Robot Vision:
Structure-from-Motion (SFM)

Prof. Friedrich Fraundorfer

SS 2023
Outline

- SfM concept
- SfM pipeline
- Image similarity using visual words
- Incremental geometry estimation
- Bundle adjustment
Structure-from-Motion (SfM) concept
Structure-from-Motion (SfM) concept

Initialize Motion
($P_1, P_2$ compatible with $F$)

Initialize Structure
(minimize reprojection error)

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

Extend structure
(Initialize new structure, refine existing structure)
Structure-from-Motion (SfM) core pipeline
Feature extraction

- Extract features (point locations and descriptors) for each of the N images
- SIFT features are recommended (best working features for matching right now)
- GPU accelerated implementations exist
Coarse matching

- To avoid NxN feature matching
- Many possible image pairs in the dataset will not have overlap, detailed feature matching will produce no matches for such pairs
- Cluster similar images by similarity using visual words
- Detailed matching will only be performed for similar images

Image similarity
Visual words

feature space cells (e.g. SIFT)
Histogram of visual words (bags of words)
Detailed matching

- Typically using an approximated nearest neighbor (ANN) algorithm
Geometric verification and epipolar graph

- Geometric verification of 2-view matches using fundamental matrix or essential matrix computation
- Epipolar graph: Is a plot of the number of geometrically verified 2-view feature matches
- Defines the sequential order for geometry processing
Geometry estimation

- Following the sequence ordering from the epipolar graph geometry is estimated for all images
- Geometry estimation is an alternating scheme:
  - Estimate camera pose of new images (position, rotation)
  - Triangulate new 3D data points seen in new image
  - Refinement by non-linear optimization (Bundle adjustment)
Geometry estimation steps

- Compute camera poses of the first two images from feature matches

\[ P = K[I|0] \]

\[ P' = K'[R'|t'] \]
Geometry estimation steps

- Computation of first 3D points by triangulation

\[ P = K[I|0] \quad P' = K'[R'|t'] \]
Geometry estimation steps

- Triangulate all feature matches of the first images

\[
P = K[I|0] \quad \quad P' = K'[R'|t']
\]
Geometry estimation steps

- First refinement of camera poses and 3D points by non-linear estimation of the re-projection error through bundle adjustment

\[
P = K[I|0] \quad \quad P' = K'[R'|t']
\]
Geometry estimation steps

- Start processing the next image

\[ P = K[I|0] \]

\[ P'' = ? \]

\[ P' = K'[R'|t'] \]
Geometry estimation steps

- First, create feature matches to all the previous, neighboring images

\[ P = K[I|0] \]

\[ P' = K'[R'|t'] \]
Geometry estimation steps

- Feature matches give correspondences to already computed 3D points
- From corresponding 2D and 3D points the pose of the new camera can be computed using the PnP-Algorithm

\[ P = K[I|0] \]
\[ P'' = ? \]
\[ P' = K'[R'|t'] \]
Geometry estimation steps

- Repeat the process starting again from triangulation of new features

\[ P = K[I|0] \]
\[ P'' = K''[R''|t''] \]
\[ P' = K'[R'|t'] \]
Bundle adjustment

- Levenberg-Marquardt optimization of re-projection error
- Parameters are camera poses and all 3D points (millions of parameters to optimize!)

\[
\min_{P_j, X_i} \left( \sum_i \sum_j \| x_{i,j} - P_j X_i \| \right)
\]
3 paradigms

sequential (incremental)

hierarchical

global
Bundle adjustment (BA)

\[ \min_{P_j, M_i} \left( \sum_i \sum_j \| p_{i,j} - P_j M_i \| \right) = \varepsilon = f(x) \]

- Objective function to be minimized

- Gauss-Newton update equation

\[ x_k = x_{k-1} + d_{k-1} \]
\[ J_{k-1}^T J_{k-1} d_{k-1} + J_{k-1}^T \varepsilon_{k-1} = 0 \]
Calculating the update vector $d$

$$J_{k-1}^T J_{k-1} d_{k-1} = -J_{k-1}^T e_{k-1}$$

$$Ax = b$$

- $J$ … npxN matrix (n … #cameras, p … #points, N … #parameters)
- residual vector $e$ is computed from $e = ||x - PM||$ for every iteration
- Then the values for the Jacobian $J$ are computed for every iteration
The Jacobian $J$ (example for 3 cameras and 4 3D points)

- $J$ … npxN matrix (n … #cameras, p … #points, N … #parameters
- $M$ … 1x3 matrix, $P$ … 1x11 matrix

White blocks are non-zero entries
\( J^T J \)

- \( J^T J \) is called the “Hessian Matrix” (symmetric matrix)
- \( U \) … 11x11 symmetric matrix, \( V \) … 3x3 symmetric matrix
- \( W \) … 11x3 matrix

\[
\begin{align*}
J_{k-1}^T J_{k-1} &= \begin{bmatrix}
U_2 & w_{21} & w_{22} \\
U_3 & w_{31} & \ \\
W^T_{21} & W^T_{31} & V_1 \\
W^T_{22} & & V_2 \\
& & V_3 \\
& & V_4
\end{bmatrix} \\
&= \begin{bmatrix}
\begin{bmatrix} d(P_2) \\ d(P_3) \\ d(M_1) \\ d(M_2) \\ d(M_3) \\ d(M_4) \end{bmatrix} \\
\begin{bmatrix} n_2 \\ n_3 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}
\end{bmatrix} \\
d_{k-1} &= -J_{k-1}^T \varepsilon_{k-1}
\end{align*}
\]
Schur complement trick/Sparse BA

\[
\begin{bmatrix}
U & W \\
W^T & V
\end{bmatrix}
\begin{bmatrix}
d(P) \\
d(M)
\end{bmatrix}
= \begin{bmatrix}
n(P) \\
v(M)
\end{bmatrix}
\]

\[
\begin{bmatrix}
I_{11(n-1)} & -WV^{-1} \\
0_{3px11(n-1)} & I_{3p}
\end{bmatrix}
\]

Multiply the above equation with this line to obtain

\[
\begin{bmatrix}
U - WV^{-1}W^T & 0_{3p} \\
W^T & V
\end{bmatrix}
\begin{bmatrix}
d(P) \\
d(M)
\end{bmatrix}
= \begin{bmatrix}
n(P) - WV^{-1}v(M) \\
v(M)
\end{bmatrix}
\]

d(P) and d(M) are separated (first row only contains d(P))

d(P) can be computed solving this equation system of type Ax=b
Only the matrix V needs to be inverted (efficiently possibly because it is block diagonal)

\[
(U - WV^{-1}W^T) d(P) = n(P) - WV^{-1}v(M)
\]

d(M) is computed by back-substitution

\[
d(M) = V^{-1}(v(M) - W^T d(P))
\]