DIRSAC: A Directed Sample and Consensus Algorithm for Localization with Quasi-Degenerate Data

Chris L Baker[a,b], William Hoff[c]

[a]National Robotics Engineering Center (chris@chimail.net)
[b]Neya Systems (chrisb@neyasystems.com)
[c]Colorado School of Mines (whoff@mines.edu)

Abstract

We propose a new method that uses an iterative closest point (ICP) algorithm to fit three-dimensional points to a prior geometric model for the purpose of determining the position and orientation (pose) of a sensor with respect to a model. We use a method similar to the Random Sample and Consensus (RANSAC) algorithm. However, where RANSAC uses random samples of points in the fitting trials, DIRSAC DIRects the sampling by ordering the points according to their contribution to the solution constraints. This is particularly important when the data is quasi-degenerate; meaning that some of the degrees of freedom of the pose are under constrained. In this case, the standard RANSAC algorithm often fails to find the correct solution. Our approach uses mutual information to avoid redundant points that result in degenerate sample sets. We demonstrate our approach on real data and show that in the case of quasi-degenerate data, the proposed algorithm outperforms RANSAC.

Keywords: RANSAC, ICP, quasi-degenerate, mutual information, 3D model fitting, point cloud

1 Introduction

A challenge for real world robotics applications is the estimation of the location and orientation (pose) of the robot in a local environment. For any robotic system to perform meaningful tasks, accurate localization information is crucial. In this paper, we focus on applications that require localization with respect to a known *a priori* surface model. We do not address the problem of continual pose tracking; rather, we focus on determining accurate localization given some rough initial pose estimate.

Recent availability of relatively low cost range sensors have led to many approaches that use range information to estimate pose. One common approach fits the sensed three-dimensional (3D) points to a prior model using an Iterative Closest Point (ICP) algorithm [1]. This approach can suffer when the data is contaminated by outliers. Traditional statistical approaches have been used to improve

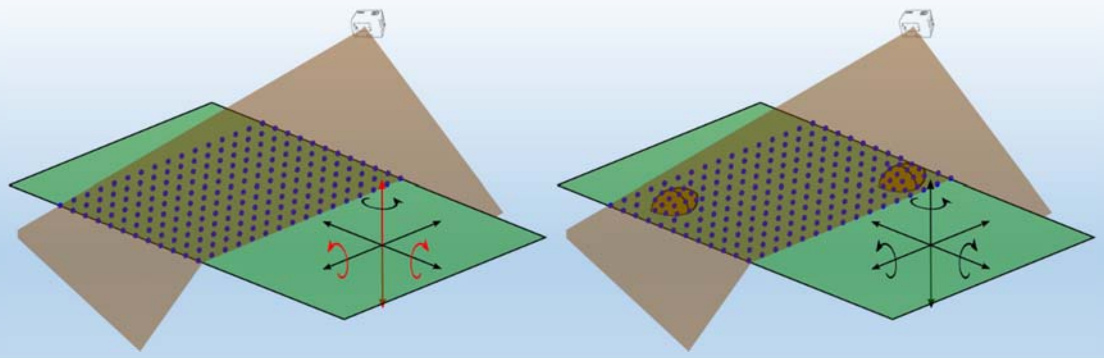


Figure 1: Illustration of a 3D laser scan of a planar surface. The left shows a fully degenerate case. The three dimensions in red are constrained, while the dimensions in black are completely unconstrained. In the right case, all six dimensions are fully constrained, but only by a very small percentage of the data. We refer to this as a “quasi-degenerate” case.

robustness to this noisy data. For example, a RANdom Sampling And Consensus (RANSAC) approach has been demonstrated to provide good results in noisy environments. RANSAC has been used in many contexts, and was first introduced in [2].

It is common for the available data to be in a degenerate, or quasi-degenerate state. By degenerate, we mean that some of the solution dimensions are not constrained by the data. By “quasi-degenerate”, we mean the case where the constraining data is indeed present, but is a small percentage of the available data, such that it would likely be omitted or considered outlier data if a random subset were chosen to compute a fit.

Consider the pathological case of an infinite plane. The constraints are along the direction orthogonal to the surface, and the orientation about the axes contained within the planar surface, shown in red in the left of Figure 1. The unconstrained dimensions are the rotation about the surface normal and the position along the dimensions parallel to the plane.

Consider the addition of small non-planar features as shown in the right of Figure 1. If points are sampled from these features, they would completely constrain the pose. The current naïve RANSAC algorithm is overwhelmed by the preponderance of planar data and would likely not consider the small but critical constraints afforded by the non-planar points. Thus, the algorithm may never select a sample set that includes the critical nonplanar points (or, may take a long time to find it). The resulting solution could be completely erroneous even though it may have low residual error.

The problem is that the naïve RANSAC algorithm is very likely to pick sample sets that are degenerate. Some of the points in such sets are redundant, in the sense that they do not add any new information in terms of constraining the solution. While there are methods to detect the degeneracy of such sample sets (as described in Section 2), it would be better to avoid selecting redundant points in the first place. If redundant points could be avoided, then there is less risk of the algorithm finding an erroneous solution.

As a side note, the example shown in Figure 1 is still not fully constrained due to a symmetry of rotation of 180 degrees around the axis orthogonal to the plane. We make the distinction here between a discrete and continuous degeneracy. A discrete degeneracy results from multiple solutions that are separated by large discrete steps in the solution space, whereas a continuous degeneracy results from equally good solutions along continuous motion in the solution space. This work focusses on the continuous degeneracies and therefore is not concerned with degeneracies due to symmetries.

This paper proposes an algorithm that selects points in such a way as to avoid degenerate sample sets. Instead of a purely random point selection, we direct the selection of points to avoid information redundancy. The algorithm, called DIRSAC, DIRects point selection within a SAmple and Consensus framework. The main idea is that we evaluate each point based on its ability to constrain the pose of the solution. We identify redundant points by computing the mutual information between points. Experimental results show significant improvements over the naïve RANSAC algorithm. A preliminary version of this work was published in [3].

The remainder of this paper is organized as follows. Section 2 discusses the current state of the art related to localization and fitting of point clouds to prior models. Section 3 describes the overall approach and Section 4 provides a simple example. Section 5 gives the details of the full approach to solve the localization problem in noisy, quasi-degenerate data. In Section 6, we describe the system configuration used to generate experimental data and describe the models used for our analysis. Results are provided in Section 7, and Section 8 provides additional discussion.

2 Previous Work

One approach for registering 3D sensed point data to surface model data is to find a rigid transformation between a small number of local data points and corresponding points on the surface model. Chen, et al randomly pick a data point and hypothesize a corresponding point on the model [4]. Then they pick a small number of nearby data points and find model points that are in the same relative positions as the data points. After these control points are successfully matched, a rigid transformation is computed and used to transform all remaining points. The solution that aligns the largest number of points is taken to be the correct solution. In the case of quasi-degenerate data, this algorithm would have a similar problem to RANSAC, in that many of subsample sets would be degenerate.

Another approach is to compute a feature vector for local neighborhoods, and then match feature vectors. Features such as "spin image" descriptors [5] and local curvature models (*e.g.,* [6]) have been used. Finding correspondences is more difficult if there are outliers in the data. Outliers are sensed points with large range errors, which can be caused by a variety of sensor effects and un-modeled objects. If the proportion of outliers is small, then robust estimation techniques can still find the correct fit of the data to the model. If the proportion of outliers is high, then another method must be used.

If a good initial guess of the pose is available, then other methods can be used. Civera, et al develop a Kalman-filter based algorithm that uses prior probabilistic information on the state of the sensor to limit the size of the random sample sets [7]. In fact, using only a single random sample is required to instantiate the model. It is not clear how this algorithm would perform in the case of quasi-degenerate data.

Similar to [8], our approach uses the classic Iterative Closest Point (ICP) algorithm within a Random Sample and Consensus (RANSAC) framework. This approach automatically finds correspondences and can tolerate a large fraction of outliers. The RANSAC algorithm randomly selects a minimum subset of points to compute a fit. The ICP algorithm then computes the point correspondences to the model, and the pose of the model in the reference frame of the sensor. Once computed, the pose information is used to transform the remaining points to the reference frame of the sensor, and the residual error between the transformed points and the measured points is measured. In addition to the residual error, a set of inliers is identified as points that fall within some error threshold. This process repeats until a reasonable probability exists that a good solution has been computed. The list of plausible solutions are searched in order of increasing error until a solution with enough inliers has been found. The method does require a good initial guess in order to converge similar to ICP. If a good initial guess is not available, there are several techniques (e.g. feature matching) that can be used. Many applications that require a precise localization within quasi-degenerate data will already employ other methods for gross localization. These methods are not the focus of this work and we assume an initial good guess is available.

In [8] the authors evaluate a few different methods of computing the pose of a robot by matching scans between different sensor views. They evaluate a set of filters to process the data prior to running the standard ICP algorithm, including RANSAC and SIFT, and their combination. Noting that the RANSAC algorithm improves accuracy, while SIFT is faster, they demonstrate that a combination of the two will likely produce a balanced approach. As several ICP algorithms do, their approach depends on matching scans point-to-point, whereas we have point-to-model matching. Although their results are not directly applicable, they do note that RANSAC and ICP together do perform better than ICP alone.

Significant effort has been made to improve the runtime speed of RANSAC while still guaranteeing a correct solution with some probability. Reviews of approaches to improving RANSAC are provided in [9] and [10]. Some approaches attempt to optimize the model verification step. For example, Matas and Chum [11] have designed a pre-evaluation stage that attempts to quickly filter out bad hypotheses. The concept of early-termination was also extended by Capel [12].

Another strategy is to bias the point selection such that points with a higher inlier probability will be treated preferentially in the random point selection process. For example [13] orders the points based on image feature matching scores, using the assumption that a higher similarity in the feature match will increase the likelihood of the point match being an inlier. Once the points are sorted, they perform RANSAC over a PROgressively larger subset from the top of this sorted set of points (PROSAC). A related method is that of [14], which groups points based on optical flow or using the result of image segmentation. Of course, these approaches are only applicable if information derived from image features is available. In our application, we assume no information is available other than the positions of the 3D sensed points in the sensor's reference frame, and a rough estimate of the sensor's pose relative to the surface model.

Approaches to help speed ICP to convergence when constraining data is sparse have also been proposed, such as [15], which weights points according to their additional geometric information. They essentially compute an interest metric for each point based on the variance of the normals of surrounding points, scaled by the distance. Using this as a weighting scheme for ICP, they demonstrate a better convergence rate and accuracy for situations where very little geometric structure would otherwise constrain the ICP fit. While they are not using this in the context of a RANSAC algorithm, this type of weighting scheme could improve convergence, especially in a quasi-degenerate environment.

Notably, our method of determining the redundant information of points could also be used in a similar way to weight ICP. The main difference in our method and their method being that we are selecting a subset of points which guarantee fully constrained fits if such constraints are available in the data. Their method simply prefers points with a high interest metric, but makes no claims on the actual ability of those points to constrain the ICP solution.

Despite the efforts mentioned above to speed up processing, RANSAC can still be slow, especially when the fraction of inliers is low. If λ is the fraction of inliers, then the number of iterations S to find a good solution (*i.e.* a sample set with no outliers) with probability η must be at least

$$S = \frac{\log(1-\eta)}{\log(1-\lambda^n)} \quad 1$$

where n is the size of each sample [16].

In the case of quasi-degenerate data, a sample set must be found that not only contains all inliers, but also is not degenerate. For this, the required number of RANSAC iterations can be much larger as Frahm et al. described in [17]. Figure 2 illustrates a comparison. The left plot shows the number of required iterations to produce a good solution with 99.5% probability, for various sample set sizes. However, when the data is quasi-degenerate, the figure on the right shows how the curves for $n = 6$ points change for varying levels of degeneracies.

Frahm's approach to mitigate this assumes that the model estimation problem can be expressed as a linear relationship $\boldsymbol{Ax} = 0$, where $\boldsymbol{x}$ is the vector of parameters to be fit, and $\boldsymbol{A}$ is a matrix containing the data from a sample set of matches. They describe an algorithm which can detect degeneracy in $\boldsymbol{A}$, and are able to estimate the actual number of degrees of freedom in the data. However, they evaluate the sample set after the points have been selected to determine if the selected points are in a degenerate configuration. This approach requires searching through the model constraint space in a non-trivial way which is costly. We would like to avoid degenerate configurations altogether by choosing points in such a way that they will provide us with non-degenerate solutions. Furthermore, the assumption of a linear data fitting relationship is not applicable to our problem of finding a rigid body transformation using ICP.

Although not specifically developed for RANSAC, in [18], Davison develops an algorithm to select points based on a mutual information constraint. He then uses this to determine which point, in a feature tracking context, would provide the most information to the state if used as an observation. As described in the next section, our approach extends this work and applies it to the current context of matching 3D data with a prior model. Using this, we are able to sort the 3D points by their added information and direct our sampling.

3 Overall Approach

The typical naïve RANSAC algorithm is unlikely to find a correct solution within the expected iterations given by Equation 1 in the presence of quasi-degenerate data. Quasi-degenerate data is

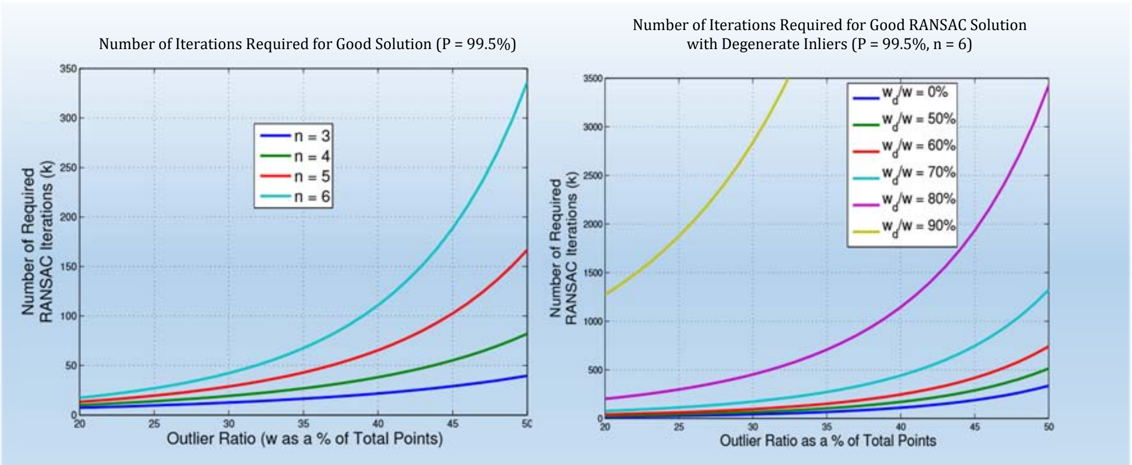


Figure 2: Comparison of the naïve version of RANSAC (left) which assumes any point subset will be a constraining subset, and (right) consideration for many subsets actually being degenerate subsets.

problematic when computing a RANSAC solution because the random selection process will result in many erroneous solutions computed with a completely degenerate set of data. In order to maintain the random nature of RANSAC, while avoiding solutions computed from redundant data, we need to direct the point sampling by avoiding groups of points that do not adequately constrain the pose solution.

We direct the point selection process in DIRSAC by computing the mutual information between sets of points which provides us with the information content overlap between the two point sets. We direct the point selection in DIRSAC as follows. First we randomly select a point and compute its mutual information with each of the remaining points. We label points with a mutual information greater than a threshold as redundant. In our experiments, we empirically found the mean of all the mutual information scores to be a good threshold. The algorithm was not particularly sensitive to the choice of threshold, and other metrics could be used such as the median. Once redundant points have been identified, we remove them from the set of available points and randomly pick from the remaining non-redundant points. A new set of mutual information scores is calculated between the currently selected points and all remaining points. Redundant points are again identified and removed for the next random selection. This process continues until we have chosen enough points to constrain the solution.

As in RANSAC, this approach is robust to outliers in the data because of the random selection of points. However, unlike the simple naïve RANSAC method, our approach avoids selecting redundant data and promotes individual trial fits with strong constraints. With this approach, the number of iterations needed to find a correct solution should be reduced.

4 Simple Example

To provide an intuitive understanding of the overall approach, consider the simple example shown in Figure 3. This figure shows a one-dimensional "surface" model, which is just a curve in the XY plane. The surface is classified as quasi-degenerate, since it consists of a straight line with only a small "bump" to constrain the pose. The surface is composed of a set of facets, and the normal vector at each facet is shown.

A sensor is positioned above the model. The pose of the sensor is uncertain, and is represented by a covariance matrix. The translational uncertainty of the sensor pose is depicted as an ellipse. The rotational uncertainty of the sensor pose is depicted as a "cone" about the coordinate axes of the sensor. The sensor samples points from the surface, shown as the red dots.

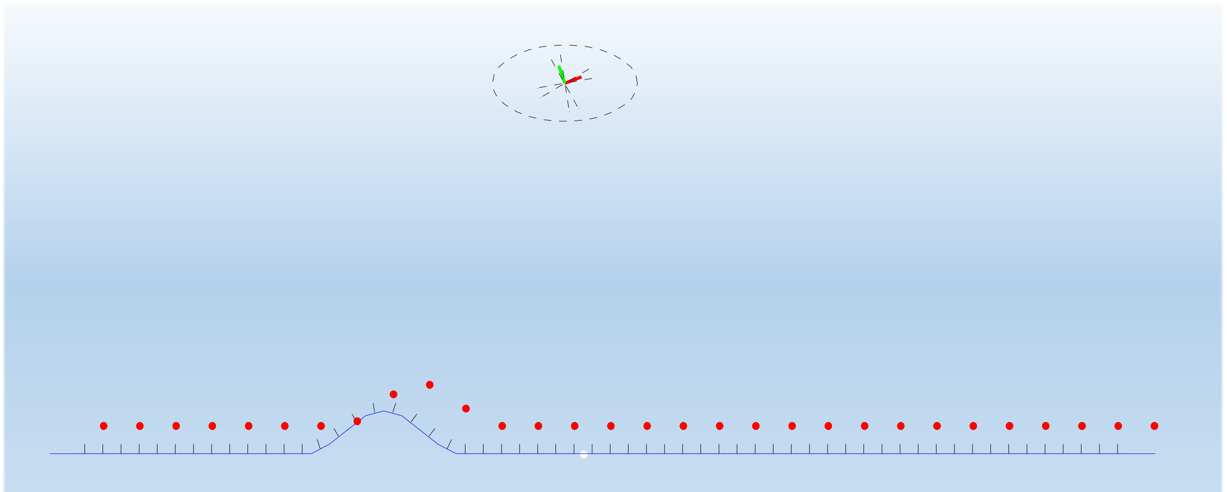


Figure 3: A simple one-dimensional surface, consisting of a set of facets. The surface normal vectors at each facet are shown. A sensor samples points from the surface, shown as the red dots.

Due to the initial error in the sensor's pose, the estimated locations of the sampled points do not align with the corresponding model points. However, alignment can be done by choosing a random subset of points and then performing the ICP algorithm to align the subset of points to the model (and thus determine the true sensor pose). The reason that a random subset is used, rather than the entire point set, is because there may be outliers in the sensed points, which can corrupt the result (in the simple example shown, there are no outliers).

However, the choice of the subset of points greatly influences the ability of the ICP algorithm to converge to the correct pose. Consider the naïve sampling of the three points shown in Figure 4 (top). These three points are all chosen from the flat part of the surface. The closest corresponding points on the model are shown.

After one iteration, the ICP algorithm aligns the three points to the model, as shown in Figure 4 (bottom), and terminates. However, the alignment is obviously not correct, as can be seen by the positions of the other points. The reason is that the three points do not sufficiently constrain the pose of the sensor.

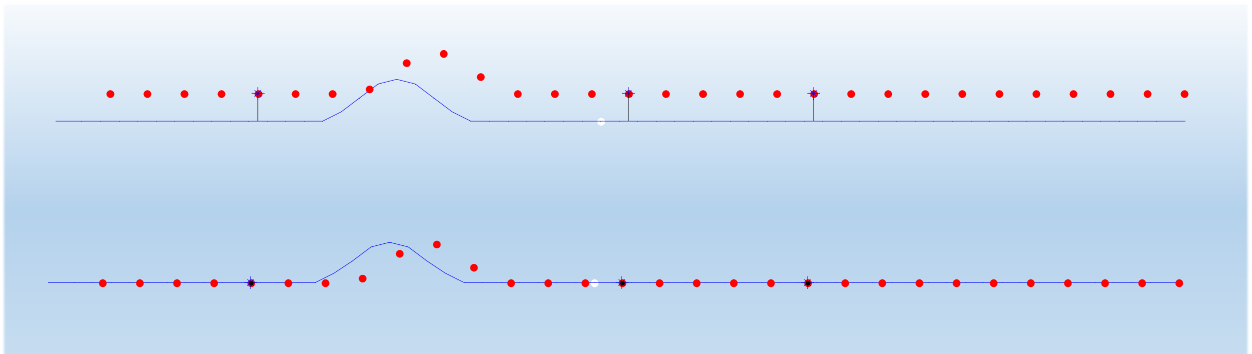


Figure 4: Performance of the ICP algorithm with a naïve sampling of points. (Top) Initial point positions, with closest corresponding points shown. (Bottom) Final convergence of ICP algorithm after 2 iterations.

Now, consider an alternative approach that uses mutual information to select the points. The first point is still chosen at random, and is marked "1st" in Figure 5. Then the mutual information between this point and all other points is computed, using Equation 2, which is described below in Section 5 .

We will avoid selecting additional points that have a large value of mutual information with the previously selected point, since these points provide very little new information to constrain the pose. As can be seen, points that are near to the first point have a large value of MI. The point with the minimum mutual information is on the “bump” and is marked with an arrow in the figure.

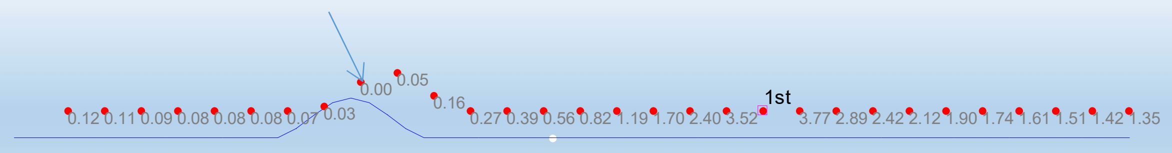


Figure 5: The mutual information between the first point and all other points are shown. The point with the minimum mutual information is marked with an arrow.

As described in Section 5, our algorithm is biased towards selecting points with low MI with previously selected points. Assume that the point marked with an arrow is selected. Next, the MI between the first two points and all the remaining points is computed. The result is shown in Figure 6. As can be seen, the next best point is also on the “bump” since this point is also useful for constraining the pose.

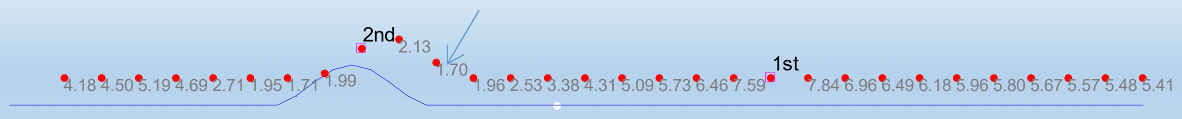


Figure 6: The mutual information between the first two points and all other points are shown. The point with the minimum mutual information is marked with an arrow.

Using these three points in the ICP algorithm results in a much better alignment. Figure 7 shows the convergence of the ICP algorithm. The top figure shows the initial alignment. The bottom figure shows the final alignment after 21 iterations.

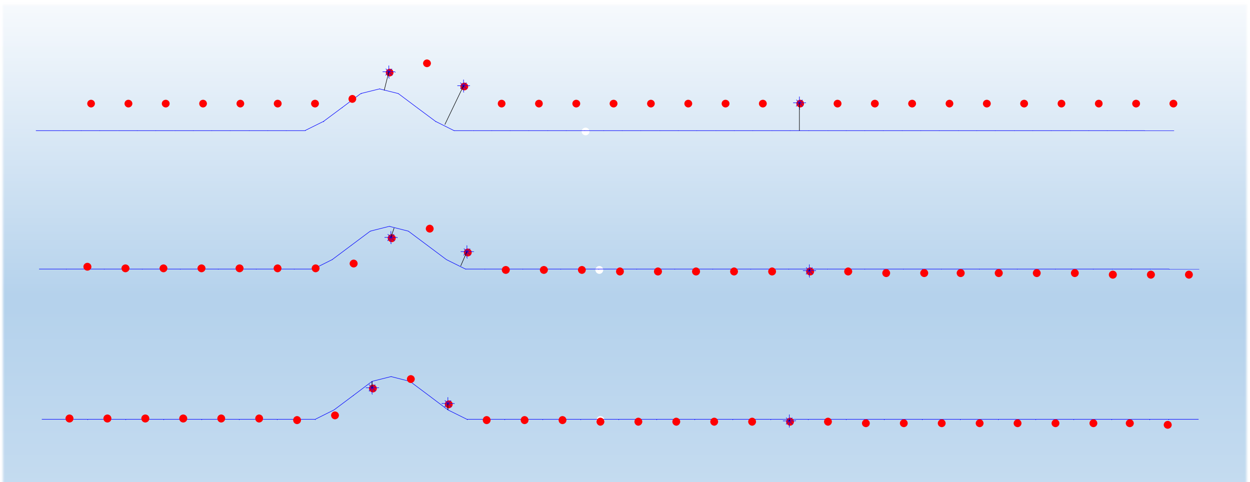


Figure 7: Performance of the ICP algorithm with directed point sampling. (Top) Initial point positions, with closest corresponding points shown. (Middle) Alignment after first iteration. (Bottom) Final convergence of ICP algorithm after 21 iterations.

5 Detailed Approach

Our goal will be to determine, given a set of selected points, what is the additional information gained by selecting an additional point. Following the derivation of Davison, in [18], we formulate the mutual information in terms of the overlapping information between range measurements, ρ_i and ρ_j and their constraints on the computed pose.

$$I(\rho_i;\rho_j) = \frac{1}{2}\log_2 \frac{|C_{\rho_i\rho_i}|}{\left|C_{\rho_i\rho_i} - C_{\rho_i\rho_j}C_{\rho_j\rho_j}^{-1}C_{\rho_j\rho_i}\right|}, \qquad 2$$

where $C_{\rho_i\rho_j}$ is the covariance between range measurement ρ_i and ρ_j. For our range measurements, we assume the error in the direction to each point is negligible and the only error is the measurement of the range. This is typical of many range sensors, which have the largest component of uncertainty along the direction to the point.

In order to implement the approach described we must first understand the Jacobian relating how changes in the pose affect changes in the range measurements. This is described in Section 5.1. We will see that this Jacobian is dependent on, among other things, an estimate of the surface normal associated with the measurement. In order to estimate this uncertainty we develop a probabilistic association of the point measurement with the model in Sections 5.2 and 5.3. This results in a probabilistic Jacobian and allows us to compute the mutual information between subsets of points, which leads to a model constraint score (Section 5.4) and finally the full DIRSAC algorithm (Section 5.5).

5.1 Relating changes in the pose to changes in range measurements

Assume that we have approximate knowledge of the pose of the sensor. The pose can be represented as 6-element vector $\boldsymbol{x}$, consisting of translation (x, y, z) and rotation (α, β, γ). We choose to adhere to one of the Euler rotation conventions, using the Z-Y-X rotation order detailed in [19], though any standard angle convention will suffice. We also have the model, represented by a set of small planar facets. We can predict a range measurement (ρ) in any direction by intersecting the ray with the model. Let $g(\boldsymbol{x})$ be a function that returns the predicted range for a given point; *i.e.*, $\rho = g(\boldsymbol{x})$. For small perturbations of $\boldsymbol{x}$, we can write the resulting perturbation of ρ. After a Taylor series expansion and linearization we have

$$\delta\rho = \frac{\partial g}{\partial \mathbf{x}}\delta\boldsymbol{x} = \vec{J}\delta\boldsymbol{x} \qquad 3$$

where $\vec{J}$ is the Jacobian which describes how small changes in $\boldsymbol{x}$ will affect the range measurement ρ. $\vec{J}$ is given by

$$\vec{J} = \begin{bmatrix} \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} & \frac{\partial g}{\partial z} & \frac{\partial g}{\partial \alpha} & \frac{\partial g}{\partial \beta} & \frac{\partial g}{\partial \gamma} \end{bmatrix} \qquad 4$$

The following derivation of $\vec{J}$ treats the translation and rotation components separately.

5.1.1 Translational Components

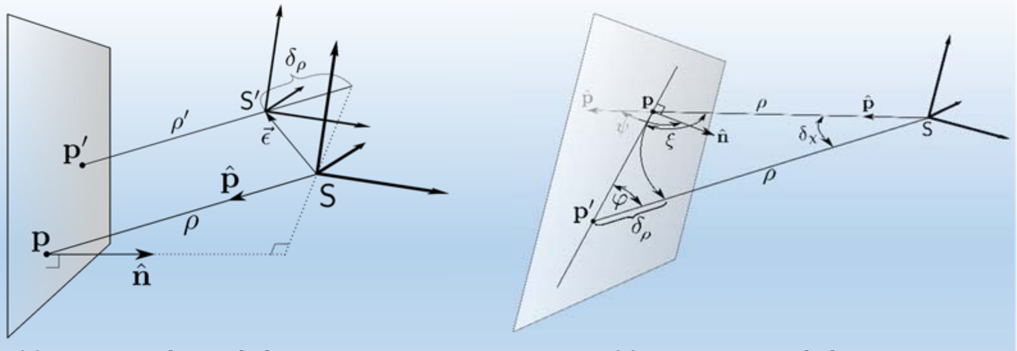
We compute the first three elements of $\vec{J}$ in Equation 4 by perturbing the sensor's position by $\vec{\epsilon}$. From Figure 8(a), the projection of $\mathbf{p}$ onto $\hat{\boldsymbol{n}}$ is the same as the projection of $\vec{\epsilon} + \hat{\boldsymbol{p}}\rho'$ onto $\hat{\boldsymbol{n}}$, or $\rho\hat{\boldsymbol{p}} \cdot \hat{\boldsymbol{n}} = [\vec{\epsilon} + \hat{p}\rho'] \cdot \hat{\boldsymbol{n}}$, where $\rho' = \rho - \delta\rho$, so $\vec{\epsilon} \cdot \hat{\boldsymbol{n}} = \delta_\rho \hat{\boldsymbol{p}} \cdot \hat{\boldsymbol{n}}$**.** Decomposing $\vec{\epsilon}$ in the $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$ dimensions, we have

$$For\ \vec{\epsilon} = (\epsilon_x, 0{,}0){:}\ \epsilon_x\hat{n}_x = \delta_\rho\hat{p}\cdot\hat{n} \Rightarrow \frac{\partial g}{\partial x} = \frac{\delta_\rho}{\epsilon_x} = \frac{\hat{n}_x}{\hat{p}\cdot\hat{n}} \qquad 5$$

$$For\ \vec{\epsilon} = \left(0, \epsilon_y, 0\right){:}\ \epsilon_y\hat{n}_y = \delta_\rho\hat{p}\cdot\hat{n} \Rightarrow \frac{\partial g}{\partial y} = \frac{\delta_\rho}{\epsilon_y} = \frac{\hat{n}_y}{\hat{p}\cdot\hat{n}} \qquad 6$$

$$For\ \vec{\epsilon} = (0{,}0, \epsilon_z){:}\ \epsilon_z\hat{n}_z = \delta_\rho\hat{p}\cdot\hat{n} \Rightarrow \frac{\partial g}{\partial z} = \frac{\delta_\rho}{\epsilon_z} = \frac{\hat{n}_z}{\hat{p}\cdot\hat{n}} \qquad 7$$

where $\hat{\epsilon} = \left[\epsilon_x, \epsilon_y, \epsilon_z\right]^T$ and $\hat{n} = \left[n_x, n_y, n_z\right]^T$. This provides us with the translational components of the Jacobian, which are the first three elements shown in Equation 4.



(a) Translational Elements (a) Rotational Elements

Figure 8: Components necessary to derive the translational and rotational elements of the $\vec{J}$. A point $\mathbf{p}$ is measured in the sensor's coordinate frame S in the direction of the unit vector $\hat{\boldsymbol{p}}$. The normal to the plane is $\hat{\boldsymbol{n}}$. The angle between the plane's normal and the point direction $\hat{\boldsymbol{p}}$ is ψ.

5.1.2 Rotational Components

Employing the law of sines and the small angle approximation with Figure 8(b), we can show that

$$\frac{\partial g}{\partial \chi} = -\rho \frac{\sin\psi}{\cos\psi} = -\rho \frac{\hat{n} \times \hat{p}}{\hat{n} \cdot \hat{p}}. \qquad 8$$

Decomposing for each angular element we have the following.

$$\frac{\partial g}{\partial \chi_\alpha} = \rho \frac{\hat{n}_z\hat{p}_y - \hat{n}_y\hat{p}_z}{\hat{n} \cdot \hat{p}}, \frac{\partial g}{\partial \chi_\beta} = \rho \frac{\hat{n}_x\hat{p}_z - \hat{n}_z\hat{p}_x}{\hat{n} \cdot \hat{p}}, \frac{\partial g}{\partial \chi_\gamma} = \rho \frac{\hat{n}_y\hat{p}_x - \hat{n}_x\hat{p}_y}{\hat{n} \cdot \hat{p}}. \qquad 9$$

We can now write the full expression for the Jacobian from Equation 4:

$$\vec{J} = \begin{bmatrix} \frac{\hat{n}_x}{\hat{p} \cdot \hat{n}} & \frac{\hat{n}_y}{\hat{p} \cdot \hat{n}} & \frac{\hat{n}_z}{\hat{p} \cdot \hat{n}} & \rho \frac{\hat{n}_z\hat{p}_y - \hat{n}_y\hat{p}_z}{\hat{n} \cdot \hat{p}} & \rho \frac{\hat{n}_x\hat{p}_z - \hat{n}_z\hat{p}_x}{\hat{n} \cdot \hat{p}} & \rho \frac{\hat{n}_y\hat{p}_x - \hat{n}_x\hat{p}_y}{\hat{n} \cdot \hat{p}} \end{bmatrix} \qquad 10$$

5.2 Covariance of Range Measurements

For N points, we can stack the $\vec{J}$'s from Equation 10 to form the relationship $\overrightarrow{\delta\rho} = \boldsymbol{J}\delta\boldsymbol{x}$, where $\boldsymbol{J}$ is an $N \times 6$ matrix, containing one row for each measurement ρ, and $\overrightarrow{\delta\rho}$ is the vector of range measurement changes resulting from the small changes in the pose ($\delta\boldsymbol{x}$). The covariance of $\overrightarrow{\delta\rho}$ is

$$C_{\overrightarrow{\delta\rho}} = E[\overrightarrow{\delta\rho}\overrightarrow{\delta\rho}^T] = E[\boldsymbol{J}\delta\boldsymbol{x}\delta\boldsymbol{x}^T\boldsymbol{J}^T], \qquad 11$$

where $E[\cdot]$ is the expected value operator. $\boldsymbol{J}$ is not a random variable and we can remove it from within the expected value and write

$$C_{\overrightarrow{\delta\rho}} = \boldsymbol{J}C_{\boldsymbol{x}}\boldsymbol{J}^T = \begin{bmatrix} C_{\rho_1\rho_1} & C_{\rho_1\rho_2} & \cdots & C_{\rho_1\rho_N} \\ C_{\rho_2\rho_1} & C_{\rho_2\rho_2} & \cdots & C_{\rho_2\rho_N} \\ \vdots & \vdots & \ddots & \vdots \\ C_{\rho_N\rho_1} & C_{\rho_N\rho_2} & \cdots & C_{\rho_N\rho_N} \end{bmatrix} \qquad 12$$

where $C_{\rho_i\rho_j} = \vec{J}_i C_{\boldsymbol{x}} \vec{J}_j^T$ for the ith and jth points. This covariance matrix for $\overrightarrow{\delta\rho}$ contains the elements required to compute the mutual information as in Equation 2.

If the covariance of the pose, $C_{\boldsymbol{x}}$ is known *a priori*, it can be used in Equation 12. However, if the *a priori* covariance is not known, we simply use a large covariance for $C_{\boldsymbol{x}}$, meaning that all the degrees of freedom are assumed to be completely unconstrained. For example, the translation uncertainty can be considered to be the size of the workspace. Similarly, the orientation uncertainty can be considered to be the full range of possible orientations. In our experiments, we use a standard deviation of $10\,m$ and 150° for position and orientation uncertainty. The absolute scale of the covariance is not important as it cancels out in the mutual information computation.

5.3 Estimation of Point Normals

The Jacobian from Equation 10 is dependent on the direction to the point $\hat{\boldsymbol{p}}$, the range to the point ρ, and the local model normal $\hat{\boldsymbol{n}}$. The sensor provides $\hat{\boldsymbol{p}}$ and ρ directly. We must estimate $\hat{\boldsymbol{n}}$. One possible method would be to estimate $\hat{\boldsymbol{n}}$ using the point cloud data. However, for noisy data, and possible un-modeled observed structure, this can be problematic. For these, the normals will erroneously imply good constraints, where the *a priori* model does not support them.

A better approach associates points with the model. This can be problematic as well since we do not know the correct associations. We can instead compute the most likely normal by probabilistically associating the sensed point with the model. As described in the next sub-sections, we start with the uncertainty of the sensor's measurement system, incorporate the uncertainty of the sensor's pose in the world, and finally associate the uncertainty of the point in the world with the model.

5.3.1 Computation of Point Covariance

The sensor provides a set of range measurements in the form of a 3D point cloud, $\boldsymbol{P}$, in the sensor's reference frame, S. Let $\boldsymbol{p}$ be a point in the cloud; then $\widehat{\boldsymbol{p}}$ is a unit vector in the direction from the origin of the sensor to $\boldsymbol{p}$, and ρ is the range to $\boldsymbol{p}$ such that $\boldsymbol{p} = \rho\widehat{\boldsymbol{p}}$. We would like to associate each point, $\boldsymbol{p}$, with a point $\boldsymbol{q}$ on the surface of a prior model, M, defined in the world's reference frame, W. Recall that $\boldsymbol{x}$ is the pose of the sensor while capturing $\boldsymbol{P}$. We can use the uncertainty of $\boldsymbol{p}$ in S, and the uncertainty of $\boldsymbol{x}$ to determine the uncertainty of $\boldsymbol{q}$. Let $\boldsymbol{q} = \boldsymbol{f}(\boldsymbol{x},\boldsymbol{p}) = \boldsymbol{R}\boldsymbol{p} + \boldsymbol{t}$ be the function that transforms $\boldsymbol{p}$ from the coordinate frame of the sensor to the coordinate frame of the model, where $\boldsymbol{R}$ is a 3×3 rotation matrix and $\boldsymbol{t}$ is the translation of the sensor with respect to the model. The Taylor series expansion and linearization around small perturbations of $\boldsymbol{p}$ and $\boldsymbol{x}$ give us the change in the position, $\delta\boldsymbol{q}$, as a function of changes in $\boldsymbol{p}$ and $\boldsymbol{x}$.

$$\delta\boldsymbol{q} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\delta\boldsymbol{x} + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{p}}\delta\boldsymbol{p} = J_x\delta\boldsymbol{x} + J_p\delta\boldsymbol{p} \qquad 13$$

Since the error in the sensor's measurement is independent of the pose $\boldsymbol{x}$, the cross covariance terms will be zero and the covariance of $\boldsymbol{q}$ in W is

$$C_{\boldsymbol{q}} = E[\delta\boldsymbol{q}\delta\boldsymbol{q}^T] = J_{\boldsymbol{x}}C_{\boldsymbol{x}}J_{\boldsymbol{x}}^T + J_{\boldsymbol{p}}C_{\boldsymbol{p}}J_{\boldsymbol{p}}^T \qquad 14$$

where $C_{\boldsymbol{x}}$ is the provided covariance of the sensor's pose $\boldsymbol{x}$ in W, and $C_{\boldsymbol{p}}$ is the covariance of $\boldsymbol{p}$ in S. Thus we are left to find the Jacobians $J_{\boldsymbol{x}}$ and $J_{\boldsymbol{p}}$.

5.3.2 Computation of Jacobian $J_{\boldsymbol{x}}$

Using the Z-Y-X Euler rotation convention, the rotation matrix is (using the shorthand notation of $cos \equiv c$, and $sin \equiv s$)

$$R_S^W = \begin{bmatrix} c\gamma c\beta & -s\gamma c\alpha + c\gamma s\beta s\alpha & s\gamma s\alpha + c\gamma s\beta c\alpha \\ s\gamma c\beta & c\gamma c\alpha + s\gamma s\beta s\alpha & -c\gamma s\alpha + s\gamma s\beta c\alpha \\ -s\beta & c\beta s\alpha & c\beta c\alpha \end{bmatrix} \qquad 15$$

Taking the partial derivative of this with respect to the elements in $\boldsymbol{x}$ gives us the Jacobian $J_{\boldsymbol{x}}$.

$$J_{\boldsymbol{x}} = \begin{bmatrix} \frac{\partial q_x}{\partial x} & \frac{\partial q_x}{\partial y} & \frac{\partial q_x}{\partial z} & \frac{\partial q_x}{\partial \alpha} & \frac{\partial q_x}{\partial \beta} & \frac{\partial q_x}{\partial \gamma} \\ \frac{\partial q_y}{\partial x} & \frac{\partial q_y}{\partial y} & \frac{\partial q_y}{\partial z} & \frac{\partial q_y}{\partial \alpha} & \frac{\partial q_y}{\partial \beta} & \frac{\partial q_y}{\partial \gamma} \\ \frac{\partial q_z}{\partial x} & \frac{\partial q_z}{\partial y} & \frac{\partial q_z}{\partial z} & \frac{\partial q_z}{\partial \alpha} & \frac{\partial q_z}{\partial \beta} & \frac{\partial q_z}{\partial \gamma} \end{bmatrix} \qquad 16$$

where $\boldsymbol{q} = [q_x, q_y, q_z]^T$ and $\boldsymbol{x} = [x, y, z, \alpha, \beta, \gamma]^T$.

5.3.3 Computation of Jacobian J_p

$J_{\boldsymbol{p}}$ in Equation 14 relates changes in $\boldsymbol{q}$ due to changes in $\boldsymbol{p}$. We use a spherical coordinate system, (ρ, θ, φ), such that $\boldsymbol{p} = [\rho \sin\varphi \cos\theta, \rho \sin\varphi \sin\theta, \rho\cos\varphi]^T$. The covariance in spherical coordinates is all zero except for the upper left entry, which is σ_ρ^2; ***i.e.***, the variance of the range noise of the sensor. We can write J_p by taking the partial derivatives with respect to $\boldsymbol{p}$.

$$J_{\boldsymbol{p}} = R_S^W \begin{bmatrix} s\phi c\theta & -\rho s\phi s\theta & \rho c\phi c\theta \\ s\phi s\theta & \rho s\phi c\theta & \rho c\phi s\theta \\ c\phi & 0 & -\rho s\phi \end{bmatrix} \qquad 17$$

We now have all the necessary components to completely specify the covariance of q, $C_{\boldsymbol{q}}$, using Equation 14.

5.3.4 Probabilistic Jacobian

We now have a probabilistic association between $\boldsymbol{q}$ and M. However, there may be multiple facet associations within M. We can compute a probability of association to each facet and compute an expected value for our Jacobian by summing over all model facets $k = 1 \dots K$ and multiplying by the probability $(P_{i,k})$ that the i^{th} data point is associated with model facet k.

$$E[J(\hat{n}_i, \hat{p}_i, \rho_i)] = \sum_{k=1}^{K} J_i(\hat{n}_k, \hat{p}_i, \rho_i) P_{i,k} = \bar{J}_i \qquad 18$$

Using $\bar{J}_i$ for the point Jacobians in Equation 12, we can now solve for $C_{\delta\vec{\rho}}$ and compute the mutual information.

We note that many model facets will only have a negligible contribution to this summation because their $P_{i,k}$ will be very small. Thus, computing the weight of all facets on the model's surface mesh is unnecessary. A reasonable approximation is to look only at triangles within 3σ of $\boldsymbol{q}$. We use an efficient method to find the closest point on the model (M) to a 3D point in space when we know the uncertainty of the pose. From Equation 14, we have the covariance of the point in the world, $\boldsymbol{C_q}$. From this we can find the most probable association of $\boldsymbol{q}$ to M for use in the ICP algorithm. Essentially, we use the Mahalanobis distance to find the nearest neighbor of a data point on the model. We do this by transforming the model M, by a special matrix $\boldsymbol{A}$. We choose $\boldsymbol{A}$ such that the new covariance, $\boldsymbol{C_{q'}} = \boldsymbol{A C_q A^T}$, is equal to the identity matrix. This is referred to as a "whitening transform" [20], and is given by $\boldsymbol{A} = \boldsymbol{\Lambda}^{-\frac{1}{2}}\boldsymbol{VT}$, where $\boldsymbol{\Lambda}$ is the diagonal matrix whose elements are the eigenvalues of $\boldsymbol{C_q}$, and $\boldsymbol{V}$ is the corresponding set of eigenvectors. This transformation by $\boldsymbol{A}$ allows us to use existing fast tools to find the closest point on M while still considering the uncertainty of $\boldsymbol{q}$.

5.4 Model Constraint Score

Before we present the full DIRSAC algorithm, we would like to present a useful metric to quantify the degeneracy of a point set. Recall from Equation 3, for a given measured point, the relationship between a small deviation in the sensor's pose and the deviation in the range measurement is given by $\delta\rho = \vec{J}\delta\boldsymbol{x}$, and for many points stacked, this relationship is $\overrightarrow{\delta\rho} = \boldsymbol{J}\delta\boldsymbol{x}$. Another interesting result is to consider the solution for the change in the pose vector, $\delta\boldsymbol{x}$, using the pseudo-inverse as $\delta\boldsymbol{x} = \boldsymbol{J}^{+}\overrightarrow{\delta\rho}$, where $\boldsymbol{J}^{+} = (\boldsymbol{J}^T\boldsymbol{J})^{-1}\boldsymbol{J}^T$.

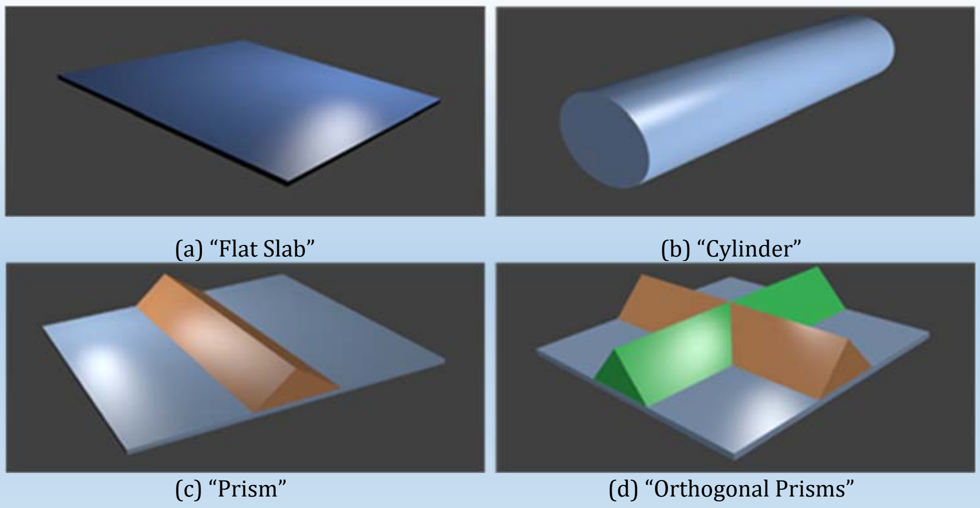


Figure 9: Four models to demonstrate the constraint score based on the singular values of $\boldsymbol{J}^T\boldsymbol{J}$. The eigenvalues of $\boldsymbol{J}^T\boldsymbol{J}$ are given in Table 1.

Computation of this pseudo-inverse requires the inversion of the 6×6 matrix, $\boldsymbol{J}^T\boldsymbol{J}$. In order to invert this, $\boldsymbol{J}^T\boldsymbol{J}$ must be full rank and the rows of $\boldsymbol{J}^T\boldsymbol{J}$ must be linearly independent. The degree to which we can or cannot compute $\boldsymbol{J}^+$ is determined by the condition number, $\kappa(\boldsymbol{J}^T\boldsymbol{J})$, computed as the ratio of the largest singular value of $\boldsymbol{J}^T\boldsymbol{J}$ to the smallest singular value. The larger $\kappa(\boldsymbol{J}^T\boldsymbol{J})$ is, the more ill-conditioned our problem becomes. We define a score in which larger numbers will imply a better constraint as

$$Score = \frac{100}{\kappa(\boldsymbol{J}^T\boldsymbol{J})} \qquad 19$$

This is a useful metric to evaluate the ability of any set of points to constrain their fit to an *a priori* model. To be full rank, the set must contain at least six points.

To illustrate the constraint score, we provide the singular values of $\boldsymbol{J}^T\boldsymbol{J}$ for four representative models in Table 1. Figure 9(a) is unconstrained in three dimensions, the two positional dimensions parallel to the plane, and the orientation about a normal to the plane. Figure 9(b) is unconstrained in two dimensions, position around the cylinder, and position parallel to the cylinder. Figure 9(c) is unconstrained in one dimension, the position parallel to the prism. Finally, Figure 9(d) is fully constrained except for symmetry, which we are not considering.

5.5 Full DIRSAC Algorithm

We now present the full DIRSAC algorithm in Figure 10. We first acquire scan data (line 1) and compute each point's Jacobian as in Equation 18. For each iteration of DIRSAC, we select a random point and compute the mutual information with the remaining points (line 6) to identify redundancy. We randomly select the next point from the non-redundant points (line 8). This continues until enough points have been selected to constrain the fit, meaning that the constraint score is above a very low threshold (0.01).

```
DIRSAC Algorithm
1:  Acquire 3D point cloud scan data
2:  Compute $\overline{J_\iota}$ as in Equation 18
3:  for the number of DIRSAC iterations do
4:      Randomly choose an initial point
5:      repeat
6:          Compute the mutual information for all remaining points with selected points
7:          Remove all points where the mutual information is greater than a threshold
8:          Randomly select next point
9:      until constraining set has been selected
10:     Compute ICP using the selected points
11:     Compute the residual error for this fit
12:     if the residual error has improved then
13:         Recompute the required number of iterations based on Equation 1
14:     end if
15: end for
16: Choose the best solution based on the residual error and inlier set
17: Compute refinement ICP using all inliers
18: The transform which best aligns the inlier set is the pose of the sensor
```

Figure 10: DIRected Sample And Consensus (DIRSAC) removes redundant data during the random selection process to bias point selection to better constraining points resulting in individual trial fits that provide good constrained solutions.

Using this set of selected points, we compute ICP, iteratively associating the points with M. As in RANSAC, we compute the residual error and identify the inliers. If the residual error improves, we re-compute the number of required trials based on Equation 1. After all iterations have completed, we compute a final refinement based on the best set of inliers (line 17) which is our final solution.

Table 1: The singular values of $\boldsymbol{J}^T\boldsymbol{J}$ for four synthetic models. The right column shows the number of free dimensions for each model. All but the last three models yield a constraint score of ∞ (fully unconstrained).

Model Name	λ_5	λ_4	λ_3	λ_2	λ_1	λ_0	Free Dims
Flat Slab	47542	36052	26718	0	0	0	3
Cylinder	313982	69487	54745	46278	0	0	2
Prism	166274	55136	47426	32536	4958	0	1
Orthogonal Prism	133891	116422	49597	41258	6966	4992	0

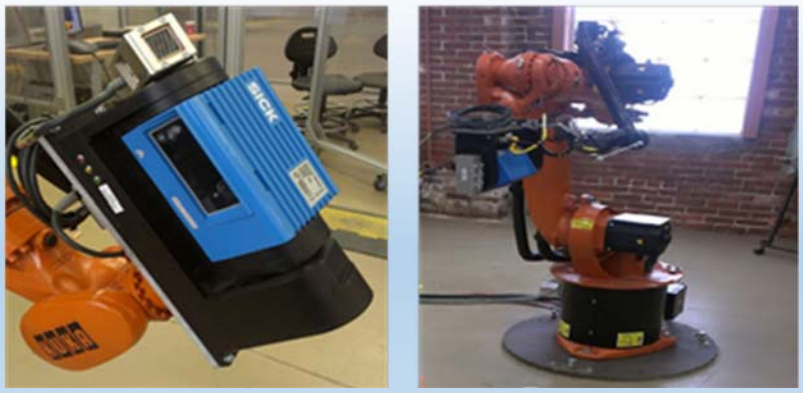

(a) (b)

Figure 11: The data collection system. A SICK LMS400 laser scanner is mounted in a nodder frame (a). The full nodder assembly is mounted on the end effector of a KR16 (b).

6 System Configuration and Experiments Performed

The approach described in this paper was developed as part of a Just-In-Time (JIT) localization system for an end effector mounted on the end of a robot arm similar to, but larger than, that shown in Figure 11, whose base was mounted to a large mobile base unit. The localization of the mobile base was accurate enough for mobility and navigation purposes and used a completely different approach. The initial starting estimate of this localization solution was made available to the JIT localization algorithm described in this paper. While good enough for general mobility, the accuracy of the final pose of the mobile base was not sufficient for the precision work required by the end effector.

However, once the mobile base unit was moved into position and its pose was accurately determined, several hours of work could be performed from that location using the highly precise positioning of the end effector. This made it economical for the algorithm described in this paper to take several seconds (or even minutes) of computation time to determine an accurate pose of the robot with respect to the workpiece. Thus, by moving the arm into several vantage points and combining the scans from multiple arm poses, we could perform scan matching to an *a priori* model of the workpiece to determine a highly accurate pose of the base of the arm in the world. Once determined, the base of the arm did not move throughout the work performed by the arm in that position.

In order to test this setup, the smaller rigidly mounted arm displayed in Figure 11 was used for testing. This section further describes the sensor data acquisition system and the different models used for experimental analysis with this application in mind.

6.1 Sensor System

Our application uses a SICK LMS400[1] mounted in a nodding cradle as shown in Figure 11(a). The SICK captures data using a spinning laser with a 70° horizontal field of view. The tilt range determines the vertical field of view, and the tilt is controlled by a very accurate harmonic drive. For our experiments the vertical field of view was set to 100°. For each line scan, the data is collected by an FPGA and time-tagged. Each line end is encoded with the tilt angle which is used to interpolate the precise angle for each range measurement. The minimum range of the SICK is $0.7m$ and the maximum range is $3.0m$.

[1] For more information on the LMS400 and other SICK sensors, visit sick.com/us/en-us/home/

The standard deviation of the range sensor error is approximately $3mm$. We convert line scans output by the FPGA to a point cloud using the interpolated angles of the range measurements.

The sensor is mounted at the end effector to capture test data as shown in Figure 11(b). We position the arm as desired and capture a single scan of data. This arm state provides a sensor pose which serves as ground truth for the pose recovery. During the analysis, position and orientation errors are added to the sensor pose and we determine the quality of the approach based on the recovery of these added errors.

6.2 Models Used in Analysis

We have performed analysis and experiments on many models with varying ranges of constraints. We present 8 models here and show results on representative examples. These models were created from scan data loaded into a 3D modeling software and surface elements were added by hand to represent the actual structure. Modeling errors may exist for many of the scenes as some extraneous surrounding structure may not have been modeled, even if it was scanned. Images of the scanning setup can be seen in Figure 11.

We report the outlier ratio for each of the models in Table 2 and consider this characteristic of the model/scan setup. The outliers were identified by performing the suggested scans and comparing the resulting scan to the model using the ground truth pose. Points with errors larger than 2σ of the expected sensor standard deviation were counted as outlier points to provide a characterization of the data.

We group the models into three groups based on their constraint score. “Non-degenerate” models are well constrained in all dimensions and have a constraint score above 4.5. “Quasi-degenerate” models are poorly constrained in some dimensions and have a constraint score between 0.5 and 4.5. Models with constraint scores below 0.5 are degenerate or almost degenerate, and we call these “extremely-degenerate”. The constraint score of each model is displayed in parenthesis in the figure’s caption as well as in the last column of Table 2. The models used are displayed in Figure 12.

7 Results

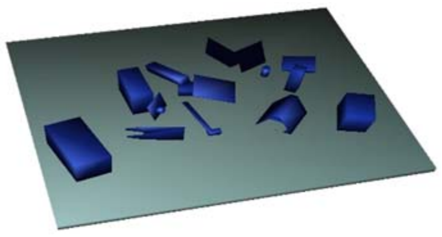
This section compares the performance of DIRSAC to RANSAC using two different methods. In Section 7.1, we measure the probability that each algorithm will choose a “good” subset of points, meaning that the selection will lead to an accurate solution. In Section 7.2 we measure the performance of the full algorithms, in terms of the solution accuracy, number of iterations, and running time.

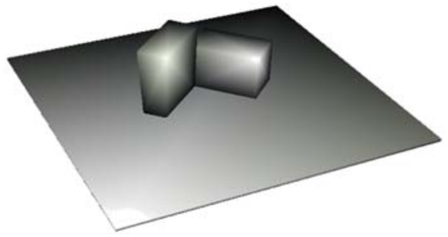
7.1 Good Solution Probability

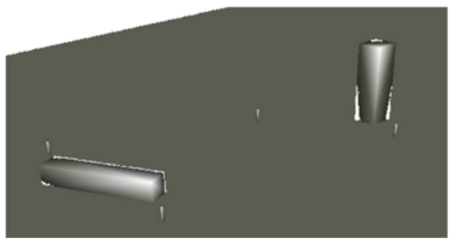
In this experiment, each algorithm (RANSAC and DIRSAC) chose a sample of 6 points. Then the ICP algorithm was used to compute the point correspondences to the model, and the pose of the model. The final pose error indicates the “quality” of the sample points chosen; *i.e.*, a sample with non-degenerate inliers will likely lead to a result with low pose error. In this experiment, the initial pose estimate for the ICP algorithm differed from the ground truth pose by 1.6 degrees in orientation and 6.7 cm in translation. These values correspond to the measured uncertainty of our robot’s positioning system. Both DIRSAC and RANSAC started from the same initial guesses.

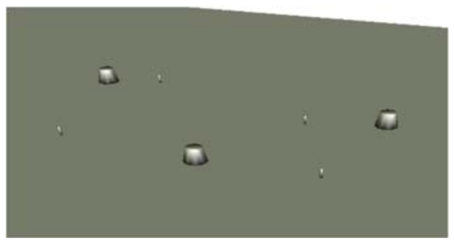
Table 3 shows the probability of the algorithm to reduce the input error by 80% and 95%, for both RANSAC and DIRSAC. We compute this by looking at 1000 trials of the algorithm and counting how many solutions reduce the input errors by 80% and by 95%. The values in the table are given in percentages. The application described in this paper could tolerate precision on the order of 0.5 degrees and around 1cm. The method used to determine the pose of the base could be in error by as much as 1.5 degrees and 6.5cm. Thus, reducing the error by 80% would provide us with pose accuracy of 0.3 degrees and 1.3cm. An even better reduction of 95% would provide us with 0.075 degrees and 0.325cm, well within our expected operating range.

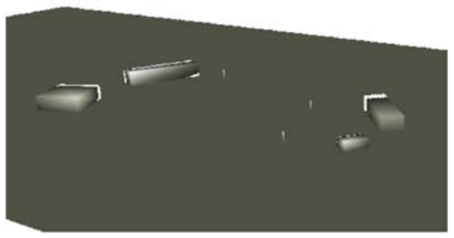
The DIRSAC algorithm shows considerable improvement over RANSAC. Especially noteworthy are the quasi-degenerate models where the probability of RANSAC to compute a solution which reduces the position error by at least 80% is only about 0.3%, whereas for DIRSAC the probability is about

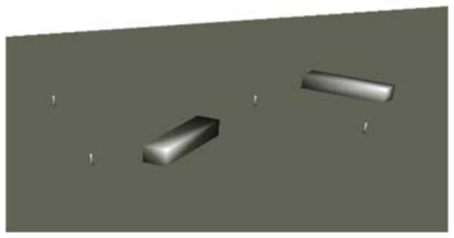

(a) “Clutter” (13.59)

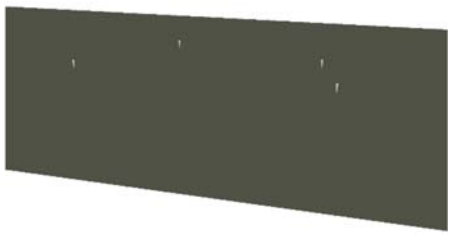

(b) “Inside Box” (4.76)


(c) “Darts with Block Standing” (4.12)


(d) “Ground Cups with Darts” (2.01)


(e) “Darts with 4 Blocks” (1.60)


(f) “Darts with Blocks” (0.40)


(g) “Darts” (0.005)

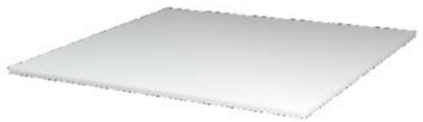

(h) “Flat Slab” (0.00)

Figure 12: Models used in the validation experiments. The constraint score is shown in parenthesis.

24.4% on average for the quasi-degenerate models. Even in the "Darts" model, which we classify as extremely-degenerate, DIRSAC has an 8.8% chance to reduce the position error and a 3.6% chance to reduce the orientation error by 80%, while RANSAC has a 0% chance for position or orientation.

Table 2: Models used with Outlier Percentages and Constraint Scores.

Model Name	Total Points	Outlier Percentage	Constraint Score
Non-Degenerate Models			
Clutter	26,356	22.0%	13.59
Inside Box	16,854	22.5%	4.76
Quasi-Degenerate Models			
Darts With Blocks Standing	27,761	15.5%	4.12
Ground Cups With Darts	27,796	15.0%	2.01
Darts With 4 Blocks	27,723	15.1%	1.60
Darts With Blocks	27,742	15.3%	0.40
Extremely-Degenerate Models			
Darts	27,830	15.3%	0.005
Flat Slab	27,836	17.7%	0.00

7.2 Full Algorithm Comparison DIRSAC vs. RANSAC

This section demonstrates how the complete DIRSAC algorithm performs compared to RANSAC. For this part of the analysis we use the same termination criteria for DIRSAC and RANSAC as suggested by Equation 1, which sets the number of iterations according to the estimated inlier ratio.

The true inlier ratio is typically unknown in advance. We use a common approach ([10], [12], and others) and update the estimated inlier ratio dynamically; ***i.e.***, as a solution is found that is the best so far, the observed inlier ratio is taken to be the true inlier ratio. Then the required number of iterations is updated using Equation 1. For these experiments we set the desired probability of achieving a correct solution to 99.5%.

For each of 1000 runs of the algorithm, we select an initial position and orientation error from a normal distribution, $N(0, 0.15m)$ and $N(0,1.5°)$, for each of the 6 pose elements. These values were chosen because they are much greater than the expected *a priori* pose errors of our actual system.

The plots in Figure 13, Figure 14, Figure 15, and Figure 16 show position error on top and orientation error on the bottom. The first column shows raw solution errors along with starting errors for reference. The trials are sorted in order of increasing initial error. The second column shows the difference in the errors and the average difference. The difference is computed as the final DIRSAC error minus the final RANSAC error. The results are displayed as a running average window of 10 solutions. The final column shows each algorithm's number of iterations and runtimes. The constraint scores are also provided in parenthesis in the figure's caption for reference.

Table 3: Probability of algorithm to compute a solution that reduces the error by 80% and 95%. The values are given in percentages.

Model Name	Position				Orientation			
	DIRSAC		RANSAC		DIRSAC		RANSAC	
Error Reduction	80%	95%	80%	95%	80%	95%	80%	95%
Non-Degenerate Models								
Clutter	25.2	2.4	6.8	0.8	7.9	0.3	4.2	0.2
Inside Box	19.2	4.8	3.0	0.3	5.9	0.1	2.3	0.0
Quasi-Degenerate Models								
Darts With 4 Blocks	21.3	2.3	0.3	0.0	16.3	0.7	0.9	0.0
Darts With Blocks Standing	44.9	6.1	0.0	0.0	35.3	2.1	0.2	0.1
Ground Cups With Darts	22.2	3.0	0.2	0.0	15.6	1.7	0.2	0.0
Darts With Blocks	9.2	1.1	0.2	0.0	23.8	1.7	1.3	0.2
Extremely-Degenerate Models								
Darts	8.8	0.2	0.0	0.0	3.6	0.0	0.0	0.0
Flat Slab	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

The results for the non-degenerate "Clutter" and "Inside Box" models are shown in Figure 13(a) and Figure 13(b). RANSAC estimates the position solution and reduces the error consistently, but not as well as DIRSAC by about $30mm$. The orientation errors are also not as accurate for RANSAC. The DIRSAC error improvement is much better. It is notable that DIRSAC requires more iterations to compute a solution, but finds the solution within a similar amount of time. This is because the directed solutions find better alignment estimates for ICP while RANSAC finds alignment estimates that are low enough to converge, but require more time to process.

The quasi-degenerate models produce very similar results (Figure 14(a), Figure 14(b), Figure 15(a), and Figure 15(b)). We see that DIRSAC finds solutions with much better position accuracy than RANSAC. This is likely because with these models, the $\boldsymbol{z}$-error is easy to resolve and produce low residual errors with high inlier ratios, but the errors in the other dimensions are more difficult to resolve. DIRSAC reduces the position error considerably better for all models (by about 100mm on average), although the orientation error is about the same. Note that DIRSAC takes more iterations (and more running time) to find solutions. This is because RANSAC terminates prematurely, finding a solution quickly that has low residual error, but is incorrect.

For the extremely-degenerate “Darts” model (Figure 16), the results indicate that neither RANSAC nor DIRSAC is able to find good solutions. This is to be expected, because the vast majority of the model points are on a plane, and there are very few model points off the plane that constrain the solution. Although DIRSAC can identify the valid constraining points, it cannot distinguish them from outlier points (*i.e.*, points with measurement errors), since both types of points appear to have low mutual information with the planar points. The number of outlier points is much larger than the number of non-outlier valid points, and so DIRSAC is much more likely to choose sample sets containing the outlier points. In fact, while both algorithms compute nearly the same number of iterations in this case, DIRSAC does take much more time.

8 Discussion

We have presented a novel approach to find the pose of a range sensor relative to an *a priori* model in the presence of quasi-degenerate data. By using mutual information between point measurements, we are able to determine the constraints a set of points will impose on the pose solution. Using this, we can direct the point selection process in a sample and consensus framework. On quasi-degenerate data, the algorithm finds significantly more accurate solutions than a naïve RANSAC approach, for a given number of iterations. The total running time of the DIRSAC algorithm is greater than that of RANSAC; however in applications where it is important to find the correct solution, using RANSAC would not be acceptable (even though it is faster). In the robotics application described in this paper, the localization process was performed only occasionally; thus the running time was acceptable. In the case of non-degenerate data, the performance of the algorithm is about the same as the naïve RANSAC approach. In the case of extremely-degenerate data, neither RANSAC nor DIRSAC is likely to find a correct solution. For these scenes a completely different approach may be needed. We also showed how to quantify the degeneracy of a model.

Another advantage of our approach is that it does not assume that the sampling of the surface is uniform. For example, some portions of the model could have higher density samplings. Such higher-density regions would cause problems for RANSAC. The higher-density regions would be similar to redundant data and would bias a solution toward those regions in a random sampling context. Our approach would handle the non-uniform sampling quite elegantly. DIRSAC would naturally identify and remove the redundancy of the higher-density regions, which would result in a reduction of the bias to those regions.

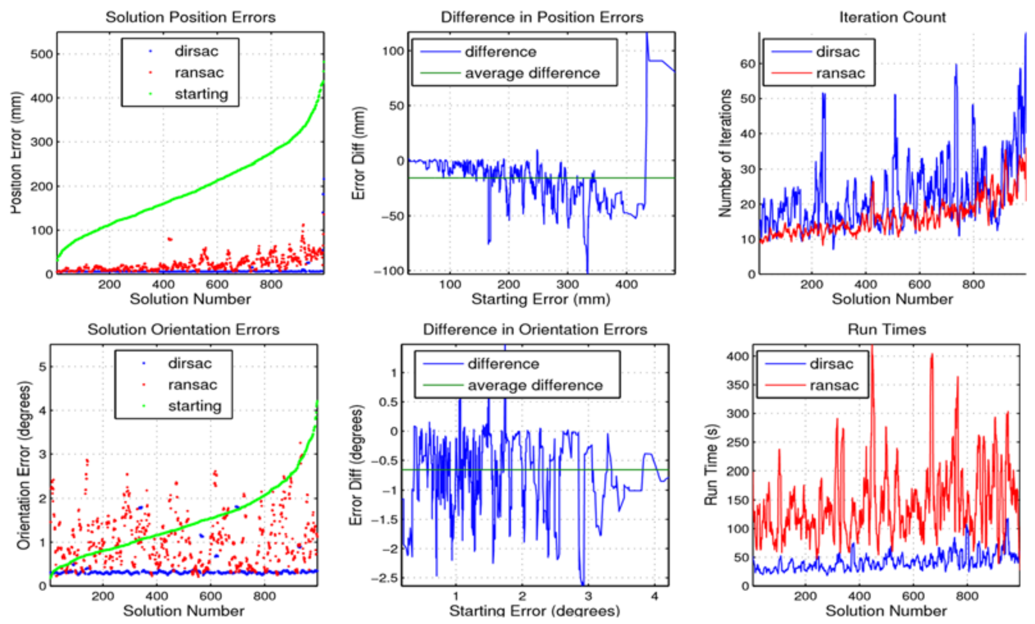
The following are suggestions for future work. In this paper we assumed a range-only measurement error which is applicable in many contexts, but this is not always the case. A possible extension would be the development of more extensive noise models and their incorporation into the algorithm. The only required modifications are the point-to-model association and the development of the Jacobian describing the constraint a point has on the sensor.

The current approach uses only range measurements. If image features are also available, this information could be incorporated into the mutual information calculation, to help guide the point selection process. This would be useful in cases where the model is highly degenerate.

Another idea is to iteratively refine the sample set by using the previously found inliers. This approach was used in [21] for fitting a fundamental matrix. As outliers are discarded during the iterative process, the mutual information could become more discriminative.

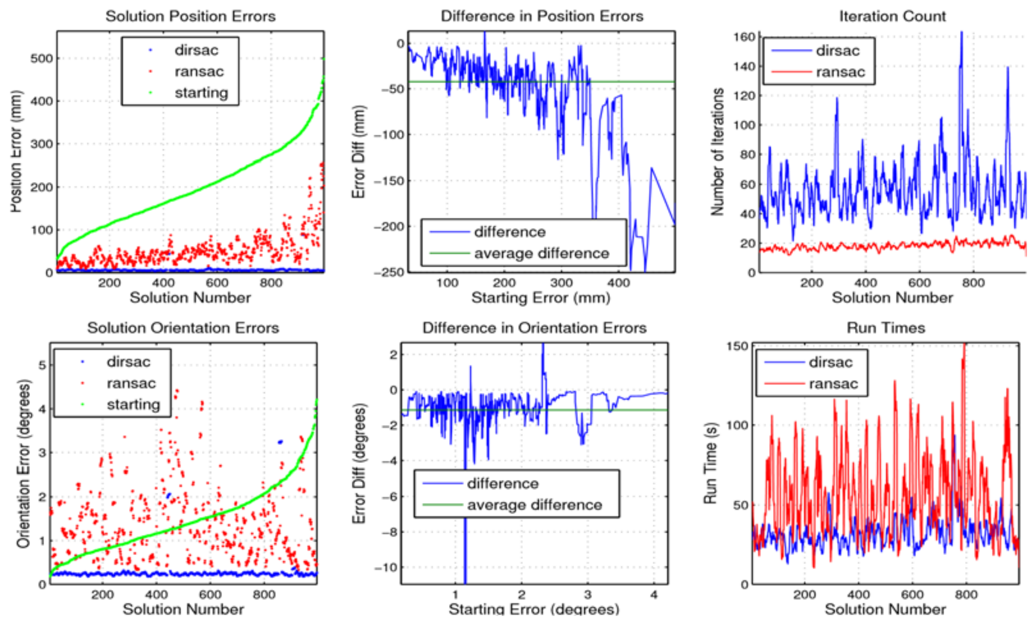
Finally, the approach presented in this paper could be useful in other applications. For example, in model building applications, where the task is to register a new scan to an existing scan, the approach could guide the selection of measurements that will allow for the most accurate registration.

“Clutter” Model Starting Error vs. Final Error



(a) “Clutter” (13.59)

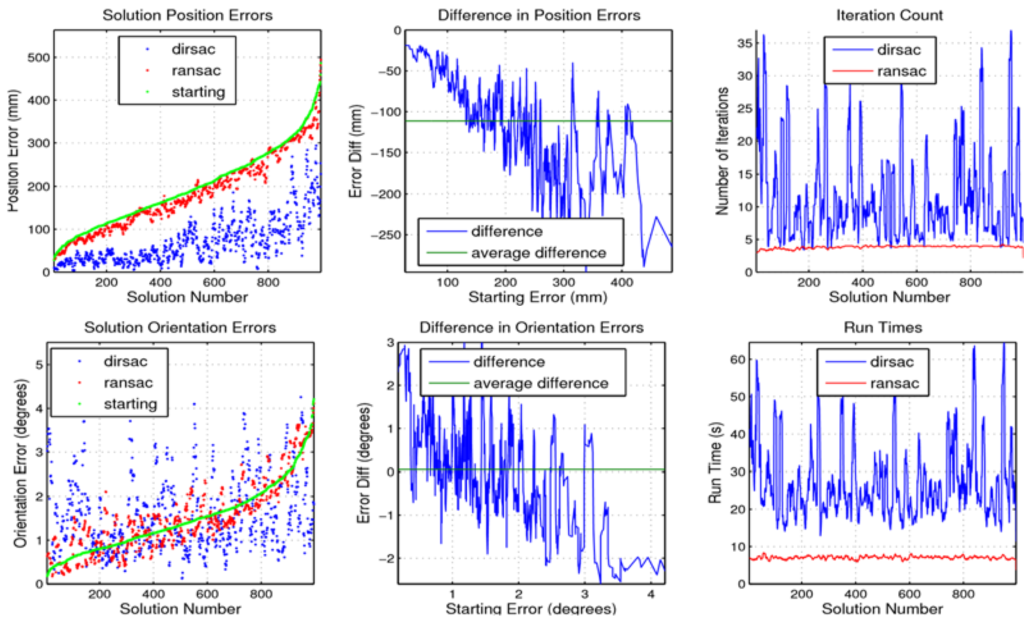
“Inside Box” Model Starting Error vs. Final Error



(b) “Inside Box” (4.32)

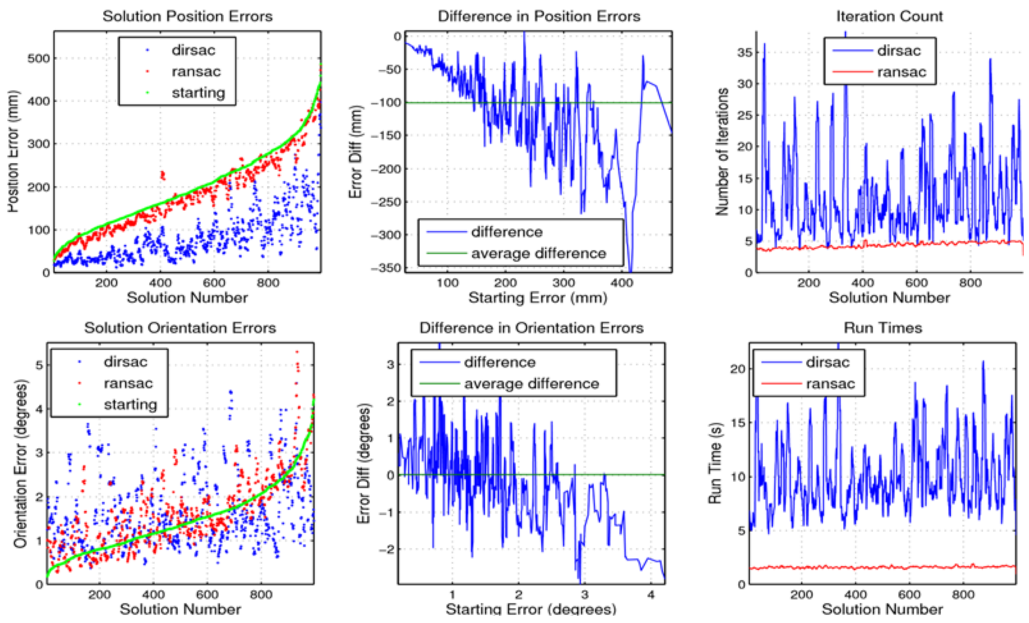
Figure 13: Convergence results for “Clutter” and ”Inside Box” models.

“Darts with Blocks Standing” Model Starting Error *vs.* Final Error



(a) “Darts with Blocks Standing” (1.79)

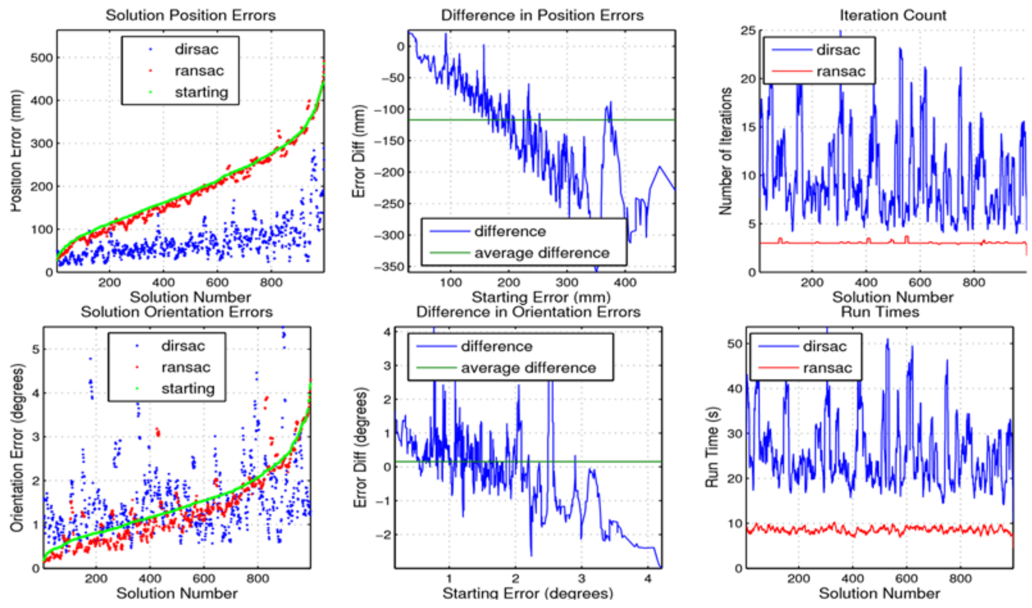
“Darts with 4 Blocks” Model Starting Error *vs.* Final Error



(b) “Darts with 4 Blocks” (1.93)

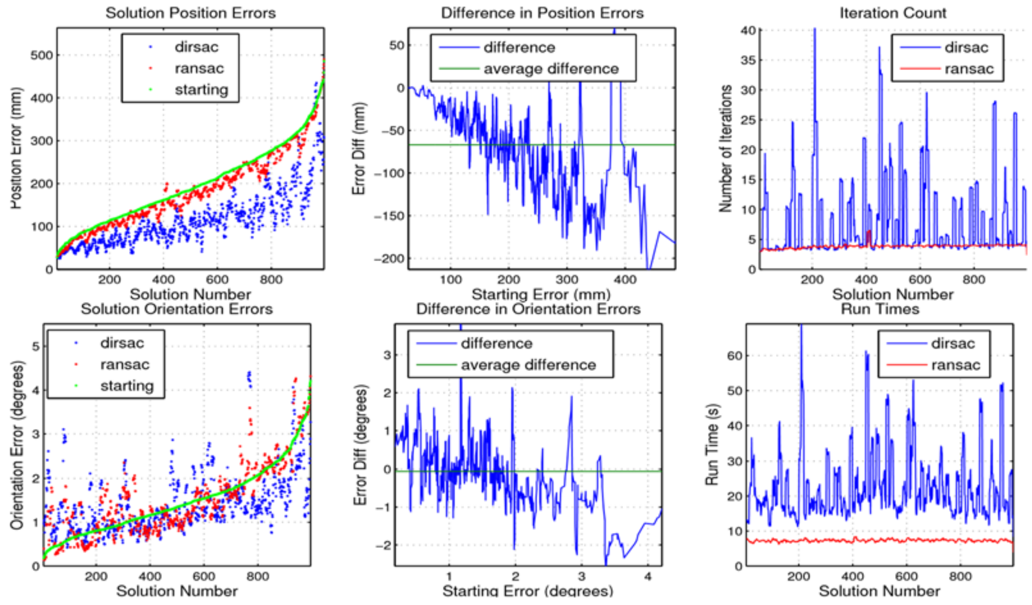
Figure 14: Convergence results for the “Darts with Blocks Standing” and “Darts with 4 Blocks” models.

"Ground Cups with Darts" Model Starting Error *vs.* Final Error



(a) "Ground Cups with Darts" (1.18)

"Darts with Blocks" Model Starting Error *vs.* Final Error



(b) "Darts with Blocks" (1.18)

Figure 15: Convergence results for the "Darts with Blocks" and "Ground Cups with Darts" (1.18) models.

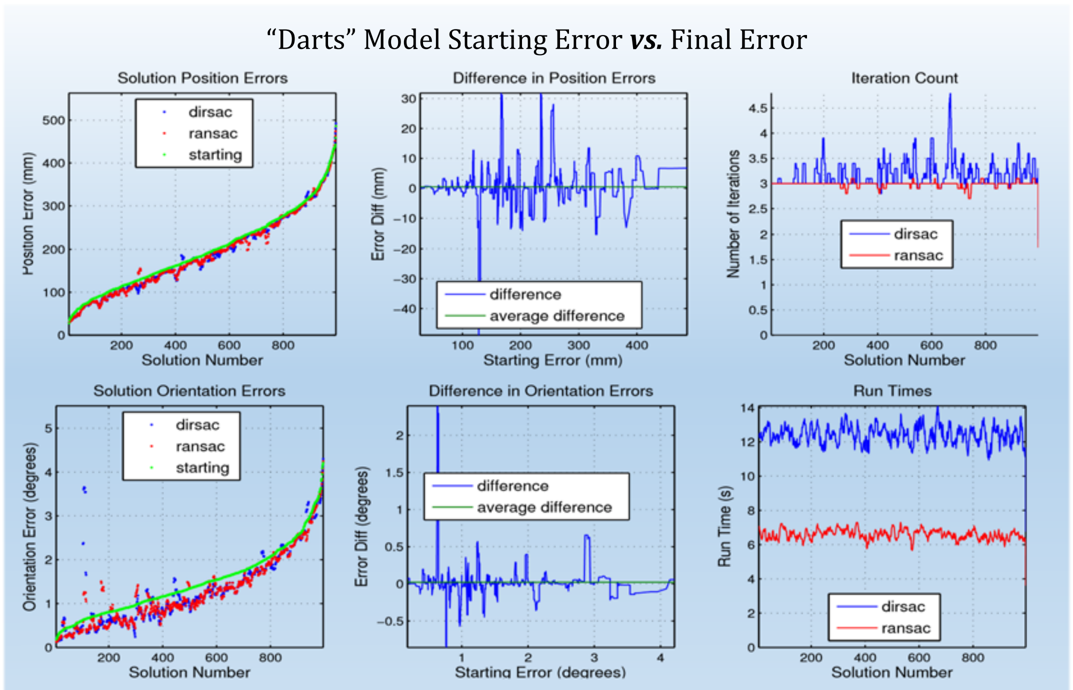


Figure 16: Convergence results for the “Darts” (0.005) model on real data.

9 Acknowledgements

This research was supported through Concurrent Technologies Corporation under subcontract NO. 120700097 & 120700098 issued under FEDERAL CONTRACT NO. W91ZLK-10-D-0005 Task 0793 issued to Carnegie Mellon University. The views, opinions, and/or findings contained in this paper are those of the authors and should not be construed as an official Department of the Air Force/Army position, policy, or decision unless so designated by other official documentation.

10 References

[1] P. Besl and H. McKay, "A Method for Registration of 3-D Shapes," *IEEE Trans on Pattern Analysis and Machine Intelligence,* vol. 14, no. 2, pp. 239-256, Feb 1992.

[2] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Graphics and Image Processing,* pp. 381-395, 1981.

[3] C. L. Baker and W. Hoff, "DIRSAC: A Directed Sampling and Consensus Approach to Quasi-Degenerate Data Fitting," in *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*, Jan..

[4] C.-S. Chen, Y.-P. Hung and J.-B. Cheng, "RANSAC-based DARCES: A New Approach to Fast Automatic Registration of Partially Overlapping Range Images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* , vol. 21, no. 11, pp. 1229-1234, 1999.

[5] A. Johnson and M. Hebert, "Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 21, no. 5, pp. 433-449, 2004.

[6] B. He, Z. Lin and Y. Li, "An Automatic Registration Algorithm for the Scattered Point Clouds Based on the Curvature Feature," *Optics & Laser Technology,* vol. 46, no. 0, pp. 53-60, 2013.

[7] J. Civera, O. G. Grasa, A. J. Davison and J. M. M. Montiel, "1-Point RANSAC for Extended Kalman Filtering: Application to Real-Time Structure from Motion and Visual Odometry," *Journal of Field Robotics,* vol. 27, no. 5, pp. 609-631, 2010.

[8] A. Hidalgo-Paniagua, M. A. Vega-Rodriguez, N. Pavon and J. Ferruz, "A Comparative Study of Software Filters Applied as a Previous Step of the ICP Algorithm in Robot Location," *Journal of Circuits, Systems, and Computers,* vol. 23, no. 08, p. 1450118, 2014.

[9] S. Choi, T. Kim and W. Yu, "Performance Evaluation of RANSAC Family," *Journal of Computer Vision,* vol. 24, no. 3, pp. 271-300, 1997.

[10] R. Raguram, J.-M. Frahm and M. Pollefeys, "A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus," in *Proceedings of the 10th European Conf on Computer Vision (ECCV): Part II*, Berlin, Heidelberg, 2008.

[11] J. Matas and O. Chum, "Randomized RANSAC with Td,d Test," *Image Vision Computing,* vol. 22, no. 10, pp. 837-842, 2004.

[12] D. P. Capel, "An Effective Bail-out Test for RANSAC Consensus Scoring.," in *BMVC*, 2005.

[13] O. Chum and J. Matas, "Matching with PROSAC - Progressive Sample Consensus," in *IEEE Computer Society Conf on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[14] K. Ni, H. Jin and F. Dellaert, "Groupsac: Efficient Consensus in the Presence of Groupings," in *Computer Vision, IEEE 12th International Conference on*, 2009.

[15] K.-R. Lee and T. Nguyen, "Robust Tracking and Mapping with a Handheld RGB-D Camera," in *Applications of Computer Vision (WACV), 2014 IEEE Workshop on*.

[16] R. Hartley and A. Zisserman, Multiple View Geometry, P. by the Press Syndicate of the University of Cambridge, Ed., Cambridge University Press, 2001.

[17] J.-M. Frahm and M. Pollefeys, "RANSAC for (Quasi-)Degenerate Data (QDEGSAC)," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition*, 2006.

[18] A. J. Davison, "Active Search for Real-Time Vision," in *Tenth IEEE Int'l Conf on Computer Vision (ICCV)*, 2005.

[19] J. J. Craig, Introduction to Robotics: Mechanics and Control (2nd Edition), Prentice Hall, 1989.

[20] H. Stark and J. Woods, Probability and Random Processes with Applications to Signal Processing, Prentice Hall, 2002.

[21] L. Moisan and B. Stival, "A Probabilistic Criterion to Detect Rigid Point Matches Between Two Images and Estimate the Fundamental Matrix," *International Journal of Computer Vision,* vol. 57, no. 3, pp. 201-218, 2004.