Self-Supervised Learning of Semantically Consistent Visual Features via a Simple 3D Prior

Sebastian Day and Oisin Mac Aodha

University of Edinburgh

Abstract. Visual features extracted from vision models like DINOv2 or Stable Diffusion enable high quality semantic correspondence between images. However, such features exhibit documented limitations that hinder accurate correspondence estimation such as the inability to disambiguate the left from right for categories exhibiting symmetries or repeated parts. Previous attempts to improve these features rely on using supervised data such as annotated keypoint pairs or camera pose information, or make restrictive topological assumptions, such as modelling the object of interest as a 3D sphere. We introduce a new self-supervised approach that allows us to learn features for topologically complex object categories using a simple prior. Instead of relying on an expensive instance-level analysis by synthesis pipeline, our approach makes use of contrastive learning and distribution matching at the global dataset-level to learn the coarse shape and appearance of a category. Combined with additional geometric losses, this enables us to replicate the global and local statistics of visual features which match those that are extracted from real images. Quantitative results are presented across several challenging object categories, demonstrating the effectiveness of our approach.

1 Introduction

Being able to decompose complex articulated and deformable object categories into structured representations that encode semantics is a key desirable capability for embodied AI systems. Semantic correspondence estimation is a specific instance of this task where the goal is to match the same local semantic parts/features across different instances of the same object category. The ability to find such correspondences is a key enabling technology for a variety of applications including robotic manipulation [37], image generation and editing [18], visual recognition [36], among others.

The previous most reliable recipe for this task involved training supervised learning-based approaches using large collections of pre-defined known semantic keypoint annotations, *e.g.* the joints on the human body [10]. However, recently it has been demonstrated that general purpose features obtained from large vision models trained using self-supervised [7, 27] or generative [30] objectives are highly effective for semantic correspondence estimation, often surpassing features obtained from bespoke supervised methods [47]. These features exhibit

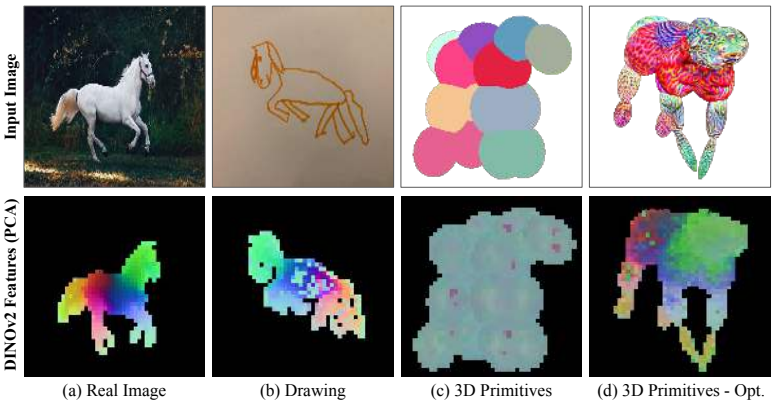

Fig. 1: Pre-trained self-supervised models like DINOv2 [27] produce local visual features that encode distinct semantic parts that generalise across different depictions of a category. In (a) and (b) (images from [27]), we observe that the features for the horse's head and tail are similar even if when coming from an image or drawing. In (c) we display the features derived from an abstract 3D collection of spheres that broadly has the same structure as a horse. However, the resulting features are not consistent with those from (a) and (b). In (d) we show features from our proposed self-supervised approach that optimises the shape and appearance of the shape from (c) such that when rendered and passed through DINOv2 they result in consistent visual features.

several key advantages: they do not require category or semantic keypoint specific fine-tuning, correspondences can be efficiently computed using distance-based matching, and they generalise better than supervised methods [23].

Despite the effectiveness of self-supervised features in semantic correspondence a prominent failure case is that of models becoming confused when there are repeated instances of a part, *e.g.* confusing the front and back wheel of a car [46]. One hypothesis to explain this behaviour is that the representations learned by these large models fail to encode information about the underlying 3D shape or structure of objects [24]. An obvious solution to this problem is to try to inject 3D knowledge into the features. Various techniques to attempt this have been proposed ranging from methods that use very simplistic, and thus inexpressive, geometric priors [24], all the way to methods that estimate category-specific 3D articulated meshes [40]. These approaches typically require some form of additional supervision, whether it is approximate camera viewpoints [24] or category-specific template meshes [33].

We propose a method for extracting visual features that is more expressive than using a single simple 3D primitive, yet much more efficient to estimate than a full complex articulated 3D shape. Inspired by how well DINOv2 features [27] generalise across different visual depictions (see Fig. 1), our method leverages a simplistic synthetic 3D representation and optimises it to match the features extracted from real images. Specifically, given an initial prior, our self-supervised approach constructs a simple 3D mesh that is represented as ellipsoids. This mesh and surface texture are learned such that when rendered to an arbitrary

view and input to a feature extractor, it results in visual features that are similar to those obtained from real images. Rather than per-instance optimisation as in analysis-by-synthesis approaches, our approach is lightweight as we make use of geometric losses and minibatch-level feature distribution matching during training. We demonstrate that our approach works with either of two minibatch-level learning algorithms and, while only having relatively few learnable parameters, can generalise to a wide variety of categories (*e.g.* quadrupeds, spiders, bicycles, etc.), and is robust to variation across instances. We report quantitative results for semantic correspondence across different object categories and demonstrate that our approach can learn semantically-aligned visual features.

2 Related Work

Most recent works on the task of semantic correspondence estimation usually follow a similar approach of taking a pre-trained feature detection model, such as the feature layers of DINOv2 [27,34] or Stable Diffusion [30], and train an adapter of the features. Features can be used directly or aggregated across multiple layers [1, 46]. Features from vision-language models (VLMs) have also been used for semantic keypoint detection [45].

Supervised methods. These approaches typically perform contrastive learning on features extracted from annotated keypoints [2, 22, 35, 46, 47]. [46] uses SoftArgMax on feature distances and an L2 penalty on distance from the target keypoint to post-process pre-trained features using contrastive learning. Pose graphs have also been explored in [13]. The limitation of supervised approaches is they require extensive expensive to obtain manual annotations. They also have a tendency to overfit to the trained keypoints and hence do not generalise [23].

Unsupervised methods. This family of methods focus on taking features from pre-trained models and adapting them through various means, *e.g.* congealing [11, 26], contrastive learning [41], or imposing structure [5, 24]. Techniques have been proposed that rely on self-consistency objectives, using either affine image transformations or known similar keypoints between images to push or pull features toward or away from one another [2,11,15,26,28,46]. Some works attempt to establish geometric transformations between images by applying image-based [33] or latent-based [28] transformations and using the transformed points for self-supervised consistency. The transformations cannot allow too large a deviation from the original and hence the geometric gain can be limited.

Some of these methods are weakly supervised as they use segmentation, pose labels or depth maps [23, 24]. Geometric constraints can be added such as flow and rigidity [5], and the learning can be constrained to a canonical mesh [25,32]. Multi-view images with pose labels can be used to learn 3D Gaussian representations which can then be used to supervise downstream tasks [44]. However, obtaining representative meshes for uncommon categories can be challenging. Canonical spaces can be more abstract, for the purposes of relating features to one another, or storing canonical pixel values, taking the form of a grid or sphere [11, 24, 26], or use point cloud representations requiring depth maps [23, 38].

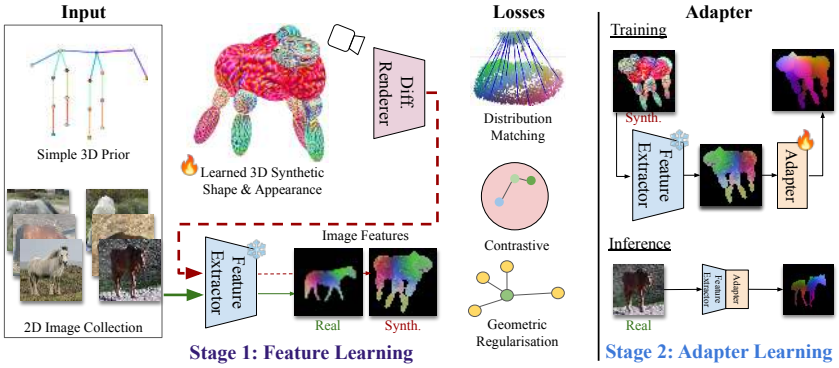

Fig. 2: Overview of our approach which consists of feature learning via canonical shape reconstruction and then feature adapter learning. In the first stage, we start with a collection of images depicting the same object category (*e.g.* a horse) and an initial simple 3D shape prior in the form of a graph. We then optimise the shape and appearance of the canonical shape such that image features derived from it match the distribution of those obtained from real images, all without requiring any instance-level supervision. In the second stage, we refine our features using an adapter by exploiting the fact that in the first stage we have generated a mechanism for creating paired weakly supervised training data that maps image features to per-pixel canonical 3D coordinates.

Reconstruction-based methods can avail of large image collections and use them to estimate a 3D model of a category [3, 21, 40] that can be optimised per image [42, 43]. These methods both use pre-trained visual features from real images for training but can be used to obtain new features using synthetic renders of the learned shape [16]. However, these models can be brittle and they require carefully constructed training pipelines and are limited to the categories they are trained on. They struggle with more complex shapes (*e.g.* spiders) when deforming the template mesh. Diffusion models have been used to generate synthetic training images for these models [14], but it is difficult to control the articulation and 3D pose, and the renders can hallucinate structures (*e.g.* additional limbs). DualPM [16] can reconstruct accurate 3D point clouds for single images but it requires a per-category mesh with manually specified pose articulations. Similar to [33], we also train a feature adapter, but our approach learns the explicit 3D model that enforces a plausible feature to 3D mapping.

3 Method

Given a dataset of real images $\{I_i\}_{i=1}^N$, depicting a category of interest, we want to find a mapping that maps each pixel on the object from one image onto the same semantic part in the other image. However, obtaining supervised correspondence annotations is time consuming and expensive, and thus we propose a fully self-supervised approach. Specifically, we take a pre-trained feature extractor ϕ as our starting point, and aim to refine the features it produces $\mathcal{F} = \phi(I) \in \mathbb{R}^{X \times Y \times D}$, with an adapter χ to disambiguate symmetric and re-

peated parts. The features we will use for matching are therefore $Q = \chi(\phi(I))$. The core idea of our approach is to learn a parameterised synthetic 3D object representation $R(\mathbf{x}; \boldsymbol{\eta})$, for 3D positions $\mathbf{x} = (x, y, z)$ and articulation η, that is optimised to generate realistic features when 2D projections/renderings $\mathcal{P}_v(R)$ of it from camera view v are passed to a pre-trained feature extractor ϕ.

The canonical representation $R(\mathbf{x}; \eta)$ is a textured 3D mesh generated from a simple input graph that can be rendered to arbitrary views with arbitrary articulations η. By providing a 3D graph as input, depicting a category's simplified skeleton topology in an approximately canonical pose, we construct the 3D representation by initialising 3D ellipsoids along each edge of the graph. The insight of Fig. 1 is that realistic looking shapes are not required to obtain good features. We exploit this by using a simple and easy-to-manipulate representation. The parameters being optimised then are the orthogonal scales $\boldsymbol{\sigma}^b = (\boldsymbol{\sigma}^b_x, \boldsymbol{\sigma}^b_y, \boldsymbol{\sigma}^b_z)$ for each b^{th} ellipsoid, the lengthwise scalar offset between the graph edge's 3D center and the ellipsoid's center δ^b, and each ellipsoid's texture.

We represent each ellipsoid as a 3D mesh with the texture parameterised as per-vertex RGB vectors, $\zeta^b_{i=1,\ldots,N_w}$ where $\zeta^b_i = (r^b_i, g^b_i, b^b_i)$ and N_w is the number of vertices per ellipsoid. As most real world object categories are symmetric about at least one plane, we enforce this symmetry by defining our graph to lie along the z-axis such that $R(x, y, z) = R(-x, y, z)$ meaning the number of parameters to be optimised is almost halved. We argue that with a suitable prior distribution for camera views, and for sufficiently many views, we should be able to optimise the canonical representation through self-supervised matching of synthetic and real feature distributions. Once we have an optimised canonical representation where by construction we know the mapping from 2D image pixels to 3D positions on the canonical shape, we train an adapter χ to map from pre-trained features to improve semantic correspondence.

3.1 Feature Learning

We start with a pre-trained feature extractor ϕ that takes an image I as input and produces a feature map $\mathcal{F} = \phi(I)$. These features are typically high-dimensional (*e.g.* the dimensionality of DINOv2 patch tokens is 768) and before we proceed we reduce this dimensionality using Princicpal Components Analysis (PCA). We do this using only features from the category being optimised and a pre-trained model $\mathcal{M}$, *e.g.* an off-the-shelf pre-trained object segmentation model such as SegmentAnything [19], to mask these features so only the relevant foreground ones are used. To match features from different images it is typical to find pairs that minimise cosine similarity. If $\mathcal{F} = \phi(I)$ is the 2D grid of features representing an input image I, then the feature vector $\mathbf{f}$ located at pixel (x, y) is denoted by $\mathcal{F}(x, y)$. Therefore, following PCA we normalise all feature vectors as $\mathbf{f} = ||\mathbf{f}||_2$ so we may optimise for the property we are interested in, *i.e.* the cosine similarity between real and synthetic features.

We seek to learn a 3D synthetic representation that replicates the spatial distribution of features produced by a pre-trained feature extractor on real images of the same category while avoiding the need for view or correspondence supervision. To do this we cannot rely on render-and-compare approaches like previous

models such as [21,40] do. Instead, we look to match the distributions of features in aggregate, independent of spatial structure. The underlying assumption being made is that for a suitable view prior distribution and for sufficiently many views, the distributions of the synthetic and real feature sets should be similar. More concretely, if we observe the category of interest from enough viewpoints we would expect the proportion of head features to leg features aggregated across all views to be approximately constant. That means we expect the aggregated histogram of all features across all views to be similar for both real images and synthetic renders.

In learning a canonical representation, one approach would be to learn features assigned to the surface coordinates, similar to SHIC [32]. But, as argued in [46], global pose information is contained within pre-trained features like DINOv2 and we argue an effective model should be able to extract more information from them by optimising a 3D source object from which to extract features, rather than optimising the features themselves on the 3D surface. By learning a representation that feeds into the feature extractor we obtain features that take advantage of the pre-learned spatial distributions of the extractor.

One of the motivations for our model is to consider uncommon non-trivial categories for which hand-crafted meshes, which are abundant for quadrupeds, are unavailable. We therefore design our model to be agnostic to the specific category as long as an approximate skeletal graph is provided as input. The graph is not expected to be carefully proportioned, *e.g.* in our later experiments all edges have been made unit length. Additionally the input articulation, in which the graph is constructed, need only be approximate as it will be randomly perturbed during training.

Distribution Matching. Given the aforementioned initial 3D shape for a category of interest we can render it from different virtual viewpoints and extract visual features from these images using a pre-trained vision backbone (*e.g.* DINOv2). However, despite the ability of methods like DINOv2 to generate semantically consistent visual features across different depictions of an object (see Fig. 1), these features will initially be substantially different from those extracted from real images depicting the category. Thus the goal is to modify the shape and appearance of the 3D template so that features extracted from renders of it are similar to those from real images.

There are several possible approaches for measuring the similarity between the two aggregate feature distributions, given that we do not have instance-level (*i.e.* image-level) correspondences. Optimal transport-based approaches [29] attempt to find the least costly means of mapping samples from a source distribution to those in a target distribution according to some cost function $\mathbf{C}$, defining the distance between samples of the source distribution and samples of the target distribution We choose optimal transport in contrast to alternative approaches such as Maximum Mean Discrepancy (MMD) [9] as, in its regularised form, it is as computationally efficient as MMD [29] but we also obtain a mapping for each of our synthetic feature samples to real feature samples which we will use at a later stage.

In optimal transport we seek to optimise a transport map $\mathbf{P}$, which assigns probability mass from one distribution to another, by minimising its inner product with the cost function or matrix $\mathbf{C}$. The cost matrix elements $\mathbf{C}_{ij}$ give the distance from source sample i to target sample j. In the discrete case, once optimised, the row $\mathbf{P}_{i,:}$ gives the transport of the i^{th} sample from the source distribution to each of the target samples. In an unregularised empirical sampling setting there would be one non-zero entry per row. A discrete measure for each distribution constrains the optimisation requiring each row and column of $\mathbf{P}$ sum to each sample's total weight via $\mathbf{P}\mathbb{1}_m = \mathbf{a}$ and $\mathbf{P}^T\mathbb{1}_n = \mathbf{b}$, where n and m are the number of source and target samples, and $\mathbf{a}$ and $\mathbf{b}$ are vectors of length equal to the number of samples n and m respectively with each entry having value $\frac{1}{n}$ and $\frac{1}{m}$ for an empirical measure. The entropy, $\mathbf{H}$, regularised discrete optimal transport problem with regularisation parameter ϵ, and set of permissible transport maps $\mathbf{U}(\mathbf{a}, \mathbf{b})$, can be written in the Kantorovich formulation [29]:

$$\mathcal{L}_\epsilon(\mathbf{a}, \mathbf{b}) = \min_{\mathbf{P} \in \mathbf{U}(\mathbf{a}, \mathbf{b})} \sum_{i,j} \mathbf{C}_{ij}\mathbf{P}_{ij} - \epsilon\mathbf{H}(\mathbf{P}), \tag{1}$$

$$\mathbf{H}(\mathbf{P}) = \sum_{i,j} \mathbf{P}_{ij}(log\mathbf{P}_{ij} - 1), \tag{2}$$

$$\mathcal{W}_\epsilon(\mathbf{a}, \mathbf{b}) = \sum_{i,j} \mathbf{C}_{ij}\mathbf{P}^*_{ij}, \tag{3}$$

where, $\mathbf{P}^*$ is our optimal transport and $\mathcal{W}_\epsilon$ is the Wasserstein distance. Unfortunately the gradients calculated for the Wasserstein distance between discrete sample sets are biased [6] but this can be remedied by using instead the Sinkhorn Divergence; the energy distance with the Wasserstein distance as metric, together with entropic regularisation [8, 31]:

$$\mathcal{L}_{distr}(\mathbf{a}, \mathbf{b}) = 2\mathcal{W}_\epsilon(\mathbf{a}, \mathbf{b}) - \mathcal{W}_\epsilon(\mathbf{a}, \mathbf{a}) - \mathcal{W}_\epsilon(\mathbf{b}, \mathbf{b}) \tag{4}$$

This linear program is optimised using the Sinkhorn algorithm [29].

Contrastive Learning. Distribution matching is not the only means of optimising the synthetic features. Self-supervised optimisation of the shape and texture can also be achieved through geometric consistency. To obtain a representation that is consistent across a large variety of arbitrary views and one that captures the full range of features we are likely to see for real images we want to enforce smooth variation in features corresponding to nearby points on the 3D surface and to pull similar features together on the surface to form coherent parts. To do this we exploit our underlying 3D geometry and use contrastive losses on the features back-projected onto the vertices of the canonical representation from the 2D features.

For a given camera view v from a set of $\mathcal{N}_V$ views we randomly sample $\mathcal{N}_S$ vertices from our mesh together with their corresponding features $\mathbf{f}_s^v$ which are those back-projected from the features onto the vertices. These are our anchor features in the contrastive loss. We merge our ellipsoids into one 3D shape by

culling those vertices appearing inside other ellipsoids and greedily adding edges from each vertex on the newly formed boundaries to the nearest vertex in the neighbouring ellipsoid. Here the neighbouring ellipsoid is the parent or child in the original 3D graph. With this fully connected graph and starting from each anchor vertex we find all vertices that are within r edges of the anchor. The corresponding features are the anchor's positive set from which we can randomly sample $\mathcal{N}_a$ features $\{\mathbf{f}_a^v\}$. The complement of this inner circle of vertices, *i.e.* the rest of the graph, is the anchor's negative set and we randomly sample an equal number of N_a features $\{\mathbf{f}_b^v\}$ to give, with constant α:

$$\mathcal{L}_{loc}(\mathbf{f}_s, \mathbf{f}_a, \mathbf{f}_b) = \frac{1}{N_a} \sum_{n}^{N_a} max(0, -\mathbf{f}_s^T \mathbf{f}_{a,n} + \mathbf{f}_s^T \mathbf{f}_{b,n} + \alpha), \tag{5}$$

$$\mathcal{L}_{contr} = \frac{1}{N_V N_s} \sum_{v}^{N_V} \sum_{i}^{N_s} \mathcal{L}_{loc}(\mathbf{f}_{s_i}^v, \mathbf{f}_{a_i}^v, \mathbf{f}_{b_i}^v). \tag{6}$$

3.2 Geometric Consistency

One of the challenges with learning from unlabelled images, whether we are using distribution matching or the contrastive loss, is it is difficult to determine which images are the front and which are the back and where the lines of symmetry should be. As a consequence it is easy for the model to learn, for example, two heads for an animal as images centered on the face are more common in biased datasets. We try to mitigate this by enforcing view-consistency. For both contrastive and distribution matching algorithms we minimise the variance at each vertex of visible features assigned to that vertex. We do this by minimising one minus the cosine similarity of visible vertex features with the vertex's mean feature $\bar{\mathbf{f}}$:

$$\mathcal{L}_{var} = \frac{1}{N_w} \sum_{w}^{N_w} 1 - \mathbf{f}_{w,vis}^T \bar{\mathbf{f}}_w. \tag{7}$$

To further exploit our geometry we use the symmetry defined for our category graph and minimise the cosine similarity of features on symmetric vertices, *i.e.* enforcing that left and right should be equivalent:

$$\mathcal{L}_{sym} = \frac{1}{N_{sym}} \sum_{i}^{N_{sym}} \mathbf{f}_i^T \mathbf{f}_{sym(i)}, \tag{8}$$

where for vertex index i, $sym(i)$ indicates the index of its symmetric vertex. The total number of unique vertices, excluding symmetries is N_{sym}. Again this is done for both learning algorithms.

Our final geometric loss term comprises $\mathcal{L}_{var}$, which optimises the consistency across views, and $\mathcal{L}_{sym}$ which encourages symmetry:

$$\mathcal{L}_{geoc} = \mathcal{L}_{var} + \mathcal{L}_{sym}. \tag{9}$$

Geometric Regularisation. In addition to the above losses, we add geometric regularisation terms. Specifically, we add an L1 loss on the scales σ and offsets δ (relative to their original positions at 0.5), a collision penalty, $\mathcal{L}_{coll}$, where non-neighbouring ellipsoids are penalised for intersecting, and a texture smoothing term $\mathcal{L}_\zeta$. The collision penalty takes the line between non-neighbouring ellipsoid centers, finds the vertices on each ellipsoid whose vector from their respective ellipsoid center is most closely aligned with this line, and uses the distance between their projected positions on the line as a measure of overlap. The texture regularisation is an L1 difference between neighbouring vertex texture parameters ζ. The complete regularisation therefore takes the form:

$$\mathcal{L}_{reg} = \lambda_\sigma \sum |\sigma| + \lambda_\delta \sum |\delta - 0.5| + \lambda_c \mathcal{L}_{coll} + \lambda_\zeta \mathcal{L}_\zeta, \tag{10}$$

which combines the following terms:

$$\mathcal{L}_{coll} = \sum_i \sum_j \mathbb{1}_{\neg neighs(i,j)} \text{ReLU}(\max_a[(c_i - c_j)^T w_a^j] - \min_b[(c_i - c_j)^T w_b^i]), \tag{11}$$

$$\mathcal{L}_\zeta = \frac{1}{N_{edges}} \sum_{i,j} |\zeta_i - \zeta_j| \mathbb{1}_{edge(i,j)}. \tag{12}$$

The final combined loss function that we optimise is:

$$\mathcal{L} = \mathcal{L}_X + \mathcal{L}_{geoc} + \mathcal{L}_{reg}. \tag{13}$$

where $\mathcal{L}_X$ is $\mathcal{L}_{distr}$ or $\mathcal{L}_{contr}$. While the model is able to learn features that are semantically similar to real ones, the constraints of the representation prevent it from being able to replicate them perfectly. To bridge that final gap we map these synthetic features in minibatches to features found in real features using optimal transport, as in the Distribution Matching section.

3.3 Feature Adapter

Having learned a 3D representation that can reconstruct the features present in real images and structure them spatially in 3D, we can now use this representation to learn features for arbitrary camera views for the category of interest. Effectively, we now have the ability to generate paired data consisting of visual features with their respective per-pixel 3D locations on our learned 3D representation by rendering the 3D shape and passing the resulting images through the pre-trained feature extractor ϕ. We can exploit the learned 3D structure of our distribution of features to learn new features possessing this geometric information. Given this paired data we can train a feature adapter that takes features $\mathcal{F}$ as input and maps them to new features corresponding to predictions of the 3D canonical positions, Q, on the representation for each image patch, *i.e.* $Q = \chi(\mathcal{F})$, along with a predicted mask $\mathbf{M}_Q$. The learned 3D representation is not guaranteed to be geometrically similar to the category thus we rely on the adapter to learn the relative spatial pattern of features. To train this adapter we

adopt an approach similar to DualPM [16] and DUSt3R [39] by regularising an L2 prediction loss with a per-pixel confidence prediction c_Q [17]:

$$\mathcal{L}_{adp} = \frac{1}{|\mathbf{M}_Q|} \sum_u c_Q(u) ||\hat{Q}_u - Q_u||^2 - \lambda_c log(c_Q(u)), \tag{14}$$

where $\mathbf{M}_Q$ is the predicted mask, $\hat{Q}_u$ and Q_u are the ground truth and predicted position maps for masked feature tokens u, c_Q is the predicted confidence map, and λ_c is a hyperparameter. The predicted mask is trained using binary cross entropy with the known synthetic mask.

Once χ has been trained, we can then predict the 3D position on our canonical 3D representation for real images by first extracting their features using the pre-trained feature extractor ϕ, and then applying the adapter to get $Q = \chi(\phi(I))$. These 3D positions act as our features as they should correspond each original backbone feature $\mathbf{f}$ with a unique position on the representation. For finding correspondences between pixel locations on two different images I_1 and I_2 with corresponding features $\mathcal{F}^1$ and $\mathcal{F}^2$, we use a squared Euclidean metric, $d(\mathbf{f}_i^1, \mathbf{f}_j^2) = ||\chi(\mathcal{F}^1)_i - \chi(\mathcal{F}^2)_j||^2$ where i and j signify the i^{th} and j^{th} feature tokens in each map corresponding to the two pixel locations

Finally, when performance matching, we can exploit the complementary advantages of the original pre-trained features and our 3D ones by combining them. First we can remove features using the predicted mask $\mathbf{M}_Q$. If we then find the top k matches using the original pre-trained features, we can refine the selection using the closest match among these k matches from our model's predictions:

$$d_{tk}(\mathbf{f}_i^1, \mathbf{f}_j^2) = \begin{cases} ||\chi(\mathcal{F}^1)_i - \chi(\mathcal{F}^2)_j||^2 & \text{if TopK}((\mathbf{f}_i^1)^T \mathbf{f}_j^2, k) \\ \infty & \text{otherwise.} \end{cases} \tag{15}$$

4 Results

Here we report both quantitative and qualitative results across several different object categories and different ablated variants of our model. Implementation and training details can be found in the appendix.

4.1 Quantitative Results

We evaluate our model on the task of semantic correspondence estimation using the PCK@0.1 metric on PF-Pascal [12] for different quadruped categories. Results presented in Table 1. We evaluate on the two template learning approaches: (i) contrastive with contrastive loss on the synthetic features without distribution matching (Eqn. 4) and (ii) distributive, global distribution matching without the contrastive loss. The results show the optimisation is successful with either algorithm. PF-Pascal is a commonly used benchmark and allows us to compare to other similar models. We first compare to SphericalMaps [24] using the protocol and test split set out in [46]. This model interpolates its final features between its spherical features and the original DINOv2 features using

DINOv2 [27]	87.5	84.3	80.0
DINOv2 [27] PCA16	83.3	85.5	80.0
SphericalMaps [24] (sphere only)	64.6	61.5	80.0
SphericalMaps [24]	91.7	85.5	80.0
Ours Contrastive	81.6	73.5	60.0
Ours Distributive	83.3	72.3	60.0
Ours Contrastive (with topk)	97.9	80.7	80.0
Ours Distributive (with topk)	93.8	84.3	80.0

Table 1: Evaluation on PF-Pascal using PCK@0.1 for cosine similarity nearest neighbour from source to target. We follow other papers and use the image resolution when calculating the threshold rather than the bounding box. For topk, we used k=5 matches.

DINOv2 [27]	53.4	59.8	51.1
DINOv2 [27] PCA16	51.8	59.8	50.7
A-CSM [20]	26.3	32.9	28.6
MagicPony [40]	42.5	42.9	41.2
Farm3D [14]	40.2	49.1	36.1
DualPM [16] (supervised)	66.9	73.2	66.8
Ours Contrastive	46.7	51.0	34.7
Ours Distributive	43.9	49.9	28.6
Ours Contrastive (with topk)	57.1	60.5	54.3
Ours Distributive (with topk)	56.2	61.9	53.3

Table 2: Keypoint matching on PF-Pascal using PCK@0.1. A-CSM, MagicPony, Farm3D, and DualPM project their 3D reconstruction vertices onto each image and assign the source feature to the nearest projected vertex which is then matched to the feature it is projected onto in the target image. Like [16, 40], we use the image resolution when calculating the threshold rather than the bounding box. For topk, we used k=5 matches.

a interpolative factor α. We compare to $\alpha = 0.8$, the default value, and $\alpha = 0.0$ which is just the sphere features prediction, most comparable to our direct results. Our model performs comparably across all four categories despite no view labels being used. We then compare our model to the set of self-supervised reconstruction models, see Table 2. Note that the image pairs used to evaluate these are no longer available and so the comparison is not exact.

To try to recreate the protocol used in [16, 20], we sampled 10k pairs of images from the PF-Pascal full dataset and cropped around the bounding box of each instance of a category in the image. The cropped images were then resized to 256×256 and keypoint transfer calculated. The original bounding box was scaled by the crop scaling and 0.1 times the scaled bounding box used as the PCK threshold. Our model out performs each of the self-supervised methods MagicPony and Farm3D, self-supervised models explicitly designed to reconstruct quadrupeds. though does not quite match the highly-supervised DualPM model [16]. Observing the test images, we find the topk model matches features DINOv2 was unable to. In the appendix we show additional qualitative examples, showing that, for the cow example, the model is able to match both eyes while DINOv2 matched only one, and was able to better match the legs. This improvement in left-right disambiguation is quantified in Table 4 where we report superior matching performance on keypoints in AwAPose that have explicit left and right confusion (*e.g.* left versus right front leg).

4.2 Qualitative Results

There are two stages of our model to examine qualitatively. There is the 3D canonical model and its spatial distribution of features and there is the 3D

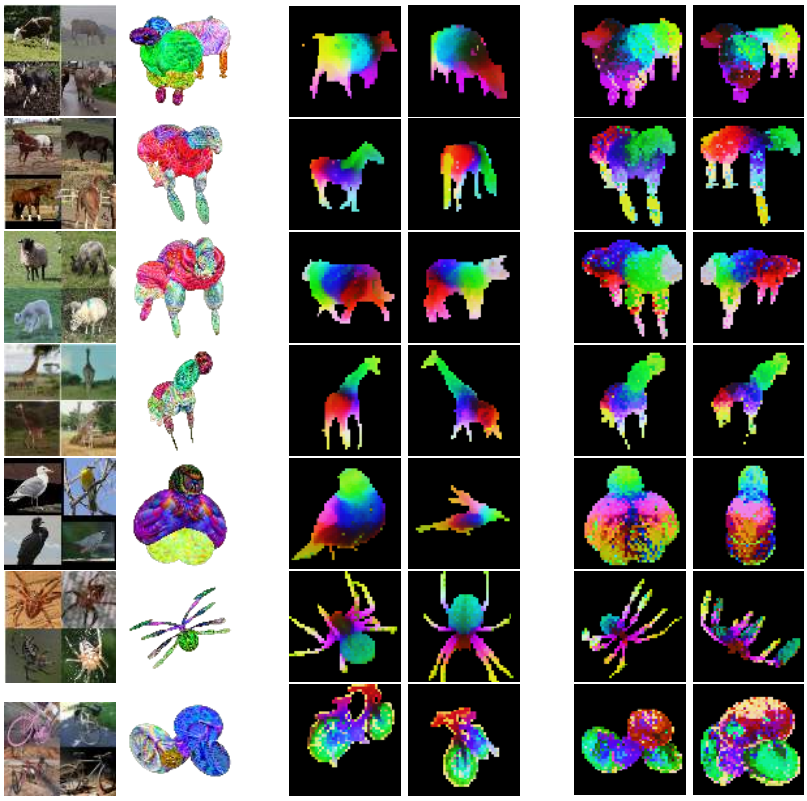

Fig. 3: Here we display the learned canonical 3D shape for different object categories using the Distributional approach. The first column shows representative training images for each category while the second column shows our learned canonical shape. The next two columns show DINOv2 features extracted from real images while the final two columns show DINOv2 features extracted for our synthetic images. The features are visualised using PCA. Our resulting features are trained to look like those from real images, but the appearance of the 3D shape is not constrained to look realistic.

position predictions from the adapter. We are aiming to learn a representation that can capture the total range of features present in real images and replicate their 3D spatial distribution such that they are sensibly placed and as consistent as possible across all views.

In Fig. 3 we display reconstruction results for different categories with varying topologies. The first three all share the exact same intialisation (*i.e.* the same input graph is provided) but develop into unique representations. In particular, each representation learns to develop a face on the texture that captures characteristics of its category so as to reconstruct features found in the dataset. The versatility of our approach enables feature reconstruction for categories that are commonly ignored by other models, *e.g.* spiders and bikes.

For the adapted features, we are primarily interested in whether we can discern semantic information that pre-trained features like DINOv2 struggle with,

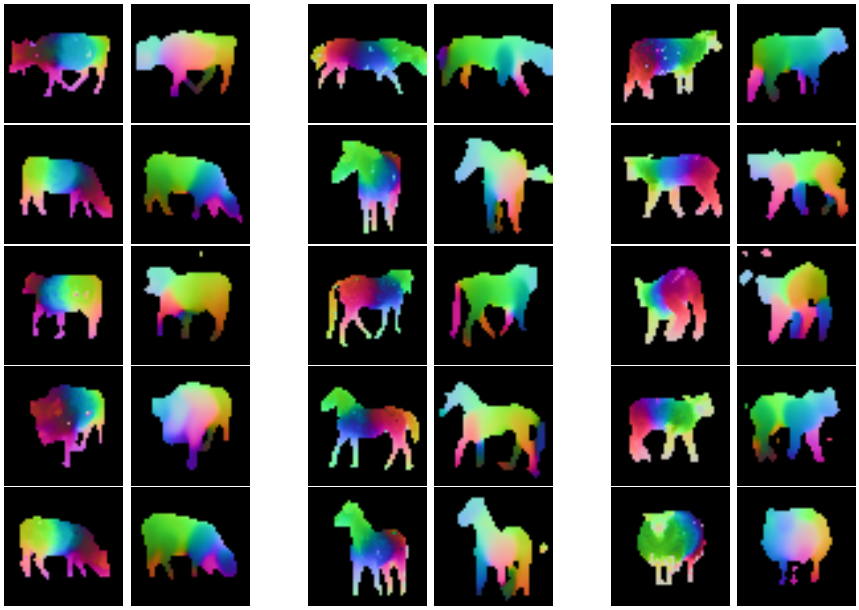

Fig. 4: Qualitative results of applying our adapter to DINOv2 features from real images using shapes learned using the Distributional approach. Columns from left to right, depict results for cows, horses, and sheep, where each row is a different image. For each column, the original DINOv2 features are on the left and the predicted 3D positions on the template representation are on the right. Our results represent the predicted 3D coordinates mapping to a location on the learned template. Unlike the original DINOv2 features, which in many instances are unable to encode left-right disambiguation, our model much more clearly separates such parts.

in particular telling left from right and, for animals, identifying each of the legs. Results for the refinement adapter are shown in Fig. 4 for three categories: cows, horses, and sheep. The adapted features for all three categories display a strong refinement of the original features, partially disambiguating left from right, with the various legs often differentiated. One of the issues we found for many categories is that our model can sometimes confuse a tail with a head or merges the tail with the back legs, a problem that can occur for real images too.

4.3 Ablations

In Table 3 we ablate our primary losses: the geometric regularisations. We also test the performance of the initial 3D prior, *i.e.* the representation prior to training, as well as the use of a version of our model that only estimates features using a single sphere (*i.e.* as in [24]), as opposed to a collection of primitives. We compare these against both our topk model, as well as the direct model without topk. However, it is worth bearing in mind that the PF-Pascal test set is small, and so to further evaluate these versions of the model, we also ablate them on the AwAPose dataset [4] where a similar trend is seen. Combining both Contrastive and Distributional approaches worsened performance. Empirically these models

	PFP.	AwA
Initial 3D Prior	7.2	1.8
$\mathcal{L}_{contr} + \mathcal{L}_{geoc}$	37.3	16.0
$\mathcal{L}_{distr} + \mathcal{L}_{geoc}$	31.3	13.8
$\mathcal{L}_{contr} + \mathcal{L}_{geoc} + \mathcal{L}_{reg}$ w/ sphere prior	34.9	15.8
$\mathcal{L}_{distr} + \mathcal{L}_{geoc} + \mathcal{L}_{reg}$ w/ sphere prior	62.7	34.6
$\mathcal{L}_{reg}$	26.5	14.0
$\mathcal{L}_{distr} + \mathcal{L}_{contr} + \mathcal{L}_{geoc} + \mathcal{L}_{reg}$	55.4	31.6
$\mathcal{L}_{contr} + \mathcal{L}_{geoc} + \mathcal{L}_{reg}$	73.5	51.6
$\mathcal{L}_{distr} + \mathcal{L}_{geoc} + \mathcal{L}_{reg}$	72.3	43.3

Table 3: Keypoint matching scores on horses in PF-Pascal (PFP) and AwAPose (AwA) for ablations of the model, evaluated using PCK@0.1. The poor performance of the initial prior shows the gains of the optimisation while the sphere prior shows having a more complex prior can improve results. Geometric regularisation, is necessary but not sufficient.

	Cow	Horse	Sheep	Giraffe	Zebra
DINOv2	48.7	54.7	56.3	69.9	59.0
DINOv2 (pca16)	46.0	53.9	48.6	67.1	59.4
Ours Contrastive	48.5	41.8	43.6	40.9	36.0
Ours Distributive	46.4	36.6	39.9	50.5	25.4
Ours Contrastive (topk)	61.5	64.3	61.3	67.4	61.3
Ours Distributive (topk)	61.4	55.7	60.4	68.0	59.1

Table 4: Matching scores for keypoints with left and right equivalents in AwAPose using PCK@0.1. In most instances we improve left-right disambiguation compared to DINOv2.

appeared to be more susceptible to developing multiple heads and possibly work against one another.

4.4 Limitations

While our approach can approximate the distribution of real image features, there can still be inconsistencies between very similar rendered views. For example, the front legs for quadrupeds can frequently change from having front leg features to back leg features to front again. There can also be artifacts due to sets of features being very similar, *e.g.* the head of an animal can sometimes gain tail features scattered among the head features. This is worse depending on how similar the original backbone features are for that category. Finally, our approach is simple by design, but as a result does not explicitly model significant object shape variation or articulations. Doing so, would necessitate a significantly more complex model, and our goal is to demonstrate effective feature learning from a simple initial 3D prior.

5 Conclusion

Reliably distinguishing object parts is an open challenge and most previous methods rely on using extensive supervision to address it. We presented a new lightweight self-supervised approach for extracting more 3D consistent part-centric visual features from a pre-trained feature extractor. We achieve this by optimising a category level shape and appearance prior and show that this can be used to train an adapter to disambiguate repeated and symmetric parts for target categories. We demonstrate competitive performance on the task of semantic correspondence estimation using two variants of our model on a variety of different object categories.

References

1. Amir, S., Gandelsman, Y., Bagon, S., Dekel, T.: Deep vit features as dense visual descriptors. In: ECCV Workshops (2021)
2. Aygün, M., Mac Aodha, O.: Demystifying unsupervised semantic correspondence estimation. In: ECCV (2022)
3. Aygun, M., Mac Aodha, O.: Saor: Single-view articulated object reconstruction. In: CVPR (2024)
4. Banik, P., Li, L., Dong, X.: A Novel Dataset for Keypoint Detection of Quadruped Animals from Images. arXiv:2108.13958 (2021)
5. Barel, N., Weber, R.S., Mualem, N., Finder, S.E., Freifeld, O.: Spacejam: a lightweight and regularization-free method for fast joint alignment of images. In: ECCV (2024)
6. Bellemare, M.G., Danihelka, I., Dabney, W., Mohamed, S., Lakshminarayanan, B., Hoyer, S., Munos, R.: The cramer distance as a solution to biased wasserstein gradients. arXiv:1705.10743 (2017)
7. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: ICCV (2021)
8. Genevay, A., Peyré, G., Cuturi, M.: Learning Generative Models with Sinkhorn Divergences. In: AISTATS (2018)
9. Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., Smola, A.: A kernel method for the two-sample-problem. NeurIPS (2006)
10. Güler, R.A., Neverova, N., Kokkinos, I.: Densepose: Dense human pose estimation in the wild. In: CVPR (2018)
11. 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)
12. Ham, B., Cho, M., Schmid, C., Ponce, J.: Proposal flow: Semantic correspondences from object proposals. PAMI (2017)
13. Hirschorn, O., Avidan, S.: A graph-based approach for category-agnostic pose estimation. In: ECCV (2024)
14. Jakab, T., Li, R., Wu, S., Rupprecht, C., Vedaldi, A.: Farm3d: Learning articulated 3d animals by distilling 2d diffusion. In: 3DV (2024)
15. Karmali, T., Atrishi, A., Harsha, S.S., Agrawal, S., Jampani, V., Babu, R.V.: Lead: Self-supervised landmark estimation by aligning distributions of feature similarity. In: WACV (2022)
16. Kaye, B., Jakab, T., Wu, S., Ruprecht, C., Vedaldi, A.: Dualpm: dual posed-canonical point maps for 3d shape and pose reconstruction. In: CVPR (2025)
17. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? In: NeurIPS (2017)
18. Kim, J., Gu, G., Park, M., Park, S., Choo, J.: Stableviton: Learning semantic correspondence with latent diffusion model for virtual try-on. In: CVPR (2024)
19. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. In: ICCV (2023)
20. Kulkarni, N., Gupta, A., Fouhey, D.F., Tulsiani, S.: Articulation-aware canonical surface mapping. In: CVPR (2020)
21. Li, Z., Litvak, D., Li, R., Zhang, Y., Jakab, T., Rupprecht, C., Wu, S., Vedaldi, A., Wu, J.: Learning the 3D Fauna of the Web. In: CVPR (2024)
22. Luo, G., Dunlap, L., Park, D.H., Holynski, A., Darrell, T.: Diffusion hyperfeatures: Searching through time and space for semantic correspondence. In: NeurIPS (2023)

23. Mariotti, O., Du, Z., Bhalgat, Y., Mac Aodha, O., Bilen, H.: Jamais vu: Exposing the generalization gap in supervised semantic correspondence. In: NeurIPS (2025)
24. Mariotti, O., Mac Aodha, O., Bilen, H.: Improving semantic correspondence with viewpoint-guided spherical maps. In: CVPR (2024)
25. Neverova, N., Novotny, D., Szafraniec, M., Khalidov, V., Labatut, P., Vedaldi, A.: Continuous surface embeddings. In: NeurIPS (2020)
26. Ofri-Amar, D., Geyer, M., Kasten, Y., Dekel, T.: Neural congealing: Aligning images to a joint semantic atlas. In: CVPR (2023)
27. Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: DINOv2: Learning robust visual features without supervision. TMLR (2024)
28. Peebles, W., Zhu, J.Y., Zhang, R., Torralba, A., Efros, A.A., Shechtman, E.: Gan-supervised dense visual alignment. In: CVPR (2022)
29. Peyré, G., Cuturi, M.: Computational optimal transport. Foundations and Trends in Machine Learning (2019)
30. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022)
31. Salimans, T., Zhang, H., Radford, A., Metaxas, D.: Improving gans using optimal transport. In: ICLR (2018)
32. Shtedritski, A., Rupprecht, C., Vedaldi, A.: Shic: Shape-image correspondences with no keypoint supervision. In: ECCV (2024)
33. Shtedritski, A., Vedaldi, A., Rupprecht, C.: Learning universal semantic correspondences with no supervision and automatic data curation. In: ICCV Workshops (2023)
34. Siméoni, O., Vo, H.V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., Yi, S.E., Ramamonjisoa, M., Massa, F., HAZIZA, D., Wehrstedt, L., Wang, J., Darcet, T., Moutakanni, T., Sentana, L., Roberts, C., Vedaldi, A., Tolan, J., Brandt, J., Couprie, C., Mairal, J., Jegou, H., Labatut, P., Bojanowski, P.: DINOv3. TMLR (2026)
35. Tang, L., Jia, M., Wang, Q., Phoo, C.P., Hariharan, B.: Emergent correspondence from image diffusion. NeurIPS (2023)
36. Tang, L., Wertheimer, D., Hariharan, B.: Revisiting pose-normalization for fine-grained few-shot recognition. In: CVPR (2020)
37. Tsagkas, N., Rome, J., Ramamoorthy, S., Mac Aodha, O., Lu, C.X.: Click to grasp: Zero-shot precise manipulation via visual diffusion descriptors. In: IROS (2024)
38. Wandel, K., Wang, H.: SemAlign3D: Semantic Correspondence between RGB-Images through Aligning 3D Object-Class Representations. In: CVPR (2025)
39. Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: Dust3r: Geometric 3d vision made easy. In: CVPR (2024)
40. Wu, S., Li, R., Jakab, T., Rupprecht, C., Vedaldi, A.: Magicpony: Learning articulated 3d animals in the wild. In: CVPR (2023)
41. Xiao, T., Liu, S., De Mello, S., Yu, Z., Kautz, J., Yang, M.H.: Learning contrastive representation for semantic correspondence. IJCV (2022)
42. Yao, C.H., Hung, W.C., Li, Y., Rubinstein, M., Yang, M.H., Jampani, V.: Lassie: Learning articulated shapes from sparse image ensemble via 3d part discovery. NeurIPS (2022)
43. Yao, C.H., Hung, W.C., Li, Y., Rubinstein, M., Yang, M.H., Jampani, V.: Hi-lassie: High-fidelity articulated shape and skeleton discovery from sparse image ensemble. In: CVPR (2023)

44. Yue, Y., Das, A., Engelmann, F., Tang, S., Lenssen, J.E.: Improving 2d feature representations by 3d-aware fine-tuning. In: ECCV (2024)
45. Zhang, H., Xu, L., Lai, S., Shao, W., Zheng, N., Luo, P., Qiao, Y., Zhang, K.: Open-vocabulary animal keypoint detection with semantic-feature matching. IJCV (2024)
46. Zhang, J., Herrmann, C., Hur, J., Chen, E., Jampani, V., Sun, D., Yang, M.H.: Telling left from right: Identifying geometry-aware semantic correspondence. In: CVPR (2024)
47. Zhang, J., Herrmann, C., Hur, J., Polania Cabrera, L., Jampani, V., Sun, D., Yang, M.H.: A tale of two features: Stable diffusion complements dino for zero-shot semantic correspondence. NeurIPS (2023)