FastJAM: a Fast Joint Alignment Model for Images

Omri Hirsch* **Ron Shapira Weber*** **Shira Ifergane** **Oren Freifeld**
The Faculty of Computer and Information Science, Ben Gurion University of the Negev (BGU), Israel
The Data Science Research Center, BGU
The School of Brain Sciences and Cognition, BGU

{omrihir,ronsha,shiraif}@post.bgu.ac.il
orenfr@bgu.ac.il

Abstract

Joint Alignment (JA) of images aims to align a collection of images into a unified coordinate frame, such that semantically-similar features appear at corresponding spatial locations. Most existing approaches often require long training times, large-capacity models, and extensive hyperparameter tuning. We introduce FastJAM, a rapid, graph-based method that drastically reduces the computational complexity of joint alignment tasks. FastJAM leverages pairwise matches computed by an off-the-shelf image matcher, together with a rapid nonparametric clustering, to construct a graph representing intra- and inter-image keypoint relations. A graph neural network propagates and aggregates these correspondences, efficiently predicting per-image homography parameters via image-level pooling. Utilizing an inverse-compositional loss, that eliminates the need for a regularization term over the predicted transformations (and thus also obviates the hyperparameter tuning associated with such terms), FastJAM performs image JA quickly and effectively. Experimental results on several benchmarks demonstrate that FastJAM achieves results better than existing modern JA methods in terms of alignment quality, while reducing computation time from hours or minutes to mere seconds. Our code is available at our project webpage, https://bgu-cs-vil.github.io/FastJAM/.

1 Introduction

Joint Alignment (JA) is the task of aligning a collection of images by estimating per-image spatial transformations such that, when applied, all images become geometrically consistent in a shared coordinate frame according to certain semantic or geometric criteria (see, *e.g.*, Figure 1). Unlike pairwise alignment, which aligns each image pair independently and often leads to error accumulation (*i.e.*, "drifting"), JA enforces a global agreement across the entire set, making JA particularly valuable for discovering shared structures between images or building a class atlas. However, achieving JA is inherently challenging: without supervision or a reference image, optimization methods frequently collapse into trivial or inconsistent solutions. Moreover, existing approaches typically require extensive computational resources, taking more than an hour [1, 2] to jointly align as few as 30 images. Recently, we proposed a method called **SpaceJAM** [3] that, partially by virtue of a new *inverse-compositional loss over dense feature maps*, significantly mitigated these computational issues, thereby solved the task in only a few minutes. Additionally, SpaceJAM set new state-of-the-art quantitative results. A natural question arises, however: is it possible to do even better in terms of both speed and performance? Fortunately, the answer is positive, as we show in the present paper.

*Equal Contribution

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

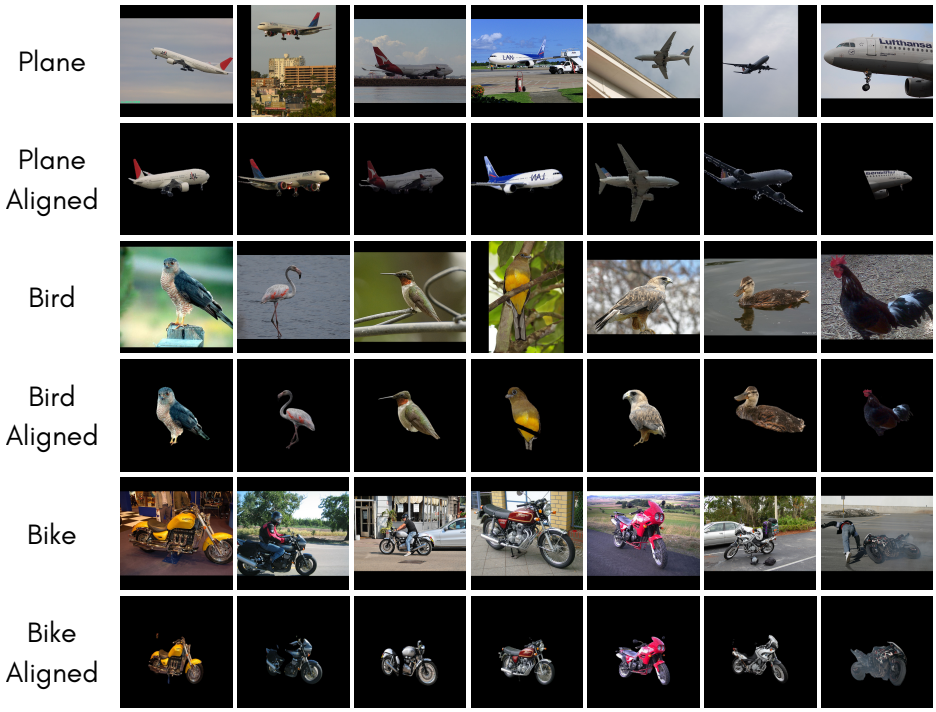

Figure 1: **Joint alignment with FastJAM.** Given a set of images of the same object, or of different objects from the same category (*e.g.*, motorbikes), our method aligns all images in seconds, compared to other methods (minutes [3] or hours [1, 2]).

Here we introduce an even more computationally-efficient method that solves the JA problem in under 50 seconds, dramatically outperforming prior approaches in terms of speed while maintaining, and in fact typically improving, alignment quality.

Traditionally, JA methods relied on classical approaches such as congealing [4, 5], which iteratively align each image towards the remaining set, or centroid-based methods that utilize a reference image or a latent template [3]. Classical techniques employed feature-based methods (*e.g.*, SIFT [6]), to establish keypoint (KP) correspondences between images. The rise of deep learning, particularly through Vision Transformers (ViTs) [7] and semantic feature extraction methods like DINO [8], has significantly advanced JA by providing richer representations that alleviate some challenges faced by traditional methods. However, even with ViT features, many difficulties persist, leading recent approaches to depend heavily on high-capacity, computationally-expensive models paired with extensive regularization [1, 2]. This reliance not only increases computational demands but also introduces complexity through the requirement of extensive hyperparameter (HP) tuning, ultimately resulting in methods that are slow and often brittle (as the HP tuning is usually dataset-specific).

Both congealing and atlas-based approaches typically rely on objective functions that fall into two main categories: geometric losses and semantic (feature-based) losses. In the geometric cases, one directly minimizes spatial discrepancies between corresponding KPs across images, leveraging explicit correspondence information. In contrast, semantic losses operate over dense feature representations and provide a smoother, globally-differentiable alignment without requiring explicit KPs. However, such dense (and typically high-dimensional) representations, often derived from high-capacity models like DINO [8], are computationally expensive when used within the JA optimization (even if the features themselves are kept frozen). Therefore, in this work we adopt the geometric loss paradigm, offering a sparse, lightweight, and scalable formulation that achieves typically higher alignment accuracy while significantly reducing the computational overhead.

Concretely, we introduce **FastJAM**, a graph-based JA framework that achieves fast and scalable alignment. Unlike prior methods that rely on dense feature maps and/or computationally-intensive optimization, FastJAM constructs a KP graph from pairwise correspondences using an off-the-shelf matcher, where nodes represent KPs and edges encode intra- and inter-image relationships. A

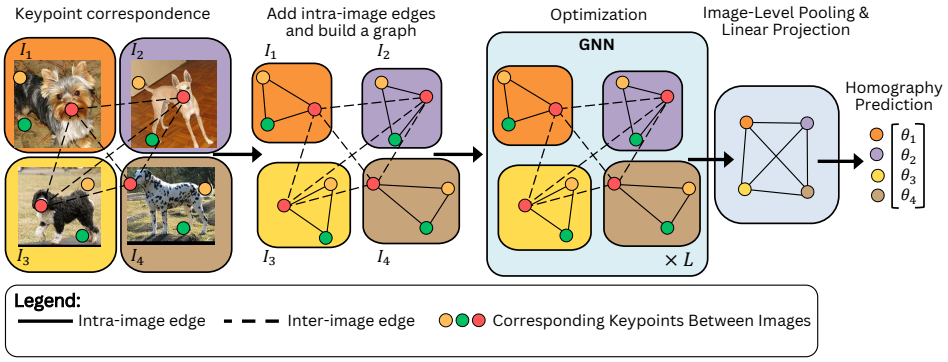

Figure 2: **Overview of the FastJAM architecture.** Given a set of images, we extract sparse keypoints (KPs) and pairwise correspondences using an off-the-shelf matcher (left; only red-dot matches are shown for clarity). A graph is built by linking KPs within each image (intra-image edges) and across matched pairs (inter-image edges). A GNN with L layers propagates alignment information through this graph (center). Image-level features are then obtained via mean pooling and used to predict per-image homography parameters ($\boldsymbol{\theta}_i$) for joint alignment.

Table 1: Comparison with recent JA methods on three SPair-71k categories [9]. Runtime is reported as average ± standard deviation in `hh:mm:ss` format.

Method	# Params	# Losses	#HP	Atlas-free	# Epochs	Runtime
Neural Congealing [1]	28.7M	8	8	✗	8000	01:18:30 ± 00:06:18
ASIC [2]	7.9M	4	5	✗	20000	01:06:38 ± 00:00:38
SpaceJAM [3]	0.016M	1	0	✓	700	00:06:00 ± 00:00:12
FastJAM (Ours)	0.13M	1	0	✓	600	**00:00:49** ± 00:00:04

Graph Neural Network (GNN) propagates alignment cues across the graph, and a readout layer produces image-level embeddings used to predict the homography parameters (as shown in Figure 2). Combined with a robust *inverse-compositional geometric loss*, FastJAM aligns an entire image set in under 50 seconds. Experiments on SPair-71k and CUB-200 show that FastJAM matches or exceeds the accuracy of contemporary JA methods while being significantly more efficient and orders of magnitude faster (see Table 1 for a comparison). Our key contributions are as follows.

1. We introduce FastJAM, a novel GNN-based framework for JA that significantly accelerates the alignment process (compared with existing methods) from hours/minutes to seconds.
2. FastJAM graph structure allows for the information from the entire image collection to propagate between images during optimization, unlike previous "image-by-image" approaches (including SpaceJAM), leading to improvements in JA quality.
3. The first Inverse-Compositional JA loss that is based on a sparse KP representation.

2 Related Work

Pairwise image alignment. Learning-based correspondence methods have substantially improved the accuracy and robustness of image matching. Sparse approaches like SuperPoint [10] and SuperGlue [11] detect KPs and compute context-aware matches using attention and GNNs, but might struggle in low-texture or repetitive regions due to their reliance on sparse detections. To overcome these limitations, dense methods such as LoFTR [12] compute pixel-wise matches using transformer-based architectures without requiring explicit KPs. Recent advances continue to close the gap between the sparse and dense paradigms. RoMa [13] combines DINOv2 features with hierarchical transformers for robust wide-baseline matching. DIFT [14] introduces efficient descriptor interpolation from diffusion models. Additional types of relevant dense features appeared in the works of Mariotti *et al*. [15], who propose spherical viewpoint maps that encode rich geometry, and Xu *et al*. [16], who incorporate directional priors to disambiguate symmetric cases. However, common issues with

dense matching approaches are that they can be computationally demanding and that they require post-processing because the geometry often breaks.

JA by feature matching. Classical JA methods utilize geometric transformations and handcrafted features. The idea of congealing [4, 5] is to iteratively align images by minimizing a global cost, typically entropy or least-squares of pixel values [17, 18, 5] or descriptors like SIFT [19, 20, 21]. Other approaches simultaneously cluster and align images to their class means [22, 23] or use template matching [24, 25, 26]. In any case, such methods are limited by the quality of the extracted features. Another classical approach models image sets as low-rank linear subspaces [27, 28, 29, 30].

Deep learning has substantially advanced image JA. Huang *et al*. [31] adapted congealing to CNN features, while Spatial Transformer Networks (STNs) [32] introduced a differentiable module for predicting spatial transformations, enabling end-to-end alignment learning. STNs have since been widely adopted in JA tasks, including congealing [33], atlas construction [34, 35, 36, 37, 38], joint clustering [39, 40], moving-camera background modeling [41, 42], and temporal synchronization of multiple videos [43]. STNs have also been combined with GANs [44] to generate high-quality canonical atlases [45, 46], albeit data demanding.

Recent works have increasingly adopted deep features as the basis for JA. DINO features [8], in particular, offer robust and semantically-rich representations well-suited for this task. Neural Congealing [1] employs test-time optimization to build class-specific atlases (e.g., birds) by aligning DINO features using rigid and non-rigid warps, predicted by a ResNet-based STN [47]. ASIC [2] utilizes DINO features, learning dense warps from input images to a canonical space through a U-Net architecture [48]. Both these methods are computationally intensive (often exceeding an hour for 30 images on an RTX 4090), require heavy regularization to avoid degenerate solutions, and are prone to instability. ASIC also typically produces fragmented or globally-incoherent alignments due to the challenges of dense warping. More recently, a previous work from our group introduced SpaceJAM [3], a more efficient solution using a lightweight ConvNet (CNN) and an inverse-compositional loss over refined DINO features, reducing runtime to a few minutes. However, its reliance on high-dimensional feature maps results in substantial memory overhead during the optimization process. In contrast, FastJAM combines sparse KP-based matching with a geometric loss, enabling much faster optimization and better scalability. FastJAM also differs from SpaceJAM in its architecture and the fact that the input to its neural net is based on the entire image collection, as opposed to SpaceJAM's single-image input.

JA by KP correspondence. Several methods tackle JA by explicitly leveraging KP correspondences across the image set and minimizing a geometric loss. Shokrollahi *et al*. [21] construct a similarity graph from KP matches to select optimal references for alignment. Safdarnejad *et al*. [49] propose a temporally-aware congealing method for video frames based on tracked KPs. FlowWeb [50] estimates dense correspondences across images and refines them jointly through graph-based optimization. ASIC [2] also incorporates an initial KP matching step, though its primary pipeline relies on dense warping. Dense matching is often computationally demanding and memory intensive, while sparse KPs offer a more efficient alternative. Building on recent successes of GNNs in pairwise alignment tasks [11, 51, 52], FastJAM capitalizes on this sparsity and introduces a GNN to aggregate alignment information across matched KPs. Combined with a fast inverse-compositional loss, this enables high-quality and regularization-free JA in seconds and with a low-memory footprint.

3 Method

In this section, we first formally introduce the JA problem (§ 3.1). We then detail how to construct a correspondence graph from pairwise image matches (§ 3.2), while in § 3.3 we detail the model architecture. Finally, we explain in § 3.4 how to perform image JA with FastJAM.

3.1 The Joint Alignment Problem and the Inverse-Compositional Framework

Given N images, $\mathcal{I} = (I_i)_{i=1}^N$, depicting different instances from the same semantic class (*e.g*., cars), the task is to facilitate JA by estimating a transformation $T^{\boldsymbol{\theta}_i} \in \mathcal{T}$ for each image such that the transformed images $(I_i \circ T^{\boldsymbol{\theta}_i})_{i=1}^N$ are spatially aligned in a shared coordinate frame $\mathcal{C}$. We assume a parametric family of transformations $\mathcal{T}$ (*e.g*., homographies), with $T^{\boldsymbol{\theta}_i}$ denoting the transformation

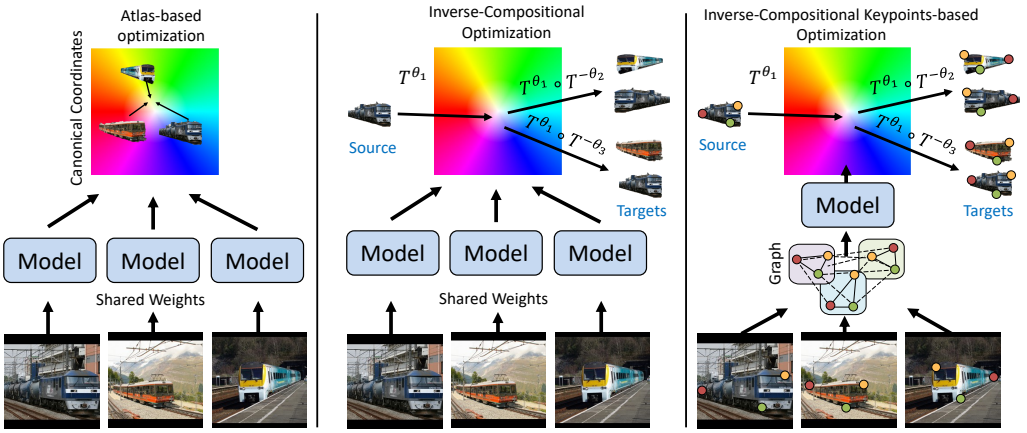

Figure 3: **Comparison of joint alignment frameworks.** *Left:* Atlas-based methods align each image independently to a canonical space $\mathcal{C}$ by minimizing variance. *Middle:* Existing inverse-compositional (IC) methods estimate $\mathcal{C}$ implicitly via relative transformations ($T^{\boldsymbol{\theta}_i} \circ T^{-\boldsymbol{\theta}_j}$), but process images independently. *Right:* **FastJAM** follows the IC paradigm, but differs from previous approaches in that 1) the loss is computed between KPs and 2) all images are processed simultaneously (the model process the entire KPs graph during its forward pass), allowing shared reasoning across all images.

for image I_i, parameterized by $\boldsymbol{\theta}_i$. Atlas-based approaches (*e.g.*, [1, 2]) usually optimize for a latent template, I_μ, jointly with the transformations. Formally, they solve

$$\underset{I_\mu, (T^{\boldsymbol{\theta}_i})_{i=1}^N \in \mathcal{T}}{\arg\min} \sum_{i=1}^{N} D(I_\mu, I_i \circ T^{\boldsymbol{\theta}_i}) + \mathcal{R}(T^{\boldsymbol{\theta}_i}; \lambda) \tag{1}$$

where D is a discrepancy measure (*e.g.*, the Euclidean distance), $\mathcal{R}(T^{\boldsymbol{\theta}_i}; \lambda)$ is a regularization term on the predicted transformations with HPs λ, and I_μ is the so-called canonical space or atlas. Due to its notion of centrality, I_μ is also known as the average or centroid image.

In contrast, FastJAM follows a congealing-inspired approach [4, 5] (particularly, Least-Squares (LS) Congealing [17, 18]) which avoids the need to maintain an explicit reference image, together with the modern Inverse Compositional (IC) approach we proposed in [3]. Concretely, the IC approach can be defined via the following loss:

$$\underset{(T^{\boldsymbol{\theta}_i})_{i=1}^N \in \mathcal{T}}{\arg\min} \sum_{i=1}^{N} \sum_{j:j\neq i} D(I_j, I_i \circ T^{\boldsymbol{\theta}_i} \circ T^{-\boldsymbol{\theta}_j}) \,. \tag{2}$$

Of note, the historical roots of the IC approach go back to the pre-DL era [18]. SpaceJAM [3], however, rather than optimizing over the warping of a single image at a time (as was done in traditional LS-congealing) simultaneously optimizes over all transformations.

The IC formulation indicates that the image collection is mapped to a shared space, since

$$I_j \approx I_i \circ T^{\boldsymbol{\theta}_i} \circ T^{-\boldsymbol{\theta}_j} \Leftrightarrow I_j \circ T^{\boldsymbol{\theta}_j} \approx I_i \circ T^{\boldsymbol{\theta}_i} \,. \tag{3}$$

As explained in [3], the IC approach obviates the need for using regularization terms. An important distinction between our work and [3] is that in [3] the optimization is based on the discrepancy between dense feature maps (*e.g.*, DINO features), while we adapt it to KPs and rely on geometric measure, as detailed below. Figure 3 illustrates the different JA approaches.

By design, and due to the inverse-compositional nature, IC losses are invariant to a single global homography. That is, for any $(T^{\boldsymbol{\theta}_i})_{i=1}^N$ and any additional transformation $T^{\boldsymbol{\theta}_0}$, the $(T^{\boldsymbol{\theta}_i} \circ T^{\boldsymbol{\theta}_0})_{i=1}^N$ transformations would give rise to the same value of the loss. The same phenomena happens in not only [3] (and, for slightly-different reasons, [38, 43]) but also many works on synchronization over groups (see, *e.g.*, [53, 54]). This is a feature, not a bug, as it simplifies the optimization considerably. Importantly: 1) While this means there are infinitely-many solutions, the implied N-fold joint correspondence is unique. 2) After the fact (*i.e.*, after the optimization is done), and for visualization purposes, for example, one can pick the value of $T^{\boldsymbol{\theta}_0}$ without affecting the quality of the

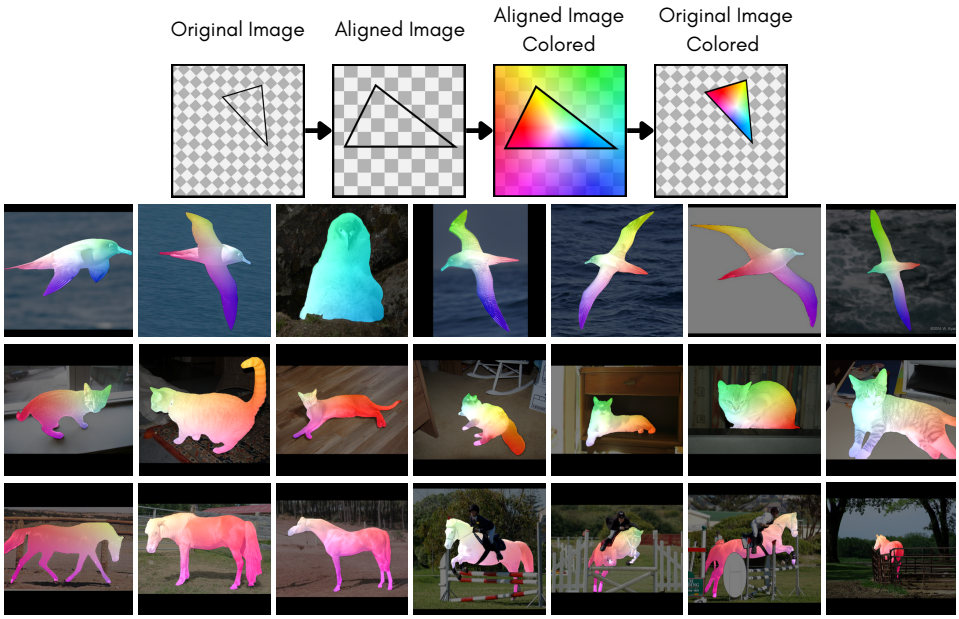

Figure 4: **Canonical Space Visualization.** We visualize the canonical space ($\mathcal{C}$) via a predefined RGB colormap. The first row shows an example of color projection from the canonical space onto a reference triangle. From the second row, we color each image I_i by applying its inverse transformation on $\mathcal{C}$ (*i.e.*, $\mathcal{C} \circ T^{-\boldsymbol{\theta}_i}$). FastJAM maps semantically similar regions to the same areas of $\mathcal{C}$, as shown by the consistent color mapping.

solution. Plausible choices include the inverse of the average homography (which can be computed in various ways, including the so-called Karcher mean [55, 56]) or the inverse of, say, $T^{\boldsymbol{\theta}_1}$. Note that the latter choice does *not* imply that the first image had any special importance in the optimization.

3.2 Graph Construction

The first stage of FastJAM involves constructing a sparse graph over KPs extracted from the image collection. This graph encodes both intra-image structure and inter-image correspondences and serves as the input to the GNN. The construction process consists of three steps, detailed below.

Object-centric region extraction. We follow [1, 2, 3] and extract object-centric masks for the image collection. We use Grounded-SAM [57], a combination of the Grounding DINO object detector and the Segment Anything Model (SAM) [8, 58]. Given a text prompt corresponding to the object category, Grounded-SAM produces a segmentation mask focusing on the object of interest. This mask is used to restrict the KP extraction and matching to the relevant region, improving robustness to background clutter and occlusions. Mask extraction takes ~ 0.4 seconds per image and can be parallelized over the GPU.

KP detection and matching. For each image I_i, we extract a set of sparse KPs $X_i = \{x_i^{(1)}, \ldots, x_i^{(M_i)}\}$ (where M_i is the number KPs in I_i) using an off-the-shelf image matcher inside the objects' mask. We use RoMa [13], but any matcher producing KP correspondences can be used. We chose RoMA due to its robustness and fast inference time (~ 0.3 seconds for an image pair on an RTX4090, which can be done in parallel across pairs). For each image pair (I_i, I_j), RoMa returns a set of KPs, (X_i, X_j), and correspondences $\mathcal{M}_{ij} \subset X_i \times X_j$ along with a confidence score for each match. To improve spatial coverage and reduce redundancy, we apply non-maximum suppression (NMS) over the matcher confidence scores using a 30×30 window, and retain the top-scoring points. We found that selecting as few as 10 KPs per image is sufficient for our framework.

Intra-image KP clustering. Consider three images, (I_1, I_2, I_3), and recall that RoMa is a *pairwise* KP extractor and matcher. Running RoMa on (I_1, I_2) produces 10 KPs on I_1 and 10 on I_2; running

it on (I_1, I_3) produces another 10 KPs on I_1 and 10 on I_3; and so on. Consequently, each image (e.g., I_1) accumulates multiple sets of KPs obtained from different pairwise runs. Importantly, these sets are not guaranteed to be identical, even though they all refer to the same image, leading to redundancy. For instance, in bird images, most of the $N-1$ KP sets extracted from I_1 are likely to include a KP near the beak tip, though at slightly different locations. To reduce this redundancy and merge semantically similar KPs within an image, we apply a fast, nonparametric clustering step. Specifically, we use Dinari and Freifeld's parallel DP-Means algorithm [59] (with `init_n`=3 and $\delta = 1$), a highly efficient variant of DP-Means [60], which itself generalizes K-Means to an unknown number of clusters. It takes ~ 0.13 seconds to cluster ~ 6000 points. After clustering, we discard the original KPs and retain only the cluster means as the representative intra-image KPs.

Graph definition. We define a single graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ over the entire image set. Each KP, $x_i^{(m)}$, is represented as a node $v \in \mathcal{V}$. We add two types of edges: (1) **Intra-image edges**: In each image I_i, we fully connect all KPs in X_i to model local spatial structure. (2) **Inter-image edges**: In each matched KP pair $(x_i^{(m)}, x_j^{(n)}) \in \mathcal{M}_{ij}$, we add an edge between the corresponding nodes. Each node $v \in \mathcal{V}$ is initialized with a vector $\boldsymbol{h}_v^{(0)}$ consisting of the KP's 2D coordinates such that $\boldsymbol{H}^{(0)}$ is the initial nodes coordinates matrix. In addition, each node is tagged with a categorical identifier indicating its source image, which is later used to perform image-level pooling. Unlike traditional graph-level readout layers that summarize the entire graph, FastJAM performs structured readout by pooling node embeddings per image. The edge structure is encoded as a binary adjacency matrix $\boldsymbol{A} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$, where $\boldsymbol{A}_{uv} = 1$ if there is an edge (intra- or inter-image) between nodes u and v. The resulting graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{H}^{(0)}, \boldsymbol{A})$ encodes both local geometric structure and cross-image semantic correspondence, and serves as input to a message-passing GNN that propagates alignment information throughout the image collection.

3.3 Model Architecture

Given the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{H}^{(0)}, \boldsymbol{A})$, our objective is to predict a transformation parameter vector $\boldsymbol{\theta}_i \in \mathbb{R}^8$ for each image I_i, representing its homography. We treat this as a structured regression task over node features: each node corresponds to a KP, and the GNN must propagate alignment-relevant information across the graph to produce per-image outputs. Formally, let $f(\mathcal{G}) = (\boldsymbol{\theta}_i)_{i=1}^N$ be a GNN that predicts the warping parameters from the graph. In our setting, where the goal is to regress image-specific transformations from KPs structured within a shared graph, it is essential to preserve the distinction between a node and its neighbors. GraphSAGE [61] achieves this by applying separate transformations to self and neighbor features, enabling more effective modeling of local asymmetries and node-specific roles which is relevant when propagating alignment cues across inter- and intra-image connections. While attention-based models such as GAT [62] offer expressive edge-aware aggregation, we found them to be empirically slower. In contrast, GraphSAGE provided a favorable balance of speed, stability, and alignment accuracy, making it well-suited for FastJAM's test-time optimization regime. The message protocol for GraphSAGE is

$$\boldsymbol{h}_v^{(l)} = \sigma\left(\boldsymbol{W}_1^{(l)}\boldsymbol{h}_v^{(l-1)} + \boldsymbol{W}_2^{(l)} \cdot \mathrm{mean}_{u\in\mathcal{N}(v)}\boldsymbol{h}_u^{(l-1)}\right) \tag{4}$$

where $\boldsymbol{h}_v^{(l)}$ is the embedding of node v at layer l, $\mathcal{N}(v)$ denotes the neighbors of v, $\boldsymbol{W}^{(l)}$ is a learnable weight matrix, and σ is a non-linear activation function. After $L = 5$ message-passing layers, we perform **per-image readout** via global average pooling over all nodes belonging to each image:

$$\boldsymbol{z}_i = \tfrac{1}{M_i}\sum\nolimits_{v\in\mathcal{V}_i} \boldsymbol{h}_v^{(L)} \tag{5}$$

where $\mathcal{V}_i \subset \mathcal{V}$ is the set of nodes from image I_i, and $\boldsymbol{z}_i \in \mathbb{R}^d$ is the resulting image-level embedding. Finally, we project the embedding of each image to the estimated homography parameters:

$$\boldsymbol{\theta}_i = \boldsymbol{z}_i^T\boldsymbol{W}_{\text{out}} \in \mathbb{R}^8\,. \tag{6}$$

3.4 FastJAM Joint Alignment

Lie-algebraic parameterization. To ensure matrix invertibility, which is essential for our IC formulation and for stable optimization [63, 3], we represent homographies using the Special Linear group $\mathrm{SL}(3)$ via a Lie-algebraic parameterization as was done in, *e.g.*, [64, 54, 3]. For details, see our supplemental material (**SupMat**).

Robust inverse-compositional KP loss. Our geometric loss builds upon the IC formulation introduced in [3], adapted to sparse KP correspondences. For each image pair (I_i, I_j), the forward warp from I_i is composed with the inverse warp from I_j (*i.e.*, $T^{\boldsymbol{\theta}_i} \circ T^{-\boldsymbol{\theta}_j}$). We penalize the discrepancy between each matched KP pair $(x_i^{(m)}, x_j^{(n)}) \in \mathcal{M}_{ij}$ after applying the IC transformation

$$\mathcal{L}_{\text{KP-IC}} = \sum_{i=1}^{N} \sum_{j \neq i} \sum_{(x_i^{(n)}, x_j^{(m)}) \in \mathcal{M}_{ij}} \rho_\sigma \left(\| x_j^{(m)} - x_i^{(n)} \circ T^{\boldsymbol{\theta}_i} \circ T^{-\boldsymbol{\theta}_j} \|_2 \right), \tag{7}$$

where $\rho_\sigma(z) = \frac{z^2}{z^2+\sigma^2}$ is the Geman-McClure robust loss function [65] with parameter σ.

This formulation allows alignment to be computed at the original KP locations without any regularization term on the warps or the need to render warped images via expensive interpolation. Compared to dense alignment over high-dimensional DINO feature maps, our sparse formulation is both significantly more efficient and more robust to missing KPs, wrong matches, and outliers.

Handling reflections. We follow [3] and explicitly check for flips every K epochs during optimization (where $K = 100$) and compute the gradient and update the model's weight only for the best configuration. We have found that only checking for horizontal flips is sufficient. This ensures that flipped images can still participate in alignment without requiring a reflection-aware parameterization.

Implementation details. All experiments were conducted on a single NVIDIA RTX 4090 GPU with 24GB of memory. We optimize FastJAM for 600 epochs using Adam [66] with a Geman-McClure robustness parameter $\sigma = 0.25$. We use pretrained Grounding-SAM [57] and RoMa [13] with the default HP once, before starting the optimization. For more details, please see our **SupMat**.

Limitations. Our main limitation is the reliance on an external image matcher to generate initial KP correspondences. While modern matchers like RoMa provide high-quality matches in many scenarios, the overall alignment quality depends on the accuracy of these correspondences. In addition, FastJAM models geometric transformations using homographies, which may be insufficient in cases involving strong non-planar deformations. Extending the model to support more expressive transformation families remains a direction for future work.

4 Results

Datasets, evaluation metrics, and baselines. We evaluate FastJAM under a test-time optimization setting, where the model is optimized independently on each image collection. We use two benchmark datasets: **SPair-71k** [9] and **CUB-200** [67] (classes and subsets). SPair-71k's test set comprises 18 object categories, each with $\sim$30 images, with annotated KPs and large intra-class variation. We report both per-category performance and average results across all categories. Following prior works, we evaluate on the first 3 categories of CUB-200 test set, each containing $\sim$30 images as well. We use the **Percentage of Correct Keypoints (PCK)** as the evaluation metric. A predicted KP is deemed correct if it falls within a normalized distance threshold α of the ground-truth location. We report mean PCK across all KPs and categories, with $\alpha = 0.1$. We average the results over 3 runs.

As in prior work [2, 3], we compare FastJAM to several baselines. Neural Best Buddies (NBB) [68] aligns image pairs using mutual nearest neighbors with Moving Least Squares warping [69], using VGG (**VGG-MLS**) or DINO (**DINO-MLS**) features. **DINO-NN** performs dense nearest-neighbor matching. **GANgealing** [46] uses GANs but is limited to seen categories. **Neural Congealing** [1] builds an explicit atlas but requires hyperparameter tuning and reported results on only three SPair-71k classes. **ASIC** [2] predicts dense warps to a canonical space, and **SpaceJAM** [3] applies an IC loss over DINO features. FastJAM instead uses sparse KPs and a graph-based model, enabling faster and more scalable alignment. We cite the results reported in [2, 3].

4.1 Qualitative Results

We illustrate the qualitative performance of FastJAM in Figure 1, Figure 4, and Figure 5. Given a collection of category-level images (*e.g.*, birds), FastJAM aligns all images within seconds, producing visually coherent and semantically consistent outputs across instances. To interpret the alignment, we

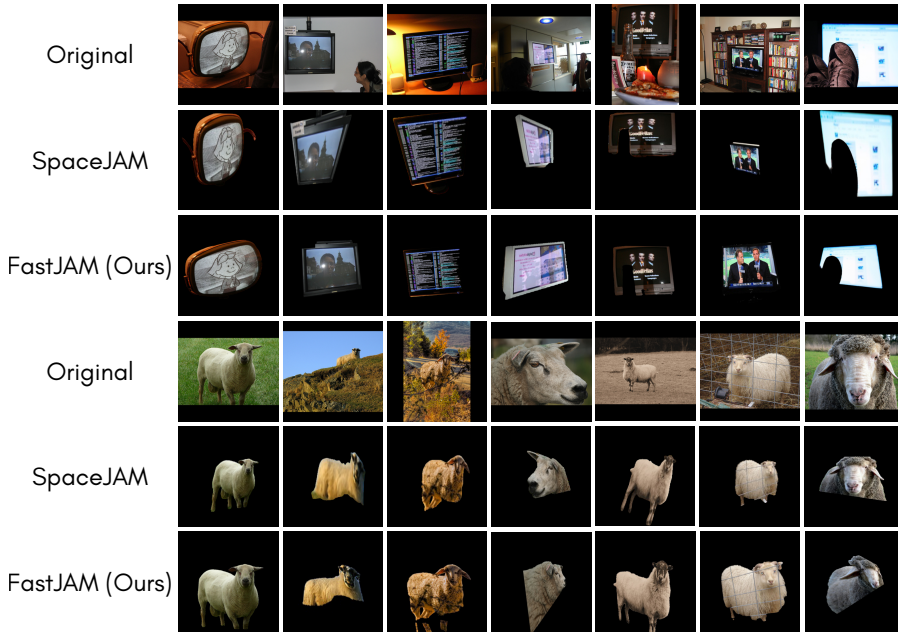

Figure 5: **JA Visual Comparison.** We compare FastJAM with SpaceJAM [3] on both rigid (TV) and non-rigid (Sheep) classes. In both cases, FastJAM alignment is visually better, where the improvement is particularly noticeable for close-up images, such as the middle or rightmost sheep.

Table 2: **SPair-71k results:** PCK@0.10 on the test set. Among test-time optimization (TTO) methods, the best is in **bold**, second-best is underlined. ($\star$) Denotes use of a reference image. ($-$) Indicates missing results. (†) Marks non-TTO methods.

Method	Aero	Bike	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Dog	Horse	Motor	Person	Plant	Sheep	Train	TV	**All**
GANgealing [46]	-	37.5†	-	-	-	-	-	67.0	-	-	23.1	-	-	-	-	-	-	57.9	-
VGG+MLS [68]	29.5	22.7	61.9	26.5	20.6	25.4	14.1	23.7	14.2	27.6	30.0	29.1	24.7	27.4	19.1	19.3	24.4	22.6	27.4
DINO+MLS [68, 70]	49.7	20.9	63.9	19.1	32.5	27.6	22.4	48.9	14.0	36.9	39.0	30.1	21.7	41.1	17.1	18.1	35.9	21.4	31.1
DINO+NN [71]	57.2	24.1	67.4	24.5	26.8	29.0	27.1	52.1	15.7	42.4	43.3	30.1	23.2	40.7	16.6	24.1	31.0	24.9	35.0
NeuCongeal [1]	-	29.1⋆	-	-	-	-	-	53.3	-	-	35.2	-	-	-	-	-	-	-	-
ASIC [2]	57.9	25.2	**68.1**	24.7	35.4	28.4	30.9	54.8	21.6	45.0	**47.2**	39.9	26.2	48.8	14.5	24.5	49.0	24.6	37.0
SpaceJAM (ViT-L) [3]	53.6	**53.5**	45.4	**47.5**	**71.0**	54.0	46.0	**66.0**	25.8	48.6	28.5	47.6	54.0	**50.7**	34.0	09.0	71.8	15.4	45.7
FastJAM (Ours)	**64.4**	43.3	60.0	29.6	58.4	**66.8**	**56.5**	63.7	**32.0**	**49.2**	40.8	**53.7**	**62.8**	49.1	**42.9**	**33.4**	**76.2**	**71.2**	**53.0**

visualize the canonical space $\mathcal{C}$ using a fixed RGB colormap. Each image I_i is colored by applying the inverse of its predicted transformation to $\mathcal{C}$, *i.e.*, $\mathcal{C} \circ T^{-\boldsymbol{\theta}_i}$. As shown in Figure 4, semantically similar regions (*e.g.*, heads, wings, tails) align to consistent areas in $\mathcal{C}$, indicating robust JA across pose and appearance. As shown in Figure 5 in comparison to [3], FastJAM alignment is visually better. Additional examples and comparisons are available in the **SupMat**.

4.2 Quantitative Results

Table 2 reports alignment accuracy on the SPair-71k test set, measured by mean PCK@0.10 across 18 object categories. FastJAM achieves the best overall performance with an average PCK of **53.0**, outperforming all competing methods. It ranks first in 11 categories and second in 6 others, with comparable performance across a wide range of object classes and viewpoints. Compared to [1, 2], FastJAM offers on-par or better accuracy while being significantly faster and more memory-efficient, validating the benefits of its sparse, graph-based formulation. While FastJAM performs competitively across most categories, we observe reduced performance on highly-symmetric objects such as *bicycles*. These cases pose inherent challenges due to visual ambiguity, where the initial matcher struggles to disambiguate symmetric parts (*e.g.*, right versus left). In such scenarios, FastJAM can propagate incorrect correspondences. Addressing this remains an interesting direction for future work. Results for the CUB-200 dataset are reported in Table 3 where FastJAM outperforms SpaceJAM and achieves comparable results to ASIC [2] (in a fraction of the computation time) on the categories benchmark and the best results across subsets (results were not reported for [2]).

Table 3: A comparison on CUB-200.

Method	CUB-200 (first 3 cate.)	**Method**	CUB-200 (Subsets)
VGG+MLS [68]	25.8	-	-
DINO+MLS [70, 68]	67.0	-	-
DINO+NN [71]	68.3	GANgealing [46]	56.8
ASIC [2]	**75.9**	NeuCongeal [1]	63.6
SpaceJAM [3]	69.6	SpaceJAM [3]	69.9
FastJAM (Ours)	75.3	FastJAM (Ours)	**73.6**

Table 4: Dense vs. KP warping runtime analysis [sec].

Loss	$B_{\max}$	N_{points}	D	Grid warping	Interpolation	Fwd. +Back	Total
Dense	10	70756	2	1.28	0.13	2.82	8.46
Dense	10	70756	25	1.29	0.77	4.12	12.36
KPs	30	16	2	0.18	———	0.36	0.36
KPs	30	16	25	0.17	———	0.34	0.34

Table 5: Ablation Study.

Ablation	CUB-200 (first 3 cate.)	SPair-71k
LoFTR [12] (No RoMa)	33.8	17.0
Linear projection	33.2	14.0
MLP (per-image, no graph)	73.5	47.8
Homography optimization (no deep net)	74.1	48.3
L2 loss (No Geman-McClure function)	61.0	33.9
No masks	74.8	41.8
No non-maximum suppression	74.3	47.2
No Lie Group	75.0	49.3
No intra-image edges	74.7	50.0
GNN Backbones		
GCN [72] (53 secs)	73.2	47.4
GAT [62] (66 secs)	75.0	49.9
GraphSAGE [61] (49 secs)	**75.3**	**53.0**

4.3 Runtime Analysis

Runtime comparison. We evaluate the computational efficiency of FastJAM by comparing its runtime against: NeuCongeal [1]; ASIC [2]; SpaceJAM [3]. As shown in Table 1, FastJAM achieves over an **order-of-magnitude speedup** (measured over three SPair-71k categories), aligning image collections in **under ∼50 seconds**, compared to 5–6 minutes for SpaceJAM and over an hour for ASIC and NeuCongeal. The reported runtime includes preprocessing (*i.e.*, pairwise matches). We also compare FastJAM and SpaceJAM on an increasing number of images ($N = 10$ to 100). The full experimental setup is available in our **SupMat**.

Dense vs. KP warping. We analyze the individual warping components of both methods on a set of 30 images (see Table 4, all reported runtime in this table are in seconds) for one epoch. We evaluate how the number of points (N_{points}) in the coordinate grid and the feature dimension (D) affect the overall warping time. For dense matching, we set $N_{\text{points}} = 266 \times 266 = 70{,}756$ (the image resolution used in SpaceJAM) and $D = 25$, corresponding to the feature dimension on which the loss is computed. For FastJAM, we set $N_{\text{points}} = 16$ and $D = 2$ (*i.e.*, 8 KPs in 2D). For completeness, we also evaluate $D \in \{2, 25\}$ for both methods. We measure the time required for (*i*) grid warping, (*ii*) interpolation (used only in dense warping), (*iii*) forward and backward warping for the IC loss (*Fwd + Back*), and the total runtime over all batches in a single epoch. For dense warping (*e.g.*, SpaceJAM), the maximum batch size is $B_{\max} = 10$, resulting in a total runtime of $3 \times$ (Fwd+Back). The key observations are: (1) reducing N_{points} is crucial for achieving fast warping, and (2) avoiding interpolation further accelerates computation, making the runtime largely independent of D.

4.4 Ablation Study

Table 5 summarizes our ablation study on CUB-200 (3 categories) and SPair-71k. Replacing RoMa with LoFTR [12] significantly reduces performance, underscoring LoFTR's limitations in cross-instance correspondence. Substituting the Geman–McClure loss [65] with an ℓ_2 loss or removing the object mask also causes notable accuracy drops, confirming the importance of robust error modeling and spatial masking. NMS and intra-image edges provide additional gains. Although removing the Lie-algebraic parameterization has little effect on accuracy, it ensures warp invertibility, essential for the IC loss, as without it about 2% of runs fail due to non-invertible matrices, whereas using it eliminates such failures entirely. Replacing the GNN with a linear projection caused a substantial performance drop, while MLP-based and direct homography optimization models also reduced accuracy, though less severely. Among GNN backbones, GraphSAGE [61] outperforms both GCN [72] and GAT [62] in accuracy and runtime.

5 Conclusion

We introduced **FastJAM**, a graph-based framework for fast and scalable image JA. By leveraging sparse KP correspondences and a lightweight GNN architecture, FastJAM propagates alignment cues across image collections and regresses per-image transformations. Our method achieves state-of-the-art alignment quality while significantly reducing runtime and memory usage.

Acknowledgments

This work was supported by the Lynn and William Frankel Center at BGU CS, by the Israeli Council for Higher Education via the BGU Data Science Research Center, RSW's work was supported by the Kreitman School of Advanced Graduate Studies. Both OH and SI were also supported by the VATAT National excellence scholarship for MSc students in AI and Data Science.

References

[1] Dolev Ofri-Amar, Michal Geyer, Yoni Kasten, and Tali Dekel. Neural congealing: Aligning images to a joint semantic atlas. In *CVPR*, 2023. 1, 2, 3, 4, 5, 6, 8, 9, 10

[2] Kamal Gupta, Varun Jampani, Carlos Esteves, Abhinav Shrivastava, Ameesh Makadia, Noah Snavely, and Abhishek Kar. ASIC: Aligning sparse in-the-wild image collections. In *ICCV*, 2023. 1, 2, 3, 4, 5, 6, 8, 9, 10

[3] Nir Barel, Ron Shapira Weber, Nir Mualem, Shahaf E Finder, and Oren Freifeld. SpaceJAM: a lightweight and regularization-free method for fast joint alignment of images. In *European Conference on Computer Vision*, pages 180–197. Springer, 2024. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

[4] Erik G Miller, Nicholas E Matsakis, and Paul A Viola. Learning from one example through shared densities on transforms. In *CVPR*. IEEE, 2000. 2, 4, 5

[5] Erik G Learned-Miller. Data driven image models through continuous joint alignment. *IEEE TPAMI*, 2006. 2, 4, 5

[6] David G Lowe. Object recognition from local scale-invariant features. In *ICCV*. IEEE, 1999. 2

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[8] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 2, 4, 6

[9] Juhong Min, Jongmin Lee, Jean Ponce, and Minsu Cho. Spair-71k: A large-scale benchmark for semantic correspondence. *arXiv preprint arXiv:1908.10543*, 2019. 3, 8

[10] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. 3

[11] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 3, 4

[12] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8922–8931, 2021. 3, 10

[13] Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Roma: Robust dense feature matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19790–19800, 2024. 3, 6, 8

[14] Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. *Advances in Neural Information Processing Systems*, 36:1363–1389, 2023. 3

[15] Octave Mariotti, Oisin Mac Aodha, and Hakan Bilen. Improving semantic correspondence with viewpoint-guided spherical maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19521–19530, 2024. 3

[16] Junyi Zhang, Charles Herrmann, Junhwa Hur, Eric Chen, Varun Jampani, Deqing Sun, and Ming-Hsuan Yang. Telling left from right: Identifying geometry-aware semantic correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3076–3085, 2024. 3

[17] Mark Cox, Sridha Sridharan, Simon Lucey, and Jeffrey Cohn. Least squares congealing for unsupervised alignment of images. In *CVPR*, 2008. 4, 5

[18] Mark Cox, Sridha Sridharan, Simon Lucey, and Jeffrey Cohn. Least-squares congealing for large numbers of images. In *ICCV*. IEEE, 2009. 4, 5

[19] Gary B Huang, Vidit Jain, and Erik Learned-Miller. Unsupervised joint alignment of complex images. In *ICCV*. IEEE, 2007. 4

[20] Wen-Yan Lin, Linlin Liu, Yasuyuki Matsushita, Kok-Lim Low, and Siying Liu. Aligning images in the wild. In *CVPR*. IEEE, 2012. 4

[21] Fatemeh Shokrollahi Yancheshmeh, Ke Chen, and Joni-Kristian Kamarainen. Unsupervised visual alignment with similarity graphs. In *CVPR*, 2015. 4

[22] Xiaoming Liu, Yan Tong, and Frederick W Wheeler. Simultaneous alignment and clustering for an image ensemble. In *ICCV*. IEEE, 2009. 4

[23] Marwan A Mattar, Allen R Hanson, and Erik G Learned-Miller. Unsupervised joint alignment and clustering using bayesian nonparametrics. *arXiv preprint arXiv:1210.4892*, 2012. 4

[24] Anil K. Jain, Yu Zhong, and Sridhar Lakshmanan. Object matching using deformable templates. *IEEE TPAMI*, 1996. 4

[25] Dariu M Gavrila. Multi-feature hierarchical template matching using distance transforms. In *ICPR*. IEEE, 1998. 4

[26] Pedro F Felzenszwalb and Joshua D Schwartz. Hierarchical matching of deformable shapes. In *CVPR*, pages 1–8. IEEE, 2007. 4

[27] Ira Kemelmacher-Shlizerman and Steven M Seitz. Collection flow. In *CVPR*. IEEE, 2012. 4

[28] Jun He, Dejiao Zhang, Laura Balzano, and Tao Tao. Iterative grassmannian optimization for robust image alignment. *Image and Vision Computing*, 2014. 4

[29] Xiaoqin Zhang, Di Wang, Zhengyuan Zhou, and Yi Ma. Robust low-rank tensor recovery with rectification and alignment. *IEEE TPAMI*, 2019. 4

[30] Yigang Peng, Arvind Ganesh, John Wright, Wenli Xu, and Yi Ma. Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *IEEE TPAMI*, 2012. 4

[31] Gary Huang, Marwan Mattar, Honglak Lee, and Erik G Learned-Miller. Learning to align from scratch. In *NeurIPS*, 2012. 4

[32] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *NeurIPS*, 2015. 4

[33] Roberto Annunziata, Christos Sagonas, and Jacques Cali. Jointly aligning millions of images with deep penalised reconstruction congealing. In *ICCV*, 2019. 4

[34] Chen-Hsuan Lin and Simon Lucey. Inverse compositional spatial transformer networks. In *CVPR*, 2017. 4

[35] Adrian Dalca, Marianne Rakic, John Guttag, and Mert Sabuncu. Learning conditional deformable templates with convolutional networks. *NeurIPS*, 2019. 4

[36] Ron Shapira Weber, Matan Eyal, Nicki Skafte Detlefsen, Oren Shriki, and Oren Freifeld. Diffeomorphic temporal alignment nets. In *NeurIPS*, 2019. 4

[37] Matthew Sinclair, Andreas Schuh, Karl Hahn, Kersten Petersen, Ying Bai, James Batten, Michiel Schaap, and Ben Glocker. Atlas-istn: joint segmentation, registration and atlas construction with image-and-spatial transformer networks. *Medical Image Analysis*, 2022. 4

[38] Ron Shapira Weber and Oren Freifeld. Regularization-free diffeomorphic temporal alignment nets. In *ICML*. PMLR, 2023. 4, 5

[39] Tom Monnier, Thibault Groueix, and Mathieu Aubry. Deep transformation-invariant clustering. *NeurIPS*, 2020. 4

[40] Romain Loiseau, Tom Monnier, Mathieu Aubry, and Loïc Landrieu. Representing shape collections with alignment-aware linear models. In *3DV*. IEEE, 2021. 4

[41] Irit Chelly, Vlad Winter, Dor Litvak, David Rosen, and Oren Freifeld. JA-POLS: a moving-camera background model via joint alignment and partially-overlapping local subspaces. In *CVPR*, 2020. 4

[42] Guy Erez, Ron Shapira Weber, and Oren Freifeld. A deep moving-camera background model. In *ECCV*. Springer, 2022. 4

[43] Avihai Naaman, Ron Shapira Weber, and Oren Freifeld. Synchronization of multiple videos. In *ICCV*, 2025. 4, 5

[44] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS*, 2014. 4

[45] Jiteng Mu, Shalini De Mello, Zhiding Yu, Nuno Vasconcelos, Xiaolong Wang, Jan Kautz, and Sifei Liu. Coordgan: Self-supervised dense correspondences emerge from gans. In *CVPR*, 2022. 4

[46] William Peebles, Jun-Yan Zhu, Richard Zhang, Antonio Torralba, Alexei A Efros, and Eli Shechtman. Gan-supervised dense visual alignment. In *CVPR*, 2022. 4, 8, 9, 10

[47] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*. Springer, 2016. 4

[48] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*. Springer, 2015. 4

[49] S Morteza Safdarnejad, Yousef Atoum, and Xiaoming Liu. Temporally robust global motion compensation by keypoint-based congealing. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI 14*, pages 101–119. Springer, 2016. 4

[50] Tinghui Zhou, Yong Jae Lee, Stella X Yu, and Alyosha A Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1191–1200, 2015. 4

[51] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. Lightglue: Local feature matching at light speed. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17627–17638, 2023. 4

[52] Christiano Gava, Vishal Mukunda, Tewodros Habtegebrial, Federico Raue, Sebastian Palacio, and Andreas Dengel. Sphereglue: Learning keypoint matching on high resolution spherical images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6134–6144, 2023. 4

[53] David M Rosen, Luca Carlone, Afonso S Bandeira, and John J Leonard. SE-Sync: A certifiably correct algorithm for synchronization over the special Euclidean group. *The International Journal of Robotics Research*, 2019. 5

[54] Federica Arrigoni and Andrea Fusiello. Synchronization problems in computer vision with closed-form solutions. *International Journal of Computer Vision*, 2020. 5, 7

[55] X. Pennec. Probabilities and statistics on Riemannian manifolds: Basic tools for geometric measurements. In *NSIP*, pages 194–198, 1999. 6

[56] Hermann Karcher. Riemannian center of mass and so called karcher mean. *arXiv preprint arXiv:1407.2087*, 2014. 6

[57] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024. 6, 8

[58] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 6

[59] Or Dinari and Oren Freifeld. Revisiting DP-means: fast scalable algorithms via parallelism and delayed cluster creation. In *Uncertainty in Artificial Intelligence*. PMLR, 2022. 7

[60] Brian Kulis and Michael I Jordan. Revisiting k-means: New algorithms via bayesian nonparametrics. *arXiv preprint arXiv:1111.0352*, 2011. 7

[61] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017. 7, 10

[62] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. Graph attention networks. *stat*, 1050(20):10–48550, 2017. 7, 10

[63] Nicki Skafte Detlefsen, Oren Freifeld, and Søren Hauberg. Deep diffeomorphic transformer networks. In *CVPR*, 2018. 7

[64] Christopher Mei, Selim Benhimane, Ezio Malis, and Patrick Rives. Homography-based tracking for central catadioptric cameras. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 669–674. IEEE, 2006. 7

[65] Stuart Geman and Donald E McClure. Statistical methods for tomographic image reconstruction. In *BISI*, 1987. 8, 10

[66] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, 2014. 8

[67] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset, 2011. 8

[68] Kfir Aberman, Jing Liao, Mingyi Shi, Dani Lischinski, Baoquan Chen, and Daniel Cohen-Or. Neural best-buddies: Sparse cross-domain correspondence. *ACM TOG*, 2018. 8, 9, 10

[69] Scott Schaefer, Travis McPhail, and Joe Warren. Image deformation using moving least squares. In *ACM SIGGRAPH 2006 Papers*, pages 533–540. ACM, 2006. 8

[70] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020. 9, 10

[71] Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep vit features as dense visual descriptors. In *ECCV Workshops*, 2022. 9, 10

[72] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. 10

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: All the claims made in the abstract are backed in the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In section § 3.4, we elaborate about the limitations of FastJAM.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: there are no theoretical results in the paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Most if not all needed information about reproducibility is given either in the paper itself, in the Supplementary or in the future to be published code.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: As described in the abstract, the code would be released upon acceptance. The datasets we used are publicly available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We describe all the details either in the paper itself or in the supplementary.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Due to the substantial computational cost associated with running all competing methods, it was not feasible to perform multiple trials necessary for estimating variance or reporting error bars. On our results, we averaged our experiments on 3 different seed runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The experiments where maybe on internal cluster, GPU RTX4090, we mention it where needed. Memory size is also indicated.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: All the data we used is publicly available.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The method presented is a generic tool for aligning images and is not directly tied to a specific application. While it may enable both beneficial and potentially harmful uses (e.g., in surveillance or misinformation), the paper does not explore these societal implications.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We did not found high risk misuse for our method.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Any credit or citation needed was provided in the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: All needed documentation would be provided with the code upon acceptance.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: the paper did not involve crowd-sourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowd-sourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The usage of LLM, as declared, was mainly for writing and editing.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

FastJAM: a Fast Joint Alignment Model for Images

Supplemental Material

Contents

This document contains the following:

Additionally, key notational conventions used throughout the main paper and this document are summarized in Table 6.

A Additional Visual Comparisons

To qualitatively assess alignment quality, we compare FastJAM against existing joint alignment methods, including SpaceJAM and ASIC. As illustrated in Figure 6, FastJAM produces more coherent and natural-looking alignments, particularly in challenging cases involving pose variation. Unlike ASIC, which applies dense warping and often introduces distortions, FastJAM preserves global structure by relying on sparse keypoints and homographic transformations.

B Additional Visualizations

This section provides supplementary qualitative results that further demonstrate the alignment capabilities of FastJAM across various categories and settings.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

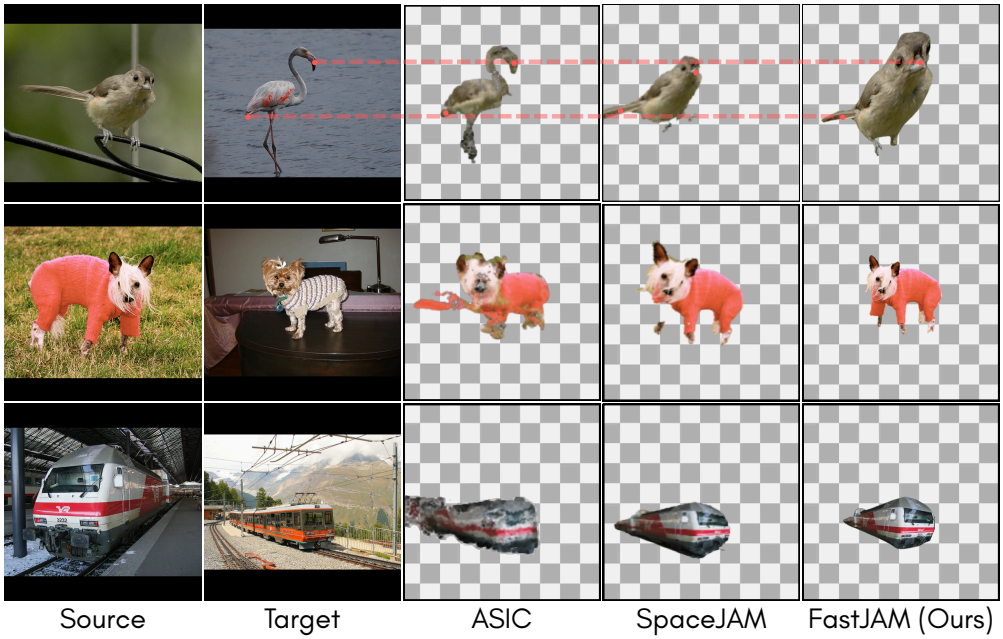

Figure 6: **Qualitative Comparison of Pairwise Alignment Methods.** Visual comparison of joint alignment results across ASIC [1], SpaceJAM [2], and FastJAM (ours) on several image pairs. ASIC, which relies on dense warping fields, often introduces spatial distortions and unrealistic deformations in low-texture or structured regions. In contrast, SpaceJAM and FastJAM apply global homographies, resulting in more coherent transformations. Notably, in the top row (bird), FastJAM produces the most geometrically consistent alignment, as evidenced by the parallel lines and precise correspondence of semantically meaningful points such as the beak and tail.

B.1 Additional Joint Alignment on CUB Dataset

Figure 7 presents further joint alignment results on additional classes from the CUB-200 dataset. For each class, the original input images are shown in the top row, while the bottom row displays their aligned counterparts. The outputs demonstrate FastJAM's ability to handle fine-grained categories and produce visually coherent canonical views across varying poses and appearances.

B.2 Pairwise Alignment

We visualize full pairwise alignment grids for two representative categories: "aeroplane" and "horse." As shown in Figure 8 and Figure 9, each grid displays how the source images (rows) are aligned to the target images (columns) using the estimated inverse-compositional warps. The diagonal entries, which correspond to self-alignments (i.e., identity transformations), are highlighted with a purple dashed-line frame. These grids illustrate FastJAM's ability to produce consistent, symmetric mappings across image pairs and maintain semantic structure throughout the alignment process.

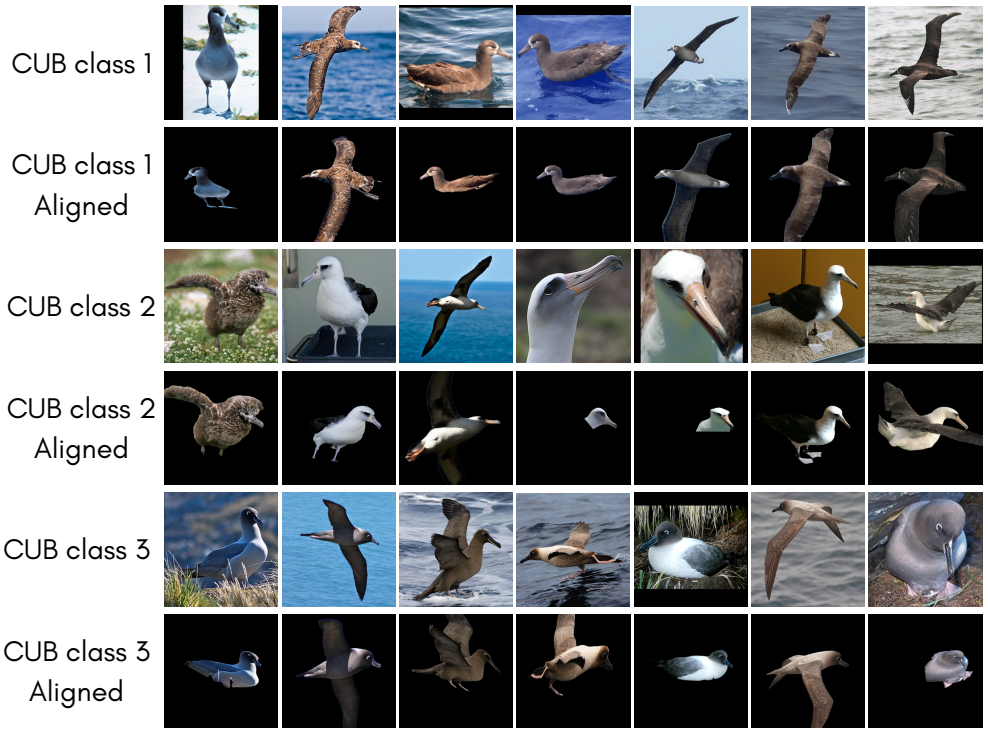

Figure 7: **Qualitative alignment results on CUB-200 classes.** Each pair of rows corresponds to a distinct semantic class from the CUB-200 [3] dataset. The top row in each pair shows the original, unaligned images; the bottom row shows the corresponding aligned images produced by FastJAM. The alignment process successfully maps semantically consistent parts (e.g., heads, wings, tails) to similar spatial locations across different instances within each class.

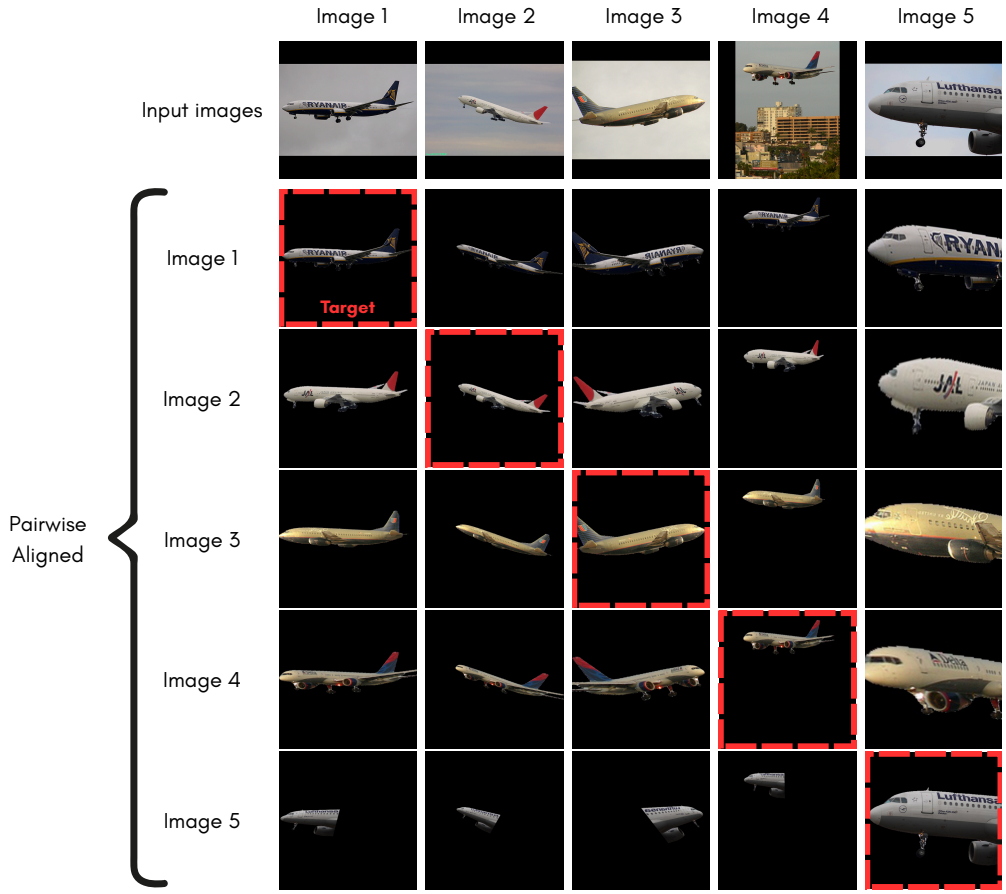

Figure 8: **Pairwise Alignment Grid – "Aeroplane" Class.** Top row: five input images from the "aeroplane" category. Below: a 5×5 alignment matrix, where each entry in column j displays the corresponding source image from row i, warped to align with target image j using the inverse-compositional transformation $T^{\theta_i} \circ T^{-\theta_j}$. Each row thus visualizes the same source image aligned to five different targets. Diagonal entries show the self-warped images (i.e., identity transformation), but with the canonical background rather than the original, and are marked with a **red** dashed-line frame. This layout highlights FastJAM's ability to achieve coherent, semantically meaningful alignments across all image pairs.

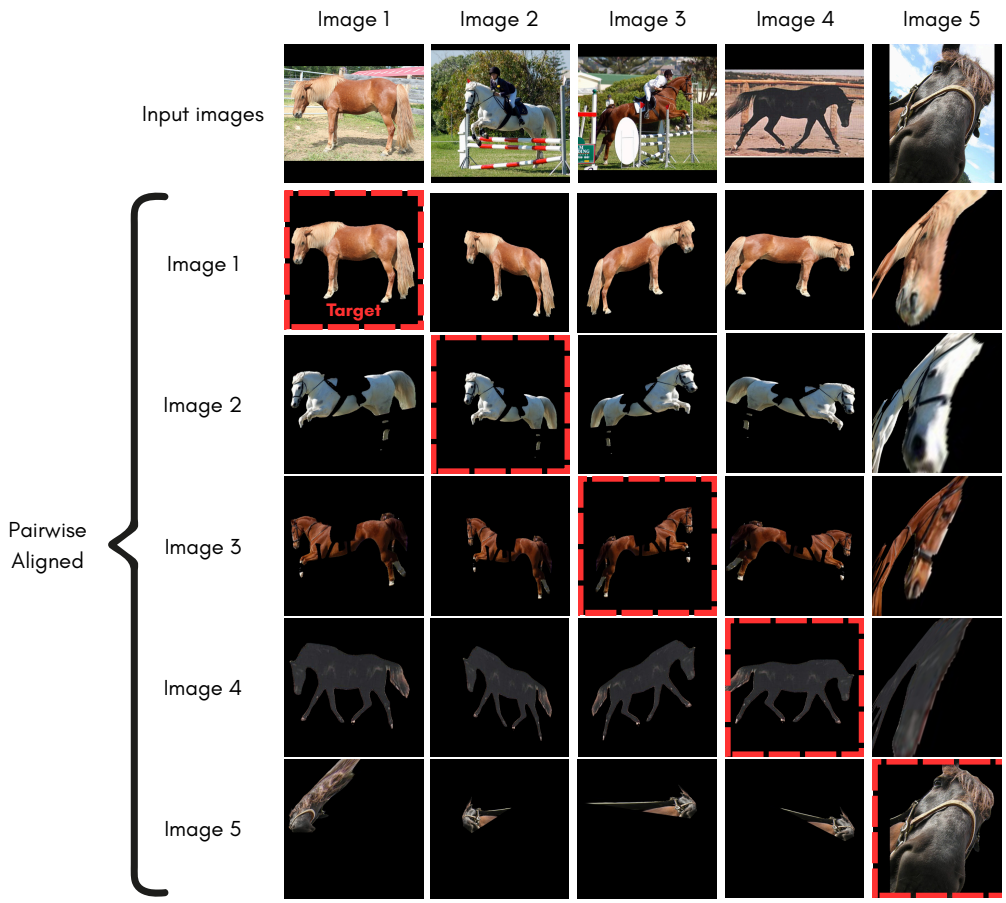

Figure 9: **Pairwise Alignment Grid – "Horse" Class.** Top row: five input images from the "horse" category. Below: a 5×5 grid showing the full pairwise alignment structure. Each image in row i, column j corresponds to the source image i aligned to the target image j using $T^{\theta_i} \circ T^{-\theta_j}$. The diagonal entries depict self-warping (identity), rendered with the canonical background rather than the original, and are marked with a **red** dashed-line frame. This visualization reveals consistent alignment behavior across the set, illustrating how FastJAM handles pose and appearance variation within a semantic class.

C Explaining the Colormap Visualization in More Detail

To provide an intuitive understanding of how FastJAM aligns images to a shared canonical space, we visualize the warped outputs using a predefined RGB colormap. As shown in Figure 10, each input image is first aligned to the canonical frame and blended with the colormap. The result is then inverse-warped back to the original image space, allowing us to visualize how semantic regions are mapped consistently across instances. This process highlights FastJAM's ability to establish meaningful correspondences without relying on dense features or explicit templates.

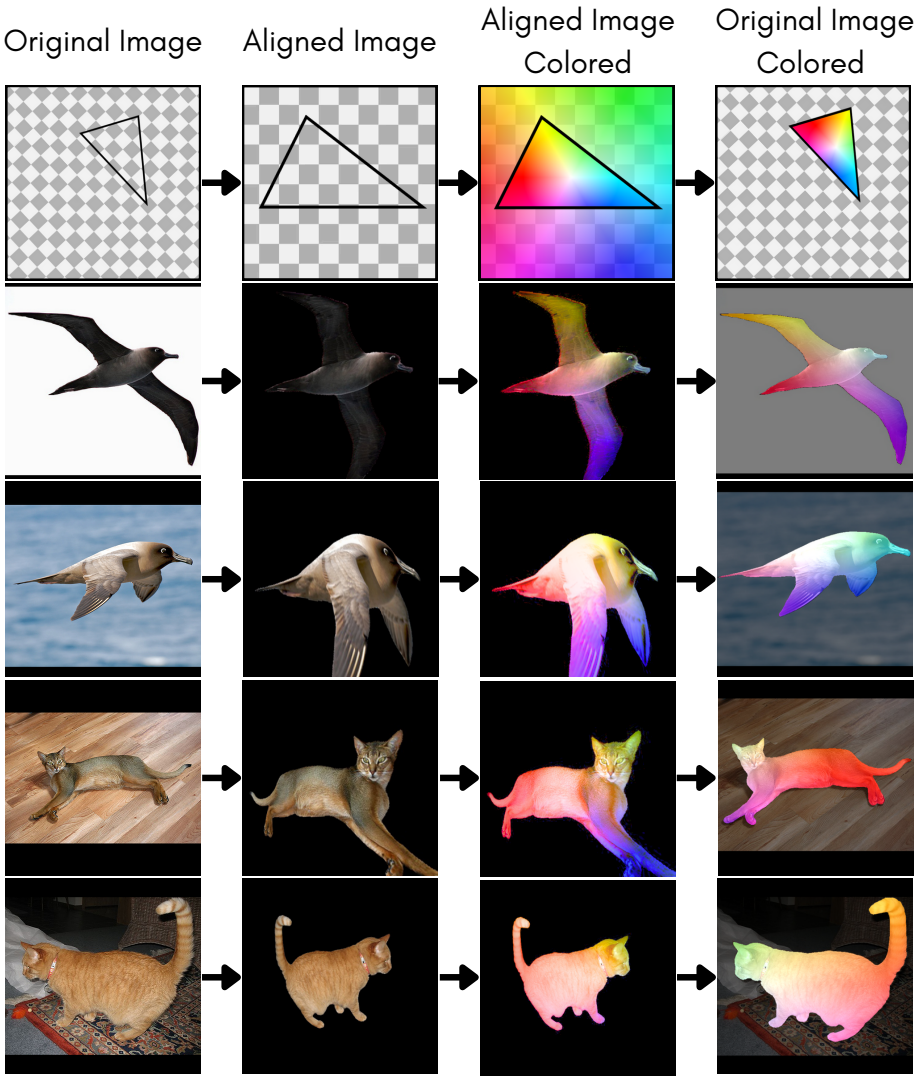

Figure 10: **Illustrated Canonical Space Visualization.** This figure provides an intuitive explanation of how FastJAM visualizes the canonical space $\mathcal{C}$ using a predefined RGB colormap I_{colormap}. For each input image I_i (left column), we first apply the estimated homography to obtain its aligned version in canonical space: $I_i \circ T^{\theta_i}$. We then blend this aligned image with the colormap via averaging: $\frac{1}{2}(I_{\text{colormap}} + (I_i \circ T^{\theta_i}))$. Finally, we apply the inverse warp to visualize the blended canonical signal in the original image frame: $\left[\frac{1}{2}(I_{\text{colormap}} + (I_i \circ T^{\theta_i}))\right] \circ T^{-\theta_i}$. This provides a visual explanation of how semantically similar regions across instances are mapped to consistent spatial locations.

D Runtime Analysis

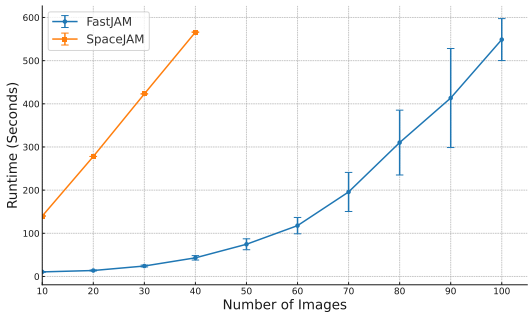

Figure 11: Runtime analysis between SpaceJAM and FastJAM over an increasing number of images. SpaceJAM PCA processing step runs out of RAM after 40 images.

As illustrated in Figure 11, FastJAM remains efficient across all tested sizes, completing alignment in under 450 seconds even for 100 images. In contrast, the current implementation of SpaceJAM runs out of RAM after $N = 40$ images due to the PCA preprocessing step.

E Model Configuration and Training Setup

We train the model for 600 epochs using the Adam optimizer with an initial learning rate of 5×10^{-3}, multiplied by 0.5 after 200 epochs without improvement. The loss function is based on the Geman-McClure formulation with a robustness parameter $\sigma = 0.25$, and no weight decay is applied. The feature extractor within the GNN uses 5 layers of hidden size 128, followed by a linear projection to an 8-dimensional homography parameter vector. A full summary of the model architecture and its 133,256 trainable parameters is provided in Table 5. To encourage geometric stability, the final projection layer is initialized to approximate the identity transformation, and transformations are parameterized using Lie algebra to ensure invertibility. Alignment is applied iteratively using a single pass of the inverse compositional (IC) spatial transformer network. During optimization, horizontal flips are checked every 100 epochs.

Table 5: GraphSAGE GNN Model Summary.

Layer (type:name)	**Output Shape**	**Param**
GraphSAGE GNN	–	–
convs.0.lin_l.weight	[128, 2]	256
convs.0.lin_l.bias	[128]	128
convs.0.lin_r.weight	[128, 2]	256
convs.1.lin_l.weight	[128, 128]	16,384
convs.1.lin_l.bias	[128]	128
convs.1.lin_r.weight	[128, 128]	16,384
convs.2.lin_l.weight	[128, 128]	16,384
convs.2.lin_l.bias	[128]	128
convs.2.lin_r.weight	[128, 128]	16,384
convs.3.lin_l.weight	[128, 128]	16,384
convs.3.lin_l.bias	[128]	128
convs.3.lin_r.weight	[128, 128]	16,384
convs.4.lin_l.weight	[128, 128]	16,384
convs.4.lin_l.bias	[128]	128
convs.4.lin_r.weight	[128, 128]	16,384
fc.weight	[8, 128]	1,024
fc.bias	[8]	8
Total	–	**133,256**

For correspondence estimation, we employ the RoMa matcher at a fixed image resolution of 560×560 for both coarse and upsampled stages. A maximum of 10 keypoints is retained per image, filtered by

non-maximum suppression (NMS) using a radius of 0.054 in normalized coordinates—corresponding to a 30×30 pixel window in the original image space. This ensures spatial coverage while avoiding redundant detections.

Table 6: Summary of Notation

Symbol	Description
$\mathcal{I} = (I_i)_{i=1}^N$	Set of input images
M_i	Number of keypoints in image I_i
$X_i = \{x_i^{(1)}, \ldots, x_i^{(M_i)}\}$	Keypoints in image I_i, $x_i^{(m)} \in [-1, 1]^2$
$\mathcal{M}_{ij}$	Set of matched keypoints between I_i and I_j
$\mathcal{T}$	Family of parametric transformations (e.g., homographies)
$T^{\boldsymbol{\theta}_i} \in \mathcal{T}$	Transformation for image I_i, parameterized by $\boldsymbol{\theta}_i$
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	Keypoint graph with intra- and inter-image edges
$f(\mathcal{G})$	GNN-based function that predicts $\{\boldsymbol{\theta}_i\}_{i=1}^N$ from a graph
$I_i \circ T^{\boldsymbol{\theta}_i}$	Image I_i warped by transformation $T^{\boldsymbol{\theta}_i}$

F External Tools and Frameworks

We gratefully acknowledge the use of several open-source libraries and resources in this project. Our GNN implementation is based on `torch_geometric`, primarily using the GraphSAGE [4] architecture, along with other variants for comparison. We used Weights & Biases for experiment tracking and visualization. For keypoint matching, we built upon the official implementation of RoMa [5], and we also incorporated components from the LoFTR framework [6]. Object-centric masks were obtained using Grounded-SAM, which combines Grounding DINO [7, 8] and the Segment Anything Model (SAM) [9]. We thank the authors of all these works for making their code and models publicly available.

G Lie Group Parameterization

A homography has 8 degrees of freedom and corresponds to an equivalence class of invertible matrices, where a representative with unit determinant can be used. Now consider

$$\mathfrak{sl}(3) = \left\{ \boldsymbol{\Theta} = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 \\ \theta_4 & \theta_5 & \theta_6 \\ \theta_7 & \theta_8 & -(\theta_1+\theta_5) \end{bmatrix} \right\} \text{ and } \quad \mathrm{SL}(3) = \left\{ \boldsymbol{H} \in \mathbb{R}^{3\times 3} : \det \boldsymbol{H} = 1 \right\}. \tag{1}$$

The space $\mathfrak{sl}(3)$ is the Lie algebra of trace-zero matrices. The matrix exponential maps $\mathfrak{sl}(3)$ into $\mathrm{SL}(3)$, yielding a smooth eight-parameter representation of homographies. Our network predicts $(\boldsymbol{\theta}_i)_{i=1}^N$, where each $\boldsymbol{\theta}_i \in \mathbb{R}^8$ is mapped to $\boldsymbol{\Theta}_i \in \mathfrak{sl}(3)$ as shown above, and the homography is obtained via $T^{\boldsymbol{\theta}_i} = \boldsymbol{H}_i = \exp(\boldsymbol{\Theta}_i)$. This construction guarantees $\det T^{\boldsymbol{\theta}_i} = 1$. In particular, $\boldsymbol{\theta}_i = \mathbf{0}_{8\times 1}$ yields the identity matrix, and $T^{-\boldsymbol{\theta}_i}$ is the inverse of $T^{\boldsymbol{\theta}_i}$.

References

[1] Kamal Gupta, Varun Jampani, Carlos Esteves, Abhinav Shrivastava, Ameesh Makadia, Noah Snavely, and Abhishek Kar. ASIC: Aligning sparse in-the-wild image collections. In *ICCV*, 2023. 2

[2] Nir Barel, Ron Shapira Weber, Nir Mualem, Shahaf E Finder, and Oren Freifeld. Spacejam: a lightweight and regularization-free method for fast joint alignment of images. In *European Conference on Computer Vision*, pages 180–197. Springer, 2024. 2

[3] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset, 2011. 3

[4] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017. 8

[5] Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Roma: Robust dense feature matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19790–19800, 2024. 8

[6] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8922–8931, 2021. 8

[7] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024. 8

[8] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 8

[9] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 8