# Large Scale Photometric Bundle Adjustment
# IDP Report

Alexander Gaul

October 2021

**Abstract**

Jointly estimating scene structure and camera parameters is a fundamental problem in computer vision, which has recently been shown to benefit from direct, photometric methods. We address the work Large Scale Photometric Bundle Adjustment by Woodford and Rosten, which is dealing with the offline optimization of camera parameters and dense geometry. Overall, the presented framework is capable of handling a large number of parameters and a variety of lighting conditions and camera intrinsics, as could be found in a collection of images downloaded from the internet. It deals with these challenges by utilizing a photometric loss that is invariant to local lighting changes and a memory-efficient implementation of the Variable Projection optimizer. The goal of this project has been to manually implement the proposed framework and reproduce the evaluation results on the Tanks and Temples dataset. We additionally compare the memory-efficient optimization method with standard Levenberg-Marquardt.

## 1 Introduction

This project deals with the article Large Scale Photometric Bundle Adjustment [14] which presents a framework for the reconstruction of camera parameters and dense scene geometry. The joint estimation of camera parameters and scene structure is a fundamental problem in computer vision with applications ranging from large-scale reconstruction of objects or buildings to online camera pose estimation. The latter has been shown to greatly benefit from minimizing a photometric error [10], rather than the geometric error of feature-based methods. Such direct methods measure the error in the domain of pixel intensities, which can explain the significant improvements. The standard approach for a large-scale dense reconstruction, however, consists of computing camera

Figure 1: Refined dense point cloud and camera poses

parameters and sparse geometry with feature-based structure from motion (SfM) [8] and a subsequent dense reconstruction with multi-view stereo (MVS) [9]. Such a method is used to generate initial parameters for our framework, which performs a photometric refinement of geometry and camera parameters to improve accuracy. It specifically deals with the challenges found in sets of images downloaded from the internet, which include different lighting conditions and camera intrinsics and a high volume of data.

Some methods have addressed jointly minimizing a photometric error over structure and camera parameters. One offline method is [4], which represents dense geometry as a triangle mesh that is regularized with a smoothness term to penalize variations of the surface normals. Optimization is performed using a first-order gradient descent solver. A sparse and direct approach to monocular visual odometry is presented in [10]. The photometric error is minimized over a window of recent frames using a second-order solver. It performs joint optimization of all model parameters, including camera motion and intrinsics and geometry, which is represented as inverse depth in a reference frame. The smoothness prior used in other methods is omitted and the algorithm does not depend on keypoint detectors or descriptors.

Similarly, our approach uses an independent, ray-based landmark representation, additionally modeling planes, and utilizes a second-order solver, but computes dense geometry. We employ the Normalized Cross Correlation (NCC) photometric score [3, 7], which is invariant to affine intensity variations over local patches, to deal with different lighting situations between images. A memory-efficient implementation of the Variable Projection optimizer [6] is capable of jointly optimizing thousands of camera parameters and millions of geometry parameters. Overall, the presented framework is capable of handling a variety of lighting conditions and camera intrinsics and a large number of parameters, as could be found in a large and diverse collection of images downloaded from the internet.

The goal is to implement the proposed framework and evaluate its performance on the Tanks and Temples training dataset [11]. Additionally, we compare the proposed optimization method to standard Levenberg-Marquardt with higher memory requirements. We do not address the reconstruction from internet images in this work.

# 2 Method

In this section, we describe the main optimization step in our framework. For this, we require problem parameters, a photometric cost function, and the specific optimization methods we apply.

## 2.1 Problem Parametrization

The pose of each view is modeled by a Euclidean transformation $\mathbf{T}_i \in \mathrm{SE}(3)$ consisting of a rotation $\mathbf{R} \in \mathrm{SO}(3)$ and translation $\mathbf{t} \in \mathbb{R}^3$. For $P$ images we thus obtain camera extrinsics $\{\mathbf{R}_i, \mathbf{t}_i\}_{i=1}^P$. Contrary to [14] we assume shared intrinsics between views, which consist of calibration and distortion parameters $\{f_x, f_y, c_x, c_y, l_1, l_2\}$. We use the standard linear calibration function

$$\kappa(\mathbf{x}) = \begin{bmatrix} f_x & 0 \\ 0 & f_y \end{bmatrix} \mathbf{x} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}. \tag{1}$$

The lens distortion $\varphi$ describes a polynomial radial distortion model with two coefficients

$$\varphi(\mathbf{x}) = \mathbf{x}(1 + l_1 r^2 + l_2 r^4) \qquad r^2 = \|\mathbf{x}\|^2. \tag{2}$$

With $\bar{\mathbf{\Theta}} = \{f_x, f_y, c_x, c_y, l_1, l_2\} \cup \{\mathbf{R}_i, \mathbf{t}_i\}_{i=1}^P$ we denote the set of camera parameters. Projecting a 3D-point $\mathbf{X} \in \mathbb{R}^3$ from world coordinates into image $j$ we obtain

$$\mathbf{x} = \kappa(\varphi(\pi(\mathbf{R}_j \mathbf{X} + \mathbf{t}_j))) \tag{3}$$

with the projection function $\pi([x, y, z]^\top) = [x/z, y/z]^\top$. We additionally require the inverse functions $\kappa^{-1}$ and $\varphi^{-1}$. The first can be determined straightforwardly, $\varphi^{-1}$ is another polynomial where we compute the first six coefficients in closed form according to [5].

Intuitively, in our formulation, a landmark $k$ is not only defined by its location $\mathbf{X}_k$ in 3D space but also by a normal $\mathbf{N}_k$ modeling the surface orientation. We parametrize a landmark $k$ as a plane $\mathbf{n}_k \in \mathbb{R}^3$ relative to some source frame $i = I_k$ and obtain

$$\mathbf{n}_k = \frac{\mathbf{R}_i \mathbf{N}_k}{(\mathbf{R}_i \mathbf{N}_k)^\top (\mathbf{R}_i \mathbf{X}_k + \mathbf{t}_i)} \tag{4}$$

by transforming location and normal into the respective coordinate system. This formulation encodes depth as well as orientation. Overall, the complete set of optimization variables is $\mathbf{\Theta} = \bar{\mathbf{\Theta}} \cup \{\mathbf{n}_k\}_{k=1}^{L}$.

To unproject the coordinates $\mathbf{x}_k$ of landmark $k$ in image $i$ onto its plane in world coordintes we compute

$$\mathbf{X}_k = \mathbf{R}_i^\top \left( \frac{\bar{\mathbf{x}}_k}{\mathbf{n}_k^\top \bar{\mathbf{x}}_k} - \mathbf{t}_i \right), \qquad \bar{\mathbf{x}}_k = \begin{bmatrix} \varphi^{-1}(\kappa^{-1}(\mathbf{x}_k)) \\ 1 \end{bmatrix}. \tag{5}$$

Overall, to reproject landmark $k$ from view $i$ to view $j$ we calculate

$$\mathbf{x}_{ijk} = \kappa(\varphi(\pi(\mathbf{R}_j \mathbf{R}_i^\top \left( \frac{\bar{\mathbf{x}}_k}{\mathbf{n}_k^\top \bar{\mathbf{x}}_k} - \mathbf{t}_i \right) + \mathbf{t}_j))). \tag{6}$$

Extending this mapping to a grid of $N$ points results in the function $\Pi_{ijk}^{\mathbf{\Theta}} : \mathbb{R}^{2\times N} \to \mathbb{R}^{2\times N}$ depending on the model parameters $\mathbf{\Theta}$ with $k$ being the landmark and $i$ and $j$ the source and target frame, respectively.

During the optimization we compute parameter update steps $\delta\mathbf{\Theta}$ and generally denote the update with $\mathbf{\Theta} \leftarrow \mathbf{\Theta} \oplus \delta\mathbf{\Theta}$. Specifically, the update of pose a $\mathbf{T}_i$, consisting of rotation $\mathbf{R}_i$ and translation $\mathbf{t}_i$, is not done additively but parametrized as $\mathbf{T}_i \leftarrow \exp(\boldsymbol{\xi}_i)\mathbf{T}_i$ where $\exp(\cdot)$ converts a 6-vector into a Euclidean transformation. Additionally, we abbreviate the derivative of the update $\partial/\partial\boldsymbol{\xi}_i|_{\boldsymbol{\xi}_i=0}$ with $\partial/\partial\mathbf{R}_i$ and $\partial/\partial\mathbf{t}_i$. Details can be found in appendix B.2.

## 2.2 Cost Formulation

Having a mapping $\Pi_{ijk}^{\mathbf{\Theta}}$ for pixels from one image into another, we can compare intensity values between different views for each landmark. For this, we additionally require visibilities $\mathcal{V}_k$ and a source frame $I_k$ for each landmark $k$, which stay fixed during the optimization. As discussed in the previous section, the landmark is parametrized relative to this source view, meaning it is anchored to a specific pixel coordinate $\mathbf{x}_k$ in the respective image. During the optimization, only its depth and normal orientation change relative to this view. We construct a $4 \times 4$ grid $\mathbf{P}_k \in \mathbb{R}^{2\times N}, N = 16$ centered on $\mathbf{x}_k$, with the spacing between points being one pixel, and extract intensity values $\mathsf{I}_i(\mathbf{P}_k)$ from image $i = I_k$. When reprojecting this grid into different views, multiple such patches can

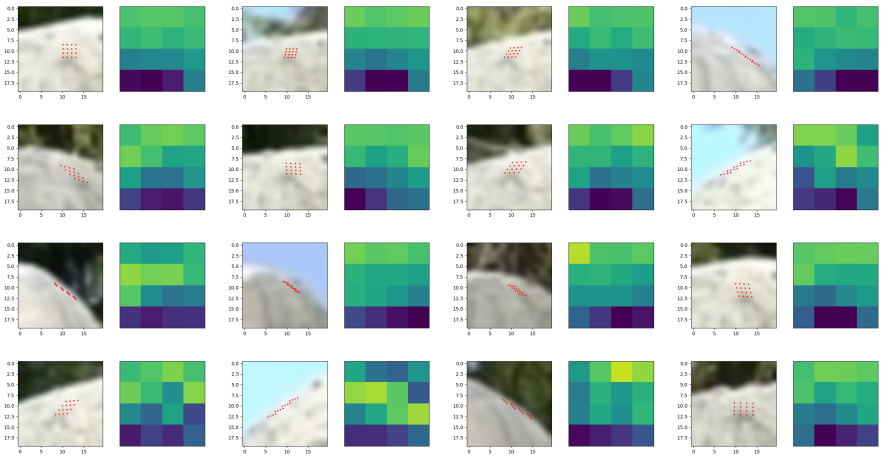

Figure 2: Grid coordinates and normalized patches from different views (after optimization)

be sampled and compared, which is visualized in figure 2. As desired, the grid is projected onto the plane and follows its orientation and the sampled patches have similar values despite different lighting conditions. We define the least-squares cost accordingly

$$E(\Theta) = \|\boldsymbol{\mathcal{E}}_{\text{reg}}\|^2 + \sum_k \sum_{j \in \mathcal{V}_k} \rho(\|\boldsymbol{\mathcal{E}}_{jk}\|^2), \qquad \rho(s) = \frac{s}{s+\tau^2} \tag{7}$$

$$\boldsymbol{\mathcal{E}}_{jk} = \Psi(\mathsf{I}_j(\Pi^{\Theta}_{ijk}(\mathbf{P}_k))) - \Psi(\mathsf{I}_i(\mathbf{P}_k)), \qquad i = I_k \tag{8}$$

$$\Psi(\bar{\mathsf{I}}) = \frac{\bar{\mathsf{I}} - \mu_{\bar{\mathsf{I}}}}{\sigma_{\bar{\mathsf{I}}}} \qquad \sigma_{\bar{\mathsf{I}}} = \|\bar{\mathsf{I}} - \mu_{\bar{\mathsf{I}}}\| \qquad \mu_{\bar{\mathsf{I}}} = \frac{\mathbf{1}^\top \bar{\mathsf{I}}}{N} \tag{9}$$

$$\boldsymbol{\mathcal{E}}_{\text{reg}} = 10^5 \left[ \frac{f_x - f_y}{f_x + f_y}, \frac{c_x - W/2}{\max(W,H)}, \frac{c_y - H/2}{\max(W,H)} \right]^\top. \tag{10}$$

The Normalized Cross Correlation $\Psi$ normalizes the patches to have zero mean and unit variance making them invariant to affine intensity transformations. The Geman-McClure Kernel [2, 1] $\rho$ robustifies the cost with $\tau = 0.5$ as $\lim_{s\to\infty} \rho(s) = 1$. $\boldsymbol{\mathcal{E}}_{\text{reg}}$ forces the focal lengths to be similar and the pixel offsets to be close to half the image width and height, respectively. This might be less relevant for our formulation with shared intrinsics, but in the original method the individual parameters for each view might be less well constrained.

## 2.3 Optimization

The optimization procedure involves computing Jacobians, the partial derivatives of the residuals with respect to the optimization variables $\mathbf{J} = \frac{\partial \boldsymbol{\mathcal{E}}}{\partial \bar{\boldsymbol{\Theta}}}$. We group Jacobians by landmark and separate the derivatives for camera and landmark paramters. To incorporate the loss functions we peform a weighting by the square-root of $\rho'(s) = \tau^2/(s+\tau^2)^2$. For landmark $k$ we obtain the Jacobians

$$\bar{\mathbf{J}}_k = \left[ \sqrt{\rho'(\|\boldsymbol{\mathcal{E}}_{jk}\|^2)} \frac{\partial \boldsymbol{\mathcal{E}}_{jk}}{\partial \bar{\boldsymbol{\Theta}}} \right]_{\forall j \in \mathcal{V}_k} \qquad \hat{\mathbf{J}}_k = \left[ \sqrt{\rho'(\|\boldsymbol{\mathcal{E}}_{jk}\|^2)} \frac{\partial \boldsymbol{\mathcal{E}}_{jk}}{\partial \mathbf{n}_k} \right]_{\forall j \in \mathcal{V}_k} \tag{11}$$

and the residual

$$\boldsymbol{\mathcal{E}}_k = \left[ \sqrt{\rho'(\|\boldsymbol{\mathcal{E}}_{jk}\|^2)} \boldsymbol{\mathcal{E}}_{jk} \right]_{\forall j \in \mathcal{V}_k}. \tag{12}$$

Using the Gauss-Newton optimization for least squares problems we compute approximate Hessians

$$\mathbf{H}_{\bar{\boldsymbol{\Theta}}} = \sum_{k=0}^{L} \bar{\mathbf{J}}_k^\top \bar{\mathbf{J}}_k + \mathbf{J}_{\text{reg}}^\top \mathbf{J}_{\text{reg}}, \qquad \mathbf{H}_{\mathbf{n}_k} = \hat{\mathbf{J}}_k^\top \hat{\mathbf{J}}_k, \qquad \mathbf{H}_{\bar{\boldsymbol{\Theta}}\mathbf{n}_k} = \bar{\mathbf{J}}_k^\top \hat{\mathbf{J}}_k, \tag{13}$$

with $\mathbf{J}_{\text{reg}}$ being the Jacobian for the intrinsics regularization, and the gradients

$$\mathbf{g}_{\bar{\boldsymbol{\Theta}}} = \sum_{k=0}^{L} \bar{\mathbf{J}}_k^\top \boldsymbol{\mathcal{E}}_k \qquad \mathbf{g}_{\mathbf{n}_k} = \hat{\mathbf{J}}_k^\top \boldsymbol{\mathcal{E}}_k. \tag{14}$$

We combine $\mathbf{H}_{\bar{\boldsymbol{\Theta}}\mathbf{n}_k}$ into $\mathbf{H}_{\bar{\boldsymbol{\Theta}}\mathbf{n}}$ and $\mathbf{H}_{\mathbf{n}_k}$ into $\mathbf{H}_{\mathbf{n}}$ for all $k$, where $\mathbf{H}_{\mathbf{n}}$ is block-diagonal because of independent landmarks. Overall, we obtain a linear system $\mathbf{H}\delta\boldsymbol{\Theta} = -\mathbf{g}$ that is structured as follows

$$\begin{bmatrix} \mathbf{H}_{\bar{\boldsymbol{\Theta}}} & \mathbf{H}_{\bar{\boldsymbol{\Theta}}\mathbf{n}} \\ \mathbf{H}_{\bar{\boldsymbol{\Theta}}\mathbf{n}}^\top & \mathbf{H}_{\mathbf{n}} \end{bmatrix} \begin{bmatrix} \delta\bar{\boldsymbol{\Theta}} \\ \delta\mathbf{n} \end{bmatrix} = - \begin{bmatrix} \mathbf{g}_{\bar{\boldsymbol{\Theta}}} \\ \mathbf{g}_{\mathbf{n}} \end{bmatrix}. \tag{15}$$

Solving this directly is prohibitively expensive due to the large number of landmark parameters, but we can exploit the block-sparse structure of $\mathbf{H}_n$. Evaluating the Schur complement, we solve for $\delta\bar{\boldsymbol{\Theta}}$ and construct a reduced camera system (RCS) to first determine the update step for the camera parameters

$$\delta\bar{\boldsymbol{\Theta}} = -\mathbf{H}_{\text{rcs}}^{-1}\mathbf{g}_{\text{rcs}} \qquad \mathbf{H}_{\text{rcs}} = \mathbf{H}_{\bar{\boldsymbol{\Theta}}} - \mathbf{H}_{\bar{\boldsymbol{\Theta}}\mathbf{n}}\mathbf{H}_{\mathbf{n}}^{-1}\mathbf{H}_{\bar{\boldsymbol{\Theta}}\mathbf{n}}^\top \qquad \mathbf{g}_{\text{rcs}} = \mathbf{g}_{\bar{\boldsymbol{\Theta}}} - \mathbf{H}_{\bar{\boldsymbol{\Theta}}\mathbf{n}}\mathbf{H}_{\mathbf{n}}^{-1}\mathbf{g}_{\mathbf{n}}. \tag{16}$$

We note that this system can be constructed by summing over individual landmarks like the original camera Hessian and gradient. Back-substituting the result into equation (15) we obtain

$$\delta \mathbf{n} = -\mathbf{H}_{\mathbf{n}}^{-1}(\mathbf{g}_{\mathbf{n}} - \mathbf{H}_{\bar{\mathbf{\Theta}}\mathbf{n}}^{\top}\delta\bar{\mathbf{\Theta}}) \tag{17}$$

which can also be computed for each landmark individually.

To perform the Levenberg-Marquardt (LM) optimization step, we additionally apply damping to the Hessian and choose its diagonal $\operatorname{diag}(\mathbf{H})$ as the default option but also use the identity matrix in our experiments. This results in

$$\delta\bar{\mathbf{\Theta}} = -(\mathbf{H}_{\text{rcs}} + \lambda \operatorname{diag}(\mathbf{H}_{\bar{\mathbf{\Theta}}}))^{-1}\mathbf{g}_{\text{rcs}} \tag{18}$$

$$\delta\mathbf{n}_k = -(\mathbf{H}_{\mathbf{n}_k} + \lambda \operatorname{diag}(\mathbf{H}_{\mathbf{n}_k}))^{-1}(\mathbf{g}_{\mathbf{n}_k} - \mathbf{H}_{\bar{\mathbf{\Theta}}\mathbf{n}_k}^{\top}\delta\bar{\mathbf{\Theta}}) \tag{19}$$

where the RCS itself was constructed by applying damping to $\mathbf{H}_{\mathbf{n}}$.

The inconvenience that remains in this formulation is the off-diagonal block $\mathbf{H}_{\bar{\mathbf{\Theta}}\mathbf{n}_k}$ that needs to be stored for each landmark, with the size depending on the number of views from which it is visible. Ignoring the correction term removes this necessity and yields the update step

$$\delta\mathbf{n}_k = -\mathbf{H}_{\mathbf{n}_k}^{-1}\mathbf{g}_{\mathbf{n}_k} \tag{20}$$

that can also be expressed as $\delta\mathbf{n}_k = -\hat{\mathbf{J}}_k^{+}\boldsymbol{\mathcal{E}}_k$, with $\mathbf{J}^{+} = (\mathbf{J}^{\top}\mathbf{J})^{-1}\mathbf{J}^{\top}$ being the matrix pseudo-inverse. This describes the situation of assuming fixed camera parameters and thus zero step $\delta\bar{\mathbf{\Theta}}$. In the Variable Projection optimizer (VarPro) this update is run repeatedly until convergence. This means that for each outer iteration, that optimizes the camera parameters, multiple inner iterations are performed which are called Embedded Point Iterations (EPIs). By default, VarPro uses the identity matrix to damp the RCS but no damping for the landmark updates.

Algorithm 1 summarizes the optimization procedure. $\lambda_0$ depends on the damping type used, we choose $10^{-6}$ for $\operatorname{diag}(\mathbf{H})$ and $L$ for the identity matrix $\mathbf{I}$. The specific landmark update depends on the chosen optimizer, the outer iteration is repeated with higher damping when the performed updated increases the cost.

**Algorithm 1:** Optimization

$\lambda \leftarrow \lambda_0$ ; $\omega \leftarrow 10$ ;
$S_0 \leftarrow E(\mathbf{\Theta}_0)$;
**for** $t = 1 : 10$ **do**
  $\mathbf{\Theta}_t \leftarrow \mathbf{\Theta}_{t-1}$ ;
  **for** $k = 1 : L$ **do**
    **for** $j \in \mathcal{V}_k$ **do**
      Linearize $\boldsymbol{\mathcal{E}}_{jk}$ ;
    Add Landmark $k$ to $\mathbf{H}_{\text{rcs}}$ and $\mathbf{g}_{\text{rcs}}$ ;
  $\bar{\mathbf{\Theta}}_t \leftarrow \bar{\mathbf{\Theta}}_t \oplus \delta\bar{\mathbf{\Theta}}$ ; /* Update camera parameters */
  **for** $k = 1 : L$ **do**
    $\mathbf{n}_{kt} \leftarrow \mathbf{n}_{kt} + \delta\mathbf{n}_k$ ; /* Update landmark parameters */
  $S_t \leftarrow E(\mathbf{\Theta}_{\mathbf{t}})$ ;
  **if** $S_t < S_{t-1}$ **then**
    $\lambda \leftarrow \lambda/10$ ; $\omega \leftarrow 10$ ; /* Reduce damping */
  **else**
    $\lambda \leftarrow \lambda\omega$ ; $\omega \leftarrow 2\omega$ ; /* Increase damping */
    **goto** 9 ; /* Retry */

# 3 Preprocessing

This framework is designed to jointly optimize camera parameters and scene structure from an existing reconstruction. We use Colmap [8, 9] with default settings (`colmap automatic_reconstructor`)

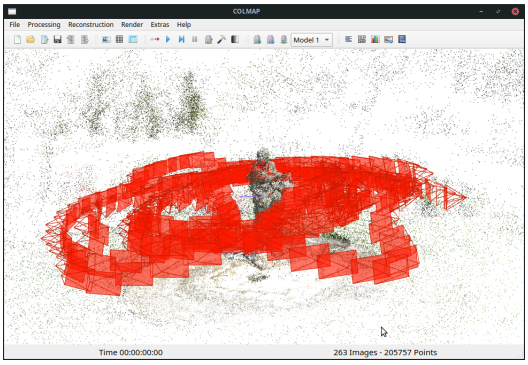


Figure 3: Sparse reconstruction with camera poses in Colmap

which produces camera parameters and a dense point cloud including normals. The reconstructions typically include a significant amount of background clutter and are thus manually cropped to the object of interest.

We require the visibilities $\mathcal{V}_k$ and source view $I_k$ per landmark. Points can be projected into each view to check if they are located on the respective image. Additionally, we need to consider occlusions. For this, we utilize the mesh generated by the poisson mesher included in Colmap and render depth maps for each view. If the rendered depth deviates from the depth of a landmark by less than 1% it is regarded as visible.

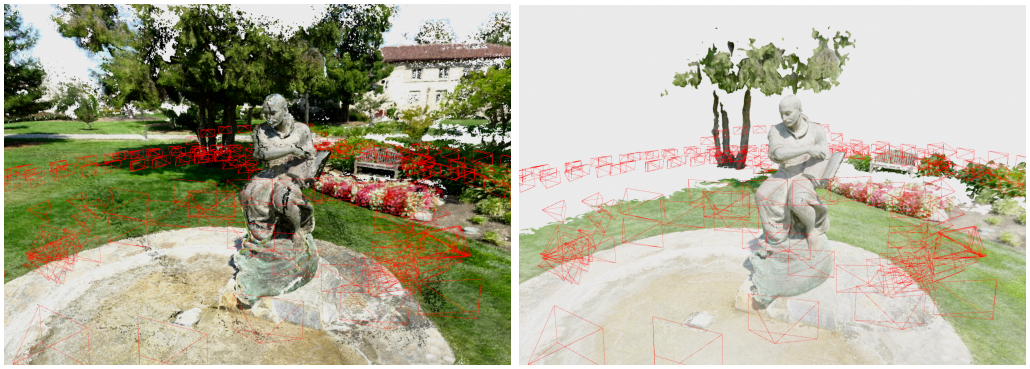

Figure 4: Colamp dense reconstruction and poisson mesh

The source view is chosen as the frame whose patch is closest to the robust mean of all patches across visible views, in order to avoid selecting a photometric outlier. We construct and equally-spaced $4 \times 4$ grid $\mathbf{X}_j \in \mathbb{R}^{3\times N}$ in 3D space centered at the position of the landmark and facing the direction of the normal. The grid is scaled in such a way that the projected points have an average distance of 1 pixel across all views. The optimization

$$I_k = \arg\min_{j\in\mathcal{V}_k} \|\bar{\mathsf{I}}_j - \boldsymbol{\mu}\|^2 \tag{21}$$

$$\boldsymbol{\mu} = \arg\min_{\hat{\boldsymbol{\mu}}\in\mathbb{R}^N} \sum_{j\in\mathcal{V}_k} \rho(\|\bar{\mathsf{I}}_j - \hat{\boldsymbol{\mu}}\|^2) \tag{22}$$

$$\bar{\mathsf{I}}_j = \Psi(\mathsf{I}_j(\kappa(\varphi(\pi(\mathbf{R}_j\mathbf{X}_k + \mathbf{t}_j))))) \tag{23}$$

is started from the unrobustified mean. Discarded are landmarks with insufficient texture where the patch of the determined source view has variance $\|\bar{\mathsf{I}}_{I_k} - \boldsymbol{\mu}_{\bar{\mathsf{I}}_{I_k}}\| < 8$, with 256 gray levels.

# 4 Implementation Details

The preprocessing has been implemented in Python to make use of easy visualization and implementation. This includes a graphical interface to inspect and crop the supplied point cloud. Unfortunately, the optimization in Python is only sufficient to reliably handle smaller datasets or initialization with Colmap on low settings. As such, the computation of the robust mean is additionally implemented in Ceres [15] using auto-differentiation. The main optimization is also implemented in C++ with the Jacobians being computed analytically. Outputs are stored as simple text files.

Linearization, Schur complement, and landmark updates are parallelized using OMP. Landmarks are grouped by source view for the construction of the RCS. The Jacobians as well as the Schur complement are computed with respect to relative poses $\mathbf{T}_j\mathbf{T}_i^{-1}$ similar to [12]. Derivates for absolute poses are determined subsequently from relative derivatives with details in appendix B.2.

The optimization is run on half size images first and then at full size with landmark grids scaled accordingly. Images are sampled with bicubic interpolation. In VarPro, structure is initially optimized alone wiht EPIs, before starting algorithm 1. We perform a maximum number of 5 EPIs per landmark and stop optimizing when the relative change of the cost is smaller $10^{-3}$.

The main optmization was also implemented in Ceres with automatic and analytic gradients. Unfortunately Ceres is limited in the number of residuals a problem can consist of, which means that the given datasets need to be subsampled to far below a hundred thousand points.

# 5 Evaluation

We evaluate the presented optimization methods on the Tanks and Temples [11] training datasets. Table 1 shows the size of our reconstructions. The ground truth was collected with a laser scanner and is publicly available for the training sets. For the evaluation, reconstructed and ground truth point clouds are aligned. After computing the distance of each reconstructed point to the ground truth, a global precision score can be obtained given a specific distance threshold. Similarly, a recall score can be determined after computing the distance of the ground truth points to the reconstruction. Default distance thresholds $\tau$ are provided for each dataset.

| | Truck | Ignatius | Meetingroom | Barn | Caterpillar | Church |
|---|---|---|---|---|---|---|
| Cameras | 251 | 263 | 371 | 410 | 383 | 507 |
| Landmarks [M] | 1.51 | 1.69 | 5.35 | 4.40 | 4.77 | 2.92 |
| Observations [M] | 122.42 | 148.56 | 325.95 | 358.94 | 462.09 | 373.75 |

Table 1: Dataset properties after preprocessing

## 5.1 Quantitative Evaluation

Table 2 and 3 show AUC scores between 0 and $2\tau$ for precision and recall capturing more information than scores at a single $\tau$. We evaluate the Colmap output and the point cloud after preprocessing separately to distinguish the effects from the main optimization step. Specifically, discarding landmarks with insufficient texture significantly decreases recall scores, whereas the optimization yields slight improvements. The preprocessing also improves precision scores slightly.

Overall, Levenberg-Marquardt delivers a small improvement over the Varialbe Projection optimizer. The mean precision and recall scores are additionally visualized in figure 5.

| | Truck | Ignatius | Meetingroom | Barn | Caterpillar | Church | Mean |
|---|---|---|---|---|---|---|---|
| LM | **0.612** | **0.695** | **0.451** | **0.408** | **0.372** | 0.556 | **0.516** |
| VarPro | 0.579 | 0.691 | 0.435 | 0.395 | 0.357 | **0.560** | 0.503 |
| Preprocessing | 0.555 | 0.613 | 0.426 | 0.385 | 0.348 | 0.555 | 0.480 |
| Colmap | 0.564 | 0.608 | 0.384 | 0.379 | 0.357 | 0.530 | 0.470 |

Table 2: Precision-AUC

| | Truck | Ignatius | Meetingroom | Barn | Caterpillar | Church | Mean |
|---|---|---|---|---|---|---|---|
| LM | 0.424 | 0.627 | 0.204 | 0.409 | 0.433 | 0.218 | 0.386 |
| VarPro | 0.407 | 0.632 | 0.204 | 0.409 | 0.424 | 0.222 | 0.383 |
| Preprocessing | 0.399 | 0.618 | 0.201 | 0.385 | 0.442 | 0.211 | 0.376 |
| Colmap | **0.477** | **0.640** | **0.291** | **0.481** | **0.544** | **0.255** | **0.448** |

Table 3: Recall-AUC

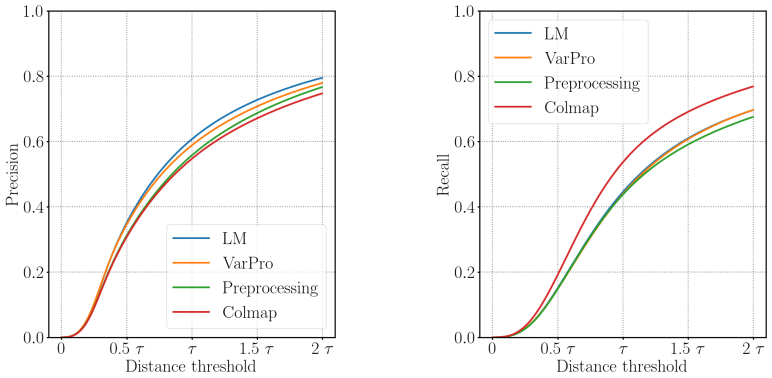


Figure 5: Mean precision and recall

## 5.2 Resource Requirements

Table 4 shows the memory consumption for the different datasets with the largest requiring over 100GB with LM. VarPro typcially requires about half the memory.

| | Truck | Ignatius | Meetingroom | Barn | Caterpillar | Church |
|---|---|---|---|---|---|---|
| LM | 33.23 | 38.41 | 79.74 | 87.39 | 109.00 | 101.54 |
| VarPro | 16.99 | 19.26 | 36.41 | 39.88 | 47.49 | 51.73 |

Table 4: Memory [GB]

The runtime is displayed in table 5. VarPro takes roughly double the time as LM with the longest optimization taking over 2 days. The increased runtime is in part caused by a higher number of backtracking steps, which might be alleviated by further fine-tuning of the damping parameters.

| | Truck | Ignatius | Meetingroom | Barn | Caterpillar | Church |
|---|---|---|---|---|---|---|
| LM | 4.53 | 7.07 | 15.47 | 15.47 | 23.47 | 36.18 |
| VarPro | 11.82 | 23.58 | 24.80 | 32.00 | 48.13 | 58.77 |

Table 5: Runtime [h]

An overview over the entire pipeline is depicted in table 6. A significant amount of time is required by the Colmap reconstruction while still being shorter than the main optimization step. At this point, calculating visibilites and 3D grids is only implemented in Python and not

multithreaded which also adds to the overall duration. Computing the robust mean in C++ has much lower runtime.

| | Truck | Ignatius | Meetingroom | Barn | Caterpillar | Church |
|---|---|---|---|---|---|---|
| Colmap (SfM+MVS) | 3.25 | 4.00 | 6.38 | 7.00 | 6.67 | 5.68 |
| Prep I (Python) | 1.92 | 1.13 | 4.05 | 6.57 | 3.97 | 3.08 |
| Prep II (C++) | 0.30 | 0.30 | 1.02 | 1.60 | 1.87 | 0.93 |

Table 6: Pipeline Runtime [h]

### 5.3 Qualitative Evaluation

Figure 6 visualizes reconstructed models with indivudal points being colored according to the distance to the ground truth. Darker colors encode larger distances with white being 0 and black being $3\tau$. Improvements can easily be observed for the Truck and Ignatius datasets. Typically the improvements affect distinct parts of the surface which comes along with a decrease in smoothness that is expected due to the lack of surface regularization. VarPro reconstructions for the objects located outdoors show a large number of significantly diverged landmarks.

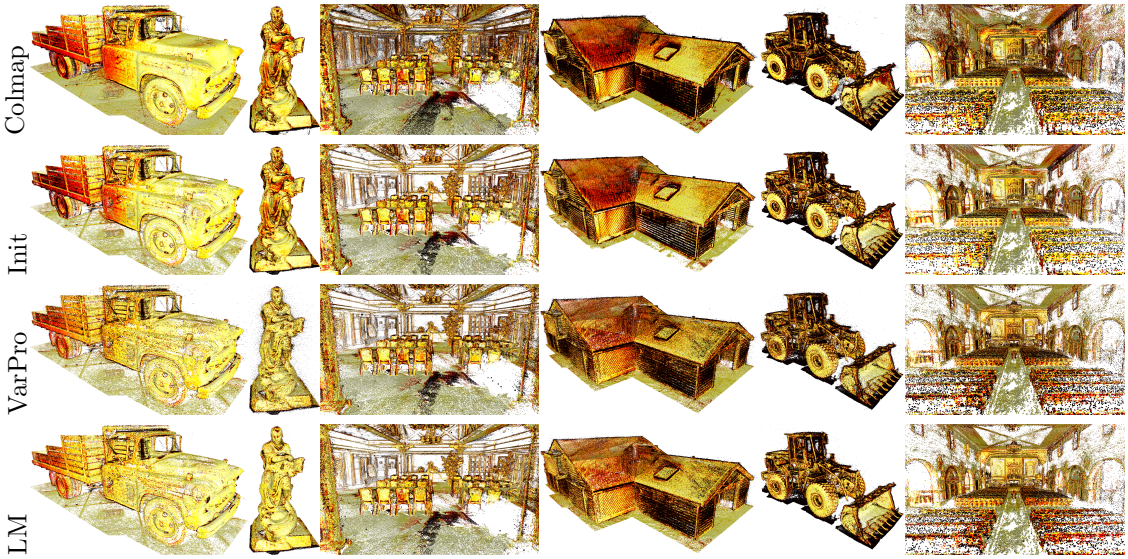


Figure 6: Precision

Figure 7 shows ground truth models where the color visualized the distance to the respective reconstructed point cloud. Visible are the gaps caused by the selection in the preprocessing. Improvements for Truck and Ignatius are located in similar areas as for the precision.

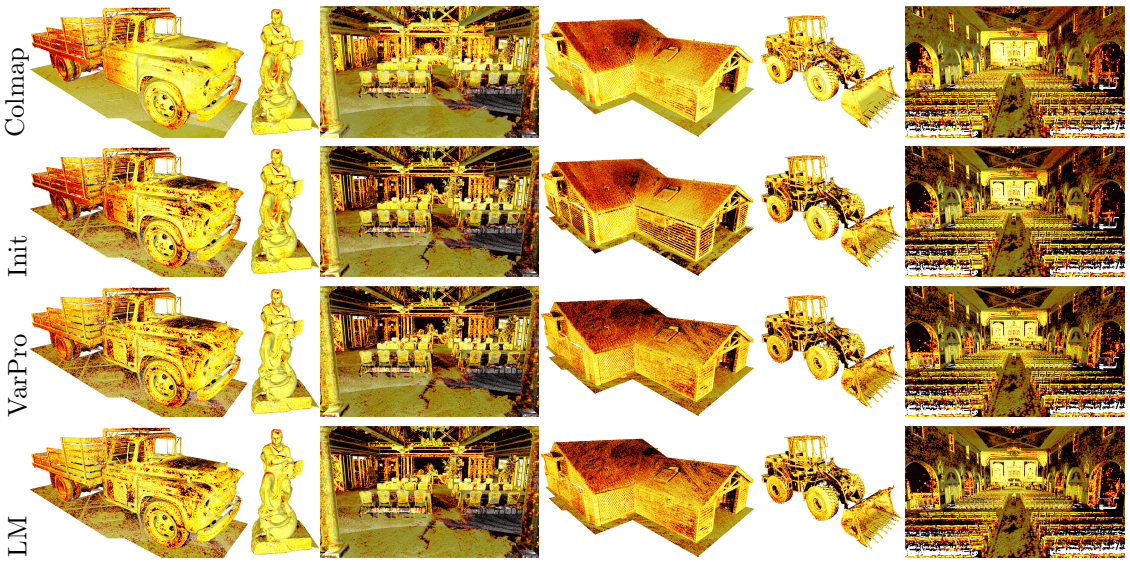


Figure 7: Recall

## 5.4 Ablation Studies

We evaluate four datasets with different damping types for LM and VarPro. The matrices used are either the identity matrix $\mathbf{I}$ or the diagonal of the hessian $\text{diag}(\mathbf{H})$ where we either apply the damping to all parameters $\boldsymbol{\Theta}$ or just to the camera parameters $\bar{\boldsymbol{\Theta}}$ without landmarks.

Tables 7 and 8 and figure 8 show the results for LM with $\lambda\,\text{diag}(\mathbf{H}_{\boldsymbol{\Theta}})$ being the default setting. Damping landmarks and camera parameters shows better performance than damping just cameras. The default setting has the best mean score for precision whereas the identity is slightly better for recall.

| | Truck | Ignatius | Meetingroom | Barn | Mean |
|---|---|---|---|---|---|
| $\lambda\,\text{diag}(\mathbf{H}_{\boldsymbol{\Theta}})$ | **0.612** | 0.695 | **0.451** | 0.408 | **0.541** |
| $\lambda\,\text{diag}(\mathbf{H}_{\bar{\boldsymbol{\Theta}}})$ | 0.593 | 0.696 | 0.445 | 0.408 | 0.535 |
| $\lambda\mathbf{I}_{\lvert\boldsymbol{\Theta}\rvert}$ | 0.583 | **0.707** | 0.445 | **0.409** | 0.536 |
| $\lambda\mathbf{I}_{\lvert\bar{\boldsymbol{\Theta}}\rvert}$ | 0.557 | 0.691 | 0.447 | 0.396 | 0.523 |
| Preprocessing | 0.555 | 0.613 | 0.426 | 0.385 | 0.495 |

Table 7: Precision-AUC for LM ablation experiments

| | Truck | Ignatius | Meetingroom | Barn | Mean |
|---|---|---|---|---|---|
| $\lambda\,\text{diag}(\mathbf{H}_{\boldsymbol{\Theta}})$ | **0.424** | 0.627 | 0.204 | 0.409 | 0.416 |
| $\lambda\,\text{diag}(\mathbf{H}_{\bar{\boldsymbol{\Theta}}})$ | 0.412 | 0.629 | 0.204 | 0.408 | 0.413 |
| $\lambda\mathbf{I}_{\lvert\boldsymbol{\Theta}\rvert}$ | 0.407 | **0.643** | 0.205 | **0.414** | **0.418** |
| $\lambda\mathbf{I}_{\lvert\bar{\boldsymbol{\Theta}}\rvert}$ | 0.392 | 0.623 | **0.206** | 0.397 | 0.404 |
| Preprocessing | 0.399 | 0.618 | 0.201 | 0.385 | 0.401 |

Table 8: Recall-AUC for LM ablation experiments

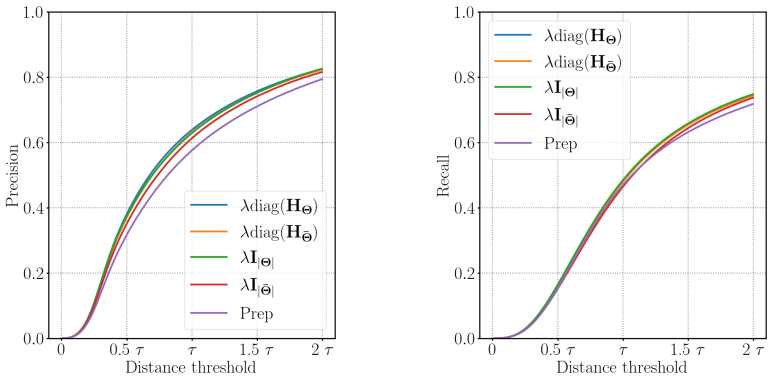


Figure 8: Mean precision and recall for LM ablation experiments

Tables 9 and 10 and figure 9 show the results for VarPro with $\lambda \mathbf{I}_{|\bar{\Theta}|}$ being the default setting presented in [14]. Damping all parameters with the diagonal of the hessian shows slightly better precision scores. The recall scores are best for the full damping with the identity matrix. Overall, it is not clear here that the default option is the best choice.

| | Truck | Ignatius | Meetingroom | Barn | Mean |
|---|---|---|---|---|---|
| $\lambda \mathbf{I}_{\|\bar{\Theta}\|}$ | 0.579 | **0.691** | 0.435 | 0.395 | 0.525 |
| $\lambda \,\text{diag}(\mathbf{H}_{\bar{\Theta}})$ | 0.572 | 0.691 | 0.423 | 0.391 | 0.519 |
| $\lambda \mathbf{I}_{\|\Theta\|}$ | 0.577 | 0.671 | 0.433 | 0.389 | 0.518 |
| $\lambda \,\text{diag}(\mathbf{H}_{\Theta})$ | **0.601** | 0.676 | **0.437** | **0.396** | **0.527** |
| Preprocessing | 0.555 | 0.613 | 0.426 | 0.385 | 0.495 |

Table 9: Precision-AUC for VarPro ablation experiments

| | Truck | Ignatius | Meetingroom | Barn | Mean |
|---|---|---|---|---|---|
| $\lambda \mathbf{I}_{\|\bar{\Theta}\|}$ | 0.407 | 0.632 | 0.204 | 0.409 | 0.413 |
| $\lambda \,\text{diag}(\mathbf{H}_{\bar{\Theta}})$ | 0.402 | 0.632 | 0.199 | 0.406 | 0.410 |
| $\lambda \mathbf{I}_{\|\Theta\|}$ | 0.410 | **0.643** | **0.206** | **0.414** | **0.418** |
| $\lambda \,\text{diag}(\mathbf{H}_{\Theta})$ | **0.420** | 0.625 | 0.203 | 0.408 | 0.414 |
| Preprocessing | 0.399 | 0.618 | 0.201 | 0.385 | 0.401 |

Table 10: Recall-AUC for VarPro ablation experiments

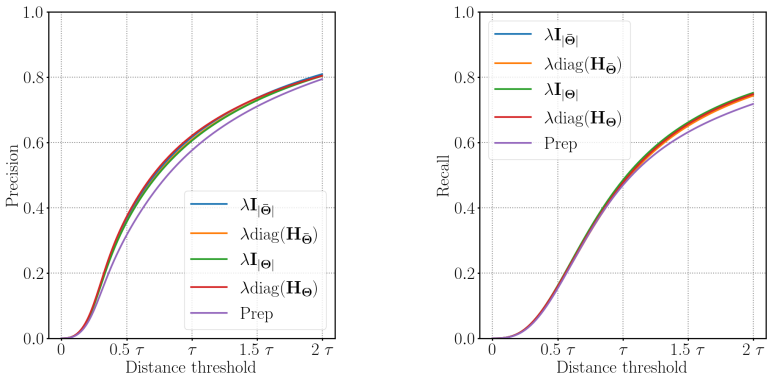


Figure 9: Mean precision and recall for VarPro ablation experiments

# 6 Conclusion and Future Work

We have compared the Variable Projection optimizer with standard Levenberg-Marquardt for which we have obtained slightly better quantitative and qualitative results. Resource requirements at twice the speed and double the memory can be seen as comparable.

Further modifications can be made to the Levenberg-Marquardt method. The off-diagonal Hessian block could also be discarded after the construction of the RCS and then recomputed for the update of the landmark parameters. This reduces the memory consumption and increases the runtime. It would be interesting to see how this trade-off compares to the resource requirements of VarPro. The implementation could benefit from further improvements. As the images are sampled from a video, landmarks are typically observed from consecutive views which could lend itself to a memory-efficient encoding. Results could possibly be stored in binary files instead of simple text files. Cache-efficiency might be improved when storing landmark Hessians.

The paper also evaluates running the MVS of Colmap with the optimized camera parameters to obtain a dense point cloud with better recall. This could also be tested for the LM reconstruction to specifically compare the accuracy of the camera parameters.

Modifications could also be made to the preprocessing stage. The low recall after this step could be improved by selecting a lower threshold for discarding landmarks, which could in turn negatively affect precision. Additionally, views with extreme angles towards a landmark might be removed from its visibilities and the computation of the robust mean. This could make the optimization more robust by filtering out extremely slanted patches. Alternatively, the corresponding residuals could be weighted during the main optimization.

# References

[1] Stuart Geman and Donald E. McClure. “Bayesian image analysis: An application to single photon emission tomography”. In: *Amer. Statist. Assoc* (1985), pp. 12–18.

[2] Michael J. Black and Anand Rangarajan. “On the unification of line processes, outlier rejection, and robust statistics with applications in early vision”. In: *International Journal of Computer Vision* (1996).

[3] Yasutaka Furukawa and Jean Ponce. “Accurate, dense, and robust multiview stereopsis”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2010).

[4] Amaël Delaunoy and Marc Pollefeys. “Photometric bundle adjustment for dense multiview 3d modeling”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2014), pp. 1486–1493.

[5] Pierre Drap and Julien Lefèvre. “An exact formula for calculating inverse radial lens distortions”. In: *Sensors* (2016).

[6] J. Hong et al. “Projective Bundle Adjustment from Arbitrary Initialization Using the Variable Projection Method”. In: *Lecture Notes in Computer Science* 9905 (2016), pp. 477–493.

[7] J. Frahm Johannes L. Schönberger E. Zheng and Marc Pollefeys. “Pixelwise view selection for unstructured multi-view stereo”. In: *Proceedings of the European Conference on Computer Vision* (2016), pp. 501–518.

[8] Johannes Lutz Schönberger and Jan-Michael Frahm. “Structure-from-Motion Revisited”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[9] Johannes Lutz Schönberger et al. “Pixelwise View Selection for Unstructured Multi-View Stereo”. In: *European Conference on Computer Vision (ECCV)*. 2016.

[10] Vladlen Koltun Jakob Engel and Daniel Cremers. “Least Squares Normalized Cross Correlation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).

[11] Arno Knapitsch et al. “Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction”. In: *ACM Transactions on Graphics* 36.4 (2017).

[12] Pablo Rodríguez Palafox. *Local Tracking and Mapping for Direct Visual SLAM*. `https://vision.in.tum.de/_media/members/demmeln/palafox2019ma.pdf`. 2019.

[13] Dinesh Atchuthan Joan Sol‘a Jeremie Deray. *A micro Lie theory for state estimation in robotics*. 2020.

[14] Oliver J. Woodford and Edward Rosten. *Large Scale Photometric Bundle Adjustment*. 2020.

[15] Sameer Agarwal, Keir Mierle, et al. *Ceres Solver*. `http://ceres-solver.org`.

# A NCC

For $N$ sampled intensity values $\bar{\mathsf{I}}$ and the Normalized Cross-Correlation

$$\Psi(\bar{\mathsf{I}}) = \frac{\bar{\mathsf{I}} - \mu}{\sigma} \qquad \sigma = \|\bar{\mathsf{I}} - \mu\| \qquad \mu = \frac{\mathbf{1}^\top \bar{\mathsf{I}}}{N} \tag{24}$$

we obtain the jacobian

$$\begin{aligned}
\frac{\partial \Psi(\bar{\mathsf{I}})}{\partial \bar{\mathsf{I}}} &= \frac{\partial}{\partial \bar{\mathsf{I}}} \frac{\bar{\mathsf{I}} - \frac{\mathbf{1}^\top \bar{\mathsf{I}}}{N}}{\sqrt{\left(\bar{\mathsf{I}} - \frac{\mathbf{1}^\top \bar{\mathsf{I}}}{N}\right)^\top \left(\bar{\mathsf{I}} - \frac{\mathbf{1}^\top \bar{\mathsf{I}}}{N}\right)}} \\
&= \frac{1}{\sigma}\left(\mathbf{I}_N - \mathbf{1}\mathbf{1}^\top \frac{1}{N}\right) + \left(\bar{\mathsf{I}} - \frac{\mathbf{1}^\top \bar{\mathsf{I}}}{N}\right) \frac{1}{-\sigma^2} \frac{1}{2\sigma} 2 \left(\bar{\mathsf{I}} - \frac{\mathbf{1}^\top \bar{\mathsf{I}}}{N}\right)^\top \left(\mathbf{I}_N - \mathbf{1}\mathbf{1}^\top \frac{1}{N}\right) \\
&= \left(\frac{1}{\sigma}\mathbf{I}_N - \frac{1}{\sigma^3}\left(\bar{\mathsf{I}} - \frac{\mathbf{1}^\top \bar{\mathsf{I}}}{N}\right)\left(\bar{\mathsf{I}} - \frac{\mathbf{1}^\top \bar{\mathsf{I}}}{N}\right)^\top\right)\left(\mathbf{I}_N - \mathbf{1}\mathbf{1}^\top \frac{1}{N}\right)
\end{aligned} \tag{25}$$

with $\mathbf{I}_N$ being the $N \times N$ identity matrix and $\mathbf{1}$ being the $N$-dimensional vector of ones.

# B Source to Target Frame

## B.1 Plane

The derivative of the reprojection with respect to plane $\mathbf{n}$ is

$$\frac{\partial}{\partial \mathbf{n}} \mathbf{R}_j \mathbf{R}_i^\top \left(\frac{\bar{\mathbf{x}}}{\mathbf{n}^\top \bar{\mathbf{x}}} - \mathbf{t}_i\right) + \mathbf{t}_j = \mathbf{R}_j \mathbf{R}_i^\top \frac{\bar{\mathbf{x}}\bar{\mathbf{x}}^\top}{-(\mathbf{n}^\top \bar{\mathbf{x}})^2}. \tag{26}$$

## B.2 Poses

Rotations are stored as quaternions $\mathbf{q} = \mathbf{v} + w$ where $\mathbf{R}(\mathbf{q})$ is the corresponding rotation matrix.

### B.2.1 Quaternion Vector Transformation

Rotating $\mathbf{a} \in \mathbb{R}^3$ according to $\mathbf{q}$ results in

$$\begin{aligned}
\mathbf{R}(\mathbf{q})\mathbf{a} &= \mathbf{v} + 2w(\mathbf{v} \times \mathbf{a}) + \mathbf{v} \times 2(\mathbf{v} \times \mathbf{a}) \\
&= \mathbf{v} + 2w(\mathbf{v} \times \mathbf{a}) + 2\left((\mathbf{v}^\top \mathbf{a})\mathbf{v} - (\mathbf{v}^\top \mathbf{v})\mathbf{a}\right).
\end{aligned} \tag{27}$$

### B.2.2 Quaternions

The Jacobian of a rotated vector with respect to the quaternion is

$$\frac{\partial \mathbf{R}(\mathbf{q})\mathbf{a}}{\partial \mathbf{q}} = 2\left[\mathbf{v}^\top \mathbf{a}\mathbf{I}_3 + \mathbf{v}\mathbf{a}^\top - 2\mathbf{a}\mathbf{v}^\top - w[\mathbf{a}]_\times \mid \mathbf{v} \times \mathbf{a}\right] \in \mathbb{R}^{3\times 4} \tag{28}$$

with $[\mathbf{a}]_\times$ being the skew symmetric matrix with entries according to $\mathbf{a}$. The Jacobian of the quaternion inversion is

$$\frac{\partial \mathbf{q}^{-1}}{\partial \mathbf{q}} = \frac{\partial \frac{\bar{\mathbf{q}}}{\|\mathbf{q}\|^2}}{\partial \mathbf{q}} = \frac{\partial \bar{\mathbf{q}}}{\partial \mathbf{q}} \frac{1}{\|\mathbf{q}\|^2} - \bar{\mathbf{q}}\mathbf{q}^\top \frac{2}{\|\mathbf{q}\|^3}. \tag{29}$$

and the Jacobian of the quaternion conjugation is

$$\frac{\partial \bar{\mathbf{q}}}{\partial \mathbf{q}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}. \tag{30}$$

Jacobians of the reprojection into a target camera coordinate system with respect to the involved quaternions are obtained as the derivative of a vector rotation

$$\frac{\partial}{\partial \mathbf{q}_j}\mathbf{R}_j\mathbf{R}_i^\top\left(\frac{\bar{\mathbf{x}}}{\mathbf{n}^\top\bar{\mathbf{x}}}-\mathbf{t}_i\right)+\mathbf{t}_j=\frac{\partial \mathbf{R}(\mathbf{q}_j)\left(\mathbf{R}_i^\top\left(\frac{\bar{\mathbf{x}}}{\mathbf{n}^\top\bar{\mathbf{x}}}-\mathbf{t}_i\right)\right)}{\partial \mathbf{q}_j}$$

$$\frac{\partial}{\partial \mathbf{q}_i}\mathbf{R}_j\mathbf{R}_i^\top\left(\frac{\bar{\mathbf{x}}}{\mathbf{n}^\top\bar{\mathbf{x}}}-\mathbf{t}_i\right)+\mathbf{t}_j=\mathbf{R}_j\frac{\partial \mathbf{R}(\mathbf{q}_i^{-1})\left(\frac{\bar{\mathbf{x}}}{\mathbf{n}^\top\bar{\mathbf{x}}}-\mathbf{t}_i\right)}{\partial \mathbf{q}_i^{-1}}\frac{\partial \mathbf{q}_i^{-1}}{\partial \mathbf{q}_i} \tag{32}$$

where $\mathbf{q}_i$ and $\mathbf{q}_j$ are the quaternions for rotation $\mathbf{R}_i$ and $\mathbf{R}_j$, respectively. The Jacobians with respect to the translation vectors are

$$\frac{\partial}{\partial \mathbf{t}_j}\mathbf{R}_j\mathbf{R}_i^\top\left(\frac{\bar{\mathbf{x}}}{\mathbf{n}^\top\bar{\mathbf{x}}}-\mathbf{t}_i\right)+\mathbf{t}_j=\mathbf{I}_3 \tag{33}$$

$$\frac{\partial}{\partial \mathbf{t}_i}\mathbf{R}_j\mathbf{R}_i^\top\left(\frac{\bar{\mathbf{x}}}{\mathbf{n}^\top\bar{\mathbf{x}}}-\mathbf{t}_i\right)+\mathbf{t}_j=-\mathbf{R}_j\mathbf{R}_i^\top. \tag{34}$$

### B.2.3 Pose Increments

For a twist $\boldsymbol{\xi}\in\mathbb{R}^6$ with the first three components representing the translational part and the last three representing the rotation vector and $\exp:\mathbb{R}^6\to \mathrm{SE}(3)$ we obtain the jacobian [13]

$$\frac{\partial \exp(\boldsymbol{\xi})\mathbf{a}}{\partial \boldsymbol{\xi}}=[\mathbf{I}_3\mid -[\mathbf{a}]_\times]\in\mathbb{R}^{3\times 6}. \tag{35}$$

For $\mathbf{T}\in\mathrm{SE}(3)$ with rotation $\mathbf{R}$ and translation $\mathbf{t}$ appearing inverted

$$\frac{\partial(\exp(\boldsymbol{\xi})\mathbf{T})^{-1}\mathbf{a}}{\partial \boldsymbol{\xi}}=\frac{\partial \mathbf{T}^{-1}\exp(-\boldsymbol{\xi})\mathbf{a}}{\partial \boldsymbol{\xi}}=\frac{\partial \mathbf{R}^{-1}\exp(-\boldsymbol{\xi})\mathbf{a}-\mathbf{t}}{\partial \boldsymbol{\xi}}=\left[-\mathbf{R}^{-1}\mid \mathbf{R}^{-1}[\mathbf{a}]_\times\right]\in\mathbb{R}^{3\times 6}. \tag{36}$$

### B.2.4 Relative Poses [12]

Formally, with $\mathbf{T}\oplus\boldsymbol{\xi}=\exp(\boldsymbol{\xi})\mathbf{T}$ and $\mathbf{T}_1\ominus\mathbf{T}_2=\log(\mathbf{T}_1\mathbf{T}_2^{-1}),\log:\mathrm{SE}(3)\to\mathbb{R}^6$ we define

$$\frac{\partial f(\mathbf{T})}{\partial \mathbf{T}}=\lim_{\boldsymbol{\xi}\to 0}\frac{f(\mathbf{T}\oplus\boldsymbol{\xi})\ominus f(\mathbf{T})}{\boldsymbol{\xi}} \tag{37}$$

for some $f:\mathrm{SE}(3)\to\mathrm{SE}(3)$. With

$$\mathrm{Adj}(\mathbf{T})=\begin{bmatrix}\mathbf{R} & [\mathbf{t}]_\times\mathbf{R}\\ \mathbf{0} & \mathbf{R}\end{bmatrix},\qquad \exp(\mathrm{Adj}(\mathbf{T})\boldsymbol{\xi})=\mathbf{T}\exp(\boldsymbol{\xi})\mathbf{T}^{-1} \tag{38}$$

we compute the relative to absolute pose derivatives

$$\begin{aligned}\frac{\partial \mathbf{T}_1\mathbf{T}_2^{-1}}{\partial \mathbf{T}_2}&=\lim_{\boldsymbol{\xi}\to 0}\frac{\log(\mathbf{T}_1(\exp(\boldsymbol{\xi})\mathbf{T}_2)^{-1}(\mathbf{T}_1\mathbf{T}_2^{-1})^{-1})}{\boldsymbol{\xi}}\\ &=\lim_{\boldsymbol{\xi}\to 0}\frac{\log(\mathbf{T}_1\mathbf{T}_2^{-1}\exp(-\boldsymbol{\xi})(\mathbf{T}_1\mathbf{T}_2^{-1})^{-1})}{\boldsymbol{\xi}}\\ &=\lim_{\boldsymbol{\xi}\to 0}\frac{\log(\exp(-\mathrm{Adj}(\mathbf{T}_1\mathbf{T}_2^{-1})\boldsymbol{\xi}))}{\boldsymbol{\xi}}\\ &=\lim_{\boldsymbol{\xi}\to 0}\frac{-\mathrm{Adj}(\mathbf{T}_1\mathbf{T}_2^{-1})\boldsymbol{\xi}}{\boldsymbol{\xi}}\\ &=-\mathrm{Adj}(\mathbf{T}_1\mathbf{T}_2^{-1})\end{aligned} \tag{39}$$

$$\begin{aligned}\frac{\partial \mathbf{T}_1\mathbf{T}_2^{-1}}{\partial \mathbf{T}_1}&=\lim_{\boldsymbol{\xi}\to 0}\frac{\log(\exp(\boldsymbol{\xi})\mathbf{T}_1\mathbf{T}_2^{-1}(\mathbf{T}_1\mathbf{T}_2^{-1})^{-1})}{\boldsymbol{\xi}}\\ &=\lim_{\boldsymbol{\xi}\to 0}\frac{\boldsymbol{\xi}}{\boldsymbol{\xi}}\\ &=\mathbf{I}_6.\end{aligned} \tag{40}$$

# C Camera Intrinsics

## C.1 Perspective Projection

With $\mathbf{X} = [x, y, z]^\top$ the Jacobian for the perspective projection is

$$\frac{\partial \pi(\mathbf{X})}{\partial \mathbf{X}} = \begin{bmatrix} 1/z & 0 & -x/z^2 \\ 0 & 1/z & -y/z^2 \end{bmatrix}. \tag{41}$$

## C.2 Distortion

For the radial distortion

$$\varphi(\mathbf{x}) = \mathbf{x}(1 + l_1 r^2 + l_2 r^4) \qquad r^2 = \|\mathbf{x}\|^2 = \mathbf{x}^\top \mathbf{x} = x^2 + y^2 \tag{42}$$

the Jacobian with respect to $\mathbf{x}$ is

$$\begin{aligned} \frac{\partial \varphi(\mathbf{x})}{\partial \mathbf{x}} &= \begin{bmatrix} 1 + l_1(2x^2 + r^2) + l_2(4x^2 r^2 + r^4) & 2xyl_1 + 4xyr^2 l_2 \\ 2xyl_1 + 4xyr^2 l_2 & 1 + l_1(2y^2 + r^2) + l_2(4y^2 r^2 + r^4) \end{bmatrix} \\ &= \mathbf{I}_2(1 + l_1 r^2 + l_2 r^4) + \mathbf{x}(2l_1 \mathbf{x}^\top + 4l_2 r^2 \mathbf{x}^\top) \\ &= \mathbf{I}_2(1 + l_1 r^2 + l_2 r^4) + \mathbf{x}\mathbf{x}^\top(2l_1 + 4l_2 r^2) \end{aligned}$$

and the Jacobian with respect to $\mathbf{l} = [l_1, l_2]^\top$ is

$$\frac{\partial \varphi(\mathbf{x})}{\partial \mathbf{l}} = [\mathbf{x}r^2, \mathbf{x}r^4].$$