1-Point RANSAC-Based Method for Ground Object Pose Estimation

Jeong-Kyun Lee, Young-Ki Baik, Hankyu Cho, Kang Kim, and Duck Hoon Kim

Qualcomm Korea YH, Seoul, South Korea
{ljeongky,ybaik,hankcho,kangk,duckhoon}@qti.qualcomm.com

Abstract. Solving Perspective-n-Point (PnP) problems is a traditional way of estimating object poses. Given outlier-contaminated data, a pose of an object is calculated with PnP algorithms of $n = \{3, 4\}$ in the RANSAC-based scheme. However, the computational complexity considerably increases along with n and the high complexity imposes a severe strain on devices which should estimate multiple object poses in real time. In this paper, we propose an efficient method based on 1-point RANSAC for estimating a pose of an object on the ground. In the proposed method, a pose is calculated with 1-DoF parameterization by using a ground object assumption and a 2D object bounding box as an additional observation, thereby achieving the fastest performance among the RANSAC-based methods. In addition, since the method suffers from the errors of the additional information, we propose a hierarchical robust estimation method for polishing a rough pose estimate and discovering more inliers in a coarse-to-fine manner. The experiments in synthetic and real-world datasets demonstrate the superiority of the proposed method.

Keywords: RANSAC, perspective-n-point, object pose estimation

1 Introduction

Perspective-n-point (PnP) is a classical computer vision problem of finding a 6-DoF pose of a camera or an object given n 3D points and their 2D projection points in a calibrated camera [10]. It has been widely used in finding ego-motion of a camera and reconstructing a scene in 3D [11,35] as well as estimating a pose and a shape of a known or arbitrary object [26,36,45].

Handling outliers is a crucial ability in practical applications of pose estimation because mislocalization errors or mismatches of point correspondences inevitably arise. However, a majority of studies on the PnP problem focus on producing high accuracy in noisy data. Instead, they depend on the RANSAC-based scheme [10] to handle data contaminated with outliers. The common scheme that PnP algorithms with $n = \{3, 4\}$ (*i.e.*, P3P [12,25,37] or P4P [1,18]) are incorporated into RANSAC distinguishes inlier 2D-3D point correspondences and produces a rough pose estimate which is subsequently polished using the inlier set. Although the stopping criterion of RANSAC iteration is adaptively determined while RANSAC processing, the average number of trials in this scheme

exponentially increases along with n in the data with a high outlier ratio. The high complexity may hinder the system running on real-world applications where fast and accurate execution is crucial, such as localizing multiple objects at once.

Ferraz et al. proposed REPPnP [9] that is similar to the popular robust estimation technique [21] with iterative re-weighting mechanism. It estimates a pose by calculating the null space of a linear system for control points in the same way of EPnP [28]. Afterward, in an iterative fashion, it assigns confidence values to the correspondences by computing algebraic errors and computes the null space of the weighted linear system. Since the repetition is empirically finished within several times, REPPnP achieves fast and accurate performance. However, as with a general M-estimator [21], its highly possible breakpoint can attain 50%. Thus, this method does not ensure working on the data with the high outlier ratio.

In this paper, we propose an efficient method for calculating the 6D pose of an object with a P1P algorithm in the RANSAC-based scheme. To simplify the 6D pose estimation problem, we assume an object is on the ground and the relationship between the ground and the camera is pre-calibrated. Then, given a 2D object bounding box and n 2D-3D point correspondences, the PnP problem is reformulated by 1-DoF parameterization (*i.e.*, a yaw angle or a depth of an object), thereby producing a pose estimate from one point correspondence. The synthetic experiments demonstrate our method is the fastest one among RANSAC-based methods. However, the proposed method suffers from erroneous 2D bounding box or ground geometry. Therefore, we also propose a hierarchical robust estimation method for improving the performance in the practical situation. In the refinement stage, it not only polishes the rough pose estimate but also secures more inliers in a coarse-to-fine manner. It consequently achieves the more robust and accurate performance in the erroneous cases and a more complex problem (*e.g.*, joint pose and shape estimation [26,36,45]).

2 Related works

As mentioned above, the PnP problem in the data containing outliers is handled with REPPnP [9] or usually the RANSAC scheme [10] where P3P [12,25,37] or P4P [1,18] are incorporated for providing minimal solutions of the PnP problem. Except for them, the existing PnP algorithms have aimed at improving the performance in noisy data. The PnP algorithms are traditionally categorized to two types of methodologies: iterative and non-iterative methods.

Iterative methods [6,13,19,33,34] find a globally optimal pose estimate by solving a nonlinear least squares problem minimizing algebraic or geometric errors with iterative or optimization techniques. Among them, the method proposed by Lu *et al.* [34] accomplishes outstanding accuracy. It reformulates an objective function as minimizing object-space collinearity errors instead of geometric errors measured on the image plane. The least squares problem is solved using the way of Horn *et al.* [20] iteratively. Garro *et al.* [13] proposed a Procustes PnP (PPnP) method reformulating the PnP problem as an anisotropic orthogonal Procustes (OP) problem [7]. They iteratively computed a solution of

the OP problem by minimizing the geometric errors in the object space until convergence, which achieved a proper trade-off between speed and accuracy.

On the other hand, non-iterative methods [17,24,28,29,47,48] quickly obtain a pose estimates by calculating solutions of a closed form. Lepetit *et al.* [28] proposed an efficient and accurate method (EPnP) for solving the PnP problem with computational complexity of $O(n)$. They defined the PnP problem as finding virtual control points, which was quickly calculated by null space estimation of a linear system. They refined the solution with the Gauss-Newton method so that its accuracy amounted to that of Lu *et al.* [34] with less computational time. Hesch *et al.* [17] formulated the PnP problem as finding a Cayley-Gibbs-Rodrigues (CGR) parameter by solving a system of several third-order polynomials. Several methods [24,47,48] employ quaternion parameterization and solve a polynomial system minimizing algebraic [47,48] or object-space [24] errors with the Grbner basis technique [27]. In particular, the method of Kneip *et al.* [24] results in minimal solutions of the polynomial system and is generalized for working on both central and non-central cameras. Li *et al.* [29] proposed a PnP approach robust to special cases such as planar and quasi singular cases. They defined the PnP problem by finding a rotational axis, an angle, and a translation vector, and solved the linear systems formulated from projection equations and a series of 3-point constraints [38]. Recently, some methods [8,41] utilize the uncertainty of observations on the image space. They estimate control points [8] or directly a pose parameter [41] by performing maximum likelihood estimation for Sampson errors reflecting covariance information of observations.

3 Proposed Method

In this paper, we handle how to reduce the computational complexity of a RANSAC-based scheme for the PnP problem in outlier-contaminated data. Many real-world applications [26,36,45] aim at estimating a 6D pose of an object on the ground, given 2D object bounding boxes from object detectors [15,31]. The additional information is not only useful to reduce the number of required points for solving the PnP problem but also reasonable due to being acquired without an extra computational loss in some applications. Hence, we propose a perspective-1-point (P1P) method for an object on the ground, which significantly raises the speed of the RANSAC-based scheme. In the following sections, we first introduce a general framework for n-point RANSAC-based pose estimation, then propose the P1P method for roughly estimating an object pose using one point sample, and finally suggest a novel robust estimation method for polishing the rough pose estimate.

3.1 General framework of n-point RANSAC-based pose estimation

We employ a general framework for n-point RANSAC-based pose estimation.[1] Given a set of 2D-3D keypoint correspondences, $\mathcal{C}$, we sample n keypoint correspondences, *i.e.*, $\{(\mathbf{x}_1, \mathbf{X}_1), ..., (\mathbf{x}_n, \mathbf{X}_n) | (\mathbf{x}_i, \mathbf{X}_i) \in \mathcal{C}\}$, and compute a pose candidate $\mathbf{T}_{cand}$ using a PnP algorithm. Then, the reprojection errors are computed

[1] Please find a flowchart of the RANSAC-based scheme in the supplementary material.

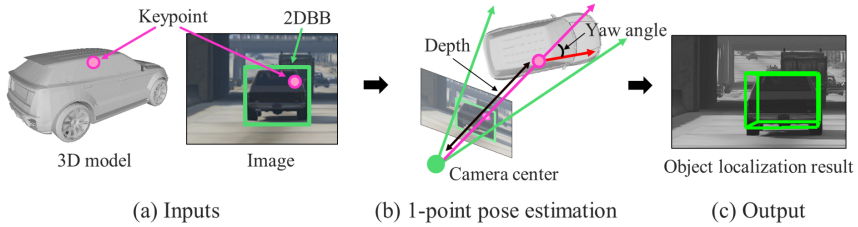


Fig. 1: A flow chart of the proposed method for pose estimation

for all the keypoint correspondences and the keypoints whose reprojection errors are within a threshold t_{in} are regarded as inliers. This process is repeated N times and we select the pose with the maximum number of inliers as the best pose estimate. The maximum number of iteration, N, can be reduced by the adaptive RANSAC technique [10] which adjusts N depending on the number of inliers while the RANSAC processing. Finally, the pose estimate is polished by minimizing the reprojection errors of the inlier keypoints or using the existing PnP algorithms.

3.2 Perspective-1-point solution

To reduce the number of points required for computing an object pose hypothesis, we need to use some prior knowledge. First, we assume that the tilt of a camera to the ground where an object of interest is placed is given as a pitch angle θ_p.[2] From the pitch angle, a rotation transformation $\mathbf{R}_{cg} \in SO(3)$ from the ground to the camera is defined as $\mathbf{R}_{cg} = e^{\omega_p}$ where $\omega_p = [-\theta_p, 0, 0]^\top \in so(3)$ and $e : so(3) \mapsto SO(3)$ is an exponential map. For example, if an object is fronto-parallel to the image plane, $\mathbf{R}_{cg} = \mathbf{I}_3$ where $\mathbf{I}_3$ is a 3×3 identity matrix. Second, besides the 3D model of the object and its 2D-3D keypoint correspondence, which are provided as input by default in the PnP problem, we assume that the 2D bounding box (2DBB) of the object of interest on an image is given as an input. Using the additional prior information, the problem is redefined as aligning the projection of the 3D bounding box (3DBB) into the back-projected rays of both the side ends of its 2DBB in a bird-eye view (BEV), thereby computing the yaw angle of the object and the depth of the keypoint as shown in Fig. 1b. Here, we formulate an equation that computes the pose of the object of interest by 1-DoF parameterization. A unique pose hypothesis per keypoint is obtained by solving the equation. Consequently, through the RANSAC process, we find the best pose parameter with the maximum number of inliers among those hypotheses and then optimize the pose.

There are four cases of the 1-point object pose estimation depending on the yaw angle of the object as shown in Fig. 2. In Fig. 2a, the red arrow and the blue rectangle denote the forward direction of the object and the 3DBB in the BEV, respectively. We denote the four corners of the rectangle by numbers as

[2] It is pre-calibrated or calculated in an online manner with [22,46].

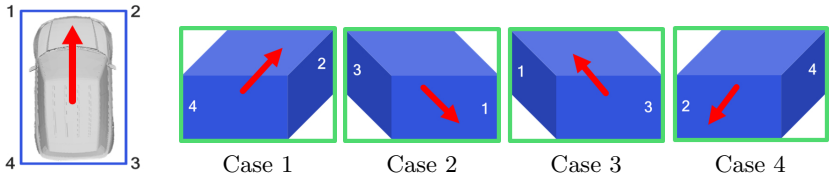


Fig. 2: Case study on the 1-point object pose estimation: (a) 3DBB of an object in the BEV and (b) four cases of the pose estimation depending on the yaw angle

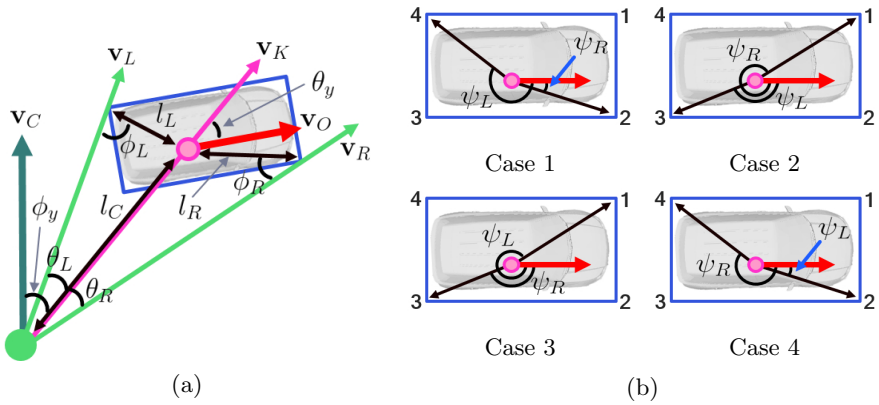


Fig. 3: Parameter definition in the BEV: (a) parameters in the camera coordinate system and (b) parameters for each case in the 3D model coordinate system

shown in Fig. 3b. In Fig. 2b, the green rectangle and the blue hexahedron denote the 2DBB of the object and the projection of the 3DBB to an image. Then, as shown in Fig. 2b, the left and right edges of the 2DBB are adjacent to two of the corners of the 3DBB depending on the yaw angle of the object, *e.g.*, in Case 1, the left and right edges of the 2DBB are paired with Corner 4 and 2 of the 3DBB, respectively. In this way, we find out the four cases[3] for the 1-point pose estimation and will describe how to compute an object pose using a point correspondence at each case.

Figure 3 shows the definition of the parameters required to formulate an equation in the BEV. In Fig. 3a, $\mathbf{v}_K$ (magenta) is a directional vector from the camera center to the keypoint, $\mathbf{v}_L$ and $\mathbf{v}_R$ (green) are the back-projected rays of

[3] In fact, the number of the cases are exactly more than 4 due to the perspective effect (*e.g.*, Corner 3 and 4 of the 3DBB is possible to be adjacent to the 2DBB) but those cases can approximate to the 4 cases of Fig. 2b if an object of interest is far enough from the camera.

the left and right edges of the 2DBB[4], $\mathbf{v}_O$ (red) is the forward direction of the object, $\mathbf{v}_C$ (dark green) is the forward direction of the camera, ϕ_y, θ_L, and θ_R are the angles between $\mathbf{v}_C$ and $\mathbf{v}_K$, $\mathbf{v}_K$ and $\mathbf{v}_L$, and $\mathbf{v}_K$ and $\mathbf{v}_R$, respectively, l_C is the length between the camera center point and the keypoint, l_L and l_R are the lengths between the keypoint and the corners of the 3DBB, and ϕ_L and ϕ_R are the angles between $\mathbf{v}_L$ and l_L, and $\mathbf{v}_R$ and l_R, respectively. In Fig. 3b, ψ_L and ψ_R are the angles between the forward direction of the object and the corners of the 3DBB. Then, our purpose is to compute a local yaw angle of the object, *i.e.*, an angle between $\mathbf{v}_K$ and $\mathbf{v}_O$, θ_y, and a depth from the camera center to the keypoint, l_C.

We first describe the 1-point pose estimation method for Case 1. We can derive an equation to compute θ_y from the sine rule as follows.

$$l_C = \frac{l_R \sin\phi_R}{\sin\theta_R} = \frac{l_L \sin\phi_L}{\sin\theta_L}, \tag{1}$$

where $\phi_R = \theta_y + \psi_R - \theta_R$ and $\phi_L = -\theta_y - \psi_L - \theta_L$. From Eq. 1, the yaw angle θ_y is derived as

$$\theta_y = \arctan\left(\frac{-\frac{l_R \sin(\psi_R - \theta_R)}{\sin\theta_R} - \frac{l_L \sin(\psi_L + \theta_L)}{\sin\theta_L}}{\frac{l_R \cos(\psi_R - \theta_R)}{\sin\theta_R} + \frac{l_L \cos(\psi_L + \theta_L)}{\sin\theta_L}}\right), \tag{2}$$

and the depth of the keypoint, l_C, is calculated using Eq. (1).[5]

Once θ_y and l_C are calculated, an object pose $\mathbf{T}_{co} \in SE(3)$, which is a transformation matrix from the object coordinates to the camera coordinates, is computed by

$$\mathbf{T}_{co} = \begin{bmatrix} \mathbf{R}_{cg} & \mathbf{0}_3 \\ \mathbf{0}_3^\top & 1 \end{bmatrix} \begin{bmatrix} e^{\omega_y} & l_C \mathbf{d}_\mathbf{x} \\ \mathbf{0}_3^\top & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_3 & -\mathbf{X} \\ \mathbf{0}_3^\top & 1 \end{bmatrix}, \tag{3}$$

where $\omega_y = [0, \phi_y + \theta_y, 0]^\top$, $\mathbf{X}$ is a 3D location of the selected keypoint, and $\mathbf{d}_\mathbf{x}$ is a normalized directional vector by back-projecting the keypoint $\mathbf{x}$ to a 3D ray, and computed by

$$\mathbf{d}_\mathbf{x} = \frac{\mathbf{R}_{cg}^\top \mathbf{K}^{-1} \hat{\mathbf{x}}}{\sqrt{\hat{\mathbf{x}}^\top \mathbf{K}^{-\top} \mathbf{R}_{cg} \mathbf{S} \mathbf{R}_{cg}^\top \mathbf{K}^{-1} \hat{\mathbf{x}}}}, \text{ where } \mathbf{S} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4}$$

Here, $\mathbf{K}$ is a camera intrinsic matrix and $\hat{\mathbf{x}}$ is the homogeneous coordinate of $\mathbf{x}$.

The poses $\mathbf{T}_{co}$ in the other cases are computed in a similar way to Case 1. The only different thing is the definitions of ψ_L and ψ_R as shown in Fig. 3b. Due to all the cases presented in Fig. 2 and the sign inside the arctangent function of Eq. B.10, we have four solutions. However, as shown in Table B.1, if we consider the ranges of θ_y, ψ_L, and ψ_R for each case, we can filter out unreliable solutions and obtain a unique solution from only one keypoint.

[4] We assume that the side edges of the 2DBB are back-projected to the planes perpendicular to the ground and thus, the projections of the planes into the ground become the rays passing through the camera center and the corners of the 3DBB.

[5] Please see the supplementary material for the details of the derivation.

Table 1: Range of angle parameters for the four cases

Parameter	Case 1	Case 2	Case 3	Case 4
θ_y	$[0, \frac{\pi}{2}]$	$[\frac{\pi}{2}, \pi]$	$[-\frac{\pi}{2}, 0]$	$[-\pi, -\frac{\pi}{2}]$
ψ_L	$[-\pi, -\frac{\pi}{2}]$	$[\frac{\pi}{2}, \pi]$	$[-\frac{\pi}{2}, 0]$	$[0, \frac{\pi}{2}]$
ψ_R	$[0, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, 0]$	$[\frac{\pi}{2}, \pi]$	$[-\pi, -\frac{\pi}{2}]$

3.3 Hierarchical robust pose estimation of objects with deformable shapes

Many practical applications [4,26,36,45] have dealt with the pose estimation of objects with deformable shapes. Unlike solving the PnP problem using a known 3D model, estimating object pose and shape simultaneously in a single image is an ill-posed problem. Thus, the existing methods [4,26,36,45] have exploited active shape models (ASM) [5]. Given a set of class-specified 3D object models as prior information, the i^{th} 3D keypoint location $\mathbf{X}_i$ in the ASM is defined by summing a mean location $\bar{\mathbf{X}}_i$ and the combinations of M basis vectors $\mathbf{B}_i^j$.

$$\mathbf{X}_i = \bar{\mathbf{X}}_i + \sum_j^M \lambda_j \mathbf{B}_i^j, \tag{5}$$

where $\lambda = \{\lambda_1, ..., \lambda_M\}$ is a set of variables depending on shape variation.

Then, the object pose and shape are jointly estimated by minimizing residuals r_i (*i.e*, reprojection errors) as follows.

$$\underset{\mathbf{T},\lambda}{\operatorname{argmin}} \sum_{i=0}^{|\mathcal{C}|} \rho(r_i(\mathbf{T}, \lambda)), \text{ where } r_i(\mathbf{T}, \lambda) = \|f(\mathbf{R}(\bar{\mathbf{X}}_i + \sum_{j=0}^{M} \lambda_j \mathbf{B}_i^j) + \mathbf{t}) - \mathbf{x}_i\|_2 \tag{6}$$

Here, $|\mathcal{C}|$ is the cardinality of the correspondence set $\mathcal{C}$, the object pose $\mathbf{T}$ is represented by decomposition into a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$, f is a projection function from a 3D coordinate to a image coordinate, and ρ represents an M-estimator [21] for robust estimation in the presence of outliers.

In our experiments, we use the Tukey biweight function defined by

$$\rho(r_i) = \begin{cases} \frac{c^2}{6}\left\{1 - \left[1 - \left(\frac{r_i}{c}\right)^2\right]^3\right\}, & \text{if } r_i \leq c \text{ and} \\ \frac{c^2}{6}, & \text{if } r_i > c, \end{cases} \tag{7}$$

where $c = 4.685s$, a scale factor s is calculated by $s = MAD(r_i)/0.6745$, and $MAD(r_i)$ is a median absolute deviation of residuals r_i. We initialize the coefficients λ as zeros and the object pose by the RANSAC process where the unknown 3D model is substituted with the mean shape $\bar{\mathbf{X}}$. Finally, optimal pose and shape minimizing Eq. (6) are estimated using the iterative reweighted least squares (IRLS) method.

Algorithm 1 Hierarchical robust pose and shape estimation

Require: $\mathcal{C}, \mathbf{T}_{init}, \tau_1, \tau_2, \tau_3$
Ensure: $\mathbf{T}, \lambda$
1: $\mathbf{T} \leftarrow \mathbf{T}_{init}, \lambda \leftarrow \mathbf{0}$
2: $\mathbf{T} \leftarrow \text{argmin}_{\mathbf{T}} \sum_{i=0}^{|\mathcal{C}|} \rho(r_i(\mathbf{T}|\lambda = 0)|c = 4.685\gamma(s|\tau_2, \tau_3))$ # First stage
3: $\mathbf{T}, \lambda \leftarrow \text{argmin}_{\mathbf{T},\lambda} \sum_{i=0}^{|\mathcal{C}|} \rho(r_i(\mathbf{T}, \lambda)|c = 4.685\gamma(s|\tau_1, \tau_2))$ # Second stage
4: $\mathcal{C}_{inlier} \leftarrow \{(\mathbf{x}_i, \mathbf{X}_i)|r_i(\mathbf{T}, \lambda) < \tau_1\}$
5: $\mathbf{T}, \lambda \leftarrow \text{argmin}_{\mathbf{T},\lambda} \sum_{i=0}^{|\mathcal{C}_{inlier}|} \|r_i(\mathbf{T}, \lambda)\|^2$ # Third stage

However, the common approach [32] for robust pose and shape estimation often gets stuck in local minima by the following reasons. First, the scale factor computed by MAD attains a breakdown point of 50% in a statistical point of view but does not produce a geometrically meaningful threshold in the data contaminated with outliers. Second, the M-estimator is sensitive to initial parameters. In particular, our P1P solution produces a more noisy pose estimate when the camera pitch angle varies or a 2D bounding box from an object detector is erroneous.

Inspired by MM-estimator [44] and annealing M-estimator [30] that reduce the sensitivity to the scale estimate and avoid to get stuck in a local minimum via an annealing process, we propose a hierarchical robust estimation method for the object pose and shape estimation. We repeat M-estimation while decreasing the scale factor. Here, we use geometrically meaningful and user-defined scale factors because the threshold empirically set by an user may be rather meaningful than the one calculated from statistical analysis in the case that camera properties such as intrinsic parameters remain constant and the input data contain outliers.

The details of the proposed method are described in Alg. 1. Given the 2D-3D correspondence set $\mathcal{C}$, an initial pose $\mathbf{T}_{init}$ by the RANSAC process, and user-defined thresholds τ_1, τ_2, and τ_3 ($\tau_1 < \tau_2 < \tau_3$), the object pose $\mathbf{T}$ and shape λ are estimated through three stage optimization. At the first stage, the roughly initialized pose is refined with the scale estimates loosely bounded to a range of $[\tau_2, \tau_3]$ by the function γ, which is a clamp function for limiting the range of an input value x to $[\alpha, \beta]$ and defined as $\gamma(x|\alpha, \beta) = \max(\alpha, \min(x, \beta))$. At the second stage, the pose $\mathbf{T}$ and shape parameter λ are jointly optimized with the scale estimates tightly bounded to a range of $[\tau_1, \tau_2]$. Finally, the pose and shape are polished using only the inlier correspondences set $\mathcal{C}_{inlier}$. The first and second stages are computed using the IRLS method and the third stage is done using the Gauss-Newton method. In a case of the PnP problem using a known 3D model, the hierarchical robust estimation method can be also used to estimate only an object pose by excluding λ in Alg. 1.

4 Experiments

4.1 Synthetic experiments

Dataset. It is assumed that a camera is calibrated with a focal length of 800 pixels and images are captured with resolution of 640×480. Object points are

randomly sampled with a uniform distribution in a cube region of $[-2,2] \times [-2,2] \times [-2,2]$. A center location and a yaw angle of an object are randomly sampled from a region of $[-4,4] \times [-1,1] \times [20,40]$ and a range of $[-\pi, \pi]$, respectively, and used to calculate the ground truth rotation matrix $\mathbf{R}_{gt}$ and translation vector $\mathbf{t}_{gt}$. We extract 300 object points and generate 2D image coordinates by projecting them onto the image plane. Then, Gaussian noise of $\sigma = 2$ pixels is added to the image coordinates. The pitch angle of a camera and a ratio of outliers are set to 0° and 50%, respectively. We design 4 types of experiments. (1) *E1*: the outlier ratio is changed from 10% to 90%. (2) *E2*: the number of object points is changed from 50 to 1000. (3) *E3*: pitch angle errors from -5° to 5° are added to the camera pose. (4) *E4*: 2D bounding box errors from -5 pixels to 5 pixels are added to 2D bounding box observations.

Evaluation metric. We use mean rotation and translation errors which have been widely used in the literature [9,47,48]. Given a rotation estimate $\mathbf{R}$ and a translation estimate $\mathbf{t}$, the translation error is measured by $e_t(\%) = \|\mathbf{t}_{gt} - \mathbf{t}\|/\|\mathbf{t}\| \times 100$ and the rotation error by $e_r(^\circ) = \max_{i=1}^{3}\{\arccos(\mathbf{r}_{gt,i} \cdot \mathbf{r}_i) \times 180/\pi\}$ where $\mathbf{r}_{gt,i}$ and $\mathbf{r}_i$ are the i^{th} column of the rotation matrices $\mathbf{R}_{gt}$ and $\mathbf{R}$. For each experiment, we calculated mean errors of 1000 independent simulations. The methods were tested on a 3.4GHz single core using MATLAB.

Variation of the outlier ratio and the number of points. Our 1-point RANSAC-based method (RNSC-P1P) is compared with RANSAC+P3P [10,25] (RNSC-P3P) and REPPnP [9]. In addition, their results are polished by the Gauss-Newton (GN) method or several PnP approaches: EPnP [28], RPnP [29], ASPnP [48], OPnP [47], and EPPnP [9].[6] In all the experiments, the inlier threshold of RANSAC is set to $t_{in} = 4$ pixels and the algebraic error threshold of REPPnP is set to $\delta_{\max} = kt_{in}/f$ where a constant $k = 1.4$ and the focal length $f = 800$ as recommended in [9].

Figures 4a and 4b show the mean rotation and translation errors in *E1*. It demonstrates that RNSC-P1P (or RNSC-P1P+PnP strategies) is superior to RNSC-P3P (or RNSC-P3P+PnP strategies). Since *E1* has no pitch angle error and P1P uses the 1-DoF rotation parameterization constrained by prior pitch information, the pose estimates of P1P are more accurate than those of P3P in this experiment. Hence, the number of inliers by P1P is higher than that by P3P as shown in Fig. 4c. Figures 4d and 4e represent the results in *E2*, which show the same tendency as the results of *E1*. REPPnP achieved better accuracy than our method because it took much more inliers by using the loose threshold. However, REPPnP frequently produced invalid object pose estimates in 1) the case that outlier ratio was more than 50% and 2) the case that a small number of point correspondences were used (*e.g.*, the number of points is less than 200 as shown in Figs. 4d and 4e), because of the effect of the high outlier ratio of 50%. On the other hand, RANSAC-based methods consistently provide valid pose estimates despite the high outlier ratio of 90%.

[6] The results of EPnP and OPnP are refined by GN and the Newton (N) method, respectively.

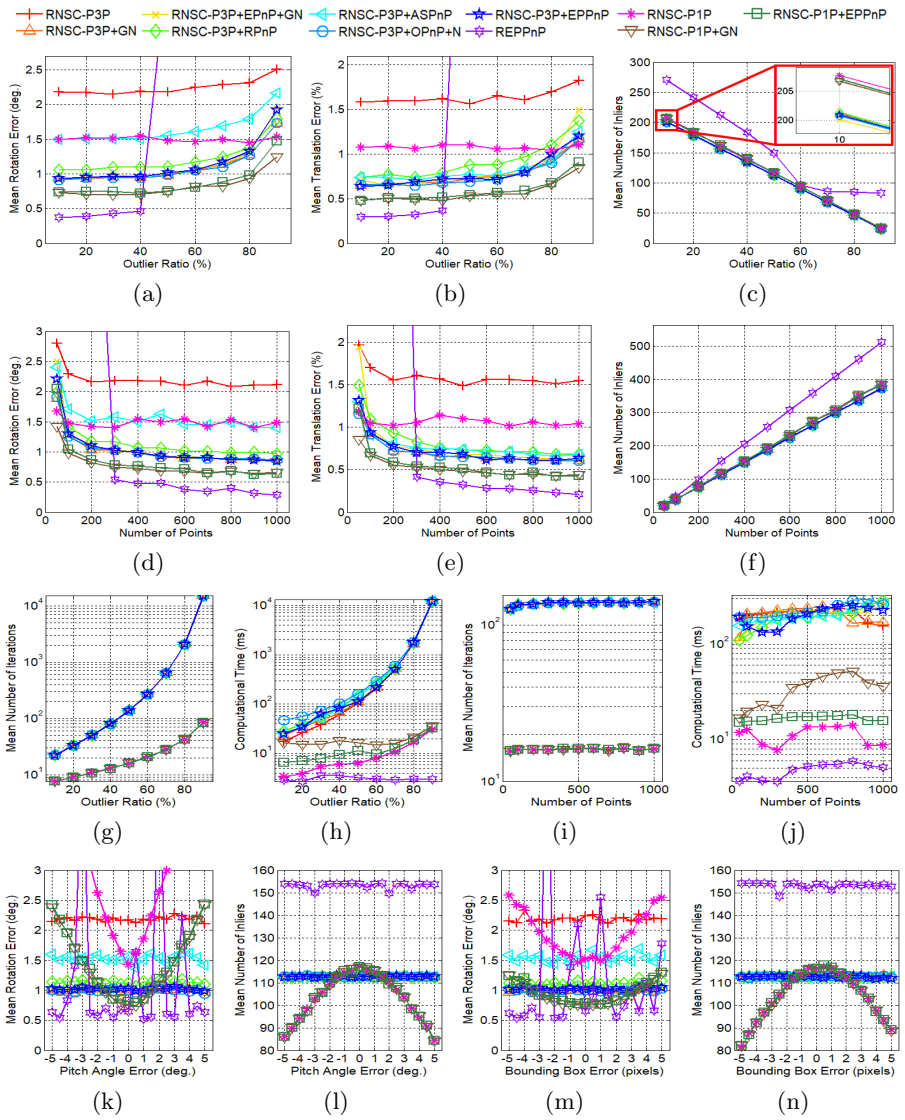
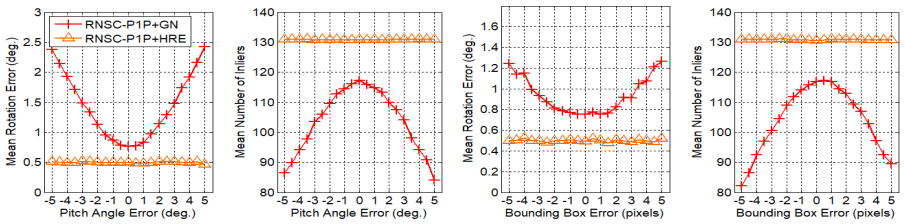


Fig. 4: Results of synthetic experiments *E1-E4*. (a), (b), (c), (g), and (h) ((d), (e), (f), (i), and (j)) represent the mean rotation errors, the mean translation errors, the mean number of inliers, the mean number of iteration of RANSAC, and the computational time on *E1* (*E2*), respectively. (k) and (l) ((m) and (n)) represent the mean rotation errors and the number of inliers on *E3* (*E4*), respectively.



(a) (b) (c) (d)

Fig. 5: Results of our hierarchical robust estimation method. (a) and (b) ((c) and (d)) represent the mean rotation errors and the number of inliers on *E3* (*E4*), respectively.

As shown in Figs. 4g and 4i, our RNSC-P1P-based methods take much less iterations than RNSC-P3P-based methods. Consequently, Figs. 4h and 4j show that the computational time of RNSC-P1P-based methods is considerably faster than that of RNSC-P3P-based methods but slower than that of REPPnP.

Pitch angle and bounding box errors Since the proposed method assumed a fixed pitch angle and used a 2D bounding box as additional information, we performed *E3* and *E4* to analyze the effect of the pitch angle and bounding box errors on our method. Figures 4k-4n show that both the pose estimation accuracy and the number of inliers decrease as the pitch angle and bounding box errors increase. The RNSC-P1P-based methods produce better performance than the RNSC-P3P-based methods in the pitch error within 1.5 degrees or the bounding box error within 1 pixel, but otherwise their performance is degraded.

Hierarchical robust estimation As mentioned in Section 3.3, the hierarchical robust estimation (HRE) method can be used in pose estimation using known 3D models. We compared RNSC-P1P+HRE with RNSC-P1P+GN in *E3* and *E4*. In the experiments, the parameters τ_1, τ_2, and τ_3 are set to 4, 6, and 12 pixels, respectively. As shown in Figs. 5b and 5d, RNSC-P1P+HRE produces the consistently higher number of inliers than RNSC-P1P+GN in spite of the effect of pitch angle and bounding box errors. It demonstrates that the scale estimates of HRE are appropriately bounded by manually determined but geometrically meaningful thresholds at each stage. Consequently, HRE converges to a global minimum even if an initial pose estimate is noisy, whereas the existing estimator converges to local minima, as shown in Figs. 5a and 5c.

4.2 Experiments on a virtual driving simulation dataset

Dataset To evaluate our method in a real application, we generated a dataset from a virtual driving simulation platform. We defined 53 keypoints for 4 types of vehicles (*i.e.*, car, bus, pickup truck, and box truck) in a similar manner with [39]. Then, we generated 2D corresponding keypoints by projecting them using ground truth poses. Ground truth bounding boxes of objects were calculated from 2D projection of the 3D vehicle models. The pitch angle θ_p of the camera

Table 2: Speed and accuracy trade-offs of various keypoint detection networks.

Model	Class acc.	Vertex err. (cm)	GMAC	Latency (ms)
Res50 256×192	0.997	5.24	5.45	13.54
Res10 256×192	0.990	7.12	2.32	5.84
Res50 128×96	0.993	7.57	1.36	7.31
Res10 128×96	0.986	10.05	0.50	3.03

was set to 0° as prior information. However, since the camera of the ego-vehicle was considerably shaking and the road surface was often slanted, the pitch angle was regarded to be very noisy. Images were captured with image resolution of 1914×1080 pixels and a horizontal FOV of 50.8°. In total, we captured 100,000 frames including 207,977 objects which were split into 142,301, 10,997, and 54,679 instances for training, validation, and testing, respectively.

Keypoint detection To detect objects' keypoints, we employ a network architecture similar to [42]. Specifically, we train ResNets[16] on top of which three to four transposed Conv-BN-ReLU blocks are stacked to upscale output feature maps. These networks take a cropped object image as input and have two output heads: one outputs prediction maps whose number of channels equals to the maximum number of pre-defined keypoints of all classes; the other predicts the label of the object class in the input. To investigate the speed and accuracy trade-offs on various backbones and input resolutions, we used two ResNets (Res50 and Res10[7]) as backbones and two resolutions (256×192 and 128×96) as input sizes. More complicated network architectures [3,40] can boost the detection performance further, but is out of scope of this paper. To compare the speed of each model, we measured FLOPS and inference speed of models on Qualcomm(R) Snapdragon(TM)[8] SA8155P's DSP units. All models were trained by using Adam optimizer [23] with 90 epochs and the learning rate was divided by 10 at 60 and 80 epochs with the initial learning rate of 0.001 and weight decay was set to 0.0001. We applied the softargmax to determine the location of each keypoint. Table 2 shows the comparison of the accuracy and speed among these models.

Evaluation We performed two types of experiments: (1) *G1*: object pose estimation using known 3D object models and (2) *G2*: object pose and shape estimation using ASM. In *G1*, we evaluate RNSC-P1P and RNSC-P3P with GN and HRE, respectively, and measure the average rotation and translation errors. In *G2*, we take the same protocol with *G1* but substitute GN to the robust estimation (RE) method of Eq. (6) and additionally measure an average vertex error between ground truth and reconstructed 3D models whose scales are adjusted using a scale difference between ground truth and estimated translation vectors

[7] Res10 models simply remove a residual block in each stage (conv2∼conv5) of Res18.
[8] Qualcomm Snapdragon is a product of Qualcomm Technologies, Inc. and/or its subsidiaries.

Table 3: Rotation, translation, and vertex errors for the experiments on known models (*G1*) and ASM (*G2*). **Bold** and *italic* mean the best and the second best, respectively.

Model		Known models (*G1*)					ASM (*G2*)			
		RNSC-P1P		RNSC-P3P			RNSC-P1P		RNSC-P3P	
		GN	HRE	GN	HRE		RE	HRE	RE	HRE
Res10	$e_r(^\circ)$	3.78	*1.56*	1.90	**1.41**	$e_r(^\circ)$	2.98	**2.20**	3.68	*2.77*
	$e_t(\%)$	6.28	*1.61*	1.76	**1.30**	e_v(cm)	30.53	**21.16**	23.08	*21.28*
Res50	$e_r(^\circ)$	3.76	*1.19*	1.46	**0.96**	$e_r(^\circ)$	2.57	**1.96**	3.16	*2.43*
	$e_t(\%)$	6.38	*1.36*	1.41	**0.93**	e_v(cm)	21.43	*21.23*	22.18	**20.77**

because of its scale ambiguity. In the experiments, the ground truth 2D bounding boxes were used and the parameters t_{in}, τ_1, τ_2, and τ_3 were set to $0.0375l$, $0.0375l$, $0.05l$, and $0.15l$, respectively, where $l = \max(w_o, h_o)$ and w_o and h_o were a width and a height of a 2D object bounding box. Table 3 presents the results for the experiments. It shows that the performance of HRE is superior to those of both GN and RE. In *G1*, RNSC-P3P-based methods are more accurate than RNSC-P1P-based methods due to pitch errors. Nevertheless, the performance of RNSC-P1P+HRE surpasses that of RNSC-P3P+GN as securing more valid inliers by HRE. *G2* is a more difficult scenario because because a pose should be calculated from an inaccurate 3D model (*i.e.*, the mean shape of ASM) Contrary to *G1*, RNSC-P1P-based methods achieve the better rotational accuracy as the constrained pitch angle rather restricts the range of a rotation estimate in the early optimization stage with large shape variation.

Computational time We tested the proposed method, *i.e.*, RNSC-P1P+HRE with the keypoint detection model of Res10-128×96 on the Snapdragon SA8155P processor. Given a 2D object bounding box, the pose and shape estimation took 4.38 ms per object where RNSC-P1P+HRE took 0.15 ms on CPU, keypoint detection took 3.03 ms on DSP, and the pre- and post-processing such as normalization, image cropping and resizing, and softargmax operation for keypoint extraction took the rest of the computational time.

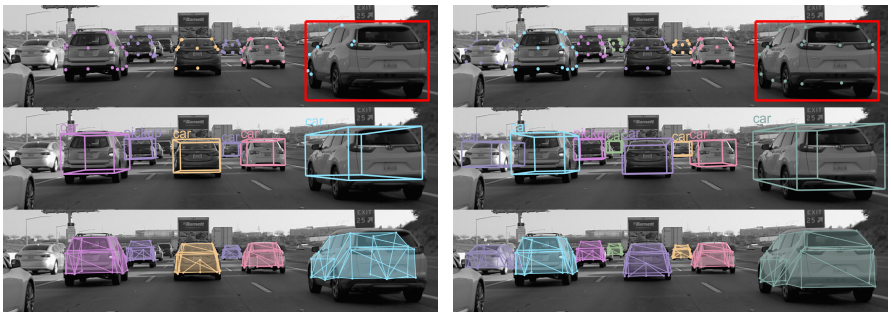
4.3 Experiments on real-world datasets

Experimental setting We used the KITTI object detection dataset [14] to evaluate quantitatively our method in real-world scenes. Following the protocol of [2], we split the KITTI training data into the *training* and *validation* sets. From the training set, we selected 764 and 1019 instance samples for training and validation of the keypoint detection network, respectively. We manually annotated keypoints and then trained the model of Res50 256×192. In addition, we captured images in a real-world scene to compare qualitatively the methods.

Evaluation We evaluate RNSC-P1P and RNSC-P3P with RE and HRE, respectively, on the validation set by measuring rotation and translation errors

Table 4: Rotation and translation errors for the experiments on the KITTI validation set. Each value represents an error at the easy/moderate/hard case.

Method	$e_r(^\circ)$	$e_a(^\circ)$
RNSC-P1P+RE	**3.203**/*4.141*/*4.283*	*0.1376*/*0.1509*/*0.1531*
RNSC-P1P+HRE	*3.365*/**4.016**/**4.071**	**0.1356**/**0.1450**/**0.1460**
RNSC-P3P+RE	4.083/5.482/5.450	0.1511/0.1651/0.1644
RNSC-P3P+HRE	4.021/5.243/5.231	0.1472/0.1616/0.1613



(a) (b)

Fig. 6: Results of RNSC-P1P+RE(a) and RNSC-P1P+HRE(b). Top, middle, and bottom images represent 2D projection of inlier keypoints, 3D bounding boxes, and shape reconstruction results using ASM, respectively.

as in Section 4.2. However, since the translation estimate has scale ambiguity, we employ the average angular error between $\mathbf{t}$ and $\mathbf{t}_{gt}$ as a translation error $e_a(^\circ)$ according to [43]. As shown in Table 4, RNSC-P1P+HRE achieves the best performance in most cases. Figure 6 shows the results of RNSC-P1P+RE and RNSC-P1P+HRE from an input image captured in a real-world scene. In the red box of Fig. 6a, RNSC-P1P+RE reconstructed the shape of the vehicle incorrectly, whereas RNSC-P1P+HRE estimated its shape completely with more inlier keypoints. It demonstrates that the proposed method works well in practical applications that require to detect objects with arbitrary shape under abrupt pitch angle variation by a shaking camera.

5 Conclusion

In this paper, we proposed an efficient method based on 1-point RANSAC to estimate a pose of an object on the ground. Our 1-point RANSAC-based method using 2D bounding box prior information was much faster than the conventional method such as RANSAC+P3P by reducing significantly the number of trials. In addition, our hierarchical robust estimation method using geometrically meaningful and multiple scale estimates produced superior results in the evaluation using synthetic, virtual driving simulation, and real-world datasets.

References

1. Bujnak, M., Kukelova, Z., Pajdla, T.: A general solution to the p4p problem for camera with unknown focal length. In: CVPR (2008)
2. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3d object detection network for autonomous driving. In: CVPR (2017)
3. Cheng, B., Xiao, B., Wang, J., Shi, H., Huang, T.S., Zhang, L.: Bottom-up higher-resolution networks for multi-person pose estimation. arXiv preprint arXiv:1908.10357 (2019)
4. Chhaya, F., Reddy, D., Upadhyay, S., Chari, V., Zia, M.Z., Krishna, K.M.: Monocular reconstruction of vehicles: Combining SLAM with shape priors. In: ICRA. pp. 5758–5765 (may 2016)
5. Cootes, T.F., Taylor, C.J., Cooper, D.H., Graham, J.: Active shape models-their training and application. CVIU **61**(1), 38–59 (1995)
6. Dementhon, D.F., Davis, L.S.: Model-based object pose in 25 lines of code. IJCV **15**(1-2), 123–141 (1995)
7. Dosse, M.B., Kiers, H.A., Ten Berge, J.M.: Anisotropic generalized procrustes analysis. Computational Statistics & Data Analysis **55**(5), 1961–1968 (2011)
8. Ferraz, L., Binefa, X., Moreno-Noguer, F.: Leveraging feature uncertainty in the PnP problem. In: BMVC (2014)
9. Ferraz, L., Binefa, X., Moreno-Noguer, F.: Very fast solution to the pnp problem with algebraic outlier rejection. In: CVPR. pp. 501–508 (2014)
10. Fischler, M.A., Bolles, R.C.: 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)
11. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multiview stereopsis. TPAMI **32**(8), 1362–1376 (2009)
12. Gao, X.S., Hou, X.R., Tang, J., Cheng, H.F.: Complete solution classification for the perspective-three-point problem. TPAMI **25**(8), 930–943 (2003)
13. Garro, V., Crosilla, F., Fusiello, A.: Solving the PnP problem with anisotropic orthogonal procrustes analysis. In: Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission. pp. 262–269 (2012)
14. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR) (2012)
15. Girshick, R.: Fast r-cnn. In: ICCV. pp. 1440–1448 (2015)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
17. Hesch, J.A., Roumeliotis, S.I.: A Direct Least-Squares (DLS) method for PnP. In: ICCV. pp. 383–390 (2011)
18. Horaud, R., Conio, B., Leboulleux, O., Lacolle, B.: An analytic solution for the perspective 4-point problem. Computer Vision, Graphics, and Image Processing **47**(1), 33–44 (1989)
19. Horaud, R., Dornaika, F., Lamiroy, B., Christy, S.: Object pose: The link between weak perspective, paraperspective, and full perspective. IJCV **22**(2), 173–189 (1997)
20. Horn, B.K., Hilden, H.M., Negahdaripour, S.: Closed-form solution of absolute orientation using orthonormal matrices. JOSA A **5**(7), 1127–1135 (1988)
21. Huber, P.J.: Robust statistics, vol. 523. John Wiley & Sons (2004)
22. Jeong, J., Kim, A.: Adaptive inverse perspective mapping for lane map generation with slam. In: URAI. pp. 38–41 (2016)

23. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
24. Kneip, L., Li, H., Seo, Y.: UPnP: An optimal O(n) solution to the absolute pose problem with universal applicability. In: ECCV (2014)
25. Kneip, L., Scaramuzza, D., Siegwart, R.: A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In: CVPR. pp. 2969–2976 (2011)
26. Krishna Murthy, J., Sai Krishna, G., Chhaya, F., Madhava Krishna, K.: Reconstructing vehicles from a single image: shape priors for road scene understanding. In: ICRA (2017)
27. Kukelova, Z., Bujnak, M., Pajdla, T.: Automatic generator of minimal problem solvers. In: ECCV. pp. 302–315 (2008)
28. Lepetit, V., Moreno-Noguer, F., Fua, P.: EPnP: An accurate O(n) solution to the PnP problem. IJCV **81**(2), 155–166 (feb 2009)
29. Li, S., Xu, C., Xie, M.: A robust o(n) solution to the perspective-n-point problem. TPAMI **34**(7), 1444–1450 (2012)
30. Li, S.Z., Wang, H., Soh, W.Y.: Robust estimation of rotation angles from image sequences using the annealing M-estimator. Journal of Mathematical Imaging and Vision **8**(2), 181–192 (1998)
31. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: ECCV. pp. 21–37 (2016)
32. Lourakis, M., Zabulis, X.: Model-Based Pose Estimation for Rigid Objects. In: International Conference on Computer Vision Systems. pp. 83–92 (2013)
33. Lowe, D.G.: Fitting parameterized three-dimensional models to images. TPAMI **13**(5), 441–450 (1991)
34. Lu, C.P., Hager, G.D., Mjolsness, E.: Fast and globally convergent pose estimation from video images. TPAMI **22**(6), 610–622 (jun 2000)
35. Mur-Artal, R., Tardós, J.D.: Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. TRO **33**(5), 1255–1262 (2017)
36. Murthy, J.K., Sharma, S., Krishna, K.M.: Shape priors for real-time monocular object localization in dynamic environments. In: IROS. pp. 1768–1774 (2017)
37. Nistér, D., Stewénius, H.: A minimal solution to the generalised 3-point pose problem. Journal of Mathematical Imaging and Vision **27**(1), 67–79 (2007)
38. Quan, L., Lan, Z.: Linear n-point camera pose determination. TPAMI **21**(8), 774–780 (1999)
39. Song, X., Wang, P., Zhou, D., Zhu, R., Guan, C., Dai, Y., Su, H., Li, H., Yang, R.: ApolloCar3D: A large 3D car instance understanding benchmark for autonomous driving. In: CVPR (2019)
40. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: CVPR. pp. 5693–5703 (2019)
41. Urban, S., Leitloff, J., Hinz, S.: MLPNP-A real-time maximum likelihood solution to the perspective-n-point problem. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences (2016)
42. Xiao, B., Wu, H., Wei, Y.: Simple baselines for human pose estimation and tracking. In: ECCV. pp. 466–481 (2012)
43. Yang, J., Li, H., Jia, Y.: Optimal essential matrix estimation via inlier-set maximization. In: ECCV. pp. 111–126 (2014)
44. Yohai, V.J.: High breakdown-point and high efficiency robust estimates for regression. The Annals of Statistics pp. 642–656 (1987)
45. Zeeshan Zia, M., Stark, M., Schiele, B., Schindler, K.: Detailed 3D representations for object recognition and modeling. TPAMI **35**(11), 2608–2623 (2013)

46. Zhang, D., Fang, B., Yang, W., Luo, X., Tang, Y.: Robust inverse perspective mapping based on vanishing point. In: IEEE International Conference on Security, Pattern Analysis, and Cybernetics. pp. 458–463 (dec 2014)
47. Zheng, Y., Kuang, Y., Sugimoto, S., Astrom, K., Okutomi, M.: Revisiting the pnp problem: A fast, general and optimal solution. In: ICCV. pp. 2344–2351 (2013)
48. Zheng, Y., Sugimoto, S., Okutomi, M.: Aspnp: An accurate and scalable solution to the perspective-n-point problem. IEICE Transactions on Information and Systems **96**(7), 1525–1535 (2013)

A RANSAC-based scheme for object pose estimation

A general framework for n-point RANSAC-based pose estimation is shown in Alg. A.1.

Algorithm A.1 Pose estimation using RANSAC and a PnP algorithm

Require: $\mathcal{C}$
Ensure: $\mathbf{T}$
$n_{best} \leftarrow 0, \mathbf{T} \leftarrow \varnothing$
for $t = 1$ **to** N **do**
  Randomly sample n 2D-3D keypoint correspondences from $\mathcal{C}$.
  $\mathbf{T}_{cand} \leftarrow$ Compute a pose candidate using the PnP algorithm from the n samples.
  $n_{cand} \leftarrow$ Count the number of inliers using $\mathbf{T}_{cand}$.
  if $n_{cand} > n_{best}$ **then**
    $\mathbf{T} \leftarrow \mathbf{T}_{cand}, n_{best} = n_{cand}$
  end if
end for
if $\mathbf{T} \neq \varnothing$ **then**
  $\mathbf{T} \leftarrow$ Refine $\mathbf{T}$ using the inlier points.
end if

B Details of perspective-1-point solution

B.1 Definition of pitch angle

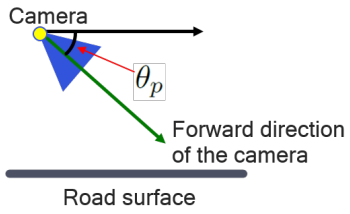


Fig. B.1: Definition of pitch angle

Since we want to simplify the PnP problem, we redefine the problem in the bird-eye view (BEV) in this paper. Figure B.1 shows the definition of a pitch angle θ_p of a camera. Once the pitch angle between the camera and the road surface is known, a 3D directional vector $\mathbf{V}_i$, *i.e.*, a back-projected ray of a pixel on an image, is approximated to a 2D directional vector $\mathbf{v}_i$ on the top-view as follows.

$$\mathbf{v}_i = \begin{bmatrix} V_X \\ V_Z \end{bmatrix}, \text{ where } \begin{bmatrix} V_X \\ V_Y \\ V_Z \end{bmatrix} = \mathbf{R}_{cg}^{-1}\mathbf{V}_i \tag{B.1}$$

B.2 Parameter calculation

Many parameters[9] such as ϕ_y, θ_L, θ_R, ψ_L, ψ_R, l_L, and l_R should be calculated prior to derivation of Eqs. (1) and (2). The parameters ϕ_y, θ_L, and θ_R are computed as follows.

$$\phi_y = \arcsin \mathbf{v}_C \cdot \mathbf{v}_K \tag{B.2}$$
$$\theta_L = \arccos \mathbf{v}_K \cdot \mathbf{v}_L \tag{B.3}$$
$$\theta_R = \arccos \mathbf{v}_K \cdot \mathbf{v}_R \tag{B.4}$$

Let's denote the i^{th} corner point as $\mathbf{p}_i$ and a selected keypoint as $\mathbf{p}_K$. Then, the parameters ψ_L, ψ_R, l_L, and l_R are calculated depending on a case as the following table.

Table B.1: Parameter computation for the four cases

	Case 1	Case 2	Case 3	Case 4
l_L	$\|\mathbf{p}_K - \mathbf{p}_4\|$	$\|\mathbf{p}_K - \mathbf{p}_3\|$	$\|\mathbf{p}_K - \mathbf{p}_1\|$	$\|\mathbf{p}_K - \mathbf{p}_2\|$
l_R	$\|\mathbf{p}_K - \mathbf{p}_2\|$	$\|\mathbf{p}_K - \mathbf{p}_1\|$	$\|\mathbf{p}_K - \mathbf{p}_3\|$	$\|\mathbf{p}_K - \mathbf{p}_4\|$
ψ_L	$-\text{acos } \mathbf{v}_f \cdot \frac{\mathbf{p}_4 - \mathbf{p}_K}{l_L}$	$\text{acos } \mathbf{v}_f \cdot \frac{\mathbf{p}_3 - \mathbf{p}_K}{l_L}$	$-\text{acos } \mathbf{v}_f \cdot \frac{\mathbf{p}_1 - \mathbf{p}_K}{l_L}$	$\text{acos } \mathbf{v}_f \cdot \frac{\mathbf{p}_2 - \mathbf{p}_K}{l_L}$
ψ_R	$\text{acos } \mathbf{v}_f \cdot \frac{\mathbf{p}_2 - \mathbf{p}_K}{l_R}$	$-\text{acos } \mathbf{v}_f \cdot \frac{\mathbf{p}_1 - \mathbf{p}_K}{l_R}$	$\text{acos } \mathbf{v}_f \cdot \frac{\mathbf{p}_3 - \mathbf{p}_K}{l_R}$	$-\text{acos } \mathbf{v}_f \cdot \frac{\mathbf{p}_4 - \mathbf{p}_K}{l_R}$

B.3 Derivation of P1P solution

Here, we show the process of the derivation of Eq. (2). First of all, we prove Case 1. From the sine rule,

$$\frac{l_L}{\sin\theta_L} = \frac{l_C}{\sin\phi_L} \text{ and } \frac{l_R}{\sin\theta_R} = \frac{l_C}{\sin\phi_R}, \tag{B.5}$$

an equation is formulated as follows.

$$l_C = \frac{l_R \sin\phi_R}{\sin\theta_R} = \frac{l_L \sin\phi_L}{\sin\theta_L} \tag{B.6}$$

[9] Please see the details of the parameters in Fig. 3.

Because $\phi_R = \theta_y + \psi_R - \theta_R$ ($\phi_R > 0$) and $\phi_L = -\theta_y - \psi_L - \theta_L$ ($\phi_L > 0$), Eq. (B.6) is substituted to

$$\frac{l_R \sin(\theta_y + \psi_R - \theta_R)}{\sin\theta_R} = \frac{l_L \sin(-\theta_y - \psi_L - \theta_L)}{\sin\theta_L}. \tag{B.7}$$

Let $\omega_R = \psi_R - \theta_R$ and $\omega_L = \psi_L + \theta_L$. According to the angle addition and subtraction formulae, the sine functions are decomposed as

$$l_R \frac{\sin\theta_y \cos\omega_R + \cos\theta_y \sin\omega_R}{\sin\theta_R} = l_L \frac{\sin\theta_y \cos\omega_L - \cos\theta_y \sin\omega_L}{\sin\theta_L}, \tag{B.8}$$

because $\psi_R - \theta_R > 0$ and $-\psi_L - \theta_L > 0$. Eq. (B.8) is reorganized with respect to $\sin\theta_y$ and $\cos\theta_y$ as

$$\sin\theta_y \left(\frac{l_R \cos\omega_R}{\sin\theta_R} + \frac{l_L \cos\omega_L}{\sin\theta_L} \right) = \cos\theta_y \left(-\frac{l_R \sin\omega_R}{\sin\theta_R} - \frac{l_L \sin\omega_L}{\sin\theta_L} \right). \tag{B.9}$$

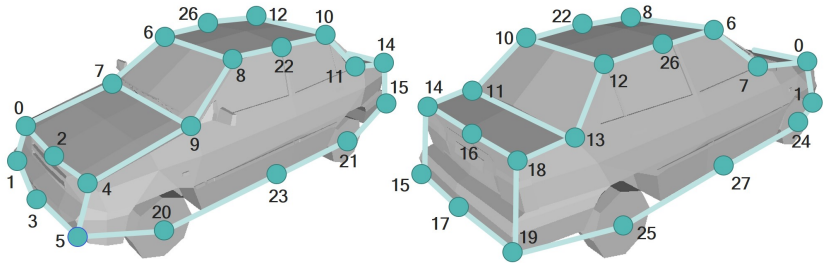
Finally, we reorganize Eq. (B.9) with respect of θ_y as

$$\theta_y = \arctan\left(\frac{-\frac{l_R \sin(\psi_R - \theta_R)}{\sin\theta_R} - \frac{l_L \sin(\psi_L + \theta_L)}{\sin\theta_L}}{\frac{l_R \cos(\psi_R - \theta_R)}{\sin\theta_R} + \frac{l_L \cos(\psi_L + \theta_L)}{\sin\theta_L}} \right). \tag{B.10}$$

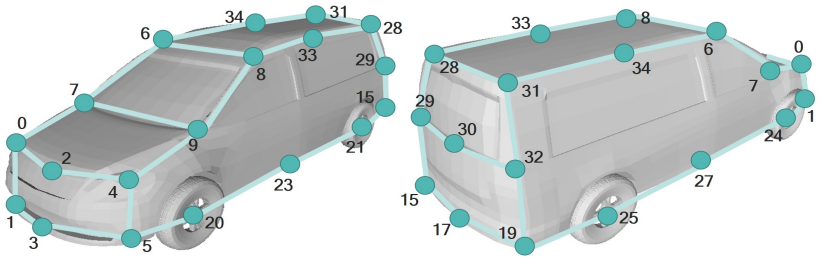
For Cases 2-4, the local yaw angle θ_y is computed in the same way of Eqs. (B.5)-(B.10).

C Definition of keypoints

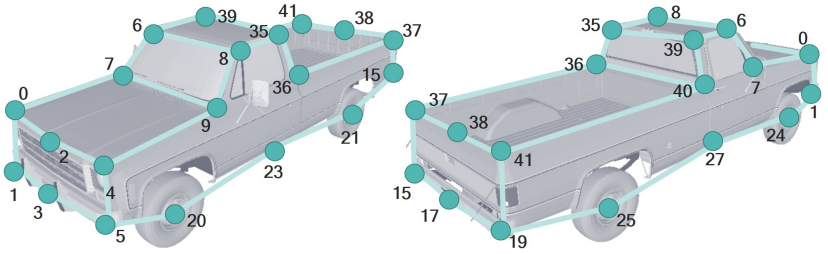
We define 53 keypoints and 4 types of vehicles: *Car*, *Van-Bus*, *PickupTruck*, and *BoxTruck*. The numbers of keypoints of *Car*, *Van-Bus*, *PickupTruck*, and *BoxTruck* are 28, 26, 26, and 30, respectively. Some keypoints are shared among the classes. The location and index of each keypoint are shown in Fig. C.1.



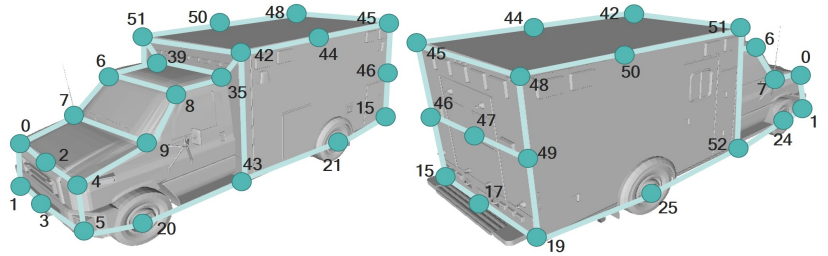
(a)



(b)



(c)



(d)

Fig. C.1: Definition of keypoints of *Car*, *Van-Bus*, *PickupTruck*, and *BoxTruck*