UNLOCKING GEODESIC GROMOV-WASSERSTEIN DISTANCES FOR 3D MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Gromov-Wasserstein Distances (GWDs) provide quantitative ways of comparing probabilistic distributions defined on different metric spaces by applying techniques from the optimal transport theory. As such, GWD can be potentially useful in a large variety of applications ranging from graph matching problems to 3D object detection. However its practical use at scale is significantly limited by cubic time complexity computations involving dense intra-space distance matrices. Even though in the Euclidean metric spaces several techniques (e.g. involving scalable kernel methods) were proposed to address it, to the best of our knowledge, analogous techniques for general geodesic distances on manifolds, or shortest-path distance on graphs in their discretized variants, were not developed. In this paper, we present **E**fficient **G**eodesic **Gro**mov-**W**asserstein methods (EGGroW), a new class of efficient algorithms designed to calculate geodesic Gromov-Wasserstein distances with entropic Sinkhorn-like approaches, leveraging recently introduced *GenusSink* methods (Choromanski et al., 2026) and the theory of random features. We provide important downstream applications, namely: 3D pose estimation and 3D template detection. In the latter setting, we formulate a partial 3D template recovery as a staged problem: capacity-constrained scene selection is followed by semi-relaxed recovery of template visibility and correspondence. Our empirical findings show that EGGroW provides accurate solutions when standard Euclidean-based techniques fail and is characterized by light computational footprint, as our theoretical analysis predicts.

1 INTRODUCTION & RELATED WORK

Wasserstein distance (Pereira & Amini, 2025; Montesuma et al., 2024; Peyré & Cuturi, 2019; Peyré, 2025; Amari et al., 2019; Genevay et al., 2016; Figalli, 2010; Ollivier et al., 2014; Ambrosio et al., 2021; Villani, 2003) is a rigorous and one of the most commonly used in machine learning approach to measuring dissimilarity between two probabilistic measures: μ and ν, defined in a given metric space $\mathcal{X}$. For $p \in [1, +\infty]$, the *Wasserstein p-distance* between μ and ν with finite p-moments is given by the following formula, where $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is the corresponding distance function:

$$W_p(\mu, \nu) \inf_{\gamma \in \Gamma(\mu,\nu)} (\mathbb{E}_{x,y\sim\gamma} d(x, y)^p)^{\frac{1}{p}} . \tag{1}$$

Here $\Gamma(\mu, \nu)$ stands for the set of all couplings of μ and ν. We define a *coupling* as a joint probablity measure on $\mathcal{X} \times \mathcal{X}$ whose marginals on the first and second factor are μ and ν respectively, i.e. the following holds for every measurable $\mathcal{A}$:

$$\gamma(\mathcal{A} \times \mathcal{X}) = \mu(\mathcal{A}), \text{ and } \gamma(\mathcal{X} \times \mathcal{A}) = \nu(\mathcal{A}). \tag{2}$$

In most practical applications of the Optimal Transport, its discretized version is considered, with μ and ν supported by m and n points respectively. In this setting, each solution (transport plan) is described by the *transport matrix* $\mathbf{T} \in \mathbb{R}^{m\times n}$. The costs of moving mass between points is encoded by the *cost matrix* $\mathbf{C} \in \mathbb{R}^{m\times n}$, where $C_{i,j}$ is the distance between ith atom of μ and jth atom of ν. The optimization problem takes the following form for $\mathbf{a} \in \mathbb{R}^m_{\geq 0}, \mathbf{b} \in \mathbb{R}^n_{\geq 0}$ encoding μ and ν ($\sum_{i=1}^m a_i = 1$, $\sum_{j=1}^n b_j = 1$) and furthermore, $\odot$ is a *Hadamard* (element-wise) product:

$$\min_{\mathbf{T}\in\mathbb{R}^{m\times n}} \mathbf{T} \odot \mathbf{C} \text{ st. } \mathbf{T}\mathbf{1}_n = \mathbf{a},\ \mathbf{T}^\top \mathbf{1}_m = \mathbf{b},\ \mathbf{T} \geq 0. \tag{3}$$

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Matrix $\mathbf{T}^* \in \mathbb{R}^{m \times n}$ takes quadratic space $O(nm)$. Standard linear-programming algorithms, used to solve Optimal Transport problem from Eq. 3, are of cubic time complexity. One of the most popular relaxations of the problem, providing significant computational speedups, is the so-called *Sinkhorn* (entropic) variation (Cuturi, 2013; Moradi, 2025; Watanabe & Isobe, 2024). It adds an entropic regularization term, so that the optimization problem becomes:

$$\min_{\mathbf{T} \in \mathbb{R}^{n \times n}} \mathbf{T} \odot \mathbf{C} + \epsilon \sum_{i,j} T_{i,j}(\log(T_{i,j}) - 1). \tag{4}$$

Sinkhorn version can be solved in time $O(nm)$ by the following iterative procedure (with the number of iterations treated as a constant), with *kernel matrix* $\mathbf{K}_\epsilon = \exp(-\frac{\mathbf{C}}{\epsilon})$ (exp and vector-divisions applied element-wise):

$$\mathbf{v}^{\text{init}} = \mathbf{1}_n, \mathbf{u} \leftarrow \frac{\mathbf{a}}{\mathbf{K}_\epsilon \mathbf{v}}, \mathbf{v} = \frac{\mathbf{b}}{(\mathbf{K}_\epsilon)^\top \mathbf{u}} \tag{5}$$

Thus Sinkhorn approach makes the Optimal Transport Problem much more computationally tractable, especially for probabilistic distributions supported on massive size discrete sets.

Gromov-Wasserstein Distance (GWD: see Sec. 2.1 for exact definition) (Mémoli, 2011; Peyré et al., 2016; Xu et al., 2019; Vayer et al., 2020) is a natural extension of the OT formulation to the scenario, where the measures μ and ν are defined on two different metric spaces $\mathcal{X}$ and $\mathcal{Y}$. Importantly $\mathcal{X}$ and $\mathcal{Y}$ do not need to be related to each other in any way, since the new formulation does not rely on any notion of inter-distances between the two spaces. Instead, it evaluates the soft coupling provided by the transport matrix $\mathbf{T}$ by comparing intra-space relationships between the points supporting measures. This feature makes it a potentially very powerful tool in a variety of ML applications, ranging from graph matching problems to 3D object detection.

However practical use at scale of the GWD techniques is significantly limited by cubic time complexity computations involving dense intra-space distance matrices (Chowdhury et al., 2021). Even though in certain settings including costs given via Euclidean distances (e.g. squared Euclidean distances, see: (Scetbon et al., 2022)), several efficient methods were proposed (e.g. involving scalable kernel methods), to the best of our knowledge, analogous analogous techniques for general geodesic distances on manifolds, or shortest-path distance on graphs in their discretized variants, were not developed. This more general setting, the key to unlocking GWD for 3D modeling (e.g. with meshes and point clouds, where distances are often defined via shortest paths in the related graphs), is the subject of this paper.

One of the biggest challenges in the geodesic setting is that costs matrices $\mathbf{C}$ seem to be much less structured, at first glance prohibiting computationally efficient operations on them (or their derivatives, such as kernel matrices) (Parashar et al., 2026). The recently proposed GenusSink algorithm (Choromanski et al., 2026) addresses this question in the regular Optimal Transport setting, providing efficient methods for matrix-vector multiplications involving kernel matrices in the geodesic setting. The only assumption (satisfied in most practical applications) is that graphs under considerations need to have bounded genus (Kotrbcík & Skoviera, 2012).

In this paper we significantly extend those techniques to computations of the Gromov-Wasserstein Distance. We present **E**fficient **G**eodesic **Gro**mov-**W**asserstein methods (EGGroW), a new class of efficient algorithms designed to calculate geodesic Gromov-Wasserstein distances with entropic approaches, leveraging the aforementioned *GenusSink* methods and the theory of random features. We provide important downstream applications, namely: 3D pose estimation and 3D template detection. Another contribution of this paper is the formulation of the 3D template detection in the given point cloud scene as a derivative of the original GWD problem. To be more specific, we formulate partial 3D template recovery as a staged problem: capacity-constrained scene selection is followed by semi-relaxed recovery of template visibility and correspondence with the use of GWDs. Our empirical findings show that EGGroW provides accurate solutions when standard Euclidean-based techniques fail and is characterized by light computational footprint, as our theoretical analysis predicts.

Two branches that EGGroW relies on: the theory of GenusSink and random features techniques, are complementary. The former gives an exact algorithm for graphs with separators of size $O(\log \log n)$ (satisfied in most practical applications), having a near-exact extension for more general bounded genus graphs. It provides favorable near-linear matrix-vector computation time with kernel matrices, particularly relevant for processing massive input graphs. The latter gives additional computa-

tional gains (at the cost of the accuracy loss), even for medium-size input graphs and is particularly straightforward to implement, as it does not involve any combinatorial operations.

2 PRELIMINARIES

2.1 PROBLEM SETTING

Consider two metric spaces: $\mathcal{X}$ and $\mathcal{Y}$, with the corresponding probabilistic measures μ and ν defined on them. We will assume that $\mathcal{X}$ and $\mathcal{Y}$ are discretized, with the corresponding sets of points $X = \{x_i\}_{i=1}^m$ and $Y = \{y_j\}_{j=1}^n$ respectively for some $m, n > 0$. We encode the discretized versions of μ and ν as: $\mathbf{a} \in \Delta_m$ and $\mathbf{b} \in \Delta_n$ ($\Delta_k \stackrel{\text{def}}{=} \{(z_1, ..., z_k) : z_i \geq 0; \sum_{i=1}^k z_i = 1\}$). The internal discretized geometries of $\mathcal{X}$ and $\mathcal{Y}$ are represented by cost matrices $\mathbf{C}_X \in \mathbb{R}^{n \times n}$ and $\mathbf{C}_Y \in \mathbb{R}^{m \times m}$. In this paper, we are interested in the setting, where $C_Z(i,j) \stackrel{\text{def}}{=} d_{G_Z}(i,j)$ ($Z \in \{X, Y\}$) for $d_G(i,j)$ denoting the length of the shortest path distance between vertices i and j in graph G. Here G_Z is a graph with the set of vertices Z and weights of edges defined as distances in the underlying metric spaces. A coupling $\mathbf{T} \in \mathbb{R}_{\geq 0}^{m \times n}$ is a soft, generally many-to-many correspondence between the two distributions $\mathbf{a}$ and $\mathbf{b}$, i.e.:

$$\sum_{j=1}^{n} T_{i,j} = a_i, \ \sum_{i=1}^{m} T_{i,j} = b_j \tag{6}$$

2.2 GROMOV-WASSERSTEIN DISTANCE, ENTROPIC GWD AND FUSED GWD (FGWD)

For a given coupling $\mathbf{T} \in \mathbb{R}_{\geq 0}^{m \times n}$, consider the following loss function:

$$\mathcal{G}(\mathbf{T}) = \frac{1}{2} \sum_{i,k=1}^{m} \sum_{j,\ell=1}^{n} \left(C_X(i,k) - C_Y(j,\ell)\right)^2 T_{ij} T_{k\ell}. \tag{7}$$

Intuitively, $\mathcal{G}(\mathbf{T})$ measures how well the coupling conducts a matching between X and Y, by looking into the discrepancy between intra-metric points' relationships (given as costs). Quadratic loss is a natural choice. Importantly the matching is soft, thus we do not require $m = n$. Probabilities at points can be interpreted as weights, providing a convenient way to measure uncertainties about the data (smaller weights imply large uncertainties). Since costs do not involve cross-metric terms, the loss defined in Eq. 7 is valid also on metric spaces that do not relate to each other in any way. As previously noticed, the admissible couplings satisfy the following:

$$\Pi(\mathbf{a}, \mathbf{b}) = \left\{\mathbf{T} \geq 0 : \mathbf{T}\mathbf{1}_n = \mathbf{a}, \quad \mathbf{T}^\top \mathbf{1}_m = \mathbf{b}\right\}. \tag{8}$$

The *Gromov-Wasserstein Distance* (GWD) is then defined as follows:

$$\mathrm{GWD}(\mathbf{a}, \mathbf{b}) = \min_{\mathbf{T} \in \Pi(\mathbf{a}, \mathbf{b})} \mathcal{G}(\mathbf{T}). \tag{9}$$

The entropic relaxation of the above definition leverages an entropy-based regularizer:

$$\mathrm{GWD}_\epsilon(\mathbf{a}, \mathbf{b}) = \min_{\mathbf{T} \in \Pi(\mathbf{a}, \mathbf{b})} \left[\mathcal{G}(\mathbf{T}) + \varepsilon \sum_{i,j} T_{ij}(\log T_{ij} - 1)\right]. \tag{10}$$

One of the standard approaches to solving entropic GWD is by the 2-level iterative procedure. Here the outer loop iterates over different transport matrices $\mathbf{T}$ and the inner loop conducts regular Sinkhorn iterations, as described in Sec. 1, but for the kernel matrix $\mathbf{K}_\epsilon$ defined as: $\mathbf{K}_\epsilon = \exp(-\frac{\mathbf{D}_\mathbf{T}}{\epsilon})$, where $\mathbf{D}_\mathbf{T}$ is given as follows for $\mathbf{p} = \mathbf{T}\mathbf{1}_n$ and $\mathbf{q} = \mathbf{T}^\top \mathbf{1}_m$: $\mathbf{D}_\mathbf{T} \in \mathbb{R}^{n \times m}$:

$$\mathbf{D}_\mathbf{T} = (\mathbf{C}_X^{\circ 2} \mathbf{p})\mathbf{1}_m^\top + \mathbf{1}_n (\mathbf{C}_Y^{\circ 2} \mathbf{q})^\top - 2\mathbf{C}_X \mathbf{T} \mathbf{C}_Y^\top, \tag{11}$$

Here $\circ$ stands for the Hadamard (element-wise) product. After each step of the inner loop, new transport matrix $\mathbf{T}$ is formed and next outer loop step is invoked.

Even though, as emphasized earlier, regular GWD does not introduce any cross-cost terms, there exists its generalization that leverages the so-called *cross-cost* matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$. Matrix $\mathbf{M}$ is usually given as: $M_{ij} = \|\mathbf{f}_i - \mathbf{g}_j\|_2^2$ for two sets of vectors $\{\mathbf{f}_i\}_{i=1}^m$ and $\{\mathbf{g}_j\}_{j=1}^n$.

That generalization effectively replaces the $\mathcal{G}(\mathbf{T})$ objective with the $\alpha\mathcal{G}(\mathbf{T}) + (1-\alpha)\mathbf{M} \circ \mathbf{T}$ (for a hyperparameter $\alpha \in [0,1]$), and we refer to it as *Fused Gromov-Wasserstein Distance* (FGWD). Its entropic relaxation is given by the following formula:

$$\mathrm{FGWD}_{\epsilon,\alpha}(\mathbf{a},\mathbf{b}) = \min_{\mathbf{T}\in\Pi(\mathbf{a},\mathbf{b})} \left[\alpha\mathcal{G}(\mathbf{T}) + (1-\alpha)\mathbf{M} \circ \mathbf{T} + \varepsilon \sum_{i,j} T_{ij}(\log T_{ij} - 1) \right]. \tag{12}$$

The corresponding kernel matrix $\mathbf{K}_{\epsilon,\alpha}$ is now given as $\mathbf{K}_{\epsilon,\alpha} = \exp(-\frac{\mathbf{Q_T}}{\epsilon})$ for:

$$\mathbf{Q_T} = \alpha\mathbf{D_T} + (1-\alpha)\mathbf{M}. \tag{13}$$

We normalize source/target structural costs by a shared, template-derived scale. Independently normalizing two spaces would erase global scale differences and define a different matching problem.

3 Efficient Geodesic Gromov-Wasserstein (EGGroW)

We are ready to present the class of the Efficient Geodesic Gromov-Wasserstein (EGGroW) algorithms. We focus on the entropic approach and start with exact algorithms that rely on the recently introduced *GenusSink* method (Choromanski et al., 2026). The only assumption we need to make regarding that exact class is that graphs under consideration have separators of size $O(\log\log(n))$ (near -exact extension exists, as shown in (Choromanski et al., 2026), for even more general bounded genus graphs). We then propose even more computationally efficient variants, but at the cost of approximating the original solution. We will focus on FGWD, as the most general variant.

3.1 Exact GWD Computations with Deterministic EGGROW

Looking at Eq. 11, 13, we conclude that the main computational bottlenecks present while computing entropic FGWD involve: multiplications with matrices $\mathbf{C}$ and $\mathbf{C}^{\circ 2}$, Hadamard products with matrices $\mathbf{M}$ and element-wise exponentiations. Rather than storing $\mathbf{C}$ and $\mathbf{C}^{\circ 2}$ explicitly, we will instead apply the efficient **deterministic** polylog-linear GenusSink algorithm (Choromanski et al., 2026) for matrix-vector multiplications involving $\mathbf{C}$. GenusSink leverages separator-based factorization of low genus graphs (Gilbert et al., 1984; Choromanski et al., 2023) for efficient computations involving a broad range of cost matrices relying on distances between graphs' vertices (defined via shortest paths).

Our first observation is that a small modification of the GenusSink algorithm can be applied to efficient multiplications with both $\mathbf{C}$ and $\mathbf{C}^{\circ 2}$ (in (Choromanski et al., 2026) the algorithm is applied to efficient multiplications with $\exp(\mathbf{C})$). We prove it in the lemma below.

Lemma 3.1 (Efficient multiplications with $\mathbf{C}$ and $\mathbf{C}^{\circ 2}$)**.** *For n-vertex graphs under consideration with separators of size $O(\log\log n)$, multiplications with matrices $\mathbf{C}$ and $\mathbf{C}^{\circ 2}$, where $C_{i,j} = d(i,j)$ is the shortest path distance between vertices i and j, can be conducted in polylog-linear time.*

Proof. We will leverage time complexity analysis of the cross_compute method from (Choromanski et al., 2026). By leveraging it, we deduce that it suffices to prove that for any two sequences of scalars $(x_1,...,x_m)$,$(y_1,...,y_n)$, matrix-vector multiplications with two matrices: $\mathbf{P} = [x_i + y_j]_{i=1,...,m}^{j=1,...,n}$ and $\mathbf{P}^{\circ 2}$ can be conducted in polylog-linear time. Note that for for any $\mathbf{v} \in \mathbb{R}^n$, we have: $(\mathbf{Pv})_i = x_i \sum_{j=1}^n v_j + \sum_{j=1}^n y_j v_j$ for $i = 1,...,m$. Thus we can compute all $(\mathbf{Pv})_i$ in time $O(m+n)$. Now let us focus on $\mathbf{P}^{\circ 2}$. We have: $\mathbf{P}^{\circ 2} = [x_i^2 + 2x_i y_j + y_j^2]_{i=1,...,m}^{j=1,...,n}$. The $O(m+n)$ time matrix-vector multiplications with $\mathbf{P}^{\circ 2}$ follows naturally from our analysis for $\mathbf{P}$ and the fact that matrix $[x_i y_j]_{i=1,...,m}^{j=1,...,n}$ trivially support $O(m+n)$ matrix-vector multiplication via its rank-1 decomposition. □

For given matrices $\mathbf{\Lambda}$, $\mathbf{V}$ ($\mathbf{V}$ potentially being a vector), define the action of $\mathbf{\Lambda}$ on $\mathbf{V}$ as: $\mathcal{S}_{\mathbf{\Lambda}}(\mathbf{V})$. We introduce extra notation rather than denoting this action simply as $\mathbf{\Lambda V}$ to highlight that its computation might not require brute-force matrix-matrix multiplication. With that notation in place,

we can re-write matrix $\mathbf{D_T}$ from Equation 11 as follows, using intermediate calculations:

$$\mathbf{A} = \mathcal{S}_{\mathbf{C}_X^{\circ 2}}(\mathbf{p}), \qquad \mathbf{B} = \mathcal{S}_{\mathbf{C}_Y^{\circ 2}}(\mathbf{q}), \tag{14}$$

$$\mathbf{U} = \mathcal{S}_{\mathbf{C}_X}(\mathbf{T}), \qquad \mathbf{V} = \mathcal{S}_{\mathbf{C}_Y}(\mathbf{U}^\top)^\top, \tag{15}$$

$$\mathbf{D_T} = \mathbf{A}\mathbf{1}_m^\top + \mathbf{1}_n\mathbf{B}^\top - 2\mathbf{V}. \tag{16}$$

We obtained exact entropic GWD algorithm, where cubic computations involving the construction of a matrix $\mathbf{D_T}$ from Equation 11 (namely, the computation of $\mathbf{C}_X\mathbf{T}\mathbf{C}_Y^\top$) were replaced by **polylog-quadratic**. We refer to this method as a **deterministic EGGroW**, or simply: **DET-EGGroW**.

3.2 Randomized Approximate EGGroW for Additional Speedups

We will now show how the additional speedups for the EGGroW algorithm can be obtained by applying the theory of random features (Rahimi & Recht, 2007; Scetbon & Cuturi, 2020). We will use the following notation.

$$\mathbf{U} = \mathbf{C}_X\mathbf{T}, \quad \mathbf{A} = \mathbf{C}_X^{\circ 2}\mathbf{p}, \quad \mathbf{B} = \mathbf{C}_Y^{\circ 2}\mathbf{q}, \tag{17}$$

and define *augmented query and key vectors* $\{\mathbf{z}_i\}_{i=1}^m, \{\mathbf{w}_j\}_{j=1}^n$ as follows, where $M_{i,j} = \|\mathbf{f}_i - \mathbf{g}_j\|_2^2$:

$$\mathbf{z}_i = \begin{bmatrix} \sqrt{2\alpha/\varepsilon}\,\mathbf{U}_{i,:} \\ \sqrt{2(1-\alpha)/\varepsilon}\,\mathbf{f}_i \end{bmatrix}, \qquad \mathbf{w}_j = \begin{bmatrix} \sqrt{2\alpha/\varepsilon}\,(\mathbf{C}_Y)_{j,:} \\ \sqrt{2(1-\alpha)/\varepsilon}\,\mathbf{g}_j \end{bmatrix}. \tag{18}$$

Each iteration of the entropic FGWD computation critically relies on the multiplications with kernel matrices $\mathbf{K_T} = [\exp(-Q_T(i,j)/\varepsilon)]_{i=1,\ldots,n}^{j=1,\ldots,m}$. Our key observation is that those matrices admit **unbiased** low-rank factorization. To see that, note first that $\mathbf{K_T}$ are related to softmax-kernel matrices:

$$\mathbf{K_T}(i,j) = \exp(-Q_T(i,j)/\varepsilon) = s_i \exp(\mathbf{z}_i^\top \mathbf{w}_j) t_j, \tag{19}$$

where

$$s_i = \exp\left(-\frac{\alpha A_i + (1-\alpha)\|\mathbf{f}_i\|_2^2}{\varepsilon}\right), \tag{20}$$

$$t_j = \exp\left(-\frac{\alpha B_j + (1-\alpha)\|\mathbf{g}_j\|_2^2}{\varepsilon}\right). \tag{21}$$

Using positive feature maps $\phi : \mathbb{R}^n \to \mathbb{R}^r$ from (Choromanski et al., 2021), for the unbiased estimation of the softmax-kernel, we approximate it as follows for $\omega_1, \ldots \omega_r \overset{\text{iid}}{\sim} \mathcal{N}(0, \mathbf{I}_r)$:

$$\begin{aligned} \exp(\mathbf{z}_i^\top \mathbf{w}_j) &\overset{\mathbb{E}}{=} \phi(\mathbf{z}_i)^\top \phi(\mathbf{w}_j), \\ \phi(\mathbf{x}) &\overset{\text{def}}{=} \frac{1}{\sqrt{r}} \exp\left(-\frac{\|\mathbf{x}\|^2}{2}\right) \left(\exp(\omega_1^\top \mathbf{x}), \ldots, \exp(\omega_r^\top \mathbf{x})\right)^\top. \end{aligned} \tag{22}$$

That directly leads to the following unbiased low-rank factorization of the kernel matrix $\mathbf{K_T}$.

$$\mathbf{K_T} \overset{\mathbb{E}}{=} (\operatorname{diag}(\mathbf{s})\Phi_X^\top)(\Phi_Y \operatorname{diag}(\mathbf{t})), \tag{23}$$

where $\Phi_X \in \mathbb{R}^{r\times m}$ and $\Phi_Y \in \mathbb{R}^{r\times n}$ are obtained by stacking (as columns) vectors $\{\phi(\mathbf{z}_i)\}_{i=1}^n$ and $\{\phi(\mathbf{w}_j)\}_{j=1}^m$ respectively. Sinkhorn scaling preserves this factorization, so the approximate coupling can be stored as $\mathbf{T} = \mathbf{L}^\top\mathbf{R}$ for low-rank matrices $\mathbf{L}, \mathbf{R}$.

In practice, centering and diagonal balancing are used to reduce the variance of the approximation. All scale factors must be retained in semi-relaxed problems because, unlike balanced Sinkhorn, separable factors can affect a learned marginal. Finally, note that vectors $\mathbf{A}$ and $\mathbf{B}$ can be computed in polylog-linear time, with the use of the techniques developed for DET-EGGroW (see: Sec. 3.1) and furthermore, $\mathbf{U}$ can be computed in either: polyloq-quadratic time (again, using techniques from Sec. 3.1) or in time $O(m^2 r)$ (where r is the number of random features) if low-rank decomposition of $\mathbf{T}$ is leveraged. Thus the resulting algorithm also runs in **near-quadratic time**, rather than cubic, as its brute-force counterpart. We refer to it as an **approximate EGGroW**, or simply: **APP-EGGroW**. The computational advantage of APP-EGGroW over DET-EGGroW is that in practice operations involving random features are better suited for modern accelerators than separator-based approaches. The price is some accuracy loss. We analyze this in more depth in Sec. 5. Table 1 summarizes the computational profiles of the brute-force algorithm, as well as two EGGroW variants.

Table 1: The three implementations of the Gromov-Wasserstein Distance calculation remove different dense objects. Here r is the number of random features for APP-EGGroW. Operator storage depends on the selected separator or analytic moment backend.

Method	Intra-space distances	Cross-space state	Status
Dense GWD (brute-force)	Dense $\mathbf{C}_X, \mathbf{C}_Y$	Dense $\mathbf{T}, \mathbf{D_T}, \mathbf{K_T}$	Exact
DET-EGGroW	Operator form	Dense $\mathbf{T}, \mathbf{D_T}, \mathbf{K_T}$	Exact if operators exact
APP-EGGroW	Operator form	$O((n+m)r)$ factors	Approximate

4 EGGroW for 3D Template Detection

In the previous section, we described in depth the class of EGGroW algorithms. We will now propose a method that applies Fused Gromov-Wasserstein Distance for the 3D template detection problem. EGGroW algorithms make this method practical from the computational standpoint.

4.1 Observation model and subsampling

Our input is a point cloud or a mesh (e.g. CAD) template $\mathbf{Y} = \{\mathbf{y}_j\}_{j=1}^n \subset \mathbb{R}^3$ and a calibrated multi-view RGB-D observation of a scene. The solver-visible data comprise depth returns, camera calibration, template geometry, and any descriptors computed from those observations. Instance labels, the generated object pose, and visibility truth are held in a separate evaluation object.

We voxel-subsample the observed scene without using instance labels and sample or aggregate the template surface into m discrete nodes $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_m\}$. Template nodes carry a normalized surface-area prior $\mathbf{b} \in \Delta_n$; the subsampled scene carries prior $\mathbf{a} \in \Delta_m$. We report the raw RGB-D count, retained scene count, voxel size, template vertex count, terminal count, and surface-area ownership construction.

4.2 Stage I: capacity-constrained scene localization

Let $\rho_s \in (0, 1]$ be the minimum scene coverage measured with respect to the scene prior $\mathbf{a}$. We solve:

$$\min_{\mathbf{T} \geq 0} \quad \alpha \mathcal{G}(\mathbf{T}) + (1-\alpha)\mathbf{M} \circ \mathbf{T} + \varepsilon \sum_{i,j} T_{ij}(\log T_{ij} - 1) \quad \text{s.t.} \quad \mathbf{T}^\top \mathbf{1}_m = \mathbf{b}, \mathbf{T}\mathbf{1}_n \leq \frac{\mathbf{a}}{\rho_s}. \tag{24}$$

The first constraint fixes the complete template marginal to $\mathbf{b}$, so the coupling has total mass one, while the scene distribution $\mathbf{p} = \mathbf{T}\mathbf{1}_n$ is learned. The second constraint limits the mass assigned to each scene point to $p_i \leq a_i/\rho_s$. Consequently, the coupling cannot concentrate all its mass on a subset of scene points whose total prior mass is less than ρ_s. To score scene relevance, we compare the learned mass at each point with its prior weight: $s_i^{\text{scene}} = p_i/a_i$. This multiplicative change accounts for nonuniform prior weights, rather than ranking points by raw correspondence mass p_i. A threshold chosen on calibration data can then define the predicted scene support for Stage II; no threshold may be chosen from test labels.

4.3 Stage II: template visibility and correspondence

Stage II fixes the predicted scene distribution $\widehat{\mathbf{p}}$ and learns the template marginal (Vincent-Cuaz et al., 2022):

$$\mathbf{T}\mathbf{1}_n = \widehat{\mathbf{p}}, \qquad \mathbf{q} = \mathbf{T}^\top \mathbf{1}_m. \tag{25}$$

The value q_j is the recovered correspondence mass assigned to node j. Two forms of partial-template control are useful:

1. a capacity $q \leq b/\rho_t$, where ρ_t is minimum template coverage; or
2. a soft marginal penalty $\tau D_{\text{KL}}(q\|r)$ around a strictly positive, truth-free reference r.

Since the template's position and orientation are initially unknown, we estimate an initial rigid alignment $(\mathbf{R}_0, \mathbf{t}_0)$ from the correspondences obtained in Stage I, discussed in App. A.1. This alignment rotates and translates the template into scene coordinates without changing its shape, and is held

fixed throughout Stage II. Using the calibrated camera views, we raycast the aligned template to obtain the visibility reference $\mathbf{q}_{\text{ray}}$. In Sec. 5.2, we set $\mathbf{r} = 0.98\mathbf{q}_{\text{ray}} + 0.02\mathbf{b}$ to retain positive support everywhere.

Let D be the template diameter and G_X, G_Y the scene and template graphs, respectively. Writing $d_G(u, v)$ for the shortest-path distance between vertices u and v on graph G, we define the costs

$$C_X(i,k) = d_{G_X}(\mathbf{x}_i, \mathbf{x}_k)/D, \qquad C_Y(j,\ell) = d_{G_Y}(\mathbf{y}_j, \mathbf{y}_\ell)/D. \tag{26}$$

To also favor matches consistent with the upstream pose, we define a cross-cost that penalizes the squared Euclidean distance between a scene point and the predicted location of a template point:

$$M_{ij}^{\text{pose}} = \frac{\|\mathbf{x}_i - (\mathbf{y}_j \mathbf{R}_0^\top + \mathbf{t}_0)\|_2^2}{D^2}. \tag{27}$$

Division by D^2 makes this quantity dimensionless. Stage II then solves the following entropically regularized FGWD$_{\epsilon,\alpha}$ objective with an additional KL penalty, using the EGGroW algorithm:

$$\begin{aligned} \min_{\mathbf{T}\geq 0} \quad & \lambda_G \mathcal{G}(\mathbf{T}) + \lambda_M \langle \mathbf{M}^{\text{pose}}, \mathbf{T} \rangle + \varepsilon \sum_{i,j} T_{ij}(\log T_{ij} - 1) + \tau D_{\text{KL}}(\mathbf{q} \| \mathbf{r}) \\ \text{s.t.} \quad & \mathbf{T}\mathbf{1}_n = \widehat{\mathbf{p}}, \qquad \mathbf{q} = \mathbf{T}^\top \mathbf{1}_m \end{aligned} \tag{28}$$

The KL term is a soft visibility penalty: $\mathbf{q}$ can depart from the raycast reference when the relational geometry favors another visible region. Additional details for how we integrate the KL term with the FGWD objective in the optimization formulation are reported in App. A.2. In contrast, the row-marginal constraint is enforced exactly; any capacity or affine-moment constraints used in a variant are likewise imposed as hard constraints. As we show in Sec. 5.2, EGGroW algorithm makes this 3D template detection algorithm computationally feasible for large-size templates and scenes. We will also show in Sec. 5.2 that proposed EGGroW-powered 3D detection method is capable of finding templates of various profiles, in particular of irregular shapes.

5 Experiments

5.1 Pose Correspondence

We evaluate EGGroW on FAUST meshes Bogo et al. (2014) (see Fig. 1). The registrations share vertex identities, giving ground-truth correspondences. We use source scan 004 and three target scans from the same subject: 007, 006, and 003. For each pair, we sample $n = 512, 1024, 2000, 4096, 6890$ vertices, construct weighted mesh graphs from triangle edges, compute sampled geodesic matrices through the full graph, and use uniform marginals. For FGWD, M_{ij} is the squared Euclidean distance between normalized 3D coordinates and we set $\alpha = 0.95$, so the objective is mostly intrinsic GWD with a small Euclidean feature term.

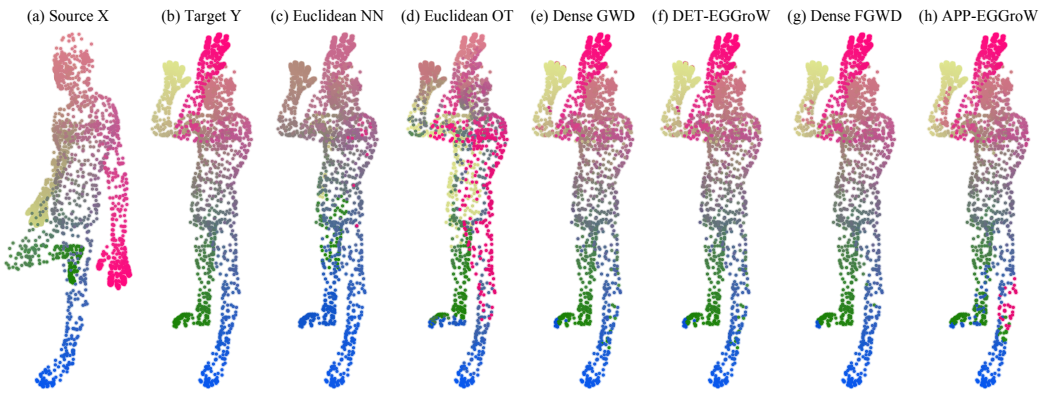

Figure 1: FAUST pose correspondence for (a) source scan 004 and (b) target scan 003. Colors are intrinsic source colors transferred to the target by each method (c)-(h).

We compare against two Euclidean baselines. Euclidean nearest neighbors (NN) independently maps each source point to the closest target point in normalized 3D space: $j(i) = \arg\min_j \|\mathbf{x}_i - \mathbf{y}_j\|^2$. This baseline is purely local and can send many source points to the same target region. Euclidean OT instead solves entropic optimal transport with cost $M_{ij} = \|\mathbf{x}_i - \mathbf{y}_j\|^2$ and uniform marginals. This enforces global mass balance, but it is still based only on extrinsic 3D proximity. Both baselines can therefore fail when unrelated body parts become close in Euclidean space. Tab. 2 shows performance, which we primarily measure using mean target-geodesic correspondence error; lower is better. Further details and results can be found in App. B.

Table 2: Mean target-geodesic correspondence error on three FAUST same-subject pose pairs at $n = 2000$ vertices. Euclidean OT and FGWD use squared Euclidean cross-feature costs.

Pair	NN	OT	GWD	FGWD	DET	APP_{64}	APP_{256}	APP_{1024}
Geodesic correspondence error ↓								
004 → 003	0.574	0.636	**0.332**	**0.166**	**0.166**	0.240	0.232	0.230
004 → 006	0.518	0.641	**0.323**	**0.108**	**0.108**	0.252	0.206	0.206
004 → 007	0.665	0.633	**0.354**	**0.223**	**0.223**	0.281	0.261	0.237
Runtime (seconds) ↓								
004 → 003	-	-	896.4	1077.2	587.0	**320.8**	357.1	535.1
004 → 006	-	-	794.3	878.8	585.3	**322.3**	354.4	539.0
004 → 007	-	-	831.1	996.5	585.0	**322.7**	357.1	531.3

Fig. 2 shows the expected accuracy–runtime tradeoff for APP-EGGroW. Increasing the number of random features improves the mean geodesic error, with 1024 features giving the best APP-EGGroW accuracy in this pose pair. At the same time, smaller feature counts are faster than the Dense GWD/FGWD baselines, and both DET-EGGroW and APP-EGGroW runtimes grow significantly slower than baseline methods.

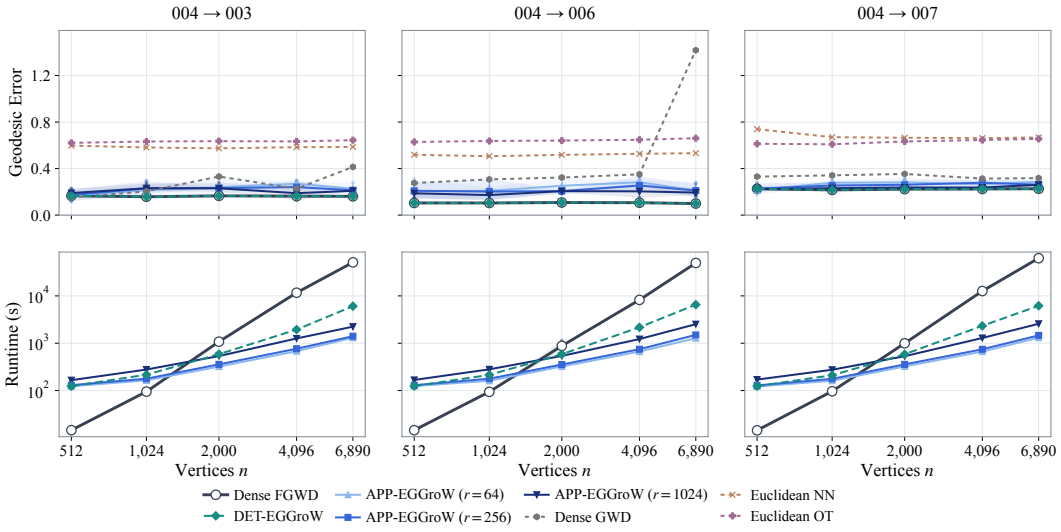

Figure 2: Error (top) and runtime (bottom) across different methods as n increases.

5.2 Template Detection

We compare Dense GWD, Dense FGWD, DET-EGGroW, APP-EGGroW, and Scetborn-Cuturi Scetbon et al. (2022) across increasing sizes meshes—Fig. 3 shows the scene (i.e., a room) and one of the templates (i.e., the octopus tail). ET-EGGroW matches dense transport plans to numerical precision and accelerates larger ribbon and point-cloud instances, but remains slower on sphere–ellipsoid meshes because of preprocessing and iteration overhead. APP-EGGroW is fastest on the larger Lego block and Octopus instances, at the cost of transport-plan approximation error. For more setup details, see Sec. 4 and App. B.3.

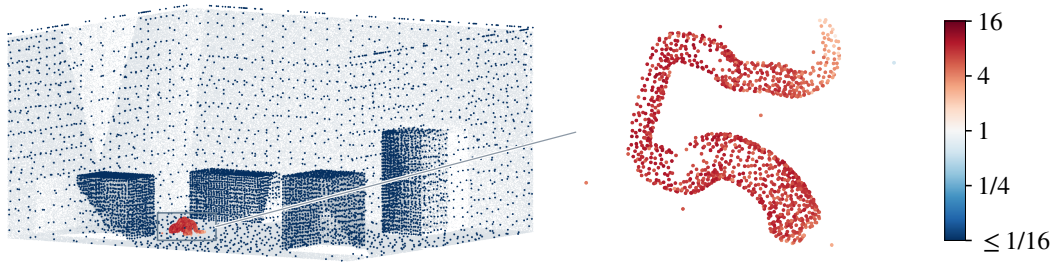

Figure 3: Octopus template localization where colors show the recovered scene density p_i/a_i; gray points provide scene context.

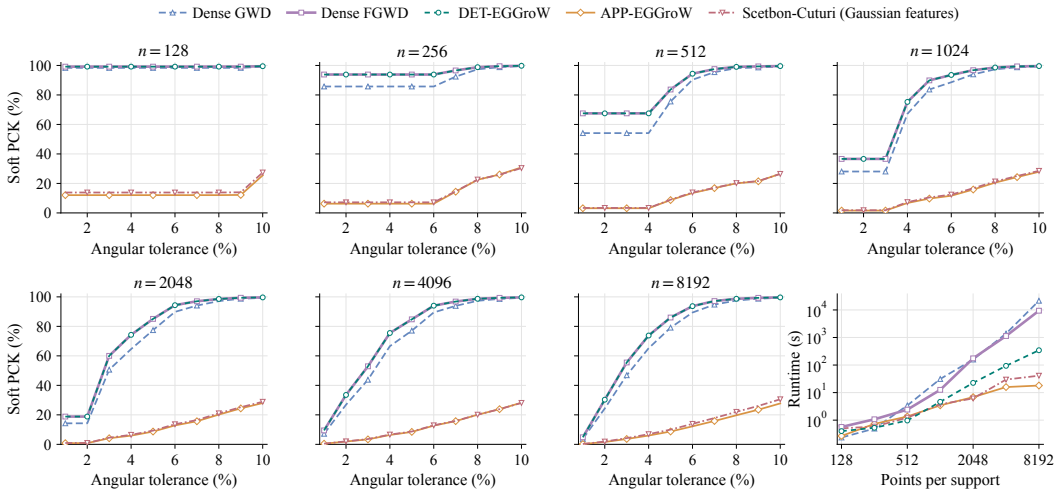

Figure 4: Mean accuracy and runtime (bottom right) at increasing mesh sizes for each of the different methods under comparison.

A successful transport plan concentrates mass around the correct correspondences. We measure this concentration using soft PCK, that is, the fraction of transport mass assigned to pairs within a prescribed error tolerance. For our surfaces, correspondence error is the angle between their known underlying spherical directions. A 10% tolerance therefore permits an angular error of 18°, capturing how reliably each method recovers the correct region of the shape. Our results show that DET-EGGroW follows Dense FGWD across the entire tolerance range, preserving the reference method's correspondence accuracy. Feature fusion is particularly helpful when matches must be precise: at $n = 1024$ and 1% tolerance, soft PCK increases from approximately 28% for Dense GWD to 37% for Dense FGWD and DET-EGGroW. Furthermore, at 10% tolerance, all three exceed 99%, indicating that most transport mass remains near the correct region of the shape. Finally, limitations and future work for both template detection and pose correspondence can be found in App. B.4.

6 Conclusion

We presented in this paper **E**fficient **G**eodesic **Gro**mov-**W**asserstein (EGGroW) algorithms for efficiently computing Gromov-Wasserstein distances, with metrics induced by shortest-path distance on graphs. EGGroW is characterized by near-quadratic computational time (as compared to the cubic of its brute-force entropic GWD counterpart) and leverages methods based on graphs factorization with small-size separators, proposed in GenusSink algorithm, as well as positive random features techniques used in efficient attention modules for Transformers. Our theoretical analysis is confirmed by experimental validation on the problems on 3D pose estimation and 3D template detection. Our proposed GWD-based template detection algorithm critically relies on the computational efficiency of EGGroW to handle large input point clouds / meshes.

AI USE STATEMENT

We have not used generative AI tools to produce the content of this paper.

ETHICS STATEMENT

We believe that our paper does not raise any questions regarding the Code of Ethics. In particular, none of our studies involves human subjects. We are also not aware of any potential conflicts of interest related to this submission.

REPRODUCIBILITY STATEMENT

We have described in depth all presented algorithms, as well as the experimental setup in which they were used, so that all our experimental results can be accurately reproduced. In particular, we dedicate a separate section just to describe in depth proposed 3D template detection algorithm (see: Sec. 4). All additional details (hyperparams, datasets used, etc.) are listed in the corresponding sections (see in particular: Sec. 5). We will open source our code upon acceptance of the manuscript.

REFERENCES

Shun-ichi Amari, Ryo Karakida, Masafumi Oizumi, and Marco Cuturi. Information geometry for regularized optimal transport and barycenters of patterns. *Neural Computation*, 31(5):827–848, 05 2019. ISSN 0899-7667. doi: 10.1162/neco_a_01178. URL https://doi.org/10.1162/neco_a_01178.

Luigi Ambrosio, Elia Brué, and Daniele Semola. Lectures on optimal transport. *UNITEXT*, 2021. URL https://api.semanticscholar.org/CorpusID:244769124.

Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. FAUST: Dataset and evaluation for 3D mesh registration. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Piscataway, NJ, USA, June 2014. IEEE.

Krzysztof Choromanski, Derek Long, Ananya Parashar, and Dwaipayan Saha. Near-linear time generalized sinkhorn algorithms for bounded genus graphs. *CoRR*, abs/2605.09782, 2026. doi: 10.48550/ARXIV.2605.09782. URL https://doi.org/10.48550/arXiv.2605.09782.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking attention with performers. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=Ua6zuk0WRH.

Krzysztof Marcin Choromanski, Arijit Sehanobish, Han Lin, Yunfan Zhao, Eli Berger, Tetiana Parshakova, Alvin Pan, David Watkins, Tianyi Zhang, Valerii Likhosherstov, Somnath Basu Roy Chowdhury, Kumar Avinava Dubey, Deepali Jain, Tamas Sarlos, Snigdha Chaturvedi, and Adrian Weller. Efficient graph field integrators meet point clouds. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 5978–6004. PMLR, 2023. URL https://proceedings.mlr.press/v202/choromanski23b.html.

Samir Chowdhury, David Miller, and Tom Needham. Quantized Gromov-Wasserstein. In *Machine Learning and Knowledge Discovery in Databases. Research Track*, volume 12977 of *Lecture Notes in Computer Science*, pp. 811–827. Springer, 2021. doi: 10.1007/978-3-030-86523-8_49.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pp. 2292–2300, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/af21d0c97db2e27e13572cbf59eb343d-Abstract.html.

Alessio Figalli. The optimal partial transport problem. *Archive for Rational Mechanics and Analysis*, 195:533–560, 2010. URL https://api.semanticscholar.org/CorpusID:15899562.

Aude Genevay, Marco Cuturi, Gabriel Peyré, and Francis R. Bach. Stochastic optimization for large-scale optimal transport. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 3432–3440, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/2a27b8144ac02f67687f76782a3b5d8f-Abstract.html.

John R. Gilbert, Joan P. Hutchinson, and Robert Endre Tarjan. A separator theorem for graphs of bounded genus. *Journal of Algorithms*, 5(3):391–407, 1984. doi: 10.1016/0196-6774(84)90019-1.

W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5):922–923, 1976. doi: https://doi.org/10.1107/S0567739476001873.

Michal Kotrbcík and Martin Skoviera. Matchings, cycle bases, and the maximum genus of a graph. *Electron. J. Comb.*, 19(3):3, 2012. doi: 10.37236/2479. URL https://doi.org/10.37236/2479.

Facundo Mémoli. Gromov–Wasserstein distances and the metric approach to object matching. *Found. Comut. Math.*, 11(4):417–487, August 2011.

Eduardo Fernandes Montesuma, Fred Ngolè Mboula, and Antoine Souloumiac. Recent advances in optimal transport for machine learning, 2024. URL https://arxiv.org/abs/2306.16156.

Sina Moradi. A survey on algorithmic developments in optimal transport problem with applications. *ArXiv*, abs/2501.06247, 2025. URL https://api.semanticscholar.org/CorpusID:275470703.

Yann Ollivier, Herve Pajot, and Cédric Villani (eds.). *Optimal Transport - Theory and Applications*, volume 413 of *London Mathematical Society lecture note series*. Cambridge University Press, 2014. ISBN 978-1-10-768949-7. URL http://www.cambridge.org/de/academic/subjects/mathematics/geometry-and-topology/optimal-transport-theory-and-applications.

Ananya Parashar, Derek Long, Dwaipayan Saha, and Krzysztof Choromanski. Manifold random features. *arXiv preprint arXiv:2602.03797*, 2026. doi: 10.48550/arXiv.2602.03797. URL https://arxiv.org/abs/2602.03797.

Luiz Manella Pereira and M. Hadi Amini. A survey on optimal transport for machine learning: Theory and applications. *IEEE Access*, 13:26506–26526, 2025. doi: 10.1109/ACCESS.2025.3539926. URL https://doi.org/10.1109/ACCESS.2025.3539926.

Gabriel Peyré. Optimal transport for machine learners. *CoRR*, abs/2505.06589, 2025. doi: 10.48550/ARXIV.2505.06589. URL https://doi.org/10.48550/arXiv.2505.06589.

Gabriel Peyré and Marco Cuturi. Computational optimal transport. *Found. Trends Mach. Learn.*, 11(5-6):355–607, 2019. doi: 10.1561/2200000073. URL https://doi.org/10.1561/2200000073.

Gabriel Peyré, Marco Cuturi, and Justin Solomon. Gromov-wasserstein averaging of kernel and distance matrices. In Maria-Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, JMLR Workshop and Conference Proceedings, pp. 2664–2672. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/peyre16.html.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, volume 20, 2007. URL https://proceedings.neurips.cc/paper/2007/hash/013a006f03dbc5392effeb8f18fda755-Abstract.html.

Meyer Scetbon and Marco Cuturi. Linear time sinkhorn divergences using positive features. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 13468–13480. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/9bde76f262285bb1eaeb7b40c758b53e-Paper.pdf.

Meyer Scetbon, Gabriel Peyré, and Marco Cuturi. Linear-time gromov wasserstein distances using low rank couplings and costs. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 19347–19365. PMLR, 2022. URL https://proceedings.mlr.press/v162/scetbon22b.html.

Titouan Vayer, Laetitia Chapel, Remi Flamary, Romain Tavenard, and Nicolas Courty. Fused Gromov-Wasserstein distance for structured objects. *Algorithms*, 13(9):212, August 2020.

Cédric Villani. Topics in optimal transportation. 2003. URL https://api.semanticscholar.org/CorpusID:118448577.

Cédric Vincent-Cuaz, Rémi Flamary, Marco Corneli, Titouan Vayer, and Nicolas Courty. Semi-relaxed Gromov-Wasserstein divergence with applications on graphs. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2110.02753.

Kazuki Watanabe and Noboru Isobe. Sinkhorn algorithm for sequentially composed optimal transports. *CoRR*, abs/2412.03120, 2024. doi: 10.48550/ARXIV.2412.03120. URL https://doi.org/10.48550/arXiv.2412.03120.

Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin. Gromov-wasserstein learning for graph matching and node embedding. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, Proceedings of Machine Learning Research, pp. 6932–6941. PMLR, 2019. URL http://proceedings.mlr.press/v97/xu19b.html.

Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797*, 2016.

A Additional details for 3D template detection

A.1 Initial rigid alignment from the Stage I coupling

Stage I provides a soft correspondence matrix $\mathbf{T}^{(1)}$, where $T_{ij}^{(1)}$ measures the correspondence mass between scene point $\mathbf{x}_i$ and template point $\mathbf{y}_j$. We use these correspondences to estimate a single rigid transformation of the template into scene coordinates. With point coordinates written as row vectors, the transformation maps

$$\mathbf{y}_j \mapsto \mathbf{y}_j \mathbf{R}^\top + \mathbf{t}.$$

We estimate the initial rotation and translation by solving

$$(\mathbf{R}_0, \mathbf{t}_0) \in \underset{\mathbf{R}\in \mathrm{SO}(3),\, \mathbf{t}\in\mathbb{R}^{1\times 3}}{\arg\min} \sum_{i=1}^{m}\sum_{j=1}^{n} T_{ij}^{(1)} \left\| \mathbf{x}_i - (\mathbf{y}_j \mathbf{R}^\top + \mathbf{t}) \right\|_2^2, \tag{29}$$

where

$$\mathrm{SO}(3) = \left\{ \mathbf{R} \in \mathbb{R}^{3\times 3} : \mathbf{R}^\top \mathbf{R} = \mathbf{I}_3,\, \det(\mathbf{R}) = 1 \right\}.$$

Thus, scene–template pairs with larger Stage I correspondence mass have greater influence on the fitted pose. We use all weighted candidate matches rather than first selecting a single match for each point.

Weighted centering. Let

$$\mathbf{p}^{(1)} = \mathbf{T}^{(1)}\mathbf{1}_n$$

be the scene marginal learned in Stage I. Since the template marginal is fixed to $\mathbf{b}$, the weighted centroids of the scene and template are

$$\bar{\mathbf{x}} = \sum_{i=1}^{m} p_i^{(1)}\mathbf{x}_i, \qquad \bar{\mathbf{y}} = \sum_{j=1}^{n} b_j\mathbf{y}_j. \tag{30}$$

For any fixed rotation $\mathbf{R}$, the optimal translation aligns these two centroids:

$$\mathbf{t} = \bar{\mathbf{x}} - \bar{\mathbf{y}}\mathbf{R}^\top.$$

We can therefore estimate the rotation using the centered coordinates

$$\widetilde{\mathbf{x}}_i = \mathbf{x}_i - \bar{\mathbf{x}}, \qquad \widetilde{\mathbf{y}}_j = \mathbf{y}_j - \bar{\mathbf{y}}.$$

Rotation and translation. Using the centered points, define the weighted cross-covariance matrix

$$\mathbf{H} = \sum_{i=1}^{m}\sum_{j=1}^{n} T_{ij}^{(1)}\widetilde{\mathbf{y}}_j^\top\widetilde{\mathbf{x}}_i \in \mathbb{R}^{3\times 3}. \tag{31}$$

This matrix summarizes how the centered template and scene points are aligned under the Stage I correspondence weights. Substituting the optimal translation into Eq. 29 reduces the problem to finding the rotation that maximizes $\mathrm{tr}(\mathbf{RH})$ over $\mathbf{R} \in \mathrm{SO}(3)$. We solve this weighted alignment problem using the Kabsch algorithm (Kabsch, 1976).

Let

$$\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

be a singular value decomposition, and define

$$\mathbf{J} = \mathrm{diag}\left(1, 1, \det(\mathbf{V}\mathbf{U}^\top)\right). \tag{32}$$

The estimated rotation and translation are then

$$\mathbf{R}_0 = \mathbf{V}\mathbf{J}\mathbf{U}^\top, \qquad \mathbf{t}_0 = \bar{\mathbf{x}} - \bar{\mathbf{y}}\mathbf{R}_0^\top. \tag{33}$$

The matrix $\mathbf{J}$ prevents the solution from introducing a reflection, so that $\mathbf{R}_0$ is a valid rotation.

Use in Stage II. We compute $(\mathbf{R}_0, \mathbf{t}_0)$ once after Stage I and hold it fixed throughout Stage II. The fitted pose is used to raycast the aligned template and construct the visibility reference $\mathbf{q}_{\mathrm{ray}}$, and it also enters the pose-conditioned cross-cost $\mathbf{M}^{\mathrm{pose}}$. Stage II therefore refines template visibility and correspondence without re-estimating the rigid pose. Computing the alignment requires only the weighted centroids, the 3×3 matrix $\mathbf{H}$, and a 3×3 singular value decomposition; it does not require an additional GWD or Sinkhorn optimization.

A.2 KL-REGULARIZED STAGE II AND GENERALIZED SINKHORN UPDATES

We describe how the KL-divergence regularizer in Stage II modifies the Sinkhorn iterations. Stage II fixes the predicted scene distribution $\widehat{\mathbf{p}}$ and learns the template distribution

$$\mathbf{q} = \mathbf{T}^\top\mathbf{1}_m.$$

As in Sec. 4.3, we use the raycast-based reference $\mathbf{r} = 0.98\mathbf{q}_{\mathrm{ray}} + 0.02\mathbf{b}$. With $\mathbf{q}_{\mathrm{ray}}$ and $\mathbf{b}$ normalized and $\mathbf{b}$ strictly positive, $\mathbf{r}$ is a strictly positive probability vector. Thus, the KL penalty encourages the learned template distribution to remain close to the raycast prediction, while the positive reference weights ($r_j > 0$) allow every template point to receive correspondence mass.

For probability vectors $\mathbf{q}, \mathbf{r} \in \Delta_n$, we use

$$D_{\mathrm{KL}}(\mathbf{q}\|\mathbf{r}) = \sum_{j=1}^{n}\left(q_j\log\frac{q_j}{r_j} - q_j + r_j\right) = \sum_{j=1}^{n} q_j\log\frac{q_j}{r_j}, \tag{34}$$

where $0\log 0 = 0$. The second equality follows because $\sum_j q_j = \sum_j r_j = 1$. The coefficient τ controls the strength of this soft visibility prior: larger values of τ encourage the learned marginal $\mathbf{q}$ to remain closer to $\mathbf{r}$.

Inner optimization problem. We use the two-level iteration described in Sec. 2.2. As defined in the main body, at outer iteration k, its Stage-II specialization is

$$\begin{aligned}\mathbf{q}^{(k)} &= (\mathbf{T}^{(k)})^\top \mathbf{1}_m,\\ \mathbf{D}_{\mathbf{T}^{(k)}} &= (\mathbf{C}_X^{\circ 2}\widehat{\mathbf{p}})\mathbf{1}_n^\top + \mathbf{1}_m(\mathbf{C}_Y^{\circ 2}\mathbf{q}^{(k)})^\top - 2\mathbf{C}_X\mathbf{T}^{(k)}\mathbf{C}_Y^\top,\\ \mathbf{Q}_{\mathbf{T}^{(k)}} &= \lambda_G \mathbf{D}_{\mathbf{T}^{(k)}} + \lambda_M \mathbf{M}^{\text{pose}},\\ \mathbf{K}_{\mathbf{T}^{(k)}} &= \exp\left(-\frac{\mathbf{Q}_{\mathbf{T}^{(k)}}}{\varepsilon}\right).\end{aligned} \tag{35}$$

Here $\mathbf{D}_{\mathbf{T}^{(k)}}$ is the gradient of the loss $\mathcal{G}$ in Eq. 7, with $\mathbf{C}_X$ and $\mathbf{C}_Y$ being the normalized shortest-path distances defined in Eq. 26.

We hold $\mathbf{Q}_{\mathbf{T}^{(k)}}$ fixed during the inner iterations and solve

$$\min_{\substack{\mathbf{T}\geq 0\\ \mathbf{T}\mathbf{1}_n=\widehat{\mathbf{p}}}} \langle \mathbf{Q}_{\mathbf{T}^{(k)}}, \mathbf{T}\rangle + \varepsilon \sum_{i,j} T_{ij}(\log T_{ij} - 1) + \tau D_{\mathrm{KL}}(\mathbf{T}^\top \mathbf{1}_m \| \mathbf{r}). \tag{36}$$

At each outer iteration, the nonlinear structural GWD term is replaced by its first-order approximation around the current coupling $\mathbf{T}^{(k)}$. The entropy and KL terms are kept in their original form in the inner problem. This gives a semi-relaxed optimization: the scene marginal $\widehat{\mathbf{p}}$ is fixed exactly, while the template marginal is learned and softly regularized toward $\mathbf{r}$. The derivation here concerns this formulation without additional capacity or affine-moment constraints.

Generalized Sinkhorn updates. We now derive how the KL penalty modifies the usual Sinkhorn column scaling. The updates are summarized in the following lemma.

Lemma A.1. *Suppose $\mathbf{Q}_{\mathbf{T}^{(k)}}$ is finite, $\varepsilon > 0$, $\tau > 0$, and $\widehat{\mathbf{p}}$ and $\mathbf{r}$ are strictly positive probability vectors. Then Eq. 36 has a unique minimizer of the form*

$$\mathbf{T} = \mathrm{diag}(\mathbf{u})\mathbf{K}_{\mathbf{T}^{(k)}}\,\mathrm{diag}(\mathbf{v}), \tag{37}$$

where

$$\mathbf{u} = \frac{\widehat{\mathbf{p}}}{\mathbf{K}_{\mathbf{T}^{(k)}}\mathbf{v}}, \qquad \mathbf{v} = \left(\frac{\mathbf{r}}{\mathbf{K}_{\mathbf{T}^{(k)}}^\top \mathbf{u}}\right)^\theta, \qquad \theta = \frac{\tau}{\tau+\varepsilon}. \tag{38}$$

All divisions and powers are applied elementwise.

Proof. The feasible set is nonempty and compact, and the entropy term makes the inner objective strictly convex, so the minimizer is unique. Introducing a multiplier η_i for each fixed row sum, stationarity at a positive optimum gives

$$Q_{ij} + \varepsilon \log T_{ij} + \tau \log \frac{q_j}{r_j} + \eta_i = 0, \tag{39}$$

where $\mathbf{q} = \mathbf{T}^\top \mathbf{1}_m$ and $\mathbf{Q} = \mathbf{Q}_{\mathbf{T}^{(k)}}$. Hence

$$T_{ij} = u_i (\mathbf{K}_{\mathbf{T}^{(k)}})_{ij} \left(\frac{r_j}{q_j}\right)^{\tau/\varepsilon}, \tag{40}$$

with $u_i = \exp(-\eta_i/\varepsilon)$. Writing

$$v_j = \left(\frac{r_j}{q_j}\right)^{\tau/\varepsilon}$$

gives $\mathbf{T} = \mathrm{diag}(\mathbf{u})\mathbf{K}_{\mathbf{T}^{(k)}}\,\mathrm{diag}(\mathbf{v})$. Since

$$q_j = v_j (\mathbf{K}_{\mathbf{T}^{(k)}}^\top \mathbf{u})_j,$$

we obtain

$$v_j^{1+\tau/\varepsilon} = \left(\frac{r_j}{(\mathbf{K}_{\mathbf{T}^{(k)}}^\top \mathbf{u})_j}\right)^{\tau/\varepsilon}, \tag{41}$$

which yields the exponent $\theta = \tau/(\tau+\varepsilon)$ in Eq. 38. The fixed row marginal gives the stated update for $\mathbf{u}$. □

Starting from $\mathbf{v}^{(0)} = \mathbf{1}_n$, the inner iterations alternate

$$\mathbf{u}^{(\ell+1)} = \frac{\widehat{\mathbf{p}}}{\mathbf{K}_{\mathbf{T}^{(k)}}\mathbf{v}^{(\ell)}}, \qquad \mathbf{v}^{(\ell+1)} = \left(\frac{\mathbf{r}}{\mathbf{K}_{\mathbf{T}^{(k)}}^{\top}\mathbf{u}^{(\ell+1)}}\right)^{\theta}. \tag{42}$$

The row update enforces the fixed scene marginal, while the column update incorporates the soft prior on template visibility. The resulting template marginal is

$$\mathbf{q} = \mathbf{v} \odot \left(\mathbf{K}_{\mathbf{T}^{(k)}}^{\top}\mathbf{u}\right). \tag{43}$$

For $\tau = 0$, $\theta = 0$ and $\mathbf{v} = \mathbf{1}_n$, so the reference distribution has no effect. As τ increases, the learned marginal is pulled more strongly toward $\mathbf{r}$. In the extreme case, as $\tau \to \infty$, the update approaches the balanced scaling that fixes the template marginal to $\mathbf{r}$.

EGGroW computations. We next show that the generalized Sinkhorn updates above can be implemented within the same EGGroW framework used for the standard entropic problem. The KL penalty changes only the scaling updates and does not alter the two kernel–vector products required at each inner iteration. DET-EGGroW evaluates the structural matrix operations used to construct $\mathbf{Q}_{\mathbf{T}^{(k)}}$ using the deterministic methods of Sec. 3.1.

For APP-EGGroW, let the positive random-feature approximation of the Stage-II kernel be

$$\widehat{\mathbf{K}}_{\mathbf{T}^{(k)}} = \operatorname{diag}(\mathbf{s})\Phi_X^{\top}\Phi_Y \operatorname{diag}(\mathbf{t}). \tag{44}$$

The products are then evaluated as

$$\begin{aligned} \widehat{\mathbf{K}}_{\mathbf{T}^{(k)}}\mathbf{v} &= \mathbf{s} \odot \Phi_X^{\top}\big(\Phi_Y(\mathbf{t} \odot \mathbf{v})\big), \\ \widehat{\mathbf{K}}_{\mathbf{T}^{(k)}}^{\top}\mathbf{u} &= \mathbf{t} \odot \Phi_Y^{\top}\big(\Phi_X(\mathbf{s} \odot \mathbf{u})\big). \end{aligned} \tag{45}$$

The features and scale factors are constructed using the Stage-II coefficients λ_G and λ_M in place of α and $1 - \alpha$, respectively. The resulting approximate coupling remains factorized:

$$\widehat{\mathbf{T}} = \mathbf{L}^{\top}\mathbf{R}, \qquad \mathbf{L} = \Phi_X \operatorname{diag}(\mathbf{s} \odot \mathbf{u}), \qquad \mathbf{R} = \Phi_Y \operatorname{diag}(\mathbf{t} \odot \mathbf{v}). \tag{46}$$

Consistent with Sec. 3.2, we retain both scale factors. In particular, the template-side factor $\mathbf{t}$ affects the learned marginal $\mathbf{q}$ and therefore cannot be absorbed or discarded. The KL penalty only modifies the scaling updates and does not require additional random features or a larger factorization.

Numerical implementation. We implement the generalized Sinkhorn updates in a numerically stable form and explicitly enforce the fixed scene marginal before returning each inner solution. For the dense implementation, we evaluate the scaling updates in the log domain. Writing LSE for log-sum-exp,

$$\begin{aligned} \log u_i &= \log \widehat{p}_i - \operatorname{LSE}_j\left(-\frac{Q_{ij}}{\varepsilon} + \log v_j\right), \\ \log v_j &= \theta\left[\log r_j - \operatorname{LSE}_i\left(-\frac{Q_{ij}}{\varepsilon} + \log u_i\right)\right]. \end{aligned} \tag{47}$$

These are the log-domain forms of the updates in Eq. equation 38 and avoid numerical underflow or overflow in the exponential kernel. APP-EGGroW evaluates the same kernel–vector products through the factorization in Eq. equation 45, while retaining the associated scale factors.

During the inner iterations, we monitor the row-marginal residual and the change in the scaling variables. After the final column update, we recompute

$$\mathbf{u} = \frac{\widehat{\mathbf{p}}}{\mathbf{K}_{\mathbf{T}^{(k)}}\mathbf{v}}$$

so that the returned coupling satisfies the fixed scene marginal $\mathbf{T}\mathbf{1}_n = \widehat{\mathbf{p}}$. If $\widehat{p}_i = 0$, the corresponding row is omitted from the log-domain scaling iterations and restored as a zero row afterward. The resulting coupling defines the next outer iterate $\mathbf{T}^{(k+1)}$, which is used to recompute $\mathbf{D}_{\mathbf{T}^{(k+1)}}$ and construct the next Stage-II kernel.

B Additional Numerical Results

B.1 Random-feature ablation of APP-EGGroW

We compare five positive random-feature constructions for APP-EGGroW on FAUST $004 \to 007$ (Table 3). For $n \in \{1{,}000, 2{,}000\}$, each construction approximates one balanced Sinkhorn update at the dense GWD iterates $T^{(20)}$ and $T^{(50)}$, with $r = 512$, $\alpha = 1$, $\varepsilon = 0.05$, and centered diagonal conditioning. Note that these are dense GWD only, whereas in the Sec. 5.1 we approximated FGWD using APP-EGGroW instead.

Structural costs are full-mesh graph shortest paths with Euclidean edge weights, jointly scaled by the larger sampled 95th percentile. Source and target use matching vertex samples and uniform marginals. Sinkhorn uses tolerance 10^{-9} and at most 500 iterations; correspondences are obtained by row-wise maximization of the coupling.

Random-feature	$n = 1{,}000$		$n = 2{,}000$	
construction	Geodesic error ↓	PCK@5% ↑	Geodesic error ↓	PCK@5% ↑
IID PRF	0.310 ± 0.082	22.50 ± 3.16	0.465 ± 0.029	18.88 ± 3.82
OPRF-IID	0.325 ± 0.092	21.84 ± 2.57	0.457 ± 0.036	19.09 ± 4.13
Orthogonal PRF	0.421 ± 0.044	20.79 ± 2.71	0.430 ± 0.113	17.95 ± 1.78
Simplex PRF	0.422 ± 0.044	20.80 ± 2.70	0.430 ± 0.113	17.97 ± 1.77
OPRF-Orthogonal	0.424 ± 0.048	20.53 ± 2.75	0.433 ± 0.116	17.78 ± 1.87
Dense-kernel reference	0.290	25.15	0.358	22.58

Table 3: Random-feature ablation on FAUST. RF entries are mean ± sample standard deviation over five/three seeds at $n = 1{,}000/2{,}000$, after averaging the two frozen updates. Geodesic error is the mean raw target distance; PCK@5% is the percentage within 5% of the sampled target geodesic diameter. The dense reference averages the same states. OPRF denotes optimized positive random features.

Among RF variants, IID features achieve the lowest mean geodesic error and highest PCK at $n = 1{,}000$. At $n = 2{,}000$, orthogonal and simplex features give lower mean geodesic errors, whereas OPRF-IID achieves the highest PCK.

B.2 Pose Correspondence

To complement the pose correspondence experiments in Sec. 5.1, Fig. 5 presents additional qualitative results from source scan 004 to target scans 006 and 007. Similar to Fig. 1, transferring intrinsic source colors to each target visualizes how well each method preserves correspondence across changes in pose. The additional poses highlight the challenges introduced by articulation, particularly when distinct limbs become close in Euclidean space despite remaining distant along the mesh. We used $\varepsilon = 0.05$ and ran each experiment for 50 outer updates with 500 inner Sinkhorn iterations per update.

Regarding further experiment setup details, we normalized structural distances by the full-source off-diagonal geodesic 95th percentile, centered each mesh, and scaled both coordinate sets by the source RMS radius. Numerical correspondences were extracted through row-wise maximization of the coupling, with normalized errors measured relative to the full target diameter and shaded bands showing the observed minimum–maximum across runs. Moreover, we timed method-specific setup and solving, excluding shared calibration, input loading, and post-solve scoring.

B.3 Template Detection

Here, we provide specific details on the setup for template detection in Sec. 5.2. We used meshes from the Thingi10k dataset Zhou & Jacobson (2016) with the following IDs: `[977256, 104968, 73083, 718339, 38644, 44381, 134543, 65444, 39946, 985159, 263199, 931879, 42370, 69987, 71999, 112544, 98020, 53750, 1514903, 362964]`.

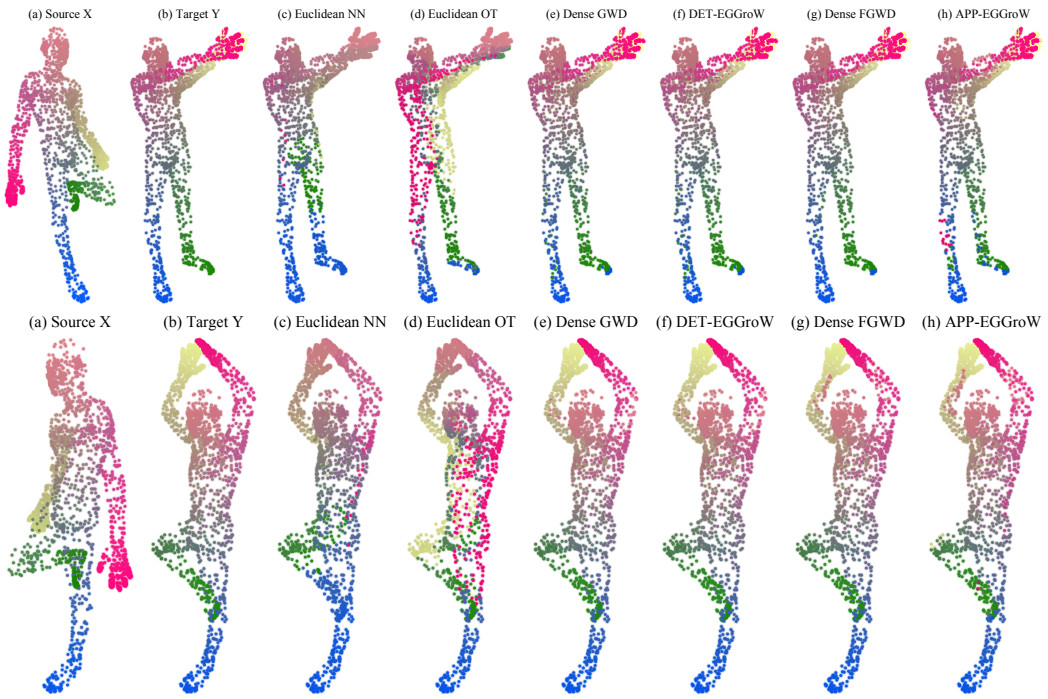

Figure 5: FAUST pose correspondence for (a) source scan 004 and (b) target scans 006 (top) and 007 (bottom). Colors are intrinsic source colors transferred to the target by each method (c)-(h).

We center each source mesh at the midpoint of its bounding box and normalize its bounding-box diagonal to one. This normalization remains fixed across support sizes and is also used for the target. For the sampled-point experiments, we draw points uniformly with respect to source surface area and use nested prefixes of the same sample sequence to vary resolution. Each support contains n points with uniform masses $a_i = b_i = 1/n$, and all methods receive identical inputs at each size.

We construct the target by applying a smooth nonrigid deformation, followed by a fixed rotation and translation, to the sampled source points. A random permutation removes their original ordering while preserving known material correspondences for evaluation. For shortest-path comparisons, we construct an undirected graph by taking the union of the directed 8-nearest-neighbor edges, weighted by Euclidean length. Disconnected components are joined using Euclidean minimum-spanning-tree bridges.

B.4 Limitations and Future Work

While our experiments cover a selected range of poses, object geometries, and scenes, a broader evaluation would help establish robustness across subjects and acquisition conditions. Template detection also depends on the quality of the observed geometry and initialization, particularly under clutter and occlusion. Future work will explore more diverse datasets, stronger geometric features, and adaptive approximation strategies to improve robustness and scalability for larger, more complex scenes.