Partial-to-Partial Shape Matching with Geometric Consistency

Viktoria Ehm[1,2]   Maolin Gao[1,2]   Paul Roetzer[3]   Marvin Eisenberger[1,2]
Daniel Cremers[1,2]   Florian Bernard[3]
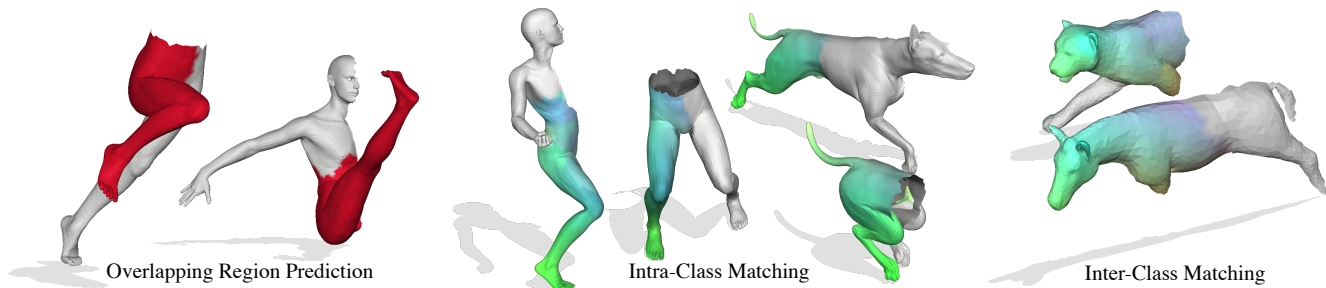[1]Technical University of Munich   [2]Munich Center for Machine Learning   [3]University of Bonn

Figure 1. We present the first **geometrically consistent partial-to-partial shape matching** solution. Our approach can find correspondences between partial shapes and thus determine the overlapping region between these shapes (left). We can find correspondences within shape classes (middle), and across shape classes (right). For the latter, we introduce a new inter-class partial-to-partial matching dataset.

Abstract

Finding correspondences between 3D shapes is an important and long-standing problem in computer vision, graphics and beyond. A prominent challenge are partial-to-partial shape matching settings, which occur when the shapes to match are only observed incompletely (e.g. from 3D scanning). Although partial-to-partial matching is a highly relevant setting in practice, it is rarely explored. Our work bridges the gap between existing (rather artificial) 3D full shape matching and partial-to-partial real-world settings by exploiting geometric consistency as a strong constraint. We demonstrate that it is indeed possible to solve this challenging problem in a variety of settings. For the first time, we achieve geometric consistency for partial-to-partial matching, which is realized by a novel integer non-linear program formalism building on triangle product spaces, along with a new pruning algorithm based on linear integer programming. Further, we generate a new inter-class dataset for partial-to-partial shape-matching. We show that our method outperforms current SOTA methods on both an established intra-class dataset and our novel inter-class dataset. The code of this work is publicly available[1].

1. Introduction

Finding correspondences between 3D shapes is an important topic in computer vision to solve tasks such as shape interpolation or texture transfer. While there is a lot of research on 3D shape matching, these works predominantly focus on full-to-full shape matching [11, 19, 25, 33, 39, 43, 44], or on partial-to-full shape matching [3, 11, 18, 29, 36, 37]. Therefore, they are not applicable to the complex problem of matching two partial shapes. The problem of partial-to-partial matching is difficult because not every part of one shape will necessarily be matched to the other shape and vice versa. At the same time, we must encourage specific parts to be matched to avoid the trivial solution of not matching anything. There is only little research on the topic of partial-to-partial shape matching.

The current SOTA method [3] requires annotated data for training, which is often unavailable rendering many applications infeasible. While the method [37] is learning-free and thus does not rely on annotations, it requires closing holes in the partial shapes to generate a full-to-full shape matching, resulting in poor results for shapes with low overlaps. It uses the property of geometric consistency to generate its results. While many methods also make use of the connectivity of meshes to generate geometrically consistent results for full-to-full [39, 43, 44] or partial-to-full [18] matching, to date, it has not been applied to partial-to-partial matching. In this work, we incorporate geometric consistency into

---

[1] https://github.com/vikiehm/gc-ppsm

a partial-to-partial shape matching setting by formulating and solving a non-linear integer program with a novel pruning algorithm. Most current SOTA shape matching methods use deep functional maps [2, 10, 11, 16, 25] to find correspondences between 3D shapes. Functional maps are not directly applicable to partial-to-partial shape matching because they cannot predict the overlapping region. We integrate deep features from the SOTA feature extractor [11] into our partial-to-partial integer program. Current partial-to-partial shape matching approaches focus solely on intra-class datasets [3]. Instead, we present a new inter-class dataset based on SMAL [47] to expand the scope of partial-to-partial shape matching evaluation. We summarize our contributions as follows:

- We present a **geometrically consistent partial-to-partial** shape matching formalism, that fuses **SOTA deep features** with a **non-linear integer programming approach** to combine the best of both worlds.
- We introduce a **pruned search algorithm** to solve the integer program in a reasonable time.
- We provide a new **inter-class** partial-to-partial dataset based on the SMAL dataset.
- We show that our approach **outperforms** SOTA supervised deep learning and combinatorial optimization algorithms in both intra-class and inter-class settings.

2. Related Work

While the entire field of 3D shape matching is vast, this paper focuses on closely related work. For a more comprehensive overview, interested readers are referred to the following surveys [38, 41].

2.1. Integer and Combinatorial Optimization in Shape Matching

Many shape correspondence formalisms involve integer or binary variables, for which permutation matrices are a popular representation. Depending on the objective function (linear or quadratic), a linear assignment problem (LAP) can be solved to global optimality in polynomial time, e.g. using Bertsekas' Auction algorithm [7], while the quadratic assignment problem (QAP) is NP-hard [31, 35] and thus several relaxed formulations have been proposed [5, 17, 22, 24].

Instead of matching vertices to vertices, the authors of [39, 43, 44] proposed to match mesh faces to faces by minimizing an elastic energy, which models the bending and stretching of the triangles locally, while constraining the matching to be geometrically consistent [9]. Geometric consistency in this case means that neighboring triangles are matched to neighboring triangles, which naturally leads to smooth results. Respective approaches leads to orientation-preserving matchings by design, but they involve solving a difficult instance of a linear integer program (i.e. its con-

straints are not totally unimodular), which limits the size of the problem it can tackle [45]. However, in practice, high resolution shapes are often available, and the mandatory down-sampling step unavoidably leads to a loss of valuable 3D information, and hence sub-optimal correspondences. Recently, [37] has proposed a tailored linear integer solver to improve the method's scalability in trade of compromising the global optimality.

Most recently, shape matching problems were also modeled as mixed integer programs [6, 20], which can (often) be solved to global optimality by branch-and-bound methods. Though their potential to be applied to partial shapes matching has been shown in their paper, an adequate quantitative study is not presented.

2.2. Partial-to-Full Shape Matching

Unlike matching full shapes, partial shapes often exhibit substantial non-isometric and irregular characteristics, making them a challenging problem to address. Nevertheless, investigating this problem is essential, given its significant practical relevance.

In [36] a method for matching partial shapes to full template shapes has been introduced. It extended the functional map framework, which has achieved great success in full-to-full matching [14, 28, 33]. This partial functional maps framework was extended to multiple partial shapes [27], where every partial shape can be further affected by potential overlaps with other parts, the possibility of missing or redundant parts, and the presence of clutter. Clutter is also addressed in [13]. Recent deep learning methods use a functional map layer [3, 11] to determine correspondences between a partial and a template shape. An inherent limitation shared by many methods based on functional maps is to get desired structural properties, like bijectivity and smoothness, as observed in previous studies [34, 42]. In [37] a scalable solver was introduced for the formalism of [39, 43, 44], which ensures smoothness utilizing geometric consistency. Respective works are also applicable to partial-to-full shape matching (by closing the holes of the partial shapes), which however increases the difficulty to solve this problem. In [18], these works were extended such that geometrically consistent partial-to-full matchings can be produced without requiring closing holes beforehand. While this work is closest to our approach, it is not applicable to partial-to-partial shape matching as the authors expect that the partial shape is completely matched to (parts of) the full shape.

2.3. Partial-to-Partial Shape Matching

Even though partial-to-partial shape matching has high practical relevance, there is still few research about it. Litany et al. [29] provide a theoretical concept for partial-to-partial shape matching, but no practical analysis. While [4]

investigates overlapping regions for pairs of partial shapes, it does not provide a method for shape matching. DPFM [3] and Sm-comb [37] are the only approaches enabling partial-to-partial shape matching enabled by deep learning (DPFM) and combinatorial optimization (Sm-comb). While DPFM needs labels for training, these are often not accessible in real-world examples. Sm-comb does not need labels, however, it requires the closing of holes in shapes, as it is only applicable to water-tight shapes. Instead, we propose a novel partial-to-partial shape matching formalism without requiring 3D ground truth labels and closing holes.

3. Partial-to-Partial Shape Matching

Our primary innovation centers around a novel integer programming framework designed for achieving geometrically consistent partial-to-partial shape matching. We show an overview of our method in Figure 2.

3.1. Notation

Shapes. We define an oriented manifold shape as $\mathcal{X} = \left(\mathbf{V}^{(\mathcal{X})}, \mathbf{F}^{(\mathcal{X})}\right)$, which consists of vertices $\mathbf{V}^{(\mathcal{X})} := \left\{\mathbf{v}_i^{(\mathcal{X})} \in \mathbb{R}^3 | 1 \le i \le |\mathbf{V}^{(\mathcal{X})}|\right\}$ and triangles $\mathbf{F}^{(\mathcal{X})} \subset \mathbf{V}^{(\mathcal{X})} \times \mathbf{V}^{(\mathcal{X})} \times \mathbf{V}^{(\mathcal{X})}$. Moreover, $\mathbf{E}^{(\mathcal{X})} \subset \mathbf{V}^{(\mathcal{X})} \times \mathbf{V}^{(\mathcal{X})}$ are edges induced by triangles. The set of edges is divided in boundary $\mathbf{E}_b^{(\mathcal{X})} \subset \mathbf{E}^{(\mathcal{X})}$ and non-boundary $\mathbf{E}_{nb}^{(\mathcal{X})} \subset \mathbf{E}^{(\mathcal{X})}$ edges.

Degenerate Triangles. To allow shrinking and expanding in our matching, which is necessary for different discretization and non-isometric shape pairs, we allow to match triangles from one shape not only to triangles from the other shape, but also to vertices and edges from the other shape. For this reason, we define a degenerate triangle as either a vertex $\mathbf{v} \in \mathbf{V}^{(\mathcal{X})}$ or an edge $\mathbf{e} \in \mathbf{E}^{(\mathcal{X})}$ and collect these in the set of extended triangles which we define as $\bar{\mathbf{F}}^{(\mathcal{X})} := \mathbf{V}^{(\mathcal{X})} \cup \mathbf{E}^{(\mathcal{X})} \cup \mathbf{F}^{(\mathcal{X})}$.

Inner Triangles. We utilize the property that a non-boundary edge $\mathbf{e} \in \mathbf{E}_{nb}^{(\mathcal{X})}$ is common to two adjacent triangles. In contrast, a boundary edge $\mathbf{e} \in \mathbf{E}_b^{(\mathcal{X})}$ is adjacent to only one triangle. As the shape $\mathcal{X}$ is oriented, non-boundary edges are found in pairs with opposing orientations within adjacent triangles $(\mathbf{v}_i^{(\mathcal{X})}, \mathbf{v}_j^{(\mathcal{X})})$ and $(\mathbf{v}_j^{(\mathcal{X})}, \mathbf{v}_i^{(\mathcal{X})})$. We use the notation $\mathcal{O}^{(\mathcal{X})} : \mathbf{E}^{(\mathcal{X})} \times \mathbf{F}^{(\mathcal{X})} \to \{-1, 0, 1\}$ to represent the orientation operator between an edge $\mathbf{e} = (\mathbf{v}_1', \mathbf{v}_2')$ and a triangle $\mathbf{f} = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$. The orientation operator represents the orientation of an edge $\mathbf{e}$ in dependence of a triangle $\mathbf{f}$, and whether this edge is adjacent to the triangle (refer to [18, 37, 43, 44] for further details).

3.2. Optimization Problem

We present a method to match two oriented and manifold shapes, denoted as $\mathcal{X}$ and $\mathcal{Y}$, where both shapes have boundaries. With this approach we extend the full-to-full matching algorithm from [39, 43, 44] and the partial-to-full matching from [18] to a partial-to-partial formulation.

Product Space We define product edges $\mathbf{E} := \left(\mathbf{E}^{(\mathcal{X})} \times \mathbf{E}^{(\mathcal{Y})}\right) \cup \left(\mathbf{E}^{(\mathcal{X})} \times \mathbf{V}^{(\mathcal{Y})}\right) \cup \left(\mathbf{V}^{(\mathcal{X})} \times \mathbf{E}^{(\mathcal{Y})}\right)$ by the combination of an edge with an edge or an edge with a vertex. Additionally, we define product triangles $\mathbf{F} := \left(\bar{\mathbf{F}}^{(\mathcal{X})} \times \mathbf{F}^{(\mathcal{Y})}\right) \cup \left(\mathbf{F}^{(\mathcal{X})} \times \bar{\mathbf{F}}^{(\mathcal{Y})}\right)$. Every product triangle represents a matching between shape $\mathcal{X}$ and shape $\mathcal{Y}$. To encode these product triangles we define a binary vector $\mathbf{\Gamma} \in \{0, 1\}^{|\mathbf{F}|}$ that forms the matching for which we optimize for. The value $\Gamma_i = 1$ means that the (degenerate or non-degenerate) triangles $f_j \in \bar{\mathbf{F}}^{(\mathcal{X})}$ and $f_k \in \bar{\mathbf{F}}^{(\mathcal{Y})}$, represented by the product triangle $\mathbf{F}_i = (f_j, f_k)$, are matched.

Uniqueness Constraints. In the context of partial shape matching, triangles can either be matched or not be matched. We denote matched triangles as *overlapping*. Our optimization approach ensures that each triangle within the partial shapes is matched at most once, emphasizing the uniqueness of correspondences in the matching process. Following the approach of prior studies [18, 43], we express this condition through the use of projection matrices. $\boldsymbol{\pi}^{(\mathcal{X})} \in \{0, 1\}^{|\mathbf{F}^{(\mathcal{X})}| \times |\mathbf{F}|}$ and $\boldsymbol{\pi}^{(\mathcal{Y})} \in \{0, 1\}^{|\mathbf{F}^{(\mathcal{Y})}| \times |\mathbf{F}|}$. For every combination between a triangle $\mathbf{f}_i^{(\mathcal{X})}$ in $\mathcal{X}$ and a product triangle $\mathbf{f}_j$, the projection entry is defined as

$$\boldsymbol{\pi}_{i,j}^{(\mathcal{X})} := \begin{cases} 1 & \text{if} \quad \mathbf{f}_j \text{ contains } \mathbf{f}_i^{(\mathcal{X})} \\ 0 & \text{else.} \end{cases} \quad (1)$$

The definition of $\boldsymbol{\pi}^{(\mathcal{Y})} \in \{0, 1\}^{|\mathbf{F}^{(\mathcal{Y})}| \times |\mathbf{F}|}$ follows similarly. Whereas in a full matching every triangle must be matched exactly once, i.e., $\boldsymbol{\pi}^{(\mathcal{X})}\mathbf{\Gamma} = \mathbf{1}$, in the partial-to-partial matching scenario we merely impose that each triangle is matched *at most* once. The uniqueness constraints are defined as

$$\boldsymbol{\pi}^{(\mathcal{X})}\mathbf{\Gamma} \le \mathbf{1}, \quad \boldsymbol{\pi}^{(\mathcal{Y})}\mathbf{\Gamma} \le \mathbf{1}, \quad (2)$$

where $\mathbf{1}$ is a vector of ones.

Minimum Triangle Matching. When minimizing the costs (which are non-negative and will be introduced in the paragraph **Cost Function** below) without considering additional constraints, the best solution would be that no triangles are matched. To avoid this degenerate solution, we enforce that at least one non-degenerate non-boundary triangle from each shape $\mathcal{X}$ and $\mathcal{Y}$ are matched. These non-degenerate non-boundary triangle-to-triangle matchings are
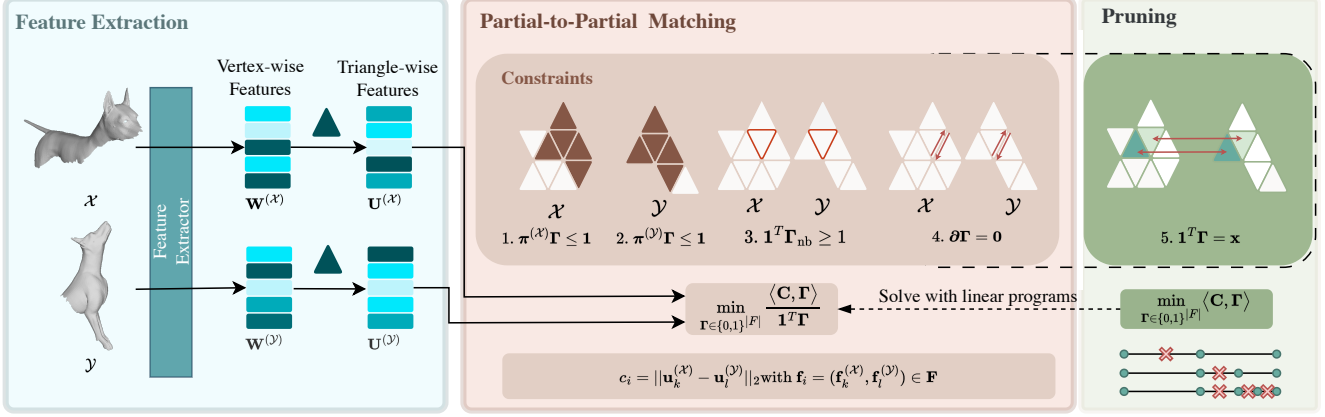
Figure 2. Overview over our **matching pipeline**: The 3D meshes are fed into a feature extractor that returns vertex-wise features. From these vertex-wise features, we generate triangle-wise features to make them accessible for our algorithm. With these features, we define an integer program that ensures that every triangle is matched at most once for both shapes, that at least one non-boundary triangle is matched (per shape), and that neighboring relationships for inner triangles are fulfilled. We tackle this non-linear integer program by solving a subset of integer linear programs (ILPs) for a specific number of matchings.

encoded in $\mathbf{\Gamma}_{\mathrm{nb}} \subset \mathbf{\Gamma}$. To ensure that at least one of these matchings exists, we constrain

$$\mathbf{1}^T \mathbf{\Gamma}_{\mathrm{nb}} \geq 1. \tag{3}$$

Neighboring Constraint. The use of the previous constraint would lead to exactly one matching (match one non-boundary triangle of shape $\mathcal{X}$ to one non-boundary triangle of shape $\mathcal{Y}$). Therefore, we define a constraint that ensures that neighboring non-boundary triangles remain neighbors after matching – this guarantees *geometric consistency in the interior*. We define $\mathbf{\mathring{E}} = (\mathbf{E}_{nb}^{\mathcal{X}} \times \mathbf{E}_{nb}^{\mathcal{Y}}) \cup (\mathbf{E}_{nb}^{\mathcal{X}} \times \mathbf{V}_{nb}^{\mathcal{Y}}) \cup (\mathbf{V}_{nb}^{\mathcal{X}} \times \mathbf{E}_{nb}^{\mathcal{Y}})$ as the set of interior product edges. For a given product triangle $\mathbf{f}_i \in \mathbf{F}$, we assign the orientation $\{-1, 1\}$ to interior product edges that are adjacent to product triangles $\mathbf{f}_j \in \mathbf{F}$, and orientation 0 if they do not coincide. This can be achieved by applying the product orientation operator $\mathcal{O}$ to edges $\mathbf{e}_i$ and triangles $\mathbf{f}_j$ in the 4-dimensional product space (cf. [43] for further insights). We define the *product orientation matrix* $\boldsymbol{\partial} \in \{-1, 0, 1\}^{|\mathbf{\mathring{E}}| \times |\mathbf{F}|}$ as

$$\partial_{i,j} := \mathcal{O}(\mathbf{e}_i, \mathbf{f}_j), \tag{4}$$

By setting $\boldsymbol{\partial}\mathbf{\Gamma} = \mathbf{0}$, we can ensure that the neighboring constraint is fulfilled, and, consequently, non-boundary triangles are matched in a geometrically consistent fashion.

Cost Function. Most feature extractors return vertex-wise features, i.e., a feature matrix $\mathbf{W}^{(\mathcal{X})} \subset \mathbb{R}^{|\mathbf{V}^{(\mathcal{X})}| \times d}$ which contains a d-dimensional feature vector for every vertex in $\mathbf{V}^{(\mathcal{X})}$. Since our algorithm operates on triangles, we set up a matrix $\mathbf{U}^{(\mathcal{X})} \subset \mathbb{R}^{|\mathbf{\bar{F}}^{(\mathcal{X})}| \times d}$ that contains a feature vector for every triangle in $\mathbf{\bar{F}}^{(\mathcal{X})}$. For every triangle, the

features are computed as the mean of the features of its vertices. We define the cost vector $\mathbf{C} \in \mathbb{R}^{|\mathbf{F}|}$, which contains the matching costs for each product triangle. The cost c_i to match a product triangle $\mathbf{f}_i = (\mathbf{f}_k^{(\mathcal{X})}, \mathbf{f}_l^{(\mathcal{Y})}) \in \mathbf{F}$ is defined by the L2 norm of the features $\mathbf{u}_k^{(\mathcal{X})}, \mathbf{u}_l^{(\mathcal{Y})}$, i.e. $c_i = ||\mathbf{u}_k^{(\mathcal{X})} - \mathbf{u}_l^{(\mathcal{Y})}||_2$. The cost vector $\mathbf{C}$ is normalized such that all $c_i \in [0, 1]$. To ensure that the algorithm is not biased towards a small number of matchings, we further normalize the objective function by the number of elements that are matched in total, i.e., we minimize the mean cost instead of the sum. Our integer program reads

$$\min_{\mathbf{\Gamma} \in \{0,1\}^{|F|}} \frac{\langle \mathbf{C}, \mathbf{\Gamma} \rangle}{\mathbf{1}^T \mathbf{\Gamma}} \tag{5}$$
$$\text{s.t.} \quad \boldsymbol{\partial}\mathbf{\Gamma} = \mathbf{0}, \boldsymbol{\pi}^{(\mathcal{X})}\mathbf{\Gamma} \leq \mathbf{1}, \boldsymbol{\pi}^{(\mathcal{Y})}\mathbf{\Gamma} \leq \mathbf{1}, \mathbf{1}^T \mathbf{\Gamma}_{\mathrm{nb}} \geq 1.$$

A further difficulty, in addition to the binary constraints, is that the division by the sum over all entries of $\mathbf{\Gamma}$ makes the objective non-linear. We solve this integer fractional minimization problem by means of a sequence of corresponding integer linear programs. To be specific, we formulate every integer linear program by adding a constraint that fixes the number of elements that have value 1 in $\mathbf{\Gamma}$ to a number $x \in \{1, 2, 3, \ldots, (|\mathbf{F}^{(\mathcal{X})}| + |\mathbf{F}^{(\mathcal{Y})}|)\}$. As such, for fixed x, the integer linear program reads

$$\min_{\mathbf{\Gamma} \in \{0,1\}^{|F|}} \langle \mathbf{C}, \mathbf{\Gamma} \rangle \tag{6}$$
$$\text{s.t.} \quad \boldsymbol{\partial}\mathbf{\Gamma} = \mathbf{0}, \boldsymbol{\pi}^{(\mathcal{X})}\mathbf{\Gamma} \leq \mathbf{1}, \boldsymbol{\pi}^{(\mathcal{Y})}\mathbf{\Gamma} \leq \mathbf{1}, \mathbf{1}^T \mathbf{\Gamma}_{\mathrm{nb}} \geq 1,$$
$$\mathbf{1}^T \mathbf{\Gamma} = x.$$

By iterating over all x and comparing the objectives (normalized by x), we can find the globally optimal solution of the fractional minimization problem (5).
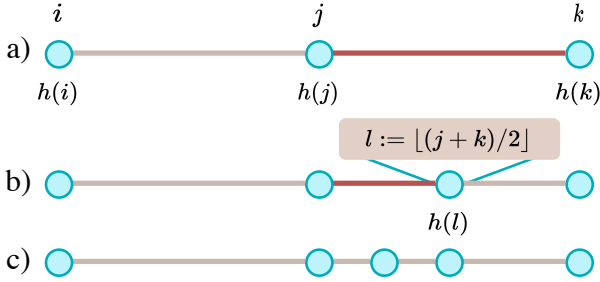
Figure 3. Illustration of our **pruning algorithm**: a) We first calculate an ILP for each initial value for x. Then, we identify intervals with potentially smaller $h(x)$ than the current minimum. b) We define new points in search space for x and calculate ILPs for these points only. c) We continue till all necessary intervals are solved.

3.3. Search Space Reduction

We define the minimum objective value of the optimization problem (6) for a specific x as $g(x)$. The normalized objective value is $h(x) := \frac{g(x)}{x}$.

To find x^* that minimizes $h(x)$, we can, in principle, compute $g(x)$ for all possible $x \in \mathbf{I} = \{1, \ldots, |\mathbf{F}^{(\mathcal{X})}| + |\mathbf{F}^{(\mathcal{Y})}|\}$, and then select x^* according to the smallest value of h. We avoid exhaustively solving all linear programs by employing a pruning algorithm. To this end, we choose a set of starting points $\mathbf{S} \subset \mathbf{I}$ and solve the optimization problem (6) for all $x \in \mathbf{S}$. For every interval given by two consecutive points $j, k \in \mathbf{S}$, we check whether the set $[j, k] \cap \mathbb{N}$ may potentially contain smaller values than $\min_{x' \in \mathbf{S}} h(x')$.

To identify those intervals with smaller $h(x)$, we analyse the difference between $g(x)$ and $g(x+1)$. Increasing x by one is equivalent to allowing an additional product triangle in the matching. This product triangle's cost is at most one (since $c_i \leq 1$ for $c_i \in \mathbf{C}$). Moreover, this additional triangle can always be matched to a boundary edge, as boundary edges have no orientation constraints. Thus the maximal cost of adding one product triangle is bounded by 1.

Hence, we know $g(x+1) \leq g(x) + 1$, which gives rise to $g(x+1) - g(x) \leq 1$. We observe the similarity between our bound and the Lipschitz constant of continuous functions. It follows that $g(k) - g(j) \leq k - j$ for any set $[j, k] \cap \mathbb{N}$, and $h(k) - h(j) \leq \sum_{i=j}^{k-1} \frac{1}{i}$ (see supplementary). Finally, for a set $[j, k] \cap \mathbb{N}$ the potentially minimal objective is given as $h(j) - h(k) - \sum_{i=j}^{k-1} \frac{1}{i}$. If this value is smaller than $\min_{x' \in \mathbf{I}} h(x')$, we add $l := \lfloor \frac{j+k}{2} \rfloor$ to $\mathbf{S}$. We repeat this process until all intervals are determined to be impossible to contain smaller minima or all possible values for x have been visited, i.e., $\mathbf{S} = \{1, \ldots, |\mathbf{F}^{(\mathcal{X})}| + |\mathbf{F}^{(\mathcal{Y})}|\}$. See Figure 3 for an illustration of our pruning algorithm.
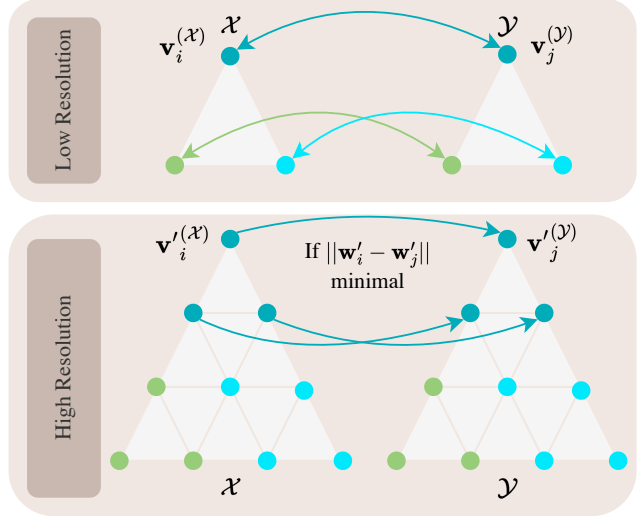

Figure 4. Our **upsampling strategy** via nearest neighbor search: Given a matching between vertices $\mathbf{v}_i^{(\mathcal{X})}$ and $\mathbf{v}_j^{(\mathcal{Y})}$ on lower resolution, we determine the matchings in high resolution by only considering the upsampled vertices (dark blue). A vertex $\mathbf{v}'_i^{(\mathcal{X})}$, which was downsampled on $\mathbf{v}_i^{(\mathcal{X})}$ can only be matched on the upsampled vertices of $\mathbf{v}_j^{(\mathcal{Y})}$ (dark blue vertices in $\mathcal{Y}$). We choose this vertex $\mathbf{v}'_j^{(\mathcal{Y})}$ that has the smallest distance in feature space of $\mathbf{W}^{(\mathcal{X})}$ and $\mathbf{W}^{(\mathcal{Y})}$.

3.4. Implementation Details

3.4.1 Upsampling to Original Resolution

To achieve reasonable running times, we perform a coarse-to-fine processing, in which we solve Problem (5) for meshes with lower resolution and then upsample the low-resolution matching to a high-resolutions matching. Let $\mathbf{v}_i^{(\mathcal{X})} \leftrightarrow \mathbf{v}_j^{(\mathcal{Y})}$ be the vertex matchings of the low resolution shapes. For upsampling, we ensure that upsampled vertices of $\mathbf{v}_i^{(\mathcal{X})}$ in shape $\mathcal{X}$ are matched to upsampled vertices of the corresponding low resolution vertex $\mathbf{v}_j^{(\mathcal{Y})}$ in shape $\mathcal{Y}$. We perform a nearest neighbor search of the given vertex features $\mathbf{W}^{(\mathcal{X})}$ and $\mathbf{W}^{(\mathcal{Y})}$ in this reduced search space. We show an illustration in Figure 4.

3.4.2 Runtime versus Optimality Trade-Off

While our algorithm leads to globally optimal solutions, it has exponential worst-case runtime. To reduce the runtime, we set a time limit to solve the individual ILP subproblems.

In [40], convergence speed is used as criterion for model selection – inspired by this, we propose a similar heuristic that is based on our observation that finding the solution that leads to the minimum mean is in many cases the fastest to compute among all values of x in problem (6). Thus, we can reduce the time budget for the ILP subproblems and select

the solution with smallest objective among all subproblems finished within the time limit. If no subproblem terminates within the time budget, we increase the time budget gradually.

4. Experimental Results

In the following, we experimentally evaluate our proposed method. We show that our method outperforms current SOTA partial-to-partial shape matching methods in terms of Intersection over Union (IoU) and Geodesic Error. Further ablation studies can be found in the supplementary.

4.1. Datasets

In our experiments, we use two different datasets:

Intra-Class Dataset First, we use a modified version of the CUTS Partial-to-Partial dataset (CP2P) [3]. The dataset is defined on the partial shapes of the SHREC16 CUTS dataset [12], which include animals and human shapes in different poses. In [3] the authors introduce the CP2P dataset by splitting the original training set into their training and test set. Instead, we use samples from the original SHREC16 CUTS test set to generate our test set CP2P TEST'24, such that we can utilize the whole SHREC16 train set for training. During this work we have observed that the original SHREC16 CUTS test split contains 47 shapes that are almost identical (up to some slight vertex jitter) to shapes in the original training set (see supplementary material for details). Thus, we consider only the remaining 153 shapes of the original test set. We propose to use this subset of shapes as test set in future research. We make this new test split publicly available. Similar to the version provided in [3], the randomly chosen samples have an overlapping region between 10-90%. As in [3], we choose 50 random sample pairs and evaluate them in both directions to obtain 100 matching pairs.

Inter-Class Dataset While the CP2P dataset provides intra-class matching, we present a new dataset called PARTIALSMAL that allows the evaluation of non-isometric inter-class shape matching. The dataset is based on the SMAL dataset [47], which includes 49 animals from 8 species. Similar to [11], we use a train/test split in different species, such that the test set contains three species that are not included in the train set that contains five species. Similar to [15], we re-mesh our dataset as presented in [34], such that methods cannot easily overfit the connectivity of the meshes. We cut the shapes in parts by configuring planes with normal vector (x, y, z), with $(x, y, z) \in \{-1, 0, 1\}^3 \backslash (0, 0, 0)$. This results in 26 different configurations for every shape. See supplementary for more information.

4.2. Experimental Setup

We compute the features required to compute matching costs (for our method and Sm-Comb [37]) with [11] based on the full-to-full matching training set of SMAL [47] and the partial-to-full training set of SHREC16 CUTS [12], respectively. During this work we became aware that for SHREC16 the test and training set used in [11] are not completely independent: TOSCA is used for pre-training of [11], while the SHREC16 dataset was constructed from TOSCA by first remeshing and then introducing missing parts. To avoid data leakage, we instead pretrain [11] using a combination of full shapes from the DT4D [26, 32], SMAL [47], FAUST [8] and SCAPE [1] datasets. During test time, no full shapes are needed. Also, notice that full shapes of the same species of the SMAL test set have not been seen during training as the train and test sets contain different animal species. We use Gurobi [21] intefaced through YALMIP [30] to solve the linear integer programs. We set the maximum time limit for every shape pair to 10 hours. Unless stated otherwise, the time limit for one ILP is set to 15 minutes. The initial interval size for the pruning algorithm is 50. If no solution is found, we first reduce the initial interval size to 25. If still no matching is found, we further downsample the meshes. For more information on how to prepare the shapes, see the supplementary material.

4.3. Comparison Methods

We compare our work with the two existing partial-to-partial shape matching approaches: Sm-comb [37] and DPFM [3]. For Sm-comb [37], we close every hole by adding one vertex in the middle of a hole and then connect this vertex with the vertices on the respective boundary, such that we obtain two full water-tight shapes, as done in the original paper. The mean of the feature vectors of all vertices along this boundary defines the feature of the added vertex. To allow for a fair comparison, we use the same SOTA features to compute matching costs for our method and Sm-comb. The supervised method DPFM [3] is readily applicable to CP2P TEST. For PARTIALSMAL, we provide a train set that is similarly generated as the test set. We cut the full shapes of the SMAL [47] and provide a total of 304 random combinations of these for training DPFM (same amount as in the CP2P train set).

4.4. Overlapping Region Prediction

Intersection over Union (IoU) Let $P \in \{0, 1\}^{(|V| \times 1)}$ and $G \in \{0, 1\}^{(|V| \times 1)}$ be vectors (predicted and ground truth, respectively), that indicate whether the respective vertex is part of the matching. With that, we quantitatively evaluate the overlapping region based on the intersection over union IoU $= \frac{|P \cap GT|}{|P \cup GT|}$.
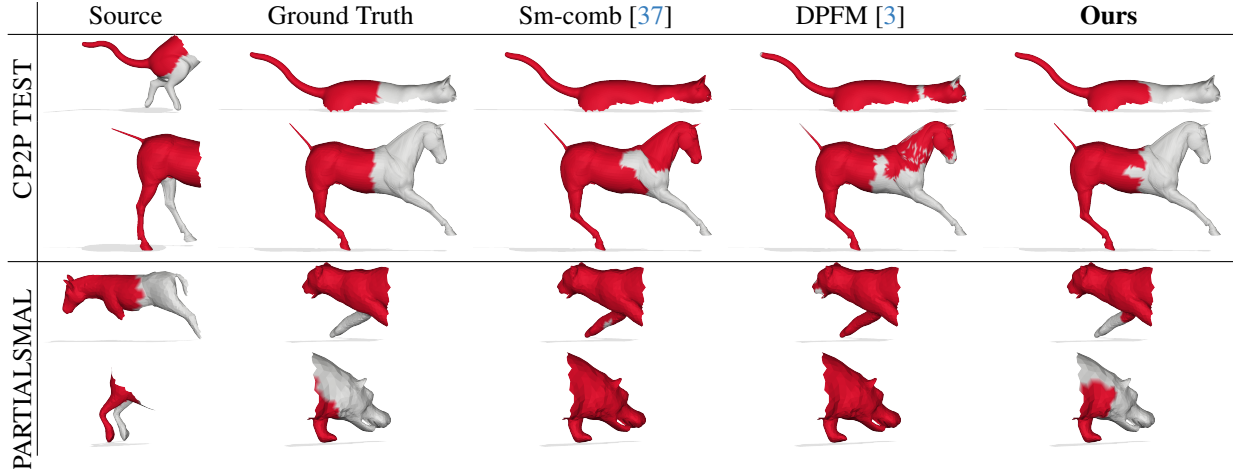
Figure 5. Comparisons of **overlapping region predictions** on CP2P TEST and PARTIALSMAL. DPFM [3] often returns noisy predictions and Sm-comb [37] predicts that the shape is matched completely or at the wrong position in the given example shapes (first two rows CP2P TEST, last two rows PARTIALSMAL). Only our method returns results close to the expected ground truth. We denote that our method performs overlapping prediction and matching simultaneously.

Results We compute the IoU for the predicted overlapping region compared to the ground truth. We observe that our method outperforms the two competing methods in terms of the number of shapes that exceed a specific IoU (see Figure 6). Additionally, we achieve a better mean IoU than the two baselines (see Table 1). In Figure 5, we show qualitative results of the overlapping region prediction.
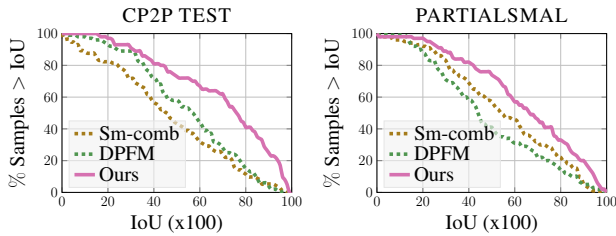

Figure 6. Comparison of **IoU scores** on CP2P TEST dataset (left) and PARTIALSMAL dataset (right). Our method outperforms current SOTA partial-to-partial methods [3, 37].

Dataset	Sm-comb [37]	DPFM [3]	Ours
CP2P TEST	57.86	54.93	**69.29**
PARTIALSMAL	54.76	48.31	**64.34**

Table 1. Comparison of mean **IoU** scores (x100). Our method outperforms Sm-comb [37] and DPFM [3] on both datasets.

4.5. Correspondence Quality

Geodesic Error We assess the geodesic error of the correspondences according to the Princeton Protocol [23]. This assessment involves normalizing the geodesic error between the ground truth and computed correspondences by the shape diameter, which is equivalent to the square root of the area of the target shape. Ours and Sm-comb [37] establish *triangle-to-triangle* correspondences, including cases with degenerate triangles, possibly resulting in non-bijective correspondences between vertices. To compare with DPFM [3] that returns vertex-wise point-to-point maps, we transform our *triangle-to-triangle* to vertex correspondences. In the DPFM [3] paper the geodesic error is compared only on the ground truth overlapping region. To better account for the partial-to-partial matching paradigm, we set all vertices on the shape that remain unmatched to an infinite geodesic error, and compute the geodesic error for the remaining vertices, such that we obtain a metric, that includes the overall performance of the matching. Therefore, the curve is expected to be lower than a comparable full-to-full matching.

Results We show that our method outperforms DPFM [3] and Sm-comb [37] in terms of geodesic error on the CP2P TEST dataset and the PARTIALSMAL dataset (see Figure 8). We show qualitative color transfer examples in Figure 7.

4.6. Faster Optimization

As ablation we evaluate our heuristic from Section 3.4.2 on the CP2P Test dataset. In Table 2 we show that we can get faster optimization while achieving similar results w.r.t. mean IoU and $\text{AUC}_{\text{GeoErr}}$. Yet,this comes at the cost of running more frequently into time limits for the linear subproblems and thus we can ensure global optimality (on the coarse level) in fewer cases. We show the optimization time for the individual values of x in (6) in the supplementary material.
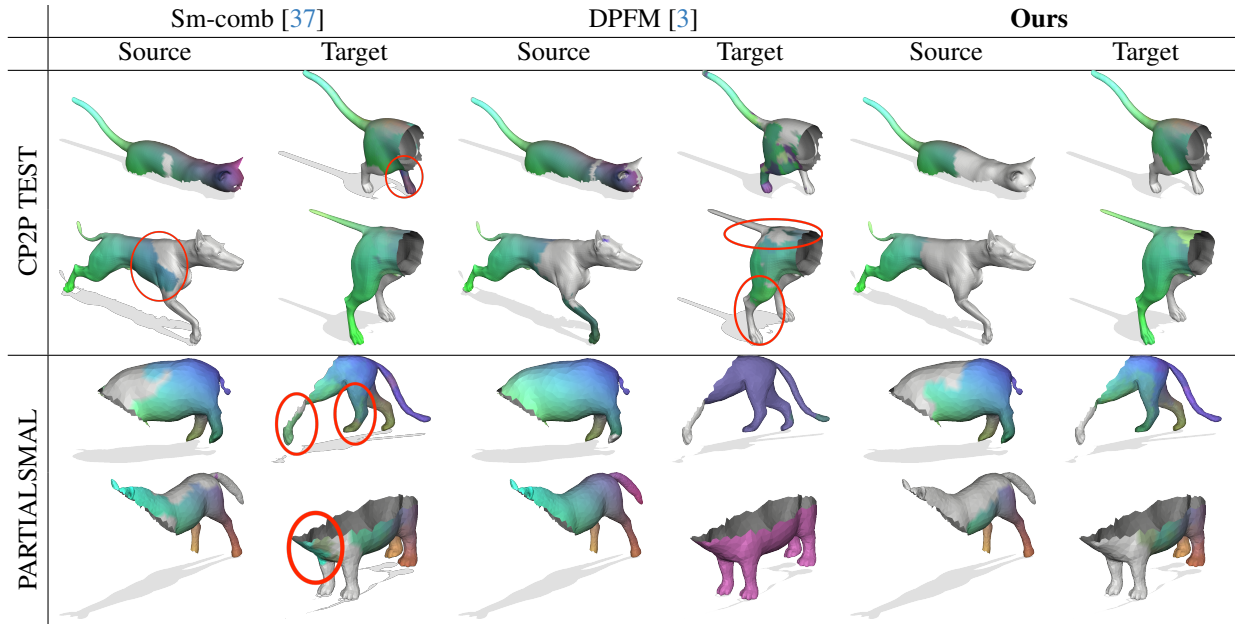
Figure 7. **Qualitative comparison** via **color transfer**. The results are shown on the predicted overlapping region of the methods. The geometric consistent matchings produced by our method visually improve matchings drastically compared to DPFM [3] and Sm-comb [37].
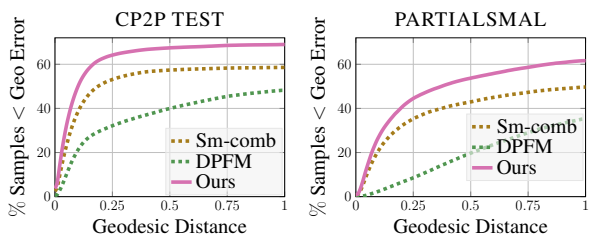

Figure 8. Comparison of **geodesic errors** on the CP2P TEST dataset (left) and on the PARTIALSMAL dataset (right). Our method outperforms current SOTA methods [3, 37].

Limit	Time (min)	mIoU	AUC$_{\text{GeoErr}}$	GlobOpt
15min	299.55	**69.29**	**61.25**	**22**%
5min	140.13	69.15	60.38	10%
1min	57.31	69.47	60.96	8%
10s	**50.68**	68.89	60.95	4%

Table 2. We **reduce the time limit** for subproblems of the minimum mean algorithm and get similar results regarding mean IoU and AUC$_{\text{GeoErr}}$, but ensure global optimality in fewer cases.

5. Discussion and Limitations

While our method outperforms current SOTA methods, there are some limitations. Even though our method uses a pruned search algorithm to reduce runtime, runtimes for solving individual ILPs are still high. Another drawback is that our neighboring constraint is only defined on the inner triangles of the matching, so that there are no rules for

matching a boundary edge on a triangle. Thus, triangles can be matched to boundaries, even though their neighborhood is not matched. Additionally, due to our coarse-to-fine scheme, we only guarantee geometric consistency at coarse levels, but not at finer levels. Further, our approach is based on explicit shape representations. Transferring analogous concepts to implicit representations [46] is an interesting direction for future work.

6. Conclusion

Our paper introduces an innovative integer non-linear approach for partial-to-partial shape matching by exploiting geometric consistency as a constraint. Our method outperforms existing approaches, showcasing SOTA results in terms of intersection over union and geodesic error. This research represents a significant stride forward in addressing real-world challenges in partial-to-partial shape matching, e.g. from 3D scanning. It differs from previous work by eliminating the need for annotated data or closing shapes. Despite potential challenges, such as high optimization time, our work provides a valuable tool for partial-to-partial shape matching. We hope to encourage further research in the practically relevant area of partial-to-partial shape matching utilizing geometric consistency.

References

[1] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. In *ACM SIGGRAPH 2005 Papers*, pages 408–416. 2005. 6

[2] Souhaib Attaiki and Maks Ovsjanikov. Ncp: Neural correspondence prior for effective unsupervised shape matching. *Advances in Neural Information Processing Systems*, 35:28842–28857, 2022. 2

[3] Souhaib Attaiki, Gautam Pai, and Maks Ovsjanikov. Dpfm: Deep partial functional maps. In *2021 International Conference on 3D Vision (3DV)*, pages 175–185. IEEE, 2021. 1, 2, 3, 6, 7, 8

[4] David Bensaïd, Noam Rotstein, Nelson Goldenstein, and Ron Kimmel. Partial matching of nonrigid shapes by learning piecewise smooth functions. In *Computer Graphics Forum*, page e14913. Wiley Online Library, 2023. 2

[5] Florian Bernard, Christian Theobalt, and Michael Moeller. DS*: Tighter Lifting-Free Convex Relaxations for Quadratic Matching Problems. In *CVPR*, 2018. 2

[6] Florian Bernard, Zeeshan Khan Suri, and Christian Theobalt. MINA: Convex mixed integer programming for non-rigid shape alignment. *CVPR*, 2020. 2

[7] Dimitri P Bertsekas. Network Optimization: Continuous and Discrete Models. Athena Scientific, 1998. 2

[8] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. Faust: Dataset and evaluation for 3d mesh registration. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3794–3801, 2014. 6

[9] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Levy. *Polygon Mesh Processing*. CRC Press, 2010. 2

[10] Dongliang Cao and Florian Bernard. Unsupervised deep multi-shape matching. In *European Conference on Computer Vision*, pages 55–71. Springer, 2022. 2

[11] Dongliang Cao, Paul Roetzer, and Florian Bernard. Unsupervised learning of robust spectral shape matching. *ACM Transactions on Graphics (TOG)*, 2023. 1, 2, 6

[12] Luca Cosmo, Emanuele Rodola, Michael M Bronstein, Andrea Torsello, Daniel Cremers, Y Sahillioglu, et al. Shrec'16: Partial matching of deformable shapes. *Proc. 3DOR*, 2(9):12, 2016. 6

[13] Luca Cosmo, Emanuele Rodola, Jonathan Masci, Andrea Torsello, and Michael M. Bronstein. Matching deformable objects in clutter. In *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016. 2

[14] Nicolas Donati, Abhishek Sharma, and Maks Ovsjanikov. Deep geometric functional maps: Robust feature learning for shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8592–8601, 2020. 2

[15] Nicolas Donati, Etienne Corman, and Maks Ovsjanikov. Deep orientation-aware functional maps: Tackling symmetry issues in shape matching. *CVPR*, 2022. 6

[16] Nicolas Donati, Etienne Corman, and Maks Ovsjanikov. Deep orientation-aware functional maps: Tackling symmetry issues in shape matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 742–751, 2022. 2

[17] Nadav Dym, Haggai Maron, and Yaron Lipman. DS++ - A Flexible, Scalable and Provably Tight Relaxation for Matching Problems. *ACM Transactions on Graphics (TOG)*, 36(6), 2017. 2

[18] Viktoria Ehm, Paul Roetzer, Marvin Eisenberger, Maolin Gao, Florian Bernard, and Daniel Cremers. Geometrically consistent partial shape matching. *arXiv preprint arXiv:2309.05013*, 2023. 1, 2, 3

[19] Marvin Eisenberger, Zorah Lähner, and Daniel Cremers. Smooth Shells: Multi-Scale Shape Registration With Functional Maps. 2020. 1

[20] Maolin Gao, Paul Roetzer, Marvin Eisenberger, Zorah Lähner, Michael Moeller, Daniel Cremers, and Florian Bernard. SIGMA: Quantum scale-invariant global sparse shape matching. In *International Conference on Computer Vision (ICCV)*, 2023. 2

[21] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. 6

[22] Stefan Haller, Lorenz Feineis, Lisa Hutschenreiter, Florian Bernard, Carsten Rother, Dagmar Kainmüller, Paul Swoboda, and Bogdan Savchynskyy. A comparative study of graph matching algorithms in computer vision. In *European Conference on Computer Vision*, pages 636–653. Springer, 2022. 2

[23] Vladimir G Kim, Yaron Lipman, and Thomas Funkhouser. Blended intrinsic maps. *ACM transactions on graphics (TOG)*, 30(4):1–12, 2011. 7

[24] Yam Kushinsky, Haggai Maron, Nadav Dym, and Yaron Lipman. Sinkhorn Algorithm for Lifted Assignment Problems. *SIAM Journal on Imaging Sciences*, 12(2):716–735, 2019. 2

[25] Lei Li, Nicolas Donati, and Maks Ovsjanikov. Learning multi-resolution functional maps with spectral attention for robust shape matching. *Advances in Neural Information Processing Systems*, 35:29336–29349, 2022. 1, 2

[26] Yang Li, Hikari Takehara, Takafumi Taketomi, Bo Zheng, and Matthias Nießner. 4dcomplete: Non-rigid motion estimation beyond the observable surface. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12706–12716, 2021. 6

[27] O. Litany, E. Rodolà, A. M. Bronstein, M. M. Bronstein, and D. Cremers. Non-rigid puzzles. *Computer Graphics Forum*, 35(5):135–143, 2016. 2

[28] Or Litany, Tal Remez, Emanuele Rodola, Alex Bronstein, and Michael Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE international conference on computer vision*, pages 5659–5667, 2017. 2

[29] Or Litany, Emanuele Rodolà, Alexander M Bronstein, and Michael M Bronstein. Fully spectral partial shape matching. In *Computer Graphics Forum*, pages 247–258. Wiley Online Library, 2017. 1, 2

[30] J. Löfberg. Yalmip : A toolbox for modeling and optimization in matlab. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. 6

[31] Eliane Maria Loiola, Nair Maria Maia de Abreu, Paulo Oswaldo Boaventura Netto, Peter Hahn, and Tania Maia Querido. A Survey for the Quadratic Assignment Problem. *European Journal of Operational Research*, 176(2): 657–690, 2007. 2

[32] Robin Magnet, Jing Ren, Olga Sorkine-Hornung, and Maks Ovsjanikov. Smooth non-rigid shape matching via effective dirichlet energy optimization. In *2022 International Conference on 3D Vision (3DV)*, pages 495–504. IEEE, 2022. 6

[33] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (TOG)*, 31(4):30:1–30:11, 2012. 1, 2

[34] Jing Ren, Adrien Poulenard, Peter Wonka, and Maks Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. *ACM Transactions on Graphics (TOG)*, 37(6), 2018. 2, 6

[35] F Rendl, P Pardalos, and H Wolkowicz. The Quadratic Assignment Problem: A Survey and Recent Developments. In *DIMACS workshop*, 1994. 2

[36] Emanuele Rodolà, Luca Cosmo, Michael M Bronstein, Andrea Torsello, and Daniel Cremers. Partial functional correspondence. In *Computer graphics forum*, pages 222–236. Wiley Online Library, 2017. 1, 2

[37] Paul Roetzer, Paul Swoboda, Daniel Cremers, and Florian Bernard. A scalable combinatorial solver for elastic geometrically consistent 3d shape matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 428–438, 2022. 1, 2, 3, 6, 7, 8

[38] Yusuf Sahillioğlu. Recent advances in shape correspondence. *The Visual Computer*, 36(8):1705–1721, 2020. 2

[39] F. R. Schmidt, T. Windheuser, U. Schlickewei, and D. Cremers. Dense elastic 3d shape matching. In *Global Optimization Methods*, pages 1–18. Springer, 2014. 1, 2, 3

[40] Johan Thunberg and Florian Bernard. Non-negative spherical relaxations for universe-free multi-matching and clustering. In *Scandinavian Conference on Image Analysis*, pages 260–277. Springer, 2023. 5

[41] Oliver Van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A survey on shape correspondence. In *Computer graphics forum*, pages 1681–1707. Wiley Online Library, 2011. 2

[42] M. Vestner, R. Litman, E. Rodola, A. Bronstein, and D. Cremers. Product manifold filter: Non-rigid shape correspondence via kernel density estimation in the product space. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[43] Thomas Windheuser, Ulrich Schlickewei, Frank R Schmidt, and Daniel Cremers. Geometrically consistent elastic matching of 3d shapes: A linear programming solution. In *2011 International Conference on Computer Vision*, pages 2134–2141. IEEE, 2011. 1, 2, 3, 4

[44] Thomas Windheuser, Ulrich Schlickwei, Frank R Schimdt, and Daniel Cremers. Large-scale integer linear programming for orientation preserving 3d shape matching. In *Computer Graphics Forum*, pages 1471–1480. Wiley Online Library, 2011. 1, 2, 3

[45] L.A. Wolsey. *Integer Programming*. Wiley, 1998. 2

[46] Zerong Zheng, Tao Yu, Qionghai Dai, and Yebin Liu. Deep implicit templates for 3d shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1429–1439, 2021. 8

[47] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 6