An Integer Linear Programming Approach to Geometrically Consistent Partial-Partial Shape Matching

Viktoria Ehm[1,2] Paul Roetzer[3,4] Florian Bernard[3,4] Daniel Cremers[1,2]
[1] Technical University of Munich [2] MCML [3] University of Bonn [4] Lamarr Institute

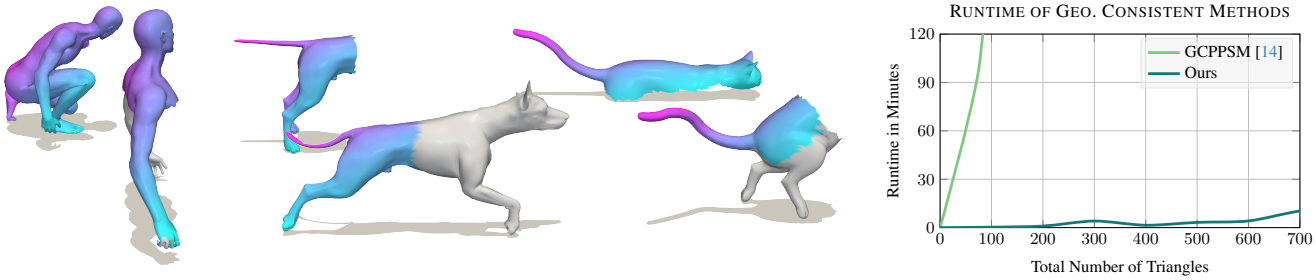

Figure 1. **(Left)** We show matching results for pairs of partial shapes computed with our method using colour transfer. **(Right)** Runtime comparison to the recent (geometrically consistent) partial-partial shape matching method GC-PPSM [14] which approaches the task using integer non-linear programming. In contrast, we propose the first integer *linear* programming approach explicitly tailored to partial-partial shape matching with geometric consistency, making our method more scalable in comparison.

Abstract

The task of establishing correspondences between two 3D shapes is a long-standing challenge in computer vision. While numerous studies address full-full and partial-full 3D shape matching, only a limited number of works have explored the partial-partial setting, very likely due to its unique challenges: we must compute accurate correspondences while at the same time find the unknown overlapping region. Nevertheless, partial-partial 3D shape matching reflects the most realistic setting, as in many real-world cases, such as 3D scanning, shapes are only partially observable. In this work, we introduce the first integer linear programming approach specifically designed to address the distinctive challenges of partial-partial shape matching. Our method leverages geometric consistency as a strong prior, enabling both robust estimation of the overlapping region and computation of neighbourhood-preserving correspondences. We empirically demonstrate that our approach achieves high-quality matching results both in terms of matching error and smoothness. Moreover, we show that our method is more scalable than previous formalisms. Our code is publicly available at https://github.com/vikiehm/partial-geco.

1. Introduction

Solving correspondence problems is a fundamental prerequisite for a wide range of tasks in computer vision, as many downstream applications depend on the ability to relate data points or features across different observations. Prominent examples include 3D reconstruction [25], where accurate correspondences enable the fusion of multiple views into a coherent geometric model, object recognition [22], where matching enables identification of known shapes, tracking [41], where correspondences across time are essential, and image matching [26], which underlies applications such as registration and stitching. Beyond that, correspondence problems arise in shape analysis, such as in 2D shape matching [38], in 2D-3D shape matching [24, 34], and in 3D shape matching [5, 10, 17, 27, 29, 43, 47]. In real-world scenarios, however, the acquisition of 3D shapes often relies on scanning devices that are limited in their ability to capture a complete surface, resulting in shapes that are only partially observed [12]. Such partial observations can occur due to occlusions, limited sensor range, or constraints on scanning time, making complete coverage impractical or impossible. Consequently, the most realistic and practically relevant setting for 3D shape matching is the partial-partial case, in which both shapes are incomplete. Despite its relevance, this scenario remains underexplored, with only a

small number of works explicitly addressing it. The main difficulty lies in its combined challenges: we must not only establish reliable correspondences between the observed regions, but also infer the a priori unknown spatial overlap between the shapes. These two requirements greatly increase the problem's complexity compared to full-full or partial-full settings, where the overlap for at least one shape is known. To address these challenges effectively, powerful regularisation strategies are necessary. As such, geometric consistency, in the form of neighbourhood preservation, serves as a particularly strong prior for regularising the resulting correspondences. Nevertheless, geometric consistency for partial-partial 3D shape matching has so far only been implemented within a non-linear programming framework [14], which can limit scalability and computational efficiency, cf. Fig. 1 right. In this work, we build upon the recently proposed shape matching formalism [32] and introduce the first geometrically consistent integer *linear* programming formalism explicitly tailored for partial-partial 3D shape matching. Our formulation is designed to exploit geometric consistency not only to identify plausible overlap regions but also to compute correspondences that preserve local neighbourhood relations. In turn, this leads to empirically observed high-quality matching results as well as robust overlap predictions. In particular, we summarise our main contributions as follows:

- We introduce the first integer *linear* programming formalism for partial-partial shape matching that explicitly incorporates geometric consistency.
- For the first time, our framework enables the integration of a predicted overlap region into the matching process.
- Through experiments, we demonstrate that our formalism scales significantly better than previous geometrically consistent partial-partial approaches.
- We further show that our method achieves high-quality matching results in terms of geodesic errors, smoothness, and overlap scores.

2. Related Works

In this section, we review works which we consider most relevant to our paper. For a broader overview of shape matching methods, we invite the interested reader to read the survey papers [37, 45, 46].

Partial Shape Matching. Matching of complete shapes, i.e. full-full shape matching has received lots of attention in the past years. In contrast, partial shape matching, i.e. shape matching where at least one shape is incomplete, is less explored (partial-full shape matching) or is even underexplored (partial-partial shape matching) due to yet-to-overcome challenges [16]. In one of the first works of partial-full shape matching, the problem is tackled by using functional maps [27] and by exploiting the resulting slanted diagonal structure of respective functional maps [30]. This idea was further used as a loss function in supervised [3] and unsupervised [3, 9, 10] learning frameworks. Other lines of work focus on preserving distances between points to act as regularisation when points are being matched from source to target shape [6, 7] or incorporate neighbourhood preservation as a hard constraint [15, 32]. Among the few works specifically tailored to the partial-partial problem, predicting the overlapping region in learning-based methods is approached using cross-attention [3] or so-called correspondence reflections [49], which both yield probabilities stating if a vertex is or is not within the overlapping region. In contrast, axiomatic methods rely on non-linear integer programming and neighbourhood preservation constraints to determine the overlapping region [14]. Our formalism explicitly allows to incorporate such overlapping region probabilities into the cost function, while at the same time it regularises the overlapping region by enforcing neighbourhood preservation.

Geometrically Consistent Shape Matching. Geometric consistency, i.e. the preservation of neighbourhood relations of surface elements, can act as a strong regulariser to compute high-quality shape matching results and can resolve ambiguities in ambiguous matching problems such as partial-partial shape matching. Yet, geometric consistency is often neglected due to resulting hard-to-solve formalisms. As such, some approaches incorporate neighbourhood information using formalisms based on the quadratic assignment problem, which, due to its NP-hardness [28], can only be solved approximately [8, 19, 23, 44]. Other lines of work require an initial set of (sparse) correspondences and either deform one shape to the other [1] or match shapes via intermediate domains [4, 39, 40]. In contrast, Windheuser et al. [47], have proposed an initialisation-free integer linear program based upon matching triangles and preserving their neighbourhood. Due to the complexity of the resulting optimisation problem, it can only be solved using coarse-to-fine strategies [48] or approximate solvers [33, 35]. Later, in SpiderMatch [31], authors propose to represent the source shape using a self-intersecting, cyclic curve and match this curve to the target shape while preserving the intersections of the curve. While this formalism is more scalable than previous works, it only enforces geometric consistency at intersection points. Building on the idea of alternative 3D shape representations, the recent work GeCo [32] represents the source shape using multiple cyclic curves and matches these curves to the target shape while preserving the neighbourhood of the individual cycles. This results in better geometric consistency compared to SpiderMatch [31] while remaining similarly scalable. A follow-up work [2], building on GeCo [32], showed that the formalism can be adapted to match shapes with up to 10k triangles. We also build our work on GeCo [32] and propose a novel formalism which is explicitly designed for partial-partial shape matching.

Symbol	Description
$\boldsymbol{X} = (\boldsymbol{V_X}, \boldsymbol{E_X}, \boldsymbol{F_X})$	3D shape
$\boldsymbol{V_X} = \boldsymbol{V_X}^{\text{bn}} \cup \boldsymbol{V_X}^{\text{in}}$	Vertices of shape $\boldsymbol{X}$ with boundary $\boldsymbol{V_X}^{\text{bn}}$ and inner $\boldsymbol{V_X}^{\text{in}}$ vertices ($\boldsymbol{V_X}^{\text{bn}} \cap \boldsymbol{V_X}^{\text{in}} = \emptyset$)
$\boldsymbol{E_X}$	Edges of shape $\boldsymbol{X}$
$\boldsymbol{F_X}$	Triangles of shape $\boldsymbol{X}$
$\boldsymbol{Y}$	3D shape $\boldsymbol{Y}$ (defined analogously to $\boldsymbol{X}$)
$\boldsymbol{C}_i$	i-th surface cycle (representing the i-th triangle $f_i \in \boldsymbol{F_X}$ if shape $\boldsymbol{X}$)
$\boldsymbol{P}_i = (\boldsymbol{V}_{\boldsymbol{P}_i}, \boldsymbol{E}_{\boldsymbol{P}_i})$	Product graph between i-th surface cycle $\boldsymbol{C}_i$ and 3D shape $\boldsymbol{Y}$
$\boldsymbol{P} = (\boldsymbol{V_P}, \boldsymbol{E_P})$	Collection of all $i = 1 \dots \lvert\boldsymbol{F_X}\rvert$ product graphs $\boldsymbol{P}_i$
c_k	Matching cost for product edge $e_k \in \boldsymbol{E_P}$
$o_{\boldsymbol{X}}$	Overlap probability of edges of shape $\boldsymbol{X}$
$o_{\boldsymbol{Y}}$	Overlap probability of vertices of shape $\boldsymbol{Y}$

Table 1. Summary of the **notation** used in this paper.

Geometrically Consistent Partial-Partial Shape Matching. Partial-partial shape matching remains under-explored and usually geometric consistency is neglected due to hard-to-solve formalisms, even though it can act as a strong regulariser to matching problems. Consequently, there exists only one work which tackles partial-partial shape matching with geometric consistency: [14] proposes a non-linear integer programming approach which builds on the formalism introduced in [47] for geometrically consistent full-full shape matching. As such, [14] leverages geometric consistency to compute the overlapping region. Yet, due to non-linearity, the resulting formalism does not scale well, see Fig. 1 right.

In our work, we propose an integer *linear* programming formalism which also uses geometric consistency as a prior to compute the overlapping region. In addition, since our formalism minimises a linear objective compared to a non-linear one [14], our approach is more scalable.

3. Background

In the following, we discuss relevant background for our method: we define shapes and an alternative shape representation (Sec. 3.1), introduce product graphs (Sec. 3.2) and define geometric consistency for partial-partial shape matching (Sec. 3.3). Our notation is summarised in Tab. 1.

3.1. 3D Shapes and Shape Representations

For our formalism, we consider partial 3D shapes (i.e. oriented 2D manifold in 3D space with boundary) which we define as follows:

Definition 1 (3D Shape)**.** *A 3D shape* $\boldsymbol{X} = (\boldsymbol{V_X}, \boldsymbol{E_X}, \boldsymbol{F_X})$ *is a triplet consisting of vertices* $\boldsymbol{V_X}$*, directed edges* $\boldsymbol{E_X} \subset \boldsymbol{V_X} \times \boldsymbol{V_X}$ *and oriented triangles* $\boldsymbol{F_X} \subset \boldsymbol{V_X} \times \boldsymbol{V_X} \times \boldsymbol{V_X}$ *such that* $\boldsymbol{X}$ *forms an oriented 2D manifold in 3D space with potential boundaries. Consequently, the set of vertices* $\boldsymbol{V_X} = \boldsymbol{V_X}^{bn} \cup \boldsymbol{V_X}^{in}$ *can be partitioned into boundary vertices* $\boldsymbol{V_X}^{bn}$ *and non-boundary vertices* $\boldsymbol{V_X}^{in}$ *such that* $\boldsymbol{V_X}^{bn} \cap \boldsymbol{V_X}^{in} = \emptyset$.

Directed edges mean that edges have an orientation, i.e. whenever an edge e is in the set of edges $e \in \boldsymbol{E}$ does not imply that its opposite edge $-e$ is in the set of edges, i.e. $e \notin \boldsymbol{E}$.

Following [32], we consider an alternative 3D shape representation using what they call *surface cycles*.

Definition 2 (Surface Cycle Shape Representation)**.** *The triangles* $\boldsymbol{F_X}$ *of a 3D shape* $\boldsymbol{X}$ *can be represented with a collection of* $\lvert\boldsymbol{F_X}\rvert$*-many surface cycles* $\boldsymbol{C}_1, \dots, \boldsymbol{C}_{\lvert\boldsymbol{F_X}\rvert}$ *such that the* i*-th triangle* $f_i \in \boldsymbol{F_X}$ *is represented with the* i*-th surface cycle* $\boldsymbol{C}_i$*. The* i*-th surface cycle* $\boldsymbol{C}_i = (\boldsymbol{V}_{\boldsymbol{C}_i}, \boldsymbol{E}_{\boldsymbol{C}_i})$ *is a cyclic chain graph consisting of the three vertices* $v_1, v_2, v_3 \in \boldsymbol{V}_{\boldsymbol{C}_i} \subset \boldsymbol{V_X}$ *and the three oriented edges* $e_1, e_2, e_3 \in \boldsymbol{E}_{\boldsymbol{C}_i} \subset \boldsymbol{E_X}$ *of the* i*-th triangle.*

The intuition is that every oriented triangle of shape $\boldsymbol{X}$ forms a directed cyclic chain graph with three edges and three vertices such that the surface of $\boldsymbol{X}$ is tiled into $\lvert\boldsymbol{F_X}\rvert$-many surface cycles which are glued together at opposite edges, see [32] and Fig. 2 for an illustration.

3.2. Shape Matching with Product Graphs

Shortest paths in product graphs have been introduced for various shape matching tasks, including shape-image matching [41], 2D-3D shape matching [24, 34, 36] and 3D shape matching [31, 32, 36]. The key idea is to compute the product graph $\boldsymbol{P}_i$ (or multiple product graphs [32]) between a cyclic chain graph $\boldsymbol{C}_i$ and another graph, i.e. a 3D shape $\boldsymbol{Y}$. We follow the definition presented in [24] in which a product graph $\boldsymbol{P}$ reads:

Definition 3 (Product graph)**.** *The product graph* $\boldsymbol{P} = (\boldsymbol{V_P}, \boldsymbol{E_P})$ *between a cyclic chain graph* $\boldsymbol{C}$ *and a 3D shape* $\boldsymbol{Y}$ *is a directed graph defined as*

$$\begin{aligned} \boldsymbol{V}_{\boldsymbol{P}_i} &= \boldsymbol{V}_{\boldsymbol{C}_i} \times \boldsymbol{V_Y}, \\ \boldsymbol{E}_{\boldsymbol{P}_i} &= \{(v, \bar{v}) \in \boldsymbol{V}_{\boldsymbol{P}_i} \times \boldsymbol{V}_{\boldsymbol{P}_i} \mid v = \left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right),\ \bar{v} = \left(\begin{smallmatrix} \bar{x} \\ \bar{y} \end{smallmatrix}\right) \\ &\qquad (x, \bar{x}) \in \boldsymbol{E}_{\boldsymbol{C}_i}, (y, \bar{y}) \in \boldsymbol{E}_{\boldsymbol{Y}}^{+}\}. \end{aligned} \tag{1}$$

Here $\boldsymbol{E}_{\boldsymbol{Y}}^{+} := \boldsymbol{E_Y} \cup \{(y, y) \mid y \in \boldsymbol{V_Y}\}$ *are the edges of shape* $\boldsymbol{Y}$ *extended with self-edges.*

In a nutshell, the product graph $\boldsymbol{P}$ contains every possible combination between edges of the cyclic chain graph $\boldsymbol{C}$ and edges/vertices of shape $\boldsymbol{Y}$ and thus encodes every potential matching between these elements. Furthermore, the product graph $\boldsymbol{P}$ is defined via the connectivity of vertices of the cyclic chain graph $\boldsymbol{C}$ and the connectivity of vertices of the shape $\boldsymbol{Y}$ such that a cyclic path within $\boldsymbol{P}$ yields a cyclic path on $\boldsymbol{Y}$, see [24, 32] for illustrations and

further discussion. We can define suitable matching costs for every edge in the product graphs (e.g. vertex-wise feature differences as done in [24, 31, 32, 34, 36]). According to these costs we find a shortest, i.e. minimum cost, cyclic path through the product graph $\boldsymbol{P}$ (by variants of Dijkstra's algorithm). This shortest cyclic path in $\boldsymbol{P}$ in turn yields a globally optimal matching between a cyclic chain graph $\boldsymbol{C}$ and a 3D shape $\boldsymbol{Y}$ (since every edge in $\boldsymbol{P}$ encodes a potential matching). Yet, for our purposes of geometrically consistent 3D shape matching we want to define multiple of these product graphs and we want to employ additional constraints. Hence, we cannot use vanilla shortest path algorithms but rather have to consider the integer linear programming formalism of the above explained shortest path problem.

3.3. Partial-Partial Geometric Consistency

We aim to find geometrically-consistent matchings between two partial shapes. This means that for every point in the resulting overlapping region (i.e. the region on both shapes which is matched) we want to preserve its neighbourhood. Formally, we define geometric consistency for partial-partial shape matching as follows

Definition 4 (Partial-Partial Geometric Consistency)**.** *We call a mapping $\sigma : \boldsymbol{V_X} \rightarrow \boldsymbol{V_Y}$ between two partial shapes $\boldsymbol{X}$ and $\boldsymbol{Y}$ geometrically consistent if for every interior pair of matched points $x, \bar{x} \in \boldsymbol{V}_{\boldsymbol{X}}^{in}$, that is connected by an edge, i.e. $(x, \bar{x}) \in \boldsymbol{E_X}$ and $\sigma(x) \neq \emptyset$, $\sigma(\bar{x}) \neq \emptyset$ it holds that either $(\phi(x), \phi(\bar{x})) \in \boldsymbol{E_Y}$ or $\phi(x) = \phi(\bar{x})$ whenever $\phi(x), \phi(\bar{x})$ map to interior vertices, i.e. $\phi(x), \phi(\bar{x}) \in \boldsymbol{V}_{\boldsymbol{Y}}^{in}$.*

In a nutshell, every pair of points on shape $\boldsymbol{X}$, that is connected by an edge, should be matched to another pair of points on shape $\boldsymbol{Y}$ which are either connected by an edge or the same vertex.

4. Partial-Partial Shape Matching

Our goal is to find a geometrically consistent matching between a partial source shape $\boldsymbol{X}$ and a partial target shape $\boldsymbol{Y}$ (i.e. two 3D shapes with boundary and unknown overlap). To this end, we build on [32] and consider an independent matching problem for each triangle, which we couple at opposite edges to enforce geometric consistency (Sec. 4.1). Furthermore, we design injectivity and surjectivity constraints such that we can account for the unknown overlap between both shapes, i.e. we allow vertices of both shapes not to be matched (Sec. 4.2). This leads to our novel integer linear program (ILP) for partial-partial 3D shape matching, which is able to incorporate fuzzy overlap predictions (Sec. 4.3). Finally, we discuss a coarse-to-fine strategy to scale to higher resolutions (Sec. 4.4). An overview of our approach can be found in Fig. 2.

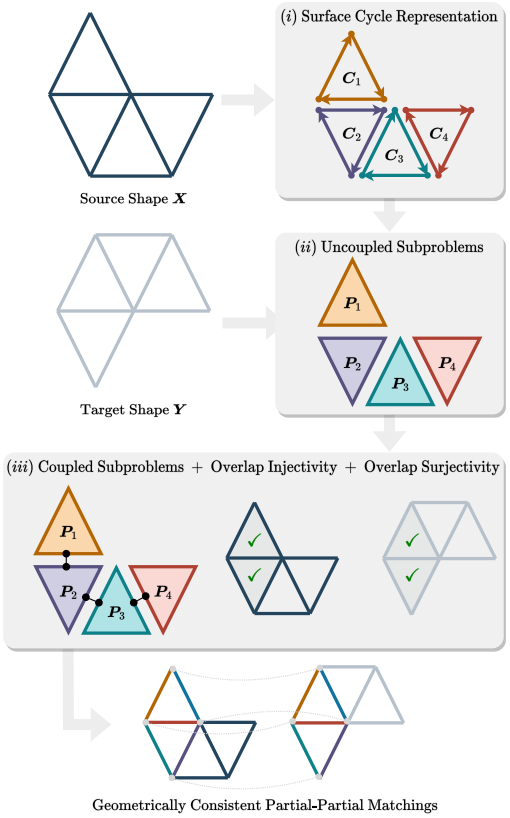

Figure 2. **Overview** over our partial-partial 3D shape matching approach. We match a partial source 3D shape $\boldsymbol{X}$ to a non-rigidly deformed partial target 3D shape $\boldsymbol{Y}$. (i) We represent the source shape using surface cycles (cf. Definition 2). (ii) For each surface cycle $\boldsymbol{C}_i$, we formulate an independent subproblem as the product graphs $\boldsymbol{P}_i$ between a $\boldsymbol{C}_i$ and the whole 3D shape $\boldsymbol{Y}$ (cf. Definition 3). (iii) By employing coupling constraints (cf. Sec. 4.1) as well as injectivity and surjectivity constraints of the overlap (cf. Sec. 4.2) we obtain a formalism for geometrically consistent partial-partial 3D shape matching (cf. Sec. 4.3).

4.1. Coupled Surface Cycle Matching Problems

We follow [32] and represent each of the $i = 1 \ldots |\boldsymbol{F_X}|$ triangles $f_i \in \boldsymbol{F_X}$ of source shape $\boldsymbol{X}$ with a surface cycles $\boldsymbol{C}_i$, cf. Definition 2. For each surface cycle $\boldsymbol{C}_i$, we compute a product graph $\boldsymbol{P}_i$ between $\boldsymbol{C}_i$ and the target 3D shape $\boldsymbol{Y}$ which gives us an indepent subproblem allowing us to match the i-th triangle of shape $\boldsymbol{X}$ to shape $\boldsymbol{Y}$. We collect all product graphs $\boldsymbol{P}_i$ in what we call the *collection of product graphs* $\boldsymbol{P}$.

Definition 5 (Collection of Product Graphs)**.** *The collection of product graphs $\boldsymbol{P} = (\boldsymbol{V_P}, \boldsymbol{E_P})$ contains all $i = 1 \ldots |\boldsymbol{F_X}|$ sets of product vertices $\boldsymbol{V}_{\boldsymbol{P}_i}$ and sets of product edges $\boldsymbol{E}_{\boldsymbol{P}_i}$ such that the vertices $\boldsymbol{V_P}$ and edges $\boldsymbol{E_P}$ of $\boldsymbol{P}$*

read

$$\boldsymbol{V_P} = \boldsymbol{V_{P_1}} \cup \dots \cup \boldsymbol{V_{P_{|\boldsymbol{F_X}|}}}, \quad \boldsymbol{E_P} = \boldsymbol{E_{P_1}} \cup \dots \cup \boldsymbol{E_{P_{|\boldsymbol{F_X}|}}}. \tag{2}$$

By definition of the surface cycles and the product graphs, the product edges $\boldsymbol{E_P}$ of the product graph collection $\boldsymbol{P}$ encode all potential matchings between edges of shape $\boldsymbol{X}$ and edges/vertices of shape $\boldsymbol{Y}$.

As mentioned earlier, our goal is to derive an integer linear programming formalism for the $|\boldsymbol{F_X}|$-many shortest cyclic path problems collected in $\boldsymbol{P}$ such that we can incorporate additional constraints. To this end, we encode the k-th product edge $e_k \in \boldsymbol{E_P}$ with a binary variable $x_k \in \{0,1\}$ where $x_k = 1$ means that product edge e_k is part of the computed matching. Furthermore, and to encode that matched edges must form a connected path in $\boldsymbol{P}$, we consider continuity constraints which read

$$\forall\, v \in \boldsymbol{V_P} : \sum_{k:e_k=(\bullet,v)\in\boldsymbol{E_P}} x_k = \sum_{j:e_j=(v,\bullet)\in\boldsymbol{E_P}} x_j. \tag{CONT}$$

Intuitively, whenever a product vertex $v \in \boldsymbol{V_P}$ has an 'active' incoming edge, it also needs to have an 'active' outgoing edge. Consequently, continuity and with that geometric consistency along the resulting path is preserved. Yet, geometric consistency between the independent subproblems is not yet ensured, i.e. the matchings of neighbouring triangles of shape $\boldsymbol{X}$ are not necessarily neighbouring on shape $\boldsymbol{Y}$.

To account for that, we adapt the idea presented in [32] and couple the independent subproblems at opposite product edges whenever a product edge amounts to non-boundary edges on shape $\boldsymbol{X}$ and shape $\boldsymbol{Y}$. To this end, we consider the following coupling constraints

$$\forall\, kj;\ e_k e_j \in \boldsymbol{E_P^{\text{in}}};\ e_k = -e_j : \quad x_k = x_j. \tag{COUPL}$$

Here, $\boldsymbol{E_P^{\text{in}}} \subset \boldsymbol{E_P}$ are all product edges of $\boldsymbol{P}$ which solely consist of non-boundary vertices on shape $\boldsymbol{X}$ and a non-boundary vertices on shape $\boldsymbol{Y}$, i.e. $\boldsymbol{E_P^{\text{in}}}$ reads

$$\boldsymbol{E_P^{\text{in}}} = \{e = \left(\binom{x}{y}, \binom{\bar{x}}{\bar{y}}\right) \in \boldsymbol{E_P} \mid x, \bar{x} \in \boldsymbol{V_X^{\text{in}}}, y, \bar{y} \in \boldsymbol{V_Y^{\text{in}}}\ \}. \tag{3}$$

These coupling constraints ensure that the neighbourhood of every surface cycle (which does not contain a boundary vertex) is preserved. We note that it is important to enforce the coupling only on non-boundary product edges. This allows us to handle the fact that the overlap is unknown, i.e. it would not be possible to match only parts of the shape if the coupling were also defined for boundary edges.

While the above constraints (CONT) and (COUPL) effectively enforce geometric consistency, we also want to ensure that resulting matchings are injective (i.e. every vertex of shape $\boldsymbol{X}$ within the overlapping region should be matched) and surjective (i.e. every vertex of shape $\boldsymbol{Y}$ within the overlapping region should be matched), which we discuss next.

4.2. Injectivity and Surjectivity of Overlap

Our goal is to have a matching which is injective and surjective within the overlapping region between both shapes, i.e. we want to match every vertex of $\boldsymbol{X}$ and every vertex of $\boldsymbol{Y}$ whenever these vertices are in the overlapping region. Yet, we do not know the overlapping region a priori. To account for that, we introduce variables $s^{\text{inj}} \in \{0,1\}^{|\boldsymbol{E_X}|}$ and $s^{\text{sur}} \in \{0,1\}^{|\boldsymbol{V_Y}|}$. Using s^{inj}, our injectivity constraints read

$$\forall\, j : e_j = (x, \bar{x}) \in \boldsymbol{E_X} : \sum_{k:e_k=\left(\binom{x}{\bullet},\binom{\bar{x}}{\bullet}\right)\in\boldsymbol{E_P}} x_k - s_j^{\text{inj}} = 1. \tag{INJY}$$

Hence, we enforce injectivity by requiring that every edge $e_j \in \boldsymbol{E_X}$ of shape $\boldsymbol{X}$ is matched exactly once whenever e_j is in the overlapping region (i.e. whenever $s_j^{\text{inj}} = 0$). Furthermore, our surjectivity constraints read

$$\forall\, j :\ v_j \in \boldsymbol{V_Y},\ v_j = y \ \vee\ v_j = \bar{y} : \sum_{k:e_k=\left(\binom{\bullet}{y},\binom{\bullet}{\bar{y}}\right)\in\boldsymbol{E_P}} x_k - s_j^{\text{sur}} \geq 1. \tag{SURJY}$$

Similarly to injectivity, we enforce surjectivity by requiring that every vertex $v_j \in \boldsymbol{V_Y}$ is matched at least once whenever v_j is in the overlapping region (i.e. whenever $s_j^{\text{sur}} = 0$).

With all of the required constraints now at hand, we can define our integer linear program (ILP) for geometrically consistent partial-partial shape matching.

4.3. ILP for Partial-Partial Shape Matching

In the following, we state our ILP for geometrically consistent partial-partial shape matching. To this end, we collect all binary variables x_k (which encode if product edge $e_k \in \boldsymbol{E_P}$ is part of the final matching) in a binary vector $x \in \{0,1\}^{|\boldsymbol{E_P}|}$. Furthermore, we define a matching cost c_k for every product edge $e_k \in \boldsymbol{E_P}$ and also collect all matching costs in a vector $c \in \mathbb{R}^+$. For the matching cost, we compute feature-differences of per-vertex features, see Sec. 5.1. Additionally, we compute probabilities if an edge, respectively vertex, of shape $\boldsymbol{X}$, respectively shape $\boldsymbol{Y}$, is within the overlapping region and collect these in vectors $o_{\boldsymbol{X}} \in [0,1]^{|\boldsymbol{E_X}|}$, respectively $o_{\boldsymbol{Y}} \in [0,1]^{|\boldsymbol{V_Y}|}$, see Sec. 5.1.

With $c, o_{\boldsymbol{X}}$, and $o_{\boldsymbol{Y}}$, we can state our ILP for geometrically consistent partial-partial shape matching.

$$\min_{x, s^{\text{inj}}, s^{\text{sur}}} c^T x + \lambda o_{\boldsymbol{X}}^T s^{\text{inj}} + \lambda o_{\boldsymbol{Y}}^T s^{\text{sur}}$$

$$\text{s.t. (CONT), (COUPL), (INJY), (SURJY),}$$

$$x \in \{0,1\}^{|\boldsymbol{E_P}|}, s^{\text{inj}} \in \{0,1\}^{|\boldsymbol{E_X}|}, s^{\text{sur}} \in \{0,1\}^{|\boldsymbol{V_Y}|} \tag{PP-ILP}$$

Here, λ is a weighting term between matching costs and overlap probabilities, see Sec. 5.1.

Even though (PP-ILP) is an integer linear formalism for partial-partial shape matching, it involves many binary variables and thus is hard to solve. This might limit its scalability (while still scaling significantly better than previous methods, cf. Fig. 1). Thus, to match higher resolution shapes, we solve (PP-ILP) on lower shape resolutions and propagate solutions to higher shape resolutions, which we discuss next.

4.4. Scaling to Higher Resolutions

Due to many binary variables and complex constraints, (PP-ILP) might only be trackable to solve for lower resolution shapes. To scale to higher resolution shapes, we use a solution found on lower resolution shapes to prune (PP-ILP) for higher resolution shapes [15, 48]. In particular, by solving (PP-ILP), we compute a mapping $\sigma^{\text{lr}} : \boldsymbol{V_X^{\text{lr}}} \to \boldsymbol{V_Y^{\text{lr}}}$ on lower resolution variants of shapes $\boldsymbol{X}$ and $\boldsymbol{Y}$, i.e. $\boldsymbol{V_X^{\text{lr}}} \subset \boldsymbol{V_X}$ and $\boldsymbol{V_Y^{\text{lr}}} \subset \boldsymbol{V_Y}$. Additionally, we maintain a mapping from higher resolution vertices to lower resolution vertices, i.e. $\gamma_{\boldsymbol{X}} : \boldsymbol{V_X} \to \boldsymbol{V_X^{\text{lr}}}$ and $\gamma_{\boldsymbol{Y}} : \boldsymbol{V_Y} \to \boldsymbol{V_Y^{\text{lr}}}$. Furthermore, we consider N-ring$(\cdot)$ to be the set of vertices which lie within the N-ring neighbourhood of a vertex on a shape (i.e. all vertices that are at most N edges away from a given vertex). With that, we can compute a set $\boldsymbol{A}$ consisting of product vertices which comprise the allowed matching pairs on higher resolution shapes.

$$\boldsymbol{A} := \left\{ \left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right) \mid x \in \boldsymbol{V_X},\ y \in \boldsymbol{V_Y},\ \gamma_{\boldsymbol{Y}}(N\text{-ring}(y)) \cap \sigma^{\text{lr}}(\gamma_{\boldsymbol{X}}(N\text{-ring}(x)) \neq \emptyset \right\} \tag{4}$$

In a nutshell, $\boldsymbol{A}$ consists of all pairs of vertices (including their N-ring neighbourhood) of higher resolution shapes which map to a pair of matched vertices on lower resolution shapes, see Fig. 3 for an illustration.

For higher resolution shapes, we employ the following additional constraints in (PP-ILP) to prune matchings which do not lie within the N-ring neighbourhood of the lower resolution matching σ^{lr}.

$$\forall\, k : e_k = (v, \bar{v}) \in \boldsymbol{E_P},\ v \notin \boldsymbol{A} \wedge \bar{v} \notin \boldsymbol{A} : x_k = 0 \tag{PRUNE}$$

We note that the above described pruning and subsequent solving of (PP-ILP) can be repeated multiple times, such that we can go from lower resolution to higher resolution shapes via intermediate resolutions.

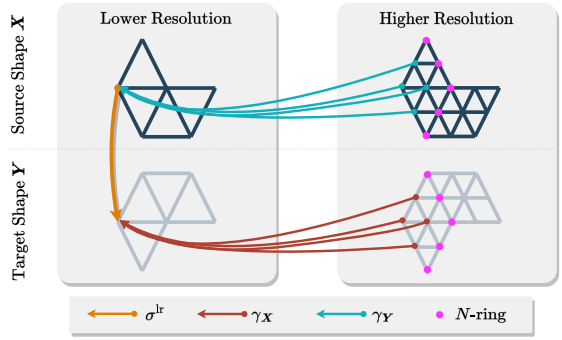

Figure 3. We **prune** instances of (PP-ILP) for pairs of higher resolution shapes by using computed matching of (the same pair of) lower resolution shapes. In particular, on higher resolution shapes, we only allow for pairs of vertices which map to a matching on lower resolutions shapes within their N-ring neighbourhood (we show $N = 1$ in the figure), i.e. in this example, all coloured vertices on higher resolution shape $\boldsymbol{X}$ can be matched to all coloured vertices higher resolution shape $\boldsymbol{Y}$.

5. Experiments

In this section, we empirically evaluate our method's performance. To this end, we first introduce the datasets and comparison methods as well as the experimental setup that we use for our method and competitors. After that, we explain the metrics that we use. Finally, we report quantitative and qualitative matching results.

5.1. Dataset & Competing Methods & Setup

Datasets. We conduct a quantitative comparison on two datasets. First, we use CP2P24 [14], which builds on SHREC16 CUTS [11] and defines a new test set for CP2P [3] such that the test set does not contain shapes which exist in the SHREC16 CUTS train set (we report results only on test set shapes). The data set consists of 76 isometric deformed animal and human shapes with varying partiality ranging from 10%-90% overlap. Second, we use PSMAL [14], which consists of 49 remeshed animal shapes from 8 different species from SMAL [50] and where partiality is introduced by cutting shapes in half using planes in 3D space, see [14] for more details. Similar to [10], we report results only on the test set of SMAL [50].

Competing Methods. We compare our method against three other approaches. DPFM [3] is a supervised learning-based approach that uses a cross-attention feature refinement network for partial-to-partial shape matching, predicting both the overlapping region and features for every vertex. EchoMatch [49] is a learning-based method which uses features computed from image foundation models [13] as input, feeds them into a DiffusionNet [42], and predicts overlapping regions based on so-called correspondences reflections. GC-PPSM [14] builds its formalism on the inte-

ger linear program for geometrically consistent 3D shape matching introduced in [47] but is specifically tailored for geometrically consistent partial-partial 3D shape matching. As such, GC-PPSM uses a non-linear integer programming formalism.

Setup. For all axiomatic methods, we use features of EchoMatch [49] to compute matching costs, i.e. we compute the cosine distance between the features of respective vertices of the product edges. Additionally, we use the overlap predictions $o_{\boldsymbol{X}}$ and $o_{\boldsymbol{Y}}$ of EchoMatch in our method. For our approach, we solve lower resolution pairs of shapes on a combined number of 600 triangles and use these solutions to prune higher resolution pairs with a combined resolution of 1000 triangles, for which we use the $N = 2$ ring neighbourhood. To this end, we downsample shapes using algorithms provided in [20] such that shapes have similarly sized triangles. We solve (PP-ILP) using off-the-shelf solver Gurobi [18] where we set a time limit of 60 for resolutions 600 and then upsample to 1000 triangles via 800 triangles, where every upsample step has a time limit of 30 minutes. We set $\lambda = 0.5$ for PSMAL and $\lambda = 0.3$ for CP2P24. Furthermore, for GC-PPSM [14] we use the experimental setup as depicted in the paper. To make geodesic errors comparable, we upsample solutions to full resolutions using the upsampling technique proposed in [14] with a neighbourhood size of 1 and choose the direction of the matching with the better objective value of the (PP-ILP).

5.2. Metrics

Intersection over Union. For each vertex on one shape, e.g. for $\boldsymbol{X}$, we can define a matching vector $m \in \{0,1\}^{(|\boldsymbol{V_X}|)}$ (by evaluating the computed matching of our method) as well as obtain a ground truth vector $g \in \{0,1\}^{(|\boldsymbol{V_X}|)}$ (by evaluating the ground-truth correspondences). Both vectors m and g indicate whether a vertex $x_i \in \boldsymbol{V_X}$ lies within the overlapping region, i.e. $g_i = 1$ respectively $m_i = 1$, or does not lie within the overlapping region, i.e. $g_i = 0$ respectively $m_i = 0$. Using these two vectors m, g, we follow [14] and use the intersection over Union (IoU), i.e. IoU $= \frac{|m \cap g|}{|m \cup g|}$, to quantify the accuracy of overlap region predictions. We report the mean IoU (mIoU) over all shapes.

Geodesic Error. We measure correspondence quality on the intersection of predicted and overlapping region using the geodesic error, for which we follow the Princeton Protocol [21] and normalise distances by the shape diameter of the full shape.

Dirichlet Energy To evaluate matching smoothness, we follow [31] and report Dirichlet Energies. To this end, we rigidly align shapes using the ground-truth matching, compute a deformation field between matched points, and evaluate the smoothness of this deformation field according to the Laplacian of the target shapes, see [31] for more details.

Geodesic Edge Distortion (GeoED) Dirichlet Energies compute an extrinsic quantity and, with that, depend on the amount of non-rigid transformation between both shapes (even for isometric deformed shapes). To circumvent this issue, we propose a new metric that better captures the intrinsic neighbourhood changes between two shapes. In particular, for every edge $e = (x, \bar{x}) \in \boldsymbol{E_X}$ on shape $\boldsymbol{X}$, we compute the geodesic distance $d_{\boldsymbol{Y}}(\sigma(x), \sigma(\bar{x}))$ on $\boldsymbol{Y}$ where $\sigma : \boldsymbol{V_X} \to \boldsymbol{V_Y}$ is the computed matching.

5.3. Results

Runtime. In Fig. 1 right, we evaluate the runtime of our method (without pruning) over 10 instances of the CP2P24 dataset. We compare to the only other geometrically consistent shape matching, which is specifically designed for partial-partial shape matching. We can see that our method scales better compared to [14], very likely due to our integer linear programming formalism (opposed to the non-linear integer programming formalism proposed in [14]).

CP2P24				
Method	IoU (↑)	GeoError (↓)	Dirichlet (↓)	GeoED (↓)
EchoMatch	<u>84.72</u>	<u>2.29</u>	<u>57.83</u>	6.15
DPFM	74.17	3.04	65.03	4.08
GC-PPSM	68.15	8.87	93.20	**1.58**
Ours	**85.28**	**2.23**	**44.94**	<u>1.69</u>

PSMAL				
Method	IoU (↑)	GeoError (↓)	Dirichlet (↓)	GeoED (↓)
EchoMatch	<u>84.72</u>	<u>4.01</u>	<u>88.05</u>	7.75
DPFM	73.63	9.72	130.44	4.13
GC-PPSM	62.27	11.74	93.12	**1.87**
Ours	**84.96**	**3.87**	**34.52**	<u>2.01</u>

Table 2. **Comparison of quantitative results** ($\times 100$) **on the CP2P24 and PSMAL dataset** in terms of smoothness, overlap prediction and correspondence quality. We indicate **best** and <u>second best</u> results.

Quantitative Evaluation. On both datasets PSMAL and CP2P24, we observe that learning-based methods produce less smooth results according to Dirichlet energies as well as GeoED, see Tab. 2. Additionally, our approach outperforms the other learning-based methods in predicting overlapping regions (mIoU) and in correspondence prediction quality (GeoErrors).

Although GC-PPSM achieves better performance in terms of GeoED, we emphasise that such metrics must always be interpreted in conjunction with overlap prediction, as all other metrics are evaluated on the intersection of the ground truth and the predicted overlapping region.

Qualitative Evaluation. We present qualitative results in Fig. 4. DPFM mainly struggles with overlap prediction

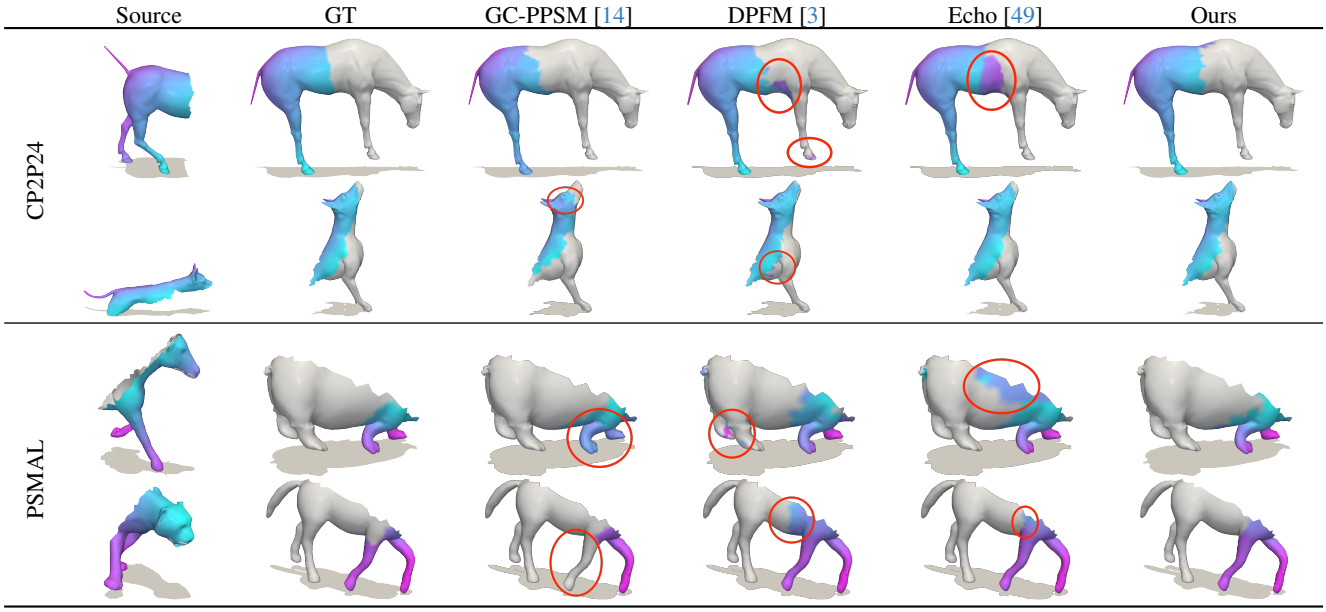

Figure 4. We show **qualitative results** and the ground-truth (GT) of shape pairs of CP2P24 (first two rows) and PSMAL (last two rows) using methods GC-PPSM [14], DPFM [3], Echo [49] and ours. We can see that our method leads to overall smoother matching results.

(see rows 3 and 4), while EchoMatch produces inaccurate correspondence predictions (row 3). GC-PPSM exhibits patch-wise inconsistencies, such as in row 2, which can be attributed to solving at a very low resolution. In contrast, our method generally yields smooth and accurate matchings, with minor inaccuracies near boundaries, likely due to the discretisation artefacts from the initial low-resolution.

5.4. Ablation Studies

We use 10 random samples of the train set of the CP2P [3] and the PSMAL [14] to determine the best parameter for the respective dataset. We show ablation studies for the neighbourhood size when upsampling and the weighting factor in the overlapping region in Tab. 3 in the supplementary.

6. Discussion & Limitations

Our approach is the first integer linear programming formalism for partial-partial shape matching, which only allows for geometrically consistent solutions. Nevertheless, since we build our formalism on GeCo [32], the solution space contains undesirable solutions, such as e.g. orientation flips. Furthermore, resulting matching results might contain multiple, disconnected overlapping regions. Yet, our method empirically shows good performance in terms of matching errors, smoothness and overlap scores.

Apart from that, even though our approach is an integer linear programming formalism (and not a non-linear integer programming formalism compared to previous work [14]), it remains computationally demanding in practice. Consequently, worst-case runtimes remain exponential and limit the scalability of our method. Nevertheless, our approach shows better scalability compared to previous works and we consider our formalism an important step towards making partial-partial shape matching more accessible.

7. Conclusion

We have presented the first integer *linear* programming formalism explicitly tailored to the challenging task of partial-partial 3D shape matching, i.e. our formalism yields neighbourhood preserving matchings of an unknown overlapping region between two partial shapes. Furthermore, by using geometric consistency as a strong prior, our approach allows to curate overlap predictions from other, data-driven approaches as e.g. learning-based methods. In addition, our formalism is more scalable than previous, non-linear integer programming approaches and shows favourable results in terms of matching quality, smoothness, and overlap scores.

Overall, there exists only few methods tackling partial-partial 3D shape matching and there are still several unsolved challenges. In that sense, we consider our work to be an important contribution for the shape matching community and we hope to inspire follow-up works dealing with partial 3D shapes and partial data in general.

8. Acknowledgements

This work was supported by the ERC Advanced Grant "SIMULACRON" (agreement #884679) and the ERC Starting Grant "Harmony" (agreement #101160648).

References

[1] Brian Amberg, Sami Romdhani, and Thomas Vetter. Optimal step nonrigid icp algorithms for surface registration. In *CVPR*, 2007. 2

[2] N. El Amrani, P. Roetzer, and F. Bernard. High-resolution 3d shape matching with global optimality and geometric consistency. *Computer Graphics Forum*, 44(5):e70208, 2025. 2

[3] Souhaib Attaiki, Gautam Pai, and Maks Ovsjanikov. Dpfm: Deep partial functional maps. In *3DV*, 2021. 2, 6, 8

[4] Alex Baden, Keenan Crane, and Misha Kazhdan. Möbius registration. In *CGF*, 2018. 2

[5] Florian Bernard, Zeeshan Khan Suri, and Christian Theobalt. Mina: Convex mixed-integer programming for non-rigid shape alignment. In *CVPR*, 2020. 1

[6] Amit Bracha, Thomas Dagès, and Ron Kimmel. On partial shape correspondence and functional maps. *ACCV*, 2024. 2

[7] Amit Bracha, Thomas Dagès, and Ron Kimmel. Wormhole loss for partial shape matching. *NeurIPS*, 2024. 2

[8] Oliver Burghard and Reinhard Klein. Efficient lifted relaxations of the quadratic assignment problem. In *CVMV*, 2017. 2

[9] Dongliang Cao and Florian Bernard. Unsupervised deep multi-shape matching. In *ECCV*, 2022. 2

[10] Dongliang Cao, Paul Roetzer, and Florian Bernard. Unsupervised learning of robust spectral shape matching. *TOG*, 2023. 1, 2, 6

[11] Luca Cosmo, Emanuele Rodola, Michael M Bronstein, Andrea Torsello, Daniel Cremers, and Y Sahillioglu. Shrec'16: Partial matching of deformable shapes. *3DOR*, 2016. 6

[12] Yan Cui, Sebastian Schuon, Sebastian Thrun, Didier Stricker, and Christian Theobalt. Algorithms for 3d shape scanning with a depth camera. *TPAMI*, 2012. 1

[13] Niladri Shekhar Dutt, Sanjeev Muralikrishnan, and Niloy J. Mitra. Diffusion 3d features (diff3f): Decorating untextured shapes with distilled semantic features. In *CVPR*, 2024. 6

[14] Viktoria Ehm, Maolin Gao, Paul Roetzer, Marvin Eisenberger, Daniel Cremers, and Florian Bernard. Partial-to-partial shape matching with geometric consistency. In *CVPR*, 2024. 1, 2, 3, 6, 7, 8

[15] Viktoria Ehm, Paul Roetzer, Marvin Eisenberger, Maolin Gao, Florian Bernard, and Daniel Cremers. Geometrically consistent partial shape matching. In *3DV*, 2024. 2, 6

[16] Viktoria Ehm, Nafie El Amrani, Yizheng Xie, Lennart Bastian, Maolin Gao, Weikang Wang, Lu Sang, Dongliang Cao, Zorah Lähner, Daniel Cremers, and Florian Bernard. Beyond complete shapes: A benchmark for quantitative evaluation of 3d shape surface matching algorithms. In *Computer Graphics Forum*, page e70186. Wiley Online Library, 2025. 2

[17] Danielle Ezuz, Behrend Heeren, Omri Azencot, Martin Rumpf, and Mirela Ben-Chen. Elastic correspondence between triangle meshes. In *CGF*, 2019. 1

[18] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. 7

[19] Benjamin Holzschuh, Zorah Lähner, and Daniel Cremers. Simulated annealing for 3d shape correspondence. In *3DV*, 2020. 2

[20] Alec Jacobson, Daniele Panozzo, et al. libigl: A simple C++ geometry processing library, 2018. https://libigl.github.io/. 7

[21] Vladimir G Kim, Yaron Lipman, and Thomas Funkhouser. Blended intrinsic maps. *TOG*, 2011. 7

[22] Whoi-Yul Kim and Avinash C. Kak. 3-d object recognition using bipartite matching embedded in discrete relaxation. *TPAMI*, 1991. 1

[23] Yam Kushinsky, Haggai Maron, Nadav Dym, and Yaron Lipman. Sinkhorn Algorithm for Lifted Assignment Problems. *SIAM Journal on Imaging Sciences*, 2019. 2

[24] Zorah Lähner, Emanuele Rodola, Frank R Schmidt, Michael M Bronstein, and Daniel Cremers. Efficient globally optimal 2d-to-3d deformable shape matching. In *CVPR*, 2016. 1, 3, 4

[25] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *ECCV*, 2024. 1

[26] Jiayi Ma, Xingyu Jiang, Aoxiang Fan, Junjun Jiang, and Junchi Yan. Image matching from handcrafted to deep features: A survey. *IJCV*, 2021. 1

[27] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. *TOG*, 2012. 1, 2

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

[29] Emanuele Rodola, Alex M Bronstein, Andrea Albarelli, Filippo Bergamasco, and Andrea Torsello. A game-theoretic approach to deformable shape matching. In *CVPR*, 2012. 1

[30] 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. 2

[31] Paul Roetzer and Florian Bernard. Spidermatch: 3d shape matching with global optimality and geometric consistency. In *CVPR*, 2024. 2, 3, 4, 7

[32] Paul Roetzer and Florian Bernard. Fast globally optimal and geometrically consistent 3d shape matching. In *ICCV*, 2025. 2, 3, 4, 5, 8

[33] Paul Roetzer, Paul Swoboda, Daniel Cremers, and Florian Bernard. A scalable combinatorial solver for elastic geometrically consistent 3d shape matching. In *CVPR*, 2022. 2

[34] Paul Roetzer, Zorah Lähner, and Florian Bernard. Conjugate product graphs for globally optimal 2d-3d shape matching. In *CVPR*, 2023. 1, 3, 4

[35] Paul Roetzer, Ahmed Abbas, Dongliang Cao, Florian Bernard, and Paul Swoboda. Discomatch: Fast discrete optimisation for geometrically consistent 3d shape matching. In *ECCV*, 2024. 2

[36] Paul Roetzer, Viktoria Ehm, Daniel Cremers, , Zorah Lähner, and Florian Bernard. Higher-order ratio cycles for fast and globally optimal shape matching. In *CVPR*, 2025. 3, 4

[37] Yusuf Sahillioğlu. Recent advances in shape correspondence. *The Visual Computer*, 2020. 2

[38] Frank R. Schmidt, Eno Töppe, and Daniel Cremers. Efficient planar graph cuts with applications in computer vision. In *CVPR*, 2009. 1

[39] Patrick Schmidt, Janis Born, Marcel Campen, and Leif Kobbelt. Distortion-minimizing injective maps between surfaces. *TOG*, 2019. 2

[40] Patrick Schmidt, Marcel Campen, Janis Born, and Leif Kobbelt. Inter-surface maps via constant-curvature metrics. *TOG*, 2020. 2

[41] Thomas Schoenemann and Daniel Cremers. A combinatorial solution for model-based image segmentation and real-time tracking. *TPAMI*, 2009. 1, 3

[42] Nicholas Sharp, Souhaib Attaiki, Keenan Crane, and Maks Ovsjanikov. Diffusionnet: Discretization agnostic learning on surfaces. *TOG*, 2022. 6

[43] Justin Solomon, Andy Nguyen, Adrian Butscher, Mirela Ben-Chen, and Leonidas Guibas. Soft maps between surfaces. In *CGF*, 2012. 1

[44] Justin Solomon, Gabriel Peyré, Vladimir G Kim, and Suvrit Sra. Entropic metric alignment for correspondence problems. *TOG*, 2016. 2

[45] Gary KL Tam, Zhi-Quan Cheng, Yu-Kun Lai, Frank C Langbein, Yonghuai Liu, David Marshall, Ralph R Martin, Xian-Fang Sun, and Paul L Rosin. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *TVCG*, 2012. 2

[46] Oliver Van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A survey on shape correspondence. In *CGF*, 2011. 2

[47] Thomas Windheuser, Ulrich Schlickewei, Frank R Schmidt, and Daniel Cremers. Geometrically consistent elastic matching of 3d shapes: A linear programming solution. In *ICCV*, 2011. 1, 2, 3, 7

[48] Thomas Windheuser, Ulrich Schlickwei, Frank R Schimdt, and Daniel Cremers. Large-scale integer linear programming for orientation preserving 3d shape matching. In *CGF*, 2011. 2, 6

[49] Yizheng Xie, Viktoria Ehm, Paul Roetzer, Nafie El Amrani, Maolin Gaoa, Florian Bernard, and Daniel Cremers. Echomatch: Partial-to-partial shape matching via correspondence reflection. In *CVPR*, 2025. 2, 6, 7, 8

[50] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *CVPR*, 2017. 6