Occupancy Flow: 4D Reconstruction by Learning Particle Dynamics

Michael Niemeyer[1] Lars Mescheder[1] Michael Oechsle[1,2] Andreas Geiger[1]
[1]Autonomous Vision Group, MPI for Intelligent Systems and University of Tübingen
[2]ETAS GmbH, Bosch Group, Stuttgart
{firstname.lastname}@tue.mpg.de

Abstract

Deep learning based 3D reconstruction techniques have recently achieved impressive results. However, while state-of-the-art methods are able to output complex 3D geometry, it is not clear how to extend these results to time-varying topologies. Approaches treating each time step individually lack continuity and exhibit slow inference, while traditional 4D reconstruction methods often utilize a template model or discretize the 4D space at fixed resolution. In this work, we present Occupancy Flow, a novel spatio-temporal representation of time-varying 3D geometry with implicit correspondences. Towards this goal, we learn a temporally and spatially continuous vector field which assigns a motion vector to every point in space and time. In order to perform dense 4D reconstruction from images or sparse point clouds, we combine our method with a continuous 3D representation. Implicitly, our model yields correspondences over time, thus enabling fast inference while providing a sound physical description of the temporal dynamics. We show that our method can be used for interpolation and reconstruction tasks, and demonstrate the accuracy of the learned correspondences. We believe that Occupancy Flow is a promising new 4D representation which will be useful for a variety of spatio-temporal reconstruction tasks.

1. Introduction

We live in a 4D world full of 3D objects in motion. An accurate and efficient representation of time-varying 3D geometry is therefore essential for us as well as for robots which navigate the very same environments. However, current 4D reconstruction approaches often require complicated multi-view setups [33,41,42,44,45,58], utilize a template model of fixed topology [2, 5, 15, 27, 30, 63, 75], or require spatio-temporally smooth inputs [48, 70], limiting the scope of possible applications to very specific tasks.

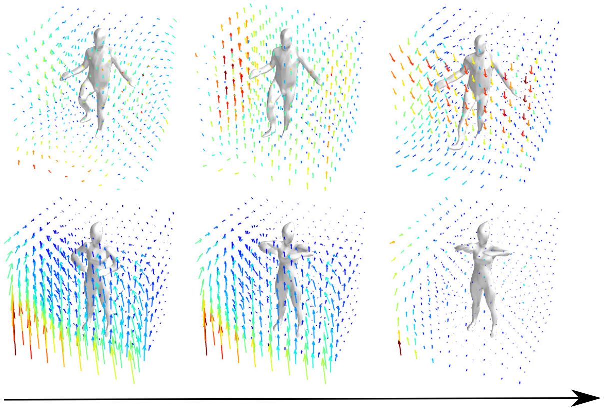


Figure 1: **Occupancy Flow.** We represent time-varying 3D geometry by a temporally and spatially *continuous* vector field which assigns a motion vector to every point in space and time, thus implicitly capturing correspondences. We demonstrate that our representation can be used for 4D reconstruction from point cloud and image sequences as well as interpolation, shape matching, and generative tasks.

Recently, learning-based approaches for recovering the 3D geometry from various forms of input have shown promising results [13,14,20,25,34,38,46,54,71]. In contrast to traditional methods, they leverage prior knowledge obtained during the training process to resolve ambiguities. In particular, recent continuous representations [13, 22, 28, 38, 39, 46, 56, 72] achieve impressive results at limited memory costs. However, it remains unclear how to extend these approaches to the task of 4D reconstruction, i.e., reconstructing 3D shapes over time. Naïvely discretizing the temporal domain would lead to high memory cost and slow inference. Furthermore, it would neither provide implicit correspondences nor a physical description of the temporal evolution. While not only being unsatisfactory from a scientific viewpoint, these problems also limit the use of existing 4D reconstruction techniques in applications where fast inference and reliable correspondences are desirable.

Contribution: In this paper, we propose a novel continuous 4D representation (Fig. 1) which implicitly models correspondences. More specifically, we parameterize a vector field with a neural network which assigns a 3D vector of motion to every 4D point in space and time. We combine this model with *Occupancy Networks* (ONet) [38] which represent shape continuously as the decision boundary of a binary classifier in 3D space. As every point in space is assigned an occupancy value as well as a continuous trajectory over time, we term our new representation *Occupancy Flow* (OFlow). Our representation is not only spatially and temporally continuous, but also *implicitly* provides correspondences at *every* point in space, so that OFlow can be seen as the continuous generalization of scene flow [67,68]. As a result, OFlow is not only suitable for reconstruction tasks, but also for a broader range of applications such as learning shape interpolations, finding correspondences between shapes, or learning probabilistic latent variable models. Furthermore, by modeling the temporal evolution of 3D shapes using continuum mechanics, our representation has a principled physical interpretation.

2. Related Work

We now discuss the most related 3D representations, 4D reconstruction techniques as well as shape registration and interpolation methods.

3D Representations: Recently, learning-based methods have shown promising results for various 3D tasks. They can broadly be categorized into voxel-based [10, 14, 21, 53, 54, 59, 74], point cloud-based [1, 20], mesh-based [25, 29, 34, 71], and continuous representations [13, 22, 28, 38, 39, 46, 56, 72].

While voxel representations can be easily incorporated in a deep learning pipeline, even variants which operate on sparse data structures are limited to relatively small resolutions of up to 256^3 or 512^3 voxels [23, 54, 61]. Point clouds constitute a more memory-efficient alternative [51, 52], but do not provide any connectivity information and hence require intensive post-processing. Mesh-based methods [25, 34, 71] do not require any post-processing, but producing the final graph consisting of edges and vertices is not straightforward, especially with deep learning methods. Therefore, the task is often simplified by deforming an initial mesh [73] or stitching multiple 3D patches [24]. More recently, continuous representations have been introduced which describe the 3D geometry implicitly [13,22,28,38,39,46,56,72]. In contrast to the methods above, these approaches are not restricted by discretization and allow for modeling arbitrary topologies. Therefore, they form an ideal basis for the Occupancy Flow representation we propose.

4D Reconstruction: Most works in the area of 4D reconstruction are restricted to a fixed domain by utilizing a template model [2, 5, 15, 27, 30, 63, 75], requiring a multi-view setup [33, 41, 42, 44, 45, 58, 64], or making strong assumptions about the motion e.g. rigidity or linearity [4, 37, 48, 65, 70].

Mustafa et al. [41, 42] perform 4D reconstruction of dynamic scenes by utilizing multiple views. However, the method requires a sufficient number of wide-baseline views to cover the scene and is limited by ambiguities in these views. Wand et al. [70] present a carefully engineered technique to reconstruct deforming 3D geometry from point clouds. While producing compelling results, their method is restricted to spatio-temporal smooth and small movements, assumes temporally dense sampling of the point clouds and is computationally costly.

Another successful line of work utilizes template models to guide the reconstruction process [2, 17, 27, 30, 63, 75]. While providing a valuable framework for classical and learning-based models, by definition those results are restricted by the quality and availability of a template model and are extremely domain specific. In addition, obtaining an adequate template is very costly, so that most existing efforts focus on particular shape categories such as human bodies, hands, or faces [7, 35, 47, 49, 55].

In contrast to all these approaches to 4D reconstruction, our deep learning based method requires neither a carefully engineered multi-view setup nor a domain-specific template model and can handle both rigid and non-rigid motion.

Shape Registration and Interpolation: In the graphics community, a large body of research has targeted tasks such as 3D shape interpolation, registration, and matching. Due to limited scope, we restrict the discussion to only the most relevant works, referring the reader to [6, 60, 66] for a thorough discussion.

Our approach to modeling time-varying geometry is related to deformation field-based methods [36, 43] which have a long-standing history in computer graphics [40, 69]. However, in contrast to our method, these approaches usually only consider vector fields on a small set of input points. Eisenberger et al. [19] calculate a deformation field over the entire embedding space, but differently from our method cannot handle volumetric changes in geometry. Slavcheva et al. [57] present a related approach which implicitly obtains correspondences by predicting the evolution of a signed distance field. However, as they require a Laplacian operator to be invariant, it only succeeds under small motions. Groueix et al. [24] introduce Shape Deformation Networks in which mesh correspondences are learned by predicting a template parameterization as well as transformations from arbitrary shapes to this template. While this approach achieves promising results for shape registration, it is highly specialized to certain object classes and requires costly fine-tuning for every registration.

In contrast to all approaches discussed in this section, our approach is not confined to shape registration, but is a general 4D reconstruction method handling a wide range of different input types and 3D topologies. Moreover, in contrast to classical vector field-based methods which require carefully engineered inference pipelines for different domains, our learning based approach can automatically obtain rich prior knowledge from observations to resolve ambiguities.

3. Method

In this section, we introduce our novel time-varying representation of 3D geometry which we term *Occupancy Flow* (OFlow). We start by formally introducing our model. Next, we explain how this representation can be learned from various types of input such as sequences of point clouds or images. Finally, the inference procedure as well as implementation details are provided. Figure 2 contains an overview of our method.

3.1. Occupancy Flow

We consider the challenging problem of estimating non-rigid 3D geometry jointly over space and time. More specifically, we are interested in inferring the evolution of a continuous 3D shape representation which implicitly and densely captures correspondences across time. We will use boldface type for vectors and vector-valued functions and regular font type for scalars and scalar functions.

Let $\mathbf{s} : [0, T] \to \mathbb{R}^3$ define the continuous *3D trajectory* of a point over the time interval $[0, T]$ such that $\mathbf{s}(0) \in \mathbb{R}^3$ and $\mathbf{s}(T) \in \mathbb{R}^3$ denote the start and end locations of the trajectory. Let further $\mathbf{v} : \mathbb{R}^3 \times [0, T] \to \mathbb{R}^3$ denote the continuous *velocity field* which describes the 3D velocity at every point in space and time. The relationship between $\mathbf{s}(\cdot)$ and $\mathbf{v}(\cdot, \cdot)$ is governed by the following differential equation

$$\frac{\partial \mathbf{s}(t)}{\partial t} = \mathbf{v}(\mathbf{s}(t), t) \tag{1}$$

with $t \in [0, T]$. When solving this ordinary differential equation (ODE) [62] for every initial condition $\mathbf{s}(0) = \mathbf{p}$ with $\mathbf{p} \in \mathbb{R}^3$ we obtain the *forward flow* $\Phi : \mathbb{R}^3 \times [0, T] \to \mathbb{R}^3$ (Fig. 2a) satisfying:

$$\frac{\partial \Phi}{\partial t}(\mathbf{p}, t) = \mathbf{v}(\Phi(\mathbf{p}, t), t) \quad \text{s.t.} \quad \Phi(\mathbf{p}, 0) = \mathbf{p} \tag{2}$$

Intuitively, the flow $\Phi(\mathbf{p}, t)$ describes the location of initial point $\mathbf{p}$ at time t when following the vector field $\mathbf{v}(\cdot, \cdot)$. In order to propagate spatial information (e.g., volumetric occupancy or mesh vertices) forward in time, we can reformulate (2) as follows

$$\Phi(\mathbf{p}, \tau) = \mathbf{p} + \int_0^\tau \mathbf{v}(\Phi(\mathbf{p}, t), t) \mathrm{d}t \tag{3}$$

where $\tau \in [0, T]$ denotes an arbitrary point in time and $\mathbf{p}$ a spatial location in $\mathbb{R}^3$. This equation can be solved with standard numerical solvers such as Runge-Kutta [62].

We can also regard $\Phi(\cdot, \tau)$ as a coordinate transformation that transforms a coordinate system at time $t = 0$ to a coordinate system at time $t = \tau$. In the field of continuum mechanics these coordinate systems are often referred to as "material coordinate system" and "spatial coordinate system", respectively [3].

We define the *backward flow* $\Psi : \mathbb{R}^3 \times [0, T] \to \mathbb{R}^3$ (Fig. 2b) as the inverse transformation of Φ. This inverse transformation can be computed by solving the reverse ODE

$$\frac{\partial \mathbf{r}(t)}{\partial t} = -\,\mathbf{v}(\mathbf{r}(t), t) \quad \text{s.t.} \quad \mathbf{r}(\tau) = \mathbf{p} \tag{4}$$

for every $(\mathbf{p}, \tau) \in \mathbb{R}^3 \times [0, T]$ and setting $\Psi(\mathbf{p}, \tau) = \mathbf{r}(0)$. As correspondences across time are implicitly captured it is sufficient to represent the 3D shape in the coordinate system at time $t = 0$. The 3D shape at other points in time can then be obtained by propagation using (3).

For representing the 3D shape at time $t = 0$ we choose the recently proposed occupancy function $f : \mathbb{R}^3 \to \{0, 1\}$ representation [38] which assigns an occupancy value to every 3D point. In contrast to mesh- or point-based representations, occupancy functions allow for representing smooth shapes at arbitrary resolution and with arbitrary topology. We parameterize both the occupancy function $f(\cdot)$ as well as the velocity field $\mathbf{v}(\cdot, \cdot)$ using neural networks

$$f_\theta : \mathbb{R}^3 \to [0, 1] \tag{5}$$

$$\mathbf{v}_{\hat{\theta}} : \mathbb{R}^3 \times [0, T] \to \mathbb{R}^3 \tag{6}$$

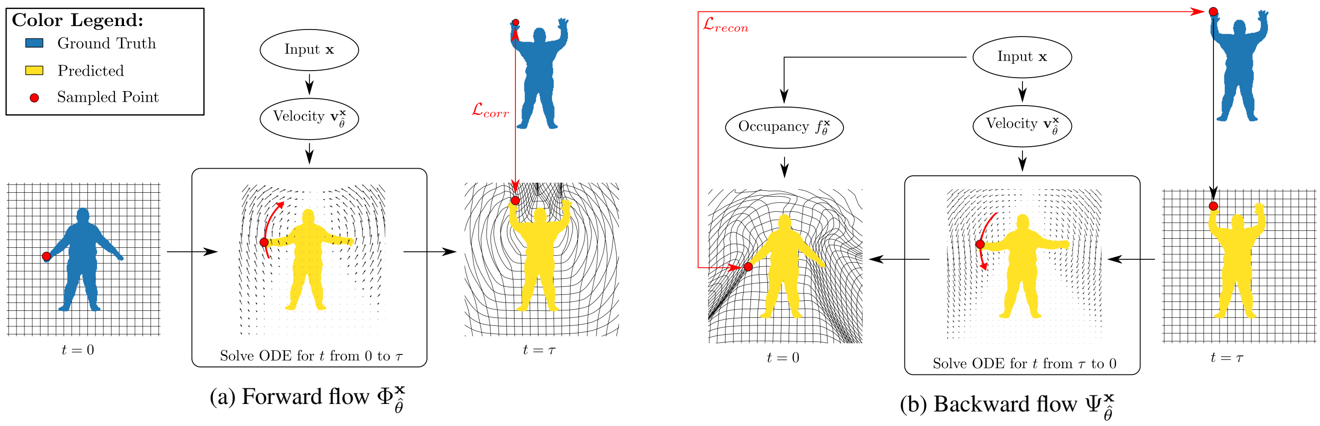
where θ and $\hat{\theta}$ denote the network parameters. In the following, we will call $f_\theta(\cdot)$ the *occupancy network* [38] and $\mathbf{v}_{\hat{\theta}}(\cdot, \cdot)$ the *velocity network*. We will now describe how the parameters of (5) and (6) can be learned from data.

3.2. Training

Our goal is to learn the parameters θ and $\hat{\theta}$ of $f_\theta(\cdot)$ and $\mathbf{v}_{\hat{\theta}}(\cdot, \cdot)$ using samples drawn from the 4D occupancy space-time volume, i.e., each sample represents the occupancy state at a particular point in space and time. Since we have chosen $t = 0$ as the reference coordinate system for representing the shape, each sample with $t > 0$ must be mapped back to its location at $t = 0$ in order to train the occupancy and the velocity networks. Towards this goal we use the backward flow $\Psi : \mathbb{R}^3 \times [0, T] \to \mathbb{R}^3$ described above (Fig. 2b). The predicted occupancy $\hat{o}_{\theta, \hat{\theta}}(\mathbf{p}, t)$ of 3D point $\mathbf{p}$ at time t is given by

$$\hat{o}_{\theta, \hat{\theta}}(\mathbf{p}, t) := f_\theta\left(\Psi_{\hat{\theta}}(\mathbf{p}, t)\right) \tag{7}$$

where we have used the notation $\Psi_{\hat{\theta}}$ to indicate that the inverse transformation depends on the parameters of the velocity network $\mathbf{v}_{\hat{\theta}}(\cdot, \cdot)$.


Color Legend:
🟦 Ground Truth
🟨 Predicted
● Sampled Point
Input $\mathbf{x}$
Velocity $\mathbf{v}^{\mathbf{x}}_{\hat{\theta}}$
$\mathcal{L}_{corr}$
$t = 0$
Solve ODE for t from 0 to τ
$t = \tau$
$\mathcal{L}_{recon}$
Input $\mathbf{x}$
Occupancy $f^{\mathbf{x}}_{\theta}$
Velocity $\mathbf{v}^{\mathbf{x}}_{\hat{\theta}}$
$t = 0$
Solve ODE for t from τ to 0
$t = \tau$

(a) Forward flow $\Phi^{\mathbf{x}}_{\hat{\theta}}$

(b) Backward flow $\Psi^{\mathbf{x}}_{\hat{\theta}}$

Figure 2: **Model Overview.** (a) During inference and to compute the correspondence loss $\mathcal{L}_{corr}$ defined in (10) we propagate points on the ground truth mesh at $t = 0$ forward in time by integrating an input-dependent vector field $\mathbf{v}^{\mathbf{x}}_{\hat{\theta}}$. We obtain the correspondence loss $\mathcal{L}_{corr}$ by taking the ℓ_2-distance between the propagated points and the ground truth points on the mesh at $t = \tau$. (b) To compute the reconstruction loss $\mathcal{L}_{recon}$ we go backward in time to transform a random point $\mathbf{p}$ into the coordinate system at $t = 0$. This allows us to compute the predicted occupancy probability $\hat{o}_{\theta,\hat{\theta}}(\mathbf{p}, \tau, \mathbf{x})$ by evaluating the occupancy network $f^{\mathbf{x}}_{\theta}$ at $t = 0$ using (8). The reconstruction loss is now given by taking the binary cross-entropy wrt. the ground truth occupancy at $t = \tau$.

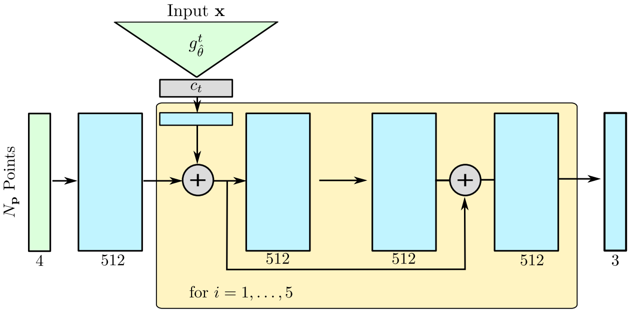


Figure 3: **Velocity Network Architecture.** Green color indicates input, cyan fully connected layers, and gray other operations. The occupancy network architecture is similar except that the input point dimension is 3 (no temporal axis), the outputs are occupancy probabilities of dimension 1, and conditional batch normalization [16, 18] is used instead of the adding operation for conditioning on input $\mathbf{x}$.[1]

To perform 4D reconstruction, the networks may also be conditioned on some additional input $\mathbf{x}$, e.g., an image sequence or a sequence of point clouds. Let $f^{\mathbf{x}}_{\theta}(\cdot)$ and $\mathbf{v}^{\mathbf{x}}_{\hat{\theta}}(\cdot, \cdot)$ denote the conditioned occupancy and velocity networks, respectively. The predicted occupancy $\hat{o}_{\theta,\hat{\theta}}(\mathbf{p}, t, \mathbf{x})$ of 3D point $\mathbf{p}$ at time t conditioned on input $\mathbf{x}$ is given by:

$$\hat{o}_{\theta,\hat{\theta}}(\mathbf{p}, t, \mathbf{x}) := f^{\mathbf{x}}_{\theta}\left(\Psi^{\mathbf{x}}_{\hat{\theta}}(\mathbf{p}, t)\right) \tag{8}$$

The model can be trained by minimizing the binary cross-entropy error (BCE) between the predicted occupancy $\hat{o}$ and the observed occupancy o of 3D point $\mathbf{p}$ at time τ:

$$\mathcal{L}_{recon}\left(\theta, \hat{\theta}\right) = \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{p},\tau,\mathbf{x},o)\in\mathcal{B}} \text{BCE}(\hat{o}_{\theta,\hat{\theta}}(\mathbf{p}, \tau, \mathbf{x}), o) \tag{9}$$

Here, $\mathcal{B}$ denotes a mini-batch which comprises samples from multiple sequences and at multiple time instances τ.

It is important to note that training our model does *not* require any correspondences across time. However, if available, additional correspondence information can be incorporated (Fig. 2a) by propagating 3D points $\mathbf{p}$ from time $t = 0$ to time $t = \tau$ using the forward flow $\Phi(\mathbf{p}, \tau)$ in (3). The correspondence loss function minimizes the ℓ_2 distance between the predicted location $\Phi^{\mathbf{x}}_{\hat{\theta}}(\mathbf{s}(0), \tau)$ and the observed location $\mathbf{s}(\tau)$ at time τ as follows

$$\mathcal{L}_{corr}\left(\hat{\theta}\right) = \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{s},\tau,\mathbf{x})\in\mathcal{B}} \left\|\Phi^{\mathbf{x}}_{\hat{\theta}}(\mathbf{s}(0), \tau) - \mathbf{s}(\tau)\right\|_2 \tag{10}$$

where $\mathbf{s}$ denotes the ground truth trajectory of a 3D point.

The gradients of (9) and (10) can be efficiently obtained using the adjoint sensitivity method [12, 50] by solving a second augmented ODE backwards in time. This way, the memory footprint can be kept constant with the tradeoff of longer computing time. For adaptive ODE solvers, relative and absolute error tolerances can be chosen to balance time and accuracy. For details we refer the reader to [12].

3.3. Inference

For a new observation $\mathbf{x}$ we predict the time varying 3D shape by first reconstructing the shape in the reference coordinate system at $t = 0$, followed by propagating the reconstruction into the future $t \in (0, T]$. While various shape

[1] See supplementary for the occupancy network architecture.

representations could be employed with our method, we utilize *Multiresolution IsoSurface Extraction (MISE)* [38] to extract a mesh $\mathcal{M}_0 = (\mathcal{V}_0, \mathcal{F}_0)$ from the prediction of the occupancy network f_θ at time $t = 0$. Here, $\mathcal{V}_0$ and $\mathcal{F}_0$ denote the vertices and the faces of mesh $\mathcal{M}_0$, respectively. For later time steps t, we use the trained velocity network $\mathbf{v}_{\hat{\theta}}$ in order to obtain the forward transformation $\Phi_{\hat{\theta}}(\mathbf{p}_i, t)$ for all vertices $\mathbf{p}_i$ in $\mathcal{V}_0$ by solving (3). The mesh at time t is given as:

$$\mathcal{M}_t = \left(\{ \Phi_{\hat{\theta}}(\mathbf{p}_i, t) \,|\, \mathbf{p}_i \in \mathcal{V}_0 \} \,, \, \mathcal{F}_0 \right) \tag{11}$$

Note that the mesh has to be extracted only once during inference. Therefore, inference for a large number of time steps is significantly faster compared to the naïve solution which extracts a mesh independently at every time step. Moreover, we implicitly obtain temporal correspondences (i.e., the mesh vertices correspond across time) even when using only the reconstruction loss (9) during training.

3.4. Implementation Details

For both the occupancy network and the velocity network we use a fully-connected ResNet-based [26] architecture shown in Fig. 3. For conditioning the occupancy network $f_\theta^{\mathbf{x}}$ and the velocity network $\mathbf{v}_{\hat{\theta}}^{\mathbf{x}}$ on a sequence of observations $\mathbf{x} = (\mathbf{x}_i)_{i=1,\ldots,L}$ with length L, we use two separate encoder networks $g_\theta^s(\mathbf{x}_1)$ and $g_{\hat{\theta}}^t(\mathbf{x})$, where the *spatial encoder* $g_\theta^s(\mathbf{x}_1)$ is only applied to the first observation $\mathbf{x}_1$ and the *temporal encoder* $g_{\hat{\theta}}^t(\mathbf{x})$ is applied to the whole sequence of L observations $\mathbf{x}$. The input $\mathbf{x}$ could for example be a sequence of images where $\mathbf{x}_i$ indicates the i-th image of this sequence. While we use the output of the spatial encoder to condition the occupancy network $f_\theta^{\mathbf{x}}$ on $\mathbf{x}$, we use the output of the temporal encoder to condition the velocity network $\mathbf{v}_{\hat{\theta}}^{\mathbf{x}}$ on $\mathbf{x}$. Depending on whether we use a sequence of point clouds or a sequence of images as input, we use a PointNet [51] or a Resnet-18 [26] for the spatial encoder g_θ^s. For the temporal encoder $g_{\hat{\theta}}^t$, we use an adjusted PointNet architecure with input dimension $3 \times L$ and a 3D convolutional network for point cloud and image input, respectively.

For training, we use the Adam optimizer [31] with learning rate 10^{-4} and train with batch size 16. More details can be found in the supplementary material.

4. Experiments

We conduct four different types of experiments to investigate the effectiveness of our approach. First, we evaluate the **representation power** of our vector field-based representation by training it to reproduce complex 3D motions. We further investigate the **reconstruction capacity** of our representation by conditioning the network on a sequence of images or noisy point clouds. We then investigate the quality of the learned **interpolations and correspondences** between two meshes or point clouds, respectively. Finally, we examine its **generative capabilities** by training a variational autoencoder [32] and investigating the quality of the latent representation.[2]

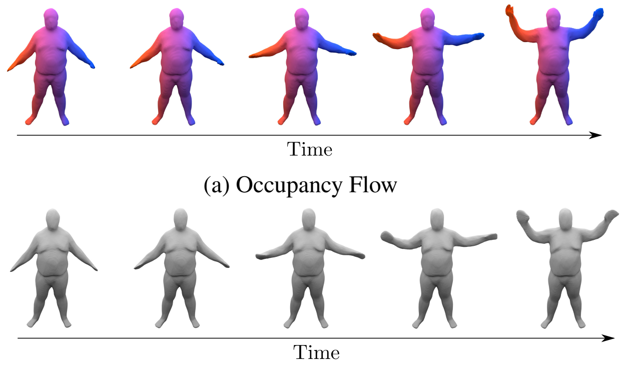
Baselines: A natural baseline for 4D reconstruction from image sequences or point clouds is to extend occupancy networks (ONet) [38] to the temporal domain by sampling points in 4D space. Similar to our method, this ONet 4D is continuous in time and space and can hence represent complex motions of 3D objects with arbitrary topology. However, in contrast to our representation, extracting meshes from this ONet 4D is time consuming (as mesh extraction is done at every frame) and does not yield correspondences across time. As an additional baseline, we implement a 4D extension of Point Set Generation Network (PSGN) [20] by predicting a set of trajectories instead of single points. For a fair comparison, we train this PSGN 4D both with and without temporal correspondences. For the former case, we evaluate the Chamfer-loss independently per time step. For the latter case we introduce a generalization of the Chamfer-loss which considers entire trajectories of points instead of independent 3D locations at each point in time.[3]

In the shape matching and interpolation experiment, we compare to nearest neighbor matching, Coherent Point Drift (CPD) [43], and 3D-Coded [24], a state-of-the-art method for finding correspondences between human shapes.

Datasets: We use the Dynamic FAUST (D-FAUST) [9] dataset which contains scans and meshes for 129 sequences of 10 real humans performing various motions such as "punching", "chicken wings", or "jumping jacks". D-FAUST is very challenging not only due to the fine structure of the human body, but also its non-rigid complex movements which include soft-tissue motion. As each sequence is relatively long (up to 1,251 time steps) and to increase the size of the dataset, we subsample each sequence into smaller clips of 17 to 50 time steps, depending on the experiment. We randomly divide all sequences into training (105), validation (6), and test (9) sequences so that the models are evaluated on combinations of individuals and motions not seen during training. In addition, we withhold one individual (12 sequences) to test generalization capabilities across individuals.

Due to the lack of publicly available datasets of time-varying non-rigid 3D geometry, we further introduce *Warping Cars*, a synthetic dataset of large scale deformations of cars. It allows to examine how well our method performs on other types of deforming objects than humans. To this end, we utilize the ShapeNet [11] "car" category and apply random displacement fields to obtain a continuous warping motion. Details of the data generation process can be found

[2] See supplementary for experiments regarding the generative model.
[3] See supplementary for a formal definition.



(a) Occupancy Flow

(b) 4D Occupancy Network

	IoU	Chamfer	Time (s)	Time w/o MC (s)
ONet 4D	**94.6 %**	**0.028**	15.509	5.802
OFlow	93.4 %	0.031	**0.716**	**0.520**

(c) Reconstruction Accuracy and Runtime

Figure 4: **Representation Power.** Correspondences are shown with the same color. While both, ONet 4D and OFlow, successfully learn to represent the complex 3D motion, only OFlow yields correspondences over time which also results in faster inference. We show inference times for all 50 time steps with and without marching cubes (MC).

in the supplementary material.

Metrics: We use volumetric IoU and Chamfer distance for evaluating the reconstruction at each time step. We refer to [38] for an in-depth description of these metrics. For evaluating the quality of the estimated correspondences, we introduce a correspondence distance as follows: The K points $\mathbf{p}^{(k)}(0)$, $k \in \{1, \dots, K\}$, of the output at $t = 0$ are assigned to the nearest neighbor $\mathbf{p}_{GT}^{(k)}(0)$ on the ground truth mesh. We then find the point $\mathbf{p}_{GT}^{(k)}(\tau)$ corresponding to $\mathbf{p}_{GT}^{(k)}(0)$ on the ground truth mesh at $t = \tau$. Similarly, we find the point $\mathbf{p}^{(i)}(\tau)$ corresponding to $\mathbf{p}^{(k)}(0)$ in the output of the method. The correspondence ℓ_2-distance at time $t = \tau$ is then defined as the average ℓ_2-distance between the points $\mathbf{p}^{(k)}(\tau)$ and $\mathbf{p}_{GT}^{(k)}(\tau)$. Note that this distance can only be computed for methods like ours that yield correspondences across time, but not ONet 4D. Similar to [20,38] we use $1/10$ times the length of the maximal edge length of the object's bounding box as unit 1.

4.1. Representation Power

In this experiment we investigate how well our Occupancy Flow model can represent 3D shapes in motion. In particular, we would like to disentangle the influence of the spatial and temporal encoders g_θ^s and $g_{\hat{\theta}}^t$ from the representation power of the Ocupancy Flow model. Towards this goal, we train our networks to reconstruct complex 3D motions without any external input $\mathbf{x}$.

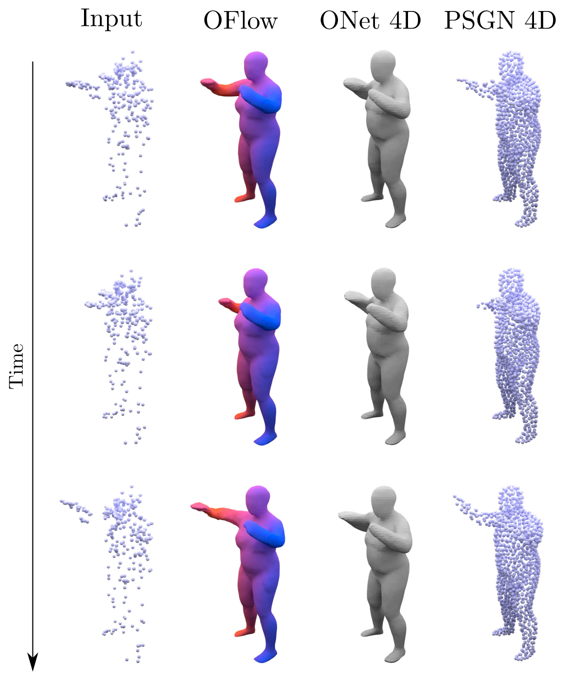


Figure 5: **4D Point Cloud Completion.** We show three equally spaced time steps between 0 and 1 for the input and the output of OFlow (w/ correspond.), ONet 4D, and PSGN 4D (w/ correspond.). The color coding for the first method illustrates correspondences across time.

For training, we select 3 sequences of length 50 from the training split of the D-FAUST dataset on which we (separately) train our networks only using the $\mathcal{L}_{recon}$ loss in (9). We compare against ONet 4D.

The results of this experiment are shown in Fig. 4. We see that our method learns an accurate representation of the deforming 3D geometry, yielding similar IOU and Chamfer values as ONet 4D. However, in contrast to ONet 4D, we only have to extract a mesh once for $t = 0$ whose vertices we then propagate forward in time by solving a time-dependent ODE, leading to much faster inference. Moreover, while both ONet 4D and our approach successfully learn to represent the complex 3D motion, only our approach yields correspondences over time.

4.2. 4D Point Cloud Completion

In this first reconstruction experiment, the input for the network are 300 discrete point trajectories, each consisting of $L = 17$ time steps. We perturb the point clouds with Gaussian noise with standard deviation 0.01. A real world scenario for this would for example be (noisy) motion capture data from a set of markers.

We train our method using the reconstruction loss $\mathcal{L}_{recon}$ in (9), which does not utilize any correspondences. Moreover, we also investigate the performance of our method when trained with both the reconstruction loss $\mathcal{L}_{recon}$ and

	IoU	Chamfer	Correspond.
PSGN 4D	-	0.108	3.234
PSGN 4D (w/ cor.)	-	0.101	0.102
ONet 4D	77.9 %	0.084	-
OFlow	79.9 %	0.073	0.122
OFlow (w/ cor.)	**81.5 %**	**0.065**	**0.094**

(a) Seen individuals

	IoU	Chamfer	Correspond.
PSGN 4D	-	0.127	3.041
PSGN 4D (w/ cor.)	-	0.119	0.131
ONet 4D	66.6 %	0.140	-
OFlow	69.6 %	0.095	0.149
OFlow (w/ cor.)	**72.3 %**	**0.084**	**0.117**

(b) Unseen individual

Table 1: **4D Point Cloud Completion (D-FAUST).** These tables show quantitative results for the 4D point cloud completion experiment on the D-FAUST dataset. We report volumetric IoU (higher is better), Chamfer distance (lower is better) and the correspondence ℓ_2-distance (lower is better) for both individuals seen during training and the unseen individual.

	IoU	Chamfer	Correspond.
PSGN 4D	-	**0.157**	3.886
ONet 4D	69.7 %	0.190	-
OFlow	**70.7 %**	0.169	**0.283**

Table 2: **4D Point Cloud Completion (Warping Cars).** This table shows quantitative results for the 4D point cloud completion experiment on the warping cars dataset.

the correspondence-based loss $\mathcal{L}_{corr}$ in (10).

We compare against ONet 4D and PSGN 4D. For a fair comparison, we train all methods with the same ResNet-based [26] PointNet [51] temporal encoder from Section 3.4. We do not use an additional spatial encoder for ONet 4D and PSGN 4D as both methods do not represent shape and motion disentangled.

The quantitative and qualitative results for the D-FAUST dataset are summarized in Table 1 and Fig. 5. We observe that OFlow outperforms ONet 4D in terms of IOU and achieves the lowest Chamfer distance compared to both PSGN variants and ONet 4D. This is surprising, as PSGN was explicitly trained on the Chamfer distance whereas OFlow was not. OFlow trained with both losses achieves the lowest correspondence ℓ_2-distance. Interestingly, OFlow trained only with the reconstruction loss achieves an only slightly worse correspondence loss even though it did not use any correspondences during training. In contrast, the PSGN variant that does not use any correspondences during training does not learn meaningful correspondences. This shows that our vector field representation is helpful for learning correspondences over time. Qualitatively (Fig. 5), we observe that OFlow learns a realistic 3D motion while ONet 4D does not. PSGN is also able to reconstruct the 3D motion, but lacks spatial connectivity.

Quantitative results for the Warping Cars dataset are shown in Table 2. We see that OFlow also works well in a very different domain and achieves the best IoU and correspondence ℓ_2-distance.

4.3. Reconstruction from Image Sequences

In this experiment we consider 4D reconstruction from a sequence of single-view images as observation $\mathbf{x}$. For all methods we use the temporal encoder architecture described in Section 3.4.

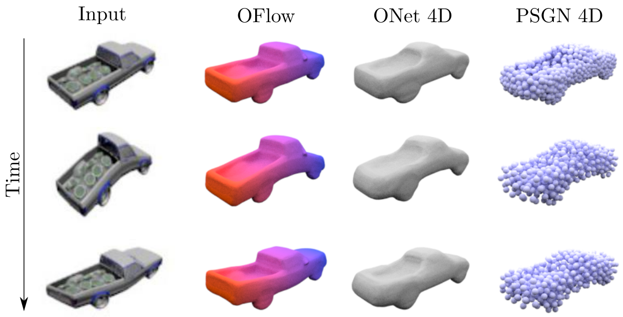


Figure 6: **Single Image 4D Reconstruction.** We show three time steps between 0 and 1 for input as well as the output of OFlow, ONet 4D and PSGN 4D. Similar to Figure 5, the color coding illustrates the correspondences.

In Table 3 and Fig. 6 we provide a summary of the quantitative and qualitative results. Similar to [38] and others, we observe that reconstruction from single-view image sequences is a harder task than 4D point cloud completion. We suspect the global image encoding as well as occlusions to be the main challenge as the viewpoint is sampled randomly for the clips which sometimes causes the motion to be invisible in the images. The quantitative performance differences are similar to the point cloud experiment. The qualitative results in Fig. 6 show that while OFlow can reconstruct the complicated 3D motion from the provided sequence reasonably well, the other methods struggle to do so. It suggests that the disentangled shape and motion representation of OFlow results in better reconstructions and biases the network towards a physically plausible motion.

4.4. Interpolation and Mesh Correspondence

The goal of the next two experiments is to investigate to which degree our method can be used for shape matching and interpolation. In both experiments, the task is to find a continuous transformation between the underlying surfaces of two randomly sampled point clouds. We train our model only using the correspondence loss (10) as recovering the 3D shape is not required in this setting.

	IoU	Chamfer	Correspond.
PSGN 4D	-	0.258	2.576
PSGN 4D (w/ cor.)	-	0.265	2.580
ONet 4D	44.0 %	0.348	-
OFlow	56.6 %	0.193	0.292
OFlow (w/ cor.)	**59.6 %**	**0.166**	**0.226**

(a) D-FAUST

	IoU	Chamfer	Correspond.
PSGN 4D	-	**0.251**	3.949
ONet 4D	55.6 %	0.319	-
OFlow	**58.2 %**	0.277	0.491
OFlow (w/ cor.)	58.0 %	0.263	**0.487**

(b) Warping cars

Table 3: **4D Reconstruction from Images** The two tables summarize the quantitative results for 4D reconstruction from image sequences.

	Correspond	Time (s)
Baseline NN	0.374	**0.004**
Coherent Point Drift [43]	0.189	343.621
OFlow	0.167	0.608
3D-Coded [24]	**0.096**	199.368

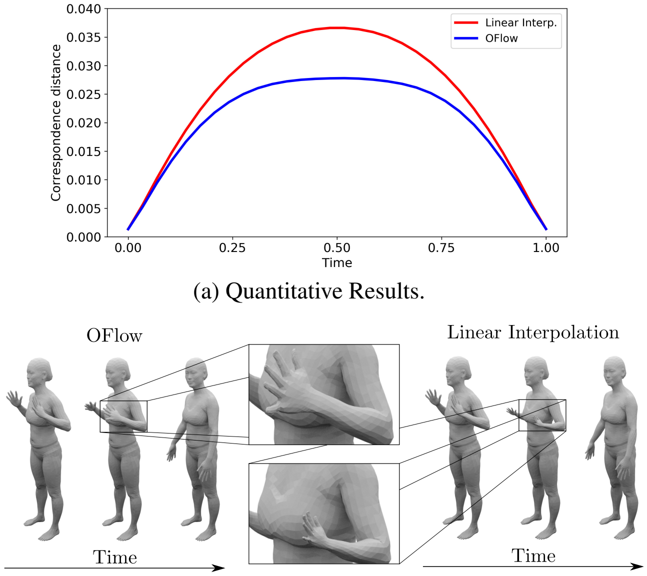
Table 4: **Shape Matching.** This table shows results for shape matching from point clouds on the D-FAUST dataset.

We first evaluate the quality of the correspondences learned by our method. We use the same splits on the D-FAUST dataset as before. We compare against nearest neighbor matching, non-rigid Coherent Point Drift [43] (CPD), and the specialized state-of-the-art learning-based method 3D-Coded [24]. While the first two find nearest neighbors or an optimal fit of GMM centroids in the second point cloud, the latter learns mappings to a human template model. For nearest neighbor matching, OFlow and 3D-Coded [24], we use two randomly sampled point clouds of size $10,000$ as input. As Coherent Point Drift [43] directly matches the point sets, we did not obtain competitive results for this method by using random point clouds so that we used the full set of vertices in this case. To adhere to community standards [8] we project predicted points which do not lie on the surface onto the final mesh for evaluation.

Our results are shown in Table 4. Even though our method is primarily concerned with 4D reconstruction, we find that it also estimates high-quality correspondences, outperforming both the nearest neighbor as well as the CPD baselines. While it performs worse than 3D-Coded, OFlow requires only a fraction of its inference time. Moreover, we remark that 3D-Coded is a highly specialized matching method including a costly fine-tuning for every registration whereas our approach is a general purpose 4D reconstruction method which estimates correspondences implicitly.

To evaluate the interpolation capabilities of OFlow, we increase the sequence length L from 17 to 30 and compare against the linear interpolation baseline. For OFlow, we predict the forward and backward motion and average the results.[4] For both methods we evaluate the correspondence ℓ_2-distance for all 30 time steps.

Quantitative and qualitative results are shown in Fig. 7.



(a) Quantitative Results.

(b) Qualitative Results.

Figure 7: **Interpolation.** The figure shows a quantitative and qualitative comparison of Occupancy Flow and the linear interpolation baseline. Occupancy Flow is able to better capture non-linear motion of non-rigid 3D shapes.

We observe that OFlow improves over the linear interpolation baseline as it is able to capture non-linear motion.

5. Conclusion

In this work, we introduced Occupancy Flow, a novel 4D representation of time-changing 3D geometry. In contrast to existing 4D representations, it does not utilize a template model, is continuous in space and time, and yields implicit temporal correspondences. Our experiments validate that it can be used effectively for shape matching and interpolation, 4D reconstruction, and generative tasks. We hence believe that Occupancy Flow is a useful representation which can be used in a wide of variety spatio-temporal tasks.

Acknowledgements

This work was supported by the Intel Network on Intelligent Systems, by Microsoft Research through its PhD Scholarship Program and by an NVIDIA research gift.

[4] See supplementary for details.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Learning representations and generative models for 3d point clouds. In *Proc. of the International Conf. on Machine learning (ICML)*, 2018. 2

[2] Thiemo Alldieck, Marcus A. Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll. Video based reconstruction of 3d people models. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2

[3] Romesh C Batra. *Elements of continuum mechanics*. Aiaa, 2006. 3

[4] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger. Bounding boxes, segmentations and object coordinates: How important is recognition for 3d scene flow estimation in autonomous driving scenarios? In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 2

[5] Amit Bermano, Thabo Beeler, Yeara Kozlov, Derek Bradley, Bernd Bickel, and Markus H. Gross. Detailed spatio-temporal reconstruction of eyelids. *ACM Trans. on Graphics*, 34(4):44:1–44:11, 2015. 1, 2

[6] Silvia Biasotti, Andrea Cerri, Alexander M. Bronstein, and Michael M. Bronstein. Recent trends, applications, and perspectives in 3d shape similarity assessment. *Computer Graphics Forum*, 35(6):87–119, 2016. 2

[7] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *ACM Trans. on Graphics*, 1999. 2

[8] Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. FAUST: dataset and evaluation for 3d mesh registration. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 8

[9] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: registering human bodies in motion. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5

[10] André Brock, Theodore Lim, James M. Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv.org*, 1608.04236, 2016. 2

[11] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *arXiv.org*, 1512.03012, 2015. 5

[12] Liang-Chieh Chen, Maxwell D. Collins, Yukun Zhu, George Papandreou, Barret Zoph, Florian Schroff, Hartwig Adam, and Jonathon Shlens. Searching for efficient multi-scale architectures for dense image prediction. In *Advances in Neural Information Processing Systems (NIPS)*, pages 8713–8724, 2018. 4

[13] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2

[14] Christopher Bongsoo Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2016. 1, 2

[15] Huseyin Coskun, Felix Achilles, Robert S. DiPietro, Nassir Navab, and Federico Tombari. Long short-term memory kalman filters: Recurrent neural estimators for pose regularization. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 1, 2

[16] Harm de Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C. Courville. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 4

[17] Jing Dong, John Gary Burnham, Byron Boots, Glen C. Rains, and Frank Dellaert. 4d crop monitoring: Spatio-temporal reconstruction for agriculture. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*, 2017. 2

[18] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2017. 4

[19] Marvin Eisenberger, Zorah Lähner, and Daniel Cremers. Divergence-free shape interpolation and correspondence. *Computer Graphics Forum*, July 2019. 2

[20] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 5, 6

[21] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2017. 2

[22] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 1, 2

[23] Ben Graham. Sparse 3d convolutional neural networks. In *Proc. of the British Machine Vision Conf. (BMVC)*, 2015. 2

[24] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. 3D-CODED: 3D correspondences by deep deformation. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 2, 5, 8

[25] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. AtlasNet: A papier-mâché approach to learning 3d surface generation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5, 7

[27] Yinghao Huang. Towards accurate marker-less human shape and pose estimation over time. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2017. 1, 2

[28] Zeng Huang, Tianye Li, Weikai Chen, Yajie Zhao, Jun Xing, Chloe LeGendre, Linjie Luo, Chongyang Ma, and Hao Li.

Deep volumetric video from very sparse multi-view performance capture. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 1, 2

[29] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 2

[30] Angjoo Kanazawa, Jason Zhang, Panna Felsen, and Jitendra Malik. Learning 3d human dynamics from video. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2

[31] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2015. 5

[32] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *Proc. of the International Conf. on Learning Representations (ICLR)*, 2014. 5

[33] Vincent Leroy, Jean-Sébastien Franco, and Edmond Boyer. Multi-view dynamic shape refinement using local temporal integration. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 1, 2

[34] Yiyi Liao, Simon Donne, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2

[35] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. on Graphics*, 2015. 2

[36] Marcel Lüthi, Thomas Gerig, Christoph Jud, and Thomas Vetter. Gaussian process morphable models. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 40(8):1860–1873, 2018. 2

[37] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[38] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2, 3, 5, 6, 7

[39] Mateusz Michalkiewicz, Jhony K. Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Deep level sets: Implicit surface representations for 3d shape inference. *arXiv.org*, 2019. 1, 2

[40] Michael I. Miller, Alain Trouvé, and Laurent Younes. Geodesic shooting for computational anatomy. *Journal of Mathematical Imaging and Vision*, 24(2):209–228, 2006. 2

[41] Armin Mustafa, Hansung Kim, Jean-Yves Guillemaut, and Adrian Hilton. General dynamic scene reconstruction from multiple view video. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2015. 1, 2

[42] Armin Mustafa, Hansung Kim, Jean-Yves Guillemaut, and Adrian Hilton. Temporally coherent 4d reconstruction of complex dynamic scenes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2

[43] Andriy Myronenko and Xubo B. Song. Point set registration: Coherent point drift. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 32(12):2262–2275, 2010. 2, 5, 8

[44] Jan Neumann and Yiannis Aloimonos. Spatio-temporal stereo using multi-resolution subdivision surfaces. *International Journal of Computer Vision (IJCV)*, 47(1-3):181–193, 2002. 1, 2

[45] Martin Ralf Oswald, Jan Stühmer, and Daniel Cremers. Generalized connectivity constraints for spatio-temporal 3d reconstruction. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2014. 1, 2

[46] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2

[47] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3d face model for pose and illumination invariant face recognition. In *Proc. of International Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, 2009. 2

[48] Yuri Pekelny and Craig Gotsman. Articulated object reconstruction and markerless motion capture from depth video. *Computer Graphics Forum*, 27(2):399–408, 2008. 1, 2

[49] Leonid Pishchulin, Stefanie Wuhrer, Thomas Helten, Christian Theobalt, and Bernt Schiele. Building statistical shape spaces for 3d human modeling. *Pattern Recognition*, 67:276–286, 2017. 2

[50] Lew S. Pontryagin, Vladimir G. Boltyanshii, Rewas V. Gamkrelidze, and Evgenii F. Mishenko. *The Mathematical Theory of Optimal Processes*. John Wiley and Sons, 1962. 4

[51] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 5, 7

[52] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 2

[53] Danilo Jimenez Rezende, S. M. Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. 2

[54] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2

[55] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: modeling and capturing hands and bodies together. *ACM Trans. on Graphics*, 36(6):245:1–245:17, 2017. 2

[56] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 1, 2

[57] Miroslava Slavcheva, Maximilian Baust, and Slobodan Ilic. Towards implicit correspondence in signed distance field

evolution. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV) Workshops*, 2017. 2

[58] Jonathan Starck and Adrian Hilton. Surface capture for performance-based animation. *IEEE Computer Graphics and Applications (CGA)*, 27(3):21–31, 2007. 1, 2

[59] David Stutz and Andreas Geiger. Learning 3d shape completion from laser scan data with weak supervision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[60] Gary K. L. Tam, Zhi-Quan Cheng, Yu-Kun Lai, Frank C. Langbein, Yonghuai Liu, A. David Marshall, Ralph R. Martin, Xianfang Sun, and Paul L. Rosin. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. 19(7):1199–1217, 2013. 2

[61] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 2

[62] Gerald Teschl. *Ordinary differential equations and dynamical systems*. American Mathematical Soc., 2012. 3

[63] Hsiao-Yu Tung, Hsiao-Wei Tung, Ersin Yumer, and Katerina Fragkiadaki. Self-supervised learning of motion capture. In *Advances in Neural Information Processing Systems (NIPS)*. 1, 2

[64] Ali Osman Ulusoy, Octavian Biris, and Joseph L. Mundy. Dynamic probabilistic volumetric models. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2013. 2

[65] Ali Osman Ulusoy and Joseph L Mundy. Image-based 4-d reconstruction using 3-d change detection. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2014. 2

[66] Oliver van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A survey on shape correspondence. *Computer Graphics Forum*, 30(6):1681–1707, 2011. 2

[67] Sundar Vedula, Simon Baker, Peter Rander, Robert T. Collins, and Takeo Kanade. Three-dimensional scene flow. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 1999. 2

[68] Sundar Vedula, Peter Rander, Robert T. Collins, and Takeo Kanade. Three-dimensional scene flow. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 27(3):475–480, 2005. 2

[69] Wolfram von Funck, Holger Theisel, and Hans-Peter Seidel. Vector field based shape deformations. *ACM Trans. on Graphics*, 25(3):1118–1125, 2006. 2

[70] Michael Wand, Philipp Jenke, Qi-Xing Huang, Martin Bokeloh, Leonidas J. Guibas, and Andreas Schilling. Reconstruction of deforming geometry from time-varying point clouds. In *Eurographics Symposium on Geometry Processing (SGP)*, 2007. 1, 2

[71] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 1, 2

[72] Weiyue Wang, Xu Qiangeng, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. DISN: Deep implicit surface network for high-quality single-view 3d reconstruction. *arXiv.org*, 2019. 1, 2

[73] Yang Wang, Yi Yang, Zhenheng Yang, Liang Zhao, Peng Wang, and Wei Xu. Occlusion aware unsupervised learning of optical flow. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[74] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. 2

[75] Qian Zheng, Xiaochen Fan, Minglun Gong, Andrei Sharf, Oliver Deussen, and Hui Huang. 4d reconstruction of blooming flowers. *Computer Graphics Forum*, 36(6):405–417, 2017. 1, 2