Computer Vision - Exercise 6

Eight-point Algorithm, RANSAC, Triangulation

19.01.2010

Tobias Weyand
RWTH Aachen
http://www.mmp.rwth-aachen.de

weyand@umic.rwth-aachen.de
Epipolar Geometry: Uncalibrated Case

\[ \hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K^{'-1} \]
Epipolar Geometry: Uncalibrated Case

\[
\hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1}
\]
Epipolar Geometry: Uncalibrated Case

\[ \hat{x}^T E \hat{x}' = 0 \quad \rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K'^{-T} E K'^{-1} \]

- \( F x' \) is the epipolar line associated with \( x' \) (\( l = F x' \) )
Epipolar Geometry: Uncalibrated Case

\[ \hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1} \]

- \( F x' \) is the epipolar line associated with \( x' \) \((l = F x')\)
- \( F^T x \) is the epipolar line associated with \( x \) \((l' = F^T x)\)
Epipolar Geometry: Uncalibrated Case

\[ \hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1} \]

- \( F x' \) is the epipolar line associated with \( x' \) (\( l = F x' \))
- \( F^T x \) is the epipolar line associated with \( x \) (\( l' = F^T x \))
- \( F e' = 0 \) and \( F^T e = 0 \)
Epipolar Geometry: Uncalibrated Case

\[ \hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1} \]

- $F x'$ is the epipolar line associated with $x'$ ($l = F x'$)
- $F^T x$ is the epipolar line associated with $x$ ($l' = F^T x$)
- $F e' = 0$ and $F^T e = 0$
- $F$ is singular (rank two)

Slide credit: Svetlana Lazebnik
Epipolar Geometry: Uncalibrated Case

\[ \hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1} \]

- \( F x' \) is the epipolar line associated with \( x' \) (\( l = F x' \))
- \( F^T x \) is the epipolar line associated with \( x \) (\( l' = F^T x \))
- \( F e' = 0 \) and \( F^T e = 0 \)
- \( F \) is singular (rank two)
- \( F \) has seven degrees of freedom
Recap: The Eight-Point Algorithm

\[ x = (u, v, 1)^T, \quad x' = (u', v', 1)^T \]

\[
(u, v, 1) \begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0
\]
Recap: The Eight-Point Algorithm

\[ x = (u, v, 1)^T, \quad x' = (u', v', 1)^T \]

\[
\begin{pmatrix}
F_{11} & F_{12} & F_{13} \\
F_{21} & F_{22} & F_{23} \\
F_{31} & F_{32} & F_{33}
\end{pmatrix}
\begin{pmatrix}
u \\ v'
\end{pmatrix}
= 0
\]

\[
\begin{pmatrix}
u u' \\ u v' \\ u \\ v \\ v'
\end{pmatrix}
= 0
\]
Recap: The Eight-Point Algorithm

\[ x = (u, v, 1)^T, \quad x' = (u', v', 1)^T \]

\[
\begin{pmatrix}
F_{11} & F_{12} & F_{13} \\
F_{21} & F_{22} & F_{23} \\
F_{31} & F_{32} & F_{33}
\end{pmatrix}
\begin{pmatrix}
u' \\
v'
\end{pmatrix}
= 0
\]

\[
\begin{pmatrix}
u u' \\
v v' \\
v u' \\
v v' \\
v u' \\
v v' \\
v u' \\
v v' \\
u u' \\
v v' \\
v u' \\
v v' \\
u u' \\
v v' \end{pmatrix}
\begin{pmatrix}
F_{11} \\
F_{12} \\
F_{13} \\
F_{21} \\
F_{22} \\
F_{23} \\
F_{31} \\
F_{32} \\
F_{33}
\end{pmatrix}
= 0
\]
Recap: The Eight-Point Algorithm

\[ x = (u, v, 1)^T, \quad x' = (u', v', 1)^T \]

\[
\begin{pmatrix}
F_{11} & F_{12} & F_{13} \\
F_{21} & F_{22} & F_{23} \\
F_{31} & F_{32} & F_{33}
\end{pmatrix}
\begin{pmatrix}
u' \\
v''
\end{pmatrix} = 0
\]

1.) Solve with SVD. This minimizes \( \sum_{i=1}^{N} (x_i^T F x_i')^2 \)

2.) Enforce rank-2 constraint using SVD
Recap: The Eight-Point Algorithm

\[ x = (u, v, 1)^T, \quad x' = (u', v', 1)^T \]

\[
\begin{pmatrix}
F_{11} & F_{12} & F_{13} \\
F_{21} & F_{22} & F_{23} \\
F_{31} & F_{32} & F_{33}
\end{pmatrix}
\begin{pmatrix}
u' \\
v'
\end{pmatrix}
= 0
\]

1.) Solve with SVD. This minimizes

\[
\sum_{i=1}^{N} (x_i^T F x_i')^2
\]

2.) Enforce rank-2 constraint using SVD

Problem: poor numerical conditioning
Problem with the Eight-Point Algorithm

\[
\begin{bmatrix}
250906.36 & 183269.57 & 921.81 & 200931.10 & 146766.13 & 738.21 & 272.19 & 198.81 \\
2692.28 & 131633.03 & 176.27 & 6196.73 & 302975.59 & 405.71 & 15.27 & 746.79 \\
416374.23 & 871684.30 & 935.47 & 408110.89 & 854384.92 & 916.90 & 445.10 & 931.81 \\
191183.60 & 171759.40 & 410.27 & 416435.62 & 374125.90 & 893.65 & 465.99 & 418.65 \\
48988.86 & 30401.76 & 57.89 & 298604.57 & 185309.58 & 352.87 & 846.22 & 525.15 \\
116407.01 & 2727.75 & 138.89 & 169941.27 & 3982.21 & 202.77 & 838.12 & 19.64 \\
135384.58 & 75411.13 & 198.72 & 411350.03 & 229127.78 & 603.79 & 681.28 & 379.48
\end{bmatrix}
\begin{bmatrix}
F_{11} \\
F_{12} \\
F_{13} \\
F_{21} \\
F_{22} \\
F_{23} \\
F_{31} \\
F_{32} \\
F_{33}
\end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
\]
### Problem with the Eight-Point Algorithm

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>250906.36</td>
<td>183269.57</td>
<td>921.81</td>
<td>200931.10</td>
<td>146766.13</td>
<td>738.21</td>
<td>272.19</td>
</tr>
<tr>
<td>2692.28</td>
<td>131633.03</td>
<td>176.27</td>
<td>6196.73</td>
<td>302975.59</td>
<td>405.71</td>
<td>15.27</td>
</tr>
<tr>
<td>416374.23</td>
<td>871684.30</td>
<td>935.47</td>
<td>408110.89</td>
<td>854384.92</td>
<td>916.90</td>
<td>445.10</td>
</tr>
<tr>
<td>191183.60</td>
<td>171759.40</td>
<td>410.27</td>
<td>416435.62</td>
<td>374125.90</td>
<td>893.65</td>
<td>465.99</td>
</tr>
<tr>
<td>48988.86</td>
<td>30401.76</td>
<td>57.89</td>
<td>298604.57</td>
<td>185309.58</td>
<td>352.87</td>
<td>846.22</td>
</tr>
<tr>
<td>164786.04</td>
<td>546559.67</td>
<td>813.17</td>
<td>1998.37</td>
<td>6628.15</td>
<td>9.86</td>
<td>202.65</td>
</tr>
<tr>
<td>116407.01</td>
<td>2727.75</td>
<td>138.89</td>
<td>169941.27</td>
<td>3982.21</td>
<td>202.77</td>
<td>838.12</td>
</tr>
<tr>
<td>135384.58</td>
<td>75411.13</td>
<td>198.72</td>
<td>411350.03</td>
<td>229127.78</td>
<td>603.79</td>
<td>681.28</td>
</tr>
</tbody>
</table>

\[
\begin{bmatrix}
F_{11} \\
F_{12} \\
F_{13} \\
F_{21} \\
F_{22} \\
F_{23} \\
F_{31} \\
F_{32} \\
F_{33}
\end{bmatrix}
\begin{bmatrix}
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1
\end{bmatrix} = 0
\]
Problem with the Eight-Point Algorithm

- Poor numerical conditioning
Problem with the Eight-Point Algorithm

- Poor numerical conditioning
- Can be fixed by rescaling the data
Recap: Normalized Eight-Point Alg.
Recap: Normalized Eight-Point Alg.

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.

[Hartley, 1995]
Recap: Normalized Eight-Point Alg.

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.

2. Use the eight-point algorithm to compute $F$ from the normalized points.
Recap: Normalized Eight-Point Alg.

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.
2. Use the eight-point algorithm to compute $F$ from the normalized points.
3. Enforce the rank-2 constraint using SVD.
Recap: Normalized Eight-Point Alg.

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.

2. Use the eight-point algorithm to compute $F$ from the normalized points.

3. Enforce the rank-2 constraint using SVD.

$$F = U D V^T = U \begin{bmatrix} d_{11} & \cdot & \cdot & \cdot \\ \cdot & d_{22} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & d_{33} \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & v_{13} \\ \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots \\ v_{31} & \cdots & v_{33} \end{bmatrix}^T$$

[Hartley, 1995]
Recap: Normalized Eight-Point Alg.

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.
2. Use the eight-point algorithm to compute $F$ from the normalized points.
3. Enforce the rank-2 constraint using SVD.

$$F = UDV^T = U \begin{bmatrix} d_{11} & d_{22} & d_{33} \\ \vdots & \vdots & \vdots \\ d_{33} & d_{33} & d_{33} \end{bmatrix} \begin{bmatrix} \nu_{11} & \cdots & \nu_{13} \\ \vdots & \cdots & \vdots \\ \nu_{31} & \cdots & \nu_{33} \end{bmatrix}^T$$

Set $d_{33}$ to zero and reconstruct $F$
Recap: Normalized Eight-Point Alg.

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.

2. Use the eight-point algorithm to compute $F$ from the normalized points.

3. Enforce the rank-2 constraint using SVD.

\[
F = UDV^T = U \begin{bmatrix} d_{11} & d_{12} & \ldots \\ d_{21} & d_{22} & \ldots \\ \vdots & \vdots & \ddots \\ d_{31} & \ldots & d_{33} \end{bmatrix} \left[ \begin{array}{c} v_{11} \\ \vdots \\ v_{31} \end{array} \right] \left[ \begin{array}{ccc} v_{11} & \cdots & v_{13} \\ \vdots & \ddots & \vdots \\ v_{31} & \cdots & v_{33} \end{array} \right]^T
\]

Set $d_{33}$ to zero and reconstruct $F$.

4. Transform fundamental matrix back to original units: if $T$ and $T'$ are the normalizing transformations in the two images, than the fundamental matrix in original coordinates is $T^T F T'$.

[Hartley, 1995]
### Comparison of Estimation Algorithms

<table>
<thead>
<tr>
<th></th>
<th>8-point</th>
<th>Normalized 8-point</th>
<th>Nonlinear least squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>Av. Dist. 1</td>
<td>2.33 pixels</td>
<td>0.92 pixel</td>
<td>0.86 pixel</td>
</tr>
<tr>
<td>Av. Dist. 2</td>
<td>2.18 pixels</td>
<td>0.85 pixel</td>
<td>0.80 pixel</td>
</tr>
</tbody>
</table>

Slide credit: Svetlana Lazebnik
Recap: RANSAC

RANSAC loop:
1. Randomly select a seed group of points on which to base transformation estimate (e.g., a group of matches)
2. Compute transformation from seed group
3. Find inliers to this transformation
4. If the number of inliers is sufficiently large, re-compute least-squares estimate of transformation on all of the inliers
   - Keep the transformation with the largest number of inliers

Slide credit: Kristen Grauman

B. Leibe
Tuesday, January 26, 2010
RANSAC Line Fitting Example

- Task: Estimate the best line
RANSAC Line Fitting Example

- Task: Estimate the best line

Sample two points
RANSAC Line Fitting Example

- Task: Estimate the best line

Fit a line to them
RANSAC Line Fitting Example

- Task: Estimate the best line

Total number of points within a threshold of line.
RANSAC Line Fitting Example

- Task: Estimate the best line

Repeat, until we get a good result.
RANSAC Line Fitting Example

- Task: Estimate the best line

Repeat, until we get a good result.
RANSAC Line Fitting Example

- Task: Estimate the best line

Repeat, until we get a good result.
Revisiting Triangulation

- Given projections of a 3D point in two or more images (with known camera matrices), find the coordinates of the point
Revisiting Triangulation

- We want to intersect the two visual rays corresponding to $x_1$ and $x_2$, but because of noise and numerical errors, they will never meet exactly.
Triangulation - Linear Algebraic Approach

Two independent equations each in terms of three unknown entries of $X$.

- Stack equations and solve with SVD.
- This approach nicely generalizes to multiple cameras.

\[
\begin{align*}
\lambda_1 x_1 &= P_1 X \\
\lambda_2 x_2 &= P_2 X
\end{align*}
\]

\[
\begin{align*}
x_1 \times P_1 X &= 0 \\
x_2 \times P_2 X &= 0
\end{align*}
\]

\[
\begin{align*}
[x_{1x}] P_1 X &= 0 \\
[x_{2x}] P_2 X &= 0
\end{align*}
\]