ARCS: Accurate Rotation and Correspondence Search

Liangzu Peng
Johns Hopkins University
lpeng25@jhu.edu

Manolis C. Tsakiris
ShanghaiTech University
mtsakiris@shanghaitech.edu.cn

René Vidal
Johns Hopkins University
rvidal@jhu.edu

Abstract

This paper is about the old Wahba problem in its more general form, which we call “simultaneous rotation and correspondence search”. In this generalization we need to find a rotation that best aligns two partially overlapping 3D point sets, of sizes $m$ and $n$ respectively with $m \geq n$. We first propose a solver, ARCS, that i) assumes noiseless point sets in general position, ii) requires only 2 inliers, iii) uses $O(m \log m)$ time and $O(m)$ space, and iv) can successfully solve the problem even with, e.g., $m, n \approx 10^6$ in about 0.1 seconds. We next robustify ARCS to noise, for which we approximately solve consensus maximization problems using ideas from robust subspace learning and interval stabbing. Thirdly, we refine the approximately found consensus set by a Riemannian subgradient descent approach over the space of unit quaternions, which we show converges globally to an $\varepsilon$-stationary point in $O(\varepsilon^{-4})$ iterations, or locally to the ground-truth at a linear rate in the absence of noise. We combine these algorithms into ARCS+, to simultaneously search for rotations and correspondences. Experiments show that ARCS+ achieves state-of-the-art performance on large-scale datasets with more than $10^6$ points with a $10^4$ time-speedup over alternative methods. https://github.com/liangzu/ARCS

1. Introduction

The villain Procrustes forced his victims to sleep on an iron bed; if they did not fit the bed he cut off or stretched their limbs to make them fit [27].

Modern sensors have brought the classic Wahba problem [75], or slightly differently the Procrustes analysis problem [31], into greater generality that has increasing importance to computer vision [34,50], computer graphics [58], and robotics [12]. We formalize this generalization as follows.

Problem 1 (simultaneous rotation and correspondence search). Consider point sets $Q = \{q_1, \ldots, q_m\} \subset \mathbb{R}^3$ and $P = \{p_1, \ldots, p_n\} \subset \mathbb{R}^3$ with $m \geq n$. Let $C^*$ be a subset of $[m] \times [m] := \{1, \ldots, m\} \times \{1, \ldots, n\}$ of size $k^*$, called the inlier correspondence set, such that all pairs $(i_1, j_1)$ and $(i_2, j_2)$ of $C^*$ satisfy $i_1 \neq i_2$ and $j_1 \neq j_2$. Assume that

$$q_i = R^*p_{j} + \epsilon_{i,j}, \quad \text{if } (i,j) \in C^* \quad (1)$$

where $\epsilon_{i,j} \sim \mathcal{N}(0, \sigma^2 I_3)$ is noise, $R^*$ is an unknown 3D rotation, and $(q_i, p_j)$ is called an inlier. If $(i,j) \notin C^*$ then $(q_i, p_j)$ is arbitrary and is called an outlier. The goal of the simultaneous rotation and correspondence search problem is to simultaneously estimate the 3D rotation $R^*$ and the inlier correspondence set $C^*$ from point sets $Q$ and $P$.

We focus on Problem 1 for two reasons. First, it already encompasses several vision applications such as image stitching [16]. Second, the more general and more important simultaneous pose and correspondence problem, which involves an extra unknown translation in (1), reduces to Problem 1 by eliminating the translation parameters (at the cost of squaring the number of measurements) [80]. As surveyed in [38], whether accurate and fast algorithms exist for solving the pose and correspondence search is largely an open question. Therefore, solving the simpler Problem 1 efficiently is an important step for moving forward.

For Problem 1 or its variants, there is a vast literature of algorithms that are based on i) local optimization via iterative closest points (ICP) [13,20,66] or graduated non-convexity (GNC) [1,76,85] or others [23,41,59], ii) global optimization by branch & bound [18,21,50,54,55,64,70,81], iii) outlier removal techniques [16,62,63,69,80], iv) semidefinite programming [39,58,71,77,79], v) RANSAC [28,51,52,72], vi) deep learning [4,9,22,37], and vii) spherical Fourier transform [12]. But all these methods, if able to accurately solve Problem 1 with the number $k^*$ of inliers extremely small, take $\Omega(mn)$ time. Yet we have:

Theorem 1 (ARCS). Suppose there are at least two inliers, $k^* \geq 2$, and that the point sets $Q$ and $P$ of Problem 1 are noiseless “in general position”. Then there is an algorithm that solves Problem 1 in $O(m \log m)$ time and $O(m)$ space.
Remark 1 (general position assumption). In Theorem 1, by “in general position” we mean that i) for any outlier \((q_i, p_j)\), we have \(\|q_i\| \neq \|p_j\|\), ii) there exists some inlier pairs \((q_{i1}, p_{j1})\) and \((q_{i2}, p_{j2})\) such that \(q_{i1}\) and \(q_{i2}\) are not parallel. If point sets \(Q\) and \(P\) are randomly sampled from \(\mathbb{R}^3\), these two conditions hold true with probability 1.

A numerical illustration of Theorem 1 is that our ARCS solver, to be described in §3, can handle the case where \(m = 10^6\), \(n = 8 \times 10^5\) and \(k^* = 2\), in about 0.1 seconds (cf. Table 1).\(^1\) However, like other correspondence-based minimal solvers for geometric vision [29,45–47,61], ARCS might be fragile to noise. That being said, it can be extended to the noisy case, leading to a three-step algorithm called ARCS+, which we summarize next.

The first step ARCS+ of ARCS+ extends ARCS by establishing correspondences under noise. ARCS+ outputs in \(O(\ell + m \log m)\) time a candidate correspondence set \(\hat{C}\) of size \(\ell\) that contains \(C^*\). Problem 1 then reduces to estimating \(R^*\) and \(C^*\) from \(P, Q, \hat{C}\), and hypothetical correspondences \(\hat{C}\), a simpler task of robust rotation search [16,63,77,85].

The second step ARCS+ of ARCS+ is to remove outliers from the previous step 1. To do so we approximately maximize an appropriate consensus over \(SO(3)\) (§4.2). Instead of mining inliers in \(SO(3)\) [10,34,43,50,64], we show that the parameter space of consensus maximization can be reduced from \(SO(3)\) to \(S^2\) and further to \([0,\pi]\) (see [16] for a different reduction). With this reduction, ARCS+ removes outliers via repeatedly solving in \(O(\ell \log \ell)\) time a computational geometry problem, interval stabbing [24] (§4.2.1). Note that ARCS+ only repeats for \(s \approx 90\) times to reach satisfactory accuracy. Therefore, conceptually, for \(\ell \geq 10^6\), it is \(10^4\) times faster than the most related outlier removal method GORE [16], which uses \(O(\ell^2 \log \ell)\) time (Table 4).

The third and final step ARCS of ARCS+ is to accurately estimate the rotation, using the consensus set from the second step (§4.3). In short, ARCS+ is a Riemannian subgradient descent method. Our novelty here is to descend in the space \(S^3\) of unit quaternions, not \(SO(3)\) [14]. This allows us to derive, based on [53], that ARCS+ converges linearly though locally to the ground-truth unit quaternion, thus obtaining the first to our knowledge convergence rate guarantee for robust rotation search.

Numerical highlights are in order (§5). ARCS+ is an outlier pruning procedure for robust rotation search that can handle extremely small inlier ratios \(k^*/\ell = 3000/10^7 = 0.03\%\) in 5 minutes; ARCS+ + ARCS+R, or ARCS+GR for short, accurately solves the robust rotation search problem with \(k^*/\ell = 10^2/10^6\) in 23 seconds (see Table 4). ARCS+ + ARCS+GR, that is ARCS+, solves Problem 1 with \(m = 10^3\), \(n = 8000\), \(k^* = 2000\) in 90 seconds (see Figure 2). To the best of our knowledge, all these challenging cases have not been considered in prior works. In fact, as we will review soon (§2), applying state-of-the-art methods to those cases either gives wrong estimates of rotations, or takes too much time (\(\geq 8\) hours), or exhausts the memory (Table 4).

2. Prior Art: Accuracy Versus Scalability

Early efforts on Problem 1 have encountered an accuracy versus scalability dilemma. The now classic ICP algorithm [13] estimates the rotation and correspondences in an alternating fashion, running in real time but requiring a high-quality and typically unavailable initialization to avoid local and usually poor minima; the same is true for its successors [20,23,41,59,66]. The GO–ICP method [81,82] of the branch & bound type enumerates initializations fed to ICP to reach a global minimum—in exponential time; the same running time bound is true for its successors [18,55,64].

The above ICP versus GO–ICP dilemma was somewhat alleviated by a two-step procedure: i) compute a candidate correspondence set \(\hat{C}\), via hand-crafted [67] or learned [30] feature descriptors, and ii) estimate the rotation from point sets indexed by \(\hat{C}\). But, as observed in [80], due to the quality of the feature descriptors, there could be fewer than 2 inliers remaining in \(\hat{C}\), from which the ground-truth rotation can never be determined. An alternative and more conservative idea is to use all-to-all correspondences \(\hat{C} := [m] \times [n]\), although now the inlier ratio becomes extremely small.

This justifies why researchers have recently focused on designing robust rotation search algorithms for extreme outlier rates, e.g., \(\geq 90\) outliers out of 100. One such design is GORE [16], a guaranteed outlier removal algorithm of \(O(\ell^2 \log \ell)\) time complexity that heavily exploits the geometry of \(SO(3)\). The other one is the semidefinite relaxation QUASAR of [77], which involves sophisticated manipulation on unit quaternions; \(\ell \approx 1000\) constitutes the current limit on the number of points this relaxation can handle. Yet another one is TEASER++ [80]; its robustness to outliers comes mainly from finding via parallel branch & bound [65] a maximum clique of the graph whose vertices represent point pairs and whose edges indicate whether two point pairs can simultaneously be inliers. This maximum clique formulation was also explored by [62] where it was solved via a different branch & bound algorithm. Since finding a maximum clique is in general NP-hard, their algorithms take exponential time in the worst case; in addition, TEASER++ was implemented to trade \(O(\ell^2)\) space for speed. One should also note though that if noise is small then the graph is sparse so that the otherwise intractable branch & bound algorithm can be efficient. Since constructing such a graph entails checking \(\binom{\ell}{2}\) point pairs, recent follow-up works [51,56,69,71,72] that use such a graph entail \(O(\ell^2)\) time complexity. While all these methods are more accurate than scalable, the following two are on the other side. FGR [85] combines graduated non-convexity
of Table 1 we have only

overlapping ratio that they are not designed to handle the noiseless case. The other reason is that the overlapping ratio 100 = 80% outliers.

Is such accuracy versus scalability dilemma of an inherent nature of the problems here, or can we escape from it?

3. ARCS: Accuracy & Scalability

Basic Idea. Although perhaps not explicitly mentioned in the literature, it should be known that there is a simple algorithm that solves Problem 1 under the assumptions of Theorem 1. This algorithm first computes the $\ell_2$ norm of each point in $Q$ and $P$ and the differences $d_{i,j} := \|q_i\|_2 - \|p_j\|_2$. Since $Q$ and $P$ are in general position (Remark 1), we have that $(q_i, p_j)$ is an inlier pair if and only if $d_{i,j} = 0$. Based on the $d_{i,j}$’s, extract all such inlier pairs. Since $k^* = 2$, and by the general position assumption (Remark 1), there exist two inlier pairs say $(q_1, p_1), (q_2, p_2)$ such that $q_1$ and $q_2$ are not parallel. As a result and as it has been well-known since the 1980’s [3, 35, 36, 57], if not even earlier [68, 75], $R^*$ can be determined from the two inlier pairs by SVD.

ARCS: Efficient Implementation. Not all the $d_{i,j}$’s should be computed in order to find the correspondence set $C^*$, meaning that the otherwise $O(mn)$ time complexity can be reduced. Our ARCS Algorithm 1 seeks all point pairs $(q_i, p_j)$’s whose norms are close, i.e., they satisfy $|d_{i,j}| \leq c$, for some sufficiently small $c \geq 0$. Here $c$ is provided as an input of ARCS and set as 0 in the current context. It is clear that, under the general position assumption of Theorem 1, the set $\mathcal{C}$ returned by ARCS is exactly the ground-truth correspondence set $C^*$. It is also clear that ARCS takes $O(m \log m)$ time and $O(m)$ space (recall $m \geq n \geq |C^*|$).

We proved Theorem 1. It is operating in the noiseless case that allows us to show that Problem 1 can be solved accurately and at large scale. Indeed, ARCS can handle more than $10^6$ points with $k^* = 2$ in about 0.1 seconds, even though generating those points has taken more than 0.2 seconds, as shown in Table 1.2 Note that in the setting of Table 1 we have only $k^* = 2$ overlapping points, a situation where all prior methods mentioned in §1 and §2, if directly applicable, in principle break down. One reason is that they are not designed to handle the noiseless case. The other reason is that the overlapping ratio $k^*/m$ of Table 1 is the minimum possible. While the achievement in Table 1 is currently limited to the noiseless case, it forms a strong motivation that urges us to robustify ARCS to noise, while keeping as much of its accuracy and scalability as possible. Such robustification is the main theme of the next section.

4. ARCS+: Robustifying ARCS to Noise

Here we consider Problem 1 with noise $\epsilon_{i,j}$. We will illustrate our algorithmic ideas by assuming $\epsilon_{i,j} \sim N(0, \sigma^2 I_h)$, although this is not necessary for actual implementation. As indicated in §1, ARCS+ has three steps. We introduce them respectively in the next three subsections.

4.1. Step 1: Finding Correspondences Under Noise

A simple probability fact is $\|q_i - R^* p_j\|_2 \leq 5.54 \sigma$ for any inlier $(q_i, p_j)$, so $|d_{i,j}| \leq 5.54 \sigma$ with probability at least $1 - 10^{-6}$ (see, e.g., [80]). To establish correspondences under noise, we need to modify3 the while loop of Algorithm 1, such that, in $O(\ell + m \log m)$ time, it returns the set $\mathcal{C}$ of all correspondences of size $\ell$ where each $(i,j) \in \mathcal{C}$ satisfies $|d_{i,j}| \leq c$, with $c$ now set to 5.54 $\sigma$. Note that, to store the output correspondences, we need an extra $O(\ell)$ time, which can not be simply ignored as $\ell$ is in general larger than $m$ in the presence of noise (Table 2). We call this modified version ARCS+1. ARCS+1 gives a set $\mathcal{C}$ that...

3The details of this modification can be found at: https://github.com/liangzu/ARCS/blob/main/ARCSplus_N.m

<table>
<thead>
<tr>
<th>$m$</th>
<th>$n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^4$</td>
<td>$8 \times 10^3$</td>
</tr>
<tr>
<td>G</td>
<td>5.9</td>
</tr>
<tr>
<td>Brute Force</td>
<td>73.8</td>
</tr>
<tr>
<td>ARCS</td>
<td>1.51</td>
</tr>
</tbody>
</table>

Table 1. Time (msec) of generating noiseless Gaussian point sets (G) and solving Problem 1 by ARCS (100 trials, $k^* = 2$).
contains all inlier correspondences $\mathcal{C}^*$ with probability at least $(1 - 10^{-6})k^*$. This probability is larger than 99.9% if $k^* \leq 10^3$, or larger than 99% if $k^* \leq 10^4$.

Remark 2 (feature matching versus all-to-all correspondences versus ARCS+). Feature matching methods provide fewer than $n$ hypothetical correspondences and thus speed up the subsequent computation, but they might give no inliers. Using all-to-all correspondences preserves all inliers, but a naïve computation needs $O(nm)$ time and leads to a large-scale problem with extreme outlier rates. ARCS+ strikes a balance by delivering in $O(n\log m)$ time a candidate correspondence set $\mathcal{C}$ of size $\ell$ containing all inliers with high probability and with $\ell \ll mn$.

For illustration, Table 2 reports the number $\ell$ of correspondences that ARCS+ typically yields. As shown, even though $\ell/(mn)$ is usually smaller than 5%, yet $\ell$ itself could be very large, and the inlier ratio $k^*/\ell$ is extremely small (e.g., $\leq 0.05\%$). This is perhaps the best we could do for the current stage, because for now only considered every point pair individually, while any pair $(q_i, p_i)$ is a potential inlier if it satisfies the necessary (but no longer sufficient) condition $|d_{ij}| \leq c$. On the other hand, collectively analyzing the remaining point pairs allows to further remove outliers, and this is the major task of our next stage (§4.2).

### 4.2. Step 2: Outlier Removal

Let there be some correspondences given, by, e.g., either ARCS+ or feature matching (cf. Remark 2). Then we arrive at an important special case of Problem 1, called robust rotation search. For convenience we formalize it below:

**Problem 2. (robust rotation search)** Consider $\ell$ pairs of 3D points $\{(y_i, x_i)\}_{i=1}^{\ell}$, with each pair satisfying

$$y_i = R^* x_i + o_i + e_i.$$  

(2)

Here $e_i \sim \mathcal{N}(0, \sigma^2 I_3)$ is noise, $o_i = 0$ if $i \in \mathcal{I}^*$, and if $i \notin \mathcal{I}^*$ then $o_i$ is nonzero and arbitrary. The task is to find $R^*$ and $\mathcal{I}^*$.

The percentage of outliers in Problem 2 can be quite large (cf. Table 2), so our second step ARCS+ here is to remove outliers. In §4.2.1, we shortly review the interval stabbing problem, on which ARCS+ of §4.2.2 is based.

### 4.2.1 Preliminaries: Interval Stabbing

Consider a collection of subsets of $\mathbb{R}$, $\{\mathcal{J}_i\}_{i=1}^L$, where each $\mathcal{J}_i$ is an interval of the form $[a, b]$. In the interval stabbing problem, one needs to determine a point $\omega \in \mathbb{R}$ and a subset $\mathcal{I}$ of $\{\mathcal{J}_i\}_{i=1}^L$, so that $\mathcal{I}$ is a maximal subset whose intervals overlap at $\omega$. Formally, we need to solve

$$\max_{\mathcal{I} \subset \mathcal{J}_i, \omega \in \mathbb{R}} |\mathcal{I}|$$

(3)

s.t. $\omega \in \mathcal{J}_i, \forall i \in \mathcal{I}$

For this purpose, the following result is known.

**Lemma 1 (interval stabbing).** Problem (3) can be solved in $O(L \log L)$ time and $O(L)$ space.

Actually, the interval stabbing problem can be solved using sophisticated data structures such as interval tree [24] or interval skip list [32]. On the other hand, it is a basic exercise to find an algorithm that solves Problem (3), which, though also in $O(L \log L)$ time, involves only a sorting operation and a for loop (details are omitted, see, e.g., [17]). Finally, note that the use of interval stabbing for robust rotation search is not novel, and can be found in GORE [16,63]. However, as the reader might realize after §4.2.2, our use of interval stabbing is quite different from GORE.

### 4.2.2 The Outlier Removal Algorithm

We now consider the following consensus maximization:

$$\max_{\mathcal{I} \subset \mathcal{J}_i, R \in SO(3)} |\mathcal{I}|$$

(4)

s.t. $||y_i - Rx_i||_2 \leq c, \forall i \in \mathcal{I}$

It has been shown in [73] that for the very related robust fitting problem, such consensus maximization is in general NP-hard⁴. Thus it seems only prudent to switch our computational goal from solving (4) exactly to approximately.

From $SO(3)$ to $S^2$. Towards this goal, we first shift our attention to $S^2$ where the rotation axis $b^*$ of $R^*$ lives. An interesting observation is that the axis $b^*$ has the following interplay with data, independent of the rotation angle of $R^*$.

**Proposition 1.** Let $v_i := y_i - x_i$. Recall $e_i \sim \mathcal{N}(0, \sigma^2 I_3)$. If $(y_i, x_i)$ is an inlier pair, then $v_i^\top b^* \sim \mathcal{N}(0, \sigma^2)$, and so $|v_i^\top b^*| \leq 4.9\sigma$ with probability at least $1 - 10^{-6}$.

Proposition 1 (cf. Appendix C) leads us to Problem (5):

$$\max_{\mathcal{I} \subset \mathcal{J}_i, b \in S^2} |\mathcal{I}|$$

s.t. $|v_i^\top b| \leq \bar{c}, \forall i \in \mathcal{I}$

$$b_2 \geq 0.$$  

(5)

⁴Interestingly, consensus maximization over $SO(2)$, i.e., the 2D version of (4), can be solved in $O(\ell \log \ell)$ time; see [17].
In (5) the constraint on the second entry $b_2$ of $b$ is to eliminate the symmetry, and Proposition 1 suggests to set $\tilde{c} := 4.9a$. Problem (5) is easier than (4) as it has fewer degrees of freedom; see also [16] where a different reduction to a 2 DoF (sub-)problem was derived for GORE.

Solving (5) is expected to yield an accurate estimate of $b^*$, from which the rotation angle can later be estimated. Problem (5) reads: find a plane (defined by the normal $b$) that approximately contains as much points $v_i$’s as possible. This is an instance of the robust subspace learning problem [25, 26, 48, 74, 83, 86, 87], for which various scalable algorithms with strong theoretical guarantees have been developed in more tractable formulations (e.g., $\ell_1$ minimization) than consensus maximization. Most notably, the so-called dual principal component pursuit formulation [74] was proved in [87] to be able to tolerate $O(\log \ell)$ outliers. Still, all these methods can handle as many outliers as we currently have (cf. Table 2), even though they can often minimize their objective functions to global optimality.

From $S^2$ to $[0, \pi]$. We can further “reduce” the degrees of freedom in (5) by 1, through the following lens. Certainly $b \in S^2$ in (5) is determined by two angles $\theta, \phi \in [0, \pi]$. Now consider the following problem:

$$\max_{I \subset [\ell], \omega \in [0, \pi]} |I|$$

s.t. $|v_i^T b| \leq \tilde{c}, \forall i \in I$

$$b = [\sin(\theta) \cos(\phi), \sin(\theta) \sin(\phi), \cos(\theta)]^T. \quad (6)$$

Problem (6) is a simplified version of (5) with $\phi$ given. Clearly, to solve (5) it suffices to minimize the function $f : [0, \pi] \to \mathbb{R}$ which maps any $\phi \in [0, \pi]$ to the objective value of (6) with $\phi = \phi_0$. Moreover, we have:

**Proposition 2.** Problem (6) can be solved in $O(\ell \log \ell)$ time and $O(\ell)$ space via interval stabbing.

Proposition 2 gives an $O(\ell \log \ell)$ time oracle to access the values of $f$. Since computing the objective value of (5) given $\theta, \phi$ already needs $O(\ell)$ time, the extra cost of the logarithmic factor in Proposition 2 is nearly negligible. Since $f$ has only one degree of freedom, its global minimizer can be found by one-dimensional branch & bound [42]. But this entails exponential time complexity in the worst case, a situation we wish to sidestep. Alternatively, the search space $[0, \pi]$ is now so small that the following algorithm ARCS$^+\omega$ turns out to be surprisingly efficient and robust: i) sampling from $[0, \pi]$, ii) stabbing in $S^2$, and iii) stabbing in SO(3).

**Sampling from $[0, \pi]$.** Take $s$ equally spaced points $\phi_j = (2j - 1) \pi / (2s), \forall j \in [s], [0, \pi]$. The reader may find this choice of $\phi_j$’s similar to the uniform grid approach [60]; in the latter Nesterov commented that “the reason why it works here is related to the dimension of the problem”.

**4.3. Step 3: Rotation Estimation**

The final step ARCS$^+_R$ of ARCS$^+$ is a refinement procedure that performs robust rotation search on the output correspondences $\tilde{I}$ of ARCS$^+\omega$. Since $\tilde{I}$ contains much fewer
outlier correspondences than we previously had (cf. Table 2 and 3), in which we simplify the notations by focusing on the point set \(\{(y_i, x_i)\}_{i \in [\ell]}\) which we assume has few outliers (say \(\leq 50\%\)). Then, a natural formulation is
\[
\min_{R \in \text{SO}(3)} \sum_{i=1}^{\ell} \|y_i - Rx_i\|_2.
\] (8)

Problem (8) appears easier to solve than consensus maximization (4), as it has a convex objective function at least. Next we present the ARCS+ algorithm and its theory.

**Algorithm.** We start with the following equivalence.

**Proposition 4.** We have \(w^\top D_i w = \|y_i - R x_i\|_2^2\), where \(w \in S^3\) is a quaternion representation of \(R\) of (8), and \(D_i \in \mathbb{R}^{4 \times 4}\) is a positive semi-definite matrix whose entries depend on \(x_i, y_i\). So Problem (8) is equivalent to
\[
\min_{w \in S^3} h(w), \quad h(w) = \sum_{i=1}^{\ell} \sqrt{w^\top D_i w}.
\] (9)

The exact relation between unit quaternions and rotations is reviewed in Appendix A, where Proposition 4 is proved and the expression of \(D_i\) is given. For what follows, it suffices to know that a unit quaternion is simply a unit vector of \(\mathbb{R}^4\), and that the space of unit quaternions is \(S^3\).

Note that the objective \(h\) of (9) is convex but both problems (8) and (9) are nonconvex (due to the constraint) and nonsmooth (due to the objective). Though (8) and (9) are equivalent, the advantage of (9) will manifest itself soon. Before that, we first introduce the ARCS+ algorithm for solving (9), ARCS+ falls into the general Riemannian subgradient descent framework (see, e.g., [53]). It is initialized at some unit quaternion \(w^{(0)} \in S^3\) and proceeds by
\[
w^{(t+1)} \leftarrow \text{Proj}_{S^3}(w^{(t)} - \gamma^{(t)} \tilde{\nabla}_w h(w^{(t)})),
\] (10)
where \(\text{Proj}_{S^3}(\cdot)\) projects a vector onto \(S^3\), \(\gamma^{(t)}\) is some stepsize, \(\tilde{\nabla}_w h(w^{(t)})\) is a Riemannian subgradient\(^5\) of \(h\) at \(w^{(t)}\).

**Theory.** Now we are able to compare (8) and (9) from a theoretical perspective. As proved in [14], for any fixed outlier ratio and \(k^* > 0\), Riemannian subgradient descent when applied to (8) with proper initialization converges to \(R^*\) in finite time, as long as i) \(\ell\) is sufficiently large, ii) all points \(y_i\) and \(x_i\) are uniformly distributed on \(S^2\), iii) there is no noise. But in [14] no convergence rate is given. One main challenge of establishing convergence rates there is that projecting on \(\text{SO}(3)\) does not enjoy a certain kind of nonexpansiveness property, which is important for convergence analysis (cf. Lemma 1 of [53]). On the other hand, projection onto \(S^3\) of (9) does satisfy such property. As a result, we are able to provide convergence rate guarantees for ARCS+. For example, it follows directly from Theorem 2 of [53] that ARCS+ (10) converges to an \(\varepsilon\)-stationary point in \(O(\varepsilon^{-4})\) iterations, even if initialized arbitrarily.

We next give conditions for ARCS+ to converge linearly to the ground-truth unit quaternion \(\pm w^*\) that represents \(R^*\). Let the distance between a unit quaternion \(w\) and \(\pm w^*\) be
\[
dist(w, \pm w^*) := \min \left\{ \|w - w^*\|_2, \|w + w^*\|_2 \right\}.
\]
If \(\text{dist}(w, \pm w^*) < \rho\) with \(\rho > 0\) then \(w\) is called \(\rho\)-close to \(\pm w^*\). We need the following notion of sharpness.

**Definition 1 (sharpness [15,44,49,53]).** We say that \(\pm w^*\) is an \(\alpha\)-sharp minimum of (9) if \(\alpha > 0\) and if there exists a number \(\rho_\alpha > 0\) such that any unit quaternion \(w \in S^3\) that is \(\rho_\alpha\)-close to \(\pm w^*\) satisfies the inequality
\[
h(w) - h(w^*) \geq \alpha \text{dist}(w, \pm w^*).
\] (11)

We provide a condition below for \(\pm w^*\) to be \(\alpha\)-sharp:

**Proposition 5.** If \(\alpha^* := k^* \eta_{\min}/\sqrt{2} - (\ell - k^*) \eta_{\max} > 0\) and if \(e_i = 0\) in Problem 2, then Problem (9) admits \(\pm w^*\) as an \(\alpha^*\)-sharp minimum. Here \(\eta_{\min}\) and \(\eta_{\max}\) are respectively
\[
\eta_{\min} := \frac{1}{k^*} \min_{w \in S^3} \sum_{i \in [\ell]} \sqrt{w^\top D_i w}, \quad \text{and}
\] (12)
\[
\eta_{\max} := \frac{1}{\ell - k^*} \max_{\{i \in [\ell] \setminus I\}} \sum_{i \in [\ell]} \sqrt{w^\top D_i w}, \quad \text{and}
\] (13)

where \(S^*\) is the hyperplane of \(\mathbb{R}^4\) perpendicular to \(\pm w^*\).

Proposition 5 is proved in Appendix B.1. The condition \(\alpha^* > 0\) defines a relation between the number of inliers \((k^*)\) and outliers \((\ell - k^*)\), and involves two quantities \(\eta_{\min}\) and \(\eta_{\max}\) whose values depend on how \(D_i\)’s are distributed on the positive semi-definite cone. We offer probabilistic interpretations for \(\eta_{\min}\) and \(\eta_{\max}\) in Appendix B.2.

With Theorem 4 of [53] and Proposition 5 we have that ARCS+ (10), if initialized properly and with suitable stepsizes, converges linearly to the ground-truth unit quaternion \(\pm w^*\), as long as \(\pm w^*\) is \(\alpha^*\)-sharp. A formal statement is:

**Theorem 2.** Suppose \(\alpha^* := k^* \eta_{\min}/\sqrt{2} - (\ell - k^*) \eta_{\max} > 0\). Let \(L_h\) be a Lipschitz constant of \(h\). Run Riemannian subgradient descent ARCS+ (10) with initialization \(w^{(0)}\) satisfying \(\text{dist}(w^{(0)}, \pm w^*) \leq \min\{\alpha^*/L_h, \rho_\alpha^*\}\) and with geometrically diminishing stepsizes \(\gamma^{(t)} = \beta^2 \gamma^{(0)}\), where
\[
\gamma^{(0)} < \min \left\{ \frac{2\epsilon_0 (\alpha^* - L_h e_0)}{L_h^2}, \frac{2\alpha^* - L_h e_0}{\epsilon_0} \right\},
\]
\[
\beta^2 \in \left[ 1 + 2 \left( \frac{\alpha^* - L_h e_0}{\epsilon_0} \right) \gamma^{(0)} + \frac{L_h^2 \gamma^{(0)} e_0^2}{\epsilon_0}, 1 \right],
\]
\[
e_0 = \min \left\{ \max \left\{ \text{dist}(w^{(0)}, \pm w^*), \frac{\alpha^*}{2L_h} \right\}, \rho_\alpha^* \right\}.
\]
Table 4. Average errors in degrees | standard deviation | running times in seconds of various algorithms on synthetic data (20 trials).

<table>
<thead>
<tr>
<th>Inlier Ratio $k^*/\ell$</th>
<th>$10^3/10^5 = 1%$</th>
<th>$10^3/10^5 = 0.1%$</th>
<th>$3 \times 10^3/5 \times 10^5 = 0.06%$</th>
<th>$3 \times 10^3/10^5 = 0.03%$</th>
<th>$10^3/10^5 = 0.01%$</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEASER++ [80]</td>
<td>out-of-memory</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RANSAC</td>
<td>0.39</td>
<td>0.20</td>
<td>29.1</td>
<td>≥ 8.4 hours</td>
<td></td>
</tr>
<tr>
<td>GORE [16,63]</td>
<td>3.43</td>
<td>2.10</td>
<td>1698</td>
<td>≥ 12 hours</td>
<td></td>
</tr>
<tr>
<td>FGR [85]</td>
<td>52.2</td>
<td>68.5</td>
<td>3.64</td>
<td>95.0</td>
<td>60.9</td>
</tr>
<tr>
<td>GNC–TLS [76]</td>
<td>3.86</td>
<td>9.51</td>
<td>0.13</td>
<td>63.4</td>
<td>50.5</td>
</tr>
<tr>
<td>ARCS+$_R$</td>
<td>9.92</td>
<td>13.1</td>
<td>0.12</td>
<td>65.2</td>
<td>48.9</td>
</tr>
<tr>
<td>ARCS+$_O$</td>
<td>0.86</td>
<td>0.29</td>
<td>1.71</td>
<td>0.99</td>
<td>0.37</td>
</tr>
<tr>
<td>ARCS+$_{OR}$</td>
<td>0.03</td>
<td>0.03</td>
<td>1.72</td>
<td>0.09</td>
<td>0.07</td>
</tr>
</tbody>
</table>

In the noiseless case ($\epsilon_i = 0$) we have each $w^{(i)}$ satisfying

$$\text{dist}(w^{(i)}, \pm w^*) \leq 3^i \epsilon_0.$$  \hspace{1cm} (14)

Remark 3 (a posteriori optimality guarantees). Theorem 2 endows ARCS+$_R$ (10) with convergence guarantee. On the other hand, a posteriori optimality guarantees can be obtained via semidefinite certification [5, 19, 78, 80].

5. Experiments

In this section we evaluate ARCS+ via synthetic and real experiments for Problem 1, simultaneous rotation and correspondence search. We also evaluate its components, namely ARCS+$_O$ (§4.2) and ARCS+$_R$ (§4.3) for Problem 2, robust rotation search, as it is a task of independent interest. For both of the two problems we compare the following state-of-the-art methods (reviewed in §2): FGR [85], GORE [16], RANSAC, GNC–TLS [76], and TEASER++ [80].

5.1. Experiments on Synthetic Point Clouds

Setup. We set $\sigma = 0.01$, $\epsilon = c = 5.54\sigma$, $n = [0.8m]$, and $s = 90$ unless otherwise specified. For all other methods we used default or otherwise appropriate parameters. We implemented ARCS+ in MATLAB. No parallelization was explicitly used and no special care was taken for speed.

Robust Rotation Search. From $N(0, I_3)$ we randomly sampled point pairs $\{(y_j, x_i)\}_{j=1}^{k^*}$ with $k^*$ inliers and noise $\epsilon_i \sim N(0, \sigma^2 I_3)$. Specifically, we generated the ground-truth rotation $R^*$ from an axis randomly sampled from $S^2$ and an angle from $[0, 2\pi]$, rotated $k^*$ points randomly sampled from $N(0, I_3)$ by $R^*$, and added noise to obtain $k^*$ inlier pairs. Every outlier point $y_j$ or $x_j$ was randomly sampled from $N(0, I_3)$ with the constraint $-c \leq ||y_j||_2 - ||x_j||_2 \leq c$; otherwise $(y_j, x_j)$ might simply be detected and removed by computing $||y_j||_2 - ||x_j||_2$.

We compared ARCS+$_O$ and ARCS+$_R$ and their combination ARCS+$_{OR}$ with prior works. The results are in Table 4. We first numerically illustrate the accuracy versus scalability dilemma in prior works (§2). On the one hand, we observed an extreme where accuracy overcomes scalability. RANSAC performed well with error $0.39$ when $k^*/\ell = 10^3/10^5$, but its running time increased greatly with decreasing inlier ratio, from 29 seconds to more than 8.4 hours. The other extreme is where scalability overcomes accuracy. Both GNC–TLS and FGR failed in presence of such many outliers—as expected—even though their running time scales linearly with $\ell$.

Table 4 also depicted the performance of our proposals ARCS+$_O$ and ARCS+$_R$. Our approximate consensus strategy ARCS+$_O$ reached a balance between accuracy and scalability. In terms of accuracy, it made errors smaller than 1 degree, as long as there are more than $3 \times 10^3/10^5 = 0.03\%$ inliers; this was further refined by Riemannian subgradient descent ARCS+$_R$, so that their combination ARCS+$_{OR}$ had even lower errors. In terms of scalability, we observed that ARCS+$_{OR}$ is uniformly faster than FGR, and is at least 1800 times faster than GORE for $k^*/\ell = 10^3/10^5 = 0.1\%$. But it had been harder to measure exactly how faster ARCS+$_{OR}$ is than GORE and RANSAC for even larger point sets. Finally, ARCS+$_{OR}$ failed at $k^*/\ell = 10^3/10^7 = 0.01\%$.

Simultaneous Rotation and Correspondence Search. We randomly sampled point sets $Q$ and $P$ from $N(0, I_3)$ with $k^*$ inlier pairs and noise $\epsilon_i \sim N(0, \sigma^2 I_3)$ (cf. Problem 1). Every outlier point was randomly and independently drawn also from $N(0, I_3)$. Figure 2 shows that ARCS+ accurately estimated the rotations for $k^* \geq 2000$ (in 90 seconds), and broke down at $k^* = 1000$, a situation where there were $k^*/m = 10\%$ overlapping points. We did not compare methods like TEASER++, GORE, RANSAC here, because giving them correspondences from ARCS+$_R$ would result unsatisfactory running time or accuracy (recall Tables 2 and 4), while feature matching methods like FPFH do not perform well on random synthetic data.

5.2. Experiments on 3DMatch

The 3DMatch\footnote{License info: https://3dmatch.cs.princeton.edu/} dataset [84] contains more than 1000 point clouds for testing, representing 8 different scenes.
Modern point clouds have more than $10^5$ points, and are naturally correspondences-less (cf. [17]). ARCS operates at that scale in the absence of noise (Table 1), while ARCS+ can handle $m, n \approx 10^5$ (Figure 2) and ARCS+$_{GR}$ can handle $\ell \approx 10^7$ correspondences (Table 4); all these are limited to the rotation-only case. To find rotation (and translation) from such point sets “in the wild”, it seems inevitable to downsample them. An interesting future work is to theoretically quantify the tradeoff between downsampling factors and the final registration performance. Another tradeoff to quantify, as implied by Remark 2, is this: Can we design a correspondence matching algorithm that better balances the number of remaining points and the number of remaining inliers? In particular, such matching should take specific pose into consideration (cf. ARCS); many methods did not.

Like TEASER++, GORE, GNC-TLS, RANSAC, our algorithm relies on an inlier threshold $c$. While how to set this hyper-parameter suitably is known for Gaussian noise with given variance, in practice the distance threshold is usually chosen empirically, as Hartley & Zisserman wrote [33]. While mis-specification of $c$ could fail the registration, certain heuristics have been developed to alleviate the sensitivity to such mis-specification; see [2, 6–8]. Finally, our experience is to set $c$ based on the scale of the point clouds.

Our outlier removal component ARCS+$_{GR}$ presented good performance (Table 3), yet with no optimality guarantees. Note that, with $s = 90$ we have $|\phi_j - \phi_j^*| \leq 1$ for some $\phi_j$, while Figure 1a shows that ARCS+$_{GR}$ gave roughly 1 degree error at $s = 90$. Theoretically justifying this is left as future work. Without guarantees, registration could fail, which might lead to undesired consequences in safety-critical applications. On the other hand, we believe that ARCS+ is a good demonstration of trading optimality guarantees for accuracy and scalability; enforcing all of the three properties amounts to requiring solving NP hard problems efficiently at large scale! In fact, since any solutions might get certified as optimal (Remark 3), bold algorithmic design ideas can be taken towards improving accuracy and scalability, while relying on other tools for optimality certification.

Acknowledgments. The first author was supported by the MINDS PhD fellowship at Johns Hopkins University. This work was supported by NSF Grants 1704458 and 1934979, and by the Northrop Grumman Mission Systems Research in Applications for Learning Machines (REALM) initiative.

---

Table 5. Success rates of methods run on the scene pairs of the 3DMatch dataset [84] for which the ground-truth rotation and translation are provided (rotation error smaller than 10 degree means a success [80]; see also the first paragraph of Appendix E).

<table>
<thead>
<tr>
<th>Scene Type</th>
<th>Kitchen</th>
<th>Home 1</th>
<th>Home 2</th>
<th>Hotel 1</th>
<th>Hotel 2</th>
<th>Hotel 3</th>
<th>Study Room</th>
<th>MIT Lab</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td># Scene Pairs</td>
<td>506</td>
<td>156</td>
<td>208</td>
<td>226</td>
<td>104</td>
<td>54</td>
<td>292</td>
<td>77</td>
<td>1623</td>
</tr>
<tr>
<td>TEASER++</td>
<td>99.0%</td>
<td>98.1%</td>
<td>94.7%</td>
<td>98.7%</td>
<td>99.0%</td>
<td>98.1%</td>
<td>97.0%</td>
<td>94.8%</td>
<td>97.72%</td>
</tr>
<tr>
<td>ARCS++$_{GR}$</td>
<td>98.4%</td>
<td>97.4%</td>
<td>95.7%</td>
<td>98.7%</td>
<td>98.1%</td>
<td>100%</td>
<td>97.3%</td>
<td>96.1%</td>
<td>97.72%</td>
</tr>
</tbody>
</table>

![Figure 2. Rotation errors of ARCS+ on synthetic Gaussian point clouds. 20 trials, $m = 10^4$, $n = [0.8m]$, $\sigma = 0.01$.](https://github.com/zgojcic/3DSmoothNet)
References


Heng Yang and Luca Carlone. A quaternion-based certifiably optimal solution to the Wahba problem with outliers. In IEEE International Conference on Computer Vision, pages 1665–1674, 2019. 1, 2


Jiaolong Yang, Hongdong Li, and Yunde Jia. Go-ICP: Solving 3D registration efficiently and globally optimally. In IEEE International Conference on Computer Vision, pages 1457–1464, 2013. 2


Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and T Funkhouser. 3DMatch: Learning the matching of local 3D geometry in range scans. In IEEE Conference on Computer Vision and Pattern Recognition, page 4, 2017. 7, 8

