CLOSURE: Fast Quantification of Pose Uncertainty Sets

Yihuai Gao[*†], Yukai Tang[*‡], Han Qi[§] and Heng Yang[§]
[†]Stanford University, `yihuai@stanford.edu`
[‡]Tsinghua University, `tangyk20@mails.tsinghua.edu.cn`
[§]Harvard University, `{hqi,hankyang}@g.harvard.edu`

Abstract—We investigate uncertainty quantification of 6D pose estimation from learned noisy measurements (*e.g.*, keypoints and pose hypotheses). Assuming *unknown-but-bounded* measurement noises, a *pose uncertainty set* (PURSE) is a subset of $\mathrm{SE}(3)$ that contains all possible 6D poses compatible with the measurements. Despite being simple to formulate and its ability to embed uncertainty, the PURSE is difficult to manipulate and interpret due to the many abstract nonconvex polynomial constraints defining it. An appealing simplification of PURSE–motivated by the bounded state estimation error assumption in robust control– is to find its *minimum enclosing geodesic ball* (MEGB), *i.e.*, a point pose estimation with minimum worst-case error bound. We contribute (i) a geometric interpretation of the nonconvex PURSE, and (ii) a fast algorithm to *inner* approximate the MEGB. Particularly, we show the PURSE corresponds to the feasible set of a constrained dynamical system or the intersection of multiple geodesic balls, and this perspective allows us to design an algorithm to densely sample the *boundary* of the PURSE through strategic random walks that are efficiently parallelizable on a GPU. We then use the miniball algorithm by Gärtner (1999) to compute the MEGB of PURSE samples, leading to an *inner* approximation of the true MEGB. Our algorithm is named CLOSURE (*enClosing baLl frOm purSe boUndaRy samplEs*) and it enables computing a *certificate of approximation tightness* by calculating the *relative ratio* between the size of the inner approximation and the size of the *outer* approximation GRCC from Tang, Lasserre, and Yang (2023). Running on a single RTX 3090 GPU, CLOSURE achieves the relative ratio of 92.8% on the LM-O object pose estimation dataset, 91.4% on the 3DMatch point cloud registration dataset and 96.6% on the LM object pose estimation dataset with an average runtime below 0.3 seconds. Obtaining comparable worst-case error bound but $398\times$, $833\times$ and $23.6\times$ faster than the outer approximation GRCC, CLOSURE enables uncertainty quantification of 6D pose estimation to be implemented in real-time robot perception applications.

I. INTRODUCTION

6D pose estimation (*i.e.*, a 3D rotation and a 3D translation) from images and point clouds is a longstanding problem in robotics and vision and finds extensive applications in localization and mapping [36], robotic manipulation [15], virtual and augmented reality [40], and autonomous driving [35].

We focus on two popular paradigms for pose estimation in this paper and aim to endow them with rigorous *uncertainty quantification*. The first paradigm is to start by detecting salient *keypoints* in the sensor data –often done using deep neural networks [12, 46, 32, 31]– and then leverage the *maximum likelihood estimation* (MLE) framework to estimate the

optimal pose that best fits the keypoint measurements. The second paradigm –initiated by PoseNet [22] and PoseCNN [41] but recently became state of the art via FoundationPose [40]– circumvents the need to detect keypoints and directly *regresses* pose hypotheses (potentially followed by another MLE step). Formally, let $x = (R, t) \in \mathrm{SO}(3) \times \mathbb{R}^3 := \mathrm{SE}(3)$ be the unknown pose to be estimated, both paradigms generate N noisy measurements $y_i \in \mathcal{Y}, i = 1, \ldots, N$ that satisfy

$$g(x, y_i) = \epsilon_i, \quad i = 1, \ldots, N, \tag{1}$$

where $g : \mathrm{SE}(3) \times \mathcal{Y} \to \mathbb{R}^m$ is a known residual function that measures the discrepancy between the measurement y_i and pose x. In the keypoint-based paradigm, y_i are keypoints; in the direct regression paradigm, y_i are regressed pose hypotheses. When y_i is a perfect (*i.e.*, noise-free) measurement, g evaluates to zero; when y_i is noisy, $\epsilon_i \in \mathbb{R}^m$ describes the measurement noise in neural network predictions. We give three instantiations of (1) that will be the focus of this paper.

Example 1 (Keypoint-based Object Pose Estimation [44, 24]). *Let $y_i = (z_i, Z_i) \in \mathbb{R}^2 \times \mathbb{R}^3$ be a pair of matched 2D image keypoint and 3D object keypoint. The function*

$$g(x, y_i) = z_i - \Pi(RZ_i + t) \tag{2}$$

describes the reprojection error *of the 3D keypoint Z_i, where $\Pi(\cdot)$ is the camera projection function.*[1]

Example 2 (Keypoint-based Point Cloud Registration [45, 12]). *Let $y_i = (a_i, b_i) \in \mathbb{R}^3 \times \mathbb{R}^3$ be a pair of matched 3D keypoints in the source and target point clouds, respectively. The function*

$$g(x, y_i) = b_i - (Ra_i + t) \tag{3}$$

describes the Euclidean error *between the keypoints.*

Example 3 (Direct Pose Regression [22, 41, 40]). *Let $y_i = (R_i, t_i) \in \mathrm{SE}(3)$ be a pose hypothesis, the function*

$$g(x, y_i) = \begin{bmatrix} \mathrm{vec}(R) - \mathrm{vec}(R_i) \\ t - t_i \end{bmatrix} \tag{4}$$

describes the relative pose *between (R, t) and (R_i, t_i), where $\mathrm{vec}(\cdot)$ vectorizes a matrix as a vector.*

Given the noisy measurements $\{y_i\}_{i=1}^N$, one then formulates

$$\min_{x \in \mathrm{SE}(3)} \sum_{i=1}^N \rho(g(x, z_i)), \tag{5}$$

---

[*] equal contribution.

[1] $\Pi : \mathbb{R}^3 \to \mathbb{R}^2$, $\Pi(v) = [v_1/v_3, v_2/v_3]^\mathsf{T}$.

whose solution provides an estimate of the unknown pose x. For example, when the noise ϵ_i is assumed to follow a standard Gaussian distribution, then choosing $\rho(g(x, z_i)) = \|g(x, z_i)\|^2$ makes the solution of (5) to be the maximum likelihood estimator. In practice, there are often measurements that do not follow (1) and lead to very large ϵ_i, commonly known as an *outlier*. To regain robustness to outliers, ρ in (5) is modified to be a *robust loss* [21, 6] and the optimization (5) is referred to as *M-estimation* (*i.e.,* MLE-like estimation).

Uncertainty Quantification. Despite the many algorithmic advances in solving (5) [8, 3, 10], *e.g.,* with certifiable global optimality guarantees [43], we argue there are two fundamental issues with the M-estimation framework. First, the starting assumption that ϵ_i follows a Gaussian-like distribution (up to the removal of outliers) is not justified. For example, [38] shows noises generated by neural network keypoint detections in Example 1 fail almost all statistical multivariate normality tests [25], regardless of whether outliers are removed. We reinforce this empirical observation in Supplementary Material and provide similar results showing real noises in keypoint matches of Example 2 also deviate far from a Gaussian distribution. Second, the solution of (5) provides a *single point estimate* that usually comes with no *uncertainty quantification*, which is crucial when the pose estimations need to be used for downstream decision-making [14, 11]. Though it is possible to use the inverse of Fisher Information at the optimal solution to approximate the uncertainty,[2] such approximation (i) again builds on the assumption that ϵ_i is Gaussian, and (ii) is known to underestimate the true uncertainty [37]. Recent works [33, 45, 9] provided uncertainty estimation for a few robot perception problems. However, the uncertainty either depends on uncheckable assumptions and cannot be computed [33, 45], or build on machinery that only applies to estimators based on expensive semidefinite relaxations [43].

Set-Membership Estimation with Noise Calibration. An alternative (albeit less popular) estimation framework that this paper advocates for the purpose of practical and rigorous uncertainty quantification is that of *set-membership estimation* (SME), widely known in control theory for system identification [29, 28, 39]. Instead of placing distributional assumptions on the measurement noise ϵ_i in (1), SME assumes the noise is *unknown but bounded*, *i.e.,*

$$\sqrt{\epsilon_i^\mathsf{T} \Lambda_i \epsilon_i} =: \|\epsilon_i\|_{\Lambda_i} \leq \beta_i, \quad i = 1, \ldots, N \tag{6}$$

for positive definite matrices $\Lambda_i \in \mathbb{S}_{++}^m$ and positive noise bounds $\beta_i > 0$. With this assumption, and invoking (1), a *Pose UnceRtainty SEt* (PURSE) can be formulated as [44]

$$S = \{x \in \mathrm{SE}(3) \mid \|g(x, y_i)\|_{\Lambda_i} \leq \beta_i, i = 1, \ldots, N\}, \tag{PURSE}$$

i.e., the set of all possible poses that are compatible with the measurements y_i and the assumption (6). The careful reader may ask "*why should we trust the unknown-but-bounded noise*

assumption (6) *in SME more than the Gaussian-like noise assumption in MLE?*" The answer is that recent statistical tools for distribution-free uncertainty calibration [34], *e.g., conformal prediction* [2], enables rigorous estimation of Λ_i and β_i from a calibration dataset. For instance, [44] demonstrated how to calibrate noises generated by a pretrained neural network for Example 1, and in Supplementary Material we show how to calibrate neural network detection noises for Example 2 and Example 3 using conformal prediction. In fact within robotics, the SME framework has recently also gained popularity in simultaneous localization and mapping (SLAM) for guaranteed error analysis [16, 30]. However, the SLAM literature used *interval analysis* to bound the estimation error of 2D poses, while in this paper we focus on uncertainty quantification of 3D rotations and translations.

Computational Challenges. Although SME provides a natural description of uncertainty and its unknown-but-bounded noise assumption can be made practical by modern uncertainty calibration tools, it does bring computational challenges because the PURSE is an abstract subset of $\mathrm{SE}(3)$ defined by many nonconvex constraints (more precisely, polynomial inequalities [44]). This makes PURSE difficult to be manipulated, *e.g.,* draw samples, visualize and interpret, extract a point estimate, compute volume, to name a few. Therefore, it is desirable to simplify the PURSE. Motivated by the common practice in robust perception-based control [14, 23, 13] that assumes the state estimation has bounded error, it is appropriate to simplify PURSE as a point estimate together with a *worst-case error bound*, *i.e.,* an *enclosing geodesic ball*[3] centered at the point estimate.[4] Since it is desired to find an enclosing geodesic ball with *minimum conservatism*, we wish to compute the *minimum enclosing geodesic ball* (MEGB), *i.e.,* the smallest geodesic ball that encloses the original PURSE (to be formulated precisely in Section II, see Fig. 1 for a graphical illustration). Towards this goal, [44] proposed an algorithm to compute the worst-case error bound for any point estimate, and [38] showed how to compute a hierarchy of enclosing geodesic balls (with decreasing sizes) that asymptotically converge to the MEGB. There are, however, two shortcomings in [44, 38]. First, both works provide *outer* approximations of the MEGB and it is unclear how conservative they are (*e.g.,* how much larger are these approximations compared to the true MEGB). Second, they rely on semidefinite relaxations for the outer approximation, which are too expensive to be practical for real-time robotics applications (see Section V).

Contributions. We contribute an algorithm that quantifies the uncertainty of PURSE in real time for Examples 1-3. The key perspective is that, *algebraically*, PURSE is defined by unstructured nonconvex constraints, but *geometrically*, PURSE corresponds to precisely the feasible sets of certain constrained dynamical systems in Examples 1-2, and the intersection of many geodesic balls in Example 3. This observation leads to a

---

[2]In linear least squares, this coincides with the covariance of the posterior distribution; in nonlinear least squares, this is called the Cramer-Rao lower bound, see [37, Section B.6] for a detailed explanation.

[3]Since $\mathrm{SE}(3)$ is a Riemannian manifold, we use the term "geodesic".

[4]Given an enclosing geodesic ball, one can draw samples from PURSE and estimate its volume by performing rejection sampling.
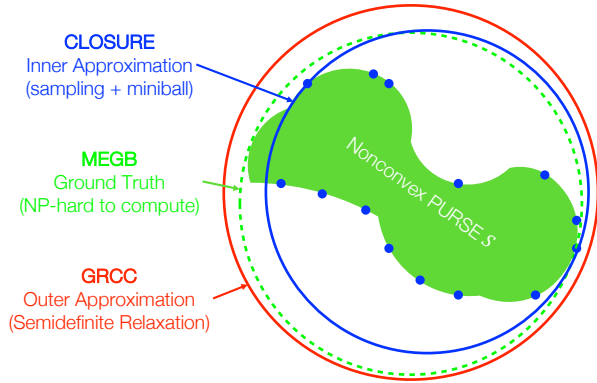
Fig. 1. Illustration of the relationship between the outer approximation (provided by the GRCC algorithm from [38]), inner approximation (provided by the proposed CLOSURE algorithm) and the ground truth MEGB. Blue dots are the boundary pose samples in PURSE, which are used to compute the inner approximation through the miniball algorithm [18].

natural algorithm: we first generate random samples in PURSE as initial states, and then perturb the states properly until it reaches the *boundary* of the feasible set. Effectively, this algorithm aims to densely sample the boundary of PURSE by *strategic random walks*, which lends itself to fast parallization. Using the samples on the PURSE boundary, we run the fast *miniball* algorithm by Gärtner [18] to return a geodesic ball enclosing the samples. We name our algorithm *enClosing baLl frOm purSe boUndaRy samplEs* (CLOSURE). This geodesic ball in turn becomes an *inner* approximation of the MEGB. By comparing the *relative ratio* between the size of the inner approximation computed by CLOSURE and the size of the outer approximation computed by [38], we obtain a *certificate of tightness* for the approximation, as shown in Fig. 1. We test CLOSURE on the LM-O dataset [7] for Example 1, the 3DMatch dataset [47] for Example 2, and the LM dataset [20] for Example 3, and demonstrate that (i) with an average runtime of 0.1879 second, CLOSURE computes inner approximations with average relative ratio 92.8% for Example 1, (ii) with an average runtime of 0.1774 second, CLOSURE computes inner approximations with average relative ratio 91.4% for Example 2 and (iii) with an average runtime of 0.2768 second, CLOSURE computes inner approximations with average relative ratio 96.6% for Example 3; all using a single NVIDIA RTX 3090 GPU. This means CLOSURE is $398\times$, $833\times$, and $23.6\times$ faster than the outer approximation algorithm in [38], respectively, making it feasible in real-time perception systems.

Paper Organization. We formalize the notion of MEGB and discuss high-level outer and inner approximation strategies in Section II. We then focus on the inner approximation: in Section III we show the geometric structure of PURSE and in Section IV we detail the CLOSURE algorithm. We provide experimental results in Section V and conclude in Section VI.

II. MINIMUM ENCLOSING GEODESIC BALL

Given the uncertainty set S defined in PURSE, denote $S_R \subseteq$ SO(3), $S_t \subseteq \mathbb{R}^3$ as the projection of S onto SO(3) and $\mathbb{R}^3$, respectively. Consider the optimization problems

$$\min_{C \in \mathrm{SO}(3)} \max_{R \in S_R} \mathrm{dist}_{\mathrm{SO}(3)}(C, R) \qquad (7)$$
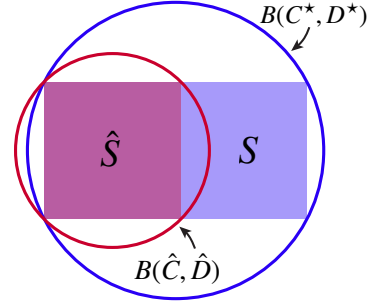

Fig. 2. MEGB $B(C^\star, D^\star)$ of a simple rectangle S and MEGB $B(\hat{C}, \hat{D})$ of the subset $\hat{S} \subset S$. Note that $\hat{D} < D^\star$ but $B(\hat{C}, \hat{D}) \not\subset B(C^\star, D^\star)$.

and

$$\min_{c \in \mathbb{R}^3} \max_{t \in S_t} \mathrm{dist}_{\mathbb{R}^3}(c, t) \qquad (8)$$

where $\mathrm{dist}_{\mathrm{SO}(3)}(C, R) := \arccos((\mathrm{tr}\left(C^\mathsf{T} R\right) - 1)/2)$ and $\mathrm{dist}_{\mathbb{R}^3}(c, t) := \|c - t\|$ denote the geodesic distance metrics in SO(3) and $\mathbb{R}^3$, respectively. Let $C^\star$ (resp. $D^\star$) and $c^\star$ (resp. $d^\star$) be the optimizer (resp. optimum) of (7) and (8), then the minimum enclosing geodesic ball (MEGB) of S_R is

$$B_{\mathrm{SO}(3)}(C^\star, D^\star) = \left\{ R \in \mathrm{SO}(3) \,|\, \mathrm{dist}_{\mathrm{SO}(3)}(R, C^\star) \leq D^\star \right\}, \quad (9)$$

and the MEGB of S_t is

$$B_{\mathbb{R}^3}(c^\star, d^\star) = \left\{ t \in \mathbb{R}^3 \mid \mathrm{dist}_{\mathbb{R}^3}(t, c^\star) \leq d^\star \right\}. \qquad (10)$$

By the minimax nature of problems (7)-(8), we have

$$S_R \subseteq B_{\mathrm{SO}(3)}(C^\star, D^\star), \quad S_t \subseteq B_{\mathbb{R}^3}(c^\star, d^\star),$$

i.e., the geodesic balls enclose the original uncertainty sets. See Fig. 1 for an illustration of the MEGB for a nonconvex PURSE. The center $(C^\star, c^\star) \in \mathrm{SE}(3)$ is the best point estimate achieving *minimum* worst-case error bounds of $(D^\star, d^\star)$. We choose to seperate the MEGB for SO(3) and $\mathbb{R}^3$ because their distance metrics have different units, *i.e.,* degrees and meters. A classical result in Riemannian geometry [1] states the MEGB exists and is unique under mild conditions. However, solving the minimax optimizations (7)(8) is known to be intractable [4]. We review several algorithmic choices for solving (7)(8).

Geodesic Gradient Descent. A general recipe leveraging the *geodesic convexity* of $f(C) := \max_{R \in S_R} \mathrm{dist}_{\mathrm{SO}(3)}(C, R)$ in C can be constructed as follows: start with an initial rotation $C^{(0)}$; then at each iteration (a) find the rotation R in S_R that attains the maximum distance to the current iterate $C^{(k)}$, and (b) move along the geodesic from $C^{(k)}$ to R by a portion of $\gamma \in (0, 1)$, which is equivalent to performing (sub)gradient descent on $f(C)$. Due to geodesic convexity of $f(C)$, this algorithm is guaranteed to converge with well-known complexity analysis [48]. However, computing the maximum distance from S_R to $C^{(k)}$ in step (a) boils down to solving a nonconvex optimization and is in general not implementable. In Supplementary Material we attempt this algorithm with step (a) implemented by the semidefinite relaxation proposed in [44] and show that it does converge but is impractical due to the excessive runtime (near one hour).

Since exactly solving (7)(8) using gradient descent is impractical, we turn to approximation algorithms.

Outer Approximation. [38] proposed an algorithm that produces a hierarchy of enclosing geodesic balls that asymptotically converge to the MEGB from above. This algorithm, however, has two drawbacks. First, the convergence is not detectable and therefore it is unclear how conservative is the computed enclosing geodesic ball compared to the true MEGB. Second, this algorithm is also built upon semidefinite relaxations and it is too slow to be practical (see Section V).

We focus on inner approximations in this paper.

Inner Approximation. If in addition to the outer approximation provided by [38], one can produce an *inner* approximation of the MEGB, then by comparing the relative ratio between the sizes of the inner and outer approximations, a certficate of approximation tightness can be obtained. The next result provides a straightforward way to compute inner approximations.

Proposition 4 (Inner Approximation of MEGB). *Let $\hat{S}_R \subseteq S_R$ and $\hat{S}_t \subseteq S_t$ be nonempty subsets. Consider the optimizations*

$$\min_{C \in \mathrm{SO}(3)} \max_{R \in \hat{S}_R} \ \mathrm{dist}_{\mathrm{SO}(3)}(C, R) \tag{11}$$

and

$$\min_{t \in \mathbb{R}^3} \max_{t \in \hat{S}_t} \ \mathrm{dist}_{\mathbb{R}^3}(c, t), \tag{12}$$

and denote their optimizers (resp. optima) to be $\hat{C}$ (resp. $\hat{D}$) and $\hat{c}$ (resp. $\hat{d}$). Then $B_{\mathrm{SO}(3)}(\hat{C}, \hat{D})$ is no greater than the MEGB (9) and $B_{\mathbb{R}^3}(\hat{c}, \hat{d})$ is no greater than the MEGB (10), i.e.,

$$\hat{D} \leq D^\star, \quad \hat{d} \leq d^\star. \tag{13}$$

Proof: In (7) and (11), for any $C \in \mathrm{SO}(3)$ denote

$$f(C) := \max_{R \in S_R} \mathrm{dist}_{\mathrm{SO}(3)}(C, R),$$
$$\hat{f}(C) := \max_{R \in \hat{S}_R} \mathrm{dist}_{\mathrm{SO}(3)}(C, R).$$

Since $\hat{S}_R \subseteq S_R$, we have

$$f(C) \geq \hat{f}(C), \quad \forall C \in \mathrm{SO}(3).$$

Therefore,

$$\min_{C \in \mathrm{SO}(3)} f(C) = D^\star \geq \hat{D} = \min_{C \in \mathrm{SO}(3)} \hat{f}(C).$$

The proof for $\hat{c} \leq c^\star$ is similar. ∎

Proposition 4 states that, given any nonempty subsets of the original PURSE S_R and S_t, solving the minimax problems (11)-(12) leads to valid inner approximations. We remark that here the word "inner" is abused in the sense that we can only guarantee the MEGB of $\hat{S}_R$ (resp. $\hat{S}_t$) is *smaller* than, but *not enclosed by*, the MEGB of the original PURSE S_R (resp. S_t). Fig. 2 gives such an counterexample. Nevertheless, we chose to use "inner" approximation to parallel the "outer" approximation in previous works. To make Proposition 4 useful, we need to strategically choose the subsets $\hat{S}_R$ and $\hat{S}_t$ such that (i) the minimax problems are easier to solve, and

(ii) the inner approximations are as tight (large) as possible. Requirement (i) is easy to satisfy if we choose the subsets as *discrete samples*, *i.e.,* we approximate the original PURSE as a point cloud of poses. In this case, the minimax problems can be solved exactly and efficiently [18], which we will further explain in Section IV. To satisfy requirement (ii), we leverage the intuition that the MEGB must touch the PURSE at a finite number of *contact points* [42, 27, 19] that lie on the boundary of the PURSE ∂S_R and ∂S_t. Therefore, if we can densely sample ∂S_R and ∂S_t, then there is high probability that the contact points will be included and the inner approximation will be close to the true MEGB. For example, the MEGB of the rectangle in Fig. 2 is exactly the same as the MEGB of the discrete point cloud containing four corners of the rectangle. Therefore, finding good samples on the boundary of the PURSE is crucial to compute tight inner approximations of the MEGB.

We summarize the outer approximation and the inner approximation in Fig. 1 for better understanding.

However, the PURSE is defined by abstract constraints and there exists no general algorithms that can sample its boundary, *i.e.,* points at which some of the inequalities become equalities. In the next section, we will show that, despite being algebraically unstructured, the PURSE has simple geometrical structures that can be exploited to sample its boundary.

III. GEOMETRIC STRUCTURES

Plugging the g functions in (2), (3), and (4) to the definition of PURSE, we explicitly write down the PURSE for Example 1

$$S = \{(R, t) \mid \|z_i - \Pi(RZ_i + t)\|_{\Lambda_i} \leq \beta_i, \forall i\}, \quad \text{(PURSE2D3D)}$$

for Example 2

$$S = \{(R, t) \mid \|b_i - Ra_i - t\|_{\Lambda_i} \leq \beta_i, \forall i\}. \quad \text{(PURSE3D3D)}$$
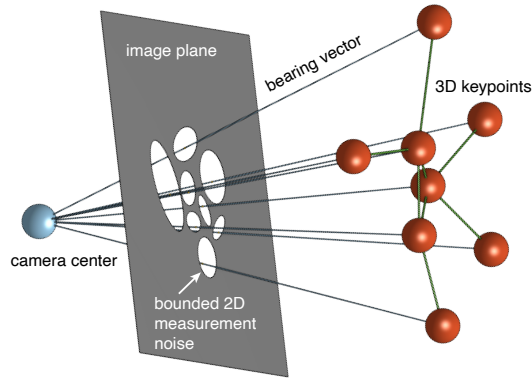
and for Example 3

$$S = \left\{(R, t) \mid \|\mathrm{vec}(R) - \mathrm{vec}(R_i)\|_{\Lambda_i^R} \leq \beta_i^R, \|t - t_i\|_{\Lambda_i^t} \leq \beta_i^t, \forall i\right\}.$$
$$\text{(PURSEreg)}$$

Note that in (PURSEreg) we separate the constraints in R and t since β_i^R and β_i^t have different units.

Although the constraints are complicated in (R, t), they actually define quite simple geometric sets.

PURSE2D3D. Each constraint in (PURSE2D3D) asks $\Pi(RZ_i + t)$ –the reprojection of Z_i– to lie inside an ellipse on the image plane. Fig. 3(a) depicts the geometric constraints: the line from the camera center to each 3D keypoint, *i.e.,* the bearing vector, needs to pass through the ellipse of bounded measurement noise. This is precisely a constrained mechanical system where there is (i) a prismatic joint between each 3D keypoint and each bearing vector, and (ii) a spherical joint between each bearing vector and the camera center. The N 3D keypoints form a rigid body (as shown by the connected links in Fig. 3(a)) that can move in 3D space subject to the bearing vectors not passing the 2D ellipses.

PURSE3D3D. Each constraint in (PURSE3D3D) enforces $Ra_i + b_i$ –the rigid transformation of a_i– to lie inside an ellipsoid centered at b_i. Fig. 3(b) depicts the geometric constraints:

(a) Constrained dynamical system for (PURSE2D3D).
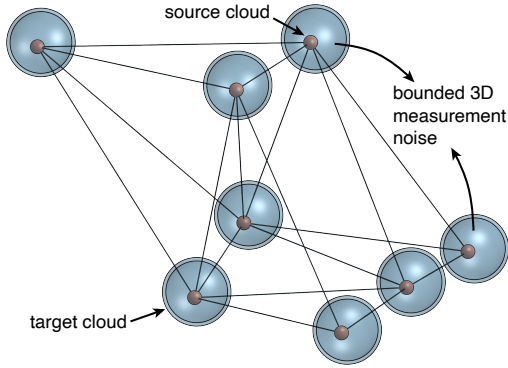
(b) Constrained dynamical system for (PURSE3D3D).

Fig. 3. Constrained dynamical systems whose feasible sets correspond to (a) (PURSE2D3D) for Example 1 and (b) (PURSE3D3D) for Example 2.

each of the ellipsoid forms a shell fixed in 3D space, and the rigid body formed by 3D points $\{a_i\}_{i=1}^N$ (again showned by the connected links) can freely move in 3D space subject to not escaping the ellipsoidal shells.

PURSEreg. Each constraint in (PURSEreg) enforces R (*resp.,* t) to lie inside a ball centered at R_i (*resp.,* t_i) with radius β_i^R (*resp.,* β_i^t). Therefore, PURSEreg is simply the intersection of N geodesic balls centered at R_i and $t_i, i = 1, \ldots, N$.

This geometric perspective inspires a natural algorithm to sample the boundary of PURSE, *i.e.,* we can start with random samples inside the PURSE and then add random perturbations to "walk" the samples on the $\mathrm{SE}(3)$ manifold until they "hit" the boundary of the PURSE. This is the CLOSURE algorithm to be engineered in the next section.

IV. THE CLOSURE ALGORITHM

The CLOSURE algorithm contains three steps, which is overviewed in Algorithm 1 and Fig. 4.

Step I: Initialize Starting Poses. Given a PURSE S, we first sample a set of initial poses S_0 from S. For Example 1 and Example 2, we use an algorithm proposed in [44] called *random sample averaging* (RANSAG). The basic idea of RANSAG is to leverage the well-established *minimal solvers* [26] to quickly generate candidate poses and check if they belong to PURSE. For Example 3, we design a convex-combinational sampler.
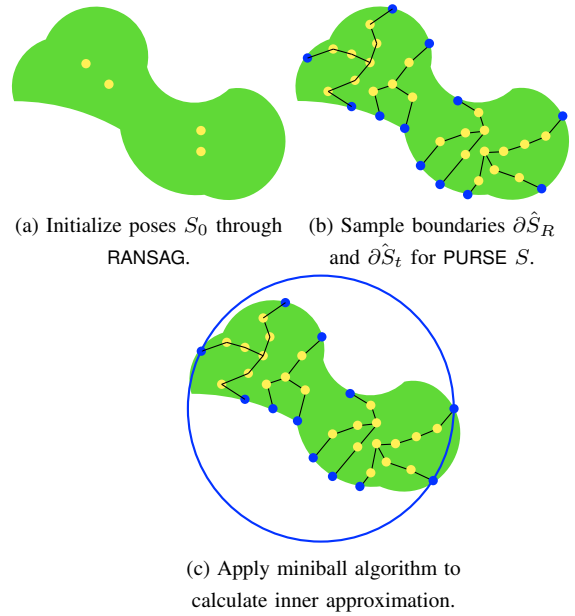

(a) Initialize poses S_0 through RANSAG.

(b) Sample boundaries $\partial \hat{S}_R$ and $\partial \hat{S}_t$ for PURSE S.

(c) Apply miniball algorithm to calculate inner approximation.

Fig. 4. Overview of the CLOSURE algorithm in 2D.

Algorithm 1: CLOSURE Overview

---

1 **Input:** a pose uncertainty set PURSE S; init sample trial number N_{sample}; simulation parameters $\mathcal{P}$;
2 **Output:** miniball center pose $s^* \in \mathrm{SE}(3)$; radius of rotation miniball $\hat{D}$; radius of translation miniball $\hat{d}$
3 $S_0 \leftarrow$ init_sample(S, N_{sample});
4 $\partial \hat{S}_R \leftarrow$ sample_rotation_boundary(S_0, S, $\mathcal{P}$);
5 $\partial \hat{S}_t \leftarrow$ sample_translation_boundary(S_0, S, $\mathcal{P}$);
6 $R^*, \hat{D} \leftarrow$ minimum_enclosing_geodesic_ball($\partial \hat{S}_R$);
7 $t^*, \hat{d} \leftarrow$ minimum_enclosing_ball($\partial \hat{S}_t$);
8 **return:** $s^* \leftarrow (R^*, t^*)$, $\hat{D}$, $\hat{d}$

- Example 1: In each RANSAG trial, we randomly select 3 constraints (*i.e.,* 3 ellipses in Fig. 3(a)) and find a 2D point in each constraint. We then apply the perspective-3-point (P3P) minimal solver [17] to obtain a candidate pose s. We add s into the set S_0 if s satisfies (PURSE2D3D).
- Example 2: In each RANSAG trial, we randomly select 3 pairs of corresponding points, and apply Arun's Method [5] to compute a candidate pose s. We put s into S_0 if s satisfies (PURSE3D3D).
- Example 3: In each trial, we randomly generate a convex combination of 10 pose hypotheses to compute a candidate pose s. We put s into S_0 if s satisfies (PURSEreg).

Step II: Strategic Random Walk. In this step, we sample poses that are close to the boundary of the PURSE, ∂S, starting from the initial samples $S_0 \subset S$ by Step I. The basic idea is to add random perturbations to every pose in S_0 until they hit the boundary of the PURSE and violate the defining constraints. We engineer several techniques to make this idea more efficient.

- **Walk away from the center**. We first find the average rotation $\bar{R}$ and average translation $\bar{t}$ of S_0 (line 3-

Algorithm 2: ∂S_R sampler

1 Input: initial poses $S_0 \subset \mathrm{SE}(3)$; PURSE S; base angular velocity magnitude ω_0; time step decay factor γ; translation perturbation scale t_p; random walk trial number N_W; iteration number N_I; perturbation number N_P; optimal perturbation number N_P^*; time step scaling number N_T;

2 Output: sampled boundary poses $\partial \hat{S}_R \subset \mathrm{SO}(3)$ as an inner approximation of ∂S_R;

3 $\bar{R} \leftarrow \mathrm{proj}_{\mathrm{SO}(3)}(\sum_{(R_j,*)\in S_0} R_j)$;

4 $\bar{t} \leftarrow \frac{1}{|S_0|}\sum_{(*,t_j)\in S_0} t_j$;

5 $\partial \hat{S}_R \leftarrow \emptyset$;

6 for $(R_0, t_0) \in S_0$ **do**

7 **for** $n \leftarrow 1$ *to* N_W **do**

8 $\omega \leftarrow \mathrm{init\_angular\_velocity}(R_0, \bar{R}, \omega_0)$;

9 $R^* \leftarrow R_0, t^* \leftarrow t_0$;

10   % Iterate N_I times so that the evolved pose gets close to ∂S_R

11   **for** $i \leftarrow 1$ *to* N_I **do**

12   % Randomize N_P translation perturbations
     for $j \leftarrow 1$ *to* N_P **do**

13 $t_j \leftarrow t^* + \mathrm{perturbation}(t_p)$;

14 $d_j \leftarrow \mathrm{dist}(R^*, t_j, \partial S)$;

15   **end**

16   % Pick out N_P^* perturbations that drag the pose away from ∂S

17 $\{j_k\}_{k=1}^{N_P^*} \leftarrow \mathrm{top\_k\_indices}(\{d_j\}_{j=1}^{N_P}, N_P^*)$;

18   **for** $k \leftarrow 1$ *to* N_p^* **do**

19   **for** $m \leftarrow 1$ *to* N_T **do**

20 $\Delta T \leftarrow \gamma^{m-1}$;

21 $R_{km} \leftarrow \mathrm{update\_rotation}(R^*, \omega, \Delta T)$;

22 $I_{km} \leftarrow \mathrm{in\_purse}(R_{km}, t_{j_k})$;

23   **end**

24   **end**

25   % Find the optimal pose that is still in S and has the maximum rotation movement

26 $m_0 \leftarrow \min\{m \mid \exists k \text{ s.t. } I_{km} = 1\}$;

27 $k_0 \leftarrow \{k \mid I_{km_0} = 1\}$;

28 $R^* \leftarrow R_{k_0 m_0}$;

29 $t^* \leftarrow t_{k_0 m_0}$;

30   **end**

31 $\partial \hat{S}_R \leftarrow \partial \hat{S}_R \cup R^*$;

32 **end**

33 end

34 return: $\partial \hat{S}_R$

Algorithm 3: ∂S_R parallel sampler

1 Input: initial poses $S_0 \subset \mathrm{SE}(3)$; PURSE S; base angular velocity magnitude ω_0; time step decay factor γ; translation perturbation scale t_p; random walk trial number N_W; iteration number N_I; perturbation number N_P; optimal perturbation number N_P^*; time step scaling number N_T;

2 Output: sampled boundary poses $\partial \hat{S}_R \subset \mathrm{SO}(3)$ as an inner approximation of ∂S_R;

3 $\omega \leftarrow \mathrm{init\_angular\_velocity}(S_0, \omega_0, N_W)$;

4 $v \leftarrow \mathrm{init\_center\_velocity}(S_0, v_0, N_W)$;

5 $(R^*, t^*) \leftarrow \mathrm{repeat}(S_0, N_W)$;

6 $\Delta T \leftarrow (1, \beta, \beta^2, \cdots, \beta^{N_T - 1})$;

7 for $i \leftarrow 1$ *to* N_I **do**

8 $R \leftarrow \mathrm{repeat}(R^*, N_P)$;

9 $t \leftarrow \mathrm{repeat}(t^*, N_P) + \mathrm{perturbation}(t_p, |S_0|N_W N_P)$;

10 $d \leftarrow \mathrm{dist}(R, t, \partial S)$;

11 $j \leftarrow \mathrm{top\_k\_indices}(d, N_P^*)$;

12 $\tilde{t} \leftarrow \mathrm{repeat}(t_{[j]}, N_T)$;

13 $\tilde{R} \leftarrow \mathrm{update\_rotation}(\mathrm{repeat}(R^*, N_P^*), \omega, \Delta T)$;

14 $I \leftarrow \mathrm{in\_purse}(\tilde{R}, \tilde{t})$;

15 $R^*, t^* \leftarrow \mathrm{find\_farthest\_rotation}(\tilde{R}, \tilde{t}, I)$;

16 end

17 return: $\partial \hat{S}_R \leftarrow R^*$

where ω_0 is angular velocity magnitude and $\tilde{u} \in \mathbb{R}^3$ is a random unit vector. Similarly, when sampling the translation boundary ∂S_t, we initialize the center velocity v as $v = v_0 ((t_0 - \bar{t})/\|t_0 - \bar{t}\| + \tilde{v})$, where $\tilde{v} \in \mathbb{R}^3$ is a random unit vector. However, in Example 1, S_t in the camera projection direction appears significantly longer than the other two directions as shown in Fig. 5(a). To balance the geometrical singularity, we apply additional normalization to the translation velocity according to the PCA analysis of S_0.

- **$\partial S \neq \partial S_R \times \partial S_t$.** A vanilla walking strategy is to perform N_I iterations of rigid body movement with a variety of perturbations and step sizes (reflected in the time step lengths) so that the evolved poses can get close enough to ∂S. However, recall that our starting goal is to sample ∂S_R and ∂S_t, not ∂S. One may think that $\partial S = \partial S_R \times \partial S_t$, but this is in general *not true*. In fact, $\partial S_R = \partial(S|_{\mathrm{SO}(3)}) \subset (\partial S)|_{\mathrm{SO}(3)}$ and $\partial S_t = \partial(S|_{\mathbb{R}^3}) \subset (\partial S)|_{\mathbb{R}^3}$. In words, the boundary of S_R (resp. S_t) is a subset of the boundary of S projected onto $\mathrm{SO}(3)$ (resp. $\mathbb{R}^3$). This means even if $(R, t) \in \partial S$, it is not guaranteed that $R \in \partial S_R$ and $t \in \partial S_t$. Please refer to Supplementary Material for an intuitive example. With this observation in mind, we sample ∂S_R and ∂S_t separately. The ∂S_R sampler is shown in Algorithm 2, and the ∂S_t sampler is similar and presented in Supplementary Material for brevity.

  When sampling ∂S_R (resp. ∂S_t), we fix the angular velocity ω (resp. center velocity v) and add random perturbations to center translation (resp. rotation). In each

4). For each pose (R_0, t_0) in S_0, we initialize N_W walks with randomized angular or center velocities while pointing outwards $\bar{R}$ and $\bar{t}$, which explicitly encourages the samples to move away from the center and explore the boundary. To be exact, when sampling the rotation boundary ∂S_R, suppose (u, θ) is the axis-angle representation of the relative rotation between R_0 and $\bar{R}$, we initialize the angular velocity ω as $\omega = \omega_0(u + \tilde{u})$,

iteration step, we first find top N_p^* perturbations out of N_p that *drags the pose away from* ∂S, *i.e.,* maximizing the distance to ∂S. The distance between a pose $(R, t) \in \text{SE}(3)$ and ∂S is defined heuristically as

$$\text{dist}(R, t, \partial S) = \min_i \left(\beta_i - \|g_i((R, t), y_i)\|_{\Lambda_i} \right), \quad (14)$$

where g_i is the estimation error defined in (2)-(4). Then we apply different step scales $\gamma^{m-1}, m = 1, \cdots, N_T$ to the pose after each perturbation. Among all the $N_p^* N_T$ poses, we find the one that is still in S and has the maximum rotation (resp. translation) movement to update the optimal pose (R^*, t^*). Finally, after N_I iterations, we add the optimal pose to $\partial \hat{S}_R$. In this way, we can sample $|S_0| N_W$ poses that are close enough to ∂S_R (resp. ∂S_t) from different approaching directions. Algorithm 2 summarizes the strategic random walk that samples ∂S_R.

- **Parallelization**. To improve the sampling speed, we implement Algorithm 2 with parallel programming using CuPy,[5] presented in Algorithm 3. We notice that (R^*, t^*) is the only variable that is updated consecutively in each iteration, while all the other loops are independent and can be executed in parallel. Specifically, there are $|S_0| N_W$ walks in total with different initial velocities. In each iteration, the N_P perturbations can be applied to the optimal poses in parallel, from which we can find N_P^* optimal perturbations that maximize the distance to ∂S according to (14). Next, we apply all N_T time step scales in a roll and broadcast to N_P^* optimal perturbations for object movement simulation. At the end of each iteration, we update the optimal poses (R^*, t^*) with the farthest movements. With in-depth analysis of the sampling algorithm, the steps with heaviest computation are 10 and 14. In these steps, $|S_0| N_W N_P$ and $|S_0| N_W N_P^* N_T$ poses[6] are checked, while checking each pose requires to calculate (PURSE2D3D)-(PURSEreg). Executing these steps in parallel significantly reduces the time cost compared with CPU based implementation.

Step III: Miniball. In this step, after acquiring a set of rotations $\partial \hat{S}_R$ and translations $\partial \hat{S}_t$ close to ∂S_R and ∂S_t respectively, we calculate the MEGB of $\partial \hat{S}_R$ and $\partial \hat{S}_t$. The *miniball* algorithm[7] introduced in [18] provides a fast implementation that can exactly compute the MEGB of a point cloud in Euclidean space. This means we can directly apply it to solve (12) and compute $B_{\mathbb{R}^3}(\hat{c}, \hat{d})$ because $\partial \hat{S}_t \subset \mathbb{R}^3$ lives in an Euclidean space. However, we cannot directly apply miniball to $\partial \hat{S}_R$ because it lives in SO(3). Fortunately, [38, Proposition A12, A13] points out, under mild conditions, computing the MEGB of a set on SO(3) is equivalent to computing the MEGB of the corresponding set on the space of unit quaternions, which in turn is equivalent to simply treating the unit quaternions as points in $\mathbb{R}^4$. Therefore, we can apply

the same algorithm after embedding the rotations in $\mathbb{R}^4$ and solve (11) to compute $B_{\text{SO}(3)}(\hat{C}, \hat{D})$. Thanks to the efficiency of the miniball algorithm, the runtime of this step is negligible compared to RANSAG and the strategic random walk.

In summary, the CLOSURE algorithm takes the PURSE as input, and outputs $B_{\text{SO}(3)}(\hat{C}, \hat{D})$ and $B_{\mathbb{R}^3}(\hat{c}, \hat{d})$, serving as inner approximations of the true MEGB. By taking advantage of efficient GPU parallel computing, CLOSURE can be executed around 0.2 seconds with satisfying performance and is suitable for real-time applications.

V. EXPERIMENTS

We test CLOSURE on three real datasets, the LM-O dataset [7] for object pose estimation (Example 1), the 3DMatch dataset [47] for point cloud registration (Example 2) and the LM dataset [20] for pose regression (Example 3). Since CLOSURE takes in the PURSE description, we briefly describe how we process the datasets to obtain PURSE descriptions.

- LM-O dataset. The LM-O test dataset contains 1214 images each capturing 8 different objects on a table and the goal here to estimate the 6D pose of each object while quantifying uncertainty. We leverage the pretrained semantic keypoint detector [31] to detect 2D semantic keypoints $\{z_i\}_{i=1}^N$ of each object that are matched to the manually labeled 3D keypoints $\{Z_i\}_{i=1}^N$ (N is around 10). To calibrate the uncertainty of the 2D keypoint detections, we follow the conformal prediction approach in [44], which produces descriptions of pose uncertainty in the form of (PURSE2D3D).
- 3DMatch dataset. The 3DMatch test dataset includes 1623 pairs of point clouds and the goal here is to estimate the 6D rigid transformation between each pair of clouds while quantifying uncertainty. We leverage the pretrained DGR network [12] that detects salient keypoint matches $\{a_i, b_i\}_{i=1}^K$ between each pair of clouds. To calibrate the uncertainty of the keypoint matches, we design a similar conformal prediction procedure as [44] with 400 pairs in the calibration dataset. The details of the conformal prediction design are presented in Supplementary Material. After conformal prediction, pose uncertainty is given in the form of (PURSE3D3D) with $N = 50$.
- LM dataset. We use 15787 images from 13 objects in the LM dataset. The goal here is to estimate the 6D pose of each object while quantifying the uncertainty. We leverage the pretrained network FoundationPose [40] to directly output pose hypotheses. To calibrate the uncertainty of the output poses, we design a conformal prediction procedure leveraging the top 10 pose hypotheses and their scores. After conformal prediction, the pose uncertainty set is given in the form of (PURSEreg).

A. Effectiveness of the Boundary Sampler

A key idea in CLOSURE is that it tries to apply strategic random walks to sample the boundary of the PURSE. Here we investigate how effective is this boundary sampling strategy on two examples. To do so, we compare CLOSURE with pure

---

[5]https://cupy.dev/, a python interface for NVIDIA CUDA library.
[6]In a typical case, $|S_0| \approx 100, N_W = 2, N_P = 150, N_P^* = 10, N_T = 15$, thus $|S_0| N_W N_P = |S_0| N_W N_P^* N_T \approx 3 \times 10^4$.
[7]https://people.inf.ethz.ch/gaertner/subdir/software/miniball.html

RANSAG, *i.e.,* sampling inside PURSE without any motivation to sample the boundary. We choose one example in the LM-O dataset and one example in 3DMatch dataset. We compare CLOSURE with pure RANSAG for 100000 trials, where CLOSURE runs approixmately 0.2 second and RANSAG runs 1 second. Fig. 5 plots the sampling results of the two algorithms. We use the stereographic projection to visualize the rotation samples (*i.e.,* we convert rotations to unit quaternions and use stereographic projection to represent unit quaternions on a 3D sphere). Note that although CLOSURE generates fewer samples, we can clearly see that the samples from CLOSURE are more spread-out than those from RANSAG only. For this reason, the geodesic balls enclosing the CLOSURE samples are larger than the geodesic balls enclosing the RANSAG samples.

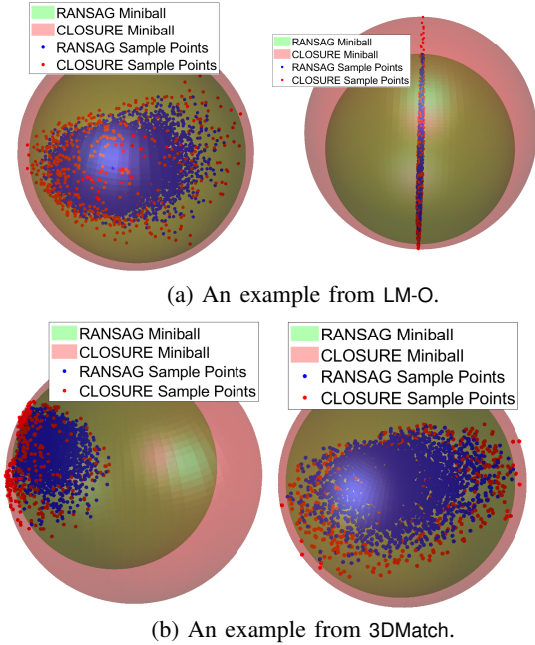(a) An example from LM-O.

(b) An example from 3DMatch.

Fig. 5. Effectiveness of PURSE boundary sampler on (a) an example from LM-O, and (b) an example from 3DMatch. Left: rotation, Right: translation.

B. Certificate of Approximation Tightness

We then perform a large-scale experiment of uncertainty quantification on all the test samples of LM-O, 3DMatch, and LM (excluding the calibration samples), where we compare the performance of CLOSURE with other baseline algorithms. Our algorithm CLOSURE runs on a single CPU with a RTX 3090 GPU, while the other baselines run on a workstations with 128 x AMD Ryzen Threadripper PRO 5995WX 64-Cores CPUs. Specifically, RANSAG and RANSAG+fmincon run with MATLAB Parallel Computing Toolbox.

1) LM-O: We implement two versions of CLOSURE
- CLOSURE: the default CLOSURE algorithm with parameters $N_{\text{sample}} = 1500$, $\omega_0 = 1$, $v_0 = 2$, $\gamma = 0.5$, $R_p = 0.2$, $t_p = 0.1$, $N_W = 2$, $N_I = 5$, $N_p = 150$, $N_T = 15$.
- CLOSURE++: the more accurate algorithm with parameters $N_{\text{sample}} = 1500$, $\omega_0 = 1$, $v_0 = 2$, $\gamma = 0.5$, $R_p = 0.2$, $t_p = 0.1$, $N_W = 10$, $N_I = 10$, $N_p = 150$, $N_T = 15$.

We implement the following baselines
- RANSAG: pure RANSAG sampling with the miniball algorithm. We choose the number of pure RANSAG trials so that its runtime is roughly one second on the workstation with parallel workers.
- RANSAG+fmincon: to encourage the RANSAG initial samples to walk to the boundary of PURSE, we explicitly solve an optimization problem

$$\text{Problem}(i) = \max_{(R,t) \in S} \|z_i - \Pi(RZ_i + t)\|_{\Lambda_i} \qquad (15)$$

where z_i is the detected 2D key point, Z_i is the 3D key point, and maximizing the objective explicitly pushes the rigid body system in Fig. 3(a) to hit the boundary of the i-th ellipse. The algorithm proceeds as in Algorithm 4 and we solve the optimization problems using Matlab function `fmincon` with parallel workers.
- GRCC: the algorithm from [38] that computes an outer approximation of the MEGB of PURSE with relaxation order $\kappa = 2$ for translation and $\kappa = 3$ for rotation. It provides an outer approximation of the MEGB (*i.e.,* $\bar{D}$ and $\bar{d}$), which is used to calculate η_R and η_t for other methods.

We investigate the runtime of these five algorithms, and more importantly, study how tight the inner and outer approximations of the MEGB are. Specifically, GRCC is the only algorithm that produces upper bounds on the size of the true MEGB. Let $\bar{D}$ and $\bar{d}$ be the radii of the enclosing geodesic balls produced by GRCC on $\text{SO}(3)$ and $\mathbb{R}^3$, respectively. The other four algorithms all produce lower bounds on the size of the true MEGB. Let $\hat{D}$ and $\hat{d}$ be the radii of the geodesic balls produced by these inner approximation algorithms. Define the *relative ratios*

$$\eta_R := \frac{\hat{D}}{\bar{D}} \leq 1, \quad \eta_t := \frac{\hat{d}}{\bar{d}} \leq 1. \qquad (16)$$

Clearly, $\eta_R = 1$ (resp. $\eta_t = 1$) certifies that the true MEGB has been found, and the closer η_R, η_t are w.r.t 1 the tighter the inner approximations are. Therefore, η_R, η_t produce certificates of approximation tightness.

Results. Table I summarizes the average runtime and the relative rotation and translation ratios. We observe that (i) CLOSURE runs below 0.2 seconds on average and attains $\eta_R > 0.92, \eta_t > 0.97$, making it the method of choice for real-time uncertainty quantification; (ii) with more runtime budget, CLOSURE++ runs below 1 second but boosts the relative ratios to almost 1; (iii) RANSAG, without the strategic random walks to sample the boundary, runs slower with worse relative ratios, again showing the value of the PURSE boundary sampler. (iv) RANSAG+fmincon attains the best η_R but the worst η_t, The reason is two-fold. First, from the set shape perspective, using the keypoint-based algorithm leads to information loss along the z-direction (depth direction), which causes extremely large uncertainty of the set S_t along the z-direction. As depicted in Fig. 5, S_t shapes like an extremely thin ellipsoid, which makes sampling the translation points near its two ends very difficult.

Algorithm 4: RANSAG+fmincon

---

1 **Input:** Calibration dataset $\mathcal{D}$, a pose uncertainty set PURSE S; RANSAG trial number N_{sample}; key point search index set $\mathcal{I}$;

2 **Output:** sampled boundary poses $\hat{\partial S} \subset \text{SE}(3)$ in PURSE; center pose $s^* \in \text{SE}(3)$; radius of minimum enclosing geodesic ball $\hat{D}$; radius of minimum enclosing ball $\hat{d}$

3 $S_0 \leftarrow$ RANSAG (S, N_{sample});

4 $\hat{\partial S} \leftarrow S_0$;

5 **for** $i \in \mathcal{I}$ **do**

6   **for** $(R_0, t_0) \in S_0$ **do**

7       $(R, t) \leftarrow$ fmincon(Problem(i), init:(R_0, t_0));

8   **if** $(R, t) \in S$ **then**

9          $\hat{\partial S} \leftarrow \hat{\partial S} \cup \{(R, t)\}$;

10   **end**

11   **end**

12 **end**

13 $R^*, \hat{D} \leftarrow$ minimum_enclosing_geodesic_ball$((\hat{\partial S})|_{\text{SO}(3)})$;

14 $t^*, \hat{d} \leftarrow$ minimum_enclosing_ball$((\hat{\partial S})|_{\mathbb{R}^3})$;

15 **return:** $\hat{\partial S}$, $s^* \leftarrow (R^*, t^*)$, $\hat{D}$, $\hat{d}$

Second, from the algorithm perspective, the RANSAG+fmincon algorithm is designed to push one of the keypoints to the boundary using any possible means. (a) In this sense, we can only guarantee that the pose (R, t) is on the boundary of the PURSE set. However, $\partial S \neq \partial S_R \times \partial S_t$. And this intuition isn't encoded in the RANSAG+fmincon algorithm. (b) Because the RANSAG+fmincon algorithm only seeks one way to push the key point to the boundary. So in general it will find the simplest way to push the key point to the boundary. Thus, it's easier to do more rotation than to use the translation at two ends of S_t. This makes the algorithm perform the worst in translation compared to the other algorithms. (v) The GRCC algorithm, despite being the only algorithm that can produce outer approximations, is too slow for real-time applications. We believe the results in Table I show that CLOSURE and CLOSURE++ can be real-time alternatives of the GRCC because the amount of underestimation of the uncertainty is very minor.

TABLE I
PERFORMANCE OF 5 METHODS ON LM-O (EXAMPLE 1)

	avg runtime (seconds)	avg η_R	avg η_t
CLOSURE	**0.1879**	0.9280	0.9781
CLOSURE++	0.7563	0.9660	**0.9918**
RANSAG	1.8368	0.8916	0.8791
RANSAG+fmincon	6.1095	**0.9726**	0.8119
GRCC	74.9724	\	\

2) 3DMatch*:* We implement two versions of CLOSURE

- CLOSURE: $N_{\text{sample}} = 1500$, $\omega_0 = 0.5$, $v_0 = 0$, $\gamma = 0.5$, $R_p = 0$, $t_p = 0.1$, $N_W = 2$, $N_I = 5$, $N_p = 150$, $N_T = 15$.
- CLOSURE++: $N_{\text{sample}} = 1500$, $\omega_0 = 0.5$, $v_0 = 0$, $\gamma = 0.5$, $R_p = 0$, $t_p = 0.1$, $N_W = 20$, $N_I = 10$, $N_p = 150$, $N_T = 15$

We implement the same RANSAG and GRCC baselines as in the LM-O case. In the new implementation of the RANSAG+fmincon algorithm under PURSE3D3D settings, we leverage the constraint pruning technique in [38] to speed up the computation of the algorithm.

Since the number of constraints in PURSE3D3D is much larger than that in PURSE2D3D, we limit the $|\mathcal{I}|$ to 50 (Setting I), and 80 (Setting II), which will lead to faster computation but worse ratio than using all of the RANSAG results. We then investigate the runtime of these algorithms as well as the certifcates of approximation tightness as defined in (16).

Results. Table II summarizes the average runtime and the relative rotation and translation ratios. We observe that (i) CLOSURE runs below 0.2 seconds on average and attains $\eta_R > 0.91, \eta_t > 0.93$, which benefits real-time uncertainty quantification. (ii) CLOSURE++ consumes more time but achieves the most accurate relative ratios. (iii) RANSAG runs slow and has the worst relative ratios. (iv) GRCC is too slow for real-time applications. (v) RANSAG+fmincon algorithm fails to achieve tight approximation ratios in both rotation and translation even with extremely high runtime. We suspect there are two reasons for this. (a) fmincon runs slow when the number of constraints is large, which makes the result worse when runtime is limited. (b) RANSAG+fmincon pushes one of the keypoints to ∂S, but it doesn't guarantee to touch S_R and S_t. Under PURSE3D3D settings, the number of keypoints is so large that it's more difficult to find the one that touches ∂S_R and ∂S_t.

TABLE II
PERFORMANCE OF 6 METHODS ON 3DMatch (EXAMPLE 2)

	avg runtime (seconds)	avg η_R	avg η_t
CLOSURE	**0.1774**	0.9140	0.9364
CLOSURE++	1.3362	**0.9318**	**0.9563**
RANSAG	1.2820	0.8480	0.8689
RANSAG+fmincon I	32.0642	0.7254	0.7518
RANSAG+fmincon II	43.2750	0.8323	0.8600
GRCC	147.8408	\	\

3) LM*:* We implement CLOSURE using the parameters listed as follows:

- CLOSURE: $N_{\text{sample}} = 200$, $\omega_0 = 0.5$, $v_0 = 0$, $\gamma = 0.5$, $R_p = 0$, $t_p = 0.1$, $N_W = 20$, $N_I = 5$, $N_p = 150$, $N_T = 10$.

We implement GRCC as in the LM-O and 3DMatch cases. We then investigate the runtime of the CLOSURE algorithm as well as the certifcates of approximation tightness as defined in (16). The random sampling baseline (like RANSAG) is not included here because we only sample from the "convex hull" defined by pose hypotheses, so it is unlikely the random samples can possibly filled the whole space.

Results. Table III summarizes the average runtime and the relative rotation and translation ratios. We observe that (i) CLOSURE runs below 0.3 seconds on average and attains $\eta_R > 0.96, \eta_t > 0.99$. (ii) GRCC is relatively faster than previous examples (Example 1 and Example 2) because: (a) we separately compute the bound for rotation and translation, which reduces the size of the optimization problem in GRCC,

(b) the constraints are simpler. However, it's still too slow for real-time applications.

TABLE III
PERFORMANCE OF 2 METHODS ON LM (EXAMPLE 3)

	avg runtime (seconds)	avg η_R	avg η_t
CLOSURE	**0.2768**	0.9659	0.9909
GRCC	6.5405	\	\

C. Time Decomposition

We implement two versions of CLOSURE

- CLOSURE: $N_{\text{sample}} = 1500$, $\omega_0 = 0.5$, $v_0 = 0$, $\gamma = 0.5$, $R_p = 0$, $t_p = 0.1$, $N_W = 2$, $N_I = 5$, $N_p = 150$, $N_T = 15$.
- CLOSURE++: $N_{\text{sample}} = 1500$, $\omega_0 = 0.5$, $v_0 = 0$, $\gamma = 0.5$, $R_p = 0$, $t_p = 0.1$, $N_W = 20$, $N_I = 10$, $N_p = 150$, $N_T = 15$

We analyze the time decomposition of CLOSURE. Our algorithm consists of three steps as described in Section IV: (1) RANSAG sampling, (2) strategic random walk, (3) miniball. The result of the time decomposition in Example 1 and Example 2 are shown in Table IV. We see that for CLOSURE, the time of the RANSAG sampling is comparable to that of strategic random walk, and the miniball time is negligible.

TABLE IV
TIME DECOMPOSITION OF CLOSURE AND CLOSURE++.

	RANSAG	Random walk	Miniball
Example 1 CLOSURE	0.0900	0.0915	0.0064
Example 1 CLOSURE++	0.0869	0.6235	0.0459
Example 2 CLOSURE	0.0883	0.0838	0.0053
Example 2 CLOSURE++	0.0874	1.1997	0.0491

D. Ablation: Sensitivity to Parameters

We study how the parameters of CLOSURE impact its efficiency, *i.e.,* how the performance (in terms of relative ratios) and runtime changes with the input parameter set. We present the results for sampling the rotation boundary ∂S_R of (PURSE3D3D) and refer the interested reader to Supplementary Material for more results.

Specifically, though a large parameter set is available to tune, the parameters that have the largest impact on the efficiency of CLOSURE are iteration step number N_I and number of the parallel workers N_W, N_T, N_P, N_P^*. We initialize the parameter set with $N_I = 5, N_W = 20, N_T = 15, N_P = 150, N_P^* = 10, \omega = 0.5, \beta = 0.5, t_p = 0.1$. When setting new values to each parameter, the rest of the parameters are kept the same as the initial values. During the study, we fix the RANSAG output of each experiment and compare the performance of different parameter values. The results are shown in Fig. 6, which demonstrate that CLOSURE's performance is quite robust to parameter tuning. For all the parameter choices, CLOSURE achieves relative ratio $\eta_R > 90\%$ and its runtime is around 1 second.

It is clear that both runtime and η_R grows as the all numbers increase, especially when the numbers start from a small value.
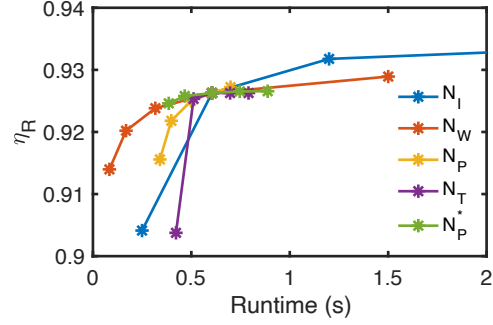

Fig. 6. The ratio-runtime curve of different parameters. $N_I \in \{2, 5, 10, 20\}$, $N_W \in \{2, 5, 10, 20, 50\}$, $N_P \in \{20, 50, 100, 150, 200\}$, $N_T \in \{5, 10, 15, 20, 25\}$, $N_P^* \in \{2, 5, 10, 15, 20\}$

However, as N_I and N_W grow higher, η_R saturates around 0.93. This might be a result that the outer approximation $\bar{D}$ provided by GRCC is not tight enough. Therefore, even if the sampler fully explores ∂S_R, the relative ratio will not reach 1. In the mean while, the sensitivity is not significant as the η_R stays above 0.9 for a wide range of parameter values.

E. How Large is the Pose Uncertainty?

The previous results focused on showing the relative ratios that are informative for the approximation performance of CLOSURE. In Fig. 7, we show the cumulative distribution of absolute sizes of the inner approximations of MEGB, just so the reader is aware of how much uncertainty is induced from the calibrated noise bounds on the learned measurements generated by modern neural networks. We can draw two observations from Fig. 7. (i) Comparing the two keypoint-based methods, *i.e.,* [31] for LM-O and DGR [12] for 3DMatch, DGR has lower uncertainty than [31]. There are two possible reasons for this: (a) 3D-3D keypoint matches better constrain the pose hypothesis space than 2D-3D keypoint matches; (b) DGR was proposed three years after [31] and it is plausible that DGR is better trained. (ii) Comparing the direct pose regression paradigm with the keypoint-based paradigm, we see that direct pose regression, in particular the state-of-the-art FoundationPose model, has much smaller uncertainty. The rotation uncertainty in FoundationPose is around 4 degrees and the translation uncertainty in FoundationPose is around 8mm. These are almost one order of magnitude better than DGR [12] and [31]. As far as we know, this is the first time such small calibrated pose uncertainty is reported in the literature.

VI. CONCLUSION

We introduced CLOSURE, a GPU-accelerated fast algorithm that can quantify pose uncertainty in real time from learned noisy measurements such as keypoints and pose hypotheses. The key perspective that led to the design of CLOSURE is that the pose uncertainty set (PURSE), despite being algebraically unstructured, has nice geometric interpretations. The key algorithmic component of CLOSURE is a strategy to generate random walks towards the boundary of the PURSE, and the key enabler for the efficiency of CLOSURE is parallel computing in GPUs. With three real-world datasets, we demonstrate the
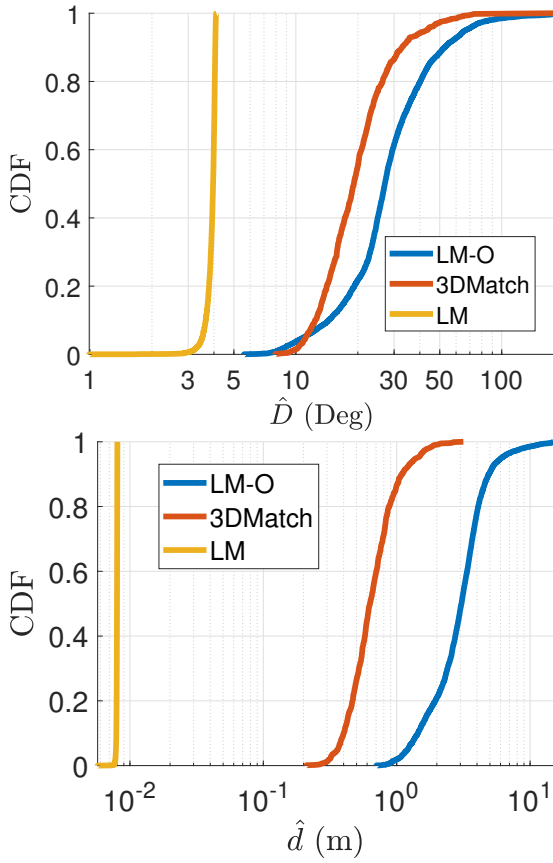
Fig. 7. Cumulative distributions of the absolute radii of the enclosing geodesic balls produced by CLOSURE for Examples 1-3. Top: rotation, bottom: translation. Note the x-axis is plotted in log scale.

efficiency and effectiveness of CLOSURE in producing tight uncertainty estimation of 6D poses.

REFERENCES

[1] Bijan Afsari. Riemannian lp center of mass: existence, uniqueness, and convexity. *Proceedings of the American Mathematical Society*, 139(2):655–673, 2011.

[2] Anastasios N Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021.

[3] Pasquale Antonante, Vasileios Tzoumas, Heng Yang, and Luca Carlone. Outlier-robust estimation: Hardness, minimally tuned algorithms, and applications. *IEEE Transactions on Robotics*, 38(1):281–301, 2021.

[4] Marc Arnaudon and Frank Nielsen. On approximating the riemannian 1-center. *Computational Geometry*, 46 (1):93–104, 2013.

[5] K Somani Arun, Thomas S Huang, and Steven D Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, (5):698–700, 1987.

[6] Michael J Black and Anand Rangarajan. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *International journal of computer vision*, 19(1):57–91, 1996.

[7] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13*, pages 536–551. Springer, 2014.

[8] Alvaro Parra Bustos and Tat-Jun Chin. Guaranteed outlier removal for point cloud registration with correspondences. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2868–2882, 2017.

[9] Luca Carlone et al. Estimation contracts for outlier-robust geometric perception. *Foundations and Trends® in Robotics*, 11(2-3):90–224, 2023.

[10] Tat-Jun Chin and David Suter. *The maximum consensus problem: recent algorithmic advances*. Springer Nature, 2022.

[11] Glen Chou, Necmiye Ozay, and Dmitry Berenson. Safe output feedback motion planning from images via learned perception modules and contraction theory. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 349–367. Springer, 2022.

[12] Christopher Choy, Wei Dong, and Vladlen Koltun. Deep global registration. In *CVPR*, 2020.

[13] Ryan K Cosner, Andrew W Singletary, Andrew J Taylor, Tamas G Molnar, Katherine L Bouman, and Aaron D Ames. Measurement-robust control barrier functions: Certainty in safety with uncertainty in state. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6286–6291. IEEE, 2021.

[14] Sarah Dean, Nikolai Matni, Benjamin Recht, and Vickie Ye. Robust guarantees for perception-based control. In *Learning for Dynamics and Control*, pages 350–360. PMLR, 2020.

[15] Xinke Deng, Yu Xiang, Arsalan Mousavian, Clemens Eppner, Timothy Bretl, and Dieter Fox. Self-supervised 6d object pose estimation for robot manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3665–3671. IEEE, 2020.

[16] Aaronkumar Ehambram, Raphael Voges, Claus Brenner, and Bernardo Wagner. Interval-based visual-inertial lidar slam with anchoring poses. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7589–7596. IEEE, 2022.

[17] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):930–943, 2003.

[18] Bernd Gärtner. Fast and robust smallest enclosing balls. In *European symposium on algorithms*, pages 325–338. Springer, 1999.

[19] Martin Henk. löwner-john ellipsoids. *Documenta Math*, 95:106, 2012.

[20] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estima-

tion of texture-less 3d objects in heavily cluttered scenes. In *Computer Vision–ACCV 2012: 11th Asian Conference on Computer Vision, Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part I 11*, pages 548–562. Springer, 2013.

[21] Peter J Huber. *Robust statistics*, volume 523. John Wiley & Sons, 2004.

[22] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.

[23] IS Khalil, JC Doyle, and K Glover. *Robust and optimal control*. Prentice hall, 1996.

[24] Laurent Kneip, Hongdong Li, and Yongduek Seo. Upnp: An optimal o (n) solution to the absolute pose problem with universal applicability. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 127–142. Springer, 2014.

[25] Selcuk Korkmaz, Dinçer Göksülük, and GÖKMEN Zararsiz. Mvn: An r package for assessing multivariate normality. *R JOURNAL*, 6(2), 2014.

[26] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Automatic generator of minimal problem solvers. In *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part III 10*, pages 302–315. Springer, 2008.

[27] Jean B Lasserre. A generalization of löwner-john's ellipsoid theorem. *Mathematical Programming*, 152: 559–591, 2015.

[28] Yingying Li, Jing Yu, Lauren Conger, and Adam Wierman. Learning the uncertainty sets for control dynamics via set membership: A non-asymptotic analysis. *arXiv preprint arXiv:2309.14648*, 2023.

[29] Mario Milanese and Antonio Vicino. Optimal estimation theory for dynamic systems with set membership uncertainty: An overview. *Automatica*, 27(6):997–1009, 1991.

[30] Mohamed Mustafa, Alexandru Stancu, Nicolas Delanoue, and Eduard Codres. Guaranteed slaman interval approach. *Robotics and Autonomous Systems*, 100:160–170, 2018.

[31] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2011–2018. IEEE, 2017.

[32] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4561–4570, 2019.

[33] David M Rosen, Luca Carlone, Afonso S Bandeira, and John J Leonard. Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group. *The International Journal of Robotics Research*, 38(2-3):95–125, 2019.

[34] Aaron Roth. Uncertain: Modern topics in uncertainty estimation, 2022.

[35] Jingnan Shi, Heng Yang, and Luca Carlone. Optimal and robust category-level perception: Object pose and shape estimation from 2-d and 3-d semantic keypoints. *IEEE Transactions on Robotics*, 2023.

[36] Yang Song, Zhuqing Zhang, Jun Wu, Yue Wang, Liang Zhao, and Shoudong Huang. A right invariant extended kalman filter for object based slam. *IEEE Robotics and Automation Letters*, 7(2):1316–1323, 2021.

[37] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022.

[38] Yukai Tang, Jean-Bernard Lasserre, and Heng Yang. Uncertainty quantification of set-membership estimation in control and perception: Revisiting the minimum enclosing ellipsoid. *arXiv preprint arXiv:2311.15962*, 2023.

[39] Ye Wang, Vicenç Puig, and Gabriela Cembrano. Set-membership approach and kalman observer based on zonotopes for discrete-time descriptor systems. *Automatica*, 93:435–443, 2018.

[40] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[41] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Robotics: Science and Systems*, 2018.

[42] Miaolan Xie. Inner approximation of convex cones via primal-dual ellipsoidal norms. Master's thesis, University of Waterloo, 2016.

[43] Heng Yang and Luca Carlone. Certifiably optimal outlier-robust geometric perception: Semidefinite relaxations and scalable global optimization. *IEEE transactions on pattern analysis and machine intelligence*, 45(3):2816–2834, 2022.

[44] Heng Yang and Marco Pavone. Object pose estimation with statistical guarantees: Conformal keypoint detection and geometric uncertainty propagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8947–8958, 2023.

[45] Heng Yang, Jingnan Shi, and Luca Carlone. Teaser: Fast and certifiable point cloud registration. *IEEE Transactions on Robotics*, 37(2):314–333, 2020.

[46] Heng Yang, Wei Dong, Luca Carlone, and Vladlen Koltun. Self-supervised geometric perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14350–14361, 2021.

[47] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and T Funkhouser. 3dmatch: Learning the matching of local 3d geometry in range scans. volume 1, page 4, 2017.

[48] Hongyi Zhang and Suvrit Sra. First-order methods for geodesically convex optimization. In *Conference on Learning Theory*, pages 1617–1638. PMLR, 2016.