Supplementary Material for Minimal Cases for Computing the Generalized Relative Pose using Affine Correspondences

Banglei Guan[1], Ji Zhao[*], Daniel Barath[2] and Friedrich Fraundorfer[3,4]

[1]College of Aerospace Science and Engineering, National University of Defense Technology, China

[2]Computer Vision and Geometry Group, Department of Computer Science, ETH Zürich

[3]Institute for Computer Graphics and Vision, Graz University of Technology, Austria

[4]Remote Sensing Technology Institute, German Aerospace Center, Germany

guanbanglei12@nudt.edu.cn zhaoji84@gmail.com dbarath@ethz.ch fraundorfer@icg.tugraz.at

1. Geometric Constraints from ACs

For AC $(\mathbf{x}_{ij}, \mathbf{x}'_{ij}, \mathbf{A})$, we get three polynomials for six unknowns $\{q_x, q_y, q_z, t_x, t_y, t_z\}$ from Eqs. (4) and (9) in the paper. After separating q_x, q_y, q_z from t_x, t_y, t_z, we arrive at equation system

$$\frac{1}{1+q_x^2+q_y^2+q_z^2} \underbrace{\begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \end{bmatrix}}_{\mathbf{M}(q_x,q_y,q_z)} \begin{bmatrix} t_x \\ t_y \\ t_z \\ 1 \end{bmatrix} = \mathbf{0}, \tag{1}$$

where the elements M_{ij} $(i = 1, \ldots, 3; j = 1, \ldots, 4)$ of the coefficient matrix $\mathbf{M}(q_x, q_y, q_z)$ are formed by the polynomial coefficients and three unknown variables q_x, q_y, q_z:

$$\mathbf{M}(q_x, q_y, q_z) = \begin{bmatrix} [2] & [2] & [2] & [2] \\ [2] & [2] & [2] & [2] \\ [2] & [2] & [2] & [2] \end{bmatrix}, \tag{2}$$

where $[N]$ denotes a polynomial of degree N in variables q_x, q_y, q_z.

Equation (1) imposes three independent constraints on six unknowns $\{q_x, q_y, q_z, t_x, t_y, t_z\}$. This constraint can be easily generalized to special cases of multi-camera motion, *e.g.*, planar motion and known vertical direction.

2. Relative Pose Under Planar Motion

2.1. Details about the Coefficient Matrix $\mathrm{M}(q_y)$

Refer to Eq. (11) in the paper, three constraints obtained from a single AC are stacked into three equations in three unknowns. The elements M_{ij} $(i = 1, \ldots, 3; j = 1, \ldots, 3)$ of the coefficient matrix $\mathbf{M}(q_y)$ are formed by the polynomial coefficients and one unknown variable q_y, which can be described as:

$$\mathbf{M}(q_y) = \begin{bmatrix} [2] & [2] & [2] \\ [2] & [2] & [2] \\ [2] & [2] & [2] \end{bmatrix}, \tag{3}$$

where $[N]$ denotes a polynomial of degree N in variable q_y.

2.2. Degenerate Case

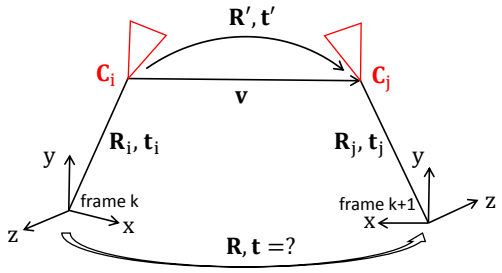


Figure 1. Planar motion of a multi-camera system.

Proposition 1. *Consider a multi-camera system which is under planar motion. Assume the following three conditions are satisfied. (1) The rotation axis is y-axis, and the translation is on xz-plane. (2) There is one AC across camera C_i in frame k and camera C_j in frame $k+1$ (C_i and C_j can be the same or different cameras). (3) The optical centers of camera C_i and C_j have the same y-coordinate. Then this case is degenerate. Specifically, the rotation can be correctly recovered, while both the translation direction and the translation scale cannot be estimated using one AC.*

Proof. Figure 1 illustrates the degenerate case described in the proposition. Note that the multi-camera reference frame

[*]Corresponding author.

Methods	17pt-Li [10]	8pt-Kneip [8]	6pt-St. [6]	4pt-Lee [9]	4pt-Sw. [14]	4pt-Liu [11]	6AC-Ven. [1]	**1AC plane**	**2AC plane**	**2AC vertical**
Timings	43.3	102.0	3275.4	26.5	22.2	3.7	38.1	**3.6**	**3.6**	17.8

Table 1. Run-time comparison of motion estimation algorithms (unit: μs).

is established on the multi-camera system, not on a certain camera coordinate system. Our proof is based on the following observation: whether a case is degenerate is independent of the relative pose solvers. Based on this point, we construct a new minimal solver which is different from the proposed solver in the paper.

(i) Since the multi-camera system is rotated by y-axis, the camera C_i in frame k and camera C_j in frame $k+1$ are under motion with known rotation axis. Thus we can use the `1AC method` [5] for perspective cameras to estimate the relative pose between C_i and C_j. This is a minimal solver since one AC provides 3 independent constraints and there are three unknowns (one unknown for rotation, two unknowns for translation by excluding scale-ambiguity). Denote the recovered rotation and translation between C_i and C_j as $(\mathbf{R}', \mathbf{t}')$, where $\mathbf{t}'$ is a unit vector. The scale of the translation vector cannot be recovered at this moment. Denote the unknown translation scale as λ.

(ii) From Fig. 1, we have

$$\begin{aligned}\begin{bmatrix}\mathbf{R} & \mathbf{t}\\ \mathbf{0} & 1\end{bmatrix} &= \begin{bmatrix}\mathbf{R}_j & \mathbf{t}_j\\ \mathbf{0} & 1\end{bmatrix}\begin{bmatrix}\mathbf{R}' & \lambda\mathbf{t}'\\ \mathbf{0} & 1\end{bmatrix}\begin{bmatrix}\mathbf{R}_i & \mathbf{t}_i\\ \mathbf{0} & 1\end{bmatrix}^{-1}\\ &= \begin{bmatrix}\mathbf{R}_j\mathbf{R}'\mathbf{R}_i^T & \lambda\mathbf{R}_j\mathbf{t}' + \mathbf{t}_j - \mathbf{R}_j\mathbf{R}'\mathbf{R}_i^T\mathbf{t}_i\\ \mathbf{0} & 1\end{bmatrix}.\end{aligned} \tag{4}$$

From Eq. (4), we have

$$\mathbf{R} = \mathbf{R}_j\mathbf{R}'\mathbf{R}_i^T, \tag{5}$$

$$\mathbf{t} = \lambda\mathbf{R}_j\mathbf{t}' + \mathbf{t}_j - \mathbf{R}_j\mathbf{R}'\mathbf{R}_i^T\mathbf{t}_i. \tag{6}$$

From Eq. (5), the rotation $\mathbf{R}$ between frame k and frame $k+1$ for the multi-camera system can be recovered. From Eq. (6), we have

$$\lambda(\mathbf{R}_j\mathbf{t}') - \mathbf{t} + (\mathbf{t}_j - \mathbf{R}\mathbf{t}_i) = \mathbf{0}. \tag{7}$$

In Eq. (7), note that $\mathbf{t} = [t_x, 0, t_z]^T$ due to planar motion. Thus this linear equation system has 3 unknowns $\{\lambda, t_x, t_z\}$ and 3 equations. Usually the unknowns can be uniquely determined by solving this equation system. However, if the second entry of $\mathbf{R}_j\mathbf{t}'$ is zero, it can be verified that λ becomes a free parameter. In other words, the translation cannot be determined and this is a degenerate case.

(iii) Finally, we exploit the geometric meaning of the degenerate case, i.e., the second entry of $\mathbf{R}_j\mathbf{t}'$ is zero. Denote the normalized vector originated from C_i to C_j as $\mathbf{v}$. Since $\mathbf{v}$ represents the normalized translation vector between C_i and C_j, the coordinates of $\mathbf{v}$ in reference of camera C_j is $\mathbf{t}'$. Further, the coordinates of $\mathbf{v}$ in frame $k+1$ is $\mathbf{R}_j\mathbf{t}'$. The second entry of $\mathbf{R}_j\mathbf{t}'$ is zero means that the endpoints of $\mathbf{v}$ have the same y-coordinate in frame $k+1$, which is the condition (3) in the proposition. □

3. Relative Pose with Known Vertical Direction

Refer to Eq. (22) in the paper, four constraints obtained from two ACs are stacked into four equations in four unknowns. The elements $\tilde{M}_{ij}$ $(i = 1, \ldots, 4; j = 1, \ldots, 4)$ of the coefficient matrix $\tilde{\mathbf{M}}(q_y)$ are formed by the polynomial coefficients and one unknown variable q_y, which can be described as:

$$\tilde{\mathbf{M}}(q_y) = \begin{bmatrix}[2] & [2] & [2] & [2]\\ [2] & [2] & [2] & [2]\\ [2] & [2] & [2] & [2]\\ [2] & [2] & [2] & [2]\end{bmatrix}, \tag{8}$$

where $[N]$ denotes a polynomial of degree N in variable q_y.

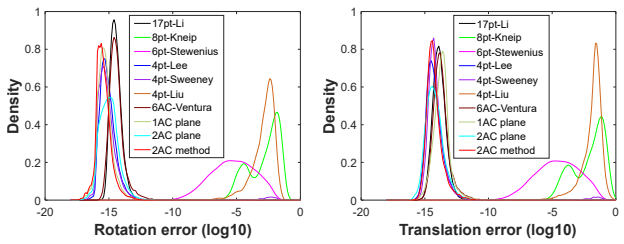
4. Experiments

4.1. Efficiency Comparison

The runtimes of the solvers are evaluated on an Intel(R) Core(TM) i7-7800X 3.50GHz. All algorithms are implemented in C++. Methods `17pt-Li`, `8pt-Kneip` and `6pt-Stewenius` are provided in the OpenGV library [7]. We implemented the `4pt-Lee` method. For methods `4pt-Sweeney`, `4pt-Liu` and `6AC-Ventura`, we used their publicly available implementations from GitHub. The average, over 10,000 runs, processing times of the solvers are shown in Table 1. The runtimes of the methods `1AC plane` , `2AC plane` and `4pt-Liu` are the lowest, because these methods solve the 4-degree polynomial equation. The `2AC vertical` which solves the 6-degree polynomial equation also requires low computation time.

4.2. Numerical Stability

Figure 2 reports the numerical stability of the solvers in the noise-free case. The procedure is repeated 10,000 times. The empirical probability density functions (vertical axis) are plotted as the function of the $\log_{10}$ estimated errors (horizontal axis). Methods `1AC plane`, `2AC plane`, `2AC vertical`, `17pt-Li`[10], `4pt-Lee` [9], `4pt-Sweeney` [14] and `6AC-Ventura` [1] are numerically stable. It can also be seen that the `4pt-Sweeney` method has a small peak, both in the rotation and translation error curves, around 10^{-2}. The `8pt-Kneip` method based on iterative optimization is susceptible to falling into local minima. Due to



(a) (b)

Figure 2. Probability density functions over estimation errors in the noise-free case (10 000 runs). The horizontal axis represents the $\log_{10}$ errors and the vertical axis represents the density. (a) reports the rotation error. (b) reports the translation error. The proposed `1AC plane` method, `2AC plane` method and `2AC vertical` are compared against `17pt-Li` [10], `8pt-Kneip` [8], `6pt-Stewénius` [6], `4pt-Lee` [9], `4pt-Sweeney` [14], `4pt-Liu` [11] and `6AC-Ventura` [1].

the use of first-order approximation of the relative rotation, the `4pt-Liu` method inevitably has greater than zero error in the noise-free case.

4.3. Planar Motion Estimation

In addition to efficiency and numerical stability, another important factor for a solver is the minimal number of required image points. The iteration number N of RANSAC can be computed by $N = \log(1-p)/\log(1-(1-\epsilon)^s)$, where s is the number of minimal image points, ϵ is the outlier ratio, and p is the success probability. For a probability of success $p = 99\%$, the RANSAC iterations needed with respect to the outlier ratio needed are shown in Figure 3. It can be seen that the iteration number of the RANSAC estimator increases exponentially with respect to the number of image points needed. For example, in a percentage of outliers $\epsilon = 50\%$, when the solvers require 1, 2, 4, 6, 8 and 17 points, the RANSAC estimator need 7, 16, 71, 292, 1177 and 603607 iterations, respectively. The proposed `1AC plane` method which only uses a single AC requires the lowest number of RANSAC iterations. Since the proposed `2AC plane` method need two ACs, the iteration number of RANSAC is also low in comparison to PC-based methods. Thus, our solvers can be used efficiently for detecting a correct inlier set when integrating them into the RANSAC framework.

We evaluate the performance of the proposed `1AC plane` method and `2AC plane` method for outlier detection in presence of outliers. The outlier ratio is set to 50%. The other configurations of this synthetic experiment are set as same as using in Figure 3(d–f) in the paper. Figure 4 shows the performance of the proposed methods against planar motion noise. It is interesting to see that the `1AC plane` method recovers more than 50% inliers and requires fewer number of RANSAC iterations,

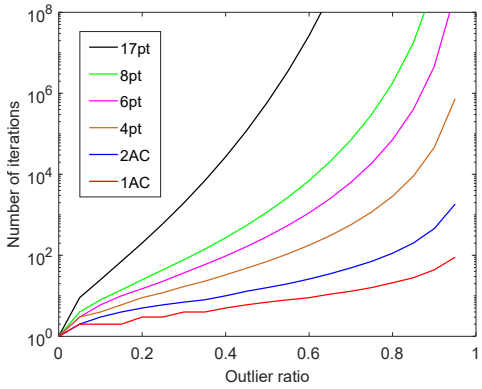
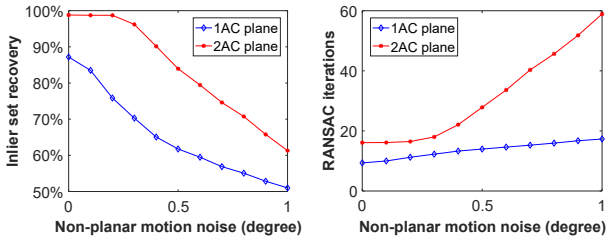


Figure 3. Comparison of the RANSAC iteration number for 99% of success probability.



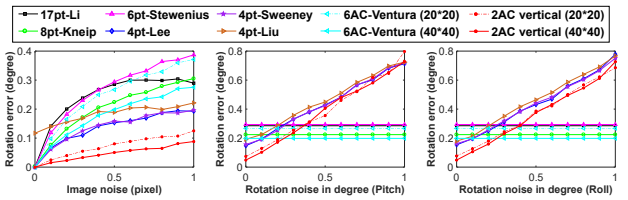
(a) (b)

Figure 4. Rotation and translation error with varying planar motion noise. The image noise is fixed at 0.5 pixel and the outlier ratio is set to 50%.

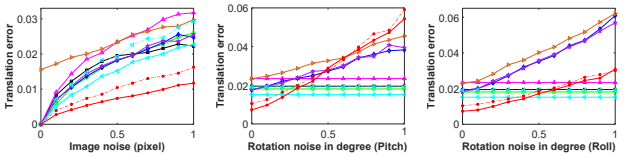
even though it performs poorly in translation estimation as shown in Figure 3(e–f) in the paper. Thus, the `1AC plane` method has the advantage of detecting a correct inlier set efficiently, which can then be used for accurate motion estimation with non-linear optimization.

4.4. Motion with Known Vertical Direction

In this section we show the performance of the proposed `2AC vertical` under forward and sideways motion. Figure 5 shows the performance of the proposed `2AC vertical` under forward motion. It can be seen that `2AC vertical` outperforms the comparative methods against image noise and provides comparable accuracy for increasing IMU noise, even though the size of the square is 20 pixels. Figure 6 shows the performance of the proposed `2AC vertical` under sideways motion. The results demonstrate that when the side length of the square is 40 pixels, the `2AC vertical` performs basically better than all compared methods against image noise and achieves comparable performance for increasing noise on the IMU data.

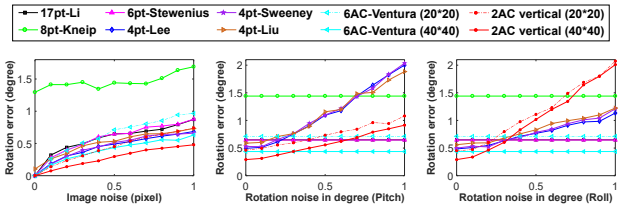


(a) $\varepsilon_{\mathbf{R}}$ with image noise (b) $\varepsilon_{\mathbf{R}}$ with pitch angle noise (c) $\varepsilon_{\mathbf{R}}$ with roll angle noise

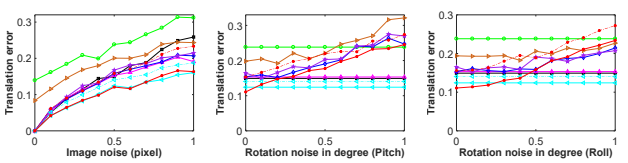


(d) $\varepsilon_{\mathbf{t}}$ with image noise (e) $\varepsilon_{\mathbf{t}}$ with pitch angle noise (f) $\varepsilon_{\mathbf{t}}$ with roll angle noise

Figure 5. Rotation and translation error under forward motion with known vertical direction. Upper row: rotation error. Bottom row: translation error. (a,d): varying image noise. (b,e) and (c,f): varying IMU angle noise and fixed 1.0 pixel std. image noise.



(a) $\varepsilon_{\mathbf{R}}$ with image noise (b) $\varepsilon_{\mathbf{R}}$ with pitch angle noise (c) $\varepsilon_{\mathbf{R}}$ with roll angle noise



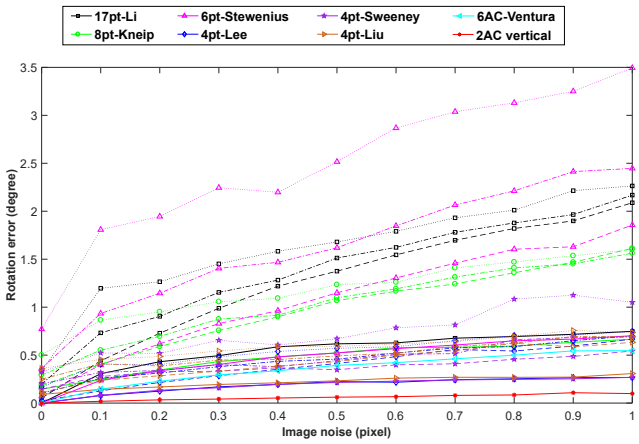
(d) $\varepsilon_{\mathbf{t}}$ with image noise (e) $\varepsilon_{\mathbf{t}}$ with pitch angle noise (f) $\varepsilon_{\mathbf{t}}$ with roll angle noise

Figure 6. Rotation and translation error under sideways motion with known vertical direction. Upper row: rotation error. Bottom row: translation error. (a,d): varying image noise. (b,e) and (c,f): varying IMU angle noise and fixed 1.0 pixel std. image noise.

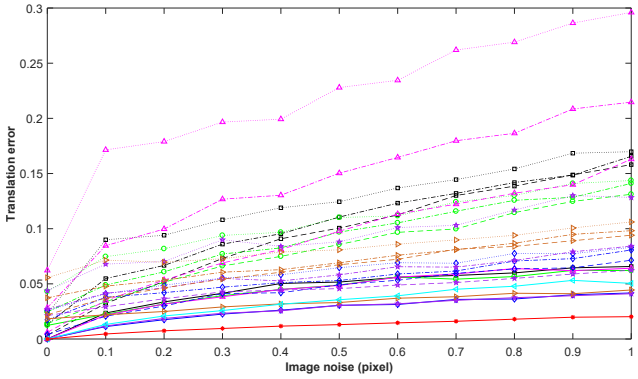
4.5. Using PCs converted from ACs

In this set of experiments, we evaluate the performance of PC-based solvers using the PCs converted from ACs. Given an AC as $(\mathbf{x}, \mathbf{x}', \mathbf{A})$, where $\mathbf{x}$ and $\mathbf{x}'$ are the image coordinates of feature point in two views and $\mathbf{A}$ is the corresponding 2×2 local affine transformation. Three generated PCs include an image point pair of AC and two hallucinated image point pairs calculated by the local affine transformation. Since local affine transformations are defined as the partial derivative, w.r.t. the image directions, of the related homography, they are valid only infinitesimally close to the image coordinates of AC. Thereby, one AC can only provide three approximate PCs the error is not zero even for noise-free input [2]. Three approximate PCs converted from one AC can be computed as follows [3]: $\mathbf{x} + [0,\ w,\ 0;\ 0,\ 0,\ w]$ and $\mathbf{x}' + \mathbf{A}[0,\ w,\ 0;\ 0,\ 0,\ w]$, where w determines the distribution area of the generated PCs. To evaluate the performance of PC-based solvers with different distribution area, w is set to 1, 5 and 10 pixels, respectively.

Take relative pose estimation with known vertical direction for an example. A total of 1000 trials are carried out in the synthetic experiment. In each test, 100 ACs are generated randomly with 40*40 support region. In the RANSAC loop, six ACs and two ACs are selected randomly for the `6AC-Ventura` method and the proposed `2AC vertical` method, respectively. The hallucinated PCs converted from a minimal num-



(a) $\varepsilon_{\mathbf{R}}$ with image noise



(b) $\varepsilon_{\mathbf{t}}$ with image noise

Figure 7. Rotation and translation error with varying image noise under random motion with known vertical direction. Solid line indicates using image point pairs of ACs. Dashed line, dash-dotted line and dotted line indicate using the hallucinated PCs, which are generated with different distribution area w = 1, 5, 10 pixels, respectively.

Part	17pt-Li [10]		8pt-Kneip [8]		6pt-St. [6]		4pt-Lee [9]		4pt-Sw. [14]		4pt-Liu [11]		6AC-Ven. [1]		**2AC plane**		**2AC vertical**	
	$\varepsilon_{\mathbf{R}}$	$\varepsilon_{\mathbf{t}}$	$\varepsilon_{\mathbf{R}}$	$\varepsilon_{\mathbf{t}}$	$\varepsilon_{\mathbf{R}}$	$\varepsilon_{\mathbf{t}}$	$\varepsilon_{\mathbf{R}}$	$\varepsilon_{\mathbf{t}}$	$\varepsilon_{\mathbf{R}}$	$\varepsilon_{\mathbf{t}}$	$\varepsilon_{\mathbf{R}}$	$\varepsilon_{\mathbf{t}}$	$\varepsilon_{\mathbf{R}}$	$\varepsilon_{\mathbf{t}}$	$\varepsilon_{\mathbf{R}}$	$\varepsilon_{\mathbf{t}}$	$\varepsilon_{\mathbf{R}}$	$\varepsilon_{\mathbf{t}}$
01 (3376 images)	0.161	2.680	0.156	2.407	0.203	2.764	0.083	1.780	0.078	1.659	0.108	1.941	0.143	2.366	0.344	2.284	**0.057**	**1.469**

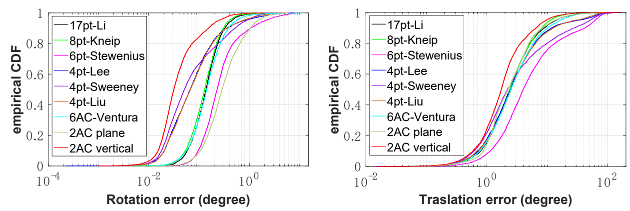
Table 2. Rotation and translation error on `nuScenes` sequences (unit: degree).

ber of ACs are used as input for the PC-based solvers. Thus, 6, 3 and 2 ACs are selected randomly for the `17pt-Li` solver [10], the `8pt-Kneip` solver [8], and the solvers `6pt-Stewénius` [6], `4pt-Lee` [9], `4pt-Sweeney` [14] and `4pt-Liu` [11], respectively. Note that the hallucinated PCs converted from ACs are only used for hypothesis generation, and the inlier set is found by evaluating the image point pairs of ACs. The solution which produces the highest number of inliers is chosen. The other configurations of this synthetic experiment are set as same as using in Figure 4(a) and (d) in the paper.

Figure 7 shows the performance of the PC-based solvers against image noise in the random motion case. The estimation results using the image point pairs of ACs are represented by solid lines. The estimation results using the hallucinated PCs generated with different distribution area are represented by dashed line (w = 1 pixel), dash-dotted line (w = 5 pixels) and dotted line (w = 10 pixels), respectively. We have the following observations. (1) The PC-based solvers using the hallucinated PCs perform worse than using the image point pairs of AC. Because the conversion error between each AC and three PCs is newly introduced. It can be seen that the estimation error of PC-based solvers using the hallucinated PCs is not zero even for image noise-free input. Moreover, the hallucinated PCs generated by each AC are near each other which may be a degenerate case for the PC-based solvers. (2) The performance of PC-based solvers is influenced by the different distribution area of hallucinated PCs. Since a smaller distribution area causes smaller conversion error between ACs and PCs, the PC-based solvers have better performance with smaller distribution area. (3) The performance of the proposed `2AC vertical` method is best. Because the AC-based solvers use the relationship between local affine transformations and epipolar lines (Eq. (9) in the paper). This is a strictly satisfied constraint and does not result in any error for noise-free input. In addition, the `2AC vertical` method is robust to image noise and performs better than the `6AC-Ventura` method.

4.6. Experiments on KITTI dataset

We also show the empirical cumulative error distributions for KITTI sequence 00. These values are calculated from the same values which were used for creating Table 2 in the paper. Figure 8 shows the proposed `2AC vertical` method offers the best overall performance in comparison to state-of-the-art methods.



(a) (b)

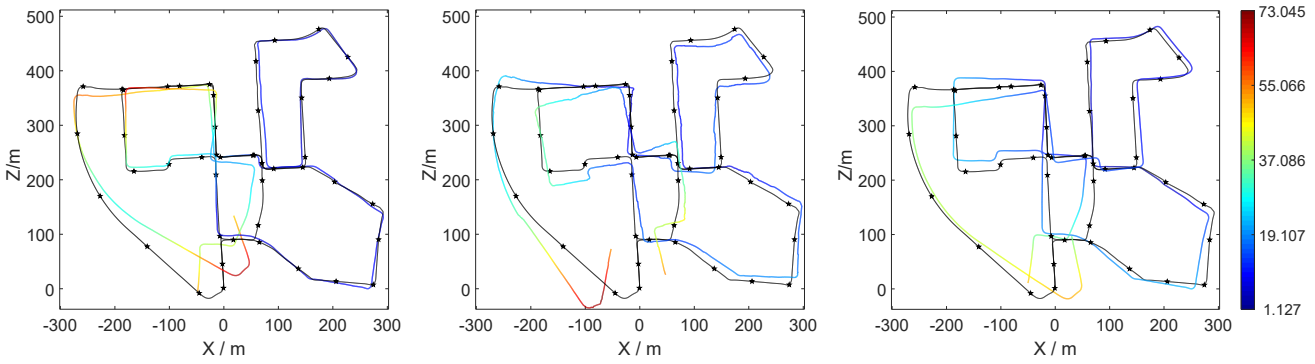
Figure 8. Empirical cumulative error distributions for KITTI sequence 00. (a) reports the rotation error. (b) reports the translation error. The proposed `2AC plane` method and `2AC vertical` are compared against `17pt-Li` [10], `8pt-Kneip` [8], `6pt-Stewénius` [6], `4pt-Lee` [9], `4pt-Sweeney` [14] and `4pt-Liu` [11].

To visualize the comparison results, the estimated trajectory for sequence 00 is plotted in Fig. 9. We are directly concatenating frame-to-frame relative pose measurements without any post-refinement. The trajectory for `2AC vertical` is compared with the two best performing comparison methods in sequence 00 based on Table 2 in the paper: `8pt-Kneip` in 6DOF motion case and `4pt-Sweeney` in 4DOF motion case. Since all methods were not able to estimate the scale correctly, in particular for the many straight parts of the trajectory, the ground truth scale is used to plot the trajectories. Then the trajectories are aligned with the ground truth and the color along the trajectory encodes the absolute trajectory error (ATE) [13]. Even though all trajectories have a significant accumulation of drift, it can still be seen that the `2AC vertical` method has the smallest ATE among the compared trajectories.

4.7. Experiments on nuScenes dataset

We also test the performance of our methods on the `nuScenes` dataset [4], which consists of consecutive keyframes from 6 cameras. All the keyframes of Part 1 are used for the evaluation and there are 3376 images in total. The ground truth pose is provided from a lidar map-based localization scheme. Similar to the experiments on `KITTI` dataset, the ACs between consecutive keyframes in each camera are established by applying the ASIFT [12] detector. All solvers are used within RANSAC.

Table 2 shows the results of the rotation and translation estimation for the Part1 of `nuScenes` dataset. The median error is used to evaluate the estimation accuracy. It can be seen that the proposed `2AC vertical` method offers the best overall performance among all the methods.



(a) `8pt-Kneip` (b) `4pt-Sweeney` (c) `2AC vertical`

Figure 9. Estimated trajectories without any post-refinement. The relative pose measurements between consecutive frames are directly concatenated. The colorful curves are the trajectories estimated by `8pt-Kneip` [8], `4pt-Sweeney` [14] and `2AC vertical`. Black curves with stars are the ground truth trajectories. Best viewed in color.

References

[1] Khaled Alyousefi and Jonathan Ventura. Multi-camera motion estimation with affine correspondences. In *International Conference on Image Analysis and Recognition*, pages 417–431, 2020. 2, 3, 5

[2] Daniel Barath and Levente Hajder. Efficient recovery of essential matrix from two affine correspondences. *IEEE Transactions on Image Processing*, 27(11):5328–5337, 2018. 4

[3] Daniel Barath, Michal Polic, Wolfgang Förstner, Torsten Sattler, Tomas Pajdla, and Zuzana Kukelova. Making affine correspondences work in camera geometry computation. In *European Conference on Computer Vision*, 2020. 4

[4] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 11621–11631, 2020. 5

[5] Banglei Guan, Ji Zhao, Zhang Li, Fang Sun, and Friedrich Fraundorfer. Minimal solutions for relative pose with a single affine correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1929–1938, 2020. 2

[6] Stewénius Henrik, Oskarsson Magnus, Kalle Aström, and David Nistér. Solutions to minimal generalized relative pose problems. In *Workshop on Omnidirectional Vision in conjunction with ICCV*, pages 1–8, 2005. 2, 3, 5

[7] Laurent Kneip and Paul Furgale. OpenGV: A unified and generalized approach to real-time calibrated geometric vision. In *IEEE International Conference on Robotics and Automation*, pages 12034–12043, 2014. 2

[8] Laurent Kneip and Hongdong Li. Efficient computation of relative pose for multi-camera systems. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 446–453, 2014. 2, 3, 5, 6

[9] Gim Hee Lee, Marc Pollefeys, and Friedrich Fraundorfer. Relative pose estimation for a multi-camera system with known vertical direction. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 540–547, 2014. 2, 3, 5

[10] Hongdong Li, Richard Hartley, and Jae-hak Kim. A linear approach to motion estimation using generalized camera models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 2, 3, 5

[11] Liu Liu, Hongdong Li, Yuchao Dai, and Quan Pan. Robust and efficient relative pose with a multi-camera system for autonomous driving in highly dynamic environments. *IEEE Transactions on Intelligent Transportation Systems*, 19(8):2432–2444, 2017. 2, 3, 5

[12] Jean-Michel Morel and Guoshen Yu. ASIFT: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009. 5

[13] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580, 2012. 5

[14] Chris Sweeney, John Flynn, and Matthew Turk. Solving for relative pose with a partially known rotation is a quadratic eigenvalue problem. In *IEEE International Conference on 3D Vision*, pages 483–490, 2014. 2, 3, 5, 6