ETH zürich

Bundle Adjustment + OKVIS

Marco Karrer

V4RL VISION FOR ROBOTICS LAB

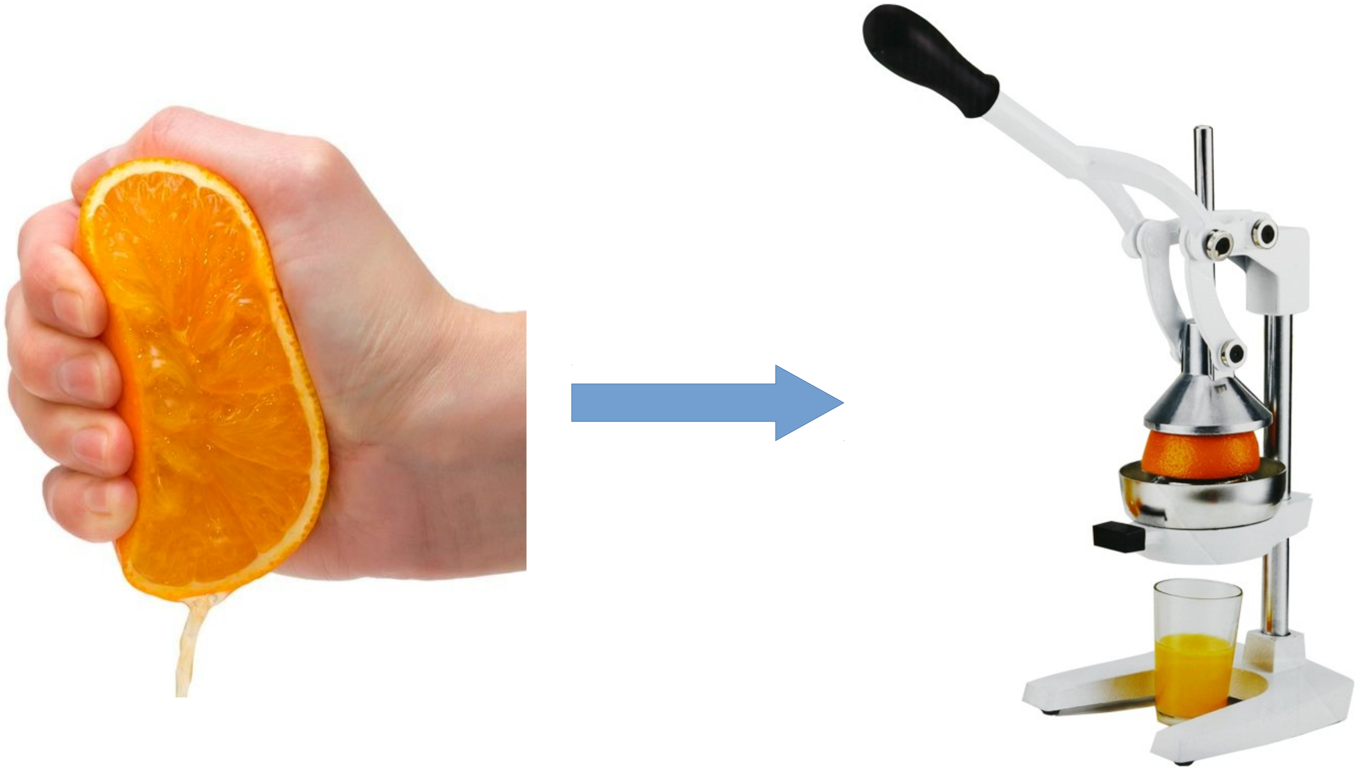
Content

1) What is Bundle Adjustment and why is it important
2) Problem Formulation
    1) General Formulation
    2) Cost Functions
3) Solving the Problem
4) Application of BA in VI-SLAM (OKVIS)
5) Relation to my work

What is Bundle Adjustment?

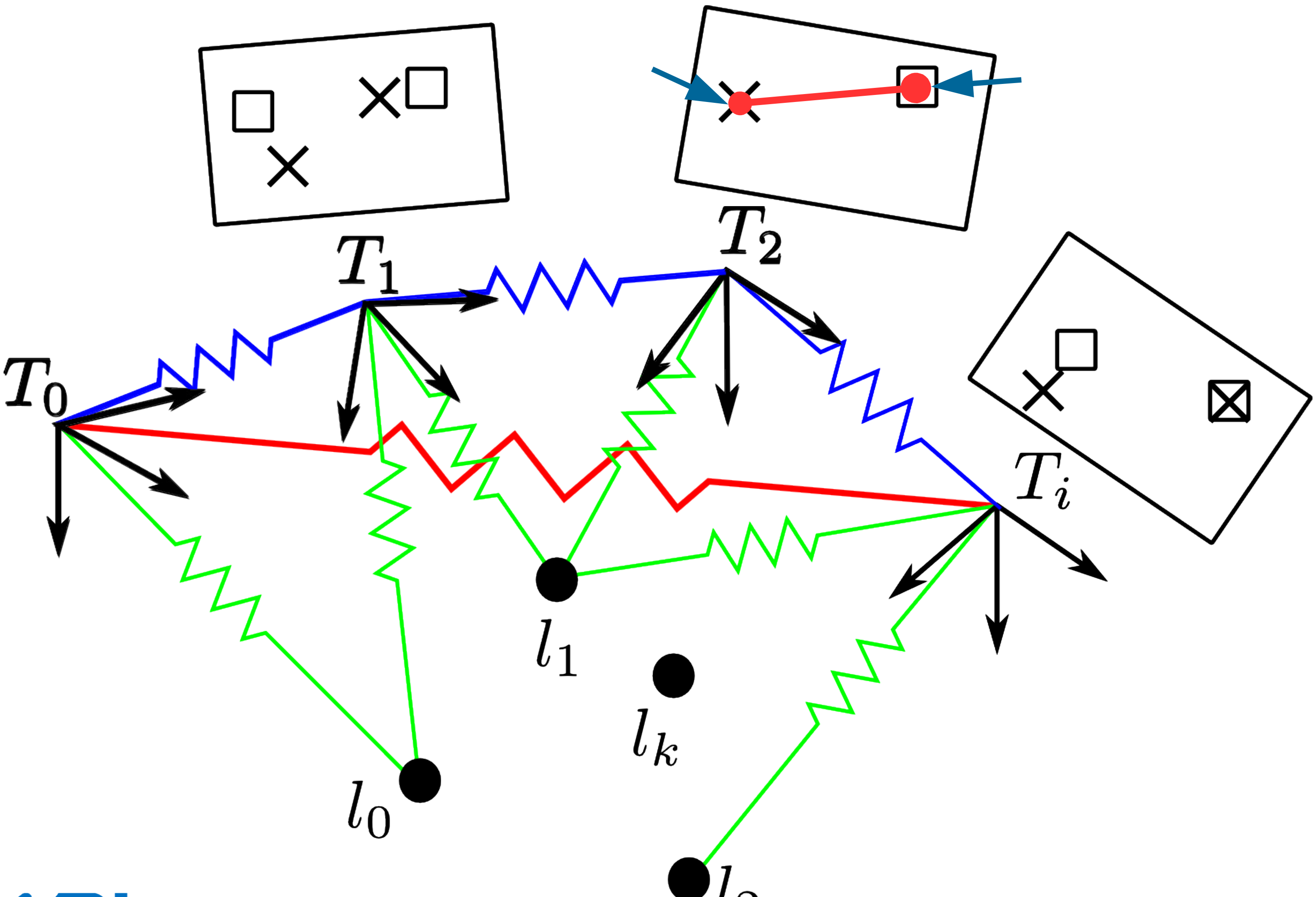
- Bundle adjustment is the problem of refining a visual reconstruction to produce jointly optimal 3D structure and viewing parameter (camera pose and/or calibration) estimates.
- In a more general view, BA is simply a non-linear optimization

Why do we need Bundle Adjustment?



- Get as good as possible estimate out of the available data

Intuitive Explaination of Bundle Adjustment



Objective Function

- General (non-linear) function of camera poses (T_k) and landmark positions (l_i)

$$\min_{T,l} \sum_{i=1}^{I} \sum_{k=1}^{K} \rho \left(\boldsymbol{z}_{i,k} - \boldsymbol{h}(T_i, l_k) \right)$$

ρ:= Cost function

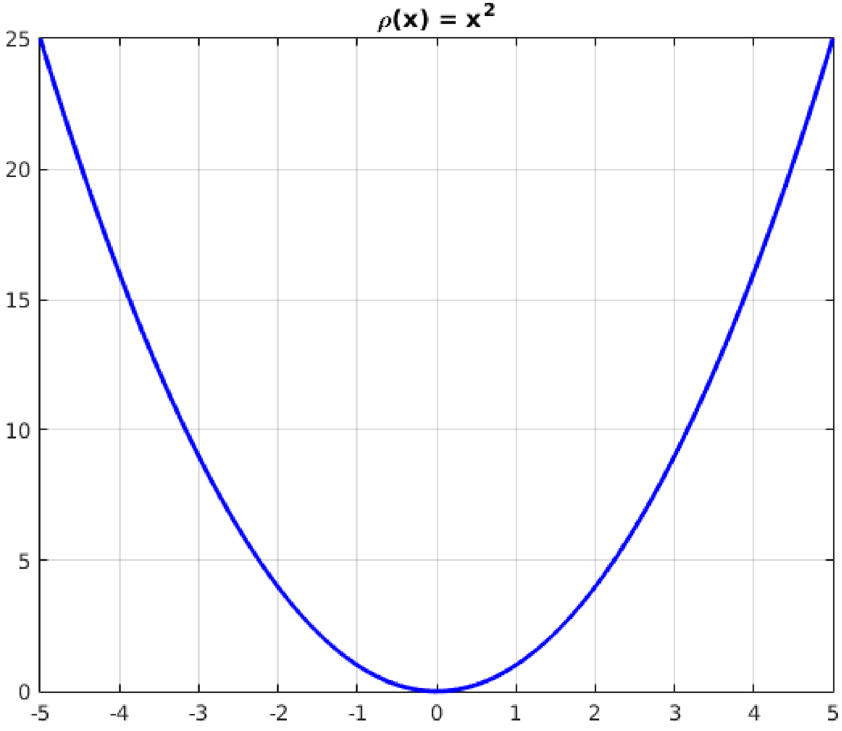
h:= Camera projection model

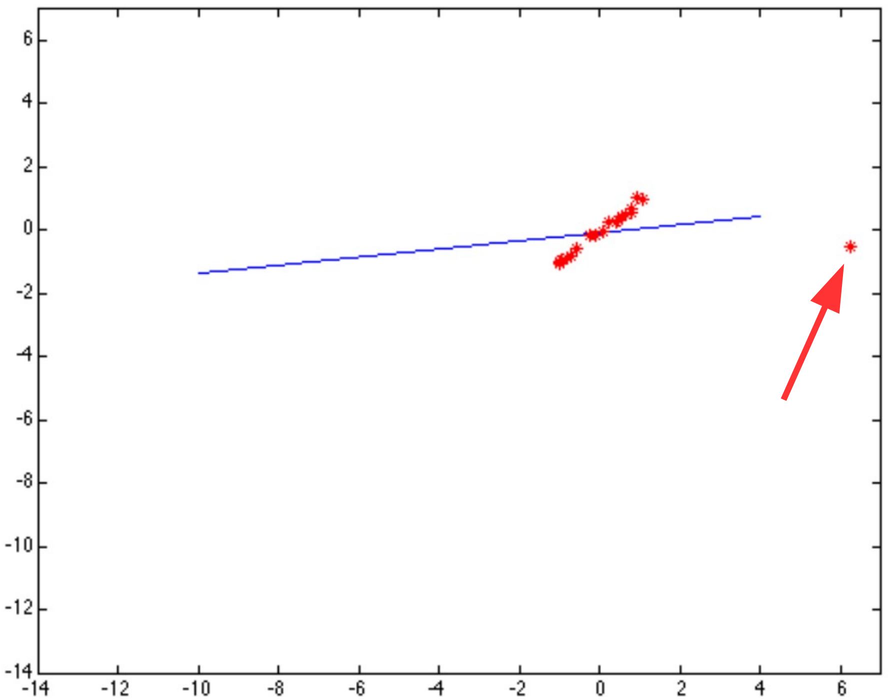
$\mathbf{z}_{i,k}$:= Measurement of landmark k in image i

- Formulation as (non-linear) least squares

Objective Function: Influence of Cost Function

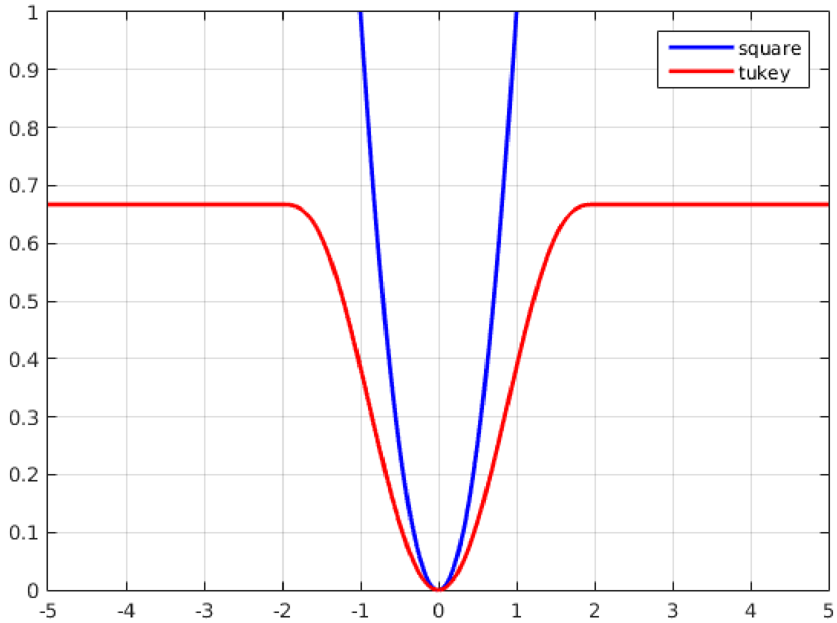
- Squared (reprojection) errors are sensitive to outliers

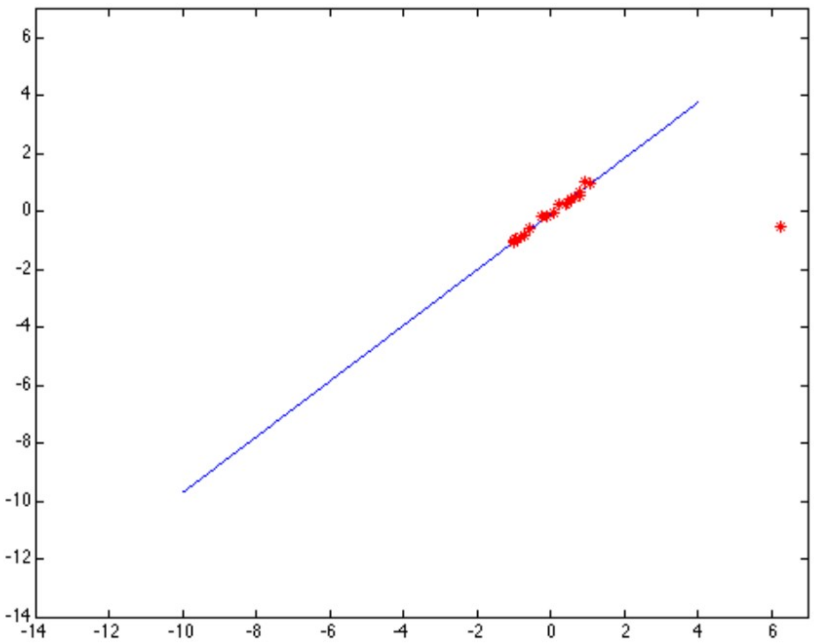




Objective Function: M-Estimators

- Robustify cost function
  - Quadratic behaviour near optimum
  - Flatened or threshold for far from optimum





VISION FOR ROBOTICS LAB

Objective Function: M-Estimators

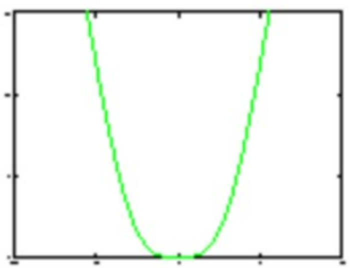
Conditions on the estimator functions (ρ)

- Bounded influence function (derivative of cost function ρ)
- Unique minimum (at zero) and convex
- At places where second derivative is undefined, the first derivative needs to be non-zero.

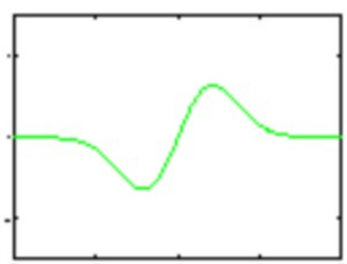
M-Estimator Functions: Popular Choices

Welsch

$$\frac{c^2}{2}[1-\exp(-(x/c)^2)]$$



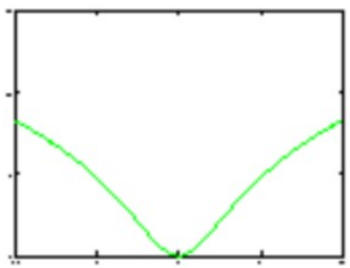
ρ-function



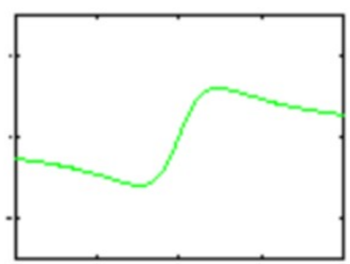
ψ (influence) function

Cauchy

$$\frac{c^2}{2}\log(1+(x/c)^2)$$



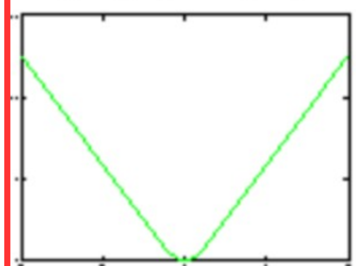
ρ-function



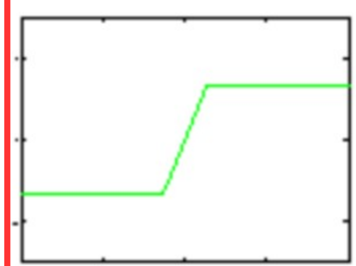
ψ (influence) function

Huber

$$\begin{cases} x^2/2 \\ k(|x|-k/2) \end{cases}$$



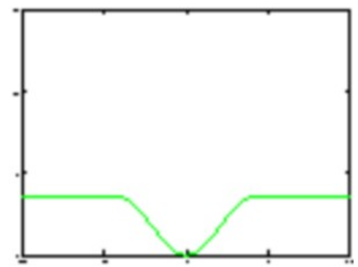
ρ-function



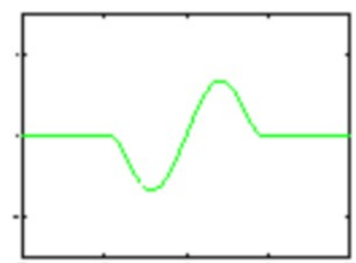
ψ (influence) function

Tukey

$$\begin{cases} \frac{c^2}{6}\left(1-[1-(x/c)^2]^3\right) \\ (c^2/6) \end{cases}$$



ρ-function



ψ (influence) function

VISION FOR ROBOTICS LAB

Solving the problem: Robust Cost Function

$$\min_p \sum_i \rho\left(r_i(p)\right), \quad r_i := z_i - h(p_i)$$

$$\sum_i \frac{\partial}{\partial p_i} \rho(r_i(p_i)) = 0$$

$$\boxed{\frac{\partial \rho}{\partial r} := \psi(r) = w(r) \cdot r}$$

$$\Leftrightarrow \sum_i \boxed{\frac{\partial \rho}{\partial r_i}(r_i)} \frac{\partial r_i(p_i)}{\partial p_i} = 0$$

$$\Rightarrow \sum_i w(r_i) r_i \frac{\partial r_i}{\partial p} = 0$$

re-weighted Least Squares problem!

Solving the problem: Algorithm

- Due to non-linearity, use iterative solver
  - Gradient descent (first order method)
    - Certain cost decrease, but slow near optimum
  - Gauss-Newton (second order)
    - Fast convergence near optimum, but tends to „overshoot“ when far from optimum
  - Levenberg-Marquardt

- Usually LM is the method of choice for BA

Solving the problem: Levenberg-Marquardt

- Slightly adapted Gauss-Newton method:

$$\boxed{\left(J^T J + \boxed{\lambda I}\right)} \delta p = -J^T \epsilon$$

$$\lambda \gg 1 \Rightarrow \quad \text{Gradient Descend Method}$$

$$\lambda < 1 \Rightarrow \quad \text{Gauss-Newton Method}$$

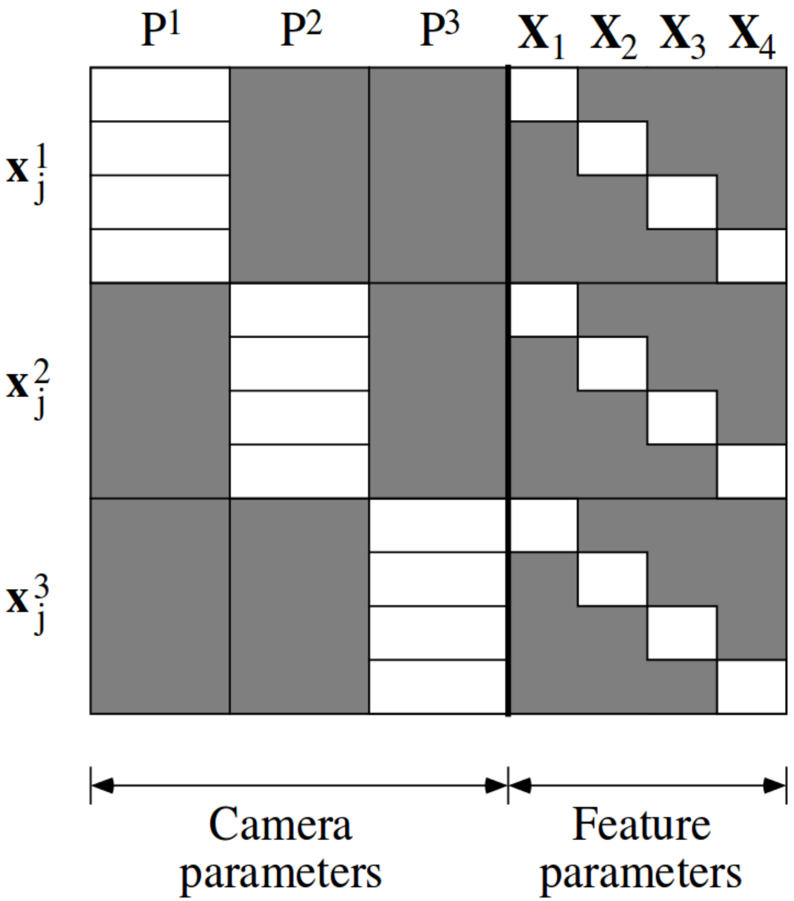
- If step reduces cost → decrease λ
- Otherwise → increase λ
- Usually λ is changed using a constant factor (e.g. 10, resp. 0.1)

Solving the problem: Structure

- Landmark in image only depend on parameters of this image → Sparsity
- In realistic problems not all Lanmarks are observed in all images

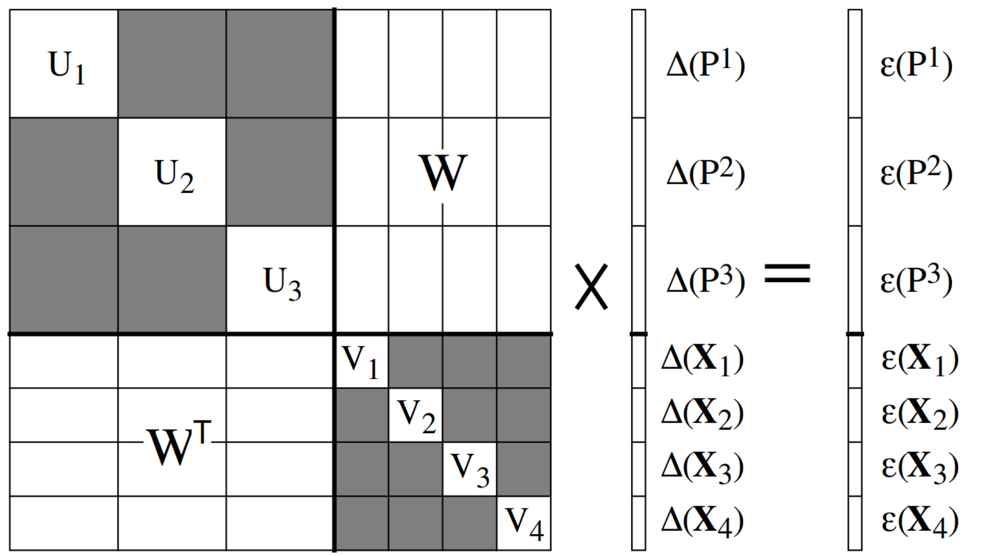
  → matrix gets sparser

And this helps me how?



Solving the problem: Structure

- Have a look at the resulting normal equation



$$\begin{bmatrix} U & W \\ W^T & V \end{bmatrix} \begin{bmatrix} \delta(T) \\ \delta(l) \end{bmatrix} = \begin{bmatrix} \epsilon(T) \\ \epsilon(l) \end{bmatrix}$$

Solving the problem: Shur Complement

$$\begin{bmatrix} U & W \\ W^T & V \end{bmatrix} \begin{bmatrix} \delta(T) \\ \delta(l) \end{bmatrix} = \begin{bmatrix} \epsilon(T) \\ \epsilon(l) \end{bmatrix}$$

Left multiply with: $\begin{bmatrix} I & -WV^{-1} \\ 0 & I \end{bmatrix}$

$$\Rightarrow \begin{bmatrix} U - WV^{-1}W^T & 0 \\ W^T & V \end{bmatrix} \begin{bmatrix} \delta(T) \\ \delta(l) \end{bmatrix} = \begin{bmatrix} \epsilon(T) - WV^{-1}\epsilon(l) \\ \epsilon(l) \end{bmatrix}$$

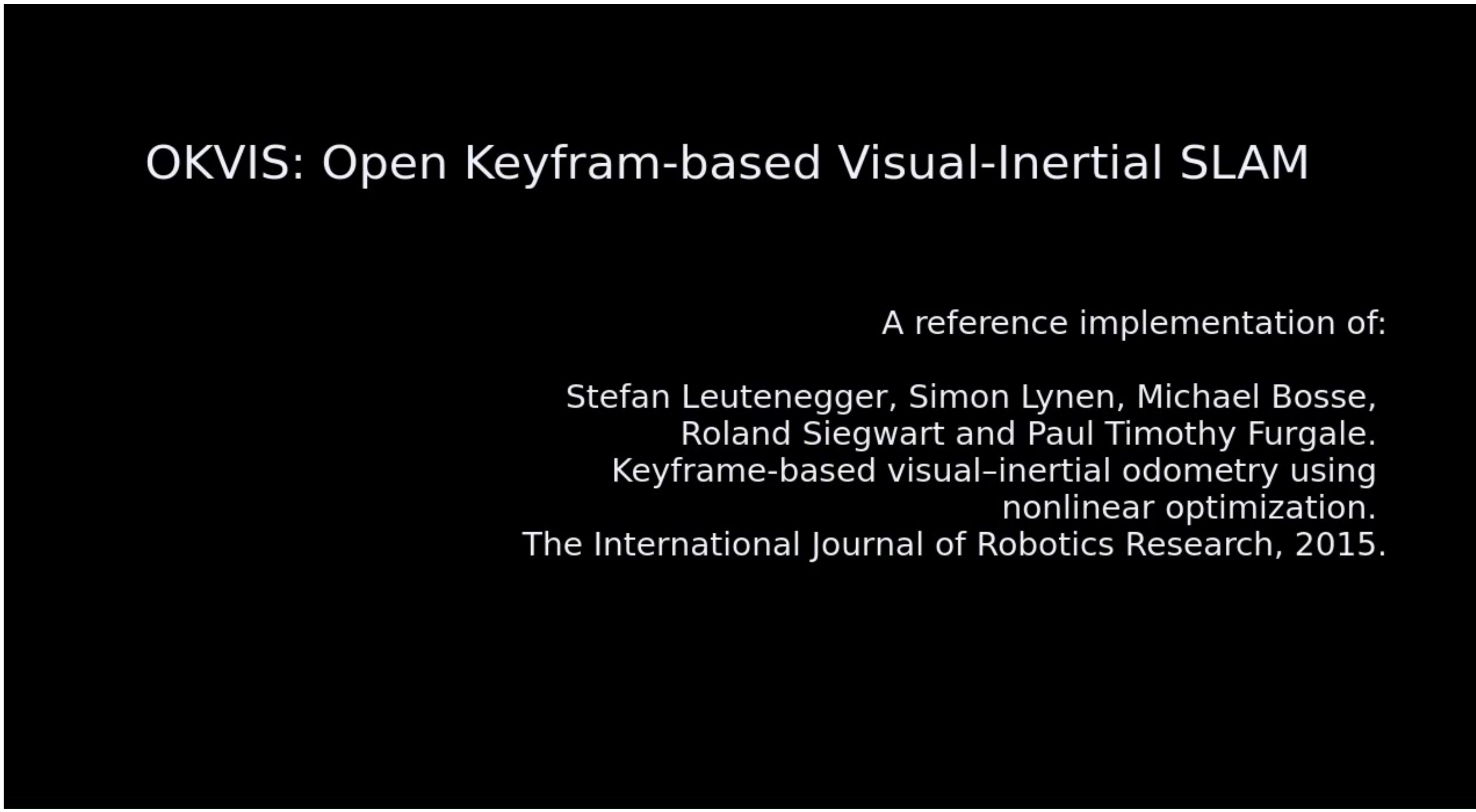
Solve the problem in two steps

$$\left(U - WV^{-1}W^T\right)\delta(T) = \epsilon(T) - WV^{-1}\epsilon(l)$$

$$\delta(l) = V^{-1}\left(\epsilon(l) - W^T\delta(T)\right)$$

Questions so far?

Open Keyframe-based Visual-Inertial SLAM: OKVIS



OVKIS: Introduction

- State vector composition

$$\boldsymbol{x} = \left[\boldsymbol{x}_R^T \ \boldsymbol{x}_L^T \ \boldsymbol{x}_C^T\right]^T$$

OKVIS: IMU-Kinematics

- Model bias terms as random walks

$$
\begin{aligned}
{}_W\dot{\mathbf{r}}_S &= \mathbf{C}_{WS}\ {}_S\mathbf{v} \\
\dot{\mathbf{q}}_{WS} &= \frac{1}{2}\boldsymbol{\Omega}({}_S\boldsymbol{\omega})\mathbf{q}_{WS} \\
{}_S\dot{\mathbf{v}} &= {}_S\widetilde{\mathbf{a}} + \mathbf{w}_{\mathrm{a}} - \mathbf{b}_{\mathrm{a}} + \mathbf{C}_{SW}\ {}_W\mathbf{g} - ({}_S\boldsymbol{\omega}) \times {}_S\mathbf{v} \\
\dot{\mathbf{b}}_{\mathrm{g}} &= \mathbf{w}_{b_{\mathrm{g}}} \\
\dot{\mathbf{b}}_{\mathrm{a}} &= -\frac{1}{\tau}\mathbf{b}_{\mathrm{a}} + \mathbf{w}_{b_{\mathrm{a}}}
\end{aligned}
$$

- Assume uncorrelated, gaussian white noise

$$
\mathbf{w} := [\mathbf{w}_{\mathrm{g}}^{\mathrm{T}}, \mathbf{w}_{\mathrm{a}}^{\mathrm{T}}, \mathbf{w}_{b_{\mathrm{g}}}^{\mathrm{T}}, \mathbf{w}_{b_{\mathrm{a}}}^{\mathrm{T}}]^{\mathrm{T}}
$$

OKVIS: Cost Function

- IMU error terms and reprojection error

$$J(\mathbf{x}) := \underbrace{\sum_{i=1}^{I} \sum_{k=1}^{K} \sum_{j \in \mathcal{J}(i,k)} \mathbf{e}_{\mathrm{r}}^{i,j,k\,\mathrm{T}} \mathbf{W}_{\mathrm{r}}^{i,j,k} \mathbf{e}_{\mathrm{r}}^{i,j,k}}_{\text{visual}} + \underbrace{\sum_{k=1}^{K-1} \mathbf{e}_{\mathrm{s}}^{k\,\mathrm{T}} \mathbf{W}_{\mathrm{s}}^{k} \mathbf{e}_{\mathrm{s}}^{k}}_{\text{inertial}}$$

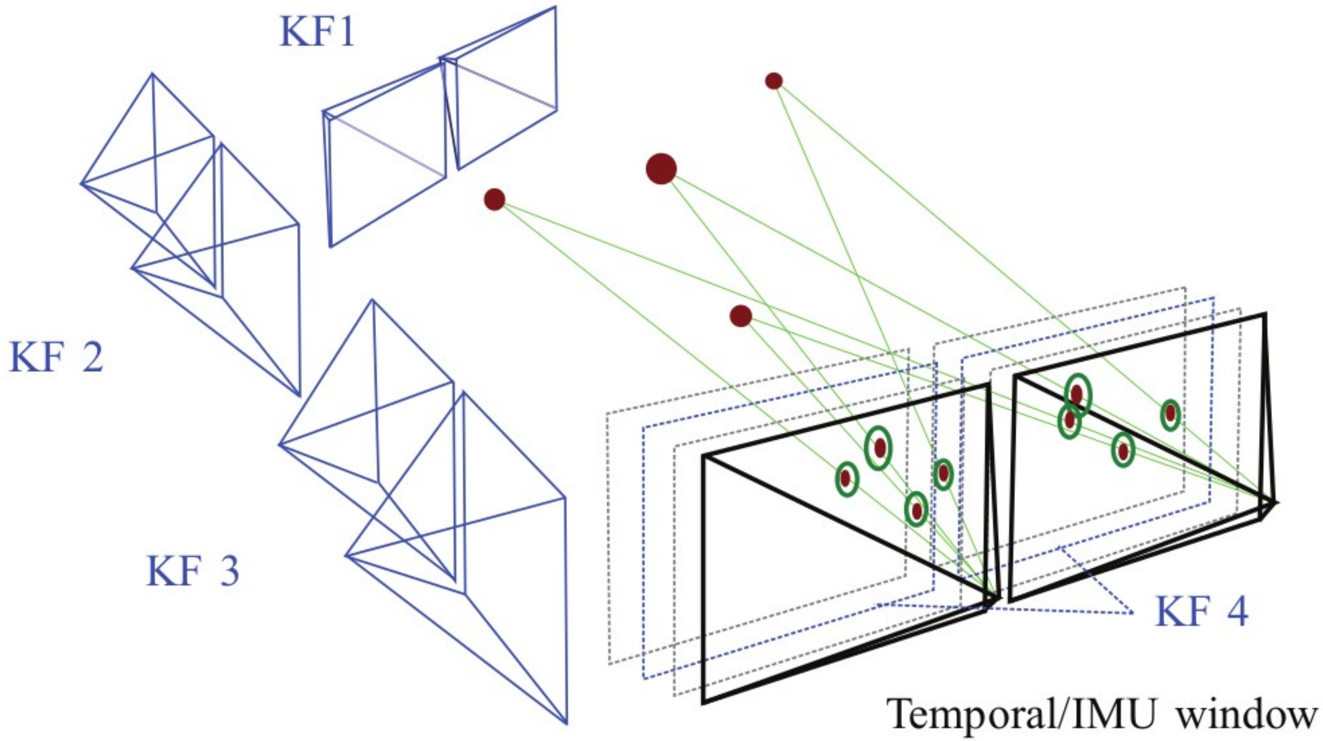
$$\mathbf{e}_{\mathrm{r}}^{i,j,k} = \mathbf{z}^{i,j,k} - \mathbf{h}_i\left(\boldsymbol{T}_{C_iS}^{k}\, \boldsymbol{T}_{SW}^{k}\, {}_W\boldsymbol{l}^{j}\right)$$

$$\mathbf{e}_{\mathrm{s}}^{k}\left(\mathbf{x}_{\mathrm{R}}^{k}, \mathbf{x}_{\mathrm{R}}^{k+1}, \mathbf{z}_{\mathrm{s}}^{k}\right) = \begin{bmatrix} {}_W\hat{\mathbf{r}}_S^{k+1} - {}_W\mathbf{r}_S^{k+1} \\ 2\left[\hat{\mathbf{q}}_{WS}^{k+1} \otimes \mathbf{q}_{WS}^{k+1\;-1}\right]_{1:3} \\ \hat{\mathbf{x}}_{\mathrm{sb}}^{k+1} - \mathbf{x}_{\mathrm{sb}}^{k+1} \end{bmatrix} \in \mathbb{R}^{15}$$

Predictions obtained by forward integration

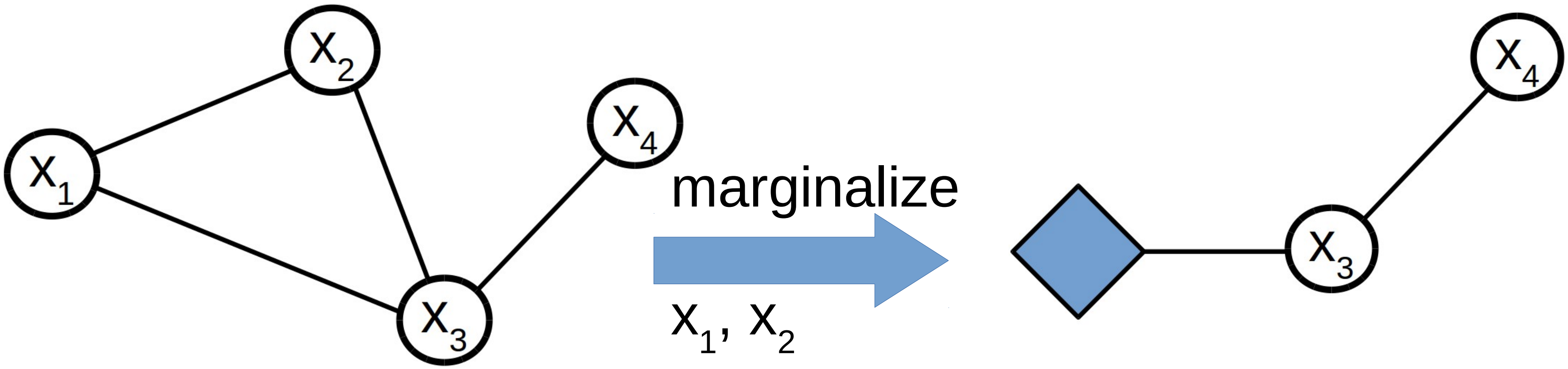
Bundle Adjustment in OKVIS

- Do the non-linear optimization on a sliding window
  - Keep computational complexity constant

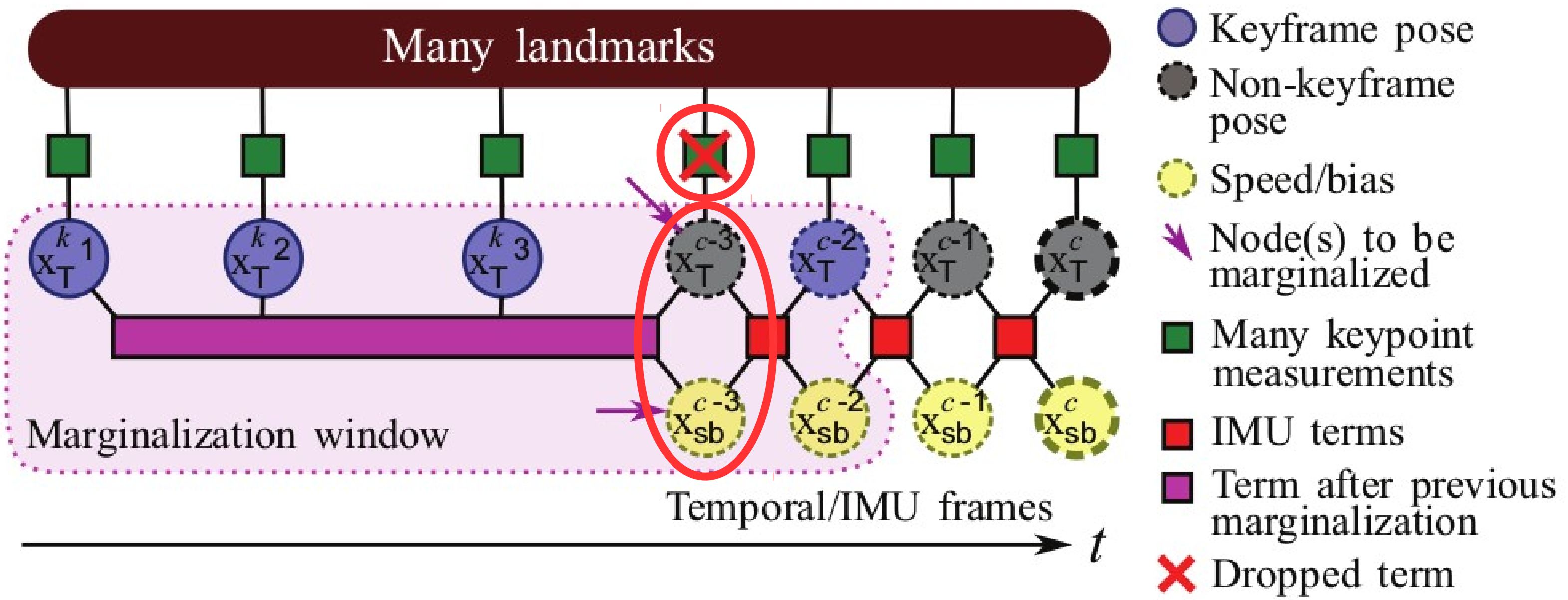


Bundle Adjustment in OKVIS: Marginalization

- Intention: take old frames and IMU measurement out of the state, without loosing all information
- Marginalization can be thought of as introducing a state that summarizes the previous states.

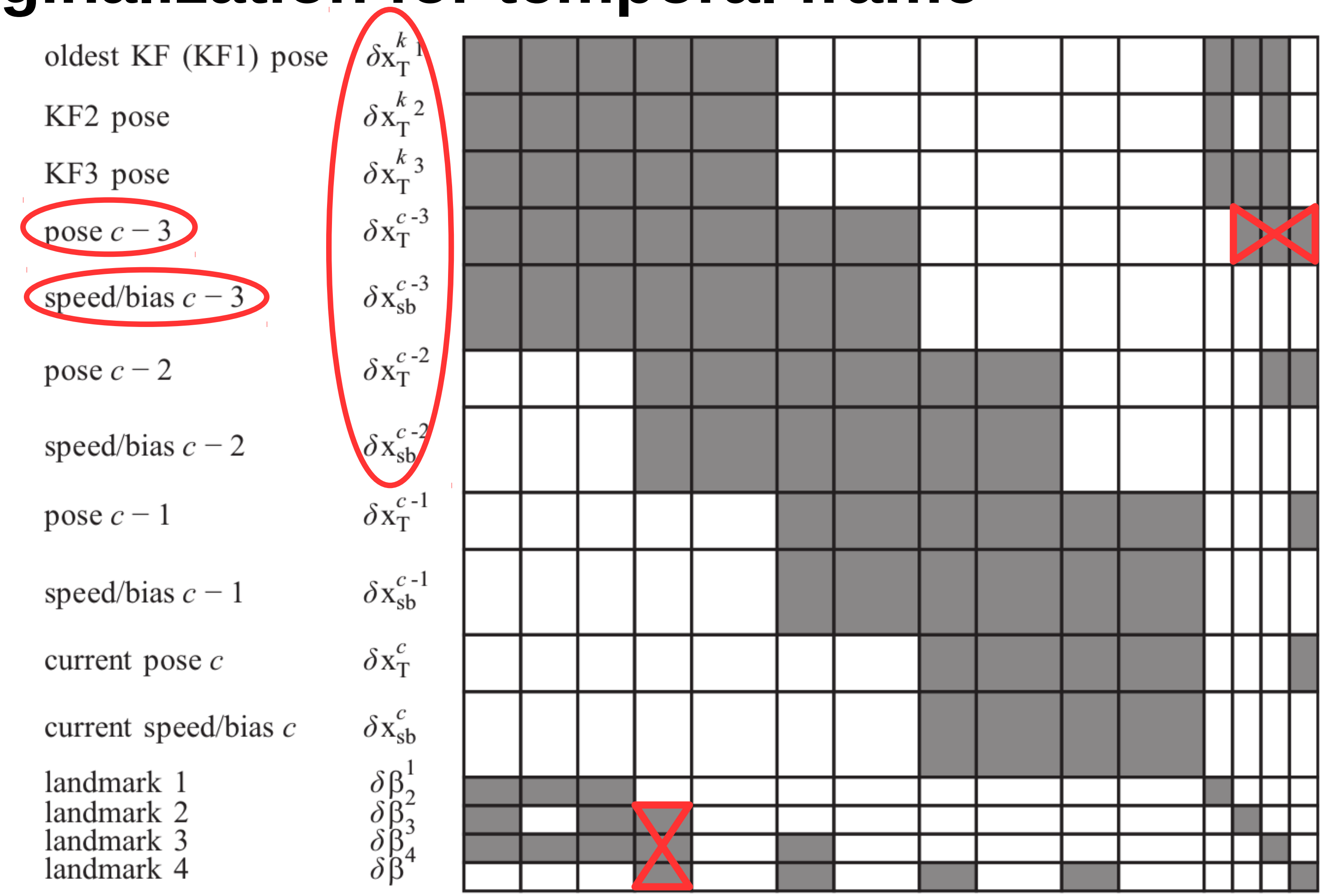


Bundle Adjustment in OKVIS: Temporal Frame Marginalization

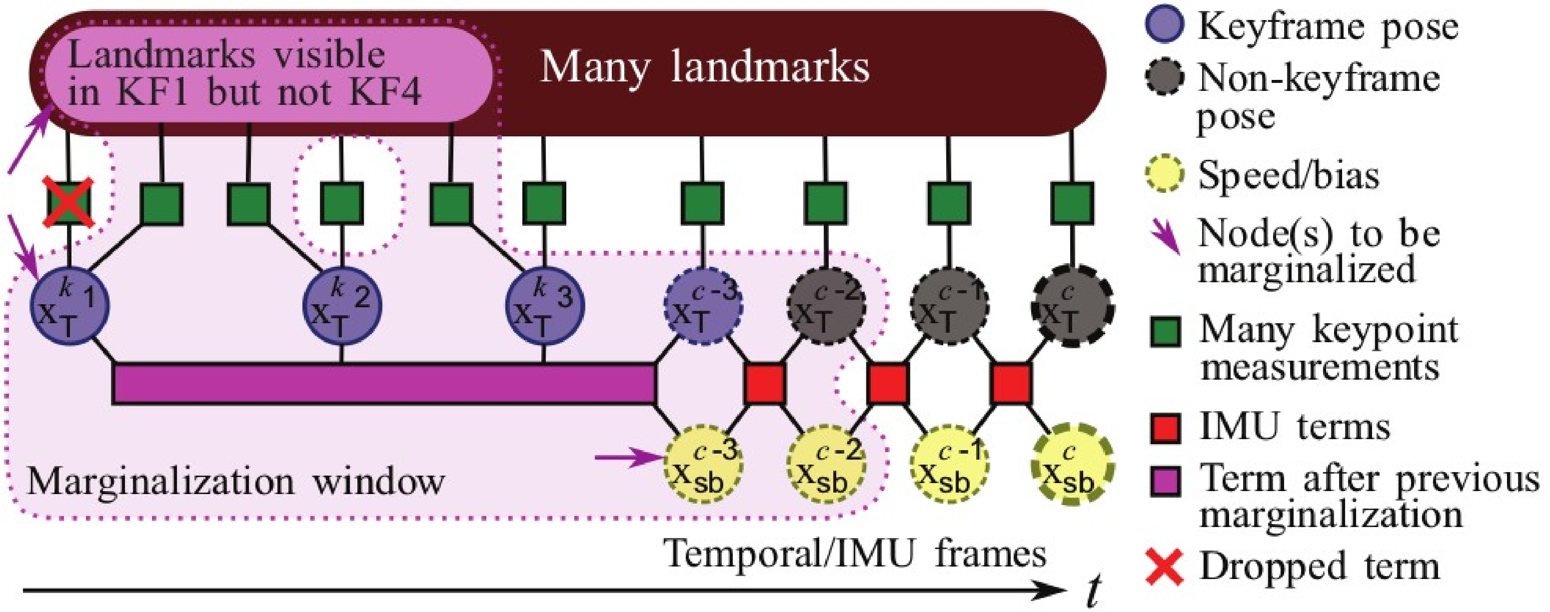


- Drop landmark measurements for this frame
- Marginalize camera pose and bias terms

Bundle Adjustment in OKVIS: Structure of the Marginalization for temporal frame



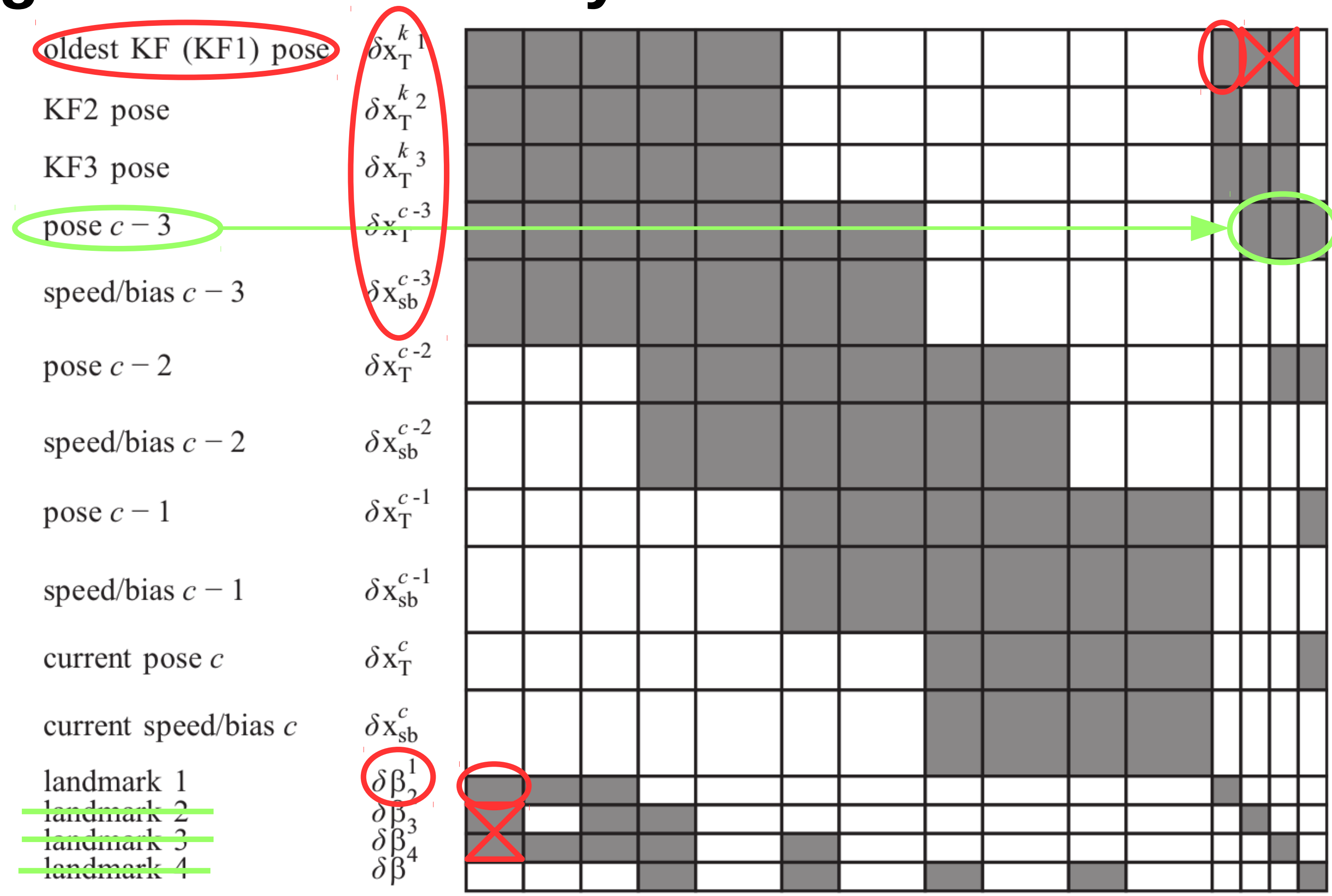
Bundle Adjustment in OKVIS: Keyframe Marginalization



- Drop landmark measurements which are not marginalized
- Marginalize out only landmarks not visible in current frames to avoid losing information from past KF observations

ETH zürich

Bundle Adjustment in OKVIS: Structure of the Marginalization for Keyframe



VISION FOR ROBOTICS LAB

Questions so far?