SpaceJAM: a Lightweight and Regularization-free Method for Fast Joint Alignment of Images Supplemental Material

Nir Barel*, Ron Shapira Weber*, Nir Mualem, Shahaf E. Finder, and Oren Freifeld

The Department of Computer Science, Ben-Gurion University of the Negev, Israel
{banir,ronsha,nirmu,finders}@post.bgu.ac.il, orenfr@cs.bgu.ac.il

1 Handling flips

As detailed in the main manuscript, flips require special care. Let (I_i, I_j) be a source and target images respectively (for simplicity, we drop the DINO ViT and learned features notion and use this notion). In this particular case, our loss function reduces to

$$\mathcal{L}_{\text{IC}} = \left\| I_j - (I_i \circ T^{\boldsymbol{\theta}_i} \circ T^{-\boldsymbol{\theta}_j}) \right\|_{\ell_2}^2 . \tag{1}$$

To incorporate flips efficiently, we consider only horizontal flips (since vertical flips could be reached through a horizontal flip + rotation) and compute the gradient only between the best matching pair. Particularly, let F^{k_i} be the k^{th} flip configuration applied to the i^{th} image, where $k \in C$ such that C holds the possible configuration (in our case, 2). The objective function is now

$$\mathcal{L}_{\text{IC}} = \sum_{i=1}^{N} \sum_{k_i=1}^{|C|} \min_{k_j \in C} \left\| I_j \circ F^{k_j} - ((I_i \circ T^{\boldsymbol{\theta}_i}) \circ F^{k_i}) \circ T^{-\boldsymbol{\theta}_j} \right\|_{\ell_2}^2 , \tag{2}$$

where (k_i, k_j) are the flips considered for the image pair (I_i, I_j).

2 Curriculum learning

To incorporate the Lie-algebraic curriculum learning during training, we gradually add more complex transformation modules, starting from SE(2) and later "release" more of the transformation parameters, to obtain (invertible) affine transformations and finally homographies. Figure 1 illustrate the process, where additional transformation parameters are "released", as illustrated by the warped images above the training timeline.

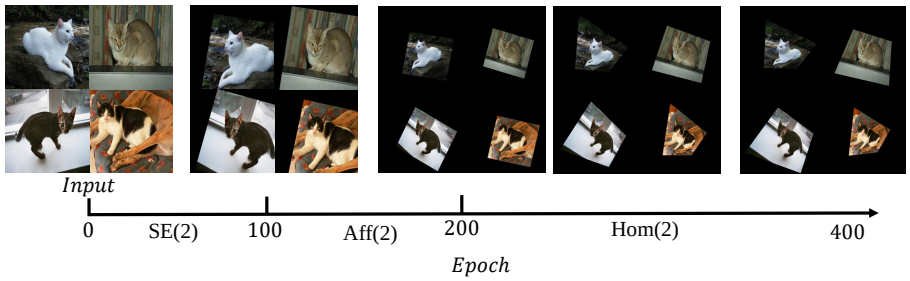


Fig. 1: Lie Algebric curriculum learning. The notation SE(2) between the epochs $(0, 100)$ states that during that interval, the training is restricted to SE(2). At epoch 100, more transformation parameters are "released" to allow for affine transformations.

3 Inverse-Compositional STN (IC-STN) [3]

The IC-STN [3], based on the classical Inverse-Compositional Lucas & Kanade algorithm [4], predicts a cascade of smaller warps and composes them. In our case, we do so by recursively feeding the STN several times with its own output. In effect,

$$(\boldsymbol{\theta}, X \circ T^{\boldsymbol{\theta}}) = \psi_{\text{STN}}(X) \tag{3}$$

$$(\boldsymbol{\theta}', X \circ T^{\boldsymbol{\theta}} \circ T^{\boldsymbol{\theta}'}) = \psi_{\text{STN}}(X \circ T^{\boldsymbol{\theta}}) \tag{4}$$

$$(\boldsymbol{\theta}'', X \circ T^{\boldsymbol{\theta}} \circ T^{\boldsymbol{\theta}'} \circ T^{\boldsymbol{\theta}''}) = \psi_{\text{STN}}(X \circ T^{\boldsymbol{\theta}} \circ T^{\boldsymbol{\theta}'}) \tag{5}$$

and so on.

4 An Additional runtime comparison

We provide an additional runtime comparison with Neural Congealing [6] and ASIC [2] on the 'Dog' and 'Bike' datasets [5]. The results are presented in Table 1.

Table 1: A comparison with recent JA methods and evaluation on 3 SPair-71K categories [5].

Method	# Params	# Losses	#HP	Atlas-free learning	#epochs	Time Cat	Time Bike	Time Dog
NeuCongealing [6]	28.7M	8	8	✗	8K	1:17:02	1:12:55	1:25:28
ASIC [2]	7.9M	4	5	✗	20K	1:06:48	1:07:40	1:06:11
SpaceJAM (Ours)	**0.016M**	**1**	**0**	✓	**0.7K**	**0:05:58**	**0:06:11**	**00:05:43**

5 Architectures

A detailed overview of the Autoencoder (AE) and Spatial Transformer Network (STN) architectures and the number of trainable parameters. Together they form our alignment module - ψ_{align}.

Table 2: Autoencoder Model Summary. GFMN = GlobalFeatureMapNormalizer.

Layer (type:depth-idx)	Output Shape	Param
Autoencoder	[1, 25, 256, 256]	–
Encoder: 1-1	[1, 3, 256, 256]	–
Sequential: 2-1	[1, 3, 256, 256]	–
Conv2d: 3-1	[1, 32, 256, 256]	832
ReLU: 3-2	[1, 32, 256, 256]	–
BatchNorm2d: 3-3	[1, 32, 256, 256]	64
Conv2d: 3-4	[1, 16, 256, 256]	528
ReLU: 3-5	[1, 16, 256, 256]	–
BatchNorm2d: 3-6	[1, 16, 256, 256]	32
Conv2d: 3-7	[1, 3, 256, 256]	51
GFMN: 3-8	[1, 3, 256, 256]	–
Decoder: 1-2	[1, 25, 256, 256]	–
Sequential: 2-2	[1, 25, 256, 256]	–
Conv2d: 3-9	[1, 16, 256, 256]	64
ReLU: 3-10	[1, 16, 256, 256]	–
BatchNorm2d: 3-11	[1, 16, 256, 256]	32
Conv2d: 3-12	[1, 32, 256, 256]	544
ReLU: 3-13	[1, 32, 256, 256]	–
BatchNorm2d: 3-14	[1, 32, 256, 256]	64
Conv2d: 3-15	[1, 25, 256, 256]	825

Table 3: STN Model Summary.

Layer (type:depth-idx)	Output Shape	Param
Conv2d-1	[1, 10, 250, 250]	1,480
AdaptiveMaxPool2d-2	[1, 10, 32, 32]	0
ReLU-3	[1, 10, 32, 32]	0
Conv2d-4	[1, 5, 28, 28]	1,255
AdaptiveMaxPool2d-5	[1, 5, 8, 8]	0
ReLU-6	[1, 5, 8, 8]	0
Linear-1	[1, 1, 32]	10,272
ReLU-2	[1, 1, 32]	0
Linear-3	[1, 1, 9]	297

We also evaluate the effect of the STN size on the resulting alignment. Figure 2 shows the average PCK@0.1 of 5 runs for the 3 subsets of the CUB200 datasets [7]. We increase the number of trainable parameters by using the same STN architecture with larger convolutional blocks in terms of # kernels and their size. Notably, the performance effectively saturates at as early as ∼15K parameters. Increasing the model further even to 24M parameters, does not results in additional gains.

6 Further Discussion of the Results

A natural question arises – why do different models perform better in some classes and worse in others? For instance, consider the 'Dogs' class of the SPair dataset. In the results presented in Table 2 in the main paper, ASIC outperforms the proposed method on that class by approximately 11 points, suggesting superior alignment. However, the visual comparison in Figure 3 reveals that dense-correspondence methods like ASIC often result in incoherent alignment. The warped images display artifacts such as holes, and the dog faces become unrecognizable. This discrepancy arises because benchmarks like SPair and CUB-200 focus on the sparse correspondence of hand-picked key points rather than measuring global

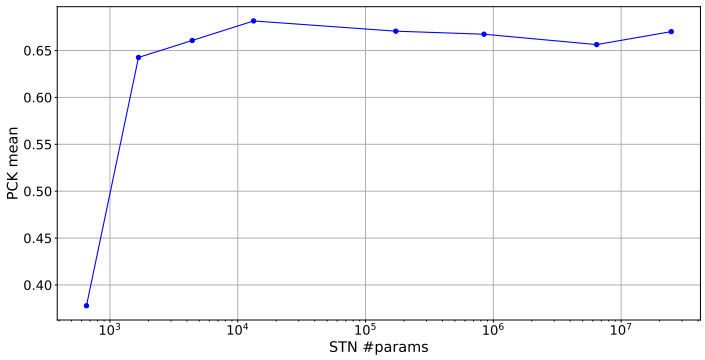
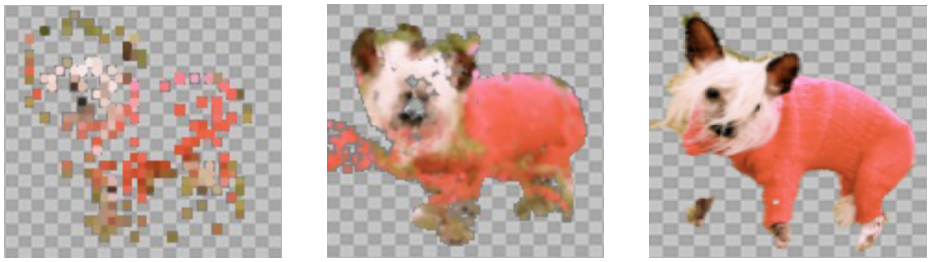


Fig. 2: Average PCK@0.1 score as a function of # trainable parameters of the STN (the x-axis is log-scaled). The model reaches saturation around $\sim$ 15K parameters.



(a) DINO+NN **(b)** ASIC **(c)** SpaceJAM (ours)

Fig. 3: Geometric fidelity of transformed images (DINO+NN and ASIC results were obtained from [2]).

alignment. In fact, the basic DINO-NN outperforms SpaceJAM on the same 'Dog' class, but yields significantly poorer visual results, as illustrated in Figure 3. This highlights the limitations of the DINO-NN approach. Additionally, our method outperforms ASIC in more than half of the classes while requiring 100x fewer parameters and achieving a 10x reduction in training time. Finally, the variance in results can also be attributed to the small set size (20-30 images) compared to the diverse poses, illuminations, and occlusions present in each set.

7 Additional Visualizations

7.1 Additional joint alignment results

More visual results of SpaceJAM's joint alignment (JA) are presented below (Figures 2-6) for the SPair-71K [5] and Samurai ('robot') [1] datasets. The figures show, from top-to-bottom: 1) input images; 2) DINO ViT features (first 3 PCs); 3) learned features 4) aligned features, and 5) aligned images. The aligned features and images are masked by the intersection of the coarse input mask and the median mask of the set (both after alignment). The atlas of the set appears at the bottom right.

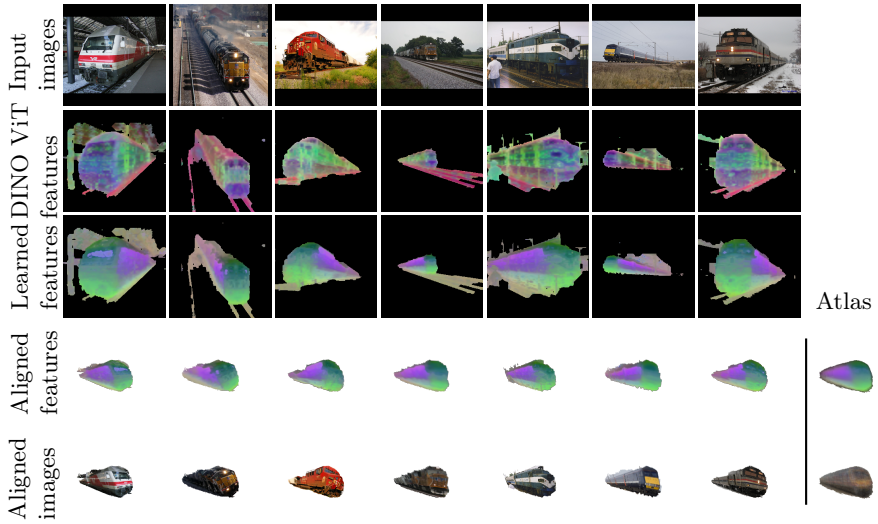


Fig. 4: Joint alignment results - "train".

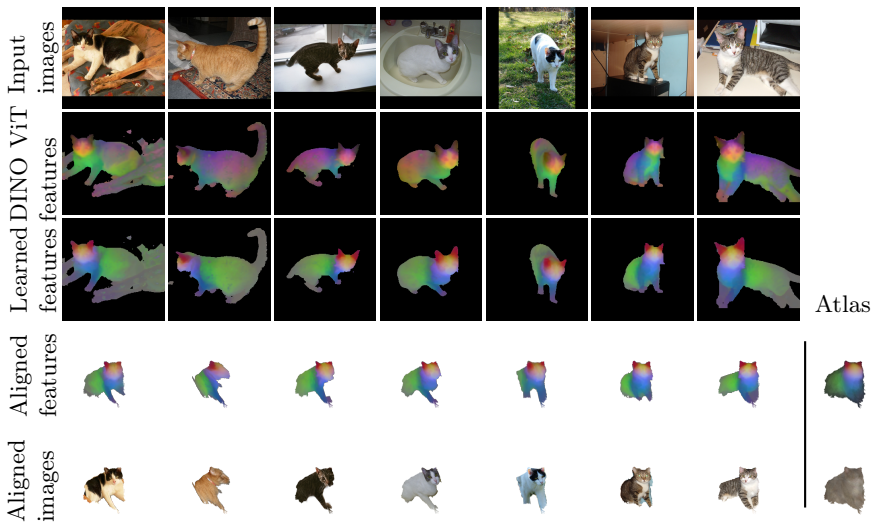


Fig. 5: Joint alignment results - "cat".

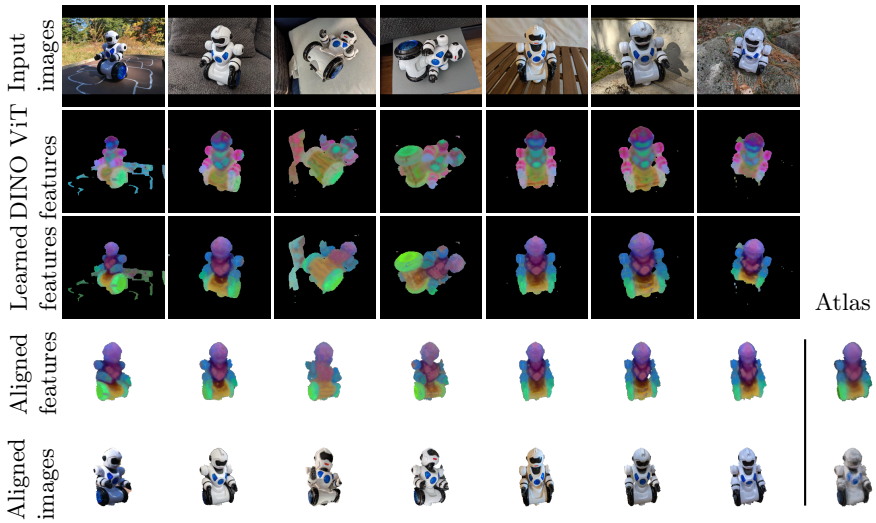


Fig. 6: Joint alignment results - "robot".

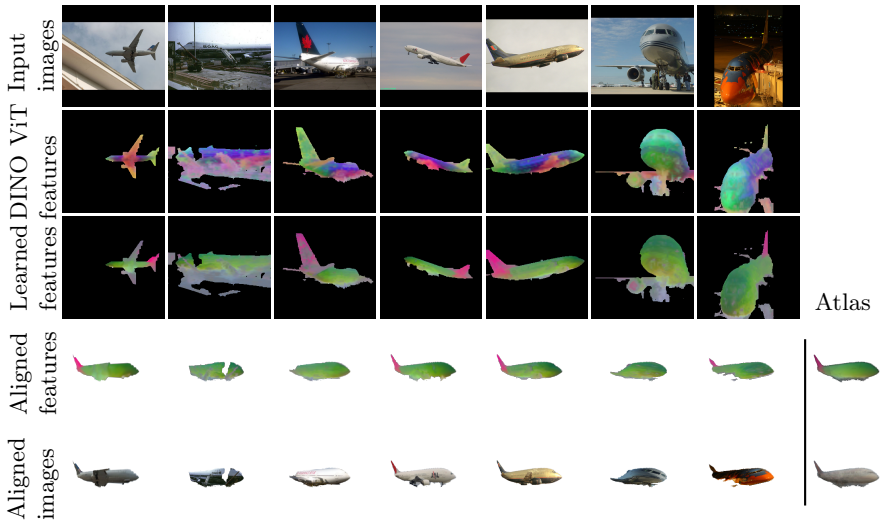


Fig. 7: Joint alignment results - "plane".

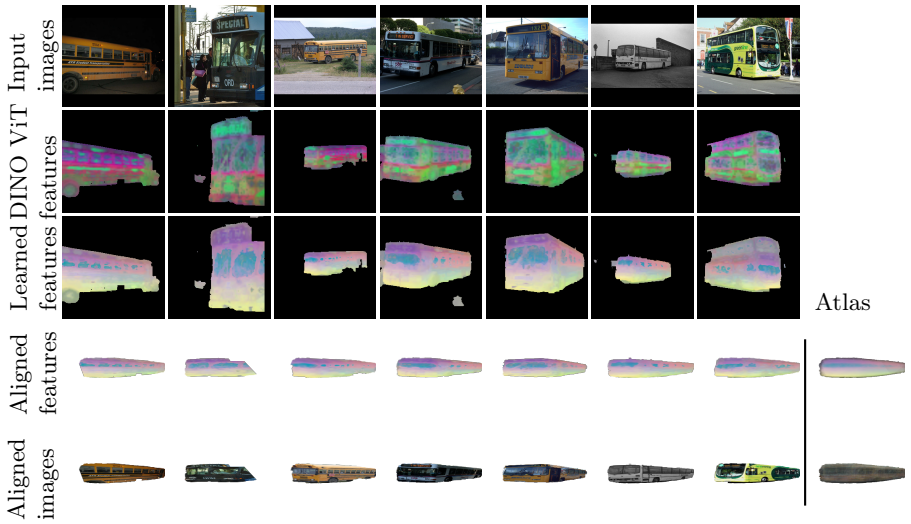


Fig. 8: Joint alignment results - "bus".

7.2 Additional pairwise alignment results

More visual results of SpaceJAM's pairwise alignment are presented below. The figures show, from top-to-bottom: 1) input images; 2) learned features overlay; 3-7) Source-to-target pairwise alignment, where the image in the red square is aligned to all other images.

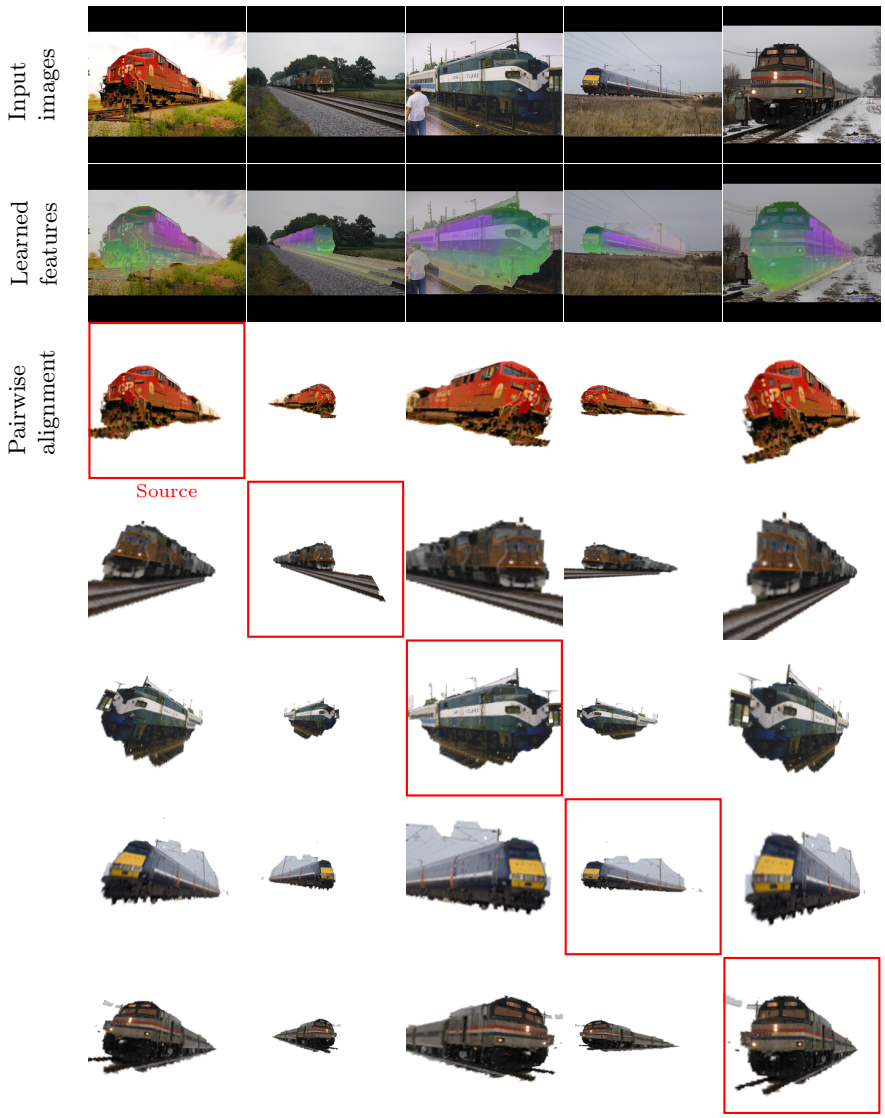


Fig. 9: Pairwise alignment results - "train".

Fig. 10: Pairwise alignment results - "bus".

References

1. Boss, M., Engelhardt, A., Kar, A., Li, Y., Sun, D., Barron, J., Lensch, H., Jampani, V.: Samurai: Shape and material from unconstrained real-world arbitrary image collections. Advances in Neural Information Processing Systems **35**, 26389–26403 (2022)
2. Gupta, K., Jampani, V., Esteves, C., Shrivastava, A., Makadia, A., Snavely, N., Kar, A.: Asic: Aligning sparse in-the-wild image collections. In: ICCV (2023)
3. Lin, C.H., Lucey, S.: Inverse compositional spatial transformer networks. In: CVPR (2017)
4. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI (1981)
5. Min, J., Lee, J., Ponce, J., Cho, M.: Spair-71k: A large-scale benchmark for semantic correspondence. arXiv preprint arXiv:1908.10543 (2019)
6. Ofri-Amar, D., Geyer, M., Kasten, Y., Dekel, T.: Neural congealing: Aligning images to a joint semantic atlas. In: CVPR (2023)
7. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset (2011)