# A Closed-Form Solution to Tensor Voting for Robust Parameter Estimation via Expectation-Maximization

Tai-Pang Wu[†‡] Jiaya Jia[‡] Chi-Keung Tang[†]

[†] The Hong Kong University of Science and Technology
[‡] The Chinese University of Hong Kong

**Abstract**

We prove a closed-form solution to second-order Tensor Voting (TV), and employ the resulting structure-aware tensors in Expectation-Maximization (EM). Our new algorithm, aptly called EM-TV, is an efficient and robust method for parameter estimation. Quantitative comparison shows that our method performs better than the conventional second-order TV and other representative techniques in parameter estimation (e.g., fundamental matrix estimation). While its implementation is straightforward, EM-TV will be available as a downloadable software library.

**Keywords**

Tensor voting, expectation-maximization, closed-form solution, parameter estimation.

## 1 Introduction

This paper re-introduces tensor voting (TV) [12] for robust parameter estimation, by developing a new algorithm for computing an optimal structure-aware tensor at a given site in the feature space. When integrated with expectation maximization (EM), our new algorithm, aptly called EM-TV, demonstrates better performance than the conventional second-order tensor voting (first-order augmentations to tensor voting was investigated in [18]).

We performed quantitative studies to demonstrate the remarkable noise robustness of EM-TV compared to the original TV and other representative techniques, using challenging synthetic and real data sets. The technical contributions of this paper are twofold:

- *Closed-form solution to tensor voting.* We first carry out theoretical studies to develop a closed-form solution to second-order tensor voting. This new solution promises a continuous, efficient and exact solution to computing an optimal structure-aware tensor at each site without discrete approximation using precomputed voting fields.

- *Robust and accurate parameter estimation.* By integrating the optimal tensor obtained at each site into an Expectation-Maximization (EM) framework, we can produce accurate parameters by combining the advantages of EM and TV. No random sampling consensus is required which is quite time consuming for a large input data set. Our new algorithm is called EM-TV.

The technical section of this paper consists of two parts: First, we develop a closed-form solution to computing an optimal structure-aware tensor at each site. The main result is Eqn (12). Next, we integrate the structure-aware tensor into a new Expectation-Maximization framework that allows for propagating neighborhood information. This EM-TV framework, where the E-step and the M-step are respectively defined in Eqns (19) and (23), permits robust *and* accurate parameter estimation in any number of dimensions. Overall, Eqns (12), (19), and (23) have more rigorous mathematical foundation and are easier to implement than the second-order TV.

## 2 Related Work

Robust techniques have been used in computer vision for many decades, and an excellent review of the theoretical foundations of robust methods in the context of computer vision can be found in [13].

The Hough transform [8] is a robust voting-based technique operating in a parameter space which is able to extract multiple models from noisy data. The most popular robust methods in computer vision remain the class of random sampling consensus (RANSAC) procedures proposed in the 1980s [4] which have spawned a lot of follow-up work. On the other hand, mean shift [3] has been widely used since its introduction to computer vision for robust feature space analysis. Adaptive mean shift [6] with variable

bandwidth in high dimensions was introduced in texture classification and has since been applied to other vision tasks.

Like RANSAC [4], robust estimators including LMedS [14] and the M-estimator [9] adopted a statistical approach. LMedS, RANSAC and the Hough transform can be expressed as M-estimators with auxiliary scale [13]. The choice of scales and parameters related to the noise level are major issues. Existing work on robust scale estimation used random sampling [17], or operate on different assumptions (e.g., more than 50% of the data should be inliers [15]; inliers have a Gaussian distribution [10]). Rather than using a Gaussian distribution, a two-step method was proposed in [20]: first, a non-Gaussian distribution is used to model inliers where local peaks of density are found by mean shift [3]. This is followed by scale parameter estimation (called TSSE in [20]). Then outliers are rejected using a RANSAC-like algorithm. The projection-based M-estimator (pbM) [2], a recent improvement on the M-estimator, uses a Parzen window for scale estimation, so the scale parameter is automatically found by searching for the normal direction (projection direction) that maximizes the sharpest peak of the density. This does not require an input scale from the user. While these recent methods can tolerate more outliers, most of them still rely on or are based on RANSAC and a great deal of random sampling is required to achieve the desired robustness.

To reject outliers, a multi-pass method using $L_\infty$-norms was proposed to successively detect outliers which are characterized by maximum errors [16].

### 2.1 Expectation Maximization

*Expectation-Maximization* (EM) has been used in handling missing data and identifying outliers [5]. EM has been widely used and shown success in computer vision, and its convergence properties were studied [11]. In essence, EM consists of two steps:

1. **E-Step**. Computing an expected value for the complete data set using the incomplete data and the current estimates of the parameters.

2. **M-Step**. Maximizing the complete data log-likelihood using the expected value of the complete data computed in the E-step.

EM is a powerful inference algorithm, but it is also well-known [5] that: 1) initialization is an issue because EM can get stuck in poor local minima, and 2) treatment of data points with small expected weights requires great care. They should not be regarded as negligible, as their aggregate effect can be quite significant.

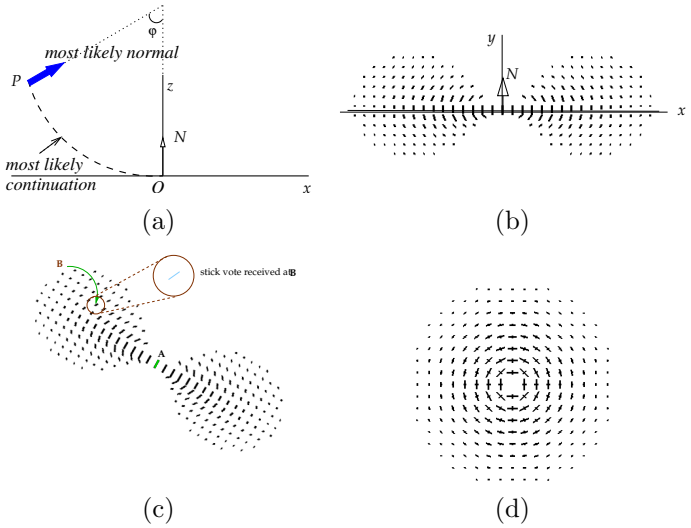



Figure 1: (a) Design of the 2D stick voting field. (b) 2D stick voting field. (c) $A$ casts a stick vote to $B$. (d) 2D ball voting field.

### 2.2 Tensor Voting

*Tensor Voting* [12] is a robust computational approach for grouping and segmentation. The approach is non-iterative, and has been applied to and demonstrated very good results in a variety of vision problems. In essence, in tensor voting, input data points are first represented as second-order symmetric *tensors*, which communicate among each other via a *voting algorithm* that uses a set of precomputed *dense voting fields*.

We will derive a *closed-form solution* for the voting algorithm, which provides an optimal means for computing a structure-aware tensor at each site in the feature space. Further, we employ such tensors to initialize the EM algorithm, which is guaranteed to converge [11], to optimize each tensor and parameter. As with EM-based method our focus is not on searching for a global optimum. By incorporating TV to EM, outliers are suppressed, inliers are reinforced, and accurate parameters are estimated.

## 3 Review of Tensor Voting

This section provides a concise review of tensor voting. In tensor voting *tensor* is used for token representation, and *voting* is used for non-iterative

token-token communication. Tensor and voting are related by a *voting field*.

### 3.1 Voting fields for the basic case

Our goal is to encode the smoothness constraint that should effectively determine if a point lies on some smooth surface in the feature space, or it is in fact an outlier. Let us consider a basic case concerning 2D smoothness: *Suppose a point $P$ in the plane is to be connected by a smooth curve to the origin $O$. Suppose also that the normal $\mathbf{N}$ to the curve at $O$ is given. What is the most likely normal direction at $P$?* [12]. Figure 1(a) illustrates the situation.

We claim that the osculating circle connecting $O$ and $P$ is the most likely connection, since it keeps the curvature constant along the hypothesized circular arc. The most likely normal is given by the *normal to the circular arc at $P$* (thick arrow in Figure 1(a)). The length of this normal, which represents the vote strength, is inversely proportional to the arc length $OP$, and also to the curvature of the underlying circular arc.

To encode proximity and smoothness (low curvature), the decay of the field takes the following form, which is a function of $r$, $\kappa$, $c$, and $\sigma$:

$$\eta_{tv}(r, \kappa, \sigma) = e^{-(\frac{r^2 + c\kappa^2}{\sigma^2})} \tag{1}$$

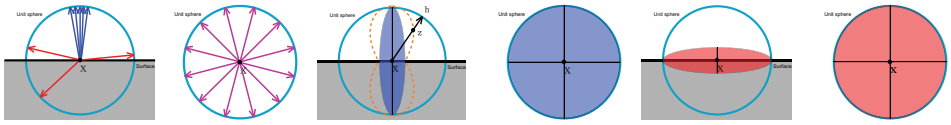
where $r$ is the arc length $OP$, $\kappa$ is the curvature, $c$ is a constant which controls the decay with high curvature. $\sigma$ is the scale of analysis, which determines the effective neighborhood size. Note that $\sigma$ is the only free parameter in the system.

If we consider all points in the 2D space, the whole set of normals thus derived constitutes the 2D stick voting field, Figure 1(b). Each normal is called a *stick vote* $[v_x \; v_y]$, thus defined as

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \eta_{tv}(r, \kappa, \sigma) \begin{bmatrix} \sin\varphi \\ \cos\varphi \end{bmatrix} \tag{2}$$

where $\varphi$ is the angle subtended by the radii of the circle incident at $O$ and $P$.

Given an input token $A$, how to use this field to *cast a stick vote* to another token $B$ for inferring a smooth connection between them? Let us assume the basic case that $A$'s normal is known, as illustrated in Figure 1(c). First, we fix the scale $\sigma$ to determine the size of the voting field. Then, we align the voting field with $A$'s normal (by translation and rotation). If $B$ is within $A$'s voting field neighborhood, $B$ receives a stick vote from the aligned field.



(a) (b) (c) (d) (e) (f)

Figure 2: Inlier/outlier and tensor illustration. (a) The normal votes at a surface point cast by points in $\mathbf{x}$'s neighborhood. Three salient outliers are present. (b) For a non-surface point, there is no preference to any normals. (c) The tensor induced by the normal observations in (a), which is represented by a $d$-D ellipsoid, where $d \geq 2$. The orange curve (dashed curve) represents the variance produced along all possible directions. (d) The tensor sum of the received votes in (b). (e) and (f) correspond to the inverse of (c) and (d), respectively.

### 3.2 Vote collection and representation

How does $B$ collect and interpret all received votes? Other input tokens cast votes to $B$ as well. Denote a received stick vote by $[v_x\ v_y]^T$. The tensor sum of all votes collected by $B$ is accumulated by tensor addition, that is, summing up the covariance matrices consisting of the votes' second order moments: $\begin{bmatrix} \sum v_x^2 & \sum v_x v_y \\ \sum v_y v_x & \sum v_y^2 \end{bmatrix}$. This is a *second order symmetric tensor*. By decomposing this tensor into the corresponding eigensystem, we obtain the most likely normal at $B$, given by the eigenvector associated with the largest eigenvalue.

Geometrically, a second-order symmetric tensor in 2D is equivalent to an ellipse. The major axis gives the general direction. The minor axis indicates the *un*certainty: if the minor axis has zero length, the tensor is a *stick tensor*, representing absolute certainty in one direction given by the major axis. If the length of the minor axis is equal to that of the major axis, the tensor is a *ball tensor*, indicating absolute uncertainty in all directions.

### 3.3 Voting fields for the general case

**Voting fields.** Now, consider the general case that no normal is available at $A$. We want to reduce this case to the basic case. Without any *a priori* assumption, *all* directions are equally likely as the normal direction at $A$. Hence, we rotate the 2D stick voting field at $A$. During the rotation, it casts a large number of stick votes to a given point $B$. All stick votes

received at $B$ are converted into second order moments, and the tensor sum is accumulated.

Then, we compute the eigensystem of the resulting tensor sum to estimate the most likely normal at $B$, given by the direction of the major axis of the resulting tensor inferred at $B$. For implementation efficiency, instead of computing the tensor sum on-the-fly as described above at a given vote receiver $B$, we precompute and store tensor sums due to a rotating stick voting field received at each quantized vote receiver within a neighborhood. We call the resulting field a *2D ball voting field*, which casts *ball tensor votes* in $A$'s neighborhood. Figure 1(d) shows the ball voting field, which stores the eigensystem of the tensor sum at each point. Note the presence of two eigenvectors at each site in Figure 1(d).

**Vote interpretation.** In $d$-dimension we can define similar stick and ball voting fields. After collecting the second order moments of the received votes, they are summed up to produce a second order symmetric tensor. Let us call this tensor sum $\mathbf{K}$ which can be visualized as an ellipsoid, represented by the corresponding eigensystem $\mathbf{K} = \sum_{i=1}^{d} \lambda_i \hat{e}_i \hat{e}_i^T$ where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$ are eigenvalues, and $\hat{e}_1, \hat{e}_2, \cdots, \hat{e}_d$ are corresponding eigenvectors. The eigenvectors determine the orientation of the ellipsoid. The eigenvalues determine the shape of the ellipsoid.

Consider any point in the feature space. While inliers are associated with their corresponding surface normals on the underlying surface[1], outliers, which have no relationship to the surface, can assume arbitrary normal directions (Figure 2(a)–(b)). If it is a surface point, the stick votes received in its neighborhood reinforce each other with a high agreement of tensor votes. The inferred tensor should be stick-like, that is, $\lambda_1 \gg \lambda_2, \cdots, \lambda_d$, indicating certainty in a single direction. On the other hand, an outlier receives a few inconsistent votes, so all eigenvalues are small. Furthermore, if it is a discontinuity or a point junction where several surfaces intersect exactly at a single point, it indicates a high *dis*agreement of tensor votes, indicating no single direction is preferred.

In our EM-TV algorithm, the inverse of $\mathbf{K}$, or $\mathbf{K}^{-1}$ will be used and therefore the most preferred direction is represented by the $\hat{e}_d$ which has the smallest eigenvalue $\lambda_d$ in the corresponding eigen-decomposition. Figure 2(e) and (f) show the corresponding inverse of Figure 2(c) and (d) respectively. Given $\mathbf{K}^{-1}$, the most significant direction is represented by $\hat{e}_d$ which has the smallest eigenvalue $\lambda_d$.

[1]This surface can be the solution to a given vision problem, such as fundamental matrix estimation and triangulation.

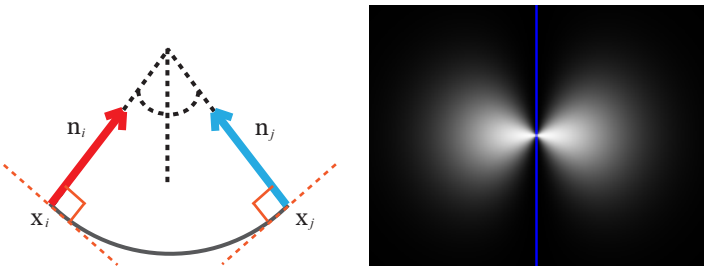



Figure 3: (a) Postulating the normal $\mathbf{n}_i = \mathbf{n}_\theta(\mathbf{x}_i, \mathbf{x}_j)$ at $\mathbf{x}_i$ using the osculating arc between $\mathbf{x}_i$ and $\mathbf{x}_j$, assuming the normal at $\mathbf{x}_j$ is $\mathbf{n}_j$. (b) Plot of Eqn. (3) in 2D.

## 4 Voting Without Voting Fields

In the following we modify the decay function $\eta_{tv}$ to encode the proximity and smoothness constraints (Eqns 3 and 4). While similar in effect to the decay function used in tensor voting, this modified function enables a closed-form solution for tensor voting in the general case (and the basic case as well), without using precomputed discrete voting fields.

Let us revisit the two cases of tensor voting. Refer to Figure 3. Consider two points $\mathbf{x}_i$ and $\mathbf{x}_j$ that are connected by some smooth structure in the solution space.

Case I: Suppose that the normal $\mathbf{n}_j$ at $\mathbf{x}_j$ is known. We want to generate at $\mathbf{x}_i$ a normal observation $\mathbf{n}_i$ so that we can calculate $\mathbf{K}_i$, where $\mathbf{K}_i$ is the second-order symmetric tensor at $\mathbf{x}_i$. $\mathbf{n}_i$ can be derived by fitting an osculating arc between the two points which can be computed in constant time. $\mathbf{K}_i$ is given by $\mathbf{n}_i\mathbf{n}_i^T$ multiplied by $\eta(\mathbf{x}_i, \mathbf{x}_j, \mathbf{n}_j)$ defined as:

$$\eta(\mathbf{x}_i, \mathbf{x}_j, \mathbf{n}_j) = c_{ij}(1 - (\mathbf{r}_{ij}^T\mathbf{n}_j)^2) \tag{3}$$

where

$$c_{ij} = \exp(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{\sigma_d}) \tag{4}$$

Eqn. (4) is an exponential function using Euclidean distance for attenuating the strength based on proximity. $\sigma_d$ is the size of local neighborhood (or the scale parameter).

In Eqn. (3), $\mathbf{r}_{ij}$ is a *unit* vector at $\mathbf{x}_j$ pointing to $\mathbf{x}_i$ and $1 - (\mathbf{r}_{ij}^T\mathbf{n}_j)^2$ is a squared-sine function[2] for attenuating the contribution according to

[2] $\sin^2\rho = 1 - \cos^2\rho$, where $\cos^2\rho = (\mathbf{r}_{ij}^T\mathbf{n}_j)^2$ and $\rho$ is the angle between $\mathbf{r}_{ij}$ and $\mathbf{n}_j$.

curvature. Similar to tensor voting, Eqn. (3) favors nearby neighbors that produce small-curvature connections. A plot of the 2D version of Eqn. (3) is shown in Figure 3(b), where $\mathbf{x}_j$ is located at the center of the image and $\mathbf{n}_j$ is aligned with the blue line. The higher the intensity, the higher the value Eqn. (3) produces at a given pixel location.

CASE II: Next, consider the general case that the normal $\mathbf{n}_j$ at $\mathbf{x}_j$ is unavailable. $\mathbf{K}_j$ at $\mathbf{x}_j$ can be any covariance matrix (typically initialized as an identity matrix). To compute $\mathbf{K}_i$, we enumerate a *complete* set of unit normals $\{\tilde{\mathbf{n}}_{\theta j}\}$ associated with the corresponding length at $\mathbf{x}_j$, indexed by all possible directions $\theta$ and each postulates a normal $\mathbf{n}_\theta(\mathbf{x}_i, \mathbf{x}_j)$ at $\mathbf{x}_i$ under the same proximity and smoothness constraints prescribed by the corresponding osculating arc.

Let $\mathbf{S}_{ij}$ be the second-order symmetric **tensor vote** obtained at $\mathbf{x}_i$ due to the complete set of normals at $\mathbf{x}_j$. We have:

$$\mathbf{S}_{ij} = \int_{\mathbf{N}_{\theta j} \in \nu} \mathbf{n}_\theta(\mathbf{x}_i, \mathbf{x}_j)\mathbf{n}_\theta(\mathbf{x}_i, \mathbf{x}_j)^T \eta(\mathbf{x}_i, \mathbf{x}_j, \tilde{\mathbf{n}}_{\theta j}) d\mathbf{N}_{\theta j} \tag{5}$$

where

$$\mathbf{N}_{\theta j} = \tilde{\mathbf{n}}_{\theta j}\tilde{\mathbf{n}}_{\theta j}^T \tag{6}$$

and $\nu$ is the space containing all possible $\mathbf{N}_{\theta j}$. For example, if $\nu$ is 2D, the complete set of unit normals $\tilde{\mathbf{n}}_\theta$ describes a unit circle. If $\nu$ is 3D, the complete set of unit normals $\tilde{\mathbf{n}}_\theta$ describes a unit sphere.

In tensor voting implementation, Eqn. (5) is typically precomputed as discrete voting fields. Such discrete voting fields (e.g. plate and ball voting fields [12]) are integrated using the 2D stick voting field shown in Figure 1, by rotating and summing the contributions using tensor addition. Although precomputed once, such discrete approximations involve uniform and dense sampling of tensor votes $\tilde{\mathbf{n}}_\theta\tilde{\mathbf{n}}_\theta^T$ in higher dimensions depending on the problem. Next, we will prove a closed-form solution to Eqn. (5), which provides an efficient and optimal solution to computing $\mathbf{K}$ without resorting to discrete and dense sampling.

## 5 A Closed-Form Solution to $\mathbf{S}_{ij}$

Without loss of generality, we consider $\mathbf{x}_i$ and $\mathbf{x}_j$ only. For simplicity of notation, set $\mathbf{r} = \mathbf{r}_{ij}$, $\tilde{\mathbf{n}}_\theta = \tilde{\mathbf{n}}_{\theta j}$ and $\mathbf{N}_\theta = \mathbf{N}_{\theta j}$. Now, using the osculating arc connection, $\mathbf{n}_\theta(\mathbf{x}_i, \mathbf{x}_j)$ can be expressed as

$$\mathbf{n}_\theta(\mathbf{x}_i, \mathbf{x}_j) = (\tilde{\mathbf{n}}_\theta - 2\mathbf{r}\mathbf{r}^T\tilde{\mathbf{n}}_\theta)\tau_\theta \tag{7}$$

where $\tilde{\mathbf{n}}_\theta$ is the unit normal at $\mathbf{x}_j$ with direction $\theta$ and $\tau_\theta$ is the length of $\tilde{\mathbf{n}}_\theta$. Let

$$\mathbf{R} = (\mathbf{I} - 2\mathbf{r}\mathbf{r}^T), \tag{8}$$

where $\mathbf{I}$ is an identity, we can rewrite Eqn. (5) into the following form:

$$\mathbf{S}_{ij} = c_{ij} \int_{\mathbf{N}_\theta \in \nu} \tau_\theta^2 \mathbf{R}\tilde{\mathbf{n}}_\theta \tilde{\mathbf{n}}_\theta^T \mathbf{R}^T (1 - (\tilde{\mathbf{n}}_\theta^T \mathbf{r})^2) d\mathbf{N}_\theta. \tag{9}$$

Following the derivation:

$$\begin{aligned}
\mathbf{S}_{ij} &= c_{ij} \int_{\mathbf{N}_\theta \in \nu} \tau_\theta^2 \mathbf{R}\tilde{\mathbf{n}}_\theta (1 - (\tilde{\mathbf{n}}_\theta^T \mathbf{r})^2) \tilde{\mathbf{n}}_\theta^T \mathbf{R}^T d\mathbf{N}_\theta \\
&= c_{ij}\mathbf{R} \left( \int_{\mathbf{N}_\theta \in \nu} \tau_\theta^2 \tilde{\mathbf{n}}_\theta (1 - \tilde{\mathbf{n}}_\theta^T \mathbf{r}\mathbf{r}^T \tilde{\mathbf{n}}_\theta) \tilde{\mathbf{n}}_\theta^T d\mathbf{N}_\theta \right) \mathbf{R}^T \\
&= c_{ij}\mathbf{R} \left( \int_{\mathbf{N}_\theta \in \nu} \tau_\theta^2 \mathbf{N}_\theta - \tau_\theta^2 \mathbf{N}_\theta \mathbf{r}\mathbf{r}^T \mathbf{N}_\theta d\mathbf{N}_\theta \right) \mathbf{R}^T \\
&= c_{ij}\mathbf{R} \left( \mathbf{K}_j - \int_{\mathbf{N}_\theta \in \nu} \tau_\theta^2 \mathbf{N}_\theta \mathbf{r}\mathbf{r}^T \mathbf{N}_\theta d\mathbf{N}_\theta \right) \mathbf{R}^T
\end{aligned}$$

The integration can be solved by integration by parts. Let $f(\theta) = \tau_\theta^2 \mathbf{N}_\theta$, $f'(\theta) = \tau_\theta^2 \mathbf{I}$, $g(\theta) = \frac{1}{2}\mathbf{r}\mathbf{r}^T \mathbf{N}_\theta^2$ and $g'(\theta) = \mathbf{r}\mathbf{r}^T \mathbf{N}_\theta$ and note that $\mathbf{N}_\theta^q = \mathbf{N}_\theta$ for all $q \in \mathbb{Z}^+$ and $\mathbf{K}_j$, in the most general form, can be expressed as a generic tensor $\int_{\mathbf{N}_\theta \in \nu} \tau_\theta^2 \mathbf{N}_\theta d\mathbf{N}_\theta$. So we have

$$\begin{aligned}
& \int_{\mathbf{N}_\theta \in \nu} \tau_\theta^2 \mathbf{N}_\theta \mathbf{r}\mathbf{r}^T \mathbf{N}_\theta d\mathbf{N}_\theta \\
= \; & [f(\theta)g(\theta)]_{\mathbf{N}_\theta \in \nu} - \int_{\mathbf{N}_\theta \in \nu} f'(\theta) g(\theta) d\mathbf{N}_\theta \\
= \; & \left[ \frac{1}{2} \tau_\theta^2 \mathbf{N}_\theta \mathbf{r}\mathbf{r}^T \mathbf{N}_\theta^2 \right]_{\mathbf{N}_\theta \in \nu} - \frac{1}{2} \int_{\mathbf{N}_\theta \in \nu} \tau_\theta^2 \mathbf{r}\mathbf{r}^T \mathbf{N}_\theta d\mathbf{N}_\theta \\
= \; & \frac{1}{2} \int_{\mathbf{N}_\theta \in \nu} \frac{d}{d\theta} [\tau_\theta^2 \mathbf{N}_\theta \mathbf{r}\mathbf{r}^T \mathbf{N}_\theta^2] d\mathbf{N}_\theta - \frac{1}{2} \mathbf{r}\mathbf{r}^T \mathbf{K}_j \\
= \; & \frac{1}{2} \int_{\mathbf{N}_\theta \in \nu} \tau_\theta^2 \mathbf{r}\mathbf{r}^T \mathbf{N}_\theta^2 + \tau_\theta^2 \mathbf{N}_\theta \mathbf{r}\mathbf{r}^T d\theta - \frac{1}{2} \mathbf{r}\mathbf{r}^T \mathbf{K}_j \\
= \; & \frac{1}{2} \left( \mathbf{r}\mathbf{r}^T \mathbf{K}_j + \mathbf{K}_j \mathbf{r}\mathbf{r}^T - \mathbf{r}\mathbf{r}^T \mathbf{K}_j \right) \\
= \; & \frac{1}{2} \mathbf{K}_j \mathbf{r}\mathbf{r}^T
\end{aligned}$$

Therefore,

$$\mathbf{S}_{ij} = c_{ij} \mathbf{R}\mathbf{K}_j \left( \mathbf{I} - \frac{1}{2}\mathbf{r}\mathbf{r}^T \right) \mathbf{R}^T \tag{10}$$

Replace $\mathbf{r}$ by $\mathbf{r}_{ij}$ such that $\mathbf{R}_{ij} = \mathbf{I} - 2\mathbf{r}_{ij}\mathbf{r}_{ij}^T$ and let $\mathbf{R}'_{ij} = (\mathbf{I} - \frac{1}{2}\mathbf{r}_{ij}\mathbf{r}_{ij}^T)\mathbf{R}_{ij}$, we obtain:

$$\mathbf{S}_{ij} = c_{ij}\mathbf{R}_{ij}\mathbf{K}_j\mathbf{R}'_{ij} \tag{11}$$

As mentioned in Section 3.3, we operate with the inverse of $\mathbf{K}_j$. Let $\mathbf{R''}_{ij} = \mathbf{R}_{ij}(\mathbf{I} + \mathbf{r}_{ij}\mathbf{r}_{ij}^T)$ and also note that $\mathbf{R}_{ij}^{-1} = \mathbf{R}_{ij}$, so the corresponding inverse of $\mathbf{S}$ is:

$$\mathbf{S}'_{ij} = c_{ij}^{-1}\mathbf{R''}_{ij}\mathbf{K}_j^{-1}\mathbf{R}_{ij} \tag{12}$$

where the initial $\mathbf{K}_j$ (or $\mathbf{K}_i$) can be either derived when the input direction is available, or simply assigned as an identity matrix otherwise.

Using Eqn. (12), we can assign at each site $\mathbf{K}_i$ a $\sum_j \mathbf{S}_{ij}$. We will call this tensor sum the *structure-aware* tensor for site $\mathbf{x}_i$, because the computation considers both geometric proximity and smoothness constraints in the presence of its neighbors $\mathbf{x}_j$ as described above.

By using an efficient data structure such as ANN tree [1] to access a constant number of neighbors $\mathbf{x}_j$ of each $\mathbf{x}_i$, where the speed of accessing nearest neighbors can be greatly increased (polylogarithmic), the computation of a structure-aware tensor is efficient. Note that the running time for our closed-from solution is $O(d^3)$, while the running time for TV is $O(u^{d-1})$, where $d$ is the dimension of the space and $u$ is the number of sampling directions for a given dimension. Because of this, TV implementation precomputes and stores the dense tensor fields.

For example, when $d = 3$ and $u = 180$ for high accuracy, our method takes 27 operation units, while TV takes 32400 operation units. Given 1980 points and the same number of neighbors, the time to compute each structure-aware tensor using our method is about 0.0001 second, while it takes about 0.1 second for TV to output the corresponding tensor. The above computation was performed on a laptop computer running on a core duo 2GHz CPU with 2GB RAM.

Now, we are ready to formulate EM-TV which makes use of structure-aware tensors in its optimization process.

# 6 EM-TV

While tensor voting is very able in rejecting outliers, it falls short of producing very accurate parameter estimation, explaining the use of RANSAC in the final parameter estimation step after outlier rejection [19].

This section describes the EM-TV algorithm for optimizing (1) the structure aware tensor $\mathbf{K}$ at each input site, and (2) the parameters of a single plane $\mathbf{v}$ of *any* dimensionality containing the inliers. The extensions to multiple planes and nonlinear model fitting, outlined in the supplemental material, are straightforward extensions based on EM-TV and our future work is to apply this to vision problems.

We first formulate three constraints to be used in EM-TV. These constraints are not mutually exclusive, where knowing the values satisfying one constraint will help computing the values of the others. However, in our case, they are all unknowns, so EM is particularly suitable for their optimization, where the expectation calculation and parameter estimation are solved alternately.

### 6.1 Constraints

**Data constraint** Suppose we have a set of clean data. One necessary objective is to minimize the following for all $\mathbf{x}_i$:

$$||\mathbf{x}_i^T \mathbf{v}|| \tag{13}$$

where $\mathbf{v}$ is the plane (or the model) to be estimated.[3] This is a typical data term that measures the faithfulness of the input data to the fitting plane.

**Orientation consistency** The plane being estimated is defined by the vector $\mathbf{v}$. Recall that the structure-aware tensor $\mathbf{K}_i$ encodes the orientation information at $\mathbf{x}_i$. If $\mathbf{x}_i$ is an inlier, the orientation information encoded by $\mathbf{K}_i$ and $\mathbf{v}$ have to be consistent. That is, the variance $\mathbf{v}^T \mathbf{K}_i^{-1} \mathbf{v}$ produced by $\mathbf{v}$ should be minimal. Otherwise, $\mathbf{x}_i$ might be generated by other models even if it minimizes Eqn. 13. Mathematically, we minimize:

$$||\mathbf{v}^T \mathbf{K}_i^{-1} \mathbf{v}||. \tag{14}$$

**Neighborhood consistency** While the estimated $\mathbf{K}_i$ helps to indicate inlier/outlier information, $\mathbf{K}_i$ has to be consistent with the local structure imposed by its neighbors (supposing they are known). If $\mathbf{K}_i$ is consistent with $\mathbf{v}$ but not the local neighborhood, either $\mathbf{v}$ or $\mathbf{K}_i$ is wrong. In practice, we minimize the following Frobenius norm:

$$||\mathbf{K}_i^{-1} - \mathbf{S}'_{ij}||_F. \tag{15}$$

The above three constraints will interact with each other in the proposed EM algorithm.

[3]Note that, in some cases, the underlying model is represented in this form: $\mathbf{x}_i^T \mathbf{v} - y_i$. But we can re-arrange it into the form given by Eqn. (13).

### 6.2 Objective Function

Define $\mathbf{O} = \{o_i = \mathbf{x}_i | i = 1, ..., N\}$ to be the set of observations. Our goal is to optimize $\mathbf{v}$ and $\mathbf{K}_i^{-1}$ given $\mathbf{O}$. Mathematically, we solve the following objective function:

$$\Lambda^* = \arg\max_{\Lambda} P(\mathbf{O}, \mathbf{R}|\Lambda) \tag{16}$$

where $P(\mathbf{O}, \mathbf{R}|\Lambda)$ is the complete-data likelihood to be maximized, $\mathbf{R} = \{r_i\}$ is a set of hidden states indicating if observation $o_i$ is an outlier ($r_i = 0$) or inlier ($r_i = 1$), and $\Lambda = \{\{\mathbf{K}_i^{-1}\}, \mathbf{v}, \sigma, \sigma_1, \sigma_2\}$ is a set of parameters to be estimated. $\sigma$, $\sigma_1$ and $\sigma_2$ are parameters imposed by some distributions, which will be explained shortly[4]. Our EM algorithm estimates the optimal $\Lambda^*$ by finding the value of the complete-data log likelihood with respect to $\mathbf{R}$ given $\mathbf{O}$ and the current estimated parameters $\Lambda'$:

$$Q(\Lambda, \Lambda') = \sum_{\mathbf{R} \in \psi} \log P(\mathbf{O}, \mathbf{R}|\Lambda) P(\mathbf{R}|\mathbf{O}, \Lambda') \tag{17}$$

where $\psi$ is a space containing all possible configurations of $\mathbf{R}$ of size $N$.

### 6.3 Expectation (E-Step)

In this section, the marginal distribution $p(r_i|o_i, \Lambda')$ will be defined so that we can maximize the parameters in the next step (M-Step) given the current parameters.

If $r_i = 1$, the observation $o_i$ is an inlier and therefore minimizes the first two conditions (Eqns 13 and 14) in Section 6.1, that is, the data and orientation constraints. In both cases, we assume that the data follows a Gaussian distribution (this explains the reason we use $\mathbf{K}_i^{-1}$ instead of $\mathbf{K}_i$). Mathematically, the observation probability of $o_i$ can be modeled by:

$$p(o_i|r_i = 1, \Lambda') \propto \exp(-\frac{||\mathbf{x}_i^T \mathbf{v}||^2}{2\sigma^2}) \exp(-\frac{||\mathbf{v}^T \mathbf{K}_i^{-1} \mathbf{v}||}{2\sigma_1^2}) \tag{18}$$

and so $p(o_i|r_i = 0, \Lambda') \propto 1 - p(o_i|r_i = 1, \Lambda')$. Since we have no prior information about inlier/outlier, we assume that the mixture probability of the observations $p(r_i = 1) = p(r_i = 0)$ equals to a constant $\alpha$ such that we have no bias to any category (inlier/outlier). In this case, $p(o_i|\Lambda') = \alpha p(o_i|r_i = 0, \Lambda') + \alpha(1 - p(o_i|r_i = 1, \Lambda')) = \alpha$.

[4]See Eqn (23).

Define $w_i = p(r_i|o_i, \Lambda')$ to be the probability of $o_i$ being an inlier. Then

$$\begin{aligned} w_i &= \frac{p(o_i|r_i = 1, \Lambda')p(r_i = 1)}{p(o_i|\Lambda')} \\ &= \frac{1}{2\pi\sigma\sigma_1} \exp(-\frac{||\mathbf{x}_i^T\mathbf{v}||^2}{2\sigma^2}) \exp(-\frac{||\mathbf{v}^T\mathbf{K}_i^{-1}\mathbf{v}||}{2\sigma_1^2}) \end{aligned} \tag{19}$$

So, in the E-Step, we compute $w_i$ using Eqn. (19) for all $i$.

### 6.4 Maximization (M-Step)

In the M-Step, we maximize Eqn. (17) using $w_i$ obtained from the E-Step. Since neighborhood information is considered, we model $P(\mathbf{O}, \mathbf{R}|\Lambda)$ as a Markov Random Field (MRF):

$$P(\mathbf{O}, \mathbf{R}|\Lambda) = \prod_i \prod_{j \in \mathcal{G}(i)} p(r_i|r_j, \Lambda)p(o_i|r_i, \Lambda) \tag{20}$$

where $\mathcal{G}(i)$ is the set of neighborhood of $i$. In theory, $\mathcal{G}(i)$ contains all the input points except $i$, since $c_{ij}$ in Eqn. 4 is always non-zero (because of the long tail of the Gaussian distribution). In practice, we can prune away the points in $\mathcal{G}(i)$ where the values of $c_{ij}$ are negligible. This can greatly reduce the size of the neighborhood. Again, using ANN tree [1], the speed of searching for nearest neighbors can be greatly increased.

Let us examine the two terms in Eqn. (20). $p(o_i|r_i, \Lambda)$ has been defined in Eqn. (18). We define $p(r_i|r_j, \Lambda)$ here. Using the third condition mentioned in Eqn. (15), we have:

$$p(r_i|r_j, \Lambda) = \exp(-\frac{||\mathbf{K}_i^{-1} - \mathbf{S}'_{ij}||_F^2}{2\sigma_2^2}) \tag{21}$$

We are now ready to expand Eqn. (17). Since $r_i$ takes only two values (0 or 1), we can rewrite $Q(\Lambda, \Lambda')$ in Eqn. (17) into the following form:

$$\sum_t^{\{0,1\}} \log \left( \prod_i \prod_{j \in \mathcal{G}(i)} p(r_i = t|r_j, \Lambda)p(o_i|r_i = t, \Lambda) \right) P(\mathbf{R}|\mathbf{O}, \Lambda')$$

After expansion,

$$\begin{aligned} Q(\Lambda, \Lambda') &= \sum_i \log(\alpha \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{||\mathbf{x}_i^T \mathbf{v}||^2}{2\sigma^2}))w_i \\ &+ \sum_i \log(\frac{1}{\sigma_1\sqrt{2\pi}} \exp(-\frac{||\mathbf{v}^T \mathbf{K}_i^{-1} \mathbf{v}||}{2\sigma_1^2}))w_i \\ &+ \sum_i \log(\exp(-\frac{||\mathbf{K}_i^{-1} - \mathbf{S}'_{ij}||_F^2}{2\sigma_2^2}))w_i w_j \end{aligned} \tag{22}$$

To maximize Eqn. (22), we set the first derivative of $Q$ with respect to $\mathbf{K}_i^{-1}$, $\mathbf{v}$, $\sigma$, $\sigma_1$ and $\sigma_2$ to zero respectively to obtain the following set of update rules:

$$\begin{aligned} \mathbf{K}_i^{-1} &= \frac{1}{\sum_{j\in\mathcal{G}(i)} w_j}(\sum_{j\in\mathcal{G}(i)} \mathbf{S}'_{ij} w_j - \frac{\sigma_2^2}{2\sigma_1^2} \mathbf{v}\mathbf{v}^T w_i) \\ \mathbf{M}\mathbf{v} &= 0 \quad \text{(solve for } \mathbf{v}\text{)} \\ \sigma^2 &= \frac{\sum_i ||\mathbf{x}_i^T \mathbf{v}||^2 w_i}{\sum_i w_i} \\ \sigma_1^2 &= \frac{\sum_i ||\mathbf{v}^T \mathbf{K}_i^{-1} \mathbf{v}|| w_i}{\sum_i w_i} \\ \sigma_2^2 &= \frac{\sum_i \sum_{j\in\mathcal{G}(i)} ||\mathbf{K}_i^{-1} - \mathbf{S}'_{ij}||_F^2 w_i w_j}{\sum_i w_i} \end{aligned} \tag{23}$$

where $\mathbf{M} = \sum_i \mathbf{x}_i \mathbf{x}_i^T w_i + \frac{\sigma^2}{\sigma_1^2} \sum_i \mathbf{K}_i^{-1} w_i$ and $\mathcal{G}(i)$ is a set of neighbors of $i$. Note that the second rule can be solved by eigen-decomposition. Eqn. (23) constitutes the set of update rules for the M-step.

There is no preference to the execution sequence for Eqn. (23), because they can be executed in parallel. The newly estimated parameters should be cached for this purpose. In each iteration, after the update rules have been executed, we re-project $\mathbf{K}_i^{-1}$ and $\mathbf{v}$ onto the feasible solution space by normalization. Also, $\mathbf{S}'_{ij}$ will be updated with the newly estimated $\mathbf{K}_i^{-1}$.

### 6.5 Implementation and Initialization

In summary, the boxed equations above (Eqns (12), (19) and (23)) are all the equations needed to implement EM-TV and therefore the implementation is straightforward.

It has been noted that initialization is important to an EM algorithm. To initialize EM-TV we set $\sigma_1$ and $\sigma_2$ to be very large values, $\mathbf{K}_i^{-1} = \mathbf{I}$ and $w_i = 1$ for all $i$. $\mathbf{S}'_{ij}$ is initialized to be the inverse of the $\mathbf{S}_{ij}$, computed using the closed-form solution presented in the previous section. These initialization values mean that at the beginning we have no preference to the surface orientation. So all the input points are initially considered as inliers. With such initialization, we execute the second and the third rules in Eqn. (23) in sequence. Note that when the first rule is being executed, the term involving $\mathbf{v}$ is ignored because of the large $\sigma_1$, thus we can obtain $\mathbf{K}_i^{-1}$ for the second rule. After that, we can start executing the algorithm from the E-step. This initialization procedure is used in all the experiments in the following sections.

# 7 Experimental Results

We first perform experiments to demonstrate the robustness of EM-TV. Then we apply EM-TV to parameter estimation.

## 7.1 Robustness of EM-TV

First, quantitative comparison will be studied to evaluate EM-TV with well-known algorithms: RANSAC [4], ASSC [20], and TV [12]. In addition, we also provide the result using the least squares method as a baseline comparison. Second, we apply our method to real data with synthetic noise where the ground truth is available, and perform comparison.

**Inlier-to-outlier (IO) ratio** We will use the *inlier-to-outlier (IO) ratio* to characterize the noise level, which can be described by

$$Z = \frac{R}{R+1} \tag{24}$$

where $Z$ is a noise percentage represented by a real number in $[0, 1]$ and $R$ is the IO ratio. Note the rapid increase in the number of outliers as $Z$ increases from 50% to 99%. Figure 4 shows a plot of $Z = \frac{R}{R+1}$ indicating that it is much more difficult for a given method to handle the same percentage increase in noise as the value of $Z$ increases. For example, according to the plot, it is much more difficult for a given method to tolerate additional 20% noise, when $Z$ is increased from 70% to 90%, than from 50% to 70%. IO ratio is therefore more suitable for studying severely corrupted data.

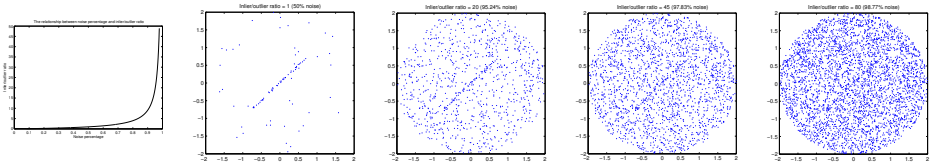


Figure 4: From left to right: Plot of $Z = \frac{R}{R+1}$, and 2D data sets with IO ratios $[1, 20, 45, 80]$ corresponding to noise percentages $[50\%, 95\%, 98\%, 99\%]$. Our EM-TV can tolerate IO ratios $\leq 51$ in this example.

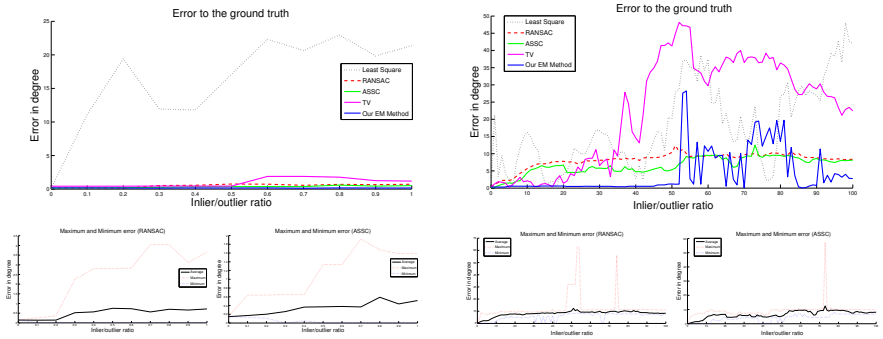



Figure 5: Error plots on SET 1 (IO ratio $= [0.1, 1]$, up to 50% noise) and SET 2 (IO ratio $= [1, 100]$, $\geq 50\%$ noise). Left: for SET 1, all the tested methods demonstrated reliable results except least-square. EM-TV is deterministic and converges quickly, capable of correcting Gaussian noise inherent in the inliers and resulting in the almost-zero error curve. Right: for SET 2, EM-TV still has an almost-zero error curve up to an IO ratio of 51 ($\simeq 98.1\%$ noise). We ran 100 trials in RANSAC and ASSC and averaged the results. The maximum and minimum errors of RANSAC and ASSC are shown below each error plot.

### 7.1.1 Robustness

We generate a set of 2D synthetic data for line fitting, by randomly sampling 44 points from a line within the range $[-1, -1] \times [1, 1]$ where the points are contaminated by Gaussian noise of 0.1 standard deviation. Random noise was added to data with different IO ratios.

The data set is partitioned into two:

- SET 1: IO ratio $\in [0.1, 1]$ with step size 0.1,
- SET 2: IO ratio $\in [1, 100]$ with step size 1.

In other words, the partition is done at 50% noise. Note from Figure 4 that the number of outliers increases rapidly after 50% noise where some data sets with different IO ratios are shown. Outliers were added within a bounding circle of radius 2.

The input scale, which is used in RANSAC, TV and our EM-TV, is estimated automatically by TSSE proposed in [20]. Note that ASSC [20] does not require any input scale.

SET 1 – Refer to the *left* of Figure 5 which shows the error produced by various methods tested on SET 1. The error is measured by the angle between the estimated line and the ground-truth. Except for the least squares method, RANSAC, ASSC, TV and EM-TV performed very well with IO ratios $\leq 1$. For RANSAC and ASSC, all the detected inliers were finally used in parameter estimation. Note that the errors measured for RANSAC and ASSC were the average errors in 100 executions[5], which are supposed to ease the random nature of the two robust methods based on random sampling. Figure 5 also shows the maximum and minimum errors of the two methods after running 100 trials. EM-TV does not have such maximum and minimum error plots, because it is deterministic.

Observe that the errors produced by our method are almost zero in SET 1, while RANSAC and ASSC have error $< 0.6$ degrees, which is still very acceptable.

SET 2 – Refer to the *right* of Figure 5 which shows the result for SET 2, from which we can distinguish the performance of the methods. TV breaks down at IO ratios $\geq 20$. After that, the performance of TV is unpredictable.

[5]It means that we have executed the *complete* algorithm 100 times. In each execution, iterative random sampling was done where the desired probability of choosing at least one sample free from outliers was set to 0.99 (default value). The source code was obtained from P. Kovesi's website http://www.csse.uwa.edu.au/∼pk/Research/MatlabFns/

EM-TV breaks down at IO ratios $\geq 51$, showing greater robustness than TV in this experiment due to the EM parameter fitting procedure.

The performance of RANSAC and ASSC were quite stable where the average errors are within 4 and 7 degrees over the whole spectrum of IO ratios considered. The maximum and minimum errors are shown in the bottom of Figure 5, which shows that they can be very large at times. EM-TV produces almost zero errors with IO ratio $\leq 51$, but then breaks down into unpredictable performance.

From the experiments on SET 1 and SET 2 we conclude that EM-TV is robust up to an IO ratio of 51 ($\simeq$98.1% noise).

#### 7.1.2 Insensitivity to choice of scale

We study the errors produced by EM-TV with different scales $\sigma_d$ (Eqn. (4)), given IO ratio of 10 ($\simeq$91% noise). Even in the presence of many outliers, EM-TV breaks down when $\sigma_d \simeq 0.7$ (the ground-truth $\sigma_d$ is 0.1), which indicates that our method is not sensitive to even large deviations of scale. Note that the scale parameter can sometimes be automatically estimated (by using TSSE in the experiment described above).

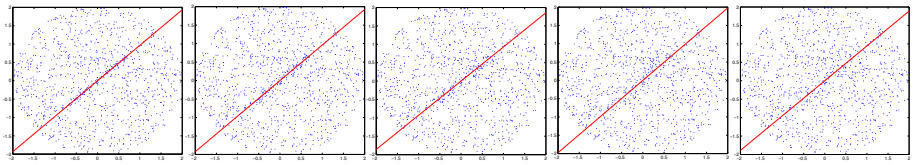
#### 7.1.3 Large measurement errors

In this experiment, we increase the measurement error by increasing the standard deviation (s.d.) from 0.01 to 0.29, while keeping IO ratio equal to 10 and the location of the outliers fixed. Some of the input data sets are depicted in Figure 6, showing that the inliers are less salient as the s.d. increases. A similar experiment was also performed in [13].

Again, we compared our method with least-squares, RANSAC, ASSC and TV, where the setting are the same as the aforementioned 2D line fitting experiments.

According to the error plot in the *top* of Figure 7, the performance of least-squares is very steady, where the angular error is around 17 degrees across the whole s.d. spectrum. This is because the number of outliers is 10 times more than the inliers thus dominating the estimation result. On the other hand, TV is very sensitive to the change of s.d.: when the s.d. is greater than 0.03, the performance is unpredictable. With increasing s.d., the performance of RANSAC and ASSC degrade gracefully while ASSC always outperforms RANSAC. The *bottom* of Figure 7 shows the corresponding maximum and minimum error in 100 executions.

On the other hand, we observe the performance of EM-TV (with $\sigma_d =$



s.d. = 0.03 s.d. = 0.06 s.d. = 0.15 s.d. = 0.2 s.d. = 0.25

Figure 6: Inputs containing various measurement errors, with IO ratio = 10 and fixed outliers location. The estimated parameters obtained using EM-TV are overlayed on the inputs. Notice the line cluster becomes less salient when s.d. = 0.25.

0.05) is extremely steady and accurate when s.d. $< 0.15$. After that, although its error plot exhibits some perturbation, the errors produced are still small and the performance is quite steady compared to other methods.

### 7.2 Fundamental Matrix Estimation

Given $p \geq 8$ correspondence pairs $\mathcal{P} = \{(\mathbf{u}_i, \mathbf{u}'_i) | 8 \leq i \leq p\}$, the goal is to estimate the $3 \times 3$ fundamental matrix $\mathbf{F} = [h]_{a,b}$, where $a, b \in \{1, 2, 3\}$, such that

$$\mathbf{u}'^T_i \mathbf{F} \mathbf{u}_i = 0 \tag{25}$$

for all $i$. $\mathbf{F}$ is of rank 2. Let $\mathbf{u} = (u, v, 1)^T$ and $\mathbf{u}' = (u', v', 1)$, Eqn. (25) can be rewritten into:

$$\mathbf{U}^T_i \mathbf{v} = 0 \tag{26}$$

where

$$\begin{aligned} \mathbf{U} &= (uu', uv', u, vu', vv'v, u', v', 1)^T \\ \mathbf{v} &= (h_{11}, h_{21}, h_{31}, h_{12}, h_{22}, h_{32}, h_{13}, h_{23}, h_{33})^T \end{aligned}$$

Noting that Eqn. (26) is a simple plane equation, if we can detect and handle the noise in the feature space, Eqn. (26) should enable us to produce a good estimation.

Since $\mathbf{F}$ is defined up to a scaling factor, to avoid the trivial solution $\mathbf{v} = 0$, many existing methods impose the hard constraint that $||\mathbf{v}|| = 1$. In EM-TV, no special treatment is needed: this constraint is automatically given by the second rule in Eqn. (23) where eigen-decomposition is used to solve for $\mathbf{v}$. Finally, we apply the method in [7] to obtain a rank-2

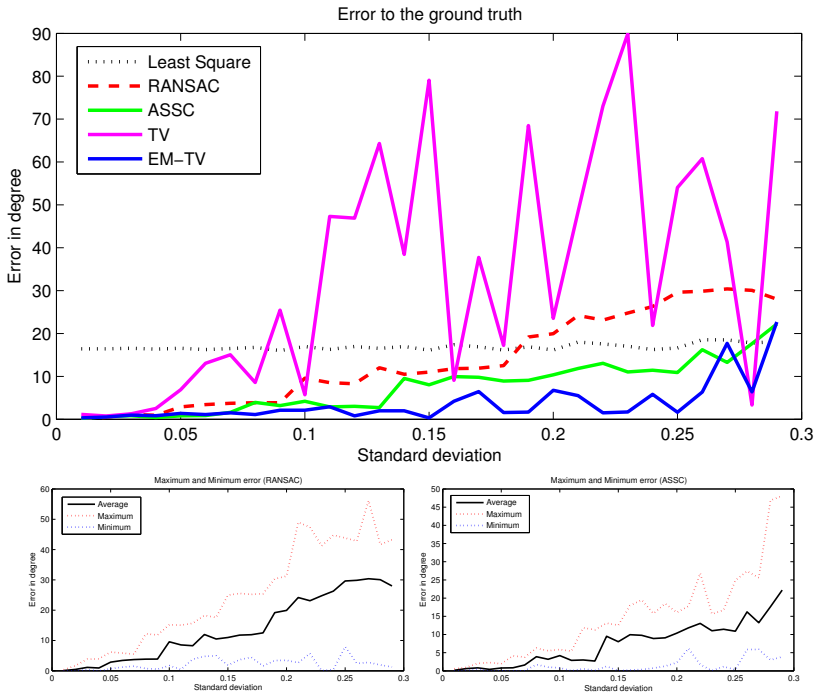



Figure 7: A test on measurement error, where the corresponding standard deviation varies from 0.01 to 0.29 with IO ratio 10.

fundamental matrix. Data normalization is similarly done as in [7] before the optimization.

We evaluate the results by estimating the fundamental matrix of the data set *Corridor*, which is available at www.robots.ox.ac.uk/∼vgg/data.html. The matches of feature points (Harris corners) are available. Figure 8 shows the plot of RMS error, which is computed by $\sqrt{\frac{1}{p}\sum_i ||\mathbf{U}_i^T \hat{\mathbf{v}}||^2}$, where $\mathbf{U}_i$ is the set of clean data, and $\hat{\mathbf{v}}$ is the 9D vector produced from the rank-2 fundamental matrices estimated by various methods. A few epipolar lines computed using EM-TV are shown on the image pair. It can be observed that RANSAC breaks down at an IO ratio $\simeq 20$, or 95.23% noise. ASSC is very stable with RMS error $< 0.15$. TV breaks down at an IO ratio $\simeq 10$. EM-TV has negligible RMS error before it starts to break down at an IO ratio $\simeq 40$. This finding echos with [7] that linear solution is sufficient when outliers are properly handled.

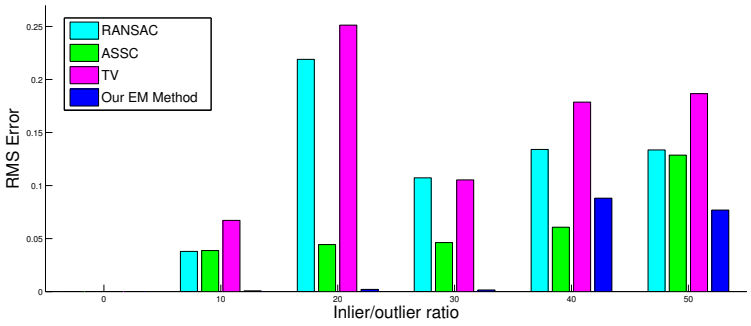



Figure 8: *Corridor*. RMS error plot of various methods.

# 8 Conclusions

We described EM-TV, where structure-aware tensors are introduced into EM optimization. A closed-form solution is presented for computing an optimal structure-aware tensor, which is used in the EM optimization for optimizing both the tensors and model parameters. We performed quantitative studies on EM-TV to demonstrate the robustness of our method in comparison to existing techniques. In the future we will extend EM-TV to extract multiple and nonlinear structures, as outlined in the supplemental material. Also we will apply this extended EM-TV framework to solve real-world vision problems such as structure-from-motion.

# References

[1] S. Arya and D. M. Mount. Approximate nearest neighbor searching. *ACM-SIAM SODA'93*, pages 271–280.

[2] H. Chen and P. Meer. Robust regression with projection based m-estimators. In *ICCV03*, pages II: 878–885, 2003.

[3] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24:603–619, May 2002.

[4] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM*, 24:381–395, 1981.

[5] D. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2003.

[6] B. Georgescu, I. Shimshoni, and P. Meer. Mean shift based clustering in high dimensions: A texture classification example. In *ICCV03*, pages 456–463, 2003.

[7] R. Hartley. In defense of the eight-point algorithm. *PAMI*, 19(6):580–593, June 1997.

[8] P. Hough. *Machine analysis of bubble chamber pictures*. Centre Européenne pour la Recherch Nucléaire (CERN), 1959.

[9] P. J. Huber. *Robust Statistics*. John Wiley & Sons, 1981.

[10] K. Lee, P. Meer, and R. Park. Robust adaptive segmentation of range images. *PAMI*, 20(2):200–205, February 1998.

[11] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley, 2008.

[12] G. Medioni, M. S. Lee, and C. K. Tang. *A Computational Framework for Segmentation and Grouping*. Elsevier, 2000.

[13] P. Meer. Robust techniques for computer vision. In *Emerging Topics in Computer Vision*, page Chapter 4, 2004.

[14] P. Rousseeuw. Least median of squares regression. *Journal of American Statistics Assoc.*, 79:871–880, 1984.

[15] P. Rousseeuw. *Robust Regression and Outlier Detection*. Wiley, 1987.

[16] K. Sim and R. Hartley. Removing outliers using the l-inf norm. In *CVPR06*, pages I: 485–494, 2006.

[17] R. Subarao and P. Meer. Beyond ransac: User independent robust regression. In *Workshop on 25 Years of RANSAC*, June 2006.

[18] D. Tong, C. Tang, P. Mordohai, and G. Medioni. First order augmentation to tensor voting for boundary inference and multiscale analysis in 3d. *PAMI*, 26(5):594–611, May 2004.

[19] W. Tong, C. Tang, and G. Medioni. Simultaneous two-view epipolar geometry estimation and motion segmentation by 4d tensor voting. *PAMI*, 26(9):1167–1184, September 2004.

[20] H. Wang and D. Suter. Robust adaptive-scale parametric model estimation for computer vision. *PAMI*, 26(11):1459–1474, Nov 2004.