Statistically Consistent Total Least-Squares Estimation of Object Scales

Arne Hasselbring[0000−0003−0350−9817] and Udo Frese[0000−0001−8325−6324]

University of Bremen, Faculty 3 – Mathematics and Computer Science,
Postfach 330 440, 28334 Bremen, Germany
{arha,ufrese}@uni-bremen.de

Abstract. Estimating object poses is a fundamental problem in computer vision in general as well as for robotic manipulation in particular. Most approaches require a known 3D model of the object. One step towards a more general formulation is to estimate the object's width, height and depth with the pose, e. g. consider a generic box, cylinder or plate instead of one with known dimensions.
This paper investigates the last stage of such a pipeline, namely least-squares estimating pose and scales from point correspondences aggregated into a fixed size matrix. Therefore it encapsulates the scaled $SO(3)$ manifold in a so-called $\boxplus$-operator and derives a Gauss-Newton based optimizer with initial guess on that.
We find that the resulting least-squares estimator is strongly biased towards small scales. The reason for that lies in the mathematical structure of the least-squares loss, where noise in recognized object points is multiplied with the to be estimated transformation matrix. This violates the least-squares assumption of additive noise. It has no effect in the prevalent use of this loss for pose estimation but affects the scale.
We propose a solution to this bias based on an approximation of total least-squares that preserves the advantage of a fixed size representation and show that it provides relatively consistent uncertainty estimates.

Keywords: Least-Squares Estimation · 9D Object Pose Estimation · RGB-D Perception

1 Introduction

Object pose estimation is an important perception skill in a variety of applications, such as household robots. In particular robot applications require a metrical pose, not just a 2D bounding box. Most modern approaches [20,25] generate point pairs $\mathbf{p}_i^O, \mathbf{p}_i^C$ in object, respectively camera frame and match these to obtain a pose. Usually the $\mathbf{p}_i^C$ come directly from the sensor, the $\mathbf{p}_i^O$ are the output of a recognition process, e. g. by keypoints or a CNN. The last stage of this matching is usually least-squares. There are various cases, but all minimize a loss that's a second order term in the coefficients of the 4×4 transformation matrix $\mathbf{T} \in SE(3)$ representing the pose. This can be directly seen, if

the points in camera frame are 3D, e. g. from a depth camera:

$$\hat{\mathbf{T}} = \arg\min_{\mathbf{T}\in SE(3)} \mathcal{L}_{\text{pp}}(\mathbf{T}), \tag{1}$$

$$\mathcal{L}_{\text{pp}}(\mathbf{T}) = \sum_i \left\|\mathbf{T}\mathbf{p}_i^O - \mathbf{p}_i^C\right\|_2^2 = \sum_i \left(\mathbf{T}\mathbf{p}_i^O - \mathbf{p}_i^C\right)^\top \left(\mathbf{T}\mathbf{p}_i^O - \mathbf{p}_i^C\right) \tag{2}$$

where $\mathcal{L}_{\text{pp}}$ is actually even linear in the rotation part of $\mathbf{T}$, which is exploited by the well-known Umeyama algorithm [23]. This is known as point cloud matching. The point matches can also be weighted by a covariance matrix $\mathbf{\Sigma}_i$, e. g. because in stereo vision, the viewing direction is the more uncertain:

$$\mathcal{L}_{\text{ppc}}(\mathbf{T}) = \sum_i \left\|\mathbf{T}\mathbf{p}_i^O - \mathbf{p}_i^C\right\|_{\mathbf{\Sigma}_i^{-1}}^2 = \sum_i \|\mathbf{\Sigma}_i^{-\frac{1}{2}} \left(\mathbf{T}\mathbf{p}_i^O - \mathbf{p}_i^C\right)\|_2^2 \tag{3}$$

Here, $\mathcal{L}_{\text{ppc}}$ becomes second order.

For perspective matches, the $\mathbf{p}_i^C$ is replaced by image coordinates $\left(\begin{smallmatrix} u_i \\ v_i \end{smallmatrix}\right)$. The resulting function is non-linear in the coefficients of $\mathbf{T}$ due to the division in the perspective projection. However, it can be approximated linearly by multiplying with the denominator, the direct linear method [8, §7.1].

$$\mathcal{L}_{\text{persp}}(\mathbf{T}) = \sum_i \left\| \begin{pmatrix} -f & 0 & u-u_0 & 0 \\ 0 & -f & v-v_0 & 0 \end{pmatrix} \mathbf{T}\mathbf{p}_i^O \right\|_2^2 \tag{4}$$

Here, f is the focal length and $\left(\begin{smallmatrix} u_0 \\ v_0 \end{smallmatrix}\right)$ the image center. The loss is again second order in the coefficients of $\mathbf{T}$.

Depth measurements d_i corresponding to $\mathbf{p}_i^O$ can also be expressed.

$$\mathcal{L}_{\text{depth}}(\mathbf{T}) = \sum_i \left(\begin{pmatrix} 0 & 0 & 1 & 0 \end{pmatrix} \mathbf{T}\mathbf{p}_i^O - d_i\right)^2 \tag{5}$$

This also allows to model an RGB-D camera more precisely with $\mathcal{L}_{\text{persp}} + \mathcal{L}_{\text{depth}}$ instead of matching points with $\mathcal{L}_{\text{pp}}$. All different variants condense the information from the raw measurements into a 13×13 matrix $\mathbf{\Omega}$.

$$\mathcal{L}(\mathbf{T}) = \bar{\mathbf{T}}^\top \mathbf{\Omega} \bar{\mathbf{T}}, \tag{6}$$

where $\bar{\mathbf{T}}$ is a flattened vector of the 12 non-constant coefficients of $\mathbf{T}$ and a constant 1. The latter allows to include the first and zeroth order terms of $\mathcal{L}(\mathbf{T})$ in $\mathbf{\Omega}$ (details in Subsect. 3.2).

This representation has been used to estimate the object's pose [20] along with a covariance matrix as uncertainty measure. Such an uncertainty measure is useful for fusing the pose with other information [21] and for higher mission control. The initial idea of our work was to extend the framework to estimating the object's pose and three-axes scale (object axes). This appears promising, as scaling is already possible with $\mathbf{T}$ in (2)-(5) and only the optimization in (1) needs to be extended to include scale as well. Sect. 3 derives how this is done in detail.

However, it turns out that the resulting least-squares estimator has a strong bias towards smaller scales and also inconsistent covariance. This is caused by the fact that noise in $\mathbf{p}_i^O$ is multiplied by $\mathbf{T}$ and thus not additive as least-squares theory assumes. Sect. 4 analyzes this observation and proposes a solution using total least-squares theory [4]. While in general total least squares is much more computationally expensive, we propose an approximation that is statistically relatively consistent in experiments and needs only slightly more time. In particular, it saves the advantage of (6) to condense an arbitrary number of points into a fixed size representation.

Thus, this paper contributes a novel algorithm to estimate object poses with three-axes scale, including the following subcontributions:

- A formalization of the space of scaling transformations as $\boxplus$-manifold.
- An optimization method based on Gauss-Newton.
- An investigation of a bias caused by noise multiplied by variables and its correction by a total least-squares approximation.

The remainder of this paper is organized as follows: after an overview of related work in Sect. 2, the Sects. 3 and 4 discuss the least-squares and total least-squares approach to the simple point-to-point loss (2). Sect. 5 extends to the perspective (4) and depth (5) loss. Finally, Sect. 6 presents experimental results.

2 Related Work

6D object pose estimation is a much researched problem, with progress measured by the BOP challenge [10]. Traditionally, CAD models and large image datasets of the objects to be detected have been required. Recently, tasks have been added to the BOP challenge that aim at handling objects without such detailed knowledge. Our work is related to the topic of category-level pose estimation. In this variant of the pose estimation problem, objects of similar shape are handled by a common detector, which in turn requires some parameterization of the instance-level properties. The object scales are one of these properties.

Wang et al. [25] propose predicting normalized object coordinates by a neural network (which they call NOCS maps). They compare regression and classification approaches for the output of the neural network. In order to estimate the transformation, they align the masked point cloud from a depth camera with the predicted NOCS map using RANSAC and a classic least-squares solution [23]. A similar method is used in [17]. Our work is applicable to that approach in that it could replace the point alignment, if uncertainty information about the NOCS map was available.

Wei et al. [26] split the transformation estimation into a learned object scale predictor and a classic RANSAC PnP algorithm to determine translation and rotation. Other approaches [6,11,12] directly predict the transformation and shape parameters using a neural network. This has the disadvantage that the neural network has to generalize over camera intrinsics (i. e. training data must be available), and it is harder to fuse the result with other input modalities.

Regarding uncertainty in pose estimation, Brachmann et al. [3] use random forests to predict noise in pixel coordinate measurements. However, this is not used to provide uncertainty information about the predicted transformation. Wursthorn et al. [27] use deep ensembles to quantify the uncertainty of pose estimation. This comes at a cost in inference time though, and their uncertainty metric provides less information than a covariance matrix. Richter-Klug and Frese [20] predict pixelwise weights using a CNN and use them for least-squares estimation, which provides a covariance matrix of the estimate. This is methodically closest to our approach is it uses the same $\boxplus$-Gauss-Newton formalism as we do.

The problem of finding an orthonormal matrix that minimizes the squared error between two sets of corresponding points has been discovered across disciplines, e. g. as Wahba's problem [24] or orthogonal Procrustes problem [22]. There are analytic solutions based on the singular value decomposition for certain variants of the problem. For instance, Kabsch's algorithm [13,14] allows weighting factors per point. Umeyama's algorithm [23] does not explicitly include weighting, but outputs a single isotropic scaling factor. Everson [5] investigates the Procrustes problem where the transformation does not need to be normal, i. e. scaling is allowed.

Computing the pose from 3D-2D correspondences is a long researched problem in computer vision [18,16,15,28]. In these works the focus lies on finding a global solution algebraically, e. g. by Gröbner bases. All mentioned methods model noisy data at most as isotropic per point and they do not provide a measure of uncertainty of the estimated transformation. In fact, it has to be defined first what a covariance matrix on transformations means.

3 Least-Squares Estimation of Pose and Scale

Starting from an image of an object, there are three steps to determine its transformation matrix ($\mathbf{T} = \mathbf{T}_{C \leftarrow O}$) and its uncertainty $\mathbf{\Sigma}$:

1. Identify pixels u_i, v_i in the image that correspond to certain 3D points $\mathbf{p}_i^O$ of the object, either with keypoints (sparse) or by a CNN (dense)
2. Aggregate measurements into a fixed size (13×13) information matrix $\mathbf{\Omega}$
   (a) Aggregate perspective measurements u_i, v_i by (4)
   (b) If available, aggregate depth measurements d_i by (5)
   (c) Alternatively convert u_i, v_i, d_i to 3D-points $\mathbf{p}_i^C$ and aggregrate them by (2)
3. Obtain the max-likelihood transformation by minimizing the aggregated loss $\bar{\mathbf{T}}^\top \mathbf{\Omega} \bar{\mathbf{T}}$ over the manifold of valid transformations $\mathbf{T}$.
   (a) Find a valid initial guess $\mathbf{T}_0$
   (b) Iterate: Perform a Gauss-Newton iteration using a $\boxplus$-operator to encapsulate the manifold
   (c) Obtain a covariance matrix $\mathbf{\Sigma}$ from the converged Gauss-Newton

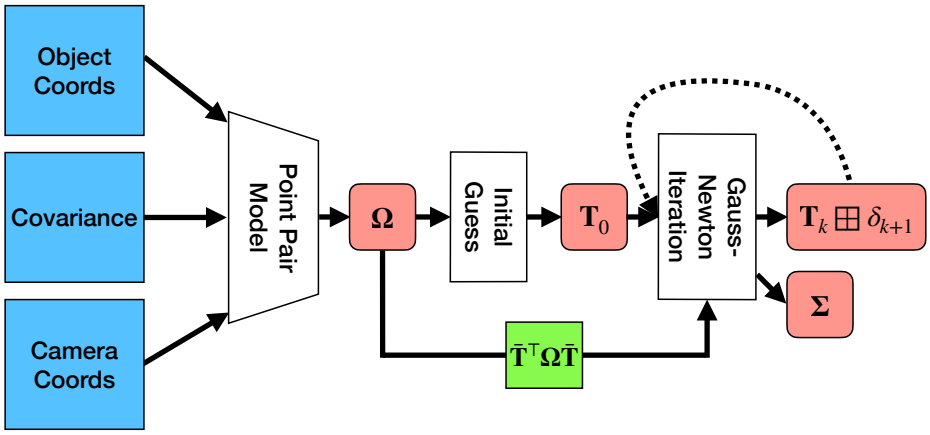

Fig. 1. System overview. The object coordinates, camera coordinates and covariance matrices are supplied externally, e. g. by a neural network from an RGB image and a depth camera. The sensor model condenses the problem to a fixed size matrix $\mathbf{\Omega}$. From this, the initial guess $\mathbf{T}_0$ is calculated. Finally, Gauss-Newton produces a δ_{k+1} that is added via $\boxplus$ to the iterate, and a covariance matrix.

An overview of the approach is given in Fig. 1.

Such an approach has been used by [20] with a CNN in the first step to estimate poses with uncertainty. In order to adapt this method to also estimate object scales, we need to extend the following steps:

- The optimizer. In our framework, this means describing the transformation space as $\boxplus$-manifold (cf. Subsect. 3.1).
- The initial guess. The previous approach was based on evaluating a fixed set of rotations and recovering the optimal translation. Now, we can still use a fixed set of rotations, but the optimal scale is not a linear function of the rotation, instead requiring the solution of a 4×4 linear equation per rotation (cf. Subsect. 3.3).

Notably, the neural network that predicts 3D points requires no change, except to output normalized coordinates. Constructing the information matrix needs no adaptation either.

Throughout the paper we use homogenous points $\mathbf{p} \in \mathbb{R}^4$ with a 1 as fourth component to flag a location vector. Occasionally, we will need only the 3 actual coordinates, this is noted as $\mathbf{p}_\blacksquare$. Similarly, for a matrix $\mathbf{A} \in \mathbb{R}^{4\times4}$, $\mathbf{A}_\blacksquare$ denotes the left/upper 3×3 part.

3.1 The Scaled $SO(3)$ Manifold

We want to model 3D transformations from object to camera coordinates that allow positive scaling along the object axes. Ignoring translation for now, these

scaling-followed-by-rotation matrices are described by the following state space:

$$\begin{aligned}\mathcal{Q} &= \{\mathbf{R}\,\mathrm{diag}(\mathbf{s}) \mid \mathbf{R} \in SO(3), \mathbf{s} \in \mathbb{R}^3_{>0}\} \\ &= \{\mathbf{Q} \mid \exists \mathbf{s} \in \mathbb{R}^3_{>0} \colon \mathbf{Q}^\top \mathbf{Q} = \mathrm{diag}(\mathbf{s})^2 \wedge \det(\mathbf{Q}) > 0\} \\ &= \{\mathbf{Q} \mid \mathbf{Q}_i^T \mathbf{Q}_j = 0 \; \forall \; i \neq j \wedge \det(\mathbf{Q}) > 0\}. \end{aligned} \tag{7}$$

$\mathcal{Q}$ forms a submanifold of GL(3). It is not closed under matrix multiplication since the scaling from the first operand would be applied to the rotated axes of the second operand. This is not necessarily a scaling of the original object axes. Instead, the rotating and scaling components must be modified separately to stay on the manifold, as seen below. This also means that, unlike $SO(3)$, this space does not form a Lie-group with matrix multiplication.

In order to use an iterative optimizer and the notion of Gaussian distributions on this space, we view it as $\boxplus$-manifold [9]. This method encapsulates the manifold structure in two closed, smooth operators defining a local parameterization (tangent space) of perturbations to a state.

$$\boxplus \colon \mathcal{Q} \times \mathbb{R}^d \to \mathcal{Q} \tag{8}$$

$$\boxminus \colon \mathcal{Q} \times \mathcal{Q} \to \mathbb{R}^d \tag{9}$$

The $\boxplus$-operator takes a a manifold element $\mathbf{Q}$ and adds a vector δ from the d-dimensional tangent space, returning a new manifold element $\mathbf{Q} \boxplus \delta$. The inverse operator $\boxminus$ returns the tangent space vector required to go from one manifold element to another, i. e. $\mathbf{Q}_1 \boxplus (\mathbf{Q}_2 \boxminus \mathbf{Q}_1) = \mathbf{Q}_2$.

In order to define these operators on our "scaled $SO(3)$" manifold, we need two auxiliary functions, σ and ρ, which extract the scaling and rotating components from $\mathbf{Q}$, respectively.

$$\begin{aligned}\sigma \colon \mathcal{Q} &\to \{\mathrm{diag}(\mathbf{s}) \mid \mathbf{s} \in \mathbb{R}^3_{>0}\} \\ \mathbf{Q} &\mapsto \sqrt{\mathbf{Q}^\top \mathbf{Q}} \end{aligned} \tag{10}$$

$$\begin{aligned}\rho \colon \mathcal{Q} &\to SO(3) \\ \mathbf{Q} &\mapsto \mathbf{Q} \cdot \sigma(\mathbf{Q})^{-1} \end{aligned} \tag{11}$$

σ is well-defined since $\mathbf{Q}^\top \mathbf{Q}$ is diagonal and positive (by definition of $\mathcal{Q}$), and thus its square-root is well-defined and still diagonal and positive. ρ is well-defined since the inverse of a positive diagonal matrix always exists. The result of ρ is indeed in $SO(3)$ since

$$\rho(\mathbf{Q})^\top \rho(\mathbf{Q}) = \sigma(\mathbf{Q})^{-1} \mathbf{Q}^\top \mathbf{Q} \sigma(\mathbf{Q})^{-1} = \sigma(\mathbf{Q})^{-1} \sigma(\mathbf{Q})^2 \sigma(\mathbf{Q})^{-1} = \mathbf{I} \tag{12}$$

$$\det(\rho(\mathbf{Q})) = \frac{\det(\mathbf{Q})}{\det(\sigma(\mathbf{Q}))} = \frac{\det(\mathbf{Q})}{\sqrt{\det(\mathbf{Q}^\top \mathbf{Q})}} = \frac{\det(\mathbf{Q})}{|\det(\mathbf{Q})|} = 1. \tag{13}$$

It is straightforward that $\mathbf{Q} = \rho(\mathbf{Q})\sigma(\mathbf{Q})$, but ρ and σ do not commute in general.

Furthermore, we need two functions based on the matrix exponential (with $[\mathbf{x}]_\times$ denoting the cross product matrix)

$$\begin{aligned} \exp_\times \colon \mathbb{R}^3 &\to SO(3) \\ \mathbf{r} &\mapsto \exp[\mathbf{r}]_\times \end{aligned} \tag{14}$$

$$\begin{aligned} \exp_\circ \colon \mathbb{R}^3 &\to \{\operatorname{diag}(\mathbf{s}) \mid \mathbf{s} \in \mathbb{R}^3_{>0}\} \\ \mathbf{s} &\mapsto \exp \operatorname{diag} \mathbf{s} \end{aligned} \tag{15}$$

and their corresponding inverses:

$$\log_\times \colon SO(3) \to \mathbb{R}^3 \tag{16}$$

$$\log_\circ \colon \{\operatorname{diag}(\mathbf{s}) \mid \mathbf{s} \in \mathbb{R}^3_{>0}\} \to \mathbb{R}^3 \tag{17}$$

$\exp_\times$ corresponds to the well-known Rodrigues formula to convert an axis-angle into a rotation matrix, and $\exp_\circ$ maps real numbers to a diagonal positive scaling matrix by exponentiating them element-wise. While the $\circ$-mappings are actually bijective, $\exp_\times$ is only injective within $\|\mathbf{r}\| < \pi$. Furthermore, for the $\circ$-functions, the same rules for exponential functions hold as for real numbers (since they basically operate element-wise on the diagonal entries). On the other hand, $\exp_\times(\mathbf{r}_1)\exp_\times(\mathbf{r}_2) = \exp_\times(\mathbf{r}_1 + \mathbf{r}_2)$ does not hold in general.

Writing $\delta \in \mathbb{R}^6 = (\delta_\times, \delta_\circ)$, the $\boxplus$- and $\boxminus$-operators on $\mathcal{Q}$ are then defined as follows:

$$\begin{aligned} \boxplus \colon \mathcal{Q} \times \mathbb{R}^6 &\to \mathcal{Q} \\ (\mathbf{Q}, \delta) &\mapsto \exp_\times \delta_\times \mathbf{Q} \exp_\circ \delta_\circ \end{aligned} \tag{18}$$

$$\begin{aligned} \boxminus \colon \mathcal{Q} \times \mathcal{Q} &\to \mathbb{R}^6 \\ (\mathbf{Q}_2, \mathbf{Q}_1) &\mapsto (\log_\times(\rho(\mathbf{Q}_2)\rho(\mathbf{Q}_1)^{-1}), \\ &\qquad \log_\circ(\sigma(\mathbf{Q}_1)^{-1}\sigma(\mathbf{Q}_2)))^\top \end{aligned} \tag{19}$$

This means that $\boxplus$ uses $\delta_\times$ as axis-angle representation of a rotation and $\delta_\circ$ as logarithmic per-axis scale factors. $\boxminus$ recovers the tangent vector that is required to move from one state to another. This behavior is formalized by three axioms[1] that need to hold for $\boxplus$-manifolds, for which the proof is given in Appendix A.

In order to incorporate object translations, we extend our state space to full 4×4 transformation matrices:

$$\mathcal{T} = \left\{ \begin{pmatrix} \mathbf{Q} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \middle| \mathbf{Q} \in \mathcal{Q}, \mathbf{t} \in \mathbb{R}^3 \right\} \tag{20}$$

$\mathbb{R}^3$ satisfies the $\boxplus$-axioms with the usual vector addition / subtraction [9]. Thus we can define $\boxplus$- and $\boxminus$-operators on $\mathcal{T}$ (with $\mathbb{R}^9$ as tangent space) that act

[1] The original publication [9] required the axiom that the zero-vector is the neutral element of $\boxplus$. This can actually be derived from the other axioms and is trivial in this case anyway.

independently on $\mathbf{Q}$ and $\mathbf{t}$, which makes $\mathcal{T}$ a 9-dimensional $\boxplus$-manifold. Note that in the optimization process itself, the $\boxminus$-operator is not needed. It is however necessary for defining the concepts of an expected value, a covariance matrix and the Mahalanobis distance on $\mathbf{T}$.

A normal distribution on a $\boxplus$-manifold is defined by

$$\mathcal{N}(\mu, \mathbf{\Sigma}) \sim \mu \boxplus \mathcal{N}(0, \mathbf{\Sigma}). \tag{21}$$

This means the uncertainty of a transformation is quantified by a covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{9\times 9}$ on the tangent space at the mean $\mu \in \mathcal{T}$.

3.2 Point Pair Sensor Model

In our context, a sensor model is a function that maps a object-in-camera transformation matrix $\mathbf{T} = \mathbf{T}_{C\leftarrow O} \in \mathcal{T}$ (the variable in the optimization) to a residual. The squared norm of this residual is then the loss to be minimized, e. g. (2)-(5). Our sensor models are linear, leading to a quadratic loss. Thus we want to express them as a matrix-vector product, which motivates the flattened transformation matrix $\bar{\mathbf{T}} \in \mathbb{R}^{13}$ defined as

$$\bar{\mathbf{T}} = \begin{pmatrix} T_{11} & T_{12} & T_{13} & T_{21} & T_{22} & T_{23} & T_{31} & T_{32} & T_{33} & 1 & T_{14} & T_{24} & T_{34} \end{pmatrix}^\top. \tag{22}$$

It consists of the rotation part in row major order, a constant 1 and the translation column. Including the 1 in the flattened transformation allows us to use a single Jacobian for affine linear functions. The position of the 1 between rotation/scaling and translation coefficients is motivated by the block decomposition for taking the Schur complement in Eq. (29). Similarly, a homogeneous vector $\mathbf{p} \in \mathbb{R}^4$ can be converted to a matrix $\bar{\mathbf{p}} \in \mathbb{R}^{4\times 13}$ such that $\mathbf{Tp} = \bar{\mathbf{p}}\bar{\mathbf{T}}$. This means that we express the matrix vector product $\mathbf{Tp}$ as a linear operation on the matrix coefficients.

$$\bar{\mathbf{p}} = \begin{pmatrix} p_1 & p_2 & p_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p_4 & 0 & 0 \\ 0 & 0 & 0 & p_1 & p_2 & p_3 & 0 & 0 & 0 & 0 & 0 & p_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_1 & p_2 & p_3 & 0 & 0 & 0 & p_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p_4 & 0 & 0 & 0 \end{pmatrix} \tag{23}$$

Using this notation, a sensor model with M measurements is represented by a Jacobian $\mathbf{J} \in \mathbb{R}^{M\times 13}$. In addition, a covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{M\times M}$ represents noise in the measurements, such that at the true transformation $\mathbf{T}^*$

$$\mathbf{J}\bar{\mathbf{T}}^* \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}) \tag{24}$$

or equivalently

$$\mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{J}\bar{\mathbf{T}}^* \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_M). \tag{25}$$

The Hessian of the resulting normalized least-squares objective $\mathcal{L}(\mathbf{T}) = \|\mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{J}\bar{\mathbf{T}}\|^2$ (up to a factor of 2) is called the *information matrix* $\mathbf{\Omega} = \mathbf{J}^\top\mathbf{\Sigma}^{-1}\mathbf{J} \in \mathbb{R}^{13\times 13}$. Equation (25) implies that $\bar{\mathbf{T}}^{*\top}\mathbf{\Omega}\bar{\mathbf{T}}^*$ is χ^2-distributed with M degrees of freedom.

The point pair sensor model assumes a set of N points of which the coordinates relative to the camera $\mathbf{p}_i^C \in \mathbb{R}^4$ and relative to the object $\mathbf{p}_i^O \in \mathbb{R}^4$ (with $\mathbf{p}_{i4}^C = \mathbf{p}_{i4}^O = 1$) are known. In a setting where the object scales should be estimated, it makes sense for the object coordinates to be normalized to the range $[0, 1]$, as done in [25]. The Jacobian for a single 3D point measurement is then

$$\mathbf{J}_{\mathrm{pp},i} = \begin{pmatrix} \mathbf{I}_3 & -\mathbf{p}_{i\blacksquare}^C \end{pmatrix} \bar{\mathbf{p}}_i^O. \tag{26}$$

If the point measurements are independent, the $3N \times 3N$ covariance matrix becomes 3×3 block-diagonal and is parameterized by $\mathbf{\Sigma}_i \in \mathbb{R}^{3\times3}$ per point. Then each point contributes the summand $\mathbf{J}_{\mathrm{pp},i}^\top \mathbf{\Sigma}_i^{-1} \mathbf{J}_{\mathrm{pp},i}$ to $\mathbf{\Omega}$.

3.3 Initial Guess

The iterative Gauss-Newton algorithm needs a starting transformation $\mathbf{T}_0$. Often RANSAC is used before the final least-squares to handle outliers which also provides an initial guess. We take here the view to aggregate all measurements into $\mathbf{\Omega}$ and thus provide an algorithm to obtain an initial guess from $\mathbf{\Omega}$.

We can decompose the optimization into rotation, scale and translation:

$$\min_{\mathbf{T}\in\mathcal{T}} \bar{\mathbf{T}}^\top \mathbf{\Omega} \bar{\mathbf{T}} = \min_{\mathbf{R}\in SO(3)} \min_{\mathbf{s}\in\mathbb{R}^3_{>0}} \min_{\mathbf{t}\in\mathbb{R}^3} \bar{\mathbf{T}}^\top \mathbf{\Omega} \bar{\mathbf{T}} \tag{27}$$

with $\mathbf{T} = \begin{pmatrix} \mathbf{Q} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}, \mathbf{Q} = \mathbf{R}\,\mathrm{diag}(\mathbf{s})$. Using the block decomposition

$$\mathbf{\Omega} = \begin{pmatrix} \mathbf{\Omega}_Q & \mathbf{\Omega}_{Qt} \\ \mathbf{\Omega}_{Qt}^\top & \mathbf{\Omega}_t \end{pmatrix} \tag{28}$$

we can eliminate the translation by Schur complement, decomposing $\bar{\mathbf{T}} = \begin{pmatrix} \bar{\mathbf{Q}} \\ \mathbf{t} \end{pmatrix}$ into the 9 rotation coefficients, including the fixed 1 as $\bar{\mathbf{Q}}$ and the 3 translation coefficients $\mathbf{t}$.

$$\min_{\mathbf{R}\in SO(3)} \min_{\mathbf{s}\in\mathbb{R}^3_{>0}} \bar{\mathbf{Q}}^\top (\mathbf{\Omega}_Q - \mathbf{\Omega}_{Qt} \mathbf{\Omega}_t^{-1} \mathbf{\Omega}_{Qt}^\top) \bar{\mathbf{Q}}. \tag{29}$$

For a given rotation $\mathbf{R}$, $\mathbf{Q}$ is linear in the scale $\mathbf{s}$. Hence the flattened $\bar{\mathbf{Q}}$ can be expressed as a matrix vector product with $\mathbf{s}$.

$$\bar{\mathbf{Q}} = \breve{\mathbf{R}} \begin{pmatrix} \mathbf{s} \\ 1 \end{pmatrix}, \quad \breve{\mathbf{R}} = \begin{pmatrix} R_{i1} & 0 & 0 & 0 \\ 0 & R_{i2} & 0 & 0 \\ 0 & 0 & R_{i3} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{i=1..3} \tag{30}$$

Given a rotation we can rewrite the optimization problem in the scale vector $\mathbf{s}$.

$$\min_{\mathbf{R}\in SO(3)} \min_{\mathbf{s}\in\mathbb{R}^3_{>0}} \begin{pmatrix} \mathbf{s}^\top & 1 \end{pmatrix} \breve{\mathbf{R}}^\top (\mathbf{\Omega}_Q - \mathbf{\Omega}_{Qt} \mathbf{\Omega}_t^{-1} \mathbf{\Omega}_{Qt}^\top) \breve{\mathbf{R}} \begin{pmatrix} \mathbf{s} \\ 1 \end{pmatrix}. \tag{31}$$

The inner optimization problem is quadratic in $\mathbf{s}$ and can thus be solved analytically. This ignores the positivity constraint, so if the optimal $\mathbf{s}$ for some $\mathbf{R}$

isn't positive, it's simply discarded. Thus for a given rotation, we can calculate the optimal scale and subsequently the optimal translation as $-\mathbf{\Omega}_t^{-1}\mathbf{\Omega}_{Qt}^{\top}\bar{\mathbf{Q}}$. The rotational space can be efficiently sampled by a grid, by selecting each of the 60 vertices of a football (truncated icosahedron) as the x-axis and rotating around it in 22.5° steps.

3.4 Gauss-Newton on the Manifold

The initial guess is refined by the Gauss-Newton algorithm on the previously defined $\boxplus$-manifold, following the principles from [9]. In each iteration the current value $\mathbf{T}_k$ is changed by some δ in the tangential space using $\boxplus$. This δ is determined optimally in a linearized way.

$$\mathbf{T}_{k+1} \approx \mathbf{T}_k \boxplus \operatorname{lin}\arg\min_{\delta\in\mathbb{R}^9} \overline{(\mathbf{T}_k \boxplus \delta)}^{\top} \mathbf{\Omega} \overline{(\mathbf{T}_k \boxplus \delta)} \tag{32}$$

This requires a linearization of the $\boxplus$-operator in order to project the information matrix to the tangent space of the manifold. This is represented by the Jacobian $\mathbf{J}_{\boxplus,\delta}(\mathbf{T}) \in \mathbb{R}^{13\times 10}$, which satisfies locally for $\delta \in \mathbb{R}^9 \approx 0$ that $\overline{\mathbf{T} \boxplus \delta} \approx \mathbf{J}_{\boxplus,\delta}(\mathbf{T}) \begin{pmatrix}\delta^{\top} & 1\end{pmatrix}^{\top}$:

$$\mathbf{J}_{\boxplus,\delta}(\mathbf{T}) = \begin{pmatrix}
0 & T_{31} & -T_{21} & T_{11} & 0 & 0 & 0 & 0 & 0 & T_{11} \\
0 & T_{32} & -T_{22} & 0 & T_{12} & 0 & 0 & 0 & 0 & T_{12} \\
0 & T_{33} & -T_{23} & 0 & 0 & T_{13} & 0 & 0 & 0 & T_{13} \\
-T_{31} & 0 & T_{11} & T_{21} & 0 & 0 & 0 & 0 & 0 & T_{21} \\
-T_{32} & 0 & T_{12} & 0 & T_{22} & 0 & 0 & 0 & 0 & T_{22} \\
-T_{33} & 0 & T_{13} & 0 & 0 & T_{23} & 0 & 0 & 0 & T_{23} \\
T_{21} & -T_{11} & 0 & T_{31} & 0 & 0 & 0 & 0 & 0 & T_{31} \\
T_{22} & -T_{12} & 0 & 0 & T_{32} & 0 & 0 & 0 & 0 & T_{32} \\
T_{23} & -T_{13} & 0 & 0 & 0 & T_{33} & 0 & 0 & 0 & T_{33} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & T_{14} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & T_{24} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & T_{34}
\end{pmatrix} \tag{33}$$

Recall that the columns correspond to the three rotation, scaling and translation parameters, respectively, and the final column captures the constant part of the linearization. The rotation parameters affect $\mathbf{Q}$ according to the cross product matrix pattern. The scaling parameters act on the columns of $\mathbf{Q}$. The translation parameters are direct offsets to the translation of $\mathbf{T}$.

The resulting unconstrained quadratic optimization problem Eq. (35) can be solved by Cholesky decomposition to obtain the Gauss-Newton step, which is then applied to the iterate via the real nonlinear $\boxplus$-operator. The matrix $\mathbf{R}$ regularizes the linearized least-squares problem, such that empirically step size 1 can be chosen in Eq. (37). After N iterations, the final estimate $\hat{\mathbf{T}}$ is available, including a covariance $\hat{\mathbf{\Sigma}}$.

$$\mathbf{J}_k = \mathbf{J}_{\boxplus,\delta}(\mathbf{T}_k) \tag{34}$$

$$\delta_{k+1} = \arg\min_{\delta \in \mathbb{R}^9} \begin{pmatrix} \delta^\top & 1 \end{pmatrix} \left(\mathbf{J}_k^\top \mathbf{\Omega} \mathbf{J}_k + \mathbf{R} \right) \begin{pmatrix} \delta \\ 1 \end{pmatrix} \tag{35}$$

$$= -\mathbf{H}_k^{-1} \mathbf{g}_k, \quad \text{with} \quad \begin{pmatrix} \mathbf{H}_k & \mathbf{g}_k \\ \mathbf{g}_k^T & c_k \end{pmatrix} = \mathbf{J}_k^\top \mathbf{\Omega} \mathbf{J}_k + \mathbf{R} \tag{36}$$

$$\mathbf{T}_{k+1} = \mathbf{T}_k \boxplus \delta_{k+1} \tag{37}$$

$$\hat{\mathbf{T}} = \mathbf{T}_N \tag{38}$$

$$\hat{\mathbf{\Sigma}} = \mathbf{H}_N^{-1}, \quad \text{with } \mathbf{R} = 0 \tag{39}$$

The idea is that the output of Gauss-Newton approximately parameterizes the distribution of the true transformation $\mathbf{T}^*$, i. e. $\mathbf{T}^* \sim \mathcal{N}(\hat{\mathbf{T}}, \hat{\mathbf{\Sigma}})$.

4 Inconsistency and Total Least Squares Correction

By now, we have a complete algorithm to obtain a 9D transformation from point pair measurements. Point measurements are aggregated into the fixed size matrix $\mathbf{\Omega}$, from which an initial guess of the transformation is made, which is refined by Gauss-Newton to yield an estimate of the distribution parameters. However, running it on sample data (cf. Subsect. 6.1, experiment *PP2*) reveals that there is a problem. There is a bias towards smaller object size of 10%. This bias is not reflected in the estimated covariance matrix, such that the results are inconsistent by a factor of 6.

The reason is that the noisy object coordinates $\mathbf{p}_i^O$ are multiplied by the optimization variable $\mathbf{T}$ in (2). This is not allowed in standard least-squares problems, where the noise must be confined to the "right-hand side of the equation", i. e. additive as noise in $\mathbf{p}_i^C$. In our case, the optimizer can scale the noise in the object coordinates down to reduce the error. This creates a bias towards smaller scales. This problem does not appear as long as only a 6D pose is fitted, since the orthonormal rotation matrix preserves the level of the noise. In fact, it has been shown that the solution to the unitarily-constrained total least squares problem is the same as the standard orthogonal Procrustes solution [1].

This leads us to the insight that for estimating scales consistently, we do need to model the problem as total least squares [7] which allows a noisy data matrix. We follow the approach from [4] and [19, §15.3] to normalize the squared residual by its variance *at the current parameters*, instead of the variance at the true parameters.

4.1 Loss Function

In standard least squares the normalization with the variance is a constant precomputed weighting factor for the minimization and included in the matrix $\mathbf{\Omega}$ in (6). By constrast the current parameters $\bar{\mathbf{T}}$ change during optimization, so normalization at the current parameters makes the loss more complex. The direct interpretation of this paradigm results in a loss function with a sum of fractions, one per scalar measurement j. We use the index j for scalar measurements, while

i indexes logical measurements (e. g. a 3D point pair is one logical measurement that contributes 3 scalar measurements).

$$\mathcal{L}_{\text{tls}}(\mathbf{T}) = \sum_{j=1}^{M} \frac{\left(\mathbf{J}_j \bar{\mathbf{T}}\right)^2}{V(\mathbf{J}_j \bar{\mathbf{T}})} \tag{40}$$

For a scalar measurement j, where the Jacobian $\mathbf{J}_j$ is just a row vector, the denominator $V(\mathbf{J}_j \bar{\mathbf{T}})$ is the quadratic form $\bar{\mathbf{T}}^\top \operatorname{Cov}(\mathbf{J}_j) \bar{\mathbf{T}}$. $\operatorname{Cov}(\mathbf{J}_j)$ depends on the specific sensor model and the noise in the quantities affecting $\mathbf{J}_j$ (e. g. $\mathbf{p}^O$ and $\mathbf{p}^C$), but again can be precomputed as it does not change during optimization. The dependency on $\bar{\mathbf{T}}$ during optimization is purely quadratic.

Therefore, the previous parameterization of the optimization problem by $\mathbf{\Omega}$ must be split into a set of matrices for the numerators and denominators, which we call $\mathbf{\Omega}_j^U$ and $\mathbf{\Omega}_j^L$, respectively.[2]

$$\mathbf{\Omega}_j^U = \mathbf{J}_j^\top \mathbf{J}_j \tag{41}$$

$$\mathbf{\Omega}_j^L = \operatorname{Cov}(\mathbf{J}_j) \tag{42}$$

$$\mathcal{L}_{\text{tls}}(\mathbf{T}) = \sum_{j=1}^{M} \frac{\bar{\mathbf{T}}^\top \mathbf{\Omega}_j^U \bar{\mathbf{T}}}{\bar{\mathbf{T}}^\top \mathbf{\Omega}_j^L \bar{\mathbf{T}}} \tag{43}$$

If there is noise only in the right hand side, $\mathbf{\Omega}_j^L$ is only non-zero at the diagonal entry corresponding to the fixed 1 of $\bar{\mathbf{T}}$ and hence the denominator constant.

There are three problems with this formula:

1. Each measurement contributes a summand, so the problem cannot be represented by a constant number of parameters.
2. The measurements are assumed to be independent which they aren't, e. g. some random variables are shared or correlated within a logical measurement (e. g. the $x/y/z$ measurements of one point).
3. This function does not have the quadratic form that Gauss-Newton expects.

For the first problem, we make the assumption that the denominators (i. e. variances of individual measurements) are not too different (i. e. within the same order of magnitude). Then, $\sum_{j=1}^{M} \frac{x_j}{y_j} \approx M \frac{\sum x_j}{\sum y_j}$ is a valid approximation and we set

$$\mathbf{\Omega}^U = \sum_{j=1}^{M} \mathbf{\Omega}_j^U \tag{44}$$

$$\mathbf{\Omega}^L = \frac{1}{M} \sum_{j=1}^{M} \mathbf{\Omega}_j^L \tag{45}$$

[2] Technically, $\mathbf{\Omega}_j^L$ is a covariance matrix, not an information matrix. We still call it $\mathbf{\Omega}$ instead of $\mathbf{\Sigma}$ due to the similar role as the information matrix in the nominator.

$$\tilde{\mathcal{L}}_{\text{tls}}(\mathbf{T}) = \frac{\bar{\mathbf{T}}^\top \boldsymbol{\Omega}^U \bar{\mathbf{T}}}{\bar{\mathbf{T}}^\top \boldsymbol{\Omega}^L \bar{\mathbf{T}}} \tag{46}$$

This is a rather strong approximation. But it is also very beneficial, because it again combines all measurements into a fixed size representation $(\boldsymbol{\Omega}^U, \boldsymbol{\Omega}^L)$.

The second problem is partially addressed by keeping the weight matrix $\boldsymbol{\Sigma}^{-\frac{1}{2}}$ in (25) corresponding to noise in the right-hand side (e. g. $\mathbf{p}_i^C$). In textbook total least squares such a weight appears in the denominator, but being scalar that can not handle correlation in $x/y/z$ components. Furthermore, this approach improves the sum-of-fractions approximation in (46) as it makes the constant part of $\boldsymbol{\Omega}_j^L$ one and hence equal for all j.

4.2 Optimization

As in normal Gauss-Newton, we replace $\bar{\mathbf{T}}$ by $\mathbf{J}_k \left(\begin{smallmatrix}\delta\\1\end{smallmatrix}\right)$ and regularize the numerator, so the modified Gauss-Newton step becomes

$$\delta_{k+1}^{\text{tls}} = \arg\min_{\delta \in \mathbb{R}^9} \frac{\begin{pmatrix}\delta^\top & 1\end{pmatrix} \left(\mathbf{J}_k^\top \boldsymbol{\Omega}^U \mathbf{J}_k + \mathbf{R}\right) \begin{pmatrix}\delta\\1\end{pmatrix}}{\begin{pmatrix}\delta^\top & 1\end{pmatrix} \mathbf{J}_k^\top \boldsymbol{\Omega}^L \mathbf{J}_k \begin{pmatrix}\delta\\1\end{pmatrix}}. \tag{47}$$

In Gauss-Newton, the linearized least squares problem is quadratic and can be solved exactly. This would be complicated here. It's also not necessary since (47) is only an approximation itself. So we opted for using the full gradient at $\delta = 0$ of the fraction, but only the scaled nominator's Hessian, making it a preconditioner. Note that the Hessian only affects the steps not the final result [19, §15.5].

$$\tilde{\delta}_{k+1}^{\text{tls}} = -\left(\frac{\mathbf{H}_k^U}{c_k^L}\right)^{-1} \left(\frac{\mathbf{g}_k^U}{c_k^L} - \frac{c_k^U}{{c_k^L}^2}\mathbf{g}_k^L\right) \tag{48}$$

$$\text{with } \begin{pmatrix}\mathbf{H}_k^U & \mathbf{g}_k^U\\ \mathbf{g}_k^{U\top} & c_k^U\end{pmatrix} = \mathbf{J}_k^\top \boldsymbol{\Omega}^U \mathbf{J}_k + \mathbf{R}, \quad \begin{pmatrix}\mathbf{H}_k^L & \mathbf{g}_k^L\\ \mathbf{g}_k^{L\top} & c_k^L\end{pmatrix} = \mathbf{J}_k^\top \boldsymbol{\Omega}^L \mathbf{J}_k \tag{49}$$

We don't reduce the fraction $\frac{\mathbf{H}_k^U}{c_L}$ here to highlight the similarity to the multi-sensor case in Subsect. 5.3.

The initial guess is not adapted and uses only $\boldsymbol{\Omega}^U$, since the scale bias is small enough to be refined by Gauss-Newton. Furthermore, the initial guess is not involved in the final covariance matrix.

4.3 Covariance Matrix

It is proven in [4] that for sufficiently high signal-to-noise ratios the posterior covariance matrix of the *unconstrained* TLS-problem (43) is given by

$$\hat{\boldsymbol{\Sigma}} = \left(\sum_{j=1}^{M} \frac{\boldsymbol{\Omega}_j^U}{\hat{\bar{\mathbf{T}}}^\top \boldsymbol{\Omega}_j^L \hat{\bar{\mathbf{T}}}}\right)^{-1}. \tag{50}$$

In our case, the approximations from Subsect. 4.1 are also employed here. Apart from that, the optimization is constrained, i. e. the $\boxplus$-Jacobian needs to be considered as in Subsect. 3.4. The result is the matrix factor in front of the gradient in (48). This also justifies using that factor instead of the full fraction's Hessian in the optimization.

$$\hat{\mathbf{\Sigma}} = \left(\frac{\mathbf{H}_N^U}{c_N^L} \right)^{-1}, \quad \text{with } \mathbf{R} = 0 \tag{51}$$

4.4 Application to the Point Pair Sensor Model

In the sensor model from Subsect. 3.2, the measurement uncertainty was represented by a single covariance matrix that normalizes the residual to unit variance. For the total least squares approach, we need to attribute the uncertainty to the object and camera coordinates. Both are assumed to be normal distributed around their true values (*), and while the coordinates within a point may be correlated, the points are assumed independent.

$$\mathbf{p}_i^O \sim \mathcal{N}(\mathbf{p}_i^{O*}, \mathbf{\Sigma}_i^O) \tag{52}$$
$$\mathbf{p}_i^C \sim \mathcal{N}(\mathbf{p}_i^{C*}, \mathbf{\Sigma}_i^C) \tag{53}$$

For the fourth homogeneous component of $\mathbf{p}_i^O, \mathbf{p}_i^C$ model zero noise. Then, with the right-hand side decorrelation/normalization in the numerator as explained above, we get the information matrix

$$\mathbf{\Omega}_{\mathrm{pp}}^U = \sum_{i=1}^{N} \mathbf{J}_{\mathrm{pp},i}^\top {\mathbf{\Sigma}_{i\blacksquare}^C}^{-1} \mathbf{J}_{\mathrm{pp},i}. \tag{54}$$

The corresponding denominator $\mathbf{\Omega}^L$ is difficult to derive from $\mathrm{Cov}(\mathbf{J}_j)$ as the flattening from the $\bar{}$-operator hinders algebraic manipulation. So, we derive it from the original measurement model (3) and then convert it into $\mathbf{\Omega}^L$ according to the flattening of $\mathbf{T}$ as $\bar{\mathbf{T}}$.

$$\bar{\mathbf{T}}^\top \mathbf{\Omega}^L \bar{\mathbf{T}} = \frac{1}{M} \sum_{j=1}^{M} V(\mathbf{\Sigma}_i^{C\,-\frac{1}{2}} \mathbf{J}_j \bar{\mathbf{T}}) \tag{55}$$
$$= \frac{1}{3N} \sum_{i=1}^{N} \mathrm{tr}\,\mathrm{Cov}\left(\mathbf{\Sigma}_i^{C\,-\frac{1}{2}} (\mathbf{T}\mathbf{p}_i^O - \mathbf{p}_i^C) \right) \tag{56}$$
$$= \frac{1}{3N} \sum_{i=1}^{N} \mathrm{tr}\left(\mathbf{\Sigma}_i^{C\,-\frac{1}{2}} \left(\mathbf{T}\mathbf{\Sigma}_i^O \mathbf{T}^\top + \mathbf{\Sigma}_i^C \right) \mathbf{\Sigma}_i^{C\,-\frac{1}{2}\top} \right) \tag{57}$$
$$= 1 + \frac{1}{3N} \sum_{i=1}^{N} \mathrm{tr}\left(\mathbf{\Sigma}_i^{C\,-\frac{1}{2}} \mathbf{T}\mathbf{\Sigma}_i^O \mathbf{T}^\top \mathbf{\Sigma}_i^{C\,-\frac{1}{2}\top} \right) \tag{58}$$

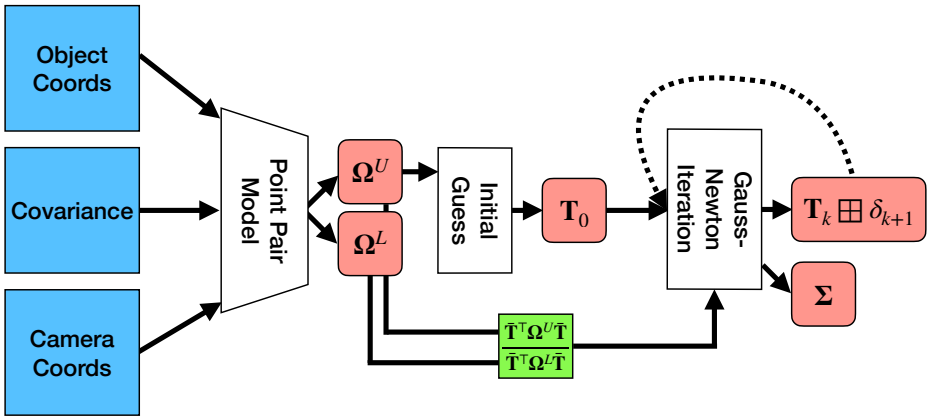

Fig. 2. The point pair model produces two matrices that represent a quadratic fraction. Only the numerator is used in the initial guess, but Gauss-Newton approximates optimization of the entire fraction.

This expression is clearly quadratic in the entries of $\mathbf{T}$ and hence can be expressed by a suitable $\mathbf{\Omega}^L$. Lemma 1 in Appendix B gives the concrete formula using the Kronecker product.

$$\mathbf{\Omega}^L_{\text{pp}} = \begin{pmatrix} \frac{1}{3N} \sum_{i=1}^{N} \mathbf{\Sigma}^{C}_{i\blacksquare}{}^{-1} \otimes \mathbf{\Sigma}^{O}_{i\blacksquare} & \mathbf{0}_{9\times 1} & \mathbf{0}_{9\times 3} \\ \mathbf{0}_{1\times 9} & 1 & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{3\times 9} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 3} \end{pmatrix}. \tag{59}$$

The resulting modification to our system is depicted in Fig. 2.

5 Extensions Beyond Point Pairs

So far, we have treated the optimization problem Eq. (3). This means that only pixels with valid depth data can be used, since that is needed to construct the $\mathbf{p}^C_i$. Although the perspective loss Eq. (4) is not sufficient to disambiguate scale, there are still ways to improve upon the point matching method. For instance, when depth data is partially available, the remaining RGB pixels can still be used to improve the estimate. Other possibilites include multiple cameras (without explicit stereo matching), or fusing the perspective data with a prior [21]. Therefore, it makes sense to define separate sensor models for a perspective and a depth camera.

The steps to obtain an information matrix from RGB and depth data are described in detail in [20]. However, the normalization by a variable variance is new, hence the sensor models are restated here.

5.1 Perspective Sensor Model

In the perspective camera model, each object pixel generates two scalar measurements. The Jacobian corresponding to Eq. (4) is

$$\mathbf{J}_{\text{persp},i} = \mathbf{P}_i \bar{\mathbf{p}}_i^O, \quad \text{with } \mathbf{P}_i = \begin{pmatrix} -f & 0 & u_i - u_0 & 0 \\ 0 & -f & v_i - v_0 & 0 \end{pmatrix} \tag{60}$$

where f, $\binom{u_0}{v_0}$ are focal length and optical center of the camera, and $\binom{u_i}{v_i}$ are the pixel coordinates where $\mathbf{p}_i^O$ was observed. From this, obtaining $\mathbf{\Omega}^U_{\text{persp}}$ is straightforward by Eqs. (41) and (44). As above, $\mathbf{p}_i^O$ is assumed to be normal distributed around its true value with covariance $\mathbf{\Sigma}_i^O$.

The denominator is obtained with the same strategy as in Subsect. 4.4, where $\mathbf{P}_i$ takes the role that $\mathbf{\Sigma}_i^{C\,-\frac{1}{2}}$ had before.

$$\bar{\mathbf{T}}^\top \mathbf{\Omega}^L_{\text{persp}} \bar{\mathbf{T}} = \frac{1}{M} \sum_{j=1}^{M} V(\mathbf{J}_{\text{persp},j} \bar{\mathbf{T}}) \tag{61}$$

$$= \frac{1}{2N} \sum_{i=1}^{N} \operatorname{tr} \operatorname{Cov} \left(\mathbf{P}_i \mathbf{T} \mathbf{p}_i^O \right) \tag{62}$$

$$= \frac{1}{2N} \sum_{i=1}^{N} \operatorname{tr} \left(\mathbf{P}_i \mathbf{T} \mathbf{\Sigma}_i^O \mathbf{T}^\top \mathbf{P}_i^\top \right) \tag{63}$$

Again with Lemma 1 the denominator is

$$\mathbf{\Omega}^L_{\text{persp}} = \begin{pmatrix} \frac{1}{2N} \sum_{i=1}^{N} (\mathbf{P}_i^\top \mathbf{P}_i)_{\blacksquare} \otimes \mathbf{\Sigma}^O_{i\,\blacksquare} & \mathbf{0}_{9\times 1} & \mathbf{0}_{9\times 3} \\ \mathbf{0}_{1\times 9} & 0 & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{3\times 9} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 3} \end{pmatrix}. \tag{64}$$

The fact that $\mathbf{\Omega}^L_{\text{persp},10,10}$ is 0 corresponds to the scale invariance of the perspective residual.

5.2 Depth Sensor Model

For the depth sensor model, each pixel is a single measurement, i. e. $N = M$. Depth $d_i \sim \mathcal{N}(d_i^*, (\sigma_i^d)^2)$ is measured at the pixel where $\mathbf{p}_i^O$ is recognized. The Jacobian corresponding to Eq. (5) is

$$\mathbf{J}_{\text{depth},i} = \mathbf{D}_i \bar{\mathbf{p}}_i^O, \quad \text{with } \mathbf{D}_i = \begin{pmatrix} 0 & 0 & 1 & -d_i \end{pmatrix}. \tag{65}$$

$\mathbf{\Omega}^U_{\text{depth}}$ is calculated from the Jacobian and the right-hand side variance $(\sigma_i^d)^2$. As in the point pair model, the weighting factor $\frac{1}{\sigma_i^d}$ is included in the nominator to make all denominators have 1 as constant part and improve the sum-of-fractions approximation.

The derivation of $\mathbf{\Omega}^L_{\text{depth}}$ is analogous using Lemma 1.

$$\bar{\mathbf{T}}^\top \mathbf{\Omega}^L_{\text{depth}} \bar{\mathbf{T}} = \frac{1}{M} \sum_{j=1}^{N} V\left(\frac{1}{\sigma_j^d} \mathbf{J}_{\text{depth},j} \bar{\mathbf{T}}\right) \tag{66}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \operatorname{tr} \operatorname{Cov} \left(\frac{1}{\sigma_i^d} \left(\mathbf{Z}\mathbf{T}\mathbf{p}_i^O - d_i \right) \right), \text{with } \mathbf{Z} = \begin{pmatrix} 0 & 0 & 1 & 0 \end{pmatrix} \tag{67}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \operatorname{tr} \left(\frac{1}{\sigma_i^d} \left(\mathbf{Z}\mathbf{T}\mathbf{\Sigma}_i^O \mathbf{T}^\top \mathbf{Z}^\top + {\sigma_i^d}^2 \right) \frac{1}{\sigma_i^d} \right) \tag{68}$$

$$= 1 + \frac{1}{N} \sum_{i=1}^{N} \operatorname{tr} \left(\left(\frac{1}{\sigma_i^d} \mathbf{Z}\right) \mathbf{T}\mathbf{\Sigma}_i^O \mathbf{T}^\top \left(\frac{1}{\sigma_i^d} \mathbf{Z}\right)^\top \right) \tag{69}$$

$$\mathbf{\Omega}^L_{\text{depth}} = \begin{pmatrix} \sum_{i=1}^{N} \left({\sigma_i^d}^{-2} \mathbf{Z}^\top \mathbf{Z} \right)_{\blacksquare} \otimes \mathbf{\Sigma}^O_{i\blacksquare} & \mathbf{0}_{9\times 1} & \mathbf{0}_{9\times 3} \\ \mathbf{0}_{1\times 9} & 1 & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{3\times 9} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 3} \end{pmatrix}. \tag{70}$$

5.3 Combining Sensor Models

The residuals in the perspective camera model all have a factor of the pixel's depth, which the depth residuals do not have. This potentially violates the assumption of variances in the same order of magnitude that underlies Eq. (45). Therefore, each sensor's contribution to the loss is normalized separately.

$$\mathcal{L}_{\text{persp+depth}}(\mathbf{T}) = \frac{\bar{\mathbf{T}}^\top \mathbf{\Omega}^U_{\text{persp}} \bar{\mathbf{T}}}{\bar{\mathbf{T}}^\top \mathbf{\Omega}^L_{\text{persp}} \bar{\mathbf{T}}} + \frac{\bar{\mathbf{T}}^\top \mathbf{\Omega}^U_{\text{depth}} \bar{\mathbf{T}}}{\bar{\mathbf{T}}^\top \mathbf{\Omega}^L_{\text{depth}} \bar{\mathbf{T}}} \tag{71}$$

Although this doubles the number of parameters involved in the optimization process, it is still a small constant instead of growing with the number of measurements. Similarly, if multiple cameras were used or other sensors, each would be normalized in a separate summand. This affects the modified Gauss-Newton step Eq. (48), where the Hessians and gradients from all S sensors have to be added.

$$\tilde{\delta}^{\text{tls}}_{k+1} = -\left(\sum_{l=1}^{S} \frac{\mathbf{H}^U_{l,k}}{c^L_{l,k}} \right)^{-1} \left(\sum_{l=1}^{S} \frac{\mathbf{g}^U_{l,k}}{c^L_{l,k}} - \frac{c^U_{l,k}}{{c^L_{l,k}}^2} \mathbf{g}^L_{l,k} \right) \tag{72}$$

The overall relation of the individual parts can be seen in Fig. 3.

For the initial guess, the information from multiple sensors has to be aggregated as well. As stated in Subsect. 4.2, the initial guess does not handle the denominator matrix. Therefore, $\mathbf{\Omega}^U_{\text{persp}}$ and $\mathbf{\Omega}^L_{\text{depth}}$ should not simply be added. At least the factor of each pixel's depth in the residual of the perspective camera model can be compensated. A good guess for this factor is the mean of depth measurements $\bar{d}$, so that the initial guess is made from $\mathbf{\Omega}^U_{\text{persp}} + \bar{d}^2 \mathbf{\Omega}^L_{\text{depth}}$.

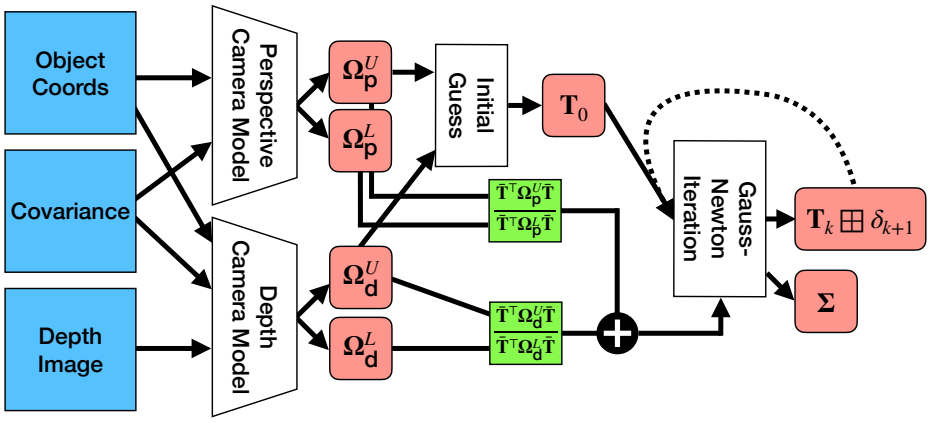

Fig. 3. Both sensor models output matrices representing a quadratic fraction. The initial guess uses both nominators. Gauss-Newton optimizes the sum of each sensor's fractions.

6 Experiments and Results

We evaluate our approach with simulated data. This way, we can focus on the statistical properties of our approach and have control over noise in the data.

Since our algorithm outputs the parameters of a distribution, we want to show that the true pose is consistent with this distribution. A measure for this is the squared Mahalanobis distance, which should be χ^2-distributed with as many degrees of freedom as the tangent space's dimensionality.

$$\chi^2 = (\mathbf{T}^* \boxminus \hat{\mathbf{T}})^\top \hat{\mathbf{\Sigma}}^{-1} (\mathbf{T}^* \boxminus \hat{\mathbf{T}}) \tag{73}$$

In particular its expected value should be 6 for pose and 9 for pose and scale.

6.1 Point Pair Model

Each experiment is a series of 1000 trials in which certain quantities are randomized, i. e. at least the point measurements. For each trial, a set of $N = 1000$ true 3D object points $\mathbf{p}_i^{O*}$ is generated, all coordinates uniformly sampled from $[0, 1]$. The corresponding true camera points $\mathbf{p}_i^{C*}$ are obtained by $\mathbf{T}^* \mathbf{p}_i^{O*}$. The object or camera points, or both, are corrupted by additive Gaussian noise to form the measurements. Finally, a transformation and its covariance is computed using the algorithm from Sect. 3 (LS) or Sect. 4 (TLS). The parameters of all experiments are summarized in Table 1.

In order to demonstrate the relevance of total least squares, we start with a simple experiment: The true object transformation $\mathbf{T}^*$ is set to the identity matrix. We start with isotropic noise, identical for all points (i. e. $\mathbf{\Sigma}^{O/C} = \sigma^2 \mathbf{I}_3$, $\sigma = 0.1$). When the noise is in the camera points only, we are in the classic least-squares case (*PP1*). Therefore, the LS estimator yields consistent results: Over

Table 1. Overview of the experiments. Common parameters are the number of points $N = 1000$, the true 3D object point distribution and the number of trials (1000).

Name	Noise	True Transformation	Algorithm
PP1	$\sigma^C = 0.1\,\mathrm{m}$	$\mathbf{T}^* = \mathbf{I}$	LS
PP2	$\sigma^O = 0.1$	$\mathbf{T}^* = \mathbf{I}$	LS
PP3	$\sigma^O = 0.1$	$\mathbf{T}^* = \mathbf{I}$	TLS
PP4	$\sigma^O \sim U([0.05, 0.1])$ rotated	$\mathbf{R}^* = \mathbf{I}$, $\mathbf{t}^* = (0\,\mathrm{m}, 0\,\mathrm{m}, 1\,\mathrm{m})$	TLS
	$\sigma^C \sim U([0.01\,\mathrm{m}, 0.1\,\mathrm{m}])$ rotated	$\mathbf{s}^* = (0.04\,\mathrm{m}, 0.08\,\mathrm{m}, 0.12\,\mathrm{m})$	
PP5	$\sigma^O \sim U([0.05, 0.1])$ rotated	$\mathbf{R}^* \sim U(SO(3))$	TLS
	$\sigma^C \sim U([0.01\,\mathrm{m}, 0.1\,\mathrm{m}])$ rotated	$\mathbf{t}^* \sim U([-1\,\mathrm{m}, 1\,\mathrm{m}]^3)$	
		$\mathbf{s}^* \sim U([0.02\,\mathrm{m}, 0.3\,\mathrm{m}]^3)$	
PD1	$\sigma^O \sim U([0.05, 0.1])$ rotated	$\mathbf{R}^* = \mathbf{I}$, $\mathbf{t}^* = (0\,\mathrm{m}, 0\,\mathrm{m}, 1\,\mathrm{m})$	TLS
	$\sigma^d = 0.01\,\mathrm{m}$	$\mathbf{s}^* = (0.04\,\mathrm{m}, 0.08\,\mathrm{m}, 0.12\,\mathrm{m})$	
PD2	$\sigma^O \sim U([0.05, 0.1])$ rotated	$\mathbf{R}^* \sim U(SO(3))$	TLS
	$\sigma^d = 0.01\,\mathrm{m}$	$\mathbf{t}^* \sim U([-0.2\,\mathrm{m}, 0.2\,\mathrm{m}]^2 \times [0.2\,\mathrm{m}, 2\,\mathrm{m}])$	
		$\mathbf{s}^* \sim U([0.02\,\mathrm{m}, 0.3\,\mathrm{m}]^3)$	

Table 2. Overview of the results. The $\overline{\chi^2}$ metric would ideally be 9. The scale ratio is the average ratio of predicted and true scales over all axes. The bias in *PP2* is noticable.

Name	$\overline{\chi^2}$	Scale Ratio
PP1	9.1003	1.0005
PP2	332.36	0.8930
PP3	10.272	0.9998
PP4	11.189	1.0022
PP5	13.842	1.0026
PD1	9.3517	0.9997
PD2	9.4690	1.0001

1000 trials, the average squared Mahalanobis distance is close to the expected 9 (cf. Table 2) and its cumulative distribution matches a χ^2 distribution with 9 degrees of freedom (cf. Fig. 4).

However, with noise in the object points, the classic least-squares approach fails (*PP2*). Although the least-squares problem is properly scaled at the true parameters (the averaged $\bar{\mathbf{T}}^{*\top}\boldsymbol{\Omega}\bar{\mathbf{T}}^*$ matches the expectation of 3000 measurements), the averaged squared Mahalanobis distance is much higher than the expected 9 degrees of freedom (cf. Table 2). Looking at the average value of the estimated scales ($\rho(\mathbf{T}_\blacksquare)$), we find that they are systematically lower than their true values. This is the effect explained in Sect. 4, that by making the object smaller, the noise's contribution to the residual is reduced. The other components in the mean of estimated transformations (calculated by fixed-point iteration over the $\boxplus$-manifold [9, §3.5]) are unbiased.

With similarly generated data, the TLS approach from Sect. 4 yields relatively consistent results (cf. *PP3* in Table 2 and Fig. 4).

We can move to more complex settings regarding the true object transformation and noise distributions (*PP4*). We set the true object transformation to a

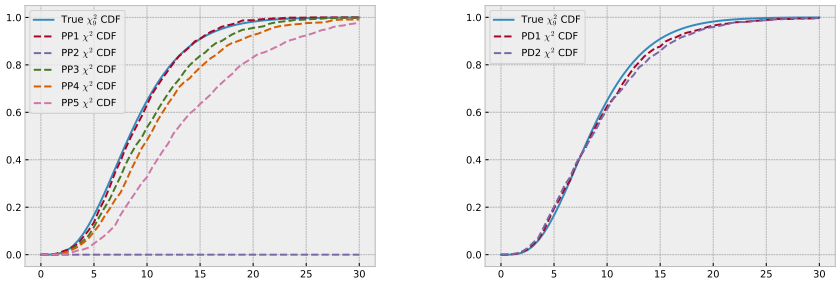

Fig. 4. Cumulative distributions of the Mahalanobis distance from the experiments (left: point pair, right: perspective and depth camera). The curve of *PP1* closely matches the true χ^2 distribution with 9 degrees of freedom. *PP2* (visible at the bottom) is shifted far off to the right due to inconsistency. *PP3*, *PP4*, and *PP5* are a bit off to the right (i. e. the estimation tends to be overconfident), as the involved approximations take increasing effect at those settings.

scale of $(0.04\,\mathrm{m}, 0.08\,\mathrm{m}, 0.12\,\mathrm{m})$ and 1 m ahead of the camera, and corrupt both object and camera points by noise. The noise is drawn for each point independently from a 3D covariance matrix with random rotation and standard deviations drawn uniformly from $[0.05, 0.1]$ for the object points and $[0.01\,\mathrm{m}, 0.1\,\mathrm{m}]$ for the camera points. In this setting, we still get an acceptable average Mahalanobis distance of ≈ 11.

As final validation, we randomize the ground truth transformation (*PP5*). The rotation is chosen uniformly over $SO(3)$ [2], translation is uniformly from $[-1\,\mathrm{m}, 1\,\mathrm{m}]$ and scales from the range $[0.02\,\mathrm{m}, 0.3\,\mathrm{m}]$, independently per axis. The latter may be a typical range for household objects to manipulate with robots. At this point, the approximations start to show (cf. Table 2). In particular, the scales of the axes of an object may differ by an order of magnitude in this experiment. This transfers to the variances of the $x/y/z$ components which therefore make the approximation Eq. (46) worse. Still, the results are far better than even the simple example without total least squares.

In summary, as long as the scales of the axes of the object are not orders of magnitude apart and the noise level is relatively low, the transformation can be estimated sufficiently consistent.

6.2 Perspective and Depth Camera Model

We now evaluate transformation estimation using the perspective and depth camera model from Sect. 5. The process is similar to the point pair experiments, where the main difference is how the true object coordinates are obtained. For each trial, a 32×32 depth image is generated by sampling uniformly within an object diameter of the object's true z-translation, i. e. the true object is a random point cloud. This depth image is then transformed into camera points using a

pixel coordinate grid and a camera matrix. The focal length is chosen depending on the object diameter and distance to the camera, such that the resulting object coordinates are approximately normalized. Both the depth image and the object points are then independently corrupted by noise. Experiment *PD1* uses similar settings to *PP4*, i. e. a fixed object transformation (cf. Table 1). For the random object transformations in *PD2*, the true translation is sampled from a different range than in *PP5*. In particular, the true z coordinate is always positive and x/y are chosen from a narrow range, such that the object center is not too far outside the "visible" patch. The results can be seen in Fig. 4 and Table 2. In both experiments, transformations can be estimated with consistent uncertainty information.

7 Conclusion and Future Work

Many object pose estimation algorithms match a set of point pairs, either 3D-3D or 3D-2D, with least-squares. We showed how this process can be extended to also estimate object-axes scales, by optimizing over the scaled $SO(3)$ manifold using a $\boxplus$-operator that encapsulates the manifold.

However, a problem comes up that's hidden in the original pose formulation already, but does not have an effect there: There is noise in both sides of the match, one of them is multiplied with the scale. Ignoring this creates a bias towards smaller scales, because that also scales the noise down. This bias is not reflected by the uncertainty estimate rendering it inconsistent.

The paper discusses a solution for this problem using total least squares by normalizing every measurement with the variance at the current parameters. Normally, this approach would not allow to combine all measurements into a fixed size representation. However, we propose an approximation to achieve this, which in the experiments proved relatively consistent.

In future work, our fusion stage will be combined with a neural network to predict object coordinates and uncertainties to construct a complete 9D pose estimation system. Further lines of research go for fusion with a prior, other sensors or fusion over time, where a system might remember the objects scale, even if the pose has changed.

Acknowledgments. The research reported in this paper has been supported by the German Research Foundation DFG, as a part of Collaborative Research Center (Sonderforschungsbereich) 1320 Project-ID 329551904 EASE - Everyday Activity Science and Engineering, University of Bremen (http://www.ease-crc.org/). The research was conducted in subproject R02 Multi-cue perception supported by background knowledge.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

A Proof of the $\boxplus$-Axioms for the Scaled $SO(3)$

The first axiom requires surjectivity of $\boxplus$ (all states can reach any other state by a single $\boxplus$-step).

$$\forall \mathbf{Q}_1, \mathbf{Q}_2 \in \mathcal{Q}\colon \mathbf{Q}_1 \boxplus (\mathbf{Q}_2 \boxminus \mathbf{Q}_1) = \mathbf{Q}_2 \tag{74}$$

Proof.

$$\begin{aligned}
&\mathbf{Q}_1 \boxplus (\mathbf{Q}_2 \boxminus \mathbf{Q}_1)\\
=&\exp_\times(\mathbf{Q}_2 \boxminus \mathbf{Q}_1)_\times \mathbf{Q}_1 \exp_\circ(\mathbf{Q}_2 \boxminus \mathbf{Q}_1)_\circ\\
=&\exp_\times \log_\times(\rho(\mathbf{Q}_2)\rho(\mathbf{Q}_1)^{-1})\mathbf{Q}_1 \exp_\circ \log_\circ(\sigma(\mathbf{Q}_1)^{-1}\sigma(\mathbf{Q}_2))\\
=&\mathbf{Q}_2\sigma(\mathbf{Q}_2)^{-1}\sigma(\mathbf{Q}_1)\mathbf{Q}_1^{-1}\mathbf{Q}_1\sigma(\mathbf{Q}_1)^{-1}\sigma(\mathbf{Q}_2)\\
=&\mathbf{Q}_2
\end{aligned}$$

The second axiom requires local injectivity of $\boxplus$ (within some neighborhood V of a state, the tangent vector transferring to another state is unique).

$$\forall \mathbf{Q} \in \mathcal{Q}, \delta \in V\colon (\mathbf{Q} \boxplus \delta) \boxminus \mathbf{Q} = \delta \tag{75}$$

For this, we need the following lemma

$$\sigma(\mathbf{Q} \boxplus \delta) = \sigma(\mathbf{Q}) \exp_\circ \delta_\circ \tag{76}$$

which is proven by using the diagonality of $\exp_\circ$ and $\mathbf{Q}^\top\mathbf{Q}$ and orthogonality of $\exp_\times$:

$$\begin{aligned}
&\sigma(\mathbf{Q} \boxplus \delta)\\
=&\sigma(\exp_\times \delta_\times \mathbf{Q} \exp_\circ \delta_\circ)\\
=&\sqrt{(\exp_\circ \delta_\circ)^\top \mathbf{Q}^\top (\exp_\times \delta_\times)^\top \exp_\times \delta_\times \mathbf{Q} \exp_\circ \delta_\circ}\\
=&\sqrt{\exp_\circ \delta_\circ \mathbf{Q}^\top \mathbf{Q} \exp_\circ \delta_\circ}\\
=&\sqrt{\exp_\circ \delta_\circ}\sigma(\mathbf{Q})\sqrt{\exp_\circ \delta_\circ}\\
=&\sigma(\mathbf{Q}) \exp_\circ \delta_\circ
\end{aligned}$$

Proof.

$$\begin{aligned}
&(\mathbf{Q} \boxplus \delta) \boxminus \mathbf{Q}\\
=&(\log_\times(\rho(\mathbf{Q} \boxplus \delta)\rho(\mathbf{Q})^{-1}), \log_\circ(\sigma(\mathbf{Q})^{-1}\sigma(\mathbf{Q} \boxplus \delta)))^\top\\
=&(\log_\times(\rho(\mathbf{Q} \boxplus \delta)\rho(\mathbf{Q})^{-1}), \log_\circ \exp_\circ(\delta_\circ))^\top\\
=&(\log_\times(\rho(\mathbf{Q} \boxplus \delta)\rho(\mathbf{Q})^{-1}), \delta_\circ)^\top\\
=&(\log_\times((\mathbf{Q} \boxplus \delta)\sigma(\mathbf{Q} \boxplus \delta)^{-1}\sigma(\mathbf{Q})\mathbf{Q}^{-1}), \delta_\circ)^\top\\
=&(\log_\times((\mathbf{Q} \boxplus \delta)(\exp_\circ \delta_\circ)^{-1}\sigma(\mathbf{Q})^{-1}\sigma(\mathbf{Q})\mathbf{Q}^{-1}), \delta_\circ)^\top
\end{aligned}$$

$$
\begin{aligned}
&= (\log_\times(\exp_\times \delta_\times \mathbf{Q} \exp_\circ \delta_\circ (\exp_\circ \delta_\circ)^{-1}\mathbf{Q}^{-1}), \delta_\circ)^\top \\
&= (\log_\times \exp_\times \delta_\times, \delta_\circ)^\top \\
&= (\delta_\times, \delta_\circ)^\top \\
&= \delta
\end{aligned}
$$

The cancellation of $\log_\times$ against $\exp_\times$ in the penultimate step is what requires V to be restricted to angles up to π.

The third axiom requires 1-Lipschitzness of the family of functions $f_\mathbf{Q} \colon \mathbb{R}^6 \to \mathcal{Q};\quad \delta \mapsto \mathbf{Q} \boxplus \delta$:

$$
\forall \mathbf{Q} \in \mathcal{Q}, \delta_1, \delta_2 \in \mathbb{R}^9 \colon \|(\mathbf{Q} \boxplus \delta_1) \boxminus (\mathbf{Q} \boxplus \delta_2)\| \leq \|\delta_1 - \delta_2\| \tag{77}
$$

Proof.

$$
\begin{aligned}
&\|(\mathbf{Q} \boxplus \delta_1) \boxminus (\mathbf{Q} \boxplus \delta_2)\|^2 \\
=&\| \log_\times(\rho(\mathbf{Q} \boxplus \delta_1)\rho(\mathbf{Q} \boxplus \delta_2)^{-1})\|^2 \\
+&\| \log_\circ(\sigma(\mathbf{Q} \boxplus \delta_2)^{-1}\sigma(\mathbf{Q} \boxplus \delta_1))\|^2 \\
=&\| \log_\times((\mathbf{Q} \boxplus \delta_1)\sigma(\mathbf{Q} \boxplus \delta_1)^{-1}\sigma(\mathbf{Q} \boxplus \delta_2)(\mathbf{Q} \boxplus \delta_2)^{-1})\|^2 \\
+&\| \log_\circ((\exp_\circ \delta_{2,\circ})^{-1}\sigma(\mathbf{Q})^{-1}\sigma(\mathbf{Q}) \exp_\circ \delta_{1,\circ}))\|^2 \\
=&\| \log_\times((\mathbf{Q} \boxplus \delta_1)(\exp_\circ \delta_1)^{-1}\sigma(\mathbf{Q})^{-1}\sigma(\mathbf{Q})(\exp_\circ \delta_2)(\mathbf{Q} \boxplus \delta_2)^{-1})\|^2 \\
+&\| \log_\circ((\exp_\circ -\delta_{2,\circ}) \exp_\circ \delta_{1,\circ})\|^2 \\
=&\| \log_\times((\mathbf{Q} \boxplus \delta_1)(\exp_\circ \delta_1)^{-1}(\exp_\circ \delta_2)(\mathbf{Q} \boxplus \delta_2)^{-1})\|^2 \\
+&\| \log_\circ \exp_\circ(\delta_{1,\circ} - \delta_{2,\circ})\|^2 \\
=&\| \log_\times(\exp_\times \delta_1 \mathbf{Q} \exp_\circ \delta_1 (\exp_\circ \delta_1)^{-1}(\exp_\circ \delta_2)(\exp_\circ \delta_2)^{-1}\mathbf{Q}^{-1}(\exp_\times \delta_2)^{-1})\|^2 \\
+&\|\delta_{1,\circ} - \delta_{2,\circ}\|^2 \\
=&\| \log_\times(\exp_\times \delta_1 (\exp_\times \delta_2)^{-1})\|^2 + \|\delta_{1,\circ} - \delta_{2,\circ}\|^2 \\
\leq&\|\delta_{1,\times} - \delta_{2,\times}\|^2 + \|\delta_{1,\circ} - \delta_{2,\circ}\|^2 \\
=&\|\delta_1 - \delta_2\|^2
\end{aligned}
$$

The $\leq$ in the penultimate step is justified by the fact that $SO(3)$ is a $\boxplus$-manifold [9].

In conclusion, the scaled $SO(3)$ with the given operators is a $\boxplus$-manifold.

B Flattening certain Expressions Involving T

To derive the denominator expression in the total least squares approach, in all sensor models we needed to express $\operatorname{tr} \mathbf{W}\mathbf{T}\mathbf{\Sigma}\mathbf{T}^\top\mathbf{W}^\top$ as $\bar{\mathbf{T}}^\top\mathbf{\Omega}^L\bar{\mathbf{T}}$, i. e. flatten it into our fixed sized representation.

Lemma 1. *Let* $\mathbf{W} \in \mathbb{R}^{d\times 4}$, $\mathbf{T} \in \mathbb{R}^{4\times 4}$ *with* $\mathbf{T}_{4\bullet} = \begin{pmatrix}0 & 0 & 0 & 1\end{pmatrix}$, $\mathbf{\Sigma} \in \mathbb{R}^{4\times 4}$, *symmetric positive semidefinite with* $\mathbf{\Sigma}_{44} = 0$. *Then*

$$\operatorname{tr}\left(\mathbf{W}\mathbf{T}\mathbf{\Sigma}\mathbf{T}^\top\mathbf{W}^\top\right) = \bar{\mathbf{T}}^\top\mathbf{\Omega}^L\bar{\mathbf{T}}, \quad \textit{with} \quad \begin{pmatrix}(\mathbf{W}^\top\mathbf{W})_{\blacksquare} \otimes \mathbf{\Sigma}_{\blacksquare} & \mathbf{0}_{9\times 1} & \mathbf{0}_{9\times 3}\\ \mathbf{0}_{1\times 9} & 0 & \mathbf{0}_{1\times 3}\\ \mathbf{0}_{3\times 9} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 3}\end{pmatrix}. \tag{78}$$

Proof.

$$\bar{\mathbf{T}}^\top\mathbf{\Omega}^L\bar{\mathbf{T}} = \operatorname{tr}\left(\mathbf{W}\mathbf{T}\mathbf{\Sigma}\mathbf{T}^\top\mathbf{W}^\top\right) = \operatorname{tr}\left(\mathbf{T}\mathbf{\Sigma}\mathbf{T}^\top\mathbf{W}^\top\mathbf{W}\right) \tag{79}$$

$$= \sum_{k,l,m,n=1}^{4,4,4,4} \mathbf{T}_{kl}\mathbf{\Sigma}_{lm}\mathbf{T}_{nm}(\mathbf{W}^\top\mathbf{W})_{nk} \tag{80}$$

$$= \sum_{k,l,m,n=1}^{3,3,3,3} \mathbf{T}_{kl}\mathbf{\Sigma}_{lm}\mathbf{T}_{nm}(\mathbf{W}^\top\mathbf{W})_{nk} \tag{81}$$

The last step holds, because from positive definiteness, the whole fourth row and column of $\mathbf{\Sigma}$ is zero, so $l = 4$ and $m = 4$ can be omitted. The same holds for $k = 4$ and $n = 4$, because $\mathbf{T}_{kl}$ or $\mathbf{T}_{nm}$ are zero.

We consider the coefficients for different products of $\mathbf{T}$ elements. There is no constant and no linear term. Each quadratic term $\mathbf{T}_{kl}\mathbf{T}_{nm}$ is multiplied with $\mathbf{\Sigma}_{lm}(\mathbf{W}^\top\mathbf{W})_{kn}$, using symmetry of $\mathbf{W}^\top\mathbf{W}$.

By flattening $\mathbf{T}$ into $\bar{\mathbf{T}}$, its row-indices k and n have stride 3 in the rows and columns of $\mathbf{\Omega}^L$ respectively. Both address an element of $\mathbf{\Sigma}_i^O$. The column-indices l and m have stride 1 and address an element of $\mathbf{W}^\top\mathbf{W}$. The two elements are multiplied. This is conveniently expressed with a Kronecker product, leading to the formula (78).

References

1. Arun, K.S.: A unitarily constrained total least squares problem in signal processing. SIAM Journal on Matrix Analysis and Applications **13**(3), 729–745 (Jul 1992)
2. Arvo, J.: Fast random rotation matrices. In: Kirk, D. (ed.) Graphics Gems III, pp. 117–120. Academic Press (1991)
3. Brachmann, E., Michel, F., Krull, A., Yang, M.Y., Gumhold, S., Rother, C.: Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image. In: 2016 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3364–3372 (2016)
4. Crassidis, J.L., Cheng, Y.: Error-covariance analysis of the total least squares problem. AIAA Journal of Guidance, Control, and Dynamics **37**(4), 1053–1063 (Jul 2014)
5. Everson, R.: Orthogonal, but not orthonormal, Procrustes problems. Tech. rep., City University of New York (Mar 1997)
6. Fu, Y., Wang, X.: Category-level 6D object pose estimation in the wild: A semi-supervised learning approach and a new dataset. In: 36th Conference on Neural Information Processing Systems (NeurIPS) (2022)

7. Golub, G.H., Van Loan, C.F.: An analysis of the total least squares problem. SIAM Journal on Numerical Analysis **17**(6), 883–893 (Dec 1980)
8. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge University Press (2003)
9. Hertzberg, C., Wagner, R., Frese, U., Schröder, L.: Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds. Information Fusion **14**(1), 57–77 (Jan 2013)
10. Hodaň, T., Sundermeyer, M., Labbé, Y., Nguyen, V.N., Wang, G., Brachmann, E., Drost, B., Lepetit, V., Rother, C., Matas, J.: BOP challenge 2023 on detection, segmentation and pose estimation of seen and unseen rigid objects. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops CV4MR Workshop (2024)
11. Irshad, M.Z., Kollar, T., Laskey, M., Stone, K., Kira, Z.: CenterSnap: Single-shot multi-object 3D shape reconstruction and categorical 6D pose and size estimation. In: IEEE International Conference on Robotics and Automation (ICRA) (2022)
12. Irshad, M.Z., Zakharov, S., Ambrus, R., Kollar, T., Kira, Z., Gaidon, A.: ShAPO: Implicit representations for multi object shape appearance and pose optimization. In: European Conference on Computer Vision (ECCV) (2022)
13. Kabsch, W.: A solution for the best rotation to relate two sets of vectors. Acta Crystallographica **A32**, 922–923 (Sep 1976)
14. Kabsch, W.: A discussion of the solution for the best rotation to relate two sets of vectors. Acta Crystallographica **A34**, 827–828 (Jul 1978)
15. Kneip, L., Furgale, P.: OpenGV: A unified and generalized approach to real-time calibrated geometric vision. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 1–8 (2014)
16. Kneip, L., Furgale, P., Siegwart, R.: Using multi-camera systems in robotics: Efficient solutions to the NPnP problem. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 3770–3776 (2013)
17. Lee, T., Lee, B.U., Kim, M., Kweon, I.S.: Category-level metric scale object shape and pose estimation. IEEE Robotics and Automation Letters **6**(4), 8575–8582 (Oct 2021)
18. Lepetit, V., Moreno-Noguer, F., Fua, P.: EPnP: An accurate O(n) solution to the PnP problem. International Journal of Computer Vision **81**(2), 155–166 (Feb 2009)
19. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes. Cambridge University Press, 3 edn. (2007)
20. Richter-Klug, J., Frese, U.: Towards meaningful uncertainty information for CNN based 6D pose estimates. In: Proceedings of the 12th International Conference on Computer Vision Systems. pp. 408–422 (2019)
21. Richter-Klug, J., Mania, P., Kazhoyan, G., Beetz, M., Frese, U.: Improving object pose estimation by fusion with a multimodal prior – utilizing uncertainty-based CNN pipelines for robotics. IEEE Robotics and Automation Letters **7**(2), 2282–2288 (Apr 2022)
22. Schönemann, P.H.: A generalized solution of the orthogonal Procrustes problem. Psychometrika **31**(1), 1–10 (Mar 1966)
23. Umeyama, S.: Least-squares estimation of transformation parameters between two point patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence **13**(4), 376–380 (Apr 1991)
24. Wahba, G.: A least squares estimate of satellite attitude. SIAM Review **7**(3), 409 (Jul 1965)

25. Wang, H., Sridhar, S., Huang, J., Valentin, J., Song, S., Guibas, L.J.: Normalized object coordinate space for category-level 6D object pose and size estimation. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2637–2646 (2019)
26. Wei, J., Song, X., Liu, W., Kneip, L., Li, H., Ji, P.: RGB-based category-level object pose estimation via decoupled metric scale recovery. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 2036–2042 (2024)
27. Wursthorn, K., Hillemann, M., Ulrich, M.: Uncertainty quantification with deep ensembles for 6D object pose estimation. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. vol. X-2-2024 (2024)
28. Zheng, Y., Kuang, Y., Sugimoto, S., Astrom, K., Okutomi, M.: Revisiting the PnP problem: A fast, general and optimal solution. In: IEEE International Conference on Computer Vision. pp. 2344–2351 (2013)