STATISTICAL ROBUSTNESS ANALYSIS OF RANDOM SAMPLING CONSENSUS METHOD

An Undergraduate Research Scholars Thesis

by

JONATHAN WEISHUHN

Submitted to the Undergraduate Research Scholars program at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by Research Advisor: Dr. Dezhen Song

May 2019

Major: Computer Science

TABLE OF CONTENTS

ABSTRACT

Analyzing the State of Computer Vision Algorithms via Robustness, Degeneracy, and Recognized Flaws

Jonathan Weishuhn
Department of Computer Science
Texas A&M University

Research Advisor: Dr. Dezhen Song
Department of Computer Science and Engineering
Texas A&M University

One of the most vocalized applications of computational interactivity today stem from our biological sense of perception, both in its promise for automation and heeding of its still prevalent weaknesses. Computer vision, as it is known, is a rapidly growing sub-field of computer science that creates use out of visual input utilizing various vision models and algorithms. Naturally these models and algorithms vary widely in terms of correctness, robustness, and degeneracy, especially when operating under disparate environments and conditions. Many publications explore the goal of developing new and robust vision models or algorithms, but less so explore the comparisons between those that already exist. The purpose of this paper is to detail the performance of Visual SLAM with other modern computer vision models (such as PTAM, ORB-SLAM, DSO, LSD, etc.) to produce a standard by which full comparisons may be drawn for both disparate environmental and conditional datasets. It is hoped that this paper will inform others in academia of the current state of computer vision models and help determine when the use of one model should be preferred over another given a certain environment and/or operating condition.

DEDICATION

I want to dedicate this thesis to my family and their never ending unconditional love, support, and guidance.

I also want to dedicate this undergraduate thesis to all those who have ever made an impact in my life, no matter how great or how small. Nearly everything that I do or enjoy is a result of some experience that I've had with those around me at one time or another.

ACKNOWLEDGEMENTS

I would like to thank Dr. Dezhen Song for accepting my application to do research in his field of robotic computer vision and matching me with Shu-Hao Yeh, who was instrumental in my studies and establishing the basis for this thesis.

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

SLAM	Simultaneous Localization and Mapping
PTAM	Parallel Tracking and Mapping
ORB-SLAM	Feature-based monocular Simultaneous Localization and Mapping
DSO	Direct Sparse Odometry
LSD	Large-Scale Direct Monocular Simultaneous Localization and Mapping
RANSAC	Random Sampling Consensus

CHAPTER I

INTRODUCTION

In the world today it has become nearly impossible to perform one's daily routine without experiencing some form of computer interaction. Be it in terms of touch on a hand held device, by sound to a home smart speaker, or even by composition via glucose reader, technology is increasingly becoming more adept and interactive with its environment in means not that different from our very own. In particular, it seems many of the most vocalized applications of computational interactivity today stem from our biological sense of perception, both in its promise for automation and heeding of its still prevalent weaknesses. Regardless of application, the need and desire for a more robust and less degenerate vision model will be paramount to progressing computer vision for the future.

One of the most critical components of this constant struggle originates from the increasingly small proximity in which humans and machines have come to work. Although major errors and incidents are less likely than ever before to occur, the mere existence of the possibility means there's always improvements to be made and lives to be saved. A prominent example today would be in the rise of the popularity of autonomous vehicles like the one shown in Figure 1. In that context, the robustness of even the simplest algorithm could quite literally be the difference between life and death. Some current computer vision algorithms have known calculation issues when the agent, or camera, turns on one of its axis in perfect rotation or if multiple reference images contain nothing but a solid white wall [1]. These problems aren't particularly unique to the field of computer vision and are derived from two fundamental measurements any algorithm will have, robustness and degeneracy. In the mathematical sense,

robustness refers to a process that can tolerate imperfect data with particular emphasis on outliers, like the white wall. Degeneracy is the quality of some process to degrade in accuracy or usefulness over a range [2]. In the context of the field of computer vision these properties represent the parameters at which algorithms can be measured for use in some particular application. As it stands today, most publications emphasizing these values only do so for one or two algorithms at a time which means in a broader survey of computer vision algorithms a researcher or industry developer would have to compare algorithms against each other under disparate contexts making it nearly impossible to gain one algorithm's true merit in terms of another.

Figure 1. Texas A&M's entry into the AutoDrive autonomous vehicle Challenge.

Visual SLAM [3], PTAM [4], ORB-SLAM [5], DSO [6], and LSD [7] are all computer vision algorithms harnessing different methodologies to perform a similar functionality, which is to categorize what is contained within an image, or set of images, within a given environment using the same principle of simultaneous localization and mapping (SLAM). Specifically, Visual SLAM is the process of creating a map or defined environment in the local area of some object, without any global context [8]. Initially conceived by Andrew Davidson, SLAM also contains the process by which the object can track itself within its own mapped local area [3]. Visual SLAM has reached a maturity level of documented strengths and weaknesses that makes it a favorable initial standard by which to measure the degeneracy and robustness of other computer vision algorithms across datasets spanning environments indoors, outdoors, and of non-optimal lighting/clarity. Moreover, to understand the metrics of these algorithms in a broad sense before direct experimentation one of their primary, underlying algorithms must first be analyzed. The objective and goal of this study was to analyze the robustness of the RANSAC (Random Sampling Consensus) algorithm [17]. As shown in Figure 2, RANSAC has far reaching derivatives that have found specific applications specializing in one aspect over another, making it a popular base for many of the computer vision models in use today. In their paper on the Performance Evaluation of RANSAC Family, Choi, Kim, and Yu discussed seven groups of RANSAC descendants that revolved around three main functions of speed, accuracy, and robustness [9].

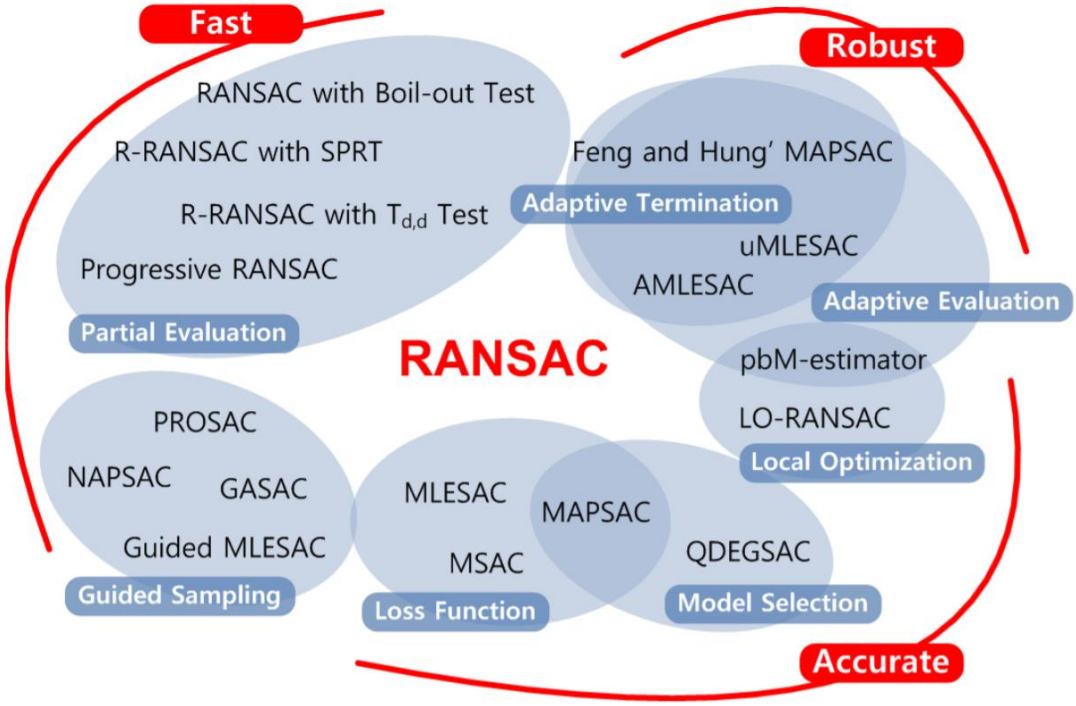

Figure 2. Various members of the RANSAC family [9].

These seven groups, depicted in Figure 2, fit roughly amongst the three main functions and relate in inner workings of each RANSAC implementation. In the accuracy category, descendants using a loss function, local optimization, and/or model selection are depicted. One example of a derivative in the loss function group is MLESAC (Maximum Likelihood SAC), which utilizes probability distribution of error by inliers and outliers to evaluate the generation and sampling of data [11]. Another form, that utilizes local optimization, is LO-RANSAC (Locally Optimized RANSAC) which theoretically improves the accuracy of RANSAC by optimizing the retrieval of its best guess estimation [12]. Model selections, such as with Torr's GRIC (Geometric Robust Information Criterion), help to make accurate estimations even when data doesn't exactly match the original model inputted [13].

The next category outlined was that of speed which was separated into two subgroups of guided sampling and partial evaluations. Guided sampling works on the basis of using prior knowledge to attach a score to newly sampled data. NAPSAC (N Adjacent Points SAC) implements RANSAC with the assumption that inliers tend to be closer to each other more often than to outliers [14]. Partial evaluation algorithms, like R-RANSAC (Randomized RANSAC), quit evaluating a given estimation if it stems too far from a predefined threshold, which reduces computing time and the number of data needed for each evaluation [15].

The third category classified included the derivatives that focused on the robustness of RANSAC. In this group these algorithms were split into those exhibiting adaptive evaluations and adaptive terminations. Adaptive evaluations dynamically change the estimation threshold automatically in order to retain accuracy when data varies noticeably. Algorithms such as AMLESAC use uniform search and gradient descent to optimize its computation for Expected Maximization [16]. Adaptive termination, on the other hand, iteratively determines a level of accuracy that is used to terminate early if it reaches a certain preset threshold.

Given the wide variety of exposure RANSAC has throughout the world today, this study sought to examine the uncertainty of rudimental form of RANSAC as it was subjugated to different configurations of inputted data sets, which simulated real world cases beyond the ideal.

CHAPTER II

RANDOM SAMPLING CONSENSUS

Description

RANSAC (or random sample consensus) is a non-deterministic algorithm used to estimate the properties of a provided model from a set of collected data containing both noise and outliers. Outliers, in this case, are defined as points that do not fit in the model given. The algorithm works on the basis that a random subset of points from the data set will be sampled, compared to a noise threshold of the model, and be refined further or discarded, which depends on how the consensus set of the sample fits the inlier threshold. It is not practical to sample each set of data to exhaustion, so an important part of the RANSAC algorithm is to calculate of the number of points needed to estimate the model's properties within the confines of a provided confidence interval. Generally, this is computed by utilizing a probability parameter which ensures that at least one of the samples estimated from the set is free of outliers. Since the algorithm does not typically sample a set to exhaustion, it will be usually implemented in an iterative way, providing a higher degree of success the more iterations it's allowed to run. The rate of success for RANSAC is directly related the number of iterations it's run, the noise of the inliers in the data set, and the proportion of outliers to inliers in the data set. It was by altering these parameters that the weaknesses and strengths of the RANSAC algorithm were observed.

Implementation

Since its first publication in 1981 by Fischler and Bolles [17], the RANSAC algorithm has seen multitudes of modifications in order to attempt to gain an advantage for some given

metric [9]. Although these algorithms were designed to perform in different ways, they've all remained closely related because they all utilize the same general RANSAC algorithm.

Figure 3 shows the five major steps in implementing the most basic RANSAC algorithm. At the vest least, implementing a generic RANSAC algorithm requires the following: a model to be estimated, a noise threshold, an assumed outlier proportion, and a probability that a set of random samples does not include an outlier. After this point the first step of the algorithm can begin.

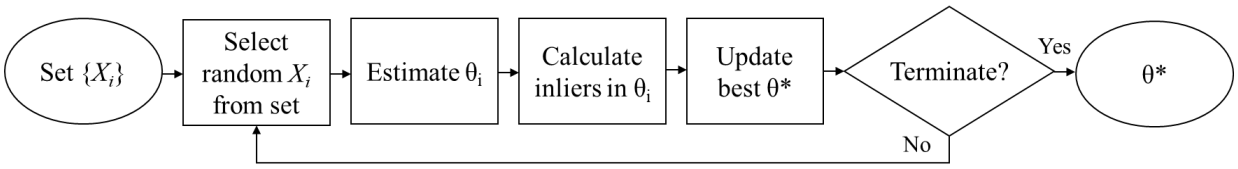

Figure 3. The rudimental flow of the RANSAC algorithm. θ is the model being estimated.

The first step in the RANSAC algorithm randomly samples a given set of data until the minimum amount of data needed to estimate the provided model is reached. In the case that the model to be estimated is that of a linear, two dimensional line, only two points from a set of points would be sampled.

The next step in the algorithm would take the sampled data and work backwards to calculate the parameters of the given model. Following the linear model as stated in the first step, this would mean calculating a slope and intercept from the two points sampled.

Once a new model is calculated from the sampled points, the implementation would then count up all of the inliers or data points from the total given set of data that fit inside the estimated model's noise threshold. Given a linear model estimation, this would require the

implementation to count all of the points from the set that fit on the line itself and within the given noise threshold value above and below it.

After the number of inlier is calculated, it is then compared to the previous maximum number of inliers. If no previous maximum exists or it exceeds the previous maximum it becomes the new maximum for the remaining cycles of iterations and its model estimation is saved for future reference.

Following the estimation and evaluation of the model for a given sample, the next step for the RANSAC algorithm is determine whether or not to continue sampling data, or to return a final best estimation. In many implementations this metric is calculated as

$$N = \frac{\log(1 - p)}{\log(1 - (1 - \epsilon)^s)} \quad (1)$$

where N is the number of iterations to be run, p is the probability that at least one of the random samples of s points is free of outliers, and ϵ is the probability that a selected point is an outlier. Once N iterations have progressed, the algorithm will then produce what it calculated to be the best estimation produced of the given model within the given set of data. For the goals and intentions of our research, this was the primary criterion used to determine the termination and thus accuracy of our RANSAC implementation.

CHAPTER III

UNCERTAINTY ANALYSIS

Purpose

To quantify the world around us in an understandable and useful way, measurements of some type or classification must be made. Since the world around us is not perfectly aligned to any one distinct model, each measurement made must be done so with an inherent assumption that it is correct to a predefined degree of accuracy. In defining this accuracy, one must first determine the acceptable amount of noise present in the collected data that can be tolerated for the specified purpose. Once all these precursory assumptions have been established, measurements performed through them can be analyzed, communicated, and utilized in a uniform way that can mitigate the uncertainty of a given value by the quantified degree. An important feature of quantifying the uncertainty of these measurements is that it allows for the benchmarking and statistical analysis of the processes that operate on them. In the context of this study, this meant the ability to determine the robustness and statistical accuracy of the RANSAC algorithm subject to pseudo-randomly generated data sets confined to preset configurations. Given the shrinking proximity of the human-machine relationship, there is a growing need for more precise and robust algorithms to ensure mutual safety as well as still maintain a viable efficiency. Since the RANSAC algorithm is the fundamental base for a large subset of computer vision models used in the academic and commercial world today, this study sought out to calculate the uncertainty that results from its implementation and observe any cases of interest that drastically skewed the resulting statistical data. In the context of these experiments this meant an uncertainty analysis would have to be performed on any resultant data. The foundation

for the resulting statistical analysis was based on the two-dimensional linear line model commonly observed on graphs with X/Y axis. Using this linear model in a RANSAC implementation will produce a resultant set of slopes and intercepts that contain both correct and incorrect estimations. Measuring the covariance of these two sets will yield a covariance matrix, which in conjunction with the chi-squared function and accompanying eigen values will produce two radii that can be used to plot an ellipse of uncertainty. This process was the procedure by which this study was able to quantify the uncertainty in the performance of an implementation of a rudimental form of the RANSAC algorithm. In the next subsection of this chapter the detailed procedure of calculating these values is further derived and developed.

Derivation

The measurement of uncertainty of the RANSAC algorithm implemented in this investigation was calculated via covariance error using explicit functions, which is a commonly used metric to analyze and visualize two-dimensional Gaussian distributed data as an ellipsoid using a given Mahalanobis radius. The first step in calculating covariance error ellipse is to calculate the variance-covariance matrix of *m* ordered sets of raw data containing *n* data points, respectively. Suppose **X** is an *m* x *n* matrix as shown in equation (2).

$$\mathbf{X} = \begin{bmatrix} X_{1,1} & \dots & X_{1,n} \\ \dots & \dots & \dots \\ X_{m,1} & \dots & X_{m,n} \end{bmatrix} \tag{2}$$

In order to derive the variance-covariance matrix one must first transform the raw values from matrix **X** into deviation scores, designated as the matrix **x** in equation (3)**.**

$$\mathbf{x} = \mathbf{X} - \begin{bmatrix} 1 \\ \dots \\ 1_m \end{bmatrix} \begin{bmatrix} 1 & \dots & 1_m \end{bmatrix} * \mathbf{X} * \frac{1}{m} \tag{3}$$

Once the deviation scores have been calculated, one must then compute the *n* x *n* deviation sums of squares and cross product matrix for **x**, which is shown below in equation (4) as **C**, which is the variance-covariance matrix used further on to plot an error ellipse of uncertainty.

$$\mathbf{C} = \mathbf{x}' * \mathbf{x} * \frac{1}{m} \tag{4}$$

Plotting a standard non-rotated requires the use of two radii, as shown in equation (5),

$$\left(\frac{\boldsymbol{x}}{\boldsymbol{r_x}}\right)^2 + \left(\frac{\boldsymbol{y}}{\boldsymbol{r_y}}\right)^2 = \mathbf{1} \tag{5}$$

where $\boldsymbol{r_x}$ and $\boldsymbol{r_y}$ are the ellipses horizontal and vertical radii, respectively. Plotting the covariance error ellipse requires only a slight modification, still assuming that the ellipse is non-rotated. Equation (6) shows that the original ellipse equation's $\boldsymbol{r_x}$ and $\boldsymbol{r_y}$ have been replaced by $\boldsymbol{\sigma_x}$ and $\boldsymbol{\sigma_y}$, respectively, in addition to being parametrized by a scale factor **s**, which is the provided Mahalanobis radius or confidence that some probability, **p**, is met.

$$\left(\frac{\boldsymbol{x}}{\boldsymbol{\sigma_x}}\right)^2 + \left(\frac{\boldsymbol{y}}{\boldsymbol{\sigma_y}}\right)^2 = \boldsymbol{s} \tag{6}$$

Using a provided **p** the Mahalanobis radius, **s**, can be calculated via the chi-squared function, shown in equation (7).

$$\boldsymbol{s} = -\mathbf{2}\boldsymbol{log}(\mathbf{1} - \boldsymbol{p}) \tag{7}$$

This effectively gives the ellipse a new set of radii of $\boldsymbol{\sigma_x}\sqrt{\boldsymbol{s}}$ and $\boldsymbol{\sigma_y}\sqrt{\boldsymbol{s}}$. For the cases when $\boldsymbol{\sigma_{xy}}$ and $\boldsymbol{\sigma_{yx}}$ are zero the above derivation works as a viable method of spread indication, but in many cases, like the ones detailed in this study, these values are non-zero and produce an ellipse that isn't axis-aligned. Thus, for the purposes experimented on in this thesis, the square roots of the eigenvalues of the covariance matrix $\boldsymbol{C} = \begin{bmatrix} \boldsymbol{\sigma_x}^2 & \boldsymbol{\sigma_{xy}} \\ \boldsymbol{\sigma_{yx}} & \boldsymbol{\sigma_y}^2 \end{bmatrix}$ are used as the final scaled radii of the

desired ellipse. Equation (8) shows the generalized ellipse equation for a given variance-covariance matrix **C**.

$$\left(\frac{\boldsymbol{x}}{\sqrt{\boldsymbol{\lambda_1}}}\right)^{\mathbf{2}} + \left(\frac{\boldsymbol{y}}{\sqrt{\boldsymbol{\lambda_2}}}\right)^{\mathbf{2}} = \boldsymbol{s} \tag{8}$$

CHAPTER IV

EXPERIMENTATION

Before performing the statistical analysis of the RANSAC algorithm, its rudimental form had to be implemented in a stable environment that would generate data consistently between trials. All experiments performed in this study were written and executed via the Python 3.7.0 programming language utilizing the libraries of MatPlotLib, Seaborn, and Scipy to process, visualize, and organize all of the data inputted into and outputted by the RANSAC implementation. Throughout this investigation, three distinct sets of experiments were conducted in order to try and best quantify the robustness state of the rudimental form of RANSAC. These experiments ranged from the analysis of the properties of inherent uncertainty, the observation of RANSAC's performance over a variety of outlier distributions, and the quantification of RANSAC's model estimation success rates over variable outlier percentages and noise thresholds.

The first experiment performed in this study was one that was meant to analyze the effect noise has on any given set of data. In the implementation, two two-dimensional points were initially plotted on a graph separated by u units. These points were then used to generated n other outlier points that were each plotted to within a given noise threshold of their respective original point. Once the two points and their outlier counterparts were initialized, both sets of points, denoted as set A and set B were inputted into a function that computed lines from a random point in A to a random point in B. After all slopes and intercepts were computed, the function then plotted the lines in contrast to the desired line along with the error ellipse produced from the matrix consisting of their slopes and intercepts. The goal of this experiment was to observe the

way in which the uncertainty of linear model estimation changed as the ratio of noise to separation changed. Since RANSAC operates on the principle of selecting random points from a set of data, it was hypothesized that this experiment could answer what effect concentrated inliers might have on its uncertainty.

The second experiment performed in this study was designed to further explore the robustness of RANSAC through the input of randomly and non-randomly distributed outliers. The procedure for a single trial of this experiment was initiated by first inputting an inlier slope and intercept, which was later meant for RANSAC to estimate. Next an outlier proportion, a probability threshold, and a noise threshold were required in order to setup the environment for the RANSAC implementation. At this point an optional outlier distribution function could also be passed to generate the outlier points in some specific configuration. After these parameters were employed, RANSAC would then begin sampling points from the full set of data, returning a best linear estimation once the iteration threshold calculated in equation (1) was met. If the returned estimation matched the predefined inlier model, then the trial would count as a success, otherwise it'd count as a failure. Before this study, literature on RANSAC had highlighted the algorithm's ability to remain robust in a set of a large distribution of presumably random outliers and little noise. The goal of this experiment was to investigate the effects changes in the distribution of outliers had on RANSAC's success rates in addition to its uncertainty.

The third and final experiment sought out in this study was to expand on the resultant set of the second experiment by altering the noise thresholds and outlier percentages rather than the outlier distributions. Similar to the previous experiment, an inlier slope and intercept were preset along with a probability threshold. From there, groups of trials were run varying the values of outlier percentage and/or noise threshold. Since it is not always possible to accurately guess the

outlier percentage and noise threshold in a real world environment, this experiment set out to determine what effects their differing values might have on RANSAC's success rates over the course of several hundred trials. Moreover, this experiment attempted to quantify which of the two parameters might be the most important to RANSAC's implementation for future applications.

CHAPTER V

RESULTS

Linear Uncertainties

Figures 4 and 5 show the results of performing the experiment analyzing the linear uncertainties associated with two generic data points that dwell on a two-dimensional plane. This experiment was conducted in two explicit configurations that first plotted two anchor points with 100 child points, each, that resided within a noise threshold range of ± 0.1 of their respective parent. Once the anchor points and their children were generated they were separated into two distinct sets, from which 100 random pairs were selected. For each of these pairs a line slope and intercept was computed and plotted in purple.

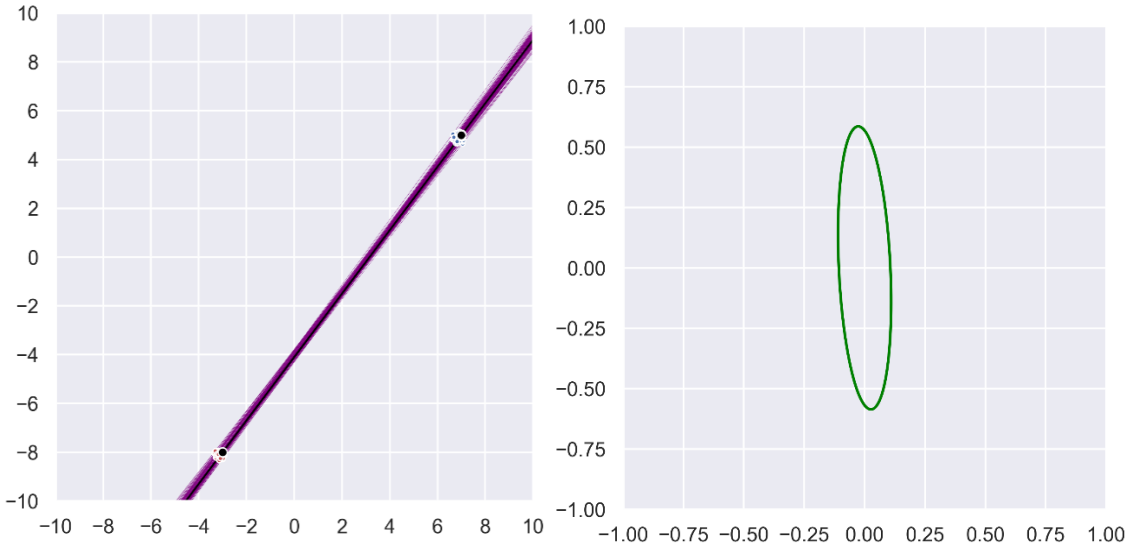

Figure 4. Viable estimation models plotted for a noise threshold of 0.1 and a point-sample distance of 16 units accompanied by an ellipse plotting the propagation of error.

Figure 4, above, shows the first configuration of the linear uncertainty experiment that generated two points disjoint by 16 units. In addition to that, Figure 4 also contains the error ellipse, plotted in green, that was generated by utilizing a covariance matrix made up of all 100 recorded slopes and intercepts.

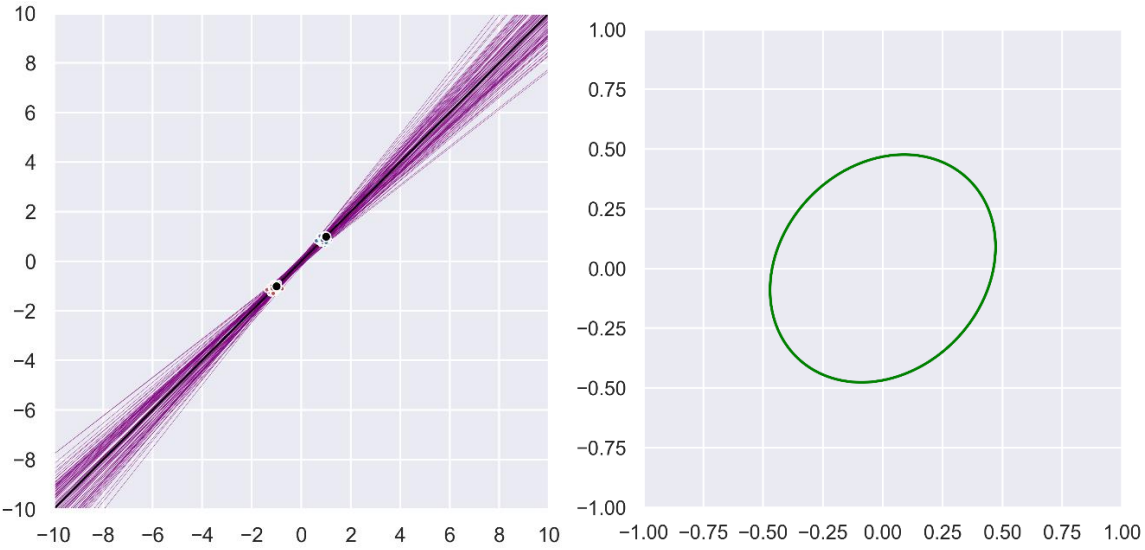

Figure 5. Viable estimation models plotted for a noise threshold of 0.1 and a point-sample distance of 1 accompanied by an ellipse plotting the propagation of error.

Figure 5, above and like Figure 4, shows the second configuration of the linear uncertainty experiment that generated two points disjoint by only 2 units. In comparison to Figure 4, Figure 5 shows just how much apart proximity plays in determining linear uncertainty. The line model in the first configuration visualizes a tight grouping of linear estimations than the second configuration even though both configurations share the same noise threshold. The error ellipses produced by the two configurations show even more quantitative differences as Figure 4's ellipse is significantly less pronounced than that of Figure 5's ellipse. This indicates that as the ratio of ratio of noise to separation increases so too does the uncertainty in linear estimation.

Outlier Distributions

Figure 6 depicts nine different configurations of outliers to inliers where each attempt to skew RANSAC's uncertainty and robustness in a different way. Subfigure 1-1 was the control case where all of the outliers were randomly distributed about the range of the expected inliers. Outliers in subfigures 1-2 and 2-1 distributed the outliers along the inlier's slope and intercept with noise threshold value at least one magnitude larger to gauge robustness in a proximity confined environment both when the outliers surrounded the inliers or kept to one side. Subfigures 1-3 and 2-2 sought to gauge how the RANSAC algorithm could handle close groupings of outliers both in and completely outside of the searched for estimation. Subfigure 2-3 was another attempt to try and skew success rates by plotting an independent slope and intercept with a larger noise but in a near perpendicular direction. The last three subfigures, 3-1, 3-2, and 3-3, tested circular, sinusoidal, and logarithmic distributions of outliers to see if non-linear functions had any prominent effects on the resultant uncertainty values and/or success rates over 1000 trials as shown in Figure 7 and Table 1, respectively.

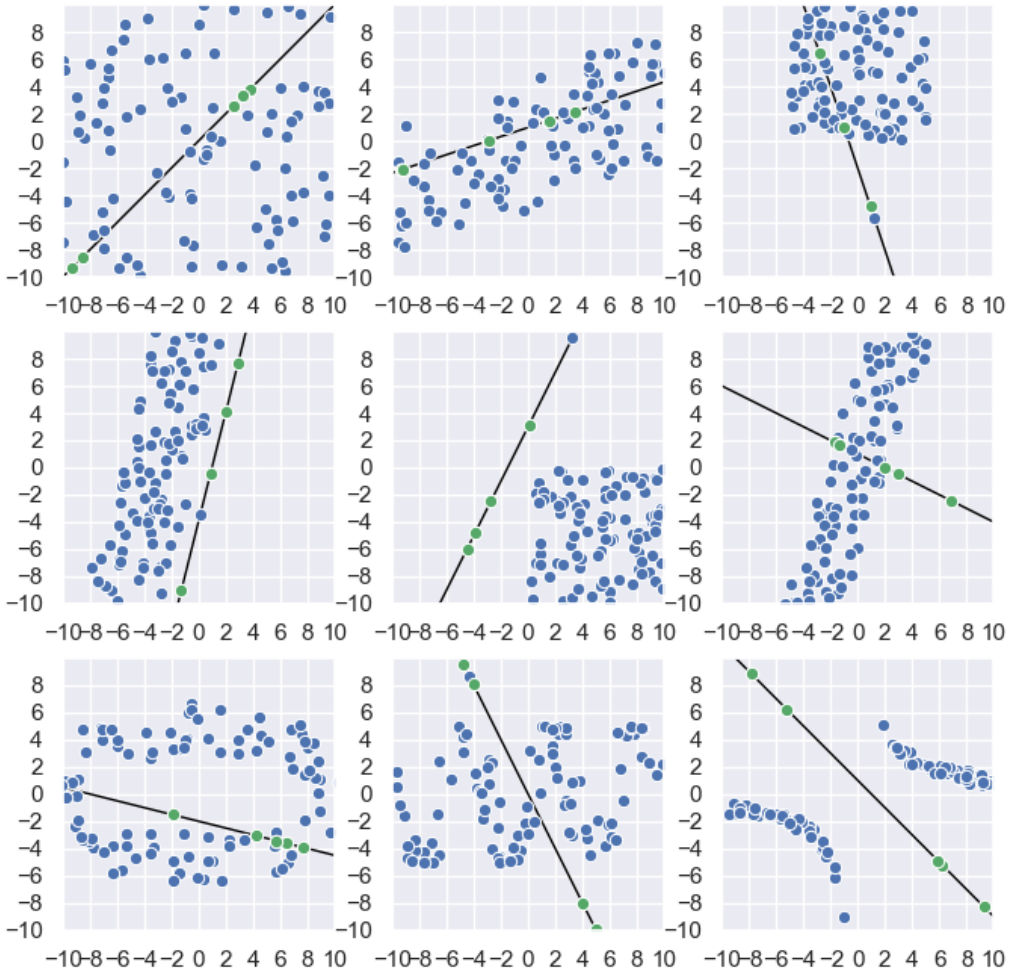

Figure 6. A 3x3 grid of subfigures (1-1 to 3-3) visualizing successful RANSAC estimations under scenarios of differing outlier distributions plotted against the correctly identified inliers.

Figure 7 visualizes the average covariance error ellipse generated after performing 1000 trials of the configurations shown in Figure 6. One may observe that as the outlier distributions tended to show visual similarities to the inlier's distributions, the error ellipses tended to span larger areas indicating that as outlier data showed resemblance to the generic linear model the larger the uncertainty was with the produced result. This finding is further reinforced with the data collected in Table 1. Subfigures with configurations, either resembling a linear model or in

close proximity, caused the success rates of RANSAC to decrease significantly when compared to the randomly distributed base case.

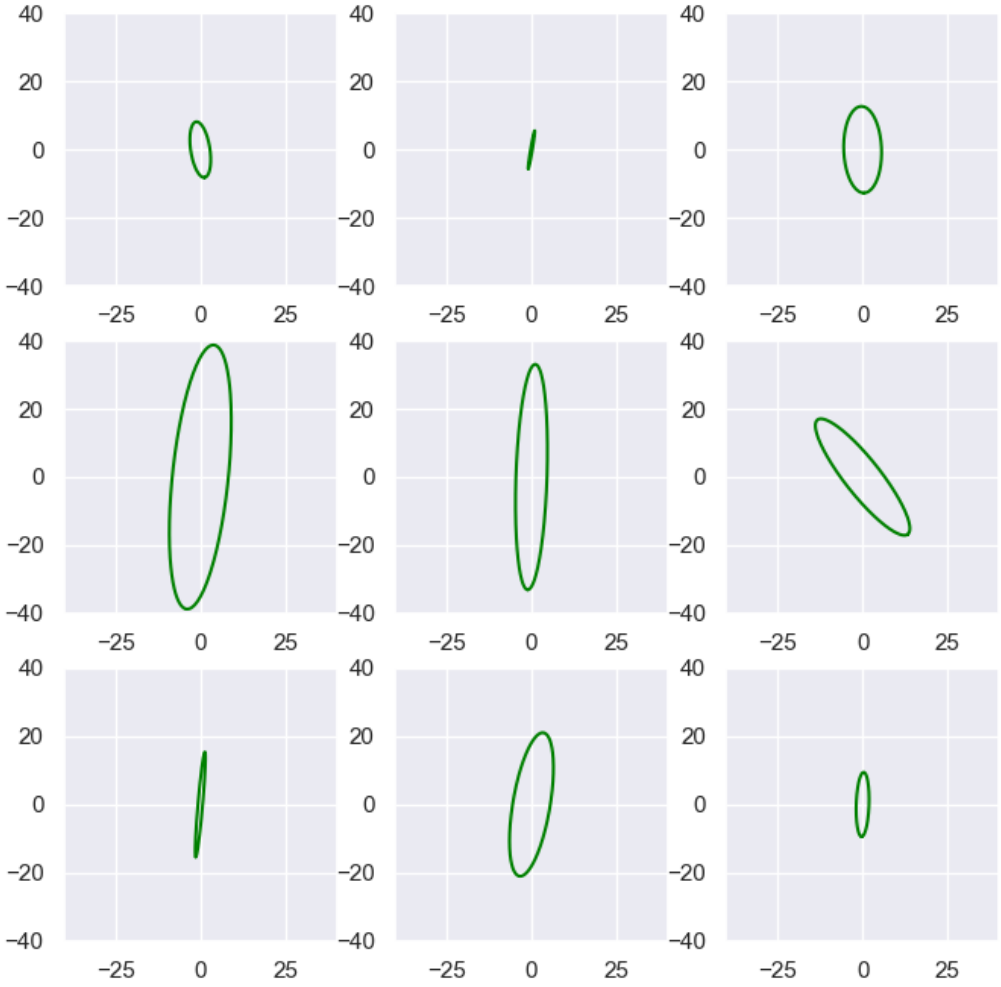

Figure 7. A 3x3 grid of subfigures (1-1 to 3-3) visualizing the error ellipses of their respective subfigures contained within Figure 6.

Table 1. Success rates of 1000 RANSAC trials run on subfigures shown in Figures 6 & 7.

0.88	0.66	0.28
0.06	0.52	0.86
0.66	0.58	0.78

Threshold Modifications

By this point in this study, it was apparent that outlier distributions had a direct effect on the robustness and uncertainty of RANSAC. What were still left unquantified was how noise thresholds and outlier percentages impacted RANSAC, and whether or not one was more important than the other. Before conducting this experiment fully, Figures 8 & 9 were generated to grasp exactly what cases would be summed in order to compute a standard success rate statistic.

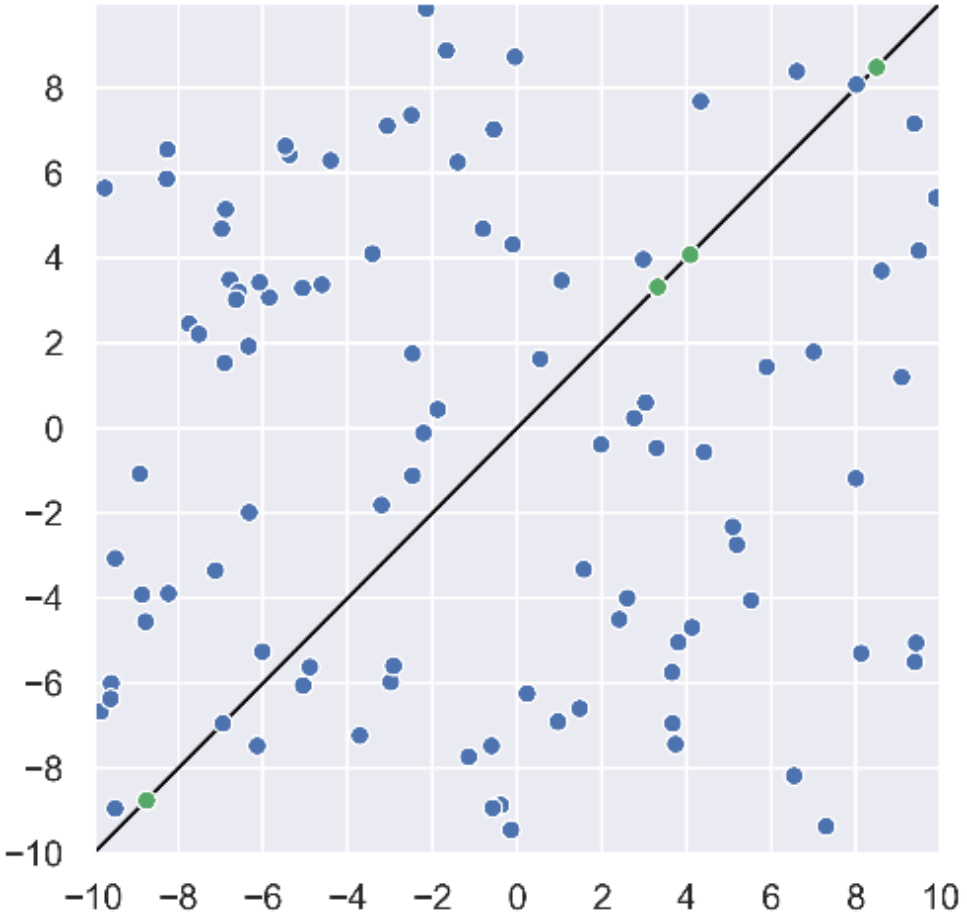

Figure 8. A plot containing correctly estimated parameters to a linear model in 95% outliers.

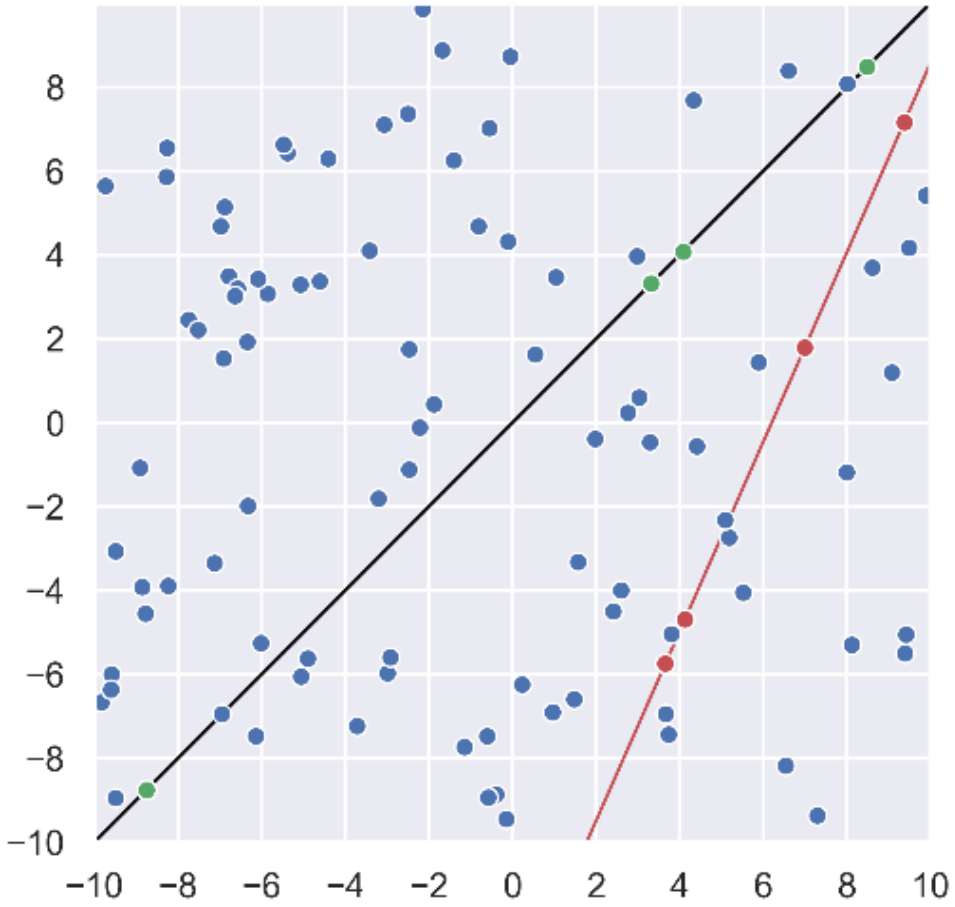

Figure 9. A plot containing incorrectly estimated parameters to a linear model in 95% outliers.

Figure 8 show a randomly distributed plot of outliers and inliers that has been run through the RANSAC algorithm and has resulted in a successful estimation. Figure 8 shows the same randomly generated plot of points but with an unsuccessful estimation plotted with what was desired. Running this process 1000 times and counting the number of successes allowed for the success rates in Table 2 to be tabulated through elementary division.

Table 2. Success rates of 1000 RANSAC trials run on a linear model with confidence of 0.99.

Noise Threshold \ Outlier Percentage	50	55	60	65	70	75	80	85	90	95
0.001	0.77	0.79	0.81	0.77	0.78	0.83	0.81	0.82	0.75	0.67
0.005	0.84	0.81	0.82	0.81	0.80	0.83	0.80	0.82	0.82	0.55
0.010	0.77	0.82	0.79	0.86	0.79	0.76	0.84	0.74	0.80	0.45
0.050	0.79	0.83	0.86	0.84	0.81	0.80	0.83	0.82	0.75	0.29
0.100	0.87	0.85	0.83	0.78	0.85	0.86	0.76	0.83	0.79	0.26
0.500	0.90	0.82	0.91	0.86	0.87	0.85	0.95	0.88	0.61	0.24
1.000	0.86	0.91	0.90	0.87	0.87	0.93	0.91	0.76	0.54	0.14

By observing the values alone, one is able to make the conclusion that, in general, as noise threshold and outlier percentage values increase the success rates of RANSAC estimation decrease. Determining if one variable is more important than the other is not a metric that appears obvious in the data collected throughout the experiment. It should be noted, however, that as the outlier percentage got larger the impact of changes in noise threshold values also got larger. It should also be noted that for this experiment the noise threshold values are relative to a distribution range of [-10, 10] on both axis. Perhaps in the future this value should be measured as a percentage of maximum scale of the data points inputted.

CHAPTER VI

CONCLUSION

Interpretation of Results

The implementation of the fundamental model of a random sampling consensus utilizing linear based model estimation reveals that much work is still left for improvements. It was discovered that through the implementation of the RANSAC algorithm, the way in which random samples are chosen and/or distributed in the collection greatly affect the resulting estimated models via substantial discrepancies in success rates. Figures 4 and 5 show how a single two-point linear model distribution with minimal noise can result in large increase to its potential error propagation. The larger the error propagation is in the collection set the greater the likelihood a false successful fit to the model will be returned since the threshold has an increased leeway to capture outlying points. This confirms the hypothesis that proximity of selected samples, in any algorithm, contribute significantly to the computed value's ending uncertainty. As proximity and/or noise increases, so too does the resultant uncertainty. This conclusion is further backed by the findings of the outlier distribution experiments. When outlier distributions exhibited closer proximities, especially to a near-linear model, success rates for RANSAC dropped significantly while uncertainty scaled magnitudes larger. The more randomly dispersed the outliers were the higher the success rates and lower the scale of uncertainty. Moreover, it was observed that so long as the distribution of outliers isn't similar to the model being estimated that RANSAC has the best chance of success in a pool of a larger percentage of outliers. These findings come as a result of the visualized data presented in Figures 6 & 7 along with their tabulated success rates in Table 1. Once the proximity and distribution of outliers were

established as key variables in the performance of RANSAC, the importance of noise threshold and outlier percentage was next to be analyzed. Given Table 2, it is obvious that both outlier percentage and noise threshold values contribute to the success rates of the RANSAC algorithm. Although the trend isn't prefect, due to the nature of random sampling, the intuitively observed statistics of the algorithm becoming less successful as the noise thresholds and outlier percentages increased remains evident. It is less obvious, however, how each does so on its own since the greatest changes in success rates followed changes in both parameters at the same time. Another observation made during this experiment was that although the noise threshold values were altered by factors of 10 and 5, this did not account for the range the data was generated in. Thus, for this portion of the study it should be stated that more experimentation is needed to fully understand the significance that the noise threshold contributes to the generic RANSAC algorithm.

Motivations & Future Work

The use and analysis of RANSAC has far reaching influences to the various disciplines of computer science as well as the many processes performing model estimation. It was for that reason that this study was conducted to analyze the robustness and degeneracy of RANSAC. Since the of the underlying tendencies of modifications to the primary parameters passed into the RANSAC algorithm have been observed, more specific and technical manifestations may now be perused. Alternatively, improvements to the base RANSAC algorithm could also be investigated as a means to mitigate its shortcomings under certain conditions. This future work might include further analysis of specific uses in computer vision applications or any new forms that try to improve its speed, robustness, or accuracy (such as with USAC [10]). Other avenues of statistical exploration could include processes like object detection or feature matching as well

(Figure 10) since both usually employ some form of a RANSAC model estimation to classify sample sets based on a provided baseline set.

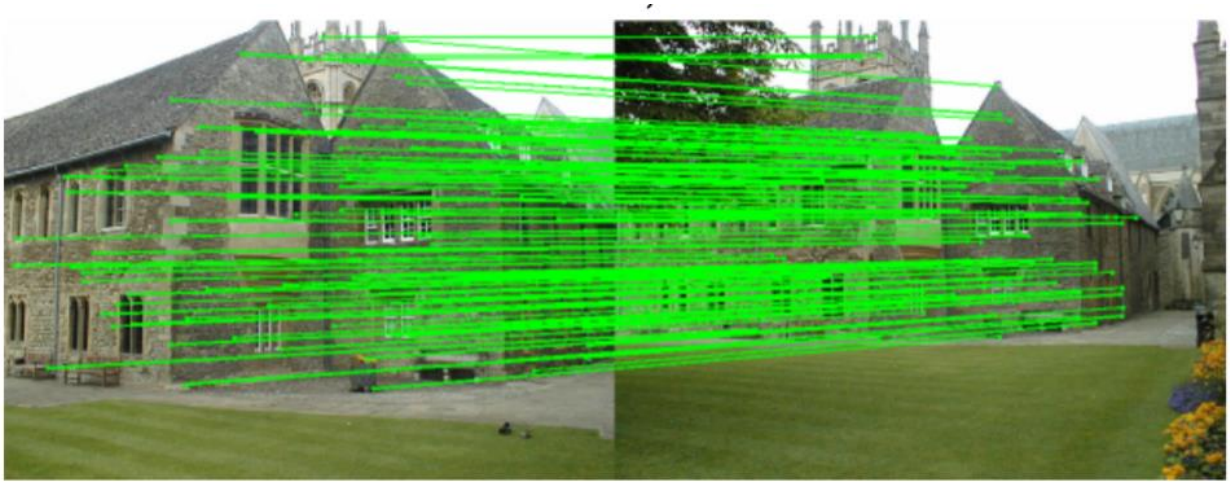

Figure 10. Feature matching using RANSAC estimation [10].

REFERENCES

[1] Hartley, R., & Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511811685

[2] Ji Zhang, Michael Kaess, and Sanjiv Singh. On Degeneracy of Optimization-based State Estimation Problems. *2016 IEEE International Conference on Robotics and Automation (ICRA).* Stockholm, Sweden. (May 16-21, 2016) 809-813

[3] J Davison, Andrew & D Reid, Ian & D Molton, Nicholas & Stasse, Olivier. (2007). MonoSLAM: real-time single camera SLAM. IEEE transactions on pattern analysis and machine intelligence. 29. 1052-67. 10.1109/TPAMI.2007.1049.

[4] Klein, Georg & Murray, David. (2007). Parallel Tracking and Mapping for Small AR Workspaces. 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality. 225-234. 10.1109/ISMAR.2007.4538852.

[5] Mur-Artal, Raul & Tardos, Juan. (2016). ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras. IEEE Transactions on Robotics. PP. 10.1109/TRO.2017.2705103.

[6] J. Engel, V. Koltun and D. Cremers, "Direct Sparse Odometry," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 3, pp. 611-625, 1 March 2018. doi: 10.1109/TPAMI.2017.2658577

[7] Engel, Jakob & Schoeps, Thomas & Cremers, Daniel. (2014). LSD-SLAM: large-scale direct monocular SLAM. Eur. Conf. Comput. Vis.. 8690. 1-16. 10.1007/978-3-319-10605-2_54.

[8] Pirchheim, Christian & Schmalstieg, Dieter & Reitmayr, Gerhard. (2013). Handling pure camera rotation in keyframe-based SLAM. *2013 IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2013*. 229-238. 10.1109/ISMAR.2013.6671783.

[9] Choi, Sunglok & Kim, Taemin & Yu, Wonpil. (2009). Performance evaluation of RANSAC family. Proceedings of the British Machine Vision Conference 2009. 24. 10.5244/C.23.81.

[10] Raguram, Rahul & Chum, Ondrej & Pollefeys, Marc & Matas, Jiri & Frahm, Jan-Michael. (2013). USAC: A Universal Framework for Random Sample Consensus. IEEE transactions on pattern analysis and machine intelligence. 35. 2022-2038. 10.1109/TPAMI.2012.257.

[11] P.H.S. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. Computer Vision and Image Understanding, 78:138–156, 2000.

[12] Ondrej Chum, Jiri Matas, and Stepan Obdrzalek. Enhancing RANSAC by generalized model optimization. In Proceedings of the Asian Conference on Computer Vision (ACCV), 2004.

[13] P.H.S. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. International Journal of Computer Vision, 50(1):35–61, 2002.

[14] D.R. Myatt, P.H.S Torr, S.J. Nasuto, J.M. Bishop, and R. Craddock. NAPSAC: High noise, high dimensional robust estimation - it's in the bag. In Preceedings of the 13th British Machine Vision Conference (BMVC), pages 458–467, 2002.

[15] J. Matas and O. Chum. Randomized RANSAC with sequential probability ratio test. In Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV), 2005.

[16] Anton Konouchine, Victor Gaganov, and Vladimir Veznevets. AMLESAC: A new maximum likelihood robust estimator. In Proceedings of the International Conference on Computer Graphics and Vision (GrapiCon), 2005.

[17] A. Fischler, Martin & C. Bolles, Robert. (1981). Random Sample Consensus: A Paradigm for Model Fitting with Applications To Image Analysis and Automated Cartography. Communications of the ACM. 24. 381-395. 10.1145/358669.358692.