Hybrid Camera Pose Estimation

Federico Camposeco[1] Andrea Cohen[1] Marc Pollefeys[1,2] Torsten Sattler[1]
[1]Department of Computer Science, ETH Zürich [2]Microsoft
{fede, acohen, pomarc, tsattler}@inf.ethz.ch

Abstract

In this paper, we aim to solve the pose estimation problem of calibrated pinhole and generalized cameras w.r.t. a Structure-from-Motion (SfM) model by leveraging both 2D-3D correspondences as well as 2D-2D correspondences. Traditional approaches either focus on the use of 2D-3D matches, known as ***structure-based*** *pose estimation or solely on 2D-2D matches (****structure-less*** *pose estimation). Absolute pose approaches are limited in their performance by the quality of the 3D point triangulations as well as the completeness of the 3D model. Relative pose approaches, on the other hand, while being more accurate, also tend to be far more computationally costly and often return dozens of possible solutions. This work aims to bridge the gap between these two paradigms. We propose a new RANSAC-based approach that automatically chooses the best type of solver to use at each iteration in a data-driven way. The solvers chosen by our RANSAC can range from pure structure-based or structure-less solvers, to any possible combination of* ***hybrid*** *solvers (*i.e. *using both types of matches) in between. A number of these new hybrid minimal solvers are also presented in this paper. Both synthetic and real data experiments show our approach to be as accurate as structure-less approaches, while staying close to the efficiency of structure-based methods.*

1. Introduction

Camera pose estimation, *i.e.*, estimating the position and orientation of a given image, is a central step in 3D computer vision approaches such as SfM [1, 11, 25], Simultaneous Localization and Mapping (SLAM) [7], and visual localization [4, 18, 22, 27, 35]. In addition, camera pose estimation plays an important role in applications such as self-driving cars [9] and augmented reality [19].

The traditional approach to camera pose estimation is to estimate the pose from a set of 2D-3D matches between pixels in a query image and 3D points in a scene model [10]. The pose is typically computed by applying a *structure-based* minimal pose solver inside a RANSAC [8] loop. There is a large body of work on absolute pose estimation from n 2D-3D matches, a problem typically referred to as n-point perspective pose (PnP). Solutions to this problem exist for calibrated [8, 10], partially calibrated [31, 34], and uncalibrated cameras [12]. When the 3D model is obtained via SfM, as is typical for visual localization [4, 18, 22, 27, 35], not all 3D points will be accurately triangulated, leading to potentially inaccurate pose estimates. An alternative to pose estimation from 2D-3D matches is *structure-less pose resectioning* [36]: The pose of a query image is estimated from a set of 2D-2D correspondences between the query and two or more images in the reconstruction. While this approach avoids the problem of inaccurately triangulated points and incompleteness of the model, structure-less pose solvers are significantly less computationally efficient (up to orders of magnitude slower).

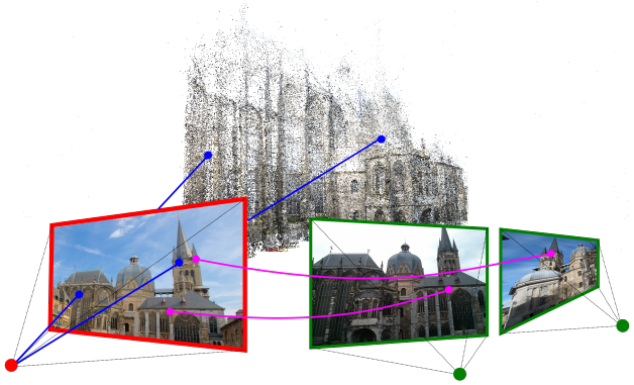

Figure 1. Visualization of 2D-2D matches (pink) and 2D-3D matches (blue) used by one of our hybrid pose solvers. The query camera is represented in red and SfM cameras in green.

The availability of both structure-based and structure-less camera pose estimation techniques leads to a set of interesting questions: Are they mutually exclusive, *i.e.*, is one always preferable over the other, or is there value in using both 2D-3D and 2D-2D matches for pose estimation? Is it best to use "pure" solvers, *i.e.*, solvers that use either 2D-3D or 2D-2D correspondences, or do *hybrid* solvers (*c.f.* Fig. 1) using both type of matches improve pose estimation performance? Should one decide prior to RANSAC which

solver to use, or is it best to select solvers in a data-driven way during RANSAC-based pose estimation?

The goal of this paper is to answer these questions. To this end, we propose *nine* novel hybrid camera pose solvers, differing by the number of 2D-3D and 2D-2D matches each one uses and whether they deal with a central or a generalized camera (or both). In order to use structure-based, structure-less, and hybrid solvers side by side, we propose a new hybrid RANSAC-variant that first samples a pose solver according to its probability of success and next selects a suitable minimal sample for this solver. Through extensive tests on both synthetic and real data, we analyze our new solvers and demonstrate that our hybrid RANSAC scheme consistently outperforms purely structure-based and structure-less approaches.

2. Background

Structure-based pose estimation. The classical procedure followed in structure-based pose estimation consists on first obtaining a set of putative 2D-3D matches. These putative matches are usually obtained from matching keypoints in a query image (or image sequence) against mean descriptors associated to each 3D point in the model. Given these matches, the camera pose is robustly inferred by employing a minimal solver inside a Hypothesize-and-Verify scheme (*e.g*. RANSAC [8]).

Different minimal solvers have been proposed in the literature, depending on whether the camera to localize is central or generalized (*i.e*., with multiple centers of projection). For the case of central cameras, [8, 14] require a minimal sample of 3 matches as they deal with a fully calibrated setting, while [34] requires 3.5 matches as a focal length is also estimated. [31] also seeks to estimate the center of projection, and thus requires 5 matches. [12] assumes unknown radial distortion, and uses a minimal set of 4 matches. [16] uses 3 matches to estimate the pose of a calibrated generalized camera, assuming that its scale w.r.t. to the 3D model is known. [28] additionally assumes a known vertical direction, requiring only 2 matches in order to compute a pose. In [16, 28], the scale of the generalized cameras w.r.t. the SfM to be known. This is not always the case in many Computer Vision scenarios, where often the 3D models available are up to scale. Addressing this, [32] deals with the case of unknown scale and unknown vertical direction, thus requiring 4 points. Since we will focus on a calibrated scenario in this paper, we will make use of some of this solvers (namely, [14, 28, 32]) alongside both structure-less and hybrid solvers in a novel RANSAC-based approach.

Structure-less pose estimation. The pose accuracy obtained by structure-based methods is limited by the quality of the 3D points that the camera observes. Additionally, the number of inlier matches a query camera can have is bounded by the number of 3D points the camera sees. Tackling these issues has been the focus of recent research. Rather than explicitly representing the scene by its 3D structure, 2D-2D correspondences between the query image (or images) against multiple images are used for pose estimation. In these cases, the query is usually matched against the most similar images (2 or more [36]) present in the SfM model, which are found using an image retrieval approach.

Alternatively, 2D-2D matches can also be obtained by following the same procedure described for structure-based pose estimation. Once the a set of 2D-3D matches is obtained, the 3D counterpart of each match can be replaced by one of the SfM camera rays used to reconstruct this 3D point. Next, a minimal solver is used inside a RANSAC loop in order to robustly estimate a camera pose. [36] proposes a minimal solver that can deal with both calibrated central cameras, using 6 matches, as well as uncalibrated ones (using 7 matches). For the problem of localizing a generalized camera, [26] proposes a 6 point solution that requires prior knowledge on the relative scale of the camera. If the vertical direction is known, only 4 matches are needed [29]. [30] estimates the scale while also assuming a known vertical direction, thus requiring 5 matches. All of these solvers share the property of being more accurate than their structure-based counterparts. However, they can be up to orders of magnitude slower, which is further aggravated by the fact that RANSAC needs to run exponentially longer due to the larger size of the minimal samples. In Sec. 3, we introduce a solver-selection step that allows the use of these type of solvers only when the current state of the problem suggests that they have a high chance of success, thus reducing the total running time of robust pose estimation.

Hybrid pose estimation. Somewhere in-between there are solvers that simultaneously use different types of matches, referred to as *hybrid* solvers, although they have not gained much attention in the past. [5] uses two 2D-3D matches and one 3D-3D match in order to estimate the pose and scale of a generalized camera. [6] uses one 3D-3D match and 3 2D-2D matches in order to find the relative pose of a stereo pair. These two approaches, however, can be twice affected by the inaccuracy of point triangulation due to the use of a 3D-3D match. Most related to the type of solvers that we explore in this work, is [13], where 2 2D-2D and 2 2D-3D matches are used to find the pose of a central camera. In section Sec. 4 we propose a number of solvers that can also deal with generalized cameras, as well as different mixtures of 2D-2D and 2D-3D matches, and show how their combination can be valuable for pose estimation.

Adapted RANSAC schemes. Since we are looking to deal with different types of matches, RANSAC needs to be adapted to sample from two different sets with potentially different inlier ratios. [5, 6] deal with this problem by keep-

ing track of the two (or more) inlier ratios and modifying the RANSAC termination criterion to take them into account. In this work, we adopt the same termination criterion, as we also face the same issue. However, we also need to devise a strategy to choose a suitable solver for RANSAC. The choice of solver should be driven by the data and the current estimate. This task is akin to model selection, where models are selected on the fly according to a probability criterion based on the current state of the problem.

3. Hybrid RANSAC for Pose Estimation

Let $\mathcal{M}_p$ be a set of putative 2D-3D matches and $\mathcal{M}_r$ be the set of putative 2D-2D matches. Traditionally, only one of these sets is used to robustly estimate the pose of a query camera using RANSAC combined with a minimal solver such as P3P or a 2D-2D structure-less pose solver such as [36]. However, we would like to exploit all of the available information, *e.g*. by using *hybrid minimal solvers* that use a combination of both 2D-3D and 2D-2D matches. In order to use such solvers, RANSAC would need to sample from both $\mathcal{M}_r$ and $\mathcal{M}_p$. Consequently, its termination criterion should be adapted to work with two separate inlier ratios, which we will call ε_r and ε_p, as presented in [5, 6]. Nevertheless, depending on the quality of the matches, *e.g*. noise in the 3D or 2D points, different number of outliers contaminating the matching sets, *etc*., as well as the quality of the solver itself, different solvers may yield better results than others. Therefore, given a set of minimal solvers S that require different mixtures of 2D-2D and 2D-3D matches, we would like RANSAC to be able to automatically choose a solver depending on the quality of the data. This choice may change from one iteration to the next based on an improved estimate of the inlier sets, *e.g*., relative inlier ratios may change when a better model is found which should affect our choice of solver.

Three questions arise when designing a RANSAC variant that can cope with two different sets of matches and inlier ratios, as well as a number of different minimal solvers:

1. How do we score a hypothesized model, *i.e*. how do we choose the best model so far?
2. Given the estimated inlier ratios $\hat{\varepsilon}_r$ and $\hat{\varepsilon}_p$, as well as our past choices of solver, which is the best solver to use for the next iteration?
3. When should we terminate?

For the first question, we adopt a classical RANSAC approach, and choose the best model as the one with the highest inlier count overall, taking into account both sets of matches. Notice that one may choose a different inlier threshold for each set of matches.

The issue of choosing a solver for the next iteration is tackled using a probability-guided sampling strategy. At each iteration, a solver is chosen according to its probability of succeeding (estimating a model from an all-inlier minimal sample) at this iteration for the first time. The intuition behind using this probability is that we want our solver to be as exploratory as possible as long as the data allows it. Therefore, we would like to select a solver with a high chance of success which has not been used enough to have found a valid solution yet. This means that its chances of finding a good solution for the first time should be high. We will refer to this probability as *success* probability, or P_s.

Let s be a solver that requires a minimal set of n matches from $\mathcal{M}_r$ and m matches from $\mathcal{M}_p$. Let ε_r and ε_p be the true inlier ratios of the sets $\mathcal{M}_r$ and $\mathcal{M}_p$, respectively. The probability of sampling an all-inlier minimal set for solver s at any iteration is given by $\varepsilon_r^n \varepsilon_p^m$. The probability of the solver not having seen an all-inlier set by iteration k_s is given by $(1 - \varepsilon_r^n \varepsilon_p^m)^{k_s - 1}$. It follows then that the probability P_s of choosing a good sample at iteration k_s of model s for the first time is given by

$$P_s = \varepsilon_r^n \varepsilon_p^m (1 - \varepsilon_r^n \varepsilon_p^m)^{k_s - 1} \quad . \tag{1}$$

Since the true inlier ratios are unknown, they can be replaced by the estimated inlier ratios $\hat{\varepsilon}_r$ and $\hat{\varepsilon}_p$ of the best model found so far. In order to compute P_s, RANSAC needs to keep track of how many times each solver has been chosen in the past. Therefore, for each solver s, a number of iterations k_s has to be stored. Notice that the more a solver is used, the less likely it is to be chosen in later iterations, therefore allowing RANSAC to eventually explore previously unused solvers.

Another factor that should affect the choice of solver is the quality of the solver itself, *i.e*. how accurate is the solution estimated given an all-inlier sample. Consequently, we will weight the computed success probability with a prior on the quality of the solver. This quality prior, referred to as P_p, is empirically chosen based on the solver's numerical stability and the size of its minimal set. The intuition behind this being that the more matches a solver requires, the more it will be affected by their noise. Therefore, we rank the solvers according to their minimal set size and their numerical stability (which is analyzed in Sec. 5.1) and assign them a normalized prior according to this ranking. Notice that this prior does not need to be recomputed during RANSAC and does not depend on the data. Finally, the solver is chosen according to this prior multiplied by its probability of success. Once the solver is picked, we then randomly sample n and m matches from $\mathcal{M}_r$ and $\mathcal{M}_p$, respectively (*c.f*. the supplementary material for additional details).

The last question to answer involves the termination criterion. For each solver s, the minimum number of iterations $\mathbf{K}_s$ that guarantees that it will find a good solution with probability P (usually set to 0.99) is given by:

$$\mathbf{K}_s = \frac{\log(1 - P)}{\log(1 - \varepsilon_r^n \varepsilon_p^m)} \quad . \tag{2}$$

Algorithm 1 Hybrid RANSAC

Require: $\mathcal{S}, \mathcal{M}_r, \mathcal{M}_p$, inlier thresholds σ_r, σ_p
Require: $\forall s \in \mathcal{S}$, minimal set sizes n_s, m_s, prior $P_p(s)$
$\forall s \in \mathcal{S}$, initialize success probability $P_s(s) = 1$,
while TRUE **do**
  Choose $s \in \mathcal{S}$ with probability $P_s(s)P_p(s)$
  $k_s \leftarrow k_s + 1$
  Randomly choose $S_r \subset \mathcal{M}_r, |S_r| = n_s$
  Randomly choose $S_p \subset \mathcal{M}_p, |S_p| = m_s$
  Compute pose $\theta = s(S_p \cup S_r)$
  I_r = inlier_count($\theta, \mathcal{M}_r, \sigma_r$)
  I_p = inlier_count($\theta, \mathcal{M}_p, \sigma_p$)
  $Inliers(\theta) \leftarrow I_r + I_p$
  $\hat{\varepsilon}_r \leftarrow I_r/|\mathcal{M}_r|, \hat{\varepsilon}_p \leftarrow I_p/|\mathcal{M}_p|$
  if $Inliers(\theta) > Inliers(\theta^*)$ **then**
    $\theta^* \leftarrow \theta$
    for all $s \in \mathcal{S}$ **do**
      Update $P_s(s, \hat{\varepsilon}_r, \hat{\varepsilon}_p)$ using (1)
      Update $\mathbf{K}_s$ using (2)
      if $k_s \geq \mathbf{K}_s$ **then**
        return best model θ^*

Thus, our RANSAC variant stops when at least one solver s has been chosen $\mathbf{K}_s$ times, as this means that a good solution for the current inlier ratios has been found with probability P. Note that at every iteration, the number of iterations k_s for the last chosen solver s has to be updated, and, if the model found is better than the best model found so far, the maximum number of iterations $\mathbf{K}_{s'}$ is updated for all solvers $s' \in \mathcal{S}$. The chosen termination criterion is optimistic in the sense that it stops as soon as one solver is finished. Its pessimist counterpart would have RANSAC run for as long as necessary for each solver to complete its iterations. We argue that this is unnecessary, as this would require RANSAC to wait for the most unsuited solvers to finish even though their chances of finding a valid model are very low (therefore needing more iterations). Indeed, when a solver is called very few times, it is usually because it relies on the set of matches with the lowest inlier ratio, therefore it is unsuited for this particular data. In practice, the optimistic criterion allows for a good trade-off between accuracy and run-time, as will be shown in Sec. 5. Our RANSAC variant is presented in Alg. 1.

4. Hybrid Minimal Solvers

In this section we detail the derivation of all new solvers which are required for our RANSAC variant (*c.f.* Alg. 1). Given a combination of n 2D-2D matches and m 2D-3D matches, we have $2m + n$ constraints: A 2D-2D match will provide 1 algebraic constraint on the pose while a 2D-3D match will provide 2 constraints. If d is the number of degrees of freedom of the camera pose, we are interested in the cases where $2m + n = d$, *i.e.*, minimal problems. In this paper we will focus on 4 different pose problems, depending on what is assumed as prior knowledge regarding the pose (*c.f.* Table 1). The first problem, referred here as **Problem 6-DoF**, is the case of a generalized camera where only its rotation and translation are unknown, and so $d = 6$. Notice that the problem of single central camera pose estimation is a specific case of this problem. **Problem 4-DoF** has $d = 4$ and refers to the *upright* case of a known-scale camera, *i.e.*, where the vertical direction is known. **Problem 7-DoF** is the problem where we do not know the scale of the generalized camera w.r.t. to the map. Thus, for this problem $d = 7$. Finally with $d = 5$, **Problem 5-DoF** is the upright version of **Problem 7-DoF**, *i.e.*, we know the vertical direction but not the scale.

	U	S	# Matches 2D	# Matches 3D	Name	Reference	Num. Sols.
Problem 6-DoF			0	3	(g)P3P	[14, 16]	4/8[1]
			2	2	**H22**	Sec. 4.2	16
			4	1	**H41**	Sec. 4.2	32
			6	0	Strless	[26, 36]	64[2]
Prob. 4-DoF	•		0	2	(g)P2P	[28]	2
	•		2	1	**uH21**	Sec. 4.2	4
	•		4	0	QEP	[29]	6
Problem 7-DoF		•	0	4	P4P+s	[32]	8
		•	1	3	**H13+s**	Sec. 4.1	16
		•	3	2	**H32+s**	Sec. 4.1	56
		•	5	1	**H51+s**	Sec. 4.1	80[3]
		•	7	0	SevenPt	[30]	140[3]
Problem 5-DoF	•	•	0	3	**uP3P+s**	Sec. 4.1	1
	•	•	1	2	**uH12+s**	Sec. 4.1	4
	•	•	3	1	**uH31+s**	Sec. 4.1	6
	•	•	5	0	FivePt	[30]	10

Table 1. Summary of Minimal Solvers. **U** and **S** stand for upright (*i.e.*, known vertical) and unknown scale, respectively. The solvers presented in this paper are in boldface.

As detailed in Table 1, most of the "pure" solvers already exist in the literature (*e.g.*, P3P), with the exception of **uP3P+s**. Conversely, most of the *hybrid* solvers are yet to be derived. This with the exception of **H22** and **H41**, which have been previously explored in [13]. However, the versions there are only meant for central cameras, thus here we offer a more general derivation. In total, there are 7 minimal solvers which our hybrid RANSAC requires and that, to the best of our knowledge, have not yet been investigated in the literature. We also offer a novel, more general derivation for 2 existing solvers. We note, however, that two of the solvers required for **Problem 7-DoF** are of very high polynomial degree and have been deemed too unstable for practical use [30]. As such, we do not investigate a full Hybrid RANSAC solution for **Problem 7-DoF**. Nevertheless, we offer derivations and synthetic precision evaluations for the more tractable cases: **H13+s** and **H32+s**.

Rotation Parameterization. Let $\mathtt{R}$ and $\mathtt{t}$ be the rotation and translation that transform elements in the global frame of reference $\{G\}$ to the camera frame of reference $\{C\}$. For the unkonwn vertical cases (**Problem 6-DoF** and **Problem 7-DoF**), we parameterize the rotation using a unit quaternion $[u_1 u_2 u_3 w]^\intercal$, setting $w = 1$, *i.e.*, $\mathtt{R} = \mathtt{R}(\mathbf{u})$. Even though setting $w = 1$ eliminates the possibility of

[1] For the generalized gP3P we have 8 solutions. 4 for P3P the case.
[2] If the query camera is central, we may have 20, 40, 56 and 64 solutions depending on the SfM cameras' configuration [36].
[3] Solvers are too numerically unstable to be useful *c.f.* Sec. 4.2 and [30].

finding a rotation with $w = 0$, this has a negligible impact on the solver's performance for real data and has been used widely in previous work [26, 29, 36].

For upright cases (**PROBLEM 4-DOF** and **PROBLEM 5-DOF**), *i.e.*, where the vertical direction in the camera frame is known, we are dealing with a rotation around a known axis. Without loss of generality we assume the vertical direction in $\{C\}$ to be $[0\ 0\ 1]^\intercal$, which can always be achieved using a pre-processing step (details of this are given in the supplemental material). This results in a rotation around the Z axis, which we choose to parameterize as

$$\mathtt{R}(a,b) = \begin{bmatrix} a & -b & 0 \\ b & a & 0 \\ 0 & 0 & 1 \end{bmatrix} , \tag{3}$$

where we must enforce that $a^2 + b^2 = 1$.

2D-3D Constraints. Given $i = 1 \dots m$ 2D-3D matches, we may write

$$\alpha_i \mathbf{v}_i + s\, \mathbf{c}_i = \mathtt{R}\mathbf{p}_i + \mathtt{t}\,, \tag{4}$$

where α_i is the depth of the i-th point in $\{C\}$, $\mathbf{v}_i$ is a ray of unit length, s is the scale of the generalized camera, $\mathbf{c}_i$ is the center of projection for the generalized measurement and $\mathbf{p}_i$ is the 3D point in $\{G\}$. In many cases it is useful to eliminate the unknown depths, thus one may rewrite (4) as

$$\lfloor \mathbf{v}_i \rfloor_\times (\mathtt{R}\mathbf{p}_i + \mathtt{t} - s\, \mathbf{c}_i) = \mathbf{0}\,, \tag{5}$$

where $\lfloor \mathbf{a} \rfloor_\times \in \mathbb{R}^{3\times 3}$, such that $\lfloor \mathbf{a} \rfloor_\times \mathbf{b} = \mathbf{a} \times \mathbf{b}$. Only 2 of the 3 equations in (5) are linearly independent, thus only the first two are taken into account. For the case where scale is known, we simply set $s = 1$, whereas for central cameras we may always set $\mathbf{c}_i = \mathbf{0}$. Notice that (5) is linear in $\mathtt{t}$ and s, two of the potential unknowns. Depending on how many 2D-3D matches are available, we manipulate (5) in two different ways.

2 *2D-3D matches or more:* In this case, $\mathtt{t}$ and s may be eliminated. For known scale problems we need only 3 of the 4 equations provided by 2 2D-3D matches to get

$$\mathbf{A}\mathtt{t} = \mathbf{B}(\mathbf{r})\,, \tag{6}$$

where $\mathbf{A} \in \mathbb{R}^{3\times 3}$ is a coefficient matrix obtained from the inputs, and $\mathbf{B}(\mathbf{r}) \in \mathbb{R}^3$ is a function of the rotation parameters $\mathbf{r}$, which can be (a, b) or $\mathbf{u}$ as explained above. In the case of unknown scale, we must use all of the 4 constraints available in the 2 2D-3D matches to isolate the linear unknowns. For this case we have

$$\mathbf{A}_s\ [\mathtt{t}^\intercal\ s]^\intercal = \mathbf{B}_s(\mathbf{r})\,, \tag{7}$$

where we now have $\mathbf{A}_s \in \mathbb{R}^{4\times 4}$ and $\mathbf{B}_s \in \mathbb{R}^4$. We can then invert either $\mathbf{A}$ or $\mathbf{A}_s$ in order to obtain the linear unknowns in terms of the rotation only.

1 *2D-3D match:* in this case we instead transform the point in $\{G\}$ and the corresponding camera frame. We do this s.t. $\mathbf{c}_1 = \mathbf{0}$, $\mathbf{v}_1 = [0\ k_1\ k_2]$ and $\mathbf{p}_1 = \mathbf{0}$ (details of this transformation are offered in the supplemental material). Doing this allows us to then rewrite (4) as $\alpha_1 \mathbf{v}_1 = \mathtt{t}$, and eliminating α_1 leads to

$$\mathtt{t} = \begin{bmatrix} 0 \\ \mathtt{t}_z \frac{k_1}{k_2} \\ \mathtt{t}_z \end{bmatrix} , \tag{8}$$

where $\mathtt{t}_z$ is the last element of the unknown translation.

2D-2D Constraints. A 3D line can be represented using Plücker coordinates [21] as $\mathbf{L} = [\mathbf{q}^\intercal\ \mathbf{q}'^\intercal]^\intercal \in \mathbb{R}^6$. Here $\mathbf{q}$ is the unit direction of the line, and $\mathbf{q}' = \mathbf{q} \times \mathbf{p}$ where $\mathbf{p}$ is any point on the line. A given pair of lines, ${}^a\mathbf{L}$ and ${}^b\mathbf{L}$, intersect in space iff ${}^a\mathbf{q} \cdot {}^b\mathbf{q}' + {}^a\mathbf{q}' \cdot {}^b\mathbf{q} = 0$. In order for the j-th line (with $j = 1 \dots n$) in the camera frame ${}^{\mathrm{C}}\mathbf{L}_j$ to intersect with its matched line in the global frame ${}^{\mathrm{G}}\mathbf{L}_j$ we must have that

$${}^{\mathrm{C}}\mathbf{q}_j^\intercal\, \mathtt{R}\, {}^{\mathrm{G}}\mathbf{q}'_j + {}^{\mathrm{C}}\mathbf{q}'^\intercal_j\, \mathtt{R}\, {}^{\mathrm{G}}\mathbf{q}_j = {}^{\mathrm{C}}\mathbf{q}_j^\intercal\, \lfloor \mathtt{t} \rfloor_\times \mathtt{R}\, {}^{\mathrm{G}}\mathbf{q}_j\,, \tag{9}$$

Notice that, as opposed to (5), (9) is homogeneous in $\mathtt{R}$, and so the scale of the rotation is irrelevant. This is important in the case of the quaternion-based rotation, since every element of $\mathtt{R}(\mathbf{u})$ has a common factor of $1/(|u|^2+1)$ in order for $\mathtt{R}$ to be a proper rotation. For (9) this nonlinear common term can be safely factored out.

Gröbner Basis Solution. For any given solver, we adopt a Gröbner basis approach to solve the resulting polynomial system. For this, we employ a Gröbner basis Solver Generator [15, 17]. The solution of the polynomial system is then given by the Gauss-Jordan reduction of an *elimination template*, and the eigen-decomposition of an action matrix with entries resulting from the elimination template. Thus, the speed and numerical stability of the resulting solver will depend on the size of the elimination template and the size of the action matrix (which also dictates the number of solutions). A more detailed discussion of this method is out of the scope of this paper, for more details see [3].

4.1. Unknown Scale Solvers

H13+s First we obtain $\mathtt{t}(\mathbf{u})$ and $s(\mathbf{u})$ using the first 2 2D-3D points (*c.f.* (7)). Then, we are left with one more 2D-3D match (2 constraints) and one 2D-2D match (1 equation). We substitute $\mathtt{t}(\mathbf{u})$ and $s(\mathbf{u})$ into (4) and into (9), leaving us with 3 equations in $\mathbf{u}$. We input these equations into the generator in [17], obtaining a solver with a 66×82 elimination template and 16 solutions.

H32+s Here, we follow the same procedure as for **H13+s**, with the difference that we substitute $\mathtt{t}$ and s into 3 2D-2D constraint equations, instead of 2 2D-3D and only 1 2D-2D. Given that the 2D-2D constraints are of higher degree than the 2D-3D, we end up with a much more complex solver compared to **H13+s**. In this case, the elimination template is of size 212×276 and the system has 56 solutions.

H51+s In this case, we cannot directly use (7), and thus we use (8) with our only 2D-3D constraint in order to eliminate the first two elements of the translation. We then substitute $\mathtt{t}$ as per (8) into (9) with $j = 1 \dots 5$, which yields a very complex solver. For this derivation, we obtain a 549×781 elimination template with 80 solutions. According to our experiments, this is not an acceptable size of elimination template to deal with in 64-bit floating point precision, and thus this solver is deemed impractical.

uP3P+s For this solver we are given 3 2D-3D matches, as in (5) *and* a known vertical direction. Since we are able to parameterize $\mathtt{R}(a, b)$ as in (3), then (5) is purely linear. As with the solver in [32], we actually have one excess constraint. Thus, we may solve the system of 6 linear equations given by 3 2D-3D constraints for the 6 unknowns $\mathtt{t}$, $\hat{a}$, $\hat{b}$, s. This solution in general will not fulfill the nonlinear constraint $\hat{a}^2 + \hat{b}^2 = 1$. To enforce this, we simply divide $\hat{a}$ and $\hat{b}$ by $\hat{a}^2 + \hat{b}^2$. This is only an approximate solution, but in practice it yields good results (*c.f.* Sec. 5.1).

uH12+s For this solver we again employ the elimination of (7). We are then left with inserting the expression for $\mathtt{t}$ and s in terms of a and b into the remaining 2D-2D equation. This leads to a system of two polynomials: one of degree 2 resulting from the previous substitution, and the constraint of the rotation parameterization (3). The solver obtained has a 6×10 template, with an action matrix of size 4.

uH31+s Similarly to **H51+s**, we start by eliminating the first two elements of $\mathtt{t}$ using (8). Then, we substitute this expression into (9) with $j = 1, 2, 3$, leading to a system of 3 polynomials in terms of a, b, $\mathtt{t}_z$ and s. Adding the rotation parameterization constraint leads to an elimination template of size 30×36 and an eigen-decomposition of size 6.

4.2. Known Scale Solvers

H22 We begin by eliminating $\mathtt{t}$ using 3 equations out of the 4 given by our 2 2D-3D matches. Afterwards we simply substitute this into the remaining 2D-3D constraint plus the two 2D-2D constraints. This yields a system of 3 polynomials in $\mathbf{u}$. We get an elimination template of size 23×39 and 16 solutions.

H41 For this solver we use (8) with only 2D-3D constraint to eliminate two translation parameters. We then substitute back into 4 (9) constraints. Since, as mentioned, all of this equations are homogeneous, we multiply out the resulting polynomial system in $\mathbf{u}$ and $\mathtt{t}_z$ by the scale of the rotation. This simplifies the algebraic representation of the problem, however we are still left with a rather large elimination template of size 244×277 and an action matrix with up to 32 valid solutions.

uH21 As in the previous solver, we eliminate the last two elements of the translation. We arrive at two 2D-2D constraints in $\mathtt{t}_z$, a and b plus one constraint on the rotation parameterization. We solve this system of three polynomials using a 8×12 template yielding an action matrix with up to 4 solutions.

5. Results

First, we evaluate the numerical stability of the solvers. This is used not only to validate our derivations, but to guide our prior $P_p(s)$ (*c.f.* Sec. 3). Afterwards, we perform two different real-data evaluations for instances of **Problem 6-DoF** and **Problem 5-DoF**.

5.1. Numerical Stability of the Minimal Solvers

For each solver we generated 10^6 random synthetic scenes and compared the obtained best pose against ground truth. For each synthetic scene we generated 3D points in the $[-1, 1] \times [-1, 1] \times [2, 10]$ cube. Each synthetic camera was placed with a random center of projection in the range $[-1, 1] \times [-1, 1] \times [-1, 1]$. We then generated a random rotation and translation (and if applicable a random scale $\in [1, 10]$) and transformed the generated camera centers and 3D points. For each trial, we use enough matches to cover the requirements of each solver. For 2D-3D matches, we projected each transformed 3D point to the transformed frame and associated it with the original 3D generated point. For 2D-2D matches, we also project the 3D points to obtain 2D measurements in the query frame, while keeping the original camera and its projected observation in the global frame. In Fig. 2 we report the precision with which we were able to obtain the camera poses. We can see that all proposed solvers are stable enough to be used in practical applications (*e.g.*, within RANSAC). In order to asses the computational cost for our solvers, we also recorded the execution time and number of real-valued solutions (*c.f.* Table 2). For comparison, a typical P3P implementation has runtimes of about 2μs.

Solver	Time		Num. of Sols.	
	Mean	Median	Mean	Median
Implementations in MATLAB (times in [ms])				
uH21	3.2	2.6	3.2	4
H13+s	19.1	17.5	6.2	6
H32+s	2,693	2,617	20.9	22
Implementations in C++ (times in [μs])				
uP3P+s	3.2	3.1	1	1
H22	7.25	7.2	6.8	6
uH12+s	9.5	9.2	3.3	4
uH31+s	32.6	32.1	4.5	4
H41	756	728	12.7	12

Table 2. Execution times and number of solutions for the minimal solvers presented in this paper. C++ was used only for the minimal problems needed for the real-world experiments. **H51+s** is not included since no practical implementation was obtained.

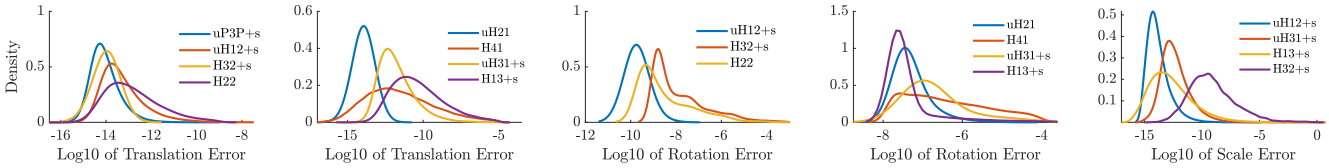

Figure 2. Numerical precision of the *nine* proposed solvers. Notice the scale changes in the X-axis. For more complex solvers (*i.e.*, larger elimination templates and more solutions, *c.f.* Sec. 4), we observe errors with a higher spread. Nevertheless, all our proposed solvers exhibit enough numerical accuracy to be useful for practical applications.

5.2. Real-data Experiments

To test our novel hybrid pose estimation framework along with our newly developed minimal solvers, we focus on **Problem 6-DoF** and **Problem 5-DoF** for the real-data experiments. We conduct these experiments by comparing our hybrid approach against the *pure* approaches for each problem. These experiments are meant to illustrate two different possible usages of our hybrid method. In the first experiment (*c.f.* Sec. 5.2.1), we address the issue of inaccurate pose estimates arising from inaccurately reconstructed 3D points. To do so, we ignore the badly triangulated 3D points and instead focus on the 2D measurements that produced them. In the second experiment (*c.f.* Sec. 5.2.2), we focus on increasing the number of correspondences by matching against all existing 3D points *plus* all the 2D observations from the SfM model which were *not* used to reconstruct any 3D point. This allows us to utilize information that is usually ignored (2D measurements in the SfM) when estimating the pose of a given camera.

Bear in mind that these results are meant to demonstrate that using hybrid matches can improve performance in real-world scenarios. As such, we do not focus on the matching procedure and thus we cannot directly compare our results to many state-of-the-art approaches, *e.g.*, [18, 22]. Nevertheless, our approach is agnostic to the matching procedure and it can be used as a drop-in replacement in many of these state-of-the-art methods.

5.2.1 Image-based Camera Localization

We employ the dataset presented in [23] as an instance of **Problem 6-DoF**. This dataset consists of an SfM model with 1.65M points, 4.3K model images and has 824 query images with ground truth poses. For each query image, we match all detected SIFT features against the SfM's 3D features using an approximate nearest neighbor search [20]. For the purely 2D-3D method, we simply take all 2D-3D correspondences to compute the pose of each image. For the 2D-2D method, for each 3D point we randomly select a camera in the model that reconstructed that point and use this camera and its original 2D measurement to recreate a 2D-2D match. Finally, for our hybrid method, we need to use both 2D-2D and 2D-3D matches. To do so, we select a subset of the initial 2D-3D matches and instead consider them as 2D-2D. To decide which 2D-3D matches we will

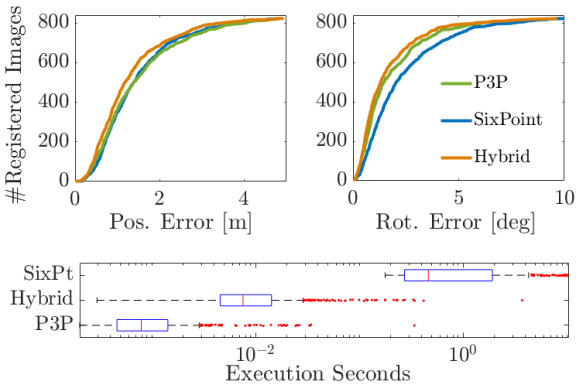

Figure 3. *Top* Comparison of the number of correctly estimated poses for the dataset in [23]. For a given threshold, we count the number of images with an error equal or below the threshold. *Bottom* Comparison of the execution time for our hybrid method compared to using either P3P [14] or SixPt [36] alone. All values are the average of 100 runs.

recast as 2D-2D, we use the covariance of the 3D points (obtained as part of the Bundle Adjustment procedure as approximated by the inverse Hessian). As in [2], we score each 3D point by the *roundness* of its covariance

$$r_k = \sqrt{\frac{\lambda_3}{\lambda_1}} \ , \tag{10}$$

where λ_i is the i-th singular value of the 3×3 (up-to-scale) covariance matrix of point k. Using this score, we then simply take 50% of the matches with highest roundness score as 2D-3D and consider the rest as 2D-2D. By applying this procedure we aim to reduce the impact of badly triangulated 3D points in the accuracy of our estimated pose.

In Fig. 3 we show the precision of our method compared to using each match type separately. For this dataset, we focus on the *registration rate* as a function of the retrieved pose precision. For a given position (or rotation) threshold, we count the number of images which have a position (or rotation) error equal or below such threshold. The poses used to compute the precision against the ground truth were obtained from the output of RANSAC directly, without performing any pose refinement. Note that, interestingly, our hybrid approach is able to increase the precision of the retrieved poses across all images. Furthermore, in Fig. 3 we can see that the average runtime of our method does not significantly differ from the less accurate P3P. Additionally in order to validate our Hybrid RANSAC solver selection scheme, we measured how often a particular solver finds

Dataset		Using 2D2D Matches							Using 2D3D Matches							Hybrid						
		Inlier Ratio		Time [ms]	Position Error [cm]				Inlier Ratio		Time [ms]	Position Error [cm]				Inlier Ratio		Time [ms]	Position Error [cm]			
					Quantiles			Mean				Quantiles			Mean				Quantiles			Mean
Name	# Images	2D2D	2D3D		25%	50%	75%		2D2D	2D3D		25%	50%	75%		2D3D	2D3D		25%	50%	75%	
Office1	9	-	-	-	-	-	-	-	0	0.657	**13.1**	4.7	5.35	7.4	6.3	0.01	0.66	21.1	4.8	5.21	7.5	**6.27**
Office2	9	-	-	-	-	-	-	-	0	0.56	**13.2**	5.2	5.9	6.5	**5.6**	0.1	0.65	19	5.1	5.9	6.5	**5.6**
Office3	33	0.54	0	169	3.5	4.4	5.2	4.5	0	0.29	272	3.9	5.3	5.9	5.8	0.46	0.33	**110**	3.2	3.8	4.5	**4.0**
Office4	9	0.74	0	25.3	4.1	5.4	7.7	**6.3**	0	0.51	21	4.3	5.83	7.9	6.6	0.72	0.54	**16.4**	4.1	5.4	8.0	6.5
Office5	15	0.58	0	115.3	3.5	5.3	7.2	5.5	0	0.31	151	4.1	5.4	6.8	5.54	0.34	0.35	**85**	3.6	4.2	5.7	**4.3**
Office6	24	0.07	0	983	4.4	5.3	6.3	**5.4**	0	0.53	**9.68**	3.4	3.7	4.0	3.8	0.04	0.54	19.1	3.4	3.5	4.1	**3.5**
Office7	9	0.70	0	51.4	3.9	4.52	5.57	**4.7**	0	0.25	125	4.4	5.1	5.8	5.1	0.63	0.27	**41.9**	4.2	4.7	5.1	**4.7**
Office8	11	0.47	0	137	3.8	4.46	5.6	4.7	0	0.41	75	4.1	4.8	5.4	4.9	0.42	0.43	**48**	4.0	4.5	5.5	**4.5**
Office9	7	0.21	0	279	5.9	6.2	6.4	7.1	0	0.49	**8.3**	4.1	5.2	6.5	5.6	0.22	0.51	15.3	4.3	5.3	5.9	**5.4**
Office10	23	0.39	0	151	5.2	6.0	6.7	6.1	0	0.45	15.7	4.7	5.7	6.1	5.8	0.38	0.45	**11.4**	4.8	5.5	5.9	**5.5**
Office11	58	0.78	0	21.3	4.4	5.1	5.5	5.0	0	0.47	**9.9**	4.3	4.9	5.57	5.1	0.77	0.47	**10**	3.8	4.3	4.9	**4.3**

Table 3. Performance on the dataset by [33], consisting of 12 image sequences and a point cloud with SIFT descriptors. As in [5, 30, 32], the position error is computed using the output of RANSAC without refinement. Notice that our hybrid approach performs better both in runtime and accuracy than using each of the matches separately. For two sequences (1 and 2), there was not enough visual overlap with "Office12" in order to localize using only 2D-2D matches.

Method	Inlier Count					
	Mean	Quantiles				
		25%	50%	75%	90%	95%
Hybrid	**260**	**74.5**	**175**	**372**	**632**	**804**
P3P	239	72.5	167.5	349	583	690
SixPoint	246	71	174	354	609	724

Table 4. Statistics on the number of inliers for the experiments on the dataset for [23]. Our hybrid method is able to consistently achieve a larger number of inliers. Results averaged over 100 runs.

the highest inlier consensus per image. For 824 images, **H22** finds the best model 214 times; **H41** finds the best model 245 times, P3P finds it 197 times and finally SixPt finds it 168 times. Thus, depending on the image being localized, one solver is more suited than another. Regarding the number of inliers obtained by our method, we measured the mean number of inliers across all images, *c.f*. Table 4.

5.2.2 SLAM Trajectory Registration

In this experiment we aim to measure the performance of our framework applied to **Problem 5-DoF**. This type of problem is particularly relevant for SfM registration, where we have two reconstructions (with different scales) which we aim to merge into one. We used the dataset in [33], which consists of 12 video sequences of an office indoor scene, as well as a 3D reconstruction of the scene obtained using SfM. Additionally, each of the 12 sequences were ran through a SLAM system, thus relative local poses within each sequence are known. An optical tracker was used to obtain accurate *positional* ground truth of each of the sequences in the dataset. Similarly to Sec. 5.2.1, we first obtain 2D-3D matches for each sequence against the model's 3D points. In contrast to the previous experiment, however, we cannot use the original reconstruction's cameras to obtain 2D-2D matches since those camera poses are not available in the dataset. Instead, we follow a similar procedure to [30]. We first take one of the sequences, "Office12", and use its ground truth pose to align it to the global frame of reference. We then compute 2D-3D matches for this sequence, and non-linearly refine the poses of the sequence by minimizing the reprojection error of the 2D-3D matches. This results in an augmented dataset, where in addition to the original set of 3D points we have a camera sequence registered to these 3D points. This allows us to then match against all original 3D points *and* the 2D observations of "Office12". Then, for the purely 2D-2D method, we consider all 2D observations of "Office12" to obtain matches. For our hybrid method, however, we match against the original 3D points to get 2D-3D matches, while we use the *unmatched* 2D observations in "Office12" (*i.e*., the image observations which are not matched to a 3D point) to obtain 2D-2D matches. Thus, for our solver we use the original 3D points *plus* the 2D measurements from "Office12" which would usually go unused.

In Table 3 we show the results of our method on an instance of **Problem 5-DoF**. For most sequences, our method is able to deliver an accuracy comparable to the state-of-art method by [30], while maintaining a runtime very close to the one achieved by the most simple 2D-3D solver.

6. Conclusions

In this paper, we have posed the question whether camera pose estimation can be improved by using both 2D-2D *and* 2D-3D matches. To answer this, we have developed a novel framework for camera pose estimation that jointly uses different minimal solvers within a new Hybrid RANSAC scheme. As needed by our Hybrid RANSAC scheme, we have derived several new minimal solvers using both 2D-2D and 2D-3D correspondences, which we evaluated on synthetic data in order to asses their numerical stability. We have evaluated our framework on two different real-world datasets, demonstrating that it generates more accurate poses compared to approaches using either 2D-2D *or* 2D-3D matches. One main strength of our approach is its ability to automatically select a suitable solver, allowing it to adapt to the quality of the provided matches. Overall, we have shown that there is a clear benefit in properly combining different types of matches for camera pose estimation. For future work, we plan to apply our framework to other geometric problems, such as uncalibrated camera pose estimation (*e.g*., by extending [24]) or rolling shutter scenarios.

Acknowledgements. This research was funded by Google.

References

[1] S. Agarwal1, N. Snavely, I. Simon, S. Seitz, and R. Szeliski. Building rome in a day. In *ICCV*, 2009. 1

[2] C. Beder and R. Steffen. Determining an Initial Image Pair for Fixing the Scale of a 3d Reconstruction from an Image Sequence. In *Joint Pattern Recognition Symposium*, 2006. 7

[3] M. Bujnak, Z. Kukelova, and T. Pajdla. Making Minimal Solvers Fast. In *CVPR*, 2012. 5

[4] F. Camposeco, A. Cohen, T. Sattler, A. Geiger, and M. Pollefeys. Toroidal Constraints for Two Point Localization Under High Outlier Ratios. In *CVPR*, 2017. 1

[5] F. Camposeco, T. Sattler, and M. Pollefeys. Minimal Solvers for Generalized Pose and Scale Estimation from Two Rays and One Point. In *ECCV*, 2016. 2, 3, 8

[6] B. Clipp, C. Zach, J.-M. Frahm, and M. Pollefeys. A New Minimal Solution to the Relative Pose of a Calibrated Stereo Camera with Small Field of View Overlap. In *ICCVL*, 2009. 2, 3

[7] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *PAMI*, 29:2007, 2007. 1

[8] M. Fischler and R. Bolles. Random Sampling Consensus: A Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography. *CACM*, 24, 1981. 1, 2

[9] C. Häne, L. Heng, G. H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys. 3D visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection. *Image and Vision Computing*, 2017. 1

[10] R. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle. Review and analysis of solutions of the three point perspective pose estimation problem. *IJCV*, 13(3), 1994. 1

[11] J. Heinly, J. L. Schönberger, E. Dunn, and J. M. Frahm. Reconstructing the world* in six days. In *CVPR*, 2015. 1

[12] K. Josephson and M. Byrod. Pose estimation with radial distortion and unknown focal length. In *CVPR*, 2009. 1, 2

[13] K. Josephson, M. Byrod, F. Kahl, and K. Astrom. Image-based localization using hybrid feature correspondences. In *CVPR*, pages 1–8. IEEE, 2007. 2, 4

[14] L. Kneip, D. Scaramuzza, and R. Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *CVPR*, 2011. 2, 4, 7

[15] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic Generator of Minimal Problem Solvers. In *ECCV*, 2008. 5

[16] Z. Kukelova, J. Heller, and A. Fitzgibbon. Efficient Intersection of Three Quadrics and Applications in Computer Vision. In *CVPR*, 2016. 2, 4

[17] K. Laurent. Polyjam Software. https://github.com/laurentkneip/polyjam, 2015. 5

[18] Y. Li, N. Snavely, D. P. Huttenlocher, and P. Fua. Worldwide Pose Estimation Using 3D Point Clouds. In *ECCV*, 2016. 1, 7

[19] S. Middelberg, T. Sattler, O. Untzelmann, and L. Kobbelt. Scalable 6-DOF Localization on Mobile Devices. In *ECCV*, 2014. 1

[20] M. Muja and D. G. Lowe. Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. In *VISAPP*, 2009. 7

[21] R. Pless. Using Many Cameras as One. In *CVPR*, 2003. 5

[22] T. Sattler, B. Leibe, and L. Kobbelt. Efficient Effective Prioritized Matching for Large-Scale Image-Based Localization. *PAMI*, 39(9):1744–1756, Sept 2017. 1, 7

[23] T. Sattler, W. Maddern, A. Torii, J. Sivic, T. Pajdla, M. Pollefeys, and M. Okutomi. Benchmarking 6DOF Urban Visual Localization in Changing Conditions. *arXiv:1707.09092*, 2017. 7, 8

[24] T. Sattler, C. Sweeney, and M. Pollefeys. On Sampling Focal Length Values to Solve the Absolute Pose Problem. In *ECCV*, 2014. 8

[25] J. L. Schönberger and J.-M. Frahm. Structure-From-Motion Revisited. In *CVPR*, June 2016. 1

[26] H. Stewénius, D. Nistér, M. Oskarsson, and K. Åström. Solutions to Minimal Generalized Relative Pose Problems. In *OMNIVIS*, 2005. 2, 4, 5

[27] L. Svärm, O. Enqvist, F. Kahl, and M. Oskarsson. City-scale Localization for Cameras with Known Vertical Direction. *PAMI*, 39(7):1455–1461, 2017. 1

[28] C. Sweeney, J. Flynn, B. Nuernberger, M. Turk, and T. Höllerer. Efficient Computation of Absolute Pose for Gravity-Aware Augmented Reality. In *ISMAR*, 2015. 2, 4

[29] C. Sweeney, J. Flynn, and M. Turk. Solving for Relative Pose With a Partially Known Rotation is a Quadratic Eigenvalue Problem. In *3DV*, 2014. 2, 4, 5

[30] C. Sweeney, L. Kneip, T. Hollerer, and M. Turk. Computing Similarity Transformations from Only Image Correspondences. In *CVPR*, 2015. 2, 4, 8

[31] B. Triggs. Camera pose and calibration from 4 or 5 known 3d points. In *ICCV*, volume 1, pages 278–284. IEEE, 1999. 1, 2

[32] J. Ventura, C. Arth, G. Reitmayr, and D. Schmalstieg. A minimal solution to the generalized pose-and-scale problem. In *CVPR*, 2014. 2, 4, 6, 8

[33] J. Ventura, C. Arth, G. Reitmayr, and D. Schmalstieg. Global Localization from Monocular SLAM on a Mobile Phone. *IEEE Transactions on Visualization and Computer Graphics*, 2014. 8

[34] C. Wu. P3.5P: pose estimation with unknown focal length. In *CVPR*, 2015. 1, 2

[35] B. Zeisl, T. Sattler, and M. Pollefeys. Camera Pose Voting for Large-Scale Image-Based Localization. In *ICCV*, 2015. 1

[36] E. Zheng and C. Wu. Structure from Motion Using Structure-less Resection. In *ICCV*, 2015. 1, 2, 3, 4, 5, 7