# Absolute Pose Estimation of Central Cameras Using Planar Regions

Robert Frohlich, Levente Tamas *Member, IEEE*, and Zoltan Kato *Senior Member, IEEE*

**Abstract**—A novel method is proposed for the absolute pose estimation of a central 2D camera with respect to 3D depth data without the use of any dedicated calibration pattern or explicit point correspondences. The proposed method has no specific assumption about the data source: plain depth information is expected from the 3D sensing device and a central camera is used to capture the 2D images. Both the perspective and omnidirectional central cameras are handled within a single generic camera model. Pose estimation is formulated as a 2D-3D nonlinear shape registration task which is solved without point correspondences or complex similarity metrics. It relies on a set of corresponding planar regions, and the pose parameters are obtained by solving an overdetermined system of nonlinear equations. The efficiency and robustness of the proposed method were confirmed on both large scale synthetic data and on real data acquired from various types of sensors.

**Index Terms**—Pose estimation, calibration, data fusion, registration, Lidar, omnidirectional camera

## 1 Introduction

Absolute pose estimation consists of determining the position and orientation of a camera with respect to a 3D world coordinate frame. It is a fundamental building block in various computer vision applications, such as robotics (*e.g.* visual odometry [1], localization and navigation [2]), augmented reality [3], geodesy, or cultural heritage [4]. The problem has been extensively studied yielding various formulations and solutions. Most of the approaches focus on a single perspective camera pose estimation using $n$ 2D–3D point correspondences, known as the *Perspective n Point* (PnP) problem [5], [6], [7]. It has been widely studied for large $n$ as well as for the minimal case of $n = 3$ (see [7] for a recent overview). Using line correspondences yields the *Perspective n Line* (PnL) problem [8], [9] (see [8] for a detailed overview). Several applications dealing with multimodal sensors make use of fused 2D radiometric and 3D depth information from uncalibrated cameras. The availability of 3D data has also became widespread. Classical image-based techniques, such as Structure from Motion (SfM) [10] provide 3D measurements of a scene, while modern range sensors (*e.g.* Lidar, ToF) record 3D structure directly. Therefore methods to estimate absolute pose of a camera based on 2D measurements of the 3D scene received more attention [7], [11], [12]. Many of these methods apply to general central cameras (both perspective and omnidirectional) that are often represented by a unit sphere [13], [14], [15], [16].

Pose estimation (also known as *external calibration*) of 2D color and 3D depth cameras was performed especially for environment mapping applications [17]. While internal calibration can be solved in a controlled environment, *e.g.* using special calibration patterns, pose estimation must rely on the actual images taken in a real environment. Popular methods rely on point correspondences such as [18], or using fiducial markers [19], which may be cumbersome to use in real life situations. This is especially true in a multimodal setting, where omnidirectional images need to be combined with other non-conventional sensors like Lidar scans providing range only data. The Lidar-omnidirectional camera calibration problem was analyzed from different perspectives. In [20], the calibration is performed in natural scenes, however point correspondences between the 2D-3D images are selected in a semi-supervised manner. In [21], calibration is tackled as an observability problem using a (planar) fiducial marker as calibration pattern. In [22] a fully automatic method is proposed based on mutual information (MI) between the intensity information from the depth sensor and the omnidirectional camera, while in [23], [24] a deep learning approach for calibration is presented. Another global optimization method uses the gradient orientation measure as described in [17]. However, these methods require range data with recorded intensity values, which are not always available. In real life applications, it is also often desirable to have a flexible one step calibration for systems which do not necessarily contain sensors fixed to a common platform.

In this work we propose a straightforward absolute pose estimation method which overcomes the majority of these limitations, *i.e.* by not using any artificial marker or intensity information from the depth data. Instead, our


- *R. Frohlich and Z. Kato are with the Department of Image Processing and Computer Graphics, University of Szeged, P.O. Box 652, H-6701 Szeged, Hungary. Fax:+36 62 546 397, Tel:+36 62 546 399. E-mail: {frohlich,kato@inf.u-szeged.hu}*
- *Z. Kato is with the Department of Mathematics and Informatics, J. Selye University, Komarno, Slovakia.*
- *L. Tamas (Corresponding author) is with the Automation Department from the Technical University of Cluj-Napoca, Memorandumului st 28, 400114 RO, Tel/fax: +40 264 401 586; and with the Department of Electrical Engineering and Information Systems, University of Pannonia, Veszprem, HU. E-mail: Levente.Tamas@aut.utcluj.ro*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



method makes use of a segmented planar region from the 2D and 3D visual data and handles the absolute pose estimation problem as a nonlinear registration task. More specifically, inspired by the 2D registration framework presented in [25], for the central camera model we construct an overdetermined set of equations containing the unknown camera pose. By solving this system of equation we obtain the required set of parameters representing the camera pose.

### 1.1 Related work

Due to the large number of applications using central camera systems, also the range of the calibration methods is rather wide. Beside solving the generic 2D-3D registration problem, several derived applications exists including medical [26], robotics [17] and cultural heritage ones [4]. For the pose estimation in known environment a good example can be found in [27], while in [28] an application is reported using spherical image fusion with spatial data. A more generic classification of the types of algorithms is presented in [29]. Beside the direct measured relative pose methods such as [30], a number of generic methods are summarized below.

Several *feature based methods* based on specific markers are used for extrinsic camera calibration [31], [32]. In the early work of [33], alignment based on a minimal number of point correspondences is proposed, while in [34], a large number of 2D-3D correspondences are used with possibly redundant or mismatched pairs. [35] was among the first addressing the extrinsic calibration of 3D Lidar and low resolution perspective color camera, which generalized the algorithm proposed in [36]. This method is based on manual point feature selection from both domains and assumes a valid camera intrinsic calibration. A similar manual point feature correspondence based approach is proposed in [20]. Recently, increasing interest is manifested in various calibration setups ranging from high-resolution spatial data [37] to low-resolution commercial cameras [38], as well as online calibration of depth and color sensors on a moving platform [22], [39].

A popular alternative to feature based matching is the color-intensity *mutual information (MI)* alignment between the 2D color image and the 3D data with intensity information such [40], [17]. Extensions to the simultaneous intrinsic-extrinsic calibration are presented in [21], which makes use of Lidar intensity information to find correspondences between the 2D-3D data. Other works are based on the fusion of IMU or GPS information for calibration [41].

A good overview of *statistical methods* based calibration methods can be found in [26]. Mutual information and particle filters are used in [17], which performs pose estimation using the whole image space of a single 2D-3D observation. The method can use both intensity and normal distribution information for the 3D data. A further extension of this approach based on gradient orientation measure is described in [42]. A gradient information extraction and global matching between the 2D color and 3D reflectivity information is presented in [22]. This has two major differences compared to our work. Our approach is not limited to Lidar systems with reflectivity information rather it is based only on depth information. On the optimization side, the proposed method is not restricted to convex problems and allows camera calibration using only a single Lidar-camera image pair.

An early and efficient *silhouette based registration* method is presented in [43], which solves a model-based vision problem using parametric description of the model. This method can be used with an arbitrary number of parameters describing the object model and is based on global optimization with the *Levenberg-Marquardt* method. A whole object silhouette based registration is proposed in [40], where the authors describe the 2D-3D registration pipeline including segmentation, pixel level similarity measure and global optimization. Although the proposed method can be used in an automatic manner, this is limited only to scenes with highly separable foreground-background parts. By an automatic segmentation of the relevant forms in panoramic images, which are registered against cadastral 3D models the segmented regions are aligned using particle swarm optimization in [44]. An extension of silhouette based registration is proposed in [45], where a hybrid silhouette and key-point driven approach is used for the registration of the 2D and 3D data. The advantage of this method is the possibility of multiple image registration as well as precise output of the algorithm.

### 1.2 Contributions

Instead of establishing 2D-3D point matches, relying on artificial markers or recorded intensity values, we propose a pose estimation algorithm which works on corresponding segmented 2D-3D regions. Since segmentation is anyway required in many real-life image analysis tasks, such regions may be available or straightforward to detect. Inspired by [25], we reformulate pose estimation as a shape alignment problem. The solution is obtained by constructing a system of non-linear equations, which is solved in the least squares sense by a standard *Levenberg-Marquardt* algorithm. The result represents the estimates for the unknown pose parameters. We formulate the problem as the pose estimation of an universal central camera, which includes omnidirectional as well as perspective cameras. Our method was quantitatively evaluated on a large synthetic dataset and proved to be robust and efficient on real data too. For the real tests we used both publicly available dataset, as well as our own captured data.

## 2 Region-based Pose Estimation

Pose estimation consists in computing the position and orientation of a camera with respect to a 3D world coordinate system $\mathcal{W}$. Herein, we are interested in central cameras, where the projection rays intersect in a single point called projection center or single effective viewpoint. Typical examples include omnidirectional cameras as well as traditional perspective cameras. A broadly used unified model for central cameras represents a camera as a projection onto the surface of a unit sphere $\mathcal{S}$ (see Fig. 1) [13],

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



[14], [15], [16], the projection center being the center of the sphere. The camera coordinate system $\mathcal{C}$ is in $\mathcal{S}$, the origin is the effective viewpoint (which is also the center of the sphere) and the $Z$ axis is the optical axis of the camera which intersects the image plane in the *principal point*. The absolute pose of our central camera is defined as the rigid transformation $(\mathbf{R}, \mathbf{t}) : \mathcal{W} \to \mathcal{C}$ acting between the world coordinate frame $\mathcal{W}$ and the camera coordinate frame $\mathcal{C}$, while the internal projection function of the camera defines how 3D points are mapped from $\mathcal{C}$ onto the image plane $\mathcal{I}$.

Let us first see the relationship between a point $\mathbf{x} = [x_1, x_2]^\top \in \mathbb{R}^2$ in the image $\mathcal{I}$ and its representation $\mathbf{x}_\mathcal{S} = [x_{\mathcal{S},1}, x_{\mathcal{S},2}, x_{\mathcal{S},3}]^\top \in \mathbb{R}^3$ on the unit sphere $\mathcal{S}$ (see Fig. 1). Note that only the half sphere on the image plane side is actually used, as the other half is not visible from image points. There are several well known geometric models for the internal projection [13], [14], [15], [16]. Following [16], the nonlinear (but symmetric) distortion of central omnidirectional cameras is represented by a surface $g$ between the image plane and the unit sphere $\mathcal{S}$, which is rotationally symmetric around $Z$. Herein, as suggested by [16], we will use a fourth order polynomial

$$g(\|\mathbf{x}\|) = a_0 + a_2\|\mathbf{x}\|^2 + a_3\|\mathbf{x}\|^3 + a_4\|\mathbf{x}\|^4, \qquad (1)$$

which has 4 parameters $(a_0, a_2, a_3, a_4)$ representing the internal parameters of the camera. Note that for a perspective camera $g$ is a plane – this special case will be discussed later in Section 2.2. The bijective mapping $\Phi : \mathcal{I} \to \mathcal{S}$ is composed of

1) lifting the image point $\mathbf{x} \in \mathcal{I}$ onto the $g$ surface by an orthographic projection

$$\mathbf{x}_g = \begin{bmatrix} \mathbf{x} \\ a_0 + a_2\|\mathbf{x}\|^2 + a_3\|\mathbf{x}\|^3 + a_4\|\mathbf{x}\|^4 \end{bmatrix} \qquad (2)$$

2) then centrally projecting the lifted point $\mathbf{x}_g$ onto the surface of the unit sphere $\mathcal{S}$:

$$\mathbf{x}_\mathcal{S} = \Phi(\mathbf{x}) = \frac{\mathbf{x}_g}{\|\mathbf{x}_g\|} \qquad (3)$$

Thus the camera projection is fully described by means of unit vectors $\mathbf{x}_\mathcal{S}$ in the half space of $\mathbb{R}^3$.

The projection of a 3D world point $\mathbf{X} = [X_1, X_2, X_3]^\top \in \mathbb{R}^3$ in the generalized spherical camera is basically a central projection onto $\mathcal{S}$ taking into account the extrinsic pose parameters ($\mathbf{R}$, $\mathbf{t}$). Thus for a world point $\mathbf{X}$ and its image $\mathbf{x} \in \mathcal{I}$, the following holds on the surface of $\mathcal{S}$:

$$\Phi(\mathbf{x}) = \mathbf{x}_\mathcal{S} = \Psi(\mathbf{X}) = \frac{\mathbf{R}\mathbf{X} + \mathbf{t}}{\|\mathbf{R}\mathbf{X} + \mathbf{t}\|} \qquad (4)$$

A classical solution of the absolute pose problem is to establish a set of 2D-3D point matches using *e.g.* a special calibration target [38], [21], or feature-based correspondences and then solve for $(\mathbf{R}, \mathbf{t})$ via the minimization of some error function based on (4). However, in many practical applications, it is not possible to use a calibration target and most 3D data (*e.g.* point clouds recorded by a Lidar device) will only record depth information, which challenges feature-based point matching algorithms.

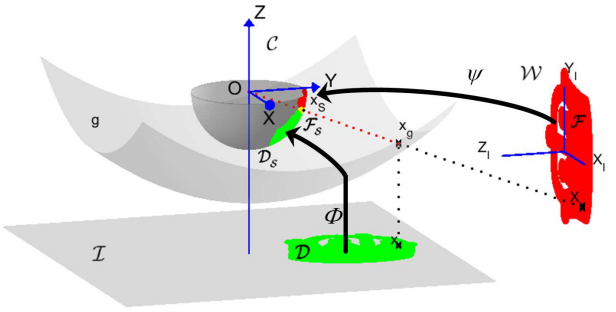


Fig. 1: Spherical camera model

Therefore the question naturally arises: what can be done when neither a special target nor point correspondences are available? Herein, we present a solution for such challenging situations. In particular, we will show that by identifying a single planar region both in 3D and the camera image, the absolute pose can be calculated. Of course, this is just the necessary minimal configuration. More such regions are available, a more stable pose is obtained. Our solution is inspired by the 2D shape registration approach of Domokos *et al.* [25], where the alignment of non-linear shape deformations are recovered via the solution of a special system of equations. Here, however, pose estimation yields a 2D-3D registration problem in case of a perspective camera and a restricted 3D-3D registration problem on the spherical surface for omnidirectional cameras. These cases thus require a different technique to construct the system of equations.

### 2.1 Absolute pose of spherical cameras

For spherical cameras, we have to work on the surface of the unit sphere as it provides a representation independent of the camera internal parameters. Furthermore, since correspondences are not available, (4) cannot be used directly. However, individual point matches can be integrated out yielding the following integral equation [46]:

$$\iint_{\mathcal{D}_\mathcal{S}} \mathbf{x}_\mathcal{S}\, \mathrm{d}\mathcal{D}_\mathcal{S} = \iint_{\mathcal{F}_\mathcal{S}} \mathbf{z}_\mathcal{S}\, \mathrm{d}\mathcal{F}_\mathcal{S}, \qquad (5)$$

where $\mathcal{D}_\mathcal{S}$ denotes the surface patch on $\mathcal{S}$ corresponding to the region $\mathcal{D}$ visible in the camera image $\mathcal{I}$, while $\mathcal{F}_\mathcal{S}$ is the surface patch of the corresponding 3D planar region $\mathcal{F}$ projected onto $\mathcal{S}$ by $\Psi$ in (4).

To get an explicit formula for the above surface integrals, the spherical patches $\mathcal{D}_\mathcal{S}$ and $\mathcal{F}_\mathcal{S}$ can be naturally parametrized via $\Phi$ and $\Psi$ over the planar regions $\mathcal{D}$ and $\mathcal{F}$. Without loss of generality, we can assume that the third coordinate of $\mathbf{X} \in \mathcal{F}$ is 0, hence $\mathcal{D} \subset \mathbb{R}^2$, $\mathcal{F} \subset \mathbb{R}^2$; and $\forall \mathbf{x}_\mathcal{S} \in \mathcal{D}_\mathcal{S} : \mathbf{x}_\mathcal{S} = \Phi(\mathbf{x}), \mathbf{x} \in \mathcal{D}$ as well as $\forall \mathbf{z}_\mathcal{S} \in \mathcal{F}_\mathcal{S} : \mathbf{z}_\mathcal{S} = \Psi(\mathbf{X}), \mathbf{X} \in \mathcal{F}$ yielding the following

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



form of (5) [46]:

$$\iint_{\mathcal{D}} \Phi(\mathbf{x}) \left\| \frac{\partial \Phi}{\partial x_1} \times \frac{\partial \Phi}{\partial x_2} \right\| \mathrm{d}x_1 \, \mathrm{d}x_2 = \iint_{\mathcal{F}} \Psi(\mathbf{X}) \left\| \frac{\partial \Psi}{\partial X_1} \times \frac{\partial \Psi}{\partial X_2} \right\| \mathrm{d}X_1 \, \mathrm{d}X_2 \quad (6)$$

where the magnitude of the cross product of the partial derivatives is known as the surface element. The above equation corresponds to a system of 2 equations only, because a point on the surface $\mathcal{S}$ has 2 independent components. However, we have 6 pose parameters (3 rotation angles and 3 translation components). To construct more equations, we adopt the general mechanism from [25] and apply a function $\omega : \mathbb{R}^3 \to \mathbb{R}$ to both sides of the equation (4), yielding

$$\iint_{\mathcal{D}} \omega(\Phi(\mathbf{x})) \left\| \frac{\partial \Phi}{\partial x_1} \times \frac{\partial \Phi}{\partial x_2} \right\| \mathrm{d}x_1 \, \mathrm{d}x_2 = \iint_{\mathcal{F}} \omega(\Psi(\mathbf{X})) \left\| \frac{\partial \Psi}{\partial X_1} \times \frac{\partial \Psi}{\partial X_2} \right\| \mathrm{d}X_1 \, \mathrm{d}X_2 \quad (7)$$

Adopting a set of nonlinear functions $\{\omega_i\}_{i=1}^{\ell}$, each $\omega_i$ generates a new equation yielding a system of $\ell$ independent equations. Hence we are able to generate sufficiently many equations. The pose parameters $(\mathbf{R}, \mathbf{t})$ are then simply obtained as the solution of the nonlinear system of equations (7). In practice, an overdetermined system is constructed, which is then solved by minimizing the algebraic error in the *least squares sense* via a standard *Levenberg-Marquardt* algorithm. Although arbitrary $\omega_i$ functions could be used, power functions are computationally favorable [25], [47] as these can be computed in a recursive manner:

$$\omega_i(\mathbf{x}_{\mathcal{S}}) = x_1^{l_i} x_2^{m_i} x_3^{n_i}, \quad \text{with } 0 \leq l_i, m_i, n_i \leq 2 \text{ and } l_i + m_i + n_i \leq 3 \quad (8)$$

Note that the left hand side of (7) is constant, hence it has to be computed only once, but the right hand side has to be recomputed at each iteration of the least squares solver as it involves the unknown pose parameters, which is computationally rather expensive for larger regions. Therefore, in contrast to [46] where the integrals on the 3D side in (7) were calculated over all points of the 3D region, here we consider a triangular mesh representation $\mathcal{F}^{\triangle}$ of the 3D planar region $\mathcal{F}$. Due to this representation, we only have to apply $\Psi$ to the vertices $\{\mathbf{V}_i\}_{i=1}^{V}$ of the triangles in $\mathcal{F}^{\triangle}$, yielding a triangular representation of the spherical region $\mathcal{F}_{\mathcal{S}}^{\triangle}$ in terms of *spherical triangles*. The vertices $\{\mathbf{V}_{\mathcal{S},i}\}_{i=1}^{V}$ of $\mathcal{F}_{\mathcal{S}}^{\triangle}$ are obtained as

$$\forall i = 1, \ldots, V : \quad \mathbf{V}_{\mathcal{S},i} = \Psi(\mathbf{V}_i) \quad (9)$$

Due to this spherical mesh representation of $\mathcal{F}_{\mathcal{S}}$, we can rewrite the integral on the right hand side of (7) adopting $\omega_i$ from (8), yielding the following system of 17 equations:

$$\iint_{\mathcal{D}} \Phi_1^{l_i}(\mathbf{x}) \Phi_2^{m_i}(\mathbf{x}) \Phi_3^{n_i}(\mathbf{x}) \left\| \frac{\partial \Phi}{\partial x_1} \times \frac{\partial \Phi}{\partial x_2} \right\| \mathrm{d}x_1 \, \mathrm{d}x_2 \approx \sum_{\forall \triangle \in \mathcal{F}_{\mathcal{S}}^{\triangle}} \iint_{\triangle} z_{\mathcal{S},1}^{l_i} z_{\mathcal{S},2}^{m_i} z_{\mathcal{S},3}^{n_i} \, \mathrm{d}\mathbf{z}_{\mathcal{S}}, \quad (10)$$

where $\Phi = [\Phi_1, \Phi_2, \Phi_3]^\top$ denote the coordinate functions of $\Phi : \mathcal{I} \to \mathcal{S}$. Thus only the triangle vertices need to be projected onto $\mathcal{S}$, and the integral over these spherical triangles is calculated using the method presented in [48]. In our experiments, we used the Matlab implementation of John Burkardt available from https://people.sc.fsu.edu/~jburkardt/m_src/sphere_triangle_quad/sphere_triangle_quad.html.

The pose parameters are obtained by solving the system of equations (10) in the least squares sense. For an optimal estimate, it is important to ensure numerical normalization and a proper initialization. In contrast to [25], where this was achieved by normalizing the input pixel coordinates into the unit square in the origin, in the above equation all point coordinates are on the unit sphere, hence data normalization is implicit. To guarantee an optimal least squares solution, initialization of the pose parameters is also important. In our case, a good initialization ensures that the surface patches $\mathcal{D}_{\mathcal{S}}$ and $\mathcal{F}_{\mathcal{S}}$, as shown in Fig. 1, overlap as much as possible. How to achieve this?

### 2.1.1 Initialization

The 3D data is given in the world coordinate frame $\mathcal{W}$, which may have an arbitrary orientation, that we have to roughly align with our camera. Thus the first step is to ensure that the camera is looking at the correct face of the surface in a correct orientation. This is achieved by applying a rotation $\mathbf{R}_0$ that aligns the normal of the 3D region $\mathcal{F}^{\triangle}$ with the $Z$ axis, *i.e.* $\mathcal{F}^{\triangle}$ will be facing the camera, since according to the camera model $-Z$ is the optical axis. Then we also apply a translation $\mathbf{t}_0$ that brings the centroid of $\mathcal{F}^{\triangle}$ into $[0, 0, -1]^\top$, which puts the region into the $Z = -1$ plane. This is necessary to ensure that the plane doesn't intersect $\mathcal{S}$ while we initialize the pose parameters in the next step.

If there is a larger rotation around the $Z$ axis, then the projected spherical patch $\mathcal{F}_{\mathcal{S}}^{\triangle}$ might be oriented very differently w.r.t. $\mathcal{D}_{\mathcal{S}}$. Using non-symmetric regions, this would not cause an issue for the iterative optimization to solve, but in other cases an additional apriori input might be needed, such as an approximate value for the vertical direction in the 3D coordinate system, which could be provided by different sensors, or might be specified for a dataset captured with a particular setup. Based on this extra information, we apply a rotation $\mathbf{R}_{\mathbf{z}}$ around the $Z$ axis that will roughly align the vertical direction to the camera's $X$ axis, ensuring a correct vertical orientation of the projection.

To guarantee an optimal least squares solution, initialization of the pose parameters is also important, which ensures

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



that the surface patches $\mathcal{D}_\mathcal{S}$ and $\mathcal{F}_\mathcal{S}^\triangle$ overlap as much as possible. This is achieved by computing the centroids of $\mathcal{D}_\mathcal{S}$ and $\mathcal{F}_\mathcal{S}^\triangle$, and initializing $\mathbf{R}$ as the rotation between them. Translation of the planar region $\mathcal{F}^\triangle$ along the direction of its normal vector will cause a scaling of $\mathcal{F}_\mathcal{S}^\triangle$ on the spherical surface. Hence an initial $\mathbf{t}$ is determined by translating $\mathcal{F}^\triangle$ along the axis going through the centroid of $\mathcal{F}_\mathcal{S}^\triangle$ such that the area of $\mathcal{F}_\mathcal{S}^\triangle$ becomes approximately equal to that of $\mathcal{D}_\mathcal{S}$.

**Algorithm 1** Absolute pose estimation algorithm for spherical cameras.

**Input:** The coefficients of $g$, 3D (triangulated) region $\mathcal{F}^\triangle$ and corresponding 2D region $\mathcal{D}$ as a binary image.
**Output:** The camera pose.

1: Produce the spherical patch $\mathcal{D}_\mathcal{S}$ from $\mathcal{D}$ using (3).
2: Produce $\mathcal{F}_\mathcal{S}^\triangle$ by prealigning $\mathcal{F}^\triangle$ as described in Section 2.1.1 using $(\mathbf{R_0}, \mathbf{t_0})$ and then $\mathbf{R_z}$, then back-projecting it onto the unit sphere $\mathcal{S}$ using (9).
3: Initialize $\mathbf{R}$ from the centroids of $\mathcal{D}_\mathcal{S}$ and $\mathcal{F}_\mathcal{S}^\triangle$ as in Section 2.1.1.
4: Initialize $\mathbf{t}$ by translating $\mathcal{F}^\triangle$ until the area of $\mathcal{F}_\mathcal{S}^\triangle$ and $\mathcal{D}_\mathcal{S}$ are approximately equal (see Section 2.1.1).
5: Construct the system of equations (10) and solve it for $(\mathbf{R}, \mathbf{t})$ using the *Levenberg-Marquardt* algorithm.
6: The absolute camera pose is then given as the composition of the transformations $(\mathbf{R_0}, \mathbf{t_0})$, $\mathbf{R_z}$, and $(\mathbf{R}, \mathbf{t})$.

The steps of the proposed algorithm for central spherical cameras using coplanar regions is summarized in Algorithm 1. For two or more non-coplanar regions, the algorithm starts similarly, by first using only one region pair for an initial pose estimation, as described in Algorithm 1. Then, starting from the obtained pose as an initial value, the system of equations is solved for all the available regions, which provides an overall optimal pose.

## 2.2 Absolute pose of perspective cameras

A classical perspective camera sees the homogeneous world point $\tilde{\mathbf{X}} = [X_1, X_2, X_3, 1]^\top$ as a homogeneous point $\tilde{\mathbf{x}} = [x_1, x_2, 1]^\top$ in the image plain obtained by a perspective projection $\mathbf{P}$:

$$\tilde{\mathbf{x}} \cong \mathbf{P}\tilde{\mathbf{X}} = \mathbf{K}[\mathbf{R}|\mathbf{t}]\tilde{\mathbf{X}}, \tag{11}$$

where '$\cong$' denotes the equivalence of homogeneous coordinates, *i.e.* equality up to a non-zero scale factor; and $\mathbf{P}$ is the $3 \times 4$ camera matrix, which can be factored into the well known $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$ form, where $\mathbf{K}$ is the $3 \times 3$ upper triangular *calibration* matrix containing the camera intrinsic parameters, while $[\mathbf{R}|\mathbf{t}]$ is the absolute pose aligning the world coordinate frame $\mathcal{W}$ with the camera frame $\mathcal{C}$.

As a central camera, the perspective camera can be represented by the spherical camera model presented in the previous section: Since we assume a calibrated camera, we can multiply both sides of (11) by $\mathbf{K}^{-1}$, yielding the normalized inhomogeneous image coordinates $\mathbf{x} = [x_1, x_2]^\top \in \mathbb{R}^2$:

$$\mathbf{x} \leftarrow \mathbf{K}^{-1}\tilde{\mathbf{x}} \cong \mathbf{K}^{-1}\mathbf{P}\tilde{\mathbf{X}} = [\mathbf{R}|\mathbf{t}]\tilde{\mathbf{X}}, \tag{12}$$

Denoting the normalized image by $\mathcal{I}$, the surface $g$ in (1) will be $g \equiv \mathcal{I}$ , hence the bijective mapping $\Phi : \mathcal{I} \to \mathcal{S}$ for a perspective camera becomes simply the unit vector of $\mathbf{x}$:

$$\mathbf{x}_\mathcal{S} = \Phi(\mathbf{x}) = \frac{\mathbf{x}}{\|\mathbf{x}\|} \tag{13}$$

Starting from the above spherical representation of our perspective camera, the whole method presented in the previous section applies without any change. However, it is computationally more favorable to work on the normalized image plane $\mathcal{I}$, because this way we can work with plain double integrals on $\mathcal{I}$ instead of surface integrals on $\mathcal{S}$. Hence applying a nonlinear function $\omega : \mathbb{R}^2 \to \mathbb{R}$ to both sides of (12) and integrating out individual point matches, we get [47]

$$\int_\mathcal{D} \omega(\mathbf{x})\, \mathrm{d}\mathbf{x} = \int_{[\mathbf{R}|\mathbf{t}]\mathcal{F}} \omega(\mathbf{z})\, \mathrm{d}\mathbf{z}. \tag{14}$$

where $\mathcal{D}$ corresponds to the region visible in the normalized *camera* image $\mathcal{I}$ and $[\mathbf{R}|\mathbf{t}]\mathcal{F}$ is the image of the corresponding *3D planar region* projected by the normalized camera matrix $[\mathbf{R}|\mathbf{t}]$. Adopting a set of nonlinear functions $\{\omega_i\}_{i=1}^\ell$, each $\omega_i$ generates a new equation yielding a system of $\ell$ independent equations. Choosing power functions for $\omega_i$ [47]

$$\omega_i(\mathbf{x}) = x_1^{n_i} x_2^{m_i}, \quad 0 \le n_i, m_i \le 3 \text{ and } (n_i + m_i) \le 4, \tag{15}$$

and using a triangular mesh representation $\mathcal{F}^\triangle$ of the 3D region $\mathcal{F}$, we can adopt an efficient computational scheme. First, let us note that this particular choice of $\omega_i$ yields 13 equations, each containing the 2D geometric moments of the projected 3D region $[\mathbf{R}|\mathbf{t}]\mathcal{F}$. Therefore, we can rewrite the integral over $[\mathbf{R}|\mathbf{t}]\mathcal{F}^\triangle$ adopting $\omega_i$ from (8) as [47]

$$\int_\mathcal{D} x_1^{n_i} x_2^{m_i} d\mathbf{x} = \int_{[\mathbf{R}|\mathbf{t}]\mathcal{F}} z_1^{n_i} z_2^{m_i}\, \mathrm{d}\mathbf{z} \approx \sum_{\forall \triangle \in [\mathbf{R}|\mathbf{t}]\mathcal{F}^\triangle} \int_\triangle z_1^{n_i} z_2^{m_i}\, \mathrm{d}\mathbf{z}. \tag{16}$$

The latter approximation is due to the approximation of $\mathcal{F}$ by the discrete mesh $\mathcal{F}^\triangle$. The integrals over the triangles are various geometric moments which can be computed using efficient recursive formulas discussed hereafter.

Since many applications deal with 3D objects represented by a triangulated mesh surface, the efficient calculation of geometric moments is well researched for 3D [49], [50]. In the 2D case, however, most of the works concentrate on the geometric moments of simple digital planar shapes [51], [52], [53], and less work is addressing the case of triangulated 2D regions, with the possibility to calculate the geometric moments over the triangles of the region.

Since in our method we have a specific case, where a 3D triangulated region $\mathcal{F}^\triangle$ is projected onto the 2D image plane $\mathcal{I}$, where we need to calculate integrals over the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



regions $\mathcal{D} \subset \mathcal{I}$ and $[\mathbf{R}|\mathbf{t}]\mathcal{F}^{\triangle} \subset \mathcal{I}$, we can easily adopt the efficient recursive formulas proposed for geometric moments calculation over triangles in 3D and apply them to our 2D regions: Since our normalized image plane $\mathcal{I}$ is at $Z = 1$, the $Z$ coordinate of the vertex points is a constant 1, hence the generic 3D formula for the $(i, j, k)$ geometric moment of a surface $S$ [49] becomes a plain 2D moment in our specific planar case:

$$M_{ijk} = \int_S x^i y^j z^k \, \mathrm{d}S = \int_S x^i y^j \, \mathrm{d}x \, \mathrm{d}y \tag{17}$$

as the last term of $M_{ijk}$ will always be 1 regardless of the value of $k$. $i$ and $j$ are integers such that $i + j = N$ is the order of the moment.

Let us now see how to compute the integral on the right hand side of (16). The projected triangulated planar surface $[\mathbf{R}|\mathbf{t}]\mathcal{F}^{\triangle}$ consists of triangles $T$ defined by vertices $(\mathbf{a}, \mathbf{b}, \mathbf{c})$ that are oriented counterclockwise. The integral over this image region is simply the sum of the integrals over the triangles. Analytically, the integral over a triangle can be written as [50], [54]

$$\int_T z_1^i z_2^j \, \mathrm{d}\mathbf{z} = \frac{2\text{area}(T) i! j!}{(i + j + 2)!} S_{ij}(T), \tag{18}$$

where

$$S_{ij}(T) = \sum_{(i_1+i_2+i_3=i)} \sum_{(j_1+j_2+j_3=j)} \Big( \frac{(i_1 + j_1)!}{i_1! j_1!} a_1^{i_1} a_2^{j_1} \frac{(i_2 + j_2)!}{i_2! j_2!} b_1^{i_2} b_2^{j_2} \frac{(i_3 + j_3)!}{i_3! j_3!} c_1^{i_3} c_2^{j_3} \Big). \tag{19}$$

Substituting (18) into (16), we get

$$\sum_{\forall T \in [\mathbf{R}|\mathbf{t}]\mathcal{F}^{\triangle}} \int_T z_1^i z_2^j \, \mathrm{d}\mathbf{z} = 2 \frac{i! j!}{(i + j + 2)!} \sum_T \text{area}(T) S_{ij}(T) \tag{20}$$

where the signed area of triangle $T$ is calculated as the magnitude of the cross product of two edges:

$$\text{area}(T) = \frac{1}{2} \|(\mathbf{b} - \mathbf{a}) \times (\mathbf{c} - \mathbf{a})\| \tag{21}$$

As shown by [49] and then by [50], the computational complexity of the term $S_{ij}(T)$ can be greatly reduced from order $O(N^9)$ to order $O(N^3)$. Based on the final generating equations proposed by [50], we can write our generating equations for 2D domain as

$$S_{ij}(T) = \begin{cases} 0 & \text{if } i < 0 \text{ or } j < 0 \\ 1 & \text{if } i = j = 0 \\ a_1 S_{i-1,j}(T) + a_2 S_{i,j-1}(T) & \\ \quad + D_{ij}(\mathbf{b}, \mathbf{c}) & \text{otherwise} \end{cases} \tag{22}$$

with

$$D_{ij}(\mathbf{b}, \mathbf{c}) = \begin{cases} 0 & \text{if } i < 0 \text{ or } j < 0 \\ 1 & \text{if } i = j = 0 \\ b_1 D_{i-1,j}(\mathbf{b}, \mathbf{c}) + b_2 D_{i,j-1}(\mathbf{b}, \mathbf{c}) & \\ \quad + C_{ij}(\mathbf{c}) & \text{otherwise} \end{cases} \tag{23}$$

and

$$C_{ij}(\mathbf{c}) = \begin{cases} 0 & \text{if } i < 0 \text{ or } j < 0 \\ 1 & \text{if } i = j = 0 \\ c_1 C_{i-1,j}(\mathbf{c}) + c_2 C_{i,j-1}(\mathbf{c}) & \text{otherwise} \end{cases} \tag{24}$$

Using only the equations (22)–(24), we can thus perform the exact computation of the contribution of every triangle to all the geometric moments of the image region in a very efficient way. The different quantities $C_{ij}(\mathbf{c})$, $D_{ij}(\mathbf{b}, \mathbf{c})$, and $S_{ij}(T)$ are computed at order $N$ from their values at order $N - 1$ using the recursive formulas given above and they are initialized to 1 at order 0. The resulting $S_{ij}(T)$ are then multiplied by the area of the triangle $T$ and summed up according to (20).

### 2.2.1 Initialization

As in Section 2.1.1, an initial rotation $\mathbf{R}_0$ is applied to ensure that the camera is looking at the correct face of the surface followed by an optional rotation $\mathbf{R}_z$ around the optical axis of the camera, that brings the up looking directional vector parallel to the camera's vertical axis, then apply a translation $\mathbf{t}_c$ to center the region in the origin. The initialization of the parameters $\mathbf{R}$ and $\mathbf{t}$ is done in a similar way as in Section 2.1.1: first the translation $\mathbf{t}$ along the $Z$ axis is initialized such that the image region $\mathcal{D}$ and the projected 3D region are of the same size, then $\mathbf{R}$ is the rotation that brings the centroid of the projected 3D region close to the centroid of the corresponding image region $\mathcal{D}$.

**Algorithm 2** Absolute pose estimation algorithm for perspective cameras

**Input:** The calibration matrix $\mathbf{K}$, 3D triangulated region $\mathcal{F}^{\triangle}$ and corresponding 2D region $\mathcal{D}$ as a binary image.
**Output:** The camera pose.

1: Produce the normalized image $\mathcal{I}$ using $\mathbf{K^{-1}}$ as in (12)
2: Prealign the 3D region $\mathcal{F}^{\triangle}$ by rotating it first with $\mathbf{R_0}$ then with $\mathbf{R_z}$ as described in Section 2.2.1, then center the region in the origin using $\mathbf{t_c}$.
3: Initialize $\mathbf{t} = [0, 0, t_z]^\top$ such that the area of the regions are roughly the same (see Section 2.2.1).
4: Initialize $\mathbf{R}$ to ensure that the regions overlap in $\mathcal{I}$ as in Section 2.2.1.
5: Construct the system of equations (16) and solve it for $(\mathbf{R}, \mathbf{t})$ using the *Levenberg-Marquardt* algorithm.
6: The absolute camera pose is then given as the composition of the transformations $\mathbf{R}_0$, $\mathbf{R}_z$, $\mathbf{t}_c$, and $(\mathbf{R}, \mathbf{t})$.

The steps of the numerical implementation of the proposed method are presented in Algorithm 2. Note that for non-coplanar regions, as in Algorithm 1, we first use a single arbitrarily selected region for an initial pose estimation, then in a second step we solve the system using all the available regions, which provides and optimal pose estimate.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence




Fig. 2: Examples of various amount of segmentation errors (*se*). First an omnidirectional image without *se*, then the same test with *se*$=12\%$, lastly the same template from a perspective test case with *se*$=20\%$.

## 3 Evaluation on Synthetic Data

For the quantitative evaluation of the proposed method, we generated different benchmark sets using $25$ template shapes as 3D planar regions and their images taken by virtual cameras. The 3D data is generated by placing $1/2/3$ 2D planar shapes with different orientation and distance in the 3D Euclidean space. Assuming that the longer side of a template shape is 1m, a set of 3D template scenes are obtained with $1/2/3$ planar regions that have a random relative distance of $\pm(1-2)$m between each other and a random relative rotation of $\pm 30^\circ$.

Both in the perspective and omnidirectional case, a 2D image of the constructed 3D scenes was taken with a virtual camera using the internal parameters of a real 3Mpx $2376 \times 1584$ camera and a randomly generated absolute camera pose. The random rotation of the pose was in the range of $\pm 40^\circ$ around all three axes. The random translation was given in the range $\pm(0.5-2)$m in horizontal and vertical directions and $(2-6)$m in the optical axis direction for the perspective camera, while the omnidirectional camera was placed at half the distance, *i.e.* $(1-3)$m in the direction of the optical axis, and $\pm(0.5-1)$m in the $X$ and $Y$ axis directions to obtain approximately equal sized image regions for both type of cameras.

In practice, we cannot expect a perfect segmentation of the regions, therefore the robustness against segmentation errors was also evaluated on synthetic data (see samples in Fig. 2): we randomly added or removed squares distributed uniformly around the boundary of the shapes, both in the 2D images and on the edges of the 3D planar regions, yielding different levels of segmentation error expressed as the percentage of the original shape's area. Using these images, we tested the robustness against 2D and 3D segmentation errors separately by performing a systematic series of tests using 1, 2, and 3 planes gradually increasing the segmentation error in each tests case individually. Based on these results we determined the noise level for each considered plane configurations such that the median rotation error around any of the axes remains under $1^\circ$ and we show combined error plots of these particular noise levels.

Theoretically, one single plane is sufficient to solve for the absolute pose, but it is clearly not robust enough. We have also found, that the robustness of the 1-plane minimal case is also influenced by the shape used: Symmetric or less compact shapes with smaller area and longer contour, and shapes with elongated thin parts often yield suboptimal results. However, such a solution can be used as an initialization for the solver with more regions. Adding one extra non-coplanar region increases the robustness by more than $4$ times! This robustness enhancement quickly saturates with the number of regions, thus in our experiments we limited the number of non-coplanar patches to 3. We also remark, that the planarity of the regions is not strictly required. In fact, the equations remain true as long as the 3D surface has no self-occlusion from the camera viewpoint (see [4] for a cultural heritage application). Of course, planarity guarantees that the equations remain true regardless of the viewpoint.

Since the proposed algorithms work with triangulated 3D data, the planar regions of the synthetic 3D scene were triangulated. For the perspective test cases a plain Delaunay triangulation of only the boundary points of the shapes were used, thus the mesh contains less but larger triangles, which are computationally favorable. For the spherical solver, however, a higher number of evenly sized triangles is desirable for a good surface approximation, which was produced by the `distmesh2D` function of [55] with the default parameters.

For a quantitative error measure, we used the rotation errors along the 3D coordinate axes as well as the overall rotation error as the rotation angle (or angular distance) $\epsilon = \hat{\mathbf{R}}\mathbf{R}^\top$, $\mathbf{R}$ being the true rotation matrix and $\hat{\mathbf{R}}$ the estimated one; and the difference between the ground truth $\mathbf{t}$ and estimated $\hat{\mathbf{t}}$ translation vectors as $\|\mathbf{t}-\hat{\mathbf{t}}\|$.

Furthermore, as a region-based back-projection error, we also measured the percentage of non-overlapping area (denoted by $\delta$) of the reference 3D shape back-projected onto the 2D image plane and of the 2D observation image. The algorithms were implemented in Matlab and all experiments were run on a standard six-core PC. A demo implementation is available at http://www.inf.u-szeged.hu/~kato/software/ The average runtime of the algorithm varies from $1-3$ seconds in the perspective case to $4-7$ seconds in the omnidirectional case, without explicit code or input data optimization. Quantitative comparisons in terms of the various error plots are shown for each test case.

### 3.1 Omnidirectional cameras

The results with $1$, $2$ and $3$ non-coplanar regions using omnidirectional camera are presented in Fig. 3. In Fig. 3a - Fig. 3d, the rotation and translation errors for various test cases are presented. In the minimal case (*i.e.* $1$ region), errors quickly increase, but using one more region stabilizes the solution: not only the error decreases but the number of correctly solved cases is also greatly increased. The $\delta$ error plot in Fig. 3e also confirms the robustness provided by more regions, while it has to be noted that with more regions the back-projection error does not improve in the way the pose parameter errors would imply, since even a smaller error in the pose yields larger non-overlapping area because of the longer boundary of the distinct regions. While using a second non-coplanar region brings

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



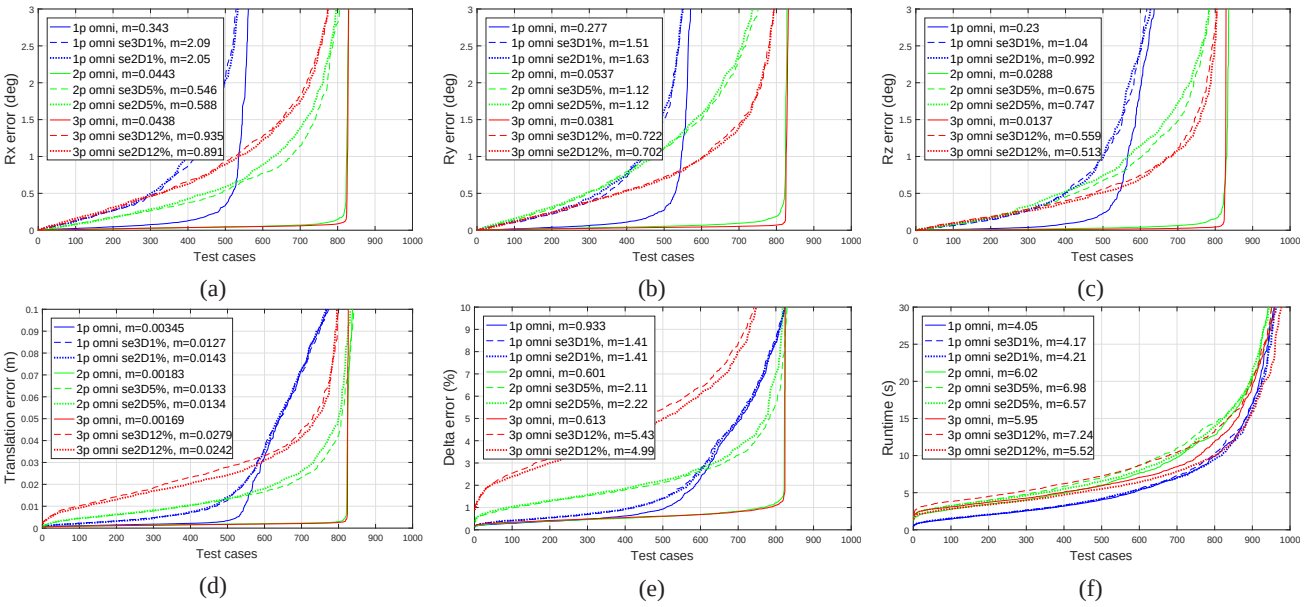


Fig. 3: Omnidirectional rotation errors along the $X$, $Y$, and $Z$ axis (first row) and translation, $\delta$ error and runtime plots (second row). $m$ denotes median error, *se2D* and *se3D* stand for segmentation error on the 2D and 3D regions respectively (best viewed in color).

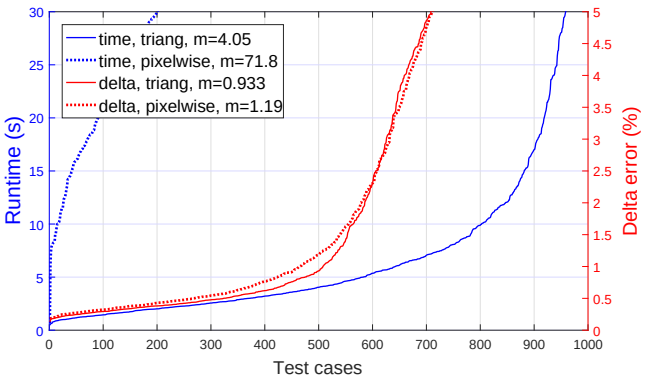


Fig. 4: Backprojection ($\delta$) errors and runtime comparison for point and triangle based spherical surface integral approximation on a 1 plane dataset (best viewed in color).

a rather big improvement, adding subsequent coplanar or non-coplanar regions yield only slight improvement over 2 regions. Fig. 8 shows the performance using 3 planes with 1 and 2 regions per plane, and 5 non-coplanar regions. Practically the increase of the number of regions above 3 does not improve significantly the performance.

While the perfect dataset is solved with median translation errors as low as 2mm (see Fig. 3d), the error is increased by an order of magnitude, but still being under 3cm, for regions corrupted with segmentation error. According to our previous experience [46], a $\delta$ below $5\%$ corresponds to a visually good result. Combining this metric with the rotation error limit of $1^\circ$, we conclude that our method is robust against segmentation errors of up to $\approx 12\%$ if at least 3 non-coplanar regions are used.

We have experimentally shown that the size of the spherical regions is greatly influencing the performance of the solver. While placing the camera closer to the scene produces larger spherical projections of the regions and the pose estimation becomes more robust, we aimed to use real world camera parameters instead, thus the camera-to-scene distance was limited. In our test cases, the median area of the spherical projections for the 1 and 3 region cases were $0.07$ and $0.13$ units respectively on the unit sphere.

For computing the spherical surface integrals, we compared two different approaches for the area approximation of the spherical regions. Our earlier approach is using standard numerical integration over the pixels projected onto the unit sphere [46], while the current one in Algorithm 1 is integrating over spherical triangles instead. The $\delta$ error and runtime of these numerical schemes are compared in Fig. 4, which clearly shows that the CPU time of Algorithm 1 is an order of magnitude faster while the precision remains the same as for the earlier scheme in [46]. The slight precision change is caused by the pixel level discretization of the regions, since the triangulated shapes are generated using a subset of the boundary pixels, thus the triangles practically average out the rasterized edges to a smoother edge, making the integrals slightly more precise.

The algorithm's CPU runtime is shown in Fig. 3f, where the slightly increased runtime of the 3D segmentation error test cases (noted by *se3D*) is due to the triangulation of the corrupted planar regions, that increases the number of triangles around the edges and thus the computational time. Practically our algorithm can solve the pose estimation problem of an omnidirectional camera in $\approx 5$ seconds using 2 regions.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



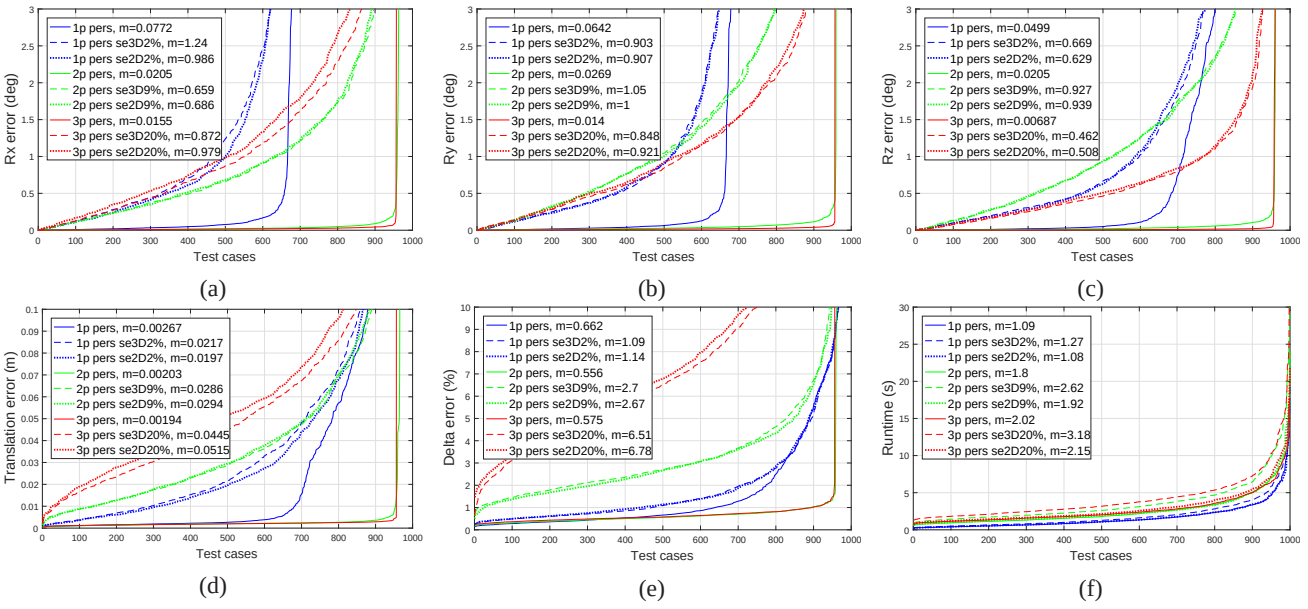


Fig. 5: Perspective pose estimation results: rotation and translation errors, $\delta$ error and algorithm runtime plots. *se2D* stands for observation segmentation error, *se3D* for template side segmentation error and *m* for median values (best viewed in color).

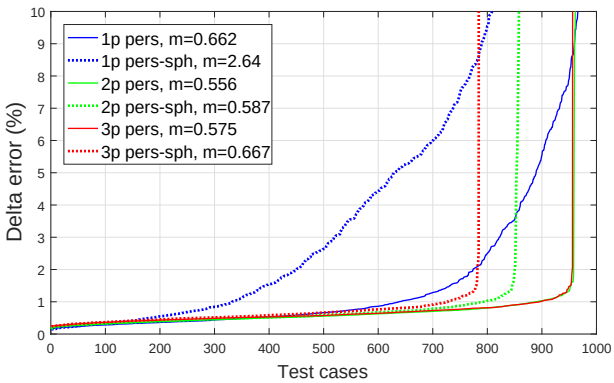


Fig. 6: Perspective pose estimation $\delta$ errors comparing the normalized image plane and the spherical solutions. *m* stands for median values (best viewed in color).

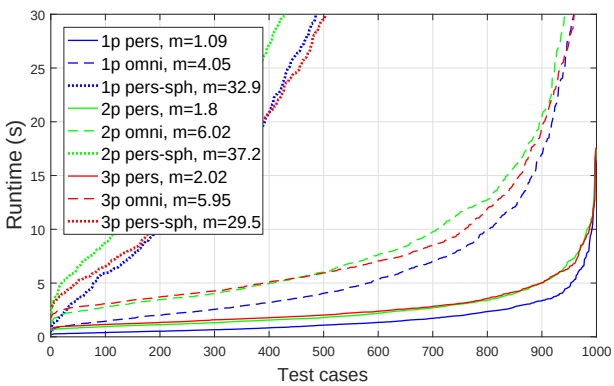


Fig. 7: Runtime comparison on test cases without segmentation errors in the omnidirectional and perspective case, the latter using both the normalized image plane and the spherical solution. *m* stands for median values (best viewed in color).

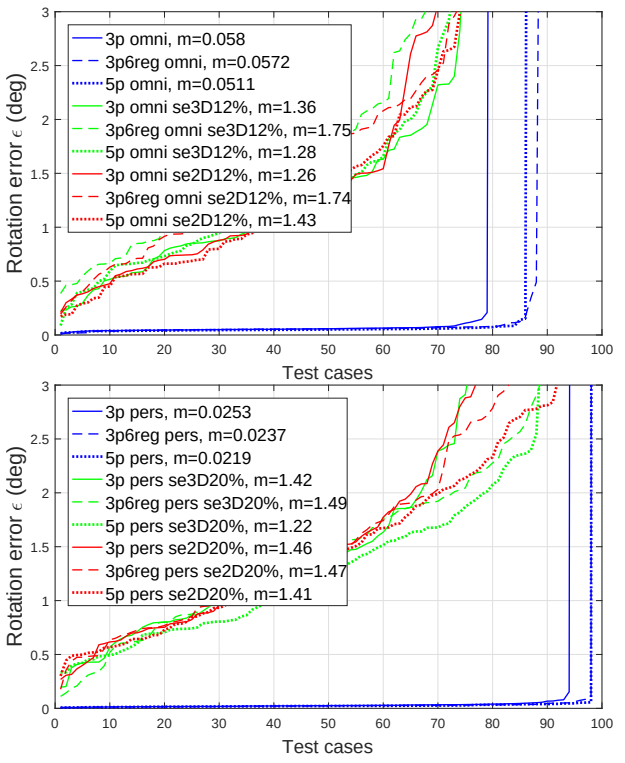


Fig. 8: Overall rotation error in function of the number of regions and planes for the omnidirectional (top) and perspective (bottom) camera. *m* stands for median values (best viewed in color).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



### 3.2 Perspective cameras

Pose estimation results using a perspective camera are presented in Fig. 5, including the same test cases with 1, 2 and 3 non-coplanar regions and with segmentation errors as in the omnidirectional case. The rotation and translation error plots in Fig. 5a - Fig. 5d clearly confirm the advantage of having more non-coplanar regions.

The median translation error (see Fig. 5d) on the perfect dataset is as low as 2mm, which increases by an order of magnitude in the presence of $20\%$ segmentation error, but still being under 5cm in case of 3 regions. The $\delta$ error plot in Fig. 5e also shows the robustness provided by the additional regions. Obviously, the back-projection error also increases in the presence of segmentation errors. However, as Fig. 5a - Fig. 5d shows, the actual pose parameters are considerably improved and the robustness greatly increases by using 1 or 2 extra non-coplanar regions. Similar to the omnidirectional case, the increase of the number of regions above 3 does not improve significantly the performance of the pose estimation (see also Fig. 8).

The algorithm's CPU time on perspective test cases is shown in Fig. 5f. The increased runtime of the 3D segmentation error test cases (noted by *se3D*) is due to the triangulation of the corrupted planar regions, that greatly increases the total number of triangles and thus the computational time. Practically our algorithm can solve the pose estimation problem of a perspective camera in around $2.5$ seconds using $2$ regions.

As mentioned in Section 2.2, a perspective camera can also be represented by the spherical model developed in Section 3.1. However, as we have shown in the previous section, this model's main limitation is the small size of the spherical regions, because a perspective camera has a narrower field of view and has to be placed at a larger distance from the scene, to produce the same size of regions on the image. The resulting spherical projections of the planar regions in median are typically $4$ times smaller than in the omnidirectional camera's case. Thus solving the perspective case using the spherical solver yields a degraded performance, as shown by the $\delta$ error plot in Fig. 6. Comparing the algorithm's runtime plot in Fig. 7 also shows that using the spherical solver for the perspective camera greatly increases the computing time due to the calculation of surface integrals on the sphere, which confirms the advantage of using the perspective solver proposed in Section 2.2, instead of a unified spherical solver.

### 3.3 Degeneracy analysis

In order to analyze the degeneracy characteristics of the proposed method, two separate tests were performed: one along the $X$ axis rotation (rotations along the $Y$ axis would have similar effects) and another one for the in-plane rotation (along the $Z$ axis). Both test cases are also linked to the robustness of the initialization step of the algorithm. For these experiments, a series of new datasets with 100 test cases each were generated using one region with non-symmetric template shapes (symmetric ones were considered separately).

In each dataset, $\mathbf{R}_X$ was set to a fixed absolute value such that the viewing angle between the plane and the camera optical axis was ranging from $50^\circ$ down to $2^\circ$ in steps of $5^\circ$, yielding gradually increased perspective distortions up to the extreme $2^\circ$ case. The other rotation parameters were set to $0$, and the translation along the optical axis was randomly generated to ensure roughly equally sized projections on the image plane. This is needed to a correct evaluation where results are not influenced by other pose parameters or the changes in the size of the projected regions. Experimental results in Fig. 9 show that if the camera's orientation is initialized within $\pm 60^\circ$ of the true rotation (*i.e.* a quite rough initialization suffices), the algorithm can robustly find the correct solution even in case of degenerate viewing angles: at least $80\%$ of the cases are solved with $\epsilon < 1^\circ$ rotation error for $\mathbf{R}_X >= 5^\circ$, but below $5^\circ$ viewing angle, the errors increase significantly. Symmetry of the template shapes doesn't affect significantly the results, since the projectively distorted shapes usually don't preserve the symmetry.

Degeneracy analysis along the $Z$ axis (in-plane rotations only) practically translates into the initialization problem of the in-plane rotation, which can be performed in a real application using a rough estimate of *e.g.* the vertical direction. As in the previous case, we generated our datasets with a pure $\mathbf{R}_Z$ rotation around the $Z$ axis going from $30^\circ$ to $90^\circ$ in absolute value, while setting all other rotations to $0$ and the translation along the optical axis was randomly generated to ensure roughly equally sized projections on the image plane. Since the initialization of $\mathbf{R}_Z = 0$ was used throughout the test cases, these rotations directly translate into a rotation error in the initialization of $\mathbf{R}_Z$. Experimental results show, that if the in-plane rotation is initialized within a $\pm 60^\circ$ ($\pm 80^\circ$ for the perspective) of the true rotation, the proposed method robustly finds a correct solution (conditioned to the use of non-symmetrical shapes) as this is visible in the second and last histogram plots of Fig. 9. In contrast to the plane rotation along $X$ and $Y$ axes, the in-plane rotation preserves the symmetry of the template shapes in our setup, thus results are greatly affected if symmetric shapes are used, but $30^\circ - 40^\circ$ initialization error is easily tolerated even for symmetric shapes.

## 4 Evaluation on real datasets

To thoroughly evaluate our method on real world test cases, we used several different 3D data recorded by commercial as well as a custom built 3D laser range finder with corresponding 2D color images captured by commercial SLR and compact digital cameras with prior calibration and radial distortion removal. Whatever the source of the 2D-3D data is, the first step is the segmentation of planar region pairs used by our algorithm. There are several automated or semi-automated 2D segmentation algorithms in the literature including *e.g.* clustering, energy-based or region growing algorithms [56]. In this work, a simple

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



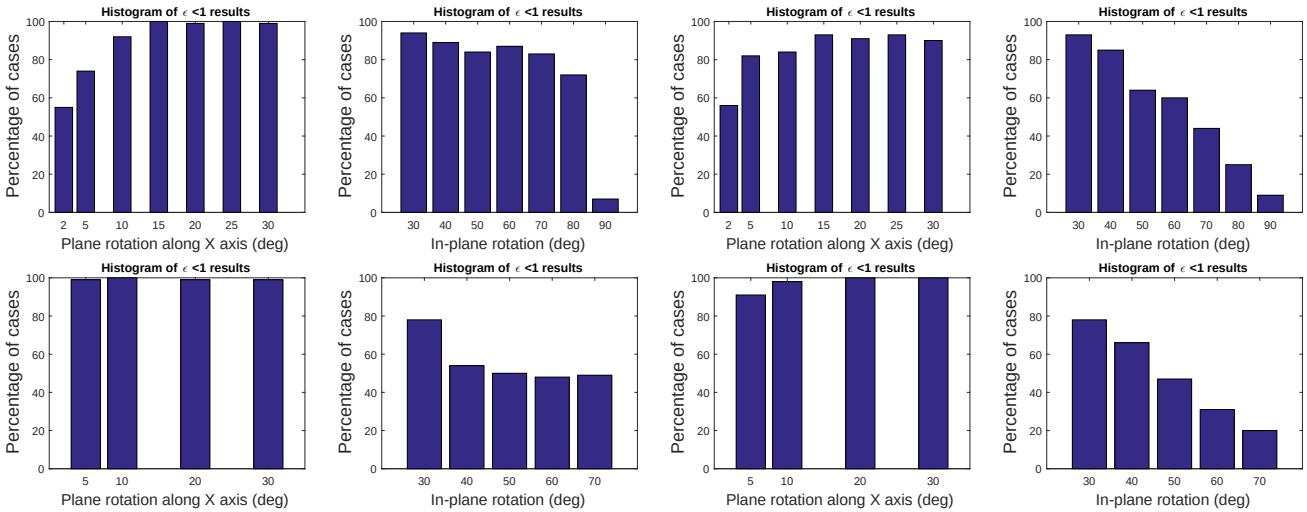


Fig. 9: Histograms of the overall rotations with errors less than 1 degrees (from left to right): for the perspective camera along the $X$ and $Z$ axis and for the omnidirectional camera along the $X$ and $Z$ axis. First row shows results on non-symmetric shapes, second row on the symmetric ones.

region growing was used which proved to be robust enough in urban environment [57]. As for 3D segmentation, a number of point cloud segmentation methods are available, *e.g.* based on difference of normals [58] or robust segmentation [59]. Like in 2D, region growing based on surface normals gave stable results for extracting planar 3D regions in our experiments. Corresponding 2D-3D regions were simply selected during the seed selection of region growing as a one-click user input. We remark, however, that a fully automatic region correspondence could be implemented by detecting and extracting planar objects like windows [60] (see *e.g.* Fig. 10) which are typically planar surfaces present in urban scenes. The 2D-3D correspondence search often can be transformed into 2D-2D image based matching, as the 3D models are built from 2D images. Application specific solutions such as building facade segmentation [61], [62] or traffic sign extraction [63] support the matching of high level features (such as windows, doors, walls, tables,...) in 2D and 3D data. Object extraction approaches relying on semantic segmentation [64], [65] or semantic scene completion [66] yield high level 2D-3D feature sets especially in semantically rich environments, from which the corresponding region pairs can be filtered out. If the segmented 3D region is a simple point cloud, the boundary of the region is detected using Alpha Shapes [67], which is then used for generating a triangular mesh. As in the synthetic case, for the omnidirectional case the method of [55] generated a uniform mesh, while for the perspective case a simple Delaunay triangulation was sufficient. The absolute pose obtained from Algorithm 1 or Algorithm 2 was used to fuse the depth and RGB data by projecting the images onto the 3D point cloud.

In Fig. 10, we show the fusion of an RGB perspective camera image and a sparse 3D point cloud recorded by a custom built 3D laser range finder containing a tilted Sick LMS200 ranger. The absolute pose of the RGB camera was computed using Algorithm 2, which was then used to back-project the RGB image onto the 3D point cloud. Despite of the relatively large displacement between the camera and the Lidar, the absolute pose was successfully estimated.

For the omnidirectional real data experiments we first tested the proposed method on 2D fish-eye camera images and a 3D triangulated building model obtained by registering a set of sparse 3D laser scans recorded by a Velodyne HDL-64E with a depth resolution up to 1cm and an angular resolution up to $0.5°$. The best results were obtained by large non-coplanar regions. Such a test case

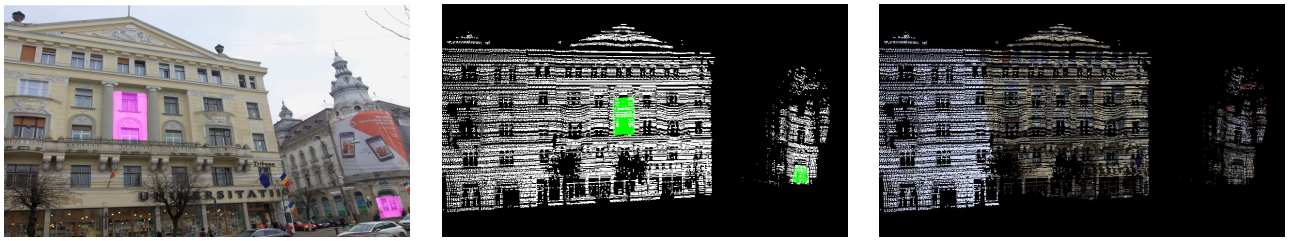

Fig. 10: Pose estimation example with (left-right) central perspective camera and custom Lidar data: color 2D image (original frame) with corresponding regions (purple); 3D data with the segmented regions (green); color information overlaid on 3D data using the estimated camera pose (best viewed in color).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



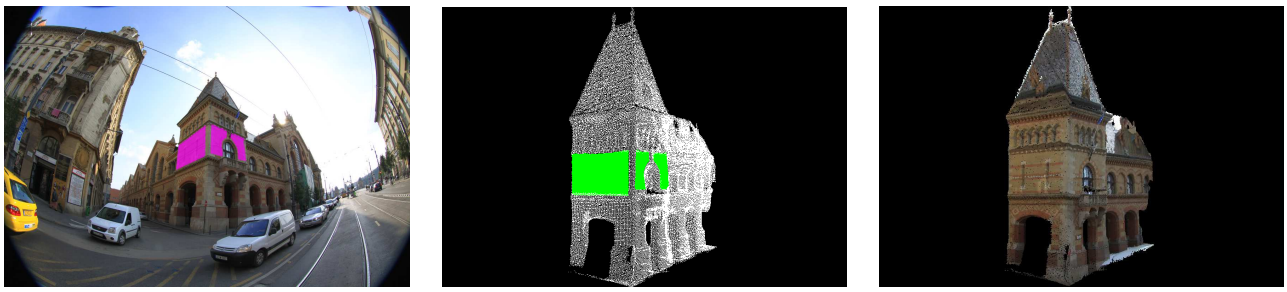

Fig. 11: Pose estimation example with (left-right) central dioptric (fish-eye) and commercial (Velodyne) Lidar images: color 2D image (original frame) with corresponding regions (purple); 3D data with the segmented region (green); color information overlaid on 3D data using estimated pose parameters (best viewed in color).

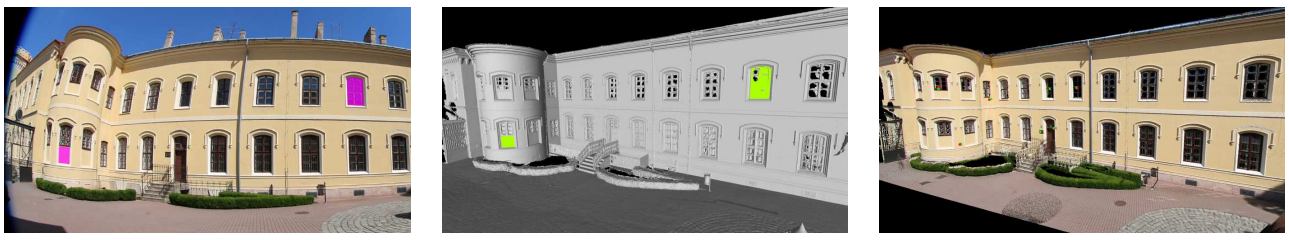

Fig. 12: Pose estimation example with omnidirectional camera image and dense Lidar data (left to right): color 2D image and 3D triangulated surface with corresponding segmented regions marked with purple and green respectively; lastly color information projected onto 3D data using the estimated extrinsic parameters, green dots mark the reference positions of the markers while red dots mark the projected positions (best viewed in color).

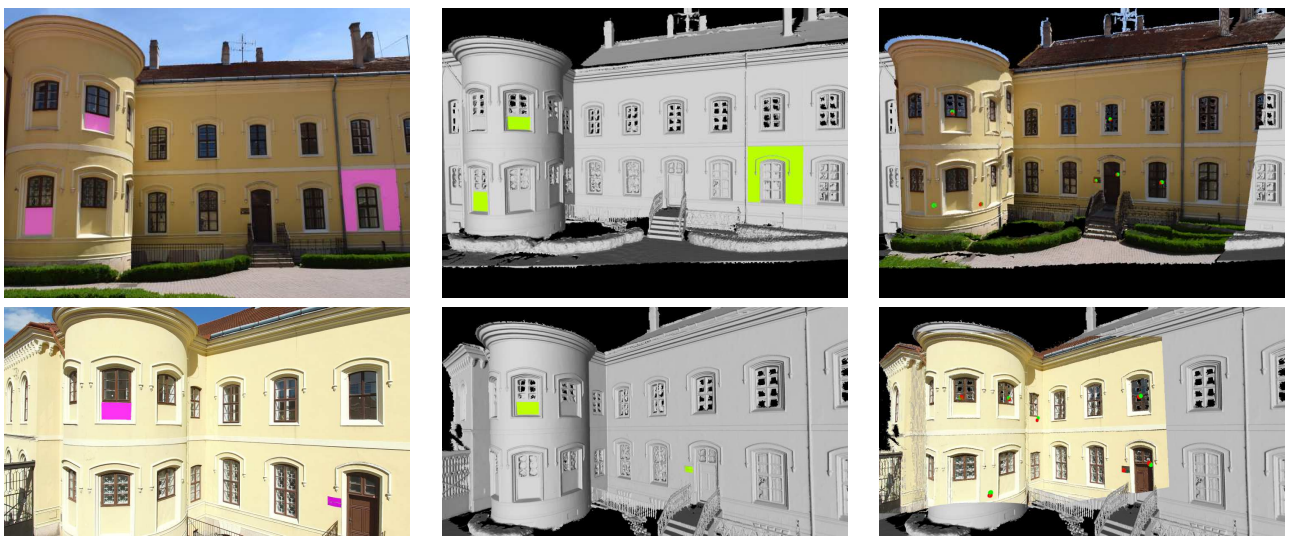

Fig. 13: Pose estimation example with perspective cameras and dense Lidar data (left to right): color 2D image and 3D triangulated surface with corresponding segmented regions marked with purple and green respectively; color information projected onto 3D data using the estimated pose, green dots mark the reference position of the markers while red dots mark the projected position. First row: wide field of view camera image; second row: normal field of view UAV camera image. (best viewed in color).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2931577, IEEE Transactions on Pattern Analysis and Machine Intelligence



is shown in Fig. 11, where the fish-eye camera image was reprojected onto the 3D surface using the absolute pose obtained by Algorithm 1. Note that in case of the omnidirectional cameras, even a relatively small rotation or translation error in the pose yields large differences in the non-linear distortions on the omnidirectional data. In spite of this sensitivity, Algorithm 1 proved to be robust enough as the segmented regions in Fig. 11 overlap well even if the total area of selected regions is relatively small compared to the whole image size.

Finally, test cases with a high precision Riegl Lidar and different cameras are shown in Fig. 12 and Fig. 13. The static Riegl scanner has a range of $400$m with a depth precision of less than $0.5$cm and angular resolution up to $0.003°$. In this dataset, the high density precise 3D model also includes the 3D positions of marker points that were set up on the building facade. Using these markers, we could evaluate the precision of our pose estimation by the forward projection of each marker from the 2D image into 3D space and then calculated the distance from their ground truth position.

For the omnidirectional case shown in Fig. 12, we used a full frame Canon EOS 5 DSLR camera with a $8$mm fish-eye lens. Segmenting only two simple, relatively small regions, the proposed Algorithm 1 estimated a precise pose with a forward projection mean error measured in the marker points of only 7cm. The ground truth marker positions are visualized in green while the projected markers in red. Note that the camera-to-scene distance was $\approx 14$m in this case. For comparison, we also show in Table 1 the error of the absolute pose obtained by the state of the art UPnP [7] method, which directly used the ground truth marker positions as input 2D-3D point matches. In spite of working with perfect point correspondences, UPnP achieved only $2$cm better forward projection error in those marker points than our method which used inherently imperfect segmented region pairs.

For the perspective case in Fig. 13, we used a full frame Nikon DSLR camera with a wide field of view $20$mm lens, one of the typical RGB cameras that comes calibrated with these Riegl scanners. The mean forward projection error of the proposed Algorithm 2 measured in the marker points was $3$cm. The advantage of using multiple regions from differently oriented surfaces is clearly visible here. In Table 1, we compare our results to the factory calibration of the setup. It was interesting to find, that at $18$m distance from the wall, the factory calibration parameters produce $20$cm mean forward projection error, due to the interchangeable camera mounting system. Applying a marker based refinement to the calibration in the scanners own software, this can be reduced to $1.3$cm, which is only slightly better than our marker-less result achieved purely using 3 segmented region pairs.

The proposed Algorithm 2 was also tested with images taken by a flying DJI Phantom 3 drone. As can be seen in Fig. 13, the viewing angle of such a camera is very different from that of a ground level imaging device. Using two corresponding segmented regions was sufficient to estimate a correct pose with a mean forward projection error of $9$cm, which is a good result considering the extreme angle of the camera and the camera-to-scene distance of $\approx 9$m. In comparison, the state of the art UPnP [7] and RPnP[6] methods using the high precision marker points as input 2D-3D point correspondences produced $2$cm and $6$cm mean error, respectively.

| | UPnP | RPnP | Riegl | Riegl(fine) | Prop. |
|---|---|---|---|---|---|
| Omni | 5 | n/a | n/a | n/a | 7 |
| Pers. HR | 0.9 | 4 | 20 | 1.3 | 3 |
| Pers. Drone | 2.2 | 6 | n/a | n/a | 9 |

TABLE 1: Comparisons on high resolution Lidar data in terms of the mean forward projection errors in marker points in cm. Note that results of UPnP [7], RPnP[6] and *Riegl(fine)* all rely on markers. *Riegl* stands for factory calibration, *Prop.* for the proposed method, and HR for high resolution full frame camera perspective test case.

| | transl. | Rx | Ry | Rz | $\delta$(%) | time(s) |
|---|---|---|---|---|---|---|
| Prop. | 0.592 | 2.970 | 0.402 | 0.393 | 12.49 | 1.23 |
| Norm. | 0.441 | 0.522 | 4.740 | 0.745 | 74.01 | 166 |
| Int. | 0.397 | 3.254 | 4.826 | 1.543 | 46.77 | 147 |

TABLE 2: Comparative results with the proposed method (Prop), normal based MI(Norm)[17] and intensity based MI (Int)[17] in terms of translation(m), rotation(deg) and $\delta$ (for reference: $\delta$ for the ground truth pose is $9.49$%) errors.

The qualitative comparison of all the mentioned methods is presented in Table 1, where n/a stands for not available, since factory calibration parameters were only available in one case, and RPnP[6] cannot be used with omnidirectional cameras. Let us emphasize, that all the point-correspondence-based methods (except the Riegl factory parameters) rely on 2D-3D special markers, that were precisely measured in 3D and 2D. Thus to achieve these results with UPnP and RPnP, a careful setup of special markers in required before data acquisition, thus both 2D and 3D data capture must be performed at the same time. In contrast, the proposed method does not require any special target or setup, hence images recorded at different time can be fused as long as at least one planar region pair is available.

### 4.1 Algorithm evaluation on the KITTI dataset

Comparison with other camera pose estimation methods from the main literature could be performed only in a limited manner due to the fundamental differences of the proposed algorithm with respect to existing ones presented in Section 1. Methods using artificial markers like the ones described in [35], [31] were tested using the codes provided by the authors. The detailed comparisons are presented in our previous work [47]. Due to the limitations of [35], [31] on real datasets, we also tested the proposed method on the KITTI dataset [68] with available ground truth information. In Fig. 14 the extrinsic calibration of a color camera and

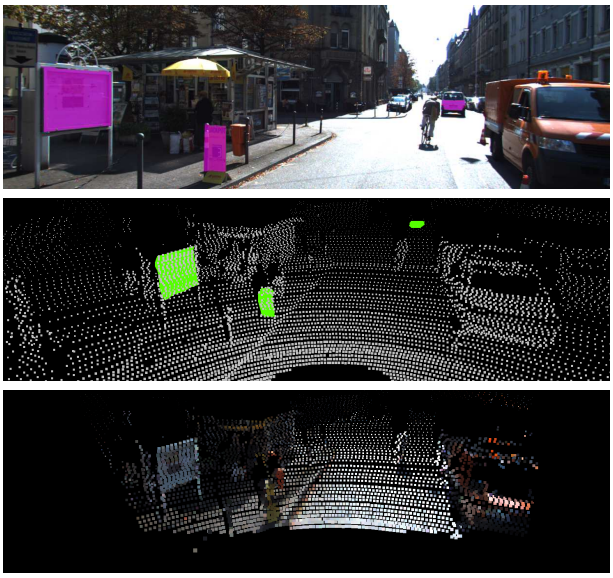

Fig. 14: Pose estimation on the KITTI dataset (top-bottom): color 2D data with the selected regions (purple); 3D data with the corresponding regions (green); color information overlaid on 3D data using the estimated camera pose.

sparse 3D Lidar data from the KITTI drive $nr = 5$ is shown. Using 3 segmented non-coplanar regions marked in purple and green in Fig. 14, the camera pose was estimated with the precision shown in Table 2.

For comparison, we used the mutual information based method described in [17] working on 3D data with intensity and normal information. The algorithm of [17] was run on the same 2D-3D data pair both in the normal based and intensity based configurations as presented in Fig. 14. The comparative results of absolute errors are also shown in Table 2. Note that while the algorithm of [17] is able to use multiple separate 2D-3D data pairs (if a sequence of such data is available with a rigid Lidar-camera setup like the KITTI dataset) to optimize the results, for a fair comparison we only provided the same single image frame and point cloud pair as the one that the proposed method was tested on. Since [17] is non-deterministic, the MI based results in Table 2 show the best ones out of 5 independent runs of the algorithm.

The results of the proposed method proved to be comparable to the results of [17], the normal based method being slightly better in the translation parameters, but worse in the rotation errors. Nevertheless the registration result of the proposed method visually was accurate, and the CPU implementation runtime was two orders of magnitude smaller than the GPU implementation of the mutual information method of [17].

## 5 Conclusion

A generic, nonlinear, explicit correspondence-less pose estimation method was proposed in this work. The absolute camera pose estimation is based on the 3D-2D registration of a common Lidar-camera planar patch. The proposed method makes use of minimal information (plain depth data from 3D and radiometric information from 2D) and is general enough to be used both for perspective and omnidirectional central cameras. The algorithm has been tested on a large scale synthetic dataset and on various real life date acquired by different types of sensors. The method could be further extended to handle internal camera parameter estimation as well. The state of the art performance of the proposed method was confirmed both on a large synthetic data set as well as on various real data experiments using different depth sensors, perspective and omnidirectional cameras.

## Acknowledgments

This work was partially supported by the NKFI-6 fund through project K120366; "Integrated program for training new generation of scientists in the fields of computer science", EFOP-3.6.3-VEKOP-16-2017-0002; the Research & Development Operational Programme for the project "Modernization and Improvement of Technical Infrastructure for Research and Development of J. Selye University in the Fields of Nanotechnology and Intelligent Space", ITMS 26210120042, co-funded by the European Regional Development Fund; the Romanian National Authority for Scientific Research and Innovation under Grant number PN-III-P1-1.1-TE-2016-0670; and Bolyai Scholarship of the Hungarian Academy of Sciences. The authors gratefully acknowledge the help of Csaba Benedek from SZTAKI in providing us with the preprocessed Velodyne Lidar scans in Fig. 11; as well as the discussions with Radu Orghidan and Cedric Demonceaux in the early stage of this work.

## References

[1] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition*, vol. 1. Washington, DC, USA: IEEE, Jun. 2004, pp. 1–8.

[2] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis, "Camera-IMU-based localization: Observability analysis and consistency improvement," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 182–201, 2014.

[3] C. Arth, M. Klopschitz, G. Reitmayr, and D. Schmalstieg, "Real-time self-localization from panoramic images on mobile devices," in *Proceedings of International Symposium on Mixed and Augmented Reality*. Basel, Switzerland: IEEE Computer Society, Oct. 2011, pp. 37–46.

[4] R. Frohlich, Z. Kato, A. Tremeau, L. Tamás, S. Shabo, and Y. Waksman, "Region based fusion of 3D and 2D visual data for cultural heritage objects," in *Proceedings of International Conference on Pattern Recognition*, IEEE. Cancun, Mexico: IEEE, Dec 2016, pp. 2404–2409.

[5] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *Int. J. Comput. Vision*, vol. 81, no. 2, pp. 155–166, Feb. 2009.

[6] S. Li, C. Xu, and M. Xie, "A robust o(n) solution to the perspective-n-point problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1444–1450, July 2012.

[7] L. Kneip, H. Li, and Y. Seo, "UPnP: an optimal O(n) solution to the absolute pose problem with universal applicability," in *Proceedings of European Conference on Computer Vision*, ser. Lecture Notes in Computer Science, vol. 8689. Zurich, Switzerland: Springer, Sep. 2014, pp. 127–142.

[8] C. Xu, L. Zhang, L. Cheng, and R. Koch, "Pose estimation from line correspondences: A complete analysis and a series of solutions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1209–1222, 2016.

[9] N. Horanyi and Z. Kato, "Generalized pose estimation from line correspondences with known vertical direction," in *International Conference on 3D Vision*. Qingdao, China: IEEE, Oct. 2017, pp. 244–253.

[10] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3D," in *ACM SIGGRAPH*. Boston, Massachusetts: ACM, 2006, pp. 835–846.

[11] F. Camposeco, T. Sattler, and M. Pollefeys, "Minimal solvers for generalized pose and scale estimation from two rays and one point," in *Proceedings of European Conference Computer Vision*, ser. Lecture Notes in Computer Science, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., vol. 9909. Amsterdam, The Netherlands: Springer, Oct. 2016, pp. 202–218.

[12] G. H. Lee, "A minimal solution for non-perspective pose estimation from line correspondences," in *Proceedings of European Conference on Computer Vision*. Amsterdam, The Netherlands: Springer, Oct. 2016, pp. 170–185.

[13] S. Baker and S. K. Nayar, "A theory of single-viewpoint catadioptric image formation," *International Journal of Computer Vision*, vol. 35, no. 2, pp. 175–196, 1999.

[14] C. Geyer and K. Daniilidis, "A unifying theory for central panoramic systems and practical applications," in *Proceedings of the 6th European Conference on Computer Vision-Part II*, ser. ECCV '00. London, UK, UK: Springer-Verlag, 2000, pp. 445–461.

[15] B. Mičušík and T. Pajdla, "Para-catadioptric camera auto-calibration from epipolar geometry," in *Proc. of the Asian Conference on Computer Vision (ACCV)*, K.-S. Hong and Z. Zhang, Eds., vol. 2. Seoul, Korea South: Asian Federation of Computer Vision Societies, January 2004, pp. 748–753.

[16] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A flexible technique for accurate omnidirectional camera calibration and structure from motion," in *Proceedings of the Fourth IEEE International Conference on Computer Vision Systems*, ser. ICVS-06. Washington, USA: IEEE Computer Society, 2006, pp. 45–51.

[17] Z. Taylor and J. Nieto, "A mutual information approach to automatic calibration of camera and lidar in natural environments," in *Australian Conference on Robotics and Automation*, Wellington, Australia, December 2012, pp. 3–5.

[18] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easily calibrating omnidirectional cameras." in *IEEE/RSJ International Conference on Intelligent Robots*. Bejing: IEEE, October 9–15 2006, pp. 5695–5701.

[19] J. Kannala and S. S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 8, pp. 1335–1340, 2006.

[20] D. Scaramuzza, A. Harati, and R. Siegwart, "Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes," in *IEEE International Conference on Intelligent Robots and Systems*, IEEE/RSJ. San Diego, USA: IEEE, October 2007, pp. 4164–4169.

[21] F. M. Mirzaei, D. G. Kottas, and S. I. Roumeliotis, "3D lidar-camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization," *The International Journal of Robotics Research*, vol. 31, no. 4, pp. 452–467, 2012.

[22] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, "Automatic targetless extrinsic calibration of a 3D lidar and camera by maximizing mutual information," in *Proceedings of the AAAI National Conference on Artificial Intelligence*, Toronto, Canada, July 2012, pp. 2053–2059.

[23] N. Schneider, F. Piewak, C. Stiller, and U. Franke, "RegNet: Multimodal sensor registration using deep neural networks," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, June 2017, pp. 1803–1810.

[24] G. Iyer, K. Ram R., J. Krishna Murthy, and K. Madhava Krishna, "CalibNet: Self-Supervised Extrinsic Calibration using 3D Spatial Transformer Networks," *ArXiv e-prints*, *1803.08181*, Mar. 2018.

[25] C. Domokos, J. Nemeth, and Z. Kato, "Nonlinear shape registration without correspondences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 943–958, May 2012.

[26] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever, "Mutual-information-based registration of medical images: a survey," *IEEE Transcations on Medical Imaging*, vol. 22, no. 8, pp. 986–1004, 2003.

[27] D. Paudel, C. Demonceaux, A. Habed, and P. Vasseur, "Localization of 2D cameras in a known environment using direct 2D-3D registration," in *Pattern Recognition (ICPR), 2014 22nd International Conference on*, Aug 2014, pp. 196–201.

[28] A. Banno and K. Ikeuchi, "Omnidirectional texturing based on robust 3D registration through euclidean reconstruction from two spherical images," *Computer Vision Image Understanding*, vol. 114, no. 4, pp. 491–499, 2010.

[29] M. Corsini, M. Dellepiane, F. Ganovelli, R. Gherardi, A. Fusiello, and R. Scopigno, "Fully automatic registration of image sets on approximate geometry," *International Journal of Computer Vision*, vol. 102, no. 1-3, pp. 91–111, 2013.

[30] T. Franken, M. Dellepiane, F. Ganovelli, P. Cignoni, C. Montani, and R. Scopigno, "Minimizing user intervention in registering 2D images to 3D models." *The Visual Computer*, vol. 21, no. 8-10, pp. 619–628, 2005.

[31] H. S. Alismail, L. D. Baker, and B. Browning, "Automatic calibration of a range sensor and camera system," in *Second Joint 3DIM/3DPVT Conference: 3D Imaging, Modeling, Processing, Visualization and Transmission*. Zurich, Switzerland: IEEE, October 2012, pp. 286–292.

[32] O. Naroditsky, E. P. Iv, and K. Daniilidis, "Automatic alignment of a camera with a line scan lidar system," in *International Conference on Robotics and Automation*. Shanghai, China: IEEE, May 2011, pp. 3429–3434.

[33] Y. Liu, T. Huang, and O. Faugeras, "Determination of camera location from 2D to 3D line and point correspondences," in *Computer Vision and Pattern Recognition, 1988. Proceedings CVPR '88., Computer Society Conference on*, Jun 1988, pp. 82–88.

[34] L. Liu and I. Stamos, "Automatic 3D to 2D registration for the photorealistic rendering of urban scenes." in *Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE Computer Society, June 2005, pp. 137–143.

[35] R. Unnikrishnan and M. Hebert, "Fast extrinsic calibration of a laser rangefinder to a camera," Carnegie Mellon University, Tech. Rep., 2005.

[36] Q. Zhang, "Extrinsic calibration of a camera and laser range finder," in *International Conference on Intelligent Robots and Systems*. Sendai, Japan: IEEE, September 2004, pp. 2301–2306.

[37] A. Mastin, J. Kepner, and J. W. F. III, "Automatic registration of lidar and optical images of urban scenes," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Miami, Florida, USA: IEEE, June 2009, pp. 2639–2646.

[38] D. Herrera C, J. Kannala, and J. Heikkila, "Joint depth and color camera calibration with distortion correction." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 10, pp. 1–8, 2012.

[39] J. Levinson and S. Thrun, "Automatic online calibration of cameras and lasers," in *Proceedings of Robotics: Science and Systems*, Berlin, Germany, June 2013.

[40] H. P. A. Lensch and W. Heidrich, "A silhouette-based algorithm for texture registration and stitching," *Graphical Models*, vol. 63, pp. 245–262, 2001.

[41] P. Nunez, P. Drews, R. Rocha, and J. Dias, "Data fusion calibration for a 3D laser range finder and a camera using inertial data," in *European Conference on Mobile Robots*, Dubrovnik, Croatia, September 2009, pp. 31–36.

[42] Z. Taylor, J. Nieto, and D. Johnson, "Multi-modal sensor calibration using a gradient orientation measure," *Journal of Field Robotics*, vol. 32, no. 5, pp. 675–695, 2015.

[43] D. G. Lowe, "Fitting parameterized three-dimensional models to images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 441–450, 1991.

[44] A. Taneja, L. Ballan, and M. Pollefeys, "Registration of spherical panoramic images with cadastral 3D models," in *Proceedings of the 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, ser. 3DIMPVT. Washington, DC, USA: IEEE Computer Society, 2012, pp. 479–486.

[45] M. Zhang, Y. Chen, and G. Wang, "Hybrid silhouette and key-point driven 3D-2D registration," in *Image and Graphics (ICIG), 2013 Seventh International Conference on*, July 2013, pp. 585–590.

[46] L. Tamas, R. Frohlich, and Z. Kato, "Relative pose estimation and fusion of omnidirectional and lidar cameras," in *Proceedings of the ECCV Workshop on Computer Vision for Road Scene Understanding and Autonomous Driving*, ser. Lecture Notes in Computer Science, L. de Agapito, M. M. Bronstein, and C. Rother, Eds., vol. 8926. Zurich, Switzerland: Springer, Sep. 2014, pp. 640–651.

[47] L. Tamas and Z. Kato, "Targetless calibration of a lidar - perspective camera pair," in *Proceedings of ICCV Workshop on Big Data in 3D Computer Vision*, IEEE. Sydney, Australia: IEEE, Dec. 2013, pp. 668–675.

[48] J. E. G. Joseph O'Rourke, Ed., *Handbook of Discrete and Computational Geometry, Second Edition (Discrete Mathematics and Its Applications)*. Chapman and Hall/CRC, 2004.
[49] J. M. Pozo, M. C. Villa-Uriol, and A. F. Frangi, "Efficient 3D geometric and zernike moments computation from unstructured surface meshes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 471–484, March 2011.
[50] P. Koehl, "Fast recursive computation of 3d geometric moments from surface meshes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2158–2163, Nov 2012.
[51] J.-G. Leu, "Computing a shape's moments from its boundary," *Pattern Recognition*, vol. 24, no. 10, pp. 949–957, 1991.
[52] M. H. Singer, "A general approach to moment calculation for polygons and line segments," *Pattern Recognition*, vol. 26, no. 7, pp. 1019 – 1028, 1993.
[53] X. Jiang and H. Bunke, "Simple and fast computation of moments," *Pattern Recognition*, vol. 24, no. 8, pp. 801–806, 1991.
[54] G. C. Best, "Helpful formulas for integrating polynomials in three dimensions (in Technical Notes and Short Papers)," *International Journal of Mathematics and Computer Science*, vol. 18, no. 86, pp. 310–312, April 1964.
[55] P.-O. Persson and G. Strang, "A simple mesh generator in matlab," *SIAM review*, vol. 46, no. 2, pp. 329–345, 2004.
[56] S. R. Vantaram and E. Saber, "Survey of contemporary trends in color image segmentation," *Journal of Electronic Imaging*, vol. 21, no. 4, pp. 1–28, 2012.
[57] M. Preetha, L. Suresh, and M. Bosco, "Image segmentation using seeded region growing," in *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*, 2012, pp. 576–583.
[58] Y. Ioannou, B. Taati, R. Harrap, and M. Greenspan, "Difference of normals as a multi-scale operator in unorganized point clouds," *ArXiv e-prints*, Sep. 2012.
[59] A. Nurunnabi, D. Belton, and G. West, "Robust segmentation in laser scanning 3d point cloud data," in *Digital Image Computing Techniques and Applications (DICTA), 2012 International Conference on*, 2012, pp. 1–8.
[60] L. Tamas and L. C. Goron, "3D semantic interpretation for robot perception inside office environments," *Eng. Appl. of AI*, vol. 32, pp. 76–87, 2014.
[61] S. I. Zolanvari, D. F. Laefer, and A. S. Natanzi, "Three-dimensional building façade segmentation and opening area detection from point clouds," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 143, pp. 134–149, 2018.
[62] M. Mathias, A. Martinović, and L. Van Gool, "ATLAS: A Three-Layered Approach to Facade Parsing," *International Journal of Computer Vision*, vol. 118, no. 1, pp. 22–48, May 2016.
[63] H. S. Lee and K. Kim, "Simultaneous traffic sign detection and boundary estimation using convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 5, pp. 1652–1663, May 2018.
[64] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba, "Semantic Understanding of Scenes Through the ADE20K Dataset," *International Journal of Computer Vision*, vol. 127, no. 3, pp. 302–321, Mar 2019.
[65] A. Dai and M. Nießner, "3DMV: Joint 3D-Multi-view Prediction for 3D Semantic Scene Segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, ser. Lecture Notes in Computer Science, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., vol. 11214. Springer, 2018, pp. 458–474.
[66] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE Computer Society, Jul 2017, pp. 190–198.
[67] N. Akkiraju and H. Edelsbrunner, "Triangulating the surface of a molecule," *Discrete Applied Mathematicss*, vol. 71, no. 1-3, pp. 5–22, Dec. 1996.
[68] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision Meets Robotics: The KITTI Dataset," *Int. J. Rob. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.

**Robert Frohlich** received the B.Sc. and M.Sc. degrees in Applied Electronics and Multimedia Technologies from the Technical University of Cluj-Napoca (Romania) in 2012 and 2014 respectively. Since then, he is a Ph.D. student at the University of Szeged (Hungary) under the guidance of Zoltan Kato. His main research interests include image registration, camera calibration, and heterogeneous 2D-3D data fusion.

**Levente Tamas** received the M.Sc. (valedictorian) and the Ph.D. degree in electrical engineering from Technical University of Cluj-Napoca, Romania, in 2005 and 2010, respectively. He took part in several postdoctoral programs dealing with 3D perception and robotics, the most recent one spent at the Bern University of Applied Sciences, Switzerland. He is currently with the Robotics Research Group, Department of Automation, Technical University of Cluj-Napoca. His current research interests include 3D perception and planning for autonomous systems. He is a member of the IEEE Robotics and Automation Society.

**Zoltan Kato** received the MS degree in computer science in 1990 from the Jozsef Attila University, Szeged, Hungary; the PhD degree in 1994 from University of Nice doing his research at INRIA - Sophia Antipolis, France; and the DSc title from the Hungarian Academy of Sciences in 2014. He has been a visiting research associate at the Computer Science Department of the Hong Kong University of Science and Technology; an ERCIM postdoc fellow at CWI, Amsterdam; and a visiting fellow at the School of Computing, National University of Singapore. In 2002, he joined the Institute of Informatics, University of Szeged, Hungary, where he is heading the Research Group on Visual Computation. His research interests include camera calibration, registration, segmentation, multiview reconstruction, statistical image models, Markov random fields, color, texture, motion, shape modeling, variational and level set methods. He is a Senior Member of IEEE and past President of the Hungarian Association for Image Analysis and Pattern Recognition.