RP \Rightarrow \frac{Z'}{f} q = \frac{Z}{f} Rp \Rightarrow q = \alpha Rp$$

Note that α is different for each point.

(2) Rotation+translation, when looking at a plane

A plane is determined by $n^T P = d \Rightarrow \frac{n^T P}{d} = 1$

$$Q = RP + t = RP + t \frac{n^T P}{d} = (R + \frac{1}{d} t n^T) P = HP$$

$$Q = HP \Rightarrow q = \alpha H p, \quad \alpha = \frac{z}{z'}$$

(3) When the camera is far from the scene, i.e. when $z \rightarrow \infty$

$$\begin{aligned} q &= \frac{f}{Z'} Q = \frac{f(RP + t)}{R_3 P + t_z} = \frac{f(R \frac{z}{f} p + t)}{R_3 \frac{z}{f} p + t_z} = \\ &= \frac{f(Rp + \frac{f}{z} t)}{R_3 p + \frac{f}{z} t_z} \xrightarrow{z \rightarrow \infty} \frac{f R p}{R_3 p} = \underbrace{\frac{f}{R_3 p}}_{\alpha} \cdot R \cdot p \end{aligned}$$

Note: When we have a homography, depth can not be recovered.

1.8 Finding H

Given two images, how many point correspondences are needed in order to recover H?

From each correspondence we get 2 equations $\begin{pmatrix} x' \\ y' \\ f \end{pmatrix} = \alpha H \begin{pmatrix} x \\ y \\ f \end{pmatrix}$. From

the third equation we take α , and substitute in first two equations). H is $3 \times 3 \Rightarrow$ has 9 unknowns \Rightarrow Need 4 correspondences in order to recover H (up to a scale factor).

There are a few methods for finding H:

(1) Using SVD, similarly to finding E in the epipolar constraints.

$$\begin{aligned} q &= \frac{f}{Z'} Q = \frac{f}{Z'} H P = \frac{f}{H_3 P} H P \Rightarrow \\ \Rightarrow q H_3 P &= f H P \Rightarrow (q H_3 I - f H) P = 0 \Rightarrow (q H_3 I - f H) p = 0 \end{aligned}$$

This is a homogenous equation $A\vec{h} = 0$, where A is $2n \times 9$

(2) Using Brightness-Constancy-Equation, see section 2.8.

(3) RANSAC - Random Simple Consensus. Usually, one uses RANSAC to find lines in an image - take each two points, and check if the line they determine has many nearby points. Application to homographies: Try defining the homography with different sets of points, and see how many other points agree with this homography.

(4) Hough: Each 4 correspondences vote for a possible homography. The homography which gets the most votes - wins. Note that the voting table is 8 dimensional (The 8 elements of H).

1.9 Orthographic projection

$$x = sX, \quad y = sY$$

Orthographic is an approximation of perspective when the origin is at infinity.

$$Q = RP + t \Rightarrow \begin{cases} x' = r_{11}x + r_{12}y + r_{13}z + t_x \\ y' = r_{21}x + r_{22}y + r_{23}z + t_y \end{cases} \Rightarrow$$

$$\dots \Rightarrow r_{23}x' - r_{13}y' = -r_{32}x + r_{31}y + (r_{23}t_x - r_{13}t_y)$$

For a given point (x, y) , (x', y') form a line - the epipolar line. Epipolar lines are parallel, since $Ax' + By' = C$, and A,B do not depend on x,y.

1.10 Finding epipolar lines in an orthographic projection

How many point correspondences are needed? From each pair of points we get one equation. We are looking for a solution for: $Ax' + By' + Cx + Dy + E = 0 \Rightarrow$ need 4 point correspondences.

1.11 Orthographic projection - finding R, t

Given point correspondences, what can be done? Can R and t be recovered? Pick one correspondence to be the origin of both images, and by that get rid of the translation. To find R , consider R as $R_z(\alpha)R_y(\beta)R_z(\gamma)$, meaning - rotation of the image, rotation around Y axes, and rotation of the image again. The first rotation rotates the epipolar lines to be horizontal, the second shifts points along the epipolar lines, and the last one rotates the epipolar lines again. Thus, only the first and last components can be recovered. In the second one, points move along the epipolar lines, and you cannot know whether this is because of rotation or their depth.

Note: Given three (non co-linear) images, can recover the rotation uniquely, since on a sphere - three angles determine the triangle uniquely.

1.12 Affine transformation

An Affine transformation is a special case of a homography, in which

$$H = \begin{bmatrix} a & b & c \\ d & e & g \\ 0 & 0 & 1 \end{bmatrix}$$

It is applicable when the field of view is narrow, or when the plane is parallel to the image.

1.13 Stereo algorithms - or - Finding correspondences

Task: Given two images, and epipolar lines in them, we want to find point correspondences for the rest of the image.

Rectification: The process of bringing two images to be parallel to the base line. Since we know the epipolar lines in the images, this can be done. After rectification, the epipolar lines are all parallel and horizontal.

Thus, we need to find correspondences only along horizontal lines, i.e., need

to find disparity of points.

Stereo algorithms can be edge-based or intensity-based.

Edge-based algorithms are less common. In these algorithms, you apply edge-detection, and then try to associate edges in both images.

Intensity-based algorithms:

$$E_{data} = \sum_{\text{correspondences}} (I(x_l, y) - I(x_r, y))^2$$

$$E_{smooth} = \sum_{\text{correspondences}} (d(x_l, y) - d(x_{l+1}, y))^2 = \sum ((x_l - x_r) - (x_{l+1} - x_{r'}))^2$$

E_{data} represents the difference of intensity in each correspondence. E_{smooth} represents the change in disparity between neighbouring pixels. We want both to be small, i.e., we want each pixel's change in intensity to be small, and we want neighbouring pixels to have similar disparities.

Trivial algorithm: Define $E = E_{data} + \lambda E_{smooth}$, and try to minimize E .

We have discussed three algorithms:

(1) Dynamic programming: Take two lines (their intensity values), and consider them as strings on which you apply dynamic programming. Moving diagonally means you match the two pixels, and pay $(I_l - I_r)^2$. Moving horizontally or vertically changes the disparity, and you pay a constant price. Find cheapest path from one corner to the other.

(2) Relaxation: Start with an initial guess, and change one pixel at a time, until you achieve a minimum.

(3) Minimal cut algorithm: First, discretize the whole 3D space. The solution we're looking for is a surface in 3D. In each X-Z slice (or Y-Z), the solution is a curve. Consider a single X-Z slice. Apply weight E_{data} to vertical edges (a change in the X value, no change in Z). Apply weight E_{smooth}

to horizontal edges (no change in X, a change in Z). Apply this also to Y-Z slices, and thus give weights for the whole 3D space. Now define a source at zero (at the camera) and a target at infinity, and find a minimal cut.

1.14 Motion Detection

Suppose we're tracking p points in f images, i.e., we have coordinates of p points in f images. Task: Recover shape (world coordinates of the p points) and motion (motion of the camera between each of the f images).

$$M = \begin{bmatrix} x_{11} & \cdots & x_{1p} \\ & & \vdots \\ x_{f1} & \cdots & x_{fp} \\ y_{11} & \cdots & y_{1p} \\ & & \vdots \\ y_{f1} & \cdots & y_{fp} \end{bmatrix}$$

M is the measurement matrix.

Consider one point as the origin, and thus get rid of translation.

Want to find: $M = T \cdot S$, $S = \text{shape} = \begin{bmatrix} X_1 & X_2 & \cdots \\ Y_1 & Y_2 & \cdots \\ Z_1 & Z_2 & \cdots \end{bmatrix}$, $T = \text{transformation} =$

$$\begin{bmatrix} r_{11}^1 & r_{12}^1 & r_{13}^1 \\ r_{11}^2 & r_{12}^2 & r_{13}^2 \\ \vdots & & \\ r_{21}^1 & r_{22}^1 & r_{23}^1 \\ r_{21}^2 & r_{22}^2 & r_{23}^2 \\ \vdots & & \end{bmatrix}. \text{ Dimensions: } M : 2f \times p, \quad T : 2f \times 3, \quad S : 3 \times p. \text{ Ideally,}$$

M should be of rank 3.

To enforce the rank of M , use SVD: $M = U\Delta V^T$, $\Delta = \text{diag}(\lambda_1, \dots, \lambda_n)$. Let $\bar{\Delta} = \text{diag}(\lambda_1, \lambda_2, \lambda_3)$, $\bar{M} = U\bar{\Delta}V^T \Rightarrow \bar{M}$ is the rank 3 approximation of M .

Define $\bar{T} = U\sqrt{\bar{\Delta}}$, $\bar{S} = \sqrt{\bar{\Delta}}V^T \Rightarrow M \approx \bar{T}\bar{S}$

This is not unique, since $M \approx \bar{T}\bar{S} = (\bar{T}A)(A^{-1}\bar{S})$.

In order to reduce ambiguity, require that each rotation satisfies $RR^T = I$:

Let $T = \begin{bmatrix} \bar{s}_1^T \\ \bar{s}_2^T \\ \vdots \\ \bar{t}_1^T \\ \bar{t}_2^T \\ \vdots \end{bmatrix}$. Require: $s_i^T s_i = 1$, $s_i^T t_i = 0$, $t_i^T t_i = 1$. Since $s_i^T =$

$\bar{s}_i^T A$, $t_i^T = \bar{t}_i^T A$, should require $\bar{s}_i^T A^T A \bar{s}_i = 1$, $\bar{s}_i^T A^T A \bar{t}_i = 0$, $\bar{t}_i^T A^T A \bar{t}_i = 1$.

Since we get 3 such equations from each frame, we get 3f equations in 6 unknowns ($A^T A$ is symmetric).

Note that this solution is still not unique, but up to a rotation B : $A^T B^T B A = A^T A$. But since this rotation is arbitrary, set the first transformation to be I.

A problem with this method is that all the p points must be in all the frames.

1.15 Finding transformations without committing to intensities

Given two images, with an unknown transformation between them. we wish to find the transformation. All the correlation methods we have discussed commit you to the intensities. What if there was a significant light change, and the intensities completely changed?

One way is to use Mutual information.

Let x, y be the two images, and consider the images as distributions. Now compare the distributions $p(x, y)$ and $p(x) \cdot p(y)$. What you are looking for is dependence between the images, a way in which pixels in one image can help predict pixels at the same location in the second image.

2 Optical Flow Fields

2.1 Introduction

Given two consecutive images, find the motion of each pixel. That is, for each (x, y) - we want to find (u, v) s.t. pixel (x, y) in image A moved to pixel $(x + u, y + v)$ in image B.

2.2 Local Information

Using only local information (information on a single pixel in each frame), one usually runs into the aperture problem: locally there is usually not enough information to know the exact movement.

2.3 Brightness Constancy Assumption

The assumption: Brightness of each world point does not change between the images. Mathematically:

$$\begin{aligned} f(x, y) &= g(x + u, y + v) \\ &\approx g(x, y) + ug_x(x, y) + vg_y(x, y) \\ \Rightarrow 0 &= \underbrace{(g - f)}_{g_t} + ug_x + vg_y \Rightarrow g_x u + g_y v + g_t = 0 \end{aligned}$$

g_t, g_x, g_y are known, and can be calculated directly from the image.

Conclusion: $(u, v) \begin{pmatrix} g_x \\ g_y \end{pmatrix} = -g_t$. i.e., we can only know the projection of (u, v) onto the direction of the gradient. This is called Normal flow

2.4 Improvement 1: Horn & Schunk

$$\min_{(u,v)} \left\{ \underbrace{\sum_{(x,y)} (g_x u + g_y v + g_t)^2}_{\text{BCE}} + \lambda \underbrace{\sum_{(x,y)} (u_x^2 + u_y^2 + v_x^2 + v_y^2)}_{\text{Smoothness}} \right\}$$

Explanation: Require, in addition to BCE, that the optical flow will be smooth.

Drawbacks:

(1) How do you choose λ ? (3) Near edges, there is no smoothness of the optical flow. This method smoothes it.

2.5 Improvement 2: Lucas & Kanade

Assume all points in a 5×5 neighborhood have the same displacement. i.e., the pixels in the vicinity all have the same (u,v) , and we want to minimize BCE for this whole vicinity. When you have different gradients in the neighborhood, will get from the BCE intersection lines.

$$Err(u, v) = \sum_{\text{vicinity}} (I_{ix}u + I_{iy}v + I_{it})^2$$

$$\frac{\partial Err}{\partial u} = 0, \frac{\partial Err}{\partial v} = 0 \Rightarrow \begin{pmatrix} \sum I_{ix}^2 & \sum I_{ix}I_{iy} \\ \sum I_{ix}I_{iy} & \sum I_{iy}^2 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -\sum I_{it}I_{ix} \\ -\sum I_{it}I_{iy} \end{pmatrix}$$

The first matrix is non-singular when you have gradients in different directions.

2.6 Other improvements and notes

(*) Above methods use 1st order Taylor approx., hence assume u and v are small. One can iterate in order to find a better approx.

$$\begin{aligned} \begin{pmatrix} u \\ v \end{pmatrix} &= \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} + \begin{pmatrix} \delta_u \\ \delta_v \end{pmatrix} \Rightarrow f(x, y) = g(x + u_0 + \delta_u, y + v_0 + \delta_v) \\ &\approx g(x + u_0, y + v_0) + \delta_u g_x(x + u_0, y + v_0) + \delta_v g_y(x + u_0, y + v_0) \end{aligned}$$

And iterate this.

(*) Refinement: One can use pyramids, and build a coarse-to-fine estimation.

(*) All the above are gradient based methods.

Region based methods: Take small regions and find correlations in the second image. One can use correlations, normalized correlations, sum-square-differences.

2.7 Introducing global constraints

$$0 = I_x u + I_y v + I_t = I_x(x - x') + I_y(y - y') + I_t = I_x x' + I_y y' + \overbrace{(I_t - I_x x - I_y y)}^m$$

$$p' = \frac{fP'}{Z'} = \frac{f(RP + T)}{R_3p + T_3} = \frac{Rp + T(f/Z)}{R_3p + T_3(f/Z)}$$

Plug this into BCE:

$$0 = I_x \underbrace{\frac{f(R_1p) + T_1(f/Z)}{R_3p + T_3(f/Z)}}_a + I_y \underbrace{\frac{f(R_2p) + T_2(f/Z)}{R_3p + T_3(f/Z)}}_b + m$$

R, T, f are global parameters, which can be recovered globally. Z is different for each point. Assume R, T, f are known.

$$0 = I_x a + I_y b + mc$$

This is a linear equation in $\frac{1}{Z}$. For an edge pixel, this defines the displacement, since this defines a single line. Epipolar constraints define another line, and by their intersection can recover the displacement.

In large uniform region, we've reduced the problem to a linear one (only epipolar constraint, the BCE does not help).

2.8 Optical flow and homographies

When a homography is applicable, for each point (x, y) , the corresponding point is known uniquely, thus the optical flow is known uniquely.

B.C.E. can be used in order to recover H:

$$I_x x' + I_y y' + m = I_x \frac{H_1 p}{H_3 p} + I_y \frac{H_2 p}{H_3 p} + m$$

$$\Rightarrow 0 = I_x H_1 p + I_y H_2 p + m H_3 p$$

Each pixel provides one such equation. We want to find:

$$\min_{h_1 \dots h_9} \sum_p (I_x H_1 p + I_y H_2 p + m H_3 p)^2$$

(Will get $A \begin{bmatrix} h_1 \\ \vdots \\ h_9 \end{bmatrix} = \vec{b}$, and solve using SVD). Note that A is constructed of

p, I_x, I_y . Also note that we do not need to find point correspondences at all this way.

Moreover, each pixel in the image contributes to the estimation of H . For a vertical edge, I_x is large, and $I_y = 0$, thus it will contribute only to the relevant part of H .

3 Segmentation

3.1 Smooth completion

Elastica

$$\min_{\Gamma(s)} \int_0^l k^2(s) ds$$

$$k = \text{the curvature} = \frac{1}{r}$$

Normal elastica is extensible yet scale-dependant.

Scale Invariant Elastica

$$\min_{\Gamma(s)} l \int_0^l k^2(s) ds$$

- This elastice is scale invariant yet it has no extensibility.
- If possible, the best curve would be a circle.
- Approximation:

$$4(\Phi_1^2 + \Phi_2^2 - \Phi_1 \Phi_2)$$

3.2 Gestalt Principles for Grouping

- Proximity: tokens that are nearby tend to be grouped.
- Similarity: similar tokens tend to be grouped together.
- Common fate: tokens that have coherent motion tend to be grouped together.
- Closure: tokens or curves that tend to lead to closed curves tend to be grouped together.
- Continuity: tokens that lead to continuous (as in joining up nicely, rather than in the formal sense): curves tend to be grouped.
- Simplicity

3.3 Connecting Between Edge Elements

We have discussed a few algorithms:

(1) Tensor Voting

- Every edge votes for all its circular edge completions, i.e. to all other edges which can sit on the same circle as it.
- Votes decay with distance and curvature

(2) Stochastic Completion Field

- Apply a random walk:

$$\dot{X} = \cos \Theta \ ; \ \dot{Y} = \sin \Theta \ ; \ \dot{\Theta} = N(0, \sigma^2)$$

That is: The change in direction is drawn from a normal distribution, and then X and Y change appropriately.

In addition, particles can die with probability $e^{-1/r}$

- Send such particles from each edge, and choose paths from one edge to the other, on which many particles have walked.

The most probable path is:

$$\alpha \int k^2(s) ds + \beta \int ds$$

(3) Shortest path

(4) Snakes

3.4 Segmentation

3.4.1 Basic algorithm

Build a graph. Each pixel is a vertex, and there's edges between adjacent pixels. The weights of the edges are the similarity between the pixels (more similar \Rightarrow Larger weight). Now find a minimal cut.

The problem with this algorithm is that it usually finds small clusters.

3.4.2 Normalization

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)}$$
$$cut(A, B) = \sum_{i \in A; j \in B} W_{ij} ; \quad assoc(A, V) = \sum_{i \in A; j \in V} W_{ij}$$

This problem is NP-complete

3.4.3 Soft segmentation

In soft segmentation, each pixel can also belong partly to A and partly to B.

Define:

$$y_i = \begin{cases} 1 & i \in A \\ -1 & i \in B \\ \text{intermediate} & \text{soft} \end{cases}$$

This can be solved using eigenvalues. Define:

W - weight matrix

D - diagonal, $D_{ii} = \sum_j W_{ij}$

L - Laplacian, $L = D - W$

And optimize the Reiley Quotient:

$$\min \frac{y^T Ly}{y^T Dy}$$

To solve:

$$Ly = \lambda Dy$$

Which is a generalized eigenvalue problem.

Explanation Assume for a minute there was no D: $\min \frac{y^T Ly}{y^T y}$. Let u_1, \dots, u_n be eigenvectors of L, with eigenvalues $\lambda_1, \dots, \lambda_n$.

$$\begin{aligned} y = \sum \alpha_i u_i &\Rightarrow Ly = \sum \alpha_i Lu_i = \sum \alpha_i \lambda_i u_i \\ &\Rightarrow \lambda_1 |y| \leq Ly \leq \lambda_n |y| \Rightarrow \lambda_1 \leq \frac{y^T Ly}{y^T y} \leq \lambda_n \end{aligned}$$

Hence, should choose y to be the eigenvector with the smallest eigenvalue.

If we bring back D now, the problem becomes a generalized eigenvalue problem.

4 Photometry

4.1 The Lambertian Model

A few properties:

- (1) Light is reflected to all directions in equal amount.
- (2) Brightness does not depend on the location of the observer.
- (3) Applicable for mat surfaces.

If so, What does determine intensity?

- (1) The color of the surface. Darker colors absorb more light, hence reflect less.
- (2) The amount of light which reaches the surface, and its direction. Note

that light energy reduces as r^2 .

Assumption: The light source is at ∞ . When the light source is at infinity, all rays reach the surface at parallel lines. Nevertheless, the angle at which it reaches the surface is still important. Per area unit, we obtain less energy as the angle becomes bigger, as determined by $\cos(n, l)$.

4.2 Mathematically

Assume Lambertian model, and light source at infinity. Definitions:

I = Intensity

ρ - Albedo (amount of light the surface reflects)

E - Energy, Intensity of the light source

$$I = \rho E \cos \Theta = \rho E (\hat{l}^T \hat{n}) = l^T n$$

For $l = E\hat{l}$; $n = \rho\hat{n}$.

4.3 DOF - same scene, different light sources

Assumptions: Lambertian model, 1 point light source at infinity, and no attached shadows.

Let I_1, I_2, \dots denote images of the exact same scene with different light sources.

$$I_1(x, y) = l_1^T \cdot n(x, y) \ ; \ I_2(x, y) = l_2^T \cdot n(x, y) \ \dots$$

l has 3 DOF, hence after 3 I 's, I will become linearly dependent:

$$\begin{aligned} I_n(x, y) &= l_n^T(x, y) \cdot n(x, y) = \\ &= (\alpha_1 l_1 + \alpha_2 l_2 + \alpha_3 l_3)^T \cdot n(x, y) = \alpha_1 I_1(x, y) + \alpha_2 I_2(x, y) + \alpha_3 I_3(x, y) \end{aligned}$$

This means that after taking 3 such images, any image taken is basically a linear combination of the first three.

4.4 Photometric Stereo

Given I_1, \dots, I_f - f Images of the same scene with different light sources, we wish to find the object dimensions and the light sources. Definitions:

f - number of images

p - number of normals to the object we wish to find.

L - Light matrix, of dimensions $f \times 3$. f rows, each contains the coordinates of a light source.

S - Shape matrix, of dimensions $3 \times p$. p columns, each contains the normal to a single point.

M - Intensities matrix, of dimensions $f \times p$. Contains p intensities in f images.

Recovering S and L

$$M = L \cdot S$$

The rank of M should be 3. In order to enforce it, one can use SVD:

$$M = U\Delta V^T \Rightarrow \Delta_3 = \text{diag}(\lambda_1, \lambda_2, \lambda_3)$$
$$\hat{L} = U\sqrt{(\Delta_3)}, \hat{S} = \sqrt{(\Delta_3)}V^T \Rightarrow M \approx \hat{L}\hat{S}$$

Reducing Ambiguity

This decomposition is not unique, since $\hat{L}\hat{S} = (\hat{L}A^{-1})(A\hat{S})$. In order to reduce ambiguity, we should require that the recovered surface could be integrable:

Let $(x, y, z(x, y))$ be the surface. By differentiating, we get:

$$t_x = (1, 0, z_x), t_y = (0, 1, z_y) \Rightarrow \hat{n} \propto t_x \times t_y = (-z_x, -z_y, 1)$$

A consistent surface is an integrable one, hence should require:

$$z_{xy} = z_{yx}$$

The normals can be found in the columns of S - each column of S is proportional to a surface normal.

Note that we are left with a Generalized Bas-Relief transformation ambiguity, which is: $\hat{z}(x, y) = \alpha x + \beta y + \gamma z(x, y)$

4.5 Introducing Shadows

There are two types of shadows: attached shadows, which are generated when the light source is behind the object ($\Theta > 90^\circ$), and cast shadows, which are generated when one object blocks the light to the other.

Note that in daylight, the atmosphere reflects light in all directions, and causes Ambient light.

Because of attached shadows, we should write:

$$I = \max\{l^T n, 0\}$$

5 Object Recognition

5.1 Recognition VS Classification

Classification - Name a type of an object.

Recognition - Recognize a specific object.

5.2 Recognition - Method 1

Given 3 model points (3D), and their corresponding 2D points in an image, one can determine a unique transformation that aligns the model with the image.

$$\Pi(SR + d)P_i = p_i$$

Where:

Π - Projection

S - Scale

R - Rotation

d - Displacement

- This is an orthographic projection
- The 3 points must be non-collinear

- Determined up to a reflection of the points around the image plane and translation in depth.

5.3 Linear combination of views

O is a set of object points. I_1, I_2, I_3 are three images of O from different views. N is a novel view of O . Then O is the linear combination of I_1, I_2, I_3 .

$$x_n^i = a_1x_1^i + a_2x_2^i + a_3x_3^i ; y_n^i = b_1x_1^i + b_2x_2^i + b_3x_3^i$$

For rotation around a fixed axis, 2 images are enough

Drawbacks: (1) You need a 3D model or exact correspondences between the images. (2) Registration of the images (???)

5.4 Classifications 1 - Structural description

This method tries to build everything out of basic building blocks. It failed...

5.5 Classifications 2 - Fragment based representaiton

- Use 2D primitives, which depend on the class we're trying to describe. When choosing building blocks, they can't be too large (large ones are too rare) or too small (small ones appear everywhere). We will look for features which carry the highest amount of information.

Finding features using mutual information

Let X denote whether there is a face in the image, and F be a candidate feature. Calculate $H(X)$, and $H(X|F)$

$$I(X, F) = H(X) - H(X|F)$$

Select features which maximize the mutual information. Finding the best threshold can also be done this way (Not sure what this means...)

This can also be done using multiple resolutions.

Weighting features

Likelihood ratio: The ratio between the probability the feature exists in a face image and the prob. it appears in a non-face image.

$$R(F) = \frac{P(F|X)}{P(F|\bar{X})}$$

Weight of F:

$$W(F) = \log(R(F))$$

Weighted decision:

$$\sum W_i F_i$$