TRPLP – Trifocal Relative Pose from Lines at Points

Supplementary Material

Ricardo Fabbri*
Rio de Janeiro State University

Timothy Duff
Georgia Tech

Hongyi Fan
Brown University

Margaret H. Regan
University of Notre Dame

David da Costa de Pinho
UENF – Brazil

Elias Tsigaridas
INRIA Paris

Charles W. Wampler
University of Notre Dame

Jonathan D. Hauenstein
University of Notre Dame

Peter J. Giblin
University of Liverpool

Benjamin Kimia
Brown University

Anton Leykin
Georgia Tech

Tomas Pajdla
CIIRC CTU in Prague†

A. Other formulations

Along with the two minor based formulations described in the main manuscript, two alternate formulations of both Chicago and Cleveland were explored, as outlined below. Experimental results using synthetic data for these two alternate formulations of Chicago and Cleveland, as well as the minor formulation of Cleveland are discussed below in Section C.

In addition, other "non-minor" formulations of Chicago were explored and implemented in MINUS for optimization and testing. Two important formulations are worth mentioning. The first is obtained by eliminating depths and other scalars from the original equations from Section 2.1 of the main paper, ending with an 11×11 system of equations only in the relative poses $\mathtt{R}_2, \mathtt{R}_3, \mathbf{t}_2, \mathbf{t}_3$ *modulo* global scale – embodying the calibrated trifocal tensor in different forms depending on the representation employed. The second reduction occurs after further eliminating translations to obtain a 6×6 system of equations in $\mathtt{R}_2, \mathtt{R}_3$, which can give better performance for the linear solves within Algorithm 1. The results of using these formulations and other more aggressive optimization strategies within MINUS are outlined below in Section D.

Alternate Chicago The first set of vector equations, (1) in the main paper, is associated to viewing points $p = 1, 2, 3$ from cameras $v = 1, 2, 3$. Eliminate $\mathbf{t}_v$ using one such equation for $p = 3$ and rearrange to:

$$\alpha_{pv}\mathbf{x}_{pv} - \alpha_{3v}\mathbf{x}_{3v} = \mathtt{R}_v(\alpha_{p1}\mathbf{x}_{p1} - \alpha_{31}\mathbf{x}_{31}), \qquad (1)$$

for $v = 2, 3$ and $p = 1, 2, 3$. The second set of vector equations used by this formulation is associated to viewing tangents from cameras $v = 1, 2, 3$, which is (5) in the main paper. Together, (1) above and (5) of the main paper are a set of 24 scalar equations with the following unknowns:

$$(\mathtt{R}_v, \mathbf{t}_v),\, v = 2, 3; \quad \alpha_{pv}, v = 1, 2, 3,\ p = 1, 2, 3;$$
$$(\epsilon_{pv}, \mu_{pv}), v = 1, 2, 3,\ p = 1, 2,$$

which are used with our additional Bertini solver in regards to the non-minor (*i.e.*, without using determinantal "visible lines" formulation of Section 3.1 of the main paper) Chicago formulation and experimentation.

Alternate Cleveland The three labeled points are the same, therefore (1) still applies. With the description in the main manuscript, for the free 3D line L, we let $(\mathbf{p}_v, \mathbf{q}_v)$ be two distinct points in three views. The back-projection of the image line is a plane whose equation in local coordinates is given in terms of a vector $\mathbf{n}_v$ normal to the plane:

$$\mathbf{n}_v^\top \mathbf{x} = 0, \text{ where } \mathbf{n}_v = \mathbf{p}_v \times \mathbf{q}_v.$$

Point $\mathbf{P}$ chosen as $\mathbf{P} = \alpha_q \mathbf{p}_1$, must lie in the back-projection planes in the other two views, giving

$$\mathbf{n}_v^\top(\mathtt{R}_v \alpha_q \mathbf{p}_1 + \mathbf{t}_v) = 0, \qquad (2)$$

*Contact: rfabbri@gmail.com, †Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague. RF is supported by UERJ Prociência and FAPERJ Jovem Cientista do Nosso Estado E-26/201.557/2014. TD and AL are supported by NSF DMS-1151297. JDH and MHR are supported by NSF CCF-1812746, with additional support for JDH from ONR N00014-16-1-2722 and for MHR from Schmitt Leadership Fellowship in Science and Engineering. BK and HF are supported by the NSF grant IIS-1910530. TP is supported by the EU Regional Development Fund IMPACT CZ.02.1.01/0.0/0.0/15 003/0000468 and EU H2020 project ARtwin 856994. This work was initiated while most authors were in residence at Brown University's Institute for Computational and Experimental Research in Mathematics – ICERM, in Providence, RI, during the Fall 2018 and Spring 2019 semesters (NSF DMS-1439786 and the Simons Foundation grant 507536).

for views 2 and 3. Eliminating the translations and rearranging we have:

$$\alpha_{3v}\mathbf{n}_v^\top \mathbf{x}_{3v} = \mathbf{n}_v^\top \mathtt{R}_v(\alpha_{31}\mathbf{x}_{31} - \alpha_q \mathbf{p}_1). \tag{3}$$

In addition, L must lie in all three back-projection planes, thus

$$\mathbf{n}_1^\top \mathbf{v} = 0, \quad \mathbf{n}_2^\top \mathtt{R}_2 \mathbf{v} = 0, \quad \mathbf{n}_3^\top \mathtt{R}_3 \mathbf{v} = 0. \tag{4}$$

The additional solver referenced below for this non-minor Cleveland formulation is defined by the polynomial system of (1), (3), and (4) with the following unknowns:

$$\mathtt{R}_v,\ \ v = 2, 3;\ \ \alpha_q;\ \ \mathbf{v};$$
$$\alpha_{pv},\ \ p = 1, 2, 3,\ \ v = 1, 2, 3.$$

Of course, we note that the above equations can partially be represented as determinants equal to zero; by non-minor we simply mean it is not focused on minors, but that they are a by-product of another type of geometrical reasoning.

B. Clarifying the proof of degrees

In the main paper, a proof regarding the number of 312 degrees and 216 for Chicago and Cleveland, respectively, was provided focusing on numerical arguments. These arguments are mathematically sound due to guarantees on the behavior of polynomial systems for these numerical methods given our assumptions listed within Section 2.2 of the main manuscript. In our main manuscript we also sketched how the proof would proceed by means of symbolic techniques. We now provide details on such a procedure, which is standard practice [2, 3].

To obtain the degree of the system, it is enough to give random values to all symbolic parameters (or coefficients), and then compute the degree of the resulting (specialized) system. This can be performed over $\mathbb{Q}$, as briefly described in the paper, or it may be more feasible to carry out computations modulo p, for a suitable prime number p. By making sure that the random values of the parameters are generic enough to be a representative of the general ones, and that the prime that we use is not a bad prime (for example that the modulo p operation does not kill terms of the polynomials), the computation of the degree is as mathematically sound as an analytic-geometric proof by hand (which would be very hard for this problem size).

Once we compute, over $\mathbb{Q}$, a lexicographical Gröbner basis, its last polynomial is a univariate polynomial of degree D, which is the problem degree. For Chicago, $D = 312$ is obtained, and for cleveland $D = 216$. Let the single variable of this last univariate polynomial be x. By solving this polynomial by usual means, one backsubstitutes x and thus finds a solution for the system. The procedure over the rationals is time consuming (several hours to days), so as a solver, this generic symbolic method as such is not useful in practice beyond proofs and other analysis.

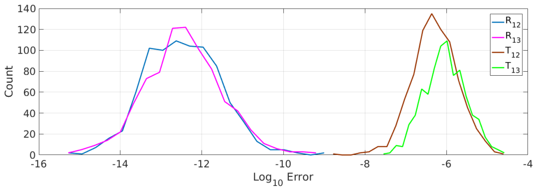

Figure 1. Errors of computed parameters with respect to the ground truth are small showing that the solver is numerically stable for the minor formulation of Cleveland.

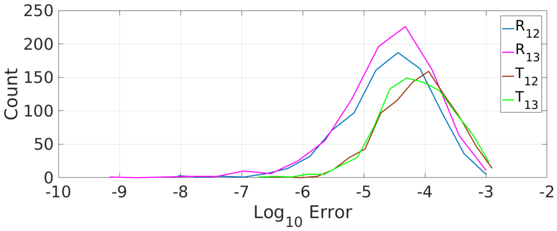

Figure 2. Errors of computed parameters with respect to the ground truth are small showing that the solver is numerically stable for the alternate formulation of Chicago.

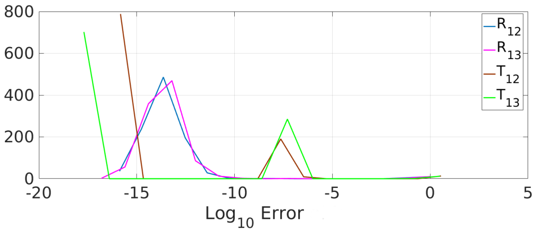

Figure 3. Errors of computed parameters with respect to the ground truth are small showing that the solver is numerically stable for the alternate formulation of Cleveland.

C. Additional Synthetic Experiments

Synthetic experiments were completed for the minor formulation of Cleveland discussed in Section 3.1 in the main manuscript, as well as the other formulations outlined above in Section A. These experiments are equivalent to those outlined in Section 4 in the main manuscript under the heading synthetic experiments.

For the three separate formulations, minor Cleveland and alternate Chicago and Cleveland, it was found that pose estimation errors are negligible as shown in Figures 1, 2, and 3, respectively.

The next set of experiments show the behavior when the correspondences are correct, but noisy. Using the same process as described in detail in Section 4 of the main paper. The result of the minor formulation of Cleveland and alternate formulations of Chicago and Cleveland are shown in Figures 4, 5, and 6, respectively. For each formulation, the median of the translation and rotation error are low, but due to the relatively high failure rate of these three formulations, there are several failures that effect the data. However, these failure cases can be detected and resolved by thresholding the maximum inlier ratio in RANSAC. In addition, the average reprojection error with respect to the ground truth point correspondences, also in Figures 4, 5, and 6, shows that for most of the test cases we have a stable and reasonable reprojection error. Again, the cases with large reprojection error can be ignored by thresholding maximum inlier ratio in RANSAC.

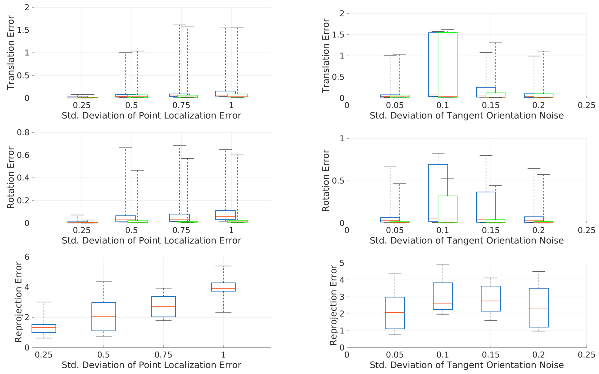

Figure 4. Distribution of trifocal pose error for the minor formulation of Cleveland in the form of translational and rotational error between cameras 1 and 2 (blue) and cameras 1 and 3 (green), as well as the reprojection error, plotted against the level of feature localization noise (left) and orientation noise (right).

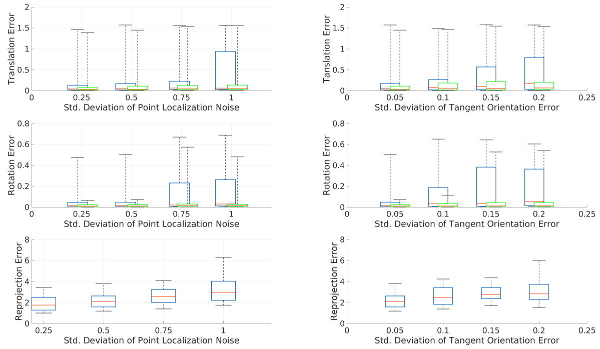

Figure 5. Distribution of trifocal pose error for the alternate formulation of Chicago in the form of translational and rotational error between cameras 1 and 2 (blue) and cameras 1 and 3 (green), as well as the reprojection error, plotted against the level of feature localization noise (left) and orientation noise (right).

These results on synthetic data sets, paired with the computational efficiency of the solvers for various formulations, highlight the efficacy of the homotopy continuation methods and their ability to solve these trifocal problems in a competitive nature.

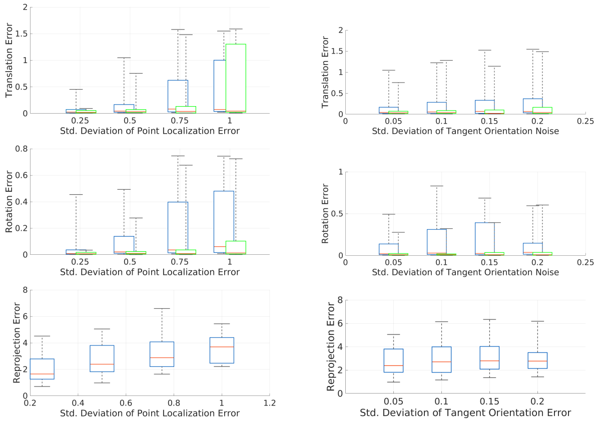

Figure 6. Distribution of trifocal pose error for the alternate formulation of Cleveland in the form of translational and rotational error between cameras 1 and 2 (blue) and cameras 1 and 3 (green), as well as the reprojection error, plotted against the level of feature localization noise (left) and orientation noise (right).

Computational efficiency For the minor formulation of Cleveland, each run of our more general purpose solver using Bertini takes about 8.97 seconds on average with a failure rate of about 17.9%. For the alternate formulation of Chicago, each run takes about 19.69 seconds on average with a failure rate of 12.4% and for the alternate formulation of Cleveland, each run takes about 11.46 seconds on average with a failure rate of 3.2%. All of these tests were done on an AMD Opteron 6378 2.4 GHz processor using 12 threads.

Implementation The minor formulation of Cleveland and the alternate formulations of both Chicago and Cleveland were implemented within a more general purpose solver involving Bertini, which utilizes the parameter homotopy method described in Algorithm 1 in the main paper. They were not implemented in MINUS since this trivial operation would only change speed, and Chicago was the focus of the paper exemplifying this process of transcribing a solver to an optimized C++ version. There are improvements that can be made to precision and error analysis using adaptive multiprecision path tracking [1], yet this comes at the expense of speed. In addition, other settings within Bertini can be employed, at the expense of reliability and causing a potential increase in failure rate. There is potential for other optimization, but that has not been explored here.

D. Tuning of the main solver MINUS

As stated in the main manuscript, MINUS can run at the milisecond scale with the 14×14 formulation, at the cost of increased failure rate. We have observed that in practice such failure rate might not be important for RANSAC, and can be controlled by performing tests to the input points and lines to rule out near-coplanar or near-collinear configurations, which make the system close to underconstrained.

In optimizing MINUS, one can constrain the number of iterations per solution path, which would yield the most effective speedup. In fact, in carrying extensive experiments with the synthetic data reported in the paper, after 10000 random solves, the maximum number of iterations for paths leading to ground-truth solutions was 1119, and for the other paths this was 253787. The discrepancy is very large. Given that the solve is about 1 microseconds per iteration, this leads to very good prospects.

Another important study is regarding the conditioning of the linearized homotopies (Jacobian matrices) as one varies the formulation. Yet another very promising idea is to vary the start system. Presently, the start system is precomputed from random parameters for the equations using monodromy. The start system can instead be sampled from the view-sphere for our synthetic data, and the closest camera could be selected matching a similar configuration of point-tangents.

In practice, we observed the following effective optimizations to the current code. First, the most important parameter to vary is the maximum number of correction steps (see Algorithm 1 in the paper); a maximum of 3 is the safe default. Increasing it to anywhere from 4 to 7 gets the runtime down to 464ms. Another is the corrector tolerance: by increasing it 10000x, MINUS will run in 200ms. This parameter can be seen by inspecting our published source code. It affects how many correction iterations are performed. The error rate for these extreme cases of 200ms can be as high as 50%. However, we believe that by performing less strict tests on reprojection error, the failure rate can be significantly lowered.

The next step for optimizing MINUS is to determine how to prune paths that take a significant length of time to track. Acceleration using SIMD has been studied, but by analyzing assembly output, most operations (complex vector multiplications and additions) are currently auto vectorized. Our tests point to the fact that reducing the representation to, say 6×6, would provide strong improvements if ill-conditioning is taken care of. They also indicate that this would improve linear-algebra solves, evaluator lengths, and instruction cache misses. These implementations are currently ongoing.

E. Creation of Mug Dataset

In this work, we created a feature-less mug dataset inspired by Nurutdinova *et al*. [5]. The reason we didn't use the original dataset from [5] is because the occlusion between mug and calibration pattern makes removing the calibration pattern cumbersome. Ten camera poses are set to capture 10 images where the calibration pattern is not occluded. After capturing images, the MATLAB calibration toolbox was used to generate the ground-truth camera pose with manually marked correspondence points on the checkerboard. Once the ground-truth was extracted, the checkerboard area was marked and deleted manually, followed by image inpainting to fill the gap in the image, as shown in Figure 7.

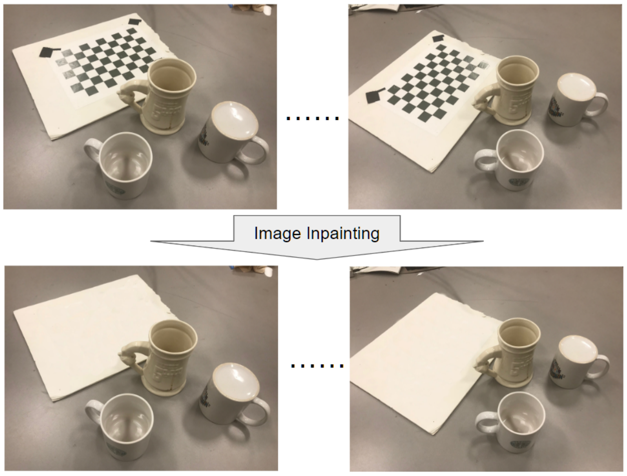

Figure 7. In construction of the mug dataset, a calibration pattern was first placed to generate the ground truth configuration of cameras. Next, the calibration pattern was removed using image inpainting for testing.

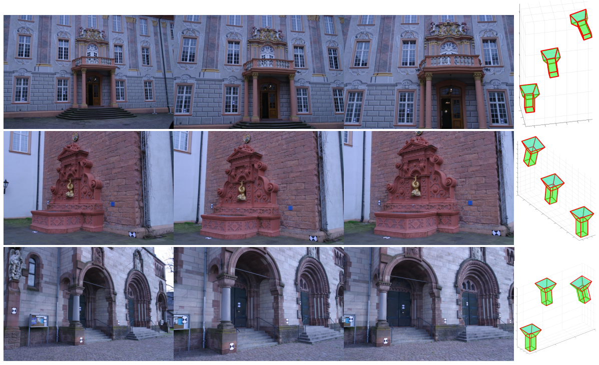

Figure 8. Trifocal relative pose estimation for additional cases from the EPFL dataset. For each row, image triplets samples are shown. The estimation results are shown on the right. Ground truth poses are in solid green and estimated poses are in red.

F. Additional Real Experiments

More real experiments that were not shown in the main paper are shown in this section. First, for texture-rich images, more cases from the EPFL dataset are shown in Figure 8 for the Chicago problem. Second, we include a quantitative comparison to other trifocal methods reported in [4] for the Chicago problem, as shown in Table 1. As in [4], we compare using the two datasets Fountain P-11 and Herz-Jesu-P8, illustrating that our method is comparable to or better than other trifocal methods.

Methods	R error (deg)	T error (deg)
TFT-L	0.292	0.638
TFT-R	0.257	0.534
TFT-N	0.337	0.548
TFT-FP	0.283	0.618
TFT-PH	0.269	0.537
MINUS (Ours)	**0.137**	**0.673**

Table 1. The pose error comparison between our method with other trifocal methods. Observe that our method has better rotation error and comparable translation error.

References

[1] D. Bates, J. Hauenstein, A. Sommese, and C. Wampler. Adaptive multiprecision path tracking. *SIAM Journal on Numerical Analysis*, 46(2):722–746, 2008. 3

[2] David Cox, John Little, and Donald O'Shea. *Using Algebraic Geometry*. Springer, 1998. 2

[3] David A. Cox, John Little, and Donald O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 2015. 2

[4] Laura Julià and Pascal Monasse. A critical review of the trifocal tensor estimation. In *The Eighth Pacific-Rim Symposium on Image and Video Technology – PSIVT'17*, pages 337–349, Wuhan, China, 2017. Springer. 4

[5] Irina Nurutdinova and Andrew Fitzgibbon. Towards pointless structure from motion: 3d reconstruction and camera parameters from general 3d curves. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2363–2371, 2015. 4