2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

A Direct Semi-Exhaustive Search Method for Robust, Partial-to-Full Point Cloud Registration

Richard Cheng, Chavdar Papozov, Dan Helmick, Mark Tjersland

***Abstract*— Point cloud registration refers to the problem of finding the rigid transformation that aligns two given point clouds, and is crucial for many applications in robotics and computer vision. The main insight of this paper is that we can directly optimize the point cloud registration problem without correspondences by utilizing an algorithmically simple, yet computationally complex, semi-exhaustive search approach that is very well-suited for parallelization on modern GPUs. Our proposed algorithm, Direct Semi-Exhaustive Search (DSES), iterates over potential rotation matrices and efficiently computes the inlier-maximizing translation associated with each rotation. It then computes the optimal rigid transformation based on any desired distance metric by directly computing the error associated with each transformation candidate $\{R, t\}$. By leveraging the parallelism of modern GPUs, DSES outperforms state-of-the-art methods for partial-to-full point cloud registration on the simulated ModelNet40 benchmark and demonstrates high performance and robustness for pose estimation on a real-world robotics problem (`https://youtu.be/q0q2-s2KSuA`).**

I. Introduction

Point cloud registration is the problem of computing the rigid transformation that aligns two given point clouds with unknown point correspondences. It is crucial in many robotic applications, including localization and pose estimation. Given two sets of point clouds, the problem is to find the rigid transformation, $\{R, t\} \in SE(3)$, that optimally aligns those point clouds. Because of the widespread importance of this problem, many solutions have been proposed over the past decades.

Most methods break this problem into two stages (often solved iteratively): (1) finding point correspondences, and (2) computing the optimal transformation given the correspondences. By far the most popular approach is Iterative Closest Point (ICP) [1]. Unfortunately, finding correspondences is extremely difficult without very good initialization, and a few outliers and/or poor correspondences can lead to large errors. Recent learning-based methods have attempted to use data to improve this correspondence matching and/or learn the entire registration pipeline, but this also introduces issues related to generalization.

The conventional wisdom has been that directly solving the point cloud registration optimization problem (see Problem (1)) is intractable. However, with the advent of powerful GPUs, we show that intelligent application of a parallelized, semi-exhaustive search method can be used to solve the point cloud registration optimization problem efficiently on a GPU. Our proposed direct semi-exhaustive search method (DSES) gives us flexibility to minimize over any norm, enabling greater robustness to outliers and partial overlap [2]. Furthermore, because the method is not data-driven, we get generalization by design.

All authors are with the Toyota Research Institute

Note that in this paper, we specifically focus on the partial-to-full point cloud registration setting (i.e. where we are trying to match a partial, observed pointcloud to a full object pointcloud derived from some known model). This is because in the partial-to-full (and full-to-full) setting, point cloud registration can be seen purely as an optimization (see Problem (1)), excepting cases of symmetry. However, in the partial-to-partial setting, there are many instances where the optimal solution to this problem provides poor point cloud registration, which necessitates a correspondence-based registration approach (as opposed to a direct optimization approach). Nevertheless, the partial-to-full point cloud registration problem is common in several robotics applications (e.g. object pose estimation), as seen in Section V-B.

The contributions of this work are as follows:

- Introduce a highly parallelizable algorithm, Direct Semi-Exhaustive Search (DSES), for robust, partial-to-full point cloud registration,
- Prove the algorithm's optimality in terms of inlier maximization between point clouds, suggesting its effectiveness both theoretically and practically,
- Demonstrate high performance and robustness of DSES for partial-to-full point cloud registration leveraging GPUs, outperforming other methods in both simulated and real-world environments.

II. Related Work

A. Classical Registration Methods

ICP is by far the most popular algorithm for solving rigid registration problems, and involves alternatively (1) finding point cloud correspondences and (2) solving a least-squares problem to compute the alignment [1]. While this method is computationally efficient, the problem becomes more difficult in the partial-to-full setting or when there are significant outliers (breaking the one-to-one point correspondence assumption). Several ICP variants have been proposed to deal with these issues, for example by introducing point-to-plane correspondences [3], setting nearest-neighbor distance thresholds [4], [5], introducing different objective functions [6], and using probabilistic matching [7], [8]. This class of solutions is very well studied, and [4], [9] provide reviews of ICP and its variants.

While such ICP methods are widely used with many open source implementations, they require significant parameter/threshold tuning and may converge to poor local

979-8-3503-7769-9/24/$31.00 ©2024 IEEE

minima without good initialization. Therefore, several methods have been proposed to tackle global alignment. GO-ICP uses a branch-and-bound method to compute the globally optimal point cloud alignment [10]. Other methods have been proposed to identify the globally optimal solution through convex relaxation [11] or mixed-integer programming [12].

However, even globally optimal solutions may yield poor point cloud registration in common real-world settings where there is significant noise and/or only partial alignment. One prominent approach is to use robust functions to reduce the importance of outliers [13], [14]. More recently, it's been show that this issue can be alleviated by adopting error metrics that promote sparsity of point-wise distance between point clouds (e.g. L_p-norm with $p \in (0, 1)$) [2], [15]. However, using these more robust error metrics significantly increase the computational cost, as minimization of the l_2-norm enables a closed form solution, which is not the case for $p \in (0, 1)$. Other methods have proposed different metrics that can achieve a similar sparsity with faster computation [16].

B. Learning-based Registration Methods

Recently, several works have looked at incorporating deep learning into the pipeline for point cloud registration. Many of these methods aim to learn/extract point correspondences between point clouds, such that a robust estimator (e.g. RANSAC) can be used to compute alignment [17]–[21]. These correspondences are typically based on keypoint detection with descriptive features [22]–[24]. The challenge lies in the need for repeatable keypoints with highly descriptive features. Therefore, other methods extract correspondences without keypoint detection using "superpoints" or hierarchical features [25]–[27].

Other works have proposed end-to-end learning to estimate the rigid transformation with a neural network. One class of solutions adopts the same framework as ICP, iteratively establishing soft correspondences and computing the transformation with differentiable weighted SVD [28]–[31]. Another class of end-to-end methods aim to extract a global feature vector for each point cloud and regresses the transformation with the global feature vectors [32]–[35]. Recently, learning-based graph matching has also been proposed to improve point cloud registration [31], [36].

While these learning-based methods have garnered significant interest, their performance suffers when applied to conditions that are not well-represented in the training set; in such conditions non-learned methods have better generalization ability [37]. In this paper, we suggest that enhanced GPU compute, rather than data-driven learning, may provide an effective and more generalizable avenue to improving point cloud registration.

III. Problem Setup

Consider we are given a source point cloud $\mathbf{X} = \{x_i \in \mathbb{R}^3 \mid i = 1, ..., N\}$ and a reference point cloud $\mathbf{Y} = \{y_j \in \mathbb{R}^3 \mid j = 1, ..., M\}$. Our goal is to compute the rigid transform, $\{R, t\} \in SE(3)$ that optimally aligns the point clouds $\mathbf{X}$, $\mathbf{Y}$. This can be expressed mathematically as,

$$\underset{R \in SO(3), t \in \mathbb{R}^3}{\operatorname{argmin}} \sum_{x_i \in \mathbf{X}} \min_{y_j \in \mathbf{Y}} \|y_j - Rx_i - t\|_p \tag{1}$$

where p is any desired norm. While most works consider $p = 2$ for computational convenience and speed, researchers have shown that $p \in (0, 1]$ or use of a truncated norm provides much more robust results [2], [15]. Since we are focused on the partial-to-full point cloud registration setting, we consider that $\mathbf{Y}$ represents the full object (for which we may have some mesh model) and $\mathbf{X}$ may represent only a portion of that object (observed from robot sensors).

IV. Our Method

We first describe a correspondence-free pure exhaustive search method to solving the point cloud registration problem. While this approach struggles to scale to large problems, it will help us frame/describe our proposed DSES method in Section IV-B, which shares the same principle but achieves greater efficiency.

A. Pure Exhaustive Search

A pure exhaustive search approach to solving the optimization problem (1) would be to discretize over all six rotational/translational DOFs for $\{R, t\} \in SE(3)$, creating a 6D grid of rotations/translations. Given point clouds $\mathbf{X}$ and $\mathbf{Y}$, we could then compute the alignment error for each discrete rotation/translation, where the nearest neighbor is computed by iterating over each point-point pair between point clouds $\mathbf{X}$ and $\mathbf{Y}$.

$$\text{ERROR}(R, t) = \sum_{x_i \in \mathbf{X}} \min_{y_j \in \mathbf{Y}} \|y_j - Rx_i - t\|_p \tag{2}$$

The $\{R, t\} \in SE(3)$ that yields the minimum error is then our optimal solution. The overall algorithm is extremely simple, and outlined in Algorithm 1.

The exhaustive search approach outlined in Algorithm 1 is optimal by definition (up to our discretization resolution) as it iterates through every possible combination in our discrete grid, and allows us to easily optimize over different metrics (not just L_2). Therefore, the main reason not to adopt this approach is timing. While the algorithm is highly amenable to GPU parallelization, since we have to discretize over 6 rotational/translational dimensions, and iterate over every point pair in point clouds $\mathbf{X}$ and $\mathbf{Y}$, the computation required for this approach scales $O(K^6MN)$, where K represents the discretization for each dimension, and N, M represent the number of points in point clouds $\mathbf{X}, \mathbf{Y}$, respectively.

Remark 1: Instead of sampling rotations/translation candidates, one could sample triplets of correspondence pairs and compute the optimal closed form poses for such triplets. However, without decent correspondences this scales poorly (two point clouds with 1000 points leads to > 10 quadrillion pose candidates). Our correspondence-free approach allows us to brute force our way around correspondences by instead focusing on discretization of 6D pose space.

B. Direct Semi-Exhaustive Search (DSES)

In this subsection, we boost the efficiency of the pure exhaustive search by iterating only over rotations, $R \in SO(3)$, and efficiently computing the inlier-maximizing translation for each rotation R instead of iterating through every potential rotation/translation combination. We also ensure that this can be easily computed leveraging CUDA kernels.

Let us define the inlier-maximizing translation in terms of a modified L_0-"norm", L_0^{m1},

$$\begin{aligned} L_0^{m1}(t|x_i, y_j) &= \min(\|y_j - Rx_i - t\|_0, 1) \\ \text{INLIERS}(t) &= \sum_{x_i \in \mathbf{X}} \min_{y_j \in \mathbf{Y}} L_0^{m1}(t|x_i, y_j). \end{aligned} \tag{3}$$

Here L_0^{m1} is simply the L_0 "norm" saturated at 1. For a given rotation, we can consider the inlier-maximizing translation:

$$t^* = \underset{t \in \mathbb{R}^3}{\operatorname{argmin}} \sum_{x_i \in \mathbf{X}} \min_{y_j \in \mathbf{Y}} L_0^{m1}(t|x_i, y_j) \tag{4}$$

As described in Lemma 2 below, the L_0^{m1}-optimal (inlier-maximizing) translation t^* for problem (4) can be computed, under mild assumptions, by taking the mode of translations between every *rotated* point-point pair.

***Lemma** 2:* Suppose that no points within $\mathbf{X}$ are the same, and that no points within $\mathbf{Y}$ are the same. Then the optimal solution to (4) is

$$t^* = \text{MODE}(\{y_j - Rx_i \mid x_i \in \mathbf{X}, y_j \in \mathbf{Y}\}). \tag{5}$$

Obviously, the L_0^{m1} norm over continuous points is practically nonsensical. However, since we discretize our points, the L_0^{m1} norm should be considered over a discrete grid. Therefore, we can consider the L_0^{m1}-minimizing solution equivalently as the inlier-maximizing solution, given some discretization distance d.

Given Lemma 2, we can iterate over every orientation $R_{\theta,\phi,\xi}$ and use (5) to compute the corresponding inlier-maximizing translation $t^*_{\theta,\phi,\xi}$ (up to our discretization δ_t). In CUDA, this can be done by counting/storing discrete translation candidates from each point-pair in an array, and taking the argmax of this array. This means we only have to iterate over a 3D grid of orientations, rather than a 6D grid of orientations *and* translations. Once we have our candidate set of rigid transformations $(R_{\theta,\phi,\xi}, t^*_{\theta,\phi,\xi})$, we *sort* them by the L_0^{m1} error; then we compute the optimal rigid transform using our desired L_p-norm by brute-force computing the error associated with the best $q\%$ of candidate rigid transforms (as defined by L_0^{m1} error). This process is completely parallelizable and summarized in Algorithm 2 and illustrated in Figure 1.

***Theorem** 3:* Suppose that no points within $\mathbf{X}$ are within the discretization distance δ_t of each other, and that no points within $\mathbf{Y}$ are within the discretization distance δ_t of each other. If we choose L_0^{m1} error as our desired L_p norm, then Algorithm 2 yields the inlier-maximizing solution to problem (1) up to our chosen discretization.

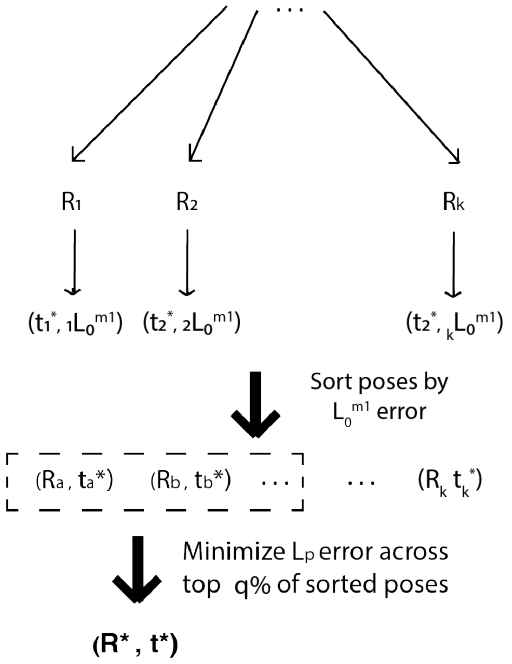


Fig. 1: Pictoral description of Algorithm 2.

One major benefit of this algorithm is that it is easy to efficiently optimize over any desired metric. Therefore, though Theorem 3 suggests the effectiveness of our method, in practice we do not aim solely for inlier-maximization (as this requires fine tuning of our discretization and is sensitive to the noise characteristics of the point clouds). Instead, we use a truncated L_1 norm which also has significiant robustness advantages over other norms (see [15]).

V. Results

All experiments in this section are run with an Intel Core i9-12900K CPU and NVIDIA A6000 GPU. Subsection V-A explores the global registration problem on a simulated dataset, and subsection V-B explores the local registration problem applied to a real-world robotics platform.

A. ModelNet40 Experiments

We conducted experiments on the ModelNet40 benchmark dataset [38], which includes 12,311 CAD models from 40 categories, and adapted the experimental conditions from [30], [31]. As done in these works, we randomly sample 2,048 points from each object, normalized into a unit sphere. To obtain random rigid transformations, we sample three Euler angle rotations in the range $[-45°, 45°]$ and translations in the range $[-0.5, 0.5]$ on each axis. We transform the source point cloud $\mathbf{X}$ using the sampled rigid transform and the task is to register it to the reference point cloud $\mathbf{Y}$.

Simulating noise and partial overlap: For both reference and observation point cloud, we randomly sample 1,024 points *independently* for the source/reference point clouds, and apply a random rigid transformation on the source point cloud. After this, we jitter the points in both point clouds by noise sampled from $\mathcal{N}(0, 0.01)$ and clipped to [-0.05,

Algorithm 1 Pure Exhaustive Search for Registration

Input: $\mathbf{X} \in \mathbb{R}^{N\times 3}$, $\mathbf{Y} \in \mathbb{R}^{M\times 3}, K \in \mathbb{Z}_+, \delta_t, \delta_R \in \mathbb{R}_+$
for $\theta = -K\delta_r; \theta \leq K\delta_r; \theta += \delta_r$ **do**
  for $\phi = -K\delta_r; \phi \leq K\delta_r; \phi += \delta_r$ **do**
    for $\xi = -K\delta_r; \xi \leq K\delta_r; \xi += \delta_r$ **do**
      for $dx = -K\delta_t; dx \leq K\delta_t; \theta += \delta_t$ **do**
        for $dy = -K\delta_t; dy \leq K\delta_t; \phi += \delta_t$ **do**
          for $dz = -K\delta_t; dz \leq K\delta_t; \xi += \delta_t$ **do**
            $e(R_{\theta,\phi,\xi}, t_{dx,dy,dz}) = 0$
            for $y_j \in \mathbf{Y}$ **do**
              min_error $= \infty$
              for $x_i \in \mathbf{X}$ **do**
                error $=$ ERROR$(y_j - R_{\theta,\phi,\xi}x_i - t_{dx,dy,dz})$
                min_error = MIN(min_error, error)
              end for
              $e(R_{\theta,\phi,\xi}, t_{dx,dy,dz})$ += min_error
            end for
          end for
        end for
      end for
    end for
  end for
end for
return $(R^{opt}, t^{opt}) = \underset{R,t}{\operatorname{argmin}}\ e(R,t)$.

0.05] on each axis. Finally, for the source point cloud $\mathbf{X}$, we sample a half-space with a random direction $\in \mathcal{S}^2$ and shift it such that 70% of the points are retained, discarding the other 30% to simulate partial-to-full overlap.

Metrics: We track mean isotropic errors (MIE) of $\mathbf{R}$ and $\mathbf{t}$ as proposed in [30], specified in units of degrees and meters, respectively. We also track mean absolute errors (MAE) of R and t used in DCP [28]. Finally, we report the recall with MAE($\mathbf{R}$)$< 1^\circ$ and MAE($\mathbf{t}$)< 0.1. The best results are marked in bold font in Tables I and II.

We compare our method to state-of-the-art point cloud registration methods RPM-Net [30], RGM [31], and Predator [24]. Other methods, such as DCP, PointNetLK, IDAM, DeepGMR, ICP, and FGR [28], [32], [39]–[41] are not directly compared, because experiments in [24], [30], [31] have already shown that these newer methods have better performance. The ModelNet40 dataset contains official train/test splits for each of the 40 object categories. The first 20 categories were used by the learned methods for training and validation, and the remaining 20 categories were used for testing of all methods.

As seen from Table I, DSES outperforms across all metrics, achieving very high recall (i.e. high reliability) and very low errors. The main instances it struggles on are categories with symmetry, where there is ambiguity with respect to some rotational degree-of-freedom.

A further advantage of DSES is its independence from data (since it is purely solving a minimization problem),

Algorithm 2 Direct Semi-Exhaustive Search (DSES)

Input: $\mathbf{X} \in \mathbb{R}^{N\times 3}$, $\mathbf{Y} \in \mathbb{R}^{M\times 3}, K \in \mathbb{Z}_+, \delta_t, \delta_R \in \mathbb{R}_+, q \in (0,1]$
// For each orientation candidate (parameterized by θ, ϕ, ξ), compute the mode t^* (and frequency M) of the translation between every point pair up to the specified discretization, δ_t.
for $\theta = -K\delta_r; \theta \leq K\delta_r; \theta += \delta_r$ **do**
  for $\phi = -K\delta_r; \phi \leq K\delta_r; \phi += \delta_r$ **do**
    for $\xi = -K\delta_r; \xi \leq K\delta_r; \xi += \delta_r$ **do**
      $t^*_{\theta,\phi,\xi} =$ MODE$(\{$ROUND$(y_j - R_{\theta,\phi,\xi}x_i\ ,\ \delta_t) \mid x_i \in \mathbf{X}, y_j \in \mathbf{Y}\})$
      $M_{\theta,\phi,\xi} =$ COUNT$_{i,j}(\{y_j - R_{\theta,\phi,\xi}x_i + t^*_{\theta,\phi,\xi} < \delta_t\})$
    end for
  end for
end for
// Sort the resulting poses $(R_{\theta,\phi,\xi}, t^*_{\theta,\phi,\xi})$ in decreasing order by frequency M, and take only the pose candidates that satisfy $M_{\theta,\phi,\xi} \geq qM^*$.
Sort pose candidates (R, t^*) by count M.
$M^* = \max_{\theta,\phi,\xi}\ M$.
while $M_{\theta,\phi,\xi} > qM^*$ **do**
  // For each pose candidate, compute the associated error by brute-force checking every nearest neighbor on the GPU.
  $e(R_{\theta,\phi,\xi}, t_{dx,dy,dz}) = 0$
  for $y_j \in \mathbf{Y}$ **do**
    min_error $= \infty$
    for $x_i \in \mathbf{X}$ **do**
      error = ERROR$(y_j - R_{\theta,\phi,\xi}x_i - t^*_{\theta,\phi,\xi})$
      min_error $=$ MIN(min_error, error)
    end for
    $e(R_{\theta,\phi,\xi}, t^*_{\theta,\phi,\xi})$ += min_error.
  end for
end while
// Return the pose that minimizes our desired error.
return $(R^{opt}, t^{opt}) = \underset{R,t}{\operatorname{argmin}}\ e(R,t)$.

making it generalize by design. To highlight this advantage, we repeated the same experiments, but rotated both the reference/source point cloud by the *same* rotation before doing point cloud registration. This effectively modifies the reference frame, but the relative rotation between reference and source point clouds was kept the same. Table II shows the results of these experiments. We see that the performance of RPM-Net, Predator, and RGM all suffer significantly, while the performance of DSES remains the same. By design, DSES is rotation/translation invariant, and can therefore generalize better to novel conditions.

Computation Time: Since our approach is based on a semi-exhaustive search, the computation time is highly dependent on the rotation/translation search range. An analysis of computation time versus search range is shown in Figure

method	MIE(R)	MIE(t)	MAE(R)	MAE(t)	Recall
RPM-Net	0.98°	0.011m	0.51°	0.005m	92.5%
Predator	1.56°	0.015m	0.83°	0.008m	82.8%
RGM	1.13°	0.011m	0.59°	0.005m	92.2%
DSES	**0.72°**	**0.007m**	**0.36°**	**0.004m**	**98.1%**

TABLE I: Point cloud registration performance on ModelNet40 experiments.

method	MIE(R)	MIE(t)	MAE(R)	MAE(t)	Recall
RPM-Net	1.33°	0.015m	0.72°	0.007m	88.6%
Predator	4.11°	0.033m	2.14°	0.017m	66.9%
RGM	2.30°	0.020m	1.19°	0.010m	85.3%
DSES	**0.72°**	**0.008m**	**0.37°**	**0.004m**	**97.2%**

TABLE II: Point cloud registration performance on ModelNet40 experiments, where reference frame is rotated.

2, and it can be seen that the algorithm is fast when we can constrain the search space by providing an initial guess. Therefore, this method is best for (1) online local point cloud registration with pose initialization, *or* (2) offline global point cloud registration without pose initialization.

B. Real-World Robot Pose Estimation

One of our motivations for developing DSES is for robust robot pose estimation. In this subsection, we utilize point cloud registration in order to correct for kinematic errors online in a mobile manipulation robot (see Figure 4). The problem is that when we command the robot to a specific joint configuration to achieve a desired gripper pose, error in the kinematics model and robot hardware (e.g. from encoder error, link flexing, etc.) create a discrepancy between the desired and actual gripper pose. To correct for this pose delta, we do point cloud registration between the expected gripper point cloud (obtained from a CAD model) and the perceived gripper point cloud. The perceived gripper point cloud is obtained from a stereo head camera running a learned stereo network to produce dense depth [42]. The computed pose delta is used to correct for this error on the robot allowing for more precise gripper positioning (see supplementary video: `https://youtu.be/q0q2-s2KSuA`). This precision is crucial for picking various objects.

We ran several experiments of the robot picking items in a real, unmodified grocery store, using point cloud registration running online to correct for kinematic errors in the gripper pose. The search range for DSES was rotations in the range [-5°, 5°] and translations in the range [-1.6, 1.6] cm. In these experiments we compare our algorithm, DSES, to the Generalized-ICP algorithm (GICP) [7] using the C++ implementation from the Point Cloud Library (PCL). As seen from Table III, our algorithm achieves smaller chamfer distance and much higher success rate, while running significantly faster. Success rate is defined as the percentage of instances where a solution is returned *and* that solution improves (decreases) the chamfer distance. By enabling more precise positioning of the arm tip pose, DSES allows us to successfully grasp items that we otherwise couldn't

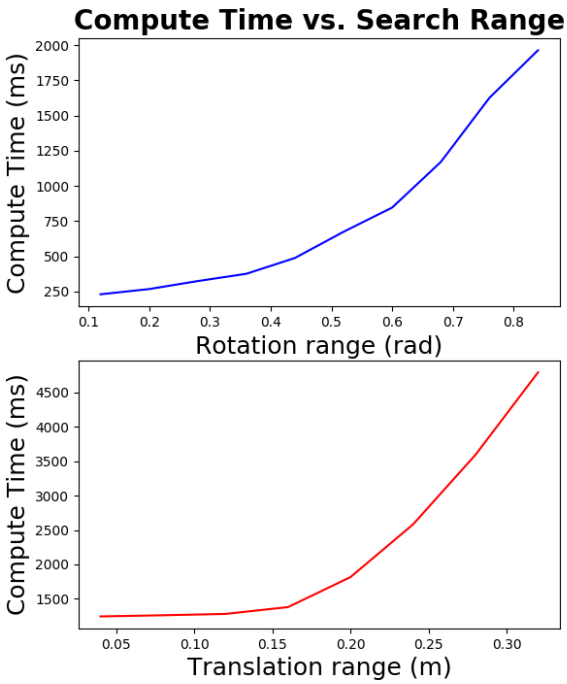


Fig. 2: (Top) Plot of computation time versus rotation range, given a translation range of ± 0.2m. (Bottom) Plot of computation time versus translation range, given a rotation range of $\pm 45°$.

method	Chamfer Distance	Success Rate	Time
None	6.32 mm	-	-
GICP	4.69 mm	68.7% (167 / 243)	235 ms
DSES	**4.36 mm**	**100%** (243 / 243)	50 ms

TABLE III: Point cloud registration performance for on-robot tool pose correction.

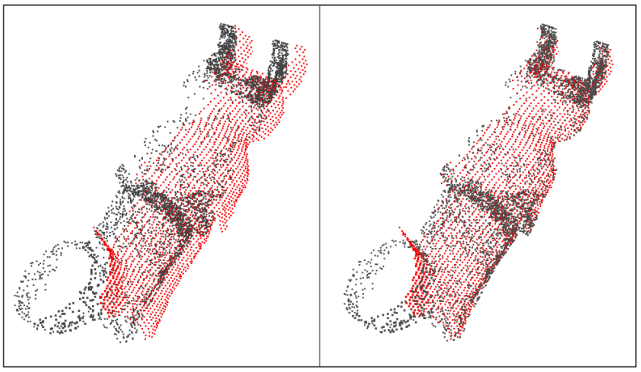
(e.g. objects with a handle or cap, where high precision is required). The supplementary video shows several instances of such grasps where DSES is crucial for successful grasping.

Note: These robot experiments test the algorithm under more realistic settings (i.e. more realistic perception noise/artifacts). However, they do not test global registration performance (as the ModelNet40 experiments do), since the problem allows for a decent initial pose based on robot forward kinematics. This initialization also enables the significantly faster computation times.

VI. Conclusion

Historically, point cloud registration has been thought of as an Expectation Maximization problem solved by an iterative process. However, the advent of extremely powerful GPUs allows us to rethink that paradigm. Our proposed DSES

Fig. 3: Image of the robot described in Section V-B picking up a jug from a grocery shelf, after using DSES to correct the gripper pose.



(a) (b)

Fig. 4: Reference arm tip point cloud (black) vs. observed arm tip point cloud (red) both (a) before pose alignment, (b) after pose alignment with DSES. While the pose delta is small, this difference impacts success of the resulting grasp.

algorithm is extremely simple, theoretically optimal, and improves upon state-of-the-art approaches for partial-to-full point cloud registration as shown in Table I. Our real-world robot experiments in Section V-B show that DSES deals well with real perception noise/artifacts and is fast for local registration problems, enabling precise robot picking.

The most important advantage of our approach is reliability (i.e. high recall). For most autonomous robotic systems, reliability takes precedence over other metrics - it is better to have an algorithm that gives approximately accurate results 100% of the time versus perfect results only 90% and poor results the other 10%. We believe that the high reliability and generalization of our approach make it ideal for many point cloud registration applications, and as GPUs become cheaper and more powerful (already the Nvidia RTX4090 boasts >80 TFLOPS), we believe its advantages will grow.

Current Limitations and Future Work

While our approach has the advantage of being able to easily trade off between reliability, speed, and problem size, it also means we must be conscious of balancing these three priorities. As seen in Figure 2, our high recall and low errors can come at the cost of higher computation time depending on the pose search range. While increasingly powerful GPUs will alleviate these issues, currently our method is most likely to provide value for (1) offline global registration, or (2) online, local registration.

As mentioned in Section III, our method optimizes Problem 1 in order to solve the point cloud registration problem. This works well for partial-to-full (and full-full) point cloud registration, but hits limitations when looking at the partial-to-partial setting. This is because the optimal solution to Problem 1 can yield poor registration in certain scenarios. We believe incorporating color into the loss function for matching can alleviate this issue for the partial-to-partial setting.

VII. Appendix

A. Proof of Lemma 2

For problem (4), consider the L_0^{m1} defined in (3). By definition, we have:

$$L_0^{m1}(t|x_i, y_j) = \begin{cases} 0 & \text{if } t = y_j - Rx_i, \\ 1 & \text{otherwise.} \end{cases} \tag{6}$$

This implies that for each $x_i \in \mathbf{X}$, all M translations

$$t_{ij} = y_j - Rx_i, \quad j = 1, \ldots, M \tag{7}$$

minimize $L_0^{m1}(t|x_i, y_j)$, where M is the number of points in the point cloud $\mathbf{Y}$. Since $L_0^{m1}(t|x_i, y_j) \in \{0, 1\}$ and no points in $\mathbf{X}$ or $\mathbf{Y}$ are repeated, the following holds

$$\min_{y_j \in \mathbf{Y}} L_0^{m1}(t|x_i, y_j) = 1 - \underset{\forall y_j \in \mathbf{Y}}{\text{COUNT}}\big(y_j - Rx_i - t\big), \tag{8}$$

where we define $\underset{\forall q}{\text{COUNT}}(p)$ as the number of times $p = 0$ across all q. Therefore,

$$\sum_{x_i \in \mathbf{X}} \min_{y_j \in \mathbf{Y}} L_0^{m1}(t|x_i, y_j) = N - \underset{\forall x_i \in \mathbf{X}, \forall y_j \in \mathbf{Y}}{\text{COUNT}}\big(y_j - Rx_i - t\big). \tag{9}$$

where N is the number of points in $\mathbf{X}$. This implies that the following optimization problems are equivalent,

$$\begin{aligned} &\underset{t \in \mathbb{R}^3}{\text{argmin}} \sum_{x_i \in \mathbf{X}} \min_{y_j \in \mathbf{Y}} L_0^{m1}(t|x_i, y_j) \\ &\qquad = \underset{t \in \mathbb{R}^3}{\text{argmax}} \underset{\forall x_i \in \mathbf{X}, \forall y_j \in \mathbf{Y}}{\text{COUNT}}\big(y_j - Rx_i - t\big). \\ &\qquad = \text{MODE}(\{y_j - Rx_i \mid x_i \in \mathbf{Y}, y_j \in \mathbf{Y}\}). \end{aligned} \tag{10}$$

where the last equality holds by definition of the mode. Therefore, we can conclude that the solution (5) optimizes problem (4).

B. Proof of Theorem 3

As seen from Lines 6-7 of Algorithm 2, for every rotation, we compute the accompanying L_0^{m1} optimal translation based on the MODE (5). Lemma 2 proves that this translation $t^*_{\theta,\phi,\xi}$ is inlier-maximizing for a given rotation, R. In addition to computing the translation, $t^*_{\theta,\phi,\xi} = \text{MODE}(\{y_j - Rx_i \mid x_i \in$

$\mathbf{Y}, y_j \in \mathbf{Y}\}$) , we also compute $M_{\theta,\phi,\xi}$. From the proof of Lemma 2, we know that the L_0^{m1}-norm can be computed as

$$\begin{aligned}\sum_{x_i \in \mathbf{X}} L_0^{m1}(t, x_i) &= N - \underset{\forall x_i \in \mathbf{X}, \forall y_j \in \mathbf{Y}}{\text{COUNT}} \left(y_j - Rx_i - t^*_{\theta,\phi,\xi}\right) \\ &= N - M_{\theta,\phi,\xi}.\end{aligned} \tag{11}$$

Therefore, the inlier-maximizing (L_0^{m1}-minimizing) solution is the one that maximizes $M_{\theta,\phi,\xi}$. In Line 12-13 of Algorithm 2, we sort through all pose candidates (R, t^*) to obtain the pose (R^{opt}, t^{opt}) that maximizes the count $M_{\theta,\phi,\xi}$, thereby maximizing the number of inliers.

Note that Lines 14-25 are unnecessary for computing the inlier-maximizing solution, but as discussed below, in practice we may desire optimization over a different metric. The post-search steps in Lines 14-25 are necessary to optimize over different norms.

REFERENCES

[1] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, 1992.

[2] S. Bouaziz, A. Tagliasacchi, and M. Pauly, "Sparse iterative closest point," vol. 32, 2013.

[3] C. Yang and G. Medioni, "Object modelling by registration of multiple range images," *Image and Vision Computing*, vol. 10, 1992.

[4] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," *Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3DIM*, 2001.

[5] K. H. Bae and D. D. Lichti, "A method for automated registration of unorganised point clouds," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 63, 2008.

[6] S. Rusinkiewicz, "A symmetric objective function for icp," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–7, 2019.

[7] A. V. Segal, D. Haehnel, and S. Thrun, "Generalized-icp," vol. 5, 2010.

[8] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Voxelized gicp for fast and accurate 3d point cloud registration," vol. 2021-May, 2021.

[9] F. Pomerleau, F. Colas, and R. Siegwart, "A review of point cloud registration algorithms for mobile robotics," *Foundations and Trends in Robotics*, vol. 4, 2015.

[10] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-icp: A globally optimal solution to 3d icp point-set registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, 2016.

[11] H. Maron, N. Dym, I. Kezurer, S. Kovalsky, and Y. Lipman, "Point registration via efficient convex relaxation," vol. 35, 2016.

[12] G. Izatt, H. Dai, and R. Tedrake, "Globally optimal object pose estimation in point clouds with mixed-integer programming," 2020.

[13] E. Trucco, A. Fusiello, and V. Roberto, "Robust motion and correspondence of noisy 3-d point sets with missing data," *Pattern Recognition Letters*, vol. 20, 1999.

[14] A. W. Fitzgibbon, "Robust registration of 2d and 3d point sets," vol. 21, 2003.

[15] H. Yang, J. Shi, and L. Carlone, "Teaser: Fast and certifiable point cloud registration," *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 314–333, 2020.

[16] J. Zhang, Y. Yao, and B. Deng, "Fast and robust iterative closest point," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, 2022.

[17] H. Deng, T. Birdal, and S. Ilic, "Ppfnet: Global context aware local features for robust 3d point matching," 2018.

[18] H. Deng, "Ppf-foldnet : Unsupervised learning of rotation invariant 3d local descriptors supplementary material additional visualizations of matching," *Eccv*, 2018.

[19] C. Choy, W. Dong, and V. Koltun, "Deep global registration," 2020.

[20] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," vol. 2019-June, 2019.

[21] A. Kurobe, Y. Sekikawa, K. Ishikawa, and H. Saito, "Corsnet: 3d point cloud registration by deep neural network," *IEEE Robotics and Automation Letters*, vol. 5, 2020.

[22] S. Ao, Q. Hu, B. Yang, A. Markham, and Y. Guo, "Spinnet: Learning a general surface descriptor for 3d point cloud registration," 2021.

[23] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C. L. Tai, "D3feat: Joint learning of dense detection and description of 3d local features," 2020.

[24] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "Predator: Registration of 3d point clouds with low overlap," 2021.

[25] H. Yu, F. Li, M. Saleh, B. Busam, and S. Ilic, "Cofinet: Reliable coarse-to-fine correspondences for robust pointcloud registration," *Advances in Neural Information Processing Systems*, vol. 34, pp. 23 872–23 884, 2021.

[26] J. Lee, S. Kim, M. Cho, and J. Park, "Deep hough voting for robust global registration," 2021.

[27] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, and K. Xu, "Geometric transformer for fast and robust point cloud registration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 143–11 152.

[28] Y. Wang and J. Solomon, "Deep closest point: Learning representations for point cloud registration," vol. 2019-October, 2019.

[29] ——, "Prnet: Self-supervised learning for partial-to-partial registration," vol. 32, 2019.

[30] Z. J. Yew and G. H. Lee, "Rpm-net: Robust point matching using learned features," 2020.

[31] K. Fu, S. Liu, X. Luo, and M. Wang, "Robust point cloud registration framework based on deep graph matching," 2021.

[32] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "Pointnetlk: Robust efficient point cloud registration using pointnet," vol. 2019-June, 2019.

[33] X. Huang, G. Mei, and J. Zhang, "Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences," 2020.

[34] R. Zhou, X. Li, and W. Jiang, "Scanet: A spatial and channel attention based network for partial-to-partial point cloud registration," *Pattern Recognition Letters*, vol. 151, 2021.

[35] H. Xu, S. Liu, G. Wang, G. Liu, and B. Zeng, "Omnet: Learning overlapping mask for partial-to-partial point cloud registration," 2021.

[36] M. Saleh, S. Dehghani, B. Busam, N. Navab, and F. Tombari, "Graphite: Graph-induced feature extraction for point cloud registration," 2020.

[37] X. Li, J. K. Pontes, and S. Lucey, "Pointnetlk revisited," 2021.

[38] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," vol. 07-12-June-2015, 2015.

[39] Q.-Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *European conference on computer vision*. Springer, 2016, pp. 766–782.

[40] J. Li, C. Zhang, Z. Xu, H. Zhou, and C. Zhang, "Iterative distance-aware similarity matrix convolution with mutual-supervised point elimination for efficient point cloud registration," vol. 12369 LNCS, 2020.

[41] W. Yuan, B. Eckart, K. Kim, V. Jampani, D. Fox, and J. Kautz, "Deepgmr: Learning latent gaussian mixture models for registration," vol. 12350 LNCS, 2020.

[42] K. Shankar, M. Tjersland, J. Ma, K. Stone, and M. Bajracharya, "A Learned Stereo Depth System for Robotic Manipulation in Homes," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, 2022.