Point triangulation through polyhedron collapse using the ℓ_∞ norm

Simon Donné, Bart Goossens, Wilfried Philips
Ghent University
iMinds-IPI-UGent
Simon.Donne@telin.ugent.be, Bart.Goossens@telin.ugent.be, philips@telin.ugent.be

Abstract

Multi-camera triangulation of feature points based on a minimisation of the overall ℓ_2 reprojection error can get stuck in suboptimal local minima or require slow global optimisation. For this reason, researchers have proposed optimising the ℓ_∞ norm of the ℓ_2 single view reprojection errors, which avoids the problem of local minima entirely. In this paper we present a novel method for ℓ_∞ triangulation that minimizes the ℓ_∞ norm of the ℓ_∞ reprojection errors: this apparently small difference leads to a much faster but equally accurate solution which is related to the MLE under the assumption of uniform noise. The proposed method adopts a new optimisation strategy based on solving simple quadratic equations. This stands in contrast with the fastest existing methods, which solve a sequence of more complex auxiliary Linear Programming or Second Order Cone Problems. The proposed algorithm performs well: for triangulation, it achieves the same accuracy as existing techniques while executing faster and being straightforward to implement.

1. Introduction

Multiview triangulation is the problem of estimating the 3D location of a physical point from observations in multiple camera views. In low-complexity people tracking methods for example, these feature points are the centroids of foreground blobs in a foreground/background segmented video sequence. In more complex methods, the features are computed using SURF [14] or similar detectors.

In realistic applications, confusion is possible between feature points corresponding to different points. This *correspondence problem* can be handled by grouped features according to their similarity and for each group triangulated the corresponding physical point. If the error of this triangulated point is too large, features are re-assigned after which the process is repeated until a satisfactory solution is reached. Clearly the performance of these methods depends crucially on that of the triangulation of separate points.

We propose a method based on minimizing the ℓ_∞ norm of the ℓ_∞ reprojection error, based on geometric insights of the 3D space. Each camera $k \in \{1, \ldots, K\}$ has a camera-specific coordinate system defined by a rotation matrix R_k and a translation $\boldsymbol{c}_k$. The relationship between the global coordinates $\boldsymbol{r} = (X, Y, Z)^T$ of a physical point and its coordinates in camera k's reference system $\boldsymbol{r}_k = (X_k, Y_k, Z_k)^T$ is given by the linear relationship $\boldsymbol{r}_k(\boldsymbol{r}) = \mathrm{R}_k \boldsymbol{r} + \boldsymbol{c}_k$. Assuming a pinhole camera model [10], the unknown $\boldsymbol{r}$ would – under ideal circumstances – be observed by camera k as a feature point with homogeneous pixel coordinates $\boldsymbol{u}_k(\boldsymbol{r}) = (x_k(\boldsymbol{r}), y_k(\boldsymbol{r}), 1) = \left(\frac{X_k(\boldsymbol{r})}{Z_k(\boldsymbol{r})}, \frac{Y_k(\boldsymbol{r})}{Z_k(\boldsymbol{r})}, 1\right)$.

In reality – due to noisy observations – camera k will observe a feature at coordinates $\tilde{\boldsymbol{u}}_k = (\tilde{x}_k, \tilde{y}_k, 1)$ rather than the ideal coordinates $\boldsymbol{u}_k(\boldsymbol{r})$. The corresponding single view *reprojection error* $\gamma_k(\boldsymbol{r})$ for camera k is

$$\gamma_k(\boldsymbol{r}) \triangleq \|\tilde{\boldsymbol{u}}_k - \boldsymbol{u}_k(\boldsymbol{r})\|, \tag{1}$$

where $\|.\|$ is a suitably chosen norm.

Many methods in literature quantify the single view reprojection error in terms of the ℓ_2 norm, i.e. the euclidean distance between the measurement and the reprojection of a hypothesized location. In this paper, we will however adopt the ℓ_∞ norm:

$$\gamma_k(\boldsymbol{r}) \triangleq \max(|\tilde{x}_k - x_k(\boldsymbol{r})|, |\tilde{y}_k - y_k(\boldsymbol{r})|). \tag{2}$$

In any case, triangulation methods minimize the *aggregated* reprojection error

$$\gamma(\boldsymbol{r}) = \|(\gamma_1(\boldsymbol{r}), \ldots, \gamma_K(\boldsymbol{r}))\|. \tag{3}$$

Traditionally, the ℓ_2 norm is used to aggregate reprojection errors. However, ℓ_∞ methods use the ℓ_∞ norm instead – accordingly, so does our proposed method. Due to the nonlinearity of the camera projection, $\gamma(\boldsymbol{r})$ can have multiple local minima when defined in terms of the ℓ_2 norm [2,6,9]. In contrast, aggregating the reprojection errors with the ℓ_∞ norm always results in a convex set of stationary points because of its pseudo-convex character [1, 3, 5, 12, 16, 17].

This absence of local minima ensures that local optimisation methods converge to the global minimum of $\gamma(\boldsymbol{r})$.

Some existing methods use a mixed-norm: the ℓ_∞ norm for aggregation but the ℓ_2 norm for the reprojection error [1, 3, 12, 17]. Minimizing this aggregated error leads to a series of auxiliary second-order cone problems which are complex and time consuming to solve. More recent methods have evaluated the use of the ℓ_∞ norm for both the aggregated and the reprojection error (we will refer to this group as the full-ℓ_∞ norm methods). This change allows expressing $\gamma(\boldsymbol{r})$ as the point-wise maximum of linear functions. Our proposed method will perform the optimization by a series of line searches, and we show that our proposed method performs slightly faster than the existing full-ℓ_∞ norm methods, while arriving at the same optimum. Notably, our proposed method does not require anything more complex than the solution of one-dimensional quadratic equations and some basic matrix algebra.

Section 2 discusses existing triangulation methods, followed by a more detailed problem outline in section 3 after which we present the proposed method in section 4. The results in section 5 show that the proposed method is slightly faster than the state of the art methods while still achieving similar accuracy. Finally section 6 reaches the conclusion and we discuss possible future work in section 7.

2. Existing work

Traditionally, triangulation methods employ the ℓ_2 norm for both the single view and aggregated reprojection error. Minimizing this error corresponds to computing the maximum-likelihood estimate (MLE) under the assumption that the observed locations $\tilde{\boldsymbol{u}}_k$ are perturbed by additive white Gaussian noise (AWGN). Early methods [7, 9, 18] can get stuck in local minima caused by the non-linearity of the camera projection. Resolving this issue can be done through a (complex) branch-and-bound approach [11] or by solving for the entire set of stationary points [2].

But the problem of local minima can also be resolved by using the ℓ_∞ norm in eq. (1): the so-called ℓ_∞ methods which result in minimax problems. Olsson et al. [17] show that the resulting cost function – the point-wise maximum – is a *quasi-convex* function. This quasi-convexity implies that the set of stationary points is convex: no local optima exist. Yet even for many existing ℓ_∞ methods, single view reprojection errors are expressed in terms of the ℓ_2 norm; we will call them *hybrid* ℓ_∞ methods [1, 3, 12, 17]. We note that the criterion optimized by the hybrid approaches cannot easily be related to the maximum-likelihood estimation of a noise model.

In contrast to hybrid methods, we will express single view reprojection errors in terms of the ℓ_∞ norm in this paper. As shown in the supplementary material, this full-ℓ_∞ norm method results in the maximum-likelihood estimation under the assumption of uniform noise on the observations $\tilde{\boldsymbol{u}}_k$, lending a statistical foundation to the proposed method. This norm was already studied in [5, 16], with applications to large-dimensional multi-view geometry problems.

The hybrid methods result in the solution of a series of SOCP problems, which is time-consuming. In the seminal work of Kahl and Hartley [12] the bisection algorithm was introduced for optimizing the hybrid cost function. In the bisection algorithm a binary search narrows an interval containing the optimal $\gamma^\star = \arg\min_{\boldsymbol{r}} \gamma(\boldsymbol{r})$. Olsson et al. [17] also propose an approach based on a series of auxiliary problems, but not based on bisection: for their method, the auxiliary problems are local approximations to the original problem such that the KKT criteria are a simple approximation to the KKT criteria of the original problem. In [1], Agarwal et al. present a survey of the ℓ_∞ norm for aggregating reprojection errors. They present the Gugat method which outperforms the techniques of Olsson et al. [17] and Kahl [12] et al. while still being based on a series of SOCP problems. They also show the use of the ℓ_∞ norm for aggregating and the ℓ_1 norm for the reprojection results in a series of LP problems. Their method is still based on a series of SOCP problems and is therefore only slightly faster than the methods by Olsson and Kahl. Finally, the authors of [3] show how the solution to the previous SOCP problem can be used as initialization to the next iteration's problem in order to speed up the solver.

Recently, full-ℓ_∞ norm has garnered some attention. In [5], the authors propose a split-Bregman approach to the optimisation problem which results in an elegant modification of existing bundle adjustment (BA) packages. In each iteration of the algorithm, a single bundle adjustment iteration is performed, as well as one evaluation of a proximal operator (which requires finding the root of a one-dimensional function). The authors show that it is faster than the Gugat method from [1], but it still requires the use of existing software packages such as SBA [13].

The authors of [16] use the full-ℓ_∞ norm as a way to detect outliers: handling the entire dataset as a single entity, outliers are removed based on their reprojection errors. We mention their approach here because it is an existing use of the full-ℓ_∞ norm, but the optimization is not continued beyond the removal of outliers.

In the proposed algorithm, we do not require a bisection algorithm, nor do we solve time-consuming auxiliary SOCP or BA problems. Rather, the proposed algorithm directly minimizes $\gamma(\boldsymbol{r})$ through a sequence of line searches. The direction of the line search is computed in a straight forward manner conform the KKT conditions, and the line searches boil down to solving quadratic equations.

3. Background theory

We will adopt the common convention that only points in front of cameras are of interest, i.e. $Z_k(\boldsymbol{r}) \geq 0$ for all k: this is the so-called cheirality constraint from [8]. Taking into account that $\boldsymbol{r}_k = Z_k(\boldsymbol{r})\boldsymbol{u}_k(\boldsymbol{r})$, it follows that

$$\gamma_k(\boldsymbol{r}) = \left\| Z_k(\boldsymbol{r})\tilde{\boldsymbol{u}}_k - \boldsymbol{r}_k(\boldsymbol{r}) \right\|_\infty / Z_k(\boldsymbol{r}). \tag{4}$$

The objective of the proposed algorithm is to minimise $\gamma(\boldsymbol{r})$:

$$\min_{\boldsymbol{r}} \max_{k} \gamma_k(\boldsymbol{r}). \tag{5}$$

Or equivalently to

$$\min_{\boldsymbol{r},\gamma} \gamma \text{ subject to } \gamma \geq \gamma_k(\boldsymbol{r}) \quad \forall k. \tag{6}$$

In order to remove the non-linearity of the constraints from the ℓ_∞ norm, we introduce the constant vectors $\boldsymbol{i}_1 = (1,0,0)^T$,$\boldsymbol{i}_2 = (-1,0,0)^T$,$\boldsymbol{i}_3 = (0,1,0)^T$,$\boldsymbol{i}_4 = (0,-1,0)^T$. Using these notations, with k indexing the K cameras and l indexing the vectors $\boldsymbol{i}_l$, the constraints can be expressed as a set of linear inequalities:

$$\begin{aligned}&\min_{\boldsymbol{r},\gamma} \gamma \text{ subject to}\\ &\quad \gamma \geq \gamma_{k,l}(\boldsymbol{r}) \overset{\triangle}{=} \boldsymbol{i}_\ell \cdot \left(\boldsymbol{r}_k(\boldsymbol{r}) - Z_k(\boldsymbol{r})\tilde{\boldsymbol{u}}_k\right)/Z_k(\boldsymbol{r}) \quad \forall k,l.\\ &\equiv (\gamma + \boldsymbol{i}_l \cdot \tilde{\boldsymbol{u}}_k) Z_k(\boldsymbol{r}) \geq \boldsymbol{i}_l \cdot \boldsymbol{r}_k(\boldsymbol{r}), \forall k,l.\end{aligned} \tag{7}$$

As $Z_k(\boldsymbol{r})$ and $\boldsymbol{r}_k(\boldsymbol{r})$ are linear functions of $\boldsymbol{r}$ and $\tilde{\boldsymbol{u}}_k$ is a known constant vector, we can write that

$$\gamma_{k,l}(\boldsymbol{r}) = \frac{\boldsymbol{a} \cdot \boldsymbol{r} + b}{\boldsymbol{c} \cdot \boldsymbol{r} + d}. \tag{8}$$

As shown by the authors of [17], such functions are all pseudo-convex. As a result the aggregated error – a point-wise maximum of these functions – is also pseudo-convex, implying that there are no local optima.

For a given value of γ, the four constraints for a given camera define a pyramid in space, containing all points whose reprojection error for that camera is smaller than γ. The feasible set for the constraints (7) is the intersection of the pyramids for all of the cameras, as shown in Figure 1 this is a convex polyhedron. Our optimization algorithm iteratively reduces the value of γ until it approaches the optimal value $\gamma^\star$. This will cause this polyhedron to collapse onto itself: due to the quasi-convexity, each polyhedron will be contained in the previous iteration's polyhedron, and each of them will in turn contain the next step's polyhedron and, eventually, the optimal point $\boldsymbol{r}^\star$.

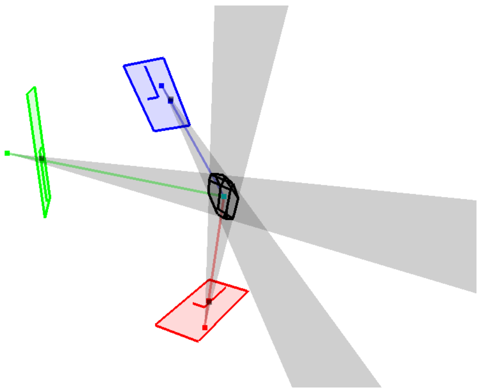
Figure 1. Example set-up. The cameras and their viewing planes are shown, as well as the pyramids they project in space for a given value of γ and the resulting polyhedron.

Over the course of the algorithm we compute improving directions in a sequence of points $\boldsymbol{r}^{(t)}$. In each subsequent point, one of the constraints is active (fulfilled with equality), otherwise problem (7) would not be in its optimum. This also means that at each point of the iteration, the current location estimate lies on the hull of the polyhedron corresponding to its ℓ_∞ norm error. Because the optimum lies in the interior of the polyhedron, we select the improving direction in terms of the gradients of the active constraints, i.e. the *inward-pointing* normal of the polyhedron faces in which the current estimate lies. In global coordinates these gradients are given by

$$\boldsymbol{g}_{k,l}(\gamma) = \mathrm{R}_k^T \left(-\boldsymbol{i}_l + (0,0,\gamma + \boldsymbol{i}_l \cdot \tilde{\boldsymbol{u}}_k)^T\right). \tag{9}$$

4. Proposed approach

The algorithm starts with an arbitrary initial estimate $\boldsymbol{r}^{(0)}$ of the sought position. We then iteratively select an improving direction and perform a line search. The only requirement for the initial point is that it must lie in front of all of the cameras (i.e. it satisfies the cheirality constraint).

In step (t) we define a polygon by using $\gamma = \gamma(\boldsymbol{r}^{(t-1)})$ in equation (7). $\boldsymbol{r}(t-1)$ lies on the hull of this polygon, and due to pseudo-convexity the optimal point $\boldsymbol{r}^\star$ must lie within the polyhedron. We select an improving direction $\boldsymbol{d}^{(t)}$ (a direction pointing towards the interior of the polyhedron) and perform a line search along this direction. This process is repeated until the KKT constraints are fulfilled.

In practice, for example due to machine precision, some inequalities may only fulfil

$$\gamma(\boldsymbol{r}) \approx \gamma_{k,l}(\boldsymbol{r}). \tag{10}$$

We therefore evaluate whether constraints are *active* using a threshold ϵ, i.e. by checking whether

$$(1-\epsilon)\gamma(\boldsymbol{r}) \leq \gamma_{k,l}(\boldsymbol{r}). \tag{11}$$

In our implementation we have used $\epsilon = 10^{-5}$.

4.1. Choice of improving direction

An improving direction points is computed in a point $\boldsymbol{r}^{(t-1)}$. At least one constraint is active, i.e. $\boldsymbol{r}^{(t-1)}$ lies on the surface of the polyhedron. Assuming that there are J active constraints, we will denote their normals by $\boldsymbol{n}_1$ through $\boldsymbol{n}_J$, in favour of brevity: which constraints correspond to the various gradients is irrelevant for the following discussion.

In the case of a single active equality we simply select the normal of this active inequality as the improving direction: $\boldsymbol{d}^{(t)} = \boldsymbol{n}_1$, which is simply the gradient descent approach.

Multiple active constraints complicate the direction choice, though. For two active constraints, $\boldsymbol{r}^{(t-1)}$ lies on the edge of two faces of the polyhedron. The chosen direction

$$\boldsymbol{d}^{(t)} = \boldsymbol{n}_1 + \boldsymbol{n}_2 \tag{12}$$

points along the interior angle bisector of the two faces and is orthogonal to the edge as shown for the two-dimensional case in figure 2.

In case of three active constraints active, $\boldsymbol{r}^{(t-1)}$ is a vertex of the polyhedron. The improving direction $\boldsymbol{d}^{(t)}$ is constructed as

$$\begin{aligned} \boldsymbol{d}' &= \boldsymbol{n}_1 \times \boldsymbol{n}_2 + \boldsymbol{n}_2 \times \boldsymbol{n}_3 + \boldsymbol{n}_3 \times \boldsymbol{n}_1 \\ s &= \boldsymbol{n}_1 \cdot \boldsymbol{d} / \left| \boldsymbol{n}_1 \cdot \boldsymbol{d} \right| \\ \boldsymbol{d}^{(t)} &= s\boldsymbol{d}' \end{aligned} \tag{13}$$

This is the vector with the same scalar product to all of the active constraint normals: the intersection of the pairwise face angle bisectors, where the sign s is used to ensure that it points to the interior. In case the three normals are coplanar (the scalar product to any normal is zero), we test for each pair of normals whether their combination according to the previous paragraph violates the third normal, i.e. whether it has a negative dot product with this third normal. If and only if none of the pairs work, the algorithm terminates.

To handle cases of four or more active constraints, we investigate each of the possible triplets of active constraints and compute a candidate improving direction $\boldsymbol{d}^{(t)}$ as before. If this candidate direction violates any of the constraints not included in the triplet, we select the next triplet and the process is repeated. If one of the triplets leads to termination of the algorithm or if none of the triplets lead to an improving direction conform all of the constraints, it can be shown that $\boldsymbol{r}^{(t-1)}$ is optimal (see the supplementary material) and the algorithm terminates. The drawback is that all possible triplets need to be examined. However, in the result section we show that the number of simultaneously active constraints is only very rarely higher than 3 or 4, such that this possibility only has a major influence.

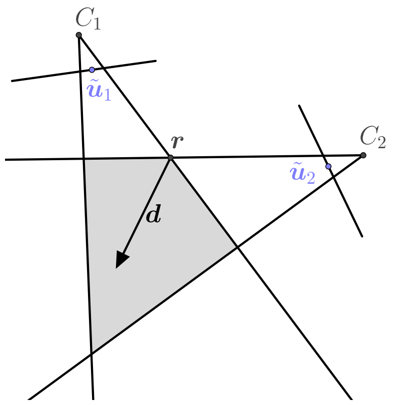

Figure 2. Illustration of the direction choice. The current estimate $\boldsymbol{r}$ and the improving direction $\boldsymbol{d}$ which is the sum of the gradients of each of the active constraints: the bisector for the corner.

In the supplementary material we show that the algorithm finishes (cannot find an improving direction) if and only if the KKT conditions are fulfilled. The current estimate is then a stationary point and due to the pseudo-convexity of the cost function, this stationary point is a global optimum. Hence, the algorithm has indeed reached its goal.

4.2. Line search

After selecting the improving direction for iteration (t), we step to the point $\boldsymbol{r}^{(t)} = \boldsymbol{r}^{(t-1)} + \alpha^{(t)}\boldsymbol{d}^{(t)}$. For notational brevity, let

$$f_{k,l}(\alpha) \triangleq \gamma_{k,l}\left(\boldsymbol{r}^{(t-1)} + \alpha\boldsymbol{d}^{(t)}\right). \tag{14}$$

For all active (k,l), $\frac{\partial}{\partial\alpha} f_{k,l}(\alpha)\big|_{\alpha=0}$ is negative. We choose the master active constraint as the one with the least negative value: the constraint which changes the least (and hence will remain active) for small steps along this line.

Now, let (k',l') be the master constraint and (k,l) any other (active or inactive) constraint. As (k',l') is active, $f_{k',l'}(\alpha)$ is a decreasing function of α for all α. On the other hand, $f_{k,l}(\alpha)$ may be increasing or decreasing and may not even be monotonic. For $\alpha = 0$, the activity of (k',l') implies that $f_{k,l}(0) \geq f_{k',l'}(0)$. Locally, the error for (k',l') is higher than the errors for all other constraints (k,l), but it decreases along the improving direction. At some value of α the graph may intersect with the graph of another constraint. Let $\alpha_{k,l}$ be the lowest strictly positive value of alpha for which $f_{k',l'}(\alpha) = f_{k,l}(\alpha)$, for any (k,l), i.e. the point where another constraint *might* become the new master constraint. As we know that $f_{k',l'}(\alpha)$ decreases with increasing α, the point $\boldsymbol{r}^{(t)} = \boldsymbol{r}^{(t-1)} + \alpha^{(t)}\boldsymbol{d}^{(t)}$ is guaranteed to have a lower reprojection error than $\boldsymbol{r}^{(t-1)}$. Finally, such an intersection is sure to exist if the direction points towards the interior of the polyhedron.

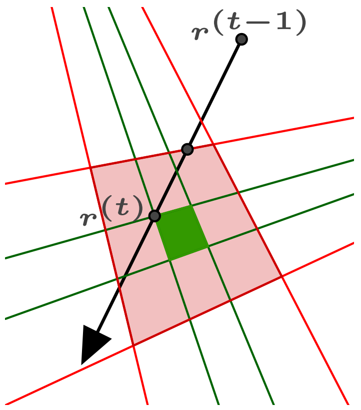

Figure 3. Example of the line search. Starting in $\boldsymbol{r}^{(t-1)}$, we step towards $\boldsymbol{r}^{(t)}$ so that it again lies on an edge of the polyhedron (the polygon in this 2D example). The unlabeled intermediate point lies only on an edge of the polyhedron and is rejected: we can still take a step along the improving direction without an increase in the cost function.

Evaluating the complementary constraint (the constraint corresponding to the same camera and the same coordinate as the master constraint, but the different sign) shows that it will always intersect for $f_{k',l'}(\alpha) = 0$, as that is where the master constraint's value equals that of its complementary constraint.

Now let $\alpha^{(t)} = \min_{k,l} \alpha_{k.l}$. This is the first point along the line search after which the master constraint *might* become inactive, and equivalently the first stationary point of the aggregated error along the search direction.

Equivalently, the selected value of $\alpha^{(t)}$ is the lowest positive value for which $\boldsymbol{r}^{(t)}$ once again lies on an edge of the polyhedron as shown for two dimensions in Figure 3 (where an edge of the polyhedron in 3D becomes a vertex of the polyhedron in 2D). This edge is defined by the master constraint and the constraint corresponding to the chosen value of α. This implies that, except for the first iteration, the improving direction will always be chosen based on at least two active constraints.

As a final note, we discuss the computation of $\alpha_{k,l}$ by solving the equation $f_{k',l'}(\alpha) = f_{k,l}(\alpha)$ for α, retaining only the smallest positive solution. As either side is a fraction of linear functions of α (see equation (8)), this equation is a quadratic equation in a single variable.

5. Results

5.1. Comparison of reprojection norms

The proposed cost function uses the ℓ_∞ norm in both the single view reprojection error and the aggregated error. Existing techniques for ℓ_∞ norm triangulation, however, sometimes use a mixed-norm: here we give a comparison between the optima of methods using ℓ_∞ norm aggregation and either ℓ_1,ℓ_2 or ℓ_∞ norm reprojection errors.

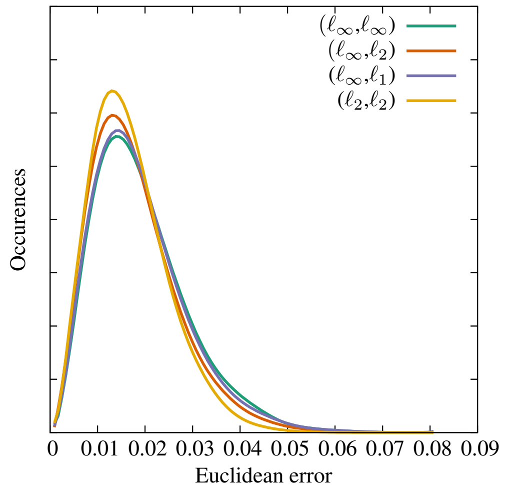

Figure 4. Accuracy for the various ℓ_∞-aggregation methods on synthetic data with Gaussian noise. The graph shows euclidean distances between the estimate and the ground truth.

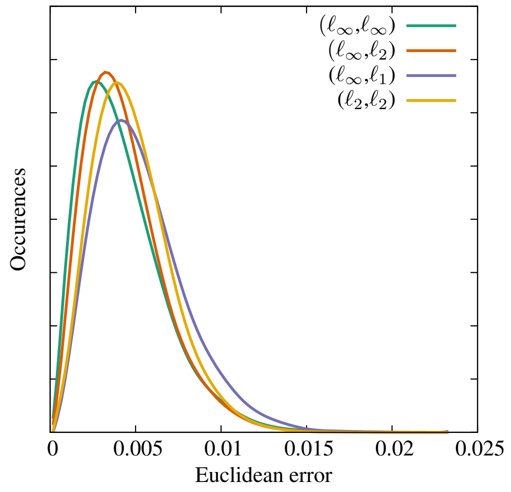

Figure 5. Accuracy for the various ℓ_∞-aggregation methods on synthetic data with Gaussian noise. The graph shows euclidean distances between the estimate and the ground truth.

The test set-up consists of 5 random cameras observing a sequence of random points. We perturb the ideal observations by the cameras either by additive white Gaussian noise or by additive white uniform noise. Figures 4 and 5 show bezier-curves fitted to the histograms for the euclidean error between the triangulated points and the ground truth. Roughly, a heavier tail can be said to correspond to less robustness: large errors occur more often. We see a predictable trend for the set with AWGN: the ℓ_∞ norm optimization is less robust than the ℓ_2 norm. When adding uniform noise, the graph implies that the ℓ_∞ norm aggregation and reprojection is the better choice: it is after all the MLE in that case (as shown in the supplementary material).

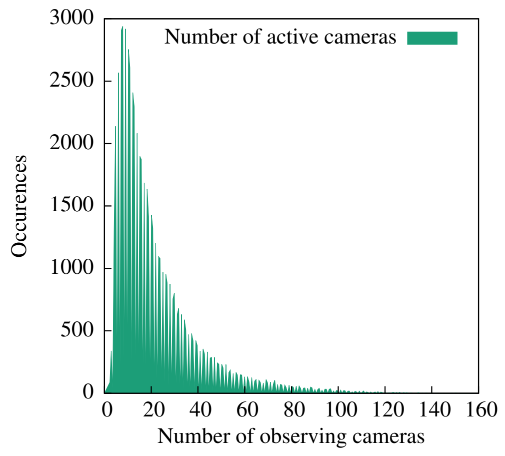

Figure 6. Histogram of the number of cameras observing a single point for the Örebro dataset.

5.2. Active inequalities and iterations

In order to illustrate the typically active number of constraints, we will use the measurements of Örebro Castle from [4]. This dataset is a collection of 761 views with a total of 58951 points visible in total, each visible in a subset of the views. In order to convey a sense of the number of views per point, we show a histogram of the number of cameras observing a point in figure 6.

In figure 7 we illustrate the number of active inequalities in subsequent iterations of the algorithm. The first iteration only ever has a single active inequality (the initial point is very unlikely to lie on an edge of the polyhedron). After that, two or three active inequalities occur more or less equally often; four is much less likely (one to two orders of magnitude), while on the entire dataset only a single point (in a single iteration) had five active inequalities.

Finally, figure 8 illustrates the distribution of the number of iterations required for a point to reach convergence.

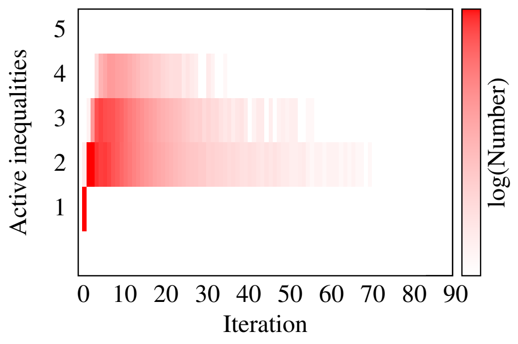

Figure 7. Heatmap (in log-scale) for the number of active constraints in a given iteration, normalized over the entire sequence.

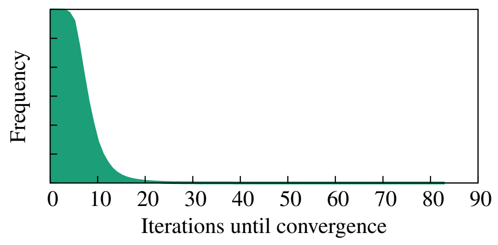

Figure 8. Distribution of the number of iterations required to reach convergence (over all points in the Örebro dataset).

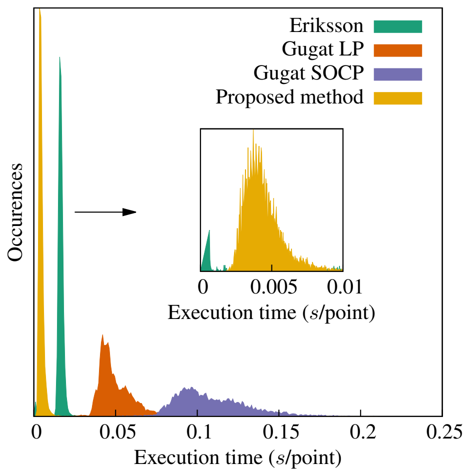

Figure 9. Run times for discussed methods on synthetic data: 10000 points each visible in exactly 10 views.

5.3. Comparison of execution speed

In order to show that our technique is faster than existing state-of-the-art methods for point triangulation, we compare with the Gugat algorithm from Agarwal [1] (both using the SOCP approach for the (ℓ_∞,ℓ_2) approach and the LP approach resulting from the (ℓ_∞,ℓ_1)-mixed norm), and the approach from Eriksson et al. [5] corresponding with the full-ℓ_∞ approach. As a first evaluation, figure 9 shows the timing results on synthetic data with a fixed 10 viewpoints per point. In realistic datasets, the number of viewpoints per point is bound to change as not all cameras observe all points (see section 5.2 and in particular figure 6).

The first real dataset we compare with is *dino*.[1] We also present results for the datasets of Alcatraz, Église du Dôme, Örebro castle, Stockholm town hall and the Vercingetorix statue from [4] and [15]: the results for the datasets as shown in Figures 10 through 15 illustrate that our proposed method executes faster than the existing methods (summarised in Table 1). All of the techniques were evaluated in MATLAB using SeDuMi as the solver for the LP and SOCP problems [19], and SBA for bundle adjustment [13].

[1] www.robots.ox.ac.uk/~vgg/data/data-mview.html

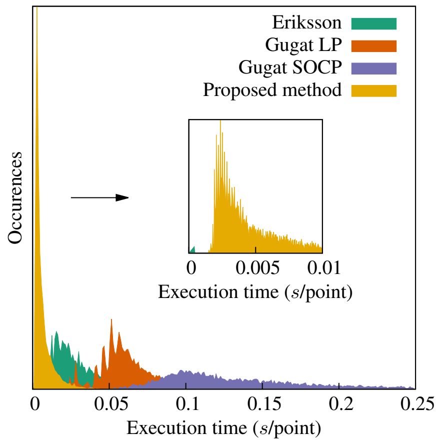

Figure 10. Run times for discussed methods on the Alcatraz dataset with 65072 points over 419 views.

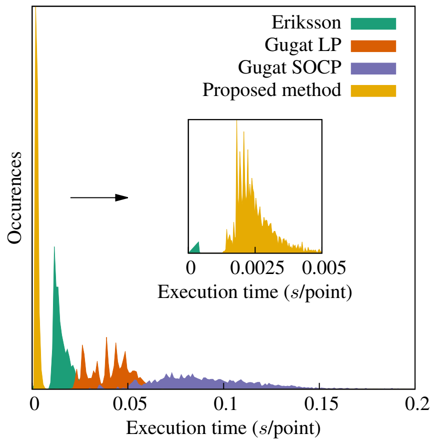

Figure 13. Run times for discussed methods on the Örebro Castle dataset with 59856 points over 761 views.

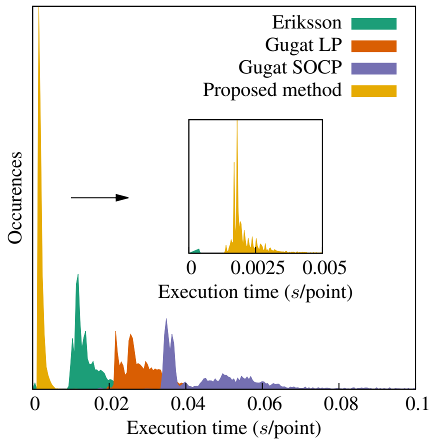

Figure 11. Run times for discussed methods on the *dino* dataset with 4983 points over 36 views.

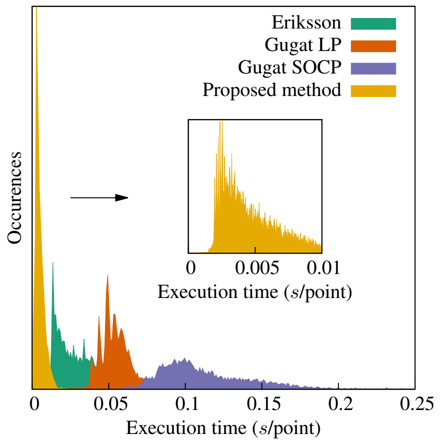

Figure 14. Run times for discussed methods on the Stockholm Town Hall dataset with 28096 points over 43 views.

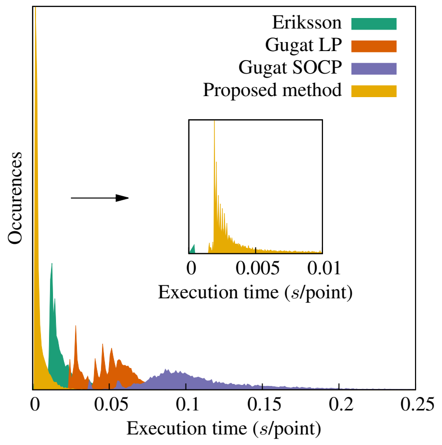

Figure 12. Run times for discussed methods on the Église du Dôme dataset with 84792 points over 85 views.

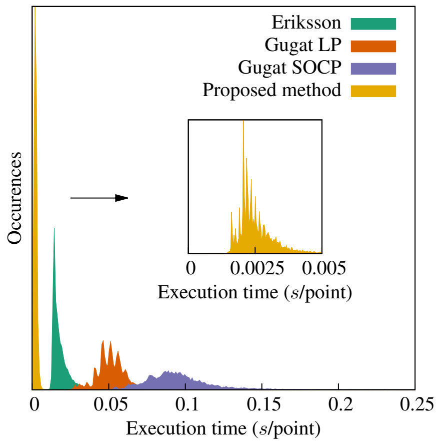

Figure 15. Run times for discussed methods on the Vercingetorix statue dataset with 10789 points over 68 views.

	Synthetic	Alcatraz	Örebro	dino	Stockholm	Église	Vercingetorix
Gugat SOCP	0.1090	0.2540	0.1338	0.0475	0.1561	0.1615	0.1131
Gugat LP	0.0498	0.0631	0.0422	0.0290	0.0548	0.0498	0.0513
Eriksson	0.0163	0.0404	0.0154	0.0140	0.0294	0.0262	0.0177
Proposed	0.0045	0.0066	0.0025	0.0022	0.0052	0.0044	0.0026

Table 1. Summary of the average execution times in seconds, over all datasets.

6. Conclusion

In this paper, we have proposed a novel technique of approaching triangulation by using the ℓ_∞ norm both for single view errors and the camera aggregation, resulting in a pseudo-convex problem without local minima.

The proposed method is based on geometrical interpretations of the cost function and its quasi-convexity of the objective function: the optimization can be seen as a series of polyhedrons representing a polyhedron collapsing onto itself until it has zero volume.

Our approach iteratively selects an improving direction and performs a line search along it. This is in contrast with the existing methods, which solve a sequence of complex problems. The less complex nature of the proposed approach results in an easier-to-implement method, while performing faster (in contrast, existing methods use advanced solvers for their auxiliary problems). This leads us to believe that a more efficient implementation would result in a larger speed benefit for the proposed method: the existing methods are implemented largely with optimized libraries while the proposed method was implemented solely in MATLAB.

7. Future Work

The presented work can be interpreted as the collapse of a polyhedron in 3D space. A promising avenue of future research is the use of this approach for the triangulation of volumes rather than points. We plan to expand this method such that general objects (which can be modelled by convex polyhedra) can be reconstructed efficiently: cameras often observe not a single point per object but rather a silhouette. We aim to generalise the proposed method in such a way as to enable an efficient polyhedral reconstruction of 3D objects using the ℓ_∞ norm.

References

[1] S. Agarwal, N. Snavely, and S. Seitz. Fast algorithms for l-infinity problems in multiview geometry. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008. 1, 2, 6

[2] M. Byrd, K. Josephson, and K. strm. Fast optimal three view triangulation. In Y. Yagi, S. Kang, I. Kweon, and H. Zha, editors, *Computer Vision ACCV 2007*, volume 4844 of *Lecture Notes in Computer Science*, pages 549–559. Springer Berlin Heidelberg, 2007. 1, 2

[3] Z. Dai, Y. Wu, F. Zhang, and H. Wang. A novel fast method for l-infinity problems in multiview geometry. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Computer Vision ECCV 2012*, volume 7576 of *Lecture Notes in Computer Science*, pages 116–129. Springer Berlin Heidelberg, 2012. 1, 2

[4] O. Enqvist, C. Olsson, and F. Kahl. Stable structure from motion using rotational consistency. Technical report, 2010. 6

[5] A. Eriksson and M. Isaksson. Pseudoconvex proximal splitting for l-infinity problems in multiview geometry. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 4066–4073, June 2014. 1, 2, 6

[6] R. Hartley, F. Kahl, C. Olsson, and Y. Seo. Verifying global minima for l2 minimization problems in multiple view geometry. *International Journal of Computer Vision*, 101(2):288–304, 2013. 1

[7] R. Hartley and P. Sturm. Triangulation. pages 146–157, 1997. 2

[8] R. I. Hartley. Cheirality invariants. In *Proceedings of DARPA Image Understanding Workshop*, pages 745–753, 1993. 3

[9] R. I. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146 – 157, 1997. 1, 2

[10] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 1

[11] F. Kahl, S. Agarwal, M. Chandraker, D. Kriegman, and S. Belongie. Practical global optimization for multiview geometry. *International Journal of Computer Vision*, 79(3):271–284, 2008. 2

[12] F. Kahl and R. Hartley. Multiple-view geometry under the l-infinity norm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(9):1603–1617, Sept. 2008. 1, 2

[13] M. A. Lourakis and A. Argyros. SBA: A Software Package for Generic Sparse Bundle Adjustment. *ACM Trans. Math. Software*, 36(1):1–30, 2009. 2, 6

[14] D. H. Nga and K. Yanai. A spatio-temporal feature based on triangulation of dense surf. In *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*, pages 420–427, Dec 2013. 1

[15] C. Olsson and O. Enqvist. Stable structure from motion for unordered image collections. In A. Heyden and F. Kahl, editors, *Image Analysis*, volume 6688 of *Lecture Notes in Computer Science*, pages 524–535. Springer Berlin Heidelberg, 2011. 6

[16] C. Olsson, A. Eriksson, and R. Hartley. Outlier removal using duality. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1450–1457, June 2010. 1, 2

[17] C. Olsson, A. Eriksson, and F. Kahl. Efficient optimization for l-problems using pseudoconvexity. In *Computer Vision,*

2007. ICCV 2007. IEEE 11th International Conference on, pages 1–8, Oct 2007. 1, 2, 3

[18] H. Stewenius, F. Schaffalitzky, and D. Nister. How hard is 3-view triangulation really? In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 686–693 Vol. 1, Oct 2005. 2

[19] J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999. Version 1.05 available from `http://fewcal.kub.nl/sturm`. 6