Robust Classification for Attitude Estimation - Draft

Will Driessen
Advisor: Dr. John Christian

Georgia Institute of Technology
Space Exploration Analysis Laboratory (SEAL)
Space Systems Design Laboratory (SSDL)
December 13, 2023

Abstract

Spacecraft are continuously fed with signals from sensors to determine their state, for example, attitude, position, and velocity. Optical Navigation (OPNAV) is a method for determining the attitude of a spacecraft based on optical measurements, usually a camera or a Light Detection and Ranging (LIDAR) sensor. These measurements of celestial bodies or other spacecraft introduce noise and outliers into large datasets due to the imperfections in the sensor. This makes the measurements difficult to realign with a model. Many methods have been introduced to decrease the impact of outliers in numerical models, as a classic least squares fit is not enough to reject the influence of outliers. Random Sample Consensus (RANSAC) is a well-established algorithm for fitting experimental models and rejecting outliers from datasets by sampling a large dataset and comparing the relative errors in the models generated by the sample. Maximum Likelihood Estimation Sample Consensus (MLESAC) is another more robust method introduced in [1], where the classification is based further on the statistics of the dataset, rather than a simple tolerance in RANSAC. A new Statistically Optimal RANSAC (SO-RANSAC) approach is introduced as a new and improved algorithm developed by XAnalytix. In the following report, RANSAC, MLESAC, and SO-RANSAC are applied to pose estimation problem. Pose estimation is a computer vision task to determine the orientation of a sensor relative to a body [2]. By applying these algorithms to the pose estimation problem, a more accurate attitude transformation can be obtained by rejecting outliers in the sensor's measurements. Various classification metrics for the estimators are presented for comparison.

Project Overview

Part of a Small Business Innovative Research (SBIR) grant awarded to XAnalytix Systems by NASA Goddard titled *Improved Autonomous Navigation Through Optimal Sensor Outliers*, XAnalytix is in development of a Statistically Optimal RANSAC (SO-RANSAC) that uses novel algorithms to more accurately estimate pose using the covariance of the dataset[3]. This initiative will explore the comparison of SO-RANSAC to known RANSAC and MLESAC methods that can be deployed for spacecraft navigation in NASA's future missions. Outlier rejection is specifically important for relative navigation, such that formation flying and rendezvous can occur seamlessly. These events are generally assisted with a sensor (LIDAR) to provide accurate 3D depth of field measurements. Preliminary results will show the feasibility of SO-RANSAC versus other robust estimators.

1 Pose Estimation

This section explores the implementation of the pose estimation that is used directly in the RANSAC and MLESAC algorithms.

1.1 Problem Definition

First, to have a pose estimation problem that can be solved through Horn's Algorithm [4], a translation and a rotation must be applied to an array of points $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$ where $\boldsymbol{x}_i = (x_i, y_i, z_i)^T$ in $\mathbb{R}^3$. A transformation $\boldsymbol{T}$, where $\boldsymbol{T}$ is a proper orthogonal matrix, is applied to the set of points $\boldsymbol{X}$. That product is translated by a vector $\boldsymbol{b} = (b_x, b_y, b_z)^T$ to obtain a set of 3D points of correspondence $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_n)$. That is,

$$\boldsymbol{Y}_i = \boldsymbol{T}\boldsymbol{X}_i + \boldsymbol{b} \tag{1.1}$$

The main objective of the pose estimation is to find the pose $(\boldsymbol{T}, \boldsymbol{b})$ from datasets $(\boldsymbol{X}, \boldsymbol{Y})$. Therefore, the goal of the robust estimator algorithms is to obtain a more accurate pose estimation over multiple trials by identifying outliers in the measured dataset $\boldsymbol{Y}$.

1.2 Euclidean Transformation Definition

To input data into the algorithm, a random set of n points $\boldsymbol{X}$ can be generated with size $(3 \times n)$. A general 3D rotation can be defined by a quaternion $\boldsymbol{q}$, which can be derived from a singular rotation θ about an arbitrary direction $\boldsymbol{a}$ that defines the Euler axis. To calculate a rotation matrix from a quaternion use the Rodriguez Rotation Formula, to generate a rotation matrix to translate from an observer frame to a body fixed frame, from [5],

$$\boldsymbol{T} = \cos\theta \boldsymbol{I}_{3\times 3} - \sin\theta[\boldsymbol{a}\times] + (1 - \cos\theta)\boldsymbol{a}\boldsymbol{a}^T \tag{1.2}$$

Here, $[\boldsymbol{a}\times]$ is a cross product matrix, or, for $\boldsymbol{a} = (a_1, a_2, a_3)$,

$$[\boldsymbol{a}\times] = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{1.3}$$

Or in terms of a quaternion $\bar{\boldsymbol{q}} = (\boldsymbol{q}, q_s)^T$,

$$\boldsymbol{T} = (q_s^2 - \boldsymbol{q}^T\boldsymbol{q})\boldsymbol{I}_{3\times 3} - 2q_s[\boldsymbol{q}\times] + 2\boldsymbol{q}\boldsymbol{q}^T \tag{1.4}$$

Therefore, for a θ rotation about the Euler axis $\boldsymbol{a}$, a quaternion of $\boldsymbol{q}$ can be obtained, and using the above equations a rotation matrix $\boldsymbol{T}$ can be obtained. Any vector $\boldsymbol{b}$ can be chosen to transform the dataset.

1.3 Adding Measurement Noise and Outliers

To add random noise to the points of correspondence, a noise standard deviation σ can be multiplied by a randomly sampled array $\boldsymbol{\epsilon}$ of size $(3\times n)$ uniformly distributed between zero and one. This noise can be added to the measured dataset with $\boldsymbol{Y}_\epsilon = \boldsymbol{Y} + \sigma\boldsymbol{\epsilon}$. The result of this step is to add Gaussian noise to the dataset. Further work could involve modeling the noise as a Gaussian mixture.

Outliers are added to the dataset by random seeding and are set to be in the domain of the measured dataset, or generally around the measured object. The number of outliers inserted into the dataset is important to know when evaluating classification metrics for these algorithms.

1.4 Pose Estimation

To solve for the pose, Horn's Algorithm [4] will be used with Singular Value Decomposition (SVD). For a set of points $(\boldsymbol{x}, \boldsymbol{y})$, with correspondence based on a rigid body translation and rotation $(\boldsymbol{T}, \boldsymbol{b})$, the pose, or transformation defined in Eqn. 1.5 can be obtained with the following steps. Restating the problem,

$$\boldsymbol{y}_i = \boldsymbol{T}\boldsymbol{x}_i + \boldsymbol{b} \tag{1.5}$$

Generally, the goal is to find $\boldsymbol{T}$ and $\mathbf{b}$ that minimize the loss function O or bring the measured and known points into alignment.

$$O = \sum_{i=1}^{n} \|\boldsymbol{y}_i - \boldsymbol{T}\boldsymbol{x}_i - \boldsymbol{b}\|^2 \tag{1.6}$$

First, the centroid of each dataset can be calculated by the following, where n is the number of points.

$$\bar{\boldsymbol{x}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i, \quad \bar{\boldsymbol{y}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{y}_i \tag{1.7}$$

The loss function can be expanded with the new definition,

$$O = \sum_{i=1}^{n} \|\boldsymbol{y}_i - \boldsymbol{T}\boldsymbol{x}_i - \bar{\boldsymbol{y}} - \boldsymbol{T}\bar{\boldsymbol{x}}\|^2 \tag{1.8}$$

Next, the datasets are each moved to the centroid by the following definition, and the loss function is rewritten and expanded in terms of $\boldsymbol{x}'_i$ and $\boldsymbol{y}'_i$.

$$\boldsymbol{x}'_i = \boldsymbol{x}_i - \bar{\boldsymbol{x}}, \quad \boldsymbol{y}'_i = \boldsymbol{y}_i - \bar{\boldsymbol{y}} \tag{1.9}$$

Inserting this into the loss function,

$$O = \sum_{i=1}^{n} \|\boldsymbol{y}'_i - \boldsymbol{T}\boldsymbol{x}'_i\|^2 \tag{1.10}$$

$$= \sum_{i=1}^{n} \|\boldsymbol{y}_i'^T\boldsymbol{y}'_i + \boldsymbol{x}_i'^T\boldsymbol{x}'_i - 2\boldsymbol{y}_i'^T\boldsymbol{T}\boldsymbol{x}'_i\| \tag{1.11}$$

Therefore, the minimization of the O occurs with the maximization of the g.

$$g = \sum_{i=1}^{n} \boldsymbol{y}_i'^T\boldsymbol{T}\boldsymbol{x}'_i \tag{1.12}$$

$$= \mathbf{Tr}\left(\sum_{i=1}^{n} \boldsymbol{T}\boldsymbol{x}_i'^T\boldsymbol{y}'_i\right) \tag{1.13}$$

From the identity $\boldsymbol{a}^T\boldsymbol{R}\boldsymbol{b} = \mathbf{Tr}\left(\boldsymbol{R}\boldsymbol{a}\boldsymbol{b}^T\right)$. Next, define a matrix $\boldsymbol{M}$ such that $g = \mathbf{Tr}\left(\boldsymbol{T}\boldsymbol{M}\right)$,

$$\boldsymbol{M} = \sum_{i=1}^{n} \boldsymbol{y}'_i\boldsymbol{x}_i'^T \tag{1.14}$$

A singular value decomposition can be applied to matrix $\boldsymbol{M}$ to break down the linear transformation into a sequence of rotations or reflections given by $\boldsymbol{U}$ and $\boldsymbol{V}$, and a matrix of singular values $\boldsymbol{\Sigma}$.

$$(\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}^T) = \mathrm{svd}(\boldsymbol{M}) \tag{1.15}$$

The rotation matrix can be calculated as

$$\boldsymbol{T} = \boldsymbol{V}\boldsymbol{U}^T \tag{1.16}$$

With the estimated attitude transformation, the translation vector $\boldsymbol{b}$ can be found as follows.

$$\boldsymbol{b} = \boldsymbol{y} - \boldsymbol{T}\boldsymbol{x} \tag{1.17}$$

Thus, with a set of points of correspondence, the Euclidean transformation that defines the measured pose can be reconstructed.

2 Random Sample Consensus - RANSAC

RANSAC, first introduced in 1981 [6] is a general method for fitting an experimental model while accounting for outliers, and in this application, RANSAC is wrapping a pose estimation algorithm. Ideally, the RANSAC will return a more accurate pose estimation with a measured dataset containing many outliers. These outliers may take the form of noisy measurements, or complete failures of the measurement system.

To begin, one must define a set number of trials N_t over which to loop the RANSAC. The statistics of this choice are explored in Sec. 2.1 and in [7], but it generally is dependent on the ratio of outliers to the number of points, and the confidence of having at least one RANSAC trial that produces a correct pose.

Next, inside the iteration, randomly sample three points of correspondence at a minimum. Then calculate the pose with the sampled points, and using this pose, transform the original points to generate the estimated measurements. Next, calculate the error between the estimated measurements and the original measurements.

To generate the consensus set $\boldsymbol{C}$, find all of the errors less than some tolerance t_e, which is usually a multiple of the applied noise, and then save the size of the consensus set. Next, check if the consensus set is larger than the largest consensus set. If so, save the pose and consensus set. These steps are spelled out in pseudocode in Alg. 1.

Algorithm 1 Random Sample Consensus

for $k \leq N_t$ **do**
  Randomly sample $(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow (\boldsymbol{x}, \boldsymbol{y})$
  Estimate pose $(\boldsymbol{T}, \boldsymbol{b})$
  Compute $\boldsymbol{Y}_k = \boldsymbol{T}\boldsymbol{X} + \boldsymbol{b}$
  Compute $\boldsymbol{e} = ||\boldsymbol{Y} - \boldsymbol{Y}_k||$
  Compute $\boldsymbol{C} = \boldsymbol{e} \leq t_e$
  Calculate $c_k = \text{size}(\boldsymbol{C})$
  if $c_{best} < c_k$ **then**
    Save $\boldsymbol{T}$, $\boldsymbol{b}$, and $\boldsymbol{C}$ as the best
    $c_k = c_{best}$
  end if
end for

Additionally, if RANSAC were to be described as a loss function of the error $\boldsymbol{e}$ to be minimized as in [8],

$$L(\boldsymbol{e}) = \begin{cases} 0 & |\boldsymbol{e}| < t_e \\ const & \text{else} \end{cases} \tag{2.1}$$

2.1 Defining the Number of Trials for a RANSAC

The number of trials for a RANSAC, N_t, can be defined in multiple ways. One can define the number of trials as a specified integer or allow the iteration to continue indefinitely until some number of inliers are found. Another way, shown in [7], of defining the number of iterations is with a probability P that at least one sample of M points contains all inliers or no outliers. That is, where e is the probability that a point is an outlier, the probability that any point is an inlier is

$$P = 1 - e \tag{2.2}$$

Now, for a sample of M points, the probability that there is one or more outliers within a sample is,

$$P = 1 - (1 - e)^M \tag{2.3}$$

Next, for N_t trials, the probability that there are not any outliers in one sample of M points is,

$$P = 1 - (1 - (1 - e)^M)^{N_t} \tag{2.4}$$

This equation can be inverted to solve for N_t.

$$N_t = \frac{\log(1 - P)}{\log(1 - (1 - e)^M)} \tag{2.5}$$

Thus, for this problem, the number of iterations of the RANSAC loop can be defined with a desired probability of a pure-inlier sample P, the sample size of M points, and the expected outlier percentage e. In Fig. 1, it can be seen that as the number of trials increases, the success confidence increases, and that for increasing outlier percentages, the number of trials increases.

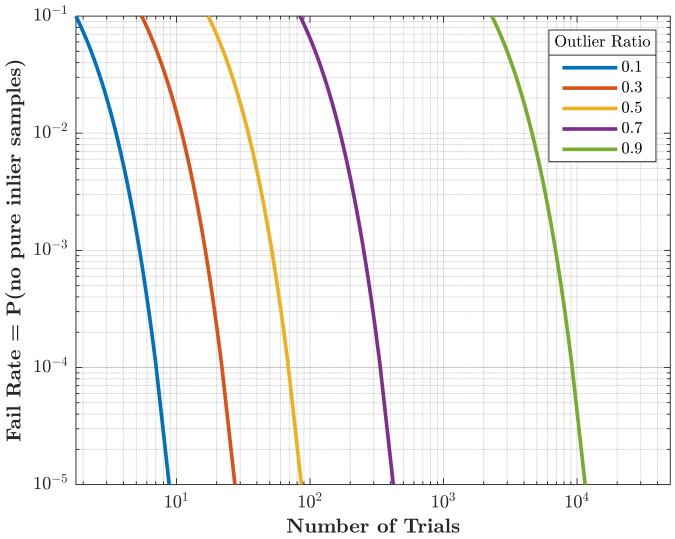

Figure 1: Trials vs Failure Rate for varying Outlier Ratios - Sample size of $M = 3$ points.

3 Maximum Likelihood Estimation - MLESAC

MLESAC, first investigated in 2000 [1], is a generalized method of RANSAC; instead of setting a threshold on the errors to determine the best model, the likelihood of inliers and outliers is investigated by inspection of the various statistical properties of the dataset.

To begin, from [9] the n-dimensional Probability Density Function (PDF) for a multivariate normal distribution is, for a symmetric positive definite diagonal covariance matrix $\boldsymbol{\Sigma}$, and mean $\boldsymbol{\mu}$,

$$f(\mathbf{x} \mid \boldsymbol{\Sigma}, \boldsymbol{\mu}) = \frac{1}{\sqrt{(2\pi)^k |\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})\right) \tag{3.1}$$

Recall that the term $(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})$ is the Maholonobis distance. For a single dimension case, the PDF is as follows,

$$f\left(x \mid \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{3.2}$$

For n one-dimensional equally distributed normal variables, the likelihood is,

$$\begin{aligned} f\left(x_1, \ldots, x_n \mid \mu, \sigma^2\right) &= \prod_{i=1}^{n} f\left(x_i \mid \mu, \sigma^2\right) \\ &= \sum_{i=1}^{n} \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \exp\left(-\frac{(x_i-\mu)^2}{2\sigma^2}\right) \end{aligned} \tag{3.3}$$

Where the product is simplified to a sum of exponential terms. Back to the single-dimension case, in MLESAC, the residuals are modeled as a Gaussian mixture with standard deviation σ, where γ is the mixing parameter between 0 and 1, or inlier ratio, and ν is the diameter of the expected range in which outliers are expected to fall [8].

$$\Pr(\boldsymbol{e}) = \gamma \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\boldsymbol{e}^2}{2\sigma^2}\right) + (1-\gamma)\frac{1}{v} \tag{3.4}$$

Applying the mixture to the n-dimensional PDF and taking the negative logarithm to minimize the negative log-likelihood, we obtain an expression for the MLESAC loss function,

$$L = -\log\left[\sum_{i=1}^{n} \gamma \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^d \exp\left(-\frac{\boldsymbol{e}_i^2}{2\sigma^2} + (1-\gamma)\frac{1}{v}\right)\right] \tag{3.5}$$

Here d is the dimension of the dataset, and for the case of pose estimation is $d = 3$. To implement this as a numerical method, an approach to estimate and minimize γ must be defined since it is not directly observable. This approach is called Expectation Minimization [1]. First, a guess for γ is determined, which begins at $\gamma_0 = \frac{1}{2}$ meaning that the dataset is a half-and-half mixture of inliers and outliers.

Next, since the probability that a given point is an inlier is of interest, γ can be guessed as a mean of the probabilities that a given datum is an inlier, over the distribution of errors for the given point. p_i is the likelihood that a given point is an inlier, and p_o is the likelihood that a given point is an outlier.

$$\gamma = \frac{1}{n}\sum_{i=1}^{n} \frac{p_i}{p_i + p_o} \tag{3.6}$$

$$p_i = \gamma \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^d \exp\left(-\frac{\boldsymbol{e}_i^2}{2\sigma^2}\right) \tag{3.7}$$

$$p_o = (1-\gamma)\frac{1}{v} \tag{3.8}$$

Then, after calculating L, one can compare it to the minimum L calculated previously and save the pose as optimal. These steps are shown in pseudocode in Alg. 2.

Algorithm 2 Maximum Likelihood Consensus

```
for k ≤ N_t do
    Randomly sample (X, Y) → (x, y)
    Estimate pose (T, b)
    Compute Y_k = TX + b
    Compute e = ||Y − Y_k||
    Set γ_0 = 1/2
    while δ > t_conv do
        Increment i = i + 1
        Estimate γ using Eqs. (3.6-3.8)
        Check for convergence, δ = |γ_{i−1} − γ_i|
    end while
    Compute L
    if L < L_min then
        Save T, b, and C as the best
        L = L_min
    end if
end for
```

It would be wise to set a maximum iteration number on the expectation minimization (Alg. 2, 7-11), to not ever experience an indeterminate solution.

4 Statistically Optimal RANSAC

Under development by XAnalytix, SO-RANSAC is a method for determining an optimal pose estimate, more accurate than that of Horn's Algorithm [4]. By using the known covariance of the points of correspondence in the pose estimation, the covariance in the pose estimation can be recovered and assists in the computation of the Mahalonobis distance. Instead of a tolerance on the errors,

like RANSAC, SO-RANSAC sets a tolerance on the Mahalonobis Distance. Note that this is the author's interpretation of SO-RANSAC.

The covariance of the input points can be defined as follows, where σ_1 is the measurement noise, and σ_2 is generally zero unless your known dataset has noise.

$$\boldsymbol{R} = \begin{bmatrix} \sigma_1^2 \boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \sigma_2^2 \boldsymbol{I}_{3\times3} \end{bmatrix} \tag{4.1}$$

Algorithm 3 Statistically Optimal RANSAC

for $k \leq N_t$ **do**
  Randomly sample $(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow (\boldsymbol{x}, \boldsymbol{y})$
  Estimate pose and covariance $(\boldsymbol{T}, \boldsymbol{b}, \boldsymbol{R}_Y)$ using $\boldsymbol{R}$
  Compute $\boldsymbol{Y}_k = \boldsymbol{T}\boldsymbol{X} + \boldsymbol{b}$
  Compute $\boldsymbol{e} = ||\boldsymbol{Y} - \boldsymbol{Y}_k||$
  Compute Maholonobis Distance $\boldsymbol{h}^2$
  Compute $\boldsymbol{C} = \boldsymbol{h}^2 \leq t_h$
  Calculate $c_k = \text{size}(\boldsymbol{C})$
  if $c_{best} < c_k$ **then**
    Save $\boldsymbol{T}$, $\boldsymbol{b}$, and $\boldsymbol{C}$ as the best
    $c_k = c_{best}$
  end if
end for

5 *Simulation and Results*

The following section describes the methods for comparing the estimation algorithms, and the results that were produced over the Monte Carlo simulation. All implementation occurred using *MATLAB R2023a*.

5.1 *Classification Metrics*

Classification is a common problem in machine learning and computer vision to categorize data into different groups based on various qualities of that data. For the example of inliers and outliers in a set of points, there are only two options for what a point is classified as. Though, if the prediction of a model is taken into account, a true or false prediction can occur. Therefore, for two classifications of data and two options for correctness, gives $2^2 = 4$ possible predictive outcomes. Each of these outcomes is defined in Fig. 2. This type of diagram is known as a Confusion Matrix and is a common method for measuring the predictions of a classification algorithm.

Assuming that an inlier is a positive class, and the outlier is the negative class, the rest follows. Thus a true positive (TP) corresponds to a correctly identified inlier, a false positive (FP) corresponds to an incorrectly identified inlier, (FN) a false negative corresponds to an incorrectly identified outlier, and a (TN) true negative corresponds to a correctly identified outlier.

It is intuitive to think that an algorithm will maximize the true predictions and minimize the false predictions to stay consistent with the model. Two outputs from the Confusion Matrix are the Precision and Recall of the algorithm, also known occasionally as confidence and sensitivity respectively. From Eq. 5.1, precision is defined as the ratio of the correctly identified inliers to all of the predicted inliers; this effect is obtained by including the falsely identified inliers in the denominator. A precision of 1 occurs when all inliers are accounted for, and no outliers are falsely identified as inliers.

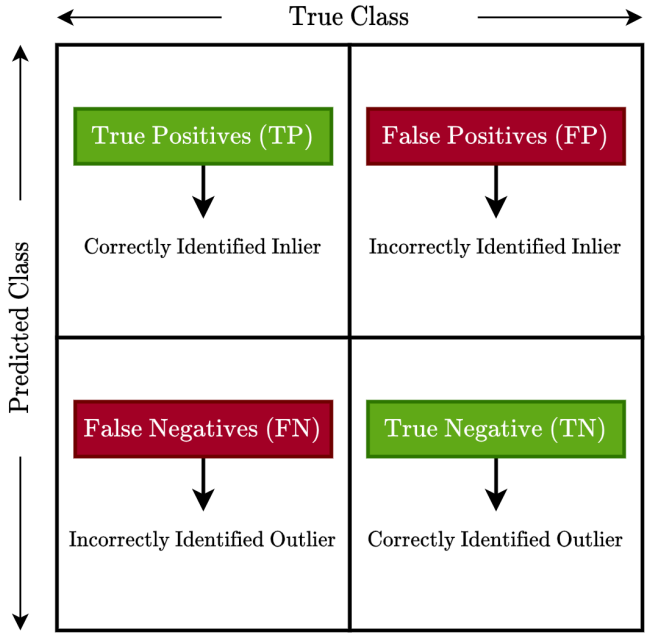

Figure 2: (2×2) Confusion Matrix.

$$\text{Precision} = P = \frac{TP}{TP + FP} \tag{5.1}$$

From Eq. 5.2, recall is defined as the ratio of the correctly identified inliers to all of the predicted inliers; this effect is obtained by including the falsely identified outliers in the denominator. A precision of 1 occurs when all inliers are accounted for, and no inliers are falsely identified as outliers.

$$\text{Recall} = R = \frac{TP}{TP + FN} \tag{5.2}$$

The F-Score is another commonly used classification metric, and it is a weighted sum of precision and recall, with β as the weighting parameter.

$$\text{F}_\beta = (1 - \beta^2)\frac{P \times R}{\beta^2 \times P + R} \tag{5.3}$$

For $\beta = 1$, the F_1 score is a common metric,

$$F_1 = \frac{2 \times P \times R}{P + R} \tag{5.4}$$

5.2 Monte Carlo Simulation

To compare the performance of these algorithms, Monte Carlo simulations were generated for various situations. First, using the vertices from a large point cloud provided by NASA containing 811 Points, Gaussian noise with $\sigma = 1 \times 10^{-5}$ was applied, and next 25% outliers were introduced to the measurement dataset, replacing previous measurement points.

For each algorithm, there is an input that determines the tolerance of the algorithm to the applied measurement noise σ when deciding whether a given datum is an inlier or outlier. These are then input into the algorithms that have a given number of trials that the algorithms have to determine the optimal pose output. For each prediction, by each algorithm, a confusion matrix can be constructed, which can calculate metrics such as precision, recall, and F1 Score, as discussed in the prior section. Figs. 3-4, show the results of the Monte Carlo simulation comparing the F1-Score of the three algorithms. Fig. 3 shows a tighter tolerance at 1σ whereas Fig. 4 shows a more realistic view of a 5σ tolerance. It can be seen that the SO-RANSAC takes fewer trials to achieve a higher score and that RANSAC and MLESAC have similar performance as you increase the multiplier on σ.

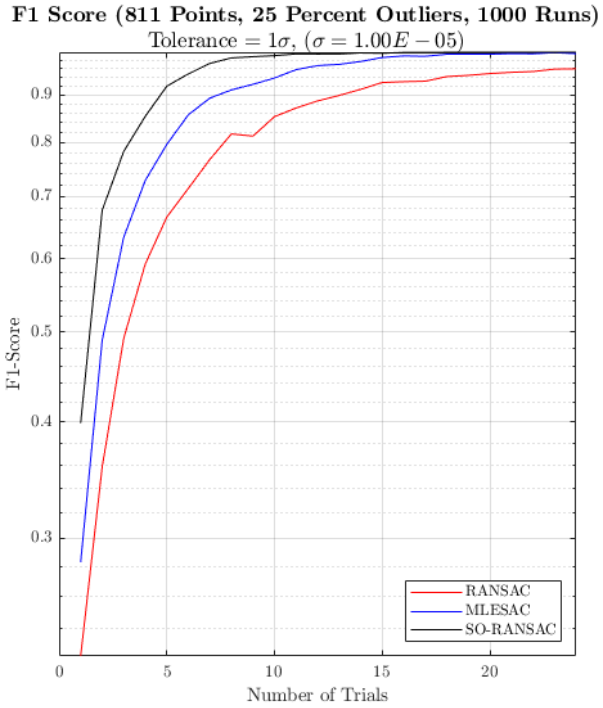

Figure 3: Algorithm F1 score comparison at 1σ.

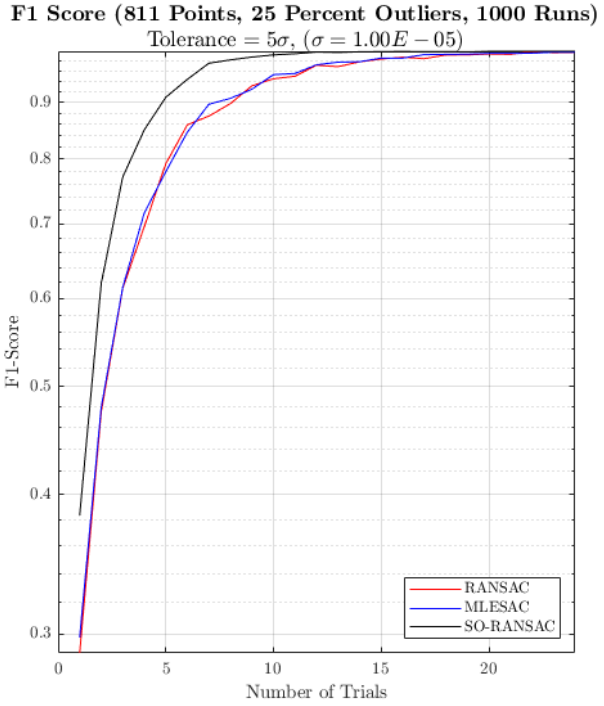

Figure 4: Algorithm F1 score comparison at 5σ.

6 Conclusion

SO-RANSAC has a higher combined precision and recall than both MLESAC and RANSAC. The main contributor to this difference is the recall, which may influence the appearance of the equally weighted F1-Score. Even though fewer trials are necessary to obtain a more accurate pose, there is still a large computational cost for SO-RANSAC, which is one to two orders of magnitude larger than the computational time of MLESAC and RANSAC. Therefore, for SO-RANSAC to give the best computational value, it must make up in accuracy, what it fails in computational cost. Further investigation into the computational cost of these algorithms is necessary to determine the situations in which one may be preferred. As it stands, MLESAC is likely the most cost-effective algorithm, while SO-RANSAC has higher combined precision and recall than the others, making it the best choice when computation time is not of concern.

Future work could also consist of generating novel ways to more accurately portray the noise of the LIDAR. Since the outlier and noise application does not take into account the line of sight of the LIDAR system, the simulation may not perfectly portray the behavior of the LIDAR in a real system. Implementing these robust estimation algorithms will be important for future refinement of robotic and remote sensing systems that use computer vision to inform navigation.

7 References

[1] P. Torr and A. Zisserman, "MLESAC: A New Robust Estimator with Application to Estimating Image Geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, Apr. 2000, ISSN: 10773142. DOI: 10.1006/cviu.1999.0832. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1077314299908329.

[2] C.-P. Lu, G. Hager, and E. Mjolsness, "Fast and globally convergent pose estimation from video images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 610–622, 2000. DOI: 10.1109/34.862199.

[3] "NASA SBIR 2023-Phase I Solicitation - Improved Autonomous Navigation Through Optimal Sensor Outliers." (2023), [Online]. Available: https://sbir.nasa.gov/SBIR/abstracts/23/sbir/phase1/SBIR-23-1-H9.03-2014.html.

[4] B. K. P. Horn, H. M. Hilden, and S. Negahdaripour, "Closed-form solution of absolute orientation using orthonormal matrices," en, *Journal of the Optical Society of America A*, vol. 5, no. 7, p. 1127, Jul. 1988, ISSN: 1084-7529, 1520-8532. DOI: 10.1364/JOSAA.5.001127. [Online]. Available: https://opg.optica.org/abstract.cfm?URI=josaa-5-7-1127.

[5] J. A. Christian, "AE 8803 - Optical Navigation - Attitude Representations Class Notes," 2023.

[6] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1, 1981, ISSN: 0001-0782. DOI: 10.1145/358669.358692. [Online]. Available: https://dl.acm.org/doi/10.1145/358669.358692.

[7] B. Ruzgiene and W. Förstner, "Ransac for outlier detection," *Geodezija ir Kartografija*, vol. 31, no. 3, pp. 83–87, 2005. DOI: 10.1080/13921541.2005.9636670. eprint: https://www.tandfonline.com/doi/pdf/10.1080/13921541.2005.9636670. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/13921541.2005.9636670.

[8] S. Choi and J.-H. Kim, "Reducing Effect of Outliers in Landmark-based Spatial Localization using MLESAC," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 2330–2335, 2008, ISSN: 14746670. DOI: 10.3182/20080706-5-KR-1001.00393. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1474667016392989.

[9] W. Holt and D. Nguyen, *Introduction to Bayesian Data Imputation*, Rochester, NY, Jun. 28, 2023. DOI: 10.2139/ssrn.4494314. [Online]. Available: https://papers.ssrn.com/abstract=4494314.