Structure from Motion

Read Chapter 7 in Szeliski’s book
<table>
<thead>
<tr>
<th>#</th>
<th>date</th>
<th>topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sep.22</td>
<td>Introduction and geometry</td>
</tr>
<tr>
<td>2</td>
<td>Sep.29</td>
<td>Invariant features</td>
</tr>
<tr>
<td>3</td>
<td>Oct.6</td>
<td>Camera models and calibration</td>
</tr>
<tr>
<td>4</td>
<td>Oct.13</td>
<td>Multiple-view geometry</td>
</tr>
<tr>
<td>5</td>
<td>Oct.20</td>
<td>Model fitting (RANSAC, EM, ...)</td>
</tr>
<tr>
<td>6</td>
<td>Oct.27</td>
<td>Stereo Matching</td>
</tr>
<tr>
<td>7</td>
<td>Nov.3</td>
<td>2D Shape Matching</td>
</tr>
<tr>
<td>8</td>
<td>Nov.10</td>
<td>Segmentation</td>
</tr>
<tr>
<td>9</td>
<td>Nov.17</td>
<td>Shape from Silhouettes</td>
</tr>
<tr>
<td>10</td>
<td>Nov.24</td>
<td><strong>Structure from motion</strong></td>
</tr>
<tr>
<td>11</td>
<td>Dec.1</td>
<td>Optical Flow</td>
</tr>
<tr>
<td>12</td>
<td>Dec.8</td>
<td>Tracking (Kalman, particle filter)</td>
</tr>
<tr>
<td>13</td>
<td>Dec.15</td>
<td>Object category recognition</td>
</tr>
<tr>
<td>14</td>
<td>Dec.22</td>
<td>Specific object recognition</td>
</tr>
</tbody>
</table>
Today’s class

- Structure from motion
  - factorization
  - sequential
  - bundle adjustment
Factorization

- Factorise observations in structure of the scene and motion/calibration of the camera

- Use all points in all images at the same time

- Affine factorisation
- Projective factorisation
Affine camera

The affine projection equations are

\[
\begin{bmatrix}
    x_{ij} \\
    y_{ij} \\
    1
\end{bmatrix} =
\begin{bmatrix}
    P_i^x \\
    P_i^y \\
    0001
\end{bmatrix}
\begin{bmatrix}
    X_j \\
    Y_j \\
    Z_j
\end{bmatrix}
\]

\[
\begin{bmatrix}
    x_{ij} \\
    y_{ij} \\
    1
\end{bmatrix} =
\begin{bmatrix}
    P_i^x \\
    P_i^y \\
    0001
\end{bmatrix}
\begin{bmatrix}
    X_j \\
    Y_j \\
    Z_j
\end{bmatrix}
\]

how to find the origin? or for that matter a 3D reference point?

**affine projection preserves center of gravity**

\[
\tilde{x}_{ij} = x_{ij} - \sum_i x_{ij} \quad \tilde{y}_{ij} = y_{ij} - \sum_i y_{ij}
\]
Orthographic factorization

(Tomasi Kanade’92)

The orthographic projection equations are

\[ \bar{m}_{ij} = \bar{P}_i \bar{M}_j, \ i = 1, \ldots, m, \ j = 1, \ldots, n \]

where

\[ \bar{m}_{ij} = \begin{bmatrix} \bar{x}_{ij} \\ \bar{y}_{ij} \end{bmatrix}, \ \bar{P}_i = \begin{bmatrix} \bar{P}_i^x \\ \bar{P}_i^y \end{bmatrix}, \ \bar{M}_j = \begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} \]

All equations can be collected for all \( i \) and \( j \)

\[ \bar{m} = \bar{P} \bar{M} \]

where

\[ \bar{m} = \begin{bmatrix} \bar{m}_{11} & \bar{m}_{12} & \ldots & \bar{m}_{1n} \\ \bar{m}_{21} & \bar{m}_{22} & \ldots & \bar{m}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{m}_{m1} & \bar{m}_{m2} & \ldots & \bar{m}_{mn} \end{bmatrix}, \ \bar{P} = \begin{bmatrix} \bar{P}_1 \\ \bar{P}_2 \\ \vdots \\ \bar{P}_m \end{bmatrix}, \ \bar{M} = [M_1, M_2, \ldots, M_n] \]

Note that \( \bar{P} \) and \( \bar{M} \) are resp. 2mx3 and 3xn matrices and therefore the rank of \( \bar{m} \) is at most 3.
Orthographic factorization

Factorize $m$ through singular value decomposition

$$\overline{m} = U\Sigma V^\top$$

An affine reconstruction is obtained as follows

$$\tilde{P} = U, \tilde{M} = \Sigma V^\top$$

Closest rank-3 approximation yields MLE!

$$\min \left\| \begin{bmatrix} \overline{m}_{11} & \overline{m}_{12} & \cdots & \overline{m}_{1n} \\ \overline{m}_{21} & \overline{m}_{22} & \cdots & \overline{m}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \overline{m}_{m1} & \overline{m}_{m2} & \cdots & \overline{m}_{mn} \end{bmatrix} - \begin{bmatrix} \overline{P}_1 \\ \overline{P}_2 \\ \vdots \\ \overline{P}_m \end{bmatrix} \begin{bmatrix} M_1 & M_2 & \cdots & M_n \end{bmatrix} \right\|$$

$$\left( \sum_{ij} ||m_{ij}||^2 = ||\overline{m} - \tilde{P}\tilde{M}||_{\text{frob}} \right)$$
Orthographic factorization

Factorize \( m \) through singular value decomposition
\[
\overline{m} = U \Sigma V^T
\]
An affine reconstruction is obtained as follows
\[
\widetilde{P} = U, \widetilde{M} = \Sigma V^T
\]

A metric reconstruction is obtained as follows
\[
\overline{P} = \widetilde{P} A^{-1}, \overline{M} = A \widetilde{M}
\]
Where \( A \) is computed from
\[
\begin{align*}
\widetilde{P}_x x^T \overline{P}_i x = 1 \\
\widetilde{P}_y y^T \overline{P}_i y = 1 \\
\widetilde{P}_x x^T \overline{P}_i y^T = 0
\end{align*}
\]
3 linear equations per view on symmetric matrix \( C \) (6DOF)

\( A \) can be obtained from \( C \) through Cholesky factorisation and inversion
Examples

Tomasi Kanade’92,
Poelman & Kanade’94
Examples

Tomasi Kanade’92, Poelman & Kanade’94
Examples

Tomasi Kanade’92,
Poelman & Kanade’94
Examples

Tomasi Kanade’92, Poelman & Kanade’94
Perspective factorization

The camera equations

$$\lambda_{ij} m_{ij} = P_i M_j, i = 1, \ldots, m, j = 1, \ldots, m$$

for a fixed image $i$ can be written in matrix form as

$$m_i \Lambda_i = P_i M$$

where

$$m_i = [m_{i1}, m_{i2}, \ldots, m_{im}], \quad M = [M_1, M_2, \ldots, M_m]$$

$$\Lambda_i = \text{diag}(\lambda_{i1}, \lambda_{i2}, \ldots, \lambda_{im})$$
Perspective factorization

All equations can be collected for all $i$ as

$$m = PM$$

where

$$m = \begin{bmatrix} m_1 \Lambda_1 \\ m_2 \Lambda_2 \\ \vdots \\ m_n \Lambda_n \end{bmatrix}, \quad P = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_m \end{bmatrix}$$

In these formulas $m$ are known, but $\Lambda_i, P$ and $M$ are unknown.

Observe that $PM$ is a product of a $3m \times 4$ matrix and a $4 \times n$ matrix, i.e. it is a rank-4 matrix.
Perspective factorization algorithm

Assume that $\Lambda_i$ are known, then $PM$ is known.

Use the singular value decomposition

$$PM = U\Sigma V^T$$

In the noise-free case

$$\Sigma = \text{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4, 0, \ldots, 0)$$

and a reconstruction can be obtained by setting:

$P = \text{the first four columns of } U\Sigma.$

$M = \text{the first four rows of } V.$
Iterative perspective factorization

When $\Lambda_i$ are unknown the following algorithm can be used:

1. Set $\lambda_{ij} = 1$ (affine approximation).

2. Factorize $PM$ and obtain an estimate of $P$ and $M$. If $\sigma_5$ is sufficiently small then STOP.

3. Use $m_i$, $P$ and $M$ to estimate $\Lambda_i$ from the camera equations (linearly) $m_i \Lambda_i = P_i M$


In general the algorithm minimizes the proximity measure $P(\Lambda, P, M) = \sigma_5$

Note that structure and motion recovered up to an arbitrary projective transformation
Further Factorization work

Factorization with uncertainty
(Irani & Anandan, IJCV’02)

Factorization for dynamic scenes
(Costeira and Kanade ‘94)
(Bregler et al. ‘00, Brand ‘01)
(Yan and Pollefeys, ‘05/’06)
Multi-Camera Factorizations

- (Static) affine cameras
- Rigidly moving object
- Camera calibration using rigid motion
  - 2D feature point trajectories as input
  - No feature point correspondences between different camera views required

All tracks of all affine cameras form rank 13 subspace!

(Angst & Pollefeys ICCV09)
Affine Structure from Motion (SfM)

- From projective to affine camera:
  \[\begin{pmatrix} x \\ y \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathbf{C}_k & \mathbf{M}_t & \mathbf{S}_n \end{pmatrix}\]

- Projective camera:
  \[\mathbf{x}_{t,k,n} = \begin{pmatrix} x \\ y \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathbf{C}_k & \mathbf{M}_t & \mathbf{S}_n \end{pmatrix}\]
Multi-Camera Factorizations

- Factorization technique
  - 3rd order tensor with rank 13 constraint

- $C$ : camera matrices
- $M$ : motion matrix
- $S$ : Shape matrix

(Angst & Pollefeys ICCV09)
Multi-Camera Factorizations

- Experiment on real data

- Minimal cases

<table>
<thead>
<tr>
<th># points per camera</th>
<th>(3, 4)</th>
<th>(4, 4)</th>
<th>(1, 3, 3)</th>
<th>(2, 3, 3)</th>
<th>(2, 2, 4)</th>
<th>(2, 2, 2)</th>
<th>(2, 2, 2, 3)</th>
<th>(2, 2, 2, 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>rank (A) ( A \cong 13 )</td>
<td>( 12 \neq 13 )</td>
<td>13 = 13</td>
<td>13 = 13</td>
<td>13 = 13</td>
<td>13 = 13</td>
<td>13 = 13</td>
<td>13 = 13</td>
<td>13 = 13</td>
</tr>
<tr>
<td>Eq. (15) solvable</td>
<td>( \times )</td>
<td>( \checkmark )</td>
<td>( \times )</td>
<td>( \checkmark )</td>
<td>( \times )</td>
<td>( \times )</td>
<td>( \checkmark )</td>
<td>( \checkmark )</td>
</tr>
<tr>
<td>Sec. 4.2 applicable</td>
<td>( \times )</td>
<td>( \checkmark )</td>
<td>( \times )</td>
<td>( \checkmark )</td>
<td>( \times )</td>
<td>( \times )</td>
<td>( \checkmark )</td>
<td>( \checkmark )</td>
</tr>
</tbody>
</table>

extra camera: 1 point

(Angst & Pollefeys ICCV09)
practical structure and motion recovery from images

- Obtain reliable matches using matching or tracking and 2/3-view relations
- Compute initial structure and motion
- Refine structure and motion
- Auto-calibrate
- Refine metric structure and motion
Sequential Structure and Motion Computation

Initialize Motion
\((P_1, P_2 \text{ compatible with } F)\)

Extend motion
(compute pose through matches seen in 2 or more previous views)

Initialize Structure
(minimize reprojection error)

Extend structure
(Initialize new structure, refine existing structure)
Computation of initial structure and motion according to Hartley and Zisserman
“this area is still to some extend a black-art”

All features not visible in all images
⇒ No direct method (factorization not applicable)
⇒ Build partial reconstructions and assemble
   (more views is more stable, but less corresp.)

1) Sequential structure and motion recovery
2) Hierarchical structure and motion recovery
Sequential structure and motion recovery

- Initialize structure and motion from two views
- For each additional view
  - Determine pose
  - Refine and extend structure
- Determine correspondences robustly by jointly estimating matches and epipolar geometry
Initial structure and motion

Epipolar geometry ↔ Projective calibration

\[ \mathbf{m}_2^T \mathbf{F} \mathbf{m}_1 = 0 \quad \mathbf{P}_1 = \begin{bmatrix} \mathbf{I} & 0 \end{bmatrix} \]

\[ \mathbf{P}_2 = \begin{bmatrix} [\mathbf{e}]_x \mathbf{F} + \mathbf{e} \mathbf{a}^T & \mathbf{e} \end{bmatrix} \]

compatible with \( \mathbf{F} \)

Yields correct projective camera setup

(Faugeras´92,Hartley´92)

Obtain structure through triangulation

Use reprojection error for minimization

Avoid measurements in projective space
Determine pose towards existing structure

\[ x_i = P_i X(x_1, \ldots, x_{i-1}) \]

Compute \( P_{i+1} \) using robust approach (6-point RANSAC)

Extend and refine reconstruction
Compute P with 6-point RANSAC

- Generate hypothesis using 6 points
  \[
  \begin{bmatrix}
  0^\top \\
  w_i x_i^\top \\
  0^\top \\
  -w_i x_i^\top \\
  y_i x_i^\top \\
  -x_i x_i^\top 
  \end{bmatrix}
  \begin{pmatrix}
  P_1 \\
  P_2 \\
  P_3
  \end{pmatrix} = 0
  \]

- Count inliers
  - Projection error
    \[d(P_i X(x_1,\ldots,x_{i-1}),x_i) < t?\]
  - 3D error
    \[d_{\Lambda}(P_i^{-1}(x_i),X) < t_{3D}?\]
  - Back-projection error
    \[d(F_{ij},x_i,x_j) < t?, \forall j < i\]
  - Re-projection error
    \[d(P_i X(x_1,\ldots,x_{i-1},x_i),x_i) < t\]
  - Projection error with covariance
    \[d_{\Lambda}(P_i X(x_1,\ldots,x_{i-1}),x_i) < t\]

- Expensive testing? Abort early if not promising
  - Verify at random, abort if e.g. P(wrong) > 0.95

(Chum and Matas, BMVC’02)
Calibrated structure from motion

- Equations more complicated, but less degeneracies
- For calibrated cameras:
  - 5-point relative motion (5DOF)
    Nister CVPR03
  - 3-point pose estimation (6DOF)
    Haralick et al. IJCV94


5-point relative motion

(Nister, CVPR03)

- Linear equations for 5 points
  
  \[
  \begin{bmatrix}
  x'_1 x_1 & x'_1 y_1 & x'_1 & y'_1 x_1 & y'_1 y_1 & y'_1 & x_1 & y_1 & 1 \\
  x'_2 x_2 & x'_2 y_2 & x'_2 & y'_2 x_2 & y'_2 y_2 & y'_2 & x_2 & y_2 & 1 \\
  x'_3 x_3 & x'_3 y_3 & x'_3 & y'_3 x_3 & y'_3 y_3 & y'_3 & x_3 & y_3 & 1 \\
  x'_4 x_4 & x'_4 y_4 & x'_4 & y'_4 x_4 & y'_4 y_4 & y'_4 & x_4 & y_4 & 1 \\
  x'_5 x_5 & x'_5 y_5 & x'_5 & y'_5 x_5 & y'_5 y_5 & y'_5 & x_5 & y_5 & 1
  \end{bmatrix}
  \begin{bmatrix}
  E_{11} \\
  E_{12} \\
  E_{13} \\
  E_{21} \\
  E_{22} \\
  E_{23} \\
  E_{31} \\
  E_{32} \\
  E_{33}
  \end{bmatrix} = 0
  \]

- Linear solution space
  scale does not matter, choose

- Non-linear constraints
  10 cubic polynomials
5-point relative motion

(Nister, CVPR03)

- Perform Gauss-Jordan elimination on polynomials

\[
\begin{array}{cccccccccccc}
\langle a \rangle & x^3 & y^3 & x^2 y & xy^2 & x^2 z & x^2 & y^2 z & y^2 & x y z & x y & x & y & 1 \\
\langle b \rangle & 1 & . & . & . & . & . & . & . & . & . & . & . & . \\
\langle c \rangle & 1 & . & . & . & . & . & . & . & . & . & . & . & . \\
\langle d \rangle & 1 & 1 & . & . & . & . & . & . & . & . & . & . & . \\
\langle e \rangle & 1 & 1 & 1 & . & . & . & . & . & . & . & . & . & . \\
\langle f \rangle & . & . & . & . & . & . & . & . & . & . & . & . & . \\
\langle g \rangle & . & . & . & . & . & . & . & . & . & . & . & . & . \\
\langle h \rangle & . & . & . & . & . & . & . & . & . & . & . & . & . \\
\langle i \rangle & . & . & . & . & . & . & . & . & . & . & . & . & . \\
\langle j \rangle & . & . & . & . & . & . & . & . & . & . & . & . & . \\
\end{array}
\]

\[
\begin{array}{|c|c|c|c|}
\hline
\langle k \rangle & \equiv & \langle e \rangle - z \langle f \rangle & 3 & 3 & 4 \\
\langle l \rangle & \equiv & \langle g \rangle - z \langle h \rangle & 3 & 3 & 4 \\
\langle m \rangle & \equiv & \langle i \rangle - z \langle j \rangle & 3 & 3 & 4 \\
\langle n \rangle & \equiv & det(B) & & & \\
\hline
\end{array}
\]
Three points perspective pose – p3p

All techniques yield 4th order polynomial

Haralick et al. recommends using Finsterwalder’s technique as it yields the best results numerically

Haralick et al., IJCV'94

\[
\begin{align*}
\frac{s_2^2 + s_3^2 - 2s_2s_3 \cos \alpha}{a^2} &= 0 \\
\frac{s_1^2 + s_3^2 - 2s_1s_3 \cos \beta}{b^2} &= 0 \\
\frac{s_1^2 + s_2^2 - 2s_1s_2 \cos \gamma}{c^2} &= 0
\end{align*}
\]
Minimal solvers
Lot’s of recent activity using Groebner bases:

- D. Nistér, *A Minimal solution to the generalised 3-point pose problem*, *CVPR 2004*
- Brian Clipp, Christopher Zach, Jan-Michael Frahm and Marc Pollefeys, *A New Minimal Solution to the Relative Pose of a Calibrated Stereo Camera with Small Field of View Overlap*, *ICCV 2009*.
- F. Fraundorfer, P. Tanskanen and M. Pollefeys. *A Minimal Case Solution to the Calibrated Relative Pose Problem for the Case of Two Known Orientation Angles*. *ECCV 2010*.
- ...
Minimal relative pose with know vertical
Fraundorfer, Tanskanen and Pollefeys, ECCV2010

Vertical direction can often be estimated
• inertial sensor
• vanishing point

\[
E = \begin{bmatrix}
  t_z \sin(y) & -t_z \cos(y) & t_y \\
  t_z \cos(y) & t_z \sin(y) & -t_x \\
  -t_y \cos(y) - t_x \sin(y) & t_x \cos(y) - t_y \sin(y) & 0
\end{bmatrix}
\]

5 linear unknowns → linear 5 point algorithm
3 unknowns → quartic 3 point algorithm
Incremental Structure from Motion

Photo Tourism

Noah Snavely, Steve Seitz, Rick Szeliski

University of Washington
Incremental structure from motion

- Automatically select an initial pair of images
Incremental structure from motion
Incremental structure from motion
Hierarchical structure and motion recovery

- Compute 2-view
- Compute 3-view
- Stitch 3-view reconstructions
- Merge and refine reconstruction
Hierarchical structure from motion

Hierarchical structure from motion

Figure 4. Two perspective views of the reconstruction of “Piazza Bra” with the Arena (Verona, Italy).
Hierarchical structure from motion

Figure 5. Top views aligned with an aerial image of “Piazza Erbe” (from Google Earth), reconstructed with SAMANTHA (left) and with BUNDLER (right).
Stitching 3-view reconstructions

Different possibilities

1. Align \((P_2,P_3)\) with \((P'_1,P'_2)\)
   \[
   \arg \min_{P' \in \mathbb{P}} d_A(P_2, P'_1 H^{-1}) + d_A(P_3, P'_2 H^{-1})
   \]

2. Align \(X,X'\) (and \(C'C'\))
   \[
   \arg \min_{H \in \mathbb{H}} \sum_j d_A(X_j, HX'_j)
   \]

3. Minimize reproj. error
   \[
   \arg \min_{H \in \mathbb{H}} \sum_j d(H^{-1}X'_j, x_j) + \sum_j d(P'HX_j, x'_j)
   \]

4. MLE (merge)
   \[
   \arg \min_{P,X} \sum_j d(PX_j, x_j)
   \]
More issues

- Repetition ambiguities

- Large scale reconstructions
Disambiguating visual relations using loop constraints

(Zach et al CVPR’10)
Repetition and symmetry detection
(Wu et al ECCV’10)
Video-only large-scale reconstruction?

- **Challenge:**
  Error accumulation yields drift of relative scale, orientation and position

- **Solution:**
  Cancel drift by closing loops (e.g. at intersections)
  Need to visually recognize locations
Solving 3D puzzles with VIPs

**SIFT features**
- Extracted from 2D images
- Variation due to viewpoint

**VIP features** (Wu et al., CVPR08)
- Extracted from 3D model
- Viewpoint invariant
3D Models with VIPs
Where am I? What am I looking at?

Image query

SIFT VIP

3D Database

Location recognition (Baatz et al., ECCV2010)
Projective ambiguity

reconstruction from uncalibrated images only
\[ \Rightarrow \text{projective ambiguity on reconstruction} \]

\[ m = PM = (PT^{-1})(TM) = P'M' \]
Factorization of Euclidean projection matrix

\[ P = K \begin{bmatrix} R^T & -R^Tt \end{bmatrix} \]

Intrinsics: \[ K = \begin{bmatrix} f_x & s & c_x \\ f_y & c_y & 1 \end{bmatrix} \] (camera geometry)

Extrinsics: \( (R, t) \) (camera motion)

Note: every projection matrix can be factorized, but only meaningful for euclidean projection matrices
The Absolute Quadric

Eliminate extrinsics from equation

$$\begin{bmatrix} R^T & -R^T t \end{bmatrix} \rightarrow KR^T R K^T \rightarrow KK^T$$

Equivalent to projection of quadric

$$P \Omega P^T = KK^T \quad \Omega^* = \text{diag}(1110)$$

Absolute Quadric also exists in projective world

$$KK^T = P \Omega^* P^T = (PT^{-1})(T \Omega^* T^T)(T^{-T} P^T)$$

$$= P' \Omega^* P'^T$$

Transforming world so that

$$\Omega' \rightarrow \Omega^*$$

reduces ambiguity to metric
Absolute quadric and self-calibration

Projection equation:

$$\omega_i^* = P_i \Omega^* P_i^T = K_i K_i^T$$

- Translate constraints on $K$
- through projection equation
- to constraints on $\Omega^*$

Absolute Quadric = calibration object which is always present but can only be observed through constraints on the intrinsics
Practical linear self-calibration

Don’t treat all constraints equal

\[
KK^T = P^* P^T \approx \begin{bmatrix} \hat{f}^2 & 0 & 0 \\ 0 & \hat{f}^2 & 0 \\ 0 & 0 & 1 \end{bmatrix}
\]

(only rough approximation, but still useful to avoid degenerate configurations)

after normalization!

(relatively accurate for most cameras)

\[
\frac{1}{0.2} (P^* P^T)_{11} - (P^* P^T)_{22} = 0
\]
\[
\frac{1}{0.01} (P^* P^T)_{12} = 0
\]
\[
\frac{1}{0.01} (P^* P^T)_{13} = 0
\]
\[
\frac{1}{0.1} (P^* P^T)_{23} = 0
\]
\[
\frac{1}{0.2} (P^* P^T)_{11} - (P^* P^T)_{33} = 0
\]
\[
\frac{1}{0.2} (P^* P^T)_{22} - (P^* P^T)_{33} = 0
\]

when fixating point at image-center not only absolute quadric \( \text{diag}(1,1,1,0) \) satisfies ICCV98 eqs., but also \( \text{diag}(1,1,1,a) \), i.e. real or imaginary spheres!

Recent related work (Gurdjos et al. ICCV’09)
Upgrade from projective to metric

- Locate $\Omega^*$ in projective reconstruction (self-calibration)
  Problem: not always unique (Critical Motion Sequences)
- Transform projective reconstruction
- to bring $\Omega^*$ in canonical position
Refining structure and motion

- Minimize reprojection error

\[
\min_{\hat{P}_k, \hat{M}_i} \sum_{k=1}^{m} \sum_{i=1}^{n} D(m_{ki}, \hat{P}_k \hat{M}_i)^2
\]

- MaximumLikelihood Estimation
  (if error zero-mean Gaussian noise)
- Huge problem but can be solved efficiently
  (Bundle adjustment)
Non-linear least-squares

\[ X = f(P) \quad \text{argmin}_P \|X - f(P)\| \]

- Newton iteration
- Levenberg-Marquardt
- Sparse Levenberg-Marquardt
Newton iteration

Taylor approximation

\[
f(P_0 + \Delta) \approx f(P_0) + J\Delta
\]

Jacobian

\[
J = \frac{\partial X}{\partial P}
\]

\[
\|X - f(P_1)\|
\]

\[
\|X - f(P_1)\| \approx \|X - f(P_0) - J\Delta\| = \|e_0 - J\Delta\|
\]

\[
\Rightarrow J^T J \Delta = J^T e_0 \Rightarrow \Delta = \left( J^T J \right)^{-1} J^T e_0
\]

\[
P_{i+1} = P_i + \Delta
\]

\[
\Delta = \left( J^T J \right)^{-1} J^T e_0
\]

\[
\Delta = \left( J^T \Sigma^{-1} J \right)^{-1} J^T \Sigma^{-1} e_0
\]

normal eq.
Levenberg-Marquardt

Normal equations

\[ J^T J \Delta = N \Delta = J^T e_0 \]

Augmented normal equations

\[ N' \Delta = J^T e_0 \quad N' = J^T J + \lambda \text{diag}(J^T J) \]

\[ \lambda_0 = 10^{-3} \]

success: \( \lambda_{i+1} = \lambda_i / 10 \)  accept

failure: \( \lambda_i = 10 \lambda_i \)  solve again

\( \lambda \) small \( \sim \) Newton (quadratic convergence)

\( \lambda \) large \( \sim \) descent (guaranteed decrease)
Levenberg-Marquardt

Requirements for minimization
• Function to compute $f$
• Start value $P_0$
• Optionally, function to compute $J$
  (but numerical ok, too)
Bundle Adjustment

\[ \hat{u}_{ij} = f(K, R_j, t_j, x_i) \]
\[ \hat{v}_{ij} = g(K, R_j, t_j, x_i) \]

- What makes this non-linear minimization hard?
  - many more parameters: potentially slow
  - poorer conditioning (high correlation)
  - potentially lots of outliers
  - gauge (coordinate) freedom
6.2.2 Iterative algorithms

The most accurate (and flexible) way to estimate pose is to directly minimize the squared (or robust) reprojection error for the 2D points as a function of the unknown pose parameters in $(R, t)$ and optionally $K$ using non-linear least squares (Tsai 1987; Bogart 1991; Gleicher and Witkin 1992). We can write the projection equations as

$$x_i = f(p_i; R, t, K)$$  \hspace{1cm} (6.41)

and iteratively minimize the robustified linearized reprojection errors

$$E_{\text{NLP}} = \sum_{i} \rho \left( \frac{\partial f}{\partial R} \Delta R + \frac{\partial f}{\partial t} \Delta t + \frac{\partial f}{\partial K} \Delta K - r_i \right),$$  \hspace{1cm} (6.42)

where $r_i = \hat{x}_i - \hat{x}_i$ is the current residual vector (2D error in predicted position), and the partial derivatives are with respect to the unknown pose parameters (rotation, translation, and optionally calibration). Note that if full 2D covariance estimates are available for the 2D feature locations, the above squared norm can be weighted by the inverse point covariance matrix, as in (6.11).
An easier to understand (and implement) version of the above non-linear regression problem can be constructed by re-writing the projection equations as a concatenation of simpler steps, each of which transforms a 4D homogeneous coordinate $p_i$ by a simple transformation such as translation, rotation, or perspective division (Figure 6.5). The resulting projection

\[ y^{(1)} = f_T(p_i; c_j) = p_i - c_j, \]  
\[ y^{(2)} = f_R(y^{(1)}; q_j) = R(q_j) y^{(1)}, \]  
\[ y^{(3)} = f_P(y^{(2)}) = \frac{y^{(2)}}{z^{(2)}}, \]  
\[ x_i = f_C(y^{(3)}; k) = K(k) y^{(3)}. \]  

(6.43)  
(6.44)  
(6.45)  
(6.46)

The advantage of this chained set of transformations is that each one has a simple partial derivative with respect to both its parameters and to its input. Thus, once the predicted value of $\tilde{x}_i$ has been computed based on the 3D point location $p_i$ and the current values of the pose parameters $(c_j, q_j, k)$, we can obtain all of the required partial derivatives using the chain rule

\[ \frac{\partial r_i}{\partial p^{(k)}} = \frac{\partial r_i}{\partial y^{(k)}} \frac{\partial y^{(k)}}{\partial p^{(k)}}, \]  

(6.47)

where $p^{(k)}$ indicates in this case one of the parameter vectors that is being optimized. (This same “trick” is used in neural networks as part of the backpropagation algorithm (Bishop 2006).)
Chained transformations

- Think of the optimization as a set of transformations along with their derivatives
Levenberg-Marquardt

- Iterative non-linear least squares [Press’92]
  - Linearize measurement equations
    \[
    \hat{u}_i = f(m, x_i) + \frac{\partial f}{\partial m} \Delta m
    \]
    \[
    \hat{v}_i = g(m, x_i) + \frac{\partial g}{\partial m} \Delta m
    \]
  - Substitute into log-likelihood equation: quadratic cost function in \( \Delta m \)
    \[
    \sum_i \sigma_i^{-2} (\hat{u}_i - u_i + \frac{\partial f}{\partial m} \Delta m)^2 + \cdots
    \]
Robust error models

- Outlier rejection
  - use robust penalty applied to each set of joint measurements
  \[
  \sum_i \sigma_i^{-2} \rho \left( \sqrt{(u_i - \hat{u}_i)^2 + (v_i - \hat{v}_i)^2} \right)
  \]
  - for extremely bad data, use random sampling [RANSAC, Fischler & Bolles, CACM’81]
Levenberg-Marquardt

- Iterative non-linear least squares [Press’92]
  - Solve for minimum \( \frac{\partial C}{\partial m} = 0 \)

\[
A\Delta m = b
\]

Hessian:

\[
A = \left[ \sum_i \sigma_i^{-2} \frac{\partial f}{\partial m} \left( \frac{\partial f}{\partial m} \right)^T + \ldots \right]
\]

error:

\[
b = \left[ \sum_i \sigma_i^{-2} \frac{\partial f}{\partial m} (u_i - \hat{u}_i) + \ldots \right]
\]
Chained transformations

• Only some derivatives are non-zero:
  \(j\)th camera, \(i\)th point
Multi-camera rig

- Easy extension of generalized bundler:
Lots of parameters: sparsity

\[ \hat{u}_{ij} = f(K, R_j, t_j, x_i) \]

\[ \hat{v}_{ij} = g(K, R_j, t_j, x_i) \]

- Only a few entries in Jacobian are non-zero
• **Schur complement** to get *reduced camera* matrix

---

**Figure 7.9** Bipartite graph for a toy structure from motion problem (a) and its associated Jacobian $J$ (b) and Hessian $A$ (c). Numbers (1–9) indicate 3D points while letters (A–D) indicate cameras. The dashed arcs and light blue squares indicate the fill-in that occurs when the structure (point) variables are eliminated.
Sparse Levenberg-Marquardt

- $N^3$ complexity for solving $\Delta = N^{-1} J^T e_0$
  - prohibitive for large problems
    (100 views 10,000 points $\sim 30,000$ unknowns)

- Partition parameters
  - partition A
  - partition B (only dependent on A and itself)
Sparse bundle adjustment

residuals $D(m_{ki}, \hat{P}_k \hat{M}_i)^2$

normal equations

$$\begin{bmatrix}
  U & W \\
  W^\top & V
\end{bmatrix}
\begin{bmatrix}
  \Delta(P) \\
  \Delta(M)
\end{bmatrix} =
\begin{bmatrix}
  \epsilon(P) \\
  \epsilon(M)
\end{bmatrix}$$

with

$$U_k = \sum_i \left( \frac{\partial \hat{m}_{ki}}{\partial \hat{P}_k} \right)^\top \frac{\partial \hat{m}_{ki}}{\partial \hat{P}_k}$$

$$\epsilon(P_k) = \sum_i \left( \frac{\partial \hat{m}_{ki}}{\partial \hat{P}_k} \right)^\top \epsilon_{ki}$$

$$V_i = \sum_k \left( \frac{\partial \hat{m}_{ki}}{\partial \hat{M}_i} \right)^\top \frac{\partial \hat{m}_{ki}}{\partial \hat{M}_i}$$

$$\epsilon(M_i) = \sum_i \left( \frac{\partial \hat{m}_{ki}}{\partial \hat{M}_i} \right)^\top \epsilon_{ki}$$

$$W_{ki} = \left( \frac{\partial \hat{m}_{ki}}{\partial \hat{P}_k} \right)^\top \frac{\partial \hat{m}_{ki}}{\partial \hat{M}_i}$$

note: tie points should be in partition A
Sparse bundle adjustment

normal equations:

\[
\begin{bmatrix}
I & -WV^{-1} \\
0 & I
\end{bmatrix}
\begin{bmatrix}
U & W \\
W^T & V
\end{bmatrix}
\begin{bmatrix}
\Delta(P) \\
\Delta(M)
\end{bmatrix}
= 
\begin{bmatrix}
\epsilon(P) \\
\epsilon(M)
\end{bmatrix}
\]

modified normal equations:

\[
\begin{bmatrix}
U - WV^{-1}W^T & 0 \\
W^T & V
\end{bmatrix}
\begin{bmatrix}
\Delta(P) \\
\Delta(M)
\end{bmatrix}
= 
\begin{bmatrix}
\epsilon(P) - WV^{-1}\epsilon(M) \\
\epsilon(M)
\end{bmatrix}
\]

solve in two parts:

\[
(U - WV^{-1}W^T) \Delta(P) = \epsilon(P) - WV^{-1}\epsilon(M)
\]

\[
\Delta(M) = V^{-1} (\epsilon(M) - W^\top \Delta(P))
\]
Sparse bundle adjustment

Jacobian of \( \sum_{k=1}^{m} \sum_{i=1}^{n} D(m_{ki}, \hat{P}_k(\hat{M}_i))^2 \) has sparse block structure

\[
J = \begin{bmatrix}
P_1 & P_2 & P_3 & M
\end{bmatrix}
\]

\( \text{im.pts. view 1} \)

\[
N = J^T J = \begin{bmatrix}
U_1 & U_2 & U_3 & W \\
U_2^T & V & \end{bmatrix}
\]

\( \text{12xm 3xn} \) (in general much larger)

Needed for non-linear minimization
Sparse bundle adjustment

- Eliminate dependence of camera/motion parameters on structure parameters

Note in general $3n > 11m$

\[
\begin{bmatrix}
  I & -WV^{-1} \\
  0 & I
\end{bmatrix} \times N = U-WV^{-1}W^T
\]

Allows much more efficient computations

e.g. 100 views, 10000 points,
solve $±1000 \times 1000$, not $±30000 \times 30000$

Often still band diagonal

use sparse linear algebra algorithms
Sparse bundle adjustment

normal equations:

\[
\begin{bmatrix}
    I & -WV^{-1} \\
    0 & I
\end{bmatrix}
\begin{bmatrix}
    U & W \\
    W^T & V
\end{bmatrix}
\begin{bmatrix}
    \Delta(P) \\
    \Delta(M)
\end{bmatrix}
= 
\begin{bmatrix}
    \epsilon(P) \\
    \epsilon(M)
\end{bmatrix}
\]

modified normal equations:

\[
\begin{bmatrix}
    U - WV^{-1}W^T & 0 \\
    W^T & V
\end{bmatrix}
\begin{bmatrix}
    \Delta(P) \\
    \Delta(M)
\end{bmatrix}
= 
\begin{bmatrix}
    \epsilon(P) - WV^{-1}\epsilon(M) \\
    \epsilon(M)
\end{bmatrix}
\]

solve in two parts:

\[
(U - WV^{-1}W^T) \Delta(P) = \epsilon(P) - WV^{-1}\epsilon(M)
\]

\[
\Delta(M) = V^{-1} (\epsilon(M) - W^T \Delta(P))
\]
Sparse bundle adjustment

- Covariance estimation

\[ \Sigma_a = \left( U - WV^{-1}W \right)^+ \]
\[ \Sigma_b = Y^T \Sigma_a Y + V^{-1} \]
\[ Y = WV^{-1} \]
\[ \Sigma_{ab} = -\Sigma_a Y \]
Direct vs. iterative solvers

- How do they work?
- When should you use them?
Large reduced camera matrices

- Use iterative preconditioned conjugate gradient
  - [Jeong, Nister, Steedly, Szeliski, Kweon, CVPR 2009]
  - [Agarwal, Snavely, Seitz, and Szeliski, ECCV 2009]
  - Block diagonal works well
  - Sometimes partial Cholesky does too
  - Direct solvers for small problems
Large-scale structure from motion
Large-scale reconstruction

- Most of the models shown so far have had ~500 images

- How do we scale from 100s to 10,000s of images?

- Observation: Internet collections represent very non-uniform samplings of viewpoint
Skeletal graphs for efficient structure from motion

Noah Snavely, Steve Seitz, Richard Szeliski
CVPR’2008
Skeletal set

- Goal: select a smaller set of images to reconstruct, without sacrificing \textit{quality} of reconstruction

- Two problems:
  - How do we measure quality?
  - How do we find a subset of images with bounded quality loss?
Skeletal set

Goal: given an image graph $G_I$, 
- select a small set $S$ of *important* images to reconstruct, bounding the loss in *quality* of the reconstruction
- Reconstruct the skeletal set $S$
- Estimate the remaining images with much faster pose estimation steps
Properties of the skeletal set

• Should touch all parts of $G$
  *Dominating set*

• Should form a single reconstruction
  *Connected dominating set*

• Should result in an *accurate* reconstruction
Example

Full graph

Skeletal graph
Representing information in a graph

What kind of information?

- No absolute information about camera positions
- Each edge provides information about the relative positions of two images
- ... but not all edges are equally informative

We model information with the uncertainty (covariance) in pairwise camera positions
Estimating certainty

- Usually measured with covariance matrix
- SfM has thousands or millions of parameters
- We only measure covariance in camera positions (3x3 matrix for every camera)
Pantheon

Full graph

Skeletal graph (t=16)
Pantheon

Skeletal reconstruction
101 images

After adding leaves
579 images

After final optimization
579 images
Trafalgar Square

2973 images registered (277 in skeletal set)
Trafalgar Square
Running time

- Stonehenge
- St. Peters
- Pantheon
- Pisa
- Trafalgar

- Full reconstruction: ~10 days
- Full reconstruction + Skeletal reconstruction: ~30 days

- Skeletal reconstruction
Building Rome in a Day

Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, Richard Szeliski

ICCV’2009
Distributed matching pipeline
The evolution of the match graph as a function of the rounds of matching, and the skeletal set corresponding to it. Notice how the second round of matching merges the two components into one, and how rapidly the query expansion increases the density of the within component connections. The last column shows the skeletal set corresponding to the final match graph. The skeletal sets algorithm can break up connected components found during the match phase if it determines that a reliable reconstruction is not possible, which is what happens in this case.
Results: Dubrovnik

(a) Dubrovnik: Four different views and associated images from the largest connected component. Note that the component captures the entire old city, with both street-level and roof-top detail. The reconstruction consists of 4,585 images and 2,662,981 3D points with 11,839,682 observed features.
Results: Rome

Colosseum: 2,097 images, 819,242 points
Trevi Fountain: 1,935 images, 1,055,153 points
Pantheon: 1,032 images, 530,076 points
Hall of Maps: 275 images, 230,182 points

(b) Rome: Four of the largest connected components visualized at canonical viewpoints [14].
Changchang Wu’s SfM code

for iconic graph
- uses 5-point+RANSAC for 2-view initialization
- uses 3-point+RANSAC for adding views
- performs bundle adjustment

For additional images
- use 3-point+RANSAC pose estimation
Rome on a cloudless day

- GIST & clustering (1h35) (Frahm et al. ECCV 2010)
- Dense Reconstruction (1h58)

SIFT & Geometric verification (11h36)

SfM & Bundle (8h35)

Some numbers
- 1PC
- 2.88M images
- 100k clusters
- 22k SfM with 307k images
- 63k 3D models
- Largest model 5700 images
- Total time 23h53
Structure from motion: limitations

• Very difficult to reliably estimate metric structure and motion unless:
  – large ($x$ or $y$) rotation \textit{or}
  – large field of view and depth variation

• Camera calibration important for Euclidean reconstructions

• Need good feature tracker
Related problems

- On-line structure from motion and SLaM (Simultaneous Localization and Mapping)
  - Kalman filter (linear)
  - Particle filters (non-linear)
Visual SLAM

- Real-time Stereo Visual SLAM
  
  (Clipp et al., IROS2010)
Open challenges

• Large scale structure from motion
  – Complete building
  – Complete city
Open source code available, e.g.

Christopher Zach (SfM, Bundle adjustment)
http://www.inf.ethz.ch/personal/chzach/opensource.html

Photo Tourism Bundler:
http://phototour.cs.washington.edu/bundler/
Next week: Optical Flow