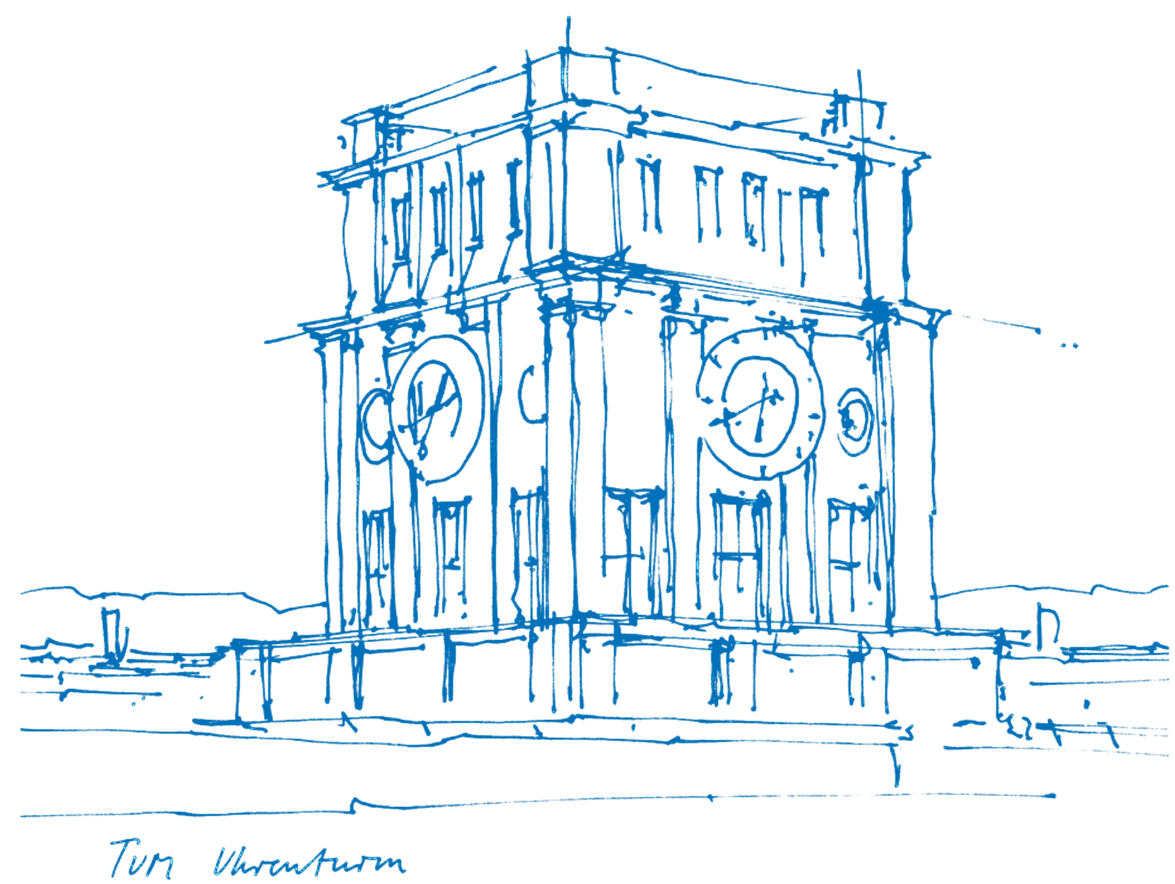
Practical Course: Vision Based Navigation

Lecture 4: Structure from Motion (SfM)

Jason Chui, Simon Klenk, Sergei Solonets
Prof. Dr. Daniel Cremers



Version: 14.11.2022

Topics Covered

- Introduction
  - Structure from Motion (SfM)
  - Simultaneous Localization and Mapping (SLAM)

- Bundle Adjustment
  - Energy Function
  - Non-linear Least Squares
  - Exploiting the Sparse Structure

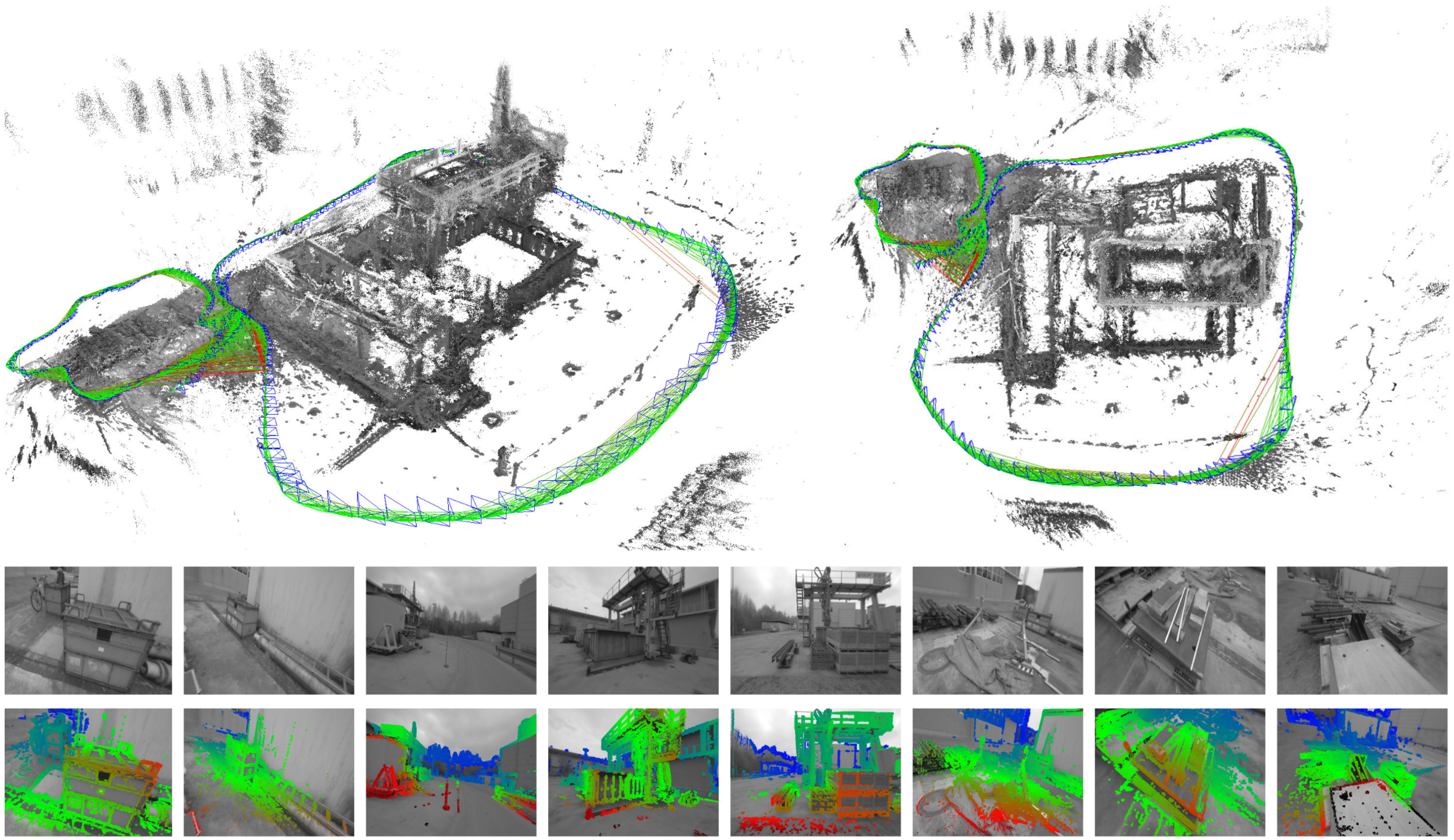
- Triangulation

Structure from Motion

Agarwal et al., “Building Rome in a day”, ICCV 2009, “Dubrovnik” image set

- 3D reconstruction using a set of unordered images
- Requires estimation of 6DoF poses

Simultaneous Localization and Mapping (SLAM)



Engel et al., “LSD-SLAM: Large-Scale Direct Monocular SLAM”, ECCV 2014

- Estimate 6DoF poses and map from sequential image data
- Update once new frames arrive

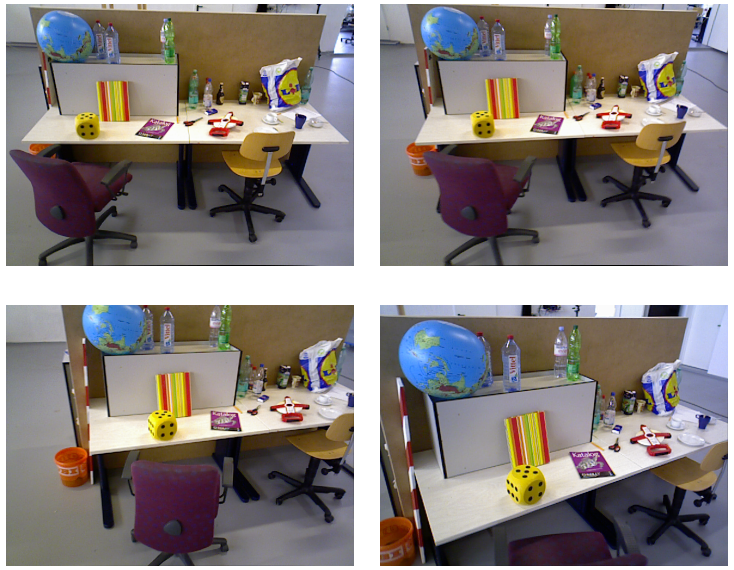
Problem Definition SfM / Visual SLAM

Estimate camera poses and map from a set of images

- Input

  Set of images $I_{0:t} = \{I_0, I_1, \ldots, I_t\}$

  Additional input possible
  - Stereo
  - Depth
  - Inertial measurements
  - Control input

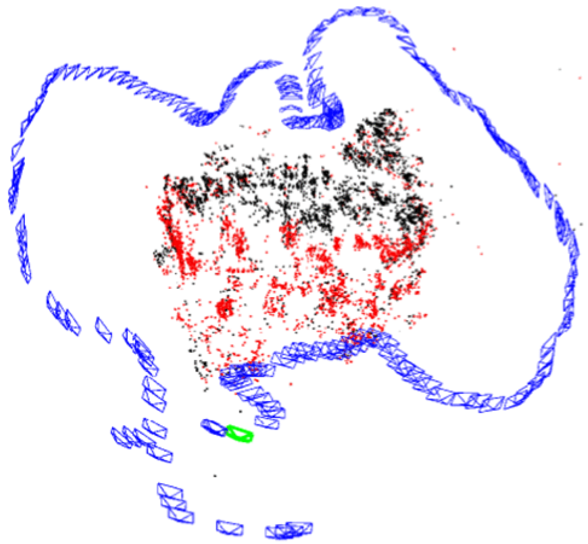


fr3/long_office_household sequence,
TUM RGB-D benchmark

- Output

  Camera pose estimates $\mathbf{T}_i \in \mathrm{SE}(3)$,
  also written as $\boldsymbol{\xi}_i = \left(\log \mathbf{T}_i\right)^{\vee}$ $\quad i \in \{0,1,\ldots,t\}$

  Environment map M



Mur-Artal et al., 2015

Typical SfM Pipeline

1) **Map initialization**
   - Using 2D-to-2D correspondences
   - Recover pose (stereo pair if available)
   - Triangulate landmarks using pose

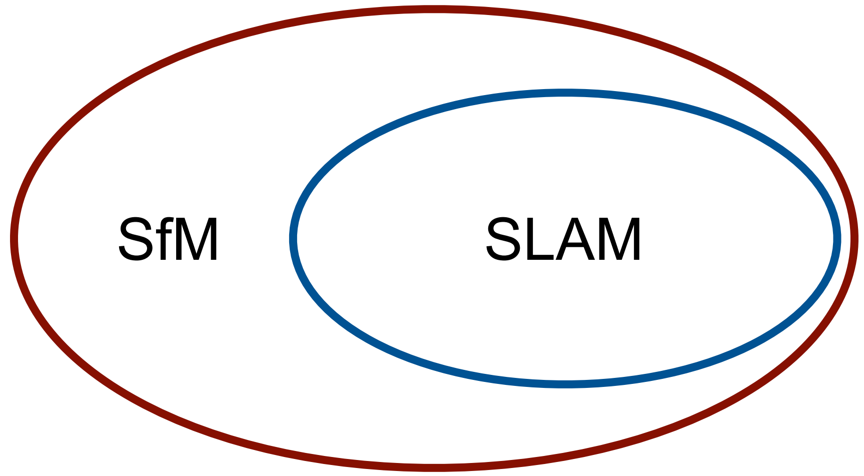
2) **Localization** with known map
   - Using 2D-to-3D correspondences

3) **Mapping** with known poses
   - Using 2D-to-2D correspondences
     → Triangulation

4) **Joint refinement** of map and poses
   - Using 2D-to-2D correspondences
     → Bundle adjustment

Iterate to add new frames

Visual SLAM



SLAM ⊂ SfM, with special focus:

- Sequential image data
- Data arrives sequentially
- Preferably realtime
- More focus on trajectory

Technical solutions:

- Windowed optimization
- Selection of keyframes
- Removal of keyframes (e.g. marginalization)

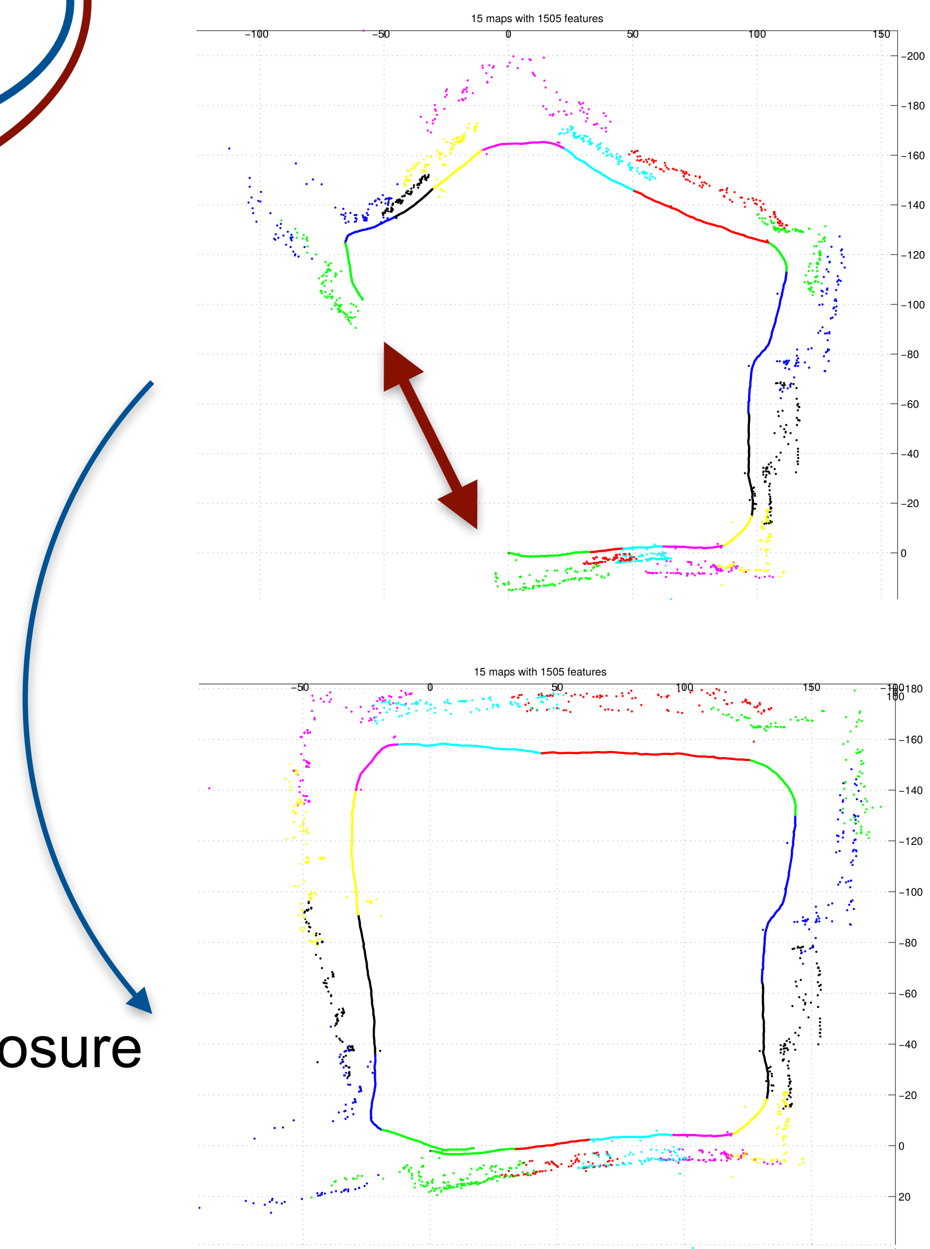
→ Accumulation of drift

- Detect loop closures
- Global mapping in separate thread (e.g. pose graph optimization)

↕

Odometry

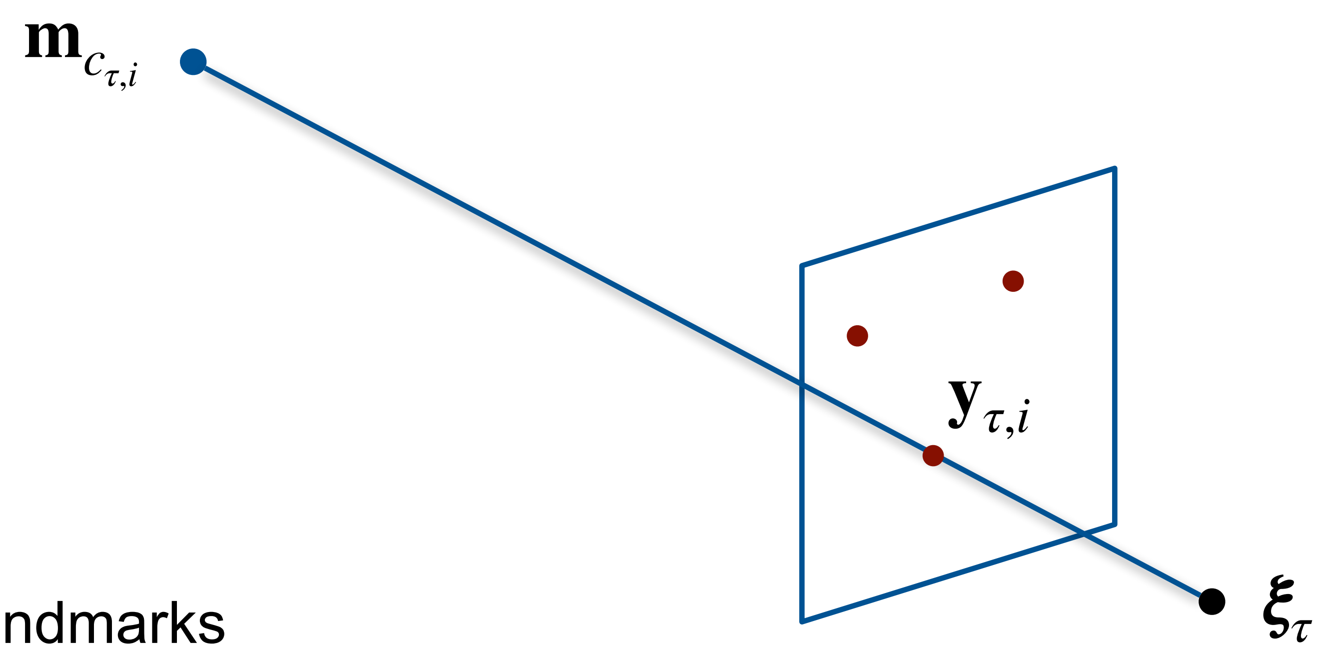
- No global mapping
- Incremental tracking only
- Local map possible



Loop closure

Clemente et al., RSS 2007

Landmarks and Features



- The map consists of 3D locations of landmarks

$$M = \{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_S\}$$

- For image τ, the set of 2D image coordinates of detected features is denoted

$$Y_\tau = \{\mathbf{y}_{\tau,1}, \mathbf{y}_{\tau,2}, \ldots, \mathbf{y}_{\tau,N}\}$$

- Known data association:
  Feature i in image τ corresponds to landmark $j = c_{\tau,i}$ $\qquad (1 \leq i \leq N, 1 \leq j \leq S)$

Bundle Adjustment Energy

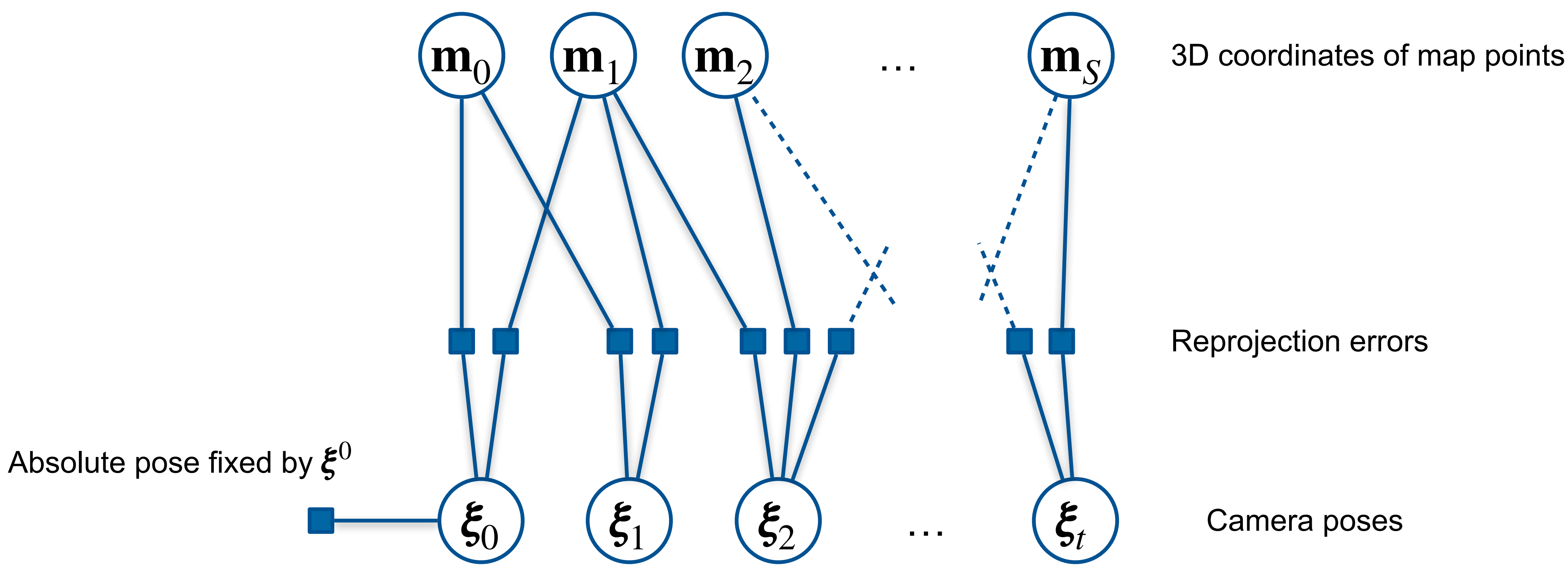
$$E\left(\boldsymbol{\xi}_{0:t}, M\right) = \frac{1}{2}\left(\boldsymbol{\xi}_0 \ominus \boldsymbol{\xi}^0\right)^\top \boldsymbol{\Sigma}_{0,\xi}^{-1}\left(\boldsymbol{\xi}_0 \ominus \boldsymbol{\xi}^0\right)$$

Absolute pose prior

$$+\frac{1}{2}\sum_{\tau=0}^{t}\sum_{i=1}^{N_\tau}\left(\mathbf{y}_{\tau,i} - h\left(\boldsymbol{\xi}_\tau, \mathbf{m}_{c_{\tau,i}}\right)\right)^\top \boldsymbol{\Sigma}_{\mathbf{y}_{\tau,i}}^{-1}\left(\mathbf{y}_{\tau,i} - h\left(\boldsymbol{\xi}_\tau, \mathbf{m}_{c_{\tau,i}}\right)\right)$$

Reprojection error

- Pose prior: Fix absolute pose ambiguity
  - In this case equivalent to keeping $\boldsymbol{\xi}_0 = \boldsymbol{\xi}^0$
  - Keep absolute pose information e.g. when first frame is marginalized
- Additional prior to fix scale ambiguity might be necessary



Energy Function as Non-linear Least Squares

- Residuals as function of state vector $\mathbf{x}$

$$\mathbf{r}^0(\mathbf{x}) := \boldsymbol{\xi}_0 \ominus \boldsymbol{\xi}^0$$

$$\mathbf{r}^y_{t,i}(\mathbf{x}) := \mathbf{y}_{t,i} - h\left(\boldsymbol{\xi}_t, \mathbf{m}_{c_{t,i}}\right)$$

$$\mathbf{x} := \begin{pmatrix} \boldsymbol{\xi}_0 \\ \vdots \\ \boldsymbol{\xi}_t \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_S \end{pmatrix}$$

- Stack the residuals in a vector-valued function und collect the residual covariances on the diagonal blocks of a square matrix

$$\mathbf{r}(\mathbf{x}) := \begin{pmatrix} \mathbf{r}^0(\mathbf{x}) \\ \mathbf{r}^{\mathbf{y}}_{0,1}(\mathbf{x}) \\ \vdots \\ \mathbf{r}^{\mathbf{y}}_{t,N_t}(\mathbf{x}) \end{pmatrix} \qquad \mathbf{W} := \begin{pmatrix} \boldsymbol{\Sigma}^{-1}_{0,\xi} & 0 & \cdots & 0 \\ 0 & \boldsymbol{\Sigma}^{-1}_{\mathbf{y}_{0,1}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \boldsymbol{\Sigma}^{-1}_{\mathbf{y}_{t,N_t}} \end{pmatrix}$$

- Rewrite energy function as $E(\mathbf{x}) = \frac{1}{2}\mathbf{r}(\mathbf{x})^\top \mathbf{W} \mathbf{r}(\mathbf{x})$

Recap: Gauss-Newton Method

- Idea: Approximate Newton's method to minimize $E(\mathbf{x})$
  - Approximate $E(\mathbf{x})$ through linearization of residuals

$$\begin{aligned}\tilde{E}(\mathbf{x}) &= \frac{1}{2}\tilde{\mathbf{r}}(\mathbf{x})^\top \mathbf{W}\tilde{\mathbf{r}}(\mathbf{x}) \\ &= \frac{1}{2}\Big(\mathbf{r}\left(\mathbf{x}_k\right) + \mathbf{J}_k\left(\mathbf{x}-\mathbf{x}_k\right)\Big)^\top \mathbf{W}\Big(\mathbf{r}\left(\mathbf{x}_k\right) + \mathbf{J}_k\left(\mathbf{x}-\mathbf{x}_k\right)\Big) \\ &= \frac{1}{2}\mathbf{r}\left(\mathbf{x}_k\right)^\top \mathbf{W}\mathbf{r}\left(\mathbf{x}_k\right) + \underbrace{\mathbf{r}\left(\mathbf{x}_k\right)^\top \mathbf{W}\mathbf{J}_k}_{=:\mathbf{b}_k^\top}\left(\mathbf{x}-\mathbf{x}_k\right) + \frac{1}{2}\left(\mathbf{x}-\mathbf{x}_k\right)^\top \underbrace{\mathbf{J}_k^\top \mathbf{W}\mathbf{J}_k}_{=:\mathbf{H}_k}\left(\mathbf{x}-\mathbf{x}_k\right)\end{aligned}$$

k iteration index

$$\mathbf{J}_k := \nabla_{\mathbf{x}}\mathbf{r}(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k}$$

- Finding root of gradient as in Newton's method leads to update rule

$$\nabla_{\mathbf{x}}\tilde{E}(\mathbf{x}) = \mathbf{b}_k^\top + \left(\mathbf{x}-\mathbf{x}_k\right)^\top \mathbf{H}_k$$

$$\nabla_{\mathbf{x}}\tilde{E}(\mathbf{x}) = 0 \qquad \text{iff} \qquad \mathbf{x} = \mathbf{x}_k - \mathbf{H}_k^{-1}\mathbf{b}_k$$

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{H}_k^{-1}\mathbf{b}_k}$$

- Pros:
  - Faster convergence than gradient descent (approx. quadratic convergence rate)
- Cons:
  - Divergence if too far from local optimum ($\mathbf{H}$ not positive definite)
  - Solution quality depends on initial guess

Structure of the Bundle Adjustment Problem

- $\mathbf{b}_k$ and $\mathbf{H}_k$ sum terms from individual residuals:

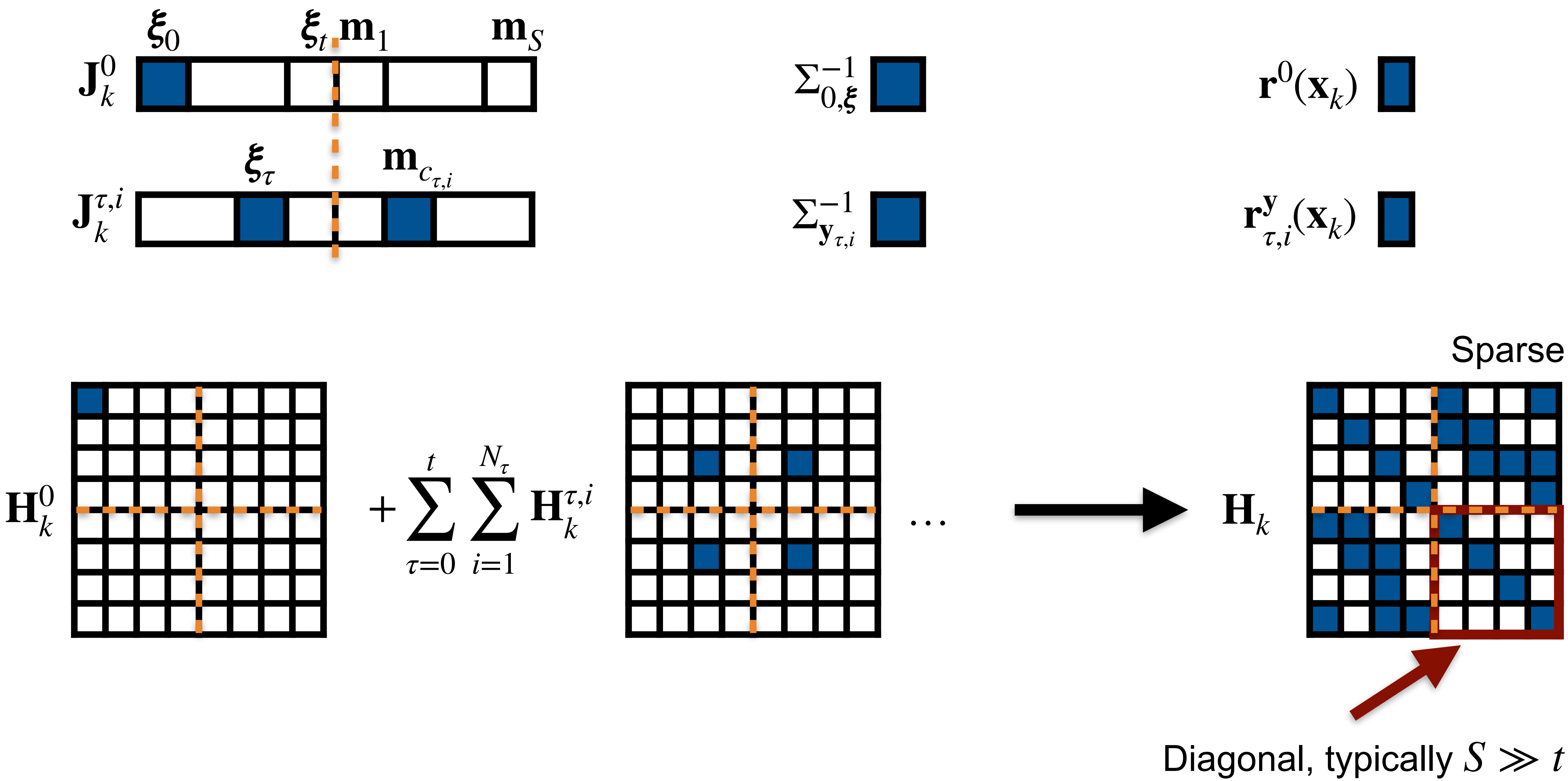
$$\mathbf{b}_k = \mathbf{b}_k^0 + \sum_{\tau=0}^{t} \sum_{i=1}^{N_\tau} \mathbf{b}_k^{\tau,i} = \left(\mathbf{J}_k^0\right)^\top \boldsymbol{\Sigma}_{0,\boldsymbol{\xi}}^{-1} \mathbf{r}^0\left(\mathbf{x}_k\right) + \sum_{\tau=0}^{t} \sum_{i=1}^{N_\tau} \left(\mathbf{J}_k^{\tau,i}\right)^\top \boldsymbol{\Sigma}_{\mathbf{y}_{\tau,i}}^{-1} \mathbf{r}_{\tau,i}^{\mathbf{y}}\left(\mathbf{x}_k\right)$$

$$\mathbf{H}_k = \mathbf{H}_k^0 + \sum_{\tau=0}^{t} \sum_{i=1}^{N_\tau} \mathbf{H}_k^{\tau,i} = \left(\mathbf{J}_k^0\right)^\top \boldsymbol{\Sigma}_{0,\boldsymbol{\xi}}^{-1} \left(\mathbf{J}_k^0\right) + \sum_{\tau=0}^{t} \sum_{i=1}^{N_\tau} \left(\mathbf{J}_k^{\tau,i}\right)^\top \boldsymbol{\Sigma}_{\mathbf{y}_{\tau,i}}^{-1} \left(\mathbf{J}_k^{\tau,i}\right)$$

$\mathbf{J}_k^0$	Jacobian of pose prior
$\mathbf{J}_k^{\tau,i}$	Jacobian of residuals for feature i in image τ

- What is the structure of these terms?

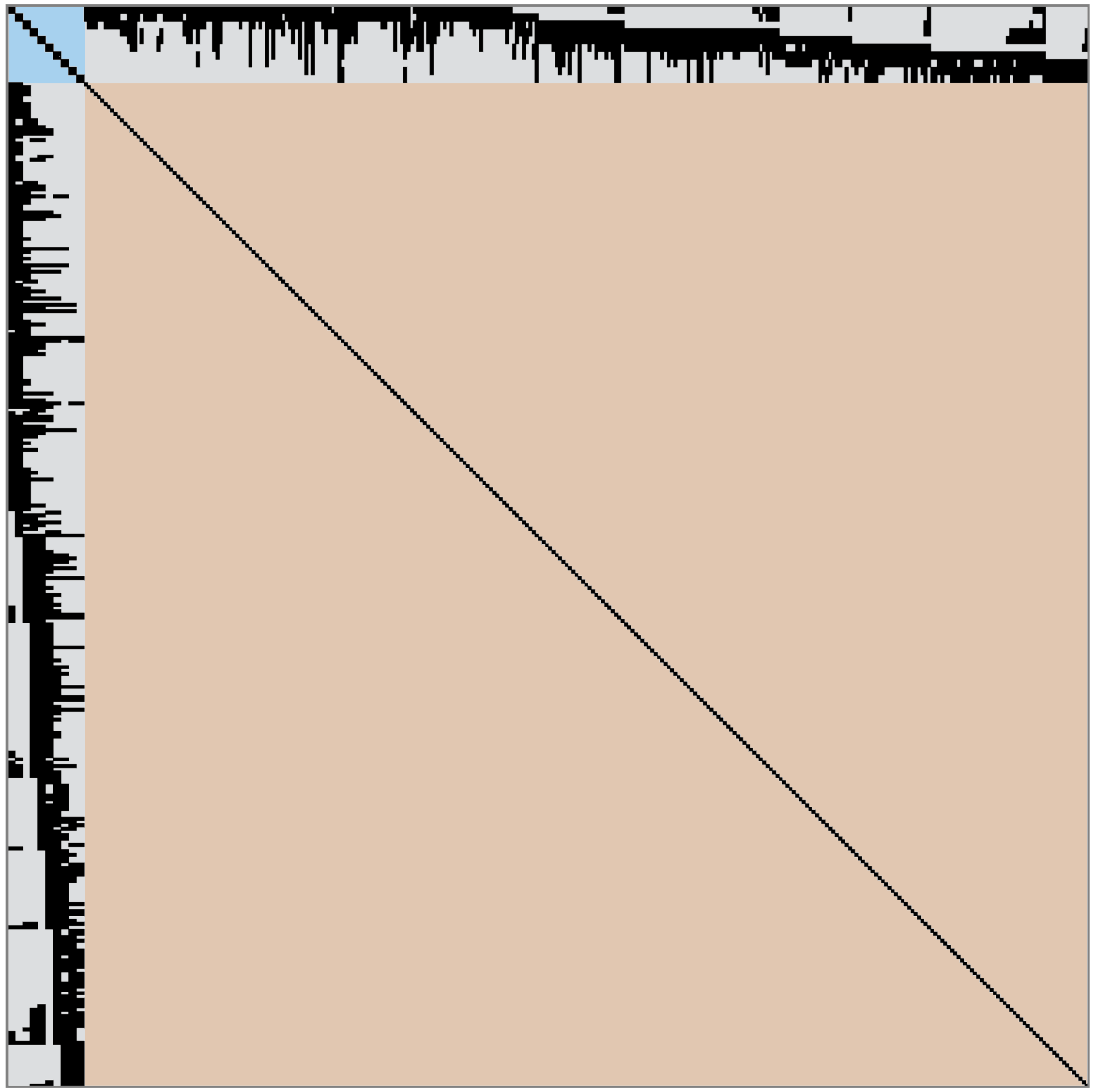
Structure of the Bundle Adjustment Problem



$$\mathbf{H}_k = \mathbf{H}_k^0 + \sum_{\tau=0}^{t}\sum_{i=1}^{N_\tau} \mathbf{H}_k^{\tau,i} = \left(\mathbf{J}_k^0\right)^\top \boldsymbol{\Sigma}_{0,\xi}^{-1} \left(\mathbf{J}_k^0\right) + \sum_{\tau=0}^{t}\sum_{i=1}^{N_\tau} \left(\mathbf{J}_k^{\tau,i}\right)^\top \boldsymbol{\Sigma}_{\mathbf{y}_{\tau,i}}^{-1} \left(\mathbf{J}_k^{\tau,i}\right)$$

Example Hessian of a BA Problem

$$H_k =$$



Lourakis et al., 2009

Large, but sparse!

How to invert efficiently?

Exploiting the Sparse Structure

- Idea:
  Apply the Schur complement to solve the system in a partitioned way

$$\mathbf{H}_k \Delta\mathbf{x} = -\mathbf{b}_k \quad \longrightarrow \quad \begin{pmatrix} \mathbf{H}_{\xi\xi} & \mathbf{H}_{\xi\mathrm{m}} \\ \mathbf{H}_{\mathrm{m}\xi} & \mathbf{H}_{\mathrm{mm}} \end{pmatrix} \begin{pmatrix} \Delta\mathbf{x}_{\xi} \\ \Delta\mathbf{x}_{\mathrm{m}} \end{pmatrix} = -\begin{pmatrix} \mathbf{b}_{\xi} \\ \mathbf{b}_{\mathrm{m}} \end{pmatrix}$$

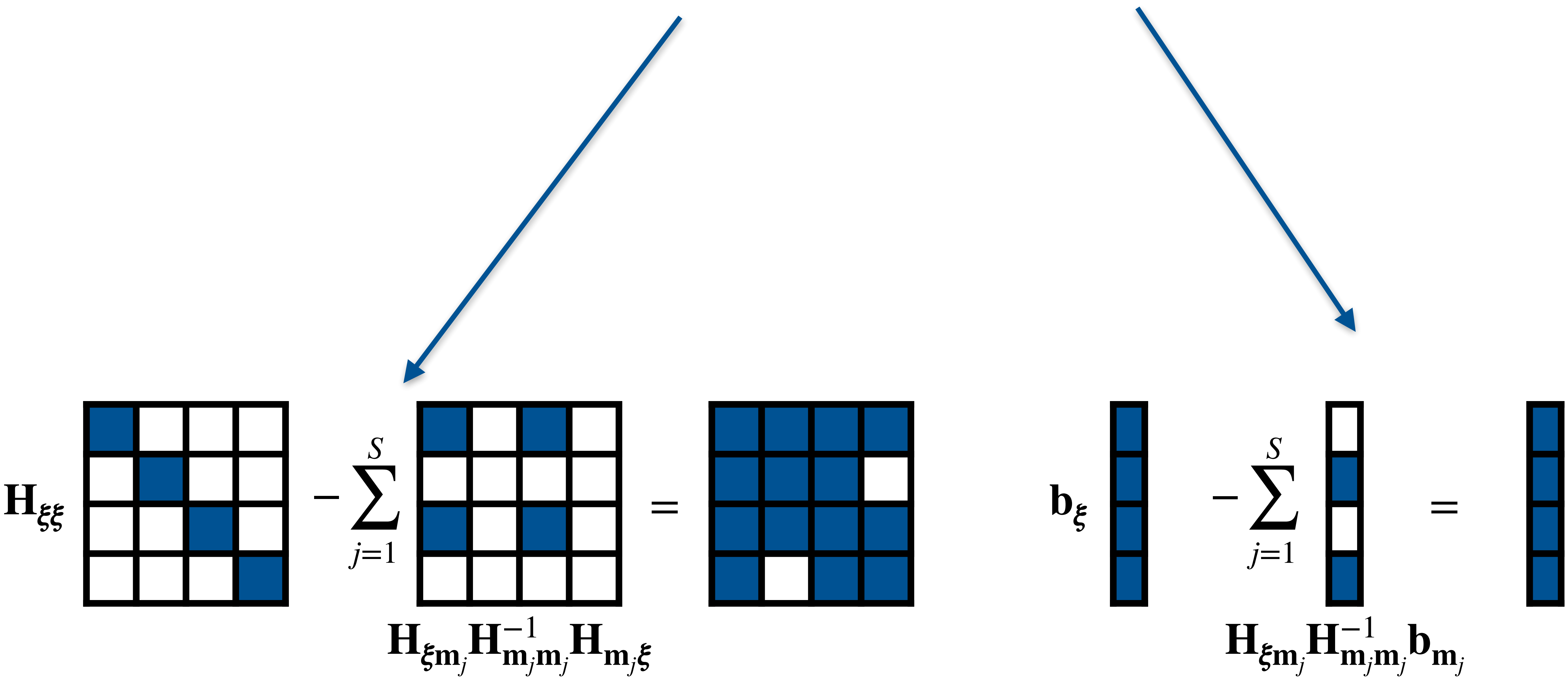
$$\longrightarrow \quad \Delta\mathbf{x}_{\xi} = -\left(\mathbf{H}_{\xi\xi} - \mathbf{H}_{\xi\mathrm{m}}\mathbf{H}_{\mathrm{mm}}^{-1}\mathbf{H}_{\mathrm{m}\xi}\right)^{-1}\left(\mathbf{b}_{\xi} - \mathbf{H}_{\xi\mathrm{m}}\mathbf{H}_{\mathrm{mm}}^{-1}\mathbf{b}_{\mathrm{m}}\right)$$

$$\longrightarrow \quad \Delta\mathbf{x}_{\mathrm{m}} = -\mathbf{H}_{\mathrm{mm}}^{-1}\left(\mathbf{b}_{\mathrm{m}} + \mathbf{H}_{\mathrm{m}\xi}\Delta\mathbf{x}_{\xi}\right)$$

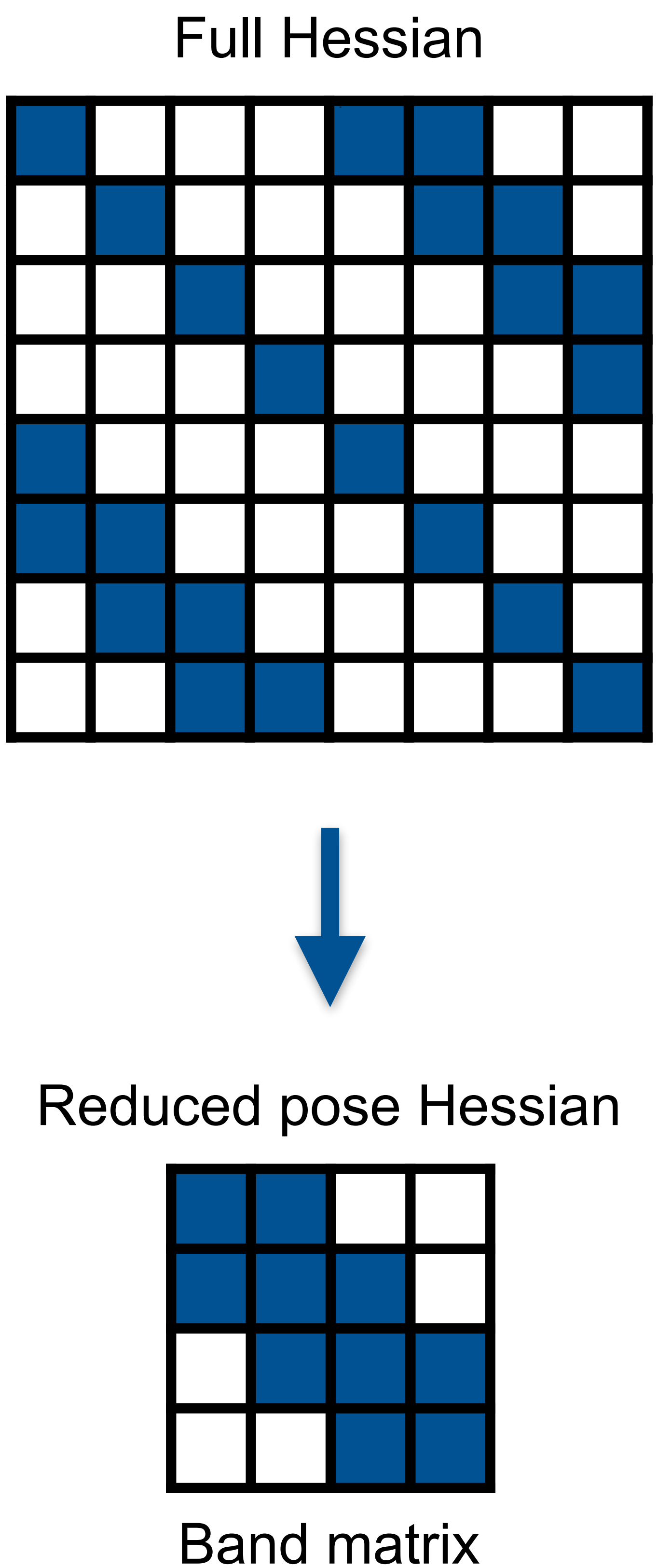
- Is this any better?

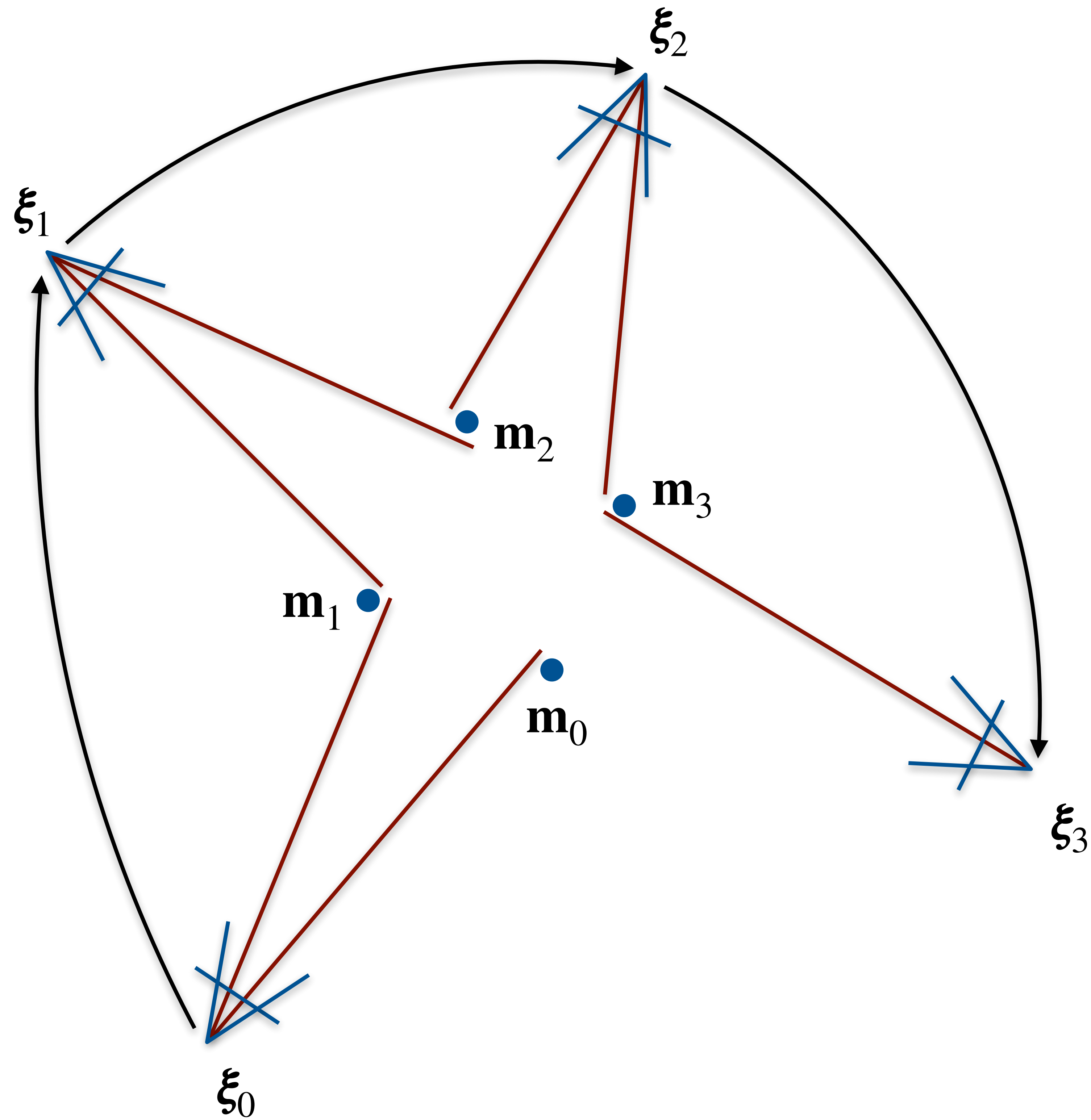
Exploiting the Sparse Structure

$$\Delta \mathbf{x}_\xi = -\left(\mathbf{H}_{\xi\xi} - \mathbf{H}_{\xi\mathbf{m}}\mathbf{H}_{\mathbf{mm}}^{-1}\mathbf{H}_{\mathbf{m}\xi}\right)^{-1}\left(\mathbf{b}_\xi - \mathbf{H}_{\xi\mathbf{m}}\mathbf{H}_{\mathbf{mm}}^{-1}\mathbf{b}_\mathbf{m}\right)$$



Effect of Loop Closures on the Hessian

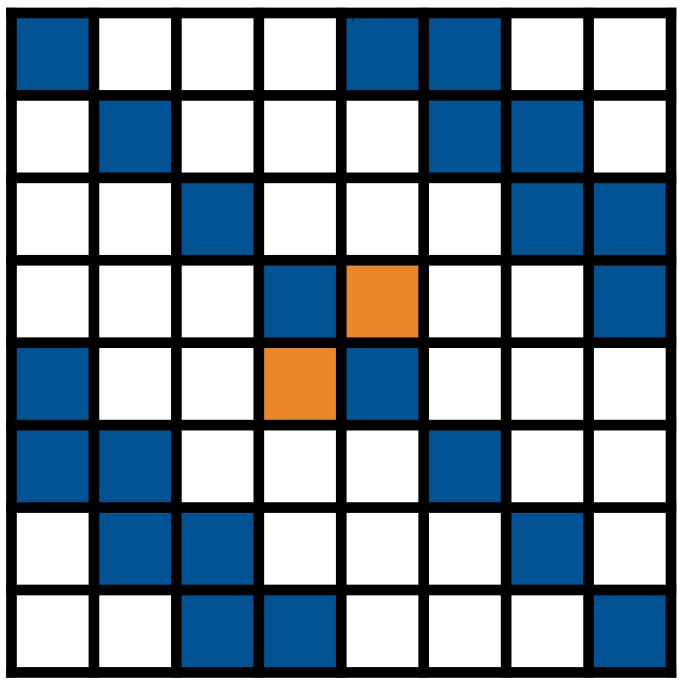




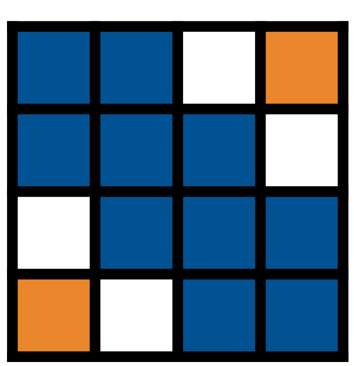
Before loop closure

Effect of Loop Closures on the Hessian

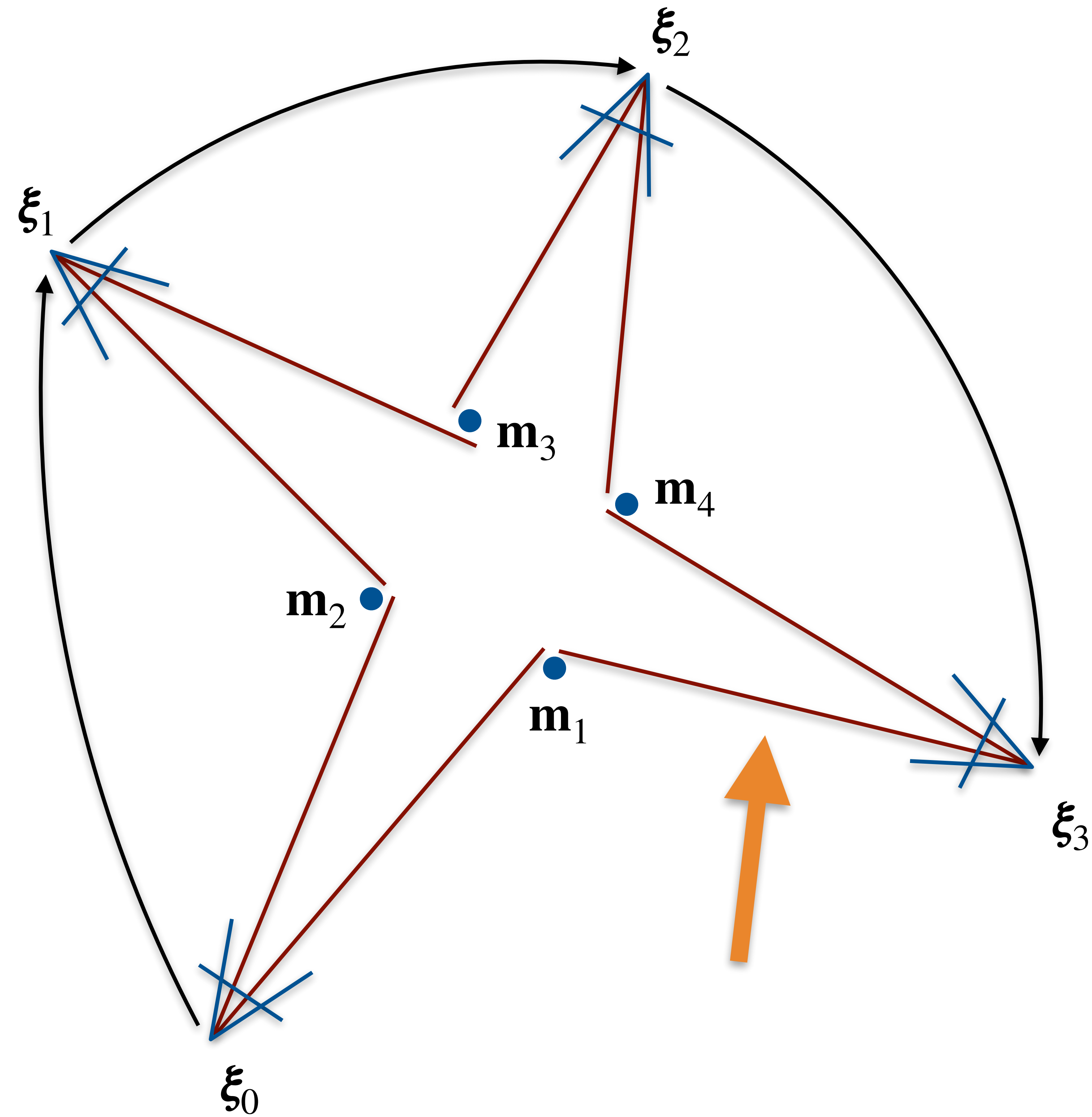
Full Hessian



Reduced pose Hessian



No band matrix: costlier to solve



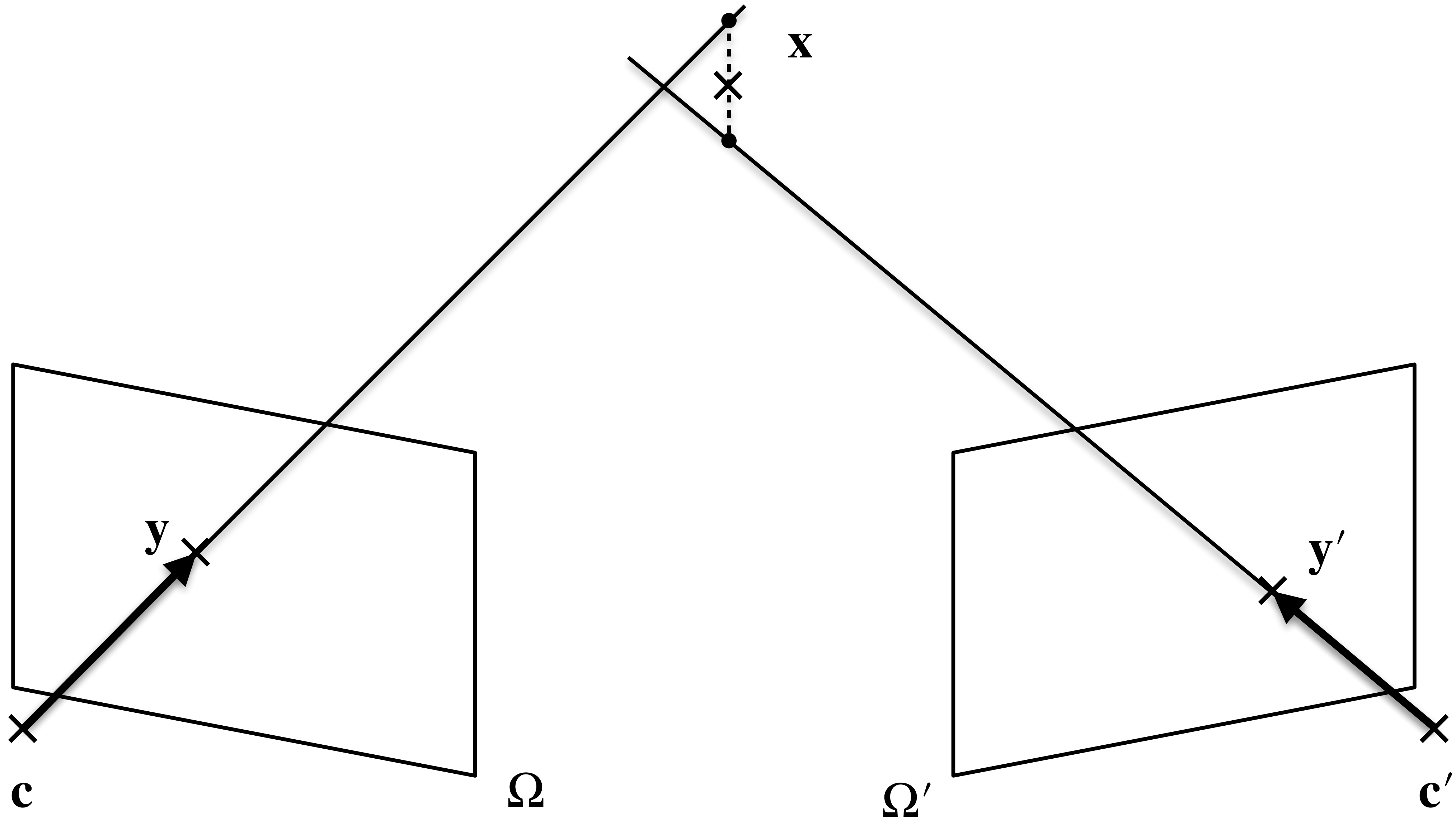
After loop closure

Further Considerations

Many methods to improve convergence / robustness / run-time efficiency, e.g.

- Use matrix decompositions (e.g. Cholesky) to perform inversions
- Levenberg-Marquardt optimization improves basin of convergence
- Heavier-tail distributions / robust norms on the residuals can be implemented using iteratively reweighted least squares
- Preconditioning
- Hierarchical optimization
- Variable reordering
- Delayed relinearization

Triangulation



- Find landmark position given the camera poses
- Ideally, the rays should intersect
- In practice, many sources of error: pose estimates, feature detections and camera model / intrinsic parameters

Triangulation

- Goal: Reconstruct 3D point $\tilde{\mathbf{x}} = (x, y, z, w)^\top \in \mathbb{P}^3$ from 2D image observations $\{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$ for known camera poses $\{\mathbf{T}_1, \ldots, \mathbf{T}_N\}$

- Linear solution: Find 3D point such that reprojections equal its projection
  - For each image i, let $\mathbf{T}_i = \begin{pmatrix} & \mathbf{p}_1 & & \\ & \mathbf{p}_2 & & \\ & \mathbf{p}_3 & & \\ 0 & 0 & 0 & 1 \end{pmatrix}$ and $\mathbf{y}_i = \begin{pmatrix} u \\ v \end{pmatrix}$
  - Projecting $\tilde{\mathbf{x}}$ yields $\mathbf{y}'_i = \pi\left(\mathbf{T}_i \tilde{\mathbf{x}}\right) = \begin{pmatrix} \mathbf{p}_1\tilde{\mathbf{x}}/\mathbf{p}_3\tilde{\mathbf{x}} \\ \mathbf{p}_2\tilde{\mathbf{x}}/\mathbf{p}_3\tilde{\mathbf{x}} \end{pmatrix}$
  - Requiring $\mathbf{y}'_i = \mathbf{y}_i$ gives two linear equations per image: $\begin{aligned} \mathbf{p}_1\tilde{\mathbf{x}} &= u\mathbf{p}_3\tilde{\mathbf{x}} \\ \mathbf{p}_2\tilde{\mathbf{x}} &= v\mathbf{p}_3\tilde{\mathbf{x}} \end{aligned}$
  - Leads to system of linear equations $\mathbf{A}\tilde{\mathbf{x}} = \mathbf{0}$, two approaches to solve:
    - Set $w = 1$ and solve non-homogeneous least squares problem
    - Find nullspace of $\mathbf{A}$ using SVD, then scale such that $w = 1$

- Non-linear least squares on reprojection errors (more accurate): $\min_{\mathbf{x}} \left\{ \sum_{i=1}^{N} \|\mathbf{y}_i - \mathbf{y}'_i\|_2^2 \right\}$

- Different solutions for different methods in the presence of noise

Exercises

Exercise sheet 4

- Implement components of SfM pipeline
- BA: Ceres can do the Schur complement →
- Triangulation: use OpenGV's triangulate function

```
ceres::Solver::Options ceres_options;
ceres_options.max_num_iterations = 20;
ceres_options.linear_solver_type =
ceres::SPARSE_SCHUR;
ceres_options.num_threads = 8;
ceres::Solver::Summary summary;
Solve(ceres_options, &problem,
&summary);
std::cout << summary.FullReport() <<
std::endl;
```

Next slide

Exercise sheet 5

- Implement components of odometry
- Similar to sheet 4, but:
  - More efficient 2D-3D matching using approximate pose of previous frame
  - New keyframe if number of matches too small
  - New landmarks by triangulation from stereo pair
  - Keep runtime bounded by removing old keyframes

Summary

SfM

- Estimate map and camera poses from set of images
- SLAM: Sequential data, real-time
- Odometry: No global mapping

Bundle Adjustment

- Non-linear least squares problem
- Sparse structure of Hessian can be exploited for efficient inversion

Triangulation

- Linear and non-linear algorithms
- Differences in the presence of noise