Structure from motion

Digital Visual Effects

Yung-Yu Chuang

with slides by Richard Szeliski, Steve Seitz, Zhengyou Zhang and Marc Pollefeys
Outline

• Epipolar geometry and fundamental matrix
• Structure from motion
• Factorization method
• Bundle adjustment
• Applications
Epipolar geometry & fundamental matrix
The epipolar geometry

$C, C', x, x'$ and $X$ are coplanar
The epipolar geometry

What if only $C, C', x$ are known?
The epipolar geometry

All points on $\pi$ project on $l$ and $l'$
The epipolar geometry

Family of planes $\pi$ and lines $l$ and $l'$ intersect at $e$ and $e'$. 
The epipolar geometry

epipolar pole = intersection of baseline with image plane
epipolar pole = projection of projection center in the other image

epipolar plane = plane containing baseline
epipolar line = intersection of epipolar plane with image
The fundamental matrix $F$

Two reference frames are related via the extrinsic parameters

$$p = Rp' + T$$
The fundamental matrix $F$

$p = Rp' + T$

Multiply both sides by $p^T[T]_x$

$[T]_x = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix}$

$p^T[T]_x p = p^T[T]_x (Rp' + T)$

$0 = p^T[T]_x Rp'$

$E p' = 0$ essential matrix
The fundamental matrix $F$

$$p^T E p' = 0$$

Let $M$ and $M'$ be the intrinsic matrices, then

$$p = M^{-1} x \quad p' = M'^{-1} x'$$

$$\Rightarrow (M^{-1} x)^T E (M'^{-1} x') = 0$$

$$\Rightarrow x^T M^{-T} E M'^{-1} x' = 0$$

$$\Rightarrow x^T F x' = 0 \quad \text{fundamental matrix}$$
The fundamental matrix $F$

- The fundamental matrix is the algebraic representation of epipolar geometry.

- The fundamental matrix satisfies the condition that for any pair of corresponding points $x \leftrightarrow x'$ in the two images:

\[
x^T F x' = 0 \quad (x^T 1 = 0)
\]
The fundamental matrix $F$

$F$ is the unique $3x3$ rank 2 matrix that satisfies $x^TFx'=0$
for all $x\leftrightarrow x'$

1. **Transpose:** if $F$ is fundamental matrix for $(x,x')$, then $F^T$
is fundamental matrix for $(x',x)$
2. **Epipolar lines:** $l=Fx'$ & $l'=F^Tx$
3. **Epipoles:** on all epipolar lines, thus $e^TFx'=0$, $\forall x'$
   $\Rightarrow e^TF=0$, similarly $Fe'=0$
4. $F$ has 7 d.o.f., i.e. $3x3-1($homogeneous$)-1($rank2$)$
5. $F$ is a correlation, projective mapping from a point $x$ to
   a line $l=Fx'$ (not a proper correlation, i.e. not invertible)
The fundamental matrix $F$

- It can be used for
  - Simplifies matching
  - Allows to detect wrong matches
Estimation of $F$ — 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

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

$$\begin{bmatrix}
    u_1 u'_1 & u_1 v'_1 & u_1 & v_1 u'_1 & v_1 v'_1 & v_1 & u'_1 & v'_1 & 1 \\
    u_2 u'_2 & u_2 v'_2 & u_2 & v_2 u'_2 & v_2 v'_2 & v_2 & u'_2 & v'_2 & 1 \\
    \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
    u_n u'_n & u_n v'_n & u_n & v_n u'_n & v_n v'_n & v_n & u'_n & v'_n & 1
\end{bmatrix} \begin{bmatrix}
    f_{11} \\
    f_{12} \\
    f_{13} \\
    f_{21} \\
    f_{22} \\
    f_{23} \\
    f_{31} \\
    f_{32} \\
    f_{33}
\end{bmatrix} = 0$$

- In reality, instead of solving $A \mathbf{f} = 0$, we seek $\mathbf{f}$ to minimize $\|A \mathbf{f}\|$ subj. $\|\mathbf{f}\| = 1$. Find the vector corresponding to the least singular value.
8-point algorithm

- To enforce that \( F \) is of rank 2, \( F \) is replaced by \( F' \) that minimizes \( \| F - F' \| \) subject to \( \det F' = 0 \).

- It is achieved by SVD. Let \( F = U\Sigma V^T \), 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 \( F' = U\Sigma' V^T \) 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

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

1. Transform input by $\hat{x}_i = Tx_i$, $\hat{x}_i' = T'x_i'$
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

Normalized least squares yields good results
Transform image to $\sim[-1,1] \times [-1,1]$
Normalized 8-point algorithm

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

\[
A = \begin{bmatrix}
  x_2(1,:)' \times x_1(1,:)' & x_2(1,:)' \times x_1(2,:)' & x_2(1,:)' \\
  x_2(2,:)' \times x_1(1,:)' & x_2(2,:)' \times x_1(2,:)' & x_2(2,:)' \\
  & x_1(1,:)' & x_1(2,:)' & \text{ones}(npts,1)
\end{bmatrix};
\]

[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;
Normalization

function [newpts, T] = normalise2dpts(pts)

    c = mean(pts(1:2,:))'; % Centroid
    newp(1,:) = pts(1,:)-c(1); % Shift origin to centroid.
    newp(2,:) = pts(2,:)-c(2);

    meandist = mean(sqrt(newp(1,:).^2 + newp(2,:).^2));
    scale = sqrt(2)/meandist;

    T = [scale 0 -scale*c(1)
          0 scale -scale*c(2)
          0 0 1 ];
    newpts = T*pts;
RANSAC

repeat
  select minimal sample (8 matches)
  compute solution(s) for F
  determine inliers
until $\Gamma(#\text{inliers},#\text{samples}) > 95\%$ or too many times
compute F based on all inliers
Results (ground truth)

- **Ground truth** with standard stereo calibration
Results (8-point algorithm)
Results (normalized 8-point algorithm)

- Normalized 8-point algorithm
Structure from motion
Structure from motion

structure for motion: automatic recovery of **camera motion** and **scene structure** from two or more images. It is a self calibration technique and called **automatic camera tracking** or **matchmoving**.
Applications

- For computer vision, multiple-view shape reconstruction, novel view synthesis and autonomous vehicle navigation.
- For film production, seamless insertion of CGI into live-action backgrounds
Matchmove

element #1  example #2  example #3
Structure from motion

2D feature tracking → 3D estimation → optimization (bundle adjust) → geometry fitting

SFM pipeline
Structure from motion

• Step 1: Track Features
  - Detect good features, Shi & Tomasi, SIFT
  - Find correspondences between frames
    • Lucas & Kanade-style motion estimation
    • window-based correlation
    • SIFT matching
KLT tracking

http://www.ces.clemson.edu/~stb/klt/
Structure from Motion

• Step 2: Estimate Motion and Structure
  - Simplified projection model, e.g., [Tomasi 92]
  - 2 or 3 views at a time [Hartley 00]
Structure from Motion

- Step 3: Refine estimates
  - “Bundle adjustment” in photogrammetry
  - Other iterative methods
Structure from Motion

• Step 4: Recover surfaces (image-based triangulation, silhouettes, stereo...)

[Images of a building and室内场景的3D建模]
Factorization methods
Problem statement
Notations

- $n$ 3D points are seen in $m$ views
- $q=(u,v,1)$: 2D image point
- $p=(x,y,z,1)$: 3D scene point
- $\Pi$: projection matrix
- $\pi$: projection function
- $q_{ij}$ is the projection of the $i$-th point on image $j$
- $\lambda_{ij}$ projective depth of $q_{ij}$

$$q_{ij} = \pi(\Pi_j p_i) \quad \pi(x, y, z) = \left(\frac{x}{z}, \frac{y}{z}\right)$$

$$\lambda_{ij} = z$$
Structure from motion

- Estimate $\prod_j$ and $p_i$ to minimize

$$\varepsilon(\Pi_1, \cdots, \Pi_m, p_1, \cdots, p_n) = \sum_{j=1}^{m} \sum_{i=1}^{n} w_{ij} \log P(\pi(\Pi_j p_i); q_{ij})$$

$$w_{ij} = \begin{cases} 1 & \text{if } p_i \text{ is visible in view } j \\ 0 & \text{otherwise} \end{cases}$$

- Assume isotropic Gaussian noise, it is reduced to

$$\varepsilon(\Pi_1, \cdots, \Pi_m, p_1, \cdots, p_n) = \sum_{j=1}^{m} \sum_{i=1}^{n} w_{ij} \left\| \pi(\Pi_j p_i) - q_{ij} \right\|^2$$

- Start from a simpler projection model
Orthographic projection

- Special case of perspective projection
  - Distance from the COP to the PP is infinite

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
= \begin{bmatrix}
x \\
y \\
1
\end{bmatrix} \Rightarrow (x, y)
\]

- Also called “parallel projection”: \((x, y, z) \rightarrow (x, y)\)
SFM under orthographic projection

2D image point → Orthographic projection incorporating 3D rotation → 3D scene point → image offset

\[ q = \Pi p + t \]

2×1 2×3 3×1 2×1

• Trick
  - Choose scene origin to be centroid of 3D points
  - Choose image origins to be centroid of 2D points
  - Allows us to drop the camera translation:

\[ q = \Pi p \]
factorization (Tomasi & Kanade)

projection of \( n \) features in one image:

\[
\begin{bmatrix}
q_1 & q_2 & \cdots & q_n
\end{bmatrix}_{2 \times n} = \prod_{i=1}^{3} \begin{bmatrix}
p_i & p_2 & \cdots & p_n
\end{bmatrix}_{3 \times n}
\]

projection of \( n \) features in \( m \) images

\[
\begin{bmatrix}
q_{11} & q_{12} & \cdots & q_{1n} \\
q_{21} & q_{22} & \cdots & q_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
q_{m1} & q_{m2} & \cdots & q_{mn}
\end{bmatrix}_{2m \times n} = \prod_{i=1}^{m} \begin{bmatrix}
\Pi_i \\
\Pi_i \\
\vdots \\
\Pi_i
\end{bmatrix}_{2m \times 3} \begin{bmatrix}
p_1 & p_2 & \cdots & p_n
\end{bmatrix}_{3 \times n}
\]

\( W \) measurement  \( M \) motion  \( S \) shape

Key Observation: \( \text{rank}(W) \leq 3 \)
Factorization Technique

- \( W \) is at most rank 3 (assuming no noise)
- We can use singular value decomposition to factor \( W \):

\[
W = M' S'
\]

- \( S' \) differs from \( S \) by a linear transformation \( A \):

\[
W = M'S' = (MA^{-1})(AS)
\]
- Solve for \( A \) by enforcing metric constraints on \( M \)
Results
Extensions to factorization methods

- Projective projection
- With missing data
- Projective projection with missing data
Bundle adjustment
**Bundle adjustment**

- $n$ 3D points are seen in $m$ views
- $x_{ij}$ is the projection of the $i$-th point on image $j$
- $a_j$ is the parameters for the $j$-th camera
- $b_i$ is the parameters for the $i$-th point
- BA attempts to minimize the projection error

\[
\min_{a_j, b_i} \sum_{i=1}^{n} \sum_{j=1}^{m} d(Q(a_j, b_i), x_{ij})^2
\]

predicted projection

Euclidean distance
Levenberg-Marquardt method

- LM can be thought of as a combination of steepest descent and the Newton method. When the current solution is far from the correct one, the algorithm behaves like a steepest descent method: slow, but guaranteed to converge. When the current solution is close to the correct solution, it becomes a Newton’s method.
Bundle adjustment
The Structure from Motion Pipeline
Applications of matchmove
Jurassic park
Enemy at the Gate, Double Negative
Enemy at the Gate, Double Negative
Photo Tourism

Exploring photo collections in 3D
VideoTrace

Video stabilization
References

Project #3 MatchMove

- It is more about using tools in this project
- You can choose either calibration or structure from motion to achieve the goal
- Calibration
- Voodoo/Icarus

- Examples from previous classes, #1, #2
- https://www.youtube.com/user/theActionMovieKid/videos