Neural Puppeteer: Keypoint-Based Neural Rendering of Dynamic Shapes[*]

Simon Giebenhain[*], Urs Waldmann[*][0000−0002−1626−9253], Ole Johannsen[0000−0002−7786−8516], and Bastian Goldluecke[0000−0003−3427−4029]

University of Konstanz, Konstanz, Germany
urs.waldmann@uni-konstanz.de
[*]Authors contributed equally.

Abstract. We introduce Neural Puppeteer, an efficient neural rendering pipeline for articulated shapes. By inverse rendering, we can predict 3D keypoints from multi-view 2D silhouettes alone, without requiring texture information. Furthermore, we can easily predict 3D keypoints of the same class of shapes with one and the same trained model and generalize more easily from training with synthetic data which we demonstrate by successfully applying zero-shot synthetic to real-world experiments. We demonstrate the flexibility of our method by fitting models to synthetic videos of different animals and a human, and achieve quantitative results which outperform our baselines. Our method uses 3D keypoints in conjunction with individual local feature vectors and a global latent code to allow for an efficient representation of time-varying and articulated shapes such as humans and animals. In contrast to previous work, we do not perform reconstruction in the 3D domain, but project the 3D features into 2D cameras and perform reconstruction of 2D RGB-D images from these projected features, which is significantly faster than volumetric rendering. Our synthetic dataset will be publicly available, to further develop the evolving field of animal pose and shape reconstruction.

1 Introduction

Neural scene representations became an emerging trend in computer vision during the last couple of years. They allow to represent scenes through neural networks which operate on 3D space, allowing for tasks like novel view synthesis of static content, generalization over object and scene classes, body, hand and face modelling and relighting and material editing. For a detailed overview please refer to [53,56]. While most such methods rely on time and memory intensive volumetric rendering, [50] proposes a method of rendering with a single network evaluation per ray. In this paper we propose a single-evaluation rendering

---

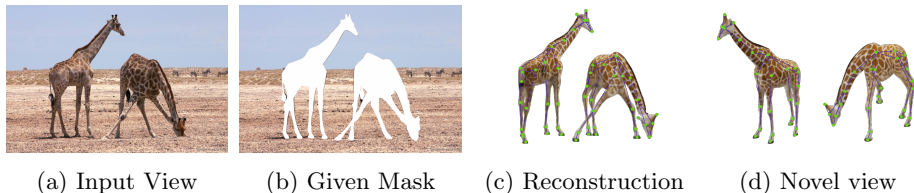(a) Input View   (b) Given Mask   (c) Reconstruction   (d) Novel view

Fig. 1: **Zero-Shot Synthetic to Real Experiment: Giraffe.** Reconstruction of a giraffe in different poses. From left to right: input image, segmentation mask used as input for keypoint estimation, estimated keypoints and shape, rendering from different perspective (cf. supplementary for more real-world examples).

formulation based on keypoints in order to represent dynamically deforming objects, like humans and animals. This allows us to render not only novel views of a known shape but also new and unseen poses by adjusting the positions of 3D keypoints. We apply this flexible approach to prevalent tasks in computer vision: the representation and reconstruction of pose for humans and animals. More specifically, we propose a silhouette based 3D keypoint detector that is based on inverse neural rendering. We decide to factor out appearance variation for keypoint detection, in order to ease the bridging of domain gaps and handle animals with few data available. Despite only relying on silhouettes, our proposed approach shows promising results when compared to the state-of-the-art 3D keypoint detector [17]. Furthermore, our approach is capable of zero-shot synthetic to real generalization, see Fig. 1.

Contributions. Our contribution is a flexible keypoint based neural scene representation and neural rendering framework called Neural Puppeteer (NePu).

We demonstrate that NePu provides valuable gradients as a differential forward map in an inverse rendering approach for 3D keypoint detection. Since we formulate the inverse rendering exclusively on 2D silhouettes, the resulting 3D keypoint detector is inherently robust with respect to transformations or domain shifts. Note that common 3D keypoint estimators require a huge amount of training samples with different texture in order to predict keypoints of the same class of shapes. Another advantage of being independent of texture is that it is easier to generalize from training with synthetic data. This can be particularly useful in cases where it is highly challenging to obtain a sufficient amount of real-world annotations, such as for wild animals (cf. Fig. 1). For animal shapes, we outperform a state-of-the-art 3D multi-view keypoint estimator in terms of Mean Per Joint Position Error (MPJPE) [17].

Unlike common practice, we shift rendering from the 3D to 2D domain, requiring only a single neural network evaluation per ray. In this sense, our approach can be interpreted as a locally conditioned version of Light Field Networks (LFNs) [50]. Our formulation is capable of learning the inter-pose variations of a single instance (constant shape and appearance) under constant lighting conditions, similar to [52]. In contrast, LFNs learn a prior over inter-object variations. We retain the rendering efficiency of LFNs and are capable of rendering color,

depth and occupancy simultaneously at 20ms per 256^2 image. This is significantly faster than NeRF-like approaches such as [52], which typically achieve less than 1 fps. Due to our fast renderer, fitting a target pose by inverse rendering can be done at $\sim$ 1fps using 8 cameras. Furthermore, we show that our keypoint-based local conditioning significantly improves the neural rendering of articulated objects, with visibly more details and quantitative improvements in PSNR and MAE over LFNs.

Code and data sets [12] to reproduce the results in the paper are publicly available at https://urs-waldmann.github.io/NePu/. We hope to inspire further work on animal pose and shape reconstruction, where our synthetic dataset can serve as a controlled environment for evaluation purposes and to experiment with novel ideas.

2   Related Work

2.1   3D Keypoint Estimation

Human keypoint estimation is a vast field with many applications [2,55,6,57]. For further reading, we refer the reader to [9,54,19,25]. The current state-of-the-art methods for 3D human keypoint estimation when trained on a single data set, i.e. the famous Human3.6M data set [16], are [45,17,14] with an average MPJPE of 18.7 mm, 20.8 mm and 26.9 mm respectively. At the time of writing, there was no code available for [45]. That is why we choose LToHP [17] as a baseline to quantitatively compare our model to. With the huge success of human keypoint estimation, the prediction of 3D keypoints for animals became a sub-branch of its own [13,34,20,3,21]. We notice that all these 3D frameworks for animals exploit 2D keypoints with standard 3D reconstruction techniques. That is why we also choose LToHP [17] as our baseline for animals, since it uses a learnable approach for the 3D reconstruction part.

Please keep in mind that our pose estimation only relies on multi-view silhouettes, while the above methods require RGB images. Using silhouettes gives more robustness to changes in texture and lighting. The only other work we are aware of that extracts keypoints from silhouettes is [5]. While [5] extracts 2D keypoints from silhouettes for quadrupeds using a template mesh and a stacked hourglass network [35], we are able to predict 3D coordinates for arbitrary shapes.

2.2   Morphable Models

The seminal morphable models for animals and humans are SMAL [65] and SMPL [26] respectively. An extension of SMPL, called SMPL-X [41], includes hands [46] and face [24]. These models have been used to estimate the 3D pose and shape from a single image [62,47,4], from multiple unconstrained images in the wild [49,64] or in an unsupervised manner from a sparse set of landmarks [29]. Because creating these models is tedious, the authors of [63] present an unsupervised disentanglement of pose and 3D mesh shape. Neural Body [43]

uses implicit neural representations where different sparse multi-view frames of a video share the same set of latent codes anchored to the vertices of the SMPL model. In this way the SMPL model provides a geometric prior for their model with which they can reconstruct 3D geometry and appearance and synthesize novel views.

While these representations are comparatively easy to handle, the need of a parametric template mesh limits them to a pre-defined class of shapes. In particular, models for quadrupeds can fit a large variety of relatively similar animals like cats and horses, but run into problems if uncommon shapes are present [65]. For example, the fitting of elephants or giraffes pose significant problems due to their additional features (trunk) or different shapes (long neck). Similar problems arise in case of humans and clothing, e.g. a free flowing coat.

On the contrary, [23,58] exploit implicit representations and reduce manual intervention completely. They disentangled dynamic objects into a signed distance field defined in canonical space and a latent pose code represented by the flow field from a canonical pose to a given shaped pose of the same identity.

2.3   Neural Fields

Neural fields are an emergent research area that has become a popular representation for neural rendering [32,60,50,33,53], 3D reconstruction and scene representation [31,38,51], geometry aware generative modelling [48,7,36] and many more. For a detailed overview, we refer the reader to [56].

While these representations often rely on a global latent vector to represent the information of interest [31,38,51,50], the importance of locally conditioning the neural field has been demonstrated in [8,44,11,36,33]. By relying on local information drawn from geometrically aligned latent vectors, these methods often obtain higher quality reconstructions and generalize better due to their translation equivariance.

Neural fields have an especially strong impact on neural rendering with the formulation of radiance-based integration over rays introduced in NeRF [32]. While NeRF and many follow-ups [60,33] achieve high-quality renderings of single scenes, [59] match their performance with relying on neural networks, implying that NeRF's core competence are meaningful gradients for optimization. [52] combines the well known NeRF pipeline [32] with a keypoint based skeleton. This allows them to reconstruct articulate 3D representation of humans by representing the pose through the 3D positions of the keypoints. However, [52] optimizes 3D coordinates in the camera system obtained from fitting the SMPL model [26] while we predict 3D world coordinates from multi-view silhouettes.

In contrast to the volumetric rendering of NeRF typically requiring hundreds of evaluations of the neural network, LFN [50] propose an alternative by instantly predicting a pixels color given the corresponding ray's origin and direction. This approach results in a much faster rendering compared to NeRF-like approaches. In this work, we embrace the idea of single-evaluation rendering [50], and employ ideas from [11] for an efficient representation and architecture. Our approach of rendering is thus much faster than [52], allowing also for faster solutions to

the inverse rendering problem of estimating keypoint locations from images. Since [50] is the only other single-evaluation-rendering method we are aware of, we quantitatively compare our method in Tab. 1 in terms of color and depth.

3 Neural Puppeteer

We will describe Neural Puppeteer in three parts. First, we discuss the encoding of the pose and latent codes, as can be seen on the upper half of Fig. 2. Second, we describe our keypoint-based neural rendering, as is depicted in the lower half of the figure. Third and last, we discuss how our pipeline can be inverted to perform keypoint estimation by fitting the pose generated from 3D keypoints to input silhouette data.

3.1 3D Keypoint Encoding

Given 3D keypoint coordinates $\mathbf{x} \in \mathbb{R}^{K \times 3}$ of a subject with K keypoints, we aim to learn an encoder network

$$\text{enc} : \mathbb{R}^{K \times 3} \to \mathbb{R}^{d_z}, \ \mathbf{x} \mapsto \mathbf{z} \tag{1}$$

that encodes a pose $\mathbf{x}$ as a d_z-dimensional global representation $\mathbf{z}$, as well as to learn a decoder network

$$\text{dec} : \mathbb{R}^{d_z} \to \mathbb{R}^{K \times 3} \times \mathbb{R}^{K \times d_f}, \ \mathbf{z} \mapsto (\hat{\mathbf{x}}, \mathbf{f}) \tag{2}$$

that reconstructs the pose $\hat{\mathbf{x}}$ and obtains local features $\mathbf{f}_k \in \mathbb{R}^{d_f}$ for each keypoint. Subsequently, we use the representation $(\mathbf{z}, \mathbf{f})$ to locally condition our neural rendering on the pose $\mathbf{x}$, as explained in section 3.2. Please note that we do not require a skeleton model, i.e. connectivity between key points, since the pose is only interpreted as a point cloud.

We build our encoder upon the vector self-attention

$$\text{VSA} : \mathbb{R}^{K \times 3} \times \mathbb{R}^{K \times d_f} \to \mathbb{R}^{K \times d_f}, \ (\mathbf{x}, \mathbf{f}) \mapsto \mathbf{f}' \tag{3}$$

introduced in [61] as a means of a neural geometric point cloud operator. Consequently, our encoder which consists of L layers produces features

$$\mathbf{f}^{(l+1)} = \text{ET}_l(\text{BN}_l(\mathbf{f}^{(l)} + \text{VSA}_l(\mathbf{x}, \mathbf{f}^{(l)}))), l \in \{0, \cdots, L-1\}, \tag{4}$$

where BN denotes a BatchNorm layer [15] and ET denotes element-wise transformations containing a 2-layer MLP, residual connection and another BatchNorm. Initial features $\mathbf{f}^{(0)} \in \mathbb{R}^{K \times d_f}$ are learned as free parameters. The final global representation $\mathbf{z}$ is obtained via dimension-wise global maxpooling,

$$\mathbf{z} = \max_{k=1,\ldots,K} \mathbf{f}_k^{(L)}. \tag{5}$$

We decode the latent vector $\mathbf{z}$ using two 3-layer MLPs. Keypoints are reconstructed using $\hat{\mathbf{x}} = \text{MLP}_{pos}(\mathbf{z})$ and additionally features $\tilde{\mathbf{f}} = \text{MLP}_{\text{feats}}(\mathbf{z})$ holding information for the subsequent rendering are extracted. Finally, these features are refined to $\mathbf{f}$ using 3 further VSA layers, which completes the decoder $\text{dec}(\mathbf{z}) = (\hat{\mathbf{x}}, \mathbf{f})$. For more architectural details we refer to the supplementary.
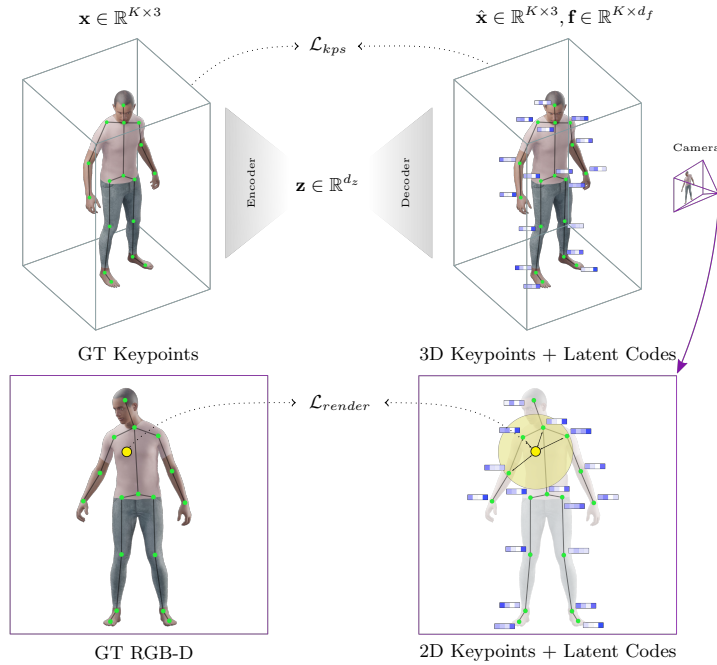
Fig. 2: **Neural Puppeteer:** Our method takes 3D keypoints (top left) and learns individual latent codes for each keypoint (top right). We project them into arbitrary camera views and perform neural rendering to reconstruct 2D RGB-D images (bottom right). The pipeline can be used to perform pose estimation by inverse rendering, optimizing the 3D keypoint positions by performing gradient descent in the latent code $\mathbf{z} \in \mathbb{R}^{d_z}$. For rendering, we use only the closest keypoints, illustated by the yellow circle around the point in question (yellow dot). Connections between keypoints are shown just for visualization and not used by the network.

3.2   Keypoint-Based Neural Rendering

Contrary to many recent neural rendering approaches, we do not rely on costly NeRF-style volumetric rendering [32]. Instead we adopt an approach similar to LFN [50], that predicts a pixel color with a single neural network evaluation.

While LFN uses a global latent vector and the ray's origin and orientation as network inputs, we directly operate in pixel coordinates. Perspective information is incorporated by projecting keypoints $\mathbf{x}$ and corresponding local features $\mathbf{f}$ into pixel coordinates. More specifically, given camera extrinsics $\mathbf{E}$ and intrinsics $\mathbf{K}$ and the resulting projection as $\pi_{\mathbf{E},\mathbf{K}}$ we obtain 2D keypoints

$$\mathbf{x}_{2D} = \pi_{\mathbf{E},\mathbf{K}}(\mathbf{x}). \tag{6}$$

Additionally, we append the depth values $\mathbf{d}$, such that the keypoint's positions are $\mathbf{x}_{2D}^* = (\mathbf{x}_{2D}, \mathbf{d}) \in \mathbb{R}^{K \times 3}$. Using this positional information we further

refine the features in pixel coordinates using $L = 3$ layers of VSA. Specifically, we define $\mathbf{f}_{2D}^{(0)} = \mathbf{f}$ and

$$\mathbf{f}_{2D}^{(l+1)} = \text{VSA}_l(\mathbf{x}_{2D}^*, \mathbf{f}_{2D}^{(l)}), l \in \{0, 1, 2\}. \tag{7}$$

The resulting refined features $\mathbf{f}_{2D}^{(L)}$ and coordinates $\mathbf{x}_{2D}^*$ are the basis for our single-evaluation rendering, as described next.

Given a pixel coordinate $\mathbf{q} \in \mathbb{R}^2$, we follow [11] to interpolate meaningful information from nearby features $\mathbf{f}_{2D}^{(L)}$ and the global representation $\mathbf{z}$. To be more specific, the relevant information is

$$\mathbf{y} = \text{VCA}((\mathbf{q}, 0), \mathbf{z}, \mathbf{x}_{2D}^*, \mathbf{f}_{2D}^{(L)}) \in \mathbb{R}^{d_f}, \tag{8}$$

where VCA denotes the vector cross-attention from [11] and we set the depth for $\mathbf{q}$ to zero.

Finally, the predictions $\hat{\mathbf{c}}$ for color, $\hat{\mathbf{d}}$ for depth and $\hat{\mathbf{o}}$ for 2D occupancy values are predicted using three feed-forward network heads

$$\text{FFN}_{\text{col}} : \mathbb{R}^{d_f} \to [0, 1]^3, \text{FFN}_{\text{dep}} : \mathbb{R}^{d_f} \to [0, 1], \text{ and } \text{FFN}_{\text{occ}} : \mathbb{R}^{d_f} \to [0, 1], \tag{9}$$

respectively, using the same architecture as in [44]. For convenience we define the color rendering function

$$\mathcal{C}_{\mathbf{E}, \mathbf{K}} : \mathbb{R}^{K \times 3} \times \mathbb{R}^{K \times d_f} \times \mathbb{R}^{d_z} \to [0, 1]^{H \times W \times 3}, \tag{10}$$

which renders an image seen with extrinsics $\mathbf{E}$ and intrinsics $\mathbf{K}$ conditioned on keypoints $\mathbf{x}$, encoded features $\mathbf{f}$ and global representation $\mathbf{z}$, by executing FFN_{col} for all pixels $\mathbf{q}$. For the depth and silhouette modalities we similarly define $\mathcal{D}_{\mathbf{E}, \mathbf{K}}$ and $\mathcal{S}_{\mathbf{E}, \mathbf{K}}$, respectively. Note, that our silhouettes contain probabilities for a pixel lying on the object.

3.3   Training

We consider a dataset consisting of M poses $\mathbf{x}_m \in \mathbb{R}^{K \times 3}$ captured by C cameras with extrinsics $\mathbf{E}_c$ and intrinsics $\mathbf{K}_c$. For each view c and pose m we have 2D observations

$$\mathbf{I}_{m,c}, \mathbf{D}_{m,c}, \mathbf{S}_{m,c}, m \in \{1, \ldots, M\}, c \in \{1, \ldots, C\}, \tag{11}$$

corresponding to color, depth and silhouette, respectively.

All model parameters are trained jointly to minimize the composite loss

$$\mathcal{L} = \lambda_{\text{pos}}\mathcal{L}_{\text{pos}} + \lambda_{\text{col}}\mathcal{L}_{\text{col}} + \lambda_{\text{dep}}\mathcal{L}_{\text{dep}} + \lambda_{\text{sil}}\mathcal{L}_{\text{sil}} + \lambda_{\text{reg}}\|\mathbf{z}\|_2, \tag{12}$$

where the different positive numbers λ are hyperparameters to balance the influence of the different losses. The keypoint reconstruction loss

$$\mathcal{L}_{\text{pos}} = \sum_{m=1}^{M} \|\mathbf{x}_m - \hat{\mathbf{x}}_m\|_2 \tag{13}$$

minimizes the mean Euclidean distance between the ground truth and reconstructed keypoint positions. The color rendering loss

$$\mathcal{L}_{\mathrm{col}} = \sum_{m=1}^{M} \sum_{c=1}^{C} \|\mathcal{C}_{\mathbf{E}_c, \mathbf{K}_c}(\mathbf{x}_m, \mathbf{f}_m, \mathbf{z}_m) - \mathbf{I}_{m,c}\|_2^2 \qquad (14)$$

is the squared pixel-wise difference over all color channels, the depth loss $\mathcal{L}_{\mathrm{dep}}$ is given by a structurally identical formula. Finally, the silhouette loss

$$\mathcal{L}_{\mathrm{sil}} = \sum_{m=1}^{M} \sum_{c=1}^{C} \mathrm{BCE}\left(\mathcal{S}_{\mathbf{E}_c, \mathbf{K}_c}(\mathbf{x}_m, \mathbf{f}_m, \mathbf{z}_m), \mathbf{S}_{m,c}\right) \qquad (15)$$

measures the binary cross entropy $\mathrm{BCE}(\hat{o}, o) = -[o \cdot \log(\hat{o}) + (1-o) \cdot \log(1-\hat{o})]$ over all pixels. Hence the silhouette renderer is trained to classify pixels into inside and outside points, similar to [31].

3.4   Pose Reconstruction and Tracking

While the proposed model learns a prior over poses along with their appearance and geometry and thus can be used to render from 3D keypoints, we can also infer 3D keypoints by solving an inverse problem, using NePu as a differentiable forward map from keypoints to images. We are especially interested in silhouette-based inverse rendering, in order to obtain robustness against transformations that leave silhouettes unchanged.

Given observed silhouettes $\mathbf{S}_c$, extrinsics $\mathbf{E}_c$ and intrinsics $\mathbf{K}_c$ for cameras $c \in \{1, \ldots, C\}$, we optimize for

$$\hat{\mathbf{z}} = \operatorname*{argmin}_{\mathbf{z}} \sum_{c=1,\ldots,C} \mathrm{BCE}\left(\mathcal{S}_{\mathbf{E}_c, \mathbf{K}_c}(\mathrm{dec}(\mathbf{z}), \mathbf{z}), \mathbf{S}_c\right) + \lambda \|\mathbf{z}\|_2. \qquad (16)$$

Following [41], we use the PyTorch [39] implementation of the limited-memory BFGS (L-BFGS) [37] to solve the optimization problem. Once $\hat{\mathbf{z}}$ has been obtained, we recover the corresponding pose as $\hat{\mathbf{x}} = \mathrm{MLP}_{\mathrm{pos}}(\hat{\mathbf{z}})$.

Since the above mentioned optimization problem does not assume any prior knowledge about the pose, the choice of an initial value is critical. For initial values too far from the ground truth pose, we observe an unstable optimization behavior that frequently diverges or gets stuck in local minima. To overcome this obstacle, we run the optimization with I different starting conditions $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(I)}$, which we obtain by clustering the 2-dimensional t-SNE [28] embedding of the global latent vectors over the training set using affinity propagation [10]. We obtain different solutions $\hat{\mathbf{z}}_i$ from minimization of Eq. (16), and as the final optimum choose the one with best IoU to the target silhouettes,

$$\hat{\mathbf{z}} = \operatorname*{argmax}_{\hat{\mathbf{z}}_i} \sum_{c=1,\ldots,C} \mathrm{IoU}\left(\mathcal{S}_{\mathbf{E}_c, \mathbf{K}_c}(\mathrm{dec}(\hat{\mathbf{z}}_i), \hat{\mathbf{z}}_i), \mathbf{S}_c\right). \qquad (17)$$

While such a procedure carries a significant overhead, the workload is amortized in a tracking scenario. When provided with a sequence of silhouettes

Table 1: *Quantitative results of the rendering on the test sets.* Comparison of the color PSNR [db] between the reconstructed RGB images and mean absolut error (MAE) [mm] for the reconstructed depth by LFN* and NePu. The local conditioning significantly improves reconstruction accuracy. See text for a discussion of the results.

	Color PSNR [dB]			Depth MAE [mm]		
	LFN*	NePu	Δ	LFN*	NePu	Δ
Sy. Cow	14.95	**19.17**	+4.22	43.4	**22.3**	−21.1
Sy. Giraffe	15.99	**21.23**	+5.24	92.1	**35.4**	−56.7
Sy. Pigeon	21.47	**28.03**	+6.56	6.9	**2.5**	−4.4
Sy. Human	20.23	**27.49**	+7.26	77.8	**20.9**	−56.9
Average	18.16	**23.98**	+5.82	55.1	**20.3**	−34.8

$\mathbf{S}_{c,1}, \ldots, \mathbf{S}_{c,T}, c \in \{1, \ldots, C\}$ of T frames, we use the above method to determine $\hat{\mathbf{z}}_1$. For subsequent frames we initialize $\hat{\mathbf{z}}_{t+1} = \hat{\mathbf{z}}_t$ and fine-tune by minimizing Eq. (16) using a few steps of gradient descent. Our unoptimized implementation of the tracking approach runs at roughly 1 second per frame using 8 cameras.

4   Experiments

In this section, we evaluate different aspects of our method. To evaluate pose estimation on previously unseen data, we compare our method to the state-of-the-art multi-view keypoint detector [17], which we train for each individual subjects using the same dataset as for NePu.

The fundamental claims of our methodology are shown by rendering novel views and poses, quantitatively evaluating color and depth estimates and comparing against a version of our framework that utilizes the LFN [50] formulation for neural rendering. For visual evidence that our method produces temporally consistent views and additional qualitative results we refer to videos and experiments in our supplemental material. In addition we compare NePu to AniNeRF [42] on Human3.6M [16].

4.1   Datasets

Our method can be trained for any kind of shape data that can be described by keypoints or skeletons. Connectivity between the keypoints neither needs to be known, nor do we make use of it in any form. We perform a thorough evaluation of our method on multiple datasets, for two types of shapes where a keypoint description often plays a major role: humans and animals.

For the human data we use the SMPL-X Blender add-on [41]. Here, we evaluate both on individual poses and captured motion sequences. The poses were obtained through the AGORA dataset [40], the animations through the
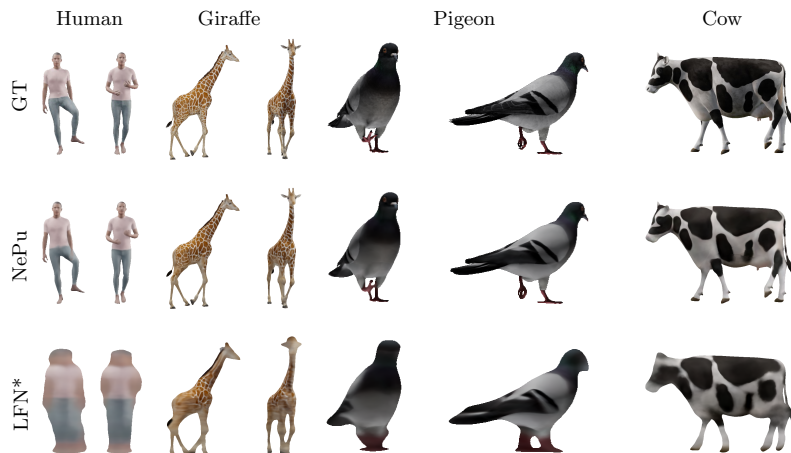
Fig. 3: **Novel view synthesis for novel poses.** Our method is capable of generating realistic renderings of the captured subject. In this figure, we show novel poses from new perspectives. See text for a discussion of the results.

AMASS dataset, and originally captured by ACCAD and BMLmovi [30,1,22]. We describe the human pose by 33 keypoints, see additional material for details.

For the training of animal data we use hand-crafted animated 3D models from different sources. To capture a variety of different animal shapes, we render datasets with a cow, a giraffe, and a pigeon. In particular the giraffe is a challenging animal for template mesh based methods, as the neck is much longer compared to most other quadrupeds. For each animal we created animations which include idle movement (e.g. looking around), walking, trotting, running, cleaning, and eating and render between 910 and 1900 timesteps. We use between 19 and 26 keypoints to describe the poses.

The keypoints for all shapes were created by tracking individual vertices. As keypoints in the interior of the shape are typically preferred, we average the position of two vertices on opposing sides of the body part. All datasets were rendered using Blender (www.blender.org) and include multi-view data of 24 cameras placed on three rings at different heights around the object. For each view and time step we generate ground truth RGB-D data as well as silhouettes of the main object. The camera parameters, and 3D and 2D keypoints for each view and timestep are also included.

4.2   Implementation Details

We implement all our models in PyTorch [39] and train using the AdamW optimizer [27] with a weight decay of 0.005 and a batch size of 64 for a total of 2000 epochs. We set the initial learning rate to $5e^{-4}$, which is decayed with a factor of 0.2 every 500 epochs. We weight the training loss in Eq. (12) with

Fig. 4: **3D point cloud reconstruction.** The 2D depth estimations from our method can be used to perform 3D reconstruction of the captured shape. The raw 3D point clouds generated by projecting the estimates from all cameras into the common reference frame already yield a good representation. The outliers originate from depth estimates at occlusion boundaries as the L_2 loss encourages smoothed depth maps and could be easily removed by filtering the point cloud.

$\lambda_{pos} = 2$, $\lambda_{col} = \lambda_{dep} = 1$, $\lambda_{sil} = 3$ and $\lambda_{reg} = 1/16$. Other hyperparameters and architectural details are presented in the supplementary.

Instead of rendering complete images during training to compute $\mathcal{L}_{col}$, $\mathcal{L}_{dep}$ and $\mathcal{L}_{sil}$, we only render a randomly sampled subset of pixels. Both for color and depth we sample uniformly from all pixels in the ground truth mask. Hence the color and depth rendering is unconstrained outside of the silhouette. To compute $\mathcal{L}_{sil}$ we sample areas near the boundary of the silhouette more thoroughly, similar to [8].

4.3   Baselines

We use different baselines: LToHP [17] for our pose estimation and tracking approach and [50] for our keypoint-based locally conditioned rendering results. In addition we compare NePu to AniNeRF [42] on Human3.6M [16].

LToHP. LToHP [17] presents two solutions for multi-view 3D human pose estimation; an algebraic and volumetric one. For details see our supplementary and [17]. We use their implementation from [18] with their provided configuration file for hyperparameters. In order to obtain the quantitative results in Tab. 2, we individually fine-tune [17] on every animal, using the same data that we trained NePu on. In animal pose estimation it is common practice to fine-tune a network that was pretrained, as in state of the art animal pose estimators like DeepLabCut [34], due to a lack of animal pose data. For all models, including our models, we select the epoch with minimum validation error for test.

LFNs. For the comparison to [50], we integrate their rendering formulation in our framework, resulting in two differences. First, their rendering operates in Plücker coordinates instead of pixel values. Second, and more importantly, we do not use local features **f** for conditioning in this baseline, but global conditioning via concatenation using **z**. In the following, we denote this model by LFN*.

AniNeRF. We trained NePu using the same training regime as AniNeRF [42]: We only trained on a single subject, using every $5th$ of the first 1300 frames of sequence "Posing" of subject "S9" for training and every $5th$ of the

Table 2: *Quantitative results for 3D keypoint estimation on the test sets.* Comparison to LToHP [17]. Values are given in MPJPE [mm] and its median of all samples [mm]. Delta shows difference to [17]. alg., vol. and vol.^{gt} indicate that [17] is trained with its algebraic model, volumetric model with root joint from alg. results and from ground truth respectively.

	MPJPE [mm]			Median [mm]		
	LToHP [17]	NePu	Δ	LToHP [17]	NePu	Δ
Sy. Cow	124 (vol.)	**15**	−109	110 (vol.)	**9**	−101
Sy. Giraffe	190 (vol.^{gt})	**67**	−123	154 (vol.^{gt})	**31**	−123
Sy. Pigeon	10.3 (alg.)	**1.3**	−9.0	6.2 (alg.)	**1.1**	−5.1
Sy. Human	**28** (alg.)	46	+18	**22** (alg.)	28	+6
Average	88	**32**	−56	73	**17**	−56

following 665 frames for testing. Like AniNeRF, we also only used cameras $0-2$ for training and camera 3 for testing.

4.4   3D Keypoint and Pose Estimation

Quantitative results for the 3D keypoint estimation are shown in Table 2. We report the MPJPE in mm and its median [mm] over all test samples. For LToHP [17] we evaluate both the algebraic and volumetric model and report the better result. In addition we report the average of MPJPE and median over all objects. We achieve a better average MPJPE and median (32mm and 17mm respectively) over all objects than LToHP (88mm and 73mm respectively). Note, however, that [17] achieves better results for humans only. We hypothesize two reasons for that. First, the human-specific pre-training of LToHP transfers well to the human data we evaluate on. Secondly, the extremities of the human body (especially arms and hands) vanish more often in silhouettes than for the animals we worked with. Example qualitative results can be found in Fig. 5.

4.5   Keypoint-Based Neural Rendering

Quantitative results for the keypoint-based neural rendering part are shown in Table 1. For color comparison we report the PSNR [dB] over all test samples, while for depth comparison we report the MAE [mm]. We achieve a better average PSNR and MAE (23.98dB and 20.3mm respectively) over all objects than LFN* (18.16dB and 55.1mm respectively).

   Qualitative results for color rendering and depth reconstruction can be found in Fig. 3 and Sec. 3.2 of our supplementary respectively. Comparing our method to the implementation without local conditioning shows the importance of the local conditioning, the renderings are much more detailed, with more precise boundaries. Fig. 4 shows projections of multiple depth maps from different viewing directions as 3D point clouds. The individual views align up nicely, yielding
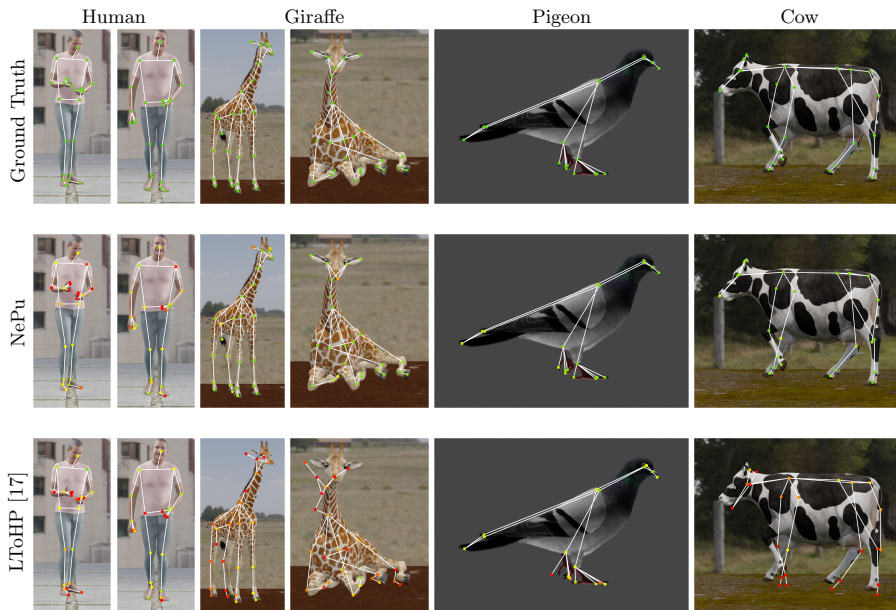
Fig. 5: **Results for pose estimation.** We show projected keypoints of the ground truth, NePu and LToHP [17] in the first, second and third row, respectively, for a human, a giraffe, a pigeon and a cow. Color coded dots indicate the keypoint location and its 3D error to the ground truth. Red dots indicate a high, green dots a low 3D error. The connections between keypoints are just for visualization and not used by the networks.

good input for further processing and analysis. The results of the novel view and pose comparison with AniNeRF [42] on Human3.6M [16] are shown in Fig. 6. Even though our rendering formulation is fundamentally different and much less developed, our results look promising, but cannot meet the quality of AniNeRF. While AniNeRF leverages the SMPL parameters to restrict the problem to computing blend weight fields, our method has to solve a more complex problem. In principle our rendering could also be formulated in canonical space and leverage SMPL to model deformations. In addition, part segmentation maps, as well as, the relation of view direction and body orientation could further help to reduce ambiguities in the 2D rendering.

Compared to A-NeRF [52] and AniNeRF [42] the fundamental differences in the rendering pipeline result in a significant speed increase. Both render at $0.25-1fps$ and $1fps$ at 512^2px, respectively, while we render at $50fps$ at 256^2px. In contrast to both methods we do not make use of optimization techniques that constrain the rendering to a bounding box of the 3D keypoints. Employing this would result in a total speed increase of $50-200\times$ at 512^2px.
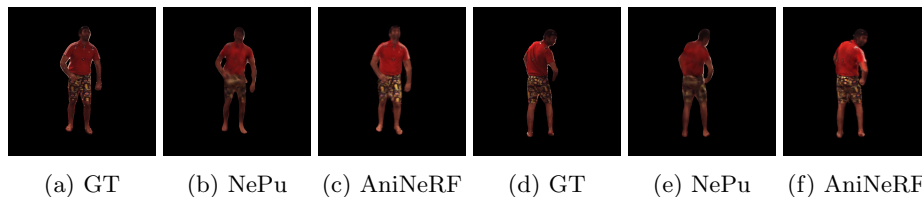
(a) GT   (b) NePu   (c) AniNeRF   (d) GT   (e) NePu   (f) AniNeRF

Fig. 6: **Novel view and pose comparison with AniNeRF.** Novel view (a-c) and novel view + novel pose (d-f) rendering on Human3.6M dataset.

5   Limitations and Future Work

In the future we plan to extend our model to account for instance specific shape and color variations, by incorporating the respective information in additional latent spaces. Moreover, our 3D keypoint encoder architecture (cf. Sec. 3.1) can for example be further improved for humans in a similar fashion to [52]. Such an approach would intrinsically be rotation equivariant and better capture the piece-wise rigidity of skeletons. Finally, while 2D rendering is much faster, it also means that we do not have guaranteed consistency between the generated views of the scene, in the sense that they are not necessarily exact renderings of the same 3D shape. This is an inherent limitation and cannot easily be circumvented, but our quantitative results indicate that it does not seem to impact quality of the rendering by much.

6   Conclusions

In this paper we present a neural rendering framework called Neural Puppeteer that projects keypoints and features into a 2D view for local conditioning. We demonstrate that NePu can detect 3D keypoints with an inverse rendering approach that takes only 2D silhouettes as input. In contrast to common 3D keypoint estimators, this is by design robust with respect to change in texture, lighting or domain shifts (e.g. synthetic vs. real-world data), provided that silhouettes can be detected. Due to our single-evaluation neural rendering, inverse rendering for downstream tasks becomes feasible. For animal shapes, we outperform a state-of-the-art 3D multi-view keypoint estimator in terms of MPJPE [17], despite only relying on silhouttes.

In addition, we render color, depth and occupancy simultaneously at 20ms per 256^2 image, significantly faster than NeRF-like approaches, which typically achieve less than 1 fps. The proposed keypoint-based local conditioning significantly improves neural rendering of articulated objects quantitatively and qualitatively, compared to a globally conditioned baseline.

7   Additional Acknowledgements

References

1. Advanced Computing Center for the Arts and Design: ACCAD MoCap Dataset, https://accad.osu.edu/research/motion-lab/mocap-system-and-data
2. Artacho, B., Savakis, A.: Unipose+: A unified framework for 2d and 3d human pose estimation in images and videos. IEEE TPAMI pp. 1–1 (2021)
3. Bala, P.C., Eisenreich, B.R., Yoo, S.B.M., Hayden, B.Y., Park, H.S., Zimmermann, J.: Automated markerless pose estimation in freely moving macaques with openmonkeystudio. Nat. Commun. **11**, 4560 (2020)
4. Biggs, B., Boyne, O., Charles, J., Fitzgibbon, A., Cipolla, R.: Who left the dogs out? 3d animal reconstruction with expectation maximization in the loop. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) Computer Vision – ECCV 2020. pp. 195–211. Springer International Publishing, Cham (2020)
5. Biggs, B., Roddick, T., Fitzgibbon, A., Cipolla, R.: Creatures great and smal: Recovering the shape and motion of animals from video. In: ACCV. pp. 3–19 (2019)
6. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: CVPR (2017)
7. Chan, E., Monteiro, M., Kellnhofer, P., Wu, J., Wetzstein, G.: pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In: arXiv (2020)
8. Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit functions in feature space for 3d shape reconstruction and completion. In: CVPR (2020)
9. Cormier, M., Clepe, A., Specker, A., Beyerer, J.: Where are we with human pose estimation in real-world surveillance? In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops. pp. 591–601 (2022)
10. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science **315**, 2007 (2007)
11. Giebenhain, S., Goldlücke, B.: Air-nets: An attention-based framework for locally conditioned implicit representations. In: 2021 International Conference on 3D Vision (3DV). pp. 1054–1064 (2021)
12. Giebenhain, S., Waldmann, U., Johannsen, O., Goldlücke, B.: Neural puppeteer: Keypoint-based neural rendering of dynamic shapes (datset) (October 2022). https://doi.org/10.5281/zenodo.7149178, https://doi.org/10.5281/zenodo.7149178
13. Günel, S., Rhodin, H., Morales, D., Campagnolo, J., Ramdya, P., Fua, P.: Deepfly3d, a deep learning-based approach for 3d limb and appendage tracking in tethered, adult *Drosophila*. eLife **8**, e48571 (2019)

14. He, Y., Yan, R., Fragkiadaki, K., Yu, S.I.: Epipolar transformers. In: CVPR (2020)
15. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd International Conference on Machine Learning (ICML). Proceedings of Machine Learning Research, vol. 37 (2015)
16. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. IEEE TPAMI **36**(7), 1325–1339 (2014)
17. Iskakov, K., Burkov, E., Lempitsky, V., Malkov, Y.: Learnable triangulation of human pose. In: ICCV (2019)
18. Iskakov, K., Burkov, E., Lempitsky, V., Malkov, Y.: Learnable triangulation of human pose (2019), https://github.com/karfly/learnable-triangulation-pytorch
19. Ji, X., Fang, Q., Dong, J., Shuai, Q., Jiang, W., Zhou, X.: A survey on monocular 3d human pose estimation. Virtual Reality & Intelligent Hardware **2**(6), 471–500 (2020)
20. Joska, D., Clark, L., Muramatsu, N., Jericevich, R., Nicolls, F., Mathis, A., Mathis, M.W., Patel, A.: Acinoset: A 3d pose estimation dataset and baseline models for cheetahs in the wild. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). pp. 13901–13908 (2021). https://doi.org/10.1109/ICRA48506.2021.9561338
21. Karashchuk, P., Rupp, K.L., Dickinson, E.S., Walling-Bell, S., Sanders, E., Azim, E., Brunton, B.W., Tuthill, J.C.: Anipose: A toolkit for robust markerless 3d pose estimation. Cell Reports **36**(13), 109730 (2021)
22. Lab, B.M.: BMLmovi Motion Capture Database, https://www.biomotionlab.ca//
23. Lei, J., Daniilidis, K.: Cadex: Learning canonical deformation coordinate space for dynamic surface representation via neural homeomorphism. In: CVPR. pp. 6624–6634 (June 2022)
24. Li, T., Bolkart, T., Black, M.J., Li, H., Romero, J.: Learning a model of facial shape and expression from 4D scans. ACM Transactions on Graphics **36**(6), 194:1–194:17 (Nov 2017), two first authors contributed equally
25. Liu, Z., Zhu, J., Bu, J., Chen, C.: A survey of human pose estimation: The body parts parsing based methods. Journal of Visual Communication and Image Representation **32**, 10–19 (2015)
26. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: Smpl: A skinned multi-person linear model. ACM Trans. Graph. **34**(6) (oct 2015), https://doi.org/10.1145/2816795.2818013
27. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
28. van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of Machine Learning Research **9**(86) (2008), http://jmlr.org/papers/v9/vandermaaten08a.html
29. Madadi, M., Bertiche, H., Escalera, S.: Deep unsupervised 3d human body reconstruction from a sparse set of landmarks. International Journal of Computer Vision **129**(8), 2499–2512 (2021)
30. Mahmood, N., Ghorbani, N., Troje, N.F., Pons-Moll, G., Black, M.J.: AMASS: Archive of motion capture as surface shapes. In: ICCV. pp. 5441–5450 (Oct 2019). https://doi.org/10.1109/ICCV.2019.00554
31. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: CVPR (2019)

32. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European Conference on Computer Vision (ECCV) (2020)
33. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. **41**(4), 102:1–102:15 (Jul 2022). https://doi.org/10.1145/3528223.3530127, https://doi.org/10.1145/3528223.3530127
34. Nath, T., Mathis, A., Chen, A.C., Patel, A., Bethge, M., Mathis, M.W.: Using deeplabcut for 3d markerless pose estimation across species and behaviors. Nat. Protoc. **14**, 2152–2176 (2019)
35. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: ECCV. pp. 483–499 (2016)
36. Niemeyer, M., Geiger, A.: Giraffe: Representing scenes as compositional generative neural feature fields. In: CVPR (2021)
37. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, New York, NY, USA, 2e edn. (2006)
38. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: CVPR (2019)
39. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019)
40. Patel, P., Huang, C.H.P., Tesch, J., Hoffmann, D.T., Tripathi, S., Black, M.J.: AGORA: Avatars in geography optimized for regression analysis. In: CVPR (Jun 2021)
41. Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A.A.A., Tzionas, D., Black, M.J.: Expressive body capture: 3d hands, face, and body from a single image. In: CVPR (2019)
42. Peng, S., Dong, J., Wang, Q., Zhang, S., Shuai, Q., Zhou, X., Bao, H.: Animatable neural radiance fields for modeling dynamic human bodies. In: ICCV. pp. 14314–14323 (October 2021)
43. Peng, S., Zhang, Y., Xu, Y., Wang, Q., Shuai, Q., Bao, H., Zhou, X.: Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In: CVPR (2021)
44. Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: ECCV (2020)
45. Reddy, N.D., Guigues, L., Pishchulin, L., Eledath, J., Narasimhan, S.G.: Tessetrack: End-to-end learnable multi-person articulated 3d pose tracking. In: CVPR. pp. 15190–15200 (2021)
46. Romero, J., Tzionas, D., Black, M.J.: Embodied hands: Modeling and capturing hands and bodies together. ACM Transactions on Graphics, (Proc. SIGGRAPH Asia) **36**(6), 245:1–245:17 (Nov 2017). https://doi.org/10.1145/3130800.3130883, http://doi.acm.org/10.1145/3130800.3130883
47. Rüegg, N., Zuffi, S., Schindler, K., Black, M.J.: Barc: Learning to regress 3d dog shape from images by exploiting breed information. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3876–3884 (June 2022)
48. Schwarz, K., Liao, Y., Niemeyer, M., Geiger, A.: Graf: Generative radiance fields for 3d-aware image synthesis. In: Advances in Neural Information Processing Systems (NeurIPS) (2020)

49. Sengupta, A., Budvytis, I., Cipolla, R.: Probabilistic 3d human shape and pose estimation from multiple unconstrained images in the wild. In: CVPR. pp. 16094–16104 (June 2021)
50. Sitzmann, V., Rezchikov, S., Freeman, W.T., Tenenbaum, J.B., Durand, F.: Light field networks: Neural scene representations with single-evaluation rendering. In: Advances in Neural Information Processing Systems (2021), https://openreview.net/forum?id=q0h6av9Vi8
51. Sitzmann, V., Zollhoefer, M., Wetzstein, G.: Scene representation networks: Continuous 3d-structure-aware neural scene representations. In: Advances in Neural Information Processing Systems (NeurIPS). vol. 32. Curran Associates, Inc. (2019), https://proceedings.neurips.cc/paper/2019/file/b5dc4e5d9b495d0196f61d45b26ef33e-Paper.pdf
52. Su, S.Y., Yu, F., Zollhöfer, M., Rhodin, H.: A-nerf: Articulated neural radiance fields for learning human shape, appearance, and pose. In: NeurIPS (2021)
53. Tewari, A., Thies, J., Mildenhall, B., Srinivasan, P., Tretschk, E., Wang, Y., Lassner, C., Sitzmann, V., Martin-Brualla, R., Lombardi, S., et al.: Advances in neural rendering. arXiv preprint arXiv:2111.05849 (2021)
54. Toshpulatov, M., Lee, W., Lee, S., Roudsari, A.H.: Human pose, hand and mesh estimation using deep learning: a survey. The Journal of Supercomputing (2022)
55. Tu, H., Wang, C., Zeng, W.: Voxelpose: Towards multi-camera 3d human pose estimation in wild environment. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) ECCV. pp. 197–212 (2020)
56. Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., Sridhar, S.: Neural fields in visual computing and beyond. https://arxiv.org/abs/2111.11426 (2021), https://arxiv.org/abs/2111.11426
57. Yang, S., Quan, Z., Nie, M., Yang, W.: Transpose: Keypoint localization via transformer. In: ICCV. pp. 11802–11812 (2021)
58. Yenamandra, T., Tewari, A., Bernard, F., Seidel, H.P., Elgharib, M., Cremers, D., Theobalt, C.: i3dmm: Deep implicit 3d morphable model of human heads. In: CVPR. pp. 12803–12813 (2021)
59. Yu, A., Fridovich-Keil, S., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks (2021)
60. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelNeRF: Neural radiance fields from one or few images. In: CVPR (2021)
61. Zhao, H., Jiang, L., Jia, J., Torr, P., Koltun, V.: Point transformer. In: International Conference on Computer Vision (ICCV) (2021)
62. Zheng, Z., Yu, T., Wei, Y., Dai, Q., Liu, Y.: Deephuman: 3d human reconstruction from a single image. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019)
63. Zhou, K., Bhatnagar, B.L., Pons-Moll, G.: Unsupervised shape and pose disentanglement for 3d meshes. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) Computer Vision – ECCV 2020. pp. 341–357. Springer International Publishing, Cham (2020)
64. Zuffi, S., Kanazawa, A., Black, M.J.: Lions and tigers and bears: Capturing non-rigid, 3d, articulated shape from images. In: CVPR (2018)
65. Zuffi, S., Kanazawa, A., Jacobs, D.W., Black, M.J.: 3d menagerie: Modeling the 3d shape and pose of animals. In: CVPR (2017)