Investigating Friction in the Deformation Behaviour Predicted by LiCROM

MELISA CIVELEKOGLU

Abstract— **Accurate and accelerated modelling of frictional interactions in complex simulations is crucial for numerous applications of physics based simulations. This paper aims to explore how the Linear Subspace Continuous Reduced Order Modelling with Neural Fields (LiCROM) framework works and how frictional forces can be integrated into it in order to enhance its capability to approximate the behaviour of frictional interactions involving deformable objects. I introduce a modified version of LiCROM that integrates frictional force calculations into the prediction of latent space dynamics via implicit time stepping. Through a series of experiments, I evaluate the impact of these frictional forces on the overall deformation predictions provided by LiCROM.**

Keywords: Physics-based simulation, Reduced-order modelling, Neural Field, Contact Handling, Friction

I. INTRODUCTION

Reduced-order modelling (ROM) can simplify complex simulations by approximating the system's solution space using a reduced kinematic representation. Through generating a number of simulated trajectories and identifying a low dimensional basis, known as the latent space, to approximate their displacements, it is possible to compute the dynamics of the system by evolving this basis. As a result, ROM is able to accelerate simulations by orders of magnitude. However, most ROM approaches rely on a specific spatial discretization, i.e. a mesh of domain $\Omega \subset \mathbb{R}^3$, limiting their adaptability and generalisation when being applied to novel or adaptive mesh configurations.

LiCROM's goal is to eliminate these restrictions by identifying the latent space over a continuous representation of the system, allowing for greater flexibility in handling different mesh resolutions and types. In a traditional ROM approach, the time-varying displacement of a mesh of n vertices, $\bar{u}(t) : \tau \to \mathbb{R}^{3n}$, given a temporal domain T, is approximated as the linear combination $\bar{u}(t) \approx \bar{U}q(t)$, where $\bar{U} \in \mathbb{R}^{3n\times r}$ is some time-independent basis and $q(t) : \tau \to Q$ is the reduced trajectory in the latent space $Q \subset \mathbb{R}^r$ with $r \ll n$. Note that a bar is placed over quantities that depend on specific spatial discretizations.

Consider each column $\bar{U}_k$ of $\bar{U}$ corresponding to the discrete displacement field over the n vertices and each row $\bar{U}_j$ corresponding to the basis for the j^{th} degree of freedom; the structure of $\bar{U}$ is inherently tied to the initial mesh discretization. In fact, the time-varying displacement of the i^{th} vertex is given by

$$\bar{u}_i(t) = \bar{W}_i q(t)$$

where $\bar{W}_i$ is a row vector with vector valued (in $\mathbb{R}^3$) entries for the three degrees of freedom at each vertex. By stacking these row vectors, we obtain $\bar{W}$, an $n \times r$ matrix of vector valued (in $\mathbb{R}^3$) entries, which maps the latent configuration $q(t)$ to the full-space displacements $\bar{u}(t)$:

$$\bar{u}(t) = \bar{W}q(t).$$

LiCROM proposes instead to use a continuous map W, which takes a point X from the reference domain and assigns subspace weights to it, allowing to express the displacement field as

$$u(X,t) = W(X)q(t).$$

Despite its capabilities in accurately accelerating deformable object simulations, LiCROM has not yet been tested extensively for deformable object simulations which involve contact friction. Accurate and efficient simulation of contact and friction is crucial for multiple applications of physics-based animations, such as video games and robotics simulations. Modelling friction affects

how bodies in the simulation interact with each other and their environment, thus introducing additional constraints on their movement and influencing the forces generated within the plane of contact, which can alter the dynamics and realism of the simulation.

In this paper, I investigate I investigate how the LiCROM framework can be adapted in order to update the latent space dynamics whilst accounting for contact friction. By doing so, my goal is to evaluate the accuracy and performance of the modified framework in predicting deformation behaviour in the presence of friction.

The remainder of this paper is organised as follows: Section 2 reviews related work and discusses current approaches to frictional simulation and reduced-order modelling using deep learning. Section 3 describes the training phase adapted from the LiCROM framework. Section 4 presents the proposed modifications to the latent space dynamics of LiCROM for the inclusion of friction. Section 5 reviews the experimental results and performance. Finally, Section 6 concludes the investigation with a discussion of the results and explores potential future research directions.

II. Related Work

A. Deformable Solid Simulation

Simulation of elastic deformable solids are significant in computer graphics modelling, first introduced by Terzopoulos et al. [26]. The finite element method (FEM) has been one of the widely used tools in deformable solid simulation, which solves the governing PDEs on a particular space discretization of the mesh. There have been significant studies done focusing on accelerating FEM especially when dealing with high-detailed models addressing. Two main categories of approach have been full space methods, which are outside the scope of this investigation, and reduced space methods, which improve performance by reducing the degrees of freedom of the system.

B. Reduced-Order Modelling

Model reduction methods aim to lessen the computational burdens of simulating large-scale dynamical systems by allowing for faster and cheaper simulations, whilst accurately representing the original large-scale system behaviour [5]. ROM methods have been used successfully in multiple fields, such as solid mechanics [1], [4], robotics [25] and fluid mechanics [28].

C. Deep Learning in Reduced-Order Modelling

While effective, traditional ROM methods often restrict the simulation state to evolve in a linear space, which can impose a limitation on their accuracy in certain problems. This motivated Lee and Carlberg [16] in introducing a framework which utilises autoencoder neural networks in projecting dynamical systems onto nonlinear manifolds. Meanwhile, DeepWarp [17] introduced a deep neural network (DNN) based framework in nonlinear deformable simulation utilising a Full Space Method, which failed to decouple mesh resolution from the simulation complexity. Fulton et al. [12] later proposed the first simulation framework utilising ROM and autoencoders for deformable solid dynamics. Continuous Reduced Order Modelling (CROM) using implicit neural networks [9] learned a discretization independent latent space, where a nonlinear decoder maps the reduced configuration q and point X from the reference domain to the associated deformed position.

D. Linear Reduced-Order Modelling with Deep Learning

Similar to the CROM framework, LiCROM aimed to learn a discretization-agnostic latent space. However, contrary to CROM, Chang et al. [7] returned to the use of a linear subspace in their LiCROM framework. As noted in their paper, this provided LiCROM with an advantage over CROM: the linearity of the basis simplified the projection to a minimization of a quadratic function, since the solution comes from solving a linear system. Through pre-factorization, this solution can be reused along with cubature points and achieve fast projection, whereas in CROM this is not possible due to its nonlinear approach.

E. Friction and Contact

Contact and collision handling is a large area of research in computer graphics. Previous research in this area include collision detection [6] and collision culling [29], [20]. Discrete approaches [3], [27] exist, which focus on detecting and resolving

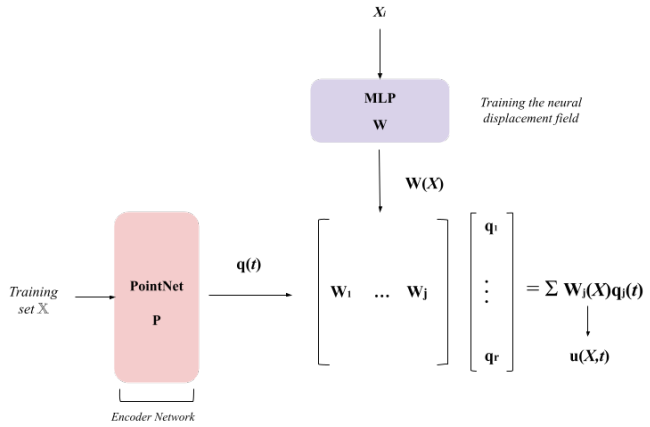

Fig. 1. *Network Architecture of Framework*. The model is trained for 1000 epochs with a learning rate of $lr = 10^{-3}$ and a batch size 16 for the network's input.

collisions at the end of each time step. Friction models typically use Coulomb's Law and accurate methods for simulating these models rely mainly on constraint based algorithms, such as used for solving the problem of rigid body stacking [2], [11], and further extended to soft materials [14].

III. Training Phase

I utilise the training method described in LiCROM for the offline stage of my implementation. Note that PyTorch Lightning is used for the implementation of the training pipeline.

The framework is trained over an observed trajectory of a deformable solid. For simplicity, consider a single trajectory sampled at instances $\{t^1, \ldots, t^m\}$. Let $\mathbb{X}$ represent the training set with entries $(\tilde{\mathbf{X}}_j, \tilde{\mathbf{u}}_j)$ which represent observations of the displacement field at time t^j. The goal is to identify a low-dimensional subspace to span these observed fields. Specifically, to find a projection $P : (\tilde{\mathbf{X}}_j, \tilde{\mathbf{u}}_j) \mapsto q_j \in Q$ and a corresponding basis $\mathbf{W}$ such that

$$\mathbf{u}_{ji} \approx \mathbf{W}(X_i)P(\tilde{X}_j, \tilde{\mathbf{u}}_j).$$

The dataset was initially generated using volumetric tetrahedral mesh discretizations using *TetWild* [13], but the network is agnostic to this knowledge during training. The training set includes full-space simulated displacement fields sampled over these meshes, which were generated via an implementation adapted from the default implicit FEM integrator from *warp* [18].

A. PointNet Encoder Architecture

The PointNet encoder P directly takes point clouds as input and is invariant under permutation of these points, which is essential for handling unordered sets. This is especially important as the input to P is an unordered set of points (X_i, u_i), where X_i and u_i each have three degrees of freedom. The key modules for the architecture are as follows: a max pooling layer acting as the symmetric function which aggregates information from all points, joint alignment networks and a local and global information combination structure [8]. The network effectively learns optimization criteria which select the most "informative" points of the point cloud. These learnt optimal values are then aggregated by the final fully connected layers to form a global descriptor and produce the output q. Note that the alignment of feature space is done on the joint alignment network in order to ensure invariance of the PointNet under input transformations, however, as latent space variables are not invariant under the transformations of displacements, the feature-transform net was removed for this implementation.

B. Neural Field Architecture

The basis W is parametrized using a 5-layer multilayer perceptron (MLP) network of width 60. As the network needs to be continuously differentiable with respect to both the spatial coordinates and the latent space vector, a continuously differentiable activation function must be adopted. Although both ELU [10] and SIREN [22] would work for this purpose, I utilise the former to follow the LiCROM framework closely as maintaining consistency with the original framework is crucial for ensuring precise and meaningful comparisons and evaluations. I use the Adam optimizer [15] and Xavier initialization for the ELU activation layer.

IV. Latent Space Dynamics

The Neo-Hookean model with rest stability [23], [19] will be used for the simulation of elastic solids. It consists of two separate energy terms:

deviatoric energy and hydrostatic energy. The deviatoric energy acts to resist the distortion of the solid whereas the hydrostatic term acts to resist the expansion and compression on the solid. As the model includes terms which resist volume changes, incompressibility becomes a hard constraint when modelling with a penalty term. However, in order to enforce this, the model would require infinite stiffness.

A. Implicit Integration

To handle these challenges, an implicit time-stepping framework is utilised for the latent space dynamics, used similarly by Fulton et al. [12] and Chang et al. [7] in minimising energy over time. The formulation for the configuration q at the end of the $(j+1)$th time step minimises the energy function as follows:

$$E(\mathbf{q}) = \int_\Omega \frac{1}{2h^2} \left\| \mathbf{W}(X)\mathbf{q} - \mathbf{u}_{\text{pred}} \right\|_g \text{dVol} + \int_\Omega \Psi(X + \mathbf{W}(X)\mathbf{q}) \, \text{dVol},$$

where h is the time-step, $\mathbf{u}_{\text{pred}}$ is the predicted position, $\Psi(x)$ represents the elastic energy density using the Neo-Hookean model and g is the kinetic energy norm.

The explicit position predictor at time step $(j+1)$ is given by

$$\mathbf{u}^{j+1}_{\text{pred}} = \mathbf{u}^j + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}},$$

where v_n is the full-space velocity computed using finite differences, given by

$$\mathbf{v}^n = \frac{u^n - u^{n-1}}{h} = \mathbf{W}(X)\frac{q^n - q^{n-1}}{h} = \mathbf{W}(X)\dot{\mathbf{q}}^n.$$

The integral over domain Ω in $E(q)$ can be approximated using cubature as

$$E(\mathbf{q}) \approx \sum_i \frac{w_i}{2h^2} \|\mathbf{W}(X_i)\mathbf{q} - \mathbf{u}_{\text{pred}}\|_g + \sum_i w_i \Psi(X_i + \mathbf{W}(X_i)\mathbf{q}).$$

B. Efficient Minimization Using Cubature Points

An et al. [1] establish an effective preprocess for cubature sampling which incrementally selects cubature sample points and assigns nonnegative weights to each selected point over a set of training poses or configurations of the system. These precomputed cubature points and weights are then used in the context of minimising the elastic energy function via gradient descent as described by Chang et al. [7].

The increment $\Delta\mathbf{u}_i$ at each cubature point i is initialised using an explicit time-stepping prediction, to be refined later through descent iterations

$$\Delta\mathbf{u}_i = h\mathbf{v}_j + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}.$$

At every iteration, the increment at each cubature point is updated by

$$\Delta\mathbf{u}_i = \alpha\left(\frac{M}{h^2}(\mathbf{W}(X_i)\mathbf{q} - \mathbf{q}_{\text{pred}}) + \frac{\partial\Psi(X_i + \mathbf{W}(X_i)\mathbf{q})}{\partial\mathbf{W}(X_i)\mathbf{q}}\right).$$

After computing the increments at all cubature points, the next step is to find the best-fit increment for the latent configuration. This involves solving a quadratic minimization problem, which aims to seeks to minimize the difference between the increments predicted by the model and the actual increments across all cubature points, weighted by w_i, as follows:

$$\Delta\mathbf{q} = \arg\min_{\Delta\mathbf{q}} \sum_i w_i \left\|\mathbf{W}(X_i)\Delta\mathbf{q} - \Delta\mathbf{u}_i\right\|^2 .$$

Since the system matrix depends only on the position and weights of the cubature points, the solution can be optimized by performing a Cholesky factorization of the system matrix once, and then solving subsequent linear systems via backsubstitution.

To improve efficiency, the function $\mathbf{W}(X_i)$ is cached at each cubature point, therefore the calculation is done once and reused until a new cubature point is introduced.

Note that a naive cubature sampling scheme is used as described by Chang et al. [7] for the purposes of accounting for remeshing of meshes unseen at the training phase.

C. Friction

Friction is the tangential resistive force arising from a sliding interface in a dynamical system [21] where two points or surfaces are in contact. In order to ensure proper contact handling, interpenetration must be resolved by projecting particles (points) along the collision normal by a distance d based on the non-penetration constraint defined as

$$C(x_i, x_j) = |x_{ij}| \geq 0,$$

where $|x_{ij}|$ represents the distance between particles i and j or between the particle and the surface with which it comes into contact. Note that the particles are assumed to be infinitesimally small.

When a contact is detected, Coulomb friction laws are applied to compute forces at each time step. I adapt the relevant force calculation scheme from [18].

The normal component of the contact force f_n is computed based on the penetration depth c and the normal stiffness k_e. The impulse j_n due to penetration is $j_n = c \cdot k_e$. Additionally, the damping impulse is calculated as $j_d = min(v \cdot n, 0) \cdot k_d$ where $v \cdot n$ is the normal component of the particle's velocity and k_d is the damping coefficient. The total contact force f_n is then $f_n = j_n + j_d$.

The friction impulse is applied in the tangent plane orthogonal to the contact normal [14]. Therefore, the friction force, f_t is determined based on the tangential velocity, v_t, which calculated by subtracting the component of velocity along the contact normal from the total velocity

$$v_t = v - n \cdot (v \cdot n).$$

Note that if the tangential velocity magnitude is greater than zero, it will be normalized to get the unit tangent vector.

Finally, f_t is constrained by the Coulomb friction model

$$f_t = min(\|v_t\| \cdot k_f, \mu \cdot |f_n|),$$

where k_f is the stiffness of friction force in soft contacts and μ is the coefficient of friction.

V. Results

Experiments were conducted for the purpose of evaluating the model when reproducing deformations on meshes seen during training under the influence of friction. The model was trained on multiple trajectories of the same mesh falling onto a plane of different slopes and different coefficients of friction from a short height.

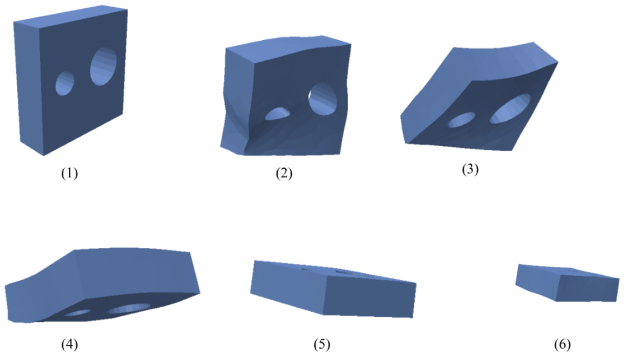

Fig. 2. *Full Space Simulation of the Cheese Mesh*. A cheese mesh is dropped on an invisible plane with friction coefficient 0.25 and plane $x+5y+2=-1.5$.

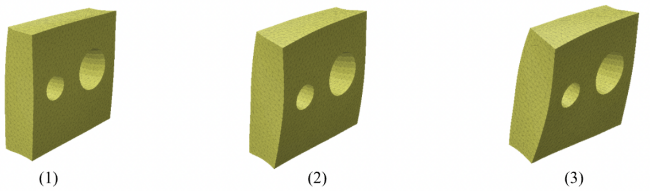

Fig. 3. *Reduced Space Simulation of the Cheese Mesh*. A cheese mesh is dropped on an invisible plane with friction coefficient 0.25 and plane $x+5y+2z=-1.5$.

Cheese Mesh. The full space simulation is given by Fig. 2. The model was trained on a total of 15 trajectories, simulated for 5 different coefficients of friction and 3 different planes. Fig. 3 shows the reduced space simulation generated at the online phase, which have not yielded accurate results.

Sphere Mesh. Similarly, the full space simulation of the sphere is given by Fig. 4 and the reduced space simulation generated is given by Fig. 5.

One possible reason for this discrepancy is the limited number of training epochs, which prevented the model from adequately capturing the complex interactions between friction and mesh deformation.

VI. Conclusion

In this study I aimed to investigate the operational dynamics of the Linear Subspace Continuous Reduced Order Modelling with Neural

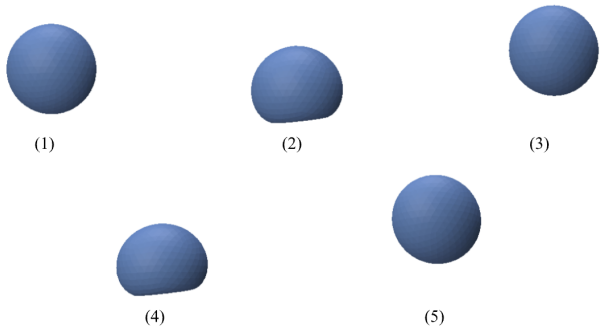

Fig. 4. *Full Space Simulation of the Sphere Mesh*. A sphere is bouncing on an invisible plane with friction coefficient 0.5 and some slope.

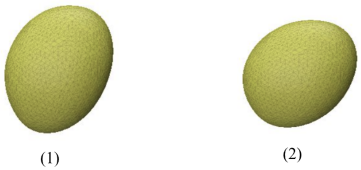

Fig. 5. *Reduced Space Simulation of the Sphere Mesh*. A sphere is bouncing on an invisible plane with friction coefficient 0.5 and some slope.

Fields (LiCROM) framework and examine scenarios where friction played a significant role in the simulation of an elastic object. The aim was to assess how incorporating frictional forces affects the deformation behavior predicted by LiCROM.

The results show a clear discrepancy between the behaviour predicted by the model trained for this paper accounting for friction and the original LiCROM model. This discrepancy is likely primarily due to the limitation of training the model for fewer epochs than those used in the original paper, due to time constraints. Consequently, this limitation likely impacted the model's performance and its ability to effectively capture the influence of friction on predicting the deformation behaviour.

An important area for improvement is hyperparameter tuning. Specifically, extending the training duration to match that of the original LiCROM framework could address the current discrepancies. A future exploration could include better tuning of hyperparameters to enhance the model's performance and provide more accurate results, which can be done by defining methods for sampling hyperparameter values and cross-validation. Additionally, the code runs on the CPU with *taichi*. GPU acceleration also remains a future research area which can allow for faster training times and thus allow for training the model on higher epochs.

Throughout the research process, several technical challenges were faced, including operating system limitations, complexities related to package dependencies of the original source code, and generating meshes and simulations of desired format during my reconstruction of the framework. While these issues were not directly related to my core research objectives, they were crucial in developing my technical skills and understanding of advanced modeling frameworks as a part of my COMP 400 objective.

Overall, this study underscores the need for continued refinement and optimization of the model, particularly in integrating frictional forces into the LiCROM framework. Future work should focus on extensive hyperparameter tuning, extended training and GPU acceleration to enhance the accuracy, applicability and speed of the model in predicting complex deformation behaviors.

VII. Acknowledgements

I would like to express my gratitude to Professor Paul Kry for his guidance, patience and support throughout my COMP 400 project.

I also thank the developers and community behind PyTorch, the Taichi programming language, TetWild, and NVIDIA Warp for facilitating this research with their tools and frameworks. Additionally, note that meshes were used from the Odedstein-Meshes repository [24].

References

[1] Steven An, Theodore Kim, and Doug James. Optimizing cubature for efficient integration of subspace deformations. *ACM transactions on graphics*, 27:165, 12 2009.

[2] David Baraff. Fast contact force computation for nonpenetrating rigid bodies. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques (SIGGRAPH '94)*, pages 23–34. ACM, 1994.

[3] David Baraff, Andrew Witkin, and Michael Kass. Untangling cloth. *ACM Transactions on Graphics*, 22, 09 2003.

[4] Jernej Barbič and Doug L. James. Real-time subspace integration for st.venant-kirchhoff deformable models. *ACM Transactions on Graphics (SIGGRAPH 2005)*, 24(3):982–990, August 2005.

[5] Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):483–531, 2015.

[6] R. Bridson, R. Fedkiw, and J. Anderson. Robust treatment of collisions, contact, and friction for cloth animation. In *ACM SIGGRAPH 2002*, pages 594–603, 2002.

[7] Yue Chang, Peter Yichen Chen, Zhecheng Wang, Maurizio M. Chiaramonte, Kevin Carlberg, and Eitan Grinspun. Licrom: Linear-subspace continuous reduced order modeling with neural fields, 2023.

[8] R. Qi Charles, Hao Su, Mo Kaichun, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, 2017.

[9] Peter Yichen Chen, Jinxu Xiang, Dong Heon Cho, Yue Chang, G A Pershing, Henrique Teles Maia, Maurizio M Chiaramonte, Kevin Thomas Carlberg, and Eitan Grinspun. CROM: Continuous reduced-order modeling of PDEs using implicit neural representations. In *The Eleventh International Conference on Learning Representations*, 2023.

[10] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *Under Review of ICLR2016 (1997)*, 11 2015.

[11] Kenny Erleben. Velocity-based shock propagation for multibody dynamics animation. *ACM Trans. Graph.*, 26, 06 2007.

[12] Lawson Fulton, Vismay Modi, David Duvenaud, David I. W. Levin, and Alec Jacobson. Latent-space dynamics for reduced deformable simulation. *Computer Graphics Forum*, 2019.

[13] Yixin Hu, Qingnan Zhou, Xifeng Gao, Alec Jacobson, Denis Zorin, and Daniele Panozzo. Tetrahedral meshing in the wild. *ACM Trans. Graph.*, 37(4):60:1–60:14, July 2018.

[14] Danny M. Kaufman, Shinjiro Sueda, Doug L. James, and Dinesh K. Pai. Staggered projections for frictional contact in multibody systems. In *ACM SIGGRAPH Asia 2008 Papers*, SIGGRAPH Asia '08, New York, NY, USA, 2008. Association for Computing Machinery.

[15] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[16] Kookjin Lee and Kevin Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *CoRR*, abs/1812.08373, 2018.

[17] Ran Luo, Tianjia Shao, Huamin Wang, Weiwei Xu, Kun Zhou, and Yin Yang. Deepwarp: Dnn-based nonlinear deformation. *CoRR*, abs/1803.09109, 2018.

[18] Miles Macklin. Warp: A high-performance python framework for gpu simulation and graphics. https://github.com/nvidia/warp, March 2022. NVIDIA GPU Technology Conference (GTC).

[19] Miles Macklin and Matthias Muller. A constraint-based formulation of stable neo-hookean materials. pages 1–7, 11 2021.

[20] Sara C. Schvartzman, Álvaro G. Pérez, and Miguel A. Otaduy. Star-contours for efficient hierarchical self-collision detection. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, New York, NY, USA, 2010. Association for Computing Machinery.

[21] Gang Sheng Chen and Xiandong Liu. Chapter 3 - friction. In Gang Sheng Chen and Xiandong Liu, editors, *Friction Dynamics*, pages 91–159. Woodhead Publishing, 2016.

[22] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020.

[23] Breannan Smith, Fernando Goes, and Theodore Kim. Stable neo-hookean flesh simulation. *ACM Transactions on Graphics*, 37:1–15, 03 2018.

[24] Oded Stein. odedstein-meshes: A computer graphics example mesh repository. 2024.

[25] Qingyang Tan, Zherong Pan, Lin Gao, and Dinesh Manocha. Realtime simulation of thin-shell deformable materials using cnn-based mesh embedding. *CoRR*, abs/1909.12354, 2019.

[26] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. *Computer Graphics*, 21(4):205–214, July 1987.

[27] Martin Wicke, Hermes Lanker, and Markus H. Gross. Untangling cloth with boundaries.

[28] D. Xiao, F. Fang, A.G. Buchan, C.C. Pain, I.M. Navon, and A. Muggeridge. Non-intrusive reduced order modelling of the navier–stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 293:522–541, 2015.

[29] Changxi Zheng and Doug L. James. Energy-based self-collision culling for arbitrary mesh deformations. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2012)*, 31(4), August 2012.