3D Reconstruction Pipelines

GVG 2021 - Lecture 13
• Previously in the GVG lecture:
  • Absolute camera pose estimation (lecture 03)
  • Homograph estimation (lecture 04)
  • Fundamental matrix estimation (lecture 10)
  • Reconstruction from two views (lecture 11)
  • Essential matrix estimation (lecture 12)

• **This lecture**: Putting things together for full 3D reconstruction
Structure-from-Motion (SfM)

Input: images

Output: (sparse) 3D point cloud, camera poses

model computed using Colmap
Sequential / Incremental SfM
Sequential / Incremental SfM

Detect interest points and extract descriptors for them, e.g., SIFT features (see lecture 6)
Sequential / Incremental SfM

- Nearest neighbor search in descriptor space to establish feature matches
- Robust model fitting via RANSAC
Model Fitting With RANdom SAmple Consensus (RANSAC)

\[ \text{While probability of missing correct model } > \eta \]
\[ \text{Estimate model from } n \text{ random data points} \]
\[ \text{Estimate support (}\#\text{inliers / robust cost func.}) \text{ of model} \]
\[ \text{If new best model} \]
\[ \text{Perform Local Optimization (LO)} \]
\[ \text{update best model, } \eta \]

Return: Model with most inliers / lowest cost

- See also USAC [Raguram et al., PAMI'13] [code] (good overview, nice implementation)
- Never use standard RANSAC!

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]

[Chum, Matas, Optimal Randomized RANSAC. PAMI 2008]

[Lebeda, Matas, Chum, Fixing the Locally Optimized RANSAC. BMVC 2012] [code]
RANSAC

2D line fitting example

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

Repeat:

1. Draw Minimal Sample
2. Estimate Model
3. Count Inliers

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

Repeat:

- Draw Minimal Sample
- Estimate Model
- Count Inliers

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

Repeat:

[Draw Minimal Sample]

[Estimate Model]

[Count Inliers]

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]

Torsten Sattler
RANSAC

2D line fitting example

Repeat:

1. Draw Minimal Sample
2. Estimate Model
3. Count Inliers

best number of inliers: 3

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

Repeat:

- Draw Minimal Sample
- Estimate Model
- Count Inliers

best number of inliers: 3

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

[Draw Minimal Sample]
[Estimate Model]
[Count Inliers]

best number of inliers: 3

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

Repeat:
- Draw Minimal Sample
- Estimate Model
- Count Inliers

best number of inliers: 4

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]

Torsten Sattler
RANSAC

2D line fitting example

Repeat:

1. Draw Minimal Sample
2. Estimate Model
3. Count Inliers

best number of inliers: 4

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
2D line fitting example

Repeat:
- Draw Minimal Sample
- Estimate Model
- Count Inliers

best number of inliers: 4

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

Repeat:

1. Draw Minimal Sample
2. Estimate Model
3. Count Inliers

best number of inliers: 4

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

Repeat:

- Draw Minimal Sample
- Estimate Model
- Count Inliers

best number of inliers: 4

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

Repeat:

1. Draw Minimal Sample
2. Estimate Model
3. Count Inliers

best number of inliers: 4

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC

2D line fitting example

Repeat:
1. Draw Minimal Sample
2. Estimate Model
3. Count Inliers

best number of inliers: 5

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC - Termination Criterion

- Let’s assume we know the **inlier ratio** $\varepsilon$ (fraction of inliers)
- Probability of picking an inlier randomly: $\varepsilon$
- Probability of picking $n$ inlier randomly: $\varepsilon^n$
- Probability of non-all inlier sample ($\geq 1$ outlier): $(1-\varepsilon^n)$
- Probability of not picking all-inlier sample in $k$ iterations: $(1-\varepsilon^n)^k$

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
RANSAC - Termination Criterion

- Terminate if \((1-\varepsilon^n)^k < \eta\)
- In practice: Compute maximum number iterations \(k_{\text{max}}\)
  - Find \(k_{\text{max}}\) such that \((1-\varepsilon^n)k_{\text{max}} = \eta\)
  - \(\iff k_{\text{max}} \ln(1-\varepsilon^n) = \ln(\eta)\)
  - \(\iff k_{\text{max}} = \frac{\ln(\eta)}{\ln(1-\varepsilon^n)}\)
  - Note: \(k_{\text{max}}(\varepsilon) > k_{\text{max}}(\varepsilon')\) if \(\varepsilon < \varepsilon'\)
- How do we know inlier ratio \(\varepsilon\)?
- In practice more than \(k_{\text{max}}(\varepsilon)\) steps necessary as not every all-inlier sample leads to best model (due to, e.g., noise, degeneracies, etc.)

[Fischler & Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. CACM 1981]
Sequential / Incremental SfM

- Extract relative rotation and translation from H/E/F matrix
- Use 2D-2D matches to **triangulate** 3D structure
Sequential / Incremental SfM

- Extract relative rotation and translation from H/E/F matrix
- Use 2D-2D matches to **triangulate** 3D structure

\[ R, t \text{ with } ||t|| = 1 \]
Sequential / Incremental SfM

- Extract relative rotation and translation from H/E/F matrix
- Use 2D-2D matches to **triangulate** 3D structure
How to select a good initial pair?

Criteria:
- Accurate relative pose ≈ many inlier matches
- Non-planar scene (planar scenes are degeneracy for F-matrix fitting)
  - Compute both H and E/F matrix
  - Select pair with large ratio #inliers(E/F) / #inliers(H)
- No pure forward motion (triangulation inaccurate / impossible)
- In practice, try out multiple initial pairs
Sequential / Incremental SfM

- Pick image(s) with large number of matches to existing cameras
- Obtain 2D-3D matches from 2D-2D matches
- Estimate absolute pose of new image

Feature Detection

Feature Matching & H/H/E/F Matrix Fitting

Two-View Initialization

Extend Motion
Sequential / Incremental SfM

- Associate existing 3D points with new features
- Triangulate new 3D points for features without associated 3D points

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure
Sequential / Incremental SfM

- Associate existing 3D points with new features
- Triangulate new 3D points for features without associated 3D points

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure
Sequential / Incremental SfM

True trajectory

Feature Detection

Feature Matching & H/H/E/F Matrix Fitting

Two-View Initialization

Extend Motion

Extend Structure

Torsten Sattler
Sequential / Incremental SfM

True trajectory
Estimated trajectory

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure
Sequential / Incremental SfM

True trajectory
Estimated trajectory

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure
Sequential / Incremental SfM

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure

True trajectory
Estimated trajectory
Sequential / Incremental SfM

- Feature Detection
- Feature Matching & H/E/F Matrix Fitting
- Two-View Initialization
- Extend Motion
- Extend Structure

True trajectory
Estimated trajectory
Sequential / Incremental SfM

True trajectory
Estimated trajectory

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure
Sequential / Incremental SfM

True trajectory
Estimated trajectory

- Feature Detection
- Feature Matching & H/H/E/F Matrix Fitting
- Two-View Initialization
- Extend Motion
- Extend Structure

Torsten Sattler
Sequential / Incremental SfM

Feature Detection

Feature Matching & H/H/E/F Matrix Fitting

Two-View Initialization

Extend Motion

Extend Structure

True trajectory

Estimated trajectory
Sequential / Incremental SfM

- Feature Detection
- Feature Matching & H/H/E/F Matrix Fitting
- Two-View Initialization
- Extend Motion
- Extend Structure

True trajectory
Estimated trajectory
Sequential / Incremental SfM

True trajectory
Estimated trajectory

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure

Torsten Sattler
Sequential / Incremental SfM

True trajectory
Estimated trajectory

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure

Torsten Sattler
Sequential / Incremental SfM

- Errors accumulate, leading to drift over time
- Adjust motion and structure frequently

Drift

True trajectory
Estimated trajectory

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure
Sequential / Incremental SfM

- Errors accumulate, leading to drift over time
- Adjust motion and structure frequently

True trajectory
Estimated trajectory

Feature Detection
Feature Matching & H/H/E/F Matrix Fitting
Two-View Initialization
Extend Motion
Extend Structure
Bundle Adjustment
Bundle Adjustment

\[ \rho_j(R_j X_i + t_j) \]

argmin

camera poses, 3D points

Can also refine camera intrinsics

Cost function is highly non-linear → refine from initialization

\[ \sum_i \sum_j \Delta_{ij} \| x_{ij} - \rho_j(R_j X_i + t_j) \|^2 \]

point i visible in image j?

Bundle Adjustment

Feature Detection

Feature Matching & H/H/E/F Matrix Fitting

Two-View Initialization

Extend Motion

Extend Structure

Torsten Sattler
Bundle Adjustment

Gradient descent

\[ \min_{X} f(X) = \min_{X} \sum_{i} \Delta_i^T \Delta_i, \quad \Delta_i = x_i - \begin{pmatrix} \frac{p_i^1 X}{p_3^1 X} \\ \frac{p_i^2 X}{p_3^2 X} \\ \frac{p_i^3 X}{p_3^3 X} \end{pmatrix}, \quad \Delta = \begin{pmatrix} \Delta_1 \\ \vdots \\ \Delta_n \end{pmatrix} \]

Initialization: \( X_k = X_0 \)

Iterate until convergence

Compute gradient: \( \nabla f(X_k) = J^T \Delta \)

Update: \( X_{k+1} = X_k - \eta \nabla f(X_k) \)

\[ J = \frac{\partial f(X)}{\partial X} : \text{Jacobian} \quad \eta : \text{Step size} \]

Slow convergence near minimum point!

slide credit: Gim Hee Lee

Torsten Sattler
Newton’s method

2\textsuperscript{nd} order approximation (quadratic Taylor expansion):

\[ f(X + \delta)|_{x=X_k} = f(X) + \nabla f(X)^T \delta + \frac{1}{2} \delta^T H \delta \bigg|_{x=X_k} \]

Hessian matrix:

\[ H = \frac{\partial^2 f(X + \delta)}{\partial^2 \delta} \bigg|_{x=X_k} \]

Find \( \delta \) that minimizes \( f(X + \delta)|_{x=X_k} \)!
Newton’s method

Differentiate and set to 0 gives:
\[ \delta = -H^{-1}\nabla f(X_k) \]

Update:
\[ X_{k+1} = X_k + \delta \]

Computation of H is not trivial (2\textsuperscript{nd} order derivatives) and optimization might get stuck at saddle point!

slide credit: Gim Hee Lee

Torsten Sattler
Bundle Adjustment

Gauss-Newton

Approximate Hessian matrix by dropping 2nd order terms:

\[ H \approx J^T J \]

Solve normal equation:

\[ J^T J \delta = -J^t \Delta \]

Might get stuck and slow convergence at saddle point!
Bundle Adjustment

Levenberg-Marquardt

Regularized Gauss-Newton with damping factor

\[
\left( J^T J + \lambda I \right) \delta = -J^t \Delta
\]

\[\lambda \to 0\]: Gauss-Newton (when convergence is rapid)

\[\lambda \to \infty\]: Gradient descent (when convergence is slow)

Adapt \(\lambda\) during optimization:

- Decrease \(\lambda\) when function value decreases
- Increase \(\lambda\) otherwise

slide credit: Gim Hee Lee
Bundle Adjustment

Reconstruction of the old inner city of Aachen, Germany, using the Bundler SfM software

slide credit: Gim Hee Lee
Bundle Adjustment

- Not covered here:
  - Sparse structure of the bundle adjustment problem
  - Efficient strategies (e.g., Schur Complement)
  - …

- Recommended reading:
  - Triggs et al., Bundle Adjustment - A Modern Synthesis, 1999

slide credit: Gim Hee Lee
Multi-View Stereo (MVS)

Input: calibrated images, camera poses, SfM model

Output: dense 3D point cloud or (textured) 3D mesh

Model computed using Colmap and Poisson Surface Reconstruction
Multi-View Stereo In a Nutshell

- Use known epipolar relation to find dense matches between images
- Create dense point cloud
3D Reconstruction Packages

- Bundler ([https://github.com/snavely/bundler_sfm](https://github.com/snavely/bundler_sfm))
- Linux (Windows also supported), open source
- SfM pipeline, MVS pipelines can read file format
- Showed nice results on internet photo collections
- Not state-of-the-art anymore
3D Reconstruction Packages

- VisualSFM (http://ccwu.me/vsfm/)
- Linux, Mac OS X, Windows, closed source
- SfM pipeline, interface to external MVS software
- Graphical User Interface
- Very efficient due to use of GPU

https://www.youtube.com/watch?v=5ceiOd8Yx3g
3D Reconstruction Packages

- OpenMVG (https://github.com/openMVG/openMVG)
- Linux, Mac OS X, Windows, open source
- SfM pipeline, MVS pipelines can read file format
- Very modular, functionality not in other packages (full multi-camera support)
- Not very efficient, no GUI
3D Reconstruction Packages

- COLMAP (https://colmap.github.io/index.html)
- Linux, Mac OS X, Windows, open source
- SfM and MVS
- Efficient pipeline, GUI
- High code quality, very great tool for research!
3D Reconstruction Packages

- COLMAP (https://colmap.github.io/index.html)
- Linux, Mac OS X, Windows, open source
- SfM and MVS (NVidia GPU required for MVS)
- Efficient pipeline, GUI
- High code quality, very great tool for research!

Demo!
3D Reconstruction Packages

- AliceVision Meshroom ([https://alicevision.org/](https://alicevision.org/))
- Linux, Windows, open source
- SfM and MVS (NVidia GPU required for both)
- Includes work by Tomas Pajdla and his PhD students
- Have not tried it yet, on my Todo list
3D Reconstruction Packages

- RealityCapture (https://www.capturingreality.com/)
- Commercial software, Windows only
- Start-up (CapturingReality) out of Slovakia, former PhD students at CVUT, recently acquired by Epic Games
- Both SfM and MVS (MVS requires NVidia GPU)
- Highly efficient, SfM takes a few minutes even for large scenes
- Very high quality (probably best software out there)
Results with RealityCapture
Results with RealityCapture
3D Reconstruction Packages

- Many more commercial packages available
  - Agisoft Metashape (https://www.agisoft.com/)
  - Pix4D (https://www.pix4d.com/)
  - ...

Torsten Sattler
Bonus: Neural Radiance Fields (NeRFs)

[Mildenhall, Srinivasan, Tancik, et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020]
Volume Rendering

\[ \hat{C}(r) = \sum_{i=1}^{N} T_i (1 - \exp(-\sigma_i \delta_i)) c_i \]

- **Final color**
- **Visibility**
- **Color at i-th sample**

\[ T_i = \exp \left( -\sum_{j=1}^{i-1} \sigma_j \delta_j \right) \]

- **Small if high density before this sample**
- **Distance to previous sample**

[Mildenhall, Srinivasan, Tancik, et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020]
Neural Radiance Fields (NeRFs)

input: 3D point and ray direction

\((x,y,z,\theta,\phi) \rightarrow (RGB\sigma)\)

output: color and density

Continuous scene representation (vs. discrete voxel volumes)

[Mildenhall, Srinivasan, Tancik, et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020]
Volume Sampling

uniform sampling inside intervals $\rightarrow$ continuous sampling of the volume

subdivide into equally sized intervals

[Mildenhall, Srinivasan, Tancik, et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020]
Hierarchical Volume Sampling

Coarse sampling (coarse network): Uniform sampling in equally-spaced intervals

“Fine” sampling (“fine” network): Sample according to observed densities

All samples are used during volume rendering

[Mildenhall, Srinivasan, Tancik, et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020]
Training

\[
\mathcal{L} = \sum_{\mathbf{r} \in \mathcal{R}} \left[ \left\| \hat{C}_c(\mathbf{r}) - \mathbf{C}(\mathbf{r}) \right\|_2^2 + \left\| \hat{C}_f(\mathbf{r}) - \mathbf{C}(\mathbf{r}) \right\|_2^2 \right]
\]

Trained individually per scene, about 1-2 days on single GPU

[Mildenhall, Srinivasan, Tancik, et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020]
Important Details

\[ \gamma(x, y, z) \rightarrow \text{volume density } \sigma \rightarrow \text{256D feature} \]

\[ \gamma(\theta, \phi) \rightarrow \text{color } c \]

density does not depend on viewing direction

Input encoding

[Tancik, Srinivasan, Mildenhall, et al., Fourier Features: Let Networks Learn High Frequency Functions in Low Dimensional Domains, NeurIPS 2020]

\[ \gamma(p) = \left( \sin(2^0 \pi p), \cos(2^0 \pi p), \ldots, \sin(2^{L-1} \pi p), \cos(2^{L-1} \pi p) \right) \]

Helps network preserve high-frequency variations, similar in spirit to positional encoding of transformers

[Mildenhall, Srinivasan, Tancik, et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020]
Neural Radiance Fields (NeRFs)

Synthetic Scenes

[Mildenhall, Srinivasan, Tancik, et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020]