FAST CONVERGENCE METHOD FOR GLOBAL OPTIMAL 4DOF REGISTRATION

Jiro Abe[a,*], Akira Tsuji[a], Junichi Abe[a]

[a] NEC Corporation, 1753 Shimonumabe, Nakahara-ku, Kawasaki, Japan
abe.j@nec.com

WG II/3

KEY WORDS: Registration, LiDAR, Global optimization, Cylindrical norm, Rectangle intersection problem, Branch and bound.

ABSTRACT:

Four degrees of freedom (4DoF) registration is a class of point cloud registration problems for finding a rigid transformation to align two point clouds under the constraint that the rigid transformation is composed of a three-dimensional (3D) translation and 1D rotation. This constraint is suitable to align scan pairs acquired using modern terrestrial Light Detection and Ranging (LiDAR) scanners, the scans of which can share the direction of gravity as the Z-axis due to such scanners using tripods or internal inclinometers. We propose a fast convergence method for global optimal 4DoF registration. The proposed method consists of (i) our newly developed 4DoF registration model formulated as an optimization problem involving the *cylindrical norm* to measure the distance between two points, and (ii) a fast convergence algorithm to find a global optimal solution of the model. We experimentally demonstrated that the proposed method reduced the number of iterations to convergence and computation time compared with a current 4DoF registration method, especially when the given scan pairs are similar but cannot be aligned, which often appears in registration of multiple point clouds.

1. INTRODUCTION

Terrestrial LiDAR scanners to reproduce the real world in a 3D digital space as-is has become widely used in many engineering fields, such as building information modeling (BIM), construction information modeling (CIM), facility management, and the equipment delivery planning. To reproduce a 3D model of an entire target architecture, multiple scans acquired at different scan points must be aligned because every scan captures only a part of the architecture visible from a scan point. Therefore, point cloud registration, which is a technique to align multiple LiDAR scans, is necessary.

This paper focuses on pairwise registration methods for aligning two point clouds by applying a rigid transformation to one side because many point cloud registration methods for multiple point clouds has been developed on the basis of the execution of pairwise registration (Theiler et al., 2015). When using pairwise registration as a sub-process of registration for multiple scans, it is not guaranteed that the two input point clouds can be aligned. Therefore, pairwise registration should be achieved quickly even when input scan pairs cannot be aligned.

One of the most common pairwise registration methods is the iterative closest point (ICP) method (Besl and McKay, 1992), which achieves highly accurate registration. However, it finds a locally optimal solution, which requires a good initial solution to obtain a correct result. Methods for finding a good initial solution to make such a method accurate are called coarse registration methods.

We propose a fast convergence method with the following three main features:

- *global optimality*: The proposed method consists of a 4DoF registration model written as an optimization problem (see Problem 1) and its global optimization algorithm (see Algorithm 1). The global optimality guarantees that the proposed method gives the best transformation to achieve the highest objective value regardless of the initial pose of input point clouds.
- *4DoF registration*: This concept was originally introduced by Cai *et al.* into global optimal registration (Cai et al., 2019). Since modern terrestrial LiDARs are equipped with tripods or internal inclinometers, registration methods only needs to search for 3D translation and 1D rotation, not 3D rotation. This concept enables global optimal registration within a practical computation time. Problem 1 is our newly developed 4DoF registration model, which involves the *cylindrical norm* (see Figure 1) instead of the standard Euclidean norm.
- *faster convergence*: Compared with the current global optimal 4DoF registration method (Cai et al., 2019), called branch-and-bound algorithm with fast match pruning (FMP+BnB), the proposed method converges faster. As shown later, the proposed method enables faster convergence, especially when input scan pairs cannot be aligned.

The cylindrical norm is the key factor to achieve the faster convergence property. By adapting the cylindrical norm, as shown in Figure 3, two of the four parameters in 4DoF registration can be simultaneously optimized via the computation geometry problem which searches the most overlapped region of multiple rectangles and can be solved in polynomial time. We demonstrate the computation efficiency of the proposed method in experiments.

1.1 Related work

Researchers have proposed various methods for coarse registration. However, with many of these methods, the possibility of successful registration depends on the random sample

* Corresponding author

consensus (RANSAC) strategy (Chen et al., 1999) or appropriate parameter tunings for every input point cloud pair on the basis of their overwrap ratio (Aiger et al., 2008). These features are inconvenient for practical use, such as difficulty for users to identify whether the success factors of registration is in the LiDAR locations or algorithm settings. Certain studies used specific structures in a point cloud, such as lines (Jaw and Chuang, 2008) or ground planes (Li et al., 2021). However, dependence on such structure are inconvenient due to their limited applicability.

To avoid these inconveniences, certain coarse registration methods, e.g., (Yang et al., 2016), have been developed on the basis of global optimization techniques, which can clearly describe the results as an exact solution of a certain objective function for evaluating the quality of rigid transformations. In the context of global optimal registration, it has been challenging to exactly solve optimization problems within a practical computation time.

To tackle this challenge, (Cai et al., 2019) introduced the concept of 4DoF registration and proposed the global optimal registration method, called FMP+BnB, which enables coarse registration for large real-world point clouds within a practical computation time. However, from our observations discussed in Section 4.2, FMP+BnB often consumes a large amount of time, especially when the input point clouds are similar but cannot be aligned, which often appears when using pairwise registration as a sub-process of registration for multiple point clouds.

1.2 Notation

Let $\mathbb{R}$, $\mathbb{R}_+$, and $\mathbb{R}_{++}$ be the sets of real numbers, non-negative real numbers, and positive real numbers, respectively. For $\alpha \in \mathbb{R}$, $|\alpha| \in \mathbb{R}_+$ denotes the absolute value of α. For $\beta \in \mathbb{R}_{++}$, we use the indicator function

$$\iota_\beta \colon \mathbb{R}_+ \to \{0,1\} \colon x \mapsto \begin{cases} 1, & (x \le \beta), \\ 0, & (\text{otherwise}). \end{cases} \tag{1}$$

Bold lowercase letters express vectors, and the superscript $(\cdot)^\top$ denotes transpose. For a vector $\boldsymbol{x} := [x_1, x_2, \cdots, x_n]^\top \in \mathbb{R}^n$, we use the Euclidean norm $\|\boldsymbol{x}\|_2 := (\sum_{i=1}^n |x_i|^2)^{1/2}$. For a 3D point $\boldsymbol{p} := [p_x, p_y, p_z]^\top \in \mathbb{R}^3$, $[\boldsymbol{p}]_{\mathrm{xy}} := [p_x, p_y]^\top \in \mathbb{R}^2$ and $[\boldsymbol{p}]_{\mathrm{z}} := p_z \in \mathbb{R}$ represent for the horizontal and vertical components, respectively. For a finite set $\mathcal{S}$, $|\mathcal{S}|$ denotes the number of elements of $\mathcal{S}$.

2. CYLINDRICAL-NORM BASED 4DOF REGISTRATION MODEL

For two given point clouds, i.e., the source point cloud $\mathcal{P} \subset \mathbb{R}^3$ and target point cloud $\mathcal{Q} \subset \mathbb{R}^3$, the goal with 4DoF registration is to find a rigid transformation $f \colon \mathbb{R}^3 \to \mathbb{R}^3$ which aligns $\mathcal{P}$ to $\mathcal{Q}$, under the constraint that f is formulated as

$$f(\boldsymbol{p}; \theta, \boldsymbol{t}) := R(\theta)\boldsymbol{p} + \boldsymbol{t}, \tag{2}$$

$$R(\theta) := \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3}, \tag{3}$$

with a rotation angle $\theta \in [0, 2\pi)$ and translation vector $\boldsymbol{t} \in \mathbb{R}^3$.

To achieve this goal, we first generate 3D keypoint matches $\mathcal{C} := (\boldsymbol{p}_i, \boldsymbol{q}_i)_{i\in\mathcal{I}}$ between $\mathcal{P}$ and $\mathcal{Q}$, where $\mathcal{I}$ is the finite index set. Note that using $\mathcal{C}$ for registration is common, and we follow the standard procedure for generating such matches (see Section 4 for the procedure used in our experiments).

In general, the $\mathcal{C}$ contain many false correspondences for reasons such as the two given point clouds only partially overwrap. Therefore, we designed our 4DoF registration model in a manner similar to robust estimation techniques. Concretely, the model is written as a problem to find an f in (2) which maximizes the number of correspondences in $\mathcal{C}$ closer than a certain constant distance, called an *inlier threshold*. Problem 1 formulates the model.

Problem 1 (Our 4DoF registration model)**.** Let $R(\theta) \in \mathbb{R}^{3\times 3}$ for $\theta \in [0, 2\pi)$ be defined in (3). For given source point cloud $\mathcal{P} \subset \mathbb{R}^3$, target point cloud $\mathcal{Q} \subset \mathbb{R}^3$, $\mathcal{C} := (\boldsymbol{p}_i, \boldsymbol{q}_i)_{i\in\mathcal{I}}$ between $\mathcal{P}$ and $\mathcal{Q}$ with a finite index set $\mathcal{I}$, and inlier threshold $(\epsilon_1, \epsilon_2) \in \mathbb{R}_{++} \times \mathbb{R}_{++}$,

$$\text{find } (\theta^\star, \boldsymbol{t}^\star) \in \operatorname*{arg\,max}_{(\theta,\boldsymbol{t})\in[0,2\pi)\times\mathbb{R}^3} E(\theta, \boldsymbol{t}; \mathcal{I}), \tag{4}$$

$$E(\theta, \boldsymbol{t}; \mathcal{I}) := \sum_{i\in\mathcal{I}} \iota_1 \left(\|R(\theta)\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_i\|_{(\epsilon_1,\epsilon_2)} \right), \tag{5}$$

where the norm $\|\cdot\|_{(\epsilon_1,\epsilon_2)} \colon \mathbb{R}^3 \to \mathbb{R}_+$ is defined as

$$\|\boldsymbol{v}\|_{(\epsilon_1,\epsilon_2)} := \max\left(\frac{\|[\boldsymbol{v}]_{\mathrm{xy}}\|_2}{\epsilon_1}, \frac{|[\boldsymbol{v}]_{\mathrm{z}}|}{\epsilon_2} \right). \tag{6}$$

We call the norm defined in (6) in Problem 1 *cylindrical norm* because of the shape of its unit ball, i.e., the set $\{\boldsymbol{v} \in \mathbb{R}^3 \mid \|\boldsymbol{v}\|_{(\epsilon_1,\epsilon_2)} \le 1\}$. Figure 1 illustrates the unit ball of the cylindrical norm in (6). The unit ball forms a cylinder with the base radius of ϵ_1 and height of $2\epsilon_2$, centered at the origin and having the Z-axis as its vertical axis.

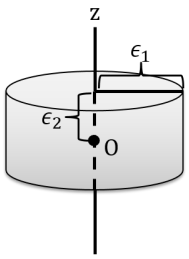


Figure 1. Unit ball of cylindrical norm $\|\cdot\|_{(\epsilon_1,\epsilon_2)}$ in (6)

Note that our model in Problem 1 is different from the current 4DoF registration model used with FMP+BnB (Cai et al., 2019) in terms of the norm to measure the distance between matched keypoints. While the current model use the standard Euclidean norm, our model uses the cylindrical norm.

To show the benefit of the cylindrical norm, we rewrite the objective function E in Problem 1 in accordance with the definition of the cylindrical norm in (6) as follows:

$$E(\theta, \boldsymbol{t}; \mathcal{I}) = \sum_{i\in\mathcal{I}} I_{\mathrm{xy}}(\theta, \boldsymbol{t}) I_{\mathrm{z}}(\boldsymbol{t}), \tag{7}$$

$$I_{\mathrm{xy}}(\theta, \boldsymbol{t}) := \iota_{\epsilon_1} \left(\|[R(\theta)\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_i]_{\mathrm{xy}}\|_2 \right) \in \{0,1\}, \tag{8}$$

$$I_{\mathrm{z}}(\boldsymbol{t}) := \iota_{\epsilon_2} \left(|[\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_i]_{\mathrm{z}}| \right) \in \{0,1\}. \tag{9}$$

Intuitively, (7)–(9) indicate that E in Problem 1 counts the number of correspondences in $\mathcal{C}$ that satisfies the following two conditions:

(i) [Horizontal condition]: The horizontal distance between $R(\theta)\boldsymbol{p}_i + \boldsymbol{t}$ and $\boldsymbol{q}_i$ is up to ϵ_1.

(ii) [Vertical condition]: The vertical distance between $\boldsymbol{p}_i + \boldsymbol{t}$ and $\boldsymbol{q}_i$ is up to ϵ_2.

The vertical condition does not depend on the rotation angle θ because the rotation matrix $R(\theta)$ in (3) has no effect on the Z-coordinate of a point. The horizontal condition does not depend on $[\boldsymbol{t}]_{\mathrm{z}}$ which is the translation along the Z-axis. In other words, θ and $[\boldsymbol{t}]_{\mathrm{z}}$ affect different conditions. We emphasize that θ and $[\boldsymbol{t}]_{\mathrm{z}}$ play an important role in achieving fast convergence with the proposed method. As shown later in Section 3.3, θ and $[\boldsymbol{t}]_{\mathrm{z}}$ can be efficiently optimized simultaneously for a fixed $[\boldsymbol{t}]_{\mathrm{xy}}$.

In Section 3, we describe finding a global optimal solution $(\theta^\star, \boldsymbol{t}^\star)$ with our 4DoF registration model.

3. GLOBAL OPTIMIZATION ALGORITHM

Algorithm 1 shows the algorithm for finding a global optimal solution to Problem 1. The algorithm consists of the following two steps:

(i) [Pruning]: To lighten the burden of the subsequent optimization step, this step reduces given $\mathcal{I}$ to a smaller subset $\mathcal{I}'$ under the constraint that $\mathcal{I}'$ preserves the global optimal solution $(\theta^\star, \boldsymbol{t}^\star)$ of Problem 1. See Section 3.1 for details of this pruning step.

(ii) [Optimization]: This step searches for a global optimal solution $(\theta^\star, \boldsymbol{t}^\star)$ of Problem 1 on $\mathcal{I}'$ by using a custom branch-and-bound (BnB) algorithm. See Sections 3.2–3.3 for details of this optimization step.

Algorithm 1 for Problem 1.
Prune $\mathcal{I}$ into a smaller subset $\mathcal{I}'$ (Algorithm 2).
Find a solution $(\theta^\star, \boldsymbol{t}^\star)$ of Problem 1 on $\mathcal{I}'$ (Algorithm 3).
return $(\theta^\star, \boldsymbol{t}^\star)$.

3.1 Prune indices

The pruning step is a preprocess to reduce $\mathcal{I}$ in Problem 1 to $\mathcal{I}'$ such that a global optimal solution is preserved in $\mathcal{I}'$, i.e.,

$$\operatorname*{arg\,max}_{(\theta,\boldsymbol{t})\in[0,2\pi)\times\mathbb{R}^3} E(\theta,\boldsymbol{t};\mathcal{I}') = \operatorname*{arg\,max}_{(\theta,\boldsymbol{t})\in[0,2\pi)\times\mathbb{R}^3} E(\theta,\boldsymbol{t};\mathcal{I}). \quad (10)$$

Proposition 1 states a sufficient condition for global optimal solutions to be preserved when an index $k \in \mathcal{I}$ is removed from $\mathcal{I}$.

Proposition 1 (A sufficient condition for removable index)**.** *In Problem 1, let $k \in \mathcal{I}$ and define $(\boldsymbol{p}_i^{(k)}, \boldsymbol{q}_i^{(k)}) := (\boldsymbol{p}_i - \boldsymbol{p}_k, \boldsymbol{q}_i - \boldsymbol{q}_k)$ for every $i \in \mathcal{I}$. Suppose that there exists $(\theta_0, \boldsymbol{t}_0) \in [0, 2\pi) \times \mathbb{R}^3$ satisfying*

$$\bar{E}_k < E(\theta_0, \boldsymbol{t}_0; \mathcal{I}), \quad (11)$$

where

$$\bar{E}_k := \max_{\theta\in[0,2\pi)} \sum_{i\in\mathcal{I}_k} \iota_{2\epsilon_1}\left(\|[R(\theta)\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)}]_{\mathrm{xy}}\|_2\right), \quad (12)$$

$$\mathcal{I}_k := \left\{ i \in \mathcal{I} \;\middle|\; |[\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)}]_{\mathrm{z}}| \le 2\epsilon_2 \right\}. \quad (13)$$

Then the index k is removable, i.e.,

$$\operatorname*{arg\,max}_{(\theta,\boldsymbol{t})\in[0,2\pi)\times\mathbb{R}^3} E(\theta,\boldsymbol{t};\mathcal{I}\setminus\{k\}) = \operatorname*{arg\,max}_{(\theta,\boldsymbol{t})\in[0,2\pi)\times\mathbb{R}^3} E(\theta,\boldsymbol{t};\mathcal{I}). \quad (14)$$

Proof. See Appendix A. □

In accordance with Proposition 2, we can find removable indices in $\mathcal{I}$ and generate $\mathcal{I}'$. The $\bar{E}_k$ in (12) is calculated by solving an optimization problem for θ, called the *max-stabbing problem* (Cai et al., 2019), which can be solved in a computational complexity of $\mathcal{O}(|\mathcal{I}_k| \log |\mathcal{I}_k|)$.

Algorithm 2 shows the process of the pruning step.

Algorithm 2 for the pruning step.
Set $\mathcal{I}' \leftarrow \mathcal{I}$, $\underline{E} \leftarrow 0$.
For $k \in \mathcal{I}$ **do**
  Compute $\mathcal{I}_k$ in (13) on $\mathcal{I}'$.
  Compute $\overline{E}_k$ in (12) and obtain corresponding θ.
  If $\overline{E}_k < \underline{E}$ **then** $\mathcal{I}' \leftarrow \mathcal{I}' \setminus \{k\}$.
  else
    Set $(\theta_0, \boldsymbol{t}_0) \leftarrow (\theta, R(\theta)\boldsymbol{p}_k + \boldsymbol{q}_k)$.
    Update $\underline{E} \leftarrow E(\theta_0, \boldsymbol{t}_0; \mathcal{I}')$.
Remove indices $k \in \mathcal{I}'$ satisfying $\overline{E}_k < \underline{E}$.
return $\mathcal{I}'$.

3.2 Overview of optimization step

In Sections 3.2–3.4, we describe the optimization algorithm to solve Problem 1. This algorithm is directly applicable to the optimization step by replacing $\mathcal{I}$ in Problem 1 with the pruned $\mathcal{I}'$ computed in Section 3.1.

To derive an optimization algorithm for Problem 1, we first rewrite the optimization problem in (4) as follows:

$$\operatorname*{maximize}_{[\boldsymbol{t}]_{\mathrm{xy}}\in\mathbb{R}^2} U([\boldsymbol{t}]_{\mathrm{xy}}; \epsilon_1, \epsilon_2, \mathcal{I}), \quad (15)$$

$$U([\boldsymbol{t}]_{\mathrm{xy}}; \epsilon_1, \epsilon_2, \mathcal{I}) := \max_{(\theta,[\boldsymbol{t}]_{\mathrm{z}})\in[0,2\pi)\times\mathbb{R}} E(\theta, \boldsymbol{t}; \mathcal{I}). \quad (16)$$

The purpose of the translation to (15)–(16) is as follows:

- As we see in Section 3.3, we can efficiently solve the *inner* optimization problem in (16), i.e., the problem simultaneously optimizing θ and $[\boldsymbol{t}]_{\mathrm{z}}$ for a fixed $[\boldsymbol{t}]_{\mathrm{xy}}$, by computation complexity of $\mathcal{O}(|\mathcal{I}| \log |\mathcal{I}|)$.
- As we see in Section 3.4, we can find a global optimal solution $[\boldsymbol{t}]_{\mathrm{z}}$ of the *outer* optimization problem in (15) by using a custom BnB algorithm which searches for the solution on a 2D space.

Note that the optimization algorithm of the proposed method is designed in a way similar to the algorithm of FMP+BnB (Cai et al., 2019) which also solves their 4DoF registration model by decomposing into an *inner* and *outer* optimization problems. Table 1 shows a comparison between FMP+BnB's algorithm and our algorithm in terms of computational cost. The *inner* optimization problem can be solved in $\mathcal{O}(|\mathcal{I}| \log \mathcal{I})$ with both FMP+BnB' and our algorithms. However, the BnB algorithm for the *outer* optimization searches on only a 2D space at our algorithm, while it searches on a 3D space at the algorithm of FMP+BnB. This difference suggests that our algorithm can converge faster than that of FMP+BnB.

	Algorithm of FMP+BnB	Ours
Computational complexity of *inner* optimization	$\mathcal{O}(\lvert\mathcal{I}\rvert \log \lvert\mathcal{I}\rvert)$	$\mathcal{O}(\lvert\mathcal{I}\rvert \log \lvert\mathcal{I}\rvert)$
Search space of *outer* optimization	$\mathbb{R}^3$	$\mathbb{R}^2$

Table 1. Computation-cost comparison between algorithm of FMP+BnB's and our algorithm

3.3 Evaluate U in (16) for a fixed $[\boldsymbol{t}]_{\mathrm{xy}}$

In accordance with the definition of E in (5), (16) can be re-written as

$$U([\boldsymbol{t}]_{\mathrm{xy}};\epsilon_1,\epsilon_2,\mathcal{I}) = \max_{\substack{\theta\in[0,2\pi),\\ [\boldsymbol{t}]_{\mathrm{z}}\in\mathbb{R}}} \sum_{i\in\mathcal{I}} F_i(\theta,[\boldsymbol{t}]_{\mathrm{z}};[\boldsymbol{t}]_{\mathrm{xy}}), \tag{17}$$

$$F_i(\theta,[\boldsymbol{t}]_{\mathrm{z}};[\boldsymbol{t}]_{\mathrm{xy}}) := \iota_1\left(\|\tilde{\boldsymbol{p}}_i(\theta,[\boldsymbol{t}]_{\mathrm{z}}) - \tilde{\boldsymbol{q}}_i\|_{(\epsilon_1,\epsilon_2)}\right), \tag{18}$$

$$\tilde{\boldsymbol{p}}_i(\theta,[\boldsymbol{t}]_{\mathrm{z}}) := R(\theta)\boldsymbol{p}_i + [0,0,[\boldsymbol{t}]_{\mathrm{z}}]^\top \in \mathbb{R}^3, \tag{19}$$

$$\tilde{\boldsymbol{q}}_i := \boldsymbol{q}_i - [[\boldsymbol{t}]_{\mathrm{xy}}^\top, 0] \in \mathbb{R}^3. \tag{20}$$

Figure 2(a) illustrates the region of θ and $[\boldsymbol{t}]_{\mathrm{z}}$, which satisfy $F(\theta,[\boldsymbol{t}]_{\mathrm{z}};[\boldsymbol{t}]_{\mathrm{xy}}) = 1$ in (18), as thin red lines.

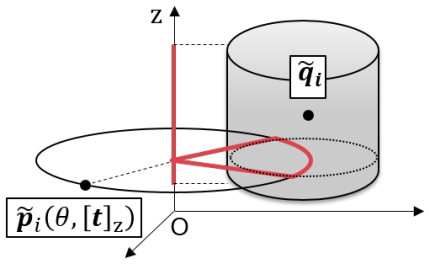
From (18)–(20) and the definition of the cylindrical norm in (6), the following condition can be derived:

$$F_i(\theta,[\boldsymbol{t}]_{\mathrm{z}};[\boldsymbol{t}]_{\mathrm{xy}}) = 1 \Leftrightarrow (\theta,[\boldsymbol{t}]_{\mathrm{z}}) \in \Theta_i \times \mathcal{T}_i, \tag{21}$$

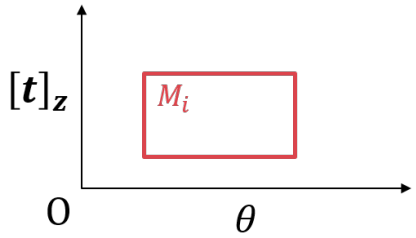
$$\Theta_i := \{\phi \in [0,2\pi) \mid \|[R(\phi)\boldsymbol{p}_i - \tilde{\boldsymbol{q}}_i]_{\mathrm{xy}}\|_2 \le \epsilon_1\}, \tag{22}$$

$$\mathcal{T}_i := \{t_z \in \mathbb{R} \mid |[\boldsymbol{p}_i]_{\mathrm{z}} + t_z - [\boldsymbol{q}_i]_{\mathrm{z}}| \le \epsilon_2\}. \tag{23}$$

The Θ_i in (22) consists of at most two intervals of $[0,2\pi)$, and the $\mathcal{T}_i$ in (23) consists of an interval of $\mathbb{R}$. Therefore, the condition in (21) shows that, as shown in Figure 2(b), the region of the parameter $(\theta,[\boldsymbol{t}]_{\mathrm{z}})$ satisfying $F(\theta,[\boldsymbol{t}]_{\mathrm{z}};[\boldsymbol{t}]_{\mathrm{xy}}) = 1$ forms (at most two) rectangles $\mathcal{M}_i$ in the parameters space.



(a)



(b)

Figure 2. (a) Region $(\theta,[\boldsymbol{t}]_{\mathrm{z}})$ satisfying $F_i(\theta,[\boldsymbol{t}]_{\mathrm{z}};[\boldsymbol{t}]_{\mathrm{xy}}) = 1$ can be reviewed as (b) rectangle regions on 2D space.

As shown in Figure 3, the optimization problem for $(\theta,[\boldsymbol{t}]_{\mathrm{z}})$ in (17) can be translated into a computation geometry problem called the *rectangle-intersection problem* (Choi et al., 2012), which searches a point in the region where the given multiple rectangles overlap the most. The rectangle-intersection problem can be solved in $\mathcal{O}(N\log N)$ for given N rectangles (Imai and Asano, 1983). In our problem, the number of rectangles is at most $2|\mathcal{I}|$. See Figure 3 and Appendix B for our version of the algorithm to solve the rectangle-intersection problem.

We can now find an optimal solution $(\theta,[\boldsymbol{t}]_{\mathrm{z}})$ of the optimization problem in (16) and evaluate $U([\boldsymbol{t}]_{\mathrm{xy}};\epsilon_1,\epsilon_2,\mathcal{I})$ for a fixed $[\boldsymbol{t}]_{\mathrm{xy}}$ with $\mathcal{O}(|\mathcal{I}|\log|\mathcal{I}|)$.

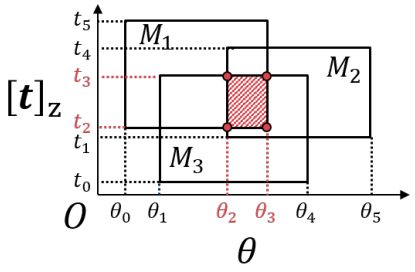


Figure 3. Optimal parameter $(\theta,[\boldsymbol{t}]_{\mathrm{z}})$, which is on most overlapped region for multiple rectangles (slashed-red region), can be found as pair of (θ_i,t_j), where θ_i and t_j are a vertex of rectangles.

3.4 Optimize U in (15) by using BnB algorithm

We now discuss to finding a global solution of an optimization problem for $[\boldsymbol{t}]_{\mathrm{xy}}$ in (15) by using a custom BnB algorithm that uses the computation of U in (16) with a fixed $[\boldsymbol{t}]_{\mathrm{xy}}$ mentioned in Section 3.3. The BnB algorithm finds a global optimal solution of non-convex problems, which splits the original maximization problem into multiple smaller sub-problems (*branches*) then searches only branches with an upper bound exceeding the highest objective value found with the algorithm. The BnB iterations stop when all upper bounds of branches that have not yet been explored are less than or equal to the current highest objective value because there is no better solution in such branches.

In the context of solving (15), the custom BnB algorithm initializes the search space with a square $\mathcal{S}_0 \subset \mathbb{R}^2$ that contains the optimal solution $[\boldsymbol{t}^\star]_{\mathrm{xy}}$, then splits S_0 into four smaller congruent squares with edges parallel to the X-axis or Y-axis. For each smaller square $\mathcal{S} \subset \mathcal{S}_0$, let $[\boldsymbol{t}_{\mathcal{S}}]_{\mathrm{xy}} \in \mathbb{R}^2$ be the center point of $\mathcal{S}$. If $[\boldsymbol{t}_{\mathcal{S}}]_{\mathrm{xy}}$ gives a higher objective value than the current best estimate $[\hat{\boldsymbol{t}}]_{\mathrm{xy}}$, $[\hat{\boldsymbol{t}}]_{\mathrm{xy}}$ is updated to $[\boldsymbol{t}_{\mathcal{S}}]_{\mathrm{xy}}$. Divide $\mathcal{S}$ into the four smaller congruent squares then, as we describe later, compute an upper bound $\overline{U}$ of each square and keep squares with $\overline{U}$ exceeding the current highest objective value $U([\hat{\boldsymbol{t}}]_{\mathrm{xy}})$ as branches, which will be explored later in the custom BnB algorithm.

Algorithm 3 shows the custom BnB algorithm described above. Note that while Algorithm 3 appears to find only an optimal horizontal translation $[\boldsymbol{t}^\star]_{\mathrm{xy}}$ via the optimization problem in (15), it simultaneously finds the optimal residual parameters $(\theta^\star,[\boldsymbol{t}^\star]_{\mathrm{z}})$ in Problem 1. Once Algorithm 3 gives an optimal solution $[\boldsymbol{t}^\star]_{\mathrm{xy}}$, the residual parameters $(\theta^\star,[\boldsymbol{t}^\star]_{\mathrm{z}})$ in Problem 1 can be obtained by evaluating U for fixed $[\boldsymbol{t}^\star]_{\mathrm{xy}}$.

To run Algorithm 3, we need to compute $\overline{U}(\mathcal{S})$ which is an upper bound of U on a given $\mathcal{S} \subset \mathbb{R}^2$. Proposition 2 shows the computation of $\overline{U}(\mathcal{S})$ such an upper bound.

Proposition 2 (An upper bound of U)**.** *In Problem 1, define U as* (16)*. Let $\mathcal{S} \subset \mathbb{R}^2$ be a bounded square, $\boldsymbol{s}_0$ be the center of $\mathcal{S}$, and $d_{\mathcal{S}}$ be the radius of $\mathcal{S}$. Then,*

$$\overline{U}(\mathcal{S}) := U(\boldsymbol{s}_0;\epsilon_1 + d_{\mathcal{S}},\epsilon_2,\mathcal{I}) \ge \max_{\boldsymbol{s}\in\mathcal{S}} U(\boldsymbol{s};\epsilon_1,\epsilon_2,\mathcal{I}). \tag{24}$$

Proof. See Appendix C. □

The inequality in (24) shows that $\overline{U}(\mathcal{S})$ is an upper bound of U on a $\mathcal{S}$. $\overline{U}(\mathcal{S})$ can be computed by evaluating U for fixed $\boldsymbol{s}_0$.

Algorithm 3 for solving (15).

Set initial matches $\mathcal{C}$ and inlier threshold (ϵ_1, ϵ_2).
Set $\mathcal{S}_0$, $[\hat{\boldsymbol{t}}]_{\mathrm{xy}} \leftarrow [\boldsymbol{t}_{\mathcal{S}_0}]_{\mathrm{xy}}$ and an empty priority queue w.
Compute $\overline{U}(\mathcal{S}_0; \epsilon_1, \epsilon_2, \mathcal{I})$.
Insert $(\mathcal{S}_0, \overline{U}(\mathcal{S}_0; \epsilon_1, \epsilon_2, \mathcal{I}))$ into w.
While w is not empty
  Pull the cube $\mathcal{S}$ with the highest $\overline{U}$ from w.
  Compute $U([\boldsymbol{t}_{\mathcal{S}}]_{\mathrm{xy}})$.
  If $U([\boldsymbol{t}_{\mathcal{S}}]_{\mathrm{xy}}) = \overline{U}(\mathcal{S}; \epsilon_1, \epsilon_2, \mathcal{I})$ **then break**.
  If $U([\boldsymbol{t}_{\mathcal{S}}]_{\mathrm{xy}}) > U([\hat{\boldsymbol{t}}]_{\mathrm{xy}})$ **then** $[\hat{\boldsymbol{t}}]_{\mathrm{xy}} \leftarrow [\boldsymbol{t}_{\mathcal{S}}]_{\mathrm{xy}}$.
  Divide $\mathcal{S}$ into 4 sub-cubes $\{\mathcal{S}_k\}_{k=1}^4$.
  Compute $\overline{U}(\mathcal{S}_k)$ for all $\mathcal{S}_k$.
  If $\overline{U}(\mathcal{S}_k) > U([\hat{\boldsymbol{t}}]_{\mathrm{xy}})$ **then** insert $(\mathcal{S}_k, \overline{U}(\mathcal{S}_k))$ into w.
return $[\hat{\boldsymbol{t}}]_{\mathrm{xy}}$.

4. EXPERIMENTS

We conducted experiments to demonstrate (i) the applicability of the proposed method to coarse registration (Section 4.1) and (ii) its computation efficiency (Section 4.2) by using the real-world datasets *Arch* and *Trees*[1]. The *Arch* dataset consists of five scans acquired with low overlap (30–40%), and the *Trees* dataset consists of six scans acquired in a forest with a large amount of underwood.

To demonstrate registration performance in a scenario in which point cloud pairs can and cannot be aligned, we prepared *regular* and *irregular* pairs of both *Arch* and *Trees* datasets.

- regular pairs: point cloud pairs that are assumed can be aligned with the dataset. We list all the regular pairs in Table 2 for the *Arch* dataset and in Table 3 for the *Trees* dataset.
- irregular pairs: all pairs of different point clouds. The X and Y coordinates were swapped in the source point cloud for all irregular pairs. Since the swapping is not a rigid transformation, all irregular pairs cannot be aligned.

The matches $\mathcal{C}$ in Problem 1 was generated by the following procedure:

1. The voxel grid down-sampling at 10cm in length and the intrinsic shape signatures (ISS) keypoint extraction (Zhong, 2009).
2. The fast point feature histograms (FPFH) feature (Rusu et al., 2009) computation. Matching keypoints on the basis of the distance between their FPFH features. Concretely, the correspondence $(\boldsymbol{p}_i, \boldsymbol{q}_i)$ is included in $\mathcal{C}$ if their FPFH features are one of the ten nearest neighbors to each other.

The inlier threshold (ϵ_1, ϵ_2) in Problem 1 was set to $(0.4, 0.4)$. All experiments were implemented in C++ and executed on an Intel Xeon 3.20GHz CPU.

4.1 Validity of proposed method for coarse registration

We first evaluated the applicability of the proposed method to coarse registration by using the regular pairs in the *Arch* and *Trees* datasets.

For the *Arch* dataset, Table 2 represents the registration accuracy of the proposed method without and with the ICP method, which is a well-known fine registration method. The registration performance was measured on the basis of the rotation error and translation error formulated as

$$E_{\mathrm{rot}} := \|R(\theta^\star) - R_{\mathrm{gt}}\|_{\mathrm{F}}, \quad E_{\mathrm{trans}} := \|\boldsymbol{t}^\star - \boldsymbol{t}_{\mathrm{gt}}\|_2 \qquad (25)$$

where $\|\cdot\|_{\mathrm{F}}$ means the Frobenius norm and $(R_{\mathrm{gt}}, \boldsymbol{t}_{\mathrm{gt}}) \in \mathbb{R}^{3\times3} \times \mathbb{R}^3$ is the ground truth of the rotation matrix and translation vector given by the dataset. The validity of the proposed method was confirmed by determining whether the result satisfies the following condition; both the rotation and translation errors decreased by using the ICP method and the translation error was sufficiently small (≤ 5 cm) after using this method, which means that the proposed method gave a good initial solution for fine registration. As shown in Table 2, the proposed method gave an initial solution for all regular pairs of the *Arch* dataset, hence it is suitable for coarse registration.

pair	proposed		with ICP		success
	E_{rot} ($\times 10^{-3}$)	E_{trans} (cm)	E_{rot} ($\times 10^{-3}$)	E_{trans} (cm)	
s1-s2	5.3	9.9	1.6	3.6	✓
s3-s1	5.2	24.2	1.2	0.7	✓
s3-s2	7.4	21.6	2.1	1.9	✓
s3-s4	5.7	10.5	0.3	0.6	✓
s2-s4	9.4	17.1	1.1	1.4	✓
s2-s5	7.1	16.3	0.9	2.2	✓
s4-s5	3.9	11.5	0.3	2.2	✓
s5-s1	2.3	25.3	0.8	3.6	✓

Table 2. Accuracy for *Arch* dataset

Figure 4 shows the results of the proposed method for the point cloud pair *s5-s1* in the *Arch* dataset. Note that pair *s5-s1* is the worst case in terms of the translation error in Table 2. As shown in Figure 4, the source and target point clouds were aligned successfully even if they scanned different planes on the arch.

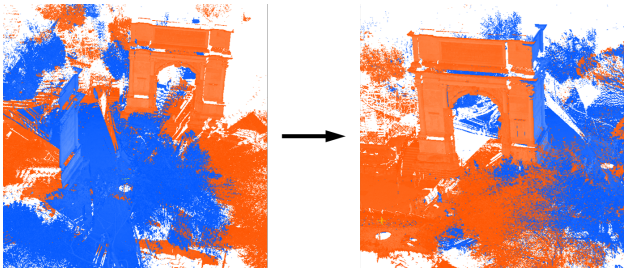


Figure 4. Registration results for point cloud pair *s5-s1* of *Arch* dataset

We evaluated the results for the *Trees* dataset under the same conditions as the *Arch* dataset. Table 3 shows the accuracy of the proposed method for the regular pairs of the *Trees* dataset. The proposed method was also suitable for coarse registration for all pairs in the *Trees* dataset.

Figure 5 shows the results of the proposed method for the point cloud pair *s2-s5*, which is the worst case in terms of the translation error in Table 3, in the *Trees* dataset.

[1] `http://www.prs.igp.ethz.ch/research/completed_projects/automatic_registration_of_point_clouds.html`

pair	proposed		with ICP		success
	E_{rot} ($\times 10^{-3}$)	E_{trans} (cm)	E_{rot} ($\times 10^{-3}$)	E_{trans} (cm)	
s1-s2	11.8	11.4	2.3	0.6	✓
s1-s3	7.1	18.2	3.4	0.7	✓
s2-s3	5.8	8.0	0.9	0.3	✓
s2-s4	5.7	29.1	1.1	1.1	✓
s2-s5	6.9	37.3	0.8	1.4	✓
s3-s4	0.6	26.1	0.3	1.2	✓
s3-s5	4.6	16.1	0.8	1.7	✓
s4-s5	8.5	16.9	0.9	0.6	✓
s4-s6	1.5	26.8	0.4	0.9	✓
s5-s6	5.4	18.3	1.0	1.0	✓

Table 3. Accuracy for *Trees* dataset

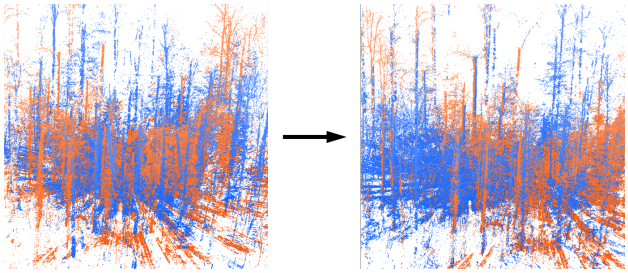


Figure 5. Registration results for point cloud pair *s2-s5* of *Trees* dataset

4.2 Computation efficiency

We now discuss the computation efficiency of the proposed method in terms of the computation time and convergence speed for finding a global optimal solution. We compared the proposed method with FMP+BnB, which also achieves global optimal 4DoF registration.

Arch dataset:
Figure 6 shows the computation times for the regular and irregular pairs in the *Arch* dataset. The proposed method reduced the worst computation time to 51% compared with FMP+BnB. The computation time of the proposed method was more stable than that of FMP+BnB, especially for the irregular pairs. Numerically, the instability of the computation time defined as

$$\text{instability} := \frac{\text{maximum time} - \text{minimum time}}{\text{minimum time}} \tag{26}$$

was calculated; instability was 3.95 for FMP+BnB and 1.82 for the proposed method.

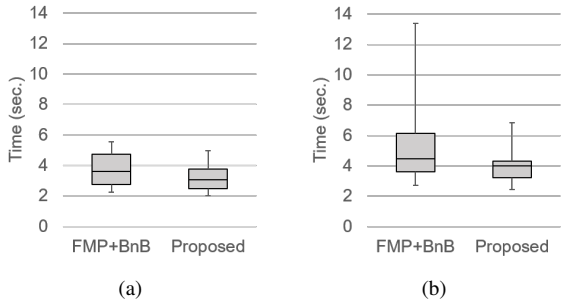


Figure 6. Computation-time comparison between FMP+BnB and proposed method (a) for regular pairs in *Arch* dataset, (b) irregular pairs in *Arch* dataset

Table 4 shows the details of the computation-time comparison. Every value in Table 4 was computed as the ratio of the computation time of the proposed method to that of FMP+BnB for a given pair. The proposed method was more efficient than FMP+BnB for all irregular pairs. The harmonic mean of all ratios was 0.77.

		target				
		s1	s2	s3	s4	s5
source	s1		.91	.93	.88	.94
	s2	.95		.90	.91	.45
	s3	.96	.95		.73	.90
	s4	.87	.86	.66		.85
	s5	.97	.32	.90	.86	

Table 4. Ratios of computation-times of proposed method to those of FMP+BnB for all irregular pairs in *Arch* dataset

Figure 7 shows the comparison of the number of iterations, i.e., the number of branches executed using the BnB algorithm, regarding convergence speed. The proposed method effectively reduced the number of iterations to 14% in the worst case. In other words, the maximum number of iterations was 7.1 times less with the proposed method than with FMP+BnB. This is the main aim of introducing the cylindrical norm and limiting the search dimension of the BnB algorithm into a 2D plane.

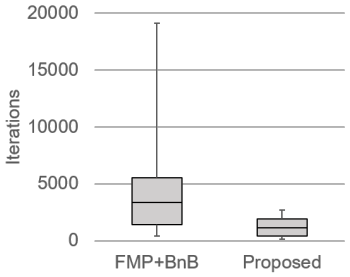


Figure 7. Number of iterations for *Arch* dataset

Trees dataset:
We also evaluated the results for the *Trees* dataset under the same conditions as the *Arch* dataset. As shown in Figure 8, the proposed method reduced the worst computation time to 51% compared with FMP+BnB. Instability was also reduced; 3.64 with FMP+BnB and 1.64 with the proposed method.

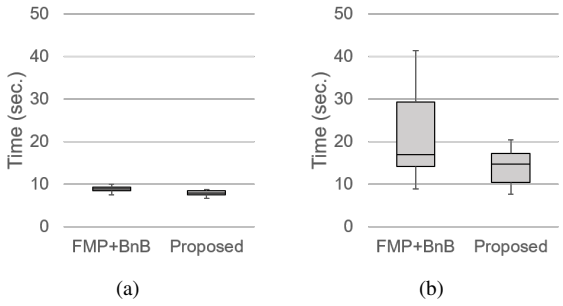


Figure 8. Computation-time comparison between FMP+BnB and proposed method (a) for regular pairs in *Trees* dataset, and (b) for irregular pairs in *Trees* dataset

Table 5 shows the computation-time ratios. The proposed method was more efficient than FMP+BnB for all pairs except *s1-s2*. The harmonic mean of all ratios was 0.68.

		target					
		s1	s2	s3	s4	s5	s6
source	s1		1.1	.90	.68	.87	.82
	s2	.99		.76	.71	.52	.64
	s3	.86	.73		.42	.84	.51
	s4	.53	.65	.42		.89	.69
	s5	.86	.52	.85	.88		.90
	s6	.72	.57	.43	.69	.86	

Table 5. Time comparison for *Trees* dataset

Figure 9 shows the comparison of the number of iterations regarding convergence speed. The proposed method reduced the number of iterations to 15% in the worst case. In other words, the maximum number of iterations was 6.7 times less than with FMP+BnB.

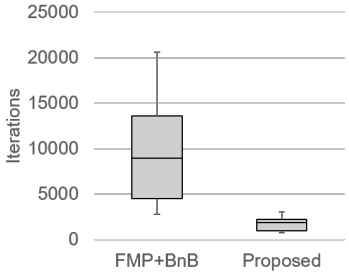


Figure 9. Number of iterations for *Trees* dataset

5. CONCLUSIONS

We proposed a fast convergence method for global optimal 4DoF registration. The proposed method consists of our newly developed 4DoF registration model (Problem 1), which includes the cylindrical norm in (6), and its global optimization algorithm (Algorithm 1). Since two parameters $([\boldsymbol{t}]_z, \theta)$ in Problem 1 can be simultaneously optimized in polynomial time $\mathcal{O}(|\mathcal{I}| \log |\mathcal{I}|)$ via the rectangle intersection problem (Figure 3), the cutom BnB algorithm of the proposed algorithm searches only a 2D space (Table 1) and can converge faster than that of FMP+BnB.

We conducted experiments to demonstrate the (i) applicability of the proposed method to coarse registration and (ii) its computation efficiency by using datasets for terrestrial LiDAR point cloud registration. Compared with FMP+BnB, the proposed method was 6.7 times faster in terms of the maximum number of iterations and 2.0 times faster in terms of the worst computation time.

REFERENCES

Aiger, D., Mitra, N. J., Cohen-Or, D., 2008. 4-points Congruent Sets for Robust Surface Registration. *ACM Transactions on Graphics*, 27(3), 85.

Besl, P. J., McKay, N. D., 1992. Method for registration of 3-d shapes. *Sensor fusion IV: control paradigms and data structures*, 1611, International Society for Optics and Photonics, 586–606.

Cai, Z., Chin, T.-J., Bustos, A. P., Schindler, K., 2019. Practical optimal registration of terrestrial LiDAR scan pairs. *ISPRS Journal of Photogrammetry and Remote Sensing*, 147, 118–131.

Chen, C.-S., Hung, Y.-P., Cheng, J.-B., 1999. RANSAC-based DARCES: A new approach to fast automatic registration of partially overlapping range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(11), 1229–1234.

Choi, D.-W., Chung, C.-W., Tao, Y., 2012. A scalable algorithm for maximizing range sum in spatial databases. *Proceedings of the VLDB Endowment*, 1088–1099.

Imai, H., Asano, T., 1983. Finding the connected components and a maximum clique of an intersection graph of rectangles in the plane. *Journal of Algorithms*, 4(4), 310–323.

Jaw, J.-J., Chuang, T.-Y., 2008. Registration of ground-based LiDAR point clouds by means of 3D line features. *Journal of the Chinese Institute of Engineers*, 31(6), 1031–1045.

Laaksonen, A., 2017. *Guide to Competitive Programming*. Springer.

Li, Z., Zhang, X., Tan, J., Liu, H., 2021. Pairwise coarse registration of indoor point clouds using 2D line features. *ISPRS International Journal of Geo-Information*, 10(1), 26.

Rusu, R. B., Blodow, N., Beetz, M., 2009. Fast point feature histograms (fpfh) for 3d registration. *2009 IEEE International Conference on Robotics and Automation*, IEEE, 3212–3217.

Theiler, P. W., Wegner, J. D., Schindler, K., 2015. Globally consistent registration of terrestrial laser scans via graph optimization. *ISPRS Journal of Photogrammetry and Remote Sensing*, 109, 126–138.

Yang, J., Li, H., Campbell, D., Jia, Y., 2016. Go-ICP: A globally optimal solution to 3D ICP point-set registration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11), 2241–2254.

Zhong, Y., 2009. Intrinsic shape signatures: A shape descriptor for 3d object recognition. *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, IEEE, 689–696.

APPENDIX A. PROOF OF PROPOSITION 1

Define a problem $\mathcal{P}[k]$ as a sub-problem of the original Problem 1, which is a constraint in which the k-th match must be aligned, i.e.,

$$\text{find } (\theta_k^\star, \boldsymbol{t}_k^\star) \in \underset{(\theta,\boldsymbol{t})\in[0,2\pi)\times\mathbb{R}^3}{\arg\max} E(\theta, \boldsymbol{t}; \mathcal{I}), \tag{27}$$

$$\text{s.t. } \|R(\theta)\boldsymbol{p}_k + \boldsymbol{t} - \boldsymbol{q}_k\|_{(\epsilon_1,\epsilon_2)} \leq 1. \tag{28}$$

According to the similar discussion in (Cai et al., 2019, Lemma 2), it is sufficient to show $E_k^* \leq \bar{E}_k < \underline{E}$, where $E_k^* := E(\theta_k^\star, \boldsymbol{t}_k^\star; \mathcal{I})$ and $\underline{E} := E(\theta_0, \boldsymbol{t}_0; \mathcal{I})$. The latter inequality, i.e., $\bar{E}_k < \underline{E}$, is satisfied from the assumption. Therefore, we prove $E_k^* \leq \bar{E}_k$.

Define $\boldsymbol{t}_k^{\star\prime} := \boldsymbol{q}_k - R(\theta_k^\star)\boldsymbol{p}_k - \boldsymbol{t}_k^\star$, then the constraint (28) can

be rewritten as $\|\boldsymbol{t}_k^{\star\prime}\|_{(\epsilon_1,\epsilon_2)} \leq 1$. Therefore,

$$E_k^* = \sum_{i\in\mathcal{I}} \iota_1 \left(\|R(\theta_k^\star)\boldsymbol{p}_i + \boldsymbol{t}_k^\star - \boldsymbol{q}_i\|_{(\epsilon_1,\epsilon_2)}\right) \tag{29}$$

$$= \sum_{i\in\mathcal{I}} \iota_1 \left(\|R(\theta_k^\star)\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)} - \boldsymbol{t}_k^{\star\prime}\|_{(\epsilon_1,\epsilon_2)}\right) \tag{30}$$

$$= \sum_{i\in\mathcal{I}} \iota_{\epsilon_1} \left(\|[R(\theta_k^\star)\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)}]_{\mathrm{xy}} - [\boldsymbol{t}_k^{\star\prime}]_{\mathrm{xy}}\|_2\right) \times \iota_{\epsilon_2} \left(|[\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)}]_{\mathrm{z}} - [\boldsymbol{t}_k^{\star\prime}]_{\mathrm{z}}|\right) \tag{31}$$

$$\leq \sum_{i\in\mathcal{I}} \iota_{(\epsilon_1+\|[\boldsymbol{t}_k^{\star\prime}]_{\mathrm{xy}}\|_2)} \left(\|[R(\theta_k^\star)\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)}]_{\mathrm{xy}}\|\right) \times \iota_{(\epsilon_2+|[\boldsymbol{t}_k^{\star\prime}]_{\mathrm{z}}|)} \left(|[\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)}]_{\mathrm{z}}|\right) \tag{32}$$

$$\leq \sum_{i\in\mathcal{I}} \iota_{2\epsilon_1} \left(\|[R(\theta_k^\star)\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)}]_{\mathrm{xy}}\|\right) \iota_{2\epsilon_2} \left(|[\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)}]_{\mathrm{z}}|\right) \tag{33}$$

$$= \sum_{i\in\mathcal{I}_k} \iota_{2\epsilon_1} \left(\|[R(\theta_k^\star)\boldsymbol{p}_i^{(k)} - \boldsymbol{q}_i^{(k)}]_{\mathrm{xy}}\|_2\right) \leq \bar{E}_k. \tag{34}$$

APPENDIX B. SOLVING RECTANGLE INTERSECTION PROBLEM

We describe an algorithm to solve the rectangle-intersection problem shown in Figure 3 by computation complexity $\mathcal{O}(N \log N)$ for N rectangles. To store the information about rectangle sides, the algorithm uses the segment tree with lazy propagation (STLP) (Laaksonen, 2017, Sec. 15.2.1), which is a tree-data structure storing information about intervals. The STLP can compute the following range updates and queries by $\mathcal{O}(\log N)$.

- $\mathrm{ADD}_{n_1}^{n_2}(x)$: add $x \in \mathbb{R}$ to every element in $[n_1, n_2)$.
- $\mathrm{MAX}_{n_1}^{n_2}()$: obtain the index $i \in [n_1, n_2)$, which gives the maximum value in $[n_1, n_2)$. We denote $\mathrm{MAX}_{\mathrm{all}}()$ as $\mathrm{MAX}_0^N()$, which obtains the index that gives a maximum value in all elements.

By using the STLP, we can solve the rectangle-intersection problem. In Algorithm 4, we denote the vertices of the i-th rectangle as $\{(\theta_{\mathrm{low}}^{(i)}, t_{\mathrm{low}}^{(i)}), (\theta_{\mathrm{low}}^{(i)}, t_{\mathrm{high}}^{(i)}), (\theta_{\mathrm{high}}^{(i)}, t_{\mathrm{low}}^{(i)}), (\theta_{\mathrm{high}}^{(i)}, t_{\mathrm{high}}^{(i)})\}$, where $\theta_{\mathrm{low}}^{(i)} < \theta_{\mathrm{high}}^{(i)}$ and $t_{\mathrm{low}}^{(i)} < t_{\mathrm{high}}^{(i)}$.

Algorithm 4 for the rectangle intersection problem.

Set array $\Theta[\,] \leftarrow \mathrm{ascending\_sort}(\bigcup_{i=0}^{N-1}\{\theta_{\mathrm{low}}^{(i)}, \theta_{\mathrm{high}}^{(i)}\})$.
Set array $T[\,] \leftarrow \mathrm{ascending\_sort}(\bigcup_{i=0}^{N-1}\{t_{\mathrm{low}}^{(i)}, t_{\mathrm{high}}^{(i)}\})$.
Build STLP with size $2N$ as S.
Initialize score and solutions $(s, \theta^*, t^*) \leftarrow (0, 0, 0)$.
For $i = 0, \ldots, 2N-1$ **do**
  If $T[i]$ is $t_{\mathrm{low}}^{(k)}$ for a k **then**
    $S.\mathrm{ADD}_{n_1}^{n_2}(1)$ for $\Theta[n_1] = \theta_{\mathrm{low}}^{(k)}$ and $\Theta[n_2] = \theta_{\mathrm{high}}^{(k)}$.
    Compute $(j, \tilde{s}) \leftarrow S.\mathrm{MAX}_{\mathrm{all}}()$.
    If $\tilde{s} > s$ **then** $(s, \theta^*, t^*) \leftarrow (\tilde{s}, \Theta[n_1], T[i])$.
  Else
    $S.\mathrm{ADD}_{n_1}^{n_2}(-1)$ for $\Theta[n_1] = \theta_{\mathrm{low}}^{(k)}$ and $\Theta[n_2] = \theta_{\mathrm{high}}^{(k)}$.
return θ^*, t^*

The sort operators in Algorithm 4 are computed by $\mathcal{O}(N \log N)$. Each loop for i is are executed by $\mathcal{O}(\log N)$; hence, the overall computation complexity is $\mathcal{O}(N \log N)$.

APPENDIX C. PROOF OF PROPOSITION 2

Define $\boldsymbol{r}_i(\theta, \boldsymbol{t}) := R(\theta)\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_i$ for every $i \in \mathcal{I}$. For

$$(\theta_{\mathcal{S}}^\star, \boldsymbol{t}_{\mathcal{S}}^\star) \in \operatorname*{arg\,max}_{(\theta,[\boldsymbol{t}]_{\mathrm{xy}},[\boldsymbol{t}]_{\mathrm{z}})\in[0,2\pi)\times\mathcal{S}\times\mathbb{R}} E(\theta, \boldsymbol{t}; \mathcal{I}), \tag{35}$$

let $\boldsymbol{r}_i^\star := \boldsymbol{r}_i(\theta_{\mathcal{S}}^\star, \boldsymbol{t}_{\mathcal{S}}^\star)$ for every $i \in \mathcal{I}$, then

$$\max_{[\boldsymbol{t}]_{\mathrm{xy}}\in\mathcal{S}} U([\boldsymbol{t}]_{\mathrm{xy}}; \epsilon_1, \epsilon_2, \mathcal{I}) = E(\theta_{\mathcal{S}}^\star, \boldsymbol{t}_{\mathcal{S}}^\star; \mathcal{I}) \tag{36}$$

$$= \sum_{i\in\mathcal{I}} \iota_{\epsilon_1} \left(\|[(\boldsymbol{r}_i^\star - \boldsymbol{t}_{\mathcal{S}}^\star + \boldsymbol{s}_0) + (\boldsymbol{t}_{\mathcal{S}}^\star - \boldsymbol{s}_0)]_{\mathrm{xy}}\|_2\right) \iota_{\epsilon_2} \left(|[\boldsymbol{r}_i^\star]_{\mathrm{z}}|\right) \tag{37}$$

$$\leq \sum_{i\in\mathcal{I}} \iota_{(\epsilon_1+d_{\mathcal{S}})} \left(\|[\boldsymbol{r}_i^\star - \boldsymbol{t}_{\mathcal{S}}^\star + \boldsymbol{s}_0]_{\mathrm{xy}}\|_2\right) \iota_{\epsilon_2} \left(|[\boldsymbol{r}_i^\star]_{\mathrm{z}}|\right) \tag{38}$$

$$\leq \max_{(\theta,t_z)\in[0,2\pi)\times\mathbb{R}} \sum_{i\in\mathcal{I}} \iota_1 \left(\|\boldsymbol{r}_i(\theta, [\boldsymbol{s}_0, t_z]^\top)\|_{(\epsilon_1+d_{\mathcal{S}},\epsilon_2)}\right) \tag{39}$$

$$= U(\boldsymbol{s}_0; \epsilon_1 + d_{\mathcal{S}}, \epsilon_2, \mathcal{I}). \tag{40}$$