Fast Globally Optimal and Geometrically Consistent 3D Shape Matching

Paul Roetzer[1,2] Florian Bernard[1,2]
[1] University of Bonn [2] Lamarr Institute

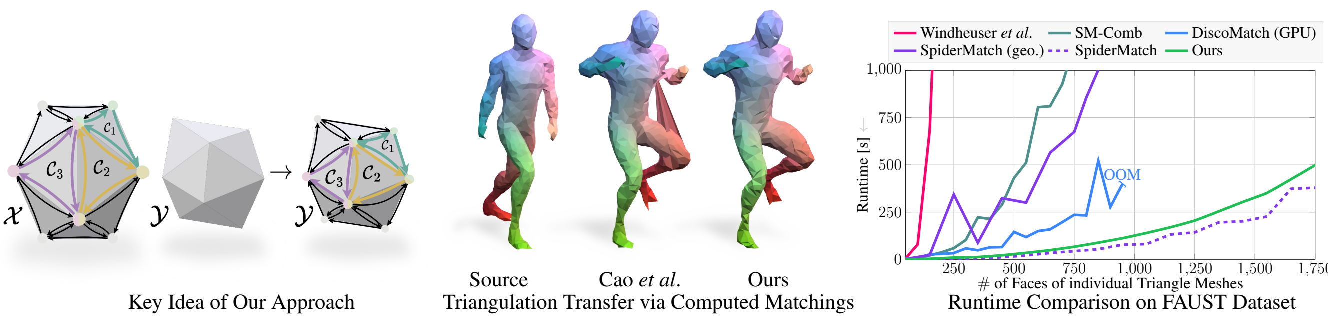

Figure 1. **Left:** Illustration of our approach for geometrically consistent matching of 3D shape $\mathcal{X}$ to 3D shape $\mathcal{Y}$. We represent shape $\mathcal{X}$ using n surface cycles $\mathcal{C}_1, \ldots, \mathcal{C}_n$ and then consistently match these to shape $\mathcal{Y}$ while preserving neighbourhood relations between the cycles. **Middle:** The triangulation- and colour-transfer between shapes using computed matchings of Cao *et al.* [13] (not geometrically consistent) and ours (geometrically consistent) illustrates the importance of geometric consistency. **Right:** Our method is the first 3D shape matching approach that yields globally geometrically consistent matchings, that is globally optimal, and scalable in practice, opposed to all competing methods (OOM stands for 'out of memory' and the dashed line indicates a weak notion of geometric consistency, see explanation of methods in Sec. 5).

Abstract

Geometric consistency, i.e. the preservation of neighbourhoods, is a natural and strong prior in 3D shape matching. Geometrically consistent matchings are crucial for many downstream applications, such as texture transfer or statistical shape modelling. Yet, in practice, geometric consistency is often overlooked, or only achieved under severely limiting assumptions (e.g. a good initialisation). In this work, we propose a novel formalism for computing globally optimal and geometrically consistent matchings between 3D shapes which is scalable in practice. Our key idea is to represent the surface of the source shape as a collection of cyclic graphs, which are then consistently matched to the target shape. Mathematically, we construct a hyper product graph (between source and target shape), and then cast 3D shape matching as a minimum-cost circulation flow problem in this hyper graph, which yields global geometrically consistent matchings between both shapes. We empirically show that our formalism is efficiently solvable and that it leads to high-quality results. Our code is publicly available.[1]

[1] https://github.com/paul0noah/geco

1. Introduction

The availability of correspondences between visual data are a key prerequisite for many visual computing tasks, including 3D reconstruction [50], loss computation for deep learning [14], protein alignment [45], anomaly detection [86], object recognition [44], shape modelling [24] and others. In many practical cases, it is desirable that correspondences preserve neighbourhood relations. For example, in 3D shape matching this could mean that when bringing a pair of neighbouring points on a source shape into correspondence with a pair of points on a target shape, the pair of points must remain neighbours on the target. Such a neighbourhood preservation can serve as strong topological prior, for example to resolve ambiguities, achieve robustness, or ensure well-posedness of an otherwise ill-posed problem. However, despite its crucial importance, neighbourhood preservation is in practice often-times overlooked. This is because many of the well-known formalisms are not efficiently solvable (e.g. the quadratic assignment problem [64], or general integer programming [76], which have been used to tackle image keypoint matching [55], graph matching [39, 91], or 3D shape matching [8, 34, 58, 89]).

In this work we specifically focus on the task of *non-rigid geometrically consistent 3D shape matching*, in which

Method	Globally Optimal	Geometr. Consistent	Scalable
MINA [8]	(✓)	✗	✗
SIGMA [34]	(✓)	✗	✗
PMSDP [58]	✗	✗	✓
Windheuser *et al.* [89]	✗	✓	✗
SpiderMatch [67]	✓	(✓)	(✓)
Ours	✓	✓	✓

Table 1. **Comparison** of axiomatic 3D shape matching methods.

neighbourhood-preserving correspondences between two given 2D manifolds (embedded in 3D space) are sought for. So far, there does not exist any 3D shape matching approach that combines the following three desirable properties: (i) global optimality, (ii) neighbourhood preservation, and (iii) scalability, see Tab. 1. While the recent approach SpiderMatch [67] fulfils (i)-(iii) to some extent, its scalability relies heavily on the choice of the cycle that is used to represent one of the shapes (cf. Fig. 1 right: solid vs. dashed purple line), thereby trading off neighbourhood preservation with scalability. Furthermore, proper geometric consistency (Definition 4) can only be enforced at intersection points of the cycle and thus also the quality of geometric consistency depends on the choice of the cycle. Inspired by this idea of representing a shape using a cycle, we represent one shape using a *collection of cycles* and show that this leads to an (in practice) efficiently solvable formalism which ensures proper geometric consistency. We summarise our main contributions as follows:

- For the first time, we present a globally geometrically consistent formalism for 3D shape matching that is efficiently solvable to global optimality in practice.
- To achieve this, we introduce a novel 3D shape representation in which we represent a 3D shape using a *collection of surface cycles*, so that 3D shape matching can be cast as finding a minimum-cost flow circulation in a *hyper product graph*.
- We experimentally show that our formalism leads to high quality and geometrically consistent matchings between two 3D shapes.
- In addition to 3D shape matching, we also show, in a proof-of-concept manner, that our formalism can generalise to specific instances of other matching problems, such as graph matching when the source graph is planar.

2. Related Work

In the following we discuss works that are most relevant to our approach. We start by discussing efficiently solvable matching problems, continue with 3D shape matching, and conclude with geometrically consistent 3D shape matching.

Efficiently Solvable Matching Problems. There are various instances of correspondence problems that can be solved efficiently. Among them is the linear assignment problem (LAP) [46], which matches points without considering their neighbourhood relations. For matching time series data, sequences, or (open) contours, the popular dynamic time warping algorithm can efficiently compute solutions while preserving neighbourhood relations [70]. Closed contours can be matched analogously via graph cuts in product graphs [71]. Certain tracking problems can be formulated as efficiently solvable flow problems [49], and model-based image segmentation can also be solved efficiently by matching a 2D contour to an image [18, 32, 74]. The matching of a 2D contour to a 3D shape can be solved using variants of Dijkstra's algorithm for finding minimum cost cycles in product graphs [48, 68]. The mentioned approaches show that there are formalisms to efficiently solve a diverse range of matching problems. Yet, these approaches do not directly generalise to 3D shape matching. In this work, we propose a novel geometrically consistent 3D shape matching formalism that is (in practice) efficiently solvable to global optimality.

3D Shape Matching is the task of finding correspondences between two non-rigidly deformed surfaces. For an in-depth overview on 3D shape matching we refer the reader to survey papers [19, 82, 85]. Many axiomatic shape matching approaches [11, 60, 65] build on the functional maps framework, efficiently solving shape matching in the spectral domain [28, 35, 56, 58, 59, 62]. In addition, functional maps have been adopted in several deep shape matching variants, either trained in a supervised [36, 52, 54, 83, 88] or in a unsupervised manner [2, 4, 13, 20, 22, 38, 51, 78]. There are also alternative approaches based on consensus maximisation [61], or using convex relaxations [15, 58] or mixed integer programming [8, 34]. Yet, many of the existing 3D shape matching methods neglect geometric consistency, as it leads to hard-to-solve formalisms.

Geometrically Consistent 3D Shape Matching. There are several approaches that have recognised the importance of geometric consistency in 3D shape matching. Some of them incorporate neighbourhood information by using formalisms based on the quadratic assignment problem (QAP) – yet, the QAP is NP-hard [64], so that respective approaches consider relaxations [12, 23, 47] or heuristics [7, 40, 79]. Opposed to such discrete formulations, there are also approaches that tackle 3D shape matching by deforming a continuous shape parametrisation using local optimisation. However, due to the severe nonconvexity of resulting problems, they rely on a good initialisation, e.g. in the form of sparse sets of landmark correspondences [72, 73, 75, 77, 81], or in the form of dense correspondences [30, 31, 87]. Windheuser *et al.* [89, 90] have modelled geometrically consistent shape matching in the discrete domain by matching triangles to triangles using an (expensive to solve) integer linear program. Re-

Symbol	Description
$\mathcal{X} = (\mathcal{V}_\mathcal{X}, \mathcal{F}_\mathcal{X})$	3D shape
$\mathcal{G}_\mathcal{X} = (\mathcal{V}_\mathcal{X}, \mathcal{E}_\mathcal{X})$	3D shape graph of $\mathcal{X}$
$\mathcal{C} = (\mathcal{V}_\mathcal{C}, \mathcal{E}_\mathcal{C})$	surface cycle (cycle in $\mathcal{G}_\mathcal{X}$)
$\{\mathcal{C}_1, \dots, \mathcal{C}_n\}$	Representation of $\mathcal{X}$ with n surface cycles
n	Number of surface cycles
$\mathcal{Y} = (\mathcal{V}_\mathcal{Y}, \mathcal{F}_\mathcal{Y})$	3D shape $\mathcal{Y}$
$\mathcal{G}_\mathcal{Y} = (\mathcal{V}_\mathcal{Y}, \mathcal{F}_\mathcal{Y})$	3D shape graph of $\mathcal{Y}$
$\mathcal{P}_i = (\mathcal{V}_{\mathcal{P}_i}, \mathcal{E}_{\mathcal{P}_i})$	Product graph (of $\mathcal{C}_i$ and mesh $\mathcal{Y}$)
$\{\mathcal{P}_1, \dots, \mathcal{P}_n\}$	Product graph collection (of $\{\mathcal{C}_1, \dots, \mathcal{C}_n\}$ and mesh $\mathcal{Y}$)
$\mathcal{H} = (\mathcal{V}_\mathcal{H}, \mathcal{E}_\mathcal{H})$	Hyper product graph (coupled product graphs $\{\mathcal{P}, \dots, \mathcal{P}_n\}$)
m	Number of hyper edges $\lvert\mathcal{E}_\mathcal{H}\rvert$
H	Vertex edge incidence matrices of $\mathcal{H}$

Table 2. Summary of the **notation** used in this paper.

cently, approximative solvers [66, 69] and extensions for partial shapes [26, 27] have been proposed. Furthermore, 3D shape matching has recently been formulated as a shortest path problem with additional constraints [67]. In this work, the surface mesh of the source 3D shape is represented using a long self-intersecting curve that traces the shape surface, and is then matched to the other shape while preserving intersections of the curve. While their idea of considering alternative 3D shape representations is promising, in the presented framework proper geometric consistency is only enforced at intersection points of the curve (see Tab. 4). In addition, runtimes grow drastically with an increasing number of intersection points, which can be seen in Fig. 1 right when comparing SpiderMatch [67] (using a curve with few intersections) to SpiderMatch (geo.) (using a curve with many intersections). Inspired by the idea of an alternative path-based 3D shape representation, we represent a 3D shape as a collection of cyclic graphs, and then tackle 3D shape matching by solving *coupled* matching problems of the individual cyclic paths. Based on the couplings, we are able to ensure global geometry consistency. Furthermore, we empirically observe that the resulting formalism is efficiently solvable to global optimality.

3. Background

We consider the task of finding a matching between a source shape $\mathcal{X}$ and a target shape $\mathcal{Y}$ (Sec. 3.1) such that the matching is geometrically consistent (Sec. 3.2). Our main notation is summarised in Tab. 2.

3.1. Shapes and Graphs

In the following, we define shapes and other relevant concepts, which are also illustrated in Fig. 2. We consider *3D shapes* represented as triangular surface mesh:

Definition 1 (3D shape)**.** *A 3D shape $\mathcal{X}$ is defined as a tuple $(\mathcal{V}_\mathcal{X}, \mathcal{F}_\mathcal{X})$ of vertices $\mathcal{V}_\mathcal{X}$ and consistently oriented (e.g. clock-wise) triangles $\mathcal{F}_\mathcal{X} \subset \mathcal{V}_\mathcal{X} \times \mathcal{V}_\mathcal{X} \times \mathcal{V}_\mathcal{X}$, such that $\mathcal{X}$ forms an orientable continuous 2D manifold (possibly with boundary) embedded in 3D space.*

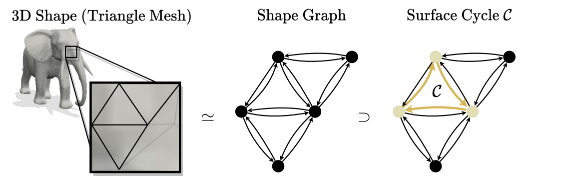

Figure 2. We represent a 3D shape (left) as a (directed) shape graph $\mathcal{G}_\mathcal{X}$ (middle). We call a cycle $\mathcal{C}$ in $\mathcal{G}_\mathcal{X}$ a *surface cycle* (right).

For our formalism it is advantageous to interpret a 3D shape as a graph, which we denote the *shape graph*:

Definition 2 (Shape graph)**.** *The* shape graph *(of 3D shape $\mathcal{X}$) is a tuple $\mathcal{G}_\mathcal{X} = (\mathcal{V}_\mathcal{X}, \mathcal{E}_\mathcal{X})$ of vertices $\mathcal{V}_\mathcal{X}$ and directed edges $\mathcal{E}_\mathcal{X} \subset \mathcal{V}_\mathcal{X} \times \mathcal{V}_\mathcal{X}$, such that each triangle in $\mathcal{F}_\mathcal{X}$ is represented by three unique and consistently directed edges.*

In addition to the shape graph (representing the whole 3D shape $\mathcal{X}$), we consider certain subgraphs of $\mathcal{G}_\mathcal{X}$:

Definition 3 (Surface cycle)**.** *A* surface cycle *(on 3D shape $\mathcal{X}$) is a tuple $\mathcal{C} = (\mathcal{V}_\mathcal{C}, \mathcal{E}_\mathcal{C})$ with (non-empty) vertices $\mathcal{V}_\mathcal{C} \subset \mathcal{V}_\mathcal{X}$ and (non-empty) directed edges $\mathcal{E}_\mathcal{C} \subset \mathcal{V}_\mathcal{C} \times \mathcal{V}_\mathcal{C}$, such that (i) $\mathcal{E}_\mathcal{C} \subset \mathcal{E}_\mathcal{X}$, and (ii) each vertex $v \in \mathcal{V}_\mathcal{C}$ has exactly one incoming and one outgoing edge in $\mathcal{E}_\mathcal{C}$.*

3.2. Geometrically Consistent 3D Shape Matching

We define geometrically consistent 3D shape matching in terms of the respective shape graphs:

Definition 4 (Geometrically Consistent Matching)**.** *The mapping $\phi : \mathcal{V}_\mathcal{X} \to \mathcal{V}_\mathcal{Y}$ is a* geometrically consistent matching *from the source shape $\mathcal{X}$ to the target shape $\mathcal{Y}$ if ϕ preserves neighbourhoods as follows: whenever two vertices $x, \bar{x} \in \mathcal{V}_\mathcal{X}$ are connected (i.e. $(x, \bar{x}) \in \mathcal{E}_\mathcal{X}$ or $(\bar{x}, x) \in \mathcal{E}_\mathcal{X}$), then their corresponding vertices $\phi(x), \phi(\bar{x}) \in \mathcal{V}_\mathcal{Y}$ must either be (i) connected (i.e. $(\phi(x), \phi(\bar{x})) \in \mathcal{E}_\mathcal{Y}$ or $(\phi(\bar{x}), \phi(x)) \in \mathcal{E}_\mathcal{Y}$), or (ii) the same vertex (i.e. $\phi(x) = \phi(\bar{x})$).*

The intuition is that neighbouring elements of $\mathcal{X}$ must be matched to neighbouring elements of $\mathcal{Y}$. We note that we require both shapes to have equal genus such that our definition of geometric consistency makes sense.

4. Our 3D Shape Matching Approach

In this section we develop our geometrically consistent 3D shape matching formulation which ensures global geometric consistency. Our approach is based on representing the

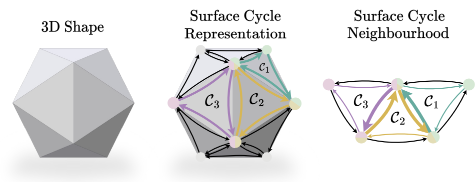

Figure 3. **Surface cycle collection** (middle) for representing the 3D shape of an icosahedron (left). Individual surface cycles are glued together via shared edges (neighbouring surface cycles share opposite edges, cf. pairs of thickened edges on the right: ‖ and ‖).

source 3D shape using a collection of surface cycles, which we then match to the target shape (Sec. 4.1). To this end, we formulate an individual subproblem for each surface cycle (Sec. 4.2.1). Further, we couple individual subproblems (Secs. 4.2.2 and 4.2.3) so that our matchings are geometrically consistent according to Definition 4. Finally, we show that our overall formalism can be interpreted as a minimum-cost circulation flow problem on a hyper product graph (Sec. 4.5).

4.1. Surface Cycle-based 3D Shape Representation

We aim to represent the surface of 3D shape $\mathcal{X}$ with a collection of surface cycles which are glued together at *opposite edges*:

Definition 5 (Opposite edge)**.** *The opposite edge to an edge* (v, w) *is defined as* $-(v, w) := (w, v)$.

We note that for every non-boundary edge $e \in \mathcal{E}_{\mathcal{X}}$ of shape $\mathcal{X}$, the opposite edge $-e \in \mathcal{E}_{\mathcal{X}}$ is also part of the shape graph $\mathcal{G}_{\mathcal{X}}$ (cf. Fig. 3 right) because the pair of opposite edges e and $-e$ belong to two neighbouring triangles.

Definition 6 (Shape as collection of surface cycles)**.** *We represent a 3D shape $\mathcal{X}$ with a collection of $n \in \mathbb{N}^+$ surface cycles $\mathcal{C}_1, \ldots, \mathcal{C}_n$ (on shape $\mathcal{X}$) that partition the surface of $\mathcal{X}$ into n polygonal patches, such that:*
(i) $\mathcal{E}_{\mathcal{C}_i} \cap \mathcal{E}_{\mathcal{C}_j} = \emptyset$ for all $i, j \in \{1, \ldots, n\}, i \neq j$, and
(ii) for every non-boundary edge $e \in \mathcal{E}_{\mathcal{C}_i}$ of shape $\mathcal{X}$ there exists j such that $-e \in \mathcal{E}_{\mathcal{C}_j}$.

Condition (i) ensures that each edge of $\mathcal{X}$ can be part of at most one surface cycle, and (ii) ensures that neighbouring surface cycles cover opposite edges, see Fig. 3. We note that from here on we assume that each surface cycle represents an individual triangle of shape $\mathcal{X}$. We discuss more general polygonal surface cycles in Sec. C.3 in the supplementary.

4.2. Our Coupled Product Graph Formalism

On a high-level, we use the product graph formalism introduced in [48] to address the matching between individual surface cycles and shape $\mathcal{Y}$, which results in geometric consistency *along the path within each individual cycle*. Further, we introduce coupling constraints that have the effect of glueing neighbouring product graphs together. Combined with additional injectivity constraints, our matching is globally geometrically consistent, i.e. it *preserves neighbourhoods between surface cycles*, see Fig. 4 for an overview.

4.2.1. Individual Surface Cycle Matching Subproblems

Each individual surface cycle $\mathcal{C}_i$ of $\mathcal{X}$ can be matched to shape $\mathcal{Y}$ by finding a cyclic path in their *product graph* $\mathcal{P}_i = (\mathcal{V}_{\mathcal{P}_i}, \mathcal{E}_{\mathcal{P}_i})$. It has been introduced in [48] for the problem of matching a 2D shape to a 3D shape and reads:

Definition 7 (Product graph $\mathcal{P}_i$)**.** *The product graph $\mathcal{P}_i = (\mathcal{V}_{\mathcal{P}_i}, \mathcal{E}_{\mathcal{P}_i})$ between the i-th surface cycle $\mathcal{C}_i$ and the 3D shape $\mathcal{Y}$ is a directed graph defined as*

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

with the extended edge set $\mathcal{E}^+_{\mathcal{Y}} := \mathcal{E}_{\mathcal{Y}} \cup \{(y, y) \mid y \in \mathcal{V}_{\mathcal{Y}}\}$.

The key idea is that each edge $(v, \bar{v}) \in \mathcal{E}_{\mathcal{P}_i}$ in the product graph can be interpreted as a (potential) matching between an edge of $\mathcal{C}_i$ and an edge (or a vertex to account for stretching/compression) of shape $\mathcal{Y}$. As shown in [48], matching surface cycle $\mathcal{C}_i$ to shape $\mathcal{Y}$ amounts to solving a cyclic shortest path problem in the product graph $\mathcal{P}_i$.

However, for our setting of *3D-to-3D shape matching*, the neighbourhood between pairs of surface cycles cannot be ensured when considering vanilla shortest path algorithms, since they would solve the n individual surface cycle matching subproblems independently. To tackle this, we consider the linear programming (LP) formalism of cyclic shortest path problems and add additional constraints to couple the individual product graphs $\mathcal{P}_1, \ldots, \mathcal{P}_n$, as explained next.

4.2.2. Subproblem Coupling

We couple the individual surface cycle matching subproblems $\mathcal{P}_1, \ldots, \mathcal{P}_n$ by glueing them together at opposite edges. To this end, we introduce *coupling constraints* $\mathcal{V}_{\mathcal{L}}$ which serve the purpose of ensuring that matchings of opposite edges are consistent, i.e. resulting matchings (or rather resulting shortest cyclic paths of neighbouring surface cycles) must go through opposite edges, see Fig. 4 (iii). Defacto, this enforces the *glueing* of neighbouring product graphs, and thus ensures geometric consistency of neighbouring surface cycles. In addition to the coupling, we want to ensure injectivity for each surface cycle edge, i.e. each such edge is matched exactly once.

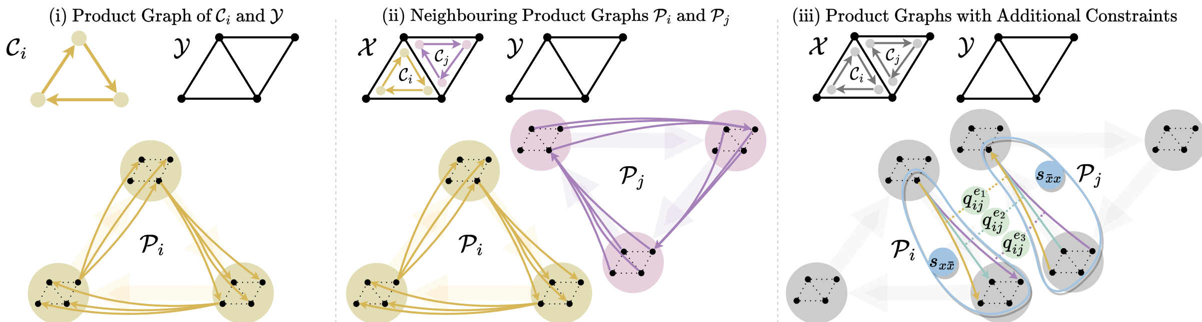

Figure 4. Conceptual summary of our formalism. **Left:** Illustration of product graph $\mathcal{P}_i$ between the surface cycle $\mathcal{C}_i$ and shape graph $\mathcal{Y}$. Each black dot inside the big yellow circle represents a product vertex. Overall, the product graph can be thought of having one copy of $\mathcal{Y}$ for each vertex of $\mathcal{C}_i$, which are appropriately connected via edges (only a subset of edges are drawn to reduce visual clutter). **Middle:** Two neighbouring product graphs $\mathcal{P}_i$ and $\mathcal{P}_j$ that arise from two neighbouring surface cycles $\mathcal{C}_i$ and $\mathcal{C}_j$. **Right:** Illustration of constrained neighbouring product graphs. We add additional coupling constraints so that opposite edges of the product graphs (i.e. edges with same colour, see coloured lines ▬,▬,▬) are coupled via coupling constraints $q_{ij}^{e_1}, q_{ij}^{e_2}, q_{ij}^{e_3} \in \mathcal{V}_{\mathcal{L}}$ (see green dots ●). Furthermore, we bundle edges of the product graphs so that each edge of the surface cycles is matched exactly once. To this end, we add injectivity constraints $s_{x\bar{x}}$, $s_{\bar{x}x} \in \mathcal{V}_{\mathcal{S}}$ (see blue dots ● within bundles of edges).

4.2.3. Surface Cycle Matching Injectivity

To enforce that each surface cycle edge is matched exactly once, i.e. it is matched to exactly one edge (or vertex) of $\mathcal{Y}$, we introduce *injectivity constraints* $\mathcal{V}_{\mathcal{S}}$. For that, let us denote the set of all edges (in the product graph) that are potential matching candidates of the edge $(x, \bar{x}) \in \mathcal{E}_{\mathcal{C}_i}$ as the *edge bundle of* $(x, \bar{x})$. For each such edge bundle, we introduce one injectivity constraint (see Fig. 4 (iii)), which has the purpose to ensure that only a single edge of each edge bundle is part of the final matching.

4.3. Constraint Matrices

As mentioned, we use the LP formalism of the independent cyclic shortest path problems and add additional constraints for coupling an matching injectivity. To this end, we consider the constraint matrix $H = \left[\begin{smallmatrix} P \\ L \\ S \end{smallmatrix}\right]$ consisting of submatrices P, L and S. Here, submatrix P represents the collection of incidence matrices of individual product graphs, submatrix L the couplings, and submatrix S the injectivity components.

The matrix P**.** The vertex edge incidence matrix $P \in \{-1, 0, 1\}^{c \times m}$ (with $m = |\mathcal{E}_{\mathcal{H}}|$ columns, i.e. one column for each product edge, and $c = \sum_{i=1}^{n} |\mathcal{V}_{\mathcal{C}_i}||\mathcal{V}_{\mathcal{Y}}|$ many rows) represents all n product graphs and is defined as $P := \text{diag}(P_1, \ldots, P_n)$, i.e. it contains the vertex edge incidence matrices P_i of individual product graphs $\mathcal{P}_i$ as diagonal blocks. To illustrate the structure of the matrix representation of one $\mathcal{P}_i$, we now discuss the simplified case of non-degenerate matchings (i.e. we do not allow edge to vertex matchings, see Sec. A in the supplementary for more details). In this case, the incidence matrix P_i reads

$$P_i := C_i^+ \otimes Y^+ - C_i^- \otimes Y^-, \tag{2}$$

with $\otimes$ being the Kroenecker product and with $C_i \in \{-1, 0, 1\}^{|\mathcal{V}_{\mathcal{C}_i}| \times |\mathcal{E}_{\mathcal{C}_i}|}$ and $Y \in \{-1, 0, 1\}^{|\mathcal{V}_{\mathcal{Y}}| \times |\mathcal{E}_{\mathcal{Y}}|}$ being the incidence matrices of surface cycle $\mathcal{C}_i$ and shape graph $\mathcal{G}_{\mathcal{Y}}$ respectively (we note that C_i has exactly two non-zeros of opposite sign in each row and column since it is the incidence matrix of a cycle). We use the notation $+$ and $-$ to split the incidence matrix into incoming and outgoing incidence matrix, respectively. In other words, C_i^+ and C_i^- as well as Y^+ and Y^- are the non-negative and non-positive entries of the incidence matrices C_i and Y, respectively. We note that the sign splitting of incidence matrices C_i and Y is necessary to account for proper edge directions in the resulting product graph. By definition, their edge directions induce edge directions of product edges of respective product graph $\mathcal{P}_i$. Each P_i has a block structure with $|\mathcal{V}_{\mathcal{Y}}| \times |\mathcal{E}_{\mathcal{Y}}|$ sized blocks and each block contains either only $+1$ or -1 entries. Further, each block contains exactly one non-zero per column, see Fig. A.1 in the supplementary.

The matrix L**.** The coupling of opposite edges of neighbouring product graphs is captured in the matrix

$$L := K \otimes \boldsymbol{I}_{|\mathcal{E}_{\mathcal{Y}}^+|}. \tag{3}$$

Here $K \in \{-1, 0, 1\}^{p \times |\mathcal{E}_{\mathcal{X}}|}$ (p is the number of undirected non-boundary edges of $\mathcal{X}$) is an incidence matrix which represents the incidence of opposite edges across all n surface cycles. K has at most one non-zero element per column and exactly two non-zeros per row (with opposite sign) and thus L has identical block structure with non-negative and non-positive blocks respectively. We note that the definition of P requires columns of the non-positive blocks of L to be permuted (so that it maps opposite edges of neighbouring product graphs), see Sec. A in the supplementary.

The matrix S**.** The bundling of edges of all n product

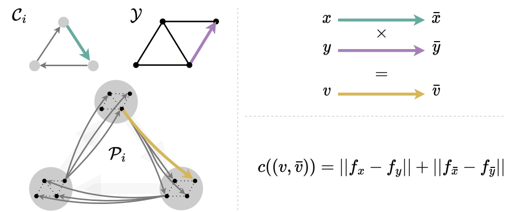

Figure 5. Visualisation of **edge cost** $c((v, \bar{v}))$ **computation** for the product edge $(v, \bar{v})$. The product edge defines a potential matching between edges $(x, \bar{x}) \in \mathcal{E}_\mathcal{X}$ of shape $\mathcal{X}$ and $(y, \bar{y}) \in \mathcal{E}_\mathcal{Y}^+$ of shape $\mathcal{Y}$. Thus, we define the matching cost $c((v, \bar{v}))$ as feature difference between features $\|f_x - f_y\|$ and $\|f_{\bar{x}} - f_{\bar{y}}\|$ at respective source vertices x, y and target vertices $\bar{x}$, $\bar{y}$.

graphs can be described using the matrix

$$S := \boldsymbol{I}_{|\mathcal{E}_\mathcal{X}|} \otimes 1^T_{|\mathcal{E}_\mathcal{Y}^+|}. \tag{4}$$

Here, S has block diagonal structure consisting of $|\mathcal{E}_\mathcal{X}|$ many non-negative blocks (i.e. $1^T_{|\mathcal{E}_\mathcal{Y}^+|}$ blocks). We note that $Sx = 1$ enforces matching injectivity by enumerating product edges such that all $|\mathcal{E}_\mathcal{Y}^+|$-many matching candidates of a single surface cycle are grouped.

4.4. Resulting Integer Linear Program

We use the previously described constraint matrix H to cast geometrically consistent shape matching as a linear program which essentially represents n many coupled cyclic shortest path problems.

Matching costs. The matching cost $c(e)$ of product edge $e = \left(\binom{x}{y}, \binom{\bar{x}}{\bar{y}}\right)$ measures how well the edge $(x, \bar{x}) \in \mathcal{E}_\mathcal{X}$ of shape $\mathcal{X}$ and the edge $(y, \bar{y}) \in \mathcal{E}_\mathcal{Y}^+$ of shape $\mathcal{Y}$ fit together, e.g. by comparing geometric properties or feature descriptors, see also Fig. 5.

Optimisation problem. We collect matching costs of all m edges of the n product graphs in the vector $c \in \mathbb{R}^m_+$. With that, we can use an indicator representation x for each edge ($x_k = 1$ means that k-th edge is part of the final matching) so that our matching formalism reads

$$\min_{x \in \{0,1\}^m} c^T x \quad \text{s.t.} \quad Hx = b. \tag{GeCo3D}$$

Here, H is the previously introduced constraint matrix and $b := \left[0_{|\mathcal{V}_\mathcal{H}| - |\mathcal{V}_S|}; 1_{|\mathcal{V}_S|}\right]$ is a column vector with all zeros except for ones in rows belonging to rows of submatrix S of H. We solve (GeCo3D) by considering its LP-relaxation, i.e. we replace $x \in \{0,1\}^m$ with $x \in [0,1]^m$. We empirically observe in our experiments, that all solutions are integral and consequently globally optimal.

Lemma 8. *Matchings between shapes $\mathcal{X}$ and $\mathcal{Y}$ obtained by solving (GeCo3D) are globally geometrically consistent according to Definition 4.*

Proof. We represent each triangle of the source shape $\mathcal{X}$ using a surface cycle. Furthermore, matchings obtained by solving (GeCo3D) preserve neighbourhood relations along the path of a matched surface cycle as well as between the surface cycles. Overall, this ensures that connected vertices on $\mathcal{X}$ are only matched to connected vertices (or the same vertex) on shape $\mathcal{Y}$. □

4.5. Hyper Product Graph Interpretation

Our optimisation problem (GeCo3D) can be interpreted as finding a minimum-cost flow circulation in a directed hyper product graph $\mathcal{H}$, see [6] for definitions of flows in directed hyper graphs. Here, $\mathcal{H}$ can be obtained by interpreting H as a vertex hyper edge incidence matrix, i.e. by interpreting rows of H as vertices and columns of H as directed hyper edges, see Fig. 6 for an illustration. For a formal definition of $\mathcal{H}$ we refer to Sec. B in the supplementary.

In general, such flow circulation problems on hypergraphs are NP-hard [5]. However, there are certain subclasses that are known to be solvable in polynomial time [10, 29, 42, 80, 84]. Unfortunately, the specific structure of our hyper graph is not listed among the known polynomial-time solvable subclasses [10, 29, 42, 80, 84]. Nevertheless, we empirically observe that all instances considered in our experiments yield an integral (and thus globally optimal) solution when solved using LP-relaxations.

5. Experiments

In this section, we experimentally evaluate our method's performance for 3D shape matching. Further, we show as proof of concept that our method is applicable to planar graph matching.

Setup. We conduct runtime experiments on an Intel Core i9 12900K with 128 GB DDR5 RAM. For solving our linear programm (GeCo3D), we use off-the-shelf solver Gurobi [37] (version 10). We explicitly resolve coupling constraints during problem construction to obtain smaller constraint matrices, see Sec. C.2 in the supplementary. To better account for shrinking and stretching, we integrate a distortion bound, see Sec. C.1 in the supplementary. For

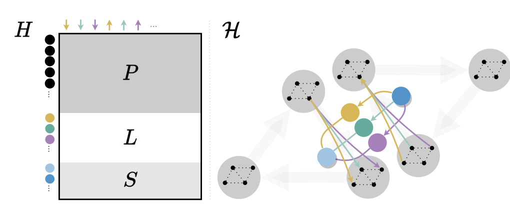

Figure 6. Illustration of interpreting our constraint matrix H as the **vertex hyper edge incidence matrix** of a hyper product graph $\mathcal{H}$, see Sec. B in the supplementary. To obtain $\mathcal{H}$, rows of H are interpreted as vertices, and columns as directed hyper edges of $\mathcal{H}$.

quantitive comparison, we decimate shapes to 1000 triangles using algorithms provided in [41]. We use the deep features [13] for all methods except for SmoothShells [28] and DiscrOpt [62] for which we use the original (axiomatic) features as reported in respective papers.

Metrics. We evaluate matching accuracy using geodesic errors, i.e. geodesic distance to ground-truth matchings. We follow the Princeton protocol [43] and normalise errors by the square-root of the shape area (see [43, Sec. 8.2]). Furthermore, we evaluate geometric consistency of matchings using Dirichlet energies, i.e. the deformation energy induced by the matching (see [67, Sec. 8]).

Methods. We compare the matching quality of the methods ULRSSM [13], SmoothShells [28], DiscrOpt [62] and SpiderMatch [67]. ULRSSM [13] is an unsupervised deep shape matching method achieving best matchings results on numerous benchmarks. Because of its good performance and its extensive evaluation, we consider it to be representative for deep shape matching methods. SmoothShells [28] is a functional map based shape alignment method which iteratively adds geometric information to compute matchings. DiscrOpt [62] is a functional map based method which relates functional maps to point-wise maps. SpiderMatch [67] is the only other method that is scalable and at the same time considers geometric consistency. Furthermore, for runtime evaluations, we only compare to other geometrically consistent methods since these aim to solve a much harder to solve problem (yet most of these methods do not scale well to relevant shape resolutions, see also Fig. 1 right). In this sense, we consider Windheuser *et al*. [89], SM-Comb [69], DiscoMatch [66] and SpiderMatch (geo.). Windheuser *et al*. [89] propose a geometrically consistent 3D shape matching formalism which they solve using LP-relaxations and iterative variable fixations. The methods SM-Comb [69] and DiscoMatch [66] aim to solve this formalism approximately on CPU and GPU, respectively. We use SpiderMatch (geo.) to indicate the SpiderMatch [67] method when using a different curve to represent the source shape: we consider a curve that covers *all edges* of the source shape and thus, contains intersections at every vertex of the source shape. This is necessary to guarantee global geometric consistency since SpiderMatch [67] can only ensure geometric consistency (according to Definition 4) at intersection points of the curve. Thus, SpiderMatch (geo.) yields a stronger notion of geometric consistency compared to SpiderMatch [67].

Datasets. We evaluate shape matching on four different datasets: remeshed FAUST [9, 21, 63] (100 near-isometric deformed human shapes from which we sample 100 test set pairs), SMAL [92] (49 non-isometric deformed animal shapes of eight species from which we sample 100 test set pairs), DT4D-H [57] (9 different classes of humanoid/game character shapes in different poses taken from DeformingThings4D [53] from which we sample 100 intra class test set pairs and 100 inter class test set pairs) and BeCoS [25] (2543 animal and humanoid shapes from various datasets of which we consider the 141 full-to-full test set pairs).

Method	FAUST	SMAL	DT4D Intra	DT4D Inter	BeCoS
ULRSSM [13]	0.031	0.048	0.033	0.041	0.057
SmoothShells [28]	0.379	0.376	0.373	0.420	0.370
DiscrOpt [62]	0.110	0.268	0.075	0.170	0.270
SpiderMatch [67]	0.029	**0.044**	0.027	0.041	0.057
Ours	**0.027**	**0.044**	**0.024**	**0.039**	**0.056**

Table 3. Comparison of **mean geodesic errors** (↓) of various shape matching methods. We can see that our method consistently outperforms other methods on all five datasets.

5.1. 3D Shape Matching

In the following, we evaluate our method's shape matching performance w.r.t. runtime and matching quality.

Runtime. In Fig. 1 right, we show runtime comparison to formalisms which consider geometric consistency. Curves show median runtimes over five instances of FAUST dataset. Among all methods, ours and SpiderMatch [67] scale best. They can handle 3D shapes with approximately twice as many vertices as their fastest competitors DiscoMatch [66], which tackles the problem proposed by Windheuser *et al*. [89]. However, we note that when considering SpiderMatch (geo.) and consequently curves with more intersections (which is necessary for global geometric consistency) the scalability of SpiderMatch (geo.) degrades drastically. This stems from the branch and bound algorithm (having exponential worst-case runtime in general) which is used to solve SpiderMatch [67]. In contrast, we empirically observe (in all of our tested instances) that the linear programming relaxation of (GeCo3D) is always tight (i.e. yields an integral optimal solution) which explains the scalability of our approach. We emphasise that matchings computed from (GeCo3D) are provably geometrically consistent according to Definition 4, while the matchings of SpiderMatch [67] are not (cf. also Tab. 4).

Full Shape Matching. In Tab. 3 and Tab. 4, we quantitatively compare the shape matching performance of various methods. Our method consistently produces best results across all datasets w.r.t. mean geodesic errors and smoothest results w.r.t Dirichlet energies. This showcases the beneficial effect of strong priors induced by geometric consistency. In addition, in Fig. 7, we show qualitative results, which further showcases high-quality matching results computed with our method. We provide more results as well as ablation studies in Sec. D.1 in the supplementary.

Partial-to-Full Shape Matching. As proof of concept we show qualitative results of matchings computed with our method in the partial-to-full setting, see Fig. 8.

Method	FAUST	SMAL	DT4D Intra	DT4D Inter	BeCoS
ULRSSM [13]	2.1	2.6	3.4	2.6	4.1
SmoothShells [28]	1.9	2.8	1.6	2.3	5.7
DiscrOpt [62]	8.9	13.6	7.5	10.7	1.1
SpiderMatch [67]	1.7	1.8	1.7	2.1	0.81
Ours	**0.46**	**0.53**	**0.48**	**0.62**	**0.77**

Table 4. Comparison of **mean Dirichlet energies** (↓) of various shape matching methods to measure smoothness of the matching. Our method consistently yields best results, which shows the effect of strong priors induced by global geometric consistency.

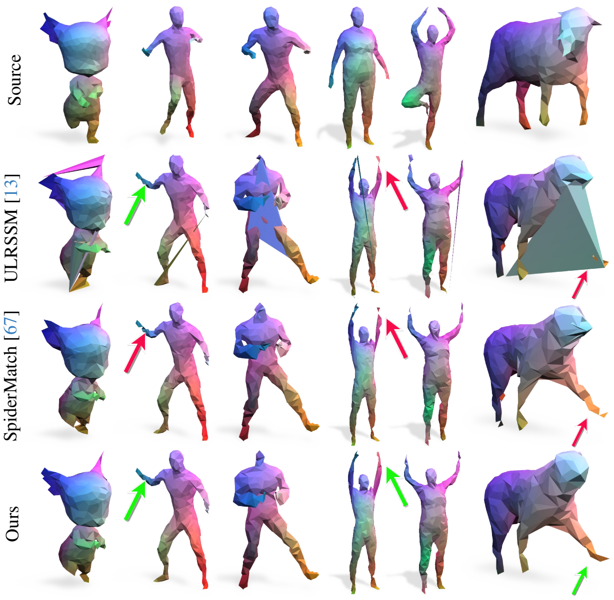

Figure 7. **Qualitative shape matching results** computed with ULRSSM [13], SpiderMatch [67] and ours. We visualise matchings using colour and triangulation transfer from source to target shape. At legs and arms of shapes we can see stronger geometric consistency of ours compared to SpiderMatch [67] (see red and green arrows and distorted triangles).

5.2. Application to Planar Graph Matching

As a proof of concept we show that our formalism directly applies to planar graph matching. To this end, we use keypoints on images of the WILLOW [16] dataset and obtain graphs by computing Delaunay triangulations of keypoints. Furthermore, we represent one of the resulting graphs using surface cycles to match it to the other graph, see Fig. 9 and in Sec. D.2 in the supplementary.

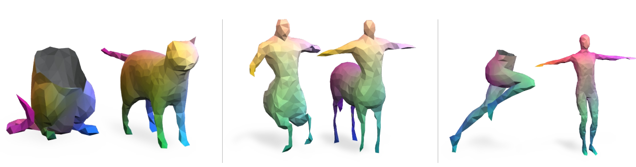

Figure 8. Qualitative results computed with ours for **partial-to-full** shape matching of test set shapes [26] from SHREC'16 [17].

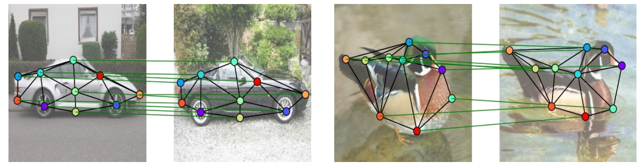

Figure 9. Examples of **planar graph matching** results using our method on ducks and cars from WILLOW [16] dataset.

6. Discussion & Limitations

Our algorithm enforces geometric consistency and thus introduces a strong prior to resolve ambiguities in 3D shape matching problems. Yet, in theory our formalism contains undesirable matchings in the solution space: On the one hand, multiple surface cycles could be matched to the same vertex on the target shape. While this ensures our optimsation problem is always feasible (since it contains the extreme case of matching all surface cycles to a single vertex), this is an undesirable solution. In the future, this could be resolved by generalising our approach towards a symmetric formalism. On the other hand, our formalism allows for inside-out flips, as each individual matching element (i.e. an edge) cannot disambiguate extrinsic orientations.

We have empirically shown that results can be computed efficiently using off-the-shelf LP-solvers [37] and that in all considered instances the linear the programming relaxations are tight, i.e. we find a globally optimal solution for all instances. Yet, we observe that our constraint matrix is not totally unimodular. We leave an in-depth analysis of our formalism, along with answering the question whether there exists a polynomial time algorithm for geometrically consistent 3D shape matching, for future works.

7. Conclusion

We have presented a novel formalism for non-rigid 3D shape matching which globally enforces geometric consistency. Our key idea is to consider a shape representation that allows to cast 3D shape matching as an integer linear program, which is efficiently solvable in practice. In addition, we illustrate that the resulting problem can be interpreted as a minimum-cost flow circulation problem in a hyper graph. Overall, we consider our work to be important for the 3D shape analysis community. Geometric consistency is crucial in most practical 3D shape matching settings, which is often times ignored (especially by learning-based methods). Our work allows to obtain such matchings efficiently and with that poses a step towards integrating geometric consistency into learning-based methods. Furthermore, we hope to inspire follow-up works on neighbourhood preserving matching formalisms within the broader field of visual computing and beyond.

Acknowledgements

We thank Johan Thunberg, Tobias Weißberg and Zorah Lähner for insightful discussions about our formalism. This work is supported by the ERC starting grant no. 101160648 (Harmony).

References

[1] MOSEK ApS. *MOSEK Optimizer API for Python 11.0.9*, 2025. 5, 6

[2] Souhaib Attaiki and Maks Ovsjanikov. Understanding and improving features learned in deep functional maps. In *CVPR*, 2023. 2

[3] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *ICCV workshops*. IEEE, 2011. 4, 5

[4] Lennart Bastian, Yizheng Xie, Nassir Navab, and Zorah Lähner. Hybrid functional maps for crease-aware non-isometric shape matching. In *CVPR*, 2024. 2

[5] Isabel Beckenbach. A hypergraph network simplex algorithm. In *GOR*, 2018. 6

[6] Isabel Leonie Beckenbach. *Matchings and flows in hypergraphs*. PhD thesis, 2019. 6

[7] Marcel Seelbach Benkner, Zorah Lähner, Vladislav Golyanik, Christof Wunderlich, Christian Theobalt, and Michael Moeller. Q-match: Iterative shape matching via quantum annealing. In *ICCV*, 2021. 2

[8] Florian Bernard, Zeeshan Khan Suri, and Christian Theobalt. Mina: Convex mixed-integer programming for non-rigid shape alignment. In *CVPR*, 2020. 1, 2

[9] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. Faust: Dataset and evaluation for 3d mesh registration. In *CVPR*, 2014. 7

[10] Kateřina Böhmová, Jérémie Chalopin, Matúš Mihalák, Guido Proietti, and Peter Widmayer. Sequence hypergraphs: paths, flows, and cuts. *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday*, 2018. 6

[11] Alexander M Bronstein, Michael M Bronstein, Ron Kimmel, Mona Mahmoudi, and Guillermo Sapiro. A gromov-hausdorff framework with diffusion geometry for topologically-robust non-rigid shape matching. *International Journal of Computer Vision*, 2010. 2

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

[13] Dongliang Cao, Paul Roetzer, and Florian Bernard. Unsupervised learning of robust spectral shape matching. *TOG*, 2023. 1, 2, 7, 8, 4, 5, 6

[14] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1

[15] Qifeng Chen and Vladlen Koltun. Robust nonrigid registration by convex optimization. In *ICCV*, 2015. 2

[16] Minsu Cho, Karteek Alahari, and Jean Ponce. Learning graphs to match. In *ICCV*, 2013. 8, 6

[17] Luca Cosmo, Emanuele Rodola, Michael M Bronstein, Andrea Torsello, Daniel Cremers, and Y Sahillioglu. Shrec'16: Partial matching of deformable shapes. *3DOR*, 2016. 8, 6, 9

[18] James Coughlan, Alan Yuille, Camper English, and Dan Snow. Efficient deformable template detection and localization without user initialization. *CVIU*, 2000. 2

[19] Bailin Deng, Yuxin Yao, Roberto M Dyke, and Juyong Zhang. A survey of non-rigid 3d registration. In *CGF*, 2022. 2

[20] Nicolas Donati, Etienne Corman, and Maks Ovsjanikov. Deep orientation-aware functional maps: Tackling symmetry issues in shape matching. In *CVPR*, 2022. 2

[21] Nicolas Donati, Abhishek Sharma, and Maks Ovsjanikov. Deep geometric functional maps: Robust feature learning for shape correspondence. In *CVPR*, 2020. 7

[22] Niladri Shekhar Dutt, Sanjeev Muralikrishnan, and Niloy J. Mitra. Diffusion 3d features (diff3f): Decorating untextured shapes with distilled semantic features. In *CVPR*, 2024. 2, 4, 5

[23] Nadav Dym, Haggai Maron, and Yaron Lipman. DS++ - A Flexible, Scalable and Provably Tight Relaxation for Matching Problems. *TOG*, 2017. 2

[24] Bernhard Egger, William A. P. Smith, Ayush Tewari, Stefanie Wuhrer, Michael Zollhoefer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, Christian Theobalt, Volker Blanz, and Thomas Vetter. 3d morphable face models—past, present, and future. *TOG*, 2020. 1

[25] Viktoria Ehm, Nafie El Amrani, Yizheng Xie, Lennart Bastian, Maolin Gao, Weikang Wang, Lu Sang, Dongliang Cao, Tobias Weißberg, Zorah Lähner, Daniel Cremers, and Florian Bernard. Beyond Complete Shapes: A Benchmark for Quantitative Evaluation of 3D Shape Matching Algorithms. *CGF*, 2025. 7

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

[27] Viktoria Ehm, Paul Roetzer, Marvin Eisenberger, Maolin Gao, Florian Bernard, and Daniel Cremers. Geometrically consistent partial shape matching. In *3DV*, 2024. 3, 9

[28] Marvin Eisenberger, Zorah Lahner, and Daniel Cremers. Smooth shells: Multi-scale shape registration with functional maps. In *CVPR*, 2020. 2, 7, 8, 6

[29] James R Evans and John J Jarvis. Network topology and integral multicommodity flow problems. *Networks*, 1978. 6

[30] Danielle Ezuz and Mirela Ben-Chen. Deblurring and denoising of maps between shapes. In *CGF*, 2017. 2

[31] Danielle Ezuz, Behrend Heeren, Omri Azencot, Martin Rumpf, and Mirela Ben-Chen. Elastic correspondence between triangle meshes. In *CGF*, 2019. 2

[32] Pedro F Felzenszwalb. Representation and detection of deformable shapes. *TPAMI*, 2005. 2

[33] Giorgio Gallo, Giustino Longo, Stefano Pallottino, and Sang Nguyen. Directed hypergraphs and applications. *Discrete applied mathematics*, 1993. 2

[34] Maolin Gao, Paul Roetzer, Marvin Eisenberger, Zorah Lähner, Michael Moeller, Daniel Cremers, and Florian Bernard. Sigma: Scale-invariant global sparse shape matching. In *ICCV*, 2023. 1, 2

[35] Andrea Gasparetto, Luca Cosmo, Emanuele Rodola, Michael Bronstein, and Andrea Torsello. Spatial maps: From

low rank spectral to sparse spatial functional representations. In *3DV*, 2017. 2
[36] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *ECCV*, 2018. 2
[37] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. 6, 8, 5
[38] Oshri Halimi, Or Litany, Emanuele Rodola, Alex M Bronstein, and Ron Kimmel. Unsupervised learning of dense shape correspondence. In *CVPR*, 2019. 2
[39] 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 *ECCV*, 2022. 1
[40] Benjamin Holzschuh, Zorah Lähner, and Daniel Cremers. Simulated annealing for 3d shape correspondence. In *3DV*, 2020. 2
[41] Alec Jacobson, Daniele Panozzo, et al. libigl: A simple C++ geometry processing library, 2018. https://libigl.github.io/. 7
[42] Robert G Jeroslow, Kipp Martin, Ronald L Rardin, and Jinchang Wang. Gainfree leontief substitution flow problems. *Mathematical Programming*, 1992. 6
[43] Vladimir G Kim, Yaron Lipman, and Thomas Funkhouser. Blended intrinsic maps. *TOG*, 2011. 7
[44] Whoi-Yul Kim and Avinash C. Kak. 3-d object recognition using bipartite matching embedded in discrete relaxation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 1991. 1
[45] Eugene Krissinel and Kim Henrick. Secondary-structure matching (ssm), a new tool for fast protein structure alignment in three dimensions. *Biological Crystallography*, 2004. 1
[46] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 1955. 2
[47] Yam Kushinsky, Haggai Maron, Nadav Dym, and Yaron Lipman. Sinkhorn Algorithm for Lifted Assignment Problems. *SIAM Journal on Imaging Sciences*, 2019. 2
[48] 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. 2, 4, 1, 3
[49] Laura Leal-Taixé and Bodo Rosenhahn. Pedestrian interaction in tracking: the social force model and global optimization methods. In *Modeling, Simulation and Visual Analysis of Crowds: A Multidisciplinary Perspective*. 2013. 2
[50] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *ECCV*, 2024. 1
[51] Lei Li, Nicolas Donati, and Maks Ovsjanikov. Learning multi-resolution functional maps with spectral attention for robust shape matching. *NIPS*, 2022. 2
[52] Qinsong Li, Shengjun Liu, Ling Hu, and Xinru Liu. Shape correspondence using anisotropic chebyshev spectral cnns. In *CVPR*, 2020. 2
[53] Yang Li, Hikari Takehara, Takafumi Taketomi, Bo Zheng, and Matthias Nießner. 4dcomplete: Non-rigid motion estimation beyond the observable surface. In *ICCV*, 2021. 7
[54] Or Litany, Tal Remez, Emanuele Rodola, Alex Bronstein, and Michael Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *ICCV*, 2017. 2
[55] Jiayi Ma, Xingyu Jiang, Aoxiang Fan, Junjun Jiang, and Junchi Yan. Image matching from handcrafted to deep features: A survey. *IJCV*, 2021. 1
[56] Filippo Maggioli, Daniele Baieri, Emanuele Rodolà, and Simone Melzi. Rematching: Low-resolution representations for scalable shape correspondence. In *ECCV*, 2024. 2
[57] Robin Magnet, Jing Ren, Olga Sorkine-Hornung, and Maks Ovsjanikov. Smooth non-rigid shape matching via effective dirichlet energy optimization. In *3DV*, 2022. 7
[58] Haggai Maron, Nadav Dym, Itay Kezurer, Shahar Kovalsky, and Yaron Lipman. Point registration via efficient convex relaxation. *TOG*, 2016. 1, 2
[59] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. *TOG*, 2012. 2
[60] Maks Ovsjanikov, Quentin Mérigot, Facundo Mémoli, and Leonidas Guibas. One point isometric matching with the heat kernel. In *CGF*, 2010. 2
[61] Thomas Probst, Danda Pani Paudel, Ajad Chhatkuli, and Luc Van Gool. Unsupervised learning of consensus maximization for 3d vision problems. In *CVPR*, 2019. 2
[62] Jing Ren, Simone Melzi, Peter Wonka, and Maks Ovsjanikov. Discrete optimization for shape matching. In *CGF*, 2021. 2, 7, 8, 6
[63] Jing Ren, Adrien Poulenard, Peter Wonka, and Maks Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. *TOG*, 2018. 7
[64] F Rendl, P Pardalos, and H Wolkowicz. The Quadratic Assignment Problem: A Survey and Recent Developments. In *DIMACS workshop*, 1994. 1, 2
[65] Emanuele Rodola, Alex M Bronstein, Andrea Albarelli, Filippo Bergamasco, and Andrea Torsello. A game-theoretic approach to deformable shape matching. In *CVPR*, 2012. 2
[66] Paul Roetzer, Ahmed Abbas, Dongliang Cao, Florian Bernard, and Paul Swoboda. Discomatch: Fast discrete optimisation for geometrically consistent 3d shape matching. In *ECCV*, 2024. 3, 7
[67] Paul Roetzer and Florian Bernard. Spidermatch: 3d shape matching with global optimality and geometric consistency. In *CVPR*, 2024. 2, 3, 7, 8, 6
[68] Paul Roetzer, Zorah Lähner, and Florian Bernard. Conjugate product graphs for globally optimal 2d-3d shape matching. In *CVPR*, 2023. 2
[69] Paul Roetzer, Paul Swoboda, Daniel Cremers, and Florian Bernard. A scalable combinatorial solver for elastic geometrically consistent 3d shape matching. In *CVPR*, 2022. 3, 7
[70] David Sankoff and Joseph B Kruskal. *Time warps, string edits, and macromolecules: the theory and practice of sequence comparison*. 1983. 2
[71] Frank R. Schmidt, Eno Töppe, and Daniel Cremers. Efficient planar graph cuts with applications in computer vision. In *CVPR*, 2009. 2
[72] Patrick Schmidt, Marcel Campen, Janis Born, and Leif Kobbelt. Inter-surface maps via constant-curvature metrics. *TOG*, 39(4):119–1, 2020. 2
[73] Patrick Schmidt, Dörte Pieper, and Leif Kobbelt. Surface maps via adaptive triangulations. *CGF*, 2023. 2
[74] Thomas Schoenemann and Daniel Cremers. A combinatorial

solution for model-based image segmentation and real-time tracking. *TPAMI*, 2009. 2, 3

[75] John Schreiner, Arul Asirvatham, Emil Praun, and Hugues Hoppe. Inter-surface mapping. In *SIGGRAPH*. 2004. 2

[76] Alexander Schrijver et al. *Combinatorial optimization: polyhedra and efficiency*. Springer, 2003. 1

[77] Avinash Sharma, Radu Horaud, Jan Cech, and Edmond Boyer. Topologically-robust 3d shape matching based on diffusion geometry and seed growing. In *CVPR*, 2011. 2

[78] Nicholas Sharp, Souhaib Attaiki, Keenan Crane, and Maks Ovsjanikov. Diffusionnet: Discretization agnostic learning on surfaces. *TOG*, 2022. 2

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

[80] Y Soun and Klaus Truemper. Single commodity representation of multicommodity networks. *SIAM Journal on Algebraic Discrete Methods*, 1980. 6

[81] Kenshi Takayama. Compatible intrinsic triangulations. *TOG*, 2022. 2

[82] 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

[83] Giovanni Trappolini, Luca Cosmo, Luca Moschella, Riccardo Marin, Simone Melzi, and Emanuele Rodolà. Shape registration in the time of transformers. *NeurIPS*, 2021. 2

[84] Klaus Truemper. Unimodular matrices of flow problems with additional constraints. *Networks*, 1977. 6

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

[86] Matthias Van Onsem, Dieter De Paepe, S Vanden Hautte, Pieter Bonte, Veerle Ledoux, Annelies Lejon, Femke Ongenae, Dennis Dreesen, and Sofie Van Hoecke. Hierarchical pattern matching for anomaly detection in time series. *Computer Communications*, 2022. 1

[87] Matthias Vestner, Roee Litman, Emanuele Rodola, Alex Bronstein, and Daniel Cremers. Product manifold filter: Non-rigid shape correspondence via kernel density estimation in the product space. In *CVPR*, 2017. 2

[88] Ruben Wiersma, Elmar Eisemann, and Klaus Hildebrandt. Cnns on surfaces using rotation-equivariant features. *TOG*, 2020. 2

[89] 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, 7, 6

[90] 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

[91] Junchi Yan, Xu-Cheng Yin, Weiyao Lin, Cheng Deng, Hongyuan Zha, and Xiaokang Yang. A short survey of recent advances in graph matching. In *ICMR*, 2016. 1

[92] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *CVPR*, 2017. 7