Two-view geometry
Announcements

• Midterm due at the beginning of class today
• Project 3 will be assigned soon
  – Groups will default to same as Project 2
  – Tentative due date: 3/30
Reading

• Reading: Szeliski, Ch. 7.2
Fundamental matrix song

https://www.youtube.com/watch?v=DgGV3l82NTk
Back to stereo

- Where do epipolar lines come from?
Two-view geometry

• Where do epipolar lines come from?

3D point lies somewhere along $r$
Fundamental matrix

- This *epipolar geometry* of two views is described by a Very Special 3x3 matrix $\mathbf{F}$, called the *fundamental matrix*.
- $\mathbf{F}$ maps (homogeneous) *points* in image 1 to *lines* in image 2!
- The epipolar line (in image 2) of point $\mathbf{p}$ is: $\mathbf{Fp}$

- *Epipolar constraint* on corresponding points: $\mathbf{q}^T \mathbf{Fp} = 0$
Fundamental matrix

- Two Special points: $e_1$ and $e_2$ (the *epipoles*): projection of one camera into the other
Fundamental matrix

- Two Special points: $e_1$ and $e_2$ (the *epipoles*): projection of one camera into the other
- All of the epipolar lines in an image pass through the epipole
Epipoles
Fundamental matrix

- Why does $F$ exist?
- Let’s derive it...
Fundamental matrix – calibrated case

\( \mathbf{K}_1 \) : intrinsics of camera 1

\( \mathbf{K}_2 \) : intrinsics of camera 2

\( \mathbf{R} \) : rotation of image 2 w.r.t. camera 1

\( \tilde{\mathbf{p}} = \mathbf{K}_1^{-1} \mathbf{p} \) : ray through \( \mathbf{p} \) in camera 1’s (and world) coordinate system

\( \tilde{\mathbf{q}} = \mathbf{K}_2^{-1} \mathbf{q} \) : ray through \( \mathbf{q} \) in camera 2’s coordinate system
Fundamental matrix – calibrated case

• \( \tilde{p}, R^T \tilde{q}, \) and \( t \) are coplanar
• epipolar plane can be represented as \( t \times \tilde{p} \)

\[
(R^T \tilde{q})^T (t \times \tilde{p}) = 0
\]
Fundamental matrix – calibrated case

\[(R^T \tilde{q})^T (t \times \tilde{p}) = 0\]

\[\tilde{q}^T R (t \times \tilde{p}) = 0\]
Fundamental matrix – calibrated case

- One more substitution:
  - Cross product with $t$ can be represented as a 3x3 matrix

\[
[t]_\times = \begin{bmatrix}
  0 & -t_z & t_y \\
  t_z & 0 & -t_x \\
  -t_y & t_x & 0 \\
\end{bmatrix}
\]

\[t \times \tilde{p} = [t]_\times \tilde{p}\]
Fundamental matrix – calibrated case

\[ \tilde{q}^T R (t \times \tilde{p}) = 0 \]

\[ \tilde{q}^T R \begin{bmatrix} t \end{bmatrix} \times \tilde{p} = 0 \]
Fundamental matrix – calibrated case

\[ \hat{p} = K_1^{-1} p \]: ray through \( p \) in camera 1’s (and world) coordinate system

\[ \hat{q} = K_2^{-1} q \]: ray through \( q \) in camera 2’s coordinate system

\[ \hat{q}^T R [t] \times \hat{p} = 0 \]

\[ \hat{q}^T E \hat{p} = 0 \]

\( E \) → the Essential matrix
Cross-product as linear operator

**Useful fact:** Cross product with a vector \( \mathbf{t} \) can be represented as multiplication with a \textit{(skew-symmetric)} 3x3 matrix

\[
\begin{bmatrix}
0 & -t_z & t_y \\
t_z & 0 & -t_x \\
-t_y & t_x & 0
\end{bmatrix}
\]

\[\mathbf{t} \times \tilde{\mathbf{p}} = [\mathbf{t}] \times \tilde{\mathbf{p}}\]
Fundamental matrix – uncalibrated case

\[
q^T K_2^{-T} R [t] \times K_1^{-1} p = 0
\]

\[F\] \rightarrow \text{the Fundamental matrix}
Properties of the Fundamental Matrix

• $Fp$ is the epipolar line associated with $p$

• $F^T q$ is the epipolar line associated with $q$

• $Fe_1 = 0$ and $F^T e_2 = 0$

• $F$ is rank 2

• How many parameters does $F$ have?
Rectified case

\[ \mathbf{R} = \mathbf{I}_{3 \times 3} \]
\[ \mathbf{t} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \]
\[ \mathbf{E} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \]
Stereo image rectification

- reproject image planes onto a common plane parallel to the line between optical centers
- pixel motion is horizontal after this transformation
- two homographies (3x3 transform), one for each input image reprojection

Relationship with homography?

Images taken from the same center of projection? Use a homography!
Questions?
Estimating F

• If we don’t know $K_1$, $K_2$, $R$, or $t$, can we estimate $F$ for two images?

• Yes, given enough correspondences
Estimating F – 8-point algorithm

• The fundamental matrix F is defined by

\[ x'\mathbf{T}Fx = 0 \]

for any pair of matches \( x \) and \( x' \) in two images.

• Let \( x=(u,v,1)^T \) and \( x'=(u',v',1)^T \),

\[
\mathbf{F} = \begin{bmatrix}
 f_{11} & f_{12} & f_{13} \\
 f_{21} & f_{22} & f_{23} \\
 f_{31} & f_{32} & f_{33}
\end{bmatrix}
\]

each match gives a linear equation

\[
uu' f_{11} + vu' f_{12} + u' f_{13} + uv' f_{21} + vv' f_{22} + v' f_{23} + uf_{31} + vf_{32} + f_{33} = 0\]
8-point algorithm

\[
\begin{bmatrix}
u_1 u_1 & v_1 u_1 & u_1 & u_1 v_1 & v_1 v_1 & v_1 & u_1 & v_1 & 1 \\
u_2 u_2 & v_2 u_2 & u_2 & u_2 v_2 & v_2 v_2 & v_2 & u_2 & v_2 & 1 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
u_n u_n & v_n u_n & u_n & u_n v_n & v_n v_n & v_n & u_n & v_n & 1 \\
\end{bmatrix}
\begin{bmatrix}
f_{11} \\
f_{12} \\
f_{13} \\
f_{21} \\
f_{22} \\
f_{23} \\
f_{31} \\
f_{32} \\
f_{33} \\
\end{bmatrix} = 0
\]

- Like with homographies, instead of solving \(A f = 0\), we seek \(f\) to minimize \(\|A f\|\), least eigenvector of \(A^T A\).
8-point algorithm – Problem?

• $\mathbf{F}$ should have rank 2
• To enforce that $\mathbf{F}$ is of rank 2, $\mathbf{F}$ is replaced by $\mathbf{F}'$ that minimizes $\|\mathbf{F} - \mathbf{F}'\|$ subject to the rank constraint.

• This is achieved by SVD. Let $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^T$, where

$$
\Sigma = \begin{bmatrix}
\sigma_1 & 0 & 0 \\
0 & \sigma_2 & 0 \\
0 & 0 & \sigma_3 \\
\end{bmatrix}
$$

let

$$
\Sigma' = \begin{bmatrix}
\sigma_1 & 0 & 0 \\
0 & \sigma_2 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}
$$

then $\mathbf{F}' = \mathbf{U}\Sigma'\mathbf{V}^T$ is the solution.
8-point algorithm

% Build the constraint matrix
A = [x2(1,:)'.*x1(1,:)'  x2(1,:)'.*x1(2,:)'  x2(1,:)' ... 
x2(2,:)'.*x1(1,:)'  x2(2,:)'.*x1(2,:)'  x2(2,:)' ... 
x1(1,:)'  x1(2,:)'  ones(npts,1) ];

[U,D,V] = svd(A);

% Extract fundamental matrix from the column of V
% corresponding to the smallest singular value.
F = reshape(V(:,9),3,3)';

% Enforce rank2 constraint
[U,D,V] = svd(F);
F = U*diag([D(1,1) D(2,2) 0])*V';
8-point algorithm

- Pros: it is linear, easy to implement and fast
- Cons: susceptible to noise
Problem with 8-point algorithm

\[
\begin{bmatrix}
    u_1u_1' & v_1u_1' & u_1' & u_1v_1' & v_1v_1' & v_1' & u_1 & v_1 & 1 \\
u_2u_2' & v_2u_2' & u_2' & u_2v_2' & v_2v_2' & v_2' & u_2 & v_2 & 1 \\
    \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
u_nu_n' & v_nu_n' & u_n' & u_nv_n' & v_nv_n' & v_n' & u_n & v_n & 1 \\
\end{bmatrix}
\begin{bmatrix}
f_{11} \\
f_{12} \\
f_{13} \\
f_{21} \\
f_{22} \\
f_{23} \\
f_{31} \\
f_{32} \\
f_{33}
\end{bmatrix} = 0
\]

Orders of magnitude difference between column of data matrix → least-squares yields poor results
Normalized 8-point algorithm

normalized least squares yields good results

Transform image to ~[-1,1]x[-1,1]
Normalized 8-point algorithm

1. Transform input by $\hat{x}_i = Tx_i$, $\hat{x}'_i = Tx'_i$
2. Call 8-point on $\hat{x}_i$, $\hat{x}'_i$ to obtain $\hat{F}$
3. $F = T'^T\hat{F}T$

$$x'^TFx = 0$$

$$\hat{x}'^TT'^{-T}\hat{F}T^{-1}\hat{x} = 0$$
Normalized 8-point algorithm

\[
A = [x2(1,:)' \cdot x1(1,:)', x2(1,:)'.*x1(2,:)', x2(1,:)' \ldots
x2(2,:)' \cdot x1(1,:), x2(2,:)'.*x1(2,:), x2(2,:)' \ldots
x1(1,:), x1(2,:)'
ones(npts,1) ];
\]

\[
[U,D,V] = \text{svd}(A);
\]

\[
F = \text{reshape}(V(:,9), 3, 3)';
\]

\[
[U,D,V] = \text{svd}(F);
F = U*\text{diag}([[D(1,1) D(2,2) 0]])*V';
\]

\% Denormalise
F = T2'*F*T1;
Results (ground truth)

**Ground truth** with standard stereo calibration
Results (8-point algorithm)
Results (normalized 8-point algorithm)
What about more than two views?

• The geometry of three views is described by a 3 x 3 x 3 tensor called the *trifocal tensor*

• The geometry of four views is described by a 3 x 3 x 3 x 3 tensor called the *quadrifocal tensor*

• After this it starts to get complicated...
Large-scale structure from motion

Dubrovnik, Croatia. 4,619 images (out of an initial 57,845).
Total reconstruction time: 23 hours
Number of cores: 352