Fast Globally Optimal and Geometrically Consistent 3D Shape Matching

Supplementary Material

A. More Details on Constraint Matrix H

In the following, we provide more details of submatrices P and L of our constraint matrix $H = \begin{bmatrix} P \\ L \\ S \end{bmatrix}$.

The matrix P. As mentioned in the main paper, $\mathcal{P}_i$ can be represented algebraically via matrix P_i. Yet, to account for degenerate edges (i.e. the extended edge set $\mathcal{E}_{\mathcal{Y}}^+$), we have to consider a slightly altered definition of P_i which reads

$$P_i \coloneqq C_i^+ \otimes \tilde{Y}^+ - C_i^- \otimes \tilde{Y}^-. \tag{5}$$

Here, C_i^+ and C_i^- are the non-negative and non-positive entries of the vertex edge incidence matrix $C_i \in \{-1,0,1\}^{|\mathcal{V}_{\mathcal{C}_i}|\times|\mathcal{E}_{\mathcal{C}_i}|}$. In contrast to the elaborations in the main paper, which, as mentioned, neglects degenerate edges, $\tilde{Y}^+$ and $\tilde{Y}^-$ are the non-negative and non-positive entries of the vertex edge incidence matrix $Y \in \{-1,0,1\}^{|\mathcal{V}_{\mathcal{Y}}|\times|\mathcal{E}_{\mathcal{Y}}|}$ concatenated column-wise with $\boldsymbol{I}_{|\mathcal{V}_{\mathcal{Y}}|}$ and $-\boldsymbol{I}_{|\mathcal{V}_{\mathcal{Y}}|}$, respectively. Thus $\tilde{Y}^+$ and $\tilde{Y}^-$ are defined as

$$\begin{aligned} \tilde{Y}^+ &\coloneqq \begin{bmatrix} Y^+ & \boldsymbol{I}_{|\mathcal{V}_{\mathcal{Y}}|} \end{bmatrix}, \\ \tilde{Y}^- &\coloneqq \begin{bmatrix} Y^- & -\boldsymbol{I}_{|\mathcal{V}_{\mathcal{Y}}|} \end{bmatrix}. \end{aligned} \tag{6}$$

With the interpretation that non-zero entries in matrices $\tilde{Y}^+$ and $\tilde{Y}^-$ resemble incoming and outgoing edges at vertices of $\mathcal{Y}$, one can see that appending identities to respective matrices can be interpreted as adding self-edges, i.e. accounting for $\mathcal{E}_{\mathcal{Y}}^+$, see also Fig. A.1 (i).

The matrix L. As mentioned in the main paper, the definition of P requires columns of the non-positive blocks of L to be permuted. We can incorporate this permutation by considering the following alternative definition of L which reads

$$L \coloneqq K^+ \otimes \boldsymbol{I}_{|\mathcal{E}_{\mathcal{Y}}^+|} - K^- \otimes \tilde{\boldsymbol{I}}_{|\mathcal{E}_{\mathcal{Y}}^+|}. \tag{7}$$

Here K^+ and K^- are the non-negative and non-positive entries of the matrix $K \in \{-1,0,1\}^{p\times|\mathcal{E}_{\mathcal{X}}|}$ (p is the number of undirected non-boundary edges of $\mathcal{X}$ and K represents the incidence of opposite edges across all n surface cycles). Furthermore, $\tilde{\boldsymbol{I}}_{|\mathcal{E}_{\mathcal{Y}}^+|}$ is a (column) permuted identity matrix with all non-positive entries. The column permutation of $\tilde{\boldsymbol{I}}_{|\mathcal{E}_{\mathcal{Y}}^+|}$ is such that opposite edges in $\mathcal{E}_{\mathcal{Y}}^+$ can be mapped to each other via $\tilde{\boldsymbol{I}}_{|\mathcal{E}_{\mathcal{Y}}^+|}$, i.e. such that $\tilde{Y}^+ = \tilde{Y}^-\tilde{\boldsymbol{I}}_{|\mathcal{E}_{\mathcal{Y}}^+|}$ and such that $\tilde{Y}^- = \tilde{Y}^+\tilde{\boldsymbol{I}}_{|\mathcal{E}_{\mathcal{Y}}^+|}$.

B. Detailed Hyper Product Graph Formalism

In this section we provide detailed definitions of the hyper product graph $\mathcal{H}$ which arises from our constraint matrix H. We provide an overview of the major concepts in Fig. A.1, where we also visualise the matrix structures of respective submatrices P, L, and S of H.

B.1. Individual Surface Cycle Matching Subproblems

Each surface cycle $\mathcal{C}_i$ can be matched to shape $\mathcal{Y}$ by finding a cyclic path in the *product graph*, which has been introduced in [48] for matching a 2D to a 3D shape, see Definition 7.

In a nutshell, each vertex $v = \binom{x}{y} \in \mathcal{V}_{\mathcal{P}_i}$ in the product graph resembles a (potential) matching between vertices $x \in \mathcal{V}_{\mathcal{X}}$ and $y \in \mathcal{V}_{\mathcal{Y}}$ of both shapes. Thus, each edge $(v, \bar{v})$ in the product graph can be interpreted as a (potential) matching between edges of $\mathcal{C}_i$ and edges of shape $\mathcal{Y}$ (or vertices of $\mathcal{Y}$ via the extended edge set $\mathcal{E}_{\mathcal{Y}}^+$ to account for stretching and compression). With that, matching surface cycle $\mathcal{C}_i$ to shape $\mathcal{Y}$ amounts to solving a cyclic shortest path problem in the respective product graph $\mathcal{P}_i$ [48].

However, a major downside for our setting of *3D-to-3D shape matching* is that the neighbourhood between pairs of surface cycles cannot be ensured when considering vanilla shortest path algorithms (that solve the n individual surface cycle matching subproblems independently). To tackle this, we couple the individual product graphs $\mathcal{P}_1, \dots, \mathcal{P}_n$ appropriately, which we explain next.

B.2. Subproblem Coupling

We couple the individual surface cycle matching subproblems of $\mathcal{P}_i$ by glueing them together via opposite edges. To this end, we introduce *coupling vertices*:

Definition 9 (Coupling vertices)**.** *For every pair of opposite product edges $e \in \mathcal{E}_{\mathcal{P}_i}$ and $-e \in \mathcal{E}_{\mathcal{P}_j}$ we add a coupling vertex q_{ij}^e. The set of coupling vertices is $\mathcal{V}_{\mathcal{L}} \coloneqq \{q_{ij}^e \mid e \in \mathcal{E}_{\mathcal{P}_i}, -e \in \mathcal{E}_{\mathcal{P}_j}\}$.*

The purpose of coupling vertices is that matchings of opposite edges are consistent, i.e. so that matchings of surface cycles cover opposite edges and thus are neighbouring, see Fig. A.1 (iii). Consequently, this de-facto enforces the *glueing*, which in turn results in global geometric consistency.

In addition to the coupling, we want to ensure matching injectivity for each edge of surface cycles. Hence, we need to ensure that each such edge is matched exactly once, which we tackle next.

B.3. Surface Cycle Matching Injectivity

We want to enforce that each surface cycle edge is matched exactly once, i.e. it is matched to exactly one edge (or vertex) of $\mathcal{Y}$. In other words, among all the potential matching candidates of a single surface cycle edge, exactly one is

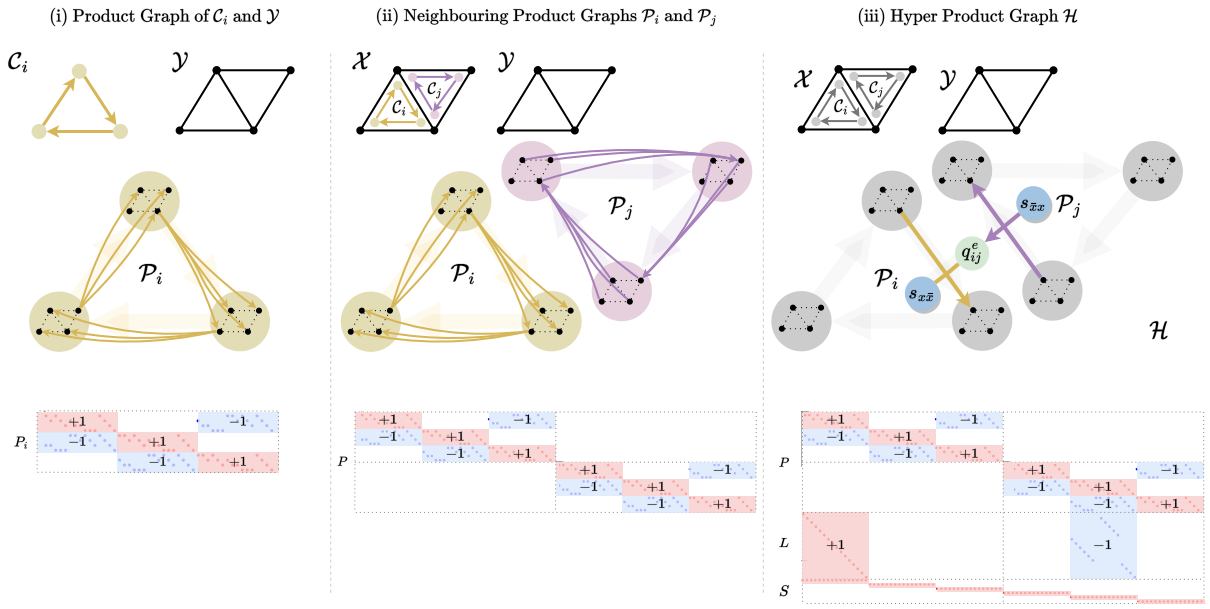

Figure A.1. Visualisation of the involved **graph structures** (top) and respective **vertex-edge incidence matrices** (bottom): (i) for an individual subproblem product graph $\mathcal{P}_i$, (ii) for two (uncoupled) subproblem product graphs $\mathcal{P}_i, \mathcal{P}_j$, and (iii) for two coupled subproblems leading to the hyper product graph $\mathcal{H}$. We note that in (iii) we illustrate two *directed hyper edges* (which have multiple source and target vertices, see thick coloured lines ▬,▬) of our hyper product graph $\mathcal{H}$. These hyper edges of $\mathcal{H}$ contain source and target vertices of product edges (see middle) in their respective sets of source and target vertices. Furthermore, each hyper edge additionally contains a vertex $q^e_{ij} \in \mathcal{V}_\mathcal{L}$ in its set of source and target vertices (so that opposite product edges are coupled, see green dot ●). Finally, each hyper edge contains an injectivity vertex $s_{x\bar{x}}, s_{\bar{x}x} \in \mathcal{V}_\mathcal{S}$ in its set of source vertices (so that each edge of the surface cycles is matched exactly once, see blue dots ●).

part of the final matching. 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 vertex (see Fig. A.1 (iii) as well as Fig. 4), which has the purpose to ensure that only a single edge of each edge bundle is part of the final matching:

Definition 10 (Injectivity vertices)**.** *For every directed edge* $(x, \bar{x}) \in \mathcal{E}_\mathcal{X}$ *of shape* $\mathcal{X}$ *we introduce one injectivity vertex. The set of injectivity vertices is* $\mathcal{V}_\mathcal{S} := \{s_{x\bar{x}} \mid (x, \bar{x}) \in \mathcal{E}_\mathcal{X}\}$.

B.4. Resulting Hyper Product Graph

Finally, we present our directed hyper product graph $\mathcal{H}$. We note that in contrast to ordinary edges (which have exactly one source and one target vertex), a *directed hyper edge* denoted as $[\![v_1, v_2, \ldots \twoheadrightarrow v_3, v_4, \ldots]\!]$ has a *set* of source vertices $\{v_1, v_2, \ldots\}$ and a *set* of target vertices $\{v_3, v_4, \ldots\}$ [33].

Our hyper graph has two different types of hyper edges, *coupled* and *uncoupled* ones. The *coupled hyper edges* refer to hyper edges that belong to *non-boundary* edges of $\mathcal{X}$ and serve the purpose of enforcing neighbourhood preservation between surface cycles:

Definition 11 (Coupled hyper edges)**.** *The set of* coupled hyper edges *is defined as*

$$\tilde{\mathcal{E}}_\mathcal{H} = \Big\{ [\![v, s_{x\bar{x}} \twoheadrightarrow \bar{v}, q^e_{ij}]\!], [\![\bar{v}, s_{\bar{x}x}, q^e_{ij} \twoheadrightarrow v]\!] \;\Big|\; s_{x\bar{x}}, s_{\bar{x}x} \in \mathcal{V}_\mathcal{S}, \; q^e_{ij} \in \mathcal{V}_\mathcal{L}, \; e := (v, \bar{v}) = \left(\binom{x}{y}, \binom{\bar{x}}{\bar{y}}\right) \in \mathcal{E}_{\mathcal{P}_i}, -e \in \mathcal{E}_{\mathcal{P}_j} \Big\}. \tag{8}$$

From above definition we can see that one coupling vertex always connects exactly two hyper edges. The *uncoupled hyper edges* refer to hyper edges that belong to boundary edges of $\mathcal{X}$ and with which we can account for partiality of the source shape $\mathcal{X}$:

Definition 12 (Uncoupled hyper edges)**.** *The set of* uncoupled hyper edges *is defined as*

$$\hat{\mathcal{E}}_\mathcal{H} = \Big\{ [\![v, s_{x\bar{x}} \twoheadrightarrow \bar{v}]\!] \mid v = \binom{x}{y}, \; \bar{v} = \binom{\bar{x}}{\bar{y}}, \; s_{x\bar{x}} \in \mathcal{V}_\mathcal{S}, \; (x, \bar{x}) \in \mathcal{E}_\mathcal{X} \text{ is boundary edge}, \; (y, \bar{y}) \in \mathcal{E}^+_\mathcal{Y} \Big\}. \tag{9}$$

This gives rise to our hyper product graph, which we also visualise in Fig. A.1 (iii):

Definition 13 (Hyper product graph)**.** *Our hyper product graph* $\mathcal{H} = (\mathcal{V}_\mathcal{H}, \mathcal{E}_\mathcal{H})$ *for matching the source shape* $\mathcal{X}$

(represented with n surface cycles) to the target shape $\mathcal{Y}$ comprises the vertex set $\mathcal{V}_\mathcal{H}$ and the set of (directed) hyper edges $\mathcal{E}_\mathcal{H}$, and is defined as

$$\begin{aligned} \mathcal{V}_\mathcal{H} &= \mathcal{V}_{\mathcal{P}_1} \cup \ldots \cup \mathcal{V}_{\mathcal{P}_n} \cup \mathcal{V}_\mathcal{S} \cup \mathcal{V}_\mathcal{L}, \\ \mathcal{E}_\mathcal{H} &= \tilde{\mathcal{E}}_\mathcal{H} \cup \hat{\mathcal{E}}_\mathcal{H} \quad \textit{with} \quad m := |\mathcal{E}_\mathcal{H}|. \end{aligned} \tag{HPG}$$

C. Practical Considerations

In this section, we discuss the implementation of the distortion bound as well as the problem size reduction of our linear program (GeCo3D). Furthermore, we discuss different choices of surface cycles.

C.1. Distortion Bound

Our hyper product graph $\mathcal{H}$ already allows to map edges of $\mathcal{X}$ to vertices and edges of $\mathcal{Y}$. Thus, our formalism already accounts for shrinking and stretching. Yet, for more flexibility, we want to additionally allow for matchings of edges of $\mathcal{Y}$ to vertices of $\mathcal{X}$. To this end, we integrate a distortion bound by creating k duplicates of vertices of every product graph $\mathcal{P}_i$. We connect the duplicates such that resulting product edges resemble edge to edge, edge to vertex or vertex to edge matchings between shapes $\mathcal{X}$ and $\mathcal{Y}$, see also the product graph definition in [48] and Fig. A.2 for a visualisation of additional product edges. This effectively allows for at most k consecutive edges of $\mathcal{Y}$ to be matched to a vertex of shape $\mathcal{X}$ (in addition to the already allowed matchings between edges of $\mathcal{X}$ and edges or vertices of $\mathcal{Y}$). In all experiments we set $k = 2$. We note that similar concepts have been used for image segmentation, see [74, Section 7.1.2].

C.2. Reduced Problem Size

For improved solver runtimes we reduce the problem size of (GeCo3D) by approximately 50%. To this end, we resolve coupling constraints during construction of matrix H (these effectively ensure that two variables hold the same value during optimisation and thus we only need to consider approximately 50% of the variables). In Fig. A.3 we show an example of full-sized matrix H for the problem of matching two tetrahedron and furthermore, in Fig. A.4, we visualise the resulting smaller matrix for the same problem of matching two tetrahedron. We emphasise that problem size reduction, as described above, does *not* prune the problem but rather describes (GeCo3D) with a smaller but equivalent optimisation problem. In other words, a solution to the reduced problem yields a uniquely defined solution to the larger problem.

C.3. Other Choices of Surface Cycles

As mentioned in the main paper, we assume that each surface cycle represents a single triangle of $\mathcal{X}$. We note that a single surface cycle can represent the boundary of polygonal patches of shape $\mathcal{X}$ that contain multiple triangles. Yet, in this case, the union of vertices of all surface cycles might not contain all vertices of shape $\mathcal{X}$, so that one would have to adjust the notion of geometric consistency in Definition 4 accordingly.

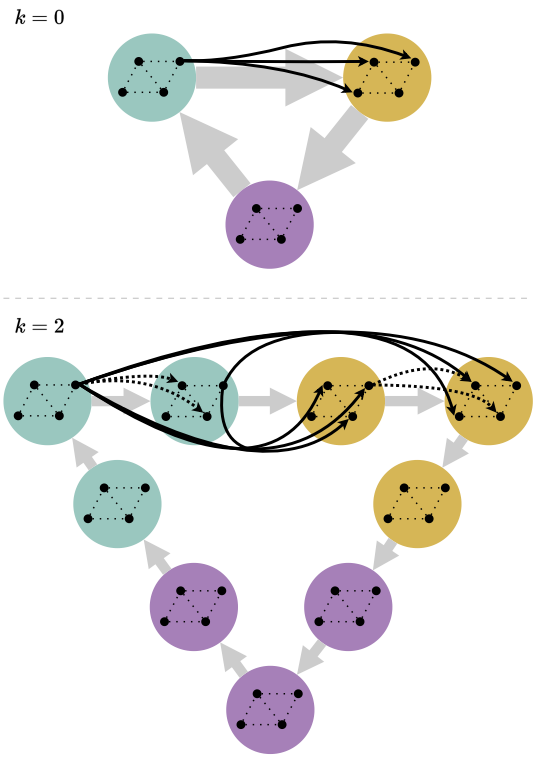

Figure A.2. Visualisation of a product graph with distortion bound $k = 0$ (**top**) and with distortion bound $k = 2$ (**bottom**). The distortion bound $k \in 2\mathbb{Z}_0^+$ is achieved by duplicating product vertices k times (duplicates are visible when comparing top and bottom graphs). We connect duplicated product vertices such that: (i) all duplicates of the same group of product vertices (e.g. all vertices within greenish circles ●) are connected (see dashed black arrows resembling potential matchings of edges of $\mathcal{Y}$ to vertices of $\mathcal{X}$) and (ii) $k-1$ duplicates as well as the original vertices of the same group are connected to $k-1$ duplicates as well as the original vertex of the subsequent group (see solid black arrows at bottom figure which connect vertices in greenish circles ● with vertices in yellow circles ● and which resemble potential matchings of edges of $\mathcal{X}$ and edges or vertices of $\mathcal{Y}$). We note that we only allow for distortion-bounds of integer multiples of 2 (so that neighbouring product graphs can still be coupled).

D. Additional Experiments

In this section, we provide additional results for shape matching and planar graph matching.

D.1. Shape Matching

Ablation Studies. In Tab. 5, we conduct ablation studies of our method. To this end, we use 10 pairs from FAUST dataset, decimate shapes to 500 triangles and evaluate mean

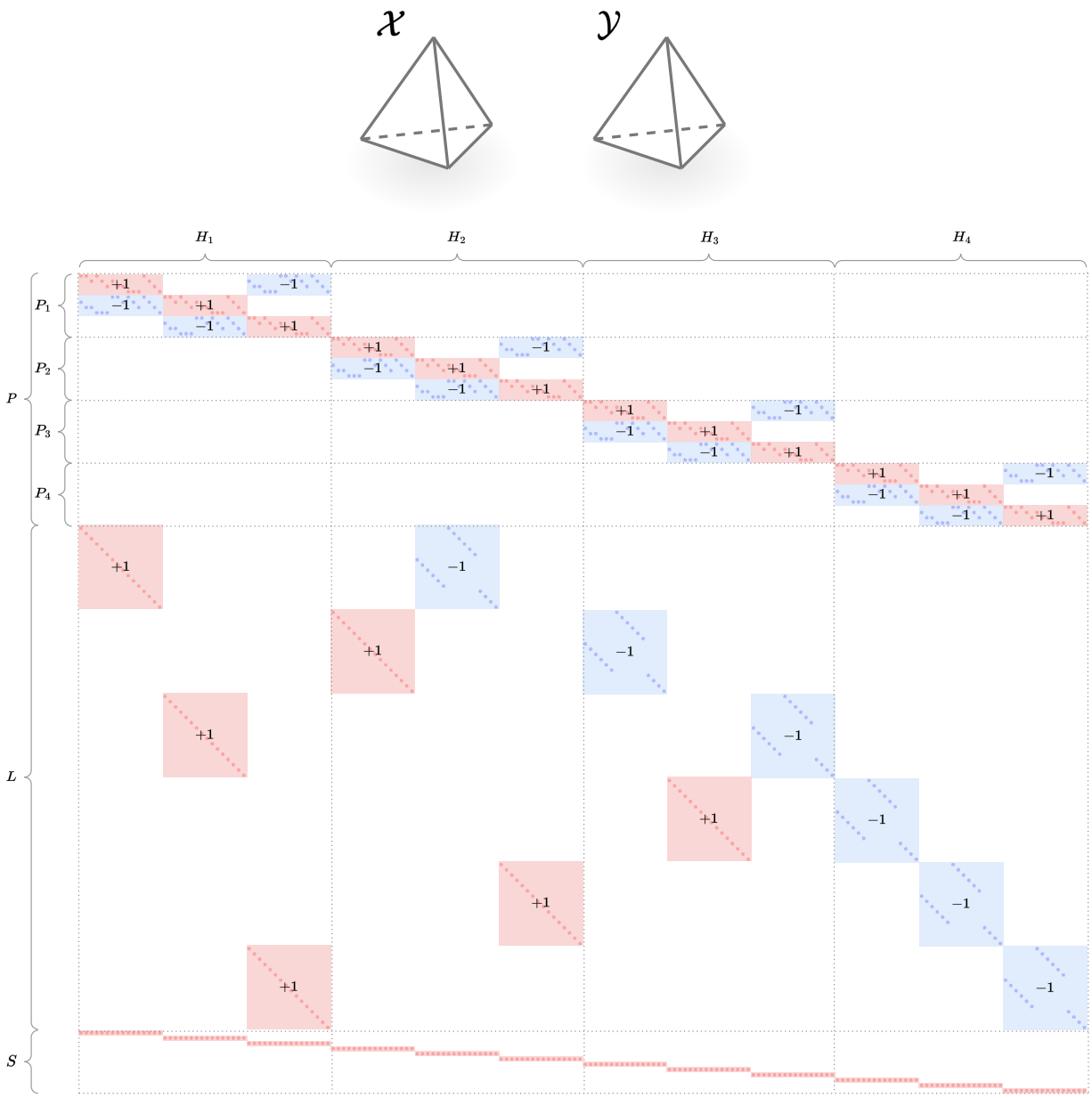

Figure A.3. For the matching of two tetrahedron shapes (**top**) we show the resulting H matrix (**bottom**). In light red and light blue we indicate the blocks of H, where within each block there is exactly one non-zero element per column.

geodesic errors in different settings. We consider our formalism (GeCo3D) with and without coupling constraints (i.e. without coupling constraints means that we only consider constraints $Px = 0$ and $Sx = 1$ and drop the constraints $Lx = 0$). Furthermore, we use different methods to compute features: wave kernel signatures (WKS) [3], features based upon image foundation models which we call Diff3F [22] (we note that we use an empty text prompt for feature extraction with Diff3F) and features computed with deep feature extractor ULRSSM [13]. For all types of features, our coupling constraints help to improve results. This shows the importance of priors induced by global geometric consistency. Overall, we obtain best results with features computed with ULRSSM [13] (which is the feature extractor that we use for our method for the rest of our shape matching experiments). Additionally, we evaluate our methods performance to shapes with different discretisation. We fix one shape to 1000 triangles, vary the

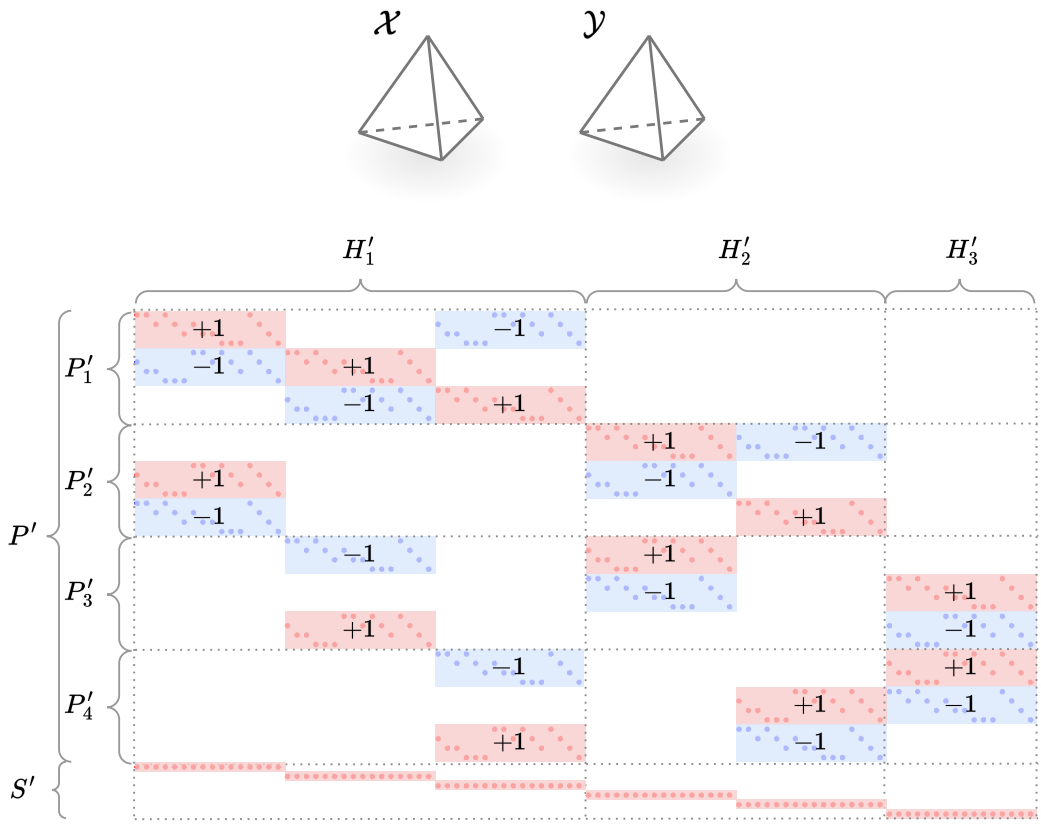

Figure A.4. We show the **reduced matrix structure** after resolving coupling constraints for the same matching problem of matching two tetrahedron as shown in Fig. A.3.

Method	Mean Geodesic Error
Ours (no coupling) + WKS [3]	0.2335
Ours + WKS [3]	0.2253
Ours (no coupling) + Diff3F [22]	0.1105
Ours + Diff3F [22]	0.0511
Ours (no coupling) + ULRSSM [13]	0.0318
Ours + ULRSSM [13]	**0.0316**

Table 5. **Ablation studies** conducted on ten pairs of FAUST dataset. We consider different methods to compute features and furthermore consider our method with and without coupling constraints. Using coupling constraints improves results for all types of features. Using features computed with ULRSSM [13] yields best results overall.

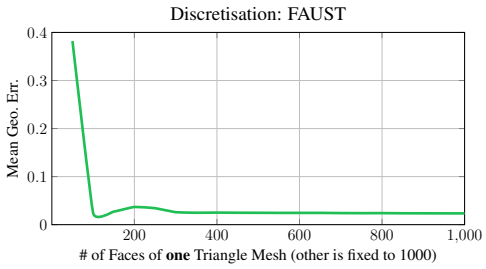

Figure A.5. Evaluation of the performance of our method under varying **discretisation**. We fix the resolution of one shape to 1000 triangles and vary the resolution of the other shape.

resolution of the other shape and plot mean geodesic errors in Fig. A.5 of five instances from FAUST. We can see that our method performs reasonably under varying shape discretisation.

Memory Footprint of Constraints. In Tab. 6, we compare the memory footprint of our method to other geometrically consistent approaches. Our has the lowest memory requirements.

Runtime. In Fig. A.6, we compare runtimes of off-the-shelf linear programming solvers Gurobi [37] and Mosek [1]. Plotted curves are median runtimes of five pairs of FAUST dataset. We can see that Mosek [1] scales poor compared to Gurobi [37].

Full Shape Matching. We show error plots of geodesic errors in Fig. A.8 and Dirichlet energy plots in Fig. A.9. Furthermore, we show additional qualitative results, includ-

# Faces	Windheuser et al.	SpiderMatch	Ours
500	0.2 GB	0.2 GB	**0.1** GB
1000	0.8 GB	1.0 GB	**0.3** GB
1500	1.8 GB	2.2 GB	**0.7** GB
2000	3.1 GB	4.0 GB	**1.2** GB

Table 6. Comparison of the **memory footprint** of the constraint matrices of Windheuser et al. [89], SpiderMatch [67] and our approach.

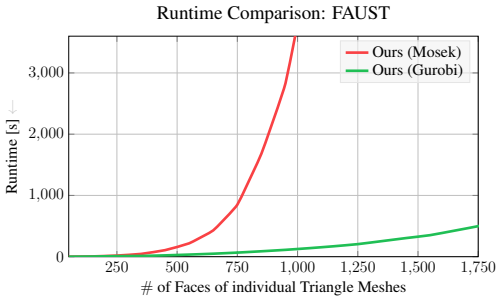

Figure A.6. **Runtime comparison** of off-the-shelf linear programming solvers Gurobi [37] and Mosek [1] when solving linear program (GeCo3D) with varying resolution of input shapes. Curves are median runtimes over five pairs from FAUST dataset.

Metric	ULRSSM	SpiderMatch	Ours
Mean geodesic errors (↓)	0.071	0.066	**0.062**
Mean Dirichlet energies (↓)	1.262	0.613	**0.444**

Table 7. Quantitative results for **partial-to-full** shape matching on the SHREC'16 cuts [17] dataset for ULRSSM [13], SpiderMatch [67] and our method.

ing results for methods by Ren *et al*. [62] and Eisenberger *et al*. [28] in Fig. A.7.

Partial-to-Full Shape Matching. In Fig. A.10, we show more qualitative results computed with our method for the partial-to-full shape matching setting and furthermore in Tab. 7 we report quantitive results on SHREC'16 [17].

D.2. Planar Graph Matching

In Fig. A.11, we show more graph matching examples of instances of WILLOW [16] dataset. We furthermore note that we only consider the graph matching problem as a proof-of-concept experiment and that we use pixel-coordinates of vertices of respective graphs as input features. We leave the integration of more elaborate features, when our method is applied to graph matching, to future works.

Figure A.7. Additional **qualitative shape matching results** on datasets DT4D-H (columns ①-⑤), FAUST (columns ⑥-⑨), and SMAL (columns ⑩-⑫). Columns ①, ③, ⑤, ⑥, ⑨, ⑩ are also shown in the main paper and are repeated here for comparison with methods DiscrOpt [62] and SmoothShells [28]. All matchings are visualised by transfering colour and triangulation from source to target shape. In column ⑫, we can see a failure mode of ours very likely caused by the fact that our solution space still contains matchings which allow for inside out flips (one side of the leg is matched to the other side of the leg).

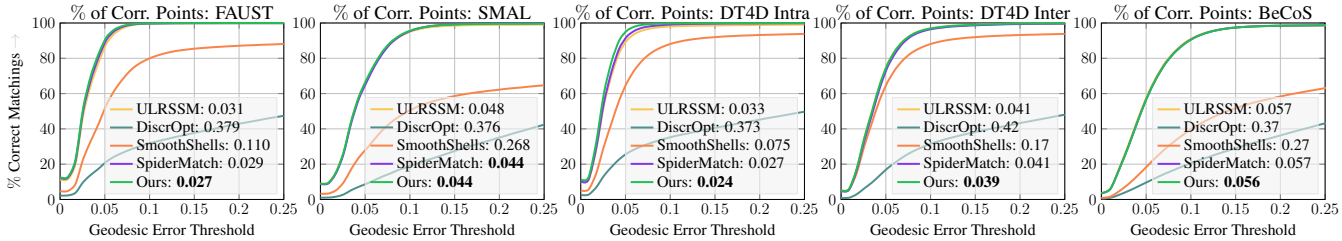

Figure A.8. **Quantitative shape matching results** on datasets FAUST, SMAL, DT4D Intra and Inter and BeCoS. We show percentage of correct points (↑) w.r.t. geodesic error thresholds (i.e. this quantifies if a matched point is within a geodesic threshold radius around ground truth point). Numbers in legends are mean geodesic errors (↓). Across all five datasets our method performs the best, very likely due to enforced geometric consistency.

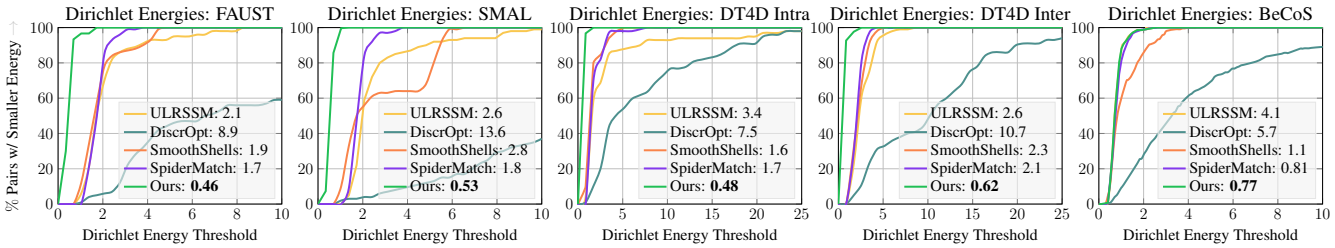

Figure A.9. We show percentage of matched points (↑) which are below a certain **Dirichlet Energy threshold** on datasets FAUST, SMAL, DT4D and BeCoS. Numbers in legends are mean Dirichlet energies across all pairs (↓). Our method consistently yields best results on all datasets very likely since it is the only method enforcing *global* geometric consistency.

Figure A.10. More qualitative results computed with our method for **partial-to-full** shape matching on test set shapes [27] from SHREC'16 [17].

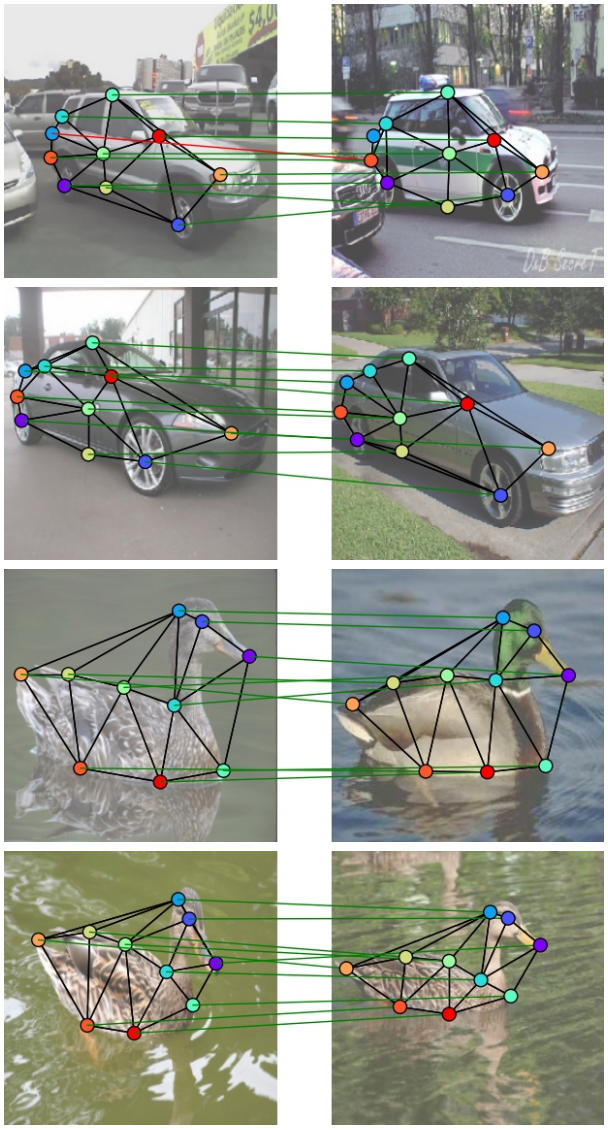

Figure A.11. More **planar graph matching** results using our method. For the cars in the top row we can see matching artefacts stemming from graph connectivity differences and enforced geometric consistency (see vertices connected with red line).