

Uncertainty-Aware Camera Pose Estimation from Points and Lines

Alexander Vakhitov¹

Luis Ferraz Colomina²

Antonio Agudo³

Francesc Moreno-Noguer³

¹SLAMcore, UK

²Kognia Sports Intelligence, Spain

³Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Spain

Abstract

Perspective-n-Point-and-Line (PnP(L)) algorithms aim at fast, accurate, and robust camera localization with respect to a 3D model from 2D-3D feature correspondences, being a major part of modern robotic and AR/VR systems. Current point-based pose estimation methods use only 2D feature detection uncertainties, and the line-based methods do not take uncertainties into account. In our setup, both 3D coordinates and 2D projections of the features are considered uncertain. We propose PnP(L) solvers based on EPnP [20] and DLS [14] for the uncertainty-aware pose estimation. We also modify motion-only bundle adjustment to take 3D uncertainties into account. We perform exhaustive synthetic and real experiments on two different visual odometry datasets. The new PnP(L) methods outperform the state-of-the-art on real data in isolation, showing an increase in mean translation accuracy by 18% on a representative subset of KITTI, while the new uncertain refinement improves pose accuracy for most of the solvers, e.g. decreasing mean translation error for the EPnP by 16% compared to the standard refinement on the same dataset. The code is available at <https://alexandervakhitov.github.io/uncertain-pnp/>.

1. Introduction

Camera localization using sparse feature correspondences is a major part of augmented or virtual reality and robotic systems. The Perspective-*n*-Point(-and-Line), or PnP(L), methods can be successfully used to estimate the pose of a calibrated camera from sparse feature correspondences. Line features can increase localization accuracy in man-made self-similar environments which lack surfaces with distinctive textures [36, 43, 15, 29], motivating the use of PnP(L) methods [41].

While vision-based localization with respect to an au-

This work has been partially funded by the Spanish government under projects HuMoUR TIN2017-90086-R, ERA-Net Chistera project IPALM PCI2019-103386 and Maria de Maeztu Seal of Excellence MDM-2016-0656.

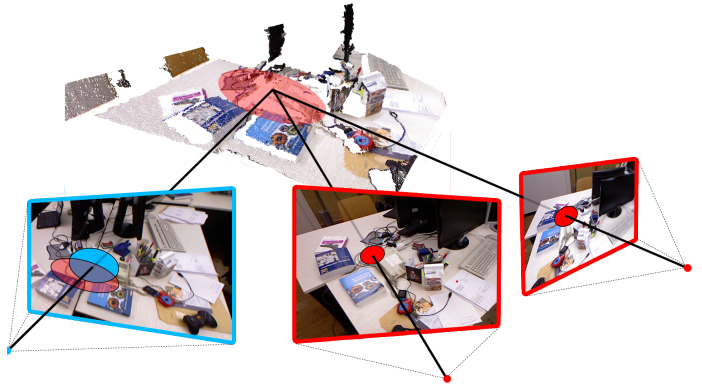


Figure 1. We propose globally convergent PnP(L) solvers leveraging a complete set of 2D and 3D uncertainties for camera pose estimation. A 3D scene model with sparse features is reconstructed from images with known poses (right cameras), and we need to find a pose of a camera on the left. The point has 2D detection uncertainty (blue ellipsoid), and 3D model uncertainty (red ellipsoid in the scene).

tomatically reconstructed map of sparse features is an important part of current robotic and AR/VR systems, commonly used PnP methods treat the features as absolutely accurate [47, 14, 20, 16]. The more recent 2D covariance-aware methods relax this assumption [40, 6], but only for the feature detections, still assuming perfect accuracy of the 3D feature coordinates.

In maps reconstructed with structure-from-motion, 3D feature coordinate accuracy can vary. Stereo triangulation errors grow quadratically with respect to the object-to-sensor distance, so the accuracy in estimating point depth can vary in several orders of magnitude, while the line stereo triangulation accuracy depends on the angle between the line direction and the baseline. Nevertheless, to the best of our knowledge, no prior method for PnP(L) was designed to take both 3D and 2D uncertainties into account.

We propose to integrate the feature uncertainty into the PnP(L) methods, see Fig. 1, which is our main contribution. We build on the classical DLS [14] and EPnP [20]

methods. Additionally, we propose a modification to the standard nonlinear refinement, which is normally used after the PnP(L) solver, to take 3D uncertainties into account. An exhaustive evaluation on synthetic data and on the two real indoor and outdoor datasets demonstrates that new PnP(L) methods are significantly more accurate than state-of-the-art, both in isolation and in a complete pipeline, e.g. the proposed DLSU method reduces the mean translation error on KITTI by 18%, see Section 4. The proposed uncertain pose refinement can improve the pose accuracy by up to 16% in exchange of an extra 5-10% of the computational time. In a synthetic setting with noise in 2D feature detections, the new methods have the same accuracy as the most accurate 2D uncertainty-aware methods [6, 40]. The code is available at <https://alexandervakhitov.github.io/uncertain-pnp/>.

2. Related work

We start with discussing how proposed methods relate to known PnP methods for arbitrary number of correspondences n , PnP(L) and 2D covariance-aware methods.

Perspective- n -point. Geometric gold standard cost [13] for pose estimation for arbitrary n is highly non-convex, and direct pose solvers rely on simplified *algebraic* costs. Early methods [28, 22, 35, 31, 8, 4, 1] were slow and inaccurate.

Starting from [24] fast direct solvers were developed [21, 14, 16, 47]. EPnP [24] was the first to provide fast and accurate pose estimate solving a least-squares system with nonlinear constraints.

EPnP was developed further in [20, 7, 6, 41]: [20] proposed adaptive PCA-based choice of control points and a fast iterative refinement step, [7] designed an EPPnP method for robust pose estimation in presence outliers. Structure-from-motion [34], visual SLAM [25], object pose estimation [38] rely on EPnP due to its robustness and fast computational time. A proposed solver EPnPU is a derivative of EPnP, providing better accuracy when feature uncertainty information is available.

Groebner basis solvers for polynomial systems are more accurate but are computationally more demanding [14, 48, 47, 16, 12]. The DLS method [14] is fast enough for real-time use and has higher accuracy compared to the EPnP, while the most accurate method OPnP [47] is significantly slower. DLS minimizes the *object space error* and uses the Cayley parameterization, and has a singularity which can be avoided [27]. In this work, we build on the DLS and take feature uncertainties into account, however relying on the OPnP-like algebraic cost instead of the object space error.

PnP(L) methods. An early DLT method [13] as well as an algorithm [4] can compute camera pose from n line correspondences, but have inferior accuracy compared to new polynomial solver-based approaches [23, 30, 45, 18, 49]. Extending EPnP or OPnP to PnP(L) [41, 42] is practical

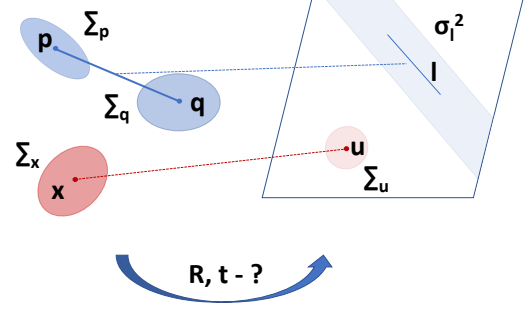


Figure 2. Schematic representation of the PnP(L) problem, see text.

since a PnP(L) method uses all available mixed correspondences at once. In this work, we propose EPnP(L)/DLSU methods, which take line uncertainty into account in order to improve the pose estimates.

PnP with uncertainty. Features are detected with varying uncertainty, and 2D uncertainty-aware PnP methods [6, 40] use it to improve the estimated pose accuracy. CEPPnP [6] builds on EPPnP and inherits the base method’s low computational complexity. MLPnP [40]

is more accurate and computationally demanding than CEPPnP, because it combines both a linear solver and a refinement into one method.

Both CEPPnP and MLPnP use only 2D feature detection uncertainty and work only for points.

In contrast, the approach we present in this paper is more accurate due to the use of both 2D and 3D feature uncertainty and works for a mixed set of line and point correspondences.

3. Method

We start with formulating the problem, and then proceed to introduce the uncertainty-aware pose solvers and the nonlinear refinement method. We conclude the section with describing the approach for obtaining the feature covariances. We denote matrices, vectors and scalars with capital, bold and italic letters, e.g. R , x , γ , and $x^{(i)}$ denotes the i -th component of x .

3.1. PnP(L) with Uncertainty

We are given a set of n_p 3D points $\{x_i\}_{i=1}^{n_p}$ and n_l 3D line segments defined by their endpoints $\{p_i\}_{i=1}^{n_l}$, $\{q_i\}_{i=1}^{n_l}$. Points and line segment endpoints are corrupted by zero-mean Gaussian noises with covariances Σ_{x_i} , Σ_{p_i} and Σ_{q_i} . The point projections $\{u_i\}_{i=1}^{n_p}$ are corrupted with zero-mean Gaussian noises with covariances Σ_{u_i} . The line segments projections $\{l_i\}_{i=1}^{n_l}$ are represented as normalized line coefficients, so $\|l_i^{(1:2)}\| = 1$. We model the line segment detection uncertainty as a zero-mean Gaussian added to the distance between the line and any point on the image

plane, with a variance $\sigma_{l,i}^2$, see Fig. 2. We consider a camera with known intrinsics, assuming that the camera calibration matrix K is an identity matrix.

Our problem is to estimate a rotation matrix R and a translation vector \mathbf{t} aligning the camera coordinate frame with the world frame. We assume knowledge of an estimate of the average scene depth \bar{d} , and we consider also the case when there is a rough initial hypothesis $\hat{R}, \hat{\mathbf{t}}$ available.

3.2. Uncertainty for Pose Estimation Solvers

In this section, we derive methods for uncertainty-aware pose estimation. We find the uncertainties for the algebraic feature projection residuals and incorporate them into the pose solvers in the form of the residual covariances. We start with point features, then move to lines.

Point residuals. Let us parameterize a point in the camera coordinate frame as $\hat{\mathbf{x}}(\theta, \mathbf{x}) = R\mathbf{x} + \mathbf{t}$, where θ encodes the camera parameters R, \mathbf{t} . The algebraic point residual is based on perspective point projection:

$$\mathbf{r}^{pt}(\theta, \mathbf{x}, \mathbf{u}) = \hat{\mathbf{x}}^{(1:2)}(\theta, \mathbf{x}) - \mathbf{u}\hat{\mathbf{x}}^{(3)}(\theta, \mathbf{x}), \quad (1)$$

where \mathbf{u} is the projected point.

By our assumptions \mathbf{x} and \mathbf{u} are corrupted with additive zero-mean Gaussian noises with covariances $\Sigma_{\mathbf{x}}, \Sigma_{\mathbf{u}}$: $\hat{\mathbf{x}}(\theta, \mathbf{x}) = \mathbb{E}\hat{\mathbf{x}} + \boldsymbol{\xi}$ and $\mathbf{u} = \mathbb{E}\mathbf{u} + \boldsymbol{\zeta}$, where $\boldsymbol{\xi}, \boldsymbol{\zeta}$ are zero-mean Gaussian noise vectors. The covariance of $\hat{\mathbf{x}}(\theta, \mathbf{x})$ is:

$$\Sigma_{\hat{\mathbf{x}}} = R\Sigma_{\mathbf{x}}R^T = \begin{bmatrix} \mathbf{S} & \mathbf{w} \\ \mathbf{w}^T & \gamma \end{bmatrix}. \quad (2)$$

Substituting into (1), we obtain

$$\mathbf{r}^{pt} = \mathbb{E}\{\hat{\mathbf{x}}^{(1:2)} - \mathbf{u}\hat{\mathbf{x}}^{(3)}\} + \boldsymbol{\xi}^{(1:2)} - \mathbf{u}\boldsymbol{\xi}^{(3)} - \boldsymbol{\zeta}\hat{\mathbf{x}}^{(3)} - \boldsymbol{\xi}^{(3)}\boldsymbol{\zeta}, \quad (3)$$

omitting the function arguments for clarity. By expressing the covariance of (3) as $\mathbb{E}\mathbf{r}^{pt}(\mathbf{r}^{pt})^T$, using the independence of $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$ we obtain the residual covariance:

$$\Sigma_{\mathbf{r}^{pt}} = \mathbf{S} + \gamma\mathbf{u}\mathbf{u}^T + (\hat{\mathbf{x}}^{(3)})^2\Sigma_{\mathbf{u}} - (\mathbf{u}\mathbf{w}^T + \mathbf{w}\mathbf{u}^T). \quad (4)$$

To compute $\Sigma_{\mathbf{r}^{pt}}$, we need to know R and $\hat{\mathbf{x}}^{(3)}$. If we approximate the model point covariance $\Sigma_{\mathbf{x}} \approx \sigma^2\mathbf{I}$, where \mathbf{I} denotes the identity, then $\Sigma_{\hat{\mathbf{x}}} \approx \sigma^2\mathbf{I}$ as well, as follows from (2). If we have a rough pose hypothesis $\hat{R}, \hat{\mathbf{t}}$, we can use it instead to approximate $\Sigma_{\mathbf{r}^{pt}}$. We propose the new solvers in two modifications. In the first case, the solver uses the average scene depth estimate \bar{d} to approximate the point depths, and an isotropic approximation to 3D point covariance (see Section 3.6 below), we dub these solvers EPnPU and DLSU. In the second case, the solver uses the pose hypothesis, typically available as an output of the RANSAC loop, to approximate the point depths and compute the 3D point covariance estimates; we denote these methods DLSU*, EPnPU*.

Line residuals. We are given the normalized 2D line segment coefficients \mathbf{l} as well as the segment 3D endpoints \mathbf{p}, \mathbf{q} , and consider the algebraic line residual following [41]:

$$\mathbf{r}^{ln}(\theta, \mathbf{p}, \mathbf{q}, \mathbf{l}) = \begin{bmatrix} \mathbf{l}^T\hat{\mathbf{p}}(\theta, \mathbf{p}) \\ \mathbf{l}^T\hat{\mathbf{q}}(\theta, \mathbf{q}) \end{bmatrix}, \quad (5)$$

where $\hat{\mathbf{p}}(\theta, \mathbf{p}), \hat{\mathbf{q}}(\theta, \mathbf{q})$ are the 3D endpoints in the camera coordinate frame, ln stands for 'line'. Let us decompose the endpoint $\hat{\mathbf{p}}(\theta, \mathbf{p}) = \mathbb{E}\hat{\mathbf{p}}(\theta, \mathbf{p}) + \boldsymbol{\eta}_{\mathbf{p}}$, where $\boldsymbol{\eta}_{\mathbf{p}}$ has the covariance $\Sigma_{\hat{\mathbf{p}}} = R\Sigma_{\mathbf{p}}R^T$. Under our model for the line detection noise, the noise-corrupted signed line-point distance is $\mathbf{l}^T\mathbf{y}_h = \mathbb{E}\{\mathbf{l}^T\}\mathbf{y}_h + \boldsymbol{\nu}_y$, where $\boldsymbol{\nu}_y$ is a zero-mean Gaussian with variance σ_l^2 , \mathbf{y}_h is an arbitrary image point, in homogeneous coordinates. If \mathbf{y}_h is a projection of a point $\mathbf{y} = \lambda_{\mathbf{y}}\mathbf{y}_h$ with depth $\lambda_{\mathbf{y}}$, then $\mathbf{l}^T\mathbf{y} = \mathbb{E}\{\mathbf{l}^T\mathbf{y}\} + \lambda_{\mathbf{y}}\boldsymbol{\nu}_y$. Therefore,

$$\mathbf{l}^T\hat{\mathbf{p}}(\theta, \mathbf{p}) = \mathbb{E}\{\mathbf{l}^T\hat{\mathbf{p}}(\theta, \mathbf{p})\} + \lambda_{\mathbf{p}}\boldsymbol{\nu}_{\mathbf{p}} + \mathbf{l}_i^T\boldsymbol{\eta}_{\mathbf{p}_i}, \quad (6)$$

where $\boldsymbol{\nu}_{\mathbf{p}_i}$ is the line detection noise. The variance is

$$\mathbb{E}(\mathbf{l}^T\hat{\mathbf{p}} - \mathbb{E}\mathbf{l}^T\hat{\mathbf{p}})^2 = \lambda_{\mathbf{p}}^2\sigma_l^2 + \mathbf{l}^T\Sigma_{\hat{\mathbf{p}}}\mathbf{l}, \quad (7)$$

where we omit the function arguments for brevity. Under our model, the noise in \mathbf{p}, \mathbf{q} and the line detection noises are assumed independent. We acknowledge that this is a simplification, however it speeds up the computations, and works in practice, as we show in the experiments. Moreover, other works, e.g. [29], rely on such a model as well, while in the offline setting one could follow [10] in using more advanced noise models for the lines. The covariance for the line residual is

$$\Sigma_{\mathbf{r}^{ln}} = \sigma_l^2\text{diag}(\lambda_{\mathbf{p}}^2, \lambda_{\mathbf{q}}^2) + \text{diag}(\mathbf{l}^T\Sigma_{\hat{\mathbf{p}}}\mathbf{l}, \mathbf{l}^T\Sigma_{\hat{\mathbf{q}}}\mathbf{l}). \quad (8)$$

In order to compute the covariance we need R and the point depths $\lambda_{\mathbf{p}}, \lambda_{\mathbf{q}}$. For EPnPLU and DLSLU, we approximate the point depths with \bar{d} given in the problem formulation and the covariances $\Sigma_{\hat{\mathbf{p}}_i}, \Sigma_{\hat{\mathbf{q}}_i}$ as isotropic, for EPnPLU* and DLSLU* we use a pose hypothesis $\hat{R}, \hat{\mathbf{t}}$ to compute these values.

So far we obtained a general form of residual covariances for point and line features under our noise model. Next we show, how to use it in the PnP(L) solvers.

3.3. EPnP with Uncertainty

We generalize the EPnP [20] and EPnPL [41] to leverage 2D and 3D uncertainties in pose prediction. EPnP starts with computing an SE3-invariant barycentric representation $\boldsymbol{\alpha}$ of a point \mathbf{x} :

$$\mathbf{x} = \mathbf{C}\boldsymbol{\alpha}, \quad (9)$$

where $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_4]$ is a matrix of the four specifically chosen *control points* in the world coordinate frame.

[20] proposed to choose the first point \mathbf{c}_1 as a mean of \mathbf{x}_i and $\mathbf{c}_2, \mathbf{c}_3$ and \mathbf{c}_4 as the maximum variance directions computed using principal component analysis (PCA). Preliminary experiments show, that when 3D noise is added to \mathbf{x} , the accuracy of the PCA version degrades. This motivated us to modify the control points choice to use the 3D uncertainties. We can get a straightforward theoretically solid PCA generalization under isotropic approximation of 3D point covariances $\Sigma_{\mathbf{x}} \approx \sigma_{\mathbf{x}}^2 \mathbf{I}$, where $\sigma_{\mathbf{x}}^2 = \frac{1}{3} \text{trace}(\Sigma_{\mathbf{x}})$. In particular, classical PCA solves a following problem to obtain the j -th principal direction:

$$\sum_i (\mathbf{z}^T \tilde{\mathbf{x}}_i)^2 \rightarrow \max_{\mathbf{z}} \text{ s.t. } \|\mathbf{z}\| = 1, \quad (10)$$

where $\tilde{\mathbf{x}}_i$ are the centered points with subtracted projections on the first $j - 1$ components, and \mathbf{z} is the sought principal direction. The covariance is $\text{cov}(\mathbf{z}^T \tilde{\mathbf{x}}) = \mathbf{z}^T \tilde{\mathbf{x}} \tilde{\mathbf{x}}^T \mathbf{z} = \mathbf{z}^T \Sigma_{\tilde{\mathbf{x}}} \mathbf{z} = \sigma_{\tilde{\mathbf{x}}}^2$ using the fact that $\|\mathbf{z}\| = 1$. Then, we modify the problem as

$$\sum_i \sigma_{\tilde{\mathbf{x}},i}^{-2} (\mathbf{z}^T \tilde{\mathbf{x}}_i)^2 \rightarrow \max_{\mathbf{z}} \text{ s.t. } \|\mathbf{z}\| = 1, \quad (11)$$

see more details in the supp. mat.

The camera pose in EPnPL is represented through the control points in the camera coordinate frame, so $\boldsymbol{\theta}_{EPnP} = \hat{\mathbf{C}} = [\hat{\mathbf{c}}_1, \dots, \hat{\mathbf{c}}_4]$, and $\hat{\mathbf{c}}_i = \mathbf{R}\mathbf{c}_i + \mathbf{t}$. The camera frame point is $\hat{\mathbf{x}}(\boldsymbol{\theta}_{EPnP}, \mathbf{x}) = \hat{\mathbf{C}}\boldsymbol{\alpha}(\mathbf{x})$. EPnP(L) uses the algebraic residuals for lines and points (1, 5), solving a problem

$$\|\text{Mvec}(\hat{\mathbf{C}})\|^2 \rightarrow \min_{\hat{\mathbf{C}}}, \quad (12)$$

where $\text{vec}(\cdot)$ denotes a vectorized matrix. The solution is given by an eigendecomposition of a 12×12 matrix $\mathbf{M}^T \mathbf{M}$. The method then looks for $\hat{\mathbf{C}}$ in the subspace of the eigenvectors of $\mathbf{M}^T \mathbf{M}$ with smallest eigenvalues.

The proposed EPnP(L)U method follows the same strategy, constructing the uncertainty-augmented matrix \mathbf{M}_U . In the previous section we noted, that to estimate the covariances of the point and line algebraic residuals (1, 5) we need to know \mathbf{R} and \mathbf{t} . However, approximating the point covariances by isotropic covariance matrices as $\Sigma_{\mathbf{x}} \approx \sigma_{\mathbf{x}}^2 \mathbf{I}$, $\Sigma_{\mathbf{p}} \approx \sigma_{\mathbf{p}}^2 \mathbf{I}$, $\Sigma_{\mathbf{q}} \approx \sigma_{\mathbf{q}}^2 \mathbf{I}$, where $\sigma^2 = \frac{1}{3} \text{trace}(\Sigma)$, we get rid of the dependency on \mathbf{R} in the residuals. By replacing the point depth with the approximate scene depth \bar{d} , we compute the point (1) and line (5) residual covariances as

$$\Sigma_{\mathbf{r}_{pt}}^{EPnP} = \sigma_{\mathbf{x}}^2 \mathbf{I} + \bar{d}^2 \Sigma_{\mathbf{u}} + \sigma_{\mathbf{x}}^2 \mathbf{u} \mathbf{u}^T. \quad (13)$$

$$\Sigma_{\mathbf{r}_{ln}}^{EPnP} = \sigma_l^2 \bar{d}^2 \mathbf{I} + \|\mathbf{l}\|^2 \text{diag}(\sigma_{\mathbf{p}}^2, \sigma_{\mathbf{q}}^2). \quad (14)$$

We also consider a case when a rough pose hypothesis is given. In this case, we still use the isotropic approximation of uncertainties, but use the pose to compute estimates of the depths of points. The method proceeds as the basis version. Next we describe our DLS-based approach.

3.4. DLS with Uncertainty

The DLS method [14] employs Cayley rotation parameterization to solve a least-squares polynomial system of the algebraic residuals for the point correspondences with the Groebner basis techniques. It relies on so-called *object space error PnP*, when one minimizes the distance between the backprojection ray of the point detection and the 3D point. However, we decided to use the algebraic residual (1), which allows for faster computations and results in a method with similar accuracy, see supp.mat. for the comparison. DLS performs eigendecomposition of a 27×27 matrix. We keep the Cayley parameterization of DLS, but reformulate the equations, and generate the new solver of the same dimension using the generator of [19].

The DLS uses the following parameterization of a point in camera coordinates: $\hat{\mathbf{x}}(\boldsymbol{\theta}, \mathbf{x}) = \mathbf{R}(\mathbf{s})\mathbf{x} + \mathbf{t}$, so $\boldsymbol{\theta}_{DLS} = [\mathbf{s}^T, \mathbf{t}^T]^T$, where $\mathbf{s} \in \mathbb{R}^3$ is a vector of the Cayley rotation parameters:

$$\mathbf{R}(\mathbf{s}) = \frac{1}{1 + \|\mathbf{s}\|^2} ((1 - \mathbf{s}\mathbf{s}^T)\mathbf{I} + 2[\mathbf{s}]_x + 2\mathbf{s}\mathbf{s}^T), \quad (15)$$

$[\mathbf{s}]_x$ denotes a cross product matrix. We use the residuals for lines and points (5, 1) with the camera parameterization $\boldsymbol{\theta}_{DLS}$. The residual covariances are obtained as in a case of EPnP. The point or line residual $\mathbf{r}_k(\mathbf{s}, \mathbf{t})$ can be expressed using the DLS parameterization as

$$\mathbf{r}_k(\mathbf{s}, \mathbf{t}) = \mathbf{A}_k \text{vec}(\mathbf{R}(\mathbf{s})) + \mathbf{T}_k \mathbf{t}, \quad (16)$$

and we denote its covariance as $\Sigma_{\mathbf{r}_k}$. The cost function of the method is

$$\frac{1}{2} \sum_{k=1}^{n_r} \mathbf{r}_k^T(\mathbf{s}, \mathbf{t}) \Sigma_{\mathbf{r}_k}^{-1} \mathbf{r}_k(\mathbf{s}, \mathbf{t}) \rightarrow \min_{\mathbf{s}, \mathbf{t}}. \quad (17)$$

We constrain the gradient of the cost by \mathbf{t} to be zero, express \mathbf{t} using the remaining unknowns and obtain the cost that depends only on $\mathbf{R}(\mathbf{s})$. Following DLS, we multiply this cost by $(1 + \|\mathbf{s}\|^2)^2$, so it becomes a polynomial of \mathbf{s} . We constrain its gradient by \mathbf{s} to be zero and obtain a third order polynomial system with three unknowns. It is solved using the generated solver, then \mathbf{t} is found using an expression obtained before, see the details of the derivation in the supp. mat.

To improve accuracy, we also use an optional non-linear refinement stage by refining the cost (17) with a Newton method starting from the output of a solver, computing the Hessian of the cost analytically.

One often refines the output of the pose solver with non-linear minimization of the gold-standard feature reprojection errors [13]. In the following section we propose a new formulation of a refinement method in order to take the full set of feature uncertainties into account.

3.5. Uncertainty-aware Pose Refinement

When the structure is fixed, to obtain optimal estimates of the camera pose one uses *motion-only* bundle adjustment [39], that is formulated as a non-linear least squares-based log-likelihood maximization. In feature-based pose estimation, one runs it as a final refinement step, initializing with the output of a pose solver. A standard 2D covariance-aware formulation of the *motion-only* bundle adjustment cost is:

$$\mathcal{L}(\theta) = \sum_{i=1}^{n_p} \|\bar{\mathbf{r}}_i^{pt}\|_{\Sigma_{\bar{\mathbf{r}}_i^{pt}}}^2 + \sum_{i=1}^{n_l} \|\bar{\mathbf{r}}_i^{ln}\|_{\Sigma_{\bar{\mathbf{r}}_i^{ln}}}^2 \rightarrow \min \theta, \quad (18)$$

where θ is the camera pose, $\bar{\mathbf{r}}_i^{pt}$, $\bar{\mathbf{r}}_i^{ln}$ are the gold standard point and line feature residuals, and $\|\cdot\|_{\Sigma}$ denotes the Mahalanobis distance with covariance Σ . The 'gold standard' residual for a point [13] is

$$\bar{\mathbf{r}}^{pt}(\mathbf{x}, \mathbf{R}, \mathbf{t}) = \mathbf{u} - \pi(\mathbf{R}\mathbf{x} + \mathbf{t}), \quad (19)$$

where the projection function π is defined as $\pi(\hat{\mathbf{x}}) := \frac{1}{\hat{\mathbf{x}}^{(3)}} \mathbf{x}^{(1:2)}$, and $\hat{\mathbf{x}} = \mathbf{R}\mathbf{x} + \mathbf{t}$. A 'gold standard' residual for a line is

$$\bar{\mathbf{r}}^{ln}(\mathbf{p}, \mathbf{q}, \mathbf{R}, \mathbf{t}) = \begin{bmatrix} \mathbf{1}^T \pi(\mathbf{R}\mathbf{p} + \mathbf{t}) \\ \mathbf{1}^T \pi(\mathbf{R}\mathbf{q} + \mathbf{t}) \end{bmatrix}. \quad (20)$$

Visual odometry systems, e.g. [25, 26], use the 2D covariance of the feature detection as the residual covariance. This corresponds to setting $\Sigma_{\bar{\mathbf{r}}^{pt}} = \Sigma_{\mathbf{u}}$, $\Sigma_{\bar{\mathbf{r}}^{ln}} = \sigma_l^2 \mathbf{I}$, and we will dub this scheme as *standard* refinement. It is often implemented based on a fast and efficient Levenberg-Marquardt method.

In our case, we wish to use the full residual covariance, including both 2D and 3D uncertainty. For the point residual it is

$$\Sigma_{\bar{\mathbf{r}}^{pt}} = \Sigma_{\mathbf{u}} + \mathbf{J}(\hat{\mathbf{x}}) \mathbf{R} \Sigma_{\mathbf{x}} \mathbf{R}^T \mathbf{J}^T(\hat{\mathbf{x}}), \quad (21)$$

where \mathbf{J} is a Jacobian of π with respect to $\hat{\mathbf{x}}$; for the line residual the covariance is

$$\Sigma_{\bar{\mathbf{r}}^{ln}} = \sigma_l^2 \mathbf{I} + \text{diag}(\mathbf{1}^T \Sigma_{\hat{\mathbf{p}}} \mathbf{1}, \mathbf{1}^T \Sigma_{\hat{\mathbf{q}}} \mathbf{1}), \quad (22)$$

where $\Sigma_{\hat{\mathbf{f}}}^{\pi} = \mathbf{J}(\hat{\mathbf{f}}) \mathbf{R} \Sigma_{\mathbf{f}} \mathbf{R}^T \mathbf{J}^T(\hat{\mathbf{f}})$, for $\hat{\mathbf{f}} = \{\hat{\mathbf{p}}, \hat{\mathbf{q}}\}$, $\mathbf{f} = \{\mathbf{p}, \mathbf{q}\}$.

The covariances in the form (21, 22) are not constant with respect to the camera pose. They cannot be used in a classical Gauss-Newton scheme. The cost (18) can be minimized using non-linear minimization, defining a *full uncertain* refinement method.

However, the *full uncertain* refinement has a downside of being computationally inefficient compared to a *standard* refinement. Therefore, we propose a technique resembling Iterative Reweighted Least Squares, in which we make Gauss-Newton iterations, but update the estimate of

the covariances (21, 22) on each step. We call it (*iterative*) *uncertain* refinement. This technique results in similar accuracy to the *full uncertain* refinement, but in terms of computational efficiency is comparable to the *standard* refinement, as we show in the experiments section below. In the following section, we explain our approach to obtaining the point and line uncertainties.

3.6. Obtaining the Uncertainties

The 2D feature uncertainties can be obtained from a feature detector, e.g. for a multiscale pyramidal detector with a scale step of κ we estimate $\Sigma_{\mathbf{u}} = \sigma_o^2 \mathbf{I}$, $\sigma_l^2 = \sigma_o^2$, where $\sigma_o = \kappa^{o-1} \epsilon$, and o is a level of the image pyramid to which the feature belongs, ϵ is the feature detection accuracy.

Uncertainty for the 3D point can be estimated after the triangulation following a standard error propagation technique, e.g. [13], Chapter 5. While there exists a single natural 3D point parameterization, the situation with line features is less clear. The line covariance formulation depends on the representation used for line triangulation, and there are several known parameterizations of lines, see [5, 46, 29]. As long as we represent a line in 3D through the endpoints of some 3D line segment, we require a method to accurately find these endpoints and their covariances. We reconstruct the endpoints as the unknowns, and for the first camera we use the point-based reprojection residuals, while for the other cameras we use the line-based residuals, see the supp. mat. This way, we can use error propagation to obtain the line endpoint uncertainties after triangulating the line.

If we have an arbitrary positive semi-definite covariance $\Sigma_{\mathbf{x}}$ of a 3D point, an isotropic approximation for it would be $\sigma_{\mathbf{x}}^2 \mathbf{I}$, where $\sigma_{\mathbf{x}}^2 = \frac{1}{3} \text{trace}(\Sigma_{\mathbf{x}})$. This approximation is optimal in the Frobenius norm sense: $\|\Sigma_{\mathbf{x}} - \sigma_{\mathbf{x}}^2 \mathbf{I}\|_F^2 = \sum_{i=1}^3 (\rho_i^2 - \sigma_{\mathbf{x}}^2)^2$, where ρ_i^2 are the singular values of $\Sigma_{\mathbf{x}}$.

We have described the new methods and now continue with evaluating them in a synthetic and real settings.

4. Experiments

We compare the proposed uncertainty-aware pose solvers to competitive pose estimation methods, in isolation and combined with a *standard* or *uncertain* refinement, see Section 3.5. We use asterisk * to denote the methods receiving pose hypothesis. We compare against EPnP [20], DLS [14], 2D covariance-aware methods CEPPnP [6] and MLPnP [40], the state-of-the-art PnP method OPnP [47] in case of points, and EPnPL and OPnPL [41] in case of points and lines mixture.

We use RANSAC [9] with P3P [17] to estimate inliers before feeding them into the solvers, while also comparing against the P3P baseline which does not use any PnP pose solver. After the solvers, we optionally run inlier filtering using the obtained pose, followed by a motion-only bundle

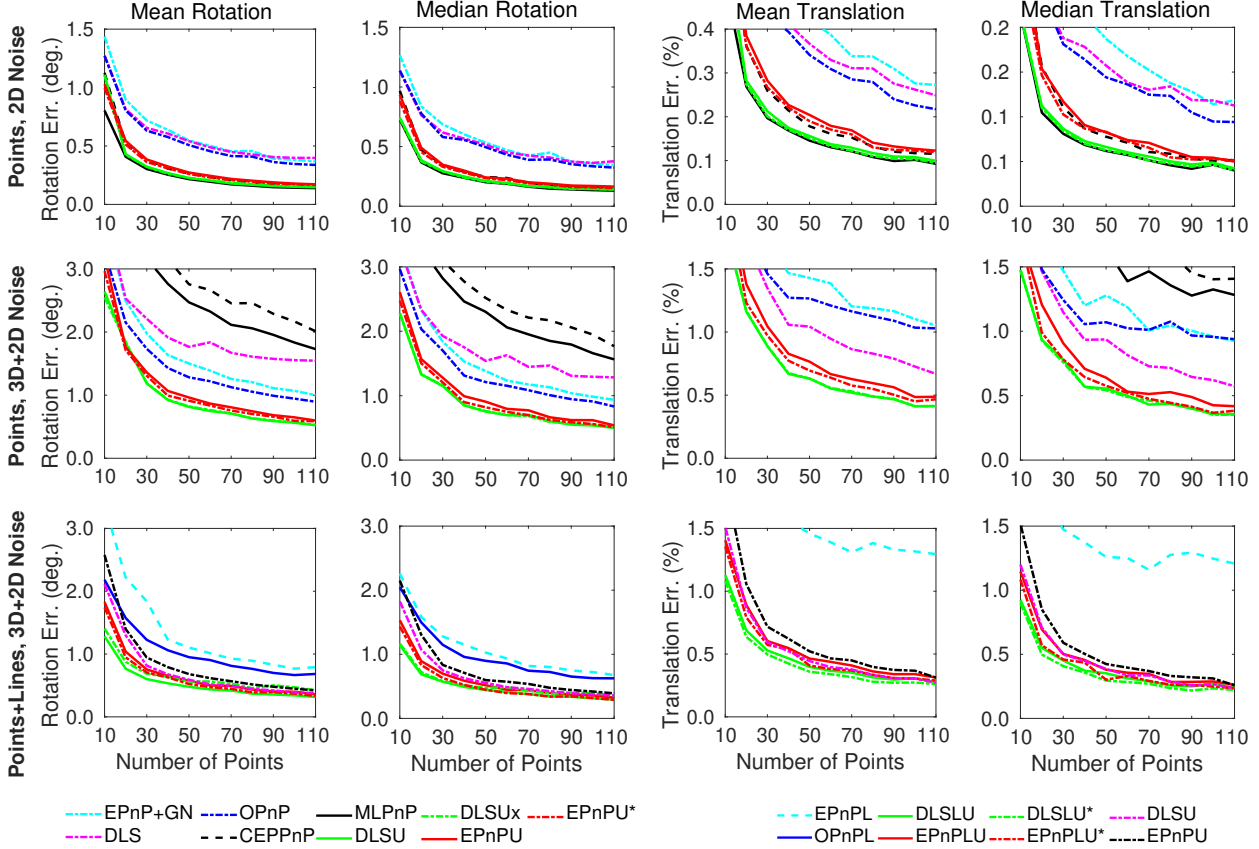


Figure 3. Pose errors in synthetic experiments. Point-based pose estimation, increasing the number of points in case of 2D noise (**Top**) and 2D+3D noise (**Center**), left legend. **Bottom**: increasing the number of lines and points in case of 2D+3D noise, right legend. We report mean and median rotation and translation errors. Asterisk denotes rough pose hypothesis input. In case of 2D noise the new approaches reach the state-of-the-art accuracy, in case of 3D noise they outperform the published methods. In case of the line features, the new solvers outperform the published EPnP and OPnP as well as the proposed uncertainty aware point-only methods. The access to a pose hypothesis does not result in better accuracy.

adjustment, inspired by the localization modules of ORB-SLAM2 [26] or COLMAP [34]. We use MATLAB implementations of the methods, run our experiments on a laptop with Core i7 1.3 GHz with 16Gb RAM.

4.1. Synthetic experiments

In the synthetic setting we compare the proposed pose estimation methods EPnP(L)U and DLS(L)U against the baselines in isolation, as well as the proposed *uncertain* refinement against the standard refinement, as defined in Section 3.5.

Metrics. We evaluate the results in terms of the absolute rotation error $e_{\text{rot}} = |\text{acos}(0.5(\text{trace}(\mathbf{R}_{\text{true}}^T \mathbf{R}) - 1))|$ in degrees and relative translation error $e_{\text{trans}} = \|\mathbf{t}_{\text{true}} - \mathbf{t}\| / \|\mathbf{t}_{\text{true}}\| \times 100$, in %, where \mathbf{R}_{true} , \mathbf{t}_{true} is the true pose and \mathbf{R} , \mathbf{t} is the estimated one.

Data generation. We assume a virtual calibrated camera with an image size of 640×480 pix., a focal length of 800 and a principal point in the image center. 3D points and endpoints of 3D line segments are generated in

the box $[-2, 2] \times [-2, 2] \times [4, 8]$ defined in camera coordinates. 3D-to-2D correspondences are then defined by projecting the 3D points under the random rotation matrix and translation vector. We move the 3D line endpoints randomly along the line by a randomly generated Gaussian shift with a standard deviation equal to 10% of the 3D line length, see [41].

We add noise of varying magnitude to the 2D point or line endpoint projections, as well as to the 3D points or line endpoints,

splitting n_{pt} points into 10 subsets with an equal number of points in order to introduce differences in the noise magnitude. Each subset is corrupted by Gaussian noise with an increasing value of standard deviation, from $\sigma = 0.05$ to $\sigma = 0.5$. We consider anisotropic covariances, which are computed by randomly picking a rotation and a triplet $\{\sigma, \sigma_1, \sigma_2\}$, where σ_1, σ_2 are random values chosen within the interval $(0, \sigma]$. The covariance axes are scaled and rotated according to a triplet of standard deviations and the rotation value, respectively. We perform exactly the same

	Points								Points + 2D Uncertainty				Points + Full Uncertainty, Proposed							
	P3P [17]		EPnP [20]		DLS [14]		OPnP [47]		CEPPnP [6]		MLPnP [40]		EPnP ^U *		EPnP ^U		DLSU*		DLSU	
	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}
KITTI [11], sequences 00-02																				
N	8.6	35.2	4.5	24.0	5.5	18.1	7.8	277.6	8.2	49.5	5.8	27.2	4.2	22.2	5.1	23.9	5.6	32.2	6.0	14.9
S	5.1	14.4	4.0	12.8	5.0	12.2	7.2	242.2	5.3	20.6	5.3	14.4	3.7	12.6	3.9	13.1	5.1	25.5	5.1	12.1
U	5.0	14.0	3.5	13.2	5.0	12.9	7.6	325.5	5.1	17.4	6.3	35.3	3.3	10.6	3.5	13.4	5.0	12.6	5.0	10.9
TUM [37], 'freiburg1' sequences																				
N	15.7	3.3	9.5	1.5	9.3	1.4	10.0	1.5	10.0	1.7	10.0	1.6	9.3	1.3	9.4	1.4	9.3	1.3	9.1	1.2
S	9.2	1.2	9.0	1.2	9.0	1.2	9.7	1.3	9.4	1.3	9.2	1.2	9.0	1.2	9.0	1.2	9.0	1.2	9.0	1.2
U	9.1	1.2	9.0	1.2	9.0	1.2	10.3	1.3	9.2	1.2	9.6	2.0	9.0	1.1	9.0	1.2	9.0	1.1	9.0	1.1

Table 1. Motion estimation from 2D-3D point correspondences on KITTI [11] TUM [37] in terms of mean absolute rotation e_{rot} (in $0.1 \times \text{deg.}$) and translation e_{trans} (in cm.) errors. We compare proposed full uncertainty-aware methods against point-based PnP and 2D uncertainty-aware methods in isolation (N), with standard (S) and proposed uncertain (U) refinement. Methods with '*' receive a pose from RANSAC, best for the dataset is in bold italic, best for each protocol (N,S or U) is in bold. The new methods outperform the baselines in most metrics, e.g. DLSU in isolation improves e_{trans} on KITTI by **3 cm (18%)** compared to the best performing baseline DLS. *Uncertain* (U) is mostly better than standard (S) for the proposed methods, e.g. e_{trans} by **2 cm (16%)** for EPnP^U* on KITTI.

addition of noise to the 3D endpoints of line segments. We add noise with different variance to the point and line end-point projections using the same mechanism increasing the standard deviation from $\sigma = 1$ to $\sigma = 10$.

We perform 400 simulation trials. The experiment settings are consistent with [47, 7, 6, 41]. We evaluate the pose solvers in isolation in a point-only and a point+line setting, providing the methods marked by '*' a pose hypothesis computed from a randomly chosen subset of three points using P3P [17]. We change $n_{\text{pt}} = 10$ to 110, in two different setups: introducing noise to the projected 2D features and introducing noise also to the 3D points or endpoints. In the case of experiments with lines and points, we generate $n_l = n_{\text{pt}}$ line correspondences in addition to points.

Results. Fig. 3 summarizes the results of the experiments. In the 2D noise experiment for points (top row), the proposed methods perform similarly with 2D methods, however for $n_{\text{pt}} < 30$ the MLPnP delivers slightly better results, probably due to additional reprojection error refinement step used in this method and not used in the other ones. When we use both 3D and 2D noise for the points (central row), the proposed methods are the most accurate, followed by the classical PnP solvers, and the 2D covariance-based methods. In point+line experiment, the new methods clearly outperform the baselines. Fig. 4 shows an analysis in terms of computational cost. The fastest are EPnP^U, EPnP, CEPPnP, MLPnP, followed by DLSU, DLS and OPnP.

In Fig. 5, we compare the proposed *uncertain* refinement against the *standard* method for point features, see Section 3.5. For the inlier filtering, we use a threshold $\tau^2 = 6^2$ for the covariance-weighted squared residuals (19) corresponding to the *standard* or the *uncertain* refinement. The data is generated as in the experiment with 3D and 2D noise. We consider EPnP, MLPnP and EPnP^U*. The *uncertain* refinement is beneficial for all considered solvers; the margin between different pose solvers after refinement

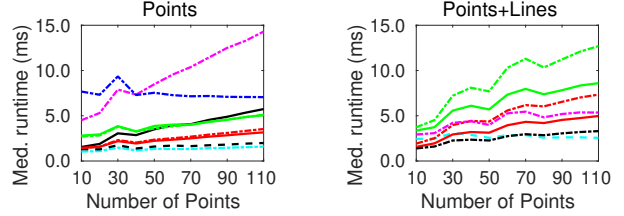


Figure 4. Runtime (ms). Methods based on points (left) or points and lines (right). See Fig. 3 for the legends.

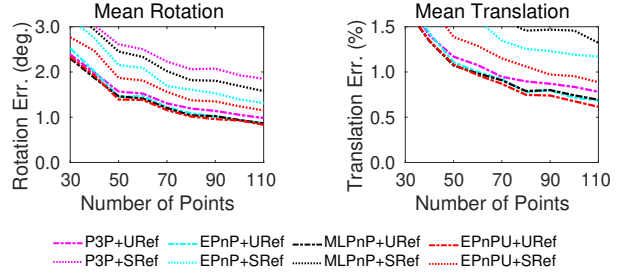


Figure 5. Comparison of methods with standard(+SRef) or uncertain(+URef) refinement, in a 2D + 3D noise setup as in Fig. 3, central row. Uncertain refinement improves accuracy, the uncertainty-aware EPnP^U* is slightly better than the other methods.

decreases, but remains, because the more accurate pose solvers can provide a better set of inliers for the final refinement step; see additional results on timing and comparison against the *full uncertain* refinement in the supp. mat.

Summarizing, the new methods outperform baselines in a synthetic setting. In the next section, we show that the same holds for the real scenarios.

4.2. Real experiments

Data. We use three monocular RGB sequences 00-02 from the KITTI dataset [11] and the first three 'freiburg1' monocular RGB sequences of the TUM-RGBD dataset [37] to evaluate the methods. We use KITTI in a monocular

	Points+Lines				Points+Lines+Uncertainty			
	EPnPL [41]	OPnPL [41]	DLSLU*	EPnPLU*				
	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}
N	2.5	37.1	10.2	650.1	6.3	18.2	3.4	25.2
S	1.8	20.4	6.8	267.4	5.2	12.2	1.8	9.8
U	1.4	12.1	9.0	497.7	5.2	12.0	1.4	9.3

Table 2. Motion estimation from 2D-3D point and line correspondences on KITTI [11] sequences 00-02. We report the mean rotation errors in $0.1 \times$ degrees and translation errors in cm, for the solvers in isolation (N), after *standard* (S) and uncertain (U) refinement, see Section 3.5. Proposed EPnPLU*, DLSLU* mostly outperform the baselines OPnPL and EPnPL, e.g. e_{trans} by **23%-52% (3 - 11 cm.)**, while EPnPL has the best rotation accuracy in isolation.

	P3P	EPnP	DLS	OPnP	CEPnP	MLnP	EPnPU	DLSU
N	3.1	4.6	16.5	10.7	5.0	7.6	6.1	8.0
S	12.1	13.2	25.1	19.5	13.7	16.0	14.7	16.7
U	11.3	12.7	24.5	18.8	13.1	15.2	13.9	16.0

Table 3. Average running time (ms) for the compared methods on KITTI in isolation (N), with standard (S) or uncertain (U) refinement.

mode, taking a temporal window of two left frames (three frames with a pose distance $> 2.5\text{cm}$ for TUM) to detect and describe features, relying on FAST [32] and ORB [33] for points and EDLines [3] and LBD [44] for lines, OpenCV implementations. We use an image pyramid with $n_o = 8$ levels and a factor $\kappa = 1.2$, the detection error $\epsilon = 1$ pix, the uncertainty is calculated as described in Section 3.6. The features are matched using standard brute-force approach, and triangulated using ground-truth camera poses. Triangulation results are refined with Ceres [2], producing also the 3D feature covariances, see supp. mat. for the detailed formulation. The next left frame in KITTI (the next RGB frame in TUM) is used for evaluation. Line detections are filtered by length (less than 25 pix. removed). We use a threshold of $\tau = 5.991$ for the covariance-weighted residual norms in RANSAC. In case of point+line combination, we generate minimal sets using only points, include the lines into the motion-only bundle adjustment for the line-aware solvers.

Protocol and metrics. We compare the methods using absolute rotation error in deg. and absolute translation error in cm. If a pose solver fails or produces less than 3 inliers, we do not use its output, but use the output of RANSAC instead, following [34, 26].

Results. We evaluate the pose solvers in isolation, see Table 1, and show a significant increase in accuracy compared to the state-of-the-art, e.g. DLSU on KITTI outperforms the closest baseline DLS by 18% in mean translation. On the TUM sequences, the mean rotation errors of the proposed methods are similar to the ones of the baselines, while there are more significant gains in translation

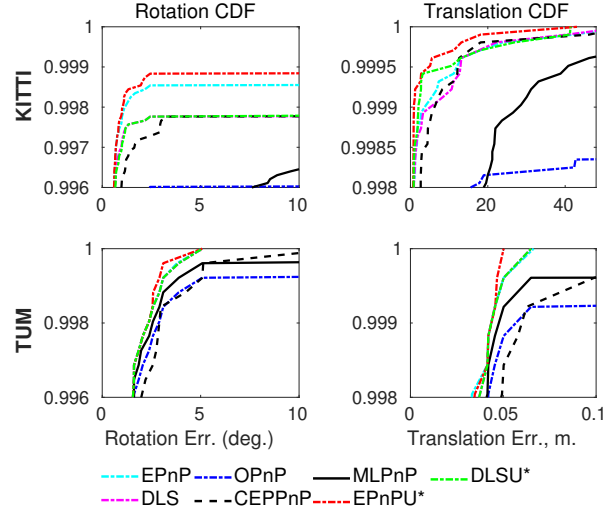


Figure 6. CDF plots for real experiments on KITTI [11] and TUM [37], U mode. EPnPU* and DLSU* are the most accurate.

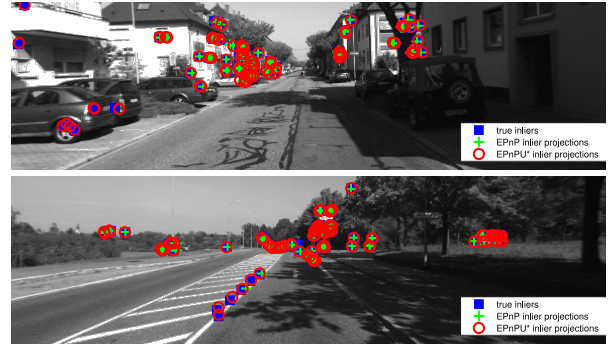


Figure 7. Comparison of inlier sets from EPnPU* (red) and EPnP (green), blue squares show the true inliers. EPnPU* inlier sets are more complete.

errors. The proposed *uncertain* refinement improves accuracy over standard refinement for most of the solvers, e.g. by 16% in case of EPnP on KITTI, see Table 1 and CDF error plots in Fig. 6; the running time of the methods is in Table 3. In the supp. mat. we give additional results, including median errors. See Fig. 7 for a visual comparison of the inlier sets estimated by EPnP and EPnPU*.

In Table 2 we report the mean errors of the points-and-lines-based pose estimation for the proposed solvers. We observe an improvement in the translation errors by almost 50% for the solvers in isolation and by 24% after the standard refinement, compared to the uncertainty-free EPnPL and OPnPL [41] methods.

5. Conclusions

We have generalized PnP(L) methods to estimate the camera pose with uncertain 2D feature detections and 3D feature locations and proposed a new pose refinement scheme. Our methods demonstrate increased accuracy and

robustness both in synthetic and real experiments.

References

- [1] Y. Abdel-Aziz, H. Karara, and M. Hauck. Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. *Photogrammetric Engineering & Remote Sensing*, 81(2):103–107, 2015. 2
- [2] S. Agarwal and K. Mierle. Ceres solver: Tutorial & reference. *Google Inc*, 2:72, 2012. 8
- [3] C. Akinlar and C. Topal. EDLines: A real-time line segment detector with a false detection control. *Pattern Recognition Letters*, 32(13):1633–1642, 2011. 8
- [4] A. Ansar and K. Daniilidis. Linear pose estimation from points or lines. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):578–589, 2003. 2
- [5] A. Bartoli and P. Sturm. Structure-from-motion using lines: Representation, triangulation, and bundle adjustment. *Computer vision and image understanding*, 100(3):416–441, 2005. 5
- [6] L. Ferraz, X. Binefa, and F. Moreno-Noguer. Leveraging feature uncertainty in the PnP problem. In *Proceedings of the BMVC 2014 British Machine Vision Conference*, pages 1–13, 2014. 1, 2, 5, 7
- [7] L. Ferraz, X. Binefa, and F. Moreno-Noguer. Very fast solution to the PnP problem with algebraic outlier rejection. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 501–508. IEEE, 2014. 2, 7
- [8] P. D. Fiore. Efficient linear solution of exterior orientation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (2):140–148, 2001. 2
- [9] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 5
- [10] W. Förstner and B. P. Wrobel. *Photogrammetric computer vision*. Springer, 2016. 3
- [11] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 7, 8
- [12] S. Hadfield, K. Lebeda, and R. Bowden. HARD-PnP: PnP optimization using a hybrid approximate representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(3):768–774, 2019. 2
- [13] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2, 4, 5
- [14] J. A. Hesch and S. I. Roumeliotis. A direct least-squares (DLS) method for PnP. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 383–390. IEEE, 2011. 1, 2, 4, 5, 7
- [15] T. Holzmam, F. Fraundorfer, and H. Bischof. Direct stereo visual odometry based on lines. In *Proceedings of the 11th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2016, pages 1–11, 2016. 1
- [16] L. Kneip, H. Li, and Y. Seo. UPnP: An optimal $O(n)$ solution to the absolute pose problem with universal applicability. In *Computer Vision—ECCV 2014*, pages 127–142. Springer, 2014. 1, 2
- [17] 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 *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2969–2976. IEEE, 2011. 5, 7
- [18] Y. Kuang, Y. Zheng, and K. Astrom. Partial symmetry in polynomial systems and its applications in computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 438–445, 2014. 2
- [19] V. Larsson, K. Astrom, and M. Oskarsson. Efficient solvers for minimal problems by syzygy-based reduction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 820–829, 2017. 4
- [20] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An accurate $O(n)$ solution to the PnP problem. *International Journal of Computer Vision*, 81(2):155–166, 2009. 1, 2, 3, 4, 5, 7
- [21] S. Li, C. Xu, and M. Xie. A robust $O(n)$ solution to the perspective-n-point problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(7):1444–1450, 2012. 2
- [22] C.-P. Lu, G. D. Hager, and E. Mjølness. Fast and globally convergent pose estimation from video images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(6):610–622, 2000. 2
- [23] F. M. Mirzaei, S. Roumeliotis, et al. Globally optimal pose estimation from line correspondences. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5581–5588. IEEE, 2011. 2
- [24] F. Moreno-Noguer, V. Lepetit, and P. Fua. Accurate non-iterative $O(n)$ solution to the PnP problem. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007. 2
- [25] R. Mur-Artal, J. Montiel, and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *Robotics, IEEE Transactions on*, 31(5):1147–1163, 2015. 2, 5
- [26] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017. 5, 6, 8
- [27] G. Nakano. Globally optimal DLS method for PnP problem with cayley parameterization. In *British Machine Vision Conference 2015, Proceedings of. BMVA*, 2015. 2
- [28] D. Oberkampf, D. F. DeMenthon, and L. S. Davis. Iterative pose estimation using coplanar feature points. *Computer Vision and Image Understanding*, 63(3):495–511, 1996. 2
- [29] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer. PL-SLAM: real-time monocular visual SLAM with points and lines. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 4503–4508. IEEE, 2017. 1, 3, 5
- [30] B. Pribyl, P. Zemck, et al. Camera pose estimation from lines using Pluecker coordinates. In *British Machine Vision Conference 2015, Proceedings of. BMVA*, 2015. 2

- [31] L. Quan and Z. Lan. Linear n-point camera pose determination. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(8):774–780, 1999. 2
- [32] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006*, pages 430–443. Springer, 2006. 8
- [33] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: an efficient alternative to SIFT or SURF. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011. 8
- [34] J. L. Schonberger and J.-M. Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113, 2016. 2, 6, 8
- [35] G. Schweighofer and A. Pinz. Globally optimal $O(n)$ solution to the PnP problem for general camera models. In *BMVC*, pages 1–10, 2008. 2
- [36] J. Sola, T. Vidal-Calleja, J. Civera, and J. M. M. Montiel. Impact of landmark parametrization on monocular EKF-SLAM with points and lines. *International journal of computer vision*, 97(3):339–368, 2012. 1
- [37] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. 7, 8
- [38] B. Tekin, S. N. Sinha, and P. Fua. Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 292–301, 2018. 2
- [39] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment: a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999. 5
- [40] S. Urban, J. Leitloff, and S. Hinz. Mlpnp - a real-time maximum likelihood solution to the perspective-n-point problem. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, III-3:131–138, 2016. 1, 2, 5, 7
- [41] A. Vakhitov, J. Funke, and F. Moreno-Noguer. Accurate and linear time pose estimation from points and lines. In *European Conference on Computer Vision*, pages 583–599. Springer, 2016. 1, 2, 3, 5, 6, 7, 8
- [42] 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*, 39(6):1209–1222, 2017. 2
- [43] G. Zhang, J. H. Lee, J. Lim, and I. H. Suh. Building a 3-D line-based map using stereo SLAM. *IEEE Transactions on Robotics*, 31(6):1364–1377, 2015. 1
- [44] L. Zhang and R. Koch. An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency. *Journal of Visual Communication and Image Representation*, 24(7):794–805, 2013. 8
- [45] L. Zhang, C. Xu, K.-M. Lee, and R. Koch. Robust and efficient pose estimation from line correspondences. In *Asian Conference on Computer Vision*, pages 217–230. Springer, 2012. 2
- [46] Y. Zhao and P. A. Vela. Good line cutting: towards accurate pose tracking of line-assisted VOVSLAM. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 516–531, 2018. 5
- [47] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi. Revisiting the PnP problem: a fast, general and optimal solution. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2344–2351. IEEE, 2013. 1, 2, 5, 7
- [48] Y. Zheng, S. Sugimoto, I. Sato, and M. Okutomi. A general and simple method for camera pose and focal length determination. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 430–437. IEEE, 2014. 2
- [49] L. Zhou, Y. Yang, M. Abello, and M. Kaess. A robust and efficient algorithm for the PnL problem using algebraic distance to approximate the reprojection distance. In *AAAI Conference on Artificial Intelligence, AAAI, Honolulu, Hawaii, USA, Jan. 2019*. 2

Uncertainty-Aware Camera Pose Estimation from Points and Lines: Supplementary Materials

Alexander Vakhitov¹ Luis Ferraz Colomina² Antonio Agudo³ Francesc Moreno-Noguer³

¹SLAMCore Ltd., UK

²Kognia Sports Intelligence, Spain

³Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Spain

1. Introduction

In the following supplementary materials, we describes results of the additional experiments, including more data on a real experiment for points and lines described in the main paper, and some additional baseline comparisons in a synthetic setup; also, we give theoretical details on the methods.

2. Details of Methods

In this part, we will describe additional theoretical details behind the proposed methods.

2.1. EP_nPU

In the following section, we outline the uncertainty-aware PCA procedure we use in this method. Recall, that we use an isotropic approximation to the point uncertainty $\Sigma_{\mathbf{x}_i} = \sigma_i^2 \mathbf{I}$.

1. **The covariance-weighted mean point.** As long as the mean point is the point with minimal sum of distances to the points, we modify this definition to use point covariances:

$$\bar{\mathbf{x}} = \operatorname{argmin}_{\mathbf{x}} \sum_{i=1}^{n_{pt}} \sigma_i^{-2} \|\mathbf{x}_i - \mathbf{x}\|^2, \quad (1)$$

$$\bar{\mathbf{x}} = \frac{1}{\sum_{i=1}^{n_{pt}} \sigma_i^{-2}} \sum_{j=1}^{n_{pt}} \sigma_j^{-2} \mathbf{x}_j, \quad (2)$$

and the covariance $\sigma_{\bar{\mathbf{x}}}^2 \mathbf{I}$ of $\bar{\mathbf{x}}$ is

$$\sigma_{\bar{\mathbf{x}}}^2 = \frac{1}{\sum_{i=1}^{n_{pt}} \sigma_i^{-2}}. \quad (3)$$

2. Compute the covariances $\sigma_{\bar{\mathbf{x}_i}}^2 \mathbf{I}$ for the centered points $\bar{\mathbf{x}}_i = \mathbf{x}_i - \bar{\mathbf{x}}$. According to the definition,

$$\sigma_{\bar{\mathbf{x}_i}}^2 = \operatorname{cov} \left(\mathbf{x}_i - \frac{1}{\sum_{i=1}^{n_{pt}} \sigma_i^{-2}} \sum_{j=1}^{n_{pt}} \sigma_j^{-2} \mathbf{x}_j \right), \quad (4)$$

and transforming this, we get

$$\sigma_{\bar{\mathbf{x}_i}}^2 = 1 + \sigma_i^2 - \frac{2}{\sum_{i=1}^{n_{pt}} \sigma_i^{-2}}. \quad (5)$$

3. The j^{th} principal direction is a solution to the following covariance-weighted problem:

$$\mathbf{z}_j = \operatorname{argmax} \sum_{i=1}^{n_{pt}} \sigma_{\bar{\mathbf{x}_i}}^{-2} (\bar{\mathbf{x}}_i^T \mathbf{z}_j)^2, \quad (6)$$

subject to $\mathbf{z}_j^T \mathbf{z}_i = 0$, $i = 1, \dots, j-1$; $\|\mathbf{z}_j\| = 1$, which follows from computing the covariance of the residuals $\operatorname{cov}(\bar{\mathbf{x}}_i^T \mathbf{z}_j) = \sigma_{\bar{\mathbf{x}_i}}^{-2}$, as explained in the paper.

Next, we move to describing the details of the computations for the DLSU method.

2.2. DLSU

In the following text, we give the detailed step-by-step formulas for the DLSU method. After formulating the cost as given in the main paper, we set the gradient of the cost by \mathbf{t} equal $\mathbf{0}$ and express the translation through the rotation parameters

$$\mathbf{t} = -\mathbf{T}^{-1} \operatorname{Avec}(\mathbf{R}(\mathbf{s})), \quad (7)$$

where $\mathbf{T} = \sum_{i=1}^{n_k} \mathbf{T}_k^T \Sigma_{\mathbf{r}_k}^{-1} \mathbf{T}_k$, $\mathbf{A} = \sum_{i=1}^{n_k} \mathbf{T}_k^T \Sigma_{\mathbf{r}_k}^{-1} \mathbf{A}_k$.

The gradient of the cost by the rotation parameters is set to be equal zero as well

$$\mathbf{A}_k^T \Sigma_{\mathbf{r}_k}^{-1} (\mathbf{A}_k - \mathbf{T}_k \mathbf{T}^{-1} \mathbf{A}) \operatorname{vec}(\mathbf{R}(\mathbf{s})) \nabla_{\mathbf{s}} \operatorname{vec}(\mathbf{R}(\mathbf{s})) = \mathbf{0}. \quad (8)$$

As long as the equations are homogeneous with respect to the vectorized rotation $\operatorname{vec}(\mathbf{R}(\mathbf{s}))$, we multiply them by $1 + \|\mathbf{s}\|^2$ following the original DLS approach. We get a 3^{rd} -order polynomial system with three unknown components of \mathbf{s} . We solve it with a generated Groebner solver, and compute \mathbf{t} using (7).

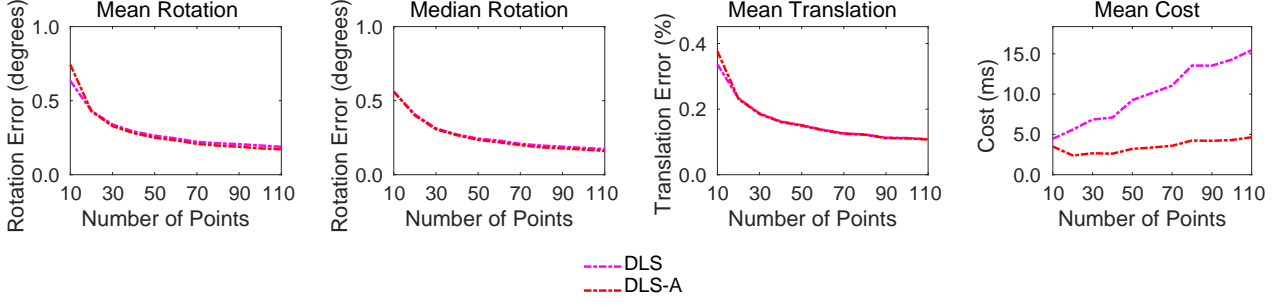


Figure 1. Pose errors and running times in a synthetic experiment with 2D noise, same conditions as in the main paper, Figure 3, top row. We compare the original DLS and the algebraic DLS-A based on algebraic distance. The latter is much faster, and the former gives slight benefits in mean errors for small point counts.

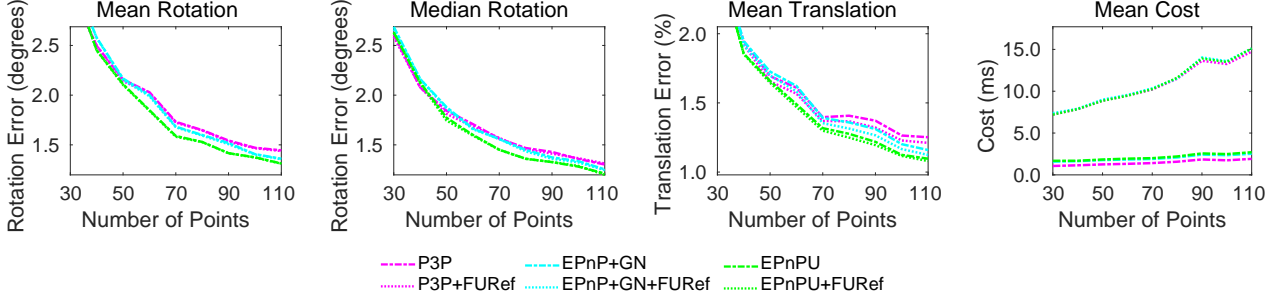


Figure 2. Pose errors and running times in a synthetic experiment with 2D+3D noise, same conditions as in the main paper, Figure 3, central row. We compare the *uncertain* and the *full uncertain* refinement methods (+FURef) for the pipelines based on P3P, EPnP+GN and proposed EPnPU solvers. *Full uncertain* refinement has highly similar accuracy to the *uncertain* refinement.

2.3. Covariance-aware line triangulation

While line representation with the 3D endpoints is clearly non-minimal, because the domain of lines in 3D is 4-dimensional, but two 3D endpoints together give a dimension of 6, the endpoint-based parameterization is still used in practice.

We assume that we are given the camera poses R_i, t_i , $i = 1, \dots, N_l$, and the line segments detected in the corresponding images, defined by the pairs $(\mathbf{x}_i^s, \mathbf{x}_i^e)$ of the endpoints in the image plane. We propose to constrain the 3D endpoints to project to the detected segment endpoints on the first image. For the other images, the projections of the endpoints should belong to the detected lines, not necessarily projecting to the 2D endpoints. This way, the endpoints would encode the spatial location of the detected segment better. We use the following cost for line triangulation, which is a sum of 2D covariance-weighted point-based residuals for the first camera, and line-based residuals for the other cameras: $L_{ln}(\mathbf{P}, \mathbf{Q}) =$

$$\|\mathbf{r}_{\mathbf{x}_1^s}^{pt}(\mathbf{p})\|_{\Sigma_{\mathbf{x}_1^s}}^2 + \|\mathbf{r}_{\mathbf{x}_1^e}^{pt}(\mathbf{q})\|_{\Sigma_{\mathbf{x}_1^e}}^2 + \sum_{i=2}^{N_l} \|\mathbf{r}_i^{ln}(\mathbf{p}, \mathbf{q})\|_{\Sigma_{1,i}}^2, \quad (9)$$

where we denote the point projection residuals $\mathbf{r}_{\mathbf{x}_1^s}^{pt}(\mathbf{p}) = \mathbf{r}_{\mathbf{x}_1^s}^{pt}(\mathbf{p}, R_1, t_1)$ and $\mathbf{r}_{\mathbf{x}_1^e}^{pt}(\mathbf{q}) = \mathbf{r}_{\mathbf{x}_1^e}^{pt}(\mathbf{q}, R_1, t_1)$, as given in (19), main paper, and $\mathbf{r}_i^{ln}(\mathbf{p}, \mathbf{q}) = \mathbf{r}_i^{ln}(\mathbf{p}, \mathbf{q}, R_i, t_i)$, as given

in (2), main paper.

We find \mathbf{p}, \mathbf{q} using Levenberg-Marquardt-based optimization of $L_{ln}(\mathbf{p}, \mathbf{q})$, initializing with the result of the DLT-based line triangulation as explained in [3].

For the error propagation, we follow a general scheme, e.g. [3], Chapter 5, getting

$$\Sigma_{\mathbf{p}, \mathbf{q}} = (\mathbf{J}^T(\mathbf{p}, \mathbf{q})\mathbf{J}(\mathbf{p}, \mathbf{q}))^{-1}, \quad (10)$$

where $\mathbf{J}(\mathbf{p}, \mathbf{q})$ denotes the Jacobian of the inverse covariance-weighted residuals. We obtain $\Sigma_{\mathbf{p}}$ as a left-upper 3×3 block of $\Sigma_{\mathbf{p}, \mathbf{q}}$, and $\Sigma_{\mathbf{q}}$ as the right-lower 3×3 block of the same matrix, which is an approximation indeed, motivated in the main paper by the simplicity of the formulation and the efficiency of computations.

3. Additional experiments

In this section, we give additional experimental results.

3.1. Median errors for points

In Table 1 we present the median errors for the real experiment on KITTI and TUM described in the main paper. While the proposed methods mostly outperform the competitive methods, the gap in terms of median errors is smaller compared to the gap in mean errors. While MLPnP excels in isolation on KITTI, it has much higher runtime because it runs reprojection cost refinement inside, while other solvers do not.

	Points								Points + 2D Uncertainty				Points + Full Uncertainty, Proposed							
	P3P [5]		EPnP [6]		DLS [4]		OPnP [10]		CEPPnP [1]		MLPnP [8]		EPnP*		EPnP*		DLSU*		DLSU	
	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}
KITTI [2], sequences 00-02																				
N	3.6	17.3	1.4	9.8	1.3	7.1	1.3	7.1	3.1	17.9	1.0	4.7	1.2	8.3	1.7	7.9	1.3	7.9	1.5	6.2
S	0.9	4.7	0.9	4.3	0.9	4.2	0.9	4.2	1.0	5.2	0.9	4.2	0.9	4.2	0.9	4.3	0.9	4.2	0.9	4.1
U	0.9	4.7	0.8	4.5	0.8	4.4	0.8	4.4	0.9	5.0	0.8	4.5	0.8	4.5	0.8	4.5	0.8	4.4	0.8	4.4
TUM [7], 'freiburg1' sequences																				
N	13.3	2.7	9.1	1.2	8.9	1.1	8.9	1.1	9.1	1.2	8.7	1.0	8.9	1.0	8.9	1.1	8.8	1.0	8.7	1.0
S	8.6	1.0	8.6	0.9	8.6	0.9	8.6	0.9	8.6	0.9	8.6	0.9	8.6	0.9	8.5	0.9	8.5	0.9	8.5	0.9
U	8.6	0.9	8.5	0.9	8.6	0.9	8.6	0.9	8.6	0.9	8.6	0.9	8.5	0.9	8.5	0.9	8.5	0.9	8.5	0.9

Table 1. Motion estimation from 2D-3D point correspondences on KITTI [2] TUM [7] in terms of median absolute rotation e_{rot} (in $0.1 \times \text{deg.}$) and translation e_{trans} (in cm.) errors. We compare proposed full uncertainty-aware methods against point-based PnP and 2D uncertainty-aware methods in isolation (N), with standard (S) and proposed uncertain (U) refinement. Methods with '*' receive a pose from RANSAC, best for the dataset is in bold italic, best for each protocol (N,S or U) is in bold. The new methods outperform the baselines in most metrics.

		Points+Lines				Points+Lines+Uncertainty			
		EPnPL [9]		OPnPL [9]		DLSLU*		EPnPLU*	
		e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}	e_{rot}	e_{trans}
KITTI [2], sequences 00-02									
mean	N	2.65	40.26	9.54	727.26	5.88	17.34	2.73	20.25
	S	1.74	12.77	7.93	344.78	5.19	13.86	1.73	9.71
med.	N	1.52	15.75	1.46	7.98	1.49	6.41	1.71	8.37
	S	0.89	4.87	0.86	4.20	0.84	4.04	0.85	4.21
TUM [7], sequences 'freiburg1'									
mean	N	11.73	1.85	10.74	1.52	9.84	1.24	12.27	1.59
	S	11.13	1.38	10.43	1.29	9.77	1.18	11.88	1.39
med.	N	9.06	1.26	8.93	1.12	8.73	0.93	8.87	1.06
	S	8.64	0.92	8.58	0.92	8.58	0.92	8.58	0.91

Table 2. Motion estimation from 2D-3D point and line correspondences on KITTI [2] sequences 00-02 and TUM [7], 'freiburg1' sequences. We report the rotation errors in $0.1 \times \text{degrees}$ and translation errors in cm, for the solvers in isolation (N) and after *standard* refinement (S). The proposed uncertainty-aware solvers outperform the uncertainty-free baselines OPnPL and EPnPL in most metrics on KITTI and in mean metrics on TUM. Median rotation errors on TUM are similar, but the proposed solvers benefit from lower median translation errors.

3.2. Full results on lines

Due to limited space in the main paper, we present here the results for the points + lines pipeline on TUM and KITTI datasets, same sequences as for the points in the main paper, see Table 2. The proposed solvers outperform the baselines in translation errors on both datasets. On TUM, the median rotation errors are similar for all tested methods, while on KITTI the proposed solvers have better rotation accuracy.

3.3. DLS Modifications

We compare the original object space error-based DLS solver [4] and our modification, DLS-A, which uses the algebraic error. We get DLS-A from the DLSU solver by providing it with unit matrices as residual covariances. See Figure 1 for the results. The proposed DLS-A is 2-3 times faster depending on a point number, but also is slightly infe-

rior for the low number of points with respect to the original DLS method.

3.4. Refinement modifications

In this section, we compare the *full uncertain* and the proposed *uncertain* refinement methods, as described in Section 3.5 of the main paper. *Full uncertain* refinement is implemented using finite-difference approximation of the Jacobian and based on MATLAB lsqnonlin function, implementing a Levenberg-Marquardt method, while *uncertain* refinement is implemented as Gauss-Newton iterations, with additional iterative re-computation of the residual covariances. The experiment is run following the setting of the main paper (2D noise + 3D noise, central row of the Figure 3, main paper). We compare a pipelines with P3P, EPnP+GN and proposed EPnP* solvers. The results in Figure 2 suggest, that there are no major differences in

accuracy of the methods.

References

- [1] L. Ferraz, X. Binefa, and F. Moreno-Noguer. Leveraging feature uncertainty in the PnP problem. In *Proceedings of the BMVC 2014 British Machine Vision Conference*, pages 1–13, 2014. 3
- [2] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 3
- [3] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2
- [4] J. A. Hesch and S. I. Roumeliotis. A direct least-squares (DLS) method for PnP. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 383–390. IEEE, 2011. 3
- [5] 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 *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2969–2976. IEEE, 2011. 3
- [6] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An accurate O(n) solution to the PnP problem. *International Journal of Computer Vision*, 81(2):155–166, 2009. 3
- [7] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. 3
- [8] S. Urban, J. Leitloff, and S. Hinz. Mlpnp - a real-time maximum likelihood solution to the perspective-n-point problem. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, III-3:131–138, 2016. 3
- [9] A. Vakhitov, J. Funke, and F. Moreno-Noguer. Accurate and linear time pose estimation from points and lines. In *European Conference on Computer Vision*, pages 583–599. Springer, 2016. 3
- [10] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi. Revisiting the PnP problem: a fast, general and optimal solution. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2344–2351. IEEE, 2013. 3


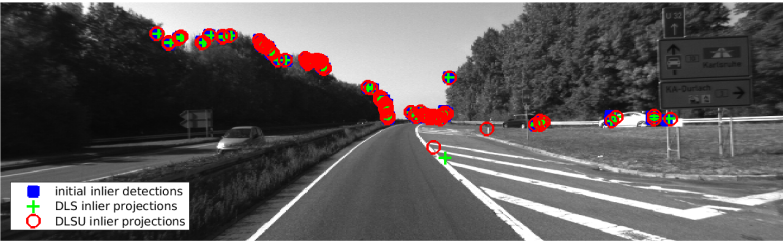
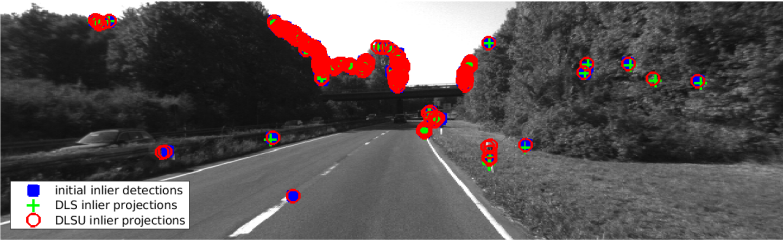
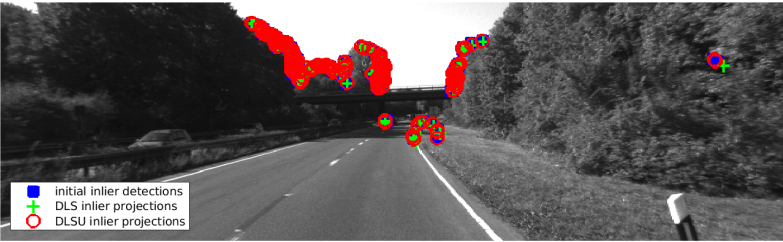
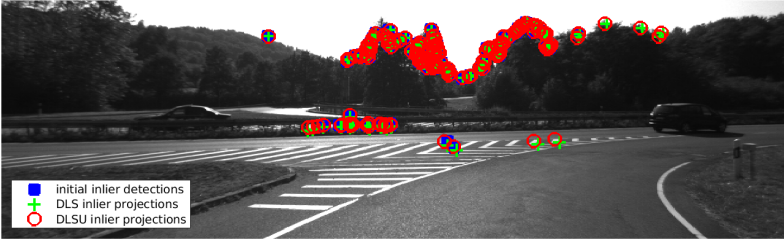
Image	$\Delta e_{\text{trans}}, \text{cm.}$
	28
	78
	14.1
	4.0
	-2.4

Table 3. We compare the DLS and DLSU filtered inlier sets reprojected onto the images, and also plot the initially estimated inlier detections. In the right column, see the improvement of absolute error by DLSU as compared to DLS, after the standard refinement. The inlier projections of DLSU are more aligned with the detections; DLSU selects closer features more often.