Lecture 14: Two-view geometry
Announcements

• Midterm graded
  – Total: 95 points
  – Mean: 79.4
  – Median: 82

• Project 3 released
Announcements

• Reading: Szeliski, Ch. 7.2
Fundamental matrix song

https://www.youtube.com/watch?v=DgGV3l82NTk
Epipolar geometry
Rectified case

- Images have the same orientation, \( t \) parallel to image planes
- Where are the epipoles?
Fundamental matrix – calibrated case

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

\[ \text{the Essential matrix} \]

\[ \tilde{q}^T E \tilde{p} = 0 \]
Fundamental matrix – uncalibrated case

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

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

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

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

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

• $F e_1 = 0$ 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

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

\[ x'^T F x = 0 \]

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

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

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

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

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

- In reality, 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

\[
\Sigma = \begin{bmatrix}
\sigma_1 & 0 & 0 \\
0 & \sigma_2 & 0 \\
0 & 0 & \sigma_3 \\
\end{bmatrix}, \quad \text{let} \quad \Sigma' = \begin{bmatrix}
\sigma_1 & 0 & 0 \\
0 & \sigma_2 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}
\]

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

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

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. $F = T^T \hat{F} T$

$$x'^T F x = 0$$

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

\[
[x1, T1] = normalise2dpts(x1);
[x2, T2] = normalise2dpts(x2);

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);

F = reshape(V(:,9),3,3)';

[U,D,V] = svd(F);
F = U*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 $3 \times 3 \times 3$ tensor called the *trifocal tensor*.

• The geometry of four views is described by a $3 \times 3 \times 3 \times 3$ 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