Joint Deep Multi-Graph Matching and 3D Geometry Learning from Inhomogeneous 2D Image Collections

Zhenzhang Ye[1], **Tarun Yenamandra**[1], **Florian Bernard**[1, 2], **Daniel Cremers**[1]
[1]Technical University of Munich
[2]University of Bonn
zhenzhang.ye@tum.de, tarun.yenamandra@tum.de, f.bernard@gmail.com, cremers@tum.de

Abstract

Graph matching aims to establish correspondences between vertices of graphs such that both the node and edge attributes agree. Various learning-based methods were recently proposed for finding correspondences between image key points based on deep graph matching formulations. While these approaches mainly focus on learning node and edge attributes, they completely ignore the 3D geometry of the underlying 3D objects depicted in the 2D images. We fill this gap by proposing a trainable framework that takes advantage of graph neural networks for learning a deformable 3D geometry model from inhomogeneous image collections, i.e., a set of images that depict different instances of objects from the same category. Experimentally, we demonstrate that our method outperforms recent learning-based approaches for graph matching considering both accuracy and cycle-consistency error, while we in addition obtain the underlying 3D geometry of the objects depicted in the 2D images.

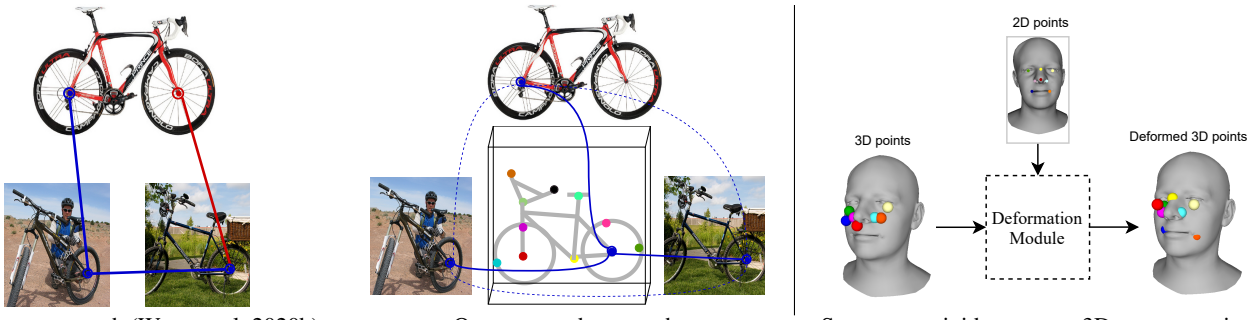
Introduction

Graph matching is a widely studied problem in computer vision, graphics and machine learning due to its universal nature and the broad range of applications. Intuitively, the objective of graph matching is to establish correspondences between the nodes of two given weighted graphs, so that the weights of corresponding edges agree as well as possible. Diverse visual tasks fit into the graph matching framework. In this work we focus in particular on the task of matching 2D key points defined in images, which has a high relevance for 3D reconstruction, tracking, deformation model learning, and many more. In this case, a graph is constructed for each image by using the key points as graph nodes, and by connecting neighbouring key points with edges, according to some suitable neighbourhood criterion. The edges contain information about geometric relations, such as the Euclidean distance between nodes in the simplest case.

Image key point matching was traditionally addressed based on finding nearest neighbours between feature descriptors such as SIFT (Lowe 2004), SURF (Bay et al. 2008). A downside to this approach is that the geometric relation between the key points are completely ignored, which is in particular problematic if there are repetitive structures that lead to similar feature descriptors. Instead, we can use a graph matching formulation to establish correspondences between key points while taking into account geometric relations between points. Yet, the sequential nature of first computing features and then bringing them into correspondence may lead to sub-optimal results, since both tasks are solved independently from each other – despite their mutual dependence. More recently, several deep learning-based graph matching methods have been proposed that learn task-specific optimal features while simultaneously solving graph matching in an end-to-end manner (Zanfir and Sminchisescu 2018; Wang, Yan, and Yang 2019a; Wang et al. 2020b; Rolínek et al. 2020). While such deep graph matching approaches lead to state-of-the-art results in terms of the matching accuracy, they have profound disadvantages, particularly in the context of 2D key point matching in image collections. On the one hand, most existing approaches only consider the matching of pairs of images, rather than the entire collection. This has the negative side-effect that so-obtained matchings are generally not cycle-consistent. To circumvent this, there are approaches that use a post-processing procedure (Wang, Yan, and Yang 2019b) to establish cycle consistency based on permutation synchronisation (Pachauri, Kondor, and Singh 2013; Bernard et al. 2018). Yet, they do not directly obtain cycle-consistent matchings but rather achieve it based on post-processing. On the other hand, and perhaps more importantly, approaches that use graph matching for 2D image key point matching have the strong disadvantage that the underlying 3D structure of the objects whose 2D projections are depicted in the images is not adequately considered. In particular, the spatial relations in the 2D image plane are highly dependent on the 3D geometric structure of the object, as well as on the camera parameters. Hence, learning graph features directly based on the image appearance and/or 2D image coordinates is sub-optimal, at best, since the neural network implicitly needs to learn the difficult task of reasoning about the underlying 3D structure.

In this work we address these issues by proposing a deep multi-graph matching approach that learns the 3D structure of objects. The main contributions are as follows:

- For the first time we propose a solution for jointly considering multi-graph matching and inferring 3D geometry from inhomogeneous 2D image collections, see Fig. 1.



Previous approach (Wang et al. 2020b) | Our proposed approach | Sparse non-rigid geometry 3D reconstruction

Figure 1: We consider a deep graph matching approach for bringing 2D image key points into correspondence. Left: Existing deep graph matching methods completely ignore the underlying 3D geometry of the 3D objects depicted in the 2D images. In addition, they lead to cycle errors, as shown by the red line. Middle: Our method obtains the underlying 3D geometry from a collection of inhomogeneous 2D images (indicated by the coloured points and the bike sketch in the centre), while at the same time guaranteeing cycle consistency. Right: To model nonlinear 3D object deformations, we infer coarse 3D geometry and in addition use a 3D deformation module to refine the underlying 3D geometry based on the 2D image key point observations.

- To effectively deal with the inhomogeneity of the image collection, in which different instances of objects of the same category are present (e.g. different types of bikes as shown in Fig. 1), we introduce a novel deformable 3D model that we directly learn from the image collection based on a graph neural network.
- Rather than performing pairwise image-to-image matching, we consider an image-to-deformable-3D-model matching formulation to guarantee cycle consistency.
- Our approach substantially outperforms the previous state of the art in learning-based graph matching approaches considering accuracy and cycle error.

Related Work

In the following we summarise the works that we consider most relevant to our approach. For a more detailed background on image key point matching we refer interested readers to the recent survey paper by Ma et al. (2021).

Feature-Based Matching. Feature descriptors extracted from images at key point locations, e.g. based on SIFT (Lowe 2004), SURF (Bay et al. 2008), or deep neural networks (Krizhevsky, Sutskever, and Hinton 2012), are often used for image matching. In order to bring extracted features into correspondence, commonly a nearest neighbour strategy (Bentley 1975) or a linear assignment problem (LAP) formulation are used (Burkard, Dell'Amico, and Martello 2012). However, these methods suffer from the problem that geometric relations between the key points in the images are not taken into account.

Graph Matching and Geometric Consistency. Geometric relations can be taken into account by modelling feature matching as graph matching problem. Here, the image key points represent the graph nodes, and the edges in the graph encode geometric relations between key points (e.g. spatial distances). Mathematically, graph matching can be phrased in terms of the quadratic assignment problem (Lawler 1963; Pardalos, Rendl, and Wolkowitz 1994; Loiola et al. 2007; Burkard, Dell'Amico, and Martello 2012). There are many existing works for addressing the graph matching problem in visual computing, including Cour, Srinivasan, and Shi (2006); Zhou and De la Torre (2016); Swoboda et al. (2017); Dym, Maron, and Lipman (2017); Bernard, Theobalt, and Moeller (2018); Swoboda et al. (2017). A drawback of these approaches is that they mostly rely on handcrafted graph attributes and/or respective graph matching cost functions based on affinity scores. In Zhang et al. (2013), a learning-based approach that directly obtains affinity scores from data was introduced. The differentiation of the power iteration method has been considered in a deep graph matching approach (Zanfir and Sminchisescu 2018). A more general blackbox differentiation approach was introduced by Rolínek et al. (2020). Various other deep learning approaches have been proposed for graph matching (Li et al. 2019; Fey et al. 2020), and some approaches also address image key point matching (Wang, Yan, and Yang 2019a; Zhang and Lee 2019; Wang et al. 2020b). In this case, optimal graph features are directly learned from the image appearance and/or 2D image coordinates, while simultaneously solving graph matching in an end-to-end manner. Although these methods consider geometric consistency, they are tailored towards matching a pair of graphs and thus lead to cycle-inconsistent matchings when pairwise matchings of more than two graphs are computed.

Synchronisation and Multi-Matching. Cycle-consistency is often obtained as a post-processing step after obtaining pairwise matchings. The procedure to establish cycle consistency in the set of pairwise matchings is commonly referred to as permutation synchronisation (Pachauri, Kondor, and Singh 2013; Zhou, Zhu, and Daniilidis 2015; Maset, Arrigoni, and Fusiello 2017; Bernard et al. 2018; Birdal and Simsekli 2019; Bernard, Cremers, and Thunberg 2021). There are also methods for directly obtaining cycle-consistent multi-matchings (Tron et al. 2017; Wang, Zhou, and Daniilidis 2018; Bernard et al. 2019). Recently, permutation synchronisation has been considered in a deep graph matching framework,

where a separate permutation synchronisation module is utilised to generalise a two-graph matching approach to the matching of multiple graphs (Wang, Yan, and Yang 2019b). However, when applying such multi-matching approaches to image key point matching they have the significant shortcoming that they ignore the underlying 3D geometry of the 2D points. This makes it extremely difficult to establish correct matchings across images, which after all depict 2D projections of 3D objects in different poses, possibly even under varying perspective projections. This also applies to the recent method by Wang, Yan, and Yang (2020), which simultaneously considers graph matching and clustering.

3D Reconstruction. 3D reconstruction obtains geometric information from 2D data. When relying on single-view input only, it is generally an ill-posed problem. Reconstruction from a single image or video using a deformable 3D prior has for example been achieved by fitting a 3D morphable model of a specific object class such as humans bodies, faces, or cars, and then finding the parameters of the model that best explain the image (Tewari et al. 2017; Bogo et al. 2016; Wang et al. 2020a). However, the availability of a suitable 3D prior is a rather strong assumption.

An alternative to address the ill-posedness of single-view reconstruction is to consider multiple views. Recent methods for multi-view reconstruction assume camera parameters and use self-supervised learning based on a neural renderer to reconstruct static and dynamic objects with novel 3D representations (Mildenhall et al. 2020; Park et al. 2020). A downside of multi-view reconstruction methods is that they require many different images of the same object, which is often unavailable in existing datasets.

Contrary to existing approaches, we simultaneously solve deep multi-graph matching and infer sparse 3D geometry from inhomogeneous 2D image collections. Our approach obtains cycle-consistent multi-matchings and does not rely on a hand-crafted template or any other prior 3D model.

Problem Formulation & Preliminaries

In this section we summarise how to achieve cycle-consistency for multiple graph matching by utilising the notion of universe points. In order to explicitly construct such universe points, we consider the sparse reconstruction of 3D key points from multiple 2D images.

Multi-Matching and Cycle Consistency. Given is the set $\{\mathcal{G}_j\}_{j=1}^N$ of N undirected graphs, where each graph $\mathcal{G}_j = (\mathcal{V}_j, \mathcal{E}_j)$ comprises of a total of m_j nodes $\mathcal{V}_j = \{v_1, \dots, v_{m_j}\}$ and n_j edges $\mathcal{E}_j = \{e_1, \dots, e_{n_j}\}$ that connect pairs of nodes in $\mathcal{V}_j$. We assume that each node represents an image key point, and that the node $v_i \in \mathbb{R}^2$ is identified with the respective 2D image coordinates. The pairwise graph matching problem is to find a node correspondence $X_{jk} \in \mathbb{P}_{m_j m_k}$ between $\mathcal{G}_j$ and $\mathcal{G}_k$. Here, $\mathbb{P}_{m_j m_k}$ is the set of $(m_j \times m_k)$-dimensional partial permutation matrices.

Let $\mathcal{X} = \{X_{jk} \in \mathbb{P}_{m_j m_k}\}_{j,k=1}^N$ be the set of pairwise matchings between all graphs in $\{\mathcal{G}_j\}_{j=1}^N$. $\mathcal{X}$ is said to be cycle-consistent if for all $j, k, l \in \{1, \dots, N\}$, the following properties hold (Huang and Guibas 2013; Tron et al. 2017; Bernard et al. 2018):

1. $X_{jj} = \mathbb{I}_{m_j}$, with the $m_j \times m_j$ identity matrix $\mathbb{I}_{m_j}$.
2. $X_{jk} = X_{kj}^T$.
3. $X_{jk} X_{kl} \leq X_{jl}$ (element-wise comparison).

When solving multi-graph matchings with pairwise matching, cycle consistency is desirable since it is an intrinsic property of the (typically unknown) ground truth matching. Rather then explicitly imposing the above three constraints, it is possible to achieve cycle consistency by representing the pairwise matching using a universe graph (Huang and Guibas 2013; Tron et al. 2017; Bernard et al. 2018):

Lemma 1 *The set $\mathcal{X}$ of pairwise matchings is cycle-consistent if there exists a collection $\{X_j \in \mathbb{P}_{m_j d} : X_j \mathbf{1}_d = \mathbf{1}_{m_j}\}_{j=1}^N$ such that $\forall X_{jk} \in \mathcal{X}$ it holds that $X_{jk} = X_j X_k^T$.*

Here, the X_j is the pairwise matching between the graph $\mathcal{G}_j$ and a universe graph $\mathcal{U} = (\mathcal{V}, \mathcal{E})$ with d universe points, where $\mathcal{V} = \{u_1, \dots, u_d\}$ denote the universe points and $\mathcal{E} = \{e_1, \dots, e_n\}$ the universe edges. Intuitively, the universe graph can be interpreted as assigning each point in $\mathcal{G}_j$ to one of the d universe points in $\mathcal{U}$. Therefore, rather than modelling the cubic number of cycle consistency constraints on $\{\mathcal{G}_j\}_{j=1}^N$ explicitly, we use an object-to-universe matching formulation based on the $\{X_j\}_{j=1}^N$.

3D Reconstruction. Though the idea of the universe graph is a crucial ingredient for synchronisation approaches (Pachauri, Kondor, and Singh 2013; Huang and Guibas 2013; Bernard et al. 2018), the universe graph is never explicitly instantiated in these methods. That is because it is merely used as an abstract entity that must exist in order to ensure cycle consistency in multi-matchings. Considering that the graphs in this work come from image collections, we assume that the nodes $u_i \in \mathbb{R}^3$ of the universe graph represent 3D points, which will allow us to address their explicit instantiation based on multiple-view geometry.

We denote the homogeneous coordinate representation of the universe point $u_i \in \mathbb{R}^3$ (represented in world coordinates) as $U_i = (u_i, 1) \in \mathbb{R}^4$. Its projection onto the j-th image plane, denoted by $\mathcal{V}_{ij} = (v_{ij}, 1) \in \mathbb{R}^3$, is given by

$$\mathcal{V}_{ij} = \lambda_{ij} K_j \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{\Pi_0} \underbrace{\begin{pmatrix} R_j & T_j \\ 0 & 1 \end{pmatrix}}_{g_j} U_i. \quad (1)$$

Here, g_j is the world-to-camera space rigid-body transformation comprising of the rotation $R_j \in \mathbb{R}^{3\times3}$ and the translation $T_j \in \mathbb{R}^3$, Π_0 is the canonical projection matrix, $K_j \in \mathbb{R}^{3\times3}$ is the intrinsic camera matrix, and $\lambda_{ij} \in \mathbb{R}$ is the scale parameter. For brevity, we define the general projection matrix $\Pi_j = K_j \Pi_0 g_j$. Let $U \in \mathbb{R}^{4\times d}$ be the stacked universe points in homogeneous coordinates, $\mathcal{V}_j \in \mathbb{R}^{3\times d}$ be the respective projection onto the j-th image plane, and $\Lambda_j = \mathrm{diag}(\lambda_{1j}, \dots, \lambda_{dj}) \in \mathbb{R}^{d\times d}$ be the diagonal scale matrix. The matrix formulation of Eq. (1) is

$$\mathcal{V}_j = \Pi_j U \Lambda_j. \quad (2)$$

Once we have a collection of N images of different objects from the same category (not necessarily the same object instance, e.g. two images of different bicycles), reconstructing

the universe points U can be phrased as solving Eq. (2) in the least-squares sense, which reads

$$\arg\min_U \sum_{j=1}^{N} ||\Pi_j U \Lambda_j - \mathcal{V}_j||_F^2. \quad (3)$$

Note that in practice the variables $U, \{\Lambda_j\}$ and $\{\Pi_j\}$ are generally unknown, so that without further constraints this is an under-constrained problem. In the next section, we will elaborate on how we approach this.

Proposed Method

Our learning framework consists of four main components. The first two components have the purpose to obtain 3D universe points, along with a deformation of these 3D points representing the underlying 3D structure of the 2D key points in the j-th image. The purpose of the other two components is to predict the matching between the 2D points of $\mathcal{G}_j$ and the 3D points of $\mathcal{U}$. Thus, rather than learning pairwise matchings between $\mathcal{G}_j$ and $\mathcal{G}_k$, we utilise an object-to-universe matching formulation. Therefore, the underlying 3D structure and cycle-consistent multi-matchings are both attained by our method. The whole pipeline is illustrated in Fig. 2 and comprises the following four main components:

1. **Learnable 3D Universe Points**: the 2D key points $\{\mathcal{V}_j\}_{j=1}^N$ of all images in the collection are used to reconstruct the 3D universe points U by incorporating a reconstruction loss that approximates Eq. (3).
2. **Deformation Module**: the retrieved universe points U are static and therefore they cannot accurately model the geometric variability present in different instances of an object from the same category (e.g. different bicycles). To address this, the universe points are non-linearly deformed by the deformation module that takes the 2D points and the (learned) 3D universe points as input.
3. **Assignment Graph Generation**: by connecting the 2D and universe points, respectively, the 2D graph and the 3D universe graph are constructed. The assignment graph is then constructed as the product of these two graphs.
4. **Graph Matching Network**: a graph matching network performs graph convolutions on the assignment graph, and eventually performs a binary node classification on the assignment graph representing the matching between the 2D graph and the universe graph.

Learnable 3D Universe Points. As discussed above, the universe points can be retrieved by minimising (3). This problem, however, is generally under-determined, since $U, \{\Lambda_j\}$ and $\{\Pi_j\}$ in (3) are generally unknown in most practical settings. Additionally, although all objects share a similar 3D geometry, the nonlinear deformations between different instances are disregarded in (3). Thus, instead of an exact solution we settle for an approximation that we later refine in our pipeline. To this end, we assume a weak perspective projection model, i.e. all universe points are assumed to have the same distance from the camera. With this condition, the diagonal of Λ_j is constant and can be absorbed into Π_j. This leads to the least-squares problem

$$\arg\min_U \sum_{j=1}^{N} ||\Pi_j U - \mathcal{V}_j||_F^2, \quad (4)$$

which can be solved in an end-to-end manner during network training based on 'backpropagable' pseudo-inverse implementations. The variable Π_j can be expressed as $\Pi_j = \mathcal{V}_j U^+$, where U^+ is the right pseudo-inverse that satisfies $UU^+ = \mathbb{I}_4$. Therefore, we solve the following problem

$$U^* = \arg\min_U \frac{1}{N} \sum_{j=1}^{N} ||\mathcal{V}_j U^+ U - \mathcal{V}_j||_F^2. \quad (5)$$

Deformation Module. The universe points retrieved in the previous step can only reflect the coarse geometric structure of the underlying 3D object, but cannot represent finer-scale variations between different instances within a particular object category. Thus, we introduce the deformation module to model an additional nonlinear deformation.

This module takes the universe points U and the 2D points $\mathcal{V}_j$ as input. As shown in the bottom left of Fig. 2, $\mathcal{V}_j$ is passed to a 2D Point Encoder. The encoder first performs a nonlinear feature transform of all input points based on multi-layer perceptron (MLP), and then performs a max pooling to get a global feature representing the input object. As can be seen in the top left in Fig. 2, an MLP is utilised to perform a nonlinear feature transform for each of the 3D points in U. Each 3D point feature is then concatenated with the same global feature from the 2D Point Encoder. The concatenated per 3D point features are fed into an MLP to compute the deformation of each point. The output is a set of per-point offsets $S \in \mathbb{R}^{3\times d}$ that are added to U to generate the deformed 3D universe points. The computation of the per-point offsets is summarised as

$$S_j = \text{MLP}\left(\text{MLP}(U) \circ \text{Encoder}(\mathcal{V}_j)\right), \quad (6)$$

where $\circ$ represents the concatenation operation.

We enforce that the projection of the deformed universe points onto the image plane should be close to the observed 2D points, similar to the reconstruction loss in Eq. (5). Since the static 3D universe points should reflect the rough geometry of the underlying 3D object, the offset S_j should be small. Therefore, we introduce the deformed reconstruction loss and the offset regulariser as

$$\mathcal{L}_{\text{def}} = \frac{1}{N} \sum_{j=1}^{N} ||\mathcal{V}_j (U^* + S_j)^+ (U^* + S_j) - \mathcal{V}_j||_F^2, \text{ and } \quad (7)$$

$$\mathcal{L}_{\text{off}} = ||S_j||_F^2. \quad (8)$$

Assignment Graph Generation. To obtain graphs from the 2D points and the deformed 3D universe points, respectively, we utilise the Delaunay algorithm (Botsch et al. 2010) to generate edges, see Fig. 2. Moreover, we define the attribute of each edge as the concatenation of the coordinates of the respective adjacent points. Note that other edge generation methods and attributes can be utilised as well.

Once the 3D universe graph $\mathcal{U}$ and the 2D graph $\mathcal{G}_j$ are generated, we construct the assignment graph $\mathcal{G}_j^A$ as the

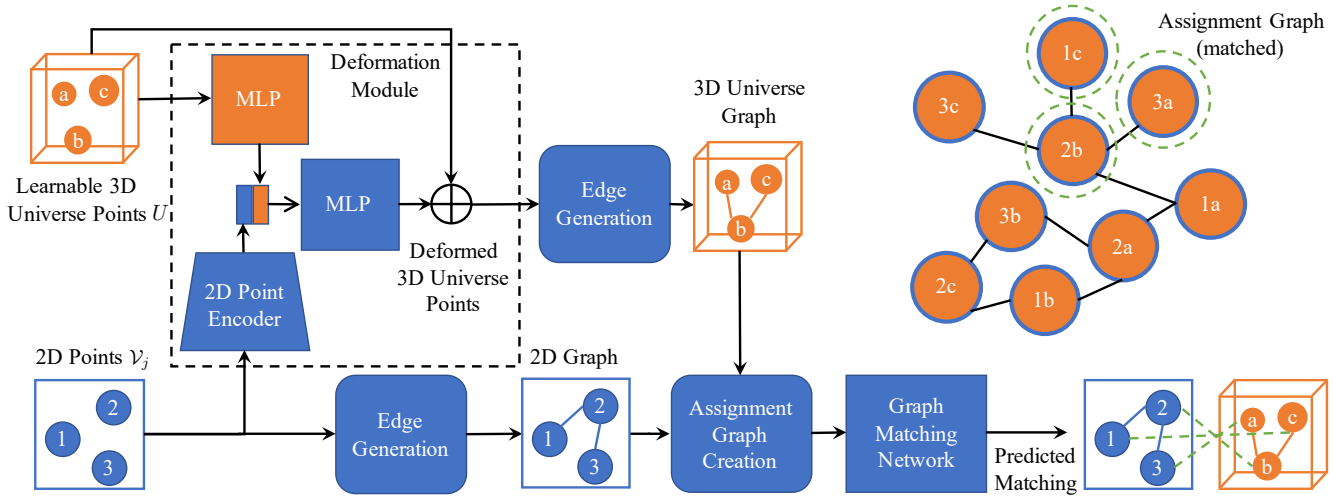


Figure 2: Overview of our algorithm. Given an image with 2D key points, we infer the corresponding image-specific 3D points in terms of a deformation of 3D universe points. The universe 3D points are learned during training for a given class of objects, while the deformations are predicted per image. We create edges and find a matching between the two graphs using a graph matching network. Since the matchings are between universe points and images, our matchings are intrinsically cycle consistent.

product graph of $\mathcal{U}$ and $\mathcal{G}_j$ following Leordeanu and Hebert (2005). To be more specific, the nodes in $\mathcal{G}_j^A$ are defined as the product of the two node sets $\mathcal{V}_j$ (of $\mathcal{G}_j$) and $\mathcal{V}$ (of $\mathcal{U}$), respectively, i.e. $\mathcal{V}_j^A = \{v_{jk} : v_{jk} = (v_j, u_k) \in \mathcal{V}_j \times \mathcal{V}\}$. The edges in $\mathcal{G}_j^A$ are built between nodes $v_{jk}, v_{mn} \in \mathcal{V}_j^A$ if and only if there is an edge between v_j and v_m in $\mathcal{E}_j$, as well as between u_k and u_n in $\mathcal{E}$. The attribute of each node and edge in $\mathcal{G}_j^A$ is again the concatenation of the attribute of corresponding nodes and edges in $\mathcal{G}_i$ and $\mathcal{U}$, respectively.

Graph Matching Network. The graph matching problem is converted to a binary classification problem on the assignment graph $\mathcal{G}^A$. For example, an assignment graph is shown on the top right of Fig. 2. Classifying nodes $\{1c, 2b, 3a\}$ as positive equals to matching point 1 to c, 2 to b and 3 to a, where numeric nodes correspond to the 2D graph, and alphabetic nodes correspond the 3D universe graph.

The assignment graph is then passed to the graph matching network (Wang et al. 2020b). A latent representation is achieved by alternatingly applying edge convolutions and node convolutions. The edge convolution assembles the attributes of the connected nodes, while the node convolution aggregates the information from its adjacent edges and updates the attributes of each node. The overall architecture is based on the graph network from Battaglia et al. (2018).

Loss Function. Similarly as existing deep graph matching approaches, we train our network in a supervised way based on the ground-truth matching matrix X_j^{gt} between $\mathcal{G}_j$ and $\mathcal{U}$. To this end, we use the matching loss

$$\mathcal{L}_{\text{match}} = \frac{1}{N} \sum_{j=1}^{N} ||X_j^{\text{gt}} - X_j||_F^2. \tag{9}$$

Furthermore, similarly as in previous work (Wang et al. 2018, 2020b), we adopt a one-to-one matching prior in terms of a soft constraint. To this end, we first convert the predicted permutation matrix X_j to a binary node label matrix $Y_j \in \{0,1\}^{m_j d \times 2}$ that we define as

$$Y_j = (1 - \text{vec}(X_j), \text{vec}(X_j)) \,. \tag{10}$$

Here, $\text{vec}(X_j)$ is the vectorisation of X_j. We can compute the corresponding index vector $y_j \in \{0,1\}^{m_j d}$ defined as

$$(y_j)_i = \arg\max_{k \in \{1,2\}} (Y_j)_{ik}. \tag{11}$$

By leveraging the auxiliary matrix $B \in \{0,1\}^{(m_j+d) \times m_j d}$ and the ground-truth permutation matrix X_j^{gt} (Wang et al. 2018), the one-to-one matching regularisation is

$$\mathcal{L}_{\text{reg}} = ||B(y - \text{vec}(X_j^{\text{gt}}))||^2. \tag{12}$$

The total loss that we minimise during training is

$$\mathcal{L} = \omega_{\text{m}} \mathcal{L}_{\text{match}} + \omega_{\text{d}} \mathcal{L}_{\text{def}} + \omega_{\text{o}} \mathcal{L}_{\text{off}} + \omega_{\text{reg}} \mathcal{L}_{\text{reg}}. \tag{13}$$

Training. We train a single network that is able to handle multiple object categories at the same time. To this end, we learn separate 3D universe points for each category, and in addition we introduce a separate learnable linear operator for each category that is applied to the global feature obtained by the 2D Point Encoder. The linear operator aims to transform the global feature to a category-specific representation, and also helps in resolving ambiguities between categories with objects that are somewhat similar (e.g. cat and dog).

In practice, we apply a warm start to learn the universe points $\mathcal{U}$, which are randomly initialised for each category. After retrieving $\mathcal{U}$, we start training the neural network on the total loss with $\omega_{\text{m}} = 1$, $\omega_{\text{d}} = 0.5$, $\omega_{\text{o}} = 0.05$ and $\omega_{\text{reg}} = 0.1$ (in all our experiments). The batch size is 16 and the number of iterations after warm start is 150k. The learning rate is 0.008 and scheduled to decrease exponentially by 0.98 after each 3k iterations.

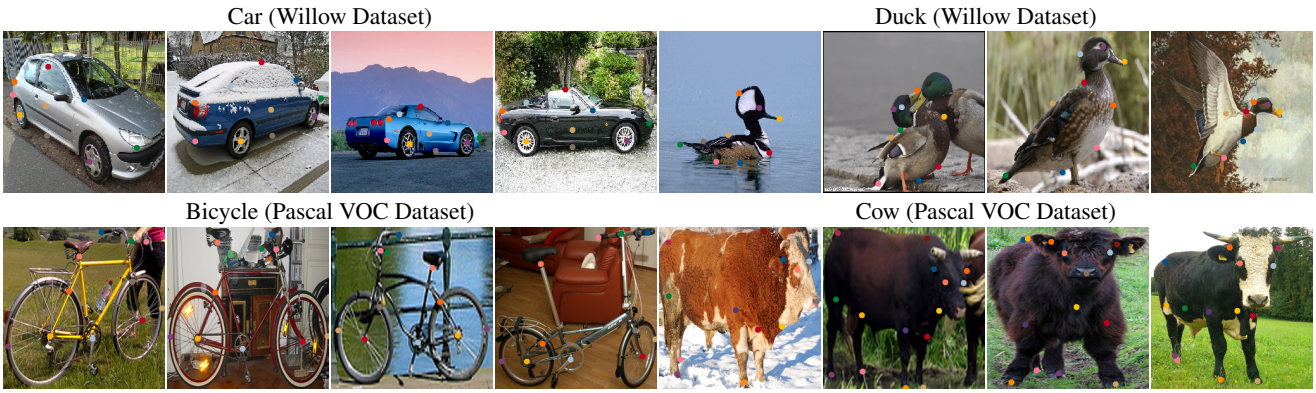


Figure 3: Qualitative results of our method on the Willow and Pascal VOC Dataset. We achieve accurate results for non-deformable objects of different types (car, bike) and reasonable results for instances of articulated objects (duck, cow).

Experiments

In the following, we evaluate our method in various settings. We compare our method to different state-of-the-art methods on two datasets, and we evaluate our deformation module based on a dataset of 3D objects.

Ablation Study. To confirm the importance of the individual components of our approach we conducted an ablation study. To this end we evaluate the accuracy on the Pascal VOC dataset in cases where we omit individual terms of the loss function, omit the warm start for learning the universe points $\mathcal{U}$, and omit deformation module, see Table 1. When we omit the one-to-one matching regulariser by setting ω_{reg} to 0, the matching accuracy is depressed substantially. When we do not conduct a warm start for finding initial universe points, the matching accuracy deteriorates. Similarly, the matching accuracy lowers without the use of our deformation module. Further, the offset regularisation and the deformed reconstruction loss can refine the universe points for each object, which brings a better matching accuracy as shown in the last two experiments. Overall, the accuracy is highest when using all components together.

Ablative setting	**Average accuracy**
$\omega_{\text{reg}} = 0$	58.11
w/o warm start	58.49
w/o deformation module	60.33
$\omega_{\text{o}} = 0$	64.19
$\omega_{\text{d}} = 0$	64.81
Ours	67.1

Table 1: Matching accuracy on the Pascal VOC dataset with the variants on regularisation terms or training strategies.

Comparisons to the state of the art. For the comparison experiments, we follow the testing protocol that was used in CSGM (Wang et al. 2020b). While all competing methods predict pairwise matchings X_{ij}, our approach predicts object-to-universe matchings X_i. Hence, we present the accuracies for pairwise matchings (written in parentheses) in addition to the accuracies for our object-to-universe matchings. Note that X_{ij} is obtained by $X_{ij} = X_i X_j^T$, which may add individual errors in X_i and X_j up, thereby leading to smaller pairwise scores. In the following, we summarise the experimental setting for each dataset and discuss our results. Parts of the matching results are visualised in Fig. 3.

Willow Dataset. We simultaneously train our model for all categories of the Willow dataset (Cho, Alahari, and Ponce 2013). It consists of images from 5 classes. It is compiled from Caltech-256 and Pascal VOC 2007 datasets, and consists of more than 40 images per class with 10 distinctively labelled features each.

We use the same training/testing split as in CSGM (Wang et al. 2020b). For training, 20 images are randomly chosen from each class and the rest are used for testing. For non-learning based methods, the affinity matrix is constructed using the SIFT descriptors (Lowe 2004) as done by Wang et al. (2018), more details are described in supplementary material. We use the 2D key point coordinates as attributes

Method	car	duck	face	motor.	bottle	**Avg.**	△	**3D**
IPFP	74.8	60.6	98.9	84.0	79.0	79.5	✗	✗
RRWM	86.3	75.5	100	94.9	94.3	90.2	✗	✗
PSM	88.0	76.8	100	96.4	97.0	91.6	✗	✗
GNCCP	86.4	77.4	100	95.6	95.7	91.0	✗	✗
ABPF	88.4	80.1	100	96.2	96.7	92.3	✗	✗
HARG	71.9	72.2	93.9	71.4	86.1	79.1	✗	✗
GMN	74.3	82.8	99.3	71.4	76.7	80.9	✗	✗
PCA	84.0	93.5	100	76.7	96.9	90.2	✗	✗
CSGM	91.2	86.2	100	99.4	97.9	94.9	✗	✗
BBGM	100.0	99.2	96.9	89.0	98.8	96.8	✗	✗
Ours	98.8	90.3	99.9	99.8	100	97.8	✓	✓
Ours	(98.7)	(86.4)	(99.9)	(99.8)	(100)	(97.0)	✓	✓

Table 2: Matching accuracy on Willow dataset, where '△' indicates whether the method guarantees the cycle consistency, and '3D' indicates that 3D geometry is obtained. Comparing to the other algorithms, our method can achieve the best average accuracy and guarantee cycle consistency.

Method	Filtering	**Avg. Acc.**	△	**3D**
GMN	y	55.3	✗	✗
PCA	y	63.8	✗	✗
CSGM	y	68.5	✗	✗
Ours	y	67.1	✓	✓
(Ours)	y	(58.9)	✓	✓
BBGM-Max	n	51.9	✗	✗
BBGM	n	61.4	✗	✗
BBGM-Multi	n	62.8	locally	✗
Ours	n	59.4	✓	✓
(Ours)	n	(42.9)	✓	✓

Table 3: Results on Pascal VOC Keypoints dataset. Note that in terms of accuracy we achieve comparable results to the previous state of the art methods GMN (Zanfir and Sminchisescu 2018), PCA (Wang, Yan, and Yang 2019a), CSGM (Wang et al. 2020b) and BBGM (Rolínek et al. 2020), while we are the only one that additionally achieves cycle consistency ('△') and reconstructs 3D geometry ('3D').

of nodes in $\mathcal{G}_i$, while the attributes of nodes in $\mathcal{U}$ are the 3D coordinates of the (learned) universe points.

Table 2 shows the accuracy of our method, on the Willow dataset, in comparison with IPFP (Leordeanu, Hebert, and Sukthankar 2009), RRWM (Cho, Lee, and Lee 2010), PSM (Egozi, Keller, and Guterman 2012), GNCCP (Liu and Qiao 2013), ABPF (Wang et al. 2018), HARG (Cho, Alahari, and Ponce 2013), GMN (Zanfir and Sminchisescu 2018), PCA (Wang, Yan, and Yang 2019a), CSGM (Wang et al. 2020b) and BBGM (Rolínek et al. 2020). Our method achieves an average accuracy of 97.8%, while also being able to reconstruct the 3D structure of objects, see Fig. 1. In the car category, our method outperforms the others noticeably. Although there is non-rigid motion in the duck category caused by articulation, our method still achieve a reasonable accuracy. Further, ours is the only one that guarantees cycle-consistent matchings.

Pascal VOC Keypoints Dataset. The Pascal VOC Keypoints dataset (Bourdev and Malik 2009) contains 20 categories of objects with labelled key point annotations. The number of key points varies from 6 to 23 for each category. Following Wang et al. (2020b), we use 7020 images for training and 1682 for testing.

We randomly sample from the training data to train our model. As shown in Table 3, in terms of matching accuracy our method is on par with the CSGM method. Moreover, the "Filtering" column denotes that keypoints missing from one of the images are filtered out before matching. This procedure is not used for our method because the universe graph contains all possible key points in one category. Nevertheless, to provide a fair comparison in the "Filtering" setting, for our method we remove elements of the (non-binary) matching matrices corresponding to keypoints that are not presented, and binarize them afterwards. Furthermore, we also report accuracies for our method without any filtering. Besides predicting accurate matchings, our method is the only one that achieves globally cycle-consistent matchings and infers 3D geometry as shown in Fig. 4. We emphasise that accuracy alone does not justifiably measure the performance of a method. Cycle consistency among the predicted matchings is also an important performance metric. More detailed results are provided in supp. mat.

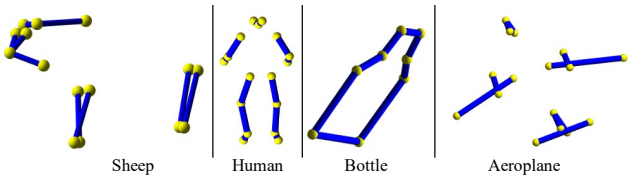


Figure 4: Illustration of 3D universe points. Examples of coarsev3D universe points from Pascal VOC dataset. Blue lines are handcrafted for better visualisation.

3D Geometry and Deformation Evaluation. The goal of this experiment is to show that the learned 3D universe points are plausible, and the deformation module can compensate for instance-specific nonlinear deformations. For this experiment, we use the 3D head dataset D3DFACs (Cosker, Krumhuber, and Hilton 2011; Li et al. 2017). We use a similar pre-processing pipeline as in i3DMM (Yenamandra et al. 2021) to obtain 8 facial landmarks on each head in the template-registered dataset. For training our model, we use 2D projections, with a pinhole camera model, of the randomly transformed 3D landmarks. During test time, we align the predicted 3D points with ground truth using Procrustes alignment to recover 3D scale and rigid transformation. The average L2 error between the ground truth 3D points and the obtained 3D universe points before and after deformations is 0.356 and 0.148, confirming the merits of the deformation module. More qualitative results are provided in supp. mat.

Conclusion

In this work we tackle the novel problem setting of simultaneously solving graph matching and performing sparse 3D reconstruction from inhomogeneous 2D image collections. Our solution achieves several favourable properties simultaneously: our matchings are cycle-consistent, which is an important property since the (unknown) ground truth matchings are cycle-consistent. Our approach does not rely on the availability of an initial 3D geometry model, so that we can train it on virtually any object category, as opposed to object-specific 3D reconstruction approaches that are for example tailored towards faces only. Instead, during training we learn a (sparse) deformable 3D geometric model directly from 2D image data. Moreover, our methods merely requires multiple images of *different object instances* of the same category. This is in contrast to typical multi-view reconstruction approaches that require multiple images of the *same object instance* from different views. We believe that the joint consideration of deep graph matching and 3D geometry inference will open up interesting research directions and that our approach may serve as inspiration for follow-up works on matching, 3D reconstruction, and shape model learning.

References

Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. 2018. Relational Inductive Biases, Deep Learning, and Graph Networks. *arXiv preprint arXiv:1806.01261*.

Bay, H.; Ess, A.; Tuytelaars, T.; and Van Gool, L. 2008. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*. Similarity Matching in Computer Vision and Multimedia.

Bentley, J. L. 1975. Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*.

Bernard, F.; Cremers, D.; and Thunberg, J. 2021. Sparse Quadratic Optimisation over the Stiefel Manifold with Application to Permutation Synchronisation. In *NeurIPS*.

Bernard, F.; Theobalt, C.; and Moeller, M. 2018. DS*: Tighter Lifting-Free Convex Relaxations for Quadratic Matching Problems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Bernard, F.; Thunberg, J.; Goncalves, J.; and Theobalt, C. 2018. Synchronisation of Partial Multi-Matchings via Non-negative Factorisations. *Pattern Recognition*.

Bernard, F.; Thunberg, J.; Swoboda, P.; and Theobalt, C. 2019. HiPPI: Higher-Order Projected Power Iterations for Scalable Multi-matching. In *Proceedings of the IEEE International Conference on Computer Vision*.

Birdal, T.; and Simsekli, U. 2019. Probabilistic Permutation Synchronization using the Riemannian Structure of the Birkhoff Polytope. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Bogo, F.; Kanazawa, A.; Lassner, C.; Gehler, P.; Romero, J.; and Black, M. J. 2016. Keep it SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image. In *European Conference on Computer Vision*.

Botsch, M.; Kobbelt, L.; Pauly, M.; Alliez, P.; and Lévy, B. 2010. *Polygon Mesh Processing*. CRC press.

Bourdev, L. D.; and Malik, J. 2009. Poselets: Body Part Detectors Trained using 3D Human Pose Annotations. *Proceedings of the IEEE International Conference on Computer Vision*.

Burkard, R.; Dell'Amico, M.; and Martello, S. 2012. *Assignment Problems: Revised Reprint*. Society for Industrial and Applied Mathematics.

Cho, M.; Alahari, K.; and Ponce, J. 2013. Learning Graphs to Match. In *Proceedings of the IEEE International Conference on Computer Vision*.

Cho, M.; Lee, J.; and Lee, K. M. 2010. Reweighted Random Walks for Graph Matching. In *European Conference on Computer Vision*.

Cosker, D.; Krumhuber, E.; and Hilton, A. 2011. A FACS Valid 3D Dynamic Action unit Database with Applications to 3D Dynamic Morphable Facial Modeling. In *Proceedings of the IEEE International Conference on Computer Vision*.

Cour, T.; Srinivasan, P.; and Shi, J. 2006. Balanced Graph Matching. *Advances in Neural Information Processing Systems*.

Dym, N.; Maron, H.; and Lipman, Y. 2017. DS++: A Flexible, Scalable and Provably Tight Relaxation for Matching Problems. *ACM Transactions on Graphics*.

Egozi, A.; Keller, Y.; and Guterman, H. 2012. A Probabilistic Approach to Spectral Graph Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Fey, M.; Lenssen, J. E.; Morris, C.; Masci, J.; and Kriege, N. M. 2020. Deep Graph Matching Consensus. In *International Conference on Learning Representations*.

Huang, Q.-X.; and Guibas, L. 2013. Consistent Shape Maps via Semidefinite Programming. In *Computer Graphics Forum*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*.

Lawler, E. L. 1963. The Quadratic Assignment Problem. *Management Science*.

Leordeanu, M.; and Hebert, M. 2005. A Spectral Technique for Correspondence Problems using Pairwise Constraints. In *IEEE International Conference on Computer Vision*.

Leordeanu, M.; Hebert, M.; and Sukthankar, R. 2009. An Integer Projected Fixed Point Method for Graph Matching and Map Inference. In *Advances in Neural Information Processing Systems*.

Li, T.; Bolkart, T.; Black, M. J.; Li, H.; and Romero, J. 2017. Learning a Model of Facial Shape and Expression from 4D Scans. *ACM Transactions on Graphics*.

Li, Y.; Gu, C.; Dullien, T.; Vinyals, O.; and Kohli, P. 2019. Graph Matching Networks for Learning the Similarity of Graph Structured Objects. In *International Conference on Machine Learning*.

Liu, Z.-Y.; and Qiao, H. 2013. GNCCP—Graduated Nonconvexity and Concavity Procedure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Loiola, E.; Abreu, N.; Boaventura-Netto, P.; Hahn, P.; and Querido, T. 2007. A Survey of the Quadratic Assignment Problem. *European Journal of Operational Research*.

Lowe, D. 2004. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*.

Ma, J.; Jiang, X.; Fan, A.; Jiang, J.; and Yan, J. 2021. Image Matching from Handcrafted to Deep Features: A Survey. *International Journal of Computer Vision*.

Maset, E.; Arrigoni, F.; and Fusiello, A. 2017. Practical and Efficient Multi-View Matching. In *Proceedings of the IEEE International Conference on Computer Vision*.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *European Conference on Computer Vision*.

Pachauri, D.; Kondor, R.; and Singh, V. 2013. Solving the Multi-way Matching Problem by Permutation Synchronization. In *Advances in Neural Information Processing Systems*.

Pardalos, P.; Rendl, F.; and Wolkowitz, H. 1994. The Quadratic Assignment Problem: A Survey and Recent Developments. Quadratic Assignment and related problem. *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science*.

Park, K.; Sinha, U.; Barron, J. T.; Bouaziz, S.; Goldman, D. B.; Seitz, S. M.; and Martin-Brualla, R. 2020. Deformable Neural Radiance Fields. *arXiv preprint arXiv:2011.12948*.

Rolínek, M.; Swoboda, P.; Zietlow, D.; Paulus, A.; Musil, V.; and Martius, G. 2020. Deep Graph Matching via Blackbox Differentiation of Combinatorial Solvers. In *European Conference on Computer Vision*.

Swoboda, P.; Rother, C.; Alhaija, H. A.; Kainmüller, D.; and Savchynskyy, B. 2017. Study of Lagrangean Decomposition and Dual Ascent Solvers for Graph Matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Tewari, A.; Zollöfer, M.; Kim, H.; Garrido, P.; Bernard, F.; Perez, P.; and Christian, T. 2017. MoFA: Model-based Deep Convolutional Face Autoencoder for Unsupervised Monocular Reconstruction. In *Proceedings of the IEEE International Conference on Computer Vision*.

Tron, R.; Zhou, X.; Esteves, C.; and Daniilidis, K. 2017. Fast Multi-Image Matching via Density-Based Clustering. In *Proceedings of the IEEE International Conference on Computer Vision*.

Wang, Q.; Zhou, X.; and Daniilidis, K. 2018. Multi-Image Semantic Matching by Mining Consistent Features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Wang, R.; Yan, J.; and Yang, X. 2019a. Learning Combinatorial Embedding Networks for Deep Graph Matching. In *Proceedings of the IEEE International Conference on Computer Vision*.

Wang, R.; Yan, J.; and Yang, X. 2019b. Neural Graph Matching Network: Learning Lawler's Quadratic Assignment Problem with Extension to Hypergraph and Multiple-graph Matching. *arXiv preprint arXiv:1911.11308*.

Wang, R.; Yan, J.; and Yang, X. 2020. Graduated Assignment for Joint Multi-Graph Matching and Clustering with Application to Unsupervised Graph Matching Network Learning. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 19908–19919. Curran Associates, Inc.

Wang, R.; Yang, N.; Stueckler, J.; and Cremers, D. 2020a. DirectShape: Photometric Alignment of Shape Priors for Visual Vehicle Pose and Shape Estimation. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

Wang, T.; Ling, H.; Lang, C.; and Feng, S. 2018. Graph Matching with Adaptive and Branching Path Following. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wang, T.; Liu, H.; Li, Y.; Jin, Y.; Hou, X.; and Ling, H. 2020b. Learning Combinatorial Solver for Graph Matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Yenamandra, T.; Tewari, A.; Bernard, F.; Seidel, H.; Elgharib, M.; Cremers, D.; and Theobalt, C. 2021. i3DMM: Deep Implicit 3D Morphable Model of Human Heads. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Zanfir, A.; and Sminchisescu, C. 2018. Deep Learning of Graph Matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Zhang, Q.; Song, X.; Shao, X.; Zhao, H.; and Shibasaki, R. 2013. Learning Graph Matching: Oriented to Category Modeling from Cluttered Scenes. In *Proceedings of the IEEE International Conference on Computer Vision*.

Zhang, Z.; and Lee, W. S. 2019. Deep Graphical Feature Learning for the Feature Matching Problem. In *Proceedings of the IEEE International Conference on Computer Vision*.

Zhou, F.; and De la Torre, F. 2016. Factorized Graph Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhou, X.; Zhu, M.; and Daniilidis, K. 2015. Multi-Image Matching via Fast Alternating Minimization. In *Proceedings of the IEEE International Conference on Computer Vision*.