Learning to Reconstruct Texture-less Deformable Surfaces from a Single View

Jan Bednařík jan.bednarik@epfl.ch
Pascal Fua pascal.fua@epfl.ch
Mathieu Salzmann mathieu.salzmann@epfl.ch
École Polytechnique Fédérale de Lausanne, Switzerland

Abstract

Recent years have seen the development of mature solutions for reconstructing deformable surfaces from a single image, provided that they are relatively well-textured. By contrast, recovering the 3D shape of texture-less surfaces remains an open problem, and essentially relates to Shape-from-Shading. In this paper, we introduce a data-driven approach to this problem. We introduce a general framework that can predict diverse 3D representations, such as meshes, normals, and depth maps. Our experiments show that meshes are ill-suited to handle texture-less 3D reconstruction in our context. Furthermore, we demonstrate that our approach generalizes well to unseen objects, and that it yields higher-quality reconstructions than a state-of-the-art SfS technique, particularly in terms of normal estimates. Our reconstructions accurately model the fine details of the surfaces, such as the creases of a T-Shirt worn by a person.

1. Introduction

In this paper, we tackle the problem of recovering the shape of complex and deforming texture-less surfaces from a single image, which is close in spirit to Shape-from-shading (SfS) with the added difficulty that we must handle complex phenomena such as sharp creases and self-shadowing. The T-shirt of Fig. 1 being worn by someone who moves illustrates that. This is in contrast to recent approaches that focus on well-textured surfaces [23, 6], or partially textured ones [38, 39, 26], and tend to produce coarse reconstructions in which fine details are lost.

SfS is one of the oldest Computer Vision problems [15, 42, 10]. Yet to this day, it remains largely unsolved because it is such an ill-posed inverse problem, except in tightly controlled lighting environments [25]. The early methods were variatonal ones that required very strong assumptions about the world, such as the presence of a single light source together with simple reflectance properties of the surfaces to be reconstructed, which are rarely satisfied. Recent ones [5, 40, 29] have focused on replacing some of these assumptions by measurements of the surface and lighting properties, yet still rely on relatively simple geometric and photometric models to remain computationally tractable.

In this paper, we show that a data driven approach enables us to operate under much weaker assumptions that *are* sufficiently well satisfied in everyday life to make the method truly practical in an environment where the lighting can be complex and inter-reflections, shadows, and sharp creases are prevalent. Fig. 1 depicts such a situation in which we outperform one of the best currently available algorithms [5]. It would seem natural to follow the most popular trend in modeling deformable surfaces and to train a Deep Net to regress from the image to the shape parameters of a surface mesh, as was done for well-textured surfaces in [26]. We will demonstrate, however, that this is not the best approach. It is more effective to train a network that predicts a dense map of depths, normals, or both, as was done by the SfS pioneers [15].

Because we do not constrain the surface to be smooth and allow the network to learn about complex effects such as self-shadowing and occlusions, we can recover very severe deformations such as the sharp folds that can be seen in Fig. 1. Furthermore, as evidenced by our experiments, training on a single surface allows us to generalize to other ones of different shape and without any re-training or fine-tuning. Our two main contributions are: (i) A data-driven SfS approach that can recover much more complex deformations than earlier ones under realistic lighting conditions and which, unlike state-of-the-art intrinsic image decomposition techniques, only requires supervision in the form of depth and normal maps which are relatively easy to obtain. (ii) A large annotated real-world dataset consisting of 26500 samples of surfaces with uniform reflectance undergoing complex deformations and viewed under complex realistic lighting. We thoroughly evaluate the performance of our method and show that it outperforms the state-of-the-art SfS approach of [5], whose code is available on the web.

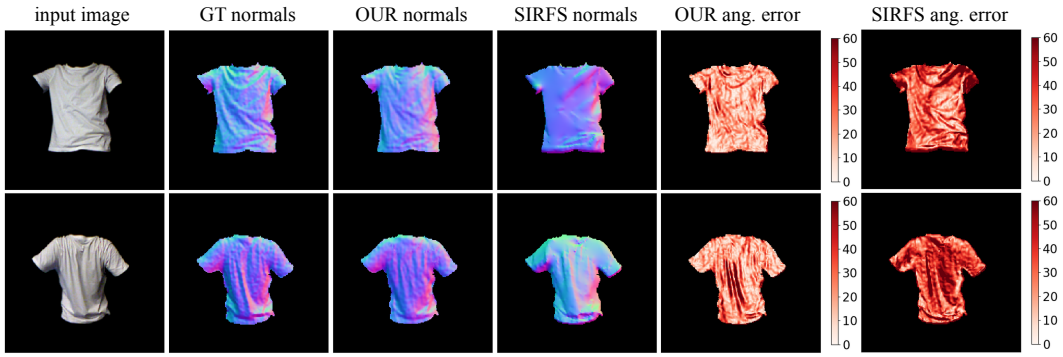


Figure 1. **Reconstructing a T-Shirt.** Comparison of our method to that of [5] on a deforming T-Shirt. The second to fourth columns depict the ground truth and recovered normals and the last two columns show the angular error in degrees. Note that the discrepancies are much smaller in our case and that the sharp creases are better recovered.

2. Related Work

Traditionally, the SfS problem has been posed as a variational problem involving the optimization of physically-inspired objective functions to impose brightness, smoothness, and integrability constraints. In its original form [15, 42, 10], the problem is underconstrained and its solution plagued by ambiguities [7, 11], which can be formally resolved only in very specific cases, such as when the camera and light source are co-located [25] or when additional stereo information is available [31].

Known lighting, and absence of interreflections and cast shadows are often assumed, as in the approach of [3] to jointly recover albedo and shape so as to explain the image as well as possible. In [24, 18], while known, the lighting is assumed to be natural, which makes it possible to treat the thee color channels of the image in a manner similar to that of photometric stereo. By contrast, our model does not require any prior knowledge about the lighting.

Assumptions are also often made about the shape. For example, in [40], quadratic functions are fitted to local image patches of different sizes, which allows the prediction of normals if the surface is sufficiently smooth, while in [14, 16], exemplars are used to provide shape priors. A different approach is to assume the direction of the normals to be correlated with their distance to the occlusion boundary [29], or to learn the smooth shape priors directly from data [5]. If the object category is known, sparse parameterization can be used instead of dense depth/normal maps or meshes. For instance, human face reconstruction approaches often rely on 3D Morphable Models [28, 34, 19, 2].

Recently, the most popular strategy has been to jointly infer two or more modalities that contribute to the image formation process, specifically normal map, depth map, surface reflectance, reflectance map and/or lighting parameters in either optimization based [5, 24, 43, 8] or learning based [29, 27, 32, 17, 34] setting. This has been one of our motivations for developing a multi-stream CNN model that outputs multiple shape representations, as will be discussed in Section 3. Even though we were inspired by Deep Net based models performing intrinsic image decomposition, our method relaxes some of the rather restricting assumptions and need for hard-to-obtain annotations. Specifically, [29] assumes a Lambertian reflectance model under Spherical Harmonics lighting, which is rarely the case in practice. [32, 17] require GT albedo and lighting annotations while [34] focuses solely on the human face object category as it relies on 3DMM representation. In [27], the surface normals are inferred as the by-product of reflectance map estimation, however, only low resolution of 64×64 px is supported and the results are only reported on synthetic data coming from a single object category. Another Deep Net based approach to directly predicting a normal map from an input image was introduced in [41] but it can only operate on infrared input images, whereas our approach takes a standard RGB image as input.

In our approach, we do not attempt to recover the lighting or reflectance explicitly, since such measurements are difficult to obtain ground-truth annotations for. Instead, we let the network learn how to handle these quantities from data. As evidenced by our experiments, even without explicitly modeling or predicting lighting and material BRDF, our network can successfully reconstruct fine surface details of complex shapes acquired under realistic conditions.

In the context of monocular 3D reconstruction of deformable surfaces, the most recent methods rely on CNNs to regress from the image to mesh vertices [26, 9]. However, while [26] can recover complex deformations, it focuses on well-textured surfaces. By contrast, [9] handles poorly-textured surfaces, but visual inspection of their results clearly shows that the method oversmoothes the shape. Our approach focuses on texture-less objects, and, as depicted by our results in Fig. 1, is able to reconstruct fine-grained deformations, such as the creases of a T-Shirt. Furthermore, we show that mesh representations are outperformed by normal- and depth-based ones for this task.

3. Our Approach

3.1. Problem Formulation

Let $\mathbf{I} \in \mathbb{R}^{H\times W\times 3}$ be an RGB image of size $W \times H$ and $\mathbf{B} \in \mathbb{N}_0^{H\times W\times 3}$ a binary mask that denotes the foreground region to be recovered. Our goal is to learn a mapping $f_{SR} : \mathbf{I} \odot \mathbf{B} \rightarrow \mathbf{S}$, where $\mathbf{S}$ represents the corresponding 3D surface. For deformable surfaces, a natural 3D representation would be a vector $\mathbf{S}_M$ containing the 3D vertex coordinates of a triangulated mesh, as in [26]. However, other representations such as a depth map $\mathbf{S}_D$ or a normal map $\mathbf{S}_N$, which are more prevalent in SfS papers, can also be used. In fact, these representations are not mutually exclusive, and we can train a network to return one or more of them, as shown in Fig. 2. In this section, we discuss this general scenario. However, in practice, we have found that the mesh-based representation was not as effective as the other two.

Given a calibrated camera, a triangulated mesh with V vertices can be expressed as a vector $\mathbf{S}_M \in \mathbb{R}^{3V}$ of 3D points in the camera coordinate frame. By contrast, a depth map and a normal map can be encoded as images instead of vectors. Specifically, the depth map $\mathbf{S}_D$ is a $W \times H$ floating point image, and the normal map $\mathbf{S}_N$ is a three-channel floating point image of size $W \times H \times 3$ representing the x, y and z coordinates of the normal vector expressed in the camera coordinate frame. Training depth maps can be easily acquired using existing depth sensors, such as the Microsoft Kinect camera. This data can be converted into ground-truth normal maps by smoothing and differentiating the depth maps as discussed in Section 4.3. By contrast, obtaining training data for 3D meshes for real images is harder and requires much more processing, since depth sensors do not provide correspondences between points on the surface. However, unlike the other two representations, 3D meshes can represent self-occluded parts of the surface, albeit at the cost of constraining the topology much more. We explain the process of obtaining the GT mesh coordinates in the supplementary material.

3.2. Shape Recovery Networks

In this work, we rely on the SegNet deep autoencoding architecture [1] depicted by Fig. 2 to regress from the image to each of our three representations.

Let $\mathbf{I}_m = \mathbf{I} \odot \mathbf{B}$ be the foreground image. The encoder performs feature extraction and outputs a latent representation tensor $\Lambda(\mathbf{I}_m) \in \mathbb{R}^{H_L\times W_L\times C_L}$, whose spatial size is $(H_L \times W_L)$ and third dimension C_L is the number of filters in the last convolutional layer of the encoder. As in many state-of-the-art intrinsic image decomposition approaches [17, 33, 34], we assume the learned features are independent of the final output modality, and thus we use the same encoder for all three shape representations. Keeping the encoder design and shape of latent representation $\Lambda(\mathbf{I}_m)$ the same for all three scenarios allows us to not only train each model separately but also jointly, which helps the model learning more robust feature extractors, which results in higher reconstruction accuracy as shown in Section 5. Note, however, that the weights of this encoder will differ if we train it, for instance, for depth map prediction only or for mesh prediction only.

Let Ψ_C, Ψ_D, Ψ_N be the decoders for mesh vertices, depth, and normals, respectively. Inspired by branched Deep Net architectures, which have been shown to perform well for intrinsic image decomposition [17, 34] and multi-task learning [35], we do not force the design of the decoders to match each other but rather adjust them to suit the output shape and/or topology.

Since the depth and normal maps both have an image-like topology, as does the output of the original SegNet, we use the same SegNet decoder architecture for these two modalities, except for the number of convolutional filters in the last convolutional layer, that is, 1 for depth and 3 for normals, and for the fact that these outputs do not need to be passed through a softmax. By contrast, when using the mesh representation, the output size is significantly smaller and shaped as a vector. We therefore take Ψ_C to be a single convolutional layer followed by average pooling and a fully connected layer to regress to the vertices' coordinates.

3.3. Loss Functions

Let us assume to be given N training samples. We represent each one as a tuple $(\mathbf{I}^n, \mathbf{B}^n, \mathbf{v}^n, \mathbf{D}^n, \mathbf{N}^n)$, that is, an input image $\mathbf{I}^n$ with corresponding foreground mask $\mathbf{B}^n$, ground-truth mesh vertices $\mathbf{v}^n$, depth map $\mathbf{D}^n$ and normal map $\mathbf{N}^n$. We define 3 loss functions for the three potential outputs of our network.

To train Ψ_C, we define the loss as the Mean Square Error between the vertex coordinates and the ground truth. That is,

$$\mathcal{L}_C = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{V}\sum_{i=1}^{V}\left\|\mathbf{v}_i^n - \Psi_C(\Lambda(\mathbf{I}_m^n))_i\right\|^2 , \qquad (1)$$

where a subscript i denotes the vertex number.

Since we use training data whose average distance to the camera is roughly constant and focus on recovering local high-frequency deformations, to train Ψ_D, we minimize the loss

$$\mathcal{L}_D = \frac{1}{N}\sum_{n=1}^{N}\frac{\sum_i |\mathbf{D}_i^n - \Psi_D(\Lambda(\mathbf{I}_m^n))_i|\mathbf{B}_i^n}{\sum_i \mathbf{B}_i^n} , \qquad (2)$$

where i denotes the image location. Note that we only take into account pixels within the binary mask $\mathbf{B}$. In other words, we handle the depth ambiguity by recovering depth

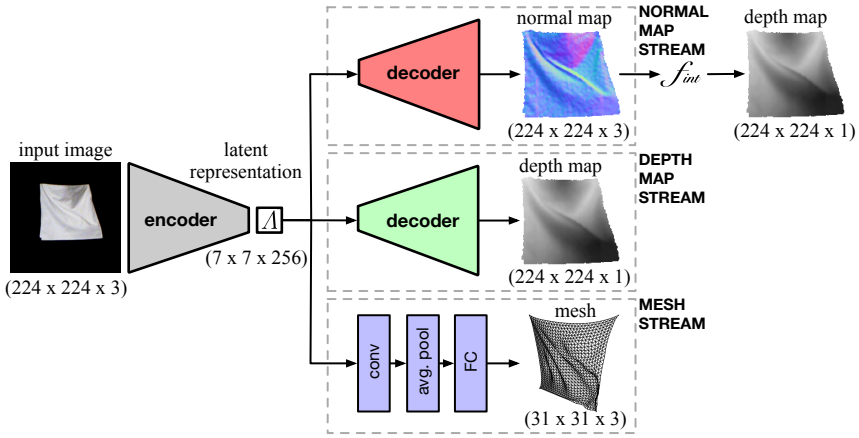


Figure 2. **Surface reconstruction architecture.** Our model is based on SegNet [1], but here we consider multiple output branches. The encoder, in gray, outputs the same latent representation Λ for the normal, depth, and vertex branch. A different decoder, shown in red, green, and blue, is then used for each branch. In other words, this creates three potential streams, which can be either trained individually or jointly. When recovering normals only, an additional integration step is required to compute depths.

variations around the mean depth of our training data, instead of using a scale invariant measure as in [12]. As will be discussed in Section 5, to evaluate accuracy at test time, we rescale the prediction so as to align it with the ground truth.

Similarly, to train Ψ_N, we define a loss $\mathcal{L}_N$ that relies on a linearized version of the cosine similarity [37] and add to it a term that favors unit length vectors. We take it to be

$$\mathcal{L}_N = \frac{1}{N}\sum_{n=1}^{N} \frac{\left[\sum_i \mathbf{B}_i^n \left(\kappa \mathcal{L}_a(\mathbf{N}_i^n, \hat{\mathbf{N}}_i^n) + \mathcal{L}_l(\hat{\mathbf{N}}_i^n)\right)\right]}{\sum_i \mathbf{B}_i^n}\,, \tag{3}$$

with

$$\mathcal{L}_a(\mathbf{N}_i^n, \hat{\mathbf{N}}_i^n) = \arccos\left(\frac{\mathbf{N}_i^n \hat{\mathbf{N}}_i^n}{\|\mathbf{N}_i^n\|\|\hat{\mathbf{N}}_i^n\| + \epsilon}\right)\frac{1}{\pi}\,, \tag{4}$$

$$\mathcal{L}_l(\hat{\mathbf{N}}_i^n) = \left(\left\|\hat{\mathbf{N}}_i^n\right\| - 1\right)^2, \tag{5}$$

where $\hat{\mathbf{N}}^n = \Psi_N(\Lambda(\mathbf{I}_m^n))$ denotes the predicted normal map, ϵ is a small positive constant that prevents divisions by zero and increases numerical stability, and κ sets the relative influence of the two terms in the loss function. In our experiments, we chose $\kappa = 10$.

4. Real-World SfS Dataset

Successful training of most of the Deep Net models depends on access to large training databases. By contrast, the datasets used in the SfS literature remain relatively small. For example, the algorithm of [40] relies on 7 rigid objects under 20 different directional lighting. In [4], an augmented version of the MIT intrinsic image dataset [13] containing 20 rigid objects under various illumination is used while the method of [20] works with a database of 6 human faces. Some authors use only synthetic data [29] while others augment the limited amount of real-data with synthetic data [40, 5].

By contrast and to fully exploit the capacity of our model while avoiding overfitting, we captured a new large dataset of real deforming surfaces. We acquired sequences of RGB images and corresponding depth maps of a rectangular piece of cloth (`cloth`), T-shirt (`tshirt`), sweater (`sweater`), hoody (`hoody`) and crumpled sheet of paper (`paper`) undergoing complex deformations and seen under varying lighting conditions. We chose the cloth for two reasons. First, being a generic piece of cotton fabric makes it universal enough to capture a wide distribution of local deformations and appearance, even if it differs globally from other objects such as garments. Second, its flat rest state makes it easy to represent by a triangular mesh, as is often done in deformable surface reconstruction [30, 6, 23, 26], and to test our mesh-based model. By contrast, the pieces of garment and a sheet of paper were chosen as a more complex real-world object to demonstrate the generality of our approach and the fact that training on `cloth` produces good results on the `tshirt`, `sweater`, `hoody` and `paper`.

The resulting dataset comprises a total of 26500 image-normals-depth triplets, which is much larger than existing real-world SfS datasets, and we make it publicly available[1]. We now describe the acquisition process.

4.1. Acquisition Setup

Fig. 3 illustrates our setup. We placed a given deformable object in front of a dark background and captured synchronized RGB images and depth maps using a

[1] cvlab.epfl.ch/texless-defsurf-data

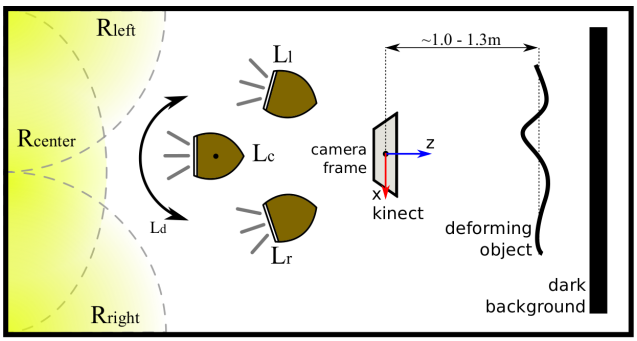
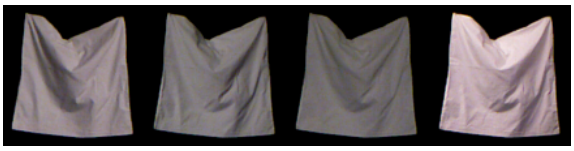


Figure 3. **Data acquisition setup.** The deformable surface is placed between a Microsoft Kinect camera and a dark background. We use three static light sources, L_r, L_l and L_c, pointing towards the back wall and a fourth one, L_d, which can move. The deformable surface is therefore lit by complex indirect lighting.



(a) (b) (c) (d)

Figure 4. **Shading effects on a rectangular piece of cloth.** We show the effects of (a) L_r, (b) L_l, (c) L_c and (d) a randomly chosen frame for L_d.

Microsoft Kinect camera, positioned so that its optical axis is roughly perpendicular to the background plane.

We used three fixed incandescent lamps positioned in the right, left and central area of the room, and a fourth one, that can move. All four were slightly slanted upwards and pointed towards the back of the room, that is, in the direction opposite to the camera's optical axis. As a result, the deforming surface was mostly illuminated by light reflecting off the walls and coming from the *radiance regions* shown in yellow in Fig. 3. This setup simulates directional but diffuse and soft lighting. The range of motion of the dynamically moving light source was chosen so that its radiance regions were slightly larger than those of the other three put together, which should result in a richer light distribution, as shown in Fig. 4. In the remainder of this section, we will consider 4 separate scenarios in which we use each lamp individually and will refer to them as L_r, L_l, L_c, and L_d, where the subscripts r, l, c and d stand for *right*, *left*, *central* and *dynamic*, respectively.

The Microsoft Kinect camera was calibrated, thus yielding known camera intrinsics. Depth maps and corresponding RGB images were aligned using OpenKinect libfreenect library. We recorded sequences at 5 FPS to avoid capturing too many duplicates of nearly identical shapes. Each recorded frame consists of a 640 x 480 RGB image and a 640 x 480 depth map. Because of the deformation undergone by the surfaces, some of the frames feature substantial motion blur, which further increases both realism and the challenge.

Figure 5. **Randomly chosen samples from our dataset.** RGB images together with corresponding GT normals and depth maps are shown for `paper`, `sweater`, `hoody`, `cloth` and `tshirt`.

4.2. Deforming Surfaces

To acquire the images, we pinned the rectangular cloth to a fixed bar along a given edge or corners and manually deformed the rest. In contrast, the T-shirt, the sweater and the hoody were worn by different people making random body motions. The T-Shirt was captured separately from the front and from the back. We were manually deforming the crumpled sheet of paper.

Altogether, this resulted in 18 sequences of 15799 samples for `cloth`, 12 sequences of 6739 samples for `tshirt`, 4 sequences of 2203 samples for `sweater`, 1 sequence of 517 samples for `hoody` and 3 sequences of 1187 samples for `paper`. A representative set of samples from our final dataset is depicted in Fig. 5

4.3. Data Preprocessing

As explained in Section 3, we assume the foreground binary mask $\mathbf{B}$ to be known for each image. To create it, we segmented the imaged objects in the RGB images by simple thresholding followed by hole filling of the biggest connected component. The masks were then used to segment the corresponding depth maps. Furthermore, the RGB images were white balanced using a standard color checker to account for the often unsatisfactory automatic white balance of the Kinect camera. Both the RGB images and depth maps were cropped to the spatial size of 224 x 224, which is

Experiment	Train obj.	Train light.	Test obj.	Test light.
cloth-cloth	cloth	L_r, L_l, L_d	cloth	L_c
tshirt-tshirt	tshirt	L_r, L_c, L_d	tshirt	L_l
cloth-tshirt	cloth	L_r, L_l, L_c, L_d	tshirt	L_r, L_l, L_c, L_d
cloth-sweater	cloth	L_r, L_l, L_c, L_d	sweater	L_r, L_l, L_c, L_d
cloth-hoody	cloth	L_r, L_l, L_c, L_d	hoody	L_l
cloth-paper	cloth	L_r, L_l, L_c, L_d	paper	L_r, L_l, L_c

Table 1. **List of the experiments we conducted.**

the expected input/output size of our model. Since the raw depth maps contain noise and holes, we performed distance based clustering and hole filling by interpolation. The normal maps were computed by differentiating the depth maps in a finite difference sense. Since the Kinect depth measurements are noisy, before computing the normal maps, we smoothed the depth maps by applying a Gaussian blur kernel of size 9×9 with $\mu = 0, \sigma = 3$. As is apparent in Fig. 5, there is still some noise in the normal maps. However, its magnitude is low enough not to significantly corrupt the fine-detailed high-frequency surface geometry, that is, the wrinkles that we are trying to model.

5. Experiments and Results

Recall from Section 3.2 that we can use the architecture depicted by Fig. 2 to recover normals, depths, or vertex coordinates either independently or jointly. In this section, we focus on independent recovery of the three modalities, which we will refer to as **Normals**, **Depth**, and **Coords**, respectively, joint recovery of two modalities, which we will denote **N+C**, **D+C** and **N+D**, respectively, and joint recovery of all three modalities denoted as **N+D+C**, where letters **N**, **D** and **C** denote the use of the normals decoder Ψ_N, depth decoder Ψ_D and/or mesh decoder Ψ_C in the final model. Note that for models predicting multiple modalities, we can evaluate the error for each individual output. We denote this by, e.g., **N+D+C**/X where X can be **N**, **D**, or **C**, depending on whether we evaluate with respect to the normals, depths, or coordinates.

We train our networks using the `cloth` and `tshirt` datasets under the four lighting scenarios introduced in Section 4. Each dataset includes separate training and testing sequences, and we can either train and test on the same object or train on one and test on the other. We can also train with one particular set of lights and test with a different one. In both cases, this allows us to gauge the generalization abilities of our approach.

Table 1 summarizes the experiments we have conducted and whose results we report below. For each one, we randomly select 100 samples from the test sequences and report results on.

5.1. Implementation Details

For all the experiments described in this section, we used the Adam [21] optimizer to train the network. We use a fixed learning rate of 0.001 and parameter κ of the loss function of Eq. 3 set to 10. For **Normals**, **Depth** and **Coords**, we simply let the optimization proceed. For **N+C**, **D+C** and **N+D**, we started by training the model employing only one of the two decoders and we applied an early stopping that halted the training once the validation loss stopped decreasing for 30 consecutive epochs. We then began estimating both decoders' outputs by minimizing the loss function $\mathcal{L}_{\text{joint2}} = \alpha\mathcal{L}_{d1} + \beta\mathcal{L}_{d2}$, where $\mathcal{L}_{d1}$ and $\mathcal{L}_{d2}$ each represent an appropriate loss function as defined in Eqs. 1, 2 and 3 and where we fixed the mixing coefficients $\alpha = 1, \beta = 3$ to promote the training of the yet untrained decoder. As in the single-decoder scenario, we used the early-stopping technique. For **N+D+C**, we similarly added the vertex-wise loss of Eq. 1 and continued training by minimizing the loss function $\mathcal{L}_{\text{joint3}} = \alpha\mathcal{L}_{\mathcal{N}} + \beta\mathcal{L}_{\mathcal{D}} + \gamma\mathcal{L}_{\mathcal{C}}$ with mixing coefficients fixed to $\alpha = 1, \beta = 1, \gamma = 3$ to promote the training of the yet untrained mesh decoder. Our implementation relies on Keras with a Tensorflow backend.

5.2. Metrics

To evaluate the accuracy of the predicted mesh coordinates, we use the mean vertex-wise Euclidean distance, similar to the MSE of Eq. 1 we used to formulate the training loss, but without squaring the distances. We will refer to this metric as m_C.

Since the depth maps we produce are subject to an inherent global scale ambiguity [12], we first align the corresponding point cloud to the ground truth using a Procrustes transformation [36]. More precisely, let $\Theta_{\mathbf{K}}(\mathbf{A})$ be the 3D point cloud associated to depth map $\mathbf{A}$ with corresponding intrinsic matrix $\mathbf{K}$, and let $\Omega(\mathbf{P}, \mathbf{D}_b)$ denote the Procrustes transformation of cloud $\mathbf{P}$ with respect to depth map $\mathbf{D}_b$. Given a set of N ground-truth depth maps $\mathbf{D}^n$ and predicted ones Δ^n, we compute accuracy in terms of the metric

$$m_D = \frac{1}{N}\sum_{n=1}^{N} \frac{\sum_i \|\Theta_{\mathbf{K}}(\mathbf{D}^n)_i - \Omega(\Theta_{\mathbf{K}}(\Delta^n), \mathbf{D}^n)_i\| \, \mathbf{B}_i^n}{\sum_i \mathbf{B}_i^n}, \tag{6}$$

where subscript i indexes the pixels.

Finally, for normal maps, we integrate the normals to recover the corresponding depth and again use the m_D metric. We also report the mean and median angular errors, which we denote as m_{AE} and d_{AE}, respectively, as well as the fractions of normals exhibiting smaller angular error than 10°, 20° and 30°.

5.3. Effectiveness of Meshes or the Lack Thereof

As mentioned in Section 1, our initial intuition was to follow the trend in deformable surface reconstruction and represent the surface as a triangular mesh. Here, we evaluate the results of such a representation, compared to depth

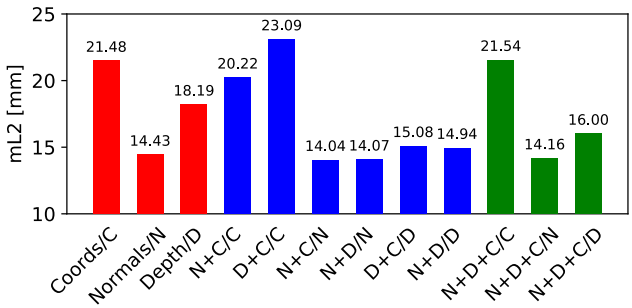


Figure 6. Comparison of models corresponding to all possible combinations of the normal map, depth map and mesh vertices decoders — (red) single decoder models, (blue) two decoders models, (green) three decoders model. The models are trained and tested on `cloth` dataset and all their available inputs are considered for comparison using m_C and m_D metrics.

and normal maps, and show that it is not as effective in our context. This is evidenced by the results in Figure 6, where we compare the models **Coords**, **Normals**, **Depth**, **N+C**, **D+C**, **N+D** and **N+D+C**, which we trained and tested on the `cloth` dataset and evaluated on all outputs available for the given model using metrics m_C for mesh coordinates and m_D for depth maps. The predicted normal maps were converted to depth maps, i.e., we did not use normal maps themselves for this comparison.

We made three key observations: (1) In case of single modality models, i.e., **Coords**, **Normals** and **Depth**, the predicted mesh coordinates **Coords/C** yield by a large margin the highest error of 21.48mm. (2) Training models **Normals** and **Depth** further using an additional modality, i.e., **N+C**, **D+C** or **N+D**, in general helps reducing the error when producing depth or normal maps on the output. However, this is not the case for multi-decoder models producing mesh vertices, where the error can even further increase. (3) We found the model **N+C/N** to achieve overall the lowest error of 14.04mm. However, this is comparable with **N+D/N** and **N+D/D**, which are not dependent on mesh vertices at all.

Given the non-trivial process required to create the GT mesh vertices, as described in the supplementary material, which is more tedious and error-prone than obtaining the depth or normal maps, and considering the negligible performance improvement, we discard the meshes from our approach in the remainder of the paper.

5.4. Separate vs Joint Learning

In Table 2, we report the accuracy of **Normals**, **Depth**, and **N+D**. In the latter case, we can evaluate either the predicted normals or the predicted depth maps, which we denote as **N+D/N** and **N+D/D**, respectively. To evaluate different scenarios we select `cloth` and `tshirt` from our datasets as the categories containing the most samples. We either train and test on the same object (`cloth`), or train on `cloth` and test on `tshirt`.

Experiment	N+D/N	Normals/N	N+D/D	Depth/D
cloth-cloth	**17.53**	17.80	**15.96**	18.18
tshirt-tshirt	16.26	**15.19**	**16.45**	18.01
cloth-tshirt	**26.26**	27.06	**30.23**	32.16

Table 2. Comparison of **Depth**, **Normals** and **N+D** in different scenarios. In general, the normal predictions yield lower m_D error than the predicted depth maps, and joint training outperforms the single-decoder models.

As can be seen in Table 2, using the normal predictions, followed by integration, tends to yield lower errors than the predicted depth maps. More importantly, training jointly on normals and depth performs best overall, which is in keeping with the idea that forcing the network to learn features that disentangle the different contributions helps [34, 22]. Interestingly, training on `cloth` and testing on `tshirt` degrades the accuracy but still yields a competitive result as we will see in the following section.

5.5. Comparing against a State-of-the-Art SfS Approach

Here, we compare our results to those of the SIRFS method [5], which we briefly described in Section 2. This choice was motivated by the fact that SIRFS constitutes a state-of-the-art SfS method whose code is publicly available, unlike that of the other contemporary SfS methods described in that section. Given the input image and segmentation mask, SIRFS performs intrinsic image decomposition into a normal map, depth map, lighting and reflectance. To compare with our method we take SIRFS's normal map and depth map predictions, integrate the normals and align to the ground-truth depth map, as explained in Section 5.2 and we do the same for our own results.

The results of this comparison are summarized in Table 3, where we evaluate several normal-based error metrics and one depth-based metric, the m_D. For the latter, we report the best results of SIRFS obtained either directly from the depth estimates, or by integrating the normal estimates. For our approach, we report the results based on the predicted normals, since we have found that they were slightly better than those obtained from our depth predictions. Note that we outperform SIRFS in all metrics. In the most challenging scenarios, `cloth-tshirt`, `cloth-sweater`, `cloth-hoody` and `cloth-paper`, our method still achieves lower errors, particularly in metrics evaluating normal quality. Figs. 1 and 7 show qualitative results for normal prediction on `tshirt`. Our method clearly outperforms SIRFS when it comes to reconstructing the finer details of local creases. Furthermore, if we train our model on the combined dataset of `cloth` and `tshirt`, the generalization capability for objects of different categories drastically increases as is shown in Table 4.

In Table 5, we compare the run-times of our approach

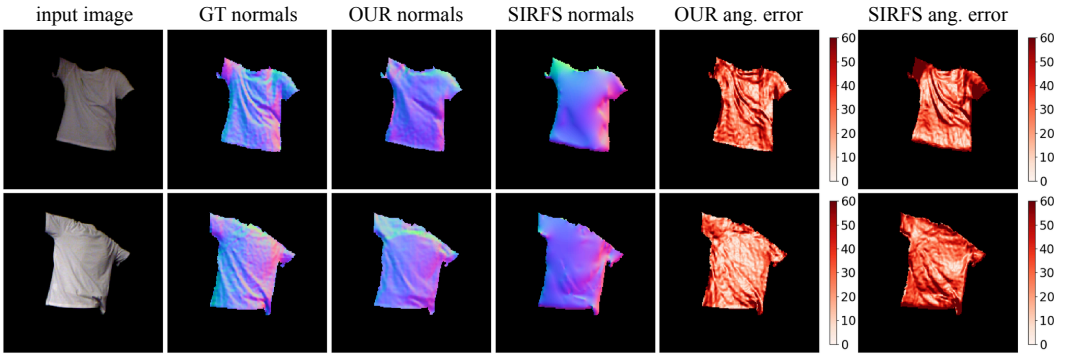


Figure 7. **Qualitative comparison to SIRFS for the `cloth-tshirt` scenario.** Even in this challenging scenario, our method is able to recover finer details than SIRFS.

experiment	method	mAE [°]	dAE [°]	$< 10°$	$< 20°$	$< 30°$	m_D [mm]
cloth-cloth	SIRFS	37.98 ± 23.18	33.52	7.25	24.93	43.96	31.55 ± 10.93
	OURS	$\mathbf{17.37 \pm 12.51}$	**14.44**	**30.6**	**68.85**	**87.29**	$\mathbf{17.53 \pm 5.50}$
tshirt-tshirt	SIRFS	30.17 ± 20.26	25.53	11.78	36.63	59.62	31.09 ± 15.03
	OURS	$\mathbf{18.07 \pm 12.71}$	**15.17**	**28.28**	**66.27**	**85.85**	$\mathbf{17.18 \pm 18.58}$
cloth-tshirt	SIRFS	30.08 ± 19.43	25.93	10.49	35.03	59.15	30.29 ± 10.42
	OURS	$\mathbf{25.74 \pm 15.81}$	**22.81**	**13.45**	**41.98**	**67.7**	$\mathbf{26.26 \pm 7.72}$
cloth-sweater	SIRFS	33.25 ± 21.60	28.11	8.94	30.7	54.02	39.51 ± 14.96
	OURS	$\mathbf{31.52 \pm 19.07}$	**28.06**	**9.25**	**30.97**	**54.25**	$\mathbf{38.93 \pm 10.36}$
cloth-hoody	SIRFS	36.84 ± 23.14	32.11	7.79	26.2	46.11	43.51 ± 13.79
	OURS	$\mathbf{32.54 \pm 21.15}$	**28.02**	**9.88**	**31.78**	**54.05**	$\mathbf{43.22 \pm 24.81}$
cloth-paper	SIRFS	56.69 ± 27.09	59.53	1.71	7.06	15.73	49.35 ± 18.51
	OURS	$\mathbf{35.53 \pm 22.16}$	**31.13**	**8.42**	**27.54**	**47.84**	$\mathbf{24.16 \pm 7.15}$

Table 3. **Comparison with the method of [5].** Our approach outperforms SIRFS in all metrics, even in the challenging `cloth-tshirt`, `cloth-sweater`, `cloth-hoody` and `cloth-paper` scenarios, with a particularly large gap for the first 5 metrics that evaluate the quality of the predicted normals. We report mean values averaged over test sets consisting of 100 samples each with standard deviations for the m_{AE} and m_D metrics.

Test set	mAE [°]	dAE [°]	$< 10°$	$< 20°$	$< 30°$	m_D [mm]
sweater	25.75 ± 16.72	22.18	14.35	43.81	68.45	$\mathbf{28.36 \pm 7.59}$
hoody	24.66 ± 17.36	20.5	17.53	48.6	71.49	$\mathbf{25.40 \pm 5.07}$

Table 4. **Evaluation of our model trained on the combined `cloth` and `tshirt` dataset.** Exposing the model not only to the generic cloth but also to the T-Shirt worn by a person at training time helps the model learn a better shapes distribution which significantly improves the predictions when tested on different garment pieces (compare these results with the corresponding ones in Table 3).

Model	SIRFS	OUR_N	OUR_D	OUR_N+D
t [s]	113.653	**0.01**	**0.01**	0.016

Table 5. **Comparison of the run-times of our approach with those of SIRFS.** We report the average time needed to process one input image of size 224×224 px.

with those of SIRFS. Note that our method performs *orders of magnitude faster*. This is due to the fact that SIRFS relies on a costly optimization, whereas, in our case, all the heavy-lifting was done at training time and inference only requires a feed-forward pass through the network.

6. Conclusion

We have introduced a framework for reconstructing the 3D shape of a texture-less, deformable surface from a single image. To this end, we have followed a data-driven approach, thus essentially learning to perform Shape-from-Shading. Our experiments have demonstrated that, while meshes have proven effective to deal with well-textured deformable surfaces, they are much less well-suited than depth- and normal-based representations for texture-less ones in our setting. Furthermore, our comparison with a state-of-the-art SfS method has shown that our reconstructions were more accurate, particularly in terms of normal quality. This is the case even when training our model on one object and testing it on a different one. We expect that such a generalizability would further increase were we to use larger amounts of training data. Therefore, in the future, we will dedicate time to creating a larger-scale dataset of texture-less, deformable objects.

Acknowledgments This work was supported in part by a Swiss National Foundation for Research grant.

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *arXiv preprint arXiv:1511.00561*, 2015.

[2] T. Bagautdinov, C. Wu, J. Saragih, P. Fua, and Y. Sheikh. Modeling Facial Geometry Using Compositional VAEs. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[3] J. Barron and J. Malik. High-Frequency Shape and Albedo from Shading Using Natural Image Statistics. *Conference on Computer Vision and Pattern Recognition*, pages 2521–2528, 2011.

[4] J. Barron and J. Malik. Shape, Albedo, and Illumination from a Single Image of an Unknown Object. In *Conference on Computer Vision and Pattern Recognition*, June 2012.

[5] J. T. Barron and J. Malik. Shape, Illumination, and Reflectance from Shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(8):1670–1687, 2015.

[6] A. Bartoli, Y. Gérard, F. Chadebecq, and T. Collins. On Template-Based Reconstruction from a Single View: Analytical Solutions and Proofs of Well-Posedness for Developable, Isometric and Conformal Surfaces. In *Conference on Computer Vision and Pattern Recognition*, 2012.

[7] P. Belhumeur, D. Kriegman, and A. Yuille. The Bas-Relief Ambiguity. *International Journal of Computer Vision*, 35(1):33–44, 1999.

[8] G. Choe, S. Narasimhan, and I. S. Kweon. Simultaneous Estimation of Near IR BRDF and Fine-Scale Surface Geometry. In *Conference on Computer Vision and Pattern Recognition*, June 2016.

[9] R. Danerek, E. Dibra, C. öztireli, R. Ziegler, and M. Gross. Deepgarment : 3D Garment Shape Estimation from a Single Image. *Eurographics*, 2017.

[10] J.-D. Durou, M. Falcone, and M. Sagona. Numerical Methods for Shape from Shading: A New Survey with Benchmarks. *Computer Vision and Image Understanding*, 109(1):22–43, 2008.

[11] A. Ecker and A. D. Jepson. Polynomial Shape from Shading. In *Conference on Computer Vision and Pattern Recognition*, pages 145–152, 2010.

[12] D. Eigen, C. Puhrsch, and R. Fergus. Depth Map Prediction from a Single Image Using a Multi-Scale Deep Network. In *Advances in Neural Information Processing Systems*, pages 2366–2374, 2014.

[13] R. Grosse, M. K. Johnson, E. H. Adelson, and W. T. Freeman. Ground Truth Dataset and Baseline Evaluations for Intrinsic Image Algorithms. In *Conference on Computer Vision and Pattern Recognition*, pages 2335–2342, 2009.

[14] T. Hassner and R. Basri. Example Based 3D Reconstruction from Single 2D Images. In *Conference on Computer Vision and Pattern Recognition*, pages 15–15, 2006.

[15] B. Horn and M. Brooks. *Shape from Shading*. MIT Press, 1989.

[16] X. Huang, J. Gao, L. Wang, and R. Yang. Examplar-Based Shape from Shading. In *3DIM*, pages 349–356, 2007.

[17] M. Janner, J. Wu, T. Kulkarni, I. Yildirim, and J. Tenenbaum. Self-Supervised Intrinsic Image Decomposition. In *Advances in Neural Information Processing Systems*, 2017.

[18] M. K. Johnson and E. H. Adelson. Shape Estimation in Natural Illumination. In *Conference on Computer Vision and Pattern Recognition*, pages 2553–2560, 2011.

[19] A. Jourabloo and X. Liu. Large-Pose Face Alignment via Cnn-Based Dense 3D Model Fitting. In *Conference on Computer Vision and Pattern Recognition*, pages 4188–4196, 2016.

[20] N. Khan, L. Tran, and M. Tappen. Training Many-Parameter Shape-from-Shading Models Using a Surface Database. In *International Conference on Computer Vision*, pages 1433–1440, 2009.

[21] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimisation. In *International Conference on Learning Representations*, 2015.

[22] T. D. Kulkarni, W. Whitney, P. Kohli, and J. B. Tenenbaum. Deep Convolutional Inverse Graphics Network. In *arXiv*, 2015.

[23] D. Ngo, J. Ostlund, and P. Fua. Template-Based Monocular 3D Shape Recovery Using Laplacian Meshes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):172–187, 2016.

[24] G. Oxholm and K. Nishino. Shape and Reflectance from Natural Illumination. In *European Conference on Computer Vision*, pages 528–541, 2012.

[25] E. Prados and O. Faugeras. Shape from Shading: A Well-Posed Problem? In *Conference on Computer Vision and Pattern Recognition*, June 2005.

[26] A. Pumarola, A. Agudo, L. Porzi, A. Sanfeliu, V. Lepetit, and F. Moreno-noguer. Geometry-Aware Network for Non-Rigid Shape Prediction from a Single View. In *Conference on Computer Vision and Pattern Recognition*, June 2018.

[27] K. Rematas, T. Ritschel, M. Fritz, E. Gavves, and T. Tuytelaars. Deep Reflectance Maps. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[28] E. Richardson, M. Sela, R. Or-el, and R. Kimmel. Learning Detailed Face Reconstruction from a Single Image. In *Conference on Computer Vision and Pattern Recognition*, pages 5553–5562, 2017.

[29] S. R. Richter and S. Roth. Discriminative Shape from Shading in Uncalibrated Illumination. In *Conference on Computer Vision and Pattern Recognition*, pages 1128–1136, 2015.

[30] M. Salzmann and P. Fua. Linear Local Models for Monocular Reconstruction of Deformable Surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):931–944, 2011.

[31] D. Samaras, D. Metaxas, P. Fua, and Y. Leclerc. Variable Albedo Surface Reconstruction from Stereo and Shape from Shading. In *Conference on Computer Vision and Pattern Recognition*, June 2000.

[32] S. Sengupta, A. Kanazawa, C. D. Castillo, and D. W. Jacobs. Sfsnet: Learning Shape, Refectance and Illuminance of Faces in the Wild. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[33] J. Shi, Y. Dong, H. Su, and S. Yu. Learning Non-Lambertian Object Intrinsics Across Shapenet Categories. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[34] Z. Shu, E. Yumer, S. Hadap, K. Sunkavalli, E. Shechtman, and D. Samaras. Neural Face Editing with Intrinsic Image Disentangling. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[35] L. Sijin, L. Zhi-Qiang, and A. Chan. Heterogeneous Multi-Task Learning for Human Pose Estimation with Deep Convolutional Neural Network. *International Journal of Computer Vision*, pages 19–36, 2015.

[36] M. B. Stegmann and D. D. Gomez. A Brief Introduction to Statistical Shape Analysis. Technical report, University of Denmark, DTU, 2002.

[37] G. Trigeorgis, P. Snape, I. Kokkinos, and S. Zafeiriou. Face Normals "In-the-Wild" Using Fully Convolutional Networks. In *Conference on Computer Vision and Pattern Recognition*, pages 340–349, 2017.

[38] A. Varol, A. Shaji, M. Salzmann, and P. Fua. Monocular 3D Reconstruction of Locally Textured Surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1118–1130, June 2012.

[39] X. Wang, M. Salzmann, F. Wang, and J. Zhao. Template-Free 3D Reconstruction of Poorly-Textured Nonrigid Surfaces. *European Conference on Computer Vision*, 2016.

[40] Y. Xiong, A. Chakrabarti, R. Basri, S. Gortler, D. Jacobs, and T. Zickler. From Shading to Local Shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1):67–79, October 2015.

[41] Y. Yoon, G. Choe, N. Kim, J.-Y. Lee, and I. S. Kweon. Fine-Scale Surface Normal Estimation Using a Single NIR Image. *European Conference on Computer Vision*, 2016.

[42] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape from Shading: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):690–706, 1999.

[43] D. Zoran, D. Krishnan, J. Bento, and B. Freeman. Shape and Illumination from Shading Using the Generic Viewpoint Assumption. In *Advances in Neural Information Processing Systems*, pages 226–234, 2014.