Estimation of Camera Locations in Highly Corrupted Scenarios: All About that Base, No Shape Trouble

Yunpeng Shi
University of Minnesota
shixx517@umn.edu

Gilad Lerman
University of Minnesota
lerman@umn.edu

Abstract

We propose a strategy for improving camera location estimation in structure from motion. Our setting assumes highly corrupted pairwise directions (i.e., normalized relative location vectors), so there is a clear room for improving current state-of-the-art solutions for this problem. Our strategy identifies severely corrupted pairwise directions by using a geometric consistency condition. It then selects a cleaner set of pairwise directions as a preprocessing step for common solvers. We theoretically guarantee the successful performance of a basic version of our strategy under a synthetic corruption model. Numerical results on artificial and real data demonstrate the significant improvement obtained by our strategy.

1. Introduction

The problem of Structure from Motion (SfM), that is, reconstructing 3D structure from 2D images, is critical in computer vision. The common pipeline for 3D reconstruction consists of the following steps: 1. Matching keypoints among images using SIFT [12]; 2. Computing the essential matrices from the matched image pairs and extracting relative camera rotations [8]; 3. Finding global camera orientations via rotation synchronization and estimating relative camera translations [1, 3, 6, 9, 13, 17]; 4. Estimating camera locations from estimated pairwise directions [1, 2, 4, 5, 6, 7, 15, 16, 17, 21, 22], where a pairwise direction between two cameras is the normalized relative location vector between them; 5. Recovering the 3D structure using bundle adjustment [20]. The key for successful 3D recovery is the accurate estimation of camera parameters, including camera locations and orientations. These parameters can be misestimated due to erroneous keypoint matching, which results in inaccurate estimates of the essential matrices [18]. This paper develops a robust and theoretically-guaranteed strategy for improving camera location estimation from corrupted pairwise directions.

1.1. Previous Works

A variety of camera location solvers have been proposed in the past two decades [18]. The least squares methods [1, 2, 5] are among the earliest solvers. However, these methods are not robust to outliers (namely, maliciously corrupted pairwise directions) and furthermore they typically produce collapsed location estimates. That is, the estimated camera locations are usually clustered around few points. The constrained least squares (CLS) method [21, 22] introduced an anti-collapsed constraint, which makes it more stable to noise but not outliers. The semidefinite relaxation (SDR) solver [17] converts the least squares problem into an SDP formulation with a nonconvex anti-collapse constraint. However, it is not outliers-robust, and its computation is challenging even after convex relaxation. Other solvers include the ℓ_∞ method [15] and the Lie-algebraic averaging method [6], but the ℓ_∞ norm is sensitive to outliers and [6] suffers from convergence to local minima and from sensitivity to outliers.

Recent outlier-robust methods have been proposed for camera location estimation. One class of solvers use outlier detection algorithms as a preprocessing step to improve their subsequent estimator. For the different problem of camera rotation estimation, cycle-consistency constraints were proposed in [15, 24] to remove outlying relative orientation measurements. For camera location recovery, the 1DSfM algorithm was proposed in [23] for removing outlying pairwise directions. It projects the 3D direction vectors onto 1D, reformulates the cycle-consistency constraints as an ordering problem and solves it using a heuristic combinatorial method. However, its convergence to the global minimum is not guaranteed. Another class of methods directly solve robust convex optimization problems and include the least unsquared deviations (LUD) algorithm [16] and the ShapeFit algorithm [7]. Exact recovery guarantees under a certain corruption model were established for ShapeFit and LUD in [7] and [11] respectively. An ADMM-accelerated version of ShapeFit, called ShapeKick, was proposed in [4]. However, it sacrifices accuracy for speed. A robust formulation for estimating the fundamental matrices was presented in [19]. However, it may suffer from convergence to local minima and requires good initialization.

1.2. Contribution of This Work

We propose a novel algorithm for detecting and removing highly corrupted pairwise directions. We use it as a preprocessing step for existing location recovery algorithms. Our method forms a statistic for any pairwise direction between two given cameras. This statistic estimates the average inconsistency of this pairwise direction with any two pairwise directions associated with an additional camera. This inconsistency is based on the

shortest path in S^2 between a direction vector and a base of a spherical triangle. We thus refer to this inconsistency and statistic as All-About-that-Base (AAB). After computing a fast version of the AAB statistic, we remove edges with large statistics and apply a preferable solver. This method is fast and easy to implement, and it can be used as a preprocessing step for any camera location solver. Most importantly, we are able to theoretically guarantee its successful classification on corrupted and uncorrupted edges. We are not aware of any other theoretically-guaranteed algorithm for removing corrupted pairwise direction measurements. We also present an iterative procedure for improving the AAB statistic, so outliers could be identified more accurately. Experiments on synthetic and real data demonstrate significant improvement of camera location accuracy by our proposed method.

2. Setting for Camera Location Estimation

A mathematical setting for camera location estimation assumes n unknown camera locations $\{\boldsymbol{t}_i^*\}_{i\in[n]} \subseteq \mathbb{R}^3$, where $[n] = \{1, 2, ..., n\}$. The ground truth pairwise direction $\boldsymbol{\gamma}_{ij}^*$ between cameras $i, j \in [n]$ is defined by

$$\boldsymbol{\gamma}_{ij}^* = \frac{\boldsymbol{t}_i^* - \boldsymbol{t}_j^*}{\|\boldsymbol{t}_i^* - \boldsymbol{t}_j^*\|}, \tag{1}$$

where $\|\cdot\|$ denotes the Euclidean norm. In practice, one often measures a corrupted pairwise direction $\boldsymbol{\gamma}_{ij}$ between cameras i and j. The mathematical problem assumes possibly corrupted pairwise measurements $\boldsymbol{\gamma}_{ij} \in E$ for some $E \subseteq [n] \times [n]$ and asks to estimate the camera locations $\{\boldsymbol{t}_i^*\}_{i\in[n]}$ up to ambiguous translation and scale. Note that E may not include all the pairs of indices, so that some values can be missing.

In order to establish theoretical guarantees and conduct synthetic data experiments for the AAB procedure, we assume that the true camera locations and corrupted pairwise directions are generated by the following slight modification of the Uniform Corruption Model UC(n, p, q, σ) [16]: Let $V = \{\boldsymbol{t}_i^*\}_{i\in[n]}$ be generated by i.i.d. $N(\mathbf{0}, \mathbf{I}_3)$ and let $G(V, E)$ be a graph generated by the Erdös-Rényi model $G(n,p)$, where p denotes the connection probability among edges. For any $ij \in E$, a corrupted pairwise direction $\boldsymbol{\gamma}_{ij}$ is generated by

$$\boldsymbol{\gamma}_{ij} = \begin{cases} \boldsymbol{v}_{ij}, & \text{w.p. } q; \\ \frac{\boldsymbol{\gamma}_{ij}^* + \sigma\boldsymbol{\epsilon}_{ij}}{\|\boldsymbol{\gamma}_{ij}^* + \sigma\boldsymbol{\epsilon}_{ij}\|}, & \text{w.p. } 1-q, \end{cases} \tag{2}$$

where $0 < q < 1$ is the probability of corruption, $\sigma \geq 0$ is the noise level and $\boldsymbol{v}_{ij}$, $\boldsymbol{\epsilon}_{ij}$ are independently drawn from a uniform distribution on S^2. The UC model of [16] assumes instead that $\boldsymbol{\epsilon}_{ij}$ are i.i.d. $N(\mathbf{0}, \mathbf{I}_3)$. We have noticed similar numerical results for data generated from both models, however, our theory described below is easier to state and verify under the uniform assumption.

3. Statistics for Corruption Reduction

We describe a statistic that may distinguish corrupted edges. It uses the geometric notion of cycle-consistency of uncorrupted edges. Cycle-consistency measures were used in [15, 23, 24] as criteria for outlier removal. For location recovery, the cycle-consistency of 3 vectors $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \boldsymbol{\gamma}_3 \in S^2$ refers to the existence of $\lambda_1, \lambda_2, \lambda_3 > 0$ such that

$$\lambda_1\boldsymbol{\gamma}_1 + \lambda_2\boldsymbol{\gamma}_2 + \lambda_3\boldsymbol{\gamma}_3 = 0. \tag{3}$$

One may easily observe that the pairwise directions $\boldsymbol{\gamma}_{ij}^*, \boldsymbol{\gamma}_{jk}^*, \boldsymbol{\gamma}_{ki}^*$ are cycle-consistent by substituting in (3) $\lambda_{ij} = \|\boldsymbol{t}_i^* - \boldsymbol{t}_j^*\|$, $\lambda_{jk} = \|\boldsymbol{t}_j^* - \boldsymbol{t}_k^*\|$ and $\lambda_{ki} = \|\boldsymbol{t}_k^* - \boldsymbol{t}_i^*\|$. However, if any of the three vectors is randomly corrupted, the consistency constraint is most probably violated. Thus, we may define a certain cycle-inconsistency measure that indicates the underlying corruption level.

Section 3.1 describes a basic measure of inconsistency of a given pairwise direction with respect to 2 other pairwise directions, where the 3 directions result from 3 unknown locations. It is referred to as the AAB inconsistency. A formula for efficiently computing it is proposed at the end of this section. Section 3.2 uses these inconsistencies to define the naive AAB statistic of a given pairwise direction that is used to remove corrupted edges. Section 3.3 discusses the iteratively reweighted AAB (IR-AAB) statistic, which aims to further improve the accuracy of naive AAB in removing corrupted edges. At last, Section 3.4 discusses some issues regarding practical implementation of naive AAB and IR-AAB.

3.1. AAB Inconsistency and Formula

We define the cycle-consistency region of $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2 \in S^2$ as $\Omega(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2) = \{\boldsymbol{\gamma} \in S^2 : \boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \boldsymbol{\gamma} \text{ are cycle-consistent}\}$. We denote by d_g the great-circle distance, i.e., the length of the shortest path on S^2. The AAB inconsistency of $\boldsymbol{\gamma}_3 \in S^2$ with respect to $\boldsymbol{\gamma}_1$ and $\boldsymbol{\gamma}_2$ is defined by

$$\begin{aligned} I_{AAB}(\boldsymbol{\gamma}_3; \boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2) &= d_g(\boldsymbol{\gamma}_3, \Omega(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2)) \\ &= \min_{\boldsymbol{\gamma}\in\Omega(\boldsymbol{\gamma}_1,\boldsymbol{\gamma}_2)} d_g(\boldsymbol{\gamma}_3, \boldsymbol{\gamma}). \end{aligned} \tag{4}$$

Figure 1 shows that $I_{AAB}(\boldsymbol{\gamma}_3; \boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2)$ is the smallest angle needed to rotate $\boldsymbol{\gamma}_3$ so that $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \boldsymbol{\gamma}_3$ are cycle-consistent.

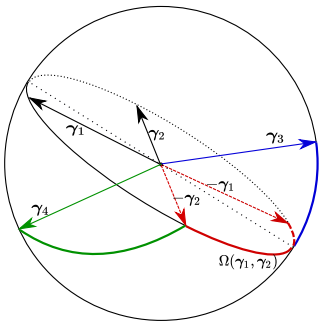

Figure 1. Clarification of the AAB Inconsistency. The red arc is the cycle-consistency region $\Omega(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2)$. Indeed, it follows from (3) that the points in $\Omega(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2)$ are linear combinations in S^2 with positive coefficients of $-\boldsymbol{\gamma}_1$ and $-\boldsymbol{\gamma}_2$. The AAB inconsistency $I_{AAB}(\boldsymbol{\gamma}_3; \boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2)$ is the distance in S^2 of $\boldsymbol{\gamma}_3$ from $\Omega(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2)$ and is the length of the blue arc. Similarly, $I_{AAB}(\boldsymbol{\gamma}_4; \boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2)$ is the length of the green arc.

The following formula for computing the AAB inconsistency is crucial for efficient implementation of the algorithms described below. Its proof appears in the supplemental material. For $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \boldsymbol{\gamma}_3 \in S^2$, $x = \boldsymbol{\gamma}_1^T\boldsymbol{\gamma}_3$, $y = \boldsymbol{\gamma}_2^T\boldsymbol{\gamma}_3$, $z = \boldsymbol{\gamma}_1^T\boldsymbol{\gamma}_2$ and

$a = I(x<yz)\cdot I(y<xz)$, where I is the indicator function,

$$I_{AAB}(\gamma_3;\gamma_1,\gamma_2) = \cos^{-1}\left(a\cdot\frac{x^2+y^2-2xyz}{1-z^2}+(a-1)\min(x,y)\right). \quad (5)$$

3.2. The Naive AAB Statistic

We initially define the naive AAB statistic of an edge $ij\in E$ as the average of the AAB inconsistencies $I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki})$ over the set $C_{ij}=\{k\in[n]: ik\in E \text{ and } jk\in E\}$. That is,

$$S^{\text{initial}}_{AAB}(ij)=\frac{1}{|C_{ij}|}\sum_{k\in C_{ij}} I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki}). \quad (6)$$

We use it as an indication for the corruption level of γ_{ij} and thus remove the edges with largest AAB statistics. Note that the AAB formula in (5) enables computation of the naive AAB statistic through vectorization instead of using a loop, and thus allows efficient coding in programming languages with an effective linear algebra toolbox. However, the average over C_{ij} can be costly and we thus advocate using a small random sample from C_{ij} of size s, where the default value of s is 50. We summarize this basic procedure of computing the AAB statistic, $S^{(0)}_{AAB}$, in Algorithm 1.

Algorithm 1 Computation of the Naive AAB Statistic

Input: $\{\gamma_{ij}\}_{ij\in E}$: pairwise directions, s: number of samples
  for each $ij\in E$ **do**
    S_{ij} = s random samples with replacement from C_{ij}
    $S^{(0)}_{AAB}(ij)=\frac{1}{s}\sum_{k\in S_{ij}} I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki})$
  end for
Output: Naive AAB statistic $\left\{S^{(0)}_{AAB}(ij)\right\}_{ij\in E}$

3.3. Iteratively Reweighted AAB

The naive AAB statistic may suffer from unreliable AAB inconsistencies when the corruption level q is high. Specifically, for an uncorrupted direction γ_{ij}, its AAB inconsistency with respect to γ_{jk} and γ_{ki} can be unreasonably high if either γ_{jk} or γ_{ki} is severely corrupted. Moreover, if many adjacent edges of ij are corrupted, then the naive AAB statistic of this edge may not accurately measure its corruption level. The main issue is not the misleading effect of neighboring edges, but the fact that only such edges are considered and relevant information from other edges is not incorporated. To overcome this issue, the iteratively reweighted AAB (IR-AAB) statistic computes a weighted mean of AAB inconsistencies and iteratively updates these weights. This results in propagation of global information from other non-neighboring edges to edge ij.

Initially, the IR-AAB procedure computes the naive AAB statistic. The reweighting strategy of IR-AAB tries to reduce the weights of $I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki})$ when either ki or kj are highly corrupted. In order to do this, at each iteration the AAB inconsistencies $I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki})$ involving suspicious edges are penalized by the reweighting function $\exp(-\tau^{(t)}x)$. The number x is the maximal value of the reweighted AAB statistics computed in previous iteration for edges ik and kj. The parameter $\tau^{(t)}$ increases iteratively and depends on the initial maximal and minimal values of inconsistencies, denoted by M and m. Figure 2 illustrates the reweighting functions with $M=1$, $m=0$ and 10 iterations. The use of slowly-decreasing reweighting functions in the first iterations ensures that only the most unreliable AAB inconsistencies are ignored. As the data is iteratively purified, the AAB inconsistencies involving "good" edges are weighted more and more. We remark that increasing $\tau^{(t)}$ corresponds to focusing more on "good" edges and ignoring more "suspicious" edges. The details of computing the IR-AAB statistic are described in Algorithm 2.

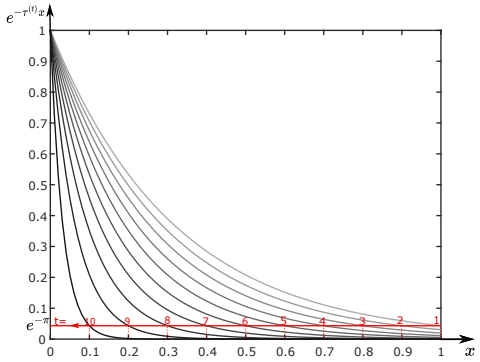

Figure 2. Demonstration of the reweighting function $\exp(-\tau^{(t)}x)$ used in IR-AAB. Here, $t\in[10]$ and the rate of decrease is $\tau^{(t)}=\pi/(1.1-0.1t)$, which increases with t. The labels on the x-axis are of the points $x_t=1.1-0.1t$, $1\le t\le 10$. At each iteration t, $\exp(-\tau^{(t)}x)<e^{-\pi}\approx 0.04$ for $x>x_t$. The red line separates for each curve the values in $[0,x_t]$ and $[x_t,1]$. Therefore, $\exp(-\tau^{(t)}x)$ gives little weight to points in $[x_t,1]$.

Algorithm 2 Computation of the IR-AAB statistic

Input: $\{\gamma_{ij}\}_{ij\in E}$: pairwise directions, s: number of samples, T: number of iterations
  Compute S_{ij}, $S^{(0)}_{AAB}(ij)$ $\forall ij\in E$ by Algorithm 1
  $M=\max_{ij\in E,k\in S_{ij}} I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki})$
  $m=\min_{ij\in E,k\in S_{ij}} I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki})$
  $L=(M-m)/T$
  for $t=1:T$ **do**
    $\tau^{(t)}=\pi/M$
    $M=M-L$
    for $ij\in E$ and $k\in S_{ij}$ **do**
      $w^{(t)}_{ij,k}=\exp\left(-\tau^{(t)}\max\left\{S^{(t-1)}_{AAB}(ki),S^{(t-1)}_{AAB}(jk)\right\}\right)$
      $w^{(t)}_{ij,k}=w^{(t)}_{ij,k}/\sum_{k\in S_{ij}} w^{(t)}_{ij,k}$
      $S^{(t)}_{AAB}(ij)=\sum_{k\in S_{ij}} w^{(t)}_{ij,k} I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki})$
    end for
  end for
Output: IR-AAB statistic: $\left\{S^{(T)}_{AAB}(ij)\right\}_{ij\in E}$

Note that IR-AAB alternatively updates the weights using the AAB statistics and then updates the AAB statistics using the new

weights. This way better weights can reduce the effect of highly corrupted edges so that the updated AAB statistics measures more accurately the corruption level of edges. Similarly, better estimates of the corruption level by the AAB statistics provide more accurate weights, which emphasize the more relevant edges. In the special practical case of repetitive patterns (e.g., due to identical windows), this procedure can help in identifying corrupted edges that are self-consistent with each other.

At last we comment that the failure mode for any AAB procedure is when there are no outliers, so the task of identifying corruptions is ill-posed. This can also happen when the noise magnitude is enormous and outliers are not distinguishable.

3.4. Numerical Considerations

As mentioned earlier, implementations for naive AAB and IR-AAB may avoid loops and use instead vectorization due to the AAB formula. An efficient Matlab code will be provided in the future supplemental webpage. For naive AAB and IR-AAB we recommend using $s = 50$ as default, and we applied this value in all of our experiments. For IR-AAB we recommend and implement the default value $T = 10$.

We note that the computational complexity of naive AAB is $O(s \cdot |E|)$, where $|E|$ is the number of edges. In general, for dense graphs the complexity is $O(s \cdot n^2)$, but for sparser graphs the complexity decreases, e.g., for sparse Erdös-Rényi graphs with $p \ll 1$, the complexity is $O_P(s \cdot p \cdot n^2)$ since $\mathbb{E}[|E|] = n(n-1)p/2$. The computational complexity of IR-AAB is also $O(s \cdot |E|)$. While IR-AAB is iterated $T = 10$ times, its main computation is due to the initial application of naive AAB, which requires the computation of the AAB inconsistencies. On the other hand the weight computations in the subsequent iterations is much cheaper. Therefore in practice, the computational complexity of naive AAB and IR-AAB are truly comparable.

For synthetic data, we demonstrate in Section 5 that a threshold on the naive AAB and IR-AAB statistics can be chosen by their corresponding histograms. We also demonstrate performance with differently chosen thresholds via ROC curves. The histograms of real data are not so simple, and thus in this case we keep half of the edges with the lowest values of the corresponding statistic. We have noticed that the less edges we keep the higher accuracy we obtain for location estimation. However, extremely low threshold results in limited number of camera locations. Demonstrations of other thresholds appear in the supplemental material.

4. Theoretical Guarantees for Outliers Removal

We show that the naive AAB statistic can be used for near-perfect separation of corrupted and uncorrupted edges. Given pairwise directions generated on an edge set E by the uniform corruption model, we denote by E_g the uncorrupted edges, namely, all edges $ij \in E$ such that $\boldsymbol{\gamma}_{ij} = \boldsymbol{\gamma}^*_{ij}$. We denote the rest of edges in E by E_b. The theorem below states that under the uniform corruption model with sufficiently small corruption probability and noise level, the naive AAB statistic is able to perfectly separate E_g as well as a large portion of E_b.

Theorem 4.1. *There exist absolute positive constants C_0, C such that for any $\epsilon \in [0,1]$ and for pairwise directions randomly generated by the uniform corruption model UC(n,p,q,σ) with $n = \Omega(1/pq\epsilon)$, $np^2(1-q)^2 \geq C_0 \log n$ and $q + \sigma < C\epsilon/\sqrt{\log n}$, there exists a set $E' \subseteq E_b$ such that $|E'| \geq (1-\epsilon)|E_b|$ and with probability $1 - O(n^{-5})$,*

$$\min_{ij \in E'} \mathbb{E}[S^{(0)}_{AAB}(ij)] > \max_{ij \in E_g} \mathbb{E}[S^{(0)}_{AAB}(ij)]. \tag{7}$$

The theorem can be extended to other synthetic models. For instance, the assumption in the UC model that the locations are sampled from a Gaussian distribution can be generalized to any distribution that generates "c-well-distributed locations", which are explained in Section 4.1.1. One can show that a compactly supported distribution with continuous and positive density satisfies this criterion with an absolute constant c (unlike the Gaussian case) and consequently the theorem may have the weaker assumption: $q + \sigma < C\epsilon$. The uniform noise assumption in the UC model of this paper can be directly extended to any compactly supported distribution. For Gaussian noise, one needs to slightly modify the theorem so the RHS of (7) is maximized over a sufficiently large subset of E_g (similarly to the LHS w.r.t. E_b).

4.1. Proof of Theorem 4.1

After reviewing preliminary results and notation in Section 4.1.1, Section 4.1.2 describes the main part of the proof. It starts with stating two essential bounds: An upper bound on the expectation of $S^{(0)}_{AAB}(ij)$ when $ij \in E_g$ and a lower probabilistic bound on the expectation of $S^{(0)}_{AAB}(ij)$ when $ij \in E_b$. The upper bound is stated in (9) and later proved in Section 4.1.3. The lower bound is stated in (10) and later proved in proved in Section 4.1.4. While the upper bound is uniform over $ij \in E_g$, the lower bound depends on the corruption level of each edge $ij \in E_b$. However, there is an absolute bound which holds within a large subset of E_b. We show that the uniform upper bound is lower than the absolute lower bound and thus conclude that with high probability the expected values of $S^{(0)}_{AAB}(ij)$ when $ij \in E_g$ are separated from the expected values of $S^{(0)}_{AAB}(ij)$ when ij is in a large subset of E_b.

4.1.1 Preliminaries

We first summarize some properties of the AAB inconsistency:
(i) $I_{AAB}(\boldsymbol{\gamma}_3;\boldsymbol{\gamma}_1,\boldsymbol{\gamma}_2) \in [0,\pi]$ $\forall\, \boldsymbol{\gamma}_1,\boldsymbol{\gamma}_2,\boldsymbol{\gamma}_3 \in S^2$.
(ii) $I_{AAB}(\boldsymbol{\gamma}_3;\boldsymbol{\gamma}_1,\boldsymbol{\gamma}_2) = 0$ iff $\boldsymbol{\gamma}_1,\boldsymbol{\gamma}_2,\boldsymbol{\gamma}_3$ are cycle-consistent.
(iii) The AAB inconsistency is rotation-invariant. That is, for any rotation $\boldsymbol{R}$: $I_{AAB}(\boldsymbol{\gamma}_3;\boldsymbol{\gamma}_1,\boldsymbol{\gamma}_2) = I_{AAB}(\boldsymbol{R}\boldsymbol{\gamma}_3;\boldsymbol{R}\boldsymbol{\gamma}_1,\boldsymbol{R}\boldsymbol{\gamma}_2)$.

We denote by $U(S^2)$ the uniform distribution on S^2 and define $Z := I_{AAB}(\boldsymbol{z};\boldsymbol{x},\boldsymbol{y})$, where $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ i.i.d. $\sim U(S^2)$. For $x \in [0,\pi]$, let $f(x) := \mathbb{E}[I_{AAB}(\boldsymbol{v}_2(x);\boldsymbol{v}_1,\boldsymbol{v})|\boldsymbol{v} \sim U(S^2)]$, where $\boldsymbol{v}_1 = (-1,0,0)^T$ and $\boldsymbol{v}_2(x) = (\cos x, \sin x, 0)^T$. The following property is proved in the supplemental material.

Lemma 4.1. *If $x \in [0,\pi]$, then $f(x) = \frac{1}{2}(x + \sin x)$.*

We will use the following definition and Lemma of [7].

Definition 4.1 (Definition 2 of [7]). *Let $G=G(V,E)$ be a graph with vertices $V=\{\boldsymbol{t}_i\}_{i=1}^n\subseteq\mathbb{R}^3$. For $\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^3$, $c>0$ and $A\subseteq V$, we say that A is c-well-distributed with respect to $(\boldsymbol{x},\boldsymbol{y})$ if the following holds for any $\boldsymbol{h}\in\mathbb{R}^3$:*

$$\frac{1}{|A|}\sum_{\boldsymbol{t}\in A}\|P_{\mathrm{Span}\{\boldsymbol{t}-\boldsymbol{x},\boldsymbol{t}-\boldsymbol{y}\}^\perp}(\boldsymbol{h})\|\geq c\|P_{(\boldsymbol{x}-\boldsymbol{y})^\perp}(\boldsymbol{h})\|. \quad (8)$$

We say that V is c-well-distributed along G if for all distinct $1\leq i,j\leq n$, the set $S_{ij}=\{\boldsymbol{t}_k\in V: ik,jk\in E(G)\}$ is c-well-distributed with respect to $(\boldsymbol{t}_i,\boldsymbol{t}_j)$.

Lemma 4.2 (Lemma 18 of [7]). *Assume that $V=\{\boldsymbol{t}_i\}_{i=1}^n$ is i.i.d. generated by $N(\mathbf{0},\mathbf{I}_3)$ and the graph $G(V,E)$ is generated by the Erdös-Rényi model $G(n,p)$. There exist absolute positive constants C_0,C_1 such that if $np^2\geq C_0\log n$, then with probability $1-n^{-5}$, the set V is $C_1/\sqrt{\log n}$-well-distributed along G.*

4.1.2 The Main Part of the Proof

Let $e_{ij}:=\measuredangle(\boldsymbol{\gamma}_{ij},\boldsymbol{\gamma}^*_{ij})$ denote the corruption level of edge $ij\in E$. We later prove in Sections 4.1.3 and 4.1.4 respectively the following two inequalities. The first one holds for any fixed $ij\in E$:

$$\mathbb{E}[S^{(0)}_{AAB}(ij)|ij\in E_g]\leq\pi\sigma(1-q)^2+\pi q(1-q)+q^2\mathbb{E}[Z]. \quad (9)$$

The second one holds with probability $1-n^{-5}$ for all $ij\in E$:

$$\mathbb{E}[S^{(0)}_{AAB}(ij)|ij\in E_b]\geq(1-q)^2\left[\frac{C'}{\sqrt{\log n}}\min(e_{ij},\pi-e_{ij})-\frac{\pi}{2}\sigma\right]+q^2\mathbb{E}[Z]. \quad (10)$$

We conclude the proof by assuming these inequalities. Recall that there exists an absolute constant C such that

$$q+\sigma<\frac{C\epsilon}{\sqrt{\log n}}. \quad (11)$$

Multiplying both sides of (11) by $3\pi(1-q)^2/2$, noting that for n sufficiently large $1-q\geq 1-C\epsilon/\sqrt{\log n}>2/3$ and thus $3q(1-q)^2/2>q(1-q)$ and setting $C'=6C$ yield

$$\pi\left[q(1-q)+\frac{3}{2}\sigma(1-q)^2\right]<(1-q)^2\frac{C'}{\sqrt{\log n}}\cdot\frac{\pi\epsilon}{4}. \quad (12)$$

Clearly (12) can be rewritten as

$$\pi\sigma(1-q)^2+\pi q(1-q)<(1-q)^2\left[\frac{C'}{\sqrt{\log n}}\cdot\frac{\pi\epsilon}{4}-\frac{\pi}{2}\sigma\right]. \quad (13)$$

Combining (9), (10) and (13) results in

$$\max_{ij\in E_g}\mathbb{E}[S^{(0)}_{AAB}(ij)]<\min_{\substack{ij\in E_b\\ \min(e_{ij},\pi-e_{ij})>\frac{\pi\epsilon}{4}}}\mathbb{E}[S^{(0)}_{AAB}(ij)]. \quad (14)$$

Let $E'=\{ij\in E_b:\min(e_{ij},\pi-e_{ij})>\pi\epsilon/4\}$. Since e_{ij} is i.i.d.$\sim U[0,\pi]$, $X_{ij}:=I(ij\notin E')$ is a Bernoulli random variable with mean $\mu=\epsilon/2$. Applying Chernoff bound [14] yields

$$\Pr\Big(\sum_{ij\in E_b}X_{ij}>2|E_b|\mu\Big)<\exp(-\Omega(|E_b|\mu)). \quad (15)$$

That is, with probability $1-\exp\left(-\Omega(n^2pq\epsilon)\right)$, $|E'|>(1-\epsilon)|E_b|$. Since $n=\Omega(1/pq\epsilon)$ this probability is sufficiently high. Thus, Theorem 4.1 is concluded if (9) and (10) are correct.

4.1.3 Proof of Inequality (9)

We investigate the distribution of $I_{AAB}(\boldsymbol{\gamma}_{ij};\boldsymbol{\gamma}_{jk},\boldsymbol{\gamma}_{ki})$ for fixed $ij\in E_g$ and $k\in C_{ij}$ in the following 3 complementary cases:

Case 1. $jk,ki\in E_g$.

In this case, $\boldsymbol{\gamma}_{ij}=\boldsymbol{\gamma}^*_{ij}+\boldsymbol{v}_{ij}$, $\boldsymbol{\gamma}_{jk}=\boldsymbol{\gamma}^*_{jk}+\boldsymbol{v}_{jk}$ and $\boldsymbol{\gamma}_{ki}=\boldsymbol{\gamma}^*_{ki}+\boldsymbol{v}_{ki}$, where $\boldsymbol{v}_{ij}=(\boldsymbol{\gamma}^*_{ij}+\sigma\boldsymbol{\epsilon}_{ij})/\|\boldsymbol{\gamma}^*_{ij}+\sigma\boldsymbol{\epsilon}_{ij}\|-\boldsymbol{\gamma}^*_{ij}$, $\boldsymbol{\epsilon}_{ij}\sim U(S^2)$ and $\boldsymbol{v}_{jk}$ and $\boldsymbol{v}_{ki}$ are defined in the same way. We note that if $\sigma=0$, then the AAB inconsistency is 0 in the current case. If $\sigma>0$, then since $\|\boldsymbol{\epsilon}_{ij}\|=1$ the AAB inconsistency is bounded as follows:

$$\begin{aligned}X^g_{ij}(k)&:=I_{AAB}(\boldsymbol{\gamma}_{ij};\boldsymbol{\gamma}_{jk},\boldsymbol{\gamma}_{ki})\\&=d_g\big(\boldsymbol{\gamma}^*_{ij}+\boldsymbol{v}_{ij},\Omega(\boldsymbol{\gamma}^*_{jk}+\boldsymbol{v}_{jk},\boldsymbol{\gamma}^*_{ki}+\boldsymbol{v}_{ki})\big)\\&\leq d_g\big(\boldsymbol{\gamma}^*_{ij}+\boldsymbol{v}_{ij},\boldsymbol{\gamma}^*_{ij}\big)+d_g\big(\boldsymbol{\gamma}^*_{ij},\Omega(\boldsymbol{\gamma}^*_{jk}+\boldsymbol{v}_{jk},\boldsymbol{\gamma}^*_{ki}+\boldsymbol{v}_{ki})\big)\\&\leq d_g\big(\boldsymbol{\gamma}^*_{ij}+\boldsymbol{v}_{ij},\boldsymbol{\gamma}^*_{ij}\big)+d_g\big(\boldsymbol{\gamma}^*_{ij},\Omega(\boldsymbol{\gamma}^*_{jk},\boldsymbol{\gamma}^*_{ki})\big)\\&\quad+\max\big(d_g(\boldsymbol{\gamma}^*_{jk},\boldsymbol{\gamma}^*_{jk}+\boldsymbol{v}_{jk}),d_g(\boldsymbol{\gamma}^*_{ki},\boldsymbol{\gamma}^*_{ki}+\boldsymbol{v}_{ki})\big)\\&\leq\frac{\pi}{2}\sigma+0+\frac{\pi}{2}\sigma=\pi\sigma. \quad (16)\end{aligned}$$

Case 2. Either $jk\in E_g$ or $ki\in E_g$, but not both in E_g.

We assume WLOG that $jk\in E_g$ and $ki\in E_b$. According to the uniform corruption model, $\boldsymbol{\gamma}_{ki}\sim U(S^2)$, $\boldsymbol{\gamma}_{jk}=\boldsymbol{\gamma}^*_{jk}+\boldsymbol{v}_{jk}$, $\boldsymbol{\gamma}_{ij}=\boldsymbol{\gamma}^*_{ij}+\boldsymbol{v}_{ij}$. For any indices ijk, let θ_{ijk} denotes the angle between $\boldsymbol{\gamma}_{ij}$ and $\boldsymbol{\gamma}_{kj}$. By choosing appropriate rotation matrix $\boldsymbol{R}$,

$$\begin{aligned}Y^g_{ij}(k)&:=I_{AAB}(\boldsymbol{\gamma}_{ij};\boldsymbol{\gamma}_{jk},\boldsymbol{\gamma}_{ki})=I_{AAB}(\boldsymbol{R}\boldsymbol{\gamma}_{ij};\boldsymbol{R}\boldsymbol{\gamma}_{jk},\boldsymbol{R}\boldsymbol{\gamma}_{ki})\\&=I_{AAB}(\boldsymbol{v}_2(\theta_{ijk});\boldsymbol{v}_1,\boldsymbol{v}), \quad (17)\end{aligned}$$

where $\boldsymbol{v}_1$ and $\boldsymbol{v}_2(\theta_{ijk})$ were defined in Section 4.1.1 and $\boldsymbol{v}\sim U(S^2)$. Lemma 4.1 and the fact that $f(x)\in[0,\pi/2]$ for $x\in[0,\pi]$ imply the inequality

$$\mathbb{E}[Y^g_{ij}(k)]=\mathbb{E}_{\theta_{ijk}}[f(\theta_{ijk})]\leq\frac{\pi}{2}. \quad (18)$$

Case 3. $jk,ki\in E_b$

Let $Z^g_{ij}(k)$ be defined as follows with distribution equivalent formulations that use an arbitrary rotation $\boldsymbol{R}$ and $\boldsymbol{x},\boldsymbol{y}\sim U(S^2)$:

$$\begin{aligned}Z^g_{ij}(k)&:=I_{AAB}(\boldsymbol{\gamma}_{ij};\boldsymbol{\gamma}_{jk},\boldsymbol{\gamma}_{ki})\overset{d}{=}I_{AAB}(\boldsymbol{R}\boldsymbol{\gamma}_{ij};\boldsymbol{R}\boldsymbol{\gamma}_{jk},\boldsymbol{R}\boldsymbol{\gamma}_{ki})\\&\overset{d}{=}I_{AAB}(\boldsymbol{R}\boldsymbol{\gamma}_{ij};\boldsymbol{x},\boldsymbol{y}). \quad (19)\end{aligned}$$

Since $\boldsymbol{R}$ is arbitrary, $Z^g_{ij}(k)$ is independent of $\boldsymbol{\gamma}_{ij}$ and for $\boldsymbol{z}\sim U(S^2)$

$$Z^g_{ij}(k):=I_{AAB}(\boldsymbol{\gamma}_{ij};\boldsymbol{\gamma}_{jk},\boldsymbol{\gamma}_{ki})\overset{d}{=}I_{AAB}(\boldsymbol{z};\boldsymbol{x},\boldsymbol{y})=Z. \quad (20)$$

At last, combining (16), (18) and (20) with probabilities $(1-q)^2$, $2q(1-q)$ and q^2 for each case respectively yields (9).

4.1.4 Proof of Inequality (10)

We investigate the distribution of $I_{AAB}(\boldsymbol{\gamma}_{ij};\boldsymbol{\gamma}_{jk},\boldsymbol{\gamma}_{ki})$ for fixed $ij\in E_b$ and $k\in C_{ij}$ in the following 3 complementary cases:

Case 1. $jk,ki\in E_g$. Observe that

$$\begin{aligned}I_{AAB}(\boldsymbol{\gamma}_{ij};\boldsymbol{\gamma}^*_{jk},\boldsymbol{\gamma}^*_{ki})&=\min_{v\in\Omega(\boldsymbol{\gamma}^*_{jk},\boldsymbol{\gamma}^*_{ki})}d_g(\boldsymbol{\gamma}_{ij},v)\\\geq\min_{v\in\mathrm{Span}\{\boldsymbol{\gamma}^*_{jk},\boldsymbol{\gamma}^*_{ki}\}}d_g(\boldsymbol{\gamma}_{ij},v)&\geq\min_{v\in\mathrm{Span}\{\boldsymbol{\gamma}^*_{jk},\boldsymbol{\gamma}^*_{ki}\}}\|\boldsymbol{\gamma}_{ij}-v\|\\&=\|P_{\mathrm{Span}\{\boldsymbol{t}^*_k-\boldsymbol{t}^*_i,\boldsymbol{t}^*_k-\boldsymbol{t}^*_j\}^\perp}(\boldsymbol{\gamma}_{ij})\| \quad (21)\end{aligned}$$

and

$$
\begin{aligned}
X_{ij}^b(k) &:= I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki}) \\
&= d_g\big(\gamma_{ij},\Omega(\gamma_{jk}^*+\boldsymbol{v}_{jk},\gamma_{ki}^*+\boldsymbol{v}_{ki})\big) \\
&\geq d_g\big(\gamma_{ij},\Omega(\gamma_{jk}^*,\gamma_{ki}^*)\big) \\
&\quad -\max\big(d_g(\gamma_{jk}^*,\gamma_{jk}^*+\boldsymbol{v}_{jk}),d_g(\gamma_{ki}^*,\gamma_{ki}^*+\boldsymbol{v}_{ki})\big) \\
&= I_{AAB}(\gamma_{ij};\gamma_{jk}^*,\gamma_{ki}^*) \\
&\quad -\max\big(d_g(\gamma_{jk}^*,\gamma_{jk}^*+\boldsymbol{v}_{jk}),d_g(\gamma_{ki}^*,\gamma_{ki}^*+\boldsymbol{v}_{ki})\big) \\
&\geq \|P_{\mathrm{Span}\{\boldsymbol{t}_k^*-\boldsymbol{t}_i^*,\boldsymbol{t}_k^*-\boldsymbol{t}_j^*\}^\perp}(\gamma_{ij})\| - \frac{\pi}{2}\sigma. \qquad (22)
\end{aligned}
$$

Denote $C_{ij}^g := \{k \in C_{ij} : ki \in E_g, jk \in E_g\}$ so that $k \in C_{ij}^g$. Note that the underlying corruption model implies that $G(V,E_g)$ is an Erdös-Rényi graph $G(n,p(1-q))$. By combining the assumption $np^2(1-q)^2 > C_0 \log n$ and Lemma 4.2, we obtain that the set of vertices V is $C_1/\sqrt{\log n}$-well-distributed along $G(V,E_g)$ for some absolute constant C_1 with high probability. This fact and (22) imply that with probability $1-n^{-5}$

$$
\begin{aligned}
&\frac{1}{|C_{ij}^g|}\sum_{k\in C_{ij}^g} I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki}) \\
&\geq \frac{1}{|C_{ij}^g|}\sum_{k\in C_{ij}^g} \|P_{\mathrm{Span}\{\boldsymbol{t}_k^*-\boldsymbol{t}_i^*,\boldsymbol{t}_k^*-\boldsymbol{t}_j^*\}^\perp}(\gamma_{ij})\| - \frac{\pi}{2}\sigma \\
&\geq \frac{C_1}{\sqrt{\log n}}\|P_{(\boldsymbol{t}_i^*-\boldsymbol{t}_j^*)^\perp}\gamma_{ij}\| - \frac{\pi}{2}\sigma = \frac{C_1}{\sqrt{\log n}}\|P_{\gamma_{ij}^{*\perp}}\gamma_{ij}\| - \frac{\pi}{2}\sigma \\
&\geq \frac{C_1\pi}{2\sqrt{\log n}}\min(e_{ij},\pi-e_{ij}) - \frac{\pi}{2}\sigma. \qquad (23)
\end{aligned}
$$

Case 2. Either $jk \in E_g$ or $ki \in E_g$, but not both in E_g.
Let $Y_{ij}^b(k) := I_{AAB}(\gamma_{ij};\gamma_{jk},\gamma_{ki})$. The arguments used for the estimates of case 2 of Section 4.1.3 and the fact that $f(x) \geq 0$ imply that $\mathbb{E}[Y_{ij}^b(k)] \geq 0$.
Case 3. $jk,ki \in E_b$
This case is exactly the same as case 3 of Section 4.1.3 and we thus use (20) for $\boldsymbol{z} \sim U(S^2)$.

At last, combining the estimates of the 3 cases with respective probabilities $(1-q)^2$, $2q(1-q)$ and q^2 yields (10).

5. Experiments on Synthetic Data

We first illustrate the ability of the statistics obtained by naive AAB, IR-AAB and 1DSfM [23] to separate corrupted and uncorrupted edges for a special synthetic dataset. The dataset was randomly generated by the uniform corruption model with $n=200$, $p=0.5$, $q=0.2$ and $\sigma=0$. Figure 3 first shows the three statistics' values of edges as a function of their corruption levels. These corruption levels are measured by the angles of the corresponding pairwise directions with the uncorrupted pairwise directions. We first note that 1DSfM may assign zero values to corrupted edges, unlike naive AAB and IR-AAB, and has the largest variance per corruption level. We also note that IR-AAB assigns negligible values to uncorrupted points, unlike naive AAB and 1DSfM, and has the lowest variance at low corruption levels. The figure also shows the histograms of the statistics for both corrupted and uncorrupted points. Since the 1DSfM statistic (which is referred to in [23] as inconsistency) obtains zero values for both corrupted and uncorrupted edges, it is hard to separate the whole histogram into two modes. On the other hand, naive AAB and IR-AAB can be nicely separated into two modes for this and other synthetic examples. For IR-AAB, but not naive AAB, this separation exactly recovers the uncorrupted edges in this particular example.

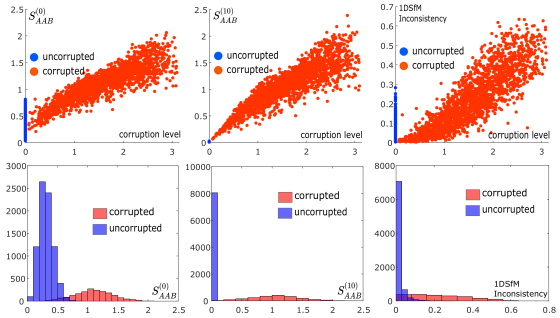

Figure 3. Demonstration of corruption identification for a synthetic dataset by naive AAB, IR-AAB and 1DSfM. The dataset was generated by the uniform corruption model UC(200,0.5,0.2,0). The 3 columns of subfigures correspond to naive AAB, IR-AAB and 1DSfM respectively. The subfigures in the first row show the correlation of the computed statistics (on y-axis) with the corruption level (on x-axis). Edges with no corruption are blue and the rest are red. The subfigures in the second row are the histograms of computed statistics for both corrupted and uncorrupted edges.

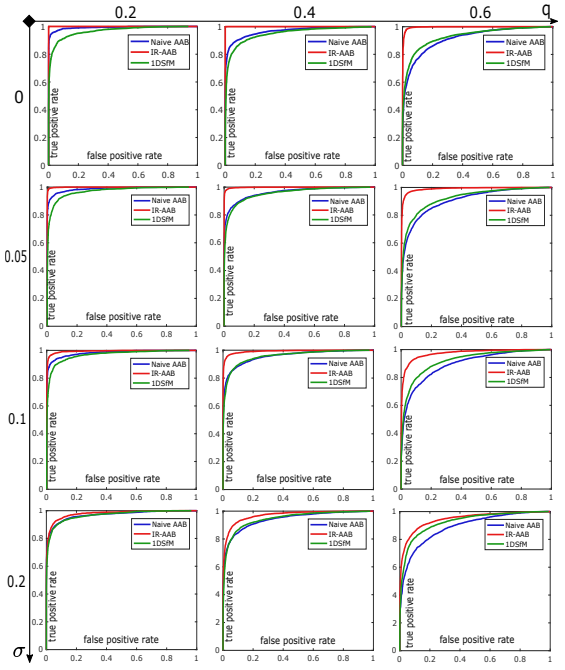

Figure 4. ROC curves for corruption detection of naive AAB, IR-AAB and 1DSfM with varying corruption and noise levels.

Next we use ROC curves to diagnose the ability of naive AAB, IR-AAB and 1DSfM to detect corrupted edges in a similar synthetic data with varying percentages of corrupted edges

and noise levels. The datasets were randomly generated by the uniform corruption model with $n = 200$, $p = 0.5$, $q = 0.2$, 0.4, 0.6 and $\sigma = 0$, 0.05, 0.1 and 0.2. For each statistic and choice of parameters, we assign 1000 equidistant thresholds between the largest and smallest values of this statistic, compute the true and false positive rates for recognizing uncorrupted points with values of this statistic above each threshold, and plot the corresponding ROC curve. We remark that the edges $ij \in E$ are recognized as corrupted when $\measuredangle(\boldsymbol{\gamma}_{ij}, \boldsymbol{\gamma}^*_{ij}) > \sin^{-1}(\sigma)$. The resulting ROC curves are shown in Figure 4, where a larger area under the ROC curve corresponds to better classification performance.

We note that classification based on IR-AAB consistently outperforms that of naive AAB and 1DSfM. Moreover, IR-AAB has a clear advantage over naive AAB and 1DSfM at low and moderate noise levels ($\sigma = 0$, 0.05, 0.1) among all levels of tested corruption. However, IR-AAB requires a certain portion of pairwise directions to be accurately estimated, and it thus does not significantly improve over the other two methods at high noise levels ($\sigma = 0.2$). Naive AAB works well when the corruption and noise levels are relatively low. However, due to the misleading effect of corrupted neighboring edges, it may misclassify uncorrupted edges when the overall corruption or noise level is high ($q = 0.6$ or $\sigma = 0.2$). The performance of 1DSfM is not competitive. Indeed, it may frequently misclassify edges even at low corruption and noise levels, since it may converge to local extrema and also the 1D projection loses information.

6. Experiments on Real Data

We consider real datasets and compare the improvement obtained by preprocessing current camera location solvers with naive AAB, IR-AAB and 1DSfM. We use the 14 datasets from [23]. For each dataset, we exactly follow the pipeline suggested by [16] for estimating camera orientations and pairwise directions. Given the estimated pairwise directions from [16], naive AAB, IR-AAB and 1DSfM are applied separately to delete 50% of the edges with the highest corresponding statistics. Different choices of 10% and 90% deleted edges are demonstrated in the supplemental material. Since the graph may not be parallel rigid after deleting edges, we extract its maximal parallel rigid component using a procedure suggested in [10]. We then apply to this component the following three different camera location solvers: LUD [16] with IRLS implementation, CLS [21, 22] with interior point method and ShapeFit [7] with ADMM implementation [4]. We remark that although 50% of edges are removed, the number of locations in the maximal parallel rigid graph is still close to the original graph. For faster implementation of LUD, only a subset of the Piccadilly dataset with 500 locations is used. For each dataset, each of the 3 statistics, and each of the 3 camera location solvers, we compute average and median distance (in meters) of the estimated camera locations to the ground truth locations[1]. The latter ones are provided by [23]. The experimental results are recorded in Table 1.

These results show significant improvement of IR-AAB for all three camera location solvers. In particular, IR-AAB works best with LUD and CLS. For example, IR-AAB with LUD outperforms naive AAB and 1DSfM with LUD on 10 out of the 14 datasets in terms of both mean and median errors. For 2 additional datasets, IR-AAB with LUD still improves over LUD. For the two remaining datasets, Ellis Island and Gendarmenmarkt, which contain highly inaccurate pairwise directions, none of the three statistics significantly improve any of the solvers. We also note that while LUD is superior to CLS, after applying IR-AAB, both algorithms are comparable. Furthermore, CLS with IR-AAB outperforms plain LUD. We observe that 1DSfM outperforms IR-AAB on a few datasets when using ShapeFit. However, 1DSfM with ShapeFit is worse than plain ShapeFit on 6 other datasets. The inconsistent results of ShapeFit are due to its instability. Indeed, its formulation has a very weak constraint that cannot avoid collapsed solutions in the presence of highly corrupted pairwise directions and when, in particular, some locations have low degrees. On the other hand, both LUD and CLS have a very strong constraint, which avoids collapsed solutions. Note that the preprocessing step results in a large component of the original graph with a possibly different topology than the original graph and thus ShapeFit may be more sensitive to the resulting subgraph, especially if it has some vertices with low degrees not present in the original graph. Due to this sensitivity, none of the preprocessing methods consistently outperforms the other ones when using ShapeFit. We remark that the instability of ShapeFit can be observed from the large variation of its estimation error using different outlier-removing methods and different datasets.

Figure 5 illustrates the improvement of the three preprocessing algorithms (naive AAB, IR-AAB and 1DSfM) over the three solvers (LUD, CLS and ShapeFit). The improvement is measured by the following formula:

$$\text{Improvement} = \frac{e_{\text{before}} - e_{\text{after}}}{e_{\text{before}}} \cdot 100\%, \tag{24}$$

where e_{before} is the mean/median error of estimated camera locations on the whole graph by the given solver without preprocessing and e_{after} is the mean/median error of the same solver after removing 50% of edges by the given preprocessing algorithm. The two datasets with highly inaccurate pairwise directions, Ellis Island and Gendarmenmarkt, were removed. The first three subfigures indicate results of mean error for each solver separately and the last subfigure demonstrates the averaged mean and median errors result among the 12 remaining datasets. It is evident that IR-AAB has the best overall performance in improving the three solvers. On the other hand, 1DSfM has the worst performance. For example, IR-AAB succeeds in improving LUD's performance with average mean-error rate of 38% and it consistently reduces the estimation error of LUD on all of these datasets. On the other hand, 1DSfM has average mean-error improvement rate for LUD of 5.7%, whereas on five datasets preprocessing by 1DSfM increases the estimation error of LUD. For comparison, naive AAB has average mean-error improvement rate for LUD of 26.5%, whereas on two datasets preprocessing by naive AAB increases the estimation error of LUD. For all of the 3 statistics, the overall improvement of preprocessing with CLS is more significant than preprocessing with LUD and ShapeFit. Indeed the averaged mean and median improvement rates of CLS

[1]For each solver, the unknown scale and shift are estimated by least squares minimization with respect to the ground truth data.

Algorithms	LUD [16]								CLS [21, 22]								ShapeFit [7]							
	None		N-AAB		IR-AAB		1DSfM		None		N-AAB		IR-AAB		1DSfM		None		N-AAB		IR-AAB		1DSfM	
Dataset	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$	$\tilde{e}$	$\hat{e}$
Alamo	0.47	1.74	0.38	0.92	**0.36**	**0.85**	0.38	1.3	1.35	2.79	0.39	0.93	**0.37**	**0.69**	0.44	1.44	0.44	1.83	0.38	0.92	**0.36**	**0.85**	0.38	2.82
Madrid Metropolis	1.84	5.94	1.28	3.57	**1.21**	**3.53**	1.46	5.79	7.1	11.2	1.48	5.28	**1.26**	**3.44**	2.73	3.59	14	27.3	1.51	17.8	**1.22**	**7.64**	4.61	29.58
Montreal N.D.	0.56	1.22	0.4	0.61	**0.39**	**0.59**	0.53	1.2	0.9	1.79	**0.4**	**0.6**	0.41	**0.6**	0.69	1.86	0.58	3.25	**0.39**	0.63	**0.39**	**0.58**	0.61	4.08
Notre Dame	0.29	0.85	0.26	0.6	**0.24**	**0.51**	0.28	1	1.05	2.12	0.36	0.86	**0.27**	**0.55**	0.61	1.47	0.24	0.96	0.23	0.58	**0.22**	**0.53**	0.24	1.27
NYC Library	2.43	6.95	0.95	2.89	**0.69**	**2.24**	1.83	5.64	5.3	8.51	1.89	4.51	**0.72**	**2.52**	4.49	7.31	13.3	14.3	0.85	5.69	**0.66**	**2.23**	13.3	14.2
Piazza Del Popolo	1.66	5.28	1.12	4.03	**0.91**	**1.54**	0.94	1.95	3.42	6.46	1.22	4.31	**0.98**	**1.57**	1.47	2.57	1.48	6.81	1.09	4.07	**0.89**	**1.51**	1	5.74
Piccadilly	2.02	3.87	1.37	3.07	**1.19**	**2.69**	2.12	3.95	3.64	5.42	1.56	3.28	**1.23**	**2.42**	3.5	5.11	14.2	13.4	5.72	14.4	11.6	13.3	**2.09**	**6.39**
Roman Forum	2.21	8.33	1.74	7.28	**1.62**	**7.13**	3.4	10.1	6.2	12.4	3.11	9.24	**2.56**	**8.58**	6.62	15.3	26.7	41	**1.53**	**12.7**	7.46	17.7	26.9	33.2
Tower of London	4.03	17.9	2.41	4.79	**2.33**	**4.36**	2.83	15.8	16	27	2.6	4.87	**2.36**	**4.34**	12.6	24.8	2.41	16.9	2.34	4.74	**2.27**	**3.92**	2.48	20.1
Union Square	7.57	11.7	**7.24**	**11.2**	7.3	11.3	7.89	12.9	8.03	12.5	**7.39**	**11.7**	7.84	**11.7**	8.54	13.6	12.9	19	**12.3**	**18.6**	12.5	18.8	13.1	19.2
Vienna Cathedral	7.26	13.1	6.86	14.9	**4.21**	**12.7**	9.05	17.4	9.59	**13.7**	10	14.8	**8.45**	13.9	8.62	15.6	28.6	36.6	28.5	36.5	28.5	**36.4**	**27.6**	**36.4**
Yorkminster	2.51	5.26	**1.61**	6.74	1.62	4.91	2	**3.69**	5.95	8.72	2.8	6.95	**2.29**	**6.36**	4.76	6.89	19.9	28.4	2.35	10.9	2.03	14.6	**1.65**	**4.51**
Ellis Island	22	**22.4**	23.5	23.7	24.7	24.6	**21.7**	22.5	**20.9**	**22**	23.4	23.6	25.3	24.7	22.1	22.4	26.7	27.7	26.5	**27.6**	26.6	27.8	**26.4**	**27.6**
Gendermenmarkt	17.5	**38.8**	15.1	40.9	**15**	41.3	17.1	40.6	20.7	**40.9**	19.4	43.1	**18.3**	42.1	19.2	42.3	32.8	51.7	32.7	52.1	32.5	52.1	**32.4**	**51.5**

Table 1. Comparison of naive AAB, IR-AAB and 1DSfM for improving 3 location solvers (LUD, CLS, ShapeFit) using 14 datasets from [23]. The median and mean distance from the estimated camera locations to the ground truth (provided in [23]) are denoted by $\tilde{e}$ and $\hat{e}$ respectively.

when preprocessing by IR-AAB is more than 50%. This is not surprising as CLS is not robust to corruption. For ShapeFit, the average improvements over the mean errors when preprocessing with naive AAB, IR-AAB and 1DSfM are 42.4%, 50.4% and 2.9% respectively. However, when considering each individual dataset, IR-AAB and naive AAB may not consistently outperform 1DSfM due to the instability of ShapeFit discussed above.

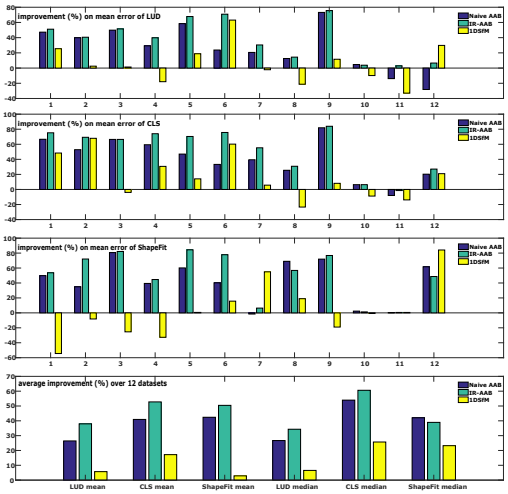

Figure 5. Percentage of improvement of location estimation by preprocessing 3 solvers with the 3 statistics. The first 3 subfigures illustrate the mean error improvement for LUD, CLS and ShapeFit respectively. Numbers 1-12 on the horizontal axis are the indices of the first 12 datasets in Table 1. Negative number of improvement rate corresponds to increase of the estimation error after removing edges. The last subfigure illustrates the averaged mean and median improvement of the 3 statistics over the first 12 datasets when preprocessing the three solvers by the three statistics.

We report the computational speed of the algorithms on the largest dataset: Roman Forum, which has 967 locations. While Piccadilly has 2226 locations, it was run with 500 locations to ease the computational time for LUD and for extracting the maximal parallel rigid component. The computations were performed on a machine with 2.5GHz Intel i5 quad core processors and 8GB memory. The total time needed to compute 1DSfM, naive AAB and IR-AAB is 2, 5 and 8 seconds respectively. For comparison, the total time to run CLS, ShapeFit and LUD is 8, 8 and 160 seconds respectively. We expect the runtime of ADMM for LUD to be comparable to that of ShapeFit. The slowest component was finding the maximal parallel rigid component. For Roman Forum, it took 550 seconds, while it took less than 200 seconds for the other datasets.

7. Conclusion

We proposed the AAB statistic for estimating the underlying corruption level on camera pairwise directions. We improved this estimation by incorporating a careful reweighting strategy. We further established theoretical guarantee on the accuracy of the non-reweighted statistic, i.e., naive AAB, for detecting corrupted edges when the corruption and noise levels are sufficiently low. The experiments on both synthetic data and real data show the significant advantage of applying the reweighting strategy with the AAB statistic. Applying our method as a preprocessing step significantly improves the performance of current camera location solvers.

This work suggests several interesting future projects. First of all, we believe that a similar strategy can be developed for improving camera orientation estimation. Second of all, we are interested in theoretically guaranteeing the reweighting strategy for segmenting corrupted and uncorrupted edges. Third of all, an interesting direction for future work is to study and provide guarantees for synthetic models that more realistically mirror real scenarios. At last, we find it important to develop a faster method for extracting the maximal parallel rigid graph so that the total runtime can be significantly reduced.

Acknowledgement

This work was supported by NSF award DMS-14-18386. We thank Soumyadip Sengupta, Thomas Goldstein, Tal Amir and Paul Hand for providing codes and real data. We also thank the anonymous reviewers and Tyler Maunu for helpful comments on an earlier version of this manuscript.

References

[1] M. Arie-Nachimson, S. Z. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, and R. Basri. Global motion estimation from point matches. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission, Zurich, Switzerland, October 13-15, 2012*, pages 81–88, 2012. 1

[2] M. Brand, M. E. Antone, and S. J. Teller. Spectral solution of large-scale extrinsic camera calibration as a graph embedding problem. In *Computer Vision - ECCV 2004, 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part II*, pages 262–273, 2004. 1

[3] A. Chatterjee and V. M. Govindu. Efficient and robust large-scale rotation averaging. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 521–528, 2013. 1

[4] T. Goldstein, P. Hand, C. Lee, V. Voroninski, and S. Soatto. Shapefit and shapekick for robust, scalable structure from motion. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VII*, pages 289–304, 2016. 1, 7

[5] V. M. Govindu. Combining two-view constraints for motion estimation. In *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), 8-14 December 2001, Kauai, HI, USA*, pages 218–225, 2001. 1

[6] V. M. Govindu. Lie-algebraic averaging for globally consistent motion estimation. In *2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004), 27 June - 2 July 2004, Washington, DC, USA*, pages 684–691, 2004. 1

[7] P. Hand, C. Lee, and V. Voroninski. Shapefit: Exact location recovery from corrupted pairwise directions. *Communications on Pure and Applied Mathematics*, 71(1):3–50, 2018. 1, 4, 5, 7, 8

[8] A. Harltey and A. Zisserman. *Multiple view geometry in computer vision (2. ed.)*. Cambridge University Press, 2006. 1

[9] R. I. Hartley, K. Aftab, and J. Trumpf. L1 rotation averaging using the weiszfeld algorithm. In *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, pages 3041–3048, 2011. 1

[10] R. Kennedy, K. Daniilidis, O. Naroditsky, and C. J. Taylor. Identifying maximal rigid components in bearing-based localization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*, pages 194–201, 2012. 7

[11] G. Lerman, Y. Shi, and T. Zhang. Exact camera location recovery by least unsquared deviations. *CoRR*, abs/1709.09683, 2017. 1

[12] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 1

[13] D. Martinec and T. Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007), 18-23 June 2007, Minneapolis, Minnesota, USA*, 2007. 1

[14] M. Mitzenmacher and E. Upfal. *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge university press, 2005. 5

[15] P. Moulon, P. Monasse, and R. Marlet. Global fusion of relative motions for robust, accurate and scalable structure from motion. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 3248–3255, 2013. 1, 2

[16] O. Özyesil and A. Singer. Robust camera location estimation by convex programming. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 2674–2683, 2015. 1, 2, 7, 8

[17] O. Özyesil, A. Singer, and R. Basri. Stable camera motion estimation using convex programming. *SIAM Journal on Imaging Sciences*, 8(2):1220–1262, 2015. 1

[18] O. Özyesil, V. Voroninski, R. Basri, and A. Singer. A survey of structure from motion. *Acta Numerica*, 26:305364, 2017. 1

[19] S. Sengupta, T. Amir, M. Galun, T. Goldstein, D. W. Jacobs, A. Singer, and R. Basri. A new rank constraint on multi-view fundamental matrices, and its application to camera location recovery. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, Hawaii, USA, June 22-25, 2017*, pages 4798–4806, 2017. 1

[20] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - A modern synthesis. In *Vision Algorithms: Theory and Practice, International Workshop on Vision Algorithms, held during ICCV '99, Corfu, Greece, September 21-22, 1999, Proceedings*, pages 298–372, 1999. 1

[21] R. Tron and R. Vidal. Distributed image-based 3-d localization of camera sensor networks. In *Proceedings of the 48th IEEE Conference on Decision and Control, CDC 2009, December 16-18, 2009, Shanghai, China*, pages 901–908, 2009. 1, 7, 8

[22] R. Tron and R. Vidal. Distributed 3-d localization of camera sensor networks from 2-d image measurements. *IEEE Trans. Automat. Contr.*, 59(12):3325–3340, 2014. 1, 7, 8

[23] K. Wilson and N. Snavely. Robust global translations with 1dsfm. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part III*, pages 61–75, 2014. 1, 2, 6, 7, 8

[24] C. Zach, M. Klopschitz, and M. Pollefeys. Disambiguating visual relations using loop constraints. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pages 1426–1433, 2010. 1, 2