Neural Cloth Simulation
HUGO BERTICHE, MEYSAM MADADI, and SERGIO ESCALERA, Universitat de Barcelona, Spain and Computer Vision Center, UAB, Spain

Fig. 1. We present a general framework for the garment animation problem through neural cloth simulation. More specifically, an unsupervised deep learning methodology inspired in physically based simulation. Ours is the first methodology able to learn cloth dynamics without any ground truth data.

We present a general framework for the garment animation problem through unsupervised deep learning inspired in physically based simulation. Existing trends in the literature already explore this possibility. Nonetheless, these approaches do not handle cloth dynamics. Here, we propose the first methodology able to learn realistic cloth dynamics unsupervisedly, and henceforth, a general formulation for neural cloth simulation. The key to achieve this is to adapt an existing optimization scheme for motion from simulation based methodologies to deep learning. Then, analyzing the nature of the problem, we devise an architecture able to automatically disentangle static and dynamic cloth subspaces by design. We will show how this improves model performance. Additionally, this opens the possibility of a novel motion augmentation technique that greatly improves generalization. Finally, we show it also allows to control the level of motion in the predictions. This is a useful, never seen before, tool for artists. We provide of detailed analysis of the problem to establish the bases of neural cloth simulation and guide future research into the specifics of this domain.

CCS Concepts: • Computing methodologies → Neural networks; Unsupervised learning; Physical simulation.

Additional Key Words and Phrases: cloth, simulation, dynamics, neural network, deep learning, unsupervised, disentangle

ACM Reference Format:
Hugo Bertiche, Meysam Madadi, and Sergio Escalera. 2022. Neural Cloth Simulation. ACM Trans. Graph. 41, 6, Article 220 (December 2022), 14 pages. https://doi.org/10.1145/3550454.3555491

1 INTRODUCTION

Cloth animation has been a focus of research during decades. Mainly due to its numerous applications in the entertainment and fashion industry. Firstly, computer graphics approaches relied on physically based simulation to animate cloth [Baraff and Witkin 1998; Hahn et al. 2014; Liu et al. 2013; Macklin et al. 2016; Müller et al. 2007; Narain et al. 2012; Pfaff et al. 2014]. While it is possible to obtain physically accurate results, these methodologies are computationally expensive, which makes them unsuitable for scenarios where real-time is a requirement, such as video-games or VR/AR. The research community, inspired by the success of deep learning in...
other 3D tasks [Arsalan Soltani et al. 2017; Han et al. 2017; Madadi et al. 2020; Omran et al. 2018; Qi et al. 2017; Richardson et al. 2016; Socher et al. 2012] and its fast inference properties, has recently shown interest in neural networks as a suitable alternative for fast garment animation.

Initially, authors proposed supervised solutions [Bertiche et al. 2020, 2021b; Gundogdu et al. 2019; Patel et al. 2020; Wang et al. 2019]. To simplify the problem, garments are usually skinned w.r.t. the underlying skeleton that drives the body motion. Then, the network task is to predict cloth deformations in rest pose. These approaches present some drawbacks. Supervised learning requires huge volumes of computationally expensive data. Moreover, this process has to be repeated for each garment and body. Also, more often than not, data requires heavy pre-processing to be ready for training. Finally, predictions usually present body penetrations, which motivates the use of post-processing or strong regularization terms. Authors of [Bertiche et al. 2021a; Santesteban et al. 2022] identified these drawbacks and proposed unsupervised learning schemes. While these approaches addressed some of the drawbacks of supervised methodologies, they do not handle cloth dynamics. On one hand, PBNS [Bertiche et al. 2021a] uses a static formulation. Then, it is unable to learn cloth dynamics. On the other hand, the authors of SNUG [Santesteban et al. 2022] propose to use an inertia term from the computer graphics literature. Nonetheless, their adaptation of the inertia term to deep learning temporally smooths cloth particle velocities (minimizes accelerations). We will see this is not the same as learning cloth dynamics (Sec. 4.5).

We present the first methodology able to learn real cloth dynamics without the need of ground truth data. By doing so, we define the first general framework for neural garment simulation. The list of our contributions is as follows:

1. **Unsupervised Cloth Dynamics.** In the computer graphics literature we can find simulation based works that recast the equations of motion as an optimization problem. This means that a similar solution can be applied to deep learning. We adapt this solution to unsupervised training of neural networks. We will show how our methodology is the first to be able to learn cloth dynamics in an unsupervised fashion.

2. **Disentangled Cloth Subspace.** We analyze the nature of the garment animation problem to motivate a novel architecture that allows an automatic disentanglement of cloth static and dynamic deformations at a subspace level. We will see how this improves model performance as well as allowing control over cloth dynamics. Additionally, we leverage the disentanglement to propose a novel motion augmentation technique that further improves model generalization.

3. **In-depth Analysis on Neural Garment Simulation.** Unsupervised garment animation differs from other deep learning tasks, supervised or unsupervised. We provide a detailed analysis on the problem to understand its peculiarities and help to establish the bases of neural simulation for garments.

The rest of the paper is as follows. In Sec. 2 we review the literature on cloth simulation and deep learning based methodologies on garments. Next, Sec. 3 describes the methodology we propose. Then, Sec. 4 contains an analysis on the different metrics, an ablation study and a comparison with the state-of-the-art. Finally, in Sec. 5 we discuss limitations and future research.

2 RELATED WORK

**Cloth simulation.** Computer graphics has been tackling the cloth animation problem for decades. The first advances in the field were done by [Weil 1986], as static geometry based models. Later, researchers developed elastic continuum models for cloth [Baraff and Witkin 1998; Carignan et al. 1992; Feynman 1986; Terzopoulos et al. 1987] that permit dynamic simulations. On the other hand, other authors [Breen et al. 1992; Haumann 1987; Provot et al. 1995] noted cloth is not a continuum, but a combination of mechanical interactions between cloth yarns. From here, alternative particle based formulations for cloth were developed, like the mass-spring model. Later, the work of [Baraff and Witkin 1998] presented triangle-based formulation for cloth that allowed fast simulation of complex garments. To this day, this work is still the foundation of many current methodologies for cloth simulation [Narain et al. 2012; Pfaff et al. 2014]. Later, [Kaldor et al. 2008, 2010] proposed modelling cloth at yarn-level to achieve highly realistic behaviour. These methodologies -while accurate- are computationally expensive, therefore, not suitable for many applications that demand real-time performance. Simpler and more efficient formulations have been developed in favor of a faster simulation [Liu et al. 2013; Macklin et al. 2016; Müller et al. 2007] at the cost of accuracy or realism. Nonetheless, realistic simulation of fine cloth dynamics in real-time is still unfeasible with standard simulation. Specially as the number of cloth triangles increase while also considering that some of these real-time applications often require computational resources for other tasks. Subspace physics has proved a valid fast alternative for soft body simulation [Pan et al. 2015; Teng et al. 2014], and recently in combination with learnt representations [Fulton et al. 2019; Shen et al. 2021]. This alternative has also been proposed for clothing [De Aguiar et al. 2010; Hahn et al. 2014; Kim et al. 2013], although in practice, collisions are not properly handled. Then, while computer graphics offers realistic and accurate solutions, efficient real-time cloth animation remains an open challenge.

**Deep learning.** During recent years, neural networks have proved their usefulness in many complex tasks. One of their main advantages is a fast inference time. Then, given their success in challenging 3D problems [Arsalan Soltani et al. 2017; Han et al. 2017; Madadi et al. 2020; Omran et al. 2018; Qi et al. 2017; Richardson et al. 2016; Socher et al. 2012], researchers have already turned to deep-based solutions for garment animation. Most of the current literature on the domain relies on supervised learning [Bertiche et al. 2020, 2021b; Gundogdu et al. 2019; Patel et al. 2020; Pfaff et al. 2020; Santesteban et al. 2019, 2021; Wang et al. 2019; Zhang et al. 2021]. To this end, it is necessary to run hundreds or thousands of offline physics based simulations to gather the data required for training. Data gathering needs to be repeated for every garment, body and fabric parameters. This hurts the scalability of supervised solutions. Additionally, supervised learning is biased towards lower frequencies, yielding overly smooth garments. Moreover, supervision does not guarantee physical constraints are satisfied. Finally, simulators display a chaotic behaviour. Garment simulation on top of very similar body
motions may result in considerably different outputs. In practice, this means noisy data, which hinders training. Authors of [Bertiche et al. 2021a] proposed PBNS, an alternative unsupervised solution that does not suffer from the drawbacks related to simulated data. To do so, they propose formulating physically based constraints as energy losses. This permits learning garment deformations without ground truth data. Nonetheless, their formulation is purely static. They do not handle cloth dynamics. Similarly, SNUG [Santesteban et al. 2022] uses the same unsupervised scheme as PBNS to learn garment deformations. With the only addition of an inertia loss term from the physics based simulation literature [Gast et al. 2015; Liu et al. 2013; Martin et al. 2011] to model dynamic deformations. However, in this work we will see how their adaptation of the inertia term is not modelling true cloth dynamics. Following this trend of unsupervised learning for garment animation, we present the first work able to learn cloth dynamics. Thus, defining the first general framework for neural cloth simulation. We also propose a novel model architecture that automatically disentangles static and dynamic cloth deformations. This, in turn, shows improved performance and interesting novel properties.

3 METHODOLOGY

The goal of this work is to define a deep-learning based methodology for the garment animation problem. This problem corresponds to the animation of cloth draped around skinned 3D body models. Following the current trend [Bertiche et al. 2021a; Santesteban et al. 2022], we propose an unsupervised training inspired on physically based simulation. We additionally achieve a disentanglement of static and dynamic cloth deformations by considering the nature of both cases in our model design.

3.1 Neural Cloth Subspace Solver

Defining the problem. Our methodology for learning cloth dynamics unsupervisedly is inspired in classical computer graphics physical simulation [Baraff and Witkin 1998; Liu et al. 2013; Müller et al. 2007]. To this end, cloth is presented as a particle system \( x \in \mathbb{R}^{N \times 3} \). Cloth solvers compute the cloth configuration at instant \( t \) from the previous cloth state, which is defined by the particle locations and velocities at \( t-1 \). Additionally, the solver must also consider external forces –colliders, wind, etc– that act on the cloth. Within the scope of this work –garment animation– the external forces correspond to gravity, which is constant, and the collisions with the underlying skinned 3D model draped with the clothes. Then, given a skinned 3D model parameterized by \( \theta \) (pose and location in our experiments, but potentially any parameterization, like body shape [Loper et al. 2015]), the solver can be written as:

\[
x_t = f(\theta_t, x_{t-1}, v_{t-1}).
\]

where \( v \) is the particle velocities. Velocities will depend on the current and previous particle locations, therefore, we can rewrite the expression as \( x_t = f(\theta_t, x_{t-1}, x_{t-2}). \) Likewise, we have that \( x_{t-1} = f(\theta_{t-1}, x_{t-2}, x_{t-3}). \) The same could be done for \( x_{t-2}, x_{t-3} \) and so on and so forth. This yields the following:

\[
x_t = \hat{f}(\theta_t, \theta_{t-1}, \theta_{t-2}, ..., \theta_0, x_0).
\]

Thus, assuming the 3D body model parameterized by \( \theta \) and the gravity are the only external forces, the cloth can be fully parameterized by the body pose history –or motion– \( \Theta_t = (\theta_t, \theta_{t-1}, \theta_{t-2}, ..., \theta_0) \).

We can intuitively assume that early poses will have less impact on the current cloth state. This allows to safely discard poses outside a given temporal window. To ensure the problem remains well-posed, the temporal window size should be sufficiently large. Garments that may show more complex, longer dynamics –like dresses or skirts– will require a larger temporal window size than garments that will not show complex dynamics –like tight pants. We also identify separately the static case, a special case of garment animation in which there is no body or cloth motion. This corresponds to the result of draping a body staying still in a given pose during an infinite –long enough– amount of time. For such case, we have \( x_t = x_{t-1} = ... = x_0 \) and \( \theta_t = \theta_{t-1} = ... = \theta_0 \), hence, the cloth can –and must– be fully parameterized using only \( \theta_t \).

Cloth subspace. Once we have been able to establish a relationship between the body motion space and cloth space, we are implicitly declaring that a cloth subspace \( Z \subset \mathbb{R}^d \) exists, with \( d << N \times 3 \).

That is, \( \Theta_t \rightarrow z_t \rightarrow x_t \), with \( z_t \in Z \). Then, inspired on subspace physics [De Aguiar et al. 2010; Hahn et al. 2014; Kim et al. 2013], our proposed model must be able to solve the next cloth configuration from the current subspace encoding:

\[
z_{t+1} = g(\Theta_{t+1}, z_t),
\]

where \( g(\cdot) \) corresponds to a neural cloth subspace solver. Additionally, it must be designed in a way that ensures that given a static sample the subspace encoding does not change, \( z_t = z_{t-1} = ... = z_0 \).

The optimal solution must naturally fall back to a static formulation when there is no input body motion. We know the static solution is a special case of the general problem, likewise, the subspace of all static cloth states \( z^S \subset Z \) is a subspace of the subspace of all possible cloth states, \( Z \subset Z \), in the same way that the body pose space is a subspace of the body motion space. That is:

\[
\Theta_t \rightarrow z^S_t, \quad \Theta_t \rightarrow x_t.
\]

We know the pose space is a single continuous manifold, henceforth, due to eq. 4, the static cloth subspace must also be a single continuous manifold. Similarly, we can extend this reasoning to the motion space and the full cloth subspace. We can then conclude that subspace \( Z \) is a higher-dimensional manifold around the static subspace \( Z^S \).

Neural network. Motivated by all of the above, we propose an encoder-decoder recurrent neural network as a suitable architecture for this problem. The model takes body motion \( \Theta_t \) as input, encodes it as \( z_t \) and finally decodes it into the predicted cloth \( x_t \). Also, we present a novel disentangled encoder that will enforce by design the expected static and dynamic behaviour. We will show how this results in an improved performance and explainability. Moreover, by disentangling static and dynamic deformations, we allow control over the level of motion in our predictions. This property will help artists to achieve the desired looks for a given application. We also define a novel motion augmentation technique that greatly increases model robustness. As it is usual in the literature [Bertiche et al. 2020, 2021a,b; Gundogdu et al. 2019; Patel et al. 2020; Santesteban et al.
the network predicts cloth deformations in rest pose. Then, the garment is skinned w.r.t. the underlying body skeleton and it is posed along with the body mesh.

3.2 Body Motion Descriptors
As explained, to fully parameterize the garment state \( x_t \), we need the body pose history, to which we refer as body motion \( \Theta_t \). To keep the problem tractable, we safely truncate the pose history using a reasonable temporal window size. The window size will be directly related to the maximum cloth motion length that our network will be able to learn. Looser garments –like skirts– require longer temporal windows. For the rest of the paper, we will refer to the truncated pose history as \( \Theta_t = \{ \theta_1, \theta_{t-1}, \theta_{t-2}, ..., \theta_{t-n} \} \). Then, during training, we predict each sample \( x_t \) using \( \Theta_t \) as network input. For inference, the model can be fed with indefinitely long sequences, as new poses \( \theta \) are used to update the hidden recurrent state.

The naive baseline solution would be to feed body pose sequences \( \Theta_t \) as joint orientations– with global body velocities. While this would suffice to avoid an ill-posed problem, this representation is sub-optimal and dynamics are entangled with static information. The model would need to learn by itself to extract body motion information from the input poses. This increases the required training data and time, as well as model capacity, while hurting generalization. We propose a set of disentangled descriptors that are more suitable for this problem.

Static descriptors. To describe the body pose it is common to use joint relative orientations (relative to the parent joint). The usual axis angle or quaternion representations suffer from a many-to-one problem and discontinuities in the rotation space. Thus, we opt for the 6D descriptors proposed in [Zhou et al. 2019]. We observe relative orientations are local descriptors. While garment deformations depend on the global body configuration. Small changes in the first joints of the kinematic tree can lead to significantly different garment states. Therefore, samples close to each other in the input space would need to be mapped to points very far from each other in the output space, which makes training and generalization more challenging. On the other hand, using global joint orientations would make the input space extremely large and noisy. Rotations around the gravity axis would create completely different inputs, other in the output space, which makes training and generalization difficult. Hence, we use as motion descriptors the first derivative of joint orientations –any rotation implies an acceleration– and the second derivative of the joint locations. Both descriptors are concatenated into a 12-dimensional descriptor per joint, which gives us \( \theta^D \in \mathbb{R}^{K \times 12} \) (K joints, 9 dimensions from the first derivative of the static descriptors and 3 additional dimensions for the joint accelerations in local space).

Skinned 3D models often have many joints in hands, feet and face which are unlikely to be relevant for garment dynamics. We remove these joints from the input.

3.3 Model
In this section we present our model architecture. As explained, we propose a recurrent encoder-decoder network architecture. Our encoder is composed of two different modules, a static and a dynamic encoder, each fed with the corresponding descriptors. Both encodings are combined by addition to be later decoded into local cloth deformations. Finally, the garment is posed along with the body. Fig. 2 depicts our model.

Static encoder. We implement this encoder as a set of 4 fully connected layers. The encoder is fed only with the current pose \( \Theta^S \), which is flattened first into a 9K-dimensional array. The output of the encoder is a static latent code \( z^S \).

Dynamic encoder. This module is implemented in two blocks. A set of fully connected layers and a Gated Recurrent Unit (GRU). First, 2 fully connected layers are applied to per-joint descriptors individually –as if joints were samples– to obtain a high-level feature array per joint. We empirically observed this to be beneficial since dynamic descriptors are a concatenation of different modalities. Later, the array is flattened and fed to an additional 2 fully connected layers. Finally, the output is passed through the GRU, which combines it with its hidden state –that encodes the history of dynamics– to obtain the dynamic latent code \( z^D \). Note that \( z^D \) is then computed with the whole motion \( \Theta \). We observe adding multiple GRUs makes training unstable and hurts model inference speed. All layers of the dynamic encoder have no bias. Without bias, zero input translates to zero output (this is not necessarily true the other way around). This ensures that a static sample will have a null \( z^D \), since time derivatives –dynamic descriptors– will be zero. Thus, addition with \( z^S \) will have no impact. Furthermore, samples with high body motion will generally produce high values for \( z^D \), creating high dynamic deformations due to a high perturbation of \( z^D \). Additionally, the hidden state of the GRU will fade away to zero as long as a constant pose (no motion) is being fed to the model.
We train our model unsupervisedly by applying losses inspired in [Bertiche et al. 2021a], within a deep learning framework, the only requirement is that it is differentiable. This makes our approach compatible with most cloth models, allowing them to be freely interchanged. For this work, we implemented a mass-spring model as in [Bertiche et al. 2021a], a squared version of the continuum formulation of [Baraff and Witkin 1998; Liu et al. 2012]. To be able to use a cloth formulation with external objects is obtained by detection and solving of collisions. Similarly, we implement a loss term that penalizes collisions and creates repelling gradients, pushing cloth vertices outside the body [Bertiche et al. 2021a,b]:

\[
L_{\text{collision}} = k_c \min(d(x_t; \theta_t) - \epsilon, 0)^2, \tag{7}
\]

where \(k_c\) is a balancing factor, \(d(\cdot; \theta)\) is the signed distance to a body mesh parameterized by \(\theta\), with negative values inside, and \(\epsilon\) is a small threshold to ensure robustness.

**Inertia Loss.** Following the laws of motion, a moving object will retain its velocity unless forces act on it. Thus, differences in the location of the cloth particles \(x_t\) and the projected location obtained with the previous velocity \(\mathbf{x}_t^{\text{proj}} = x_{t-1} + v_{t-1}\Delta t = 2x_{t-1} - x_{t-2}\) are due to other acting forces. A similar observation has already been made in the context of simulation [Gast et al. 2015; Liu et al. 2013; Martin et al. 2011]. This led to the possibility of obtaining the next garment state by finding the critical points of the following expression:

\[
h(x_t) = \frac{1}{2} (x_t - \mathbf{x}_t^{\text{proj}})^T M (x_t - \mathbf{x}_t^{\text{proj}}) + \Delta t^2 E(x_t), \tag{8}
\]

**Bending Loss.** We implement our bending term for out-of-plane deformations as the squared difference of the angle between adjacent faces w.r.t. the angle in the rest garment [Pfaff et al. 2014]. Then, for each pair of adjacent faces:

\[
L_{\text{bending}} = k_b \frac{l^2}{8a} (\phi_t - \phi^R)^2, \tag{6}
\]

where \(k_b\) is the bending stiffness, \(l\) is the length of the common edge, \(a\) is the summation of the area of both faces, \(\phi_t\) is the dihedral angle, \(\phi^R\) is the dihedral angle in the rest garment. With this formulation, the garment will try to retain its original shape. Scaling the loss as a function of the edge length and triangles area makes it agnostic to mesh resolution and connectivity.

**Collisions.** In computer graphics simulations, cloth interaction with external objects is obtained by detection and solving of collisions. Similarly, we implement a loss term that penalizes collisions and creates repelling gradients, pushing cloth vertices outside the body [Bertiche et al. 2021a,b]:

\[
L_{\text{collision}} = k_c \min(d(x_t; \theta_t) - \epsilon, 0)^2, \tag{7}
\]

where \(k_c\) is a balancing factor, \(d(\cdot; \theta)\) is the signed distance to a body mesh parameterized by \(\theta\), with negative values inside, and \(\epsilon\) is a small threshold to ensure robustness.

**Inertia Loss.** Following the laws of motion, a moving object will retain its velocity unless forces act on it. Thus, differences in the location of the cloth particles \(x_t\) and the projected location obtained with the previous velocity \(\mathbf{x}_t^{\text{proj}} = x_{t-1} + v_{t-1}\Delta t = 2x_{t-1} - x_{t-2}\) are due to other acting forces. A similar observation has already been made in the context of simulation [Gast et al. 2015; Liu et al. 2013; Martin et al. 2011]. This led to the possibility of obtaining the next garment state by finding the critical points of the following expression:

\[
h(x_t) = \frac{1}{2} (x_t - \mathbf{x}_t^{\text{proj}})^T M (x_t - \mathbf{x}_t^{\text{proj}}) + \Delta t^2 E(x_t), \tag{8}
\]
where \( \mathbf{M} \) is the mass matrix of the particle system, \( \Delta t \) is the time step of the simulation and \( E(\cdot) \) is the potential representing internal and external forces. This, similar to [Santesteban et al. 2022], leads to the following loss term for each particle:

\[
L_{\text{inertia}} = \frac{1}{2\Delta t^2} m(x_t - x_t^{\text{proj}})^2, 
\]

where \( m \) is the particle mass. Since \( x_t^{\text{proj}} \) depends on \( x_{t-1} \) and \( x_{t-2} \), we need to run the model for \( \Theta_{t-1} \) and \( \Theta_{t-2} \) as well. It is crucial not to back-propagate gradients through \( x_{t-1} \) and \( x_{t-2} \). Otherwise, while the loss value gets lower, the model will not show true cloth dynamics. The reason for this is that \( x_t \) would have influence in the location of \( x_{t-1} \) and \( x_{t-2} \) by generating pulling or pushing gradients. Thus, information would be travelling back in time. Note how simulation based related works from which we extract this loss term do not optimize \( x_{t-1} \) or \( x_{t-2} \) to satisfy eq. 8. One important observation is that this loss will penalize differences in velocities. Thus, whenever the underlying body skeleton joints present no rotations or accelerations, there will not be accelerations in the cloth —since it is attached by blend weights to the skeleton— and gradients will not to back-propagate the inertia loss only through \( x_t \). While the static loss terms can be safely applied to all 3 predictions, we observe this creates a sampling bias that hurts performance.

Balancing terms of each loss are related to desired fabric properties. For the cloth term, the values will depend on the chosen cloth material model. Usually within the range [5, 15] for structural stiffness and [0, 1] for shearing. For bending, we use values in the range \([1e-5, 1e-4]\). For collisions, we set \( k_c \) to a value similar to the chosen structural stiffness and a threshold \( \epsilon = 4 \text{mm} \). Higher values for \( k_c \) will compromise other metrics without improving generalization. Collision-free predictions will depend mostly on training data distribution. Particle mass \( m \) or mass matrix \( \mathbf{M} \) is computed from vertex area and the chosen fabric surface density. Then, the temporal window will depend on the looseness of the garment. We go from 0.5 seconds for tighter garments up to 2 seconds for looser garments. Training times until convergence will differ greatly depending on garment, body and motions. From 1 hour for simpler garments up to a day for garments that show complex and rich dynamics. Inference is extremely fast, as we show in Tab. 4. We refer to the supplementary code for additional details. The code will be publicly released.

4.2 Metrics

Traditionally, specially for supervised approaches, lower values for error metrics indicate better predictions. This is not the case for this specific unsupervised problem. Furthermore, the metrics will behave differently for the static case. For that reason, it must be considered and evaluated separately.

**Cloth Model.** This metric must be related to the cloth material model used during training. For mass-spring, edge elongation is the most suitable. For continuous formulations, the strain or triangle deformation is a better choice. This metric will measure in-plane cloth deformations. Cloth is resistant to stretching forces, thus, lower values are desired in this case. Some formulations allow modelling shearing forces separately. Cloth is not resistant to shearing, and thus, lower values do not necessarily imply better predictions.

**Bending.** Measured as the average error on dihedral angles w.r.t. rest angles for each pair of adjacent faces. Cloth is not resistant to bending. Lower is not necessarily better. A higher bending stiffness will reduce this error, but generate different wrinkles. Thus, a lower error does not imply better predictions, but a stiffer fabric. Note that a null bending error would only be achieved by the template garment in rest pose. Nonetheless, abnormally high values for this metric might suggest a failed simulation. Ultimately, this property must be assessed qualitatively.

**Collisions.** Expressed as the percentage of vertices placed within the body. Collisions are to be avoided. For this metric, lower values are desired. For the dynamic case this metric shows an interesting

---

1https://www.mixamo.com/
We analyze the effect of different cloth material models. To this end, with different cloth models. We test the mass-spring formulation we simulate in a static fashion the same garment, body and poses when dynamics begin to appear, increasing the value of this metric. We can state that training converged when this metric plateaus. Dy-value of this metric and its evolution during training will greatly

\textbf{Gravity.} Computed as the potential energy of the predicted garments. This energy is defined relative to an arbitrary \(0\). Then, it is not possible to define a \textit{goal} for this metric. Its value will also depend on the garment, fabric density, 3D body and pose data. \textbf{Static case:} the optimal solution is achieved when the garment has reached an equilibrium state. For this case, given the same aforementioned experiment conditions, lower is better. Other metrics must be considered as well. A lower cloth stiffness would allow further stretching in the direction of gravity. In such case, we could not conclude the approach converges to a more optimal solution. On the other hand, we can state that training converged when this metric plateaus. \textbf{Dy-namic case:} adding motion to the problem changes the behaviour of this metric. For example, a spinning skirt will raise against gravity when dynamics begin to appear, increasing the value of this metric. Likewise, jumping sequences will make the garment \textit{float} when the falling motion begins. This means the garment is not always at its lowest position. For this reason, it is not possible to conclude that lower values are better. Usually, the static formulation gives lower values for gravity. Therefore, a dynamic model with a lower gravity metric might be due to a lack of cloth dynamics.

\textbf{Inertia Loss.} Measured as the error between \(x_t\) and \(x^\text{proj}_t\) weighted by the particle mass. As explained in Sec. 3.4, lower values do not translate into true cloth dynamics. We empirically observe it behaves the other way around. As training advances and cloth dynamics are being learnt, the value of this metric increases. On the contrary, under the static formulation, this metric will usually decrease. Similar to how gravity metric indicates model convergence for the static case, this metric does the same for the dynamic case. Convergence in training but divergence in validation shows overfitting. This metric also gives an intuition of the level of motion in the predictions. The value of this metric and its evolution during training will greatly depend on garment, fabric, body and motion data.

It is very important to note that all metrics need to be considered at once. The improvement of a single metric is not sufficient to make conclusions regarding the results.

### 4.3 Cloth Model

We analyze the effect of different cloth material models. To this end, we simulate in a static fashion the same garment, body and poses with different cloth models. We test the mass-spring formulation used by authors of [Bertiche et al. 2021a], the continuous formulation of [Baraff and Witkin 1998] and the Saint Venant Kirchhoff (StVK) elastic material model as in [Santesteban et al. 2022]. We adapt the material model of [Baraff and Witkin 1998] by squaring their constraints for stretching and shearing. We depict the obtained results in Fig. 3. As shown, the chosen cloth formulation has almost no impact on the qualitative result. Nonetheless, we observe some differences worth mentioning. As opposed to continuous formulations, mass-spring fabric parameters depend on mesh resolution and connectivity. On the other hand, we find that StVK formulation has much more difficulties achieving convergence. We can see this in the gravity plot adjoined to Fig. 3. This formulation may be sub-optimal within the deep learning optimization framework. Finally, the formulation of [Baraff and Witkin 1998] allows explicit control of shearing stiffness. We observed each cloth model scores lower in their respective strain—as expected—and thus, it is not useful to compare strains quantitatively.

### 4.4 Ablation

\textbf{Batch size.} Unsupervised garment animation is quite sensitive to batch size. The reason for this is as follows. Model evolution during training is similar to physical simulation. In the static case, each input sample \(\theta_i\) has a theoretical optimal output \(x_i\). Under supervised

![Image](https://example.com/image.jpg)

\textbf{Fig. 3.} Comparison of cloth material models. Our methodology is compatible with any differentiable formulation for cloth. We test and compare three different ones as a static optimization problem: A) mass-spring model as in [Bertiche et al. 2021a], B) the continuous formulation proposed by [Baraff and Witkin 1998] and C) the Saint Venant Kirchhoff material used in [Santesteban et al. 2022]. For static optimization, we can use gravity as a measure of convergence.

| Strain (mm) | Bending (rads) | Collision (%) | Gravity |
|------------|----------------|---------------|---------|
| Raw        | 0.35463        | 0.0168        | 0.087418| 0.1845  |
| 6D         | 0.37374        | **0.01372**   | 0.10778 | 0.1811  |
| 6D+G       | **0.35145**    | 0.01475       | **0.073881** | **0.1765** |

Table 2. Ablation study on network architecture and data augmentation. Experiments with + contain the \textit{improvements} from the previous rows. First row: single encoder. Second row: disentangled encoder. Third row: pose mirroring. Last row: motion augmentation. First and second experiment have no data augmentation. Second, third and fourth experiment have the same architecture. We see how a disentangled encoder and the proposed data augmentations have a beneficial impact. Additionally, we see gravity and inertia show no significant difference. Therefore, these improvements do not compromise other cloth properties.

| Strain   | Collisions (%) | Gravity | Inertia |
|----------|----------------|---------|---------|
| Entangled| 9.7840         | 1.101   | 1.068   | 1.889   |
| Disentangled| 8.2391     | 0.694   | 1.066   | 1.891   |
| +Mirror  | 8.2146         | 0.505   | 1.067   | 1.864   |
| +Aug. motions | **7.6241** | **0.323** | 1.068   | 1.905   |

Table 1. Static descriptors ablation. We run experiments for each static descriptor under a static formulation. First, raw joint orientation data as quaternions or axis angles. Second, 6D descriptors as in [Zhou et al. 2019]. Finally, our proposed static descriptors. Our approach shows better generalization and convergence without compromising cloth integrity.
Fig. 4. Training progress for different batch sizes. The upper row corresponds to static experiments. The lower row corresponds to dynamic experiments. Using bigger batch sizes significantly increases convergence of the predictions. On the dynamic problem, bigger batches result in a more stable training.

Fig. 5. Ablation on dynamic input features. Baseline features contain body pose and root joint velocity. The other experiments contain body pose and joint rotation speeds and accelerations. For the local-space features, we unpose accelerations. We evaluate collision (left) and inertia (right) metrics. Upper row corresponds to training data. Lower row to validation data. We see local-space features generalize better—fewer collisions—while showing a similar level of dynamics as the training data—same inertia curves. We omit the color labels of the training plots (upper row) since they show a very similar behaviour.

Fig. 6. Ablation on temporal window size. We compare neural simulation metrics for a t-shirt with a temporal window size of 0.1 seconds, 0.5 seconds and 1 second. We can see the level of dynamics—inertia metric—is much lower with a small temporal window. We can also observe a slight decrease in gravity metric in this case. This is an example of the effect of dynamics on gravity explained in Sec. 4.2. On the other hand, we see that further increasing temporal window size has almost no impact in the results. We omit other metrics since their values is almost equal.

Fig. 7. Data augmentation ablation. Collision (left) and inertia (right) metrics evolution for validation set during training for different augmentation techniques. First: no augmentation. Next: pose mirroring. Finally, a novel motion augmentation technique. The latter is only possible thanks to disentangled encoders. As cloth dynamics are being learnt by the network, collisions slightly increase. Pose mirroring reduces this effect, while motion augmentation mitigates it almost completely. We also see the evolution of cloth dynamics—inertia metric—is not compromised by these changes.

training, gradients in the output will point directly towards $x_i$, with stronger magnitudes for more erroneous predictions. This is not the case for unsupervised garment animation. Gradients will try to greedily update the output cloth by pointing to an intermediate state $\hat{x}_i$, similar to how a simulation would compute intermediate states until achieving the fully converged solution $x_i$. Moreover, there is no guarantee that $\hat{x}_i$ will be closer in space to $x_i$ than previous predictions. The path the model has to follow to convergence is not straight-forward. Furthermore, gradient magnitudes cannot be used as a measure of convergence, except in the extremely unlikely case
that their value is 0 for all the samples in our dataset. On top of that, we have to consider special constraints found in deep learning. Network updates, specially for small batches, can undo the work of previous iterations. Then, the network gets stuck in sub-optimal local minima. This produces stiff garments with wrinkling patterns that repeat across different poses. With dynamics, the problem becomes more complex, since the gradients for \( x_t \) depend on \( x_{t-1} \) and \( x_{t-2} \), which are also predictions. Fig. 4 shows training metrics for static and dynamic problems with different batch sizes. For the static case, we see that increasing batch size greatly improves model convergence. For the dynamic case, we see that using a small batch size makes the training noisy. There is no plot for batch size 16 for dynamics since small batches make training unstable and usually fails.

**Static descriptors.** We study the impact of the proposed static descriptors under a static formulation. This means training without \( L_{\text{inertia}} \). We test three different static descriptors. Tab. 1 contains the results obtained. The first row corresponds to raw pose data as quaternions or axis angle representations. Next, 6D descriptors as in [Zhou et al. 2019]. Finally, the static descriptors proposed in Sec. 3.2. The proposed descriptors present fewer collisions, showing better generalization. They also achieve higher convergence, since gravity is lower. Finally, we see also a lower strain value, which means it did not compromise cloth integrity to minimize the other metrics.

**Dynamic descriptors.** We test three different alternatives as motion descriptors. First, as baseline, we use body pose and root joint velocity. This descriptor is the minimum requirement to avoid an ill-posed problem. Second, we gather joint rotation speed and accelerations without unposing (full-space). Finally, our proposed descriptors, after unposing joint accelerations (local-space). Fig. 5 depicts the training plots for collisions and inertia. We omit other metrics since their values barely differ. At left, we have the collision metric. At right, the inertia metric. The first row corresponds to training set plots, and the lower row to the validation set. It is interesting to notice that training plots (top row) are almost the same. For validation (bottom row), we observe full-space descriptors show more collisions. These descriptors have a higher variability. Thus, it is a much more challenging task for the network to learn the whole input space. This results in worse generalization. The other two descriptors are local, then, validation set distribution is more likely to fall in the same distribution learnt by the network. For the inertia metric, training and validation plots –although differ in absolute values– show similar curves. We see the baseline features diverge from the other curves. This indicates a lower generalization. The model has difficulties in extracting meaningful motion information from the baseline representation. Providing of explicit joint rotation speeds and accelerations helps the network to better learn the dependency between body motion and dynamic cloth deformations.
Architecture. We perform an analysis on model architecture by comparing our disentangled approach against a single encoder. For this experiment, the entangled encoder is fed with static and dynamic descriptors. Next, the GRU receives the output encoding. Finally, the decoder predicts garment deformations. The result of this comparison is shown in the first two rows of Tab. 2. First row corresponds to a single entangled encoder. The second row corresponds to the proposed disentangled architecture. We see that separate encoders have better generalization – lower strain and collisions – than a single encoder. Additionally, disentangled encoders make training much more stable. Training with an entangled encoder usually fails. This justifies our motivations in the network design.

Temporal window. As presented in eq. 2, predictions need the pose history. We analyze the effect of different temporal window sizes. For this experiment, we neurally simulate a t-shirt with a window size of 0.1 seconds, 0.5 seconds and 1 second. We present the metrics of this experiment in Fig. 6. As we can see from the inertia metric, a smaller window gives a lower level of dynamics. During training, a reduced input data has a direct impact on the discriminative power of the network. The model cannot properly differentiate motions that are similar within the 0.1 second window, but have a different past. Because of this, predictions converge to the average of the motions that are close to each other in this reduced input space, which lowers the level of dynamics. Additionally, we observe a lower value for the gravity metric for the smaller window size. This is related to the discussion on the effect of dynamics on gravity in Sec. 4.2. On the contrary, we observe an even larger temporal window size has no significant impact in cloth dynamics – for this specific case – but increases the required VRAM and time for training. Therefore, for each specific encoder, body and motions, it is important to find a window size that achieves a compromise between dynamics and efficient training. Note that during inference, sequence length can be arbitrarily large. We additionally test the effect of $\Delta t$ by training at different frame rates, 15 and 60 (for all other experiments the frame rate is 30). We notice the model is able to learn realistic cloth dynamics even at lower frame rates. Furthermore, at 15 fps training is significantly faster. On one hand, the amount of samples is halved. On the other hand, for the same temporal window, the length of the pose sequences is smaller, which means less operations. Finally, the gradients generated by $L_{\text{inertia}}$ are larger, and dynamics take less time to appear. On the contrary, increasing the frame rate makes the task much more challenging for the same reasons (but opposed). Nonetheless, training at higher frame rate allows learning finer cloth dynamics. We refer to the supplementary video for a comparison of a motion learnt at different frame rates.

Augmentation. As the model learns cloth dynamics, collisions in the validation set increase. We study the possibility of mitigating this effect with data augmentation. On one hand, we use standard pose mirroring with probability of 50%. On the other hand, leveraging our disentangled approach, we devise a novel motion augmentation technique. To do so, during training, we shuffle the dynamic latent code $z^d$ for a portion of the samples from each batch (20% in this experiment). We can apply only the static loss terms to the augmented samples. We do not back-propagate gradients to the encoders for these samples. This will give the decoder more data points from $\mathcal{Z}$ and around $\mathcal{Z}^s$, increasing generalization. Tab. 2 shows the effect of the different augmentations. Second row, no augmentation. Third row, pose mirroring. Last row, motion augmentation. We see that for both augmentations, strain and collision metrics are noticeably reduced, specially for motion augmentation. Moreover, we see it almost completely mitigates the increase of collisions as cloth dynamics are being learnt. This is shown in Fig. 7, left plot (collisions). It is also important to notice that gravity and inertia metrics show very little difference. This means the generalization improvement does not compromise other properties.

We show qualitative results of our methodology for different garments, bodies and motions in Fig. 1, 8 and 9. Fig. 1 shows results for SMPL model with different body shapes and the mannequin model from Mixamo. The samples show rich and meaningful cloth dynamics. Fig. 8 depicts predictions for a dress during a spinning sequence. As observed, as the motion begins, the dress coils up the mannequin body as we would expect to happen. Finally, in Fig. 9 we can see how our methodology can generalize to different articulated 3D bodies and garments.

4.5 State-of-the-art Comparison

We evaluate our methodology against recent unsupervised approaches for garment animation. On one hand, a quasi-static solution, PBNS [Bertiche et al. 2021a]. On the other hand, a methodology that claims to model dynamic cloth deformations, SNUG [Sanstebetan et al. 2022]. SNUG authors provide of a checkpoint we use for comparison. Nonetheless, their methodology is adapted to work with

| Strain  | Bending | Collision (%) | Gravity | Inertia |
|--------|---------|---------------|---------|---------|
| PBNS   | 3.18    | 0.20          | 0.274   | 0.6274  | 0.937   |
| SNUG   | 3.89    | 0.20          | 0.283   | 0.6299  | 0.553   |
| Ours   | 4.2     | 0.15          | 0.276   | 0.6403  | 1.219   |

| DoF    | PBNS | SNUG | Ours |
|--------|------|------|------|
| 10K    | 770.5 | 454.5 | 853.9 |
| 250K   | 250.5 | -    | 206.3 |

Table 3. Quantitative comparison with state-of-the-art: PBNS [Bertiche et al. 2021a] and SNUG [Sanstebetan et al. 2022]. SNUG achieves a much lower value for the inertia term. Nonetheless, see in Fig. 10 that their approach shows no cloth dynamics. As explained in Sec. 4.2, lower values do not translate to cloth dynamics.
Neural Cloth Simulation

Fig. 10. Qualitative comparison for different kinds of motions. The motions are: a) jumping, b) leap forward, c) quasi-static pose, d) quasi-static pose, e) jumping, f) dancing jump, g) flapping arm motion and h) fast spin. We use PBNS public code. For samples a, b, c and d, we use SNUG public checkpoint. For samples e, f, g and h, we implement and train SNUG based on authors public code and paper. Since PBNS uses a static formulation, it shows no cloth dynamics. Then, we see SNUG appears to be unable to show any meaningful dynamics. This is mainly due to an incorrect implementation of the inertia loss. Additionally, looking at the quasi-static samples (c and d), we notice wrinkling patterns that repeat for most poses. This gives a stiff and less realistic look. Finally, our approach shows dynamic cloth deformations consistent with body motion. We also show results obtained with standard simulation as a reference.

Fig. 11. Motion control. In this image we show a still frame of a spinning motion. Thanks to the disentanglement of static and dynamic cloth deformations achieved by the network design, it is possible to control the level of motion in the cloth. To do so, we scale the latent dynamic code to linearly interpolate and extrapolate from the static subspace to the full subspace. Leftmost to middle samples: linear interpolation from static latent code \( z^S \) to latent code \( z \). Middle to rightmost samples: linear extrapolation.

the body shape variability of SMPL. For fairness, we also implement and train their methodology based on their public code and paper using constant body shape. Tab. 3 shows quantitative results for each model. Qualitative evaluation is included in Fig. 10. Note we additionally include, as a reference, qualitative results obtained with standard simulation (ArcSim [Narain et al. 2012]). We show samples for different body motions: a) jumping, b) leap forward, c) quasi-static pose, d) quasi-static pose, e) jumping, f) dancing jump, g) flapping arm motion and h) fast spin. We use PBNS public code. We know PBNS to be a static solution and therefore, as expected, it does not show dynamic cloth deformations. It is interesting to notice that for quasi-static samples, our solution and PBNS converge to a
We propose a new standard for measuring the performance of pose-driven animation. In the literature we find authors that consider only the forward pass of the network as running time. This is misleading. SNUG authors report in their paper a performance of 454.5fps. Unfortunately, their runtime GPU code is not open, and so it cannot be adapted to other data sets. According to the authors, it includes collision post-processing by all-pairs testing of cloth and body primitives, parallelized on the GPU.

4.6 Cloth Subspace Disentanglement and Motion Control

The methodology presented in this work is designed to automatically learn disentangled subspaces for static and dynamic cloth deformations. To prove our model is effectively doing so, we linearly interpolate and extrapolate between the static cloth subspace $\mathcal{S}$ and the full cloth subspace $\mathcal{Z}$. To this end, we scale the dynamic latent code given by the dynamic encoder $Z^D$ with $w \in [0, 2]$. That is, $z = z^S + wz^D$. This can be seen in Fig. 11. From left to right, the values for $w$ go from 0 to 2 with steps of 0.2. The sample in the middle is the standard network output. This is learnt automatically by the network without any sample or latent code manipulation.

5 CONCLUSIONS AND LIMITATIONS

We presented a general framework for unsupervised garment animation. Contrary to previous related works, our work is the first methodology able to learn cloth dynamics without ground truth data. Additionally, we devise a novel disentangled architecture that improves generalization, opens the possibility of a new motion augmentation technique that greatly increases robustness and allows for motion control, which is a useful never seen before property for artists. Also, we provide of detailed analysis and insights on neural cloth simulation that will help future research on the domain. We proved the effectiveness of our methodology with different 3D avatars and garments. Note how the models obtained with our methodology are specific for a given avatar and garment. For new garments or bodies, our methodology needs to be applied again.
to obtain the corresponding model. Also, while in this work we did not explore body shape generalization, the works of [Bertiche et al. 2021a; Santeeteban et al. 2022] proved it. Trained models can generalize to the chosen body parameterization (see Eq. 1).

Removing the need of gathering ground truth data is a huge advantage in the domain, already noted by previous works [Bertiche et al. 2021a; Santeeteban et al. 2022]. Nonetheless, simulation cannot be completely skipped. Instead of being an offline process, simulation and training are now the same. The first time the network sees a given training sample, its corresponding $x^{t(i)}$ will be an estimation implicitly computed from body motion (transferred to the cloth through garment blend weights). The next epoch, this estimation will be more accurate. So on and so forth. This means that fine cloth dynamics take time to appear, since they have to be indeed simulated within the network. It would not be possible for the network to learn these dynamics without going through these intermediate states. This effect is even greater for looser garments.

This means that supervised training will always be much faster, even without considering that unsupervised losses are more computationally expensive. Nonetheless, the methodology still has a significant advantage over supervised approaches, since similar sample motions will contribute to each others neural simulations. On the contrary, even for very similar motions, data gathering through simulation has to be done from scratch for every sequence. Another advantage of unsupervised training is that the network will converge to the simplest solution. That is, similar motions will show similar deformations. On the other hand, offline simulations may show a huge variability for similar motions. Because of this, supervised training needs to deal with noisy data that makes the task much more challenging. Finally, supervised training shows a bias towards lower frequencies and does not satisfy physical constraints without explicit regularization. Then, overall, unsupervised training needs to deal with noisy data that makes the task much more challenging. Nonetheless, simulation cannot be completely skipped. Instead of being an offline process, simulation and training are now the same. The first time the network sees a given training sample, its corresponding $x^{t(i)}$ will be an estimation implicitly computed from body motion (transferred to the cloth through garment blend weights). The next epoch, this estimation will be more accurate. So on and so forth. This means that fine cloth dynamics take time to appear, since they have to be indeed simulated within the network. It would not be possible for the network to learn these dynamics without going through these intermediate states. This effect is even greater for looser garments.

This means that supervised training will always be much faster, even without considering that unsupervised losses are more computationally expensive. Nonetheless, the methodology still has a significant advantage over supervised approaches, since similar sample motions will contribute to each others neural simulations. On the contrary, even for very similar motions, data gathering through simulation has to be done from scratch for every sequence. Another advantage of unsupervised training is that the network will converge to the simplest solution. That is, similar motions will show similar deformations. On the other hand, offline simulations may show a huge variability for similar motions. Because of this, supervised training needs to deal with noisy data that makes the task much more challenging. Finally, supervised training shows a bias towards lower frequencies and does not satisfy physical constraints without explicit regularization.

**ACKNOWLEDGMENTS**

This work has been partially supported by the Spanish project PID2019-105039GB-100 (MINECO/FEDER, UE) and CERCA Programme/Generalitat de Catalunya. This work is partially supported by ICREA under the ICREA Academia programme.

**REFERENCES**

Amir Arsalan Soltani, Haibin Huang, Jiaying Wu, Tejas D Kulkarni, and Joshua B Tenenbaum. 2017. Synthesizing 3d shapes via modeling multi-view depth maps and silhouettes with deep generative networks. In Proceedings of the IEEE conference on computer vision and pattern recognition. 1511–1519.

David Baraff and Andrew Witkin. 1998. Large steps in cloth simulation. In Proceedings of the 25th annual conference on Computer graphics and interactive techniques 43–54.

David Baraff, Andrew Witkin, and Michael Kass. 2003. Untangling cloth. ACM Transactions on Graphics (TOG) 22, 3 (2003), 862–870.

Hugo Bertiche, Meyyas Madadi, and Sergio Escalera. 2020. CLOTH3D: clothed 3d humans. In European Conference on Computer Vision. Springer, 344–359.

Hugo Bertiche, Meyyas Madadi, and Sergio Escalera. 2021a. PBNS: Physically Based Neural Simulation for Unsupervised Garment Pose Space Deformation. ACM Trans. Graph. 30, 6, Article 198 (dec 2021). 14 pages. 

Hugo Bertiche, Meyyas Madadi, Emilio Tyison, and Sergio Escalera. 2021b. DeepSD: Automatic deep skinning and pose space deformation for 3d garment animation. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 5471–5480.

David E. Breeden, Donald H. House, and Phillip H. Getto. 1992. A physically-based particle model of woven cloth. The Visual Computer 8, 5 (1992), 264–277.

Michel Carriga, Ying Yang, Nadja Magenheit Thalmann, and Daniel Thalmann. 1992. Dressing animated synthetic actors with complex deformable clothes. ACM Siggraph Computer Graphics 26, 2 (1992), 99–104.

Edilson De Aguiar, Leonid Sigal, Adrien Treuille, and Jessica K Hodgins. 2010. Stable spaces for real-time clothing. ACM Transactions on Graphics (TOG) 29, 4 (2010), 1–9.

Carl Richard Feynman. 1986. Modeling the appearance of cloth. Ph. D. Dissertation. Massachusetts Institute of Technology.

Lawson Fulton, Viswan Mod, David Davanen, David I. L. Welin, and Alek Jacobson. 2019. Latent-space Dynamics for Reduced Deformable Simulation. Computer Graphics Forum (2019).

Theodore F. Gast, Craig Schroeder, Alexey Stormahik, Chenfanfu Jiang, and Joseph M. Teran. 2015. Optimization integrator for large time steps. IEEE transactions on visualization and computer graphics 21, 10 (2015), 1103–1115.

Erhan Gundogdu, Victor Constantin, Amrollah Seifoddini, Minh Dang, Mathieu Salzmann, and Pascal Fua. 2019. Garnet: A two-stream network for fast and accurate 3d cloth draping. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 8739–8748.

Fabian Hahn, Bernhard Thomaszewski, Stelian Coros, Robert W Sumner, Forrestor Cole, Mark Meyer, Tony DeRose, and Markus Gross. 2014. Subspace clothing simulation using adaptive bases. ACM Transactions on Graphics (TOG) 33, 4 (2014), 1–9.

Xiaoguang Han, Chang Gao, and Yizhou Yu. 2017. DeepSketch2Face: a deep learning based sketching framework for editing drawings and caricature modeling. ACM Transactions on Graphics (TOG) 36, 4 (2017), 1–12.

David Haumann. 1987. Modeling the physical behavior of flexible objects. Topics in Physically-based Modeling, Eds. Barr, Barrell, Haumann, Kass, Platt, Teszepoulos, and Witkin. SIGGRAPH Course Notes (1987).

Jonathan Kaldor, Doug L. James, and Steve Marschner. 2008. Simulating knitwear cloth at the yarn level. In ACM SIGGRAPH 2008 papers. 1–9.

Jonathan M Kaldor, Doug L. James, and Steve Marschner. 2010. Efficient yarn-based cloth with adaptive contact linearization. In ACM SIGGRAPH 2010 papers. 1–10.

Doyub Kim, Woongj Koh, Rahul Narain, Kayvon Fatahalian, Adrien Treuille, and James O’Brien. 2019. Near-exhaustive precomputation of secondary cloth effects. ACM Transactions on Graphics (TOG) 33, 4 (2014), 1–9.

Tian Tian Liu, Adam W Bargteil, James O’Brien, and Ladalash Kavan. 2013. Fast simulation of mass-spring systems. ACM Transactions on Graphics (TOG) 32, 6 (2013), 1–7.

Matthew Lopez, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. 2015. SMPL: A skinned multi-person linear model. ACM transactions on graphics (TOG) 34, 6 (2015), 1–16.

Miles Macklin, Matthias Muller, and Nuttapong Chentanez. 2016. XFBPD: position-based simulation of compliant constrained dynamics. In Proceedings of the 9th International Conference on Motion in Games. 49–54.

Meyyas Madadi, Hugo Bertiche, and Sergio Escalera. 2020. SMPLR: Deep learning based SMPL reverse for 3D human pose and shape recovery. Pattern Recognition (2020), 107472.

Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. 2019. AMASS: Archive of motion capture as surface shapes. In Proceedings of
the IEEE International Conference on Computer Vision. 5442–5451.
Sebastian Martin, Bernhard Thomaszewski, Edan Grinspun, and Markus Gross. 2011.
Example-based elastic materials. In ACM SIGGRAPH 2011 papers. 1–8.
Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. 2007. Position
based dynamics. Journal of Visual Communication and Image Representation 18, 2
(2007), 109–118.
Rahul Narain, Armin Samii, and James F O’Brien. 2012. Adaptive anisotropic remeshing
for cloth simulation. ACM transactions on graphics (TOG) 31, 6 (2012), 1–10.
Mohamed Omran, Christoph Lassner, Gerard Pons-Moll, Peter Gehler, and Bernt Schiele.
2018. Neural body fitting: Unifying deep learning and model based human pose and
shape estimation. In 2018 international conference on 3D vision (3DV). IEEE, 484–494.
Zherong Pan, Hujun Bao, and Jin Huang. 2015. Subspace dynamic simulation using
rotation-strain coordinates. ACM Transactions on Graphics (TOG) 34, 6 (2015), 1–12.
Chaitanya Patel, Zhourongqengh Liao, and Gerard Pons-Moll. 2020. Tailorset: Predicting
clothing in 3d as a function of human pose, shape and garment style. In Proceedings
of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 7365–7375.
Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W Battaglia. 2020.
Learning mesh-based simulation with graph networks. arXiv preprint arXiv:2010.03409 (2020).
Tobias Pfaff, Rahul Narain, Juan Miguel De Joya, and James F O'Brien. 2014. Adaptive
tearing and cracking of thin sheets. ACM Transactions on Graphics (TOG) 33, 4
(2014), 1–9.
Xavier Provot et al. 1995. Deformation constraints in a mass-spring model to describe
rigid cloth behaviour. In Graphics interface. Canadian Information Processing Society,
147–147.
Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep
learning on point sets for 3d classification and segmentation. In Proceedings of the
IEEE conference on computer vision and pattern recognition. 652–660.
Elad Richardson, Matan Sela, and Ron Kimmel. 2016. 3D face reconstruction by learning
from synthetic data. In 2016 fourth international conference on 3D vision (3DV). IEEE,
460–469.
Igor Santesteban, Miguel A Otaduy, and Dan Casas. 2019. Learning-based animation
of clothing for virtual try-on. In Computer Graphics Forum, Vol. 38. Wiley Online
Library, 355–366.
Ken Shoemake. 1985. Animating rotation with quaternion curves. In Proceedings of the
12th annual conference on Computer graphics and interactive techniques. 245–254.
Richard Socher, Brody Huval, Bharath Bat, Christopher D Manning, and Andrew Y Ng.
2012. Convolutional-recursive deep learning for 3d object classification. In Advances
in neural information processing systems. 656–664.
Yun Teng, Miguel A Otaduy, and Theodore Kim. 2014. Simulating articulated subspace
self-contact. ACM Transactions on Graphics (TOG) 33, 4 (2014), 1–9.
Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. 1987. Elastically de-
formable models. In Proceedings of the 14th annual conference on Computer graphics
and interactive techniques. 205–214.
Tuanfeng Y Wang, Tianjia Shao, Kai Fu, and Niloy J Mitra. 2019. Learning an intrinsic
garment space for interactive authoring of garment animation. ACM Transactions
on Graphics (TOG) 38, 6 (2019), 1–12.
Jerry Wed. 1986. The synthesis of cloth objects. ACM Siggraph Computer Graphics 20,
4 (1986), 49–54.
Meng Zhang, Tuanfeng Y Wang, Duygu Ceylan, and Niloy J Mitra. 2021. Dynamic
neural garments. ACM Transactions on Graphics (TOG) 40, 6 (2021), 1–15.
Yi Zhou, Connelly Barnes, Jingwan Lu, Jinshi Yang, and Hao Li. 2019. On the continuity
of rotation representations in neural networks. In Proceedings of the IEEE/CVF
Conference on Computer Vision and Pattern Recognition. 5745–5753.
