gDLS*: Generalized Pose-and-Scale Estimation Given Scale and Gravity Priors

Victor Fragoso
Microsoft
victor.fragoso@microsoft.com

Joseph DeGol
Microsoft
joseph.degol@microsoft.com

Gang Hua[1]
Wormpex AI
ganghua@gmail.com

Abstract

Many real-world applications in augmented reality (AR), 3D mapping, and robotics require both fast and accurate estimation of camera poses and scales from multiple images captured by multiple cameras or a single moving camera. Achieving high speed and maintaining high accuracy in a pose-and-scale estimator are often conflicting goals. To simultaneously achieve both, we exploit a priori *knowledge about the solution space. We present gDLS*, a generalized-camera-model pose-and-scale estimator that utilizes rotation and scale priors. gDLS* allows an application to flexibly weigh the contribution of each prior, which is important since priors often come from noisy sensors. Compared to state-of-the-art generalized-pose-and-scale estimators (*e.g. *gDLS), our experiments on both synthetic and real data consistently demonstrate that gDLS* accelerates the estimation process and improves scale and pose accuracy.*

1. Introduction

Estimating the pose and scale from multiple images taken from multiple cameras or multiple images from one moving camera (*e.g.*, a SLAM [6, 18, 19, 34, 46] trajectory) is an essential step in many augmented reality (AR) [32, 39, 41, 49, 47], 3D mapping [3, 11, 31, 38, 42, 44], and robotics applications [15, 20, 21, 30, 50, 54]. Consider hologram sharing services (*e.g.*, Azure Spatial Anchors [1]) as an example. These services have a reference map and need to localize query images accurately (so that holograms are positioned correctly) and quickly (to maintain a nice user experience). However, as Figure 1 shows, current methods leave room for improvement in terms of both accuracy and processing time. In this work, we propose gDLS*, a multi-camera pose-and-scale estimator that exploits scale and gravity priors to improve accuracy and speed. Despite using additional information, gDLS* computes its parameters with linear complexity in the number of points and multiple optimal solutions in a single shot, avoiding iterative optimization procedures.

[1] This work was done while at Microsoft.
[2] josephdegol.com/pages/GDLSStar_CVPR20.html

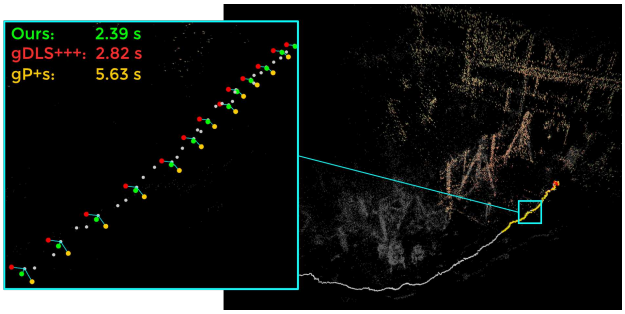


Figure 1. Estimating the pose and scale accurately and quickly is essential in many applications in AR, 3D mapping, and robotics. We introduce gDLS*, a pose-and-scale estimator that exploits scale and/or gravity priors to improve accuracy and speed. Compared to state-of-the-art estimators (*e.g.*, gDLS+++ [44] and gP+s [48]), gDLS* achieves more accurate estimates in less time when registering a set of cameras to an existing 3D reconstruction. The right image above shows an existing 3D reconstruction (gray) with aligned cameras and points for gDLS* (green), gDLS+++ (red), and gP+s (orange). The left image shows a zoomed view of the positions of the aligned cameras. The white points are the expected camera positions and the green/red/orange points are the estimate positions for each method. The cyan line between indicates a match, where longer lines indicate more error.

Using single camera pose estimators (*e.g.*, [2, 7, 16, 23, 26, 29, 33, 53]) to develop a multi-camera pose-and-scale estimator is cumbersome, and their estimates tend to be inaccurate [22, 43]. Instead, many multi-camera pose-and-scale estimators [22, 43, 44] use the generalized camera model [12, 37] to elegantly treat the collection of cameras as one generalized camera, yielding accuracy improvements. Despite their improvements, these estimators often produce erroneous results due to noisy input data and numerical instabilities in their underlying polynomial solvers.

Given the need of accurate pose and scale estimates by many applications in AR, 3D mapping, and robotics, some algorithms [41, 42, 52] exploit inertial measurements (*e.g.*, gravity directions). Most of these approaches assume that the gravity or down directions are reliable, and include this extra knowledge as part of their mathematical derivation to simplify the problem. However, the gravity directions can still be noisy due to the nature of these sensors

and can affect the accuracy of the estimates. In contrast, gDLS* adopts a generalized-camera model with regularizers that encode scale and rotation priors (*e.g.*, gravity direction). These regularizers allow a user to independently control the contribution of each individual prior, which is beneficial to reduce the effect of noise present in each prior.

We show using synthetic data that gDLS* is numerically stable and resilient to noise. We demonstrate this by (1) varying pixel noise and sample size and showing that gDLS* estimates transformations with errors that are no worse than current estimators; and (2) varying the noise in the scale and gravity priors and showing that gDLS* maintains accuracy and speed. We then use real data (*i.e.* [10, 40]) to evaluate gDLS* when registering a set of cameras to an existing 3D reconstruction. Our extensive experiments show that gDLS* is significantly faster and slightly more accurate than current pose-and-scale estimators (*i.e.*, [22, 43, 44, 48]). Moreover, the experiments show that a rotation prior based on gravity directions improves rotation and translation estimates while achieving significant speed-ups. On the other hand, a scale prior mainly improves scale estimates while modestly enhancing translation estimates and speed.

In summary, the **contributions** of this work are (1) gDLS*, a novel and generalized formulation of gDLS [43] that includes scale and gravity priors that computes its parameters with an $\mathcal{O}(n)$ complexity; (2) a novel evaluation protocol for pose-and-scale estimators that reports rotation, translation, and scale errors; and (3) extensive experimental results showing that gDLS* consistently improves pose accuracy in less time.

2. Related Work

Estimating the position and orientation of a camera is crucial for many applications because they need to accurately register computer-generated content into the real-world, localize an agent (*e.g.*, a visually impaired person) within an environment, and autonomously navigate (*e.g.*, self-driving cars). While these applications use camera pose estimators to operate, most of the estimators have focused on localizing single cameras. Although these estimators [2, 8, 23, 25, 26, 29, 51, 53] have achieved impressive performance and accuracy, many applications [21] have started to adopt multi-camera systems. This is because a multi-camera system can provide additional information that allows an application to estimate its pose more accurately. For this reason, this section reviews existing work on multi-camera pose and pose-and-scale estimators.

2.1. Multi-Camera Pose Estimators

Chen and Chang [4] and Nister and Stewenius [35] proposed gP3P, a minimal estimator which requires three 2D-3D correspondences to estimate the pose of a multi-camera system. gP3P computes up to eight solutions by finding the intersections of a circle and a ruled quartic surface. Lee *et al.* [14] also introduced a minimal estimator that utilizes Plücker lines to estimate the depth of each point. Subsequently, it estimates the position of each point w.r.t. to the frame of reference of the multi-camera system. Then it estimates the absolute pose of the multi-camera system.

Unlike previous minimal solvers, Kneip *et al.* [22] introduced UPnP, an efficient minimal and non-minimal pose estimator derived from a least-squares reprojection-error-based cost function. Inspired by DLS [16], UPnP reformulates the cost function as one depending only on a unit-norm quaternion. UPnP finds the optimal rotation by solving a polynomial system that encodes the vanishing of the cost gradient at the optimal unit-norm quaternion via a Gröbner-basis solver [24, 28].

Table 1. gDLS* compares favorably to existing state-of-the-art pose-and-scale estimators because it maintains all the properties of other estimators while also being the only estimator that enables the use of gravity and scale priors.

Reference	gP+s [48]	gDLS [43]	gDLS+++ [44]	UPnP [22]	Ours -
Year	2014	2014	2016	2014	2019
Generalized Camera	✓	✓	✓	✓	✓
Geometric Optimality	✓	✓	✓	✓	✓
Linear Complexity		✓	✓	✓	✓
Multiple Solutions	✓	✓	✓	✓	✓
Similarity Transformation	✓	✓	✓		✓
Singularity-Free Rotation	✓		✓	✓	✓
Gravity Prior					✓
Scale Prior					✓

2.2. Pose-and-Scale Estimators

Different from multi-camera pose estimators, pose-and-scale estimators compute the pose of a multi-camera system and a scale value; this value scales the positions of the cameras in order to align a 3D representation into the frame of reference of the multi-camera system more accurately.

Ventura *et al.* [48] proposed gP+s, a minimal pose-and-scale estimator that requires four 2D-3D correspondences and a Gröbner basis polynomial solver [24, 28]. However, gP+s can also work with more than four points. Kukelova *et al.* [27] introduced another minimal pose-and-scale estimator that avoids using a Gröbner basis polynomial solver, leading to impressive speed-ups but a decrease in accuracy.

Unlike previous estimators, Sweeney *et al.* [43] presented gDLS, an estimator derived from a least-squares reprojection-error cost function. gDLS derives a cost function that depends only on a rotation matrix. Inspired by DLS [16], gDLS solves a polynomial system that encodes the vanishing of the cost gradient at the optimal Cayley-Gibbs-Rodrigues angle-axis vector using the DLS polynomial solver (a Macaulay solver). Unfortunately, its Macaulay solver can be slow since it requires obtaining the

eigenvectors of a 27×27 action matrix. To alleviate this issue, Sweeney *et al*. [44] introduced gDLS+++, a gDLS-based estimator using a unit-norm quaternion. Thanks to the rotation representation of gDLS+++, it can use the efficient UPnP polynomial solver.

Different from previous methods, gDLS* is one of the first estimators to incorporate scale and rotation priors. As we show in Section 4, these priors improve both speed and accuracy. Moreover, gDLS* maintains many of the desirable properties of current solvers: (1) uses a generalized camera model which elegantly simplifies the formulation; (2) computes multiple optimal solutions in a single shot, avoiding iterative optimization procedures; (3) scales linearly when building its parameters; and (4) uses a singularity-free rotation representation. See Table 1 for a brief comparison of estimator properties.

3. Pose-and-Scale Estimation using Priors

The goal of gDLS* is to provide hints about the scale and rotation parameters of the similarity transformation using a generalized pose-and-scale estimator (*e.g.*, gDLS [43]). Thanks to the prevalence of inertial sensors in mobile devices, these priors are readily available. For instance, a rotation prior can be obtained from the gravity direction using measurements from inertial sensors, and a scale prior can be obtained from the IMU [36], GPS, or known landmark sizes [5].

One of the design considerations of gDLS* is the ability to control the contribution of each of the priors independently. This allows the user to either disable or enable each of the priors. When enabling the priors, gDLS* allows a user to set a weight for each prior to control their confidence. In Section 4, we test a range of weights, but we plan to explore in future work how to set these weights automatically using the variance of the noise of the sensors. Because gDLS* is based on the pose-and-scale formulations of gDLS [43, 44], we first describe the pose-and-scale formulation and then present our modifications that enable the use of scale and rotation priors.

3.1. gDLS - A Pose-and-Scale Estimator Review

Given n 2D-3D correspondences, gDLS computes the scale and pose of a non-central camera by minimizing the following least-squares cost function:

$$J(R, \mathbf{t}, s, \boldsymbol{\alpha}) = \sum_{i=1}^{n} \| \alpha_i \mathbf{r}_i - (R\mathbf{p}_i + \mathbf{t} - s\mathbf{c}_i) \|^2, \quad (1)$$

where $\mathbf{r}_i$ is a unit-vector indicating the direction from the position of the camera $\mathbf{c}_i$ to a 3D point $\mathbf{p}_i$; α_i is the depth of the point $\mathbf{p}_i$ with respect to the camera position $\mathbf{c}_i$; $\boldsymbol{\alpha}$ is a vector holding the depths; $R \in \mathrm{SO}(3)$ is a rotation matrix; $\mathbf{t} \in \mathbb{R}^3$ is a translation vector; and $s \in \mathbb{R}$ is the scale; see Fig. 2 for a visual representation of Eq. (1).

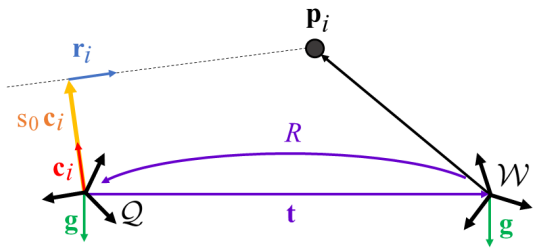


Figure 2. Estimating the pose of a multi-camera system $\mathcal{Q}$ requires the estimation of R and $\mathbf{t}$, while the scale s adjusts the camera positions $\mathbf{c}_i$ so that $\mathcal{W}$ and $\mathcal{Q}$ use the same metric scale. gDLS* can use the gravity directions $\mathbf{g}$ to impose a rotation prior and a scale-prior s_0 to place the cameras at the right scale.

The pose-and-scale formulation shown in Eq. (1) accumulates the errors between the transformed i-th 3D point $(R\mathbf{p}_i + \mathbf{t} - s\mathbf{c}_i)$ and the same point described with respect to the camera $\alpha_i \mathbf{r}_i$. The rotation R, the translation $\mathbf{t}$, and $s\mathbf{c}_i$ transform a 3D point from a world coordinate system to the coordinate system of a generalized camera.

To find the minimizer $(R^\star, \mathbf{t}^\star, s^\star, \boldsymbol{\alpha}^\star)$, gDLS [43] first rewrites $J(R, \mathbf{t}, s, \boldsymbol{\alpha})$ as a function that only depends on the rotation matrix. As Hesch and Roumeliotis [16] and Sweeney *et al*. [43, 44] demonstrated, the translation $\mathbf{t}$, scale s, and depth α_i can be written as a linear function of the rotation matrix R. Thus, it is possible to re-write the pose-and-scale least-squares cost formulation as follows:

$$\begin{aligned} J(R) &= \sum_{i=1}^{n} \| \alpha_i(R)\mathbf{r}_i - (R\mathbf{p}_i + \mathbf{t}(R) - s(R)\mathbf{c}_i) \|^2 \\ &= \mathrm{vec}(R)^\intercal M \mathrm{vec}(R), \end{aligned} \quad (2)$$

where $\mathrm{vec}(R)$ is a vectorized form of the rotation matrix, and M is a square matrix capturing the constraints from the input 2D-3D correspondences; the dimensions of M depend on the vectorization and representation of $\mathrm{vec}(R)$.

Given the cost function $J(R)$, gDLS finds the optimal rotation $R^\star$ by solving a polynomial system representing the constraint that the gradient $\nabla_{\mathbf{q}} J(R^\star) = 0$ is null with respect to the rotation parameters $\mathbf{q}$, and rotation-parameter constraints (*e.g.*, ensuring a unit-norm quaternion).

3.2. Incorporating Priors via Regularizers

In order to impose scale and rotation priors to Eq. (1), gDLS* uses regularizers. Adding these regularizers leads to the following least-squares cost function:

$$J' = J(R, \mathbf{t}, s, \boldsymbol{\alpha}) + \lambda_s \left(s_0 - s\right)^2 + \lambda_g \| \mathbf{g}_{\mathcal{Q}} \times R\mathbf{g}_{\mathcal{W}} \|^2, \quad (3)$$

where s_0 is the scale prior; $\mathbf{g}_{\mathcal{Q}}$ and $\mathbf{g}_{\mathcal{W}}$ are the gravity directions of the multi-camera setting and world, respectively; the symbol $\times$ represents the cross-product operator; and λ_s and λ_g are weights controlling the contribution of the scale

and rotation priors, respectively. These weights (*i.e.*, λ_s and λ_g) must be greater than or equal to zero.

The scale regularizer $\lambda_s \left(s_0 - s\right)^2$ imposes a penalty by deviating from the scale prior s_0. On the other hand, the rotation prior $\lambda_g \|\mathbf{g}_{\mathcal{Q}} \times R\mathbf{g}_{\mathcal{W}}\|^2$ imposes a misalignment penalty between the transformed world gravity direction $R\mathbf{g}_{\mathcal{W}}$ and the query gravity direction $\mathbf{g}_{\mathcal{Q}}$.

As discussed earlier, the first step to solve for pose and scale is to re-write the cost J' as a function that only depends on the rotation matrix. To do so, it is mathematically convenient to define

$$\mathbf{x} = \begin{bmatrix} \alpha_1 & \dots & \alpha_n & s & \mathbf{t}^\intercal \end{bmatrix}^\intercal. \tag{4}$$

The gradient evaluated at the optimal $\mathbf{x}^\star$ must satisfy the following constraint: $\nabla_{\mathbf{x}} J' \big|_{\mathbf{x}=\mathbf{x}^\star} = 0$. From this constraint, we obtain the following relationship:

$$\begin{aligned} \mathbf{x} &= \left(A^\intercal A + P\right)^{-1} A^\intercal W\mathbf{b} + \left(A^\intercal A + P\right)^{-1} P\mathbf{x}_0 \\ &= \begin{bmatrix} U \\ S \\ V \end{bmatrix} W\mathbf{b} + \lambda_s s_o \mathbf{l} \end{aligned} \tag{5}$$

where

$$\begin{aligned} A &= \begin{bmatrix} \mathbf{r}_1 & & & \mathbf{c}_1 & -I \\ & \ddots & & \vdots & \vdots \\ & & \mathbf{r}_n & \mathbf{c}_n & -I \end{bmatrix}, \mathbf{b} = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_n \end{bmatrix} \\ P &= \begin{bmatrix} 0_{n\times n} & & \\ & \lambda_s & \\ & & 0_{3\times 3} \end{bmatrix}, W = \begin{bmatrix} R & & \\ & \ddots & \\ & & R \end{bmatrix}, \end{aligned} \tag{6}$$

and $\mathbf{x}_0 = \begin{bmatrix} 0_n^\intercal & s_0 & 0_3^\intercal \end{bmatrix}^\intercal$. Inspired by gDLS [43] and DLS [16], we partition $\left(A^\intercal A + P\right)^{-1} A^\intercal$ into three matrices U, S, and V such that the depth, scale, and translation parameters are functions of U, S, and V, respectively. These matrices and the vector $\mathbf{l}$ can be computed in closed form by exploiting the sparse structure of the matrices A and P; see the supplemental material for the full derivation.

Eq. (5) provides a linear relationship between the depth, scale, and translation and the rotation matrix. Consequently, these parameters are computed as a function of the rotation matrix as follows:

$$\begin{aligned} \alpha_i(R) &= \mathbf{u}_i^\intercal W\mathbf{b} + \lambda_s s_o \mathbf{l}_i \\ s(R) &= SW\mathbf{b} + \lambda_s s_o \mathbf{l}_{n+1} \\ \mathbf{t}(R) &= VW\mathbf{b} + \lambda_s s_o \mathbf{l}_{\mathbf{t}}, \end{aligned} \tag{7}$$

where $\mathbf{u}_i^\intercal$ is the i-th row of matrix U, $\mathbf{l}_j$ is the j-th entry of the vector $\mathbf{l}$, and $\mathbf{l}_{\mathbf{t}}$ corresponds to the last three entries of the vector $\mathbf{l}$. Note that we can obtain the exact same relationships for depth, scale, and translation obtained by Sweeney *et al*. for gDLS [43, 44] when $\lambda_s = 0$.

In order to re-write the regularized least-squares cost function (*i.e.*, Eq. (3)) as clearly as possible, we define

$$\begin{aligned} \mathbf{e}_i &= \alpha_i(R)\mathbf{r}_i - \left(R\mathbf{p}_i + \mathbf{t}(R) - s(R)\mathbf{c}_i\right) \\ &= \boldsymbol{\eta}_i + \mathbf{k}_i \\ \boldsymbol{\eta}_i &= \mathbf{u}_i^\intercal W\mathbf{b}\mathbf{r}_i - R\mathbf{p}_i - VW\mathbf{b} + SW\mathbf{b}\mathbf{c}_i \\ \mathbf{k}_i &= \lambda_s s_0 \left(\mathbf{l}_i \mathbf{r} - \mathbf{l}_{\mathbf{t}} + \mathbf{l}_{n+1}\mathbf{c}_i\right). \end{aligned} \tag{8}$$

The residual $\mathbf{e}_i$ is divided into two terms: $\boldsymbol{\eta}_i$, the residual part considering the unconstrained terms; and $\mathbf{k}_i$ the residual part considering the scale-prior-related terms. Note again that when $\lambda_s = 0$, $\mathbf{k}_i$ becomes null and $\mathbf{e}_i$ becomes the residual corresponding to gDLS [43, 44].

Using the definitions from Eq. (8), and the scale, depth, and translation relationships shown in Eq. (7), we can now re-write the regularized least-squares cost function shown in Eq. (3) as follows:

$$\begin{aligned} J' &= J'_{\text{gDLS}} + J'_s + J'_g \\ &= \text{vec}(R)^\intercal M \text{vec}(R) + 2\mathbf{d}^\intercal \text{vec}(R) + k \end{aligned} \tag{9}$$

where

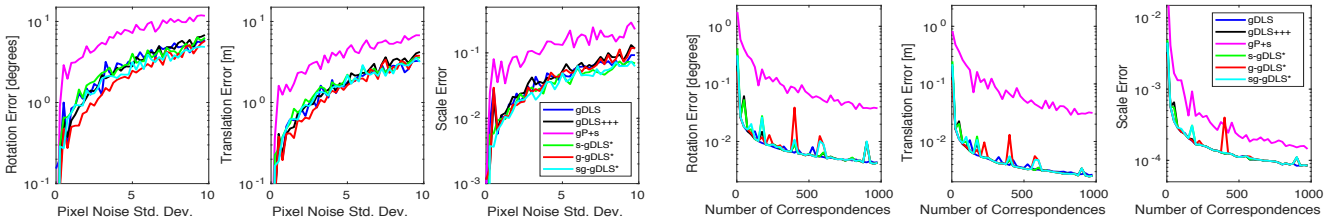
$$\begin{aligned} J'_{\text{gDLS}} &= \sum_{i=1}^{n} \mathbf{e}_i^\intercal \mathbf{e}_i = \sum_{i=1}^{n} \boldsymbol{\eta}_i^\intercal \boldsymbol{\eta}_i + 2\mathbf{k}_i^\intercal \boldsymbol{\eta}_i + \mathbf{k}_i^\intercal \mathbf{k}_i \\ &= \text{vec}(R)^\intercal M_{\text{gDLS}} \text{vec}(R) + 2\mathbf{d}_{\text{gDLS}}^\intercal \text{vec}(R) + k_{\text{gDLS}} \\ J'_s &= \lambda_s \left(s_0 - S(R)\right)^2 \\ &= \text{vec}(R)^\intercal M_s \text{vec}(R) + 2\mathbf{d}_s^\intercal \text{vec}(R) + k_s \\ J'_g &= \lambda_g \|\mathbf{g}_{\mathcal{Q}} \times R\mathbf{g}_{\mathcal{W}}\|^2 = \text{vec}(R)^\intercal M_g \text{vec}(R) \\ M &= M_{\text{gDLS}} + M_s + M_g \\ \mathbf{d} &= \mathbf{d}_{\text{gDLS}} + \mathbf{d}_s \\ k &= k_{\text{gDLS}} + k_s. \end{aligned} \tag{10}$$

The parameters of Eq. (9) (*i.e.*, $M_{\mathbf{gDLS}}$, M_s, M_g, $\mathbf{d}_{\text{gDLS}}$, $\mathbf{d}_s$, k_{gDLS}, and k_s) can be computed in closed form and in $\mathcal{O}(n)$ time; see the supplemental material for the closed form solutions of these parameters.

An important observation is that Eq. (9) generalizes the unconstrained quadratic function of gDLS shown in Eq. (1). When both priors are disabled, *i.e.*, $\lambda_g = \lambda_s = 0$, then $J'(R) = J(R)$. Also, note that the weights λ_g and λ_s allow the user to control the contribution of each of the priors independently. This gives gDLS* great flexibility since it can be adapted to many scenarios. For instance, these weights can be adjusted so that gDLS* reflects the confidence on certain priors, reduces the effect of noise present in the priors, and fully disables one prior but enables another.

3.3. Solving for Rotation

Given that the prior-based pose-and-scale cost function (*i.e.*, Eq. (3)) depends only on the rotation matrix, the next



(a) Robustness to pixel noise with minimal samples **(b)** Accuracy as a function of sample size

Figure 3. Rotation, translation, and scale errors as a function of **(a)** pixel noise when estimating pose and scale using a minimal sample of correspondences, and **(b)** sample size. s-gDLS*, g-gDLS*, and sg-gDLS* produce comparable errors to that of gDLS [43, 44]. On the other hand, gP+s [48] produces the highest errors.

step is to find R such that it minimizes Eq. (9). To achieve this, gDLS* represents the rotation matrix R using a quaternion $\mathbf{q} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}^\intercal$. To compute all the minimizers of Eq. (9), gDLS* follows [16, 22, 43, 44] and builds a polynomial system that encodes the first-order optimality conditions and the unit-norm-quaternion constraint, *i.e.*,

$$\begin{cases} \frac{\partial J'}{\partial q_j} = 0, & \forall j = 1, \ldots, 4 \\ q_j \left(\mathbf{q}^\intercal \mathbf{q} - 1 \right) = 0, & \forall j = 1, \ldots, 4 \end{cases} . \tag{11}$$

The polynomial system shown in Eq. (11) encodes the unit-norm-quaternion constraint with

$$\frac{\partial \left(\mathbf{q}^\intercal \mathbf{q} - 1 \right)^2}{\partial q_j} = q_j \left(\mathbf{q}^\intercal \mathbf{q} - 1 \right) = 0, \forall j. \tag{12}$$

Eq. (12) yields efficient elimination templates and small action matrices, which delivers efficient polynomial solvers as Kneip *et al.* [22] shows. In fact, gDLS* adopts the efficient polynomial solver of Kneip *et al.* [22] as we leverage their rotation representation

$$\text{vec}(R) = \begin{bmatrix} q_1^2 & q_2^2 & q_3^2 & q_4^2 & q_1 q_2 & q_1 q_3 & q_1 q_4 & q_2 q_3 & q_2 q_4 & q_3 q_4 \end{bmatrix}^\intercal . \tag{13}$$

Given this representation, the dimensions of the parameters of the regularized least-squares cost function shown in Eq. (9) become $M \in \mathbb{R}^{10 \times 10}$, $\mathbf{d} \in \mathbb{R}^{10}$, and $k \in \mathbb{R}$.

Because gDLS* uses the solver of Kneip *et al.* [22], it efficiently computes eight rotations. After computing these solutions, gDLS* discards quaternions with complex numbers, and then recovers the depth, scale, and translation using Eq. (7). Finally, gDLS* uses the computed similarity transformations to discard solutions that map the input 3D points behind the camera.

Our gDLS* derivation can be generalized. Imposing scale and translation priors via the regularizers is general enough to be adopted by least-squares-based estimators (*e.g.*, DLS [16] and UPnP [22]). This is because the regularizers are quadratic functions that can be added without much effort into their derivations.

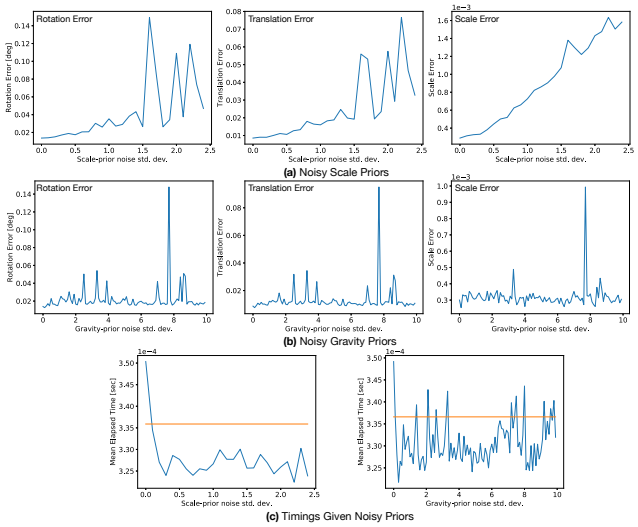


Figure 4. **(a)** gDLS*'s accuracy slowly degrades as the noise increases in the scale prior. **(b)** gDLS*'s accuracy is barely affected by noise in the gravity prior. **(c)** Time is not affected when using noisy scale (left) and gravity (right) priors. The orange line shows the best timing of gDLS+++ [44], the second fastest estimator in our experiments.

4. Experiments

This section presents experiments that use (i) synthetic data to demonstrate the numerical stability and robustness of gDLS* and (ii) real data to show the performance of gDLS* in registering a SLAM trajectory to a pre-computed point cloud. We test three gDLS* configurations: scale-only-regularized (s-gDLS*), gravity-only-regularized (g-gDLS*), and scale-gravity-regularized (sg-gDLS*). For all experiments except the ablation study, λ_s and λ_g are fixed to 1. We compare to several state-of-the-art pose-and-scale estimators: gP+s [48], gDLS [43], gDLS+++ [44], and UPnP [22]. All implementations are integrated into Theia-SfM [45]. For all experiments, we use one machine with two 2.10 GHz Intel Xeon CPUs and 32 GB of RAM.

Datasets. For the SLAM trajectory registration, the experiments use two publicly available SLAM datasets: the TUM RGBD dataset [40] and the KITTI dataset [10]. These

Table 2. Estimation times in seconds for gP+s [48], gDLS [43], gDLS+++ [44], UPnP [22], s-gDLS* (s column), g-gDLS* (g column), and sg-gDLS* (sg column) for rigid ($s_0 = 1$) and similarity ($s_0 = 2.5$) transformations. The top six rows show results for the TUM dataset, and the last six rows show results for the KITTI dataset. A gravity prior tends to deliver fast estimates, while a scale prior modestly slows down the estimation. On the other hand, scale and gravity priors tend to be modestly faster than gDLS+++.

	Rigid Transformation $[s_0 = 1]$							Similarity Transformation $[s_0 = 2.5]$					
	[48]	[43]	[44]	[22]	s	g	sg	[48]	[43]	[44]	s	g	sg
Fr1 Desk	10.77	15.38	5.79	9.60	8.11	**3.39**	5.80	10.54	15.38	5.79	8.12	**3.82**	6.89
Fr1 Room	8.86	12.23	4.51	5.52	6.66	**2.85**	4.70	8.55	11.98	4.61	7.96	**3.24**	5.40
Fr2 LargeNoLoop	5.63	8.10	2.76	2.23	2.52	**2.01**	2.10	6.46	8.25	2.82	2.49	2.40	**2.39**
Fr1 Desk2	4.99	7.72	2.96	4.20	3.89	**1.67**	3.65	5.04	7.47	2.59	3.99	**1.86**	3.38
Fr2 Pioneer SLAM	21.06	27.77	10.82	8.27	12.08	**6.48**	8.08	17.10	25.42	9.34	11.84	**7.41**	9.70
Fr2 Pioneer SLAM 2	3.39	3.49	1.93	**0.88**	1.17	1.02	1.40	3.16	3.66	1.96	1.17	1.09	**0.99**
Drive 1 $(10^{-2}$ [sec])	1.66	2.32	0.93	1.17	0.90	**0.58**	0.90	1.31	1.79	1.19	1.0	**0.61**	0.87
Drive 9	0.32	0.49	0.23	**0.14**	0.34	0.23	0.28	0.35	0.48	**0.22**	0.50	**0.22**	0.26
Drive 19	0.51	0.29	0.80	0.27	0.29	**0.26**	0.29	0.57	0.90	0.39	0.29	**0.26**	0.29
Drive 22	0.12	0.22	0.12	0.07	0.12	**0.06**	0.11	0.12	0.19	0.07	0.11	**0.06**	0.12
Drive 23 $(10^{-2}$ [sec])	3.40	4.72	2.14	2.18	2.19	**1.45**	2.04	3.25	4.71	2.07	2.34	**1.55**	1.94
Drive 29	1.15	1.95	**0.76**	1.19	1.13	0.77	1.11	1.16	1.96	0.75	1.14	**0.74**	1.12

datasets provide per-frame accelerometer estimates, which we use to compute one gravity direction for each SLAM trajectory. Specifically, we low pass filter and smooth the accelerations (because the gravity acceleration is constant within the high frequency noise) to get an estimate of the gravity vector for each image. Then, we take the mean of all these estimates to get a final gravity vector estimate. The final result is one gravity vector for each trajectory.

Error Metrics. All the experiments report rotation, translation, and scale errors. The rotation error is the angular distance [13, 17] between the expected and the estimated rotation matrix. The translation error is the L2 norm between the expected and the estimated translation. Lastly, the scale error is the absolute difference between the expected and the estimated scale values.

4.1. Robustness to Noisy Synthetic Data

This experiment consists of three parts: (1) measuring robustness to pixel noise with minimal samples (*i.e.*, four 2D-3D correspondences); (2) measuring accuracy as a function of the size of a non-minimal sample (*i.e.*, more than four 2D-3D correspondences); and (3) testing how noise in scale and gravity priors effects solution accuracy and run time. For all experiments, we execute $1,000$ trials using 10 randomly positioned cameras within the cube $[-10, 10] \times [-10, 10] \times [-10, 10]$, and 300 random 3D points in the cube $[-5, 5] \times [-5, 5] \times [10, 20]$. For each trial, we transform the 3D points by the inverse of a randomly generated ground truth similarity transformation (i.e., a random unit vector direction and random rotation angle between $0°$ and $360°$, random translation between 0 and 5 in (x, y, z), and random scale between 0 and 5).

gDLS* is robust to pixel noise with minimal samples. We generate random minimal samples with zero-mean Gaussian noise added to the pixel positions. We vary the noise standard deviation between 0 and 10 and measure

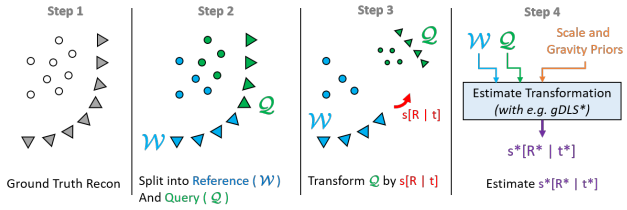


Figure 5. Evaluation Protocol: (1) Reconstruct a scene using Theia-SfM; (2) Split the reconstruction into Query ($\mathcal{Q}$) and Reference ($\mathcal{W}$) parts; (3) Transform $\mathcal{Q}$ using a similarity transform S; and (4) Estimate similarity transform $S^\star$ aligning $\mathcal{Q}$ and $\mathcal{W}$.

the rotation, translation, and scale errors. The results of this experiment can be seen in Fig. 3(a). We observe that s-gDLS*, g-gDLS*, and sg-gDLS* perform similarly to gDLS [43] and gDLS+++ [44] when comparing their rotation and translation errors. Also, we see that the rotation and translation errors produced by gP+s [48] are the highest. All methods produce similar scale errors.

Accuracy of gDLS* improves with non-minimal samples. For this experiment, we vary the size of the sample from 5 to 1000 2D-3D correspondences and fix the standard deviation to 0.5 for the zero-mean Gaussian pixel noise. Fig. 3(b) shows that s-gDLS*, g-gDLS*, and sg-gDLS* produce comparable rotation, translation, and scale errors to that of gDLS and gDLS+++. On the other hand, gP+s produced the largest errors.

gDLS* is numerically stable. From Fig. 3, we conclude that s-gDLS* (green), g-gDLS* (red), sg-gDLS* (cyan) are numerically stable because the errors are similar to that of gDLS+++.

gDLS* is robust to noise in scale and gravity priors. For this experiment, we gradually increase the noise in the scale and gravity priors. In Fig. 4(a), we see that noise in the scale prior slowly increases the rotation, translation, and

Table 3. Rotation, translation, and scale errors of gP+s [48], gDLS [43], gDLS+++ [44], UPnP [22], and gDLS* using a unit scale (*i.e.*, $s_0 = 1$) and gravity priors. The first six rows show results for the TUM dataset, and the last six rows show results for the KITTI dataset. The smallest errors are shown in bold. We observe that gDLS* and UPnP perform equivalently when comparing rotation and translation errors. However, gDLS* produces the lowest errors among the pose-and-scale estimators (*i.e.*, gP+s, gDLS, and gDLS+++).

	Rigid Transformation $[s_0 = 1]$													
	R_{error} [deg] (10^{-1})					$\mathbf{t}_{\text{error}}$ (10^{-2})					s_{error} (10^{-3})			
	[48]	[43]	[44]	[22]	Ours	[48]	[43]	[44]	[22]	Ours	[48]	[43]	[44]	Ours
Fr1 Desk	2.78	2.58	2.72	1.89	**1.57**	2.02	1.92	1.91	1.36	**1.19**	1.46	1.33	1.29	**0.68**
Fr1 Room	2.05	1.95	1.99	1.05	**0.98**	1.09	1.05	1.09	0.55	**0.52**	1.48	1.40	1.39	**0.32**
Fr2 LargeNoLoop	1.90	1.61	1.62	1.35	**1.32**	6.77	5.77	6.01	4.70	**4.45**	4.04	4.15	4.39	**1.39**
Fr1 Desk2	2.34	2.13	2.26	1.63	**1.25**	2.03	1.85	1.90	1.38	**1.06**	1.35	1.15	1.26	**0.49**
Fr2 Pioneer SLAM	1.50	1.51	1.47	**0.76**	0.87	1.29	1.18	1.19	**0.59**	0.70	2.77	2.73	2.67	**0.87**
Fr2 Pioneer SLAM 2	1.63	1.49	1.51	**0.97**	1.08	1.93	1.77	1.80	**1.22**	1.34	6.81	7.48	7.44	**0.88**
Drive 1	0.44	0.40	0.42	0.34	**0.33**	0.27	0.25	0.24	0.17	**0.16**	0.72	0.66	0.61	**0.02**
Drive 9	1.15	1.10	1.15	**0.64**	1.13	0.43	0.40	0.44	**0.13**	0.20	6.27	5.79	6.14	**0.05**
Drive 19	3.42	3.57	3.30	**2.48**	3.04	0.83	0.85	0.80	**0.63**	0.73	4.99	5.64	5.54	**0.01**
Drive 22	0.66	0.62	0.66	**0.31**	0.63	0.28	0.27	0.30	**0.16**	0.27	1.90	1.70	1.67	**0.94**
Drive 23	0.74	0.58	0.62	0.84	**0.56**	0.19	0.18	0.19	0.12	**0.09**	1.28	1.16	1.28	**0.03**
Drive 29	1.00	1.06	1.06	**0.56**	0.82	0.34	0.35	0.35	**0.21**	0.26	1.60	3.39	1.73	**0.75**

Table 4. Rotation, translation, and scale errors of gP+s [48], gDLS [43], gDLS+++ [44], and gDLS* using a scale prior of $s_0 = 2.5$ and gravity priors. The first six rows show results for the TUM dataset, and the last six rows show results for the KITTI dataset. The smallest errors are shown in bold. We see that gDLS* produces the smallest errors in almost every case.

	Similarity Transformation $[s_0 = 2.5]$											
	R_{error} [deg] (10^{-1})				$\mathbf{t}_{\text{error}}$ (10^{-2})				s_{error} (10^{-3})			
	[48]	[43]	[44]	Ours	[48]	[43]	[44]	Ours	[48]	[43]	[44]	Ours
Fr1 Desk	2.77	2.59	2.72	**2.23**	5.50	5.21	5.22	**4.55**	3.59	3.27	3.23	**1.56**
Fr1 Room	2.02	1.99	1.99	**1.29**	2.79	2.78	2.79	**1.71**	3.74	3.41	3.48	**0.91**
Fr2 LargeNoLoop	1.90	1.57	1.62	**1.49**	13.9	12.3	12.8	**10.1**	10.7	10.8	11.0	**3.66**
Fr1 Desk2	2.29	2.13	2.26	**1.77**	4.37	4.14	4.32	**3.28**	3.22	2.88	3.14	**1.37**
Fr2 Pioneer SLAM	1.43	1.48	1.47	**1.17**	3.67	3.44	3.50	**2.42**	6.98	6.81	6.68	**1.65**
Fr2 Pioneer SLAM 2	1.84	1.49	1.51	**1.27**	5.23	4.49	4.58	**3.69**	15.8	18.7	18.6	**2.42**
Drive 1	0.43	0.40	0.42	**0.33**	0.47	0.44	0.43	**0.28**	1.73	1.62	1.54	**0.05**
Drive 9	1.17	**1.09**	1.15	1.13	1.11	1.06	1.16	**0.50**	15.7	14.4	15.3	**0.12**
Drive 19	3.60	3.27	3.57	**3.09**	2.13	1.91	2.06	**1.80**	14.6	14.0	14.1	**0.04**
Drive 22	0.64	**0.62**	0.66	**0.62**	0.77	**0.75**	0.82	**0.75**	4.40	4.27	4.19	**2.34**
Drive 23	0.75	0.59	0.62	**0.57**	0.58	0.51	0.56	**0.24**	3.13	2.87	3.20	**0.14**
Drive 29	1.00	1.09	1.06	**0.82**	0.80	0.89	0.88	**0.65**	3.81	4.46	4.33	**1.86**

scale errors. Conversely, in Fig. 4(b), noise in the gravity prior has little effect on the final accuracy. Lastly, Fig. 4(c) shows that noise has a minimal effect on the solution time.

4.2. SLAM Trajectory Registration

The goal of this experiment is to measure the accuracy of an estimated similarity transformation which registers a SLAM trajectory (a collection of images from a moving camera) to a pre-computed 3D reconstruction. This experiment uses both scale and gravity priors for gDLS*. Part of this experiment considers a unit-scale similarity transformation, which makes it equivalent to a rigid transformation. In the latter case, the experiment also includes UPnP [22], a state-of-the-art multi-camera pose estimator that only estimates a rigid transformation (*i.e.*, no scale estimation).

For each dataset and method combination, we run 100 trials. Each estimator is wrapped in RANSAC [9] to estimate the transformations and the same parameters are used for all of the scale-and-pose experiments. RANSAC labels correspondences with more than 4 pixels of reprojection error as outliers. Because we use RANSAC, all methods tend to converge to accurate solutions (Tables 3 and 4); however, the speed of convergence can differ significantly (Table 2).

While there exist datasets and clear methods to evaluate visual-based localization or SfM reconstructions (*e.g.*, [5, 38]), there is not a well established methodology to evaluate pose-and-scale estimators. Previous evaluation procedures (*e.g.*, [43, 44]) mostly show camera position errors, but discard orientation and scale errors. To evaluate the registration of a SLAM trajectory, we propose a novel evaluation procedure as illustrated in Fig. 5: (1) reconstruct the trajectory using Theia-SfM; (2) remove a subset of images with their corresponding 3D points and tracks to create a new query set (the remaining images, points, and tracks are

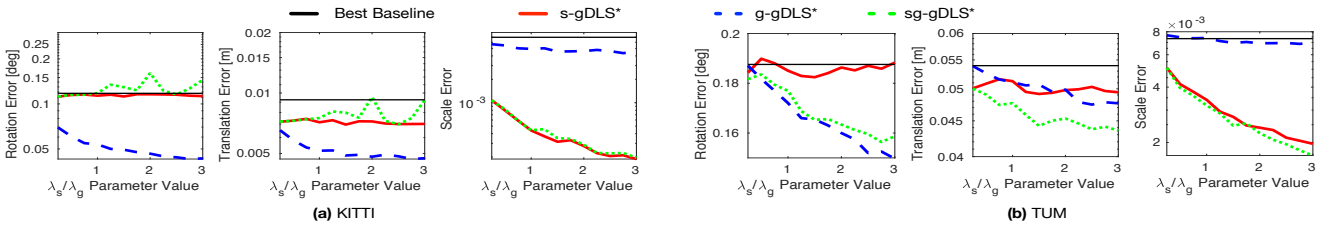


Figure 6. Average rotation, translation, and scale errors of the best baseline (best performing baseline for a given metric) and gDLS* as a function of λ_s and λ_g on **(a)** KITTI and **(b)** TUM datasets. A gravity prior (g-gDLS*) tends to reduce rotation and translation errors and modestly improves scale errors. A scale prior (s-gDLS*) tends to improve scale accuracy and modestly reduces translation errors. The combination of scale and gravity priors (sg-gDLS*) tends to reduce translation and scale errors and improves rotation estimates (see **(b)**).

the reference reconstruction); (3) apply a similarity transformation to describe the reconstruction in a different frame of reference with a different scale; and (4) estimate the similarity transformation. To compute the input 2D-3D correspondences, the evaluation procedure matches the features from the query images to the features of the reference reconstruction and geometrically verifies them. From these matches and reconstruction, the procedure builds the 2D-3D correspondences by first computing the rays pointing to the corresponding 3D points using the camera positions.

The gravity prior significantly improves speed. Table 2 shows the average estimation times for both rigid ($s_0 = 1$) and similarity ($s_0 = 2.5$) transformations. We observe that both priors help the estimators find the solution much faster than many baselines (see sg columns). In particular, a gravity only prior (see g columns) can speed up gDLS* significantly while producing good estimates (see Sec. 4.3). On the other hand, a scale only prior (see s columns) can modestly accelerate gDLS*.

Incorporating scale and rotation priors consistently improves accuracy. Tables 3 and 4 present the average rotation, translation, and scale errors of 100 trials, each estimating rigid and similarity transformations, respectively. Both Tables show six TUM trajectories at the top and six KITTI trajectories at the bottom. The scale priors s_0 are shown at the top of both Tables. Note that UPnP does not estimate scales, so it is not included in similarity transformation sections. Table 3 shows that gDLS* and UPnP produce the most accurate rotation and translation estimates, and that gDLS* produces the most accurate scale estimates. Table 4 shows that gDLS* tends to produce the most accurate rotation and translation estimates, and that gDLS* produces the most accurate scale estimates.

4.3. Ablation Study

This study aims to show the impact on the estimator accuracy of the weights λ_s and λ_g as they vary. We use the same TUM and KITTI datasets and RANSAC configuration as in previous experiments. We vary the weights from 0.25 to 3 using increments of 0.25 and run 100 trials for each weight. To summarize the results, we average the rotation, translation, and scale errors.

The priors improve accuracy and speed when used individually or together. Fig. 6 shows the results of this study. We see that on average a gravity prior (g-gDLS*) significantly improves rotation and translation errors, while modestly improving scale errors. On the other hand, a scale prior (s-gDLS*) on average significantly improves the scale errors, while modestly improving translation errors. Finally, both gravity and scale priors improve translation and scale errors and can help the estimator improve rotation errors.

From these results, we can conclude that accurate priors can greatly improve accuracy estimates (thereby also improving speed). However, we know from Fig. 4 that noisy priors can also degrade accuracy. Thus, for future work, we will explore how to automatically set λ_s and λ_g based on the noise of the priors to maximize accuracy and speed.

5. Conclusion

This work presents gDLS*, a novel pose-and-scale estimator that exploits scale and/or gravity priors to improve accuracy and speed. gDLS* is based on a least-squares re-projection error cost function which facilitates the use of regularizers that impose prior knowledge about the solution space. This gDLS* derivation is general because these regularizers are quadratic functions that can easily be added to other least-squares-based estimators. Experiments on both synthetic and real data show that gDLS* improves speed and accuracy of the pose-and-scale estimates given sufficiently accurate priors. The gravity prior is particularly effective, but the scale prior also improves the translation and scale estimates. These findings make gDLS* an excellent estimator for many applications where inertial sensors are available such as AR, 3D mapping, and robotics.

Acknowledgment

Gang Hua was supported in part by the National Key R&D Program of China Grant 2018AAA0101400 and NSFC Grant 61629301.

References

[1] Azure spatial anchors. https://azure.microsoft.com/en-us/services/spatial-anchors/. 1

[2] Martin Bujnak, Zuzana Kukelova, and Tomas Pajdla. A general solution to the p4p problem for camera with unknown focal length. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008. 1, 2

[3] Federico Camposeco, Andrea Cohen, Marc Pollefeys, and Torsten Sattler. Hybrid camera pose estimation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[4] Chu-Song Chen and Wen-Yan Chang. Pose estimation for generalized imaging device via solving non-perspective n point problem. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2002. 2

[5] Joseph DeGol, Timothy Bretl, and Derek Hoiem. Improved structure from motion using fiducial marker matching. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 3, 7

[6] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2014. 1

[7] Luis Ferraz, Xavier Binefa, and Francesc Moreno-Noguer. Very fast solution to the pnp problem with algebraic outlier rejection. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1

[8] Luis Ferraz, Xavier Binefa, and Francesc Moreno-Noguer. Very fast solution to the pnp problem with algebraic outlier rejection. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[9] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 7

[10] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *Intl. Journal of Robotics Research (IJRR)*, 2013. 2, 5

[11] Marcel Geppert, Peidong Liu, Zhaopeng Cui, Marc Pollefeys, and Torsten Sattler. Efficient 2d-3d matching for multi-camera visual localization. *ArXiV preprint arXiv:1809.06445*, 2018. 1

[12] M. D. Grossberg and S. K. Nayar. A general imaging model and a method for finding its parameters. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2001. 1

[13] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *Intl. Journal of Computer Vision (IJCV)*, 103(3):267–305, 2013. 6

[14] Gim Hee Lee, Bo Li, Marc Pollefeys, and Friedrich Fraundorfer. Minimal solutions for pose estimation of a multi-camera system. In *Robotics Research: The 16th Intl. Symposium ISRR*, pages 521–538. Springer, 2016. 2

[15] Lionel Heng, Benjamin Choi, Zhaopeng Cui, Marcel Geppert, Sixing Hu, Benson Kuan, Peidong Liu, Rang Nguyen, Ye Chuan Yeo, Andreas Geiger, et al. Project autovision: Localization and 3d scene perception for an autonomous vehicle with a multi-camera system. *ArXiV preprint arXiv:1809.05477*, 2018. 1

[16] Joel A Hesch and Stergios I Roumeliotis. A direct least-squares (DLS) method for PnP. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2011. 1, 2, 3, 4, 5

[17] Du Q Huynh. Metrics for 3d rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 35(2):155–164, 2009. 6

[18] Eagle S Jones and Stefano Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *The Intl. Journal of Robotics Research*, 30(4):407–430, 2011. 1

[19] Georg Klein and David Murray. Parallel tracking and mapping on a camera phone. In *Proc. of the IEEE Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, 2009. 1

[20] Laurent Kneip, Margarita Chli, and Roland Y Siegwart. Robust real-time visual odometry with a single camera and an imu. In *Proc. of the British Machine Vision Conference (BMVC)*, 2011. 1

[21] Laurent Kneip, Paul Furgale, and Roland Siegwart. Using multi-camera systems in robotics: Efficient solutions to the npnp problem. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2013. 1, 2

[22] Laurent Kneip, Hongdong Li, and Yongduek Seo. Upnp: An optimal o(n) solution to the absolute pose problem with universal applicability. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2014. 1, 2, 5, 6, 7

[23] Laurent Kneip, Davide Scaramuzza, and Roland Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *Proc. of the IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2011. 1, 2

[24] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Automatic generator of minimal problem solvers. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2008. 2

[25] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Closed-form solutions to minimal absolute pose problems with known vertical direction. In *Proc. of the Asian Conf. on Computer Vision (ACCV)*, 2010. 2

[26] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Real-time solution to the absolute pose problem with unknown radial distortion and focal length. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2013. 1, 2

[27] Zuzana Kukelova, Jan Heller, and Andrew Fitzgibbon. Efficient intersection of three quadrics and applications in computer vision. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[28] Viktor Larsson, Magnus Oskarsson, Kalle Astrom, Alge Wallis, Zuzana Kukelova, and Tomas Pajdla. Beyond grobner bases: Basis selection for minimal solvers. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[29] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o(n) solution to the pnp problem. *Intl. Journal of Computer Vision (IJCV)*, 81(2):155, 2009. 1, 2

[30] Jesse Levinson, Michael Montemerlo, and Sebastian Thrun. Map-based precision vehicle localization in urban environments. In *Robotics: Science and Systems*, 2007. 1

[31] Yi Ma, Stefano Soatto, Jana Kosecka, and S Shankar Sastry. *An invitation to 3-d vision: from images to geometric models*, volume 26. Springer Science & Business Media, 2012. 1

[32] Pierre Martin, Eric Marchand, Pascal Houlier, and Isabelle Marchal. Mapping and re-localization for mobile augmented reality. In *Proc. of the IEEE Intl. Conf. on Image Processing (ICIP)*, 2014. 1

[33] P. Miraldo and H. Araujo. A simple and robust solution to the minimal general pose estimation. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014. 1

[34] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017. 1

[35] David Nistér and Henrik Stewénius. A minimal solution to the generalised 3-point pose problem. *Journal of Mathematical Imaging and Vision*, 27(1):67–79, 2007. 2

[36] Gabriel Nützi, Stephan Weiss, Davide Scaramuzza, and Roland Siegwart. Fusion of imu and vision for absolute scale estimation in monocular slam. *Journal of Intelligent & Robotic Systems*, 2011. 3

[37] Robert Pless. Using many cameras as one. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2003. 1

[38] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, et al. Benchmarking 6dof outdoor visual localization in changing conditions. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 7

[39] Dieter Schmalstieg and Tobias Hollerer. *Augmented reality: principles and practice*. Addison-Wesley Professional, 2016. 1

[40] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the Intl. Conf. on Intelligent Robot Systems (IROS)*, 2012. 2, 5

[41] Chris Sweeney, John Flynn, Benjamin Nuernberger, Matthew Turk, and Tobias Höllerer. Efficient computation of absolute pose for gravity-aware augmented reality. In *Proc. of the IEEE Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, 2015. 1

[42] Chris Sweeney, John Flynn, and Matthew Turk. Solving for relative pose with a partially known rotation is a quadratic eigenvalue problem. In *Proc. of the Intl. Conf. on 3D Vision (3DV)*, 2014. 1

[43] Chris Sweeney, Victor Fragoso, Tobias Höllerer, and Matthew Turk. gDLS: A scalable solution to the generalized pose and scale problem. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2014. 1, 2, 3, 4, 5, 6, 7

[44] Chris Sweeney, Victor Fragoso, Tobias Höllerer, and Matthew Turk. Large scale sfm with the distributed camera model. In *Proc. of the IEEE Intl. Conf. on 3D Vision (3DV)*, 2016. 1, 2, 3, 4, 5, 6, 7

[45] Christopher Sweeney, Tobias Hollerer, and Matthew Turk. Theia: A fast and scalable structure-from-motion library. In *Proc. of the ACM Intl. Conf. on Multimedia*, 2015. 5

[46] K. Tsotsos, A. Chiuso, and S. Soatto. Robust inference for visual-inertial sensor fusion. In *Proc. of the Intl. Conference on Robotics and Automation (ICRA)*. 2015. 1

[47] Jonathan Ventura, Clemens Arth, Gerhard Reitmayr, and Dieter Schmalstieg. Global localization from monocular slam on a mobile phone. *IEEE Transactions on Visualization and Computer Graphics*, 20(4):531–539, 2014. 1

[48] Jonathan Ventura, Clemens Arth, Gerhard Reitmayr, and Dieter Schmalstieg. A minimal solution to the generalized pose-and-scale problem. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1, 2, 5, 6, 7

[49] Jonathan Ventura and Tobias Höllerer. Wide-area scene mapping for mobile visual tracking. In *Proc. of the IEEE Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, 2012. 1

[50] Stephan Weiss, Markus W Achtelik, Simon Lynen, Michael C Achtelik, Laurent Kneip, Margarita Chli, and Roland Siegwart. Monocular vision for long-term micro aerial vehicle state estimation: A compendium. *Journal of Field Robotics*, 30(5):803–831, 2013. 1

[51] Changchang Wu. P3. 5p: Pose estimation with unknown focal length. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[52] Bernhard Zeisl, Torsten Sattler, and Marc Pollefeys. Camera pose voting for large-scale image-based localization. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2015. 1

[53] Yinqiang Zheng, Yubin Kuang, Shigeki Sugimoto, Kalle Astrom, and Masatoshi Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2013. 1, 2

[54] Julius Ziegler, Henning Lategahn, Markus Schreiber, Christoph G Keller, Carsten Knöppel, Jochen Hipp, Martin Haueis, and Christoph Stiller. Video based localization for bertha. In *Proc. of the IEEE Intelligent Vehicles Symposium*, 2014. 1