Triangulation: Why Optimize?

Seong Hun Lee
seonghunlee@unizar.es
Javier Civera
jcivera@unizar.es

I3A
University of Zaragoza
Zaragoza, Spain

Abstract

For decades, it has been widely accepted that the gold standard for two-view triangulation is to minimize the cost based on reprojection errors. In this work, we challenge this idea. We propose a novel alternative to the classic midpoint method that leads to significantly lower 2D errors and parallax errors. It provides a numerically stable closed-form solution based solely on a pair of backprojected rays. Since our solution is rotationally invariant, it can also be applied for fisheye and omnidirectional cameras. We show that for small parallax angles, our method outperforms the state-of-the-art in terms of combined 2D, 3D and parallax accuracy, while achieving comparable speed.

1 Introduction

Locating the 3D point given its projections in multiple views is called triangulation. This classic yet fundamental problem in computer vision has immediate relevance to many applications, including visual odometry [8], simultaneous localization and mapping (SLAM) [23] and structure-from-motion (SfM) [28]. As such, achieving fast and accurate triangulation has been a goal of many research endeavors in the past decades.

For two views of known calibration and pose, the problem could be solved ideally if one finds the intersection of two backprojected rays corresponding to the same point. However, the two rays are most likely skew due to noisy measurements and inaccurate camera model. Since it is not obvious how to estimate the 3D position of the point from two skew rays, different methods have been proposed. Mainly, they can be classified into three types: (1) midpoint methods [3, 4, 30] that find the (weighted) midpoint of the common perpendicular between the two rays, (2) linear least squares methods [15], and (3) optimal methods that "minimally" correct the two rays to make them intersect [15, 16, 19]. Note that all these three types of methods produce solutions that minimize some cost function; the (weighted) midpoint minimizes the (weighted) sum of squared distances to each ray, linear least squares methods minimize the algebraic errors, and optimal methods minimize a cost function based on either *image* reprojection errors [15, 16, 20] or *angular* reprojection errors [19, 26]. The most common cost functions are the L_1 norm (sum of magnitude), L_2 norm (sum of squares) and L_∞ norm (maximum) of the reprojection errors.

In this work, we suggest a different approach. Instead of minimizing geometric or algebraic errors, we find a midpoint between a certain pair of points on each ray. Like the classic midpoint method, our method takes the two rays as input. Therefore, it is invariant to changes of camera rotation and applicable for perspective, fisheye and omnidirectional cameras. Unlike the classic method, however, the two points on each ray are not necessarily on the common perpendicular. We will see that our midpoint method bears a striking similarity

to the classic method in the formulation, and yet it offers a significant performance gain in 2D and parallax accuracy. Compared to the optimal methods, our method yields much lower 3D errors at low parallax and similar 2D errors to those of L_2 and L_∞ optimal methods. This motivates the question: In two-view triangulation, why optimize if there is a better way?

The main contributions of this paper are the following:

- We propose a novel method belonging to a group called the generalized weighted midpoint (GWM) method. We show that our method outperforms existing ones (including the classic midpoint method and the state-of-the-art optimal methods) in terms of combined 2D, 3D and parallax accuracy.
- Additionally, we propose a test of the adequacy (similar to the cheirality check) that identifies unreliable results and a weighting scheme that enhances 2D accuracy.
- We perform an extensive evaluation and analysis of various methods, revealing an intricate link between 3D accuracy and parallax estimation. This will provide an intuitive explanation of why our midpoint method performs better than the others.

2 Related Work

One of the earliest works that addressed the two-view triangulation problem is [21] where the depth of a 3D point is estimated using simple algebra. For robustness, later works mostly adopted geometric approaches, such as the midpoint method [3, 4] and the minimization of the epipolar distance [9, 10] or reprojection error [14]. Among those, the last approach has become the *de facto* standard in computer vision [13].

Optimal methods refer to those triangulation methods that minimize the cost based on reprojection errors. Assuming that the image measurements are independently perturbed by the noise in the same distribution of certain types, the optimal methods find the maximum likelihood (ML) solution. For Gaussian and Laplacian distribution, the ML solution is to minimize the L_2 norm or L_1 norm of the reprojection errors, respectively [18]. This can be found in closed form by solving a polynomial of degree six or eight [15]. Alternatively, the L_2 solution can be obtained using iterative correction methods [16, 20]. While these iterative methods do not guarantee global optimality, they were shown to be faster and more stable. For a uniform distribution, minimizing the L_∞ norm leads to the ML estimate for the lower bound of the noise [11], and the solution is obtained by solving a quartic polynomial [24]. Unlike the L_1 and L_2 cost, the L_∞ cost has a simple shape with a single minimum, but it is relatively more sensitive to noise and outliers [12].

These optimal methods assume that the image measurement errors follow certain distributions. However, this assumption is neither justified nor likely [11]. An equally (if not more) justified alternative is to assume that the noise model applies to the bearing measurements instead of the image. For fisheye or omnidirectional cameras, the angular reprojection errors are more suitable than the image reprojection errors [22, 26]. Also, formulating the triangulation problem in terms of angular errors leads to much simpler ML solutions [19, 26].

Although the existing optimal methods can provide relatively good 3D results in many cases [15, 19], none of them are theoretically optimal in terms of 3D errors. In fact, the discrepancy between 2D optimality and 3D accuracy has already been reported by Hartley and Sturm [15]. They found that in Euclidean reconstruction, the midpoint and the linear least squares method achieve higher 3D accuracy than the L_1 or L_2 optimal methods, despite consistently (and sometimes significantly) larger 2D errors. In this work, we provide additional insights on this matter. Furthermore, we show that a simple modification to the midpoint method can substantially reduce the 2D errors while maintaining 3D accuracy.

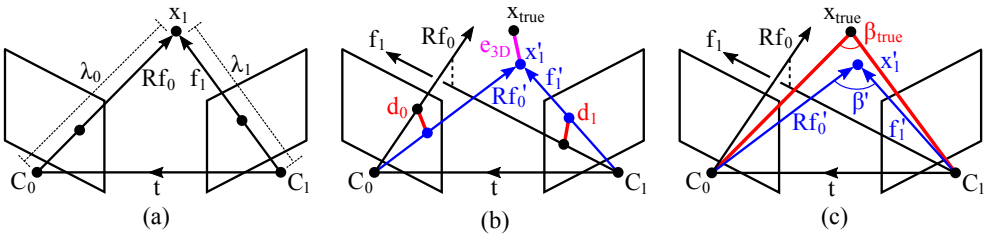

Figure 1: **(a)** Epipolar geometry when the two backprojected rays intersect. All vectors are in the coordinate system of C_1. **(b)** 3D error (e_{3D}) measures the Euclidean distance between the estimate ($\mathbf{x}'_1$) and the true position of the point ($\mathbf{x}_{\text{true}}$), while 2D error (d_0, d_1) measures the offset between the observation and the reprojection of the estimated point in each frame. **(c)** After the triangulation, one can estimate the parallax (β') from the corrected rays.

3 Preliminaries

Throughout the paper, we use bold letters for vectors and matrices, and light letters for scalars. The Euclidean norm of a vector $\mathbf{v}$ is denoted by $\|\mathbf{v}\|$, and the unit vector by $\widehat{\mathbf{v}} = \mathbf{v}/\|\mathbf{v}\|$. The angle between two lines $\mathbf{L}_0$ and $\mathbf{L}_1$ is denoted by $\angle(\mathbf{L}_0,\ \mathbf{L}_1) \in [0, \pi/2]$.

Consider a 3D point observed by two cameras C_0 and C_1. We define $\mathbf{x}_0 = [x_0, y_0, z_0]^\intercal$ and $\mathbf{x}_1 = [x_1, y_1, z_1]^\intercal$ as the unknown 3D coordinates of the point in the camera reference frame C_0 and C_1, respectively. Let $\mathbf{R}$ and $\mathbf{t}$ be the known rotation and translation between the two cameras, such that $\mathbf{x}_1 = \mathbf{R}\mathbf{x}_0 + \mathbf{t}$. Assuming that the camera calibration matrix $\mathbf{K}$ is known, the normalized image coordinates $\mathbf{f}_0 = [x_0/z_0, y_0/z_0, 1]^\intercal$ and $\mathbf{f}_1 = [x_1/z_1, y_1/z_1, 1]^\intercal$ can be obtained by $\mathbf{f}_0 = \mathbf{K}^{-1}\mathbf{u}_0$ and $\mathbf{f}_1 = \mathbf{K}^{-1}\mathbf{u}_1$, where $\mathbf{u}_0 = (u_0, v_0, 1)^\intercal$ and $\mathbf{u}_1 = (u_1, v_1, 1)^\intercal$ are the homogeneous pixel coordinates of the point observation in each frame.

In the ideal situation (Fig. 1a), the two backprojected rays intersect, satisfying the epipolar constraint [21], i.e., $\mathbf{f}_1 \cdot (\mathbf{t} \times \mathbf{R}\mathbf{f}_0) = 0$. Then, the intersection is given by $\mathbf{x}_1 = \lambda_0 \mathbf{R}\widehat{\mathbf{f}}_0 + \mathbf{t}$ or $\mathbf{x}_1 = \lambda_1 \widehat{\mathbf{f}}_1$ for some scalar depth λ_0 and λ_1. However, this rarely happens due to inaccuracies in the image measurements and the camera model. Inferring a 3D point from two skew rays requires a nontrivial method.

Once the estimate of the 3D point ($\mathbf{x}'_1$) is obtained using some triangulation method, its accuracy can be evaluated in several ways. One way is to compute the 3D error, i.e., $e_{3D} = \|\mathbf{x}'_1 - \mathbf{x}_{\text{true}}\|$. Another way is to compute the 2D error (aka the reprojection error), i.e.,

$$d_i = \|\mathbf{K}\left(\mathbf{f}_i - \mathbf{f}_i'\right)\| = \left\|\mathbf{K}\left(\mathbf{f}_i - \left([0\ 0\ 1]\ \mathbf{x}'_i\right)^{-1}\mathbf{x}'_i\right)\right\| \quad \text{for} \quad i = 0, 1, \tag{1}$$

where $\mathbf{x}'_0 = \mathbf{R}^\intercal(\mathbf{x}'_1 - \mathbf{t})$. These two errors are illustrated in Fig. 1b. Note that the 2D error represents the deviation from the measurement, whereas the 3D error represents the deviation from the ground truth. Also, unlike the 3D error, the 2D error of a 3D point can be evaluated in different norms, e.g., L_1 norm $(d_0 + d_1)$, L_2 norm $\left(\sqrt{d_0^2 + d_1^2}\right)$ and L_∞ norm $(\max(d_0, d_1))$. Besides 2D and 3D accuracy, we can also evaluate the accuracy of the resulting parallax angle (see Fig. 1c). The parallax error is defined as follows:

$$e_\beta = |\beta_{\text{true}} - \beta'| = |\angle(\mathbf{x}_{\text{true}}, \mathbf{x}_{\text{true}} - \mathbf{t}) - \angle\left(\mathbf{x}'_1, \mathbf{x}'_1 - \mathbf{t}\right)|. \tag{2}$$

We define the "raw parallax" as the angle between the original backprojected rays:

$$\beta_{\text{raw}} = \angle(\mathbf{R}\mathbf{f}_0, \mathbf{f}_1). \tag{3}$$

This gives a rough estimate of the parallax angle independently of the translation and the triangulation method.

4 Proposed method

4.1 Generalized Weighted Midpoint (GWM) Method

A GWM method consists of three steps: (1) Given two backprojected rays corresponding to the same point, estimate the depth along each ray (λ_0, λ_1) using some method. (2) Compute the 3D point on each ray at depth λ_0 and λ_1, i.e., $\mathbf{t}+\lambda_0\mathbf{R}\widehat{\mathbf{f}}_0$ and $\lambda_1\widehat{\mathbf{f}}_1$ in C_1. (3) Obtain the final estimate of the 3D point by computing their weighted average.

The classic midpoint method [3, 4, 15] is one such type of method where the two points on each ray are the closest pair of points with the equal weight. Fig. 2 shows another possible example of the generalized weighted midpoint.

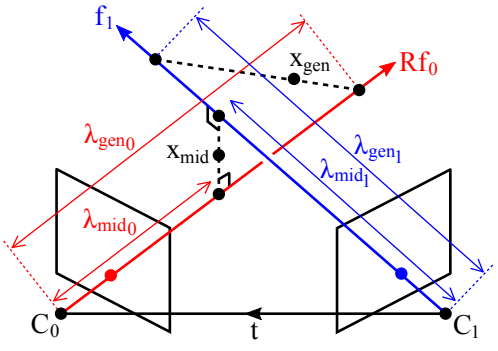

Figure 2: The classic midpoint and another example of the generalized weighted midpoint.

4.2 Alternative Midpoint Method

We propose an alternative midpoint method that belongs to the generalized weighted midpoint method. First, consider the case where the two backprojected rays happen to intersect (see Fig. 1a). In this case, the most sensible solution is the point of intersection, and the corresponding depths along the rays can be obtained using the sine rule:

$$\lambda_0 = \frac{\sin(\angle(\mathbf{f}_1,\mathbf{t}))}{\sin(\angle(\mathbf{R}\mathbf{f}_0,\mathbf{f}_1,))}\|\mathbf{t}\| = \frac{\|\widehat{\mathbf{f}}_1\times\mathbf{t}\|}{\|\mathbf{R}\widehat{\mathbf{f}}_0\times\widehat{\mathbf{f}}_1\|}, \quad \lambda_1 = \frac{\sin(\angle(\mathbf{R}\mathbf{f}_0,\mathbf{t}))}{\sin(\angle(\mathbf{R}\mathbf{f}_0,\mathbf{f}_1,))}\|\mathbf{t}\| = \frac{\|\mathbf{R}\widehat{\mathbf{f}}_0\times\mathbf{t}\|}{\|\mathbf{R}\widehat{\mathbf{f}}_0\times\widehat{\mathbf{f}}_1\|}. \quad (4)$$

We use this formula to estimate the depths even when the two rays are skew. Computing the 3D points on each ray at depth λ_0 and λ_1, respectively, we get

$$\mathbf{t}+\lambda_0\mathbf{R}\widehat{\mathbf{f}}_0 = \mathbf{t}+\frac{\|\mathbf{f}_1\times\mathbf{t}\|}{\|\mathbf{R}\mathbf{f}_0\times\mathbf{f}_1\|}\mathbf{R}\mathbf{f}_0 \quad \text{and} \quad \lambda_1\widehat{\mathbf{f}}_1 = \frac{\|\mathbf{R}\mathbf{f}_0\times\mathbf{t}\|}{\|\mathbf{R}\mathbf{f}_0\times\mathbf{f}_1\|}\mathbf{f}_1, \quad (5)$$

Taking the midpoint between these two points leads to

$$\mathbf{x}'_1 = \frac{1}{2}\left(\mathbf{t}+\frac{\|\mathbf{f}_1\times\mathbf{t}\|}{\|\mathbf{R}\mathbf{f}_0\times\mathbf{f}_1\|}\mathbf{R}\mathbf{f}_0+\frac{\|\mathbf{R}\mathbf{f}_0\times\mathbf{t}\|}{\|\mathbf{R}\mathbf{f}_0\times\mathbf{f}_1\|}\mathbf{f}_1\right). \quad (6)$$

Note that letting $\mathbf{p} = \mathbf{R}\widehat{\mathbf{f}}_0\times\widehat{\mathbf{f}}_1$, $\mathbf{q} = \mathbf{R}\widehat{\mathbf{f}}_0\times\mathbf{t}$ and $\mathbf{r} = \widehat{\mathbf{f}}_1\times\mathbf{t}$ allows us to write (4) as

$$\lambda_0 = \frac{\|\mathbf{r}\|}{\|\mathbf{p}\|}, \quad \lambda_1 = \frac{\|\mathbf{q}\|}{\|\mathbf{p}\|}. \quad (7)$$

Interestingly, these are in a similar form to the depths given by the classic midpoint method[1]:

$$\lambda_{\text{mid}0} = \frac{\widehat{\mathbf{p}}\cdot\mathbf{r}}{\|\mathbf{p}\|}, \quad \lambda_{\text{mid}1} = \frac{\widehat{\mathbf{p}}\cdot\mathbf{q}}{\|\mathbf{p}\|}. \quad (8)$$

[1] We provide the derivation in the Appendix.

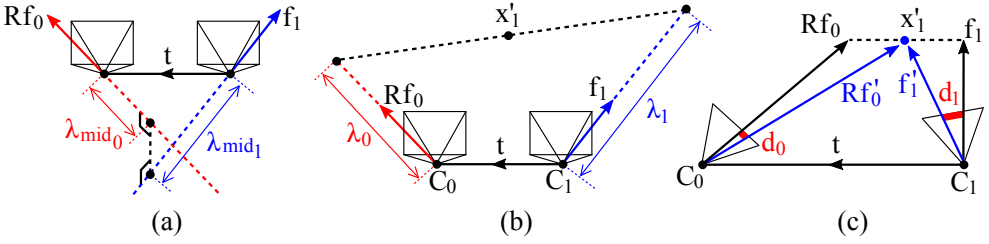

Figure 3: **(a)** A scenario where the classic midpoint method gives negative depths. The cheirality check will identify and remove this point. **(b)** In the same situation, our midpoint method will give positive depths. The triangulation result satisfies the cheirality constraint, but it is most likely inaccurate. **(c)** For unweighted midpoint methods, the frame with a smaller depth tends to get a larger reprojection error. In this example, $\lambda_1 < \lambda_0$ and $d_1 > d_0$.

The difference is in the numerator; (7) has the magnitude of $\mathbf{r}$ and $\mathbf{q}$, whereas (8) has their projection onto $\mathbf{p}$. As a result, we always get $\lambda_0 \geq \lambda_{\text{mid}0}$ and $\lambda_1 \geq \lambda_{\text{mid}1}$. In most cases, this means that our midpoint will be located farther than the classic midpoint. Fig. 2 depicts one such example. When we estimate that a point is located farther away from the cameras, it usually results in a lower estimate of the parallax angle, as will be shown in Section 5.

4.3 Cheirality and Test of Adequacy

We say that the cheirality constraint [13] is violated when a triangulated point has negative depth(s). This can happen for many reasons, such as spurious data association or the noise in the image point near the epipole. Normally, it does not pose a serious problem because we can easily check the cheirality for each point and discard the bad ones. For the classic midpoint method, this can be done by checking the signs of the depths given by (8). For our midpoint method, however, this is not possible because the depths given by (7) are always positive. Fig 3a and 3b illustrate the difference between the two methods. In our method, the depths alone cannot tell us whether or not the triangulation result is reliable.

Therefore, we use a different method to test the adequacy; we discard the point correspondence if changing the sign of at least one depth to negative leads to a smaller distance between the two points on each ray, i.e.,

$$\|\mathbf{t}+\lambda_0\widehat{\mathbf{R}\mathbf{f}}_0-\lambda_1\widehat{\mathbf{f}}_1\|^2 \geq \min\Big(\|\mathbf{t}+\lambda_0\widehat{\mathbf{R}\mathbf{f}}_0+\lambda_1\widehat{\mathbf{f}}_1\|^2,\|\mathbf{t}-\lambda_0\widehat{\mathbf{R}\mathbf{f}}_0-\lambda_1\widehat{\mathbf{f}}_1\|^2,\|\mathbf{t}-\lambda_0\widehat{\mathbf{R}\mathbf{f}}_0+\lambda_1\widehat{\mathbf{f}}_1\|^2\Big) \quad (9)$$

For the classic midpoint method, letting $\lambda_0 = |\lambda_{\text{mid}0}|$ and $\lambda_1 = |\lambda_{\text{mid}1}|$ gives effectively the same result as the cheirality check. For example, (9) holds in Fig 3a because the two points are closest when $\lambda_0 = -|\lambda_{\text{mid}0}|$ and $\lambda_1 = -|\lambda_{\text{mid}1}|$.

4.4 Inverse Depth Weighted Midpoint

The unweighted midpoint given by (6) often entails disproportionate reprojection errors in the two images. Fig. 3c shows an example. Notice that the ray with a smaller depth tends to yield a larger reprojection error. To compensate this imbalance, we propose to use the inverse depth [6] as a weight:

$$\mathbf{x}'_1 = \frac{\lambda_0^{-1}\left(\mathbf{t}+\lambda_0\widehat{\mathbf{R}\mathbf{f}}_0\right)+\lambda_1^{-1}\left(\lambda_1\widehat{\mathbf{f}}_1\right)}{\lambda_0^{-1}+\lambda_1^{-1}} \overset{(7)}{=} \frac{\|\mathbf{q}\|}{\|\mathbf{q}\|+\|\mathbf{r}\|}\left(\mathbf{t}+\frac{\|\mathbf{r}\|}{\|\mathbf{p}\|}\left(\widehat{\mathbf{R}\mathbf{f}}_0+\widehat{\mathbf{f}}_1\right)\right). \quad (10)$$

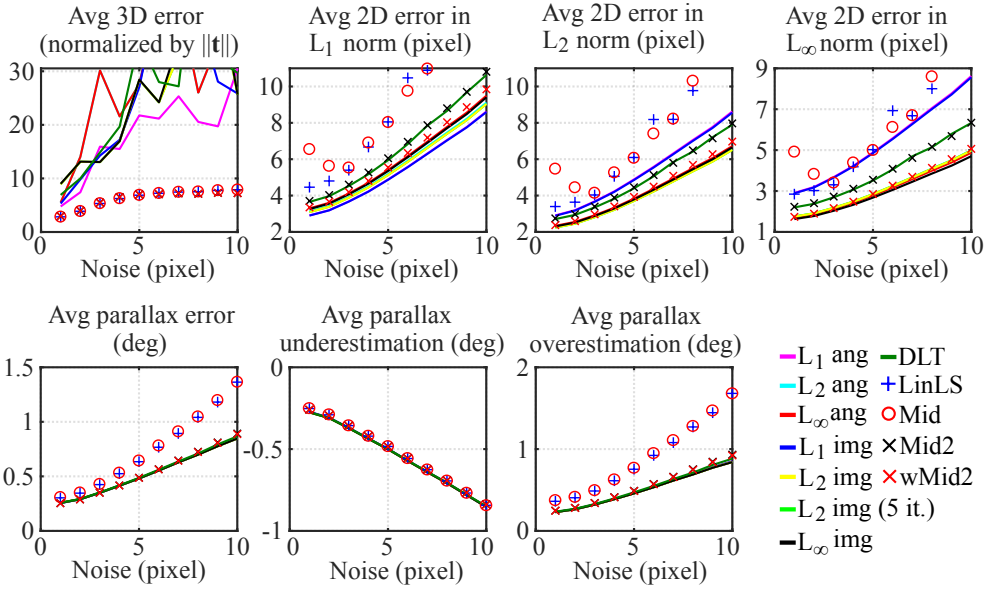

Figure 4: Triangulation accuracy over different noise levels in the image measurements. **3D results (top 1st column):** LinLS, Mid, Mid2 and wMid2 perform almost equally and all significantly better than the rest. Among the optimal methods, L_1 ang performs best. **2D results (top 2nd–4th column):** LinLS and Mid perform worst across all norms. Mid2 performs much better than those two, and wMid2 performs consistently better than Mid2. Expectedly, optimal methods perform best in their respective error criterion. However, the differences among them are smaller in L_1 norm than in the other two norms. **Parallax results (bottom row):** LinLS and Mid perform worst, and the rest almost equally. Looking at the under- and overestimation of the parallax separately, we notice that the low accuracy of LinLS and Mid is caused by their bias to overestimate the parallax on average.

5 Evaluation Results

We evaluate the following methods: Lee and Civera's L_1, L_2 and L_∞ optimal angular methods ('L_1 ang', 'L_2 ang', 'L_∞ ang') [19], Hartley and Sturm's L_1 and L_2 optimal methods ('L_1 img', 'L_2 img') and linear methods ('DLT', 'LinLS') [13, 15], Lindstrom's L_2 method with five iterations ('L_2 img (5 it.)') [20], Nister's L_∞ method ('L_∞ img') [24], the classic midpoint method ('Mid') [3, 4, 15], and our method without and with the weighting ('Mid2', 'wMid2'). The evaluation was performed on synthetic datasets generated as follows: A set of 8×8 point clouds of 5,000 points each are generated with a Gaussian radial distribution $\mathcal{N}(0,(d/4)^2)$ where d is the distance from the world origin. Each point cloud is centered at $[0,0,d]^\intercal$ for $d = 2^n$ with $n = -1,0,...,+6$, and their image projections are perturbed by Gaussian noise $\mathcal{N}(0,\sigma^2)$ for $\sigma = 1,2...,8$. The size and the focal length of the images are $1,024^2$ pixels and 512 pixel, respectively. We have four configurations for the camera poses: (1) 'orbital' - two cameras at $[\pm 0.5,0,0]^\intercal$ pointing at the point cloud center, (2) 'lateral' - two cameras at $[\pm 0.5,0,0]^\intercal$ pointing at $[0,0,\infty]^\intercal$, (3) 'forward' - two cameras at $[0,0,\pm 0.5]^\intercal$ pointing at the point cloud center, and (4) 'diagonal' - two cameras at $\pm[\sqrt{3}/6,\sqrt{3}/6,\sqrt{3}/6]^\intercal$ pointing at $[0,0,\infty]^\intercal$. The poses are slightly perturbed with uniform noise $\mathcal{U}(0,0.01)$. In total, the datasets provide over a million unique triangulation problems.

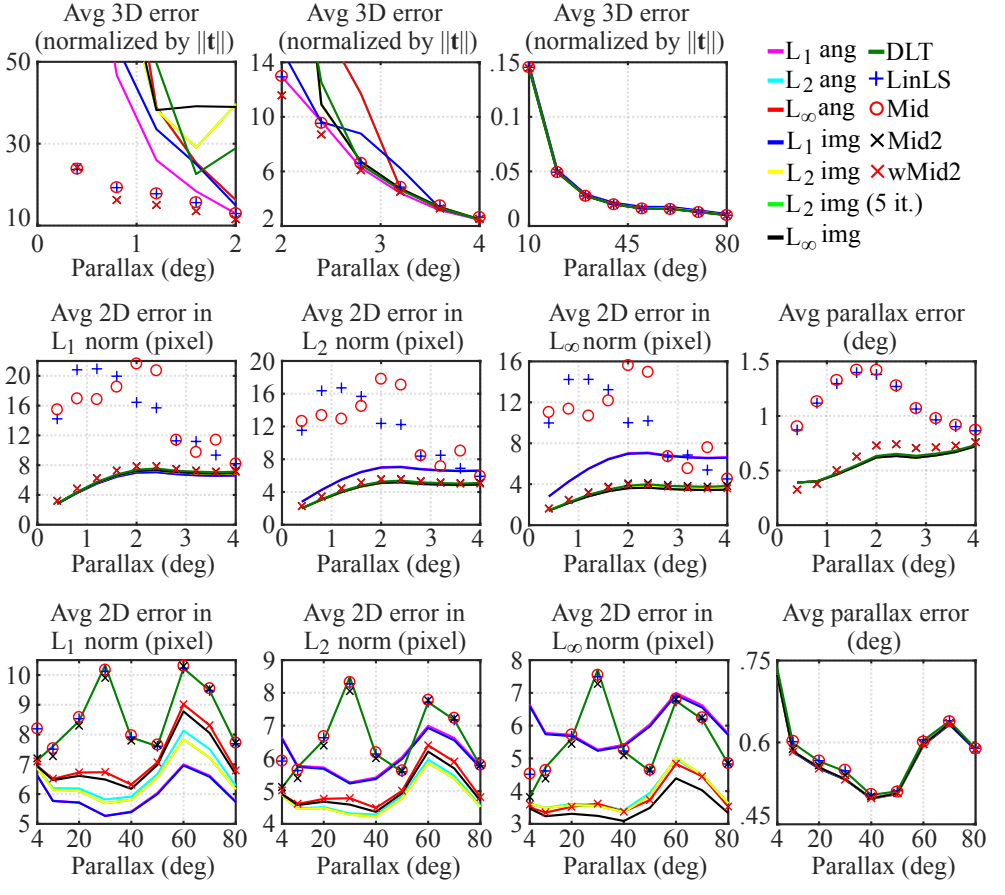

Figure 5: Triangulation accuracy over different parallax angles β_{raw} (3).
3D results (top row): For parallax under 2 deg, LinLS, Mid, Mid2 and wMid2 outperform the rest. For parallax between 2–4 deg, L_1 ang and the four aforementioned methods perform best. For parallax over 4 deg, all methods perform almost equally.
2D results for small parallax (mid 1st–3rd column): LinLS and Mid perform significantly worse than the rest in all norms. Aside from those two, L_1 methods perform much worse than the rest in L_2 and L_∞ norm, yet best in L_1 norm. The remaining methods perform similarly.
2D results for large parallax (bottom 1st–3rd column): In all norms, LinLS, Mid, Mid2 and DLT perform consistently worse than wMid2, L_2 and L_∞ methods. The latter perform similarly and much better than L_1 methods in L_2 and L_∞ norm, yet worse in L_1 norm.
Parallax results (last column): For low-parallax points, LinLS and Mid perform worst (see Fig. 6 for more details). For high-parallax points, all methods perform equally well.

We aggregate the results in Fig. 4 and 5. Our observations agree with [15] in that:

1. Generally, greater noise and lower parallax lead to larger 3D errors. All methods yield almost equally low 3D errors for high-parallax points (> 4 deg).
2. 2D and 3D errors are not well correlated. For example, LinLS and Mid perform best in 3D, but worst in 2D.

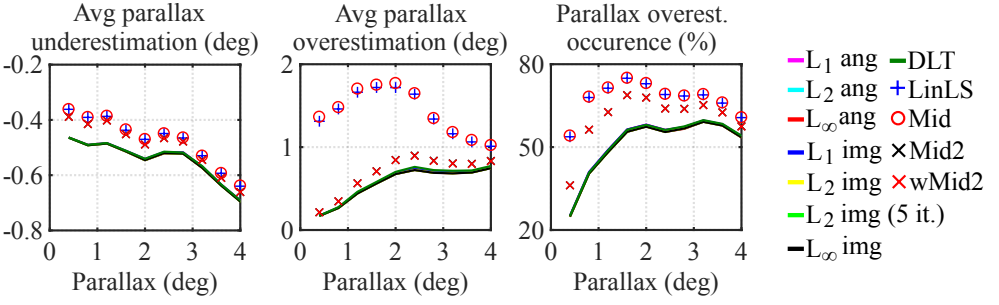

Figure 6: Parallax accuracy for low-parallax points. LinLS and Mid are relatively more biased to overestimate the small parallax angles, both in magnitude and frequency. The same goes for Mid2 and wMid2, but to a lesser extent.

Additionally, we report the following findings:

1. It is difficult to tell which method is the best in terms of 2D accuracy. For example, L_1 methods yield the lowest 2D errors in L_1 norm, but relatively larger errors in L_2 and L_∞ norm. It is not obvious which norm is more important. That said, some methods can still perform consistently better than others; wMid2, L_2 and L_∞ methods consistently outperform LinLS, Mid, Mid2 and DLT in all 2D error criteria.
2. As shown in Fig. 6, LinLS and Mid are clearly more biased to overestimate the small parallax angles (< 4 deg). This explains their relatively low 3D errors at low parallax. Fig. 7 illustrates a simplified example of this effect.
3. Our methods (Mid2 and wMid2) achieve the best overall accuracy in 3D + parallax.

The last finding can be solely attributed to the low-parallax points, for which our methods show similar 3D accuracy to that of Mid, but much better parallax accuracy. The latter can be explained by the fact that Mid2 always yields larger depths than Mid (as discussed in Section 4.2), which in effect lowers the estimated parallax angle on average (as shown in Fig. 6). Since Mid tends to overestimate small parallax angles, this works to our advantage at low parallax.

The first plot in Fig. 6 shows that Mid, LinLS and our two methods underestimate small parallax angles slightly less than the rest. It is no coincidence that precisely these four methods achieve the best 3D accuracy; Fig. 7 suggests that underestimating a small parallax angle has a severe impact on 3D accuracy. At low parallax, it seems that our methods hit the sweet spot by (1) underestimating less than the optimal methods and DLT (thus achieving lower 3D errors) and (2) overestimating less than Mid and LinLS (thus achieving lower parallax errors).

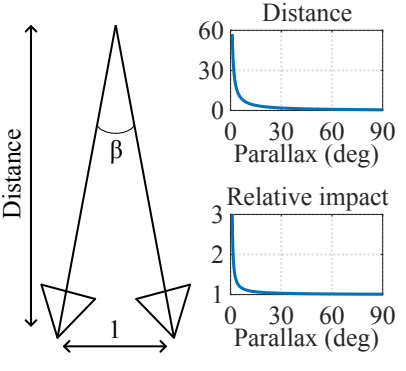

Figure 7: **Top:** In this 2D example, the distance is proportional to $\cot(\beta/2)$. **Bottom:** Relative impact is defined as $|f(\beta)/g(\beta)|$ where $f(\beta)$ and $g(\beta)$ are the distance errors when the parallax is under- and overestimated by 0.5 deg, respectively. Underestimating the parallax angle yields a larger distance error than overestimating it by the same degree, and especially more so for smaller parallax.

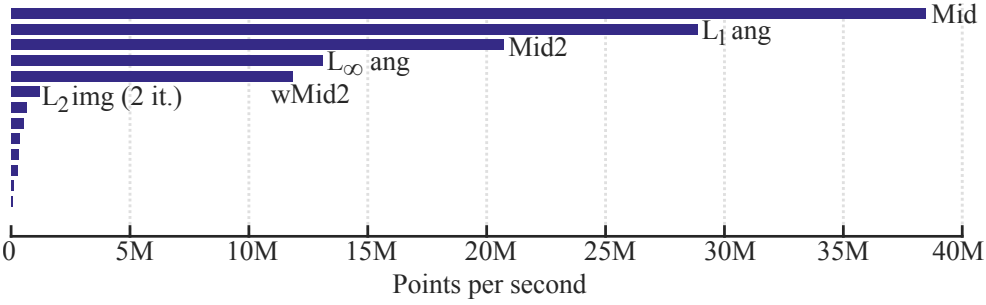

Figure 8: Speed of computing a 3D point. In descending order: Mid (38M), L_1 ang (29M), Mid2 (21M), L_∞ ang (13M), wMid2 (12M), L_2 img 2 it. (1.2M), L_2 ang (650K), L_2 img 5 it. (550K), LinLS (350K), DLT (330K), L_∞ img (270K), L_2 img (120K), L_1 img (65K).

We suspect that some of the large 2D errors of Mid and LinLS at low parallax are related to their large parallax overestimation (see the second row of Fig. 5). Large reprojection errors mean that the rays were corrected by a large amount, and pivoting two almost parallel rays (i.e., rays that correspond to a low-parallax point) will most likely increase the angle between them. This link between the 2D and the parallax error could partially explain why our methods yield smaller 2D errors than Mid and LinLS for low-parallax points.

We also found that the 2D errors of wMid2 resemble those of L_∞ methods (see the bottom row of Fig. 5). Note that the L_∞ optimal solutions yield the equal reprojection errors in the two views [19, 24]. This suggests that our inverse depth weighting (Section 4.4) not only reduces the overall 2D errors, but also balances the reprojection errors in the two images.

Fig. 8 compares the speed of each method. We included the test of the adequacy (Section 4.3) in both our methods. All methods were implemented in C++ using the Eigen library [1], compiled using GCC [2] with -O3 level optimization, and run on a laptop CPU (Intel i7-4810MQ, 2.8 GHz). Although wMid2 is almost two times slower than Mid2 and three times than Mid, it is still at least ten times faster than the state-of-the-art method [20]. Notice that even though (7) and (8) are similar, Mid is still almost twice faster than Mid2. This happens for two reasons: First, we avoid computing the square root in (8) by multiplying the numerator and denominator by $\|\mathbf{p}\|$ and get $\lambda_{\text{mid}0} = \frac{\mathbf{p}^\top \mathbf{r}}{\mathbf{p}^\top \mathbf{p}}$ and $\lambda_{\text{mid}1} = \frac{\mathbf{p}^\top \mathbf{q}}{\mathbf{p}^\top \mathbf{p}}$. Second, the test of adequacy described in Section 4.3 takes longer than the standard cheirality check. Nevertheless, given that the computational cost of triangulation is relatively small compared to other operations (e.g., point matching, pose estimation and structure refinement) [15], our methods provide an excellent trade-off between speed and accuracy.

6 Discussions

6.1 On Optimality

A reasonable doubt is that the proposed (weighted) midpoint method may be in fact optimal in some error criterion. If we consider algebraic errors, it is obvious that our midpoint minimizes $\|\mathbf{x}'_1 - \mathbf{g}(\mathbf{R},\mathbf{t},\mathbf{f}_0,\mathbf{f}_1)\|$ where $\mathbf{g}(\cdot)$ is the vector-valued function given by the right-hand side of (6) or (10). However, there are infinitely many other error functions for which the global minimum is $\mathbf{g}(\mathbf{R},\mathbf{t},\mathbf{f}_0,\mathbf{f}_1)$, and it is hard to tell which one is geometrically meaningful (at least, we could not find it or disprove its existence).

Another reasonable doubt is that, among the GWM methods, there could be a better one than ours, since we have not proved the optimality of the proposed weighted midpoint. A more fundamental question is then: What accuracy measure should we choose to define the optimality, knowing that there is a discrepancy between different types of errors, e.g., image/angular reprojection errors in different norms, 3D and parallax errors? At least for our method, it seems that the weighting affects mostly the 2D errors, so there might be a certain weighting scheme that guarantees 2D optimality without compromising 3D and parallax accuracy. A more elaborate error analysis and the comparison of different weightings within the GWM framework remain for future work. For more discussion of the optimality in geometric vision, we refer to [11] and Appendix 3 of [13].

6.2 On Practical Implications

In many SfM pipelines, two-view triangulation is used to initialize the 3D map points prior to the bundle adjustment [25, 28, 29]. Since the points with small parallax angles are associated with large 3D uncertainty, they are usually discarded. This strategy is viable if there are enough correspondences with large parallax angles. However, it is not ideal, as (1) low-parallax points can be useful for camera orientation estimation [5, 27] and (2) if the goal itself is to reconstruct the scene with large depths compared to the baseline. For problems such as reconstruction from small-baseline (or accidental) motions [7, 31], small parallax angles are quite common, so our method could be relevant. Extending our method to multiple views for reconstructing low-parallax scenes would be an interesting future direction.

7 Conclusions

Triangulation from two views with known calibration and pose is an age-old problem in computer vision. Existing methods formulate the problem as the minimization of some cost function, most commonly reprojection errors. In this paper, we asked ourselves if this is really the best approach. To this end, we proposed a novel variant of the classic midpoint method that does not minimize geometric or algebraic errors.

We found that all the existing methods we evaluated perform poorly at low parallax, producing large errors in either 2D, 3D or parallax. On the other hand, our midpoint method achieves very good overall accuracy. We also showed that incorporating the inverse depth weighting can further reduce the 2D errors. Although our method is not theoretically optimal, it provides, with speed and simplicity, a superior balance of 2D, 3D and parallax accuracy in practice.

Acknowledgement

We thank Christopher Mei at Microsoft for valuable discussions. This work was partially supported by the Spanish government (project PGC2018-096367-B-I00) and the Aragón regional government (Grupo DGA-T45_17R/FSE).

Appendix

In the following, we derive (8). In literature, the formula has been used many times without derivation [17, 19, 20]. For the sake of completeness, we present the full derivations here.

In doing so, we will use the following properties of the dot product and the cross product operations:

$$\widehat{\mathbf{a}} \times (\widehat{\mathbf{a}} \times \mathbf{b}) = \widehat{\mathbf{a}}(\widehat{\mathbf{a}} \cdot \mathbf{b}) - \mathbf{b}. \tag{11}$$

$$(\widehat{\mathbf{a}} \times \mathbf{b}) \cdot (\widehat{\mathbf{a}} \times \mathbf{c}) = \mathbf{b} \cdot \mathbf{c} - (\widehat{\mathbf{a}} \cdot \mathbf{b})(\widehat{\mathbf{a}} \cdot \mathbf{c}). \tag{12}$$

$$(\mathbf{a} \times \mathbf{b}) \times (\mathbf{a} \times \mathbf{c}) = (\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}))\,\mathbf{a}. \tag{13}$$

Next, we introduce the following lemma:

Lemma 1 (The Closest Pair of Points on Two Skew Lines)
Consider two skew lines $\mathbf{L}_0(s_0) = \mathbf{c}_0 + s_0\mathbf{m}_0$ *and* $\mathbf{L}_1(s_1) = \mathbf{c}_1 + s_1\mathbf{m}_1$ *in 3D space. Let* $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$ *and* $(\mathbf{r}_0, \mathbf{r}_1)$ *be the two points on each line that form the closest pair. Then,*

$$\mathbf{r}_0 = \mathbf{c}_0 + \frac{(\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1) \cdot (\widehat{\mathbf{m}}_1 \times \mathbf{t})}{\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2} \widehat{\mathbf{m}}_0 \tag{14}$$

and

$$\mathbf{r}_1 = \mathbf{c}_1 + \frac{(\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1) \cdot (\widehat{\mathbf{m}}_0 \times \mathbf{t})}{\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2} \widehat{\mathbf{m}}_1. \tag{15}$$

Proof. In geometry, it is a well-known fact that the closest pair of points on two skew lines lie on the common perpendicular to both lines. In other words, the vector $\mathbf{r}_0 - \mathbf{r}_1$ is perpendicular to both $\mathbf{L}_0$ and $\mathbf{L}_1$. Therefore, for some scalar τ,

$$\mathbf{r}_0 - \mathbf{r}_1 = \tau\,(\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1)\,. \tag{16}$$

Since point $\mathbf{r}_0$ and $\mathbf{r}_1$ are respectively located along $\mathbf{L}_0$ and $\mathbf{L}_1$, we can write

$$\mathbf{r}_0 = \mathbf{c}_0 + \lambda_0\widehat{\mathbf{m}}_0 \quad \text{and} \quad \mathbf{r}_1 = \mathbf{c}_1 + \lambda_1\widehat{\mathbf{m}}_1. \tag{17}$$

for some scalar λ_0 and λ_1. Then, (16) becomes

$$\mathbf{t} + \lambda_0\widehat{\mathbf{m}}_0 - \lambda_1\widehat{\mathbf{m}}_1 = \tau\mathbf{n}, \tag{18}$$

where $\mathbf{n} = \widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1$. This makes a system of three equations (in each coordinate x, y and z) with three unknowns λ_0, λ_1, and τ. Removing τ from the equations leads to

$$\frac{t_x + \lambda_0 m_{0x} - \lambda_1 m_{1x}}{n_x} = \frac{t_y + \lambda_0 m_{0y} - \lambda_1 m_{1y}}{n_y} \tag{19}$$

and

$$\frac{t_y + \lambda_0 m_{0y} - \lambda_1 m_{1y}}{n_y} = \frac{t_z + \lambda_0 m_{0z} - \lambda_1 m_{1z}}{n_z}. \tag{20}$$

Note that $\mathbf{t} = [t_x, t_y, t_z]^\intercal$, $\mathbf{n} = [n_x, n_y, n_z]^\intercal$, $\widehat{\mathbf{m}}_0 = [m_{0x}, m_{0y}, m_{0z}]^\intercal$ and $\widehat{\mathbf{m}}_1 = [m_{1x}, m_{1y}, m_{1z}]^\intercal$. From (19) and (20), we get

$$\lambda_0 = \frac{\lambda_1\,(m_{1x}n_y - m_{1y}n_x) + t_y n_x - t_x n_y}{m_{0x}n_y - m_{0y}n_x} \tag{21}$$

and

$$\lambda_0 = \frac{\lambda_1\,(m_{1y}n_z - m_{1z}n_y) + t_z n_y - t_y n_z}{m_{0y}n_z - m_{0z}n_y}. \tag{22}$$

Equating the right-hand sides of (21) and (22) leads to

$$\lambda_1 = \frac{A-B}{C-D}, \tag{23}$$

where

$$\begin{aligned}
A &= (t_z n_y - t_y n_z)(m_{0x} n_y - m_{0y} n_x) \\
B &= (t_y n_x - t_x n_y)(m_{0y} n_z - m_{0z} n_y) \\
C &= (m_{1x} n_y - m_{1y} n_x)(m_{0y} n_z - m_{0z} n_y) \\
&= (\widehat{\mathbf{m}}_1 \times \mathbf{n})_z (\widehat{\mathbf{m}}_0 \times \mathbf{n})_x, \\
D &= (m_{1y} n_z - m_{1z} n_y)(m_{0x} n_y - m_{0y} n_x) \\
&= (\widehat{\mathbf{m}}_1 \times \mathbf{n})_x (\widehat{\mathbf{m}}_0 \times \mathbf{n})_z.
\end{aligned}$$

We can rearrange $A-B$ into

$$A - B = n_y \mathbf{t} \cdot \begin{pmatrix} m_{0y} n_z - m_{0z} n_y \\ m_{0z} n_x - m_{0x} n_z \\ m_{0x} n_y - m_{0y} n_x \end{pmatrix}. \tag{24}$$

The latter term in the dot product of (24) is equal to $\widehat{\mathbf{m}}_0 \times \mathbf{n}$. Thus,

$$\begin{aligned}
A - B &= n_y \mathbf{t} \cdot (\widehat{\mathbf{m}}_0 \times (\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1)) \\
&\overset{(11)}{=} n_y \mathbf{t} \cdot (\widehat{\mathbf{m}}_0 (\widehat{\mathbf{m}}_0 \cdot \widehat{\mathbf{m}}_1) - \widehat{\mathbf{m}}_1) \\
&= n_y ((\widehat{\mathbf{m}}_0 \cdot \widehat{\mathbf{m}}_1)(\widehat{\mathbf{m}}_0 \cdot \mathbf{t}) - \widehat{\mathbf{m}}_1 \cdot \mathbf{t}) \\
&\overset{(12)}{=} -n_y (\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1) \cdot (\widehat{\mathbf{m}}_0 \times \mathbf{t}).
\end{aligned} \tag{25}$$

We can rearrange $C-D$ into

$$\begin{aligned}
C - D &= (\widehat{\mathbf{m}}_1 \times \mathbf{n})_z (\widehat{\mathbf{m}}_0 \times \mathbf{n})_x - (\widehat{\mathbf{m}}_1 \times \mathbf{n})_x (\widehat{\mathbf{m}}_0 \times \mathbf{n})_z \\
&= ((\widehat{\mathbf{m}}_1 \times \mathbf{n}) \times (\widehat{\mathbf{m}}_0 \times \mathbf{n}))_y \\
&\overset{(13)}{=} ((\mathbf{n} \cdot (\widehat{\mathbf{m}}_1 \times \widehat{\mathbf{m}}_0))\, \mathbf{n})_y \\
&= \left(-\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2 \mathbf{n}\right)_y \\
&= -\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2 n_y.
\end{aligned} \tag{26}$$

Substituting (25) and (26) into (23) gives

$$\lambda_1 = \frac{(\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1) \cdot (\widehat{\mathbf{m}}_0 \times \mathbf{t})}{\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2}. \tag{27}$$

Finally, substituting (27) into (17) leads to (15). Equation (14) is derived analogously. ■

By substituting $\mathbf{R}\mathbf{f}_0$ and $\mathbf{f}_1$ into $\mathbf{m}_0$ and $\mathbf{m}_1$, we can use lemma 1 to obtain $\lambda_{\text{mid}0}$ and $\lambda_{\text{mid}1}$ in (8).

$$\lambda_{\text{mid}0} = \frac{(\widehat{\mathbf{R}\mathbf{f}_0} \times \widehat{\mathbf{f}}_1) \cdot (\widehat{\mathbf{f}}_1 \times \mathbf{t})}{\|\widehat{\mathbf{R}\mathbf{f}_0} \times \widehat{\mathbf{f}}_1\|^2}, \quad \lambda_{\text{mid}1} = \frac{(\widehat{\mathbf{R}\mathbf{f}_0} \times \widehat{\mathbf{f}}_1) \cdot (\widehat{\mathbf{R}\mathbf{f}_0} \times \mathbf{t})}{\|\widehat{\mathbf{R}\mathbf{f}_0} \times \widehat{\mathbf{f}}_1\|^2}. \tag{28}$$

Letting $\mathbf{p} = \widehat{\mathbf{R}\mathbf{f}_0} \times \widehat{\mathbf{f}}_1$, $\mathbf{q} = \widehat{\mathbf{R}\mathbf{f}_0} \times \mathbf{t}$ and $\mathbf{r} = \widehat{\mathbf{f}}_1 \times \mathbf{t}$ leads to (8).

References

[1] Eigen Library v3. http://eigen.tuxfamily.org.

[2] GNU Compiler Collection (gcc). https://www.gnu.org/software/gcc.

[3] P. A. Beardsley, A. Zisserman, and D. W. Murray. Navigation using affine structure from motion. In *Eur. Conf. on Computer Vision*, pages 85–96, 1994.

[4] P. A. Beardsley, A. Zisserman, and D. W. Murray. Sequential updating of projective and affine structure from motion. *Intl. Journal of Computer Vision*, 23(3):235–259, 1997.

[5] J. Civera, A. J. Davison, and J. M. M. Montiel. Interacting multiple model monocular slam. In *IEEE Intl. Conf. on Robotics and Automation*, pages 3704–3709, 2008.

[6] J. Civera, A. J. Davison, and J. M. M. Montiel. Inverse depth parametrization for monocular SLAM. *IEEE Trans. Robot.*, 24(5):932–945, 2008. ISSN 1552-3098. doi: 10.1109/TRO.2008.2003276.

[7] Julie Delon and Bernard Rougé. Small baseline stereovision. *Journal of Mathematical Imaging and Vision*, 28(3):209–223, 2007.

[8] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: fast semi-direct monocular visual odometry. In *IEEE Intl. Conf. on Robotics and Automation*, pages 15–22, 2014.

[9] C. G. Harris. Determination of ego-motion from matched points. In *Third Alvey Vision Conference*, pages 189–192, 1987.

[10] C.G. Harris and J.M. Pike. 3D positional integration from image sequences. *Image and Vision Computing*, 6(2):87 – 90, 1988.

[11] R. Hartley and F. Kahl. Optimal algorithms in multiview geometry. In *Asian Conf. on Computer Vision*, pages 13–34, 2007.

[12] R. Hartley and F. Schaffalitzky. L_∞ minimization in geometric reconstruction problems. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 504–509, 2004.

[13] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[14] R. I. Hartley. Euclidean reconstruction from uncalibrated views. In *Proceedings of the Second Joint European - US Workshop on Applications of Invariance in Computer Vision*, pages 237–256, 1994.

[15] R. I. Hartley and P. Sturm. Triangulation. *Comput. Vis. Image Underst.*, 68(2):146–157, 1997.

[16] K. Kanatani, Y. Sugaya, and H. Niitsuma. Triangulation from two views revisited: Hartley-Sturm vs. optimal correction. In *British Machine Vision Conference*, pages 173–182, 2008.

[17] Y. Kanazawa and K. Kanatani. Reliability of 3-D reconstruction by stereo vision. *IECE Trans. Inf. & Syst.*, E78-D(10):1301–1306, 1995.

[18] Qifa Ke and Takeo Kanade. *Robust subspace computation using L1 norm*. School of Computer Science, Carnegie Mellon University Pittsburgh, PA, 2003.

[19] S. Lee and J. Civera. Closed-form optimal two-view triangulation based on angular errors. In *IEEE Int. Conf. on Computer Vision*, 2019.

[20] P. Lindstrom. Triangulation made easy. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1554–1561, 2010.

[21] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(5828):133–135, 1981.

[22] B. Mičušík and T. Pajdla. Structure from motion with wide circular field of view cameras. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(7):1135–1149, 2006.

[23] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robot.*, 31(5):1147–1163, 2015. doi: 10.1109/tro.2015.2463671.

[24] D. Níster. *Automatic Dense Reconstruction from Uncalibrated Video Sequences*. PhD thesis, KTH, 2001.

[25] S. Nousias, M. Lourakis, and C. Bergeles. Large-scale, metric structure from motion for unordered light fields. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2019.

[26] J. Oliensis. Exact two-image structure from motion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(12):1618–1633, 2002.

[27] C. Pirchheim, D. Schmalstieg, and G. Reitmayr. Handling pure camera rotation in keyframe-based SLAM. In *IEEE International Symposium on Mixed and Augmented Reality*, pages 229–238, 2013.

[28] J. L. Schönberger and J. Frahm. Structure-from-Motion revisited. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 4104–4113, 2016.

[29] N. Snavely, S. M. Seitz, and R. Szeliski. Photo Tourism: Exploring photo collections in 3D. *ACM Trans. Graph.*, 25(3):835–846, 2006.

[30] K. Yang, W. Fang, Y. Zhao, and N. Deng. Iteratively reweighted midpoint method for fast multiple view triangulation. *IEEE Robot. Autom. Lett.*, 4(2):708–715, 2019.

[31] F. Yu and D. Gallup. 3D reconstruction from accidental motion. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 3986–3993, 2014.