Two-view geometry
(cont’d)
Multi-view geometry
Three questions:

(i) **Correspondence geometry**: Given an image point $x$ in the first view, how does this constrain the position of the corresponding point $x'$ in the second image?

(ii) **Camera geometry (motion)**: Given a set of corresponding image points $\{x_i \leftrightarrow x'_i\}$, $i=1,\ldots,n$, what are the cameras $P$ and $P'$ for the two views?

(iii) **Scene geometry (structure)**: Given corresponding image points $x_i \leftrightarrow x'_i$ and cameras $P, P'$, what is the position of (their pre-image) $X$ in space?
Outline

• 2-view geometry
• essential matrix, fundamental matrix
• properties
• estimation
Mathematical formulation

Goal: given point in left image, we want to compute the equation of the line on the right image
Definitions

How do epipolar lines change when we double distance between two cameras?

**Epipolar plane:** plane defined by 2 camera centers & candidate 3D point (green)
(also defined by 2 camera centers any 1 points in either image plane)

**Epipolar lines:** intersection of epipolar plane and image planes (red)

**Epipoles:** projection of camera center 1 in camera 2 (& vice versa) (orange)
(set of all epipolar lines intersect at the epipoles)
Special case

Epipolar geometry is purely determined by camera extrinsics and camera intrinsics.
Projecting from camera coordinate system to image coordinates:

\[
\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = K \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} f s_x & f s_\theta & o_x \\ 0 & f s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}
\]
Projecting from camera coordinate system to \textit{normalized} image coordinates

If $K$ is known, work with warped image

\[
\begin{bmatrix}
    x' \\
    y' \\
    1
\end{bmatrix} = K^{-1} \begin{bmatrix}
    x \\
    y \\
    1
\end{bmatrix}
\]

\[\lambda x' = X\]

To simplify notation, we’ll use $x$ instead of $x'$
Recall

Dot product: \( \mathbf{a} \cdot \mathbf{b} = ||\mathbf{a}|| \ ||\mathbf{b}|| \cos \theta \)

Cross product: \( \mathbf{a} \times \mathbf{b} = ||\mathbf{a}|| \ ||\mathbf{b}|| \sin \theta \mathbf{n} \)

Cross product matrix: \( \mathbf{a} \times \mathbf{b} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \equiv \hat{\mathbf{a}} \mathbf{b} \)

Important property (skew symmetric): \( \hat{\mathbf{a}}^T = -\hat{\mathbf{a}} \)
Recall

Dot product: \( \mathbf{a} \cdot \mathbf{b} = \| \mathbf{a} \| \| \mathbf{b} \| \cos \theta \)

Cross product: \( \mathbf{a} \times \mathbf{b} = \| \mathbf{a} \| \| \mathbf{b} \| \sin \theta \mathbf{n} \)

Cross product matrix: \( \mathbf{a} \times \mathbf{b} = \hat{\mathbf{a}} \mathbf{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \)

\( \mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) = \text{volume of parallelepiped} \)
\( = 0 \text{ for coplanar vectors} \)
Calibrated 2-view geometry

\[ X_2 = RX_1 + T \]

\[ X_1 = \lambda_1 x_1, \quad X_2 = \lambda_2 x_2 \]
**Epipolar geometry**

\[
X_2 = RX_1 + T
\]

\[
X_1 = \lambda_1 x_1, \quad X_2 = \lambda_2 x_2
\]

\[
\lambda_2 x_2 = R\lambda_1 x_1 + T
\]

*Take (left) cross product of both sides with \( T \)*

\[
\lambda_2 \hat{T} x_2 = \hat{T} R\lambda_1 x_1 + \hat{T} T \quad \overset{=} \quad 0
\]

*Take (left) dot product of both sides with \( x_2 \)*

\[
\lambda_2 x_2^\top \hat{T} x_2 = x_2^\top \hat{T} R\lambda_1 x_1 \quad \overset{=} \quad 0
\]

\[
x_2^\top \hat{T} R x_1 = 0
\]
We will not handle the case of the conic being underdetermined ($n < 5$). From the SVD we take the "right singular vector" (a column from $V$) which corresponds to the smallest singular value, $\Sigma$. This is the solution, $c$, which contains the coefficients of the conic that best fits the points. We reshape this into the matrix $C$, and form the equation $x^T C x = 0$.

To recap, note that although the expression for a conic looks nonlinear, it is only the known variables (the coordinates of the $x_i$'s) that appear nonlinearly; we were able to write the problem in homogeneous least squares form since the coefficients appear linearly.

3.3. Two View Geometry

We now consider the geometry of two calibrated cameras viewing a scene. We assume that the cameras are related by a rigid body motion ($R, T$). (Figure from MaSKS Ch. 5.)

Since the cameras are calibrated, we have $K_1 = K_2 = I$. The cameras are centered at $o_1$ and $o_2$, respectively. The vectors $e_1$ and $e_2$ are the epipoles, and can be intuitively thought of as any of the following:

- The points where the baseline pierces the image planes
- The projection of the other camera's optical center onto each image plane
- The translation vector $T$ (up to a scale factor)
- The direction of travel (focus of expansion)

Simply the coplanar constraint applied to 3 vectors from camera 2’s coordinate system

$$x_2 \cdot (T \times R x_1) = 0$$
Epipolar geometry

\[
x_2^\top \hat{T} R x_1 = 0
\]

\[
x_2^\top Ex_1 = 0
\]

E is known as the *essential* matrix.
Fundamental matrix

(Faugeras and Luong, 1992)

In uncalibrated case, we need to account for camera intrinsics:

$$\lambda x = KX$$

$$E = \hat{T}R$$

$$F = K_2^{-T}EK_1^{-1}$$
Essential matrix

\[
\mathbf{x}_2^\top E \mathbf{x}_1 = 0
\]

Maps a \((x_1,y_1)\) point from left image to line in right image (and vice versa)

But how is this different from a Homography (also a 3X3 matrix)?
Epipoles

\[ x_2^T Ex_1 = 0 \]

We’ll write epipolar lines as 3-vectors: \[ l_2 = Ex_1 \]

Note that all epipolar lines in an image plane intersect at the epipole. Equivalently, the epipole has a distance of zero from every epipolar line: \[ e_2^T l_2 = 0, \forall x_1, \] and similarly \[ e_1^T l_1 = 0, \forall x_2. \]

For this to hold true, \[ e_2^T E \] and \[ E e_1 \] must be zero vectors, i.e.,

\[ e_2^T E = 0, \quad E e_1 = 0 \]

Thus \( e_1 \) and \( e_2 \) are vectors in the right and left null space of \( E \), respectively, i.e., the left and right singular vectors of \( E \) with singular value 0.
Outline

• 2-view geometry

• essential matrix, fundamental matrix

• properties

• estimation
Overview

Fundamental matrices:

\[ \mathbf{x}_2^T \mathbf{F} \mathbf{x}_1 = 0 \]

8 DOFs because of scale ambiguity

Rank 2

Essential matrices:

\[ \mathbf{x}_2^T \mathbf{E} \mathbf{x}_1 = 0 \]

\[ \mathbf{x}_2^T \hat{\mathbf{T}} \hat{\mathbf{R}} \mathbf{x}_1 = 0 \]

More-or-less behaves like a cross-product (skew symmetric matrix)
Q. How many DOFs are needed to specify an essential matrix?

3 (rotations) + 2 (translation direction)

Q. Can any 3x3 matrix be an essential matrix?

No…

E is the product of a rotation and skew-symmetric matrix

Singular values of E = (sigma,sigma,0)

[rotations do not effect singular values]

Q. Given E, can we uniquely recover R,t?

Almost. It is unique up to easy-to-deal with symmetries
Fig. 8.12. The four possible solutions for calibrated reconstruction from E. Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates 180° about the baseline. Note, only in (a) is the reconstructed point in front of both cameras.
Background: SVDs of skew symmetric matrices

Any skew-symmetric matrice \((A = -A^T)\) can be thought of as a cross-product

\[
a \times b = \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \equiv \hat{a}b
\]

SVD of a skew-symmetric matrix:

\[
\hat{a} = \begin{bmatrix} -e_2 & e_1 & e_3 \end{bmatrix} \begin{bmatrix} ||a|| & 0 & 0 \\ 0 & ||a|| & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} e_1^T \\ e_2^T \\ e_3^T \end{bmatrix} \text{ where } e_3 = a / ||a||
\]

One singular value is 0 and the other two = ||a||

\[
\hat{a} = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix} \begin{bmatrix} ||a|| & 0 & 0 \\ 0 & ||a|| & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} e_1^T \\ e_2^T \\ e_3^T \end{bmatrix}
\]
Recovering $T,R$ from $E$

1. Universal scale ambiguity

Doubling $T$ results in same epipolar lines

Let’s fix $||T|| = 1$

Numerous methods for recovering $t,R$ from $E$ exist: SVD, Louget-Higgen’s alg, etc.
Recovering $T$ from $E$

SVD-based approach for noise-free $E$ (Szeliski Chap 7.2)

\[
\mathbf{x}_2^T E \mathbf{x}_1 = 0
\]

Take (left-handside) cross product of $E = [t]_x \mathbf{R}$ with $t$

\[
\hat{t} \mathbf{r}^T E = 0.
\]

Implies that translation vector = epipole in right image (in homogenous coordinates)
Recovering T from E

Set translation direction = smallest left singular vector of E

But we can’t distinguish E from -E, so we only know direction up to a sign

Aside: \( v_2 \) = epipole in left image

\[
E = [\hat{t}] \times R = U \Sigma V^T = \begin{bmatrix} u_0 & u_1 & \hat{t} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_0^T \\ v_1^T \\ v_2^T \end{bmatrix}
\]
Recovering R from E
SVD-based approach (Szeliski Chap 7.2)

Recall skew-symmetric decomposition (for unit-norm vector)

\[
[t]_\times = SZR_{90\circ}S^T = \begin{bmatrix} s_0 & s_1 & \hat{t} \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} s_0^T \\ s_1^T \\ \hat{t}^T \end{bmatrix}
\]

\[
E = [\hat{t}]_\times R = SZR_{90\circ}S^T R = U\Sigma V^T
\]

By matching orthogonal and diagonal matrices, S = U, Z = Sigma

\[
R_{90\circ}U^T R = V^T
\]

\[
R = UR_{90\circ}V^T
\]

\[
R = \pm UR_{\pm90\circ}V^T
\]

Generate 4 possible rotations and keep 2 with determinant = 1 (non-reflections)
Fig. 8.12. The four possible solutions for calibrated reconstruction from E. Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates 180° about the baseline. Note, only in (a) is the reconstructed point in front of both cameras.
Properties (fundamental matrix)

\[
x_2^T K_2^{-T} \hat{T} R K_1^{-1} x_1 = 0
\]

\[
x_2^T F x_1 = 0
\]

Q. How many DOFs are needed to specify \( F \)?

\[8 = 9 - 1 \text{ (for scale)}\]

Q. Can any 3x3 matrix be a fundamental matrix?

No! epipoles are still in the null space, implying \( \text{rank}(F) = 2 \)

Proof: Let \( e_2 = K_2 T \)

\[e_2^T F = 0\]

(similar argument for \( e_1 \); c.f. Invitation to 3D Vision, Chap 6.2)
Properties (fundamental matrix)

\[ F = UV^T = \begin{bmatrix} u_0 & u_1 & e_1 \end{bmatrix} \begin{bmatrix} \sigma_0 & \sigma_1 & 0 \\ \end{bmatrix} \begin{bmatrix} v_0^T \\ v_1^T \\ e_0^T \end{bmatrix} \]

Two non-zero singular values are not (in general) equal

Singular vectors with zero singular value are the eipoles
Essential and Fundamental Matrices

\[
E = \hat{T} R \\
x_2^T E x_1 = 0 \\
E = \begin{bmatrix} u_0 & u_1 & e_2 \end{bmatrix} \begin{bmatrix} \sigma & \sigma \\ \sigma & 0 \end{bmatrix} \begin{bmatrix} v_0^T \\ v_1^T \\ e_1^T \end{bmatrix}
\]

"Proof": properties of skew-symmetric matrices

\[
F = K_2^{-T} E K_1^{-1} \\
x_2^T F x_1 = 0 \\
F = \begin{bmatrix} u_0 & u_1 & e_2 \end{bmatrix} \begin{bmatrix} \sigma_1 & \sigma_2 \\ \sigma_1 & 0 \end{bmatrix} \begin{bmatrix} v_0^T \\ v_1^T \\ e_1^T \end{bmatrix}
\]

Proof: scale ambiguity

where e1, e2 are epipoles in right and left images
Formal characterizations

Ma et al, An Invitation to 3D Vision

Theorem 5.1 (Characterization of the essential matrix). A non-zero matrix \( E \in \mathbb{R}^{3 \times 3} \) is an essential matrix if and only if \( E \) has a singular value decomposition (SVD): \( E = U \Sigma V^T \) with

\[
\Sigma = diag\{\sigma, \sigma, 0\}
\]

for some \( \sigma \in \mathbb{R}_+ \) and \( U, V \in SO(3) \).

Remark 6.1. Characterization of the fundamental matrix. A non-zero matrix \( F \in \mathbb{R}^{3 \times 3} \) is a fundamental matrix if \( F \) has a singular value decomposition (SVD): \( E = U \Sigma V^T \) with

\[
\Sigma = diag\{\sigma_1, \sigma_2, 0\}
\]

for some \( \sigma_1, \sigma_2 \in \mathbb{R}_+ \).
Outline

• 2-view geometry
• essential matrix, fundamental matrix
• properties
• estimation
Estimation (fundamental matrix)

Assume we have a corresponding pair of points: in noise-free case….

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} = 0 \iff \begin{bmatrix} xx' & xy' & x & yx' & yy' & y & x' & y' & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$
Given m point correspondences \((x_i, y_i)\) and \((x'_i, y'_i)\):

\[
\begin{bmatrix}
 x_1 x'_1 & x_1 y'_1 & x_1 & y_1 x'_1 & y_1 y'_1 & y_1 & x'_1 & y'_1 & 1 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
 x_m x'_m & x_m y'_m & x_m & y_m x'_m & y_m y'_m & y_m & x'_m & y'_m & 1 \\
\end{bmatrix}
\begin{bmatrix}
 F_{11} \\
 F_{12} \\
 F_{13} \\
 F_{21} \\
 F_{22} \\
 F_{23} \\
 F_{31} \\
 F_{32} \\
 F_{33} \\
\end{bmatrix} = 0
\]

\[AF(:) = 0\]
Estimation (fundamental matrix)

Given \( m \) point correspondences \((x_i, y_i)\) and \((x'_i, y'_i)\):

\[
\begin{bmatrix}
  x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
  x_mx'_m & x_my'_m & x_m & y_mx'_m & y_my'_m & y_m & x'_m & y'_m & 1 \\
\end{bmatrix}
\begin{bmatrix}
  F_{11} \\
  F_{12} \\
  F_{13} \\
  F_{21} \\
  F_{22} \\
  F_{23} \\
  F_{31} \\
  F_{32} \\
  F_{33}
\end{bmatrix} = 0
\]

\[ AF(\cdot) = 0 \]

noisy case: \( \min_{||F||=1} ||AF(\cdot)||^2 = \min_F \sum_i (x_i^T F x'_i)^2 \)

Is this reasonable error to minimize?
Recall: distance of point from a line

https://en.wikipedia.org/wiki/Distance_from_a_point_to_a_line

\[
\text{distance}(ax + by + c = 0, (x_0, y_0)) = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}.
\]

\(\mathbf{x}_i^T F \mathbf{x}_i\) is scaled euclidean distance of \((x'_i, y'_i)\) from line defined by \((x_i, y_i)\)
The eight-point algorithm

- **Meaning of error**: \( \sum_{i=1}^{N} (x_i^T F x'_i)^2 \): sum of squared distances between points \( x_i \) and epipolar lines \( F x'_i \) (or points \( x'_i \) and epipolar lines \( F^T x_i \)) multiplied by a scale factor
- **Nonlinear approach**: minimize

\[
\sum_{i=1}^{N} \left[ d^2(x_i, F x'_i) + d^2(x'_i, F^T x_i) \right]
\]
8-point algorithm

Longuet-Higgens

Given \( m \) point correspondences...

\[
\begin{bmatrix}
  x_1 x'_1 & x_1 y'_1 & x_1 & y_1 x'_1 & y_1 y'_1 & y_1 & x'_1 & y'_1 & 1 \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
  x_m x'_m & x_m y'_m & x_m & y_m x'_m & y_m y'_m & y_m & x'_m & y'_m & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
  F_{11} \\
  F_{12} \\
  F_{13} \\
  F_{21} \\
  F_{22} \\
  F_{23} \\
  F_{31} \\
  F_{32} \\
  F_{33}
\end{bmatrix} = 0
\]

Orders of magnitude difference
Between column of data matrix
\( \rightarrow \) least-squares yields poor results
“In Defence of the 8-point Algorithm”
(Hartley, PAMI’97)

Transform image to $[-1,1] \times [-1,1]$

\[
\begin{bmatrix}
\frac{2}{700} & 0 & -1 \\
\frac{2}{500} & -1 & 1
\end{bmatrix}
\]

SVD now produces good results

(0,0) (700,0) (1,1)
(0,500) (700,500) (-1,1)
(-1,-1) (1,-1)
Final “annoying” issue

Least squares solution won’t produce F that satisfies rank 2
(or rank-2 E with 2 identical singular values)

Solution: find the closest F/E (Frobenius norm) with SVD

\[ X = U \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} V^T \]

Closest fundamental matrix: set \( \sigma_3 = 0 \)

Closest essential matrix: set \( \sigma_3 = 0, \sigma = .5*(\sigma_1+\sigma_2) \)
Rank-2 Fundamental Matrix
7-point algorithm

Since $F$ are rank-deficient, we can estimate them with $m=7$ correspondences.

Given $m$ point correspondences…

Think: how many points do we need?

Unlike a homography, where each point correspondence contributes two constraints (rows in the linear system of equations), for estimating the essential/fundamental matrix, each point only contributes one constraint (row). [because the Longuet-Higgins / Epipolar constraint is a scalar eqn.]

Thus need at least 8 points.

Hence:
The Eight Point algorithm!

Solving Homogeneous Systems

Assume that we need the non-trivial solution of:

$$Ax = 0$$

with $m$ equations and $n$ unknowns, $m \geq n - 1$ and rank($A$) = $n-1$

Since the norm of $x$ is arbitrary, we will look for a solution with norm $||x|| = 1$

Self-study

Least Square solution

We want $Ax$ as close to 0 as possible and $||x|| = 1$:

Define the following cost:

This cost is called the LAGRANGIAN cost and $\lambda$ is called the LAGRANGIAN multiplier. The Lagrangian incorporates the constraints into the cost function by introducing extra variables.

Self-study

$$AF(:)=0$$

Idea: search for null vector of $A_{Mx9}$ that satisfies additional constraints (reshaped 3x3 matrix has 0 singular value)

1) $A$ is rank 7. Find 2 vectors that span null space of $A$, $F_1$ and $F_2$.

2) Find alpha such that $\text{Determinant}(\alpha F_1 + (1-\alpha) F_2) = 0$

[3rd order polynomial in alpha with at least one real solution]
Aside: what if cameras are calibrated?

Turns out we only need 5 points, but need to find roots to 10th degree polynomial

[Nister 04]
Recall: RANSAC

RANSAC loop:
1. Select feature pairs (at random)
2. Compute transformation $T$ (exact)
3. Compute *inliers* (point matches where $|p_i' - Tp_i|^2 < \varepsilon$)
4. Keep largest set of inliers
5. Re-compute least-squares estimate of transformation $T$ using all of the inliers
Fundamental matrix estimation with RANSAC
Outline

• 2-view geometry
• essential matrix, fundamental matrix
• properties
• estimation
• stereo
Three questions:

(i) **Correspondence geometry:** Given an image point $x$ in the first view, how does this constrain the position of the corresponding point $x'$ in the second image?

(ii) **Camera geometry (motion):** Given a set of corresponding image points $\{x_i \leftrightarrow x'_i\}$, $i=1,\ldots,n$, what are the cameras $P$ and $P'$ for the two views?

(iii) **Scene geometry (structure):** Given corresponding image points $x_i \leftrightarrow x'_i$ and cameras $P$, $P'$, what is the position of (their pre-image) $X$ in space?
Stereo
Basic Stereo Algorithm

For each epipolar line
  For each pixel in the left image
    • compare with every pixel on same epipolar line in right image
    • pick pixel with minimum match cost

Improvement: match *windows*
  • (Normalized) Correlation, Sum of Squared Difference (SSD), Sum of Absolute Differences (SAD), etc…
Triangulation for Rectified Stereo Pairs

Top-down view where world coordinates are centered between cameras

\[ \frac{x_L}{f} = \frac{X + b/2}{Z} \quad \frac{x_R}{f} = \frac{X - b/2}{Z} \quad \frac{y_L}{f} = \frac{y_R}{f} = \frac{Y}{Z} \]

\[ \Rightarrow \quad X = \frac{b(x_L + x_R)}{2(x_L - x_R)} \quad Y = \frac{b(y_L + y_R)}{2(x_L - x_R)} \quad Z = \frac{bf}{(x_L - x_R)} \]

\[ d = x_L - x_R = \frac{bf}{Z} \] is the disparity between corresponding left and right image points

- inverse proportional to depth \( Z \)
- disparity increases with baseline \( b \)
Disparity Maps

\[ d = x_L - x_R = \frac{bf}{Z} \]

Recall: Simple Stereo System

\[
\begin{align*}
X &\quad Z \\
Y &\quad \text{located at} \\
(0,0,0) &
\end{align*}
\]

left camera

\[
\begin{align*}
x &\quad x \\
y &\quad y \\
Z &\quad \text{located at} \\
(T_x,0,0) &
\end{align*}
\]

right camera

Recall: Stereo Disparity

Important equation!

Note: Depth and stereo disparity are inversely proportional

Stereo Example

From Middlebury stereo evaluation page

http://www.middlebury.edu/stereo/

Disparity values (0-64)

Disparity values (0-64)

Note how disparity is larger (brighter) for closer surfaces.

If we double the size of scene geometry and baseline, what happens to disparity?
Numerical stability

How do we characterize the error in depth $Z$ given an error in disparity $d$, in terms of scene + camera?

$$d = x_L - x_R = \frac{bf}{Z}$$

Scene + camera variables: $Z,f,b$

Dependant variable: $d = \text{function}(Z,f,b)$

1. Error increases quadratically with depth (hard to reconstruct far away points)
2. Error inversely proportional to baseline (larger baselines increase numerical stability)
Disparity maps (in practice)

- Small matching window (better localization)
- Large matching window (better detection)
Variational stereo

Penalize differences in nearby disparities (a “1-d” flow problem!)

$$\min_{u,v} E_{\text{intensity}} + E_{\text{smooth}}$$

$$E_{\text{intensity}}(d) = \int \int (I_2(x + d(x, y), y) - I_1(x, y))^2 dxdy$$

$$E_{\text{smooth}}(d) = \int \int \|\nabla d(x, y)\|^2 dxdy$$

1. Linearize $E_{\text{intensity}}$ term and solve with least squares

2. Add robust error terms $\rho(\cdot)$ to handle discontinuities
Coarse-to-fine stereo

Gaussian pyramid of image \( H \)

estimate disparities

upsample

estimate disparities

Gaussian pyramid of image \( I \)
Discrete disparity estimation

\[ z \in \{-5 \ldots 5\} \]

\[ \phi_i(z_i) = \rho(||I_2(x_i + z_i, y_i) - I(x_i, y_i)||) \]

\[ \psi_{ij}(z_i, z_j) = \rho(z_i - z_j) \]

\[ E(z) = \sum_{i \in V} \phi_i(z_i) + \sum_{i,j \in E} \psi_{ij}(z_i, z_j) \]

Solve with GraphCuts
Special case: single-scan-line consistency

Left Image

Right Image

Dissimilarity Values
(1-NCC) or SSD
Disparity Space Image (DSI)

Left scanline

Right scanline

Invalid entries due to constraint that disparity >= low value (0 in this case)

Invalid entries due to constraint that disparity <= high value (64 in this case)

However, I'm going to keep the full image around, including invalid values (I think it is easier to understand the pixel coordinates involved)

If we rearrange the diagonal band of valid values into a rectangular array (in this case of size 64 x N), that is what is traditionally known as the DSI coordinate in left scanline (e.g. N)
Representing the cost of all scanline correspondences
Ordering Constraint

We want one with lowest "cost" (Lowest sum of dissimilarity scores along the path).

We would like to choose the "best" path.

Constraints on Path

It is common to impose an ordering constraint on the path. Intuitively, the path is not allowed to "double back" on itself.

Ordering constraint... ...and its failure

An Optimal Scanline Strategy

• We want to find best path, taking into account ordering constraint and the possibility of occlusions.

Algorithm we will discuss now is from Cox, Hingorani, Rao, Maggs, "A Maximum Likelihood Stereo Algorithm," Computer Vision and Image Understanding, Vol 63(3), May 1996, pp.542-567.
Occlusions

Constraints on Path

It is common to impose an ordering constraint on the path. Intuitively, the path is not allowed to "double back" on itself.

Occlusions

Dealing with Occlusions

Left scanline

Right scanline

An Optimal Scanline Strategy

• We want to find best path, taking into account ordering constraint and the possibility of occlusions.

Algorithm we will discuss now is from Cox, Hingorani, Rao, Maggs, "A Maximum Likelihood Stereo Algorithm," Computer Vision and Image Understanding, Vol 63(3), May 1996, pp.542-567.
Occlusions

Lowest Cost Path

Want one with lowest "cost" (Lowest sum of dissimilarity scores along the path)

We would like to choose the "best" path.

Constraints on Path

It is common to impose an ordering constraint on the path. Intuitively, the path is not allowed to "double back" on itself.

Ordering Constraint

Dealing with Occlusions

An Optimal Scanline Strategy

• We want to find best path, taking into account ordering constraint and the possibility of occlusions.

Algorithm we will discuss now is from Cox, Hingorani, Rao, Maggs, "A Maximum Likelihood Stereo Algorithm," Computer Vision and Image Understanding, Vol 63(3), May 1996, pp.542-567.
Compute partial scanline costs

Three cases:
- Matching patches. Cost = dissimilarity score
- Occluded from right. Cost is some constant value.
- Occluded from left. Cost is some constant value.

\[
C(i,j) = \min\{\ C(i-1,j-1) + \text{dissimilarity}(i,j) \\
C(i-1,j) + \text{occlusionConstant}, \\
C(i,j-1) + \text{occlusionConstant}\};
\]

Dynamic Programming

Each pixel in DSI is now marked with a disparity value or occlusion label. In practice, enforce upper bound on disparity by computing diagonal band of DSI.
Results

Recap: want to find lowest cost path from upper left to lower right of DSI image. At each point on the path we have three choices: step left, step down, step diagonally. Each choice has a well-defined cost associated with it. This problem just screams out for Dynamic Programming! (which, indeed, is how Cox et.al. solve the problem)

Occlusion Filling

Simple trick for filling in gaps caused by occlusion. = left occluded Fill in left occluded pixels with value from the nearest valid pixel preceding it in the scanline. Similarly, for right occluded, look for valid pixel to the right.

Example

Result of DP alg with occlusion filling.

Result of DP alg. Black pixels = occluded.
Welcome to the Middlebury Stereo Vision Page, formerly located at www.middlebury.edu/stereo. This website accompanies our taxonomy and comparison of two-frame stereo correspondence algorithms [1]. It contains:

- An on-line evaluation of current algorithms
- Many stereo datasets with ground-truth disparities
- Our stereo correspondence software
- An on-line submission script that allows you to evaluate your stereo algorithm in our framework

How to cite the materials on this website:
We grant permission to use and publish all images and numerical results on this website. If you report performance results, we request that you cite our paper [1]. Instructions on how to cite our datasets are listed on the datasets page. If you want to cite this website, please use the URL “vision.middlebury.edu/stereo”.

References:
# Stereo—best algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Avg. Rank</th>
<th>Tsukuba ground truth</th>
<th>Venus ground truth</th>
<th>Teddy ground truth</th>
<th>Cones ground truth</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>nonocc</td>
<td>all</td>
<td>disc</td>
<td>nonocc</td>
</tr>
<tr>
<td>AdaptingBP [17]</td>
<td>2.8</td>
<td>1.11</td>
<td>1.37</td>
<td>5.79</td>
<td>0.10</td>
</tr>
<tr>
<td>DoubleBP2 [35]</td>
<td>2.9</td>
<td>0.88</td>
<td>1.29</td>
<td>4.76</td>
<td>0.13</td>
</tr>
<tr>
<td>DoubleBP [15]</td>
<td>4.9</td>
<td>0.88</td>
<td>1.29</td>
<td>4.76</td>
<td>0.15</td>
</tr>
<tr>
<td>SubPixDoubleBP [30]</td>
<td>5.6</td>
<td>1.24</td>
<td>1.76</td>
<td>5.98</td>
<td>0.12</td>
</tr>
<tr>
<td>AdaptOvrSegBP [33]</td>
<td>9.9</td>
<td>1.69</td>
<td>2.04</td>
<td>5.64</td>
<td>0.14</td>
</tr>
<tr>
<td>SymBP+occ [7]</td>
<td>10.8</td>
<td>0.97</td>
<td>1.75</td>
<td>5.09</td>
<td>0.16</td>
</tr>
<tr>
<td>PlaneFitBP [32]</td>
<td>10.8</td>
<td>0.97</td>
<td>1.83</td>
<td>5.26</td>
<td>0.17</td>
</tr>
<tr>
<td>AdaptDispCalib [36]</td>
<td>11.8</td>
<td>1.19</td>
<td>1.42</td>
<td>6.15</td>
<td>0.23</td>
</tr>
<tr>
<td>Segm+visib [4]</td>
<td>12.2</td>
<td>1.30</td>
<td>1.57</td>
<td>6.92</td>
<td>0.79</td>
</tr>
<tr>
<td>C-SemiGlob [19]</td>
<td>12.3</td>
<td>2.61</td>
<td>3.29</td>
<td>9.89</td>
<td>0.25</td>
</tr>
<tr>
<td>SO+orders [29]</td>
<td>12.8</td>
<td>1.29</td>
<td>1.79</td>
<td>6.83</td>
<td>0.25</td>
</tr>
<tr>
<td>DistinctSM [27]</td>
<td>14.1</td>
<td>1.21</td>
<td>1.75</td>
<td>6.39</td>
<td>0.16</td>
</tr>
<tr>
<td>CostAggr+occ [39]</td>
<td>14.3</td>
<td>1.38</td>
<td>1.96</td>
<td>7.14</td>
<td>0.44</td>
</tr>
<tr>
<td>OverSegmBP [26]</td>
<td>14.5</td>
<td>1.69</td>
<td>1.97</td>
<td>8.47</td>
<td>0.51</td>
</tr>
<tr>
<td>SegmentSupport [28]</td>
<td>15.1</td>
<td>1.25</td>
<td>1.62</td>
<td>6.68</td>
<td>0.25</td>
</tr>
<tr>
<td>RegionTreeBP [18]</td>
<td>15.7</td>
<td>1.39</td>
<td>1.64</td>
<td>6.85</td>
<td>0.22</td>
</tr>
<tr>
<td>EnhancedBP [24]</td>
<td>16.6</td>
<td>0.94</td>
<td>1.74</td>
<td>5.05</td>
<td>0.25</td>
</tr>
</tbody>
</table>
Outline

• 2-view geometry
• essential matrix, fundamental matrix
• properties
• estimation
• stereo
• multiview stereo
Dense multi view stereo

- Reconstruct the 3D position of the points corresponding to (all the) pixels in a set of images.
- Key assumption: We know the relative position, orientation, $K$, of all the cameras.
- **Number of cameras $>> 2$**
Trinocular stereo (version 0)

1. Pick 2 views, find correspondences
2. For each matching pair, reconstruct 3D point
3. If can’t find correspondence near projected location, reject

Version 1: generalize 3x3 fundamental matrix to a 3x3x3 trifocal tensor
(constraints points and lines across 3 images)
Multiview stereo (version 0)

- Pick one reference view
- For each point and for each candidate depth
  - keep depths with low SSD error in all other views
    (or any photoconsistency measure)

Problem: not all points are visible in all other views (occlusion and visibility major nuisance!)
Multiview stereo (version 1)

Hypothesize depths in a “smart” order where occluding points are found first.

Use knowledge of occluding points to smartly select view for photoconsistency check.

Store photoconsistent color in a 3D voxel grid (don’t need a reference image).

Reconstruct shape and appearance.
Speedup: plane sweeps

Validate voxels in a plane by computing their appearance in a virtual view using all N cameras.

Keep track of image-specific occlusion masks.

What is the transformation that warps image N to virtual view?
Voxel coloring
What about other camera setups?
Panoramic depth ordering

Layers radiate inwardly/outwardly

Seitz & Dyer
Space carving

Kutulakos & Seitz

Initialize voxel grid to all ‘1’s
Repeatedly choose a voxel on current surface:
  Project to visible images
  Carve out if not photoconsistent
Convergence

Consistency Property

- The resulting shape is photo-consistent
  > all inconsistent points are removed

Convergence Property

- Carving converges to a non-empty shape
  > a point on the true scene is never removed
Calibrated Image Acquisition

Calibrated Turntable

Selected Dinosaur Images

Selected Flower Images
Voxel Coloring Results

Dinosaur Reconstruction
72 K voxels colored
7.6 M voxels tested
7 min. to compute on a 250MHz SGI

Flower Reconstruction
70 K voxels colored
7.6 M voxels tested
7 min. to compute on a 250MHz SGI
Silhouette carving

Backproject binary silhouettes and find intersection
In limit of infinite cameras, this will produce convex hull reconstruction of object
Outline

• essential matrix, fundamental matrix
  (point-to-line correspondence, SVD properties)
• stereo
  (variational, discrete graph labelling, dynamic programming)
• multiview
  (volumetric models, visibility reasoning, patch-based methods)
Long-standing leader

Accurate, Dense, and Robust Multi-View Stereopsis

Yasutaka Furukawa and Jean Ponce, Fellow, IEEE

Easier to approximate surface by dense set of local planar patches

Patch-based Multiview Stereo (PMVS)
Pipeline: feature detection

Find sparse matches over pairs of images (using interest points + matching)
Triangulate to find sparse 3D points \{p\}
At each point $p$, estimate normal $N(p)$ and visibility $V_i(p)$ in each image using photometric consistency check (NCC over ~9x9 pixels).
Pipeline: patch expansion

Expand set of points \( \{ p \} \) by looking for hypothesizing 2D neighbors in visible images, backprojecting, and verifying photoconsistency.

Fig. 5. (a) Given an existing patch, an expansion procedure is performed to generate new ones for the neighboring empty image cells in its visible images. The expansion procedure is not performed for an image cell (b) if there already exists a neighboring patch reconstructed there, or (c) if there is a depth discontinuity when viewed from the camera. See text for more details.
Pipeline: filter out outlier patches

Fig. 7. The first filter enforces global visibility consistency to remove outliers (red patches). An arrow pointing from \( p_i \) to \( I_j \) represents a relationship \( I_j \in V(p_i) \). In both cases (left and right), \( U(p) \) denotes a set of patches that is inconsistent in visibility information with \( p \).
Pipeline: construct mesh

Convert set of 3D patches (*surfel* model) into polygonal mesh

Represent surface implicitly using a volumetric signed distance function
Solve differential equation that equates gradients of function to normals
Results

Fig. 16. Final mesh models: From left to right and top to bottom: roman, temple, dino, skull, face-1, face-2, body, city-hall, wall, fountain, brussels, steps-1, steps-2, steps-3, and castle datasets. Note that the mesh models are rendered from multiple viewpoints for fountain and castle datasets to show their overall structure.
Outline

• essential matrix, fundamental matrix
  (point-to-line correspondence, SVD properties)

• stereo
  (variational, discrete graph labelling, dynamic programming)

• multiview
  (volumetric models, visibility reasoning, patch-based methods)