Abstract

Shape-from-Template (SfT) methods estimate 3D surface deformations from a single monocular RGB camera while assuming a 3D state known in advance (a template). This is an important yet challenging problem due to the under-constrained nature of the monocular setting. Existing SfT techniques predominantly use geometric and simplified deformation models, which often limits their reconstruction abilities. In contrast to previous works, this paper proposes a new SfT approach explaining 2D observations through physical simulations accounting for forces and material properties. Our differentiable physics simulator regularises the surface evolution and optimises the material elastic properties such as bending coefficients, stretching stiffness and density. We use a differentiable renderer to minimise the dense reprojection error between the estimated 3D states and the input images and recover the deformation parameters using an adaptive gradient-based optimisation. For the evaluation, we record with an RGB-D camera challenging real surfaces exposed to physical forces with various material properties and textures. Our approach significantly reduces the 3D reconstruction error compared to multiple competing methods. For the source code and data, see https://4dqv.mpi-inf.mpg.de/phi-SfT/.

1. Introduction

Reconstructing general deformable, temporally-coherent surfaces in 3D from monocular RGB videos is a longstanding, challenging and ill-posed problem. It was studied under different assumptions, and methods addressing it can be roughly classified into (template-free) non-rigid structure from motion (NRSfM) [8, 14], (template-based) shape-from-template (SfT) [34, 40], and neural 3D mesh regression [27]. The objective of SfT is: Given a known initial 3D state (a template) of an observed deformable scene or an object, reconstruct all its 3D states observed in the entire image sequence [45]. Recent learning-based SfT methods encode prior knowledge in neural network weights [13, 47]. This offers multiple advantages over a vast body of previous, model-based works [34, 36, 38, 40, 45, 46, 61], such as the ability to handle larger deformations, a broader spectrum of supported types of motions and deformations (including highly nonlinear ones), and real-time operation. One of the pivotal limitations of both classical and neural SfT methods is that they capture general 3D states well but not fine local surface deformations. This is a consequence of the non-awareness of the physical fold formation process attributable to the elastic properties of the materials and forces acting on them. As a result, existing methods can only reconstruct predominantly global deformations.

This paper proposes $\phi$-SfT (from Greek $\phi$υσικη meaning physics): A new analysis-by-synthesis SfT method which addresses several limitations of the current state of the art and improves the accuracy of monocular non-rigid 3D reconstruction by a significant margin; see Fig. 1 for an overview. Our approach explicitly models the physical fold formation process, and its parameters are physically meaningful. $\phi$-SfT does not require training data. We enable gradient-based optimisation by employing two components: A differentiable renderer and a differentiable physics simulator. Our core idea is to use the latter as a regulariser during the optimisation of our objective function. Next, the differentiable renderer ensures that the reprojections of the recovered 3D states accurately match the observed images. In contrast to earlier photometric terms used for SfT [61].
differentiable rendering allows us for the first time to define the reprojection error densely per pixel and not only per vertex. We can thus exploit the information present in the texture regardless of the mesh resolution. $\phi$-SfT is significantly more accurate than related methods and supports finer-scale local folds, which is shown on a wide spectrum of deformations in extensive experiments (Sec. 4). In summary, this paper has the following technical contributions:

- A new optimisation-based SfT approach with a physics-based deformation model that ensures high physical fidelity and realism for the surface evolution, outputting temporally smooth 3D shape sequences that are aware of the forces acting on the object (Sec. 3).
- Differentiable rendering for SfT to encourage the 2D projections of the reconstructed 3D structure to match the observed images. The differentiability allows optimising for the deformation parameters by minimising a dense per-pixel photometric energy (Sec. 3.2).
- A new dataset of real deforming surfaces recorded in a way to facilitate the quantitative evaluation of reconstruction methods against reference depth maps (Sec. 4.1). The dataset contains surfaces of various textures and materials, exposed to different external forces. We release our dataset and source code to encourage future research.

2. Related Work

The methods for monocular non-rigid 3D reconstruction differ in the assumptions they make about the input, available prior knowledge and how they model the deformations. This section reviews methods that can be classified into non-rigid structure from motion (NRSfM), shape from template (SfT) and monocular 3D mesh reconstruction.

NRSfM operates on 2D point tracks over the input monocular views. Earlier NRSfM methods were designed for sparse measurements and modelled deformations with linear subspaces along with various priors [2, 8, 12, 37, 53]. More recent techniques [3, 14, 17, 26] allow reconstructing dense image points observed in a reference frame. They impose constraints on spatial point locations to infer smooth and continuous deforming surfaces. A sparse NRSfM method relying on principles of continuum mechanics represents a deformable object using an estimated (not simulated) elastic model and a low-rank force field acting on it [1]. Even though the force prior has a direct physical interpretation, this model still shares most limitations with other methods. Diff-NRSfM [39] assumes the observed structure preserves its differentiable structure and infinitesimal planarity. This method produces impressive results for smooth surfaces but struggles to reconstruct fine-scale folds, unlike our $\phi$-SfT.

Recently, neural NRSfM approaches both for sparse [35, 55] and dense [44, 50] cases were proposed in the literature. Some of them need to be trained for each object category [35, 44], whereas N-NRSfM [50] and PAUL [55] run on unknown data. Some 2D keypoint lifting approaches for 3D human pose estimation, such as Chen et al. [10], require only 2D data for supervision and share similarities with neural sparse NRSfM.

SfT algorithms operate directly on images and assume a known 3D surface prior as input [34, 40, 45, 61]. These methods minimise the 3D-2D reprojection error and impose geometric constraints such as surface inextensibility [40, 45] or isometry [5, 34, 61]. Recent neural SfT methods [13, 18, 41, 47] predict 3D surfaces from monocular images relying on datasets with different template states. Our $\phi$-SfT contrasts with other SfT methods in that it uses temporal information and a differentiable physics simulator as a regulariser for high-fidelity 3D surface tracking instead of approximating the underlying physical properties via geometric constraints. Moreover, none of these methods uses a per-pixel differentiable photometric loss which ensures that 3D estimates accurately reproject into the 2D images.

Physics-based priors in 3D human performance capture is an emerging field, although there is some early work on it relying on multi-view data [51]. Rempe et al.’s [43] method and PhysCap [48, 49] show that integrating physics laws into an objective for sparse 3D human motion capture improves the accuracy of the 3D estimates. The proposed constraints reduce the artefacts arising from the monocular setting (e.g., unnatural jitter and body leaning, foot sliding and foot-floor penetration). Several methods for 3D human performance capture include clothes deformations, such as Guo et al. [19] and Li et al. [28]. The method of Guo et al. operates on point clouds and optimises the states of the simulated clothes so that they match the inputs. The cloth motion is expressed through a combination of skin friction, gravity and forces attributed to the material (elasticity). Thus, their focus is cloth state recovery from sparse point cloud measurements, which provide a strong 3D shape cue, whereas we assume a single 3D template and operate on monocular videos; this inverse problem is much more ill-posed. Li et al. generate training data with a physics-based simulator on-the-fly and use it to train a neural network for 3D human performance capture, including clothes deformations. Thus, they do not impose hard physics-based constraints as we do with the differentiable physics simulator. Liang et al. [29] use 3D supervision for physics-based cloth simulation. The work by Weiss et al. [58] recovers material parameters of a physics simulator in an analysis-by-synthesis policy to solve an inverse elasticity problem. In contrast to our method, they additionally require depth inputs for a strong 3D cue and do not recover local surface deformations. The idea of combining a differentiable physics and a differentiable graphics engine has been recently explored in contexts different from ours, i.e., estimation of material properties and visuomotor control [21, 23, 31].
Monocular 3D mesh reconstruction approaches can be trained on extensive collections of unstructured views in the desired object category. Some works \cite{6, 11, 57} require 3D supervision, while others, similar to ours, do not: In an early work, Cashman et al. \cite{9} show that almost-rigid object categories, like dolphins, can be reconstructed from image collections. Kanazawa et al. \cite{60} relax the need for input annotations. Li et al. \cite{62} extend \cite{60} to video input and estimate a temporally consistent coarse mesh reconstruction for weakly articulated objects. LASR \cite{60} further relaxes the need for an initial coarse template. We differ from these by the usage of a physics-based deformation model, and we focus on recovering local surface deformations.

3. Our Approach

We propose \( \phi \)-SfT, a new method for the 3D reconstruction of a deforming surface (such as cloth) from a monocular RGB video \( \{I_t\}_{t \in [1, \ldots, T]} \) with known intrinsics. As is common for SfT methods \cite{34, 61}, we assume that the camera is static and take as input a flat rest shape of the target deformable surface \( S_t \) for \( t = 1 \) with a corresponding texture map \( T \). We also assume that a segmentation mask separating a foreground object and background is available. To encourage physically plausible deformations, we use a full physical model, described in Sec. 3.1, that explicitly models forces acting on the surface and the underlying elastic properties of the material. Sec. 3.2 presents the objective function we employ to relate the 2D observations to the estimated reconstructions. We then describe how we optimise the objective function for the physical parameters in Sec. 3.3 and provide implementation details in Sec. 3.4. See Fig. 2 for an overview of \( \phi \)-SfT.

Physics-based simulators are widely used in computer graphics for 3D simulations \cite{4, 33}, and differentiable versions exist \cite{29}. Our idea is to use a differentiable physics simulator as a regulariser (deformation model prior) in monocular non-rigid 3D reconstruction. Its usage in SfT is, unfortunately, not straightforward. To integrate the physics-based simulator into our framework, we have to make several crucial improvements to it. First, the idealised assumptions of simulated environments cannot account for the variety of forces and effects causing surface deformations in the real world (e.g., wind turbulence). We take inspiration from the recent work on physically plausible 3D human motion capture \cite{49}. This approach uses a virtual force acting on the root joint of a human skeleton to account for the effects the physics model does not consider. We, therefore, introduce corrective forces accounting for mismatched assumptions about the natural scene (Sec. 3.1). Second, while most simulators used in computer graphics are designed to create simulations following 3D reference motions, it now has to be driven by the 2D input images, and the gradients need to be backpropagated from the image-based losses. Hence, our energy function includes a dense photometric loss and a silhouette loss (Sec. 3.2). Lastly, the optimisation strategy suitable for 3D simulations is not the best choice for our analysis-by-synthesis \( \phi \)-SfT approach—optimising for deformed surface states, material properties and forces—and we propose an adaptive training strategy instead (Sec. 3.3).

3.1. Deformation Model

We seek to reconstruct a sequence of deforming surfaces as 2D manifold meshes in 3D space with fixed topology (edges \( E \), thereby providing temporal correspondences. A surface in this sequence can be parameterised as a triangular mesh \( S_t = \{V_t, E\} \) where the state of the \( i \)-th vertex in \( V_t \) comprises its 3D position \( x^i_t \in \mathbb{R}^3 \) and its velocity \( v^i_t \in \mathbb{R}^3 \).

Surface Parametrisation At the core of our method, we model deformations of the non-rigid surface as a physical process, i.e., as elastic deformations resulting from internal stretching and bending forces as well as external forces acting on the surface. We thus do not treat the mesh states \( S_t \) as parameters but instead use a physical parametrisation.

We initialise the differentiable cloth simulator from the template \( S_1 \) and generate \( S_t \) with physics simulation \( PS \) according to the material parameters and external forces:

\[
S_t = PS(S_{t-1}; \phi_{t-1}),
\]

where \( \phi_{t-1} \) are the estimated physics parameters, i.e.,

\[
\phi_{t-1} = \{d, S, B, w, F_{t-1}\}.
\]

Here, material density \( d \in \mathbb{R} \), stretching stiffness \( S \in \mathbb{R}^{24} \) (resistance to stretching), and bending stiffness \( B \in \mathbb{R}^{15} \) (resistance to bending and folding) all together describe the elastic properties of the material and, hence, determine the cloth’s internal forces. We also optimise for the wind force \( w \in \mathbb{R}^3 \). However, the wind model is not sufficient to fully describe all the external forces in the scene, such as hand contacts and wind turbulence. We seek to correct for these model insufficiencies by additionally defining a set of corrective forces \( F = \{F_t \in \mathbb{R}^{|V_t| \times 3}\}_{t \in [1, \ldots, T-1]} \). Note that these vary across vertices and across time. They can, in principle, account for any physical force that the simulator does not explicitly model. In the following, we use the shorthand \( \phi = \{d, S, B, w, F\} \).

Physics Simulator For \( PS \), we follow the cloth simulation pipeline introduced by Narain et al. \cite{32, 33, 32}. In the continuous domain, physics-based simulation can be formulated as a time-varying partial differential equation \cite{4}:

\[
\frac{\partial^2 x}{\partial t^2} = M^{-1}f(x, v),
\]

3950
where \((x, v)\) is the vertex state, and \(M\) is a diagonal matrix of the mass distribution derived from the material density \(d\) and surface area. \(f(\cdot)\) are the forces, *i.e.*, internal forces which are a function of cloth elastic properties \(S\) and \(B\) as well as external forces such as wind \(w\) and corrective forces \(F_i\). We follow the elastic model for cloth material of Wang et al. [56] for describing the effects of \(d, S\) and \(B\). This model approximates various nonlinear and anisotropic behaviours seen in cloths when subject to external forces.

In practice, we are given the known position \(x_{t-1}\) and velocity \(v_{t-1}\) of the system at time \(t-1\). Our goal is to determine the new position \(x_t = x_{t-1} + \Delta x\) and velocity \(v_t = v_{t-1} + \Delta v\) at time \(t\) with a time step size \(h=1\). To that end, (3) can be transformed into a first-order differential equation and solved for \(\Delta x\) and \(\Delta v\) with the implicit, backward Euler method [4]:

\[
\begin{pmatrix}
\Delta x \\
\Delta v
\end{pmatrix} = h \begin{pmatrix}
\mathbf{M}^{-1}f(x_t, v_t)
\end{pmatrix},
\]

which is nonlinear due to \(f\). To turn (4) into a linear system, \(f\) can be approximated via linearisation:

\[
f(x_t, v_t) = f_{t-1} + \frac{\partial f}{\partial x} h(v_{t-1} + \Delta v) + \frac{\partial f}{\partial v} \Delta v,
\]

where the Jacobians \(\frac{\partial f}{\partial x}\) and \(\frac{\partial f}{\partial v}\) are evaluated at \(f_{t-1}\). Thus, a simple cloth simulation process involves solving for \(v_t\) using (4) and (5), and computing the subsequent simulation state as \(x_t = x_{t-1} + h v_t\).

There can additionally be self-collisions and collisions with a dynamic obstacle mesh \((x_t^{\text{obs}}, v_t^{\text{obs}})\) during simulation. Harmon et al. [20] determine the collision response at the impact zones to update the vertex positions accordingly:

\[
x_t = x_t + \text{collision response}(x_t, v_t, x_t^{\text{obs}}, v_t^{\text{obs}}).
\]

Since we want to use end-to-end gradient-based optimisation, we need to backpropagate gradients through this extra step. However, due to the high dimensionality of the dynamical system when modelling cloth, a naïve gradient computation for the general system (4) and (6) (the collision response) can become impractical. Liang et al. [29] propose a solution for this problem, and we proceed with their approach. Specifically, they use implicit differentiation for (4) and (6), where the gradient of the latter is approximated via QR decomposition of a much smaller constraint matrix. For more details on the backward pass, please refer to [29]. We next describe the objective function we use to optimise for the parameters \(\phi\) of the differentiable simulator.

### 3.2. Our Objective Function

We now have a physical deformation model that is parametrised by \(\phi\), and that outputs a 3D mesh \(S_t\) for time
t. We solve for the optimal parameters \( \phi^* \) by minimising the objective function \( E = E_p + \lambda E_s \) (with \( \lambda \in \mathbb{R} \)):

\[
\phi^* = \arg\min_{\phi} E(\phi). \tag{7}
\]

Since we are only given RGB images for time \( t>1 \), we define a photometric energy term \( E_p \) in the image space. Specifically, \( E_p \) is an \( \ell_1 \) RGB data term to encourage photometric consistency between the reconstructed surface rendered into 2D and the input frames:

\[
E_p = \sum_{t=2}^{T} ||R(S_t, T) - I_t||_1, \tag{8}
\]

where \( R(\cdot) \) is a differentiable renderer outputting a perspective projection of the input mesh (textured with \( T \)) onto the image plane with known intrinsics. We implement \( R \) with Soft Rasterizer [30], which introduces useful gradients by composing probability maps of rendered triangles into the final image. This allows us 1) to define \( E_p \) densely over all pixels, instead of just the vertices; 2) use the information in the high-resolution \( T \) that would have been ignored if we had used a photometric term only at the vertices.

While the photometric energy term works well for local corrections, it does not provide a signal for mismatches that are farther apart in the image space. To get a signal even for larger, coarser errors, we add a silhouette energy term that encourages consistency between the foreground segmentation mask of the input frames and the rendered 2D surface:

\[
E_s = \sum_{t=2}^{T} ||G(B_t; \sigma) - G(\tilde{B}_t; \sigma)||_1. \tag{9}
\]

Here, \( B \) and \( \tilde{B} \) are the foreground binary segmentation masks of the reconstructed and the input images, respectively. \( G(\cdot; \sigma) \) represents a Gaussian filter of standard deviation \( \sigma \). The Gaussian filter smooths the binary masks, extending the spatial area where informative gradients are obtainable. Without it, non-zero gradients would be obtained only at pixels located immediately next to the silhouettes of both binary masks. Thus, if the silhouettes do not match almost perfectly at a pixel, the gradient would be zero there, providing no signal to the network as to the target direction to move the mesh’s triangles. Importantly, both the ground-truth mask \( B \) and the rendered mask \( \tilde{B} \) are processed in the same way, ensuring that \( E_s \) is well-behaved.

Given our model and the objective function, we next look at how we solve the optimisation problem.

### 3.3. Optimisation

Our goal is to obtain the optimal physical parameters \( \phi^* \) via (7). We use iterative, gradient-based optimisation to that end. Since both the simulator \( PS \) and the renderer \( R \) are differentiable, we can back-propagate gradients from the objective function \( E \) through the rendering to \( S_t \) and from there further through the physics simulation to the physical parameters \( \phi \) and further to all earlier meshes (see Fig. 2).

#### Initialisation

To obtain a sufficiently accurate initial guess for the elastic properties \( d, S, \) and \( B \), we set them to the average values of ten different real fabrics [56]. The wind and corrective forces \( F \) are initialised to 0, i.e., a zero vector. Note that this initialisation leads \( PS \) to initially generate surfaces \( \{S_t\}_t \) that are identical to the template \( S_1 \).

#### Adaptive Optimisation Scheme

Since simulation is a temporal process, \( \phi \) for the early frames directly influence the reconstructions of the later frames. In addition, later frames are initially reconstructed at lower fidelity than earlier ones. Therefore, similar to tracking a surface, we exploit the temporal frame order and do not optimise for all \( t \geq 1 \) from the start. Instead, we adaptively grow the active temporal window, i.e., the latest time \( t_a \) up to which all earlier frames \( t \leq t_a \) participate in the optimisation. We start with the first five frames as active and optimise \( E \) (7) for them. Once the energy of the latest frame decreases below a threshold \( b \), we add the next frame to the optimisation; see Fig. 3. \( b \) is set to the energy that the fifth frame has when the sixth frame is added. We assume the fifth frame to be well-reconstructed since it is early in the sequence. (In case the optimisation cannot reach the threshold, we set a maximum number of iterations, after which we progress regardless.) This adaptive scheme speeds up optimisation by converging to a reasonable guess for \( \phi \) before later frames become active. Moreover, it allocates more iterations to frames with more challenging deformations. We evaluate the adaptive optimisation scheme experimentally in Sec. 4.3.

### 3.4. Implementation Details

We implement \( \phi \)-SIFT in Pytorch [42]. (7) is solved with the Adam optimiser [25] with learning rate \( 10^{-3} \). The adaptive optimisation scheme leads to several hundred iterations in most cases, which takes 16–24 hours on an Nvidia RTX 8000 GPU. Due to the sequential nature of the simulator, we compute the objective function for all active times \( t \) for each optimisation iteration. We set \( \sigma=7px, \lambda=0.5 \) for \( E_s \) and apply the corrective forces \( F \) by modifying the velocity of vertex \( i \) at time \( t \): \( v_i^t = v_i^t + F_i^t \) (because both mass and time steps are constant). We keep the wind air density fixed at 1kg/m³ and optimise only for the wind velocity.
The images in our real scenes have resolution 1920×1080 pixels. We also pre-process the real scenes (recorded with an RGB-D camera): We first segment out the background from the captured images and point clouds by depth thresholding. We next use Poisson surface reconstruction [24] on the template (at $t = 1$), which yields $\sim 300$ vertices on average. We then determine the initial rigid pose relative to a flat sheet (which is required by the simulator) by iterative closest point (ICP) [7], and initialise the simulator with it. We obtain the texture map $\mathbf{T}$ by projecting the vertices of the template mesh $\mathbf{S}_t$ onto the image space of the first image $\mathbf{I}_1$ with known camera intrinsics.

4. Experimental Results

We evaluate our technique on real and synthetic data. We recorded the $\phi$-SfT real dataset (Sec. 4.1) of natural sequences using a monocular RGB camera and the depth camera Azure Kinect. The latter is used to obtain pseudo-ground-truth segmentations and deformations. Qualitative and quantitative results on this dataset in Sec. 4.2 show our technique clearly outperforms the state of the art by capturing a wider variety of deformations and local folds. In addition, we generate a new $\phi$-SfT synthetic dataset of four monocular RGB sequences of naturally deformable surfaces using physics-based simulation; see our supplement for further details and evaluations, likewise demonstrating the superior performance of $\phi$-SfT. We also perform an ablation study in Sec. 4.3 that demonstrates the importance of corrective forces and other design choices.

4.1. The $\phi$-SfT Real Dataset

We have recorded a new dataset of deformable surfaces to allow quantitative evaluations of reconstruction methods on real data against pseudo ground truth. The dataset includes nine sequences of various surface shapes and textures, including differing material properties due to differences in the cloths’ fabric and weaving (e.g., there are more and less elastic and more and less dense materials). The texture pattern varies from fine-grained and regular to more global and irregular patterns; the cloth size ranges from $55 \times 55 \text{cm}$ to $95 \times 95 \text{cm}$. The surfaces are exposed to external forces, i.e., gravity, wind, and hand contacts. Each sequence is simultaneously recorded using a monocular RGB and depth camera and has a length of about 40 frames, such that they focus on challenging folds. To obtain the texture map $\mathbf{T}$ under the same lighting conditions as the recorded sequence, we start with a flat rest state for $t = 1$ and obtain $\mathbf{T}$ from it. Then, pseudo-ground-truth deformations for each frame are reconstructed as point clouds through backprojection, using the depth images and known camera intrinsics. The point clouds are provided in absolute distance units (meters). Fig. 4 shows an overview of the recorded sequences.

Figure 4. We record a new $\phi$-SfT real dataset of nine sequences with reference depth data to facilitate quantitative comparisons of monocular 3D surface reconstruction methods. Our setup consists of a synchronised RGB camera and an Azure Kinect. The depth camera provides depth maps serving as pseudo ground truth.

4.2. Comparison with Existing Methods

We compare our technique to SfT methods, namely Yu et al.’s Direct, Dense, Deformable (DDD) [61], Ngo et al.’s Ngo2015 [34] and Shimada et al.’s IsMo-GAN [47], as well as Sidhu et al.’s Neural NRSfM (N-NRSfM) [50] and Parashar et al.’s Diff-NRSfM [39].

Since NRSfM methods accept 2D point correspondences, we track 2D points densely across the input images with multi-frame subspace flow (MFSF) [15, 16], as suggested in [50]. The first frame of the sequence is selected as a keyframe for tracking. We provide DDD with the required hierarchy of coarse-to-fine templates and Ngo2015 with the same template as ours. To demonstrate the necessity for the physical simulation based on the internal stretching and bending forces, we show results of the “Only $F$” baseline where the only forces are the correctives $\{F_i\}_i$. As other forces and parameters are omitted, we implement this baseline as optimisation of per-vertex offsets over time.

Due to the monocular scale and depth ambiguities, we align reconstructions of all methods to the ground truth in a rigid-body fashion. For the first frame, we determine the transformation matrix using Procrustes alignment [54], which is further refined with rigid ICP [7] for later frames.

For quantitative evaluation, we compute the Chamfer distance between the pseudo-ground-truth point cloud from Kinect $G = \{g_i \in \mathbb{R}^3\}_i$ and points $M = \{m_j \in \mathbb{R}^3\}_j$ sampled from the reconstructed mesh:

$$Ch_{G,M} = \frac{1}{|G|} \sum_{g \in G} \min |m \in M| ||g - m||^2 + \frac{1}{|M|} \sum_{m \in M} \min |g \in G| ||m - g||^2.$$  \hspace{1cm} (10)

We report average Chamfer distance per sequence $\bar{Ch}_{G,M}$. Figs. 1 and 6 show that $\phi$-SfT substantially outperforms all tested methods qualitatively; see the supplement for more visualisations of our results. Tab. 1 shows that $\bar{Ch}_{G,M}$ of $\phi$-SfT is on average lower than $\bar{Ch}_{G,M}$ of competing methods.
Our results confirm that SfT and NRSfM, both relying on simple geometric prior assumptions (such as surface smoothness, isometry or small local deformations), cannot cope with the elaborate fold patterns present in our dataset. The results of N-NRSfM follow the silhouettes of the input images better than DDD and IsMo-GAN, thanks to 2D point tracking, even though its $Ch_{G,M}$ is the highest (due to rigidity assumption for the initialisation [50, 52]). Its 3D surfaces are somewhat rugged, and the surface pattern allows to recognise the observed texture in the second row. Moreover, as expected, more fine-grained textures result in more accurate 2D point tracking by MFSF. DDD does not track large deformations and silhouettes well in our tests. IsMo-GAN, trained on rather smooth surfaces, cannot reproduce local folds and barely captures the contours. Diff-NRSfM produces reasonable reconstructions in smooth regions owing to its differentiable structure-preserving formulation. However, it is sensitive to noise in 2D measurements, which leads to visual artefacts in the regions with challenging folds. Ngo2015 failed on three scenes (partially on S7 and entirely on S8 and S9). Ngo2015 mostly produces physically implausible results but achieves lower $Ch_{G,M}$ on a few scenes; qualitative observations, however, remain the same for all sequences (Figs. 1 and 6). This suggests that an isometry prior [34] is less effective compared to our physics-based elastic model, which can even express non-isometric deformations (depending on the parameters). When removing all forces from the model except for the corrective, the results degrade in quality and average error increases $>50\%$, see “Only F” in Fig. 6 and Tab. 1. This experiment demonstrates that the accuracy of the full $\phi$-SfT model does not rely solely on the corrective forces. In contrast, $\phi$-SfT estimates temporally coherent surfaces and captures all significant folds while missing only small nuances. As seen in Fig. 1, our physics-based model provides a reasonable prior for self-occluded surface parts. Moreover, our method is not sensitive to initialisation. We empirically find no issue with always initialising with our default material parameters.

See Fig. 5 for depth maps reconstructed by $\phi$-SfT. This alternative way to visualise the results allows us to study and perceive even smaller surface details. We also strongly encourage the readers to watch our supplementary video.

4.3. Ablative Study

We conduct an ablative study on the various design choices to integrate the physics simulator into the SfT setting. We test the following modes: 1) Operation without corrective forces $F_s$, 2) Influence of the adaptive training by considering all frames from the start (Sec. 3.3), and 3) Disabling the silhouette term (9). On average across all sequences, we obtain $Ch_{G,M} = 10.69, 6.21, 5.33,$ and $4.48$ for w/o $F_s$, w/o adaptive, w/o $E_s$, and the full model, respectively. Omitting $F_s$ always leads to a significant error increase, and abandoning our adaptive training policy increases it by $39\%$. Fig. 7 shows qualitative results confirming the statistics over all sequences, i.e., the largest errors are present in the colour-coded error maps when $F_s$ is disabled. The second most crucial component of $\phi$-SfT is the adaptive training strategy. Note that $E_s$ helps when the structure deforms and significantly changes its shape in the re-projection. See our supplement for the complete report.

4.4. Discussion and Future Directions

We use PyTorch3D [42] for rendering; it only supports a single light source, which can differ from the illumination during the recording. There is, therefore, a systematic mismatch between the rendered reconstructions during training and the input images. Still, $\phi$-SfT achieves high accuracy, as our differentiable rendering loss is empirically robust to small noise and mismatches in colour and shading. The runtime of our approach is comparably high (i.e., twice as long as N-NRSfM [50]). This is due to single-threading...
5. Conclusion

We introduced φ-SfT, a new optimisation-based SfT method that models deformations with a physical simulator and uses differentiable rendering to define a reprojection energy term over all pixels, exploiting texture information independent of the mesh resolution. Experiments on the new dataset demonstrate that our approach improves the reconstructions qualitatively and quantitatively by a significant margin over competing techniques of several method classes. Especially remarkable is φ-SfT’s accuracy in folded surface regions. We believe that the proposed technique has a high potential for future research, and we hope to see more improvements on physically principled methods for monocular non-rigid 3D reconstruction.

Acknowledgements. We thank Shaifali Parashar for her help with the source code of Diff-NRSfM [39]. This work was supported by the ERC Consolidator Grant 4DRepLy (770784).
References

[27] Xuefeng Li, Sifei Liu, Shalini De Mello, Kihwan Kim, Xiaolong Wang, Ming-Hsuan Yang, and Jan Kautz. Online adaptation for consistent mesh reconstruction in the wild. In Advances in Neural Information Processing Systems (NeurIPS), 2020. 1, 3
Albert Pumarola, Antonio Agudo, Lorenzo Porzi, Alberto Mathieu Perriollat, Richard Hartley, and Adrien Bartoli.

Nikhila Ravi, Jeremy Reizenstein, David Novotny, Tayshaifali Parashar, Mathieu Salzmann, and Pascal Fua.

Marco Paladini, Alessio Del Bue, Marko Stošić, Marija Shaifali Parashar, Daniel Pizarro, Adrien Bartoli, and Jonas ¨Ostlund, Aydin Varol, Dat Tien Ngo, and Pascal Fua.


Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh


