Robust Model Fitting in Computer Vision: Theory and Algorithms

Lecture Notes

July 17, 2025

Abstract

These notes provide a comprehensive treatment of robust model fitting techniques in computer vision, with particular emphasis on geometric primitives such as lines and circles. We explore both classical least squares approaches and modern robust methods including RANSAC and its variants. The mathematical foundations are developed rigorously, from the Direct Linear Transformation (DLT) algorithm to M-estimators and greedy consensus maximization strategies.

Contents

1 Introduction to Robust Fitting

1.1 Motivation and Problem Formulation

The problem of *robust model fitting* arises ubiquitously in computer vision applications where we must estimate analytical expressions derived from geometric constraints in the presence of outliers. Unlike standard noise, outliers represent data points that significantly deviate from the expected model with statistical characteristics that are fundamentally different from the inlier distribution.

Definition 1.1 (Robust Fitting Problem)**.** Given a set of observations $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, the robust fitting problem seeks to estimate parameters $\theta \in \mathbb{R}^p$ of a model $f(\mathbf{x}; \theta)$ such that:

$$\theta^* = \arg\min_\theta \sum_{i=1}^{n} \rho(r_i(\theta)) \tag{1}$$

where $r_i(\theta) = y_i - f(\mathbf{x}_i; \theta)$ is the residual and $\rho : \mathbb{R} \to \mathbb{R}^+$ is a robust loss function.

The choice of ρ fundamentally determines the robustness properties of the estimator. Classical least squares uses $\rho(r) = r^2$, which, as we shall demonstrate, lacks robustness to outliers.

1.2 Applications in Computer Vision

Robust fitting finds applications across numerous computer vision tasks:

(i) **Sports Analytics**: Fitting lines of a soccer field from camera views enables real-time player tracking and tactical analysis. This requires fitting multiple line models simultaneously while handling occlusions and perspective distortions.

(ii) **Document Rectification**: Mobile document scanning applications detect quadrilaterals by fitting four line equations, then compute a homography to rectify the perspective distortion.

(iii) **3D Reconstruction**: Establishing point correspondences between images requires robust estimation of the fundamental matrix $\mathbf{F} \in \mathbb{R}^{3\times3}$ relating epipolar geometry.

(iv) **Object Detection**: Template matching and object localization often involve fitting geometric primitives to edge features extracted from images.

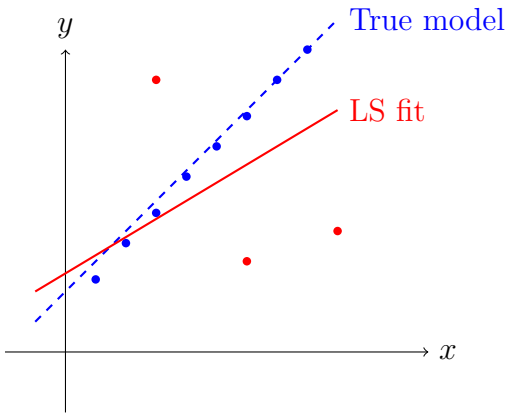

Figure 1: Illustration of outlier influence on least squares fitting

2 Classical Least Squares Approaches

2.1 Ordinary Least Squares for Line Fitting

We begin with the fundamental problem of fitting a straight line to a set of 2D points. The parametric form of a line is:

$$y = mx + b \tag{2}$$

where m is the slope and b is the y-intercept. Given observations $\{(x_i, y_i)\}_{i=1}^{n}$, ordinary least squares (OLS) seeks:

$$(m^*, b^*) = \arg\min_{m,b} \sum_{i=1}^{n} (y_i - mx_i - b)^2 \tag{3}$$

2.1.1 Matrix Formulation and Solution

Define the design matrix $\mathbf{A} \in \mathbb{R}^{n\times 2}$ and parameter vector $\theta \in \mathbb{R}^2$:

$$\mathbf{A} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix}, \quad \theta = \begin{bmatrix} m \\ b \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \tag{4}$$

The optimization problem becomes:

$$\theta^* = \arg\min_{\theta} \|\mathbf{y} - \mathbf{A}\theta\|_2^2 \tag{5}$$

Theorem 2.1 (Normal Equations)**.** The optimal parameters for the least squares problem are given by:

$$\theta^* = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y} \tag{6}$$

provided that $\mathbf{A}^T\mathbf{A}$ is invertible (i.e., $\text{rank}(\mathbf{A}) = 2$).

Proof. Taking the gradient of the objective function:

$$\nabla_\theta \|\mathbf{y} - \mathbf{A}\theta\|_2^2 = \nabla_\theta (\mathbf{y} - \mathbf{A}\theta)^T (\mathbf{y} - \mathbf{A}\theta) \tag{7}$$

$$= -2\mathbf{A}^T (\mathbf{y} - \mathbf{A}\theta) \tag{8}$$

Setting the gradient to zero yields the normal equations:

$$\mathbf{A}^T \mathbf{A}\theta = \mathbf{A}^T \mathbf{y} \tag{9}$$

When $\mathbf{A}^T\mathbf{A}$ is invertible, we obtain the closed-form solution. □

2.2 Limitations of Vertical Distance Minimization

The ordinary least squares approach measures error along the vertical axis, which introduces several fundamental limitations:

1. **Inability to fit vertical lines**: When the line approaches vertical ($m \to \infty$), the formulation breaks down as we cannot express x as a function of y.
2. **Asymmetric treatment of variables**: The choice of dependent vs. independent variable artificially privileges one coordinate axis.
3. **Scale dependency**: The error metric depends on the coordinate system orientation.

3 The Direct Linear Transformation (DLT) Algorithm

3.1 Implicit Line Representation

To overcome the limitations of parametric representations, we adopt the implicit form:

$$ax + by + c = 0 \tag{10}$$

where $(a, b, c) \in \mathbb{R}^3$ are the line parameters. This representation elegantly handles all line orientations, including vertical lines (where $b = 0$).

Remark 3.1. The implicit representation introduces a scale ambiguity: the lines defined by (a, b, c) and $(\lambda a, \lambda b, \lambda c)$ for any $\lambda \neq 0$ are identical. This necessitates a normalization constraint.

3.2 Algebraic Error Minimization

Rather than minimizing geometric distance (which leads to nonlinear optimization), we minimize the *algebraic error*:

Definition 3.1 (Algebraic Error)**.** For a point (x_i, y_i) and line parameters $\theta = [a, b, c]^T$, the algebraic error is:

$$r_i^{alg}(\theta) = ax_i + by_i + c \tag{11}$$

The relationship between algebraic and geometric errors is:

$$r_i^{geom}(\theta) = \frac{|ax_i + by_i + c|}{\sqrt{a^2 + b^2}} = \frac{\left|r_i^{alg}(\theta)\right|}{\sqrt{a^2 + b^2}} \tag{12}$$

3.3 Constrained Least Squares Formulation

The DLT problem is formulated as:

$$\theta^* = \arg\min_\theta \|\mathbf{A}\theta\|_2^2 \quad \text{subject to} \quad \|\theta\|_2 = 1 \tag{13}$$

where the design matrix is:

$$\mathbf{A} = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix} \in \mathbb{R}^{n \times 3} \tag{14}$$

Theorem 3.1 (DLT Solution via SVD)**.** Let $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ be the singular value decomposition of $\mathbf{A}$, where:

- $\mathbf{U} \in \mathbb{R}^{n \times n}$ and $\mathbf{V} \in \mathbb{R}^{3 \times 3}$ are orthogonal matrices
- $\mathbf{\Sigma} = \text{diag}(\sigma_1, \sigma_2, \sigma_3)$ with $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq 0$

Then the optimal solution is $\theta^* = \mathbf{v}_3$, the last column of $\mathbf{V}$.

Proof. Using the orthogonality of $\mathbf{U}$ and the substitution $\mathbf{w} = \mathbf{V}^T\theta$:

$$\|\mathbf{A}\theta\|_2^2 = \left\|\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\theta\right\|_2^2 \tag{15}$$

$$= \left\|\mathbf{\Sigma}\mathbf{V}^T\theta\right\|_2^2 \quad \text{(orthogonal invariance)} \tag{16}$$

$$= \|\mathbf{\Sigma}\mathbf{w}\|_2^2 = \sum_{i=1}^{3} \sigma_i^2 w_i^2 \tag{17}$$

Since $\|\theta\|_2 = \|\mathbf{w}\|_2 = 1$, minimizing the objective requires placing all weight on the smallest singular value:

$$\mathbf{w}^* = [0, 0, 1]^T \implies \theta^* = \mathbf{V}\mathbf{w}^* = \mathbf{v}_3 \tag{18}$$

□

3.4 Preconditioning for Numerical Stability

Numerical conditioning is crucial for stable DLT computation, especially when dealing with higher-order geometric entities.

Algorithm 3.1 (Normalized DLT)**.** 1. Compute data centroid and scale:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{19}$$

2. Apply normalization transformation:

$$\mathbf{T} = \begin{bmatrix} s_x & 0 & -s_x\bar{x} \\ 0 & s_y & -s_y\bar{y} \\ 0 & 0 & 1 \end{bmatrix} \tag{20}$$

   where s_x and s_y are chosen such that the normalized data has unit variance.

3. Solve DLT for normalized points: $\tilde{\theta}^* = \text{DLT}(\{\mathbf{T}\mathbf{p}_i\})$

4. Denormalize the solution: $\theta^* = \mathbf{T}^T\tilde{\theta}^*$

4 Robust Estimation Methods

4.1 The Breakdown Point and Robustness

Definition 4.1 (Breakdown Point)**.** The breakdown point ϵ^* of an estimator is the largest fraction of arbitrarily corrupted observations that the estimator can handle while still producing a bounded estimate close to the true value.

For ordinary least squares, the breakdown point is $\epsilon^* = 0$, meaning a single outlier with sufficient leverage can arbitrarily corrupt the estimate. This motivates the development of robust alternatives.

4.2 M-Estimators

M-estimators generalize maximum likelihood estimation by replacing the squared loss with robust alternatives:

$$\theta^* = \arg\min_\theta \sum_{i=1}^{n} \rho(r_i(\theta)) \tag{21}$$

Common choices for ρ include:

Name	$\rho(r)$	Influence Function $\psi(r) = \rho'(r)$
L2 (Least Squares)	r^2	$2r$
L1 (Least Absolute)	$\lvert r\rvert$	$\operatorname{sign}(r)$
Huber	$\begin{cases} r^2/2 & \lvert r\rvert \le k \\ k\lvert r\rvert - k^2/2 & \lvert r\rvert > k \end{cases}$	$\begin{cases} r & \lvert r\rvert \le k \\ k\operatorname{sign}(r) & \lvert r\rvert > k \end{cases}$
Tukey's Bisquare	$\begin{cases} \frac{k^2}{6}[1-(1-r^2/k^2)^3] & \lvert r\rvert \le k \\ k^2/6 & \lvert r\rvert > k \end{cases}$	$\begin{cases} r(1-r^2/k^2)^2 & \lvert r\rvert \le k \\ 0 & \lvert r\rvert > k \end{cases}$

Table 1: Common M-estimator loss functions and their derivatives

4.3 RANSAC: Random Sample Consensus

RANSAC takes a fundamentally different approach by explicitly modeling the presence of outliers through a consensus framework.

4.3.1 The Consensus Set

Definition 4.2 (Consensus Set)**.** Given parameters θ and threshold ϵ, the consensus set is:

$$\mathcal{C}(\theta) = \{i : |r_i(\theta)| \le \epsilon\} \tag{22}$$

RANSAC maximizes the cardinality of the consensus set:

$$\theta^* = \arg\max_\theta |\mathcal{C}(\theta)| \tag{23}$$

4.3.2 The RANSAC Algorithm

Algorithm 4.1 (RANSAC)**. Input:** Data points $\mathcal{D}$, inlier threshold ϵ, confidence p
Output: Model parameters θ^*

1. Initialize: $\mathcal{C}^* = \emptyset$, $k = 0$
2. **while** $k < N$ **do**:
   (a) Randomly sample minimal set $\mathcal{S} \subset \mathcal{D}$ with $|\mathcal{S}| = m$
   (b) Fit model: $\theta_k = \text{FitModel}(\mathcal{S})$
   (c) Evaluate consensus: $\mathcal{C}_k = \{i : |r_i(\theta_k)| \leq \epsilon\}$
   (d) **if** $|\mathcal{C}_k| > |\mathcal{C}^*|$ **then** $\mathcal{C}^* = \mathcal{C}_k$, $\theta^* = \theta_k$
3. Refine: $\theta^* = \text{LeastSquares}(\mathcal{C}^*)$

4.3.3 Determining the Number of Iterations

The number of iterations N is determined probabilistically to ensure finding at least one outlier-free sample with confidence p:

Theorem 4.1 (RANSAC Iteration Count)**.** Given inlier ratio w and desired confidence p, the required number of iterations is:

$$N = \frac{\log(1-p)}{\log(1-w^m)} \tag{24}$$

where m is the minimal sample size.

Proof. The probability of selecting an all-inlier minimal sample is w^m. The probability of failing to select such a sample in N attempts is $(1-w^m)^N$. Setting this equal to $1-p$ and solving for N yields the result. □

Example 4.1. For line fitting ($m = 2$) with 50% inliers ($w = 0.5$) and 99% confidence ($p = 0.99$):

$$N = \frac{\log(0.01)}{\log(1-0.5^2)} = \frac{\log(0.01)}{\log(0.75)} \approx 17 \tag{25}$$

4.4 MSAC and MLESAC Variants

4.4.1 MSAC: M-estimator Sample Consensus

MSAC refines RANSAC by considering the magnitude of residuals within the inlier band:

$$L_{MSAC}(\theta) = \sum_{i=1}^{n} \min(|r_i(\theta)|, \epsilon) \tag{26}$$

This formulation corresponds to a truncated L1 loss, providing better discrimination between competing models with similar consensus set sizes.

4.4.2 MLESAC: Maximum Likelihood Sample Consensus

MLESAC models the error distribution explicitly as a mixture:

$$p(r_i) = \gamma \mathcal{N}(0, \sigma^2) + (1 - \gamma)\mathcal{U}(-v, v) \tag{27}$$

where γ is the inlier ratio, and maximizes the likelihood of the observed residuals.

5 Advanced Topics and Applications

5.1 Multi-Model Fitting

Real-world scenarios often require fitting multiple models simultaneously. Consider detecting multiple lines in an edge map or multiple planes in a point cloud.

Definition 5.1 (Multi-Model Fitting Problem)**.** Given data $\mathcal{D}$, find models $\{\theta_j\}_{j=1}^{K}$ and assignments $\{z_i\}_{i=1}^{n}$ where $z_i \in \{1, \ldots, K, K+1\}$ (with $K+1$ denoting outliers) that minimize:

$$\sum_{j=1}^{K} \sum_{i:z_i=j} \rho(r_i(\theta_j)) + \lambda \cdot |\{i : z_i = K+1\}| \tag{28}$$

5.2 Adaptive Threshold Selection

The inlier threshold ϵ critically affects RANSAC performance. Adaptive methods estimate ϵ from the data:

Algorithm 5.1 (Adaptive RANSAC)**.** 1. Initial fit using conservative ϵ_0

2. Estimate noise level from inlier residuals:

$$\hat{\sigma} = \text{MAD}(\{r_i : i \in \mathcal{C}^*\})/0.6745 \tag{29}$$

   where MAD is the median absolute deviation.

3. Update threshold: $\epsilon = k\hat{\sigma}$ (typically $k \in [2.5, 3]$)

4. Iterate until convergence

5.3 Geometric Constraints and Manifold Fitting

When fitting models with geometric constraints (e.g., orthogonal lines, concentric circles), the parameter space forms a manifold. The optimization becomes:

$$\theta^* = \arg\min_{\theta \in \mathcal{M}} \sum_{i=1}^{n} \rho(r_i(\theta)) \tag{30}$$

where $\mathcal{M}$ represents the constraint manifold.

Example 5.1 (Orthogonal Line Pairs)**.** For two orthogonal lines with parameters $\theta_1 = [a_1, b_1, c_1]^T$ and $\theta_2 = [a_2, b_2, c_2]^T$, the constraint is:

$$a_1 a_2 + b_1 b_2 = 0 \tag{31}$$

This reduces the degrees of freedom from 6 to 5.

5.4 Performance Analysis and Guarantees

Theorem 5.1 (RANSAC Success Probability)**.** The probability that RANSAC finds the correct model after N iterations is:

$$P_{success} = 1 - (1 - w^m)^N \tag{32}$$

where the bound is tight when outliers are uniformly distributed.

For structured outliers (e.g., points from another model), the actual performance can be significantly better than this worst-case bound.

6 Implementation Considerations

6.1 Efficient Sampling Strategies

6.1.1 Guided Sampling

Rather than uniform random sampling, guided sampling exploits prior information:

- **Proximity-based**: Sample points that are spatially close
- **Feature-based**: Use appearance similarity to guide sampling
- **Progressive sampling**: Adaptively update sampling distribution based on previous iterations

6.1.2 Deterministic Sampling

For small datasets, exhaustive evaluation of all $\binom{n}{m}$ minimal samples may be feasible and guarantees finding the optimal solution.

6.2 Parallel Implementation

RANSAC's iterations are inherently parallel, making GPU implementation attractive:

Algorithm 6.1 (Parallel RANSAC)**.**

1. Distribute iterations across P processors
2. Each processor maintains local best model
3. Global reduction to find overall best model
4. Load balancing through dynamic work distribution

6.3 Degeneracy Detection

Degenerate configurations (e.g., collinear points for circle fitting) must be detected and rejected:

Definition 6.1 (Degenerate Configuration)**.** A minimal sample $\mathcal{S}$ is degenerate if it does not uniquely determine the model parameters, i.e., $\dim(\text{null}(\mathbf{A}_{\mathcal{S}})) > 1$.

6.4 Quality Metrics

Beyond consensus set size, additional metrics help evaluate model quality:

1. **Residual statistics**: Mean, median, and variance of inlier residuals
2. **Distribution tests**: Kolmogorov-Smirnov test for residual normality
3. **Geometric consistency**: Preservation of known geometric relationships
4. **Cross-validation**: Performance on held-out data