MATCHING MISALIGNED TWO-RESOLUTION METROLOGY DATA

A Dissertation

by

YAPING WANG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

August 2016

Major Subject: Industrial Engineering

ABSTRACT

Multi-resolution metrology devices co-exist in today's manufacturing environment, producing coordinate measurements complementing each other. Typically, the high-resolution device produces a scarce but accurate dataset, whereas the low-resolution one produces a dense but less accurate dataset. Research has shown that combining the two datasets of different resolutions makes better predictions of the geometric features of a manufactured part. A challenge, however, is how to effectively match each high-resolution data point to a low-resolution point that measures approximately the same physical location. A solution to this matching problem appears a prerequisite to a good final prediction.

This dissertation solves this metrology matching problem by formulating it as a quadratic integer programming, aiming at minimizing the maximum inter-point-distance difference (maxIPDdiff) among all potential correspondences. Due to the combinatorial nature of the optimization model, solving it to optimality is computationally prohibitive even for a small problem size. In order to solve real-life sized problems within a reasonable amount of time, a two-stage matching framework (TSMF) is proposed. The TSMF approach follows a coarse-to-fine search strategy and consists of down-sampling the full size problem, solving the down-sampled problem to optimality, extending the solution of the down-sampled problem to the full size problem, and refining the solution using iterative local search.

Many manufactured parts are designed with symmetric features; that is, many part surfaces are invariant (are mapped to themselves) to certain intrinsic reflections and/or rotations. Dealing with parts surfaces with symmetric features makes the metrology matching problem even more challenging. The new challenge is that, due to this symmetry, alignment performance metrics such as maxIPDdiff and root mean square error are not able to differentiate between (a) correct solutions/correspondences that are orientationally consistent with the underlying true correspondences and (b) incorrect but seemingly correct solutions that can be obtained by applying the surface's intrinsic reflections and/or rotations to a correct set of correspondences. To address this challenge, a filtering procedure is proposed to supplement the TSMF approach. Specifically, the filtering procedure works by generating a solution pool that contains a group of plausible candidate sets of correspondences and subsequently filtering this pool in order to select a correct set of correspondences from the pool.

Numerical experiments show that the TSMF approach outperforms two widely-used point set registration alternatives, the iterative closest point (ICP) and coherent point drift methods (CPD), in terms of several performance metrics. Moreover, compared to ICP and CPD, the TSMF approach scales very well as the instance size increases, and is robust with respect to the initial misalignment degree between the two datasets. The numerical results also show that, when enhanced with the proposed filtering procedure, TSMF exhibits much better alignment performance than TSMF without filtering, CPD and ICP in terms of both orientation correctness of the selected solution and several other performance metrics. Furthermore, in terms of computational performance, TSMF (with and without filtering) can solve real-life sized metrology data matching problems within a reasonable amount of time. Therefore, they are both well suitable to serve as an off-line tool in the manufacturing quality control process.

To my dear husband, parents, brothers, and children

for their constant support and unconditional love.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Dr. Erick Moreno-Centeno, for his support and guidance throughout my studies at Texas A&M University. His knowledge and experience contributed many of the inspiring ideas to this dissertation. I will forever be thankful for his kind support in every aspect of my PhD study.

I am deeply grateful to my committee member, Dr. Yu Ding, who has served as my research mentor since the beginning of my PhD study and coauthor on a research paper. I greatly appreciate for his commitment and tremendous efforts on providing invaluable discussions and insightful guidance during our weekly research meeting and the writing of research paper.

I would like to extend my sincere gratitude to my two other PhD committee members, Dr. Guy Curry, and Dr. Vivek Sarin, for their constructive suggestions and support throughout the course of this research, and their their careful review of this dissertation. I also owe many thanks to Dr. Natarajan Gautam and Dr. Richard M. Feldman for their kind help and generous support during my PhD study.

I own many thanks to my lab colleague and my good friend, Dr. Adolfo Escobedo-Pinto. Adolfo is a wonderful and generous friend whom I know I can always ask for advice and suggestions. His self-discipline, perseverance, and positive outlook has inspired me to complete this journey. I also want to thank Dr. Arash Pourhabib, Mr. Payman Dehghanian, and Dr. Kisuk Sung for their friendship and many enjoyable discussions/conversations. Sincere acknowledgements are also extended to my other lab colleagues, Mr. Ahmed M. Marzouk, Mr. Eric Brown, and Mr. Christopher Lourenco for their generous assistance and feedback on my research presentations.

My time at TAMU was made enjoyable in great part thanks to many wonderful friends and ISEN staff members. I sincerely thank my friends, Dr. Su Zhao, Dr. Xue Han, and Ms. Yu Fu (too many good fiends to list here but you know who you are), for their kind help and friendship. I also thank Ms. Judy Meeks, Ms. Erin Roady, Ms. Jaime Vykukal, Mr. Mark Hopcus, and Mr. Shannon Caldwell who have been genuinely nice and helpful.

Sincere gratitude are also extended to my friend and life mentor, Ms. Yifeng Ren and Dr. Shaojun Wang for their friendship, generous help and encouragement.

I am deeply grateful to my beloved husband, Dr. Jinfeng Ren, for being a true supporter, and a best friend during both my good and bad times. He has always been there to listen, share my pain and constantly encourage me to bravely conquer every single obstacle I encountered during my PhD study. These past several years was not an easy ride. I truly thank him for accompanying me through this long journey. I also deeply grateful to my parents-in-law for their generous help and support.

I especially thank my dear parents and three elder brothers for their unconditional love and care, enormous support and encouragement. I love them so much and I would not have made it so far without them.

I also want to thank my two lovely kids, Steven and Katherine, for their pure love, patience and understanding during the time I was not able to accompany them. I hope what I have done can influence them and make them more braver to achieve their own dreams in the future.

I am also grateful that my PhD study was financially supported by the Department Graduate Teaching Assistantship, the Advanced Research Project Agency-Energy (ARPA-E) under the Green Electricity Network Integration (GENI) project "Robust Adaptive Transmission Control" and also in part by the National Science Foundation(NSF) award No. CMMI-1451036.

NOMENCLATURE

B&B	Branch-and-Bound
CCMM	Contact Coordinate Measuring Machine
CPD	Coherent Point Drift
Filter1	Filtering option 1
Filter2	Filtering option 2
Filter3	Filtering option 3
FPS	Farthest Point Sampling
greedyDownsampling	greedy Down-sampling approach
HR	High-Resolution
ICP	Iterative Closest Point
IPD	Inter-Point-Distance
LR	Low-Resolution
maxIPDdiff	maximum IPD difference
MILP	Mixed Integer Linear Programming
minMaxQIP	min-Max Quadratic Integer Program
OCMM	Optical Coordinate Measuring Machine
PCA	Principle Component Analysis
QAP	Quadratic Assignment Problem
RMSE	Root Mean Square Error
RPSR	Rigid Point Set Registration
sumIPDdiff	summation IPD difference
TSMF	Two-Stage Matching Framework
XiaHeur	Heuristic matching algorithm proposed by Xia et. al.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION*

1.1 Motivation and Research Objective

To ensure the dimensional quality of manufactured products, metrology equipment is needed to take coordinate measurements. Two lines of metrology devices co-exist today: one is the contact coordination measuring machine (CCMM) [1] with a mechanical touch probe and the other is optical coordination measuring machine (OCMM) [2] equipped with a laser scanning sensory system. Figure 1.1 illustrates a manufactured part being measured by the two metrology devices.

In this pair, CCMM is the high-resolution (HR) device which can measure up to the resolution of 0.5 μm. Comparatively, OCMM is the low-resolution (LR) one whose resolution is usually one order of magnitude lower than that of the CCMM [3]. On the other hand, OCMM, due to its use of the laser scanning mechanism, can take dense measurements from a medium to large sized part reasonably fast, say in hours, while using a CCMM on the same part may take considerably longer time, say days. Even then the CCMM measurements do not cover the part's surface as nearly dense as those by OCMM. In the end, the two resulting metrology datasets have measurements of *different resolutions* and *different surface-covering densities*. They form a pair of datasets complementing, rather than replacing, one another, as the LR data, with its dense coverage, capture the local and global shape feature better, while the HR data, albeit scarce in number, does describe by each of its data points the true yet unknown surface in a more accurate and precise manner.

*Part of the material in this section is reprinted with permission from "Match misaligned two-resolution metrology data" by Yaping Wang, Erick Moreno-Centeno and Yu Ding, 2016. IEEE Transactions on Automation Science and Engineering (In Press), manuscript no. T-ASE-2015-493, article DOI: 10.1109/TASE.2016.2587219, July/2016 ©2016 IEEE.

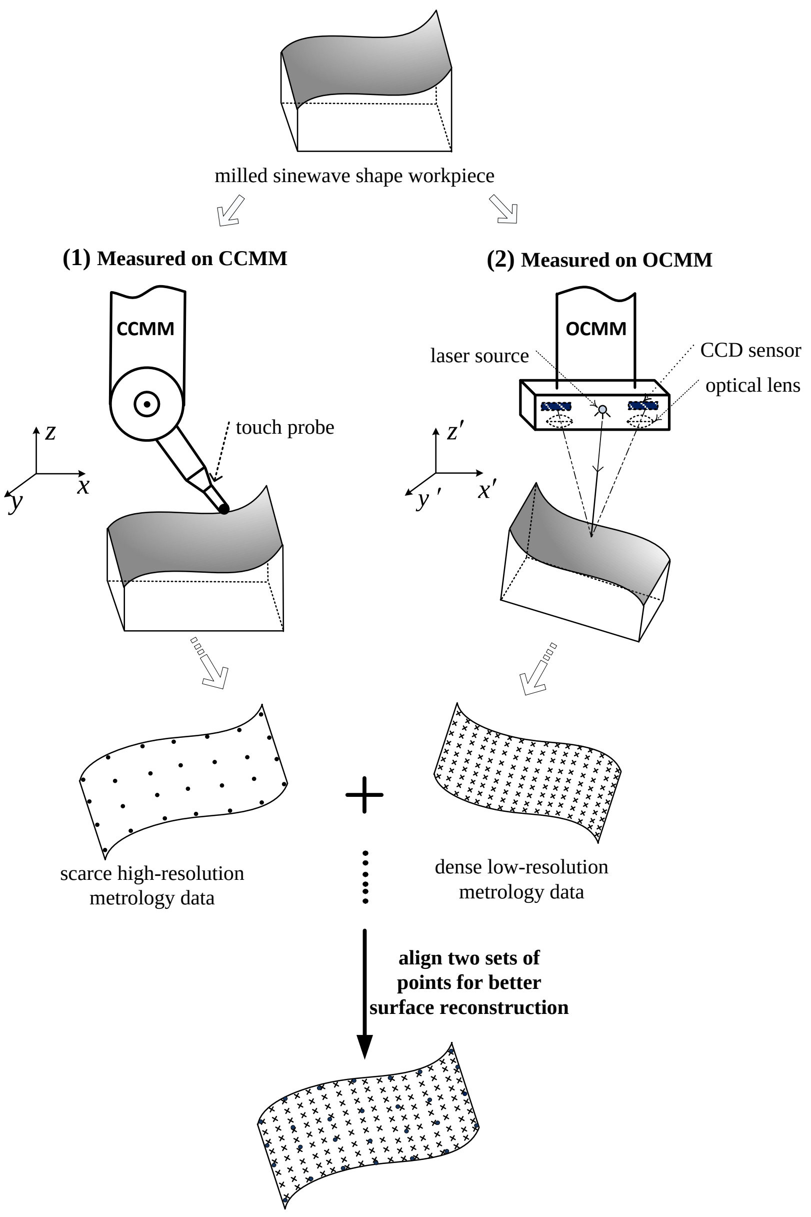

milled sinewave shape workpiece
(1) Measured on CCMM
(2) Measured on OCMM
CCMM
OCMM
laser source
CCD sensor
optical lens
touch probe
z
x
y
z′
y ′
x′
+
scarce high-resolution metrology data
dense low-resolution metrology data
align two sets of points for better surface reconstruction

Figure 1.1: Two-resolution metrology data

Researchers recognize the need and benefit of combining the two-resolution metrology datasets. For instance, Xia et. al. [4] has shown that combining the two-resolution datasets produces better prediction quality of the underlying surface feature than only using one of them. A prerequisite in achieving an effective combination is to match each HR data point to an LR data point that measures approximately the same physical location. This matching is, however, a challenge because the two datasets are often misaligned. By “misalignment”, we mean that the coordinates of a data point cannot serve as a unique reference to the physical location on the part surface where the measurement was actually taken. Given two datasets, it is not immediately clear which point in one dataset corresponds to a selected point in the other dataset. Misalignment happens, nearly inevitably, because 1) the coordinate systems used to record the measurements on CCMM and OCMM are usually different; and 2) the part is typically re-oriented between the two measuring tasks, and hence has different poses while being measured.

The objective of our research is to develop a robust algorithm, i.e., misalignment insensitive, for matching the two metrology datasets so as to lay a sound foundation that enables the neighborhood linkage model in [4] to be applied for producing better surface feature predictions.

1.2 Problem Definition

Our problem is within the class of problems referred to as rigid point set registration (RPSR) problems (a.k.a. point matching problems). Given two finite point sets $\mathcal{A}$ and $\mathcal{B}$, each on a different coordinate system, the RPSR is to find a rigid transformation and/or point-to-point correspondences that minimize(s) the misalignment between transformed point set $\mathcal{A}$ and point set $\mathcal{B}$. The term “set of correspondences” or simply “correspondences” is used here to specify a complete set of point-to-point assignment between two point sets, whereas “a pair of matching points” only specifies one single point-to-point assignment, i.e. if $a_i \in \mathcal{A}$ is matched to $b_j \in \mathcal{B}$, then, (a_i, b_j) is called a pair of matching points. Henceforth the terms *dataset* and *point* *set* are used interchangeably.

The metrology data matching problem addressed in this study is defined as follows: Given a sparse HR dataset and a dense LR dataset that are obtained by measuring the same part surface, we want to find point-to-point correspondences from the HR dataset to the LR dataset under an one-to-one (injection) function such that each HR point is matched to an LR point measuring approximately the same physical location.

Unlike a generic RPSR problem, our metrology data matching problem has the following unique characteristics: 1) Each dataset is a collection of unstructured coordinate points. Specifically, the datasets do not include any additional information concerning the nature of the points or the relationships between points, e.g., no labels, polygon mesh representation or other features like intensity or texture of the surface available. 2) The misalignment between the two datasets may be arbitrarily large. 3) The cardinality of the HR dataset is significantly smaller than that of the LR dataset, as the HR dataset has significantly lower density than the LR dataset, yet both datasets fully cover the part surface. In summary, the distinctive characteristic of our problem is the drastic density and cardinality differences between the datasets, distinguishing our problem from the RPSR problems previously addressed, including those whose datasets have no or negligible density differences [5–7], or a much smaller cardinality difference [8], or no appreciable density or cardinality difference [9].

Many manufactured parts are designed with symmetric features; that is, many part surfaces are invariant (are mapped to themselves) to certain intrinsic reflections and/or rotations. Dealing with parts surfaces with symmetric features makes the metrology matching problem even more challenging. The new challenge is that, due to this symmetry, alignment performance metrics such as maxIPDdiff and root mean square error are not able to differentiate between (a) correct solutions/correspondences that are orientationally consistent with the underlying true correspondences and (b) incorrect but seemingly correct solutions that can be obtained by applying the surface's intrinsic reflections and/or rotations to a correct set of correspondences.

1.3 Research Approach

There are three typical strategies to solve a RPSR problem:

1. Establish the point-to-point correspondences first and then recover the rigid body transformation based on the obtained correspondences (see, e.g., [10,11]). Then, with the established correspondences at hand, one can employ a closed-form least square solution (see, e.g., [12,13]) to recover the rigid transformation that optimally aligns (i.e., minimizes the L_2 distances between) the two metrology datasets.

2. First estimate the rigid body transformation, that best aligns the two datasets, and then find the correspondences (see, e.g., [14,15]). Once the two datasets are aligned by applying the estimated transformation, one can simply use a closest-point criterion to obtain the correspondences.

3. Find the transformation and the correspondences jointly (see, e.g., [5,6,16]). This strategy is generally implemented by either alternating between recovering the transformation parameters and determining the correspondences until convergence or optimizing transformation and correspondences simultaneously using a single probabilistic or optimization model.

Our solution approach follows the first strategy and intends to solve the RPSR problem by focusing on finding the point-to-point correspondences between the two datasets without the need of computing the underlying rigid body transformation beforehand. To establish the point-to-point correspondences, inspired by the matching heuristic proposed in [4], our approach makes use of the invariance property of inter-point-distance (IPD) of rigid body transformations (IPD first introduced in [17]). IPD is defined for any two points/measurements in the same dataset and is calculated as the Euclidean distance between the two points. The invariance property of IPD means the following: given any pair of physical points in the manufactured part, the Euclidean distance between the two points remains the same after applying any rigid body transformation. However, in our context, the invariance property of IPD only holds approximately because of the resolution scale difference between CCMM and OCMM and the randomness in the measurement locations due to the different measuring plans in CCMM and OCMM. Specifically, given a pair of physical points that were (approximately) measured in both datasets the distance between the pair of measurements in the first dataset should be approximately equal to the distance between the pair of measurements in the second dataset. Moreover, recall that both datasets cover the part surface evenly and the LR dataset is significantly denser than the HR one. Therefore, it is reasonable to assume each HR point has a corresponding LR point physically residing so close to it that we can deem both points represent approximately the same physical location on the part surface, and thus, they are considered to be a pair of matching points.

The invariance property of IPD allows us to compare the intrinsic pairwise distances internal to one dataset to those internal to the other dataset. To compare the internal pairwise distances (i.e. IPDs) of two pairs of matching points, with each pair in a respective dataset, we compute the IPD difference associated with these two pairs of points and use this difference as a criterion (i.e., a dissimilarity measure) to evaluate how good one pair is matched to the other pair. Let us denote by

$H = \{h_i \in \mathbb{R}^d : i = 1, \ldots, n_h\}$ the HR dataset and $L = \{l_s \in \mathbb{R}^d : s = 1, \ldots, n_l\}$ the LR data set, where $n_l \gg n_h$, and d is usually 2 or 3. Then, given two pairs of matching points (h_i, l_s) and (h_j, l_t), the associated IPD difference is $\Big| \|h_i - h_j\| - \|l_s - l_t\| \Big|$.

Our approach aims to find the best correspondences between the two datasets whose largest IPD difference is minimized. This goal is achieved by formulating our matching problem as a quadratic integer programming model (QIP) — see details in Section 3.1. However, due to the combinatorial nature of the QIP model, solving its linearized version using a general mixed integer linear programming (MILP) solver is computationally prohibitive even for a small problem size (e.g., 16 HR points and 100 LR points). Even with the help of an effective search space pruning method (discussed later in Section 3.2), it is still difficult to solve to optimality a medium-sized problem (e.g., 16 HR points and 400 LR points). Therefore, our goal is to obtain a near optimal solution for large size problems within a reasonable amount of time.

To achieve this goal, we propose a two-stage matching framework (TSMF) combining the branch-and-bound (B&B) search method and approximation algorithms. More specifically, our approach follows a coarse-to-fine search strategy, entailing the following major actions: (1) down-sample both datasets to smaller sizes; (2) find the optimal correspondences for the down-sampled problem; (3) extend the optimal correspondences of the down-sampled problem to the full datasets and find a complete set of correspondences, and (4) finally, employ an iterative local search procedure to refine this complete set of correspondences until there is no appreciable improvement.

Finally, to address the difficulty of matching metrology data for manufactured parts surfaces with symmetric features, a filtering procedure is proposed to supplement the proposed TSMF approach. Specifically, the filtering procedure works by generating a solution pool that contains a group of plausible candidate sets of correspondences and subsequently filtering this pool in order to select a correct set of correspondences from the pool.

1.4 Organization of the Dissertation

The rest of the dissertation is organized as follows. Section 2 reviews the literature that either focuses on RPSR methods that may be applied to our specific problem or that shares strong similarities with our approach. Section 3 presents our mathematical formulation for the misaligned two-resolution metrology data matching problem. Section 4 describes the details of the proposed TSMF approach and demonstrates the merits of TSMF by comparing it to two widely-used RPSR algorithms on real-life sized problem instances. As a complementary utility to the proposed TSMF approach, Section 5 proposes a filtering procedure to enhance the performance of TSMF on metrology matching problems where the part surface has symmetric features. Finally, Section 6 summarizes the dissertation and discusses some future research directions.

2. LITERATURE REVIEW*

The RPSR problem arises in many different fields, such as computer vision, image processing, pattern recognition, computational biology, etc., and has thus been extensively studied. Since a few outstanding surveys were published recently, see, e.g., [18–23], we do not intend to give another comprehensive review here. Instead, this literature review focuses on the RPSR methods that may be applied to our specific problem or that share strong similarities with our approach; most of them also take unstructured data point sets as input datasets. In this review, we categorize the RPSR solution methods into four different groups: local deterministic optimization methods, probabilistic methods, heuristic and meta-heuristic methods and global optimization methods.

2.1 Local Deterministic Optimization Methods

Local deterministic methods intend to minimize the misalignment between the datasets using local neighborhood search. The most famous method is the iterative closest point (ICP) method introduced by Besl and McKay [5]. ICP iteratively registers the two point sets by alternating between the transformation estimation and the correspondences determination. Specifically, for each data point in one data set, ICP first matches it with a point in the other data set based on the closest-point criterion, then estimates the transformation (i.e., it finds the transformation that minimizes the L2 distance between the data sets) based on the established correspondences, and then applies the estimated transformation to better algin the

*Part of the material in this section is reprinted with permission from "Match misaligned two-resolution metrology data" by Yaping Wang, Erick Moreno-Centeno and Yu Ding, 2016. IEEE Transactions on Automation Science and Engineering (In Press), manuscript no. T-ASE-2015-493, article DOI: 10.1109/TASE.2016.2587219, July/2016 ©2016 IEEE.

data sets. This three-step process is repeated until the registration error is below a pre-specified tolerance. ICP is widely used in many different RPSR applications due to its simplicity and good performance. The main shortcoming is that ICP can easily be trapped in a local minimum [7] without a good initial alignment. To circumvent this shortcoming, different variants of ICP have been proposed [24, 25]. Another drawback is that ICP does not guarantee to return a set of one-to-one correspondences [26]. More recently, Linh and Hiroshi combined ICP and nested annealing aiming to find the globally optimal alignment between the two point sets [27]—yet, the authors pointed out that this algorithm is still likely to converge to local minima.

In addition to ICP, two other local optimization methods are available. Pottmann et al. [28] proposed a registration approach using instantaneous kinematics and a local quadratic approximation of a squared distance function of a surface, which they demonstrated to have better convergence than ICP. Mitra et al. [29] used a gradient descent based optimization technique to update the rigid transformation parameters iteratively by setting the partial derivatives of the residual error to zero and solving the resulting linear systems. Even though the gradient descent method is more stable and converges faster than ICP and its variants, a good solution from the method still heavily depends on the starting position of the point sets [21]. In fact, all the aforementioned local optimization methods require more or less a good initial transformation estimation to work properly, which limits considerably their success in handling the arbitrarily large misalignment in our problem.

2.2 Probabilistic Methods

Probabilistic point matching methods can be further divided into two subgroups. The methods in the first subgroup model one or both of the datasets using a Gaussian mixture model (GMM) and cast the registration process as a maximum likelihood estimation (MLE) problem (see, e.g. [6, 30–33]). In other words, these methods aim to maximize the likelihood that one dataset fits another via an expectation-maximization (EM) algorithm. One well-known approach in this subgroup is called coherent point drift (CPD) [6]. CPD poses the RPSR registration of two datasets as a probability density estimation problem and models one dataset as GMM centroids. The best alignment is achieved by fitting the GMM centroids to the other dataset by maximizing the likelihood. CPD is able to preserve the topological structure of the point sets and can efficiently handle large size datasets for both rigid and non-rigid cases. Lu et. al. [34] proposed an accelerated CPD algorithm that can register large 3D point clouds more quickly than CPD by further accelerating the Gaussian summation process during the calculation of correspondence probability matrix of CPD. In [35], Eckart et. al. proposed a GMM-based point cloud dataset registration algorithm that applies a so-called dual-model E-M framework to achieve faster and better convergence for a wider range of initial misalignments. However, this algorithm only outperforms alternative methods when the maximum misalignment angle is less than 90 degrees and the translation is less than the length of the dataset. The methods in the second subgroup are generally known as the robust point matching (RPM) algorithms, which combine the so-called "soft assign" technique and deterministic annealing to determine the correspondences [36–38]. It has been shown [38] that the process of alternating between soft assignment of correspondences and transformation estimation is equivalent to the EM algorithm used in the first subgroup. In comparison to ICP and its variants, probabilistic methods are more robust to initial misalignment between the two datasets. However, they can still be trapped in local minima if the misalignment degree is relatively large; for instance, CPD can handle a misalignment up to 70 degrees but not greater [6]. and the accelerated CPD in [34] was only tested for handling misalignment up to 25 degrees.

2.3 Heuristic and Meta-Heuristic Methods

The third group of methods use either heuristic or meta-heuristic algorithms to find the correspondences or to estimate the transformation parameters. Xia et. al. [4] proposed a fast heuristic matching algorithm, referred to as XiaHeur hereafter, which is based on the IPD invariance property of rigid body transformations. Specifically, XiaHeur first randomly selects one HR point as anchor point and then provisionally matches it to an LR point to form an anchor pair. With this anchor pair, XiaHeur matches the remaining HR points, one at a time; specifically, XiaHeur matches each HR point with an unmatched LR point that results in the the smallest IPD difference between these newly formed matching pair and the anchor pair. Once all the remaining HR points have been matched to an LR point, we obtain one provisional set of correspondences. After this, XiaHeur matches the HR anchor point to the next LR point to form a new anchor pair and repeats the process of matching the remaining HR points. This is done until all LR points have been tried to form an anchor pair with the HR anchor point. The final set of correspondences, chosen among all the obtained provisional sets of correspondences, is the set of correspondences with the smallest maximum IPD difference. The pseudo-code of XiaHeur is included in Appendix A. Being a heuristic algorithm, XiaHeur is fast and easy to execute but does not control the resulting maximum IPD difference in each provisional set of correspondences. Indeed, as shown in the computational results in Section 4, it produces a poor set of correspondences with quite a sizeable maximum IPD difference.

As for meta-heuristic algorithms, both genetic algorithms (GA) and simulated annealing (SA) are popular choices. A GA was used in [39] to find the transformation parameters that minimize a modified Hausdorff distance between two sets of extracted image features. However, before performing the actual alignment for two images, they resort to a preprocessing step of extracting the feature points from both images based on the global and local curvatures, which can not be done with only unstructured unlabeled point coordinates in our problem setting. While in [8], GA was employed to find good correspondences for free-form surfaces and then the transformation is found using least square fitting method. SA was used in [40], in conjunction with ICP, to deal with two partially overlapping datasets; specifically, SA was used to alleviate the local-optimal-entrapping shortcoming of ICP, while ICP was used to speed up SA. Since this hybrid approach relies on the dead reckoning technique [41] to obtain a coarse position estimation, its applicability is limited.

All the GA or SA based methods mentioned above can avoid being trapped to a poor local optima to some extent, but still not guaranteed to reach a global optima eventually.

2.4 Global Optimization Methods

Global optimization methods cast the RPSR problem as a global mathematical optimization model and aim to align data point sets with any initial misalignment. This group of methods intend to find either an optimal global solution through a branch-and-bound (B&B) based approach or a practical near-to-optimal solution by combining the B&B approach and some approximation algorithms. Li et. al. [42] presented a method based on the B&B search to globally register two given 3D images. This method is not applicable to our problem because it assumes equal sizes of the two sets and no translation between them. Gelfand et. al. [11] proposed a method for registering 3D shapes based also on the pair-wise distance consistency (i.e., the IPD invariant property), but this approach relies on strong distinctive features of the input shapes to perform well. The work presented in [43] and [7] employed B&B

for image matching applications. However, both methods are specialized for 2D datasets, and it is not a trivial task to generalize them to 3D cases. For instance, extending the geometric B&B method in [7] to higher dimension will considerably enlarge the search space for the transformation function. Moreover, the method in [7] only deals with data point sets with equal size that is up to relatively small size of 100. Raviv et. al. [10] proposed a non-rigid registration method for 3D shapes that shares a similar coarse-to-fine matching strategy to our approach (elaborated in Section 4). The method in [10] has two limitations, however, making it not suitable for our problem: 1) it requires the datasets to have mesh structure (smooth geometric measure); 2) its exact coarse matching model can only handle point sets with the same cardinality (see constraint (3.5) in [10]); and 3) the method in [10] gradually add unmatched points from neighborhood during each refining iteration, while our approach extends an exact coarse partial set of correspondences to a complete set in one step and then iteratively refine it via a local search procedure. Recently, Brown et. al. proposed a B&B-based globally optimal 2D-3D registration algorithm [44]. But, this algorithm relies on features not found on unstructured 3D cloud points.

To evaluate the performance of TSMF, we choose one representative algorithm from each of the first three groups and compare them to our proposed TSMF, as the methods in the last group are not applicable to our problem. In the first group, ICP is selected due to its popularity and good performance. In the second group, CPD is chosen because of its robustness compared to ICP and ICP's variants. XiaHeur is selected from the third group because it is fast and simple, and thus, is most likely adopted in industrial practice. It is worth pointing out that CPD and ICP algorithms are not randomized algorithms; they are deterministic algorithms. Even though CPD models the rigid registration problem as a probability density probability problem, the Expectation Maximum algorithm to solve the registration problem is deterministic. Thus, running ICP multiple times on the same instance will always produce the exact same solution (the same is true for CPD).

3. PROBLEM FORMULATION*

This section presents the mathematical formulation for the misaligned two-resolution metrology matching problem. We first introduce a quadratic integer programming formulation and then briefly describe its linearized version. Last, we introduce an effective search space pruning technique that can help greatly decrease the computational time of solving the linearized optimization problem.

3.1 Quadratic Integer Program and its Linearization

Given that the invariance property of IPD holds for our problem, we formulate the misaligned metrology data matching problem as a min-max quadratic integer program (*minMaxQIP*). We first introduce a few notations. Denote by d_{ij}^H the IPD between point h_i and point h_j in the HR dataset, i.e., $d_{ij}^H = \|h_i - h_j\|$. The IPD for the LR dataset, d_{st}^L, is likewise defined. Denote by x_{is} the binary assignment variable, such that $x_{is} = 1$ if h_i is matched to l_s, and $x_{is} = 0$ otherwise. As such, the *minMaxQIP* formulation is as follows.

*Part of the material in this section is reprinted with permission from "Match misaligned two-resolution metrology data" by Yaping Wang, Erick Moreno-Centeno and Yu Ding, 2016. IEEE Transactions on Automation Science and Engineering (In Press), manuscript no. T-ASE-2015-493, article DOI: 10.1109/TASE.2016.2587219, July/2016 ©2016 IEEE.

$$\min_{x} \max_{\substack{i,j=1,\ldots,n_h \\ s,t=1,\ldots,n_l}} \left|d_{ij}^{H} - d_{st}^{L}\right| x_{is} x_{jt}, \tag{3.1}$$

$$\textbf{s. t.} \quad \sum_{s=1}^{n_l} x_{is} = 1, \quad i = 1, \ldots, n_h; \tag{3.2}$$

$$\sum_{i=1}^{n_h} x_{is} \leq 1, \quad s = 1, \ldots, n_l; \tag{3.3}$$

$$x_{is} \in \{0, 1\}, \quad i = 1, \ldots, n_h; s = 1, \ldots, n_l. \tag{3.4}$$

The objective here is to minimize the maximum IPD difference (referred to as *maxIPDdiff* hereafter) across all potential correspondences between the two datasets. Constraint (3.2) ensures that each HR point is matched to exactly one LR point. Constraint (3.3) forces an LR point to be assigned to at most one HR point. Constraints (2) and (3) together make sure that the whole HR set is matched to a subset of the LR set under an one-to-one (injective) function.

MinMaxQIP is mathematically equivalent to a min-max version of the quadratic assignment problem (QAP) [45], which is proven to be NP-hard [46] and considered indeed one of the hardest combinatorial optimization problems. The state-of-the-art exact algorithms for QAP can only solve problems with up to 35 facilities [47], which is equivalent to 35 HR points and 35 LR points in our context. For manufacturing applications, we need an approach that can solve much larger instances (e.g., an HR dataset size of about 100 and an LR dataset over 1,000) within a reasonable amount of time. To address the challenge brought forth by the larger problem size, we devised a coarse-to-fine matching strategy such that we only need to solve a much smaller size of the minMaxQIP problem to optimality, where the much smaller minMaxQIP problem is referred to as the down-sampled problem.

To prepare for our solution procedure, we linearize the minMaxQIP model. First, we define new binary variables z_{isjt} to replace the quadratic term $x_{is}x_{jt}$ in the objective function, and add (3.6) to ensure that z_{isjt} is 1 when both x_{is} and x_{jt} are 1. Then, we change the original min-max objective function to a minimization one by defining a new continuous variable u to replace the inner maximization, i.e., $\mathbf{max}\left|d_{ij}^{H} - d_{st}^{L}\right| z_{isjt}$. To reflect that u is the maximum over all combinations of i, j, s, t, constraint (3.7) is added, which says that the maximum over all possible terms is greater than or equal to every one of individual terms. The linearized model is given below.

$$\min_{x} \quad u, \tag{3.5}$$

$$\textbf{s. t.}\ (2-4),$$

$$x_{is} + x_{jt} \leq z_{isjt} + 1, \quad i, j = 1, ..., n_h;\ s, t = 1, ..., n_l;\ i < j, s \neq t; \tag{3.6}$$

$$u \geq \left|d_{ij}^{H} - d_{st}^{L}\right| z_{isjt}, \quad i, j = 1, ..., n_h;\ s, t = 1, ..., n_l;\ i < j, s \neq t; \tag{3.7}$$

$$z_{isjt} \in \{0, 1\}, \quad i, j = 1, ..., n_h;\ s, t = 1, ..., n_l;\ i < j, s \neq t. \tag{3.8}$$

3.2 Search Space Pruning Technique

Before delving into the details of our proposed two-stage solution approach in next section, we present that a simple search space pruning can greatly reduce the solution time of the linearized optimization problem.

Let u^* be the optimal solution (maxIPDdiff) of the linearized model. The key of the search space pruning is to find a tight upper bound for u^*, which we call the search space threshold, denoted by $\overline{T}$. That is, $\overline{T}$ is a value that we know for sure is larger than u^*, but we hope is not much larger than u^*. Consequently, one can reject all possible pairwise correspondences whose IPD difference is greater than $\overline{T}$. Specifically, given two HR points (h_i, h_j) and two LR points (l_s, l_t), if the IPD difference between them is greater than $\overline{T}$, then only one HR point can be matched to one of the two LR points. This is to say, if h_i is matched to l_s, then h_j cannot be matched to l_t, or vice versa.

Finding a proper $\overline{T}$ is essential for the search space pruning technique to work effectively. $\overline{T}$ should be as small as possible to effectively eliminate sufficient amount of the potential pairwise correspondences. But it cannot be too small; otherwise it may block off the underlying optimal solution; that is, it may render the pruned model infeasible — in which case one would need to increase $\overline{T}$ and resolve the model.

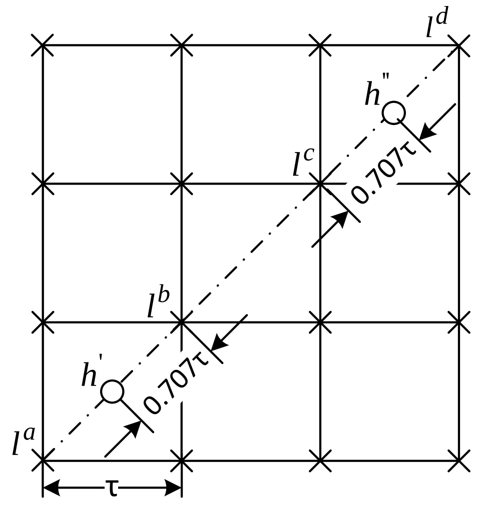


Figure 3.1: Ideal case for choosing a proper $\overline{T}$

To find an effective and safe $\overline{T}$, we consider an ideal situation where measurements in both datasets are perfectly evenly spaced over a flat part surface. A small section of a hypothetical part surface under this ideal situation is shown in Figure 3.1, where each cross represents an LR point and each circle stands for an HR point, and τ denotes the maximum distance between an LR point and its closest neighbor in the LR dataset. As shown in Figure 3.1, to estimate the largest possible IPD difference, we examine the worst case scenario where every HR point sits almost at the center of its closest four surrounding LR points, and HR point h' (h'') sits a little bit closer to LR point l^b (l^c) than to LR point l_a (l^d). As such, the HR points h' and h'' should be matched to the LR points l^b and l^c, respectively, and the IPD difference between this two pairs of matching points is 1.414τ. Under the ideal case, this 1.414τ is approximately the largest value u^* can take, as one can imagine that no matter where we move the HR points, the IPD difference is likely to get no greater. This understanding suggests that $\overline{T}$ can be set to 1.414τ. In practice, of course, the datasets are not perfectly evenly spaced and the part surfaces are usually curved. Consequently, 1.414τ may not be an upper bound of u^*. We believe that this value still represents an effective threshold. To be safer, we relax $\overline{T}$ to 1.5τ. Our later numerical analysis in Section 4 shows that this search space pruning technique on average eliminates almost 80% of binary variables and never yielded an infeasible pruned model.

It should be noted that, in this paper, we assume that the measurements in both HR and LR datasets are evenly spaced over the part surface. This assumption is realistic because the evenly spaced measurements can be readily obtained using today's metrology technology [4]. Said this, we acknowledge that there might be circumstances where taking evenly measurements throughout the surface may not be desirable. For example, if the surface is very wiggly, it may be preferred to take denser measurements near the locations with high curvature than other relative flatter areas so that the critical surface features are captured without an undue increase of measurements (especially in the HR dataset). Under these circumstances, it is desirable and practical to maintain the measurements' evenness only locally (with higher density measurements evenly distributed over the high curvature areas and lower density measurements evenly distributed over the not very curvy locations). The Appendix B explains in detail why our choice of $\overline{T} = 1.5\tau$ is also appropriate under this circumstance.

To implement the search space pruning technique, for two pairs of potential matching points (h_i, l_s) and (h_j, l_t), we do not define variable z_{isjt} if their IPD difference is greater than $\overline{T}$. To mathematically reflect this in the linearized model (i.e. equation (2)-(8)), we include constraint (3.9) to the linearized model and change (3.6) to (3.10) as below.

$$
\begin{aligned}
& x_{is} + x_{jt} \leq 1, \qquad i,j = 1,\ldots,n_h;\ s,t = 1,\ldots,n_l; \\
& \qquad\qquad i < j;\ s \neq t;\ \text{if } \left|d_{ij}^H - d_{st}^L\right| > \overline{T}, \qquad (3.9) \\
& x_{is} + x_{jt} \leq z_{isjt} + 1,\ i,j = 1,\ldots,n_h;\ s,t = 1,\ldots,n_l; \\
& \qquad\qquad i < j;\ s \neq t;\ \text{if } \left|d_{ij}^H - d_{st}^L\right| \leq \overline{T}. \qquad (3.10)
\end{aligned}
$$

A nice property of the pruning technique is that the optimal objective value of the pruned model is independent of the value of $\overline{T}$ in the following sense. If $\overline{T} < u^*$ then the pruned model is infeasible (and thus one would need to increase $\overline{T}$ and resolve the model); while if $\overline{T} \geq u^*$ then the optimal objective value of the pruned model will be equal to u^*. To see this, note that the pruning technique only eliminates the pairwise correspondences (corresponding to the binary variables z_{ijst} of the linearized model) whose IPD differences are greater than $\overline{T}$. In other words, only sub-optimal solutions are discarded. Therefore, as long as $\overline{T} \geq u^*$, the pruned model has the same optimal objective value as the unpruned model.

4. TWO-STAGE MATCHING FRAMEWORK*

Even though the search space pruning technique significantly decreases the solution time of small instances, it does not do so sufficiently to medium-to-large instances. Thus, in order to tackle problems with real-life sizes, we relax our optimization goal from solving to optimality to finding a robust, near-optimal solution and devise a two-stage matching framework (TSMF) to accomplish this relaxed goal.

We start with an overview of our solution framework. Our solution approach is conducted in two stages and each stage comprises two steps. The first stage of TSMF aims to obtain the optimal correspondences for a subset of the HR and LR data points and its steps are: 1) down-sample both datasets; 2) find the optimal correspondences for the down-sampled problem by solving it to optimality using B&B.

The second stage of TSMF extends the partial set of correspondences (i.e. the optimal correspondences for the down-sampled problem) found at the first stage to the original problem; its two steps are: 1) extend the partial set of correspondences of the down-sampled problem to a complete set of correspondences on the full datasets (i.e. find LR correspondences for the HR points that were not in the down-sampled HR dataset); 2) refine the complete set of correspondences through an iterative local search until there is no appreciable improvement. Figure 4.1 summarizes the proposed framework.

*Part of the material in this section is reprinted with permission from "Match misaligned two-resolution metrology data" by Yaping Wang, Erick Moreno-Centeno and Yu Ding, 2016. IEEE Transactions on Automation Science and Engineering (In Press), manuscript no. T-ASE-2015-493, article DOI: 10.1109/TASE.2016.2587219, July/2016 ©2016 IEEE.

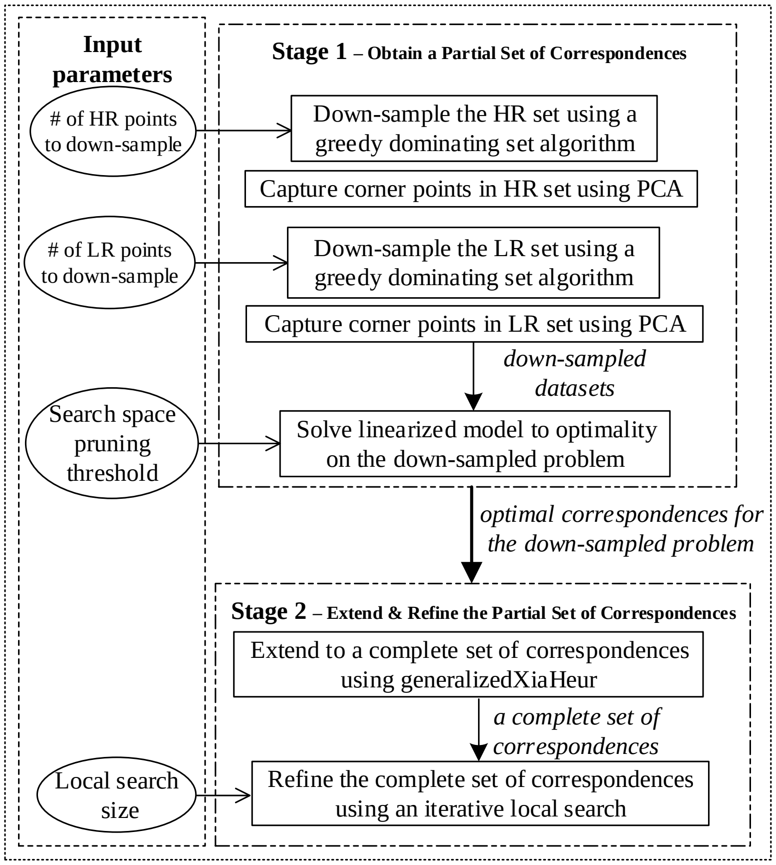

Input parameters
# of HR points to down-sample
# of LR points to down-sample
Search space pruning threshold
Local search size
Stage 1 – Obtain a Partial Set of Correspondences
Down-sample the HR set using a greedy dominating set algorithm
Capture corner points in HR set using PCA
Down-sample the LR set using a greedy dominating set algorithm
Capture corner points in LR set using PCA
down-sampled datasets
Solve linearized model to optimality on the down-sampled problem
optimal correspondences for the down-sampled problem
Stage 2 – Extend & Refine the Partial Set of Correspondences
Extend to a complete set of correspondences using generalizedXiaHeur
a complete set of correspondences
Refine the complete set of correspondences using an iterative local search

Figure 4.1: Flowchart of two-stage matching framework

Remark: If there are no down-sampled LR points close to an HR point, then the solve-to-optimality step may not give a satisfactory partial set of correspondences, which eventually lead to a poor matching solution for the original problem. Our proposed TSMF approach has one mechanism to prevent this issue from happening and one mechanism to remedy the issue if it happens. 1) Prevention mechanism: the proposed greedyDownsampling algorithm in Part A of Subsection IV is devised to prevent the issue by making the down-sampled dataset spread out on the surface as evenly as possible. The evenness makes it very likely that each down-sampled HR point has a down-sampled LR point reasonably close to it. Indeed, our computational experiments results also confirmed that, for all instances, each point in the down-sampled HR dataset has a down-sampled LR point reasonably close to it. 2) Remedy mechanism: the iterative local search procedure is designed to refine (and correct) any coarse correspondence due to the imperfect results of the down-sampling process.

4.1 First Stage - Obtaining a Partial Set of Correspondences

4.1.1 Step 1: Down-sampling Both Datasets

When down-sampling both datasets, we have two objectives: a) that the optimal correspondences of the down-sampled problem are close to the optimal correspondences for the full datasets; b) that the sizes of down-sampled sets should be small enough so that the down-sampled problem can be efficiently solved to optimality using a general MILP solver. To achieve these objectives, a down-sampling algorithm needs to meet two requirements: a) the down-sampled points should be nearly evenly spread over the part surface; b) the resulting down-sampled set should contain a desired number of points.

To fulfill the two requirements, we propose a greedy down-sampling approach, called *greedyDownsampling*, that combines the dominating set method and principal component analysis (PCA). Specifically, the final down-sampled set comprises a set of dominating points returned by the dominating set method and the corner points detected by PCA. Note that the greedyDownsampling approach is applied to each of the two datasets in the same manner. Next, we present the details of the two components of the greedyDownsampling approach.

The idea behind the dominating set method is as follows: If each data point is either part of the sampled set, or very close to a data point in the sampled set, then the set of sampled points is guaranteed to spread evenly over the part surface.

This is because the full dataset is evenly spaced over the part surface. This idea can be implemented by solving the minimum dominating set problem on an undirected graph $G = (V, E)$ appropriately constructed on the full dataset. A dominating set is a subset D of V such that every vertex not in D is adjacent to at least one vertex in D. The minimum dominating set problem is to find a dominating set with minimum cardinality. For our purposes, $G = (V, E)$ is constructed as follows: Vertex set V comprises all data points in the full dataset, and there is an edge between each data point and other points residing within a certain distance of it. We denote this distance by R_n. Building G in this way guarantees evenness of the down-sampled set (i.e. the dominating set).

The minimum dominating set problem is NP-hard [48]. Yet, for our purposes, using a greedy algorithm to find an approximate solution is good enough. The greedy algorithm starts with an empty dominating set D and iteratively appends to D the vertex $v \in V$ with the maximum degree and updates G by removing that newly added point and all vertices adjacent to it until G becomes empty. Since both datasets are arbitrarily indexed for identification, for both datasets, the algorithm always selects the median point as the first dominating point so that the two separate down-sampling process (one for each dataset) start approximately from the same physical location of the part surface; and we choose the vertex with the smallest index to break ties when there is more than one vertex having the maximum degree.

Recall that the second requirement of our down-sampling approach is to obtain a desired number of points from the full dataset. To achieve this, one needs to set R_n such that the greedy algorithm returns a dominating set of the desired size or very close to the desired size. Since the dominating set's cardinality increases monotonically as R_n decreases, to find the proper R_n, one can simply do a binary search over a plausible range of R_n. A safe initial range for R_n is between zero and a half of the longest between-point Euclidean distance in the respective dataset. The binary search procedure starts with R_n taking the middle value of the initial range and uses it to construct graph G. With G constructed, our procedure checks if the cardinality of the returned dominating set is close enough to the desired number of down-sample points (say within 5%). If it is, the binary search stops; otherwise, (a) if the dominating set's cardinality is smaller than the desired number, the binary search continues on the lower half of the current R_n range and decreases the current R_n to the midpoint of this lower half range, or, (b) if the dominating set's cardinality is larger than the desired size, the binary search continues on the upper half of the current R_n range and increases R_n to the midpoint of this upper half range.

The other component in the greedyDownsampling approach is PCA, which is used to compensate the dominating set method for its tendency not to include the edge/corner points. Specifically, we use the first two principal components of the dataset to obtain four corner points, two for each principal component. To get the first two corner points, we project the full dataset to the first principal component and choose the two points whose projection is farthest apart. The other two points are obtained similarly using the second principal component.

4.1.2 Step 2: Solve the Down-sampled Problem to Optimality

After down-sampling both datasets, we find the optimal solution (i.e., a partial set of correspondences for the full datasets) for the down-sampled sets by solving the linearized (minMaxQIP) model to optimality. This is done by using a general MILP solver but we take advantage of the search space pruning technique described in Section 3.2.

This step ensures that each point in the down-sampled HR set is matched to its best LR correspondence in the down-sampled LR dataset. The found correspondences is of course only a partial set of the full correspondences, but having it creates a good basis for improvement in the second stage.

4.2 Second Stage - Extend the Partial Set of Correspondences and Refine the Complete Solution

Once the first stage is completed, each of the two datasets can be thought of having two subsets, the matched subsets and the unmatched subsets comprising the remaining data points, namely that $H = H^{(matched)} \cup H^{(unmatched)}$ and $L = L^{(matched)} \cup L^{(unmatched)}$, so that each point in $H^{(matched)}$, there is a point in $L^{(matched)}$ that is matched to it. Our objective in the second stage is to find a point correspondence in $L^{(unmatched)}$ for every point in $H^{(unmatched)}$, conditioned on the partial set of point correspondences that have already been formed between $H^{(matched)}$ and $L^{(matched)}$.

The existence of a set of matched pairs between the HR and LR datasets in fact provides a set of anchor pairs, to borrow the term from XiaHeur. It motivates us to follow the idea of that heuristic to match the remaining HR data points to their LR counterparts. Acknowledging that XiaHeur is not robust in its matching outcome, the two steps in this stage are devised to safeguard the solution quality.

4.2.1 Step 1: Generalizing XiaHeur by Using Multiple Anchor Pairs

Through our investigation, we found that using the plain version of XiaHeur is not robust because it heavily relies on a single anchor pair. To see this, consider the example in the left panel of Figure 4.2. In the upper subfigure, there are two HR data points, illustrated by a solid circle and a solid triangle, respectively, while in the bottom subfigure, there is a group of LR datapoints, one illustrated by a solid circle and the rest illustrated by crosses. The single anchor pair comprises the two solid circles, denoted by h^{a_1} and l^{a_1}, respectively. The solid triangle point, named h^u, is the unmatched HR point. With one anchor pair (h^{a_1}, l^{a_1}), there are multiple plausible LR points that could be matched to h^u. As illustrated in the left panel of Figure 4.2, when considering a degree of measurement uncertainty up to Δ, all LR points residing within the two dashed circles could have the same merit to be matched to h^u. Some solutions could even appear on the opposite direction relative to l^{a_1}, as compared to that between h^u and h^{a_1}.

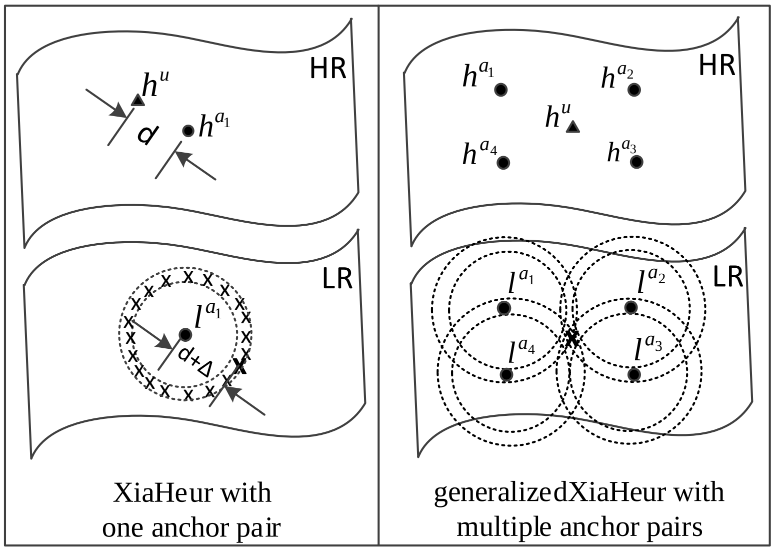


Figure 4.2: Advantage of using multiple anchor pairs

In this step, we propose a generalized version of XiaHeur, called *generalizedXiaHeur*, to overcome this drawback of XiaHeur. The major change made in generalizedXiaHeur is to use the multiple anchor pairs—those formed in the first stage of our solution framework. Specifically, for an unmatched HR point h^u, generalizedXiaHeur finds its matching point in the LR dataset such that the largest IPD difference between this new matching pair and each of the anchor pairs is minimized; see the illustration in the right panel of Figure 4.2. Through extensive numerical studies, we believe that the generalizedXiaHeur provides a remarkably robust match outcome, even in the presence of measurement noises in the datasets.

4.2.2 Step 2: Iterative Local Search

The generalizedXiaHeur, albeit its robust performance, is still a heuristic, thus leaving room for further improvement. Thus, we propose to use an iterative local search procedure to refine the complete set of correspondences obtained by generalizedXiaHeur.

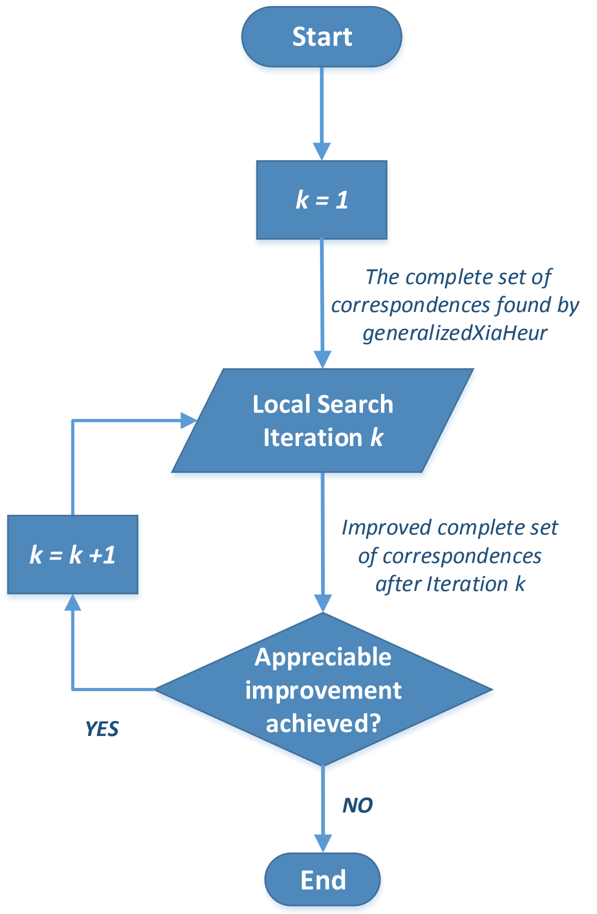

Start
k = 1
The complete set of correspondences found by generalizedXiaHeur
Local Search Iteration k
Improved complete set of correspondences after Iteration k
k = k +1
Appreciable improvement achieved?
YES
NO
End

Figure 4.3: Iterative local search procedure

The iterative local search procedure comprises a sequence local search iterations. As illustrated in Figure 4.3, each local search iteration aims to find a better set of correspondences than the best set of correspondences found so far. Moreover, such searches are limited to the sets of correspondences that are “close/local” to the current best set of correspondences. The sequence of local search iterations terminates when there is no appreciable improvement. Note that the input for each local search is the current best set of correspondences; specifically, the input for the first local search is the set of correspondences found by generalizedXiaHeur and the input for each subsequent local search is the set of correspondences found by the preceding local search iteration. The reminder of this section explains one local search iteration.

The basic idea of (one iteration) local search is illustrated in Figure 4.4 where each dot in the top sinewave represents an HR point and each cross in the bottom sinewave stands for an LR point. In the input of the local search, each HR point is matched to an LR point (denoted by a bold cross). During the local search, each HR point is allowed to be re-matched to any LR point in the neighborhood of that HR point's current LR correspondence (the crosses within the circle centered at the respective current LR correspondence).

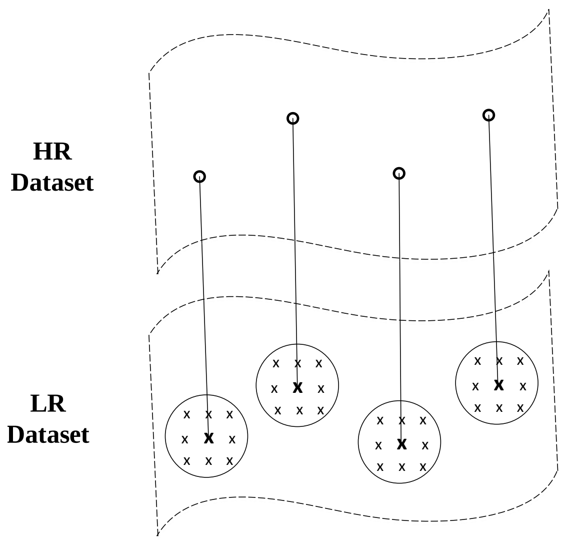


Figure 4.4: Local search illustration for one iteration

Given a neighborhood size, the local search can be done by solving a modified linearized minMaxQIP model, called *local search model*. Specifically, the local search model is very similar to the linearized minMaxQIP model (eqns. (3.5) to (3.8)) except that, in the local search model, each HR point can only be matched with one of the LR points in the neighborhood of that HR point's current LR correspondence. We do not give the local search model explicitly because we do not solve it directly but solve it as described in the reminder of this subsection.

Solving the local search model to optimality is very computationally expensive for real-life sized problems, even when using small neighborhood sizes and even after applying the search space pruning technique. In contrast, a MILP solver is very fast in determining the feasibility of the local search model after applying to it a search space pruning threshold, called $\overline{T}_{LS}$. This large complexity difference is because one can greatly simplify the local search model if one is only interested in checking its feasibility. Specifically, to check the feasibility of the model one can drop the objective function, previously u, (equivalently, set it to a constant in the MILP solver, say zero), and consequently eliminate all the constraints related to u as they are not needed for the feasibility check. With these changes, we give below the *feasibility-check model* that one needs to solve to determine the feasibility of the local search model given a specific $\overline{T}_{LS}$.

$$\begin{aligned}
&\textbf{min} \quad 0 \\
\textbf{s. t.} &\sum_{s \in NH_i} x_{is} = 1, \quad i = 1, ..., n_h; && (4.1)\\
&x_{is} = 0, \qquad i = 1, ..., n_h; s \in \{1, ..., n_l\} \setminus NH_i && (4.2)\\
&\sum_{i=1}^{n_h} x_{is} \leq 1, \qquad s = 1, ..., n_l; && (4.3)\\
&x_{is} + x_{jt} \leq 1, \quad i, j = 1, ..., n_h;\ s \in NH_i;\ t \in NH_j;\ if\ \left| d_{ij}^H - d_{st}^L \right| > \overline{T}_{LS} && (4.4)\\
&x_{is} \in \{0, 1\}, \qquad i = 1, ..., n_h;\ s = 1, ..., n_l. && (4.5)
\end{aligned}$$

In this model, NH_i denotes the set of LR points in the neighborhood of the i-th HR point's current LR correspondence; constraint (4.1) forces each HR point, say point h_i, to be matched to exactly one of the LR points in NH_i; constraint (4.2) ensures that an HR point, say point h_i, is not matched to an LR point outside NH_i; constraint (4.3) guarantees that each LR point is only matched to at most one HR point; and constraint (4.4) excludes all correspondences whose maxIPDdiff is greater than $\overline{T}_{LS}$. (To further expedite the feasibility check, one can properly set a parameter in the MILP solver to emphasize feasibility over optimality; in CPLEX, this is to set parameter *MIPEmphasis* to 1.)

Next we explain how to find the local search model's optimal solution by taking advantage of the MILP solver's efficiency to solve the feasibility-check model. Note that the feasibility-check model is feasible if and only if the applied $\overline{T}_{LS}$ is greater than or equal to the maxIPDdiff of the local search model's optimal solution. Therefore, as explained below, one can perform a binary search over $\overline{T}_{LS}$ in order to find the optimal solution to the local search model.

In the binary search, the initial range of $\overline{T}_{LS}$ is between zero and the maxIPDdiff of the best set of correspondences found so far. The binary search starts with $\overline{T}_{LS}$ taking the midpoint of its initial range and solves the feasibility-check model built using that $\overline{T}_{LS}$. If the model is feasible, then the binary search continues on the lower half of $\overline{T}_{LS}$'s current range and decreases $\overline{T}_{LS}$ to the midpoint of that lower half range; otherwise, the binary search continues on the upper half of $\overline{T}_{LS}$'s current range and increases $\overline{T}_{LS}$ to the midpoint of that upper half range. The binary search proceeds until the range length is within a predefined tolerance, say 0.01.

4.3 Computational Experiments and Results

This section describes the experimental setup and compares the performance of TSMF to that of widely-used point set registration methods: ICP and CPD. For this purpose, we used two 3D metrology datasets of a milled sinewave surface. All experiments were done on a Linux (CentOS 5.4) machine with Intel E5-1620 3.4GHz processor and 32GB RAM.

4.3.1 Experimental Setup

4.3.1.1 Datasets

The HR and LR 3D metrology datasets (see Figure 1.1) were obtained from a manufactured part of size 101×101×51 (mm), measured by a CCMM (*Sheffield Discovery II D-8* with a TB 20 touch probe) and a OCMM (*LDI Surveyor DS-2020* with a RPS 150 laser unit), respectively. The resolutions of CCMM and OCMM are roughly 5 μm and 50 μm, respectively. The two datasets were originally obtained by the study in [4], and each dataset consists of 1,560 data points that are evenly spaced over the surface. The typical number of data points collected by a CCMM over this size of product is usually an order, or orders, of magnitude fewer than that collected by an OCMM. The reason that the study of [4] collected the same number of data points in the HR set as in the LR set is because the study needed the additional HR data points for validation purposes. In fact, the largest number of data points used as the HR set in [49] is 80, and the remaining 1,480 HR points were used to assess the quality of the combined prediction made by their proposed model. In this research, we believe it is practical to increase the HR data points slightly but not substantially more. So, we chose 100 as the maximum number of points in HR set, while using all the 1,560 LR points.

To test the effectiveness and scalability of TSMF, we generated six instances as test cases of various sizes. Smaller sized datasets were created through thinning two original datasets. Sizes of all six instances are listed in the top row of Table 4.1. Each instance size is indicated by its name which comprises two parts: the number before "×" denotes the cardinality of the HR dataset, whereas the number after "×" denotes the cardinality of the LR dataset. The original LR dataset is plotted in Figure 4.5.

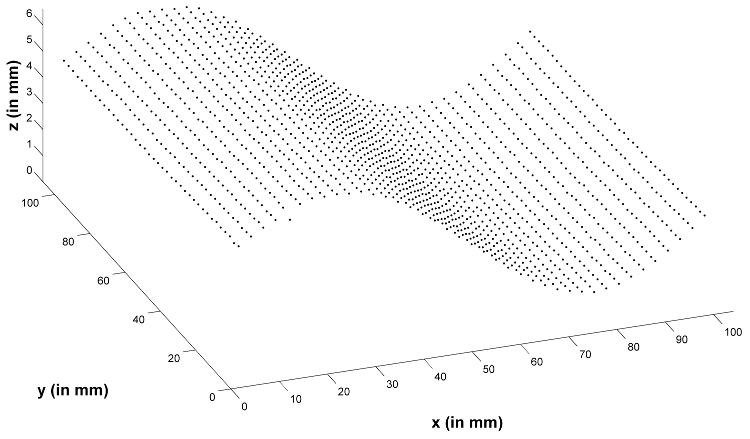


Figure 4.5: Plot of the low-resolution sinewave data

4.3.1.2 TSMF Settings and Implementation

There are four key parameters used in TSMF: (1) the search space pruning threshold ($\overline{T}$); (2) the neighborhood size of the iterative local search step in the second stage; and (3-4) the down-sampled sizes of the HR and LR datasets.

As mentioned in Section 3.2, $\overline{T}$ is chosen to be 1.5 times the maximum distance between an LR point and its closest neighbor in the LR dataset. The neighborhood size of the local search is set to 10; this provides a good balance between the search size and the time required to solve each local search.

To set the last two algorithmic parameters—the down-sampled sizes of the HR and LR datasets—we conducted extensive experiments and determined that the largest down-sampled problem size that a MILP solver can solve to optimality within a couple of minutes is roughly 10 HR points by 180 LR points. In the meanwhile, we also observed that, for each HR dataset, eight data points are enough to form a good anchor set leading to an effective generalizedXiaHeur step.

For the above reasons, we chose to down-sample every HR dataset into eight HR points: four using PCA and four using the dominating set algorithm. In contrast, the sizes of the down-sampled LR datasets were proportional to the sizes of the respective original LR dataset. Specifically, we down-sampled the largest LR dataset into 180 points—four using PCA and 176 using the dominating set algorithm. Together with the eight down-sampled HR points, this formed a combined set of 8×180 points, which is in the ballpark of the problem size that a MILP solver can solve to optimality in a desirable duration. Note that, for the largest LR dataset, the dominating set algorithm chose 176 points out of the 1560 points; which is roughly 11.3%. Thus, for the remaining LR datasets, we used the dominating set algorithm to select roughly the same 11.3% of points out of the respective LR dataset; i.e. obtaining 91 and 46 points from the original LR datasets of size 800 and size 400, respectively. Table 4.1 summarizes the down-sample set sizes for each test instance.

TSMF was implemented in C++ using Concert Technology interface for the MILP solver CPLEX (version 12.4). The PCA function we used at the down-sampling step is from the Armadillo C++ linear algebra library [50].

Instance	100×1560	64×1560	50×800	32×800	25×400	16×400
HR points	8	8	8	8	8	8
LR points	180	180	95	95	50	50

Table 4.1: Desired number of points to down-sample

4.3.1.3 CPD and ICP Implementations and Settings

Multiple ICP implementations are available online. We selected the ICP code developed by Per Bergström due to its popularity. The MATLAB code can be downloaded from *http://www.mathworks.com/matlabcentral/fileexchange/12627-iterative-closest-point-method.* Since no initial starting transformation is available in our experiments for ICP, we changed the parameter *init_flag* from default value 1 to 0 to reflect this fact. All other input parameters were left as default. As we want to match the entire HR dataset to a subset of the LR dataset, when applying ICP to our data instances we treat the LR and HR datasets as *model* and *data*, respectively.

For CPD, we chose its most recent implementation code in MATLAB, available at *https://sites.google.com/site/myronenko/research/cpd.* Given the nature of our problem, we selected the rigid registration option of CPD, i.e., *opt.method = 'rigid'*. Out of nine remaining input parameters of CPD, we changed three parameters to a non-default value: 1) *opt.scale = 0* to disallow scaling in the context of a rigid body transformation; 2) *opt.corresp = 1* to compute the correspondences at end of the registration; 3) *opt.normalize = 0* to disallow dataset normalization. We do not normalize the data because doing so results in the best CPD performance for solving our problem.

4.3.2 Results and Performance Analysis

In this subsection, we conduct the following analyses:

- Evaluate the performance of greedyDownsampling.
- Show the effectiveness of search space pruning technique.
- Compare our TSMF to XiaHeur (with and without local search) and show the effectiveness of the local search.

- Evaluate TSMF's performance against ICP and CPD.

We want to note that as the density of each HR dataset is different and the cardinality ratio of the HR dataset and LR dataset in each instance is also different (thus the underlying maxIPDdiff is different for each instance), throughout this section, we report all performance metrics as a multiple of the average smallest IPD in the LR set, denoted by w. Specifically, w is calculated for each instance by averaging the distances between each LR point and its closest neighbor in the LR dataset. This allows us to compare the results across the different instances.

The first analysis is about the performance of the greedyDownsampling method. We compare it with an down-sampling alternative, the Farthest Point Sampling (FPS) method proposed in [51]. Since it is difficult to compare these two down-sampling methods directly, what we choose to do is the following. For each down-sampling method, we first down-sample both datasets using one of the methods and then solve the down-sampled problem to optimality using the search space pruning technique. The down-sampling method that results in a better solve-to-optimality solution, i.e., a smaller maxIPDdiff, is deemed as a better option.

Table 4.2 presents the performance comparison results of the greedyDownsampling and FPS methods. For each instance, the numbers in columns 2 and 3 represent the maxIPDdiff (expressed in multiples of w) of the optimal solution of the down-sampled problem obtained using FPS and greedyDownsampling respectively. The greedyDownsampling method outperforms FPS for all instances. In addition, the execution time of greedyDownsampling and FPS are comparable across all instances and both took less than one second. Thus, the greedyDownsampling method suits our purposes better.

Instance	FPS maxIPDdiff (w)	greedyDownsampling maxIPDdiff (w)
100×1560	2.0324	1.3979
64×1560	1.9704	1.3866
50×800	2.1425	2.0197
32×800	2.1379	2.0311
25×400	1.6056	1.4966
16×400	1.2542	1.0418

Table 4.2: Performance comparison down-sampling methods

Next, we evaluate the effectiveness of the search space pruning technique when it is employed to solve the down-sampled to optimality in the first stage of TSMF. We first show the effectiveness of the suggested pruning threshold $\overline{T} = 1.5\tau$ in terms of percentage of solution time reduced and percentage of binary variables pruned after applying the suggested $\overline{T} = 1.5\tau$ to the solve-to-optimality model. Then, we further demonstrate the effectiveness of the suggested $\overline{T} = 1.5\tau$ by studying how the solution time and number of pruned binary variables change as a function of $\overline{T}$.

Table 4.3 summarizes the performance results of applying the suggested $\overline{T}$ value of 1.5τ to the solve-to-optimality model for each instance size. The percentage of solution time reduced and the percentage of binary variables pruned are calculated by comparing the with-pruning results to the without-pruning results. On average, the search space pruning technique eliminated 78% of the binary variables and reduced the solution time by 86.3%. Overall, the search space pruning technique is very effective in reducing the solution time of the solve-to-optimality model by eliminating a significant amount of binary variables from the model; moreover, its effectiveness increases as the problem sizes become larger. Note that we compute the value of τ (in the suggested $\overline{T} = 1.5\tau$) based on the down-sampled LR dataset instead of the full LR dataset since the search pruning technique is applied when solving the down-sampled problem to optimality.

Instance	100×1560	64×1560	50×800	32×800	25×400	16×400
w/o Search Space Pruning Time (s)	1616.5	1400.1	1701.9	280.6	94.4	51.6
w/ Search Space Pruning Time (s)	86.8	90.3	57.5	33.9	47.4	10.6
% of Solution Time Reduced	94.6%	93.6%	96.6%	87.9%	49.8%	79.5%
% of Binary Variables Pruned	84.2%	83.4%	81.0%	79.5%	70.0 %	67.4%

Table 4.3: Effectiveness of search space pruning technique

To further demonstrate the effectiveness of the suggested $\overline{T} = 1.5\tau$, a sensitivity study is performed by applying to the solve-to-optimality model each of the following candidate $\overline{T}$ values: 0.5τ, 0.75τ, 1τ, 1.5τ, 2τ, 2.5τ, 3τ, 3.5τ, and 4τ. Table 4.4 tabulates the main sensitivity study results. For each $\overline{T}$ value, three performance metrics are reported: solution time, percentage of solution time reduced, and percentage of binary variables pruned. Note that the results for $\overline{T}$ values of 0.5τ, 0.75τ, 2.5τ and 3.5τ are not recorded in Table 4.4. This is because 0.5τ and 0.75τ yield infeasible pruned solve-to-optimality models; and 2.5τ (resp. 3.5τ) leads to the same pruned solve-to-optimality model as 2τ (resp. 3τ). Table 4.4 shows that, for all instance sizes, 1τ also gives the same results (and the same pruned model) as the suggested 1.5τ. The equivalence of the models obtained by using $1\tau,\ 2\tau,$ and 3τ and $1.5\tau,\ 2.5\tau,$ and 3.5τ, respectively is due to our conservative definition of τ as the *maximum* distance between an LR point and its closest neighbor in the LR dataset. Therefore, we decide to use 1.5τ as in our tests it always yielded the same pruned model as 1τ, yet we prefer to err on the safer side. In general, the solution time decreases significantly as $\overline{T}$ decreases from 4τ to 1.5τ. Specifically, on average, decreasing $\overline{T}$ from 4τ to 1.5τ saves 51% of the original solution time. An interesting observation is that, the solution time does not always increase as the value of $\overline{T}$ increases for the two smallest instances. For example, for instance 25×400, the solution time decreases by 8.7 seconds when $\overline{T}$ increases from 2τ to 4τ. These counter-intuitive results are rare (and only occur in the smallest instances); moreover they can be explained by the well-known variability of the solvers' solution times (most noticeable when the solution times are small). Despite this, 1.5τ always requires significantly less solution time compared to 2τ, 3τ and 4τ. In sum, the effectiveness of the suggested $\overline{T}$ value of 1.5τ in reducing the solution time is significant for all instance sizes, especially for large instances.

Instance		1τ	1.5τ	2τ	3τ	4τ
	Time (s)	10.6	10.6	30.4	33.3	43.3
16×400	% of Time Reduced	79.5%	79.5%	41.1%	35.5%	16.1%
	% of Variables Pruned	67.4%	67.4%	40.4%	20.0%	8.0%
	Time (s)	47.7	47.4	80.2	47.7	71.5
25×400	% of Time Reduced	49.5%	49.8%	15.0%	49.5%	24.3%
	% of Variables Pruned	70.0%	70.0%	44.0%	23.5%	11.0%
	Time (s)	33.9	33.9	69.0	116.3	193.0
32×800	% of Time Reduced	87.9%	87.9%	75.4%	58.6%	31.2%
	% of Variables Pruned	79.5%	79.5%	60.4%	43.6%	29.5%
	Time (s)	58.0	57.5	239.1	179.1	477.4
50×800	% of Time Reduced	96.6%	96.6%	86.0%	89.5%	71.9%
	% of Variables Pruned	81.0%	81.0%	63.2%	47.0%	33.0%
	Time (s)	89.6	90.3	204.4	500.0	964.9
64×1560	% of Time Reduced	93.6%	93.6%	85.4%	64.3%	31.1%
	% of Variables Pruned	83.4%	83.4%	67.6%	53.1%	40.0%
	Time (s)	86.9	86.8	336.5	593.9	1227.6
100×1560	% of Time Reduced	94.6%	94.6%	79.2%	63.3%	24.1%
	% of Variables Pruned	84.2%	84.2%	69.1%	55.1%	42.3%

Table 4.4: Sensitivity study results with different $\overline{T}$ values

The third analysis shows the local search's effectiveness. Table 4.5 compares four alternative approaches: XiaHeur, XiaHeur with local search, TSMF without local search and full TSMF (i.e. TSMF with local search).

Instance	XiaHeur		XiaHeur + local search		TSMF without local search		Full TSMF	
	maxIPD -diff(w)	Time (s)	maxIPD -diff(w)	Time (s)	maxIPD -diff(w)	Time (s)	maxIPD -diff(w)	Time (s)
100×1560	39.63	1.7	24.79	33.8	1.465	86.9	0.128	162.1
64×1560	35.47	1.2	23.15	10.0	1.427	89.5	0.052	120.4
50×800	33.20	0.3	19.31	5.8	2.203	58.0	0.054	73.8
32×800	25.53	0.2	15.91	2.6	2.679	34.4	0.045	43.3
25×400	0.02	0.1	0.02	0.4	1.497	47.5	0.025	51.2
16×400	8.99	0.05	4.26	1.3	1.404	10.8	0.014	12.3

Table 4.5: Performance of local search and XiaHeur

In Table 4.5, the results of each alternative approach are tabulated in a pair of columns, where the maxIPDdiff and solution time of applying that particular alternative approach for all six instances are tabulated in the left column and the right column, respectively. By doing a pair-wise comparison for all approaches in Table 4.5, one can observe that: 1) full TSMF significantly outperforms both XiaHeur and XiaHeur with local search in five out of six instances except for the second smallest instance, where the maxIPDdiff obtained by applying XiaHeur is only slightly smaller than that obtained by full TSMF; 2) comparing the first two pairs of columns shows that local search improved the solution quality of XiaHeur by 34%, and comparing the last two pairs of columns shows that the solution quality was improved by 97% by including local search in TSMF; 3) even though XiaHeur and XiaHeur with local search outperform TSMF with respect to solution time, all solution times of TSMF are within a reasonable limit so that TSMF can very well serve as an off-line application. In summary, full TSMF produces significantly better solutions than both XiaHeur and XiaHeur with local search within a reasonable amount of time. In addition, the local search is very effective in improving the solution quality. More importantly, local search appears to give greater improvements when starting from a better solution.

The remainder of this subsection evaluates the overall performance of TSMF by comparing it to both ICP and CPD algorithms. Since the misalignment between the two datasets is not known and can be arbitrarily large, it is important to check the robustness of each approach to the change of the underlying rigid transformation between the two datasets. For this purposes, we created 100 variants for each of the six original instances listed in Table 4.1. These 100 variants of each original instance are created by applying 100 uniformly distributed random rotation matrices and random translations (i.e., 100 random rigid body transformations) to the HR dataset of that original instance. In this paper, we generate these 100 uniformly distributed random rotation matrices using the random rotation matrix generation approach proposed in [52]. Note that our TSMF approach is insensitive to the change of initial misalignment degree between the two datasets, but practically, it is reasonable to only allow the manufactured part rotate within the range of -90° and 90° degrees along axes X and Y, and rotate any degree along the vertical axis Z (see Figures 1.1 and 4.5). This restriction allows us to make fair comparison between TSMF and other alternative methods.

To reach an unbiased conclusion on the performance evaluation, we use three registration error metrics: maxIPDdiff, the summation of IPD differences (sumIPDdiff), and the root mean squared error (RMSE). Metric maxIPDdiff is used because it is the metric optimized by TSMF. Metric sumIPDdiff is reported because it is used as the objective function of many IPD-based point set registration algorithms (see, e.g. [7,10]). RMSE is also reported because it is a very popular measure of alignment error between the two datasets in point set registration problems (see, e.g. [5,24,25]). Moreover, RMSE is also the objective function that ICP aims to optimize. For each test instance, RMSE is calculated as follows: 1) estimate the rigid body transformation between the two datasets based on the obtained correspondences; 2) apply the estimated transformation to one dataset in order to align the two datasets; 3) calculate RMSE as the square root of the average squared Euclidean distance of all point correspondences.

Table 4.6 summarizes the performance of TSMF, ICP and CPD on all 600 test cases (100 variants per instance size) in terms of maxIPDdiff. Specifically, Table 4.6's last four columns give the minimum maxIPDdiff, average maxIPDdiff, standard derivation of maxIPDdiff, and maximum maxIPDdiff for the 100 variants of each instance size, respectively. Table 4.6's third column gives the number of variants, out of the 100, on which TSMF outperforms ICP and CPD. Figure 4.6 visualizes the comparison among the three methods via an error-bar plot with respect to maxIPDdif. Each error-bar is plotted using the minimum value and the maximum value from Table 4.6.

Similar comparisons were done using RMSE and sumIPDdiff. Table 4.7 and Figure 4.7 give the results using RMSE. Table 4.8 and Figure 4.8 gives the results using sumIPDdiff. It should be noted that the 100 variants of each instance size in Table 4.6, 4.7 and 4.8 (also in the three corresponding Figures 4.6, 4.7 and 4.8) are 100 different instance of the same instance size each with a different misalignment degree between the two datasets. So, for each instance size, the results for ICP and CPD are not 100 different runs of ICP and CPD on the same instance. Therefore, the variance showed in the results of ICP and CPD is due to their sensitivity to the misalignment degree change between the two datasets.

As expected, since the optimization model formulation of TSMF is a fully IPD-based formulation, it is insensitive to the change of the initial misalignment between the two datasets. In contrast, both ICP and CPD are very sensitive, thus not robust to the change of the misalignment between the two datasets (overall, ICP is more sensitive than CPD). From Figures 4.6 and 4.8, it is clear that TSMF always outperforms ICP and CPD; specifically, for every instance size, TSMF's solution has lower maxIPDdiff and sumIPDdiff than the best solution of ICP and CPD. With respect to RMSE: 1) TSMF outperforms ICP in all 100 test variants of four out of six instance sizes except for the largest and the second smallest instances, where our TSMF performs better than ICP for 80 out of 100 test variants and for 98 out of 100 test variants, respectively; 2) TSMF also outperforms CPD in all 100 test variants of the five largest instances except for the smallest instance where TSMF outperforms CPD in only 12 test variants. The average computation time of ICP and CPD is short and in seconds. Even though TSMF is slower than ICP and CPD, its solution time is well acceptable for it to be practically useful in an off-line precision inspection setting—especially considering that obtaining the HR dataset can take hours. Finally, it is clear from Figures 4.6 to 4.8 that TSMF's performance scales very well with respect to every performance metric, while ICP's and CPD's performances deteriorate as the instance size increases.

		# of variants worse	maxIPDdiff (w)			
Instance	Method	than TSMF	Min	Avg.	Std. Dev.	Max
100×1560	TSMF	-	0.13	0.13	0	0.13
	ICP	100	1.04	2.05	1.32	13.05
	CPD	100	2.26	3.21	0.51	3.59
64×1560	TSMF	-	0.052	0.052	0	0.052
	ICP	100	1.03	2.72	2.71	11.60
	CPD	100	2.28	3.26	0.41	3.60
50×800	TSMF	-	0.054	0.054	0	0.054
	ICP	100	1.03	3.05	1.67	10.24
	CPD	100	3.62	5.12	1.25	6.33
32×800	TSMF	-	0.045	0.045	0	0.045
	ICP	100	1.02	3.00	1.88	9.49
	CPD	100	4.05	4.79	0.88	10.57
25×400	TSMF	-	0.025	0.025	0	0.025
	ICP	100	0.04	1.70	0.67	4.00
	CPD	100	1.43	1.43	0.002	1.44
16×400	TSMF	-	0.014	0.014	0	0.014
	ICP	100	0.06	1.64	0.77	4.21
	CPD	100	0.03	0.40	0.47	1.04

Table 4.6: Performance comparison in terms of maxIPDdiff

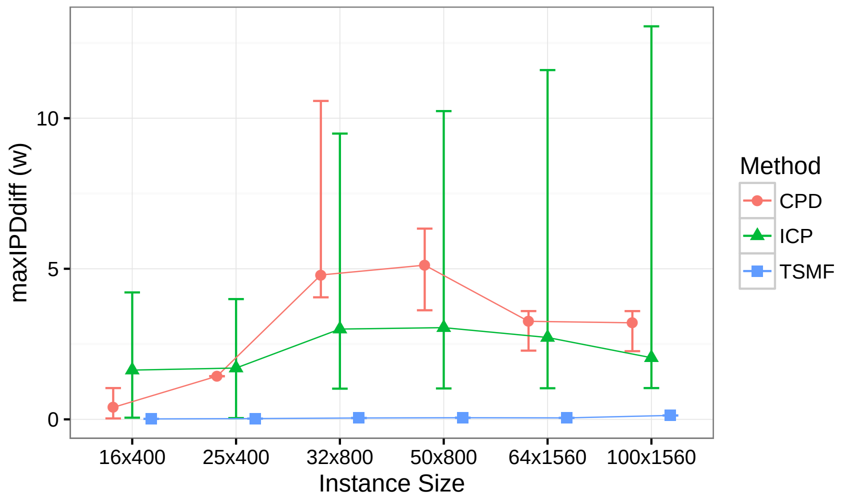


Figure 4.6: Performance in terms of maxIPDdiff

Instance	Method	# of times worse than TSMF	RMSE (w) Min	Avg.	Std. Dev.	Max
100×1560	TSMF	-	0.59	0.59	0	0.59
	ICP	80	0.34	0.78	0.25	1.85
	CPD	100	0.62	0.99	0.20	1.22
64×1560	TSMF	-	0.06	0.06	0	0.06
	ICP	100	0.41	0.85	0.25	1.97
	CPD	100	0.68	1.03	0.20	1.32
50×800	TSMF	-	0.08	0.08	0	0.08
	ICP	100	0.30	0.98	0.30	1.86
	CPD	100	2.11	1.57	0.21	2.11
32×800	TSMF	-	0.07	0.07	0	0.07
	ICP	100	0.35	1.01	0.29	2.04
	CPD	100	1.04	1.41	0.26	2.70
25×400	TSMF	-	0.38	0.38	0	0.38
	ICP	98	0.29	0.65	0.19	1.20
	CPD	100	0.62	0.70	0.04	0.77
16×400	TSMF	-	0.41	0.41	0	0.41
	ICP	100	0.33	0.64	0.13	1.08
	CPD	12	0.13	0.28	0.09	0.46

Table 4.7: Performance comparison in terms of RMSE

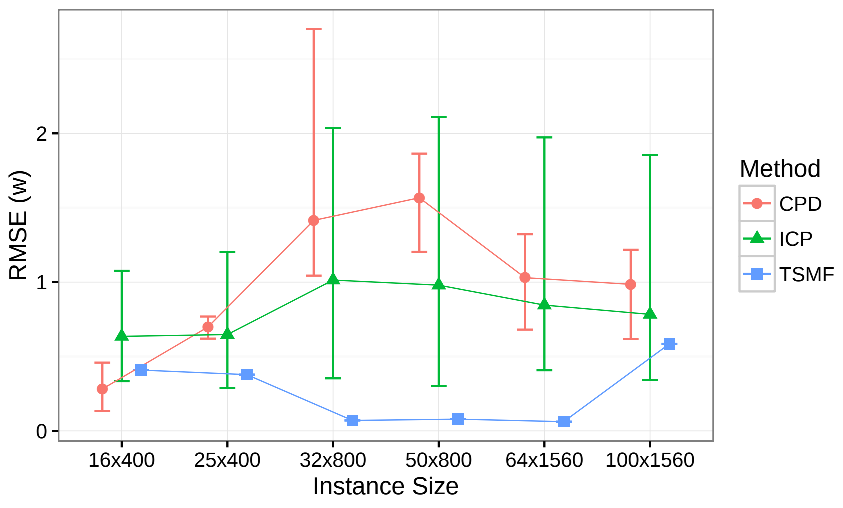


Figure 4.7: Performance in terms of RMSE

Instance	Method	# of times worse than TSMF	sumIPDdiff (w) Min	Avg.	Std. Dev.	Max
100×1560	TSMF	-	150.5	150.5	0	150.5
	ICP	100	742.0	1907.0	754.0	5885.7
	CPD	100	2502.7	3594.5	499.0	4162.9
64×1560	TSMF	-	27.2	27.2	0	27.2
	ICP	100	327.8	936.4	433.1	2420.8
	CPD	100	1220.6	1591.9	194.1	1927.9
50×800	TSMF	-	16.0	16.0	0	16.0
	ICP	100	193.9	675.8	339.9	1912.8
	CPD	100	1119.5	1504.8	234.4	1883.3
32×800	TSMF	-	6.43	6.43	0	6.43
	ICP	100	91.2	308.8	145.2	1122.6
	CPD	100	362.8	565.2	135.4	576.7
25×400	TSMF	-	2.01	2.01	0	2.01
	ICP	100	4.1	146.6	58.9	312.1
	CPD	100	173.5	181.0	5.11	196.9
16×400	TSMF	-	0.68	0.68	0	0.68
	ICP	100	1.47	54.4	16.1	90.1
	CPD	100	1.09	6.8	7.6	21.1

Table 4.8: Performance comparison in terms of sumIPDdiff

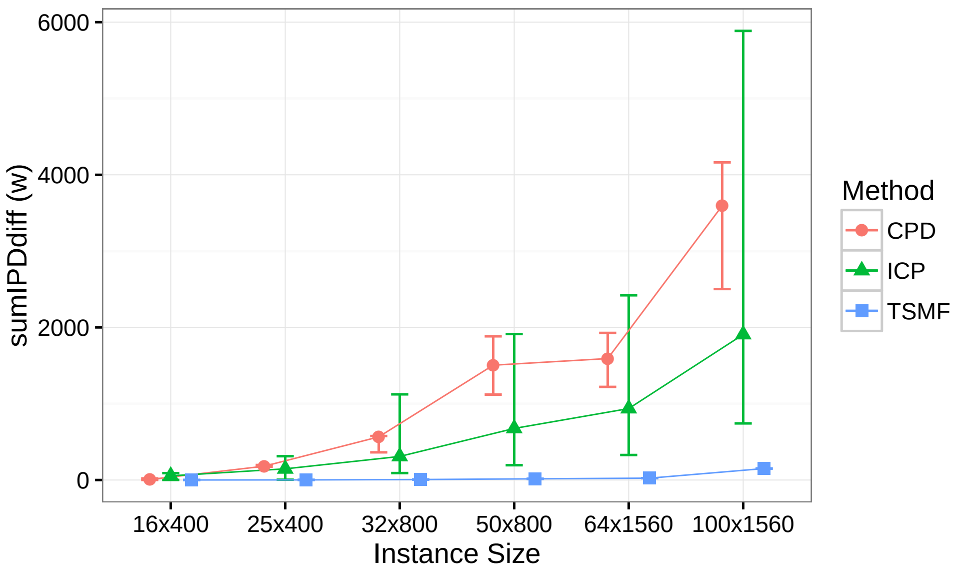


Figure 4.8: Performance in terms of sumIPDdiff

4.4 Summary

In this section, we proposed a two-stage matching framework (TSMF) approach to address the two-resolution metrology matching problem with arbitrary misalignment between the two metrology datasets. Our TSMF approach is robust (specifically, it is insensitive to the misalignment degree) and is able to find a near-to-optimal solution for real-life sized problems within a reasonable amount of time. The first of the two stages comprising the approach aims to find a partial set of correspondences by down-sampling the problem and then solving the down-sampled problem to optimality. The second stage extends the partial set of correspondences obtained in the first stage to a complete set of correspondences (i.e. a complete solution for the original full problem) and further refines the extended set of correspondences by performing an iterative local search.

Numerical experiments showed that TSMF can solve real-life-sized metrology matching problems within couple of minutes. This makes TSMF well suitable to serve as an off-line tool. Moreover, TSMF outperforms ICP and CPD in all 600 test instances in terms of both maxIPDdiff and sumIPDdiff, and almost always produces better solution than ICP and CPD with respect to RMSE for the five largest instance sizes. Unlike ICP and CPD, our approach is robust with respect to the initial misalignment degree between the two datasets, and it scales very well with respect to all three performance metrics as the instance size increases.

5. METROLOGY DATA MATCHING PROBLEM FOR MANUFACTURED PARTS WITH SYMMETRIC FEATURES

Many manufactured parts have symmetric features. For instance, the milled workpiece with a sinewave shaped surface used throughout this dissertation has two types of symmetries: rotation symmetry and reflection symmetry. This is because the measurements were only taken from the top sinewave surface of the workpiece. Specifically, as shown in Figure 5.1, a 180 degree rotation of the top sinewave surface of the milled workpiece along the dashed line of rotation symmetry will make the rotated sinewave coincide with itself; a reflection along the dotted plane of symmetry of the workpiece will also make the sinewave coincide with itself.

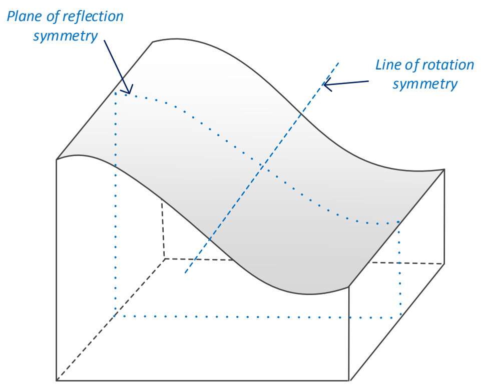


Figure 5.1: Milled part with sinewave surface

Throughout this dissertation, a *correct* set of correspondences refers to a set of correspondences that is orientationally consistent with the underlying true set of correspondences.

It is important to note that, if the symmetric features of the part surface are designed and manufactured perfectly, no point set registration algorithm can solve the two-resolution metrology data matching problem in terms of finding the correct set of correspondences. This is because the maximum IPD differences of the correct set of correspondences and of its reflected and rotated correspondences are all zero; moreover, this is also true for any other performance metric, such as the sumIPDdiff and RSME. This is illustrated in Figure 5.2, where the top and bottom represent the HR datasets and LR datasets, respectively. Specially, in Figure 5.2, from left to right, the three pairs of two datasets represent the correct set of correspondences, the reflection of the correct set of correspondences, and the rotation of the correct set of correspondences, respectively. Note that in each of the three columns of Figure 5.2, the lowercase letters label the correspondences between the two datasets. More specifically, the measurements/points with the same letters are matched; equivalently, when two measurements/points are matched it is as if the respective measured physical locations on the part surface were matched. Evidently, if the sinewave surface is manufactured perfectly, then every metric (e.g. maxIPDdiff, sumIPDdiff, and RSME) would be zero for all three sets of correspondences.

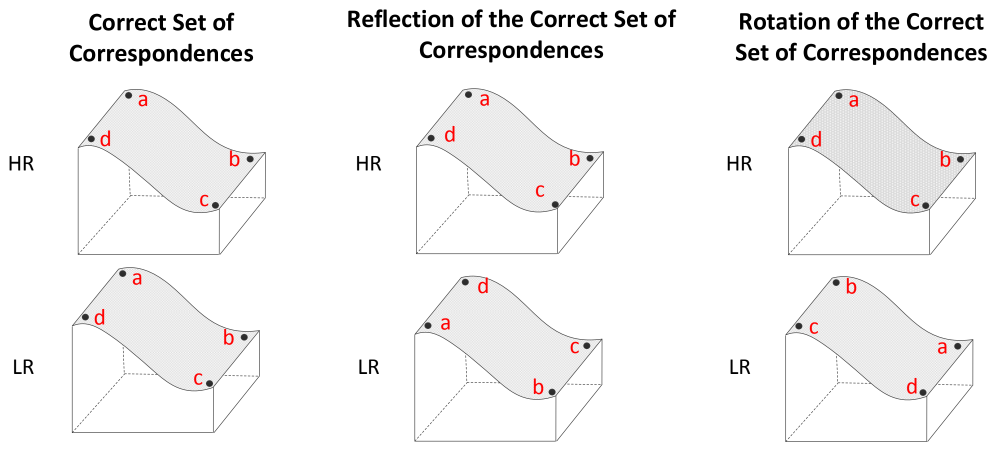

Correct Set of Correspondences
Reflection of the Correct Set of Correspondences
Rotation of the Correct Set of Correspondences
HR
LR
a
b
c
d

Figure 5.2: Correct set of correspondences, reflection and rotation of a correct set of correspondences for a manufactured part with symmetric features

However, even if the nominal design of the part surface is perfect, in practice, the manufacturing process of the surface is not. Then, provided that the scale of the imperfection of the manufactured sinewave surface (e.g. possible dents on the surface) is greater than the resolution scale of the two datasets, one can take advantage of the imperfection to distinguish the correct set of correspondences from the reflected and rotated ones. This is precisely what the algorithm developed in this section does.

As illustrated in Figure 5.2, we need to differentiate between three distinct types of sets of correspondences. Specifically, we need to differentiate them in terms of whether each HR point is matched to an LR point that measures (approximately) the same physical location on the part surface. Throughout the reminder of this dissertation, we use the following terminology:

- A *correct set of correspondences* (illustrated in Column 1 of Figure 5.2) refers to a set of correspondences that is orientationally consistent with the underlying true set of correspondences. In other words, a correct set of correspondences matches each HR point to an LR point that measures (approximately) the same physical location on the part surface.

- An *incorrect set of correspondences* matches each HR point to an LR point that measures a clearly distinct (sometimes even a "far away") physical location on the part surface. As hinted in Figure 5.2, there are two special types of incorrect sets of correspondences that are worth defining because of their close resemblance of correct sets of correspondences:
  - A *reflection of a correct set of correspondences* (illustrated in Column 2 of Figure 5.2) refers to a set of correspondences that: (a) incorrectly matches physical locations on the part surface (e.g. in Column 2 of Figure 5.2, the top left corner of the physical part is matched to the bottom left corner and the top right corner is matched to the bottom right corner), and more importantly, (b) whose incorrect matchings can be corrected by reflecting the correspondences. For example, in Column 2 of Figure 5.2, if the labels in LR dataset are reflected along the plane of reflection symmetry illustrated in Figure 5.1—i.e., labels 'a' and 'b' are exchanged with labels 'd' and 'c', respectively—, then the newly matched measurements will represent the same physical location.
  - A rotation of a correct set of correspondences (illustrated in the Column 3 of Figure 5.2) refers to a set of correspondences that: (a) incorrectly matches physical locations on the part surface (e.g. in Column 3 of Figure 5.2, the top left corner of the physical part is matched to the top right corner and the bottom left corner is matched to the bottom right corner), and more importantly, (b) whose incorrect matchings can be corrected by rotating the correspondences. For example, in Column 3 of Figure 5.2,

if the labels in LR dataset are rotated 180 degrees along the dashed line of rotation symmetry illustrated in Figure 5.1—i.e., labels 'a' and 'c' are exchanged with labels 'b' and 'd', respectively—, then the newly matched measurements will represent the same physical location.

This section aims to improve the TSMF approach to address the difficulty of matching metrology data with symmetric features. To achieve this, we first propose to generate a group of plausible candidate sets of correspondences which we call *solution pool*, and then devise a filtering procedure to select the correct set of correspondences from the solution pool. The remainder of this section is organized as follows: Section 5.1 discusses where and how to generate a solution pool. Section 5.2 describes the proposed filtering procedure, followed by a discussion of three filtering options. Finally, Section 5.4 presents the experimental results and evaluates the performance of the three proposed filtering options.

5.1 Solution Pool Generation

There are three places during TSMF where a solution pool can be generated. First, a solution pool can be generated during the solve-to-optimality step by using the solution pool feature of the MILP solver. Second, a solution pool can be generated during the process of extending the (optimal) partial set of correspondences to a complete one by keeping all the sets of correspondences whose maxIPDdiff is less than a pre-specified threshold. Third, a solution pool can be generated during the iterative local search step by using the solution pool feature of the MILP solver. As explained in the two ensuing paragraphs, one must generate the solution pool during the solve-to-optimality step.

First, we argue that it is not viable to generate the solution pool during the process of extending the partial set of correspondences or during the iterative local search step. Note that both generalizedXiaHeur and local search do not change (or change very little) the orientation of the solve-to-optimality solution. More specifically, generalizedXiaHeur preserves the solution orientation because it uses the solve-to-optimality solution as anchor points during the solution extension process. Similarly, the local search procedure only searches a small neighborhood to refine the correspondences of the extended solution and thus only adjusts the orientation of the solve-to-optimality solution by a very small degree. Consequently, an incorrect partial set of correspondences from the solve-to-optimality step will inevitably cause the solution pools generated at later places during TSMF (i.e. after extending the partial set of correspondences and after iterative local search procedure) to contain only incorrect sets of correspondences.

Next, we argue that generating the solution pool during the solve-to-optimality step allows us to include at least one correct partial set of correspondences in the pool. Note that, due to the symmetry of the part surface and the not-so-perfect evenness of the down-sampled datasets, the optimal partial set of correspondences obtained at the solve-to-optimality step might not be correct. However, it is reasonable to expect that the objective value of a correct partial set of correspondences at the solve-to-optimality step is close to that of the optimal solution of the solve-to-optimality step. Therefore, by including a group of feasible partial sets of correspondences whose objective values are close to the optimal objective value, one can be reasonably certain that the solution pool will contain at least one correct set of correspondences. Hereafter, we refer to this solution pool as the *basic pool*.

Next, Subsection 5.1.1 describes how to generate the basic pool using the solution pool feature of CPLEX, and then Subsection 5.1.2 discusses the derivation of two other solution pools variants based on the basic pool.

5.1.1 Basic Pool Generation

The basic pool comprises a group of feasible partial sets of correspondences whose objective values are close to the optimal objective value. Thus, the basic pool can be easily generated by invoking the solution pool feature of a MILP solver during the solve-to-optimality step. The solution pool feature is provided by most MILP solvers to allow one to generate and store multiple intermediate feasible solutions of a MILP model. Since the MILP solver we have been using throughout this dissertation is CPLEX 12.4, below we describe how to generate the basic pool only in CPLEX 12.4 (through its concert technology C++ interface).

To generate the basic pool, we need to set the following two CPLEX parameters before calling the *solve* method (i.e. the function called to solve a MILP model): *SolnPoolIntensity* and *SolnPoolGap*. The following two paragraphs give more details on how we set these two parameters to generate the basic pool.

The parameter *SolnPoolIntensity* controls the level of effort that CPLEX exerts to generate the solution pool. Specifically, *SolnPoolIntensity* can be set to an integer value between 0 and 4; higher values indicate higher levels of effort to generate a larger number of feasible solutions in the solution pool. The default value of *SolnPoolIntensity* is 0, which only keeps a small number of solutions in the basic pool. From some preliminary numerical experiments, we observed that, one needs to set *SolnPoolIntensity* to the most aggressive level of 4 to ensure there is at least one correct set of correspondences in the basic pool.

The parameter *SolnPoolGap* sets a relative tolerance on the objective values of the solutions included in the pool. For instance, setting *SolnPoolGap* to 1.0 directs CPLEX to only keep solutions whose objective values are less than 2 times the optimal objective value. In order to ensure that the basic pool only includes the feasible solutions with objective values close to the optimal objective value, *SolnPoolGap* needs to be set to a reasonably small percentage. Through extensive experiments, we found that setting *SolnPoolGap* to 1.0 is a good choice. Indeed, our computational results in Section 5.4 show that setting *SolnPoolGap* to 1.0 ensures that the resulting basic pools of all six test instances contain at least one correct (partial) set of correspondences. In fact, for all six instances, a correct partial set of correspondences in the basic pool has an objective value which is less than 1.71 times the optimal objective value; moreover, for four out of six instances, a correct partial set of correspondences in the basic pool has an objective value which is less than 1.5 times the optimal objective value. Therefore, setting *SolnPoolGap* to 1.0 is a conservative choice. In addition, setting *SolnPoolGap* to 1.0 also on average reduces the basic pool size by at least 60% compared to the basic pool generated without imposing a limit on *SolnPoolGap*. This is important because generating a basic pool with significantly smaller size can save substantial amount of computational time required to complete the pool generation.

In this paragraph, we describe the *populate* method, an alternative way to generate the basic pool in CPLEX, and explain why the aforementioned *solve* method is preferred to the *populate* method. (Remark: this paragraph can be omitted without loss of continuity). The *populate* method intends to generate solutions in addition to those found during the regular branch and bound (B&B) process. Specifically, the *populate* method also explores those nodes in the B&B tree that would be pruned by the default optimality based B&B algorithm of CPLEX [53]. However, one main drawback of the *populate* method is that it is prone to generate many duplicated solutions in the pool. Through extensive experiments, we found this duplication can be alleviated to some extent by simultaneously directing CPLEX (a) to set *SolnPool-Replace* (used to designate the strategy for replacing a solution in the solution pool when the solution pool has reached its capacity) to 2 to produce a diverse set of solutions in the pool, (b) to set *PopulateLim* (used to limit the number of MILP solutions generated for the solution pool) to a larger number, and to set *solnPool-Capacity* (used to limit the number of solutions kept in the solution pool) to a small percentage of *PopulateLim*. However, these improved settings do not mitigate the solution duplication problem to a satisfactory level.

5.1.2 Two Solution Pool Variants of the Basic Pool

Two variants of the basic pool can be derived during two other steps of TSMF. Specifically, the first variant, termed *extended pool*, is derived by extending with generalizedXiaHeur each partial set of correspondences in the basic pool to a complete set of correspondences. The second variant, termed *refined & extended pool*, is derived by refining each complete set of correspondences in the extended pool using the iterative local search procedure. The extended pool allows us to filter the solution pool after applying the generalizedXiaHeur algorithm. The refined & extended pool allows us to filter the solution pool after applying the iterative local search procedure (i.e. at end of the TSMF).

Note that, as illustrated in Figure 5.3, there is a one-to-one correspondence between the solutions in the basic pool and those in the two other solution pool variants. Specifically, each solution in the basic pool has exactly one extended version in the extended pool and one extended & refined version in the extended & refined pool. Therefore, we use a unique solution index when referring to a solution regardless of its status. Specifically, as shown in Figure 5.3, solution B_i in the basic pool corresponds to solution E_i in the extended pool and solution R_i in the extended & refined pool, respectively. Hereafter, with a slight abuse of notation, since the three pools contain the same set of solutions (with different solution statuses), we use the generic term "solution pool" to refer to this set of solutions. In other words, we want to refer to the set of solutions without referring to their specific statuses. Similarly, we refer to solution i as the i^{th} solution in the solution pool, and solution i could correspond to any of its three solution statuses (i.e. B_i, E_i, and R_i) depending on the context.

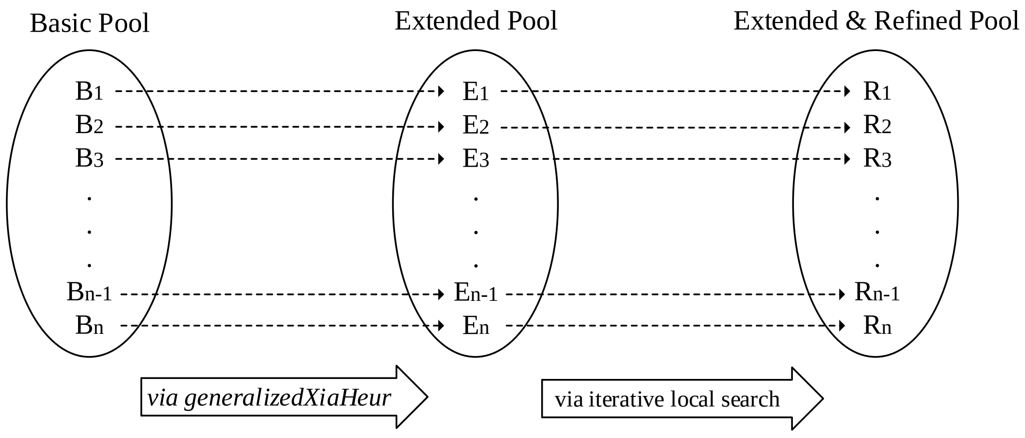


Figure 5.3: One-to-one correspondences between three solution pools

5.2 Filtering Procedure

The objective of the filtering procedure is to select a correct set of correspondences from the solution pool. As mentioned earlier, provided that the scale of the imperfection of the manufactured surface (e.g. possible dents on the surface) is greater than the resolution scale of the two datasets, one can take advantage of the imperfection to distinguish the correct set of correspondences from the reflected and rotated ones. Specifically, it is reasonable to expect that the correct set of correspondences has the minimum RMSE once the two datasets are properly aligned. However, our preliminary experiments showed that the set of correspondences with the minimum RMSE is not necessarily a correct one—it may be a reflection or a rotation of a correct set of correspondences. Moreover, this is also true the other performance metrics maxIPDdiff and sumIPDdiff. Therefore, here we present a filtering procedure that aims to find a correct set of correspondences even when the "optimal" set of correspondences is not correct. In short, our filtering procedure aims to find the correct set of correspondences with minimum RMSE—which we term as the best correct set of correspondences. Next, we first explain a core component of the filtering procedure and then describe the filtering procedure in detail.

A core component of the filtering procedure is to estimate the rigid body transformation between the two datasets based on a given set of correspondences in the solution pool. Specifically, given a set of correspondences, estimating the rigid body transformation entails finding the rotation matrix and translation vector so that both datasets are best aligned (in terms of a least square error criterion). There exists multiple rigid transformation estimation algorithms in the literature. Eggert et al. [12] compared four popular rigid transformation recovery algorithms (i.e., a singular value decomposition (SVD)-based algorithm [54], orthonormal matrix based algorithm [55], unit quaternion based algorithm [56], and dual quaternion based algorithm [57]) through extensive numerical experiments, and concluded that, under typical real-world noise levels, all four methods are equally robust. We choose the SVD-based algorithm because its implementation is widely available in linear algebra packages.

It should be noted that we assume the axes of the 3D coordinate systems of both CCMM and OCMM (used for obtaining the HR and LR datasets respectively) satisfy the right-hand rule. Thus, the correct rigid transformation between the two datasets must comprise a proper rotation and a translation. In particular, due to our assumption, the correct rigid transformation cannot contain a reflection (i.e., an improper rotation). Therefore, given a set of correspondences, if the estimated rigid transformation contains a reflection (i.e., the determinant of the estimated rotation matrix is -1 [12, 54]), then it indicates that the given set of correspondences is incorrect (specifically, it is a reflection of a correct set of correspondences). Thus, a reflection of a correct set of correspondences can be detected directly. In contrast, one cannot detect directly a rotation of a correct set of correspondences. This is because the rigid transformation estimated from either a correct set of correspondences or a rotation of a correct set of correspondences comprises a proper rotation and a translation. Therefore, in order to differentiate between correct sets of correspondences and reflections of correct sets of correspondences, one must rely on the minimum RMSE criterion.

Based on the above analyses, we propose a filtering procedure that combines the minimum RMSE criterion and a reflection detection component. Below, we describe the two steps of the filtering procedure in detail.

Step 1 - *Calculate and rank the RMSE of all solutions in the pool.*

This step calculates the RMSE of all solutions in the pool and ranks them accordingly from smallest to largest. RMSE is calculated as follows: (a) estimate the rigid body transformation between the two datasets based on the obtained correspondences; (b) apply the estimated transformation to one dataset in order to align the two datasets; (c) calculate RMSE as the square root of the average squared Euclidean distance of all point correspondences.

Step 2 - *Find the non-reflected solution with minimum RMSE.*

First, find the solution with the minimum RMSE in the pool. Then, check whether the estimated rigid transformation based on that solution contains a proper rotation. If so, this solution is the non-reflected solution with the minimum RMSE and should be selected as the final output. Otherwise, the considered solution is a reflection of a correct set of correspondences; and one should repeat this step by examining the solution with the next smallest RMSE in the pool until a non-reflected solution is found.

5.3 Three Filtering Options

Having described the filtering procedure, we now discuss the three possible places during TSMF where the filtering procedure can be performed. As illustrated in Figure 5.4, the first place to perform the filtering procedure is after the solve-to-optimality step, and the filtering is performed on the partial sets of correspondences in the basic pool. The second place to perform the filtering procedure is after extending each partial set of correspondences in the basic pool by generalizedXiaHeur and the filtering is performed on the extended complete sets of correspondences in the extended pool. The third place to apply the filtering procedure is after refining each complete set of correspondences in the extended pool by iterative local search, and the filtering is performed on the complete sets of correspondences in the refined & extended pool. We remark that, in all three filtering options, the filtering procedure would be applied exactly in the same manner; that is, they only differ on the solution pool to which the filtering procedure is applied.

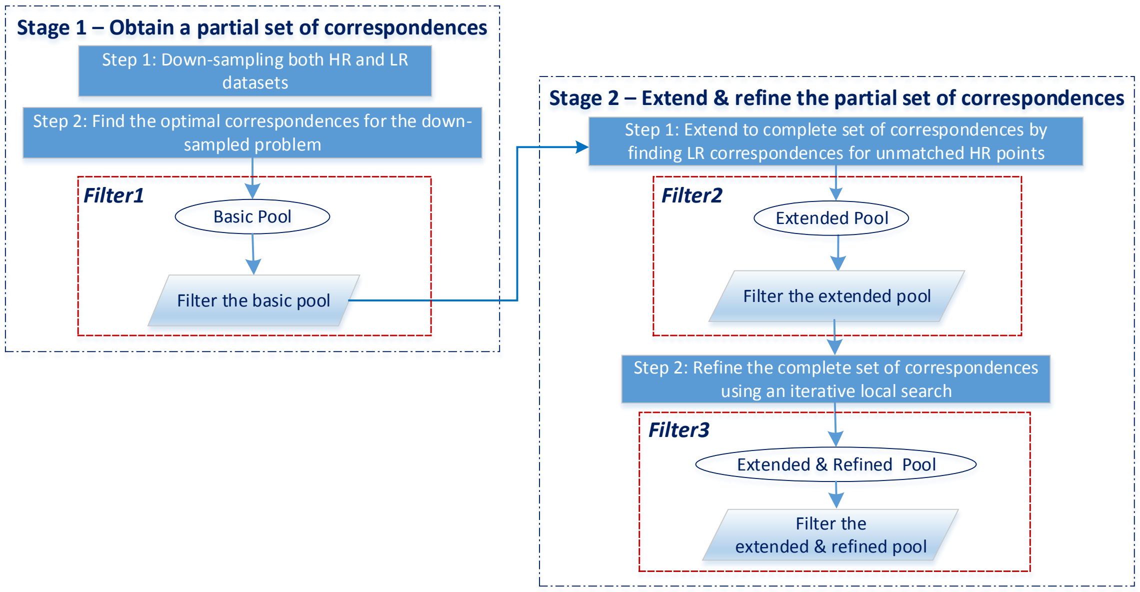

Stage 1 – Obtain a partial set of correspondences
Step 1: Down-sampling both HR and LR datasets
Step 2: Find the optimal correspondences for the down-sampled problem
Filter1
Basic Pool
Filter the basic pool
Stage 2 – Extend & refine the partial set of correspondences
Step 1: Extend to complete set of correspondences by finding LR correspondences for unmatched HR points
Filter2
Extended Pool
Filter the extended pool
Step 2: Refine the complete set of correspondences using an iterative local search
Filter3
Extended & Refined Pool
Filter the extended & refined pool

Figure 5.4: Three possible places during TSMF to perform filtering

Hereafter, we refer to the above three filtering options as *Filter1*, *Filter2* and *Filter3*, respectively. For instance, if Filter1 option is chosen, then the filtering procedure is performed on the basic pool after the solve-to-optimality step. Performance evaluation of these three filtering options will be conducted in Section 5.4.

5.4 Computational Experiments and Results

This section first describes the implementation and experimental setup of the filtering procedure, then presents the computational results and compares the alignment performance of the three filtering options. For this purpose, we still use the same six metrology data instances of a milled sinewave surface with the same TSMF parameters used in Subsection 4.3.1. All the numerical experiments are conducted on the same computing environment — Linux (Ubuntu 14.04) machine with Intel E5-1620 3.4GHz processor and 32G RAM.

5.4.1 Implementation and Experimental Setup

The proposed filtering procedure was implemented in C++ using the Concert Technology interface of CPLEX 12.4 and was integrated into TSMF. Hereafter, we refer to this integrated program as *TSMFwithFilter* and to the original (non-filtered) TSMF simply as *TSMF*. The SVD function used in the filtering procedure is provided by the Armadillo C++ linear algebra library [50].

Since incorporating the filtering procedure into TSMF does not affect TSMF's insensitivity to the change of misalignment between the two datasets, it is only necessary to test the three filtering options of TSMFwithFilter on the original data instance of each instance size and not on all 600 instances (100 randomly rotated and translated variants for each of the six instance sizes).

It should be noted that, by default, CPLEX invokes the so-called deterministic parallel mode when solving a MIP model. This mode directs CPLEX to apply as much parallelism as possible (i.e., use the maximum number of threads available) while still achieving deterministic results. That is, repeated running on the same instance with the same parameter settings on the same computing platform will follow exactly the same solution path, yielding the same solution values and performance. However, even with the same instance and CPLEX parameter settings, running TSMFwithFilter on a different computing platform may lead to different computational results.

5.4.2 Results and Performance Analysis

In this subsection, we examine the effectiveness of the three proposed filtering options and identify the best one (i.e. the most effective one). The key characteristic of the effectiveness of a filtering option is whether the final selected solution is a correct set of correspondences. Then, we compare the best filtering option with the original TSMF and the two alternative methods, CPD and ICP, in terms of alignment performance.

Comparison of the alignment performance of the three filtering options

The main objective of this comparison is to find out which filtering option(s) select a correct set of correspondences and whether their computational times are reasonable. Since the underlying true set of correspondences of the two datasets is known, the orientation correctness of the selected set of correspondences is checked by visually comparing the selected set of correspondences to the correct set of correspondences once the two datasets are aligned to one coordinate system. Appendix C illustrates and explains in detail this visual examination of the alignment performance.

Table 5.1 summarizes the main results of three filtering options for each instance size. Columns 1 and 2 give the instance size and the filtering option. Column 3 gives the number of candidate sets of correspondences in the solution pool of each instance size. Column 4 gives the index of the selected set of correspondences by the respective filtering option. Column 5 indicates whether the selected set of correspondences is correct; specifically, the results in this column summarize Figures C.1 to C.6 in Appendix C. Columns 6 - 8 record the running time of solve-to-optimality step (including the basic pool generation time), the local search time, and the total running time of TSMFwithFilter. The last three columns report the maxIPDdiff, sumIPDdiff and RMSE of the selected set of correspondences after local search, respectively. Note that, as in Section 4.3, hereafter, we report maxIPDdiff, sumIPDdiff and RMSE as a multiple of w—the average distance between each LR point and its closest neighbor in the LR dataset. Note that the computational time for greedyDownsampling, generalizedXiaHeur and the filtering procedure are not included in Table 5.1 as they are really fast; specifically, their average running time per instance are 0.05 seconds, 0.57 seconds and 0.006 seconds, respectively.

Instance	Filter Option	# of Solns in Pool	Selected Soln Index	Selected Soln is Correct?	Solve-To-Opt Time (s)	Local Search Time (s)	Total Time (s)	maxIPD -diff (w)	sumIPD -diff (w)	RMSE (w)
100X1560	Filter1	16	12	Yes	213.7	67.3	281.4	0.057	62.018	0.019
	Filter2		12	Yes	213.9	66.5	281.6	0.057	62.018	0.019
	Filter3		12	Yes	213.4	1275.5	1490.0	0.057	62.018	0.019
64X1560	Filter1	38	28	No	13923	30.4	13954.0	0.984	509.273	0.378
	Filter2		28	No	13889	30.6	13921.0	0.984	509.273	0.378
	Filter3		30	Yes	13895	1342.0	15238.0	0.059	31.553	0.020
50X800	Filter1	46	1	No	95.1	15.9	111.1	0.054	16.004	0.079
	Filter2		16	No	94.8	15.2	110.7	0.054	16.004	0.079
	Filter3		40	Yes	95.2	919.2	1015.1	0.050	15.772	0.020
32X800	Filter1	96	61	No	80.5	6.5	87.1	1.075	186.598	0.759
	Filter2		23	No	80.1	6.2	87.0	0.045	6.435	0.070
	Filter3		10	Yes	80.1	884.8	965.6	0.046	6.939	0.020
25X400	Filter1	87	76	No	39.0	4.1	43.3	0.049	4.768	0.305
	Filter2		37	No	38.7	3.6	42.6	0.027	2.069	0.074
	Filter3		54	Yes	38.6	391.2	430.0	0.021	2.068	0.009
16X400	Filter1	29	16	No	23.5	1.4	25.0	0.049	1.845	0.283
	Filter2		13	No	23.1	2.3	25.5	0.747	26.987	0.335
	Filter3		8	Yes	23.0	49.7	72.9	0.015	0.691	0.008

Table 5.1: Main results of three filtering options

From Table 5.1, we conclude that Filter3 outperforms Filter1 and Filter2 in terms of successfully selecting a correct set of correspondences[1]. In particular, Table 5.1 shows that Filter3 always selected a correct set of correspondences. In contrast, Filter1 and Filter2 output a correct set of correspondences only for the largest instance.

[1]Note that, due to the minimum-RMSE criterion used in the filtering procedure, the Filter3 may also reject some other correct sets of correspondences if there are more than one correct set of correspondences in the extended & refined pool.

In addition, Table 5.1 shows that the correct set of correspondences selected by Filter3 almost always outperform those selected by Filter1 and Filter2 in terms of maxIPDdiff, sumIPDdiff and RMSE. The only exceptions are: 1) All filtering options output the exact same set of correspondences for the largest instance; and 2) Filter2 slightly beats Filter3 in terms of maxIPDdiff and sumIPDdiff for instance 32×800.

In contrast, when it comes to computational time, Filter1 and Filter2 on average run 5.6 times faster than Filter3. This is not surprising because Filter3 needs to perform the iterative local search process (the second most computationally expensive component in TSMF) on every set of correspondences in the solution pool. A close inspection reveals that Filter3 took less than 25 minutes in five out of six instances, but took more than three hours for instance 64×1560. Note that, for instance 64×1560, the vast majority of the run time was spent on generating the basic pool. While a run time of more than three hours may seem excessive, it is important to consider that the measuring process of the HR dataset may also take hours. Therefore, we believe that Filter3 is well suitable to serve as an off-line tool.

Give the above results, we next evaluate the performance of the two main components/guiding principles of the filtering procedure—the reflection detection component and the minimum RMSE criterion. For this purpose, Table 5.2 reports, for each instance and filtering option, the confusion matrix tabulating the accuracy performance of the reflection detection component of the filtering procedure. Similarly to the evaluation of the alignment performance, the results in Table 5.2 were obtained via a visual inspection.

Instance			**Filter1**		**Filter2**		**Filter3**	
			Predicted Class		Predicted Class		Predicted Class	
			Reflection	*Rotation*	*Reflection*	*Rotation*	*Reflection*	*Rotation*
16x400	True Class	*Reflection*	4	0	1	0	3	0
		Rotation	0	1	0	1	0	1
	Accuracy		100%		100%		100%	
25x400	True Class	*Reflection*	0	0	3	0	0	0
		Rotation	0	1	0	1	0	1
	Accuracy		100%		100%		100%	
32x800	True Class	*Reflection*	0	0	0	0	6	0
		Rotation	0	1	0	1	0	1
	Accuracy		100%		100%		100%	
50x800	True Class	*Reflection*	0	0	2	0	3	0
		Rotation	0	1	0	1	0	1
	Accuracy		100%		100%		100%	
64x1560	True Class	*Reflection*	0	0	0	0	0	0
		Rotation	0	1	0	1	0	1
	Accuracy		100%		100%		100%	
100x1560	True Class	*Reflection*	2	0	0	0	0	0
		Rotation	1	1	0	1	0	1
	Accuracy		75%		100%		100%	
Overall Accuracy			92%		100%		100%	

Table 5.2: Confusion matrices of the reflection detection component of the three filtering options for each instance

Table 5.2 shows that, on every instance, the reflection detection component in Filter2 and Filter3 has 100% accuracy in correctly differentiating between reflections and proper rotations. In contrast, the reflection detection component in Filter1 is only fully accurate for the five smallest instances. We conclude that the reflection detection component is extremely accurate. Moreover, Table 5.2 also shows that, for some instances, several of the sets of correspondences with the smallest RSMEs implied a reflection; and, in all of these cases, the reflection detection component correctly filtered them out. Therefore, we also conclude that the reflection detection component is useful and supplements the minimum RMSE criterion.

Important Remark: The reflection detection component in Filter2 (resp. Filter1) correctly detected all (resp. almost all) of the reflected set of correspondences for all six instances; however, as shown in Table 5.1, Filter1 and Filter2 failed to select a correct set of correspondences among these non-reflected ones. Specifically, it selected an incorrect proper rotation (see, e.g., the third column in Figure 5.2). which is a proper rotation of itself). This is because all correct solutions in the basic pool (resp. extended pool) have a larger RMSE than that of the properly rotated solution selected by Filter1 (resp. Filter2). Therefore, by comparing the results in Table 5.1 and Table 5.2, we conclude that minimum RMSE criterion is not reliable when applied to Filter1 and Filter2. This is mainly because the solutions in the basic pool and the extended pool do not have as good alignment quality as those in the extended & refined pool. Below we explain the underlying reason for the poor performance of Filter1 and Filter2.

The failure of Filter1 can be mainly attributed to two aspects: the not-so-evenness of the down-sampled datasets and the sparsity of down-sampled HR points (around eight points). On one hand, the not-so-evenness of the down-sampled datasets may distort the RMSE of the solutions. This distortion can potentially increase the RMSE of the correct solution so that it becomes larger than that of the incorrect ones. Also, the not-so-evenness of the down-sampled datasets increases the dissimilarity level between the two down-sampled datasets. This larger dissimilarity between the two datasets introduces large "noises" (combined with two types of symmetries of the sinewave surface) that cause Filter1 to fail[2]. On the other hand, the sparsity may cause the down-sampled HR points to lay almost on a plane, which makes

[2]Due to large level of "noises" of two down-sampled datasets, SVD-based reflection detection component simply classifies a solution to either reflection or proper rotation randomly depending on which class has the smaller RMSE. Besides, this large "noises" combined with the two types of symmetries of the part surface makes the minimum RMSE criterion (i.e., the second step of the filtering procedure) not reliable.

it impossible to differentiate between a proper rotation and a reflection (i.e. in the planar case, both a reflection and a proper rotation are equivalent). Yet, as expected, this rarely happens; as shown in Table 5.2, only Filter1 confused one proper rotation as a reflection for the largest instance.

Filter2 fails mainly because of the fact that the solution extension step by generalizedXiaHeur does not improve the correspondences' quality of the anchor pairs. The correspondences' quality of the anchor pairs is not good because the anchor pairs correspondences are determined based on the not-so-even down-sampled datasets. Therefore, in sum, Filter2 fails for the similar reasons underlying Filter1's failure.

In contrast, Filter3, after applying the iterative local search step, does rematch the anchor pairs. Moreover, in general, the solutions in extended & refined pool have better alignment quality than those in the basic pool and extended pool. This allows the proposed filtering procedure to work more effectively. In other words, better solution quality makes it more likely for the SVD-based rigid transformation estimation algorithm to correctly detect reflections and proper rotations, and for the overall performance of Filter3.

Comparison of Filter3 with other alternative methods

Now we compare the best filtering option, Filter3, with TSMF, CPD and ICP in terms of both solution orientation correctness and three performance metrics (maxIPDdiff, sumIPDdiff, and RMSE). The comparison results are tabulated in Table 5.3.

Instance	Method	Correct Soln?	maxIPDdiff (w)	sumIPDdiff (w)	RMSE (w)
100X1560	Filter3	Yes	0.057	62.02	0.019
	TSMF	No	0.130	150.50	0.59
	CPD	No	2.26	2502.70	0.62
	ICP	No	1.04	742.00	0.34
64X1560	Filter3	Yes	0.059	31.55	0.020
	TSMF	No	0.052	27.20	0.06
	CPD	No	2.28	1220.60	0.68
	ICP	Yes	1.03	327.80	0.41
50X800	Filter3	Yes	0.050	15.77	0.020
	TSMF	No	0.054	16.00	0.08
	CPD	Yes	3.62	1119.50	2.11
	ICP	No	1.03	193.90	0.30
32X800	Filter3	Yes	0.046	6.94	0.020
	TSMF	No	0.045	6.43	0.07
	CPD	Yes	4.05	362.80	1.04
	ICP	No	1.02	91.20	0.35
25X400	Filter3	Yes	0.021	2.07	0.009
	TSMF	No	0.025	2.01	0.38
	CPD	No	1.43	173.50	0.62
	ICP	Yes	0.04	4.10	0.29
16X400	Filter3	Yes	0.015	0.69	0.008
	TSMF	No	0.014	0.68	0.41
	CPD	No	0.03	1.09	0.13
	ICP	No	0.06	1.47	0.33

Table 5.3: Compare Filter3 with other methods

First, we compare Filter3 to the original TSMF. Table 5.3 shows that Filter3 always outperforms TSMF because TSMF missed the correct set of correspondences for all six instances. Moreover, in terms of both maxIPDdiff and sumIPDdiff, Filter3 gives very comparable solutions for the five smallest instances, and outperforms TSMF for the largest instance by one order of magnitude. In terms of RMSE, Filter3 outperforms TSMF for all six instances; for some instances by one order of magnitude.

Second, we compare Filter3 to two other widely-used rigid registration algorithms, CPD and ICP. Table 5.3 shows that both CPD and ICP are only able to output a correct set of correspondences for two out of six instances, while Filter3 always gives the correction set of correspondences. Moreover, Filter3 outperforms both CPD and ICP for all six instances in terms of maxIPDdiff, sumIPDdiff and RMSE.

5.5 Summary

This section aims to address the difficulty of matching two-resolution metrology data for manufactured parts with symmetric geometric features. To achieve this, a filtering procedure is proposed to improve the TSMF approach in terms of finding a correct set of correspondences between the two metrology datasets with symmetric geometric features. More specifically, we propose to generate such solution pool containing a group of plausible candidate sets of correspondences and devise a filtering procedure to select the correct set of correspondences from that solution pool. We also discussed where and how to generate a solution pool and introduced three filtering options, each filtering a different solution pool variant at a different place during TSMF.

We conducted two main performance comparisons in this section: 1) comparison of the alignment performance of the proposed three filtering options; and 2) comparison of Filter3 with other alternative methods (i.e., TSMF, CPD and ICP). The first performance comparison showed that Filter3 outperforms Filter1 and Filter2, and always selected a correct set of correspondences. In contrast, Filter1 and Filter2 output a correct set of correspondences only for the largest instance. Moreover, the correct solution selected by Filter3 almost always outperforms those selected by Filter1 and Filter2 in terms of maxIPDdiff, sumIPDdiff and RMSE. Even though Filter1 and Filter2 run much faster than Filter3, Filter3 is still well suitable to serve as an off-line tool. The second performance comparison showed that the best filtering option, Filter3, always outperforms TSMF for all six instances in terms of successfully selecting a correct set of correspondences. In addition, when compared to CPD and ICP, Filter3 outperforms them for four out of six instances as both CPD and ICP are also able to output a correct set of correspondences for two instances. In terms of three performance metrics (i.e., maxIPDdiff, sumIPDdiff and RMSE), Filter3 also always produces better results than CPD and ICP and either gives comparable or one order of magnitude better results than TSMF.

6. SUMMARY AND FUTURE WORK

6.1 Summary

Aligning two-resolution metrology data is a very important and challenging problem. The problem is challenging in two ways: (1) the problem is computationally prohibitive for off-the-shelf optimization solvers; (2) the existing heuristic approaches to address the problem are not robust and often lead to solutions of very poor quality (this is especially true for problems with a large degree of misalignment). In this dissertation, we proposed a two-stage matching framework (TSMF) to provide a competitive and robust solution to this problem. The TSMF approach can serve as a good off-line tool to aid the geometric quality control process of manufactured parts. Moreover, to address the difficulty of matching metrology data for manufactured parts with symmetric features, a filtering procedure is proposed to enhance the TSMF approach. Specifically, the filtering procedure aims to select set of correspondences that are orientationally consistent with the underlying true set of correspondences.

The proposed TSMF approach aims to establish the correspondences between two fully-overlapping metrology data with different resolutions, dramatic cardinality difference and arbitrarily large degree of misalignment. The TSMF approach follows a coarse-to-fine strategy and contains two stages. The first stage obtains a coarse alignment (i.e. a partial set of correspondences) by solving a down-sampled problem. The second stage extends the partial set of correspondences obtained from solving to optimality the down-sampled problem to a complete one on the original full datasets, and refines the extended set of correspondences through an iterative local search.

Numerical results showed that TSMF outperforms two widely used algorithms, ICP and CPD, in all instances with respect to both maximum inter-point-distance difference (maxIPDdiff) and summation inter-point-distance difference (sumIPDdiff) metrics and almost always obtains a better solution than ICP and CPD with respect to the root mean square error (RMSE) metric. Compared to ICP and CPD, TSMF is robust to the initial misalignment degree between the two metrology datasets, and its performance, in terms of all performance metrics, scales very well as the instance size increases.

To address the difficulty of matching metrology data for manufactured parts with symmetric features, a filtering procedure is proposed to improve the TSMF approach in terms of selecting a set of correspondences that is orientationally consistent with the underlying true set of correspondences. Our approach works by generating a solution pool that contains a group of plausible candidate sets of correspondences and subsequently filtering this solution pool in order to select a correct set of correspondences from that solution pool.

With respect to the alignment performance of our proposed filtering procedure, the numerical results showed that, TSMF-with-filtering exhibits much better alignment performance than TSMF-without-filtering, CPD and ICP in terms of both orientation correctness of the selected solution and three quantitative performance metrics.

Furthermore, when it comes to computational performance, TSMF can solve real-life sized metrology data matching problems within a couple of minutes. Even though TSMF-with-filtering takes relatively significantly more time than TSMF-without-filtering, its worst-case running time is still acceptable considering that that the measuring process of the HR dataset may also take a comparable amount of time. Therefore, we believe that both TSMF-with-filtering and TSMF-without-filtering are well suited to serve as off-line tools in the manufacturing quality control process.

6.2 Future Work

6.2.1 Development of Bayesian Alignment Model to Solve the Metrology Matching Problem

We plan to explore the direction of developing a Bayesian alignment approach as an alternative method to address the metrology data matching problem for part surfaces with symmetric features. Specifically, our goal is to develop a Bayesian alignment approach similar to the one in [16] to solve the metrology problem. There are two main incentives to develop such a Bayesian approach: 1) to develop a probabilistic model that accounts for different sources of noises in the metrology data that have not been modeled and are not easily incorporated in the TSMF approach; and 2) to develop an alternative method to address the challenges posed by metrology problems of part surfaces with symmetric features.

Thus, the research goal would be to develop a computationally efficient Bayesian alignment approach capable of handling large metrology problem sizes within a reasonable amount of time. To achieve this goal, we propose to devise a more efficient computational method than the Markov chain Monte Carlo (MCMC) sampling procedure developed in [16] to solve the Bayesian alignment model. This task is important because the MCMC procedure in [16] is unable to solve, within a reasonable amount of time, alignment problems as large as those arising in the metrology matching problem context (see, e.g., [16] and [49]).

6.2.2 Extend our TSMF approach to Other Applications

Another promising direction is to explore the possibility of generalizing our framework to other practical applications where point set registration techniques paly a key role. For instance, remote sensing is another area on which the proposed matching framework could have potential applications. One application in remote sensing is forestry survey which shares similar two-resolution data matching characteristics as our metrology data matching problem. In forestry survey, forestry scientists need to combine a scanning LiDAR (Light Detection And Ranging) with a profiling LiDAR to survey forestry better [49]. More specifically, a scanning LiDAR covers a relatively small geographical area of about one square mile and thus provides high-resolution forestry data. In contrast, a profiling LiDAR system covers much larger geographical areas (as large as some counties in Texas), and thus results in low-resolution forestry data. Such forestry survey applications have their own special characteristics and thus require novel application of existing point set registration methods or new methodologies and algorithms to address them.

6.2.3 Parallelization of TSMF

Another promising but not trivial research direction is to parallelize the TSMF approach. With increasingly easier and cheaper access to the powerful high performance computing environment, it is worth the effort to research on the parallelization of the proposed TSMF approach as an effective parallelization would allow us to be able to solve problems with larger size and possibly in a faster speed.

REFERENCES

[1] I. Ainsworth, M. Ristic, and D. Brujic, “CAD-based measurement path planning for free-form shapes using contact probes,” *The International Journal of Advanced Manufacturing Technology*, vol. 16, no. 1, pp. 23–31, 2000.

[2] R. Minguez, A. Arias, O. Etxaniz, E. Solaberrieta, and L. Barrenetxea, “Framework for verification of positional tolerances with a 3D non-contact measurement method,” *International Journal on Interactive Design and Manufacturing*, vol. 10, pp. 85–93, May 2016.

[3] T.-S. Shen, J. Huang, and C.-H. Menq, “Multiple-sensor integration for rapid and high-precision coordinate metrology,” *IEEE/ASME Transactions on Mechatronics*, vol. 5, no. 2, pp. 110–121, 2000.

[4] H. Xia, Y. Ding, and B. K. Mallick, “Bayesian hierarchical model for combining misaligned two-resolution metrology data,” *IIE Transactions*, vol. 43, no. 4, pp. 242–258, 2011.

[5] P. J. Besl and N. D. McKay, “A method for registration of 3-D shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[6] A. Myronenko and X. Song, “Point set registration: Coherent point drift,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 12, pp. 2262–2275, 2010.

[7] F. Pfeuffer, M. Stiglmayr, and K. Klamroth, “Discrete and geometric branch and bound algorithms for medical image registration,” *Annals of Operations Research*, vol. 196, no. 1, pp. 737–765, 2012.

[8] K. Brunnstrom and A. Stoddart, "Genetic algorithms for free-form surface matching," in *Proceedings of the 13th International Conference on Pattern Recognition*, vol. 4, (Vienna, Austria), pp. 689–693, Aug 1996.

[9] J. Williams and M. Bennamoun, "A multiple view 3D registration algorithm with statistical error modeling," *IEICE Transactions on Information and Systems*, vol. 83, no. 8, pp. 1662–1670, 2000.

[10] D. Raviv, A. Dubrovina, and R. Kimmel, "Hierarchical framework for shape correspondence.," *Numerical Mathematics: Theory, Methods & Applications*, vol. 6, no. 1, pp. 245–261, 2013.

[11] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann, "Robust global registration," in *Proceedings of the Third Eurographics Symposium on Geometry Processing*, (Vienna, Austria), pp. 197–206, Jul 2005.

[12] D. W. Eggert, A. Lorusso, and R. B. Fisher, "Estimating 3-D rigid body transformations: A comparison of four major algorithms," *Machine Vision and Applications*, vol. 9, no. 5-6, pp. 272–290, 1997.

[13] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 4, pp. 376–380, 1991.

[14] X. Li and I. Guskov, "Multiscale features for approximate alignment of point-based surfaces.," in *Proceedings of the Third Eurographics Symposium on Geometry Processing*, (Vienna, Austria), pp. 217–226, Jul 2005.

[15] J. Yang, H. Li, and Y. Jia, "Go-ICP: Solving 3D registration efficiently and globally optimally," in *Proceedings of IEEE International Conference on Computer Vision*, (Sydney, Australia), pp. 1457–1464, Dec 2013.

[16] P. J. Green and K. V. Mardia, "Bayesian alignment using hierarchical models, with applications in protein bioinformatics," *Biometrika*, vol. 93, no. 2, pp. 235–254, 2006.

[17] S. Ranade and A. Rosenfeld, "Point pattern matching by relaxation," *Pattern Recognition*, vol. 12, no. 4, pp. 269–275, 1980.

[18] B. Zitova and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing*, vol. 21, no. 11, pp. 977–1000, 2003.

[19] J. Salvi, C. Matabosch, D. Fofi, and J. Forest, "A review of recent range image registration methods with accuracy evaluation," *Image and Vision Computing*, vol. 25, no. 5, pp. 578–596, 2007.

[20] O. Van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or, "A survey on shape correspondence," *Computer Graphics Forum*, vol. 30, no. 6, pp. 1681–1707, 2011.

[21] G. K. Tam, Z.-Q. Cheng, Y.-K. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X.-F. Sun, and P. L. Rosin, "Registration of 3D point clouds and meshes: A survey from rigid to nonrigid," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 7, pp. 1199–1217, 2013.

[22] B. Bellekens, V. Spruyt, R. Berkvens, and M. Weyn, "A survey of rigid 3D pointcloud registration algorithms," in *Proceedings of the Fourth International Conference on Ambient Computing, Applications, Services and Technologies*, (Rome, Italy), pp. 8–13, Aug 2014.

[23] X. Li and S. S. Iyengar, "On computing mapping of 3D objects: A survey," *ACM Computing Surveys*, vol. 47, pp. 34:1–34:45, December 2014.

[24] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proceedings International Conference 3-D Digital Imaging and Modeling*, pp. 145–

152, IEEE, 2001.

[25] Y. Liu, "Improving ICP with easy implementation for free-form surface matching," *Pattern Recognition*, vol. 37, no. 2, pp. 211–226, 2004.

[26] H. Chui and A. Rangarajan, "A new point matching algorithm for non-rigid registration," *Computer Vision and Image Understanding*, vol. 89, no. 2, pp. 114–141, 2003.

[27] T. N. Linh and H. Hiroshi, "Global iterative closet point using nested annealing for initialization," *Procedia Computer Science*, vol. 60, pp. 381 – 390, 2015.

[28] H. Pottmann, S. Leopoldseder, and M. Hofer, "Registration without ICP," *Computer Vision and Image Understanding*, vol. 95, no. 1, pp. 54–71, 2004.

[29] N. J. Mitra, N. Gelfand, H. Pottmann, and L. Guibas, "Registration of point cloud data from a geometric optimization perspective," in *Proceedings of Second Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, (Nice, France), pp. 22–31, Jul 2004.

[30] G. McNeill and S. Vijayakumar, "A probabilistic approach to robust shape matching," in *Proceedings of IEEE International Conference on Image Processing*, (Atlanta, GA), pp. 937–940, Oct 2006.

[31] W. M. Wells III, "Statistical approaches to feature-based object recognition," *International Journal of Computer Vision*, vol. 21, no. 1-2, pp. 63–98, 1997.

[32] B. Luo and E. R. Hancock, "A unified framework for alignment and correspondence," *Computer Vision and Image Understanding*, vol. 92, no. 1, pp. 26–55, 2003.

[33] B. Jian and B. C. Vemuri, "A robust algorithm for point set registration using mixture of Gaussians," in *Proceedings of the Tenth IEEE International Conference on Computer Vision*, vol. 2, (Beijing, China), pp. 1246–1251, Oct 2005.

[34] M. Lu, J. Zhao, Y. Guo, and Y. Ma, "Accelerated coherent point drift for automatic three-dimensional point cloud registration," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, pp. 162–166, February 2016.

[35] B. Eckart, K. KIm, A. Troccoli, A. Kelly, and J. Kautz, "MLMD: Maximum likelihood mixture decoupling for fast and accurate point cloud registration," in *2015 International Conference on 3D Vision*, (Lyon, France), pp. 241–249, October 2015.

[36] A. Rangarajan, H. Chui, E. Mjolsness, S. Pappu, L. Davachi, P. Goldman-Rakic, and J. Duncan, "A robust point-matching algorithm for autoradiograph alignment," *Medical Image Analysis*, vol. 1, no. 4, pp. 379–398, 1997.

[37] S. Gold, A. Rangarajan, C.-P. Lu, S. Pappu, and E. Mjolsness, "New algorithms for 2D and 3D point matching: Pose estimation and correspondence," *Pattern Recognition*, vol. 31, no. 8, pp. 1019–1031, 1998.

[38] H. Chui and A. Rangarajan, "A feature registration framework using mixture models," in *Proceedings of IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, (Hilton Head Island, South Carolina), pp. 190–197, Jun 2000.

[39] F. Meskine, N. Taleb, M. C. El-Mezouar, K. Kpalma, and A. Almhdie, "A rigid point set registration of remote sensing images based on genetic algorithms & hausdorff distance," *World Academy of Science, Engineering and Technology*, vol. 7, pp. 1095–1100, 2013.

[40] J. Luck, C. Little, and W. Hoff, “Registration of range data using a hybrid simulated annealing and iterative closest point algorithm,” in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 4, (San Francisco, CA), pp. 3739–3744, Apr 2000.

[41] H. Rashid and A. K. Turuk, “Dead reckoning localisation technique for mobile wireless sensor networks,” *IET Wireless Sensor Systems*, vol. 5, no. 2, pp. 87–96, 2015.

[42] H. Li and R. Hartley, “The 3D-3D registration problem revisited,” in *Proceedings of IEEE International Conference on Computer Vision*, (Rio de Janeiro), pp. 1–8, Oct 2007.

[43] T. M. Breuel, “Implementation techniques for geometric branch-and-bound matching methods,” *Computer Vision and Image Understanding*, vol. 90, no. 3, pp. 258–294, 2003.

[44] M. Brown, D. Windridge, and J.-Y. Guillemaut, “Globally optimal 2D-3D registration from points or lines without correspondences,” in *The IEEE International Conference on Computer Vision (ICCV)*, pp. 2111–2119, December 2015.

[45] R. E. Burkard, “Quadratic assignment problems,” *European Journal of Operational Research*, vol. 15, no. 3, pp. 283–289, 1984.

[46] S. Sahni and T. Gonzalez, “P-complete approximation problems,” *Journal of the ACM*, vol. 23, no. 3, pp. 555–565, 1976.

[47] R. Burkard, S. Karisch, and F. Rendl, “QAPLIB - A Quadratic Assignment Problem Library,” 2002. Available at `http://anjos.mgi.polymtl.ca/qaplib/`.

[48] R. G. Michael and S. J. David, *Computers and Intractability: A Guide to the Theory of NP-completeness.* San Francisco: WH Freeman & Co., 1979.

[49] H. Xia, *Bayesian hierarchical model for combining two-resolution metrology data.* Phd Dissertation, Texas A&M University, College Station, TX, United States, December 2008.

[50] C. Sandeson, "Armadillo: An open source c++ linear algebra library for fast prototyping and computationally intensive experiments," tech. rep., NICTA, 2010.

[51] C. Wang, M. M. Bronstein, A. M. Bronstein, and N. Paragios, "Discrete minimum distortion correspondence problems for non-rigid shape matching," in *Proceedings of International Conference on Scale Space and Variational Methods in Computer Vision*, vol. 6667, pp. 580–591, Lecture Notes in Computer Science, Springer, 2012.

[52] J. Arvo, *Graphics Gems III*, ch. III, pp. 117–120. San Diego, CA, USA: Academic Press, Inc., 1992.

[53] IBM, "CPLEX Online Manual – Algorithm of the populate procedure," May 2016. Available at `http://www-01.ibm.com/support/knowledgecenter/SSSA5P_12.2.0/ilog.odms.cplex.help/Content/Optimization/Documentation/CPLEX/_pubskel/CPLEX601.html`.

[54] H. T. S. Arun, K. S. and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1987.

[55] S. N. Berthold K. P. Horn, H. M. Hilden, "Closed-form solution of absolute orientation using orthonormal matrices," *Journal of the Optical Society America*, vol. 7, pp. 1127–1135, July 1988.

[56] B. K. P. Horn, “Closed-form solution of absolute orientation using unit quaternions,” *Journal of the Optical Society of America*, vol. 4, no. 4, pp. 629–642, 1987.

[57] M. W. Walker, L. Shao, and R. A. Volz, “Estimating 3-D location parameters using dual number quaternions,” *CVGIP: Image Understanding*, vol. 54, no. 3, pp. 358–367, 1991.

APPENDIX A

XIAHEUR PSEUDO-CODE

This appendix gives the Pseudo-code of XiaHeur. First, we define some useful notations. Since the datasets are labelled arbitrarily only for identification, without loss of generality, we let HR anchor point be the first point in the HR data set and denote it by h_1^a; similarly, denote the LR point in each anchor pair and unmatched HR points by l_k^a (for $k = 1, \dots, |LR|$) and h_i (for $i = 2, \dots, |HR|$), respectively. Then, the pseudo-code of XiaHeur is given below.

```
1: for each anchor pair $(h_1^a, l_k^a)$ do
2:     for each unmatched HR point $h_i$ do
3:         Find the unmatched LR point, $l_j$, with the minimum IPD difference, $\left| \|h_1^a - h_i\| - \|l_k^a - l_j\| \right|$
4:         Record this minimum IPD difference as $IPDdiff_i$
5:         if $IPDdiff_i$ is less than a prescribed threshold then
6:             $l_j$ is considered as the LR match for $h_i$
7:         else
8:             break        ▷ No valid match exists for $h_i$, and continue with the next anchor pair
9:         end if
10:    end for
11:    Record the complete correspondences found when using the current anchor pair as "reference"
12:    Record $\max_i \{IPDdiff_i\}$ as the maximum IPD difference ($maxIPDdiff$) for this complete correspondences
13: end for
14: return the complete correspondences with the smallest $maxIPDdiff$ as the solution
```

Figure A.1: Pseudo-code of XiaHeur

Note that we make minor changes to the original XiaHeur algorithm described in [4] to meet our matching goal of minimizing the largest IPD difference between the two datasets: 1) no specific threshold (i.e. $\bar{\omega}$ in Section 3.2 of [4]) is used to determine whether there exists a so-called consistent match for a specific anchor pair; instead, for each anchor pair, XiaHuer simply finds a provisional set of correspondences using the minimum largest IPD difference criterion. 2) only one best set of correspondences is selected among all provisional sets of correspondences, as opposed to keep all so-called consistent matches.

APPENDIX B

JUSTIFICATION OF SUGGESTED SEARCH SPACE PRUNING THRESHOLD FOR WIGGLY SURFACES

This appendix illustrates that the suggested pruning threshold $\overline{T} = 1.5\tau$ (and especially our definition of τ — the maximum distance between an LR point and its closest neighbor in the LR set) is also appropriate for wiggly surfaces where the measurements may not be evenly spaced throughout the part surface. Specifically, for wiggly surfaces, the preferred measurement plans for both HR and LR datasets may still maintain the measurements' evenness locally (with higher density measurements evenly distributed over high curvature areas and lower density measurements evenly distributed over not very curvy locations). Throughout this appendix, the terms measurements and points are used interchangeably.

Consider a hypothetical wiggly part surface shown in Figure B.1. This surface comprises a relatively flat section with a sparse set of evenly spaced measurements and a relatively high curvature section with a dense set of evenly spaced measurements. In Figure B.1, each cross represents an LR point and each circle denotes an HR point; each HR point sits almost at the center of its closest four surrounding LR points; h' sits a little bit closer to l^b than to l^a, and h'' is slightly closer to l^c than to l^d; and thus HR points h' and h'' should be matched to the LR points l^b and l^c, respectively. Note that this setup, just like the setup in Subsection 3.2, was created in order to have the largest possible IPD difference between correct pairs of matchings. Hereafter, for brevity, we denote the line segment and its length between two points, say points A and B, by AB and $|AB|$, respectively. The triangle inequality implies that $|h'h''| < |h'l^b| + |l^bl^c| + |l^ch''|$, which in turn implies that

$|h'h''| - |l^b l^c| < |h'l^b| + |l^c h''|$ (that is, the IPD difference between the two pairs of matching points is less than $|h'l^b| + |l^c h''|$). Therefore, a safe upper bound for the largest possible IPD difference is $|h'l^b| + |l^c h''|$. Now, due to the surface curvature, $|h'l^b| \approx 0.707\tau$ (where, 0.707τ is half of the diagonal length of a square with side length of τ); and similarly, $|h''l^c| \approx 0.353\tau$. Finally, since $0.707\tau + 0.353\tau = 1.06\tau$, we conclude that our suggested threshold value of $\overline{T} = 1.5\tau$ is a safe upper bound on the maximum possible IPD difference.

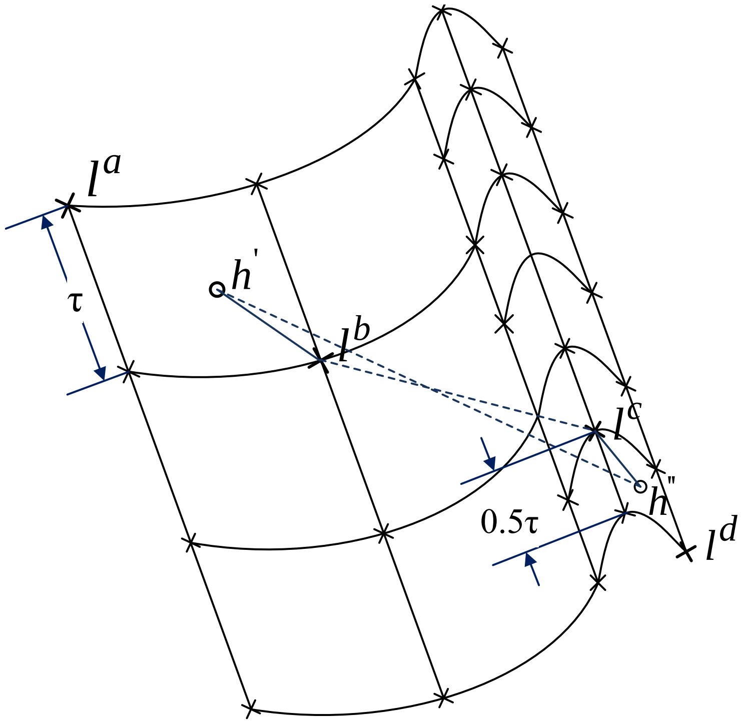


Figure B.1: Illustration of uneven case on wiggly surface

Remark: The above discussion only considers the case where one HR point is selected from the relatively flat surface section and the other HR point is selected from the relatively high curvature surface section. Two other possible cases are: 1) both HR points are selected from the relatively flat section; or 2) both HR points are selected from the high curvature section. For these two cases, one can follow the derivation discussed in Subsection 3.2 to justify 1.5τ's appropriateness. Note that, for case 2), the largest possible IPD difference under an ideal flat situation is approximately 0.707τ which needs to be reasonably relaxed to a larger amount to account for the high curvature around the two pairs of matching points, but this larger amount should still be well below 1.5τ.

APPENDIX C

VISUAL ILLUSTRATION FOR ALIGNMENT PERFORMANCE OF THREE FILTERING OPTIONS

The key characteristic of the effectiveness of a filtering option is whether the final selected solution is a correct set of correspondences. Therefore, to compare the alignment performance of the three filtering options (i.e., Filter1, Filter2 and Filter3), one main objective is to find out which filtering option(s) output a correct set of correspondences (i.e. a set of correspondences that is orientationally consistent with the underlying set of correspondences). Since, in our test datasets, the underlying true set of correspondences of the two datasets is known, the orientation correctness of the selected set of correspondences can be checked by visually comparing the selected set of correspondences to the correct set of correspondences once the two datasets are aligned. This appendix explains and illustrates in detail the visual examination of alignment performance of the three filtering options.

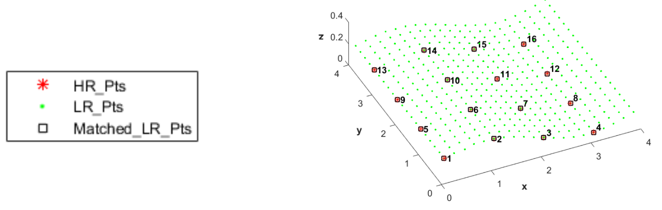
To visually evaluate the alignment performance of the three filtering options on each instance, a multi-plots figure was drawn to illustrate the alignment performance of each selected solution. The multi-plots figures for the six instances, from smallest to largest, are shown in Figures C.1 to C.6. Each multi-plots figure comprises two components: the top component is a single plot of the true set of correspondences; and the bottom component is a 3-by-3 subplot matrix. The true set of correspondences serves as a comparison basis to visually inspect whether a selected solution is orientated correctly. In the bottom 3-by-3 subplot matrix, each row corresponds to a filtering option and each column corresponds to a place during TSMF where one of filtering options was performed. More specifically, the three rows, from top to bottom, plot the solutions selected by Filter1, Filter2 and Filter3, respectively; and the three columns, from left to right, represent the following places during TSMF: *after solve-to-optimality*, *after extending solution by generalizedXiaHeur*, and *after local search*, respectively. Thus, each subplot cell in the 3-by-3 subplot matrix shows how the selected solution by the respective filtering option looks like at the respective place during TSMF. To draw each subplot cell, we first estimate the rigid body transformation (from HR dataset to LR dataset) based on the solution selected by the respective filtering option at the respective place during TSMF, and then apply the estimated transformation to align the HR dataset to the same coordinate system of the LR dataset. All LR points are denoted by a green dot; each HR point is labelled by a red asterisk along with its index, and its LR correspondence is marked by the black square closest to it.

Another way to read the 3-by-3 subplot matrix is row-wise. Specifically, each row, from left to right, represents how the respective selected solution evolves through TSMF.

In addition, under each alignment subplot cell of the 3-by-3 subplot matrix, we provide the quantitative metrics for the respective solution at that respective place during TSMF. Specifically, the quantitative metrics report its RMSE, maxIPDdiff, and whether or not that solution is detected as a reflection.

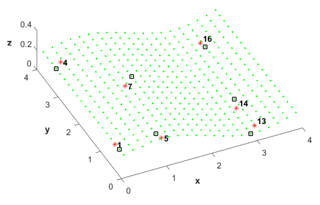
From Figures C.1 to C.6, we can see that Filter3 always selects a correct set of correspondences and it outperforms Filter1 and Filter2 for the five smallest instances and ties with Filter1 and Filter2 for the largest instance. It should be noted that, for the largest instance 100×1560, all filtering options select the same correct set of correspondences, i.e. solution 12. Similarly, for instance 64×1560, Filter1 and Filter2 select the same solution.

Correct Set of Correspondences



After SolveToOpt **After Extension** **After LocalSearch**

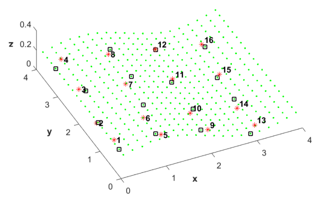
Filter1



(a) *det(R)=+1, non-reflection*

*RMSE=**RMSE.min**=0.685w*

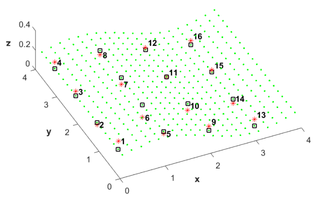
maxIPDdiff = 1.072w



(b) *det(R)=+1, non-reflection*

RMSE=0.629w

maxIPDdiff = 1.337w

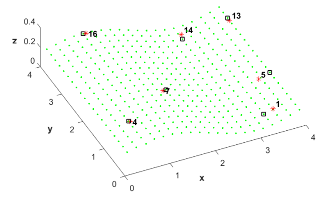


(c) *det(R)=+1, non-reflection*

RMSE=0.283w

maxIPDdiff = 0.049w

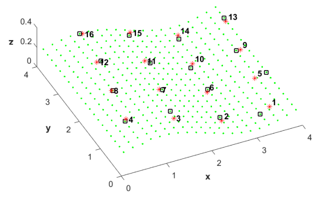
Filter2



(d) *det(R)=+1, non-reflection*

RMSE=0.689w

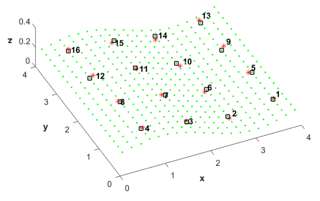
maxIPDdiff = 1.458w



(e) *det(R)=+1, non-reflection*

*RMSE=**RMSE.min**=0.589w*

maxIPDdiff = 1.458w

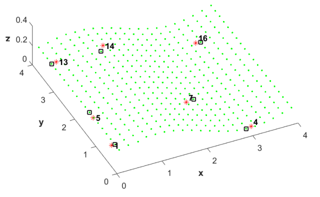


(f) *det(R)=+1, non-reflection*

RMSE=0.335w

maxIPDdiff = 0.747w

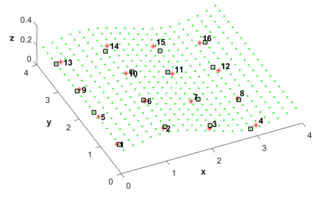
Filter3



(g) *det(R)=+1, non-reflection*

RMSE=0.786w

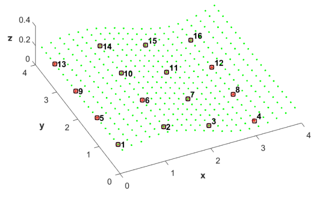
maxIPDdiff = 1.763w



(h) *det(R)=+1, non-reflection*

RMSE=0.651w

maxIPDdiff = 1.763w



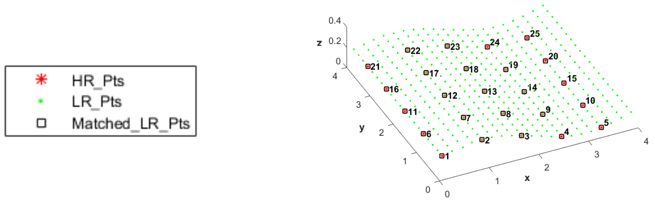
(i) *det(R)=+1, non-reflection*

*RMSE=**RMSE.min**=0.008w*

maxIPDdiff = 0.015w

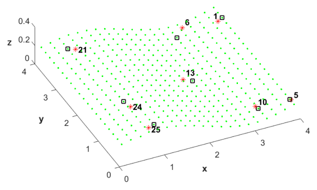
Figure C.1: Selected solutions' alignment performance of instance 16×400

Correct Set of Correspondences



After SolveToOpt **After Extension** **After LocalSearch**

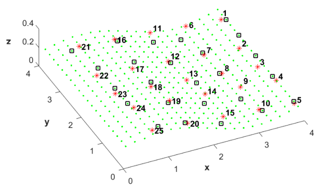
Filter1



(a) *det(R)=+1, non-reflection*

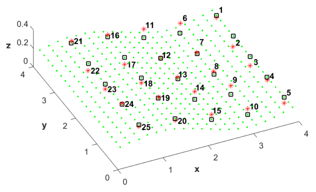
*RMSE=**RMSE.min**=0.852w*

maxIPDdiff = 1.532w



(b) *det(R)=-1, reflection*

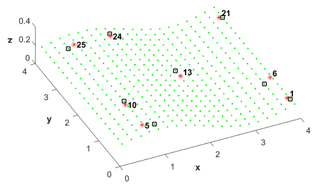
maxIPDdiff = 1.532w



(c) *det(R)=-1, reflection*

maxIPDdiff = 0.049w

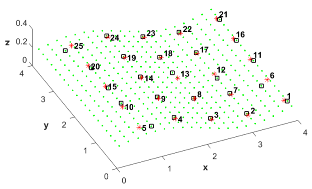
Filter2



(d) *det(R)=+1, non-reflection*

RMSE=0.935w

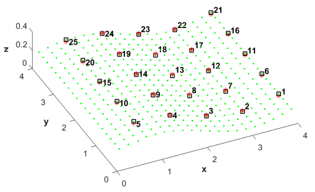
maxIPDdiff = 1.929w



(e) *det(R)=+1, non-reflection*

*RMSE=**RMSE.min**=0.935w*

maxIPDdiff = 1.929w

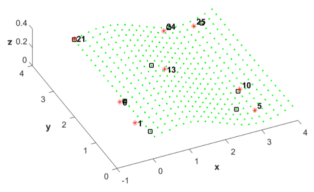


(f) *det(R)=+1, non-reflection*

RMSE=0.074w

maxIPDdiff = 0.028w

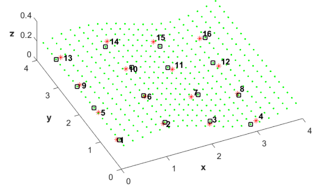
Filter3



(g) *det(R)=+1, non-reflection*

RMSE=1.409w

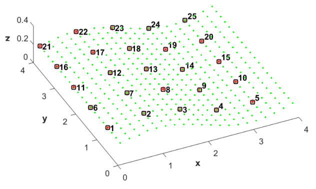
maxIPDdiff = 2.892w



(h) *det(R)=+1, non-reflection*

RMSE=1.409w

maxIPDdiff = 2.892w



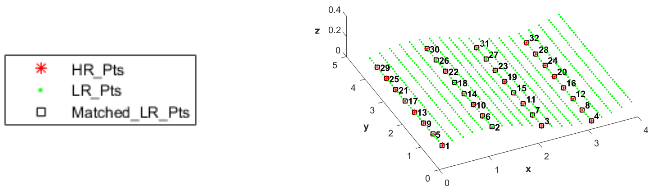
(i) *det(R)=+1, non-reflection*

*RMSE=**RMSE.min**=0.009w*

maxIPDdiff = 0.022w

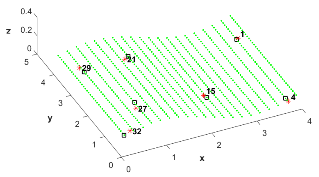
Figure C.2: Selected solutions' alignment performance of instance 25×400

Correct Set of Correspondences



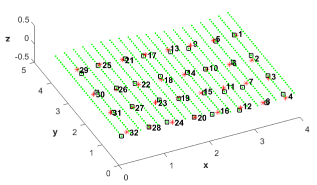
After SolveToOpt **After Extension** **After LocalSearch**

Filter1


(a) *det(R)=+1, non-reflection*

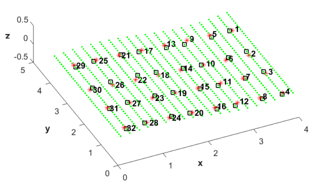
*RMSE=**RMSE.min**=0.953w*

maxIPDdiff = 2.070w


(b) *det(R)=+1, non-reflection*

RMSE=0.992w

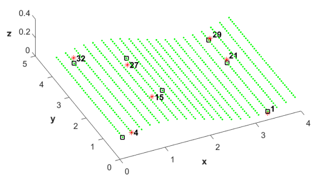
maxIPDdiff = 2.333w


(c) *det(R)=+1, non-reflection*

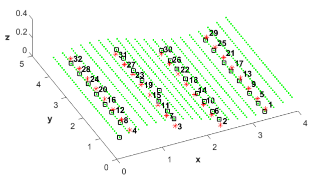
RMSE=0.759w

maxIPDdiff = 1.075w

Filter2

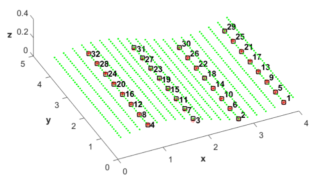

(d) *det(R)=-1, reflection*

maxIPDdiff = 2.134w


(e) *det(R)=+1, non-reflection*

*RMSE=**RMSE.min**=0.886w*

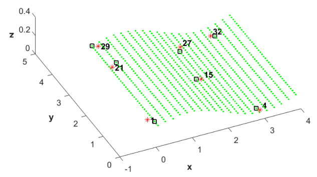
maxIPDdiff = 2.134w


(f) *det(R)=+1, non-reflection*

RMSE= 0.070w

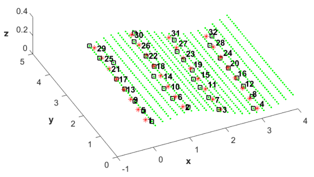
maxIPDdiff = 0.045w

Filter3


(g) *det(R)=+1, non-reflection*

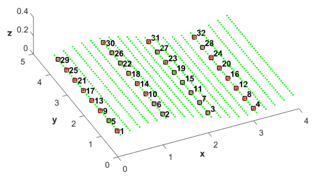
RMSE= 1.608w

maxIPDdiff = 2.580w


(h) *det(R)=+1, non-reflection*

RMSE= 1.222w

maxIPDdiff = 2.580w

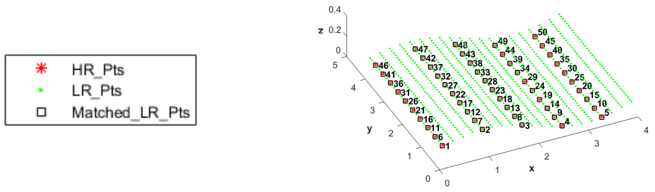

(i) *det(R)=+1, non-reflection*

*RMSE=**RMSE.min**=0.020w*

maxIPDdiff = 0.046w

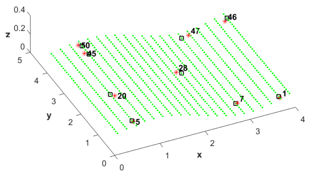
Figure C.3: Selected solutions' alignment performance of instance 32×800

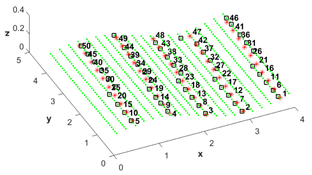
Correct Set of Correspondences

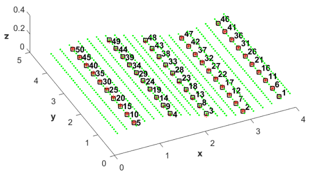


After SolveToOpt **After Extension** **After LocalSearch**

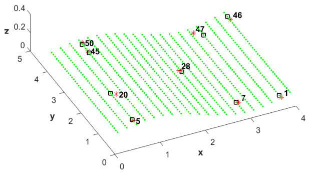
Filter1

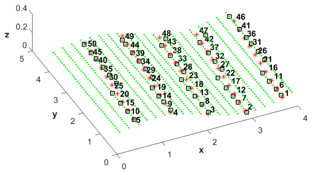

(a) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.945w*
maxIPDdiff = 2.020w

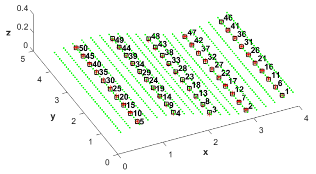

(b) *det(R)=+1, non-reflection*
RMSE=0.897w
maxIPDdiff = 2.203w


(c) *det(R)=+1, non-reflection*
RMSE=0.079w
maxIPDdiff = 1.054w

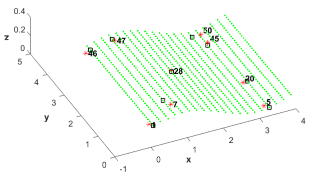
Filter2

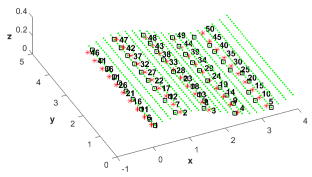

(d) *det(R)=+1, non-reflection*
RMSE=1.148w
maxIPDdiff = 2.674w


(e) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.845w*
maxIPDdiff = 2.674w


(f) *det(R)=+1, non-reflection*
RMSE= 0.079w
maxIPDdiff = 0.054w

Filter3


(g) *det(R)=+1, non-reflection*
RMSE=1.675w
maxIPDdiff = 3.997w


(h) *det(R)=+1, non-reflection*
RMSE= 1.218w
maxIPDdiff = 3.997w

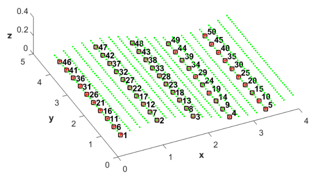
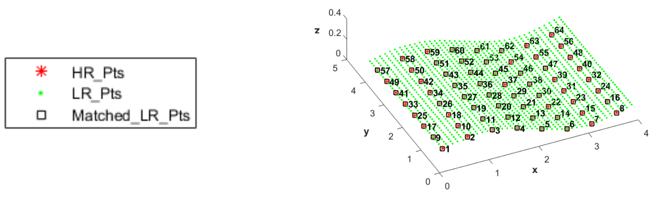

(i) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.020w*
maxIPDdiff = 0.050w

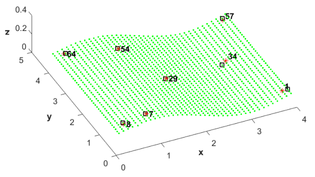
Figure C.4: Selected solutions' alignment performance of instance 50 × 800

Correct Set of Correspondences

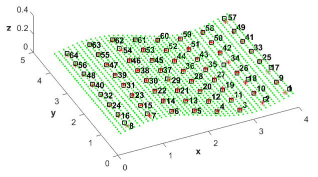


After SolveToOpt **After Extension** **After LocalSearch**

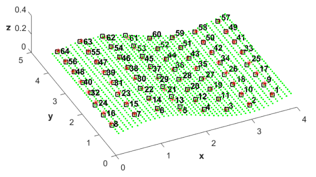
Filter1



(a) *det(R)=+1, non-reflection* *RMSE=**RMSE.min**=0.715w* *maxIPDdiff = 1.439w*

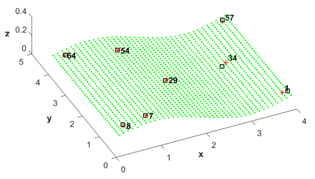


(b) *det(R)=+1, non-reflection* *RMSE=**RMSE.min**=0.436w* *maxIPDdiff = 1.439w*

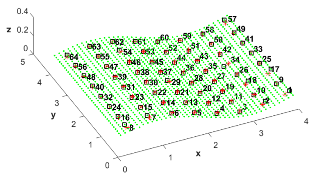


(c) *det(R)=+1, non-reflection* *RMSE=0.378w* *maxIPDdiff = 0.984w*

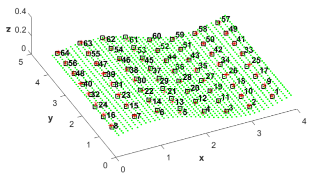
Filter2



(a) *det(R)=+1, non-reflection* *RMSE=**RMSE.min**=0.715w* *maxIPDdiff = 1.439w*

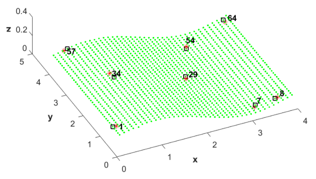


(b) *det(R)=+1, non-reflection* *RMSE=**RMSE.min**=0.436w* *maxIPDdiff = 1.439w*

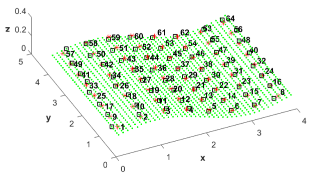


(c) *det(R)=+1, non-reflection* *RMSE=0.378w* *maxIPDdiff = 0.984w*

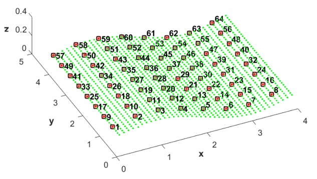
Filter3



(d) *det(R)=+1, non-reflection* *RMSE=1.115w* *maxIPDdiff = 2.260w*



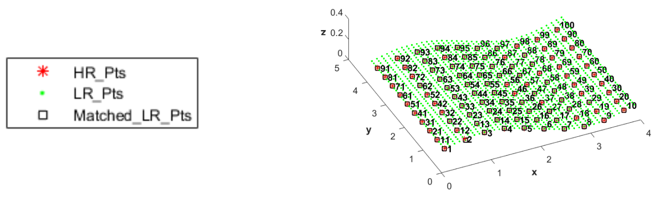
(e) *det(R)=+1, non-reflection* *RMSE=0.874w maxIPDdiff = 2.264w*



(f) *det(R)=+1, non-reflection* *RMSE=**RMSE.min**=0.020w* *maxIPDdiff = 0.059w*

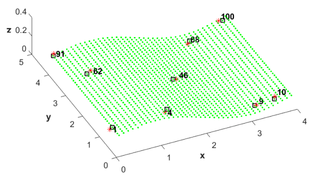
Figure C.5: Selected solutions' alignment performance of instance 64 × 1560

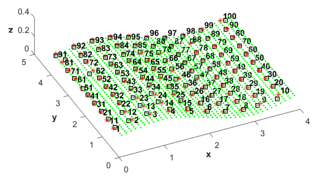
Correct Set of Correspondences

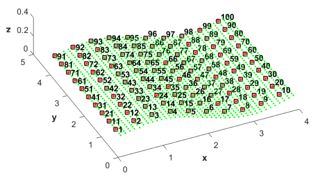


After SolveToOpt **After Extension** **After LocalSearch**

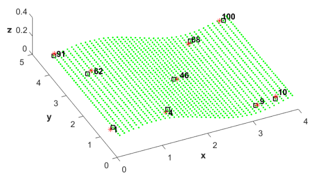
Filter1

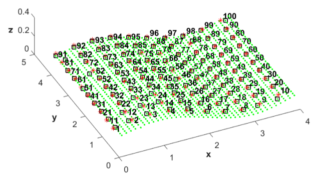

(a) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.916w*
maxIPDdiff = 2.070w

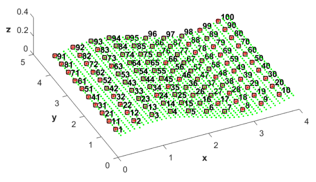

(b) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.555w*
maxIPDdiff = 2.249w


(c) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.019w*
maxIPDdiff = 0.057w

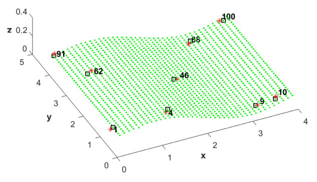
Filter2

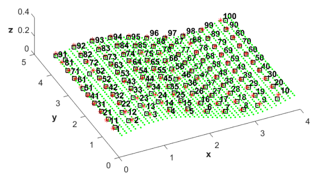

(d) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.916w*
maxIPDdiff = 2.070w


(e) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.555w*
maxIPDdiff = 2.249w


(f) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.019w*
maxIPDdiff = 0.057w

Filter3


(g) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.916w*
maxIPDdiff = 2.070w


(h) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.555w*
maxIPDdiff = 2.249w

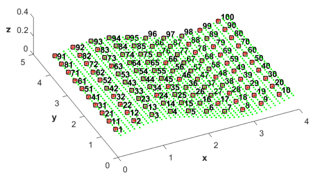

(i) *det(R)=+1, non-reflection*
*RMSE=**RMSE.min**=0.019w*
maxIPDdiff = 0.057w

Figure C.6: Selected solutions' alignment performance of instance 100×1560