N° d'ordre NNT : 2026ECDL0011

THESE DE DOCTORAT DE L'ECOLE CENTRALE DE LYON

membre de l'Université de Lyon

Ecole Doctorale N°512
InfoMaths - Informatique et Mathématiques de Lyon

Spécialité de doctorat : Informatique

Soutenue publiquement le 24 avril 2026, par :

Ruochen CHEN

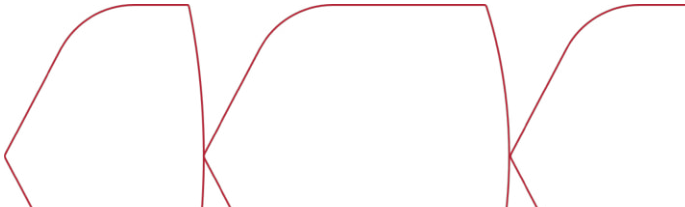
Simulating Continuous Physics on Discretized Neural Cloths : Introducing Constraints, Representations, and Operators to Bridge the Gap

---

Devant le jury composé de :

Hyewon SEO, Directrice de recherche, University of Strasbourg	Présidente
Vladislav GOLYANIK, Chercheur senior et chef d'équipe, Max Planck Institute for Informatics	Rapporteur
Di HUANG, Professeur des universités, Beihang University	Rapporteur
Liming CHEN, Professeur des universités, Ecole Centrale de Lyon	Directeur de thèse
Shaifali PARASHAR, Chargée de recherche, LIRIS, INSA-Lyon	Co-encadrante de thèse

Unité de recherche : LIRIS - Laboratoire d'InfoRmatique en Image et Systèmes d'information



Abstract

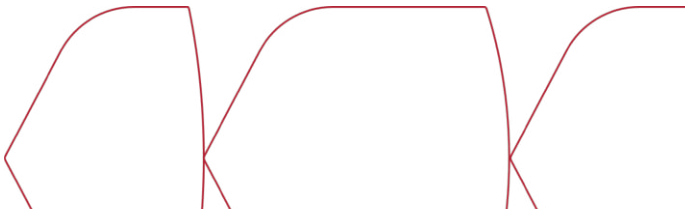
Modeling the deformation of surfaces, particularly garments and cloth, is a foundational task across computer vision, computer graphics, and robotics. Despite significant recent advancements, a fundamental gap remains between the continuous nature of physical fabrics and the discrete structures used in digital modeling and learning systems. Traditional physics-based simulators provide high accuracy but are computationally expensive, while emerging data-driven neural surrogates have yet to fully overcome the challenges posed by this discretization gap. This thesis aims to address these challenges by introducing novel constraints, representations, and operators, establishing a comprehensive framework to bridge the gap between continuous physics and discretized neural cloths.

To this end, we identify three core challenges and propose three corresponding contributions. We first introduce GAPS, a geometry-aware, physics-based, self-supervised neural garment draping framework. GAPS enforces inextensibility through locally computed covariance-based measures while adaptively relaxing constraints in collision regions, yielding stable and realistic draping without expensive post-processing, complemented by an RBF-based skinning that improves robustness for loose garments. Second, we propose PolyFit, a continuous and differentiable surface representation based on local polynomial n-jet functions that models each surface patch with a compact set of coefficients, substantially reducing dimensionality while providing closed-form derivatives of arbitrary order. We demonstrate its effectiveness through two applications: PolySfT, a learning-free method for monocular 3D surface reconstruction, and OneFit, a self-supervised neural draping model that predicts garment deformation directly in the functional coefficient space, achieving garment-agnostic and resolution-agnostic generalization while being up to an order of magnitude faster than existing baselines. Finally, we present FNOpt, a self-supervised cloth simulation framework that addresses the resolution dependence of conventional neural simulators. By meta-learning a neural optimizer parameterized by a Fourier Neural Operator, FNOpt operates in the spectral domain and learns dynamics between function spaces, making it naturally resolution-agnostic. This enables zero-shot super-resolution: a model trained solely on coarse meshes can produce

high-fidelity simulations at fine resolutions, recovering wrinkles and geometric details absent from the training data.

These technical and methodological contributions are intended to improve the robustness, fidelity, and generalization capability of neural garment modeling, and we hope they represent a meaningful step toward more generic, scalable, and efficient simulation of deformable objects.

Keywords: Cloth simulation, Garment draping, Self-supervised learning, Surface representation, Neural operators, Deformable surfaces.



Résumé

La modélisation de la déformation des surfaces, en particulier des vêtements et des tissus, constitue une tâche fondamentale en vision par ordinateur, en informatique graphique et en robotique. Malgré des avancées récentes significatives, un écart fondamental persiste entre la nature continue des tissus physiques et les structures discrètes utilisées dans les systèmes de modélisation et d'apprentissage numériques. Les simulateurs traditionnels fondés sur la physique offrent une grande précision mais sont coûteux en temps de calcul, tandis que les substituts neuronaux émergents guidés par les données n'ont pas encore pleinement surmonté les défis posés par cet écart de discrétisation. Cette thèse vise à répondre à ces défis en introduisant de nouvelles contraintes, représentations et opérateurs, établissant un cadre complet pour combler le fossé entre la physique continue et les simulations neuronales discrétisées de tissus.

À cette fin, nous identifions trois défis principaux et proposons trois contributions correspondantes. Nous introduisons d'abord GAPS, un cadre de drapage neuronal auto-supervisé de vêtements, tenant compte de la géométrie et fondé sur la physique. GAPS impose l'inextensibilité par des mesures locales fondées sur la covariance tout en relâchant de manière adaptative les contraintes dans les régions de collision, produisant un drapage stable et réaliste sans post-traitement coûteux, complété par un habillage (skinning) à base de RBF qui améliore la robustesse pour les vêtements amples. Ensuite, nous proposons PolyFit, une représentation de surface continue et différentiable fondée sur des fonctions polynomiales locales de type n-jet, qui modélise chaque patch de surface par un ensemble compact de coefficients, réduisant considérablement la dimensionnalité tout en fournissant des dérivées analytiques de tout ordre. Nous démontrons son efficacité à travers deux applications : PolySfT, une méthode sans apprentissage pour la reconstruction monoculaire de surfaces 3D, et OneFit, un modèle de drapage neuronal auto-supervisé qui prédit la déformation des vêtements directement dans l'espace des coefficients fonctionnels, atteignant une généralisation indépendante du vêtement et de la résolution tout en étant jusqu'à un ordre de grandeur plus rapide que les méthodes existantes. Enfin, nous présentons FNOpt, un cadre de simulation de tissus auto-supervisé qui traite la dépendance à la résolution des simulateurs neuronaux conventionnels. En méta-apprenant un

optimiseur neuronal paramétré par un opérateur neuronal de Fourier, FNOpt opère dans le domaine spectral et apprend la dynamique entre espaces fonctionnels, le rendant naturellement indépendant de la résolution. Cela permet une super-résolution sans apprentissage supplémentaire : un modèle entraîné uniquement sur des maillages grossiers peut produire des simulations haute fidélité à des résolutions fines, retrouvant des plis et des détails géométriques absents des données d'entraînement.

Ces contributions techniques et méthodologiques visent à améliorer la robustesse, la fidélité et la capacité de généralisation de la modélisation neuronale de vêtements, et nous espérons qu'elles représentent un pas significatif vers une simulation plus générique, extensible et efficace des objets déformables.

Mots-clés : Simulation de tissus, Drapage de vêtements, Apprentissage auto-supervisé, Représentation de surfaces, Opérateurs neuronaux, Surfaces déformables.

Acknowledgments

I would like to express my sincere gratitude to all those who have supported me throughout this journey.

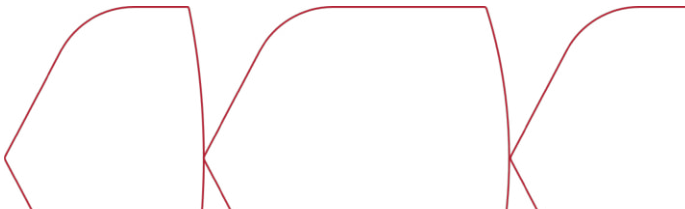
Foremost, I am deeply grateful to my supervisors, Shaifali Parashar and Liming Chen, for giving me the opportunity to undertake this thesis project and for their invaluable guidance, unwavering patience, and continuous encouragement throughout this research. Their expertise and insightful feedback have been instrumental in shaping this work and my development as a researcher.

I would like to extend my gratitude to my thesis jury members, Hyewon Seo, Vladislav Golyanik, and Di Huang, for graciously accepting to dedicate their valuable time and expertise to the evaluation of this work.

I would also like to thank the members and former members of the LIRIS at École Centrale de Lyon, who have been both great friends and colleagues, in alphabetical order, Alexandre Chapin, Guillaume Duret, Zezeng Li, Mu Liang, Liqun Liu, Bruno Machado, Léo Schneider, Rui Yang, Shanthika Naik, Ziqian Zhang; with a special thank to Thuy Tran, which has been my closest collaborator and friend during the last two years; Boulbaba Ben Amor and Gilles Gesquière as jury members of my Individual Follow-up Committee (CST); Emmanuel Dellandréa for organizing team seminars; Rahul Narain for the discussion on the latest project. Both the formal research exchanges and informal conversations have enriched my understanding and made this journey more enjoyable.

There are also other researchers or students with whom I had the chance to interact and that I would like to thank here: Adrien Bartoli, Bourgoin Mickael, Plihon Nicolas, François Schweitzer, Ziyi Chen, Lipeng Cai, Xun Gong.

Finally, I would like to thank my parents and my grandparents for their unconditional love and support during these three years.



Contents

Abstract **3**

Résumé **5**

Acknowledgments **7**

List of Figures **13**

List of Tables **15**

Acronyms **17**

Notations **19**

1 Introduction **21**
1.1 General Context 21
1.2 Key Challenges 23
1.2.1 Challenge 1: Enforcing Physical Realism in Neural Simulation 23
1.2.2 Challenge 2: Scalable Continuous Representation 24
1.2.3 Challenge 3: Resolution-Agnostic Learning 24
1.3 Contributions 25
1.4 Publications 27
1.5 Outline 28

2 Literature Review **31**
2.1 Introduction 31
2.2 Cloth and Garment Simulation 31
2.2.1 Physics-Based Cloth Simulation 32
2.2.2 Supervised Cloth Simulation 33
2.2.3 Self-Supervised Neural Simulation 35
2.3 Surface Representations for Deformable Geometry 38
2.3.1 Explicit Surface Representations 38

2.3.2 Implicit Representations 39
2.3.3 Radiance Fields and Volumetric Splatting 40
2.3.4 Application to Garment Modeling 40
2.4 Neural Operators and Resolution-Agnostic Learning 42
2.5 Template-Based 3D Reconstruction 44
2.5.1 Deformation priors 45
2.5.2 Solving Strategies 46

3 Technical Background 49
3.1 Introduction 49
3.2 Surface Elasticity and Deformation Theory 50
3.2.1 Continuous Formulation 50
3.2.2 Discretization Strategies 53
3.2.3 Material Models and Constitutive Laws 54
3.3 Parametric Human Body Modeling 55
3.4 Cloth Simulation as Optimization 57
3.4.1 Time Integration 57
3.4.2 Optimization-Based Reformulation 58
3.4.3 Collision Handling 59
3.5 Fourier Neural Operators Formulation 60

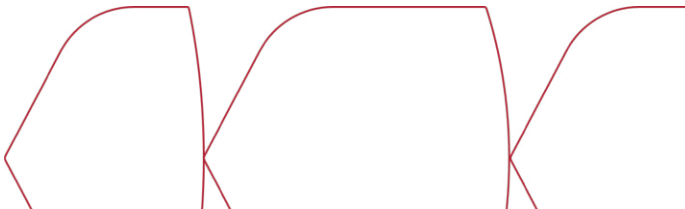
4 GAPS: Geometry-Aware, Physics-Based, Self-Supervised Neural Garment Draping 65
4.1 Introduction 66
4.2 Geometry-Aware Modeling 68
4.2.1 Garment Deformation Modeling 68
4.2.2 Body-Participation In Garment Dynamics 70
4.3 Method 71
4.3.1 Model 72
4.3.2 Losses 73
4.4 Experiments 75
4.5 Conclusion 81

5 Patch-based Representation and Learning for Efficient Deformation Modeling 83

5.1 Introduction . . . 84
5.2 PolyFit . . . 85
5.3 PolySfT . . . 87
5.4 OneFit . . . 87
5.5 Experiments . . . 90
5.5.1 PolyFit . . . 90
5.5.2 PolySfT . . . 94
5.5.3 OneFit . . . 97
5.6 Conclusions . . . 101

6 FNOpt: Resolution-Agnostic, Self-Supervised Cloth Simulation using Meta-Optimization with Fourier Neural Operators 107
6.1 Introduction . . . 108
6.2 Simulating Cloth via FNOpt . . . 109
6.2.1 Mathematical Foundation . . . 109
6.2.2 Training FNO . . . 110
6.2.3 Training Neural Optimizer via Meta-learning . . . 111
6.2.4 Inference . . . 112
6.3 Experiments . . . 113
6.3.1 SOTA Comparison . . . 114
6.3.2 Boundary Condition Generalization . . . 116
6.3.3 Ablation Studies . . . 118
6.4 Simulation under Varying Material Coefficients . . . 130
6.5 Conclusion . . . 130

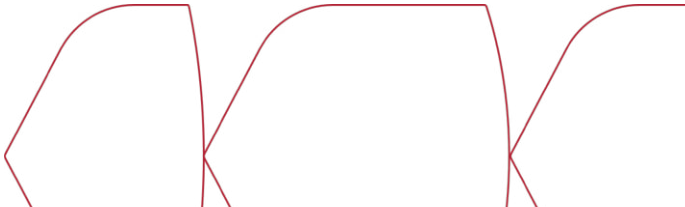
7 Conclusion 133
7.1 Summary of contributions . . . 133
7.2 Future perspectives . . . 134



List of Figures

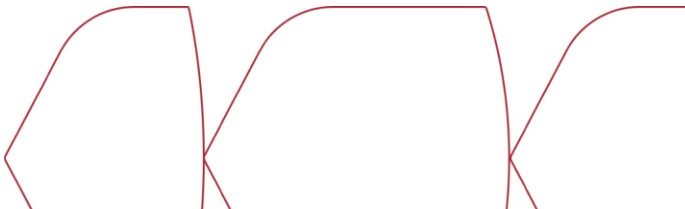
1.1 Some applications of digital garments modeling . . . 22
1.2 Contribution 1 . . . 25
1.3 Contribution 2 . . . 26
1.4 Contribution 3 . . . 27
2.1 Comparison of neural networks and neural operators . . . 43
2.2 Applications of FNO . . . 44
3.1 SMPL parametric human model . . . 56
3.2 Fourier Neural Operator architecture . . . 61
4.1 GAPS vs SOTA methods in terms of stretching . . . 66
4.2 Body-participation computation . . . 70
4.3 GAPS method overview . . . 72
4.4 GAPS draping results on various garments . . . 75
4.5 GAPS results on dress draping . . . 77
4.6 GAPS generalization to diverse bodies . . . 78
4.7 GAPS comparison with PBS on dress . . . 79
4.8 GAPS RBF-based skinning comparison . . . 80
4.9 RBF-based skinning ablation . . . 80
5.1 Patch-based representation and learning . . . 84
5.2 PolyFit and PolySfT illustration . . . 86
5.3 OneFit pipeline . . . 88
5.4 STN canonicalization effect . . . 92
5.5 Jet order fitting error . . . 93
5.6 PolySfT reconstruction results . . . 96
5.12 Single vs multi-garment OneFit . . . 99
5.7 Error maps on phi-SfT dataset . . . 102
5.8 Error maps DeepSfT vs ours . . . 103
5.9 PolySfT stability test . . . 103
5.10 OneFit intra-class generalization . . . 104
5.11 OneFit resolution-agnostic results . . . 104
5.13 OneFit vs SOTA on dress . . . 105

5.14 OneFit vs SOTA on tight garments . . . 105
5.15 OneFit on challenging poses . . . 105
6.1 FNOpt generalization overview . . . 108
6.2 FNOpt pipeline . . . 110
6.3 Error maps at training resolution . . . 119
6.4 Error maps at finer resolutions . . . 120
6.5 Per-frame error progression . . . 121
6.6 Speed generalization . . . 121
6.7 Handle placement generalization . . . 122
6.8 Non-square mesh generalization . . . 124
6.9 Optimizer runtime comparison . . . 125
6.10 Optimizer convergence comparison . . . 125
6.11 Error comparison with MGN . . . 126
6.12 Upsampling method comparison . . . 127
6.13 Runtime vs iterations trade-off . . . 128
6.14 Repulsive loss ablation . . . 129
6.15 Material coefficient variations . . . 130



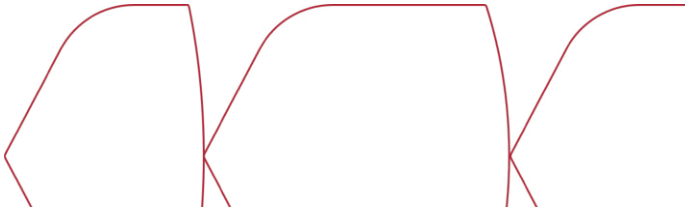
List of Tables

2.1 Overview of clothed human datasets . . . 36
4.1 GAPS inextensibility and collision metrics . . . 76
4.2 Collision-awareness ablation study . . . 78
4.3 Inextensibility loss ablation study . . . 78
4.4 GAPS timing performance . . . 81
5.1 Patch fitting chamfer distance . . . 92
5.2 Patch fitting chamfer distance . . . 93
5.3 PolyFit fitting metrics . . . 94
5.4 Comparison with point-based encoders . . . 94
5.5 PolyFit training data study . . . 94
5.6 Reconstruction results on Kinect-Paper . . . 95
5.7 Phi-SfT quantitative comparison . . . 96
5.8 Multi-garment collision metrics . . . 100
5.9 OneFit fine-tuning vs training . . . 100
5.10 OneFit ablation study . . . 100
5.11 OneFit timing performance . . . 101
6.1 Resolution generalization metrics . . . 117
6.2 3D error comparison across resolutions . . . 118
6.3 Iteration count ablation . . . 123
6.4 FNOpt ablation study . . . 127
6.5 Repulsive loss ablation study . . . 129
6.6 Optimizer comparison at varying iterations . . . 131
6.7 Iteration budget ablation study . . . 131
6.8 Per-sequence iteration budget comparison . . . 132



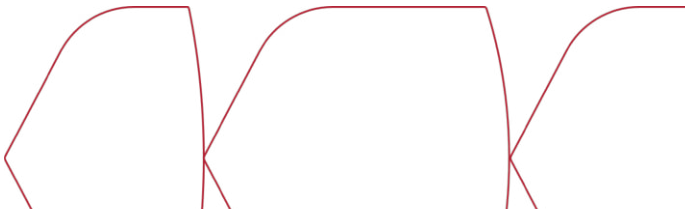
Acronyms

ACVD	Approximated Centroidal Voronoi Diagrams
ARAP	As-Rigid-As-Possible
CNN	Convolutional Neural Network
CPU	Central Processing Unit
FEM	Finite Element Method
FFT	Fast Fourier Transform
FNO	Fourier Neural Operator
GNN	Graph Neural Network
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
LBS	Linear Blend Skinning
L-BFGS	Limited-memory Broyden–Fletcher–Goldfarb–Shanno Algorithm
MLP	Multilayer Perceptron
ODE	Ordinary Differential Equation
PBD	Position-Based Dynamics
PBS	Physics-Based Simulation
PCA	Principal Component Analysis
PDE	Partial Differential Equation
RBF	Radial Basis Function
SDF	Signed Distance Field
SfT	Shape-from-Template
SMPL	Skinned Multi-Person Linear Model
StVK	Saint Venant–Kirchhoff
UDF	Unsigned Distance Field
UV	2D Parametric Coordinates (u, v)
VTO	Virtual Try-On



Notations

Symbol	Description	Defined
$\mathcal{M}$	Surface manifold	§3.2.1
V, E, F	Mesh vertices, edges, and faces	§3.2.2
$\mathcal{B}$	Body vertices set	§4.2.2
$\mathcal{G}$	Garment vertices set	§4.2.2
$\mathbf{x}_i$	Vertex position in 3D space, $\mathbf{x}_i \in \mathbb{R}^3$	§3.2.2
$\mathbf{n}_i$	Normal vector at vertex or patch i	§3.2.1
$\mathbf{G}$	First fundamental form (metric tensor)	§3.2.1
$\mathbf{B}$	Second fundamental form (curvature tensor)	§3.2.1
$\mathbf{F}$	Deformation gradient	§3.2.1
$\mathbf{E}$	Green–Lagrange strain tensor	§3.2.1
H, K	Mean and Gaussian curvature	§3.2.1
E_Y	Young's modulus	§3.2.3
ν	Poisson's ratio	§3.2.3
μ, λ	Lamé coefficients	§3.2.3
$\mathcal{E}$	Total system energy	§3.2.1
Δt	Time step size	§3.4.1
$\mathbf{x}, \mathbf{v}, \mathbf{a}$	Position, velocity, and acceleration vectors	§3.4
$\mathcal{P}$	Participation matrix (body-garment interaction)	§4.2.2
$\mathcal{W}$	Garment blend weights	§4.2.2
$\mathcal{L}_{\text{cloth}}$	Cloth physics loss (stretching, bending, collision)	§4.3.2
e_{CD}	Chamfer distance error metric	§6.3.1
e_{3D}	3D vertex position error	§6.3.1



CENTRALE LYON

LIRIS

CHAPTER 1

Introduction

1.1 General Context

Modeling the deformation of surfaces lies at the core of many applications in computer vision, computer graphics, and robotics. From virtual try-on, sports analysis and character animation to soft robot manipulation and deformable tissue modeling for surgery, the ability to accurately represent and predict how surfaces deform under external forces is essential for achieving realism, reliability, and physical plausibility.

In recent years, the demand for high-fidelity digital representations of deformable surfaces has increased rapidly across domains. In virtual try-on and e-commerce, realistic simulation of garment draping on diverse body shapes is crucial for ensuring consistency between virtual previews and physical products. In medical imaging and minimally invasive surgery, accurate three-dimensional reconstruction and deformation modeling of soft organs enables precise localization, planning, and intervention. In digital entertainment and sports analysis, physically plausible cloth and body motion enhances visual realism while supporting biomechanical understanding. Similarly, in robotics, tasks such as fabric folding, dressing assistance, and manipulation of flexible objects fundamentally rely on robust models of surface deformation.

Among these applications, cloth and garment deformation poses particularly unique and challenging problems. Most fabrics are elastic materials that tend to deform near-isometrically, bending easily while resisting in-plane stretch, but body shape and contact constraints can force them to stretch or compress in specific regions. As a result, garments exhibit complex, strongly anisotropic and nonlinear behavior, while involving high-dimensional state spaces due to fine geometric detail.

In virtual try-on and e-commerce, this challenge is driven by the growing ex-

pectation of visual fidelity and physical consistency. Traditional two-dimensional overlay techniques are increasingly replaced by three-dimensional simulations that allow full-view inspection and dynamic interaction. In character animation and video games, realistic cloth dynamics significantly contribute to perceived realism: garments are expected to respond naturally to body motion rather than behaving as rigid shells. Achieving this realism requires simulation methods that balance physical accuracy and computational efficiency.

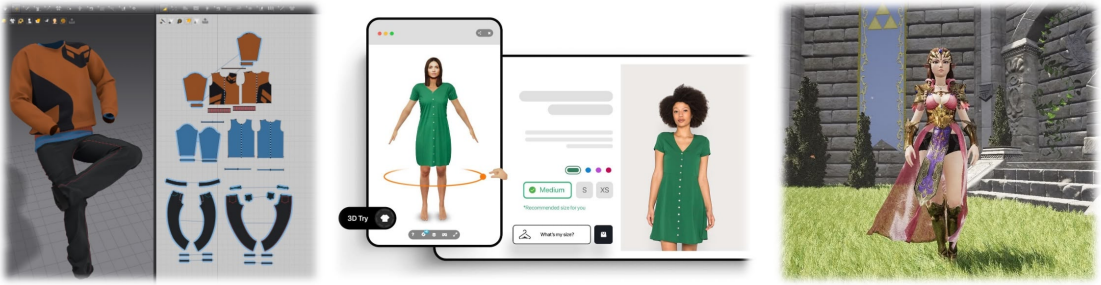


FIGURE 1.1 : Some applications of digital garments modeling. Left: fashion design; Middle: virtual try-on; Right: game characters.

Behind these applications lies a fundamental challenge: the mismatch between the continuous nature of the physical world and the discrete nature of digital representations. To make simulation tractable, fabric is traditionally modeled as a continuous thin shell governed by continuum mechanics and Partial Differential Equations (PDEs). Its deformation is determined not only by intrinsic material properties, ranging from nearly inextensible (close to isometric) to highly elastic, but also by environmental factors such as external forces and boundary constraints imposed by the body. This interplay between material behavior and environmental conditions makes fabric deformation inherently complex and difficult to model. In contrast, digital systems rely on discrete structures such as polygonal meshes or point clouds to process, learn, and render these surfaces.

This discrepancy makes it difficult for discrete representations to faithfully preserve continuous physical properties, such as remaining inextensible during body-garment interactions while exhibiting controlled, uniform extension when fitting to larger bodies or under other boundary constraints, without introducing numerical artifacts. Moreover, the computational cost of mesh-based simulation grows rapidly with increasing resolution, limiting scalability in practical applications.

Furthermore, while deep learning offers a promising avenue for accelerating surface deformation modeling, most existing data-driven approaches inherit the limitations of the discrete representations on which they operate. These methods typically learn mappings between fixed-dimensional discrete spaces, rather than capturing the underlying continuous physical laws. As a consequence, they often struggle to generalize across resolutions, mesh topologies, and material configurations.

This leads to the central question of the thesis: *How can we bridge the gap between continuous physical laws and discrete neural simulation to achieve physically plausible, scalable, and resolution-agnostic cloth and garment simulation?*

To address this question, this thesis focuses on bridging the gap between continuous surface dynamics and discrete digital representations, with garment simulation as the primary application domain. Motivated by the limitations of discrete representations discussed above, we investigate new constraints, alternative representations and learning-based operators to better support physical realism, generalization, and efficiency in modern garment and cloth simulation pipelines.

While our main focus is on garment simulation, the proposed contributions are directly applicable to broader 3D deformation modeling tasks. For instance, we demonstrate applications in 3D surface reconstruction from monocular images, where similar challenges of preserving geometric properties arise.

1.2 Key Challenges

We identify three key challenges in building physically plausible, scalable, and generalizable neural cloth simulation systems.

1.2.1 Challenge 1: Enforcing Physical Realism in Neural Simulation

The first challenge concerns maintaining physical plausibility in learning-based garment simulation. In the continuous physical world, cloth deformation exhibits varying degrees of elasticity depending on material properties. For typical garment fabrics, cloth tends to deform nearly isometrically, bending easily but resisting in-plane stretching. External forces (such as gravity or tension) and geometric constraints (such as collision or body contact) can induce controlled local deformation.

However, neural simulation methods often produce non-physical results with implausible stretching, particularly when the draped garment penetrates the body.

While existing physics-based losses such as St. Venant-Kirchhoff (StVK) strain energy enforce isometry through the metric tensor (first fundamental form), they typically apply uniformly across the entire mesh. This uniform enforcement creates a fundamental trade-off: strictly enforcing inextensibility leads to locking and overly stiff behavior during collision resolution, while relaxing the constraint permits widespread nonphysical stretching. Simple pairwise distance constraints (e.g., mass-spring models) applied in learning frameworks tend to produce local artifacts or require expensive post-processing.

What is needed is an adaptive, collision-aware constraint that intelligently balances isometry preservation and collision resolution—maintaining strict inextensibility where geometrically feasible while permitting controlled stretching only where necessary to resolve body-garment penetration.

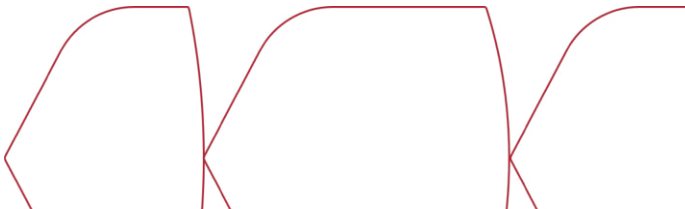
1.2.2 Challenge 2: Scalable Continuous Representation

The second challenge comes from the discrete mesh representation itself. Encoding a surface as vertices (V, E, F) links the complexity of the model directly to the sampling density. As higher visual quality is demanded, vertex counts rise, dramatically increasing the dimensionality of the deformation problem.

Moreover, discrete representations lack closed-form derivatives. Computing higher-order quantities typically relies on numerical finite differences, which are sensitive to noise and mesh irregularity. There is a need for a compact, analytical representation that can express complex surface deformations with far fewer parameters while providing analytic derivatives. Without such a representation, neural models remain tied to high-dimensional output spaces, limiting scalability and efficiency.

1.2.3 Challenge 3: Resolution-Agnostic Learning

The third challenge is the strong spatial resolution dependence of current neural simulators. Deep learning models such as Convolutional Neural Networks (CNNs) or Graph Neural Networks (GNNs) typically learn mappings defined on specific discretizations. A model trained on a coarse mesh learns dynamics tied to that discretization. To produce high-fidelity simulations at fine-resolution, one must usually train on expensive high-resolution data, which limits the deployment of neural simulators in multi-resolution environments.



This severely limits generalization. When applied to a high-resolution mesh to recover fine wrinkles, such models often fail or produce overly smooth results. They learn discrete behaviors of specific mesh nodes rather than the underlying continuous operator. What is missing is a method that is resolution-agnostic, which learns mappings in function spaces rather than between discrete vectors, enabling models trained on coarse data to generalize to fine meshes in a zero-shot manner.

1.3 Contributions

This thesis proposes three contributions to address these challenges, progressing from enforcing physical realism, to enabling scalable representation, and finally to achieving resolution-independent learning.

Contribution 1: GAPS: Geometry-aware Constraint for Neural Garment Simulation.

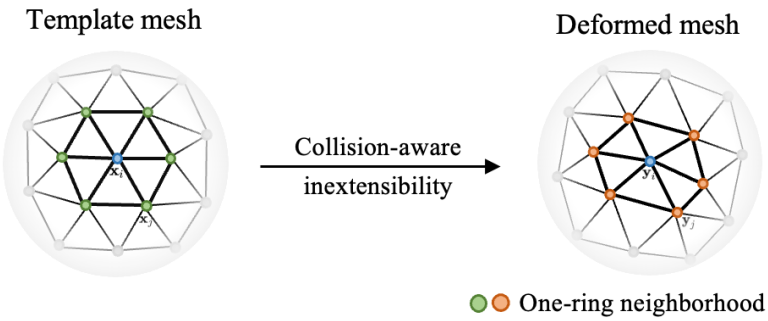


FIGURE 1.2 : Covariance-based local inextensibility constraint. We enforce isometry by preserving the singular values of the local covariance matrix over one-ring neighborhoods and enabling adaptive relaxation in collision regions.

We address the physical realism challenge by introducing an adaptive, collision-aware inextensibility constraint for self-supervised neural garment draping. Unlike existing methods that uniformly enforce isometry (e.g., StVK strain energy) and suffer from unrealistic stretching during collision resolution, GAPS adaptively modulates constraint strength based on local body-garment penetration depth. In collision-free regions, the constraint strictly preserves inextensibility; in penetration zones, it progressively relaxes, permitting controlled stretching only where geometrically necessary to resolve body-garment interference.

Technically, we achieve this through a covariance-based geometric formulation that preserves local rigidity by enforcing the preservation of singular values of covariance matrices over one-ring neighborhoods. We further introduce an RBF-based geometry-aware skinning method that computes body-garment participation weights from local distance fields, robustly handling both tight-fitting and loose garments without manual parameter tuning. We further validated the adaptive constraint in monocular shape-from-template reconstruction Tran et al. [2025], demonstrating its broader applicability beyond neural simulation.

Contribution 2: Continuous Analytical Representation from Discrete Data.

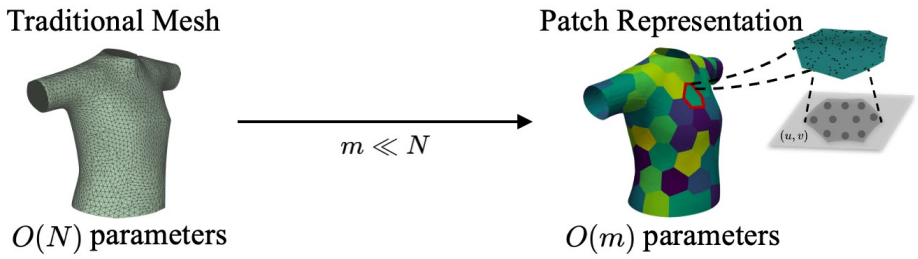


FIGURE 1.3 : From mesh to analytical patches. A dense mesh stores $O(N)$ vertex coordinates, while our PolyFit representation uses m polynomial patches with compact coefficients, reducing parameters to $O(m)$ where $m \ll N$.

To overcome the scalability limitations inherent in mesh-based formulations, we introduce PolyFit, a continuous and differentiable surface representation based on local n-jet functions. PolyFit models each surface patch with a compact set of polynomial coefficients, providing an analytical approximation of the underlying discrete geometry, including first- and higher-order derivatives.

This functional representation replaces thousands of vertex coordinates with a lightweight set of coefficients, dramatically reducing dimensionality while enabling fast, scalable, and resolution-independent computation. We demonstrate its versatility through two downstream tasks: PolySfT, which performs monocular 3D reconstruction using the analytical patch structure, and OneFit, which learns garment deformation directly in the functional space. In OneFit, this representation yields garment-agnostic and resolution-agnostic generalization, while offering substantial improvements in computational efficiency.

Contribution 3: Learning a Resolution-Agnostic Dynamics Operator.

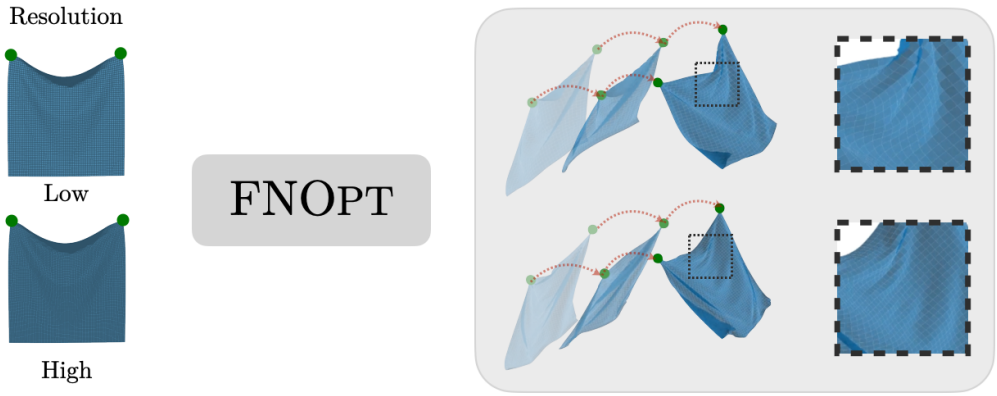


FIGURE 1.4 : Resolution-agnostic learning via Fourier Neural Operator. FNOpt learns dynamics in the spectral domain, enabling zero-shot super-resolution: trained on coarse meshes (low resolution), it generalizes to fine meshes (high resolution) and recovers high-frequency wrinkles absent from training data.

Finally, we address the resolution-dependency of neural simulators with FNOPT. Instead of learning a direct state-to-acceleration mapping tied to a specific mesh resolution, we meta-learn a neural optimizer parameterized by a Fourier Neural Operator (FNO). Although training losses are computed on discretized meshes, the FNO backbone operates in the spectral domain, learning a mapping between function spaces that is naturally resolution-independent. As a result, a model trained only on coarse meshes can perform zero-shot super-resolution: when evaluated on fine meshes, it produces realistic high-frequency wrinkles that were not explicitly present in the training data. FNOPT demonstrates that by learning optimization dynamics in continuous function space, we can largely decouple simulation fidelity from training resolution, representing a major step toward truly resolution-agnostic learned simulators.

1.4 Publications

Here is the list of publications produced during this thesis:

First-Author Publications

- Ruochen Chen, Liming Chen, and Shaifali Parashar. GAPS: Geometry-aware, physics-based, self-supervised neural garment draping. In *International Confe-*

rence on 3D Vision (3DV), 2024

Contributions: **R. Chen** developed the RBF-based skinning methodology, implemented the full framework, conducted the experiments, and wrote the first draft of the manuscript.

- Ruochen Chen, Thuy Tran, and Shaifali Parashar. Patch-based representation and learning for efficient deformation modeling. In *International Conference on 3D Vision (3DV)*, 2026

  Contributions: **R. Chen** conceptualized the learning frameworks of PolyFit, PolySfT, and OneFit, conducted the experiments, and wrote the first draft of the manuscript. **T. Tran** assisted in developing the fine details of the pipeline and provided feedback on the demonstration of results.

- Ruochen Chen*, Thuy Tran*, and Shaifali Parashar. FNOpt: Resolution-agnostic, self-supervised cloth simulation using meta-optimization with fourier neural operators. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2026

  Contributions: **R. Chen** and **T. Tran** contributed equally to this work. **R. Chen** developed the methodology, conducted the experiments, and contributed to the paper writing. **T. Tran** conceptualized the use of Fourier Neural Operators for resolution-agnostic cloth learning and wrote the methodology section.

Co-authored Publications

- Thuy Tran, Ruochen Chen, and Shaifali Parashar. Image-guided shape-from-template using mesh inextensibility constraints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025

  Contributions: **T. Tran** conceptualized the frame-wise approach, implemented the architecture, and performed the main evaluations. **R. Chen** developed the window-based approach and assisted in conducting a subset of the experiments.

1.5 Outline

This thesis is organized as follows:

* Equal contribution.

Section 2 surveys the relevant literature, covering cloth simulation methods, surface representations for deformable geometry, neural operators for resolution-agnostic learning, and template-based 3D reconstruction.

Section 3 then establishes the mathematical and computational foundations: surface elasticity and deformation theory, parametric human body modeling, the optimization-based formulation of implicit time integration, and the PDE perspective on cloth dynamics.

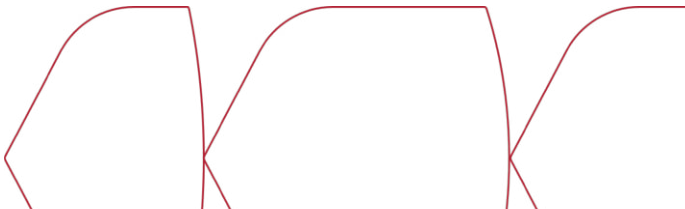
Section 4 addresses the first challenge identified in this thesis: enforcing physical realism in neural garment simulation. We introduce GAPS (Geometry-Aware, Physics-based, Self-supervised Neural Garment Draping), which improves upon existing self-supervised neural simulators by incorporating a collision-aware inextensibility loss. Unlike uniform constraint enforcement methods (e.g., StVK) that suffer from locking, this adaptive loss employs local covariance matrices to preserve isometry in collision-free regions while progressively relaxing in penetration zones, allowing garments to stretch only when geometrically necessary. We further propose an RBF-based geometry-aware skinning method that robustly handles both tight-fitting and loose garments. Experimental results demonstrate that GAPS produces physically plausible draping without expensive collision post-processing.

Section 5 tackles the second challenge: scalable continuous representation. We propose PolyFit, a continuous analytical surface representation based on local polynomial n-jet functions. By approximating each surface patch with a compact set of coefficients, PolyFit reduces dimensionality while providing closed-form derivatives of arbitrary order. We demonstrate its effectiveness through two applications: PolySfT, a learning-free monocular 3D reconstruction method, and OneFit, a garment-agnostic neural draping model that predicts deformation in the functional coefficient space rather than vertex positions. OneFit achieves cross-topology and cross-resolution generalization, while being an order of magnitude faster than baseline methods.

Section 6 addresses the third challenge: resolution-agnostic learning. We introduce FNOpt, a meta-learned neural optimizer parameterized by a Fourier Neural Operator. Unlike conventional neural simulators that learn discrete mappings tied to specific mesh resolutions, FNOpt operates in the spectral domain and learns dynamics between function spaces. This enables zero-shot super-resolution: a model trained only on coarse meshes can produce high-fidelity simulations on fine meshes, recovering wrinkles and details absent from the training data. We validate FNOpt's resolution-

agnostic generalization and superior performance against state-of-the-art baselines across multiple mesh resolutions.

Finally, Section 7 concludes the thesis by synthesizing our key findings and discussing their broader implications for neural surface modeling and garment simulation. It also identifies current limitations and outlines promising directions for future research, including removing skinning dependencies, handling topology changes, and extending to multi-layer clothing systems.



CENTRALE LYON

LIRIS

CHAPTER 2

Literature Review

2.1 Introduction

This chapter surveys the existing work most closely related to the contributions of this thesis. Each subsection reviews a specific research area, positions the prior art with respect to its strengths and limitations, and concludes with a paragraph that clarifies how our contributions differ from or build upon these works.

Section 2.2 reviews the evolution of cloth simulation from classical physics-based methods to modern learning-based approaches, covering numerical simulation methods, supervised neural simulators, self-supervised frameworks, and collision handling techniques.

Section 2.3 examines surface representations for deformable geometry modeling, comparing explicit meshes, point clouds, implicit neural fields, and parametric patch-based formulations.

Section 2.4 discusses neural operators for learning partial differential equations, with particular emphasis on Fourier Neural Operators and their resolution-agnostic properties.

Section 2.5 surveys template-based 3D reconstruction (Shape-from-Template) methods, categorizing them by deformation priors and solution strategies.

2.2 Cloth and Garment Simulation

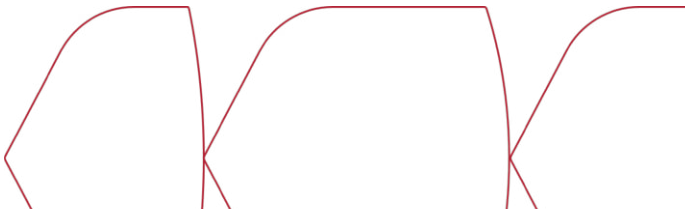
This section reviews the evolution of cloth simulation from classical physics-based methods to modern learning-based approaches. We discuss physics-based simulation methods, supervised regression methods, self-supervised energy minimization frameworks, and collision handling techniques.

2.2.1 Physics-Based Cloth Simulation

Physics-based cloth simulation can be conceptually decoupled into three complementary aspects: the mechanical model (spatial discretization), material modeling (constitutive laws defining elastic forces), and time integration (advancing the state over time). We categorize the existing works into three representative families: continuum-based methods, mass-spring systems, and constraint-based methods.

Continuum-Based Methods. Continuum-based methods model cloth as a continuous sheet, providing high physical accuracy. Early works Terzopoulos et al. [1987] modeled cloth dynamics using a continuum formulation, in which the potential energy functions were derived based on elasticity theory. The cloth was then discretized into a rectangular mesh and the resulting ordinary differential equation (ODE) of motion was solved by applying a semi-implicit integration method. Carignan et al. [1992] use the same formulation to simulate the cloth using explicit integration. However, realistically capturing the highly non-linear behavior of cloth is difficult: fabric hardly stretches under normal conditions, which gives rise to highly stiff equations of motion. Traditional numerical solvers struggle to handle this stiffness efficiently. To ensure stability, one must either significantly decrease the timestep size, thereby drastically increasing the computational time Breen et al. [1994], or heavily reduce the stiffness, which inevitably leads to unnaturally stretchy cloth. To alleviate this stability issue, Baraff and Witkin [1998] developed an implicit integration formulation on triangular cloth and handled contact and geometric constraints directly, allowing larger step-size control while keeping the simulation stable. To this day, this work is still the foundation of many current methodologies for high-fidelity cloth simulation including ARCSim Narain et al. [2012], which is widely used and has become a standard framework leveraging Finite Element Methods (FEM). Later, Kaldor et al. [2008] proposed modeling cloth at the yarn-level to achieve highly realistic macroscopic behavior. While accurate, these continuum and yarn-level methods suffer from high computational complexity which limits their real-time applications.

Mass-Spring Systems. Developed roughly in parallel with early continuum approaches, mass-spring systems offer a conceptually simpler and more computationally efficient alternative. By representing the garment as a network of point masses connected by elastic springs, some works propose simpler elastic formulations trading accuracy for speed Breen et al. [1992], Provot [1995]. However, they can suffer from the aforementioned over-stretching issues when the system is not perfectly tuned. To



maintain cloth inextensibility in such networks, Bridson et al. [2002] proposed an iterative refinement as post-processing to minimize changes in edge lengths while also taking into account collisions, friction and contacts. Later, Liu et al. [2013] developed an implicit Euler time integration scheme for classical Hookean mass–spring systems and solved the energy minimization problem using an efficient block coordinate descent algorithm.

Constraint-Based Methods. To tackle the strict real-time constraints of video games and interactive animation, constraint-based methods have seen widespread adoption. Instead of classical force-based simulation, Position-Based Dynamics (PBD) Müller et al. [2007] handles constraints formulated via constraint functions, directly working with positions while omitting the velocity layer. The original PBD suffers from the fact that the stiffness of the simulated material depends on the time step and the number of constraint projection iterations, a problem later addressed in Extended Position-Based Dynamics (XPBD) Macklin et al. [2016]. Owing to their robustness, numerical stability, and ease of implementation, PBD is extremely popular in interactive applications Nvcloth [2018]. In order to maintain cloth inextensibility in a constraint framework, Bender and Bayer [2008] proposed an impulse-based constraint to minimize changes in lengths. However, this constraint was posed everywhere on the cloth and did not yield a significant improvement. While PBD achieves faster simulation on low-resolution meshes than classical physics-based methods, this comes at the cost of limited accuracy and physical correctness. Moreover, performance degrades significantly at high mesh resolutions.

Unlike PBD, which enforces constraints through direct position projections, Projective Dynamics Bouaziz et al. [2023] formulates these constraints as quadratic energy terms. This formulation elegantly bridges position-based methods and classical FEM approaches, enabling efficient optimization with a constant Hessian while achieving improved physical accuracy.

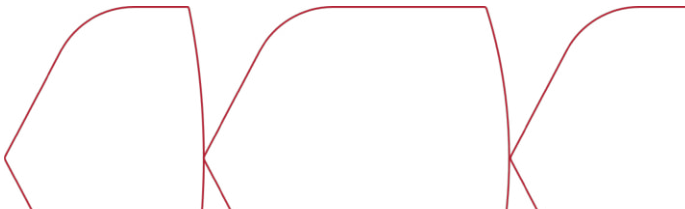
2.2.2 Supervised Cloth Simulation

In contrast to traditional physics-based simulation, which typically requires solving large systems of nonlinear equations at each time step, learning-based methods aim at regressing the desired output via a single function learned by neural networks. Various methods learn the static draped garment over the input body Patel et al. [2020], Gundogdu et al. [2020], Tiwari et al. [2020], Zakharkin et al. [2021], De Luigi

et al. [2023], often termed *neural garment draping*, while others take into account the dynamic motion Bertiche et al. [2020], Santesteban et al. [2019], Zhang et al. [2021b], Wang et al. [2019], Lähner et al. [2018], Guan et al. [2012], Tiwari and Bhowmick [2023], Li et al. [2024a], known as *neural garment simulation*. Supervised methods learn a parametric garment deformation using ground truth supervision generated by Physics-Based Simulation (PBS) methods Narain et al. [2012], Optitex [2018], NVIDIA Flex [2018], CLO Virtual Fashion [2018]. These methods learn a single function that directly predicts the garment positioning over the input body by incorporating both local and global characteristics of the body and garment geometry. Most methods predict vertex-wise additive displacements with respect to a template mesh, then pose the garment using skinning weights often transferred from the body mesh, with a few exceptions using 3D point clouds Zakharkin et al. [2021], Gundogdu et al. [2020], Ma et al. [2021b]. Santesteban et al. [2021] learn a generative deformation subspace in an unposed, deformed canonical space with a novel self-supervised collision term, though ground truth is still required. Lähner et al. [2018] employ a conditional generative adversarial network to synthesize high-frequency garment wrinkles as detailed normal maps. To resolve penetrations, some works apply a post-processing refinement step Santesteban et al. [2019], though at the cost of inference speed.

In such supervised deformation pipelines, the cloth behavior is largely dominated by the underlying body geometry and motion. To fully capture the physical interactions arising between cloth vertices themselves, Libao et al. [2023b] trains graph-based neural networks on trajectories generated by offline simulators. Their model builds on MeshGraphNets Pfaff et al. [2021], which has shown strong performance across various mesh-based simulation tasks.

Owing to the statistical nature of supervised regression, these methods tend to learn smooth, low-frequency approximations of the deformation, effectively averaging across the diverse ground-truth simulations. As a result, the predicted garments often exhibit over-smoothed, plastic-like behavior. In addition, generating large training datasets is costly, either through PBS simulation or multi-camera acquisition, and the learned models implicitly rely on a one-to-one mapping between body inputs and garment outputs. Beyond the data burden, the trained networks tend to overfit to specific mesh resolutions, boundary conditions and motion patterns seen during training, which limits their ability to generalize under distribution shifts such as



finer meshes or faster motions. Because explicit physical constraints are not enforced, their predictions may deviate from physically accurate behavior in unseen scenarios.

To facilitate these learning-based approaches, extensive 3D clothed human datasets have been constructed, as summarized in Tab. 2.1. We broadly categorize them into synthetic and real-world groups. Synthetic datasets are generated using physics-based simulation software such as ARCSim Narain et al. [2012] or simulators integrated into open-source and commercial tools (e.g., Blender, CLO3D, Marvelous Designer). These datasets typically provide dynamic sequences with large frame counts and controlled conditions. In contrast, real-world datasets are captured from multi-view scanners and generally contain fewer frames but exhibit authentic material behavior and occlusions. Most datasets provide SMPL body parameters and garment meshes or point clouds, though GarmentCodeData Korosteleva et al. [2024] instead includes rich anthropometric body measurements and sewing pattern annotations. While the scale of these datasets varies from hundreds of samples (Deep Fashion3D) to millions of frames (CLOTH3D, 3DPeople), they inevitably cover only a subset of possible garment dynamics. Consequently, supervised models trained on them often struggle to generalize to unseen motion or topologies outside the training distribution.

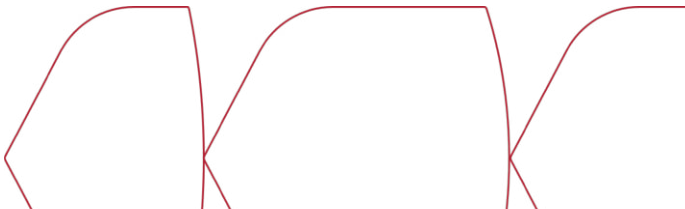
2.2.3 Self-Supervised Neural Simulation

Modern self-supervised methods train neural networks by directly minimizing the physics-based energy functions, obtaining differentiable neural simulators without the need for simulation data generated a priori. This relaxes the requirement on simulating large-scale datasets by physics-based simulators for training supervised models. Moreover, the per-time-step numerical integration methods used in physics-based simulators are shifted to a forward network pass. Such novel formulation is remarkable for cloth simulation, especially in the contemporary era of deep learning and large models.

While the pioneering work Bertiche et al. [2021] models static deformations using the potential energy of a mass-spring system, it specifically employs a cloth consistency loss that minimizes Laplacian differences and edge-length discrepancies to maintain surface smoothness. Later, Santesteban et al. [2022b], Bertiche et al. [2022] are able to model dynamic cloth deformations by introducing an inertia term and recurrent neural networks. However, the error margin is much higher in these methods and only material-based control over strain and bending forces produces

Dataset	Dynamic	Subjects	Outfits	Frames	Data Format	Software
CLOTH3D Bertiche et al. [2020]	✓	8.5k	7k	2.1M	SMPL + Garments	Blender
CLOTH4D Zou et al. [2023]	✓	1k	1k	100k	Mesh + Garments	CLO3D
VTO Santesteban et al. [2019]	✓	17	2	120k	SMPL + Garments	ARCSim
D-LAYERS Shao et al. [2023]	✓	1	4.9k	700k	SMPL + Garments	Blender
BEDLAM Black et al. [2023]	✓	271	111	380k	SMPL(-X) + Garments	CLO3D
ReSynth Ma et al. [2021b]	✓	3	24	30k	SMPL(-X) + Point Clouds	Deform Dynamics
TailorNet Patel et al. [2020]	×	9	9	55k	SMPL + Garments	Marvelous Designer
3DPeople Pumarola et al. [2019]	×	80	-	2.5M	Mesh + Garments	-
4D-DRESS Wang et al. [2024]	✓	32	64	78k	Scans + SMPL(-X) + Garments	-
4DHumanOutfit Armando et al. [2023]	✓	20	14	459k	Scans + SMPL + Garments	-
CAPE Ma et al. [2020]	✓	11	15	140k	SMPL+D	-
Multi-Garment Net Bhatnagar et al. [2019]	×	356	712	712	Scans + (SMPL+D) + Garments	-
Deep Fashion3D Heming et al. [2020]	×	-	563	2078	Scans + SMPL + Garments	-
BUFF Zhang et al. [2017]	✓	6	6	11k	Scans + SMPL	-

TABLE 2.1 : Overview of clothed human datasets. Synthetic datasets are highlighted in gray. *Dynamic*: whether temporal sequences are provided. *Data Format*: 3D representations of human bodies and garments. '–' indicates data not available or not applicable.



unrealistic stretches. Grigorev et al. [2023] uses graph neural networks to learn temporally coherent drapings of several garment meshes.

To better handle loose-fitting garments, Pan et al. [2022] propose introducing a set of virtual bones that parameterize garment deformation as a function of body pose, thereby extending skeletal control beyond the body surface. In a related direction, Santesteban et al. [2021] present a diffused body representation that generalizes blend shapes and skinning weights from the body mesh to the surrounding space, allowing garment vertices to be driven by a continuous body-dependent field. Bertiche et al. [2022] applies iterative Laplacian smoothing of the skinning weights of each garment vertex w.r.t. its neighbours.

In order to generalize to different garment topologies, several methods have demonstrated their applicability to clothes with arbitrary topology by encoding them as a point-cloud De Luigi et al. [2023], Gundogdu et al. [2020], graph Grigorev et al. [2023], Tiwari et al. [2023], or a UV map Zhang et al. [2021a].

Collision Handling. Collision handling is a critical component in cloth simulation. In learning-based methods, the classical detect-then-resolve pipeline is replaced by differentiable penalty terms integrated into the training loss. For cloth-body collision, most methods adopt a penetration-depth penalty Santesteban et al. [2022b], Bertiche et al. [2022], Grigorev et al. [2023], often combined with a post-processing correction step (Algorithm 1). For self-collision, the repulsive loss Lee et al. [2023] penalizes proximity between non-adjacent vertices. Beyond these standard formulations, several works have explored alternative self-collision strategies. Grigorev et al. [2024] uses the contour length of the self-intersection as an unsupervised loss. However, this length-based objective can sometimes resist resolution when the contour reduction direction opposes the true penetration-resolving direction. Liao et al. [2024] instead directly minimize the volume enclosed by self-intersecting cloth regions. This approach is more robust to complex initialization states, but requires that the garment mesh can be easily closed to form a watertight surface. In addition, it has higher computational cost due to remeshing and Global Intersection Analysis (GIA) operations.

Relations to our work. In this thesis, we build upon the self-supervised learning paradigm to overcome the computational bottleneck of classical physics-based simulation while avoiding the data dependency and generalization issues of supervised methods. While previous self-supervised methods Bertiche et al. [2021, 2022] enforce geometric constraints uniformly across the garment surface, this often leads to conflict

in regions where stretching is physically necessary to accommodate the body shape. In contrast, our approach (Section 4) introduces an *adaptive inextensibility loss* that selectively relaxes strain constraints based on local geometry. By formulating this through local covariance matrices Pauly et al. [2002], we allow the fabric to stretch only where geometrically required while preserving realistic stiffness elsewhere, thereby eliminating the need for explicit collision handling schemes.

2.3 Surface Representations for Deformable Geometry

Surface representations fundamentally determine the computational efficiency, differentiability, and generalization capability of deformation modeling systems. This section reviews existing representations and their suitability for modeling non-rigid deformations, and their application to garment geometry.

2.3.1 Explicit Surface Representations

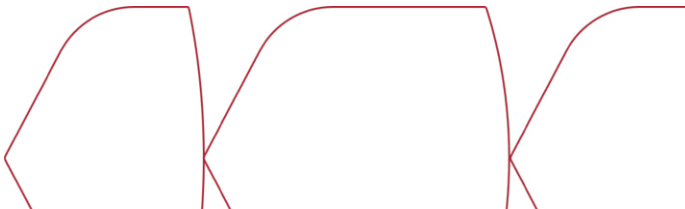
Explicit representations define geometry through direct coordinates or topological connectivity.

Triangle meshes remain the most ubiquitous representation in computer graphics and vision due to their maturity and efficient rendering support Botsch et al. [2010]. Their piecewise-linear nature, however, makes the computation of higher-order differential properties sensitive to noise, mesh irregularity, and sampling resolution.

Point-based representations represent surfaces as a collection of points or particles Szeliski and Tonnesen [1992]. By dispensing with fixed topology, they naturally handle large deformations and topological changes. However, maintaining surface smoothness requires complex interaction potentials, which can be computationally expensive.

Voxel grids Choy et al. [2016] generalize 2D pixels to the 3D domain, offering a regular structure that is naturally compatible with 3D convolutional architectures. While well-suited for volumetric data processing, they are fundamentally limited by their cubic memory complexity, which constrains the achievable resolution. As a result, voxel grids often struggle to preserve high-frequency surface details and can introduce discretization artifacts when representing smooth geometric shapes.

Parametric surfaces such as B-splines or NURBS provide smoother approximations with fewer parameters than dense meshes Farin [2002]. Subdivision surfaces offer a



flexible alternative by recursively refining a coarse polygonal mesh into a smoother, more detailed surface Warren and Weimer [2001]. While efficient for smooth shapes, they struggle to compactly represent intricate, high-frequency details without a significant increase in control points.

State-of-the-art explicit neural surface representations Yang et al. [2018], Groueix et al. [2018], Bednarik et al. [2020] typically rely on a neural atlas formulation. In these approaches, each local chart is defined by a neural network-based parameterization over a fixed open domain, such as a 2D square. The surface patches obtained by applying the parameterizations to the parametric domain are then unioned to form the represented surface. The learned parametric mappings enable accurate shape reconstruction and support arbitrary resolutions with minimal memory overhead. Following Groueix et al. [2018], Bednarik et al. [2020] proposed new losses based on higher-order analytical surface properties to reduce patch collapse and overlap. However, stitching these patches into a coherent, artifact-free surface remains a significant challenge Deng et al. [2020]. Low and Lee [2022] propose an atlas-based explicit neural surface representation that learns flexible chart domains via an implicit occupancy field, enabling low-distortion parameterizations with as few as three charts.

Truncated Taylor expansions, or n-jets, provide an analytical local fit for surfaces Cazals and Pouget [2005]. They have been widely used for estimating differential quantities (e.g., normals and curvatures) on unorganized point sets Ben-Shabat and Gould [2020]. Unlike numerical differentiation on discrete meshes, jets yield closed-form expressions for derivatives of arbitrary order, which facilitates gradient-based optimization on surface geometry.

2.3.2 Implicit Representations

Neural implicit representations define a surface as the zero level set of a scalar field $F(x, y, z) = 0$, where the field is parameterized by a multilayer perceptron (MLP) that maps spatial coordinates to implicit values. These methods can employ either signed distance fields (SDFs), which are restricted to closed surfaces, or unsigned distance fields (UDFs) Long et al. [2023], which can represent open surfaces but may produce double or crusted geometry due to ambiguity near zero-level sets. The continuous parameterization offers memory efficiency and resolution-independent representations, enabling straightforward isosurface extraction via marching cubes Lewiner et al. [2003], Guillard et al. [2022]. Patch-based implicit representations have also been

used to model generic 3D shapes Tretschk et al. [2020], Deprelle et al. [2019]. Despite their expressive power, neural implicit formulations incur high computational costs during dense field queries, and their lack of an explicit, compact state makes them difficult to integrate directly into real-time physics-based simulation loops.

2.3.3 Radiance Fields and Volumetric Splatting

In recent years, novel view synthesis and 3D scene reconstruction have been revolutionized by radiance fields and volumetric representations, most notably Neural Radiance Fields (NeRF) Mildenhall et al. [2020] and 3D Gaussian Splatting (3DGS) Kerbl et al. [2023]. These data-driven methods excel at capturing complex view-dependent appearances and fine geometric structures without relying on explicit topological priors. Extensions to dynamic scenes have shown promising results in modeling non-rigid deformations from monocular or multi-view video.

However, despite their immense success in achieving high visual fidelity, these representations are not inherently suitable for physics-based simulation or tasks requiring strict geometric constraints. While NeRF represents geometry implicitly as a continuous density field, and 3DGS relies on an explicit but disconnected collection of volumetric splats, both lack a defined, continuous surface topology. Consequently, extracting a clean, manifold surface mesh with consistent connectivity from them remains a non-trivial challenge, often requiring hybrid approaches that tether these volumetric splats back to explicit mesh proxies to enable structural control Waczyńska et al. [2024], Gao et al. [2024]. In the context of garment modeling and cloth simulation, where enforcing physical laws (such as inextensibility and collision avoidance) and computing differential quantities (like bending and strain) rely on explicit surface topology, template-based and continuous patch-based representations offer distinct advantages.

2.3.4 Application to Garment Modeling

Diverse geometric representations have been explored for modeling garments. One common strategy extends parametric body models to represent clothing as vertex-wise offsets, often referred to as SMPL+D Pons-Moll et al. [2017], Ma et al. [2020], Alldieck et al. [2018, 2019b,a], Bhatnagar et al. [2019]. This approach models garments as offsets relative to the underlying body template (e.g., SMPL Loper et al. [2015]) and leverages linear blend skinning to articulate the clothed geometry.

However, since the body and cloth are modeled in a unified manner, this approach is limited to representing geometry that remains close to the body surface.

Another common strategy in learning-based simulation is to predict per-vertex displacement from the input garment mesh template Santesteban et al. [2019, 2022b], Bertiche et al. [2022]. Such approaches leverage the mesh template's explicit connectivity, enabling tight integration with downstream physics-based simulation and real-time inference.

Neural implicit representations offer an alternative modeling paradigm for garments. Some methods Saito et al. [2021a], Chen et al. [2021], Tiwari et al. [2021], Palafox et al. [2021], Chen et al. [2022] represent the clothed human as a unified implicit surface conditioned on pose and shape, while others Corona et al. [2021], De Luigi et al. [2023] model the clothing and underlying body separately. Because standard signed distance functions (SDFs) assume closed, watertight geometry, the latter approaches typically adopt unsigned distance fields (UDFs), which can represent open garment surfaces more accurately. Despite their expressive power, implicit methods require dense 3D queries during inference and are difficult to integrate into physics-based simulation loops that rely on per-vertex computations.

Point-based representations have also been explored for garment geometry. Zakharkin et al. [2021] trains a draping function that maps the outfit latent code and the SMPL parameters to a point cloud, and uses neural point-based graphics to render appearance, offering flexibility for handling topology changes. Parallel to this, Ma et al. [2021b] record the body surface points on a UV positional map, and decodes a per-point displacement based on the points' local body pose and garment geometry features. However, point clouds alone struggle with fine surface details. Moreover, the lack of explicit surface connectivity makes it difficult to define meaningful local geometric priors or constraints, and the need for dense point cloud reconstruction during training scales poorly.

Finally, the idea of modeling clothing surfaces as collections of local patches has emerged as an expressive mid-level abstraction. In the implicit domain, PatchNets Tretschk et al. [2020] represents local patches as continuous signed distance functions in a canonical space to achieve shape-agnostic generalizability. Similarly for garments, Li et al. [2023a] decomposes clothing into 2D panels represented by implicit signed distance functions in UV space, lifted to 3D through learned mappings.

On the explicit side, methods like Bednarik et al. [2020] learn continuous map-

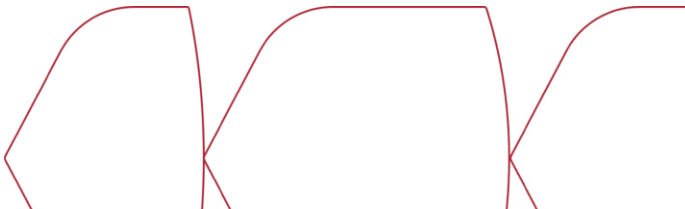
pings from a 2D parameter space to 3D point clouds, offering the unique advantage of analytical access to exact differential surface properties without requiring triangulation. Parallel to this, Ma et al. [2021a] adopts a large set of articulated local parametric elements attached to a minimally clothed body to capture topologically varying clothing with fast inference.

Relations to our work. Our approach adopts a local, analytical surface representation based on patch-wise n-jet polynomial fitting. Unlike Ben-Shabat and Gould [2020], which uses a PointNet Qi et al. [2017a]-based module to infer point-wise differential quantities, our method performs explicit patch-wise surface fitting aimed at efficient deformation modeling. Unlike atlas-based methods, our representation is trained on lightweight synthetic analytic patches, is local by construction, and generalizes to wide range of 3D surface deformations. By employing patch-wise n-jet coefficients as the state variables of deformation, we enable closed-form derivatives that drive both test-time 3D reconstruction via inverse optimization and self-supervised draping tasks.

2.4 Neural Operators and Resolution-Agnostic Learning

The methods discussed thus far, encompassing both physics-based and learning-based approaches, typically operate on fixed-resolution discretizations. Standard neural network architectures (e.g., CNNs on regular grids, GNNs on graphs) are designed to operate on finite-dimensional vectors and are inherently tied to the discretization used during training. This resolution dependence poses a fundamental barrier to scalable simulation: high-resolution training data is expensive to generate, yet coarse-resolution models cannot capture fine-scale dynamics such as wrinkles and folds.

Many phenomena and dynamical systems of interest in science and engineering can be modeled as nonlinear partial differential equations (PDEs). To efficiently solve PDEs whose closed-form solutions are typically unknown, neural networks have been widely adopted. However, their reliance on specific discretizations limits their ability to generalize across resolutions. To overcome this, *neural operators* have been developed to learn mappings between infinite-dimensional function spaces Li et al. [2020, 2021], Kovachki et al. [2023]. By design, they are discretization-invariant, enabling models trained on coarse meshes to evaluate on continuous domains and



generalize to arbitrarily fine resolutions at test time.

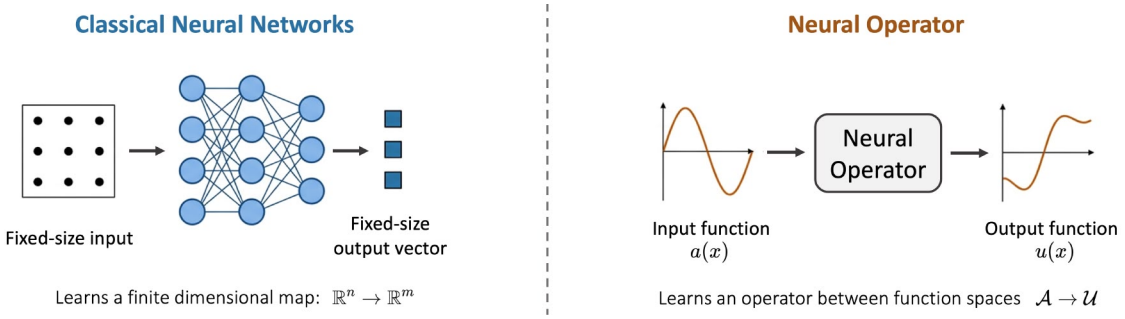


FIGURE 2.1 : Comparison of standard neural networks and neural operators.

Various architectures of neural operators have been studied. Graph neural operators (GNO) Li et al. [2020] perform kernel integration on graph structures and can handle irregular geometries. However, like other graph-based methods, GNO is limited by computational complexity with long-range global interactions on the graph. To overcome this limitation, the Fourier Neural Operator (FNO) was introduced to represent the kernel integration in the spectral domain by leveraging the Fourier transform, enabling the use of the discrete fast Fourier transform (FFT) to improve efficiency. The mathematical formulation of Neural Operators, specifically the Fourier Neural Operator (FNO), is detailed in Section 3.5.

From these outstanding properties, FNO has become a standard in scientific computing and has been applied to various domains including fluid and solid mechanics Li et al. [2022], Choubineh et al. [2023], Khorrami et al. [2025], geoscience Yang et al. [2021], Wen et al. [2023], weather forecasting Pathak et al. [2022], Kurth et al. [2023], Bonev et al. [2023] and inverse-design problems Zhou et al. [2024], as shown in Figure 2.2. Notably, Pathak et al. [2022], Kurth et al. [2023] uses an adaptive Fourier Neural Operator (AFNO) to produce high-resolution weather forecasts capable of generating week-long predictions in less than 2 seconds, demonstrating the significant potential of FNO for industrial downstream applications.

While purely data-driven Fourier Neural Operators have achieved remarkable success across diverse scientific domains, their performance heavily relies on the availability of large-scale, high-fidelity simulation data. Generating such extensive curated datasets is often computationally prohibitive. To alleviate this strict data dependency, recent advancements have introduced the Physics-Informed Neural Operator (PINO) Li et al. [2024b]. By incorporating the governing partial differential

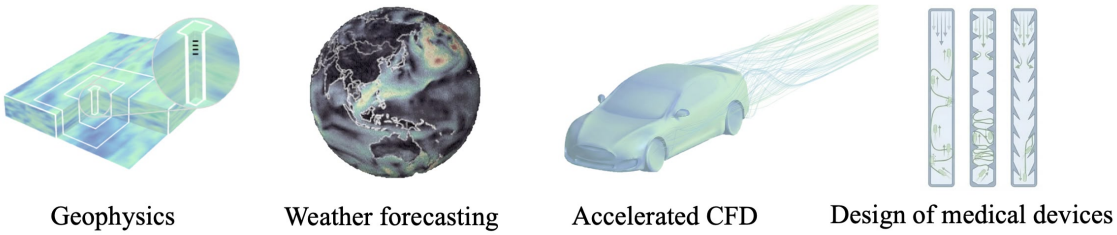


FIGURE 2.2 : Notable applications of the FNO and its variants.

equations (PDEs) as soft constraints into the training loss, PINO enables the model to learn physically consistent operator mappings with limited or even no paired ground-truth data, effectively combining the expressiveness of neural operators with the generalizability of physics-informed learning.

Relations to our work. The property of discretization invariance is highly desirable for cloth simulation, as it allows the model to consistently capture fine-level details, such as folds and wrinkles, across varying mesh resolutions. While the advanced computational efficiency and global receptive field of the Fourier Neural Operator (FNO) make it an ideal backbone for this task, purely data-driven FNOs remain strictly dependent on large-scale datasets. Inspired by the physics-informed paradigm that mitigates this data dependency, our proposed FNOPT framework (detailed in Section 6) takes a step further to address the complex dynamics of soft bodies.

Instead of training the neural operator to directly map initial conditions to future states like a standard PDE solver, FNOPT recasts the implicit time integration of cloth dynamics as an energy minimization problem. Specifically, it meta-learns a neural optimizer parameterized by an FNO. By evaluating the gradients of discrete physics-based energy functions (e.g., stretching, bending, and collision), FNOPT utilizes the FNO backbone to iteratively predict optimal state updates in a completely self-supervised manner. To the best of our knowledge, this is the first work to apply FNO to cloth simulation. By bridging the gap between physics-informed operator learning and iterative numerical optimization, FNOPT opens up new possibilities for robust resolution-agnostic cloth simulation.

2.5 Template-Based 3D Reconstruction

Shape-from-Template (SfT), or template-based reconstruction, aims to recover the shape of a deformable object from monocular RGB imagery, assuming a reference

3D template is available in a rest configuration. The foundational mathematical formulation was established by Bartoli et al. [2015], who posed the isometric SfT problem rigorously, proved its well-posedness, and derived the first algebraic closed-form solution. This seminal work laid the theoretical groundwork for subsequent developments in the field.

2.5.1 Deformation priors

Since monocular shape estimation is an ill-posed inverse problem, resolving depth ambiguities requires strong regularization. In this section, we discuss the various deformation priors proposed in the literature.

Deformation Priors. Deformation priors explicitly model the mathematical properties of the mapping between the template and the deformed observation. Among metric-based priors, the strongest prior is *isometry*, which assumes the object deforms without stretching or shrinking (e.g., paper). Intuitively, this means that geodesic distances between any two points on the surface remain unchanged during deformation. Mathematically, this enforces the preservation of the Riemannian metric tensor. If $\mathbf{J}_\Delta$ and $\mathbf{J}_\phi$ represent the Jacobians of the template embedding and the deformed surface respectively, the isometric constraint requires:

$$\mathbf{J}_\phi^\top \mathbf{J}_\phi = \mathbf{J}_\Delta^\top \mathbf{J}_\Delta. \tag{1}$$

An alternative to isometry is *inextensibility*, which enforces preservation of Euclidean distance rather than geodesic distances on the surface. For real-world deformable materials that exhibit some degree of elasticity, *quasi-isometric* constraints can be employed to limit excessive deformation while allowing controlled stretching, and can be implemented as As-Rigid-As-Possible (ARAP) Sorkine and Alexa [2007], Parashar et al. [2015].

Other deformation priors relax this assumption: *conformal* maps Bartoli et al. [2015], Parashar et al. [2020a] preserve local angles, allowing for stretching. *Equiareal* prior preserve area on the surface. Different from these geometric priors, *Mechanical* priors Malti et al. [2013, 2015], Malti and Herzet [2017], Haouchine and Cotin [2017] leverage thin-shell theory to model material stress and strain using continuum mechanics, enabling the reconstruction of moderately extensible materials like rubber or skin.

Temporal Priors. Temporal priors leverage the correlation between consecutive video frames to resolve ambiguities that are unsolvable in static images. These constraints typically enforce temporal smoothness by minimizing the change in velocity or acceleration of surface points over time Yu et al. [2015], Tran et al. [2025], which effectively acts as a low-pass filter on the deformation trajectory.

2.5.2 Solving Strategies

For the solving strategies, existing approaches can be broadly categorized into energy-based, analytical, and learning-based methods.

Analytical Methods. Analytical methods Chhatkuli et al. [2016], Casillas-Perez et al. [2019, 2021] reformulate the reprojection and deformation constraints as systems of Partial Differential Equations (PDEs) to derive closed-form solutions from a set of correspondences between the 3D template and the image. Building on the theoretical framework of Bartoli et al. [2015], Chhatkuli et al. [2016] further improved stability by integrating differential quantities over surface patches. Analytical approaches are fast and do not require initialization, but they rely on accurate feature correspondences, which are often noisy or missing in practice, and can be sensitive to derivative noise.

Learning-Based Methods. Learning-based methods employ neural networks to regress 3D geometry directly from input images, bypassing the need for explicit correspondences. Early supervised approaches like DeepSfT Fuentes-Jimenez et al. [2022] use Fully Convolutional Networks (FCNs) trained on synthetic data to predict shape. However, these methods are typically *template-specific*, meaning the geometric prior is baked into the network weights. Consequently, they often require retraining for new objects and are inherently restricted to scenarios similar to the training distribution, limiting generalization to unseen deformations. To reduce dependence on synthetic data and improve generalization, recent works employ weak supervision. For instance, WS-DeepSfT Luengo-Sanchez et al. [2025] trains a registration DNN on real RGB video sequences in a weakly supervised manner using optical-flow-based image-space matching and regularization losses, and does not require dense 3D ground truth or depth sensors, while achieving improved robustness and accuracy over prior optimization and learning-based methods.

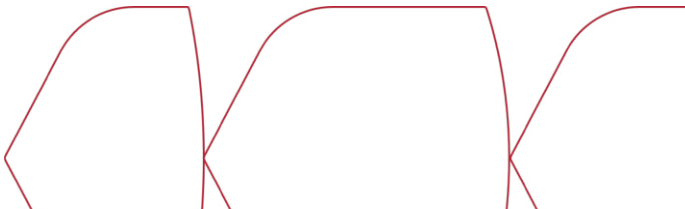
Energy-Based Methods. Energy-based methods formulate reconstruction as an optimization problem minimizing a cost function $\mathcal{L}(\mathbf{x}) = \mathcal{L}_{\text{data}} + \lambda\mathcal{L}_{\text{prior}}$, where the data term measures reprojection error and the prior term penalizes deviations from

the deformation models described above. Early works relaxed the non-convex isometry constraint to convex inextensibility bounds to ensure global convergence Salzmann et al. [2008]. Later approaches tackled the non-convex energy directly using iterative solvers to handle elastic Malti et al. [2013, 2015] or volumetric Parashar et al. [2015] deformations.

While the methods discussed above rely on explicit correspondences, a recent trend within energy-based approaches integrates *physics simulation* as a regularization mechanism, enabling correspondence-free reconstruction. Methods like Φ-SfT Kairanda et al. [2022] employ differentiable physics engines as forward models, optimizing physical parameters (stiffness, mass, external forces) by backpropagating photometric errors through the simulator. This constrains the solution space to physically plausible deformation trajectories. To mitigate the computational cost of numerical simulation (FEM), hybrid approaches such as Physics-Guided SfT Stotko et al. [2024] replace the physics solver with learned neural surrogates, achieving orders-of-magnitude speedup while preserving physical validity. Stotko and Klein [2025] extends this paradigm to jointly recover geometry and spatially-varying appearance, demonstrating that physics-informed regularization is crucial for disentangling shading from shape.

Distinct from approaches relying on forward simulation, Kairanda et al. [2025] performs analysis-by-synthesis optimization of a continuous neural deformation field under a Kirchhoff–Love thin-shell prior, balancing photometric consistency with physically plausible surface dynamics. Tran et al. [2025] propose an unsupervised, correspondence-free SfT framework that optimizes an image-driven energy using a time-conditioned deformation network predicting mesh vertex offsets under inextensibility regularization, achieving high-fidelity reconstructions from only image observations.

Relations to our work. In this thesis, we formulate the Shape-from-Template problem as a test-time inverse optimization task (Section 5). Unlike previous methods that depend on dense correspondences or purely data-driven priors, we leverage our proposed continuous surface representation to model the template deformation. By employing polynomial descriptions with closed-form derivatives, our PolyFit-based framework enables robust gradient-based optimization directly on the photometric and geometric losses, effectively recovering accurate 3D shapes for synthetic and real datasets.



CENTRALE LYON

LIRIS

CHAPTER 3

Technical Background

3.1 Introduction

This chapter establishes the mathematical and computational foundations that underpin the methods developed in this thesis. We present core theoretical tools, including continuum surface mechanics, discretization strategies, time integration, optimization-based simulation formulations, and parametric body modeling, that are shared across the subsequent chapters.

Section 3.2 introduces the continuous formulation of deformable surface mechanics, including metric tensors, deformation gradients, curvature, thin shell elasticity, and the governing partial differential equation of cloth dynamics. It then discusses how these continuous laws are discretized onto triangle meshes, and reviews constitutive material models governing elastic behavior.

Section 3.3 introduces parametric human body models, particularly SMPL and linear blend skinning, which provide the body geometry and kinematic structure used across all garment simulation tasks.

Section 3.4 derives the optimization-based formulation of implicit time integration, which recasts Newton's equations of motion as an energy minimization problem and forms the theoretical basis for self-supervised neural simulation.

Section 3.5 presents the framework of neural operators for solving PDEs, detailing the Fourier Neural Operator (FNO) architecture which enables resolution-agnostic learning.

3.2 Surface Elasticity and Deformation Theory

This section establishes the mathematical foundations for modeling deformable surfaces, particularly cloth and garment materials. We begin with the continuous formulation rooted in differential geometry and thin shell elasticity, discuss discretization strategies onto triangle meshes, and finally review the constitutive material models governing elastic behavior.

3.2.1 Continuous Formulation

The dynamics of cloth and other thin elastic materials can be described by a partial differential equation governing the spatiotemporal evolution of the surface. The seminal work of Terzopoulos et al. [1987] introduced such elastically deformable models to computer graphics, formulating the governing equation as:

$$\frac{\partial}{\partial t}\left(\mu\frac{\partial x}{\partial t}\right) + \gamma\frac{\partial x}{\partial t} + \frac{\delta\mathcal{E}}{\delta x} = f(x,t), \tag{2}$$

where $x(p,t)$ is the position of a material point $p \in \Omega$ at time t, $\mu(p)$ is the mass density, $\gamma(p)$ is the damping density, $\mathcal{E}(x)$ is the elastic energy of deformation, and $f(x,t)$ represents external forces. In Terzopoulos et al. [1987], the elastic energy is constructed directly from the first and second fundamental forms of the surface, penalizing deviations of the current metric and curvature from their rest-state values. Modern formulations refine this approach using deformation gradients and constitutive models from continuum mechanics, but the underlying structure remains the same: the energy $\mathcal{E}$ decomposes into membrane (in-plane) and bending (out-of-plane) contributions, whose variational derivative $\delta\mathcal{E}/\delta x$ yields the internal elastic forces.

In the following, we present the mathematical quantities that define this elastic energy: the metric tensor and deformation gradient for in-plane behavior, and the curvature tensor for bending.

Metric tensor and surface inextensibility. Consider a surface patch parameterized by coordinates $(u,v) \in \Omega \subset \mathbb{R}^2$, embedded in $\mathbb{R}^3$ as $\mathbf{x}(u,v,t)$. The intrinsic geometry of the surface is captured by the *first fundamental form*, or *metric tensor*:

$$g_{ij} = \frac{\partial \mathbf{x}}{\partial u^i} \cdot \frac{\partial \mathbf{x}}{\partial u^j}, \quad i,j \in 1,2, \tag{3}$$

where $u^1 = u$ and $u^2 = v$. The metric tensor $\mathbf{G} = [g_{ij}]$ encodes geodesic distances and local area measurements.

A deformation is *inextensible* (or *isometric*) if it preserves the metric tensor everywhere:

$$g_{ij}(u, v, t) = \bar{g}_{ij}(u, v), \quad \forall (u, v) \in \Omega, \tag{4}$$

where $\bar{g}_{ij}$ denotes the rest-state metric. This constraint characterizes materials that can bend freely but resist stretching and compression, effectively approximating woven textiles.

Deformation Gradient and Strain. The local deformation is characterized by the deformation gradient $\mathbf{F} \in \mathbb{R}^{3\times 2}$, which maps infinitesimal vectors from the reference domain to the deformed surface:

$$\mathbf{F} = \begin{bmatrix} \frac{\partial \mathbf{x}}{\partial u} & \frac{\partial \mathbf{x}}{\partial v} \end{bmatrix}. \tag{5}$$

The metric tensor relates to $\mathbf{F}$ via $\mathbf{G} = \mathbf{F}^T\mathbf{F}$, making the connection between the first fundamental form and the deformation gradient explicit.

Deviations from the rest state are quantified by the *Green-Lagrange strain tensor* $\mathbf{E}$:

$$\mathbf{E} = \frac{1}{2}(\mathbf{G} - \bar{\mathbf{G}}). \tag{6}$$

For cloth materials with planar rest states, $\bar{\mathbf{G}} = \mathbf{I}$, which simplifies the strain to

$$\mathbf{E} = \frac{1}{2}(\mathbf{F}^T\mathbf{F} - \mathbf{I}). \tag{7}$$

Curvature and the second fundamental form. While the first fundamental form captures the intrinsic (in-plane) geometry, the *second fundamental form* characterizes the extrinsic (out-of-plane) bending of the surface. Let $\mathbf{n}$ denote the unit surface normal. The second fundamental form is defined as:

$$b_{ij} = -\frac{\partial \mathbf{n}}{\partial u^i} \cdot \frac{\partial \mathbf{x}}{\partial u^j} = \mathbf{n} \cdot \frac{\partial^2 \mathbf{x}}{\partial u^i \partial u^j}, \quad i, j \in \{1, 2\}. \tag{8}$$

From the matrix $\mathbf{B} = [b_{ij}]$ and the metric tensor $\mathbf{G}$, the principal curvatures κ_1, κ_2 are obtained as the eigenvalues of $\mathbf{G}^{-1}\mathbf{B}$. Two scalar curvature invariants are of particular importance:

- The *mean curvature* $H = \frac{1}{2}(\kappa_1 + \kappa_2) = \frac{1}{2}\operatorname{tr}(\mathbf{G}^{-1}\mathbf{B})$, which measures the average

bending of the surface.

- The *Gaussian curvature* $K = \kappa_1\kappa_2 = \det(\mathbf{G}^{-1}\mathbf{B})$, which is an intrinsic quantity by Gauss's Theorema Egregium and remains invariant under isometric deformations.

For cloth, the mean curvature governs the bending energy: a flat sheet has $H = 0$, and deviations from zero indicate the presence of folds or wrinkles.

Thin shell elasticity. Cloth and garment materials have a thickness h that is negligible compared to their lateral dimensions. Such bodies are classified as *thin shells*: three-dimensional elastic solids that can be faithfully represented by their two-dimensional mid-surface. Under the Kirchhoff-Love hypothesis Wempner et al. [2003], material fibers initially normal to the mid-surface are assumed to remain straight and normal after deformation, and the transverse normal stress is neglected. This assumption implies that the deformation state of the shell is entirely determined by the mid-surface geometry, i.e., by the first fundamental form $\mathbf{G}$ (which captures stretching and shearing) and the second fundamental form $\mathbf{B}$ (which captures bending). As a consequence, the total elastic energy decomposes into membrane and bending components Terzopoulos et al. [1987], Grinspun et al. [2003]:

$$\mathcal{E}_{\text{total}} = \mathcal{E}_{\text{membrane}} + \mathcal{E}_{\text{bending}}. \tag{9}$$

The *membrane energy* penalizes in-plane deformations (stretching and shearing), and is defined as an integral of the strain energy density $\psi(\mathbf{E})$ over the mid-surface:

$$\mathcal{E}_{\text{membrane}} = \int_{\Omega} \psi(\mathbf{E})\, dA, \tag{10}$$

where the specific form of ψ depends on the constitutive model (see Section 3.2.3).

The *bending energy* penalizes out-of-plane deformations, i.e., changes in curvature relative to the rest state. A standard formulation, derived from the Kirchhoff-Love theory, models the bending energy as:

$$\mathcal{E}_{\text{bending}} = \int_{\Omega} \kappa_b (H - \bar{H})^2\, dA, \tag{11}$$

where κ_b is the flexural stiffness and $\bar{H}$ is the rest-state mean curvature.

3.2.2 Discretization Strategies

Practical simulation requires discretizing the continuous governing equations onto a finite computational domain. The standard discrete substrate for cloth is a triangle mesh, on which the surface is approximated as a collection of planar triangles. On such a piecewise linear surface, the metric tensor is constant per triangle and is determined entirely by the edge lengths. Inextensibility in the discrete setting then amounts to preserving the length of every mesh edge: $\|\mathbf{x}_i - \mathbf{x}_j\| = \bar{l}_{ij}$.

Given this triangulated domain, mechanical forces must be defined to simulate elastic behavior. We present two approaches in order of historical development: mass-spring systems, which offer computational simplicity, and the finite element method, which provides continuum-consistent accuracy.

Mass-spring systems. Early cloth simulators relied on mass-spring systems Provot [1995], which model the surface as a network of point masses connected by massless springs along the mesh edges. The membrane energy is approximated by summing the potential energy of individual springs:

$$\mathcal{E}_{\text{spring}} = \sum_{(i,j)\in\mathcal{E}} \frac{k_s}{2} \left(\|\mathbf{x}_i - \mathbf{x}_j\| - \bar{l}_{ij}\right)^2. \tag{12}$$

While computationally efficient, the spring constant k_s is a phenomenological parameter whose effective macroscopic behavior depends on the mesh topology and connectivity. This makes it difficult to simulate materials with well-defined, mesh-independent physical parameters. Nevertheless, mass-spring models remain widely used in learning-based simulation due to their simplicity and differentiability.

While mass-spring systems offer practical efficiency, they lack a direct connection to continuum mechanics. The finite element method addresses this limitation.

Finite Element Method (FEM). Although FEM has been a cornerstone of structural engineering since the 1950s, its application to cloth simulation in computer graphics emerged only in the late 1990s Gan et al. [1995], Eischen et al. [1996], Etzmuß et al. [2003]. It provides a continuum-consistent spatial discretization that preserves the physically measurable material parameters (Young's modulus, Poisson's ratio) introduced in the continuous setting Narain et al. [2012]. For a linear triangle element, the deformation gradient $\mathbf{F}$ is assumed constant over the element and can

be computed as

$$\mathbf{F} = \mathbf{D}_s \mathbf{D}_m^{-1}, \tag{13}$$

where $\mathbf{D}_s = [\mathbf{x}_1 - \mathbf{x}_0, \mathbf{x}_2 - \mathbf{x}_0]$ stacks the deformed edge vectors and $\mathbf{D}_m$ stacks the corresponding reference-edge vectors. The Green–Lagrange strain tensor $\mathbf{E}$ is then computed from $\mathbf{F}$ as defined in Eq. (6), and the elastic energy of the element is obtained by integrating the strain-energy density $\psi(\mathbf{E})$ over the triangle area A:

$$\mathcal{E}_{\text{element}} = A \cdot \psi(\mathbf{E}(\mathbf{F})). \tag{14}$$

These element-level energies are assembled over the mesh, and nodal forces are obtained as $\mathbf{f} = -\nabla_{\mathbf{x}} \mathcal{E}_{\text{total}}$, yielding the global system used for time integration.

3.2.3 Material Models and Constitutive Laws

Constitutive laws define the relationship between deformation (strain) and restoring forces (stress).

St. Venant-Kirchhoff (StVK) model. The StVK model is a geometrically nonlinear extension of linear elasticity, capable of handling arbitrarily large rotations. Its energy density is quadratic in the Green-Lagrange strain $\mathbf{E}$:

$$\psi_{\text{StVK}}(\mathbf{E}) = \mu |\mathbf{E}|_F^2 + \frac{\lambda}{2} \text{tr}(\mathbf{E})^2, \tag{15}$$

where μ, λ are the Lamé coefficients, related to Young's modulus E_Y and Poisson's ratio ν by:

$$\mu = \frac{E_Y}{2(1+\nu)}, \quad \lambda = \frac{E_Y \nu}{(1+\nu)(1-2\nu)}. \tag{16}$$

Discrete Bending Models. The bending model is essential for obtaining realistic visual details of cloth, i.e. dynamic wrinkles and folds. Since piecewise linear meshes have zero curvature inside triangles and infinite curvature at edges, bending energy is defined via the *dihedral angle* θ_e between adjacent faces. A widely adopted discrete bending model was first proposed in Bridson et al. [2005] and has been extended in subsequent works Wang et al. [2011], Narain et al. [2012]. The energy is formulated as:

$$\mathcal{E}_{\text{bend}} = \sum_{edges} \frac{k_b l_e^2}{8(A_0 + A_1)} \theta_e^2 \tag{17}$$

where k_b is the material bending stiffness coefficient, l_e is the length of the edge, A_0 and A_1 are the areas of the two triangles sharing the edge, and θ_e is the dihedral angle between these triangles. This formulation is widely adopted in recent learning-based cloth simulation methods Santesteban et al. [2022a], Bertiche et al. [2022].

3.3 Parametric Human Body Modeling

Accurate cloth simulation requires a reliable representation of the human body that provides both geometric and kinematic information. Parametric body models Anguelov et al. [2005], Brégier et al. [2025], Loper et al. [2015] offer a compact and structured way to describe human shape and pose, and have therefore become a standard component in garment simulation and virtual try-on pipelines. Early parametric models such as SCAPE Anguelov et al. [2005] employed PCA-based shape spaces derived from 3D body scans. Recently, unified models like Anny Brégier et al. [2025] extends parametric representation to span the entire human lifespan, addressing demographic gaps in earlier models.

Among these models, SMPL Loper et al. [2015] is the most widely adopted statistical representation of the human body. SMPL models the body surface as a triangulated mesh driven by a low-dimensional set of shape parameters β and pose parameters θ. Starting from a template mesh in a rest pose, body shape variations are captured by linear blend shapes, while pose-dependent deformations account for non-rigid effects induced by articulation. Formally, the unposed body surface is expressed as:

$$T(\beta, \theta) = \mathcal{T} + B_S(\beta) + B_P(\theta), \tag{18}$$

where $\mathcal{T}$ denotes the mean template mesh, and B_S and B_P represent shape- and pose-dependent displacements (commonly referred to as shape and pose blendshapes), respectively.

Finally, the articulated mesh is obtained using the Linear Blend Skinning (LBS) function $W(\cdot)$. The complete SMPL model $M(\beta, \theta)$ maps shape and pose parameters to vertices:

$$M(\beta, \theta) = W(T(\beta, \theta), J(\beta), \theta, \mathcal{W}), \tag{19}$$

where $J(\beta)$ are the joint locations regressed from the body shape, and $\mathcal{W}$ are the skinning weights. Specifically, for each vertex $\boldsymbol{x}_k$ of the deformed template $T(\beta, \theta)$,

the skinning transformation is defined as:

$$\hat{\boldsymbol{x}}_k = \sum_{j=1}^{J} w_{k,j}\, \mathbf{G}_j(\theta, J(\beta))\, \boldsymbol{x}_k, \tag{20}$$

where $w_{k,j} \in \mathcal{W}$ satisfying $\sum_j w_{k,j} = 1$, and $\mathbf{G}_j(\theta, J(\beta))$ represents the rigid transformation associated with joint j.

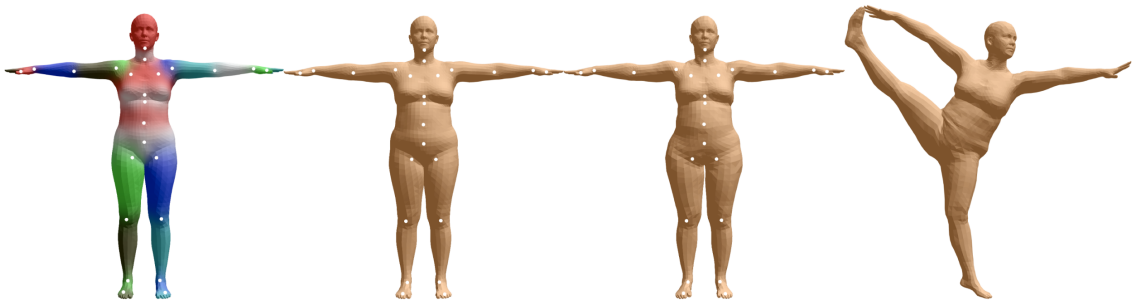


FIGURE 3.1 : SMPL parametric human model (extracted from Loper et al. [2015]). (a) The template mesh visualized with color-coded skinning weights and joint locations shown as white markers. (b) Applying the shape-dependent blendshape produces a body-specific deformation and the corresponding joints. (c) Adding the pose-dependent blendshape. (d) The final articulated surface is obtained by linear blend skinning.

The skinning weights $\mathcal{W} = \{w_{k,j}\}$ play a crucial role in defining how the surface deforms with the skeletal motion. Intuitively, a weight $w_{k,j}$ represents the influence of the j-th joint on the k-th vertex. Vertices near the center of a bone typically have a weight close to 1 for that bone, while vertices near joints (e.g., elbows and knees) blend influences from adjacent bones to produce smooth skin deformations. In the context of SMPL, these weights are not manually defined but are learned from 3D scan data to minimize reconstruction error across a large dataset of poses.

The original SMPL framework has spawned several extensions: SMPL-X Pavlakos et al. [2019] augments the model with articulated hands and facial expressions, SMPL+H Romero et al. [2022] provides detailed hand geometry, STAR Osman et al. [2020] incorporates sparse pose-corrective deformations, and SUPR Osman et al. [2022] offers a unified representation trained on large-scale 3D scan data. The concise parameterization of SMPL enables efficient control of body motion and shape while maintaining geometric consistency. This property has facilitated its widespread adoption across diverse computer vision applications, including human–scene interac-

tion Hassan et al. [2019], Xu et al. [2023], monocular avatar reconstruction Xiu et al. [2022], Saito et al. [2021b] and Human Mesh Recovery Zhang et al. [2023].

For cloth simulation applications, SMPL provides both a stable skeletal structure and a continuous body surface that naturally drive garment deformation. Since garments do not possess an intrinsic skeleton, a standard practice is to transfer the skinning weights from the underlying SMPL body to the garment vertices to serve as a strong geometric prior for articulation. While the naive nearest-neighbor transfer works reasonably well for tight-fitting clothing, defining appropriate skinning weights for loose garments remains a significant challenge. To enable animation and motion-driven applications, the AMASS Mahmood et al. [2019], Punnakkal et al. [2021] motion capture dataset was introduced, which parametrizes diverse human motions using SMPL's parameter space. This comprehensive database of SMPL-parameterized motion data has become instrumental for animation and dynamic simulation tasks.

3.4 Cloth Simulation as Optimization

A central insight that connects classical physics-based simulation to modern learning-based approaches is that the implicit time integration step can be reformulated as an energy minimization problem Martin et al. [2011], Gast et al. [2015]. This section derives the optimization-based formulation that forms the theoretical basis for the self-supervised neural simulation methods reviewed in Section 2.2.

3.4.1 Time Integration

To numerically simulate time-dependent processes, the continuous time domain is partitioned into a sequence of uniform discrete steps. The simulation time at the n-th step is denoted by t_n and corresponds to a *time step*. The temporal resolution is defined by the *time step size* Δt, given by $\Delta t = t_{n+1} - t_n$. The timestep index $n \in \mathbb{N}$ starts from zero, so that $t_0 = 0\,\mathrm{s}$ represents the initial time of the simulation.

In the discrete setting, we use the time variable t and the step index n interchangeably as subscripts. For instance, $\mathbf{x}_t$, $\mathbf{v}_t$, and $\mathbf{f}_t$ denote the position, velocity, and force vectors at the n-th time step (i.e., at time t_n).

As the simplest form of implicit integration, the implicit backward Euler time

integration is given by:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \Delta t \mathbf{v}_{t+1} \tag{21}$$
$$\mathbf{v}_{t+1} = \mathbf{v}_t + \Delta t M^{-1} \mathbf{f}_{t+1} \tag{22}$$

where M is the mass matrix.

3.4.2 Optimization-Based Reformulation

Combining Eqs. (21) to (22) by eliminating $\mathbf{v}_{t+1}$ gives:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{x}_t + \Delta t(\mathbf{v}_t + \Delta t M^{-1}\mathbf{f}_{t+1}) \\ &= \mathbf{x}_t + \Delta t \mathbf{v}_t + \Delta t^2 M^{-1}\mathbf{f}_{t+1} \end{aligned} \tag{23}$$

which can be rephrased as:

$$M(\mathbf{x}_{t+1} - (\mathbf{x}_t + \Delta t \mathbf{v}_t)) - \Delta t^2 \mathbf{f}_{t+1} = \mathbf{0} \tag{24}$$

Letting $\hat{\mathbf{x}}_t = \mathbf{x}_t + \Delta t \mathbf{v}_t$, we can rewrite the above equation as:

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}} \mathcal{E}(\mathbf{x}) \tag{25}$$

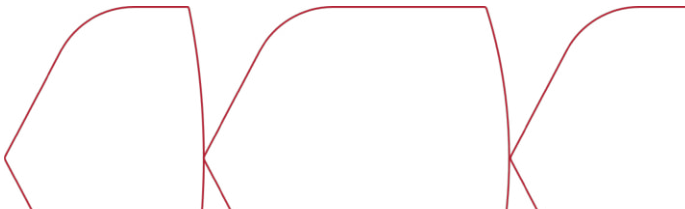
where

$$\mathcal{E}(\mathbf{x}) = \frac{1}{2\Delta t^2}\|\mathbf{x} - \hat{\mathbf{x}}_t\|_M^2 + P(\mathbf{x}) \tag{26}$$

Here, $\frac{1}{2\Delta t^2}\|\mathbf{x} - \hat{\mathbf{x}}_t\|_M^2 = \frac{1}{2\Delta t^2}(\mathbf{x} - \hat{\mathbf{x}}_t)^T M(\mathbf{x} - \hat{\mathbf{x}}_t)$ is the inertia term, and $P(\mathbf{x})$ is the potential energy for internal and external forces $\mathbf{f}$.

It is worth noting that the inertia term $\frac{1}{2\Delta t^2}\|\mathbf{x} - \hat{\mathbf{x}}_t\|_M^2$ can be equivalently reformulated in terms of acceleration $\mathbf{a}$ as $\frac{1}{2}\Delta t^2 \langle \mathbf{a},\ M\mathbf{a} \rangle$, where $\mathbf{a} = (\mathbf{x} - \hat{\mathbf{x}}_t)/\Delta t^2$.

This optimization perspective is fundamental: first, it allows us to improve both the efficiency and robustness of the solvers by leveraging advanced optimization techniques. Furthermore, by training a neural network to directly minimize this physics-based energy $\mathcal{E}(\mathbf{x})$, one obtains a differentiable neural simulator without



requiring simulation data. This relaxes the dependency on large-scale datasets generated by classical simulators for training supervised models, while the per-time-step numerical integration is replaced by a single forward network pass.

3.4.3 Collision Handling

Beyond elastic energy, cloth simulation often needs to account for collisions between the garment and the body, as well as self-collisions within the garment itself. In classical simulation, collision handling is decomposed into two stages: *collision detection*, which identifies geometric primitives (vertices, edges, triangles) that are in close proximity or interpenetrating, and *collision response*, which resolves the detected intersections by adjusting positions or velocities. In the detection stage, a *broad phase* first prunes the search space using spatial data structures such as Bounding Volume Hierarchies (BVH) Klosowski et al. [1998], followed by a *narrow phase* resolution. For dynamic scenarios, Continuous Collision Detection (CCD) Provot [1997] can be employed to prevent tunneling artifacts caused by large time steps.

In learning-based methods, however, this detect-then-resolve pipeline is replaced by differentiable penalty formulations that can be directly integrated into the optimization objective or training loss.

Cloth-body collision. A widely adopted formulation penalizes interpenetration using a power function of the penetration depth Santesteban et al. [2022b], Bertiche et al. [2022], Grigorev et al. [2023]:

$$\mathcal{L}_{\text{collision}} = \sum_{\text{vertices}} k_c \cdot d_c^p, \tag{27}$$

where k_c is a balancing factor, p is the penalty exponent that controls the sharpness of the collision response (typically $p = 2$ or $p = 3$), and $d_c = \max(\varepsilon - d(x), 0)$ quantifies the degree of interpenetration. Here, $d(x)$ is the signed distance between a garment vertex and the body surface, and the threshold distance ε is a small positive constant introduced to enhance stability.

Nevertheless, the soft penalty alone cannot guarantee collision-free results at inference time, and a post-processing step is typically required to resolve residual body-cloth intersections, see Algorithm 1. This leads to two drawbacks: (i) the inference time is significantly increased, and (ii) the garment is over-stretched to ensure non-penetration, compromising predicted garment details such as wrinkles

Algorithm 1 Post-processing cloth-body collision correction

Input: Cloth vertices $\mathbf{x}_c$, body vertices $\mathbf{x}_b$, body normals $\mathbf{n}_b$, penetration tolerance ε
Output: Collision-free cloth vertices $\mathbf{x}_c^{\text{fixed}}$

▷ Find nearest body surface vertices for each cloth vertex
$\mathbf{i} \leftarrow \textsc{NearestNeighbor}(\mathbf{x}_c, \mathbf{x}_b)$
$\mathbf{x}_b \leftarrow \mathbf{x}_b[\mathbf{i}]$, $\mathbf{n}_b \leftarrow \mathbf{n}_b[\mathbf{i}]$
▷ Compute penetration depth for each cloth vertex
$\mathbf{p} \leftarrow \langle \mathbf{n}_b, \mathbf{x}_c - \mathbf{x}_b \rangle - \varepsilon$
$\mathbf{p} \leftarrow \min(\mathbf{p}, 0)$
▷ Apply normal-based corrective displacement
$\Delta\mathbf{x} \leftarrow -\mathbf{n}_b \odot \mathbf{p}$
$\mathbf{x}_c^{\text{fixed}} \leftarrow \mathbf{x}_c + \Delta\mathbf{x}$
return $\mathbf{x}_c^{\text{fixed}}$

and dynamics.

Remark 1. If objects move at sufficiently high velocities or if the timestep is chosen too large, collisions may fail to be detected, allowing objects to pass through each other between consecutive simulation steps. Throughout this thesis, we assume that the timestep is small enough such that these situations do not arise.

Self-collision. Self-collision detection has $\mathcal{O}(N^2)$ complexity. Similar to cloth-body collision, learning-based methods formulate it as a penalty term in the loss function. The repulsive loss Lee et al. [2023] effectively handles self-collision by constraining non-adjacent vertices to maintain sufficient distance:

$$\mathcal{L}_{\text{rep}} = \lambda_{\text{rep}} \sum_{i=1}^{N} \sum_{j \in \mathcal{A}_i} -\log\big(\|\mathbf{x}_i - \mathbf{x}_j\|^2\big), \tag{28}$$

where $\mathcal{A}_i \coloneqq \{\, j \nsim i : \|\mathbf{x}_i - \mathbf{x}_j\| < \delta \,\}$, $(j \nsim i)$ denotes non-adjacent vertices, and δ is a pre-defined threshold.

3.5 Fourier Neural Operators Formulation

Section 2.4 introduced Neural Operators as a powerful paradigm for overcoming the resolution dependence of standard deep learning architectures. Building upon that context, this section details their mathematical foundation. To solve a general partial differential equation (PDE), neural operators learn the solution operator of

an entire PDE family directly in infinite-dimensional function spaces. They establish a direct mapping from parametric input conditions (e.g., initial states, boundary constraints, or physical coefficients) to continuous solution fields by capturing global spatial interactions through an iterative kernel integral operator $\mathcal{K}$, formulated as:

$$\mathcal{K}(v)(x) = \int_D \kappa(x, y)v(y)\, dy, \tag{29}$$

where v is the latent feature representation of the input function and κ is a learnable kernel function. Fundamentally, the design and parameterization of the kernel function $\kappa(x, y)$ encode specific physical and geometric inductive biases, giving rise to distinct families of neural operators. For instance, without imposing any geometric priors, parameterizing the kernel via multi-layer perceptrons leads to the Graph Neural Operator (GNO) Li et al. [2020], which is highly flexible for irregular meshes but computationally expensive. Constraining the kernel to a separable form yields the DeepONet architecture Lu et al. [2019]. Furthermore, introducing data-dependent kernels extends self-attention mechanisms to function spaces, resulting in Transformer Neural Operators Hao et al. [2023].

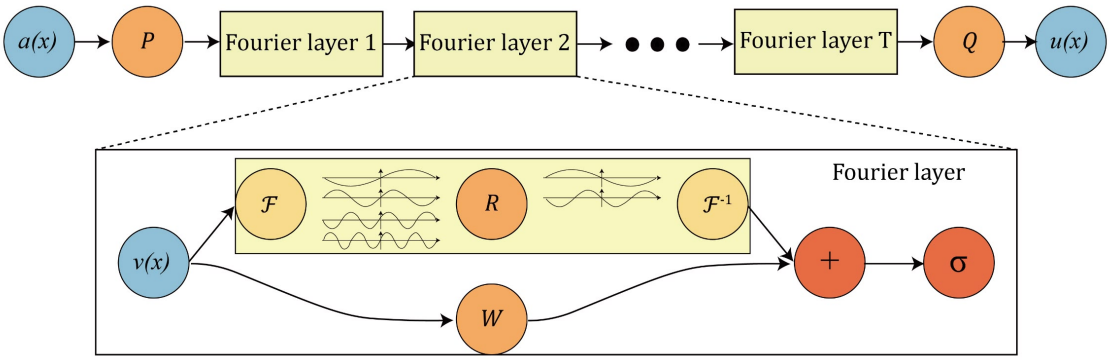


FIGURE 3.2 : The Fourier Neural Operator (FNO) architecture (image from Li et al. [2021]). The input function $a(x)$ is first lifted to a higher-dimensional latent representation by a pointwise neural network P. The representation is then updated through L successive Fourier layers. Within each Fourier layer, the input is processed by two parallel branches: a global spectral branch that computes the Fast Fourier Transform ($\mathcal{F}$), filters the signal using a learnable complex-valued weight tensor R, and applies the Inverse Fourier Transform ($\mathcal{F}^{-1}$); and a local spatial branch that applies a linear transform W. Their outputs are added and passed through a non-linear activation σ. Finally, a projection network Q maps the latent representation to the target output dimension.

Among these diverse architectures, imposing translation invariance leads to

the Fourier Neural Operator (FNO) Li et al. [2021], whose overall architecture is illustrated in Figure 3.2. By evaluating global spatial interactions efficiently in the frequency domain, FNO has emerged as one of the most successful neural operators for solving complex partial differential equations.

Continuous Operator Learning and the Convolution Theorem. If we impose a physical assumption that the interaction between two points in space depends solely on their relative distance (translation invariance), the kernel function simplifies to $\kappa(x, y) = \kappa(x - y)$. Mathematically, the integral then strictly transforms into a standard spatial convolution $\kappa * v$. However, computing global convolutions directly in the spatial domain on high-resolution grids is computationally prohibitive. The FNO elegantly resolves this bottleneck by exploiting the Convolution Theorem: a convolution in the spatial domain is strictly equivalent to a pointwise multiplication in the frequency domain.

Therefore, FNO abandons learning the complex kernel κ directly in the spatial domain. Instead, it utilizes the Fast Fourier Transform (FFT) to convert the signal into the frequency domain, where it directly learns a complex-valued weight tensor R_ϕ. This enables FNO to achieve global integration with a highly efficient computational complexity of $\mathcal{O}(N \log N)$.

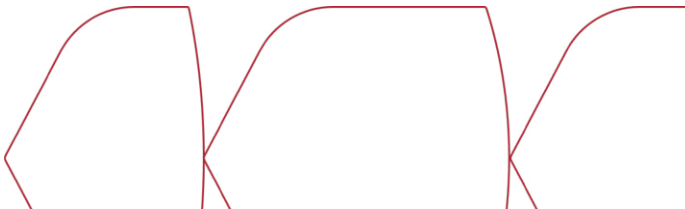
Physical and Mathematical Significance of Frequency Truncation. In the frequency domain, FNO does not retain all frequency modes. Instead, it applies a manual truncation, keeping only the lowest $k_{\max}$ frequency modes for the multiplication with R_ϕ, while filtering out the remaining high-frequency components. This design carries profound physical and mathematical significance. In the vast majority of PDE-driven physical systems, the dominant energy and macroscopic global structures are typically concentrated in the low-frequency spectrum. Conversely, the extremely high-frequency components often represent minor local fluctuations or numerical noise induced by discretization. Truncating high frequencies not only maps the infinite-dimensional operator to a compact, finite-dimensional parameter space but also serves as an implicit regularization mechanism, effectively preventing the model from overfitting to the discrete high-frequency noise of specific grids.

Complementarity of the Global and Local Branches. Within each Fourier layer, the latent representation splits into two parallel branches. Alongside the FFT-based global spectral branch, there is a local linear branch W (e.g., a 1×1 convolution or pointwise MLP) operating directly in the spatial domain. The complementarity

between these two branches is crucial: the Fourier transform fundamentally assumes that the signal is globally periodic; however, practical physical boundaries are often non-periodic. The W branch recovers the high-frequency spatial details lost due to the spectral truncation. The outputs of the two branches are subsequently merged via a residual connection.

Resolution Agnosticism. Benefiting from its parameterization in the frequency domain, FNO is inherently resolution-agnostic. The learnable weight R_ϕ is defined exclusively on a fixed, finite number of low-frequency modes. This implies that regardless of how finely the spatial domain is discretized during inference (i.e., the magnitude of the grid N), the parameter dimension of R_ϕ remains strictly unchanged. To perform inference at a higher resolution, the model simply applies zero-padding in the frequency domain and samples more spatial points during the inverse Fast Fourier Transform ($\mathcal{F}^{-1}$). This underlying mathematical principle guarantees the model's capacity to perform zero-shot super-resolution evaluations without requiring any modification to the network parameters.

Advantages in Cloth Dynamics Simulation. In the context of cloth and soft-body physical simulations investigated in this thesis, the propagation of deformation and stress often exhibits long-range dependencies. For instance, tugging on a single corner of a garment instantaneously transmits wrinkles and forces across the entire fabric. Traditional CNNs or GNNs often struggle with such problems due to their localized receptive fields or the prohibitive computational complexity associated with long-range graph message passing. In contrast, the FNO architecture intrinsically possesses a global receptive field via its frequency-domain computation, enabling it to capture long-range physical interactions instantaneously. Furthermore, its resolution-agnostic property allows us to efficiently train the model on low-resolution grids and directly generate cloth dynamics with rich and realistic wrinkle details on high-resolution meshes during inference. Consequently, FNO provides a promising framework for building generalizable cloth dynamics solvers, which forms the architectural foundation of our FNOpt method detailed in Section 6.



CENTRALE LYON

LIRIS

CHAPTER 4

GAPS: Geometry-Aware, Physics-Based, Self-Supervised Neural Garment Draping

Summary

Recent neural, physics-based modeling of garment deformations allows faster and visually aesthetic results as opposed to the existing methods. Material-specific parameters are used by the formulation to control the garment inextensibility. This delivers unrealistic results with physically implausible stretching. Oftentimes, the draped garment is pushed inside the body which is either corrected by an expensive post-processing, thus adding to further inconsistent stretching; or by deploying a separate training regime for each body type, restricting its scalability. Additionally, the flawed skinning process deployed by existing methods produces incorrect results on loose garments.

In this chapter, we introduce a geometrical constraint to the existing formulation that is collision-aware and imposes garment inextensibility wherever possible. Thus, we obtain realistic results where draped clothes stretch only while covering bigger body regions. Furthermore, we propose a geometry-aware garment skinning method by defining a body-garment closeness measure which works for all garment types, especially the loose ones. [1]

1. Code is publicly available at https://github.com/Simonhfls/GAPS.

4.1 Introduction

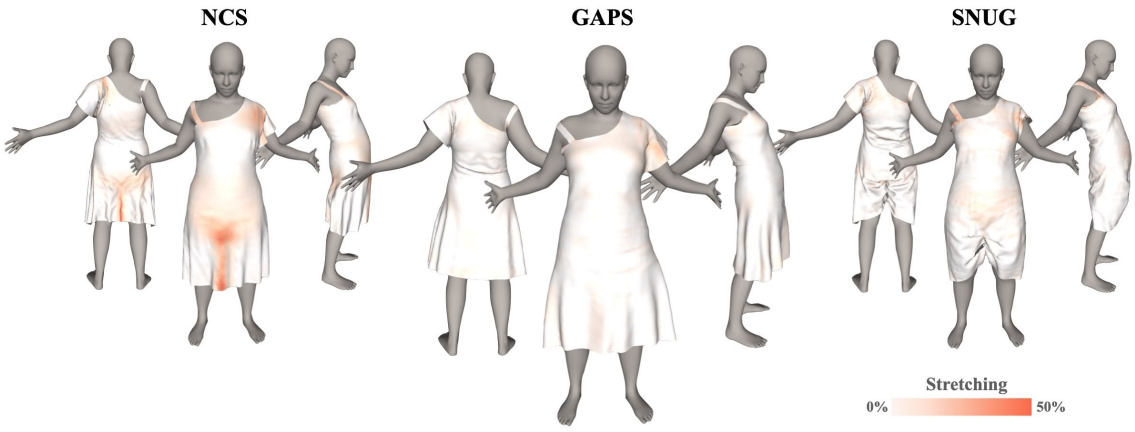


FIGURE 4.1 : SNUG Santesteban et al. [2022b] and NCS Bertiche et al. [2022] perform self-supervised draping of garments. However, they are prone to unrealistic stretching arising due to their efforts to prevent garment-body collisions. To overcome this issue, we propose GAPS which performs geometrically aware mechanism to drape a garment by stretching only if necessary. This controls body-garment collisions automatically without any post-processing Santesteban et al. [2022b] or restrictive measures Bertiche et al. [2022]. Furthermore, our geometry-aware skinning allows better handling of loose garments.

Modeling digital garments for real-time applications, such as video game simulations, fashion design, and e-commerce virtual try-on systems is a challenging yet important problem to solve. Traditionally, clothes are modeled using Physics-Based Simulation (PBS) Nealen et al. [2006], Baraff and Witkin [1998], Narain et al. [2012] of continuum fabric Macklin et al. [2016] or highly detailed yarn Cirio et al. [2014] as per Newton's laws of motion. While these techniques are capable of delivering precise simulations, their high computational complexity makes them unsuitable for real-time and web-based applications.

With the success of deep learning in major 3D learning tasks Qi et al. [2017a,b], Soltani et al. [2017], Omran et al. [2018], Richardson et al. [2016], supervised learning of garment draping Gundogdu et al. [2020], Tiwari et al. [2020], Pfaff et al. [2021], Wang et al. [2019], Zhang et al. [2021b], Patel et al. [2020], Corona et al. [2021] gained interest. They used a neural network fusing a given garment and a body (a joint pose and shape representation generally parametrized using SMPL Loper et al. [2015]) to produce drapings similar to PBS, but at a faster rate.

A major limitation of these supervised approaches is that the supervision requires high quality 3D meshes of draped garments over various bodies which not only

requires expensive setup (3D scanners and multi-view cameras) but also a huge amount of manual intervention. In addition, all these methods leave significant smoothing artefacts that often result in visually unaesthetic drapings.

Recently, Bertiche et al. [2021] proposed a self-supervised learning of garment draping by enforcing a static physical consistency of garments while training. Inspired by Martin et al. [2011], Santesteban et al. [2022b], Bertiche et al. [2022] reformulated motion constraints in PBS as an optimization problem and enforced physical consistency on both static and dynamic garment deformations. Not only such a formulation is computationally more efficient than PBS and supervised methods but it also does not incur smoothing artefacts. However, a major shortcoming of Santesteban et al. [2022b], Bertiche et al. [2022] is the assumption that the plasticity or elasticity of worn clothes depends solely on material properties. When a garment fits onto a body, it remains inextensible in most parts and stretches only when wrapped over bigger body regions. Current models use only material parameters and limit the strain and bending forces to allow only small changes. This causes two major problems: 1) Unrealistic fittings with physically implausible local stretches that are incoherent with the physical nature of clothes, see Fig. 6.1. 2) Unsolvable body-garment collisions due to the naive strain/bending minimization. To overcome 2), Santesteban et al. [2022b] relies on heavy post-processing that forcefully pushes the collided garment vertex outside the body. This impairs run-time performance and contributes to further incoherent stretching. Bertiche et al. [2022] uses data augmentation to reduce collisions and limits the training to single body shape. This is impractical as it requires to re-train for each body shape. It leads to longer training duration (reported up to 24 hours for loose garments such as dresses). Santesteban et al. [2022b] simultaneously trains to drape a garment over several body types. However, it does not handle cloth dynamics well. In addition, current methods use flawed garment-skinning whose accuracy is dependent on the tightness of the body which degrades their performance poorly while draping loose garments as seen Fig. 6.1.

In this chapter, we propose GAPS: a geometry-aware, physics-based, self-supervised neural garment draping method to fix the above-mentioned problems. Geometry-based deformation modeling is widely popular in many computer vision problems such as 3D reconstruction Parashar et al. [2020b, 2021], Bednarik et al. [2020], Perriollat et al. [2008] and shape matching Bednarik et al. [2021], Donati et al. [2020]. To our best knowledge, GAPS is the first neural method that combines physics-based

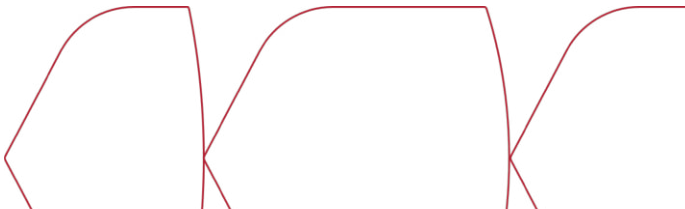
formulation with geometrical constraints on deformations to force collision-aware garment inextensibility. It is built upon Santesteban et al. [2022b] and imposes inextensibility by preserving local distances and areas, which is only progressively relaxed in case of body-garment collisions. Consequently, it replicates the true cloth behaviour: maintains inextensibility, stretches only over bigger body regions and does not penetrate the body. Furthermore, we propose a novel, geometry-aware skinning method demonstrates strong adaptability across a wide range of garment topologies as seen Fig. 6.1. Existing point-based methods Saito et al. [2021a], Chen et al. [2021] learn a backward transformation field between posed body and canonical pose which fails on loose garments such as skirts. More advanced strategies such as Ma et al. [2022] learn implicit neural garment body interactions but they don't generalise well. Contrary to these works, our proposed skinning is rather simple and able to generalise well. Our experiments demonstrate a significant qualitative and quantitative improvement over state-of-the-art methods.

4.2 Geometry-Aware Modeling

We now discuss our main contributions. First, we describe geometry-based deformation modeling which together with the existing physics-based modeling allows us to learn true cloth behaviour. Then, we describe the geometry-aware method to compute body-participation in garment dynamics to compute garment skinning. This allows efficient draping of all types of garments.

4.2.1 Garment Deformation Modeling

Even if the garments are made of highly stretchable material, they stretch only while fitting over larger body regions. Depending on the body shape and the dynamic motion, the stretching is generally localized to small areas while the majority of the draped garment remains inextensible; thus preserving local lengths, angles and areas. Geometrically, surfaces that deform while preserving these local properties are defined using *isometry* Lee [2003]: a geodesic-preserving map on smooth manifolds. It is imposed by preserving *metric tensors* which are locally defined using surface jacobians. While this differential modeling delivers promising results, the computation of local jacobians and hessians leads to slow neural networks Bednarik et al. [2020, 2021], Das et al. [2022], which is not desirable. We represent garments using meshes



which are a discrete representation of a continuous space. Therefore, we relax isometry by enforcing inextensibility locally on each mesh vertex. Given $\mathcal{X} = (x_i)_{i=1}^{N}$, a mesh containing N vertices x_i that transforms into $\mathcal{Y} = (y_i)_{i=1}^{N}$, a naive inextensibility imposition would be force each edge around y_i within its local neigborhood $y_j, j \in \mathcal{N}_i$ to comply with the corresponding one around x_i in terms of length. Consequently, we impose the constraint that

$$D_i = \sum_{j \in \mathcal{N}_i} (\|x_i - x_j\| - \|y_i - y_j\|)^2 = 0. \tag{30}$$

While imposing this constraint minimizes inextensibility, it delivers jittery results. This is because under this scheme, each edge length will be optimized independently causing slight jitters.

Instead, we impose a restriction on covariance matrix. For x_i, it is given by

$$\mathbf{C}_{x_i} = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} (x_j - \overline{x}_i)(x_j - \overline{x}_i)^\top, \tag{31}$$

where $\overline{x}_i$ is the mean of x_j. We impose inextensibility as local rigidity around each vertex. Given that $y_i = \mathbf{R}x_i + \mathbf{T}$, where $\mathbf{R}$ and $\mathbf{T}$ are a 3D rotation and a translation respectively, we assume that $y_j = \mathbf{R}x_j + \mathbf{T}$. Thus, we obtain

$$\mathbf{C}_{y_i} = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} (\mathbf{R}x_j - \mathbf{R}\overline{x}_i)(\mathbf{R}x_j - \mathbf{R}\overline{x}_i)^\top = \mathbf{R}\mathbf{C}_{x_i}\mathbf{R}^\top. \tag{32}$$

Using SVD, we can write $\mathbf{C}_{x_i} = \mathbf{U}\mathbf{S}_{x_i}\mathbf{U}^\top$. It can be easily verified that $\mathbf{C}_{y_i} = \mathbf{R}\mathbf{U}\mathbf{S}_{x_i}\mathbf{U}^\top\mathbf{R}^\top = \mathbf{V}\mathbf{S}_{x_i}\mathbf{V}^\top$. Thus inextensibility implies preservation of singular values of corresponding covariance matrix. It implies that $\sigma_i \in \mathbf{S}_{x_i} = diag(\sigma_1, \sigma_2, \sigma_3)$ will satisfy the characteristic polynomial of $\mathbf{C}_{y_i}$, i.e,

$$\det(\mathbf{C}_{y_i} - \alpha\sigma_i\mathbf{I}_{3\times3}) = 0, \tag{33}$$

where $\mathbf{I}_{3\times3}$ is identity and $\alpha = 1$. We use the above constraint to enforce realistic cloth behaviour. It has 3 benefits: 1) σ_i are derived from template garment and therefore, can be precomputed. 2) We avoid computing singular values of the garment in current timestep during optimization, which is expensive and may suffer from numerical instabilities due to vanishing gradients. 3) We can control the degree of stretching

just by changing α. In section 6.2, we will explain how α can be progressively changed to stretch the garment in order to avoid body collisions.

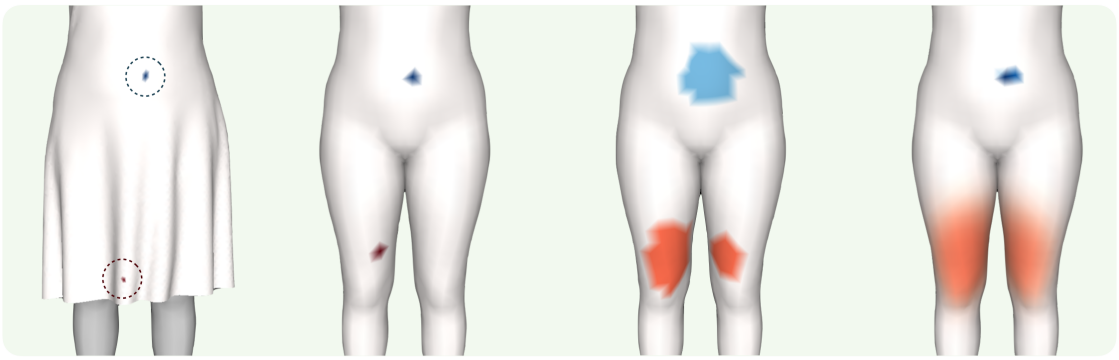


FIGURE 4.2 : Body-participation computation for garment vertices close (in blue) or far (in red) to the body.

4.2.2 Body-Participation In Garment Dynamics

In order to dynamically fit a garment over a body, it is required to compute entire body's participation towards the movement of each garment vertex. As described in Loper et al. [2015]: for each body vertex, it can be computed as a weighted blend of body-joints, with weights describing the impact of body-joints towards the motion at a given vertex. Assuming that the garments follow body motions, Bertiche et al. [2021], Santesteban et al. [2022b], Bertiche et al. [2022] assign each garment vertex the blending weights of its nearest body vertex directly. While this may be effective for tightly-fit garments such as T-shirts, it is extremely suboptimal for looser garments like dresses. In order to compensate, Bertiche et al. [2022] applies iterative Laplacian smoothing of the weights of each garment vertex w.r.t. its neighbours. On a loose garment, there is a high possibility for a garment vertex and its neighbours to match to the same body vertex, causing undesired artefacts.

To overcome these drawbacks, we introduce a simple, geometry-aware computation of blend weights premised on two fundamental observations: 1) For a given garment vertex, the influence of the blend weights of a body vertex is negatively correlated with their respective Euclidean distances. 2) On loose garments, the blend weights are less likely to depend on the proximate body vertex. We use the Gaussian Radial Basis Function (RBF) to compute the participation of each body vertex to a specific

garment vertex. We write a Gaussian RBF kernel as

$$\varphi(r) = e^{-\frac{r^2}{km_i^2}}, \tag{34}$$

where m_i is the Euclidean distance between a garment vertex $g_i \in \mathcal{G}$ and its nearest body vertex. Then, we define a participation matrix $\mathcal{P}$ where each element p_{ij} signifies the degree of participation of body vertex $b_j \in \mathcal{B}$ in g_i as

$$\mathcal{P} = \begin{bmatrix} \varphi(d_{11} - m_1) & \dots & \varphi(d_{1N_b} - m_1) \\ \vdots & \ddots & \vdots \\ \varphi(d_{N_g1} - m_{N_g}) & \dots & \varphi(d_{N_gN_b} - m_{N_g}) \end{bmatrix} \tag{35}$$

where d_{ij} represents the Euclidean distance between g_i and b_j, N_g and N_b are the number of elements in $\mathcal{G}$ and $\mathcal{B}$ respectively and $k = 0.5$. The new garment blend weights, $\tilde{\mathcal{W}}$, are then given by

$$\tilde{\mathcal{W}} = \text{normalize}(\mathcal{P} * \mathcal{W}), \tag{36}$$

where $\mathcal{W}$ are the blend weights described in Loper et al. [2015]. The normalization makes the cumulative sum of each row to 1.

Fig. 4.2 shows the body participation measures for garment vertices that drape close (in blue) or far (in red) to the body. Picking the closest body vertex and considering its blend weights might work for garment vertices close to the body. However, for the vertices on looser regions, it is clear that motion dynamics is dependent on a larger body region. Naively choosing k-nearest neighbours might work for vertices on looser regions but not on the tight ones. Our simple and effective body-participation method can automatically find the best-fitting region for any type of garment, irrespective of its topology which allows flawless garment skinning.

4.3 Method

We integrate our geometric formulations with PBS principles to improve the performance of learning-based cloth simulations.

4.3.1 Model

Our model is developed within an unsupervised learning framework proposed in Santesteban et al. [2022b]. It incorporates a Gated Recurrent Unit (GRU) module R outputting vertex displacement relative to the template garment $\mathcal{T}$ (unposed and undeformed). This GRU module consists of four layers, each yielding an output size of 256, with tanh activation functions. After combining it with $\mathcal{T}$ to obtain the unposed, deformed garment ($\mathcal{M}_t$), we subsequently apply the skinning function S which articulates $\mathcal{M}_t$. The garment blend weights $\tilde{\mathcal{W}}$ for skinning are estimated using Body-Participation module, described in section 4.2.2. The network is trained by optimizing a composite loss function, which includes our newly proposed inextensibility loss (described in section 4.2.1) in addition to the PBS losses. The architecture of our model is depicted in Fig. 4.3.

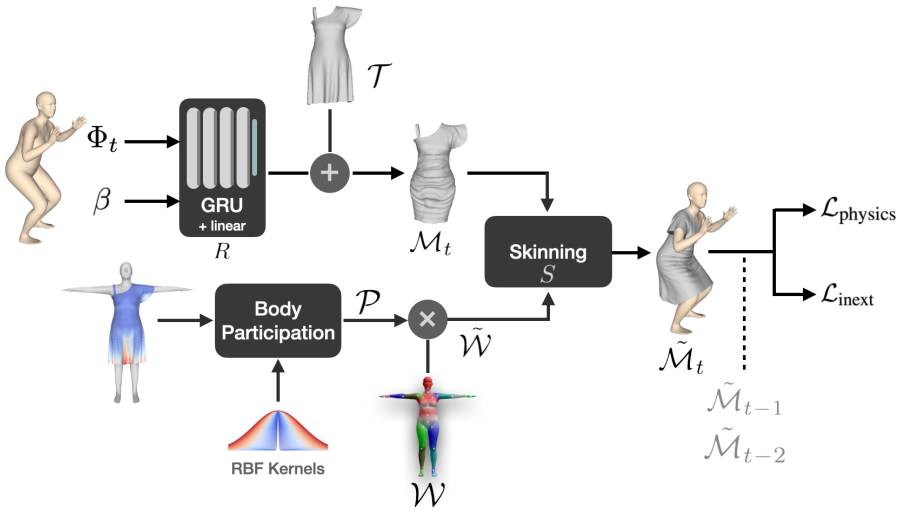


FIGURE 4.3 : Overview of our method. We added RBF based Body Participation module on Santesteban et al. [2022b] to estimate blend weights based on relative garment-body geometry.

The garment is modeled as

$$\begin{aligned} \mathcal{M}_t &= \mathcal{T} + R(\Phi_t, \beta), \\ \tilde{\mathcal{M}}_t &= S(\mathcal{M}_t, J(\beta), \theta_t, \tilde{\mathcal{W}}), \end{aligned} \tag{37}$$

where the regressor R takes as input the body shape β and the motion parameter $\Phi_t = \{\theta_t, v_t\}$ containing the current body pose θ_t and global velocity v_t of the root

joint. The skinning function S articulates $\mathcal{M}_t$ based on joint locations $J(\beta)$, current body pose and the estimated garment blend weights $\tilde{\mathcal{W}}$ using the Body-Participation module.

4.3.2 Losses

In order to learn the garment deformations, we know that the physical equilibrium of forces related to strain, gravity, collision and bending (defined in PBS) should be achieved. Like Santesteban et al. [2022b], we enforce them as losses to be minimized. We express them as

1) Strain. We use the Saint-Venant–Kirchhoff (StVK) material model for simulating deformation. It is given by

$$\mathcal{L}_{\text{strain}} = \mu \|\mathbf{E}\|_F^2 + \frac{\lambda}{2}\text{tr}(\mathbf{E})^2, \tag{38}$$

where $\mathbf{E} = \frac{1}{2}(\mathbf{F}^\top\mathbf{F} - \mathbf{I})$ represents the Green-Lagrange strain tensor (see Equation (7)). μ and λ are the Lamé coefficients reflecting the material properties.

2) Gravity. We incorporate gravity by minimizing the potential energy of the garment, given by

$$\mathcal{L}_{\text{gravity}} = \sum_{\text{vertices}} -mg^\top x, \tag{39}$$

where m is the particle mass and g is the gravitational acceleration.

3) Collision. It penalizes penetration between the body and the garment. For each garment vertex, it is given by

$$\mathcal{L}_{\text{collision}} = \sum_{\text{vertices}} k_c * d_c^3, \tag{40}$$

where k_c is a balancing factor, and $d_c = \max(\epsilon - d(x), 0)$ quantifies the degree of interpenetration. $d(x)$ is the signed distance between garment vertex and body surface, and ϵ is a small positive constant introduced to enhance stability.

4) Bending. We use a different formulation than Santesteban et al. [2022b], Bertiche et al. [2022]. We express it as a balance between smoothness in local garment areas and global coherence with the original template (to retain natural

drape characteristics). It is given by

$$\mathcal{L}_{\text{bending}} = \sum_{\text{edges}} k_b \frac{l^2}{8a}(\alpha(\theta_t - \theta_r)^2 + (1-\alpha)\theta_t^2), \tag{41}$$

where k_b is the bending stiffness, θ_t denotes the dihedral angle of the deformed garment, θ_r signifies the corresponding dihedral angle in the rest state of the template garment, l is the common edge length and a is the summation of the areas of both faces. The balancing coefficient $\alpha \in [0, 1]$ is proportional to the distance between the garment and the current edge in the rest pose.

5) Inertia. In order to convert PBS to an optimisation problem, Santesteban et al. [2022b] proposed the intertia loss. It is given by

$$\mathcal{L}_{\text{inertia}} = \sum_{\text{vertices}} \frac{1}{2\Delta t^2} m(x - x^{[t-1]} - \Delta t v^{[t-1]})^2, \tag{42}$$

where Δt is the simulation time step, $x^{[t]}$ and $x^{[t-1]}$ specify the particle's position at times t and $t-1$, respectively. As suggested in Bertiche et al. [2022], we do not to back-propagate $\mathcal{L}_{\text{inertia}}$ through previous frames.

Together, all these physical losses are given by

$$\mathcal{L}_{\text{physics}} = \mathcal{L}_{\text{strain}} + \mathcal{L}_{\text{bending}} + \mathcal{L}_{\text{collision}} + \mathcal{L}_{\text{gravity}} + \mathcal{L}_{\text{inertia}}. \tag{43}$$

Inextensibility Loss. Our novel geometric loss that enforces inextensibility (as per eq. 33) within the one-ring local area associated with each vertex, is given by

$$\mathcal{L}_{\text{inext}} = k_i \sum_{\text{vertices}} \sum_{j=1}^{3} |\det(\mathbf{C} - k_{\text{ext}}\sigma_j \mathbf{I}_{3\times 3})|, \tag{44}$$

where k_i is a balancing factor, $\mathbf{C}$ is the covariance matrix within 1-ring neighbourhood and σ_j are the singular values of covariance matrix of the 1-ring neighbourhood around the corresponding point in garment template $\mathcal{T}$. k_{ext} decides the degree of extension: $k_{\text{ext}} = 1$ indicates inextensibility and $k_{\text{ext}} > 1$ indicates stretching. Given that garment stretches only while covering large body regions, k_{ext} should be 1 and changed only if the body-garment collision is detected. Moreover, to not be affected by slight perturbations, the garment should only be gradually stretched upon collision

detection. Considering this, we write

$$k_{\text{ext}} = 1 + \min(10d_c, 0.03) \times \min(e, 100), \tag{45}$$

where e is the current epoch. We first allow the network to stabilize through 100 epochs and then gradually stretch the garment locally in case of body-garment collisions.

The overall loss is

$$\mathcal{L} = \mathcal{L}_{\text{physics}} + \mathcal{L}_{\text{inext}}. \tag{46}$$

FIGURE 4.4 : T-shirt, Dress, Skirt, and Gown draped using GAPS.

4.4 Experiments

Training. During the training phase, the Gated Recurrent Unit (GRU) layers are initialized with random hidden states. The batch size is 128, the learning rate is set to $1e{-}3$ for the initial 10 epochs and $1e{-}4$ for the rest. The balancing weight for inextensibility loss k_i is set to $2e8$. All other hyperparameters as well as material parameters are set according to Santesteban et al. [2022b]. We randomly sample the shape parameters from $\mathcal{U}(-2, 2)$ for each batch. We select 60 train sequences and 5 validation sequences from AMASS dataset Mahmood et al. [2019] containing around 11000 poses. We compare with SNUG Santesteban et al. [2022b] and NCS Bertiche et al. [2022]: the only state-of-the-art methods closest to ours. For a fair comparison, we train them on the same motion sequences as ours. GAPS and SNUG are trained on variable bodies whereas NCS is trained on a single body. The training code for SNUG is unavailable so we have implemented it. Our training losses are slightly better than the reported numbers in the paper.

	T-shirt			Dress			Skirt			Gown		
	ε_e	ε_a	ε_c	ε_e	ε_a	ε_c	ε_e	ε_a	ε_c	ε_e	ε_a	ε_c
SNUG	7.815	13.796	5.631 (±2.453)	4.833	6.084	3.113 (±1.496)	3.062	4.063	0.365 (±0.788)	5.906	7.434	3.515 (±3.376)
NCS	3.615	5.311	0.522 (±0.513)	4.164	5.391	0.368 (±0.556)	4.386	5.677	2.076 (±0.668)	4.782	6.686	0.633 (±1.041)
GAPS	**3.343**	**5.290**	**0.116 (±0.420)**	**3.051**	**3.825**	**0.196 (±0.308)**	**2.337**	**3.019**	**0.026 (±0.131)**	**3.639**	**4.249**	**0.123 (±0.225)**

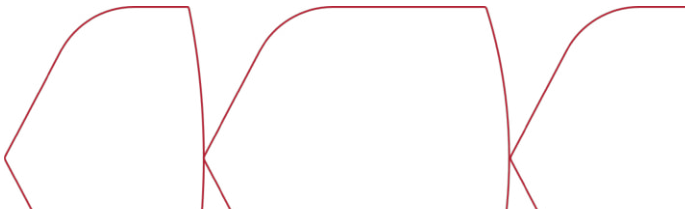
TABLE 4.1 : Inextensibility and Collision metrics on AMASS 86_07. GAPS fits all the garments with minimal collisions and minimal garment extensions.

Error metrics. We report ε_{e} and ε_{a} as the mean difference (in %) between the edge lengths and areas between the template and the draped garment. We report ε_{c} as the % of draped garment vertices colliding with the body.

Quantitative Evaluation. We consider the validation sequence AMASS 86_07 on a single body and drape four garments shown in Fig. 4.4. In general, each garment should be able to fit on this body without stretching. However, due to the dynamic motion of the body, stretching is inevitable. Tab. 4.1 shows the results. GAPS shows the best performance on all datasets. The drapings are obtained with minimal stretch and collisions, thanks to our collision-aware imposition of inextensibility and geometry-aware skinning. We remind that NCS performs body-specific draping. It is trained only on the body that we used in this experiment while others are trained on variable bodies. It performs well (close to ours) on T-shirt which is a simple, tight-fitting garment. On other garments, which are all more challenging than T-shirt, it degrades significantly. SNUG performs a post-processing to fix body-garment collisions which makes $\varepsilon_c = 0$ but it considerably increases ε_a and ε_e. We compute all metrics before post-processing, thus evaluating only the method's performance. Fig. 4.5 shows the ε_a and ε_c on the entire AMASS 86_07 sequence with a Dress draped on it. Unlike NCS, the stretches produced by GAPS seem to be quite realistic with minimal stretching in loose garment areas.

Performance on Diverse Bodies. Both GAPS and SNUG are trained on variable bodies. We compare their scalability. Fig. 4.6 shows diverse body shapes where SNUG shows strong body-garment penetrations. It heavily depends on post-processing to generate visually comparable results. In contrast, GAPS' collision-aware inextensibility loss allows visually remarkable results. We have omitted results for extremely thin bodies as they are similar for all methods.

Performance on Loose Garments. We revisit dress to compare the performance on loose garments. SNUG assumes that garment motion is close to the body's and uses the blend weights proposed in Loper et al. [2015] for garment vertices and therefore, fails while fitting loose garments. NCS uses a Laplacian smoothing to



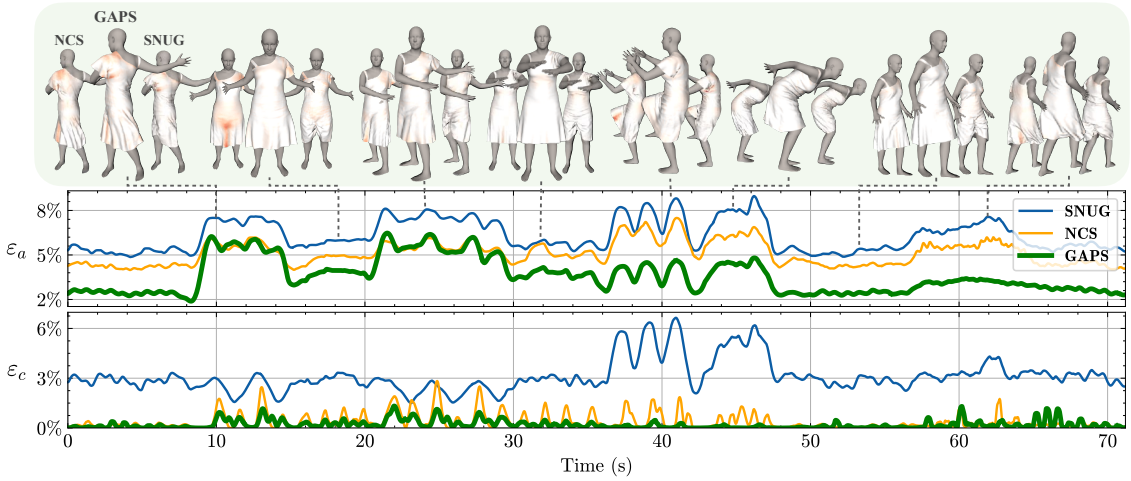


FIGURE 4.5 : Results on AMASS sequence 86_07 with Dress. GAPS shows the best performance with minimal collision and realistic stretching (indicated in orange).

estimate the blend weights better in order to fit loose garments. However, this requires an iteration parameter to be set. We fine-tune this parameter (set to 50) in order to get the best possible performance on the dress. Fig. 4.7 shows the results. SNUG can not deal with loose garments. On looser areas, we observed a big spike in strain loss which hinders convergence. NCS can not produce realistic results despite the careful selection of the smoothing iterations; the results are similar to SNUG. In contrast, GAPS' geometry-aware skinning is robust, generates realistic results quite close to the PBS (used in Santesteban et al. [2021]). Fig. 4.8 compares skinning of the template garment onto a posed body using different skinning methods. The closest vertex based skinning shows large discontinuities between the legs. The use of K-nearest vertices reduces these discontinuities, however with an inevitable compromise with accuracy due to the fixed number of neighbours. The Laplacian smoothing based skinning of NCS is similar to the k-nearest vertices one, which explains its degraded performance on loose garments.

Ablation Study. We study the individual impact of various components within the GAPS framework.

1) Collision-awareness. By fixing $k_{\text{ext}} = 1$ in $\mathcal{L}_{\text{inext}}$ (see eq. (44)), we impose the garment to remain intextensible everywhere. Tab. 4.2 shows the % of vertices in collision, ε_c. We see that incorporated collision-awareness in GAPS is crucial for it to drape realistically.

2) Inextensibility. Tab. 4.3 shows that explicit enforcement of inextensibility loss

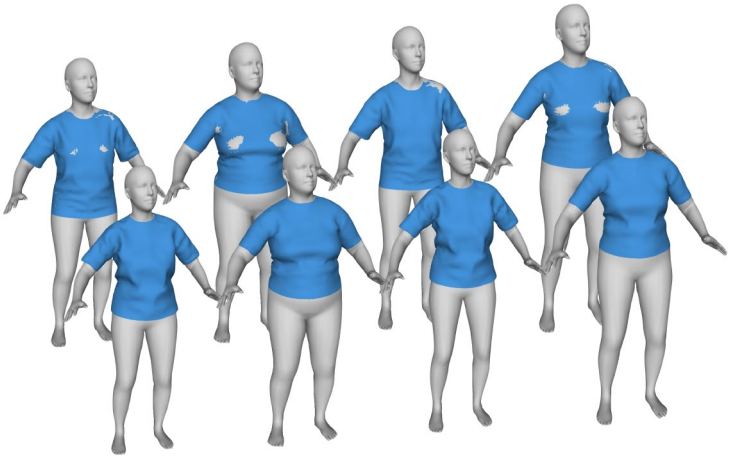


FIGURE 4.6 : Results on diverse bodies. SNUG (top) shows significant garment penetration while GAPS (bottom) shows remarkable performance.

	T-shirt	Dress	Skirt	Gown
GAPS	0.116	0.196	0.026	0.123
GAPS ($k_{\text{ext}} = 1$)	5.191	5.572	2.992	7.027
GAPS without $\mathcal{L}_{\text{inext}}$	5.344	5.924	3.105	7.580

TABLE 4.2 : The collision error ε_c evaluated across varying configurations demonstrates that incorporating collision-awareness is crucial for GAPS to effectively resolve body-garment intersections.

$\mathcal{L}_{\text{inext}}$ on top of physics-based losses $\mathcal{L}_{\text{physics}}$ forces GAPS to maintain inextensibility especially while draping over smaller bodies which are expected to be as inextensible as possible. This shows the complementary nature of strain loss, eq (38) and $\mathcal{L}_{\text{inext}}$ (44).

3) RBF-based skinning. As demonstrated in Fig. 4.9, the RBF-based skinning method employed in GAPS enhances realism for loose garments and generalizes well to tight garments. Note that draping results for tight garments, such as T-shirt, are similar whether the RBF-based skinning method is employed or not.

Timing Performance. SNUG takes less than 1 hour to converge for tight garments

	Avg body		Chubby		Obese	
	ε_e	ε_a	ε_e	ε_a	ε_e	ε_a
GAPS	3.05	3.83	4.32	5.06	6.41	6.44
No $\mathcal{L}_{\text{inext}}$	3.84	5.13	4.38	5.54	6.43	6.94

TABLE 4.3 : ε_e and ε_a evaluated on Dress across various body types shows the importance of inextensibility loss.

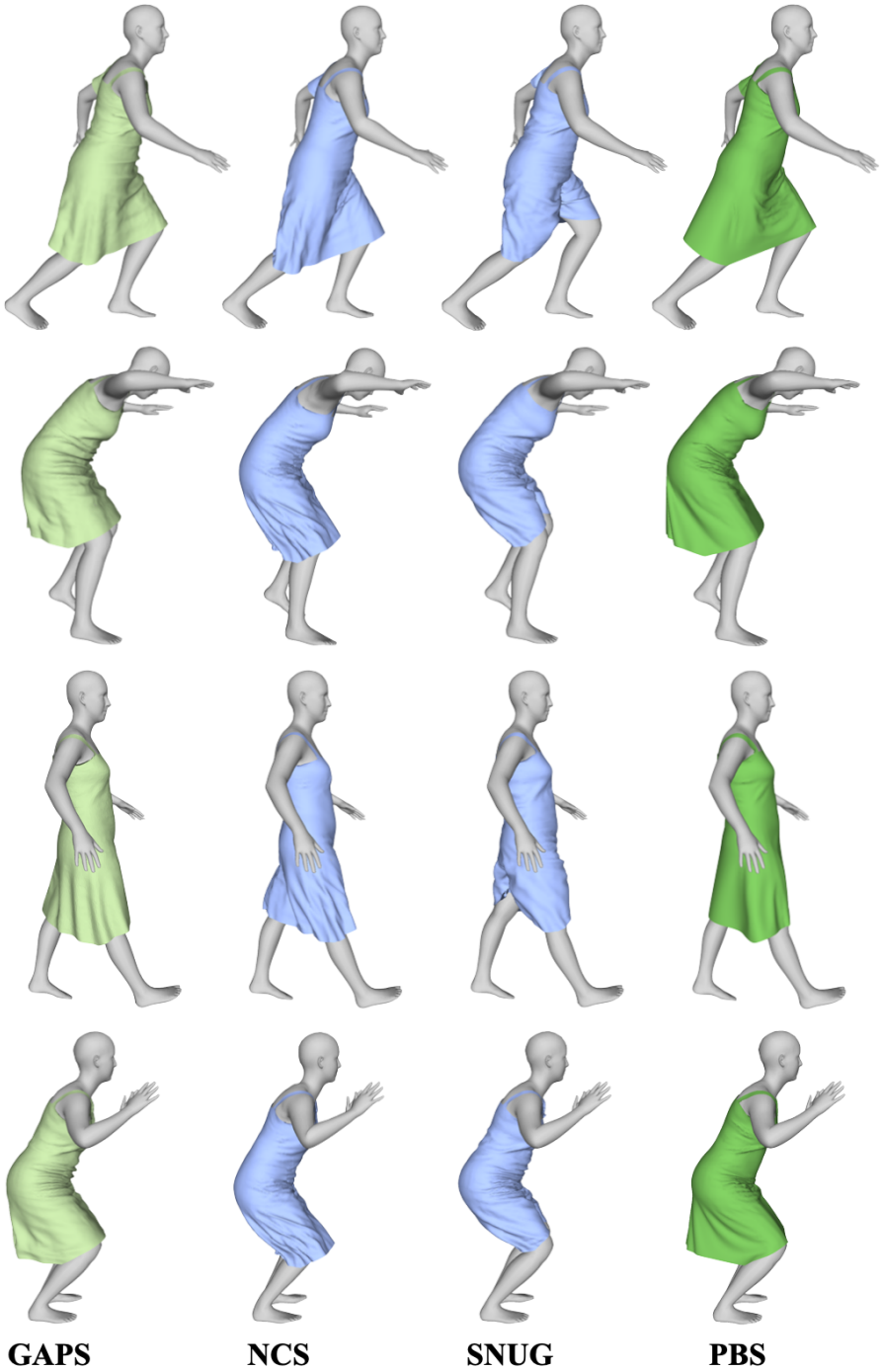


FIGURE 4.7 : Results on Dress. GAPS shows realistic results quite close to the PBS.

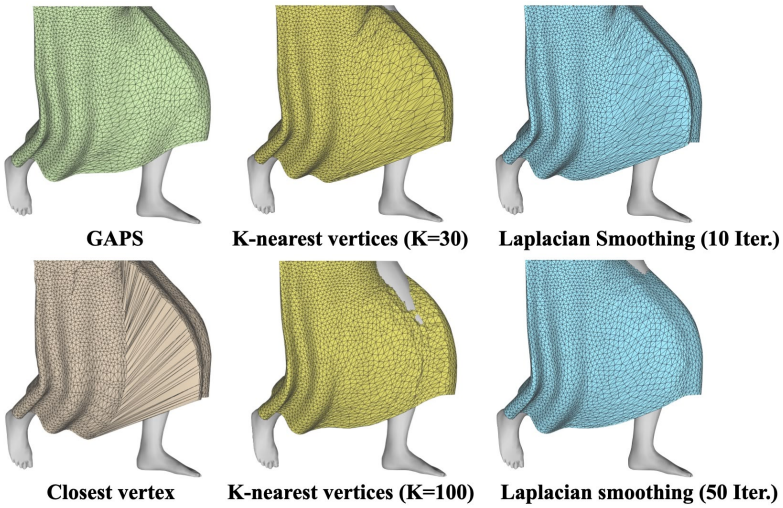


FIGURE 4.8 : Comparing the geometry-aware, RBF-based skinning in GAPS with others.

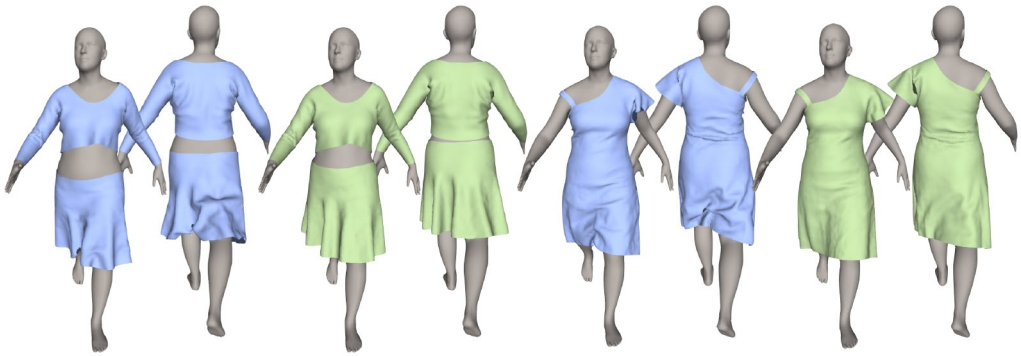


FIGURE 4.9 : GAPS with (in green) RBF-based skinning and without it (in blue).

with less than 10k vertices but will take a significantly longer time (up to 24 hours) for looser garments because of the spike in strain loss. NCS takes 1-24 hours. Our method takes 1 hour for tight garments and up to 8 hours for looser garments. All training is carried out on 8 NVIDIA V100 GPUs.

As for run-time performance, we measure the processing duration from the stage of raw body pose data to the final body and garment meshes. For a fair comparison, we use a CMU motion sequence comprising 2,175 frames to evaluate the runtime. The tests are executed on an Intel Xeon Gold 6146 and NVIDIA V100. For SNUG, we used the checkpoint and associated script provided by the author. For NCS, we use the author's script for training and prediction. Tab. 4.4 shows the comparison. SNUG takes greater runtime due to the post-processing. The authors suggested the possibility of parallelizing the collision post-processing component on the GPU to

	Train	Runtime
SNUG	1 - 24 h	23.7 ms
NCS	1 - 24 h	3.3 ms
GAPS	1-8 h	2.7 ms

TABLE 4.4 : Timing performance.

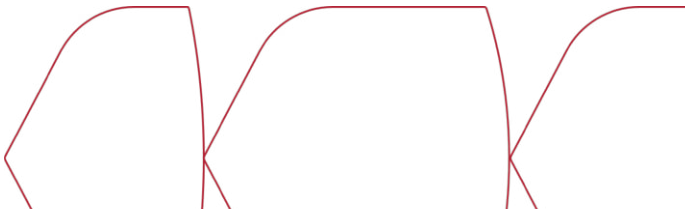
enhance runtime performance. However, their runtime GPU code is not publicly available. Our approach achieves the fastest performance without requiring a GPU. **Summary of methods.** SNUG is trained on various body shapes; it is scalable but relies on post-processing to overcome body-garment collisions, which are significant. It cannot handle loose garments. NCS is body-specific and trained on a single body; it is highly restrictive but it fits decently well with lesser collisions and smaller extensions than SNUG. It produces some unrealistic stretches and does not drape loose clothes well, due to its suboptimal skinning approach. GAPS is scalable, minimizes body-garment collisions and garment extensions. It produces realistic stretches and drapes all types of garments well.

Limitations. Though temporal neural networks have been incorporated in our approach, there is still a necessity for better handling of the cloth dynamics. Additionally, while the implementation of our collision-aware inextensibility loss has substantially mitigated body-garment collision issues, challenges persist in extreme body poses and shapes. Furthermore, our current framework does not yet address cloth self-collision and cannot deal with multi-layer and topology-varying clothing settings.

4.5 Conclusion

We presented GAPS: a geometry-aware, physics-based, self-supervised garment draping method. It incorporates explicit collision-aware inextensibility enforcement which encourages realistic drapings where the garment stretches only while fitting over larger body regions. Furthermore, the collision-awareness module significantly reduces body-garment collisions, eliminating the need for expensive post-processing or restrictive training. In addition, we proposed a geometry-aware skinning approach which automatically computes the right body-participation to a garment dynamics and therefore, can handle a variety of garments. This allows us to obtain significantly

better draping results, even on loose garments. Most importantly, our modeling is generic and can be easily incorporated with others.



CENTRALE LYON

LIRIS

CHAPTER 5

Patch-based Representation and Learning for Efficient Deformation Modeling

Summary

In this chapter, we present a patch-based representation of surfaces, PolyFit, which is obtained by fitting jet functions locally on surface patches. Such a representation can be learned efficiently in a supervised fashion from both analytic functions and real data. Once learned, it can be generalized to various types of surfaces. Using PolyFit, the surfaces can be efficiently deformed by updating a compact set of jet coefficients rather than optimizing per-vertex degrees of freedom for many downstream tasks in computer vision and graphics. We demonstrate the capabilities of our proposed methodologies with two applications: 1) Shape-from-template (SfT): where the goal is to deform the input 3D template of an object as seen in image/video. Using PolyFit, we adopt test-time optimization that delivers competitive accuracy while being markedly faster than offline physics-based solvers, and outperforms recent physics-guided neural simulators in accuracy at modest additional runtime.

2) Garment draping. We train a self-supervised, mesh- and garment-agnostic model that generalizes across resolutions and garment types, delivering up to an order-of-magnitude faster inference than strong baselines.

5.1 Introduction

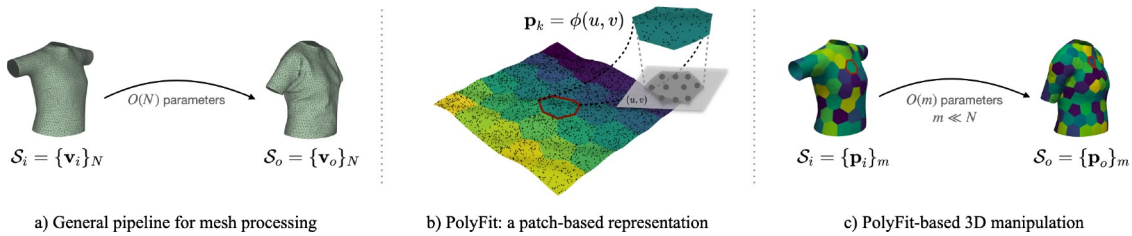


FIGURE 5.1 : **Patch-based representation and learning.** Against conventional per-vertex parameterizations of mesh deformation (a), we present PolyFit (b) which obtains a simplified patch-wise representation by fitting jet functions with limited parameters; consequently simplifying the transformations to modification of patch parameters only (c) and reducing the computational overhead by a large margin.

3D surface deformation is central to many computer vision and graphics applications such as animation Grigorev et al. [2023], video editing Parashar and Bartoli [2019] and medical imaging Lamarca et al. [2020], to name but a few. A common practice is to discretize deformable surfaces as meshes and formulate deformations with per-vertex unknowns, either optimized or predicted. Even when driven by reduced controls or regularizers, the underlying degrees of freedom scale with mesh resolution, which makes optimization and inference costly and hinders cross-resolution generalization. Another possibility is to use parametric representations such as splines Bookstein [1989] or NURBS Piegl [1991] to reduce the number of parameters to be estimated; however, such techniques are practical only for simpler geometries. As the geometries grow more complex, the number of parameters required for accurate representation usually explodes; thus defeating the purpose of using a parametric representation as a low-dimensional deformation state. Although modern learning-based representations such as AtlasNet Groueix et al. [2018] and other neural representations Yang et al. [2023], Park et al. [2019], Wang et al. [2021], Long et al. [2023] alleviate some limitations, their large parameter counts and heavy training typically hinder their usage in surface deformation pipelines that require a compact, controllable state.

In this chapter, we present a patch-wise, jet-based representation of surfaces which allows an efficient surface deformation with significantly reduced number of parameters to be estimated. Our proposed representation *PolyFit* divides the surfaces into small patches that are represented by simple jet functions, as seen

in Figure 6.1. It is learned in a supervised fashion using analytic functions which are cheap to generate. If needed, its accuracy can be further improved by fine-tuning with a small number of samples of a given object type. To deform surfaces, we directly update the patch-wise n-jet coefficients, thereby limiting the number of parameters to be estimated which leads to efficient processing of the deformations. We showcase the efficiency of our proposed methodology with two well-known problems in computer vision and graphics. First, we propose *PolySfT*: a learning-free, polynomial fitting approach to solve SfT Salzmann et al. [2008], Bartoli et al. [2015], Kairanda et al. [2022], Stotko et al. [2024], Fuentes-Jimenez et al. [2022] where the goal is to deform a given 3D template as seen in the images. We show that it performs much faster with competitive accuracy than the existing best-performing approaches. Second, we propose *OneFit*: a self-supervised polynomial fitting methodology to learn the draping of the garment. Existing methodologies Chen et al. [2024], Santesteban et al. [2022b], Bertiche et al. [2022] produce mesh-specific or garment-specific solutions. A few exceptions are Grigorev et al. [2023], Tiwari and Bhowmick [2023], Tiwari et al. [2023], which can handle multiple garments at various mesh resolutions. However, Tiwari and Bhowmick [2023], Tiwari et al. [2023] are not designed to generate temporally consistent garment deformations as their training process does not incorporate temporal data. In contrast, OneFit is temporally coherent, mesh- and garment-agnostic. Trained on a single garment, OneFit is able to handle different inter-class and intra-class garment variations. Moreover, due to its compact representation, it trains faster than existing methods and provides up to an order-of-magnitude faster inference than strong baselines.

5.2 PolyFit

PolyFit allows a jet-based representation on surfaces described with points/meshes. It is possible to represent the entire surface $\mathcal{S}$ with a single function or it can be subdivided using Approximated Centroidal Voronoi Diagrams (ACVD) clustering Valette and Chassery [2004], which efficiently constructs uniform tessellations of a given surface area, into desired number of patches. Each patch k is passed into PolyFit which computes orientation $\mathcal{O}_k = (s_k, \mathbf{R}_k, \mathbf{T}_k)$ and a parametric n-jet function $\phi_{\mathcal{S}_k}(u, v)$ with respect to a canonical uv space, as seen in Figure 5.2(a). Given a set of 3D points $\mathbf{p}_k$ on $\mathcal{S}_k$, PolyFit yields a smooth representation $\mathcal{S}_k := \{\mathcal{O}_k, \phi_{\mathcal{S}_k}(u, v)\}$ such that $\mathbf{p}_k = s_k \mathbf{R}_k \phi_{\mathcal{S}_k}(u, v) + \mathbf{T}_k$. Depending on its orientation, projecting $\mathcal{S}_k$ onto

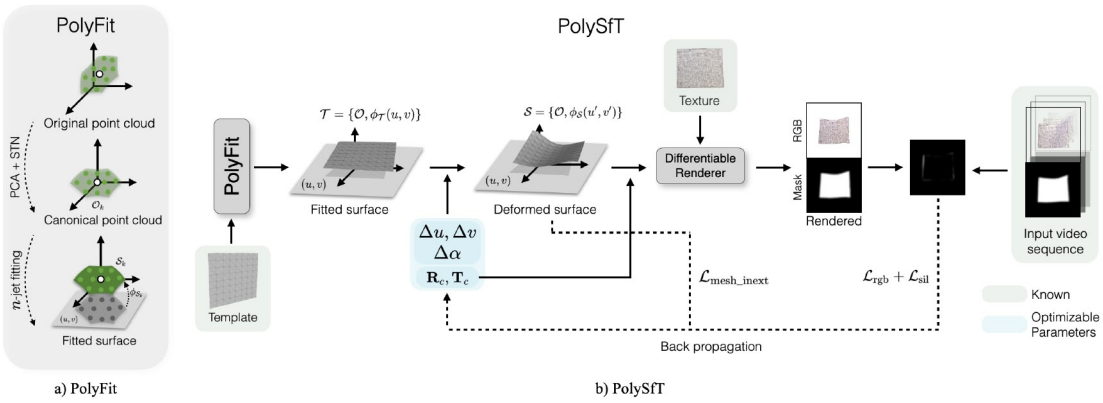


FIGURE 5.2 : a) **PolyFit**. It orients the input patch to improve bijectivity with the uv-plane and obtains an analytic representation using n-jet fitting. b) **PolySfT**. Using PolyFit, the input 3D template is deformed to match input images by estimating the offsets of fitted jet coefficients $\Delta\alpha$, uv displacements Δuv, as well as a rigid transformation $(\mathbf{R}_c, \mathbf{T}_c)$.

a local 2D frame may produce foldovers (overlaps in (u, v)), so the surface is no longer a single-valued height graph, breaking the bijectivity of $\phi_{\mathcal{S}_k}$. To mitigate this issue, we leverage Principal Component Analysis (PCA) to transform each patch into a canonical space of maximally planar patch representations, which empirically reduces such degeneracies. We then use a rigid Spatial Transformer Network (STN) Jaderberg et al. [2015] parameterized by unit quaternions to refine the orientation and promote a near-bijective height-graph parameterization before n-jet fitting (see Figure 5.4 in the supplement for an illustrative example).

Following the explicit representation of surfaces in terms of height function, $z(u, v)$, from a canonical uv space, an n^{th} order truncated Taylor expansion of z (also known as n-jet), is given by $z(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{i} \alpha_{i-j,j} u^{i-j} v^{j}$. The combinations of (α, n) allow an analytic representation of various non-trivial geometries, whose n^{th} order derivatives can be computed precisely. Moreover, given sufficient point samples, $z(u, v)$ can be obtained by fitting an n^{th} order jet in a least squares sense Cazals and Pouget [2005]. Therefore, canonical representation of surfaces, in which every point is parameterized by a diffeomorphism $\phi_{\mathcal{S}_k} : (u, v) \mapsto (u, v, z(u, v))^{\top}$, can be oriented using $\mathcal{O}_k = \{s_k, \mathbf{R}_k, \mathbf{T}_k\}$ to fit any smooth surface patch embedded in $\mathbb{R}^3$.

5.3 PolySfT

PolySfT (see Figure 5.2(b)) leverages PolyFit to efficiently deform a textured 3D template to match the input images, thereby recovering the 3D shape observed in video. We consider a single-patch representation of the template $\mathcal{T} = \{\mathcal{O}, \phi_{\mathcal{T}}(u, v)\}$, where $\mathcal{O} = (s, \mathbf{R}, \mathbf{T})$ and $\phi_{\mathcal{T}} = (u, v, \sum_{i=0}^{n} \sum_{j=0}^{i} \alpha_{i-j,j} u^{i-j} v^{j})^{\top}$. It is deformed using the offset jet coefficients $\Delta\alpha$, uv displacements $(\Delta u, \Delta v)$ as well as a global rotation, $\mathbf{R}_c$ and translation $\mathbf{T}_c$ related to rigid motion of the object. Assuming camera intrinsics are known (a common assumption in SfT), the resulting meshes are converted into RGB and mask images using a differentiable renderer Laine et al. [2020]. These renderings are compared to the target input video sequence by computing pixel-wise RGB and silhouette losses similar to Kairanda et al. [2022]. The gradients of the losses are used to refine the optimizable parameters. The reconstructed surface is thus modeled by $\mathbf{R}_c \phi_{\mathcal{S}}(u + \Delta u, v + \Delta v; \alpha + \Delta\alpha) \ + \mathbf{T}_c$ followed by transformation given by $\mathcal{O}$, to bring the reconstruction from canonical orientation to real one.

Losses. To guide the surface reconstruction, we employ a combination of photometric and geometric losses. Specifically, we adopt an RGB and silhouette loss as defined in Kairanda et al. [2022] to align the reconstructed mesh with the observed imagery. In addition, we apply mesh inextensibility loss to enforce edge-preserving constraints between the deformed and template mesh, $\mathcal{M}_{\mathcal{P}}$ and $\mathcal{M}_{\mathcal{T}}$ respectively:

$$\mathcal{L}_{\text{mesh\_inext}} = k_{\text{mi}} \sum_{i=1}^{n_{\text{edge}}} (e_i(\mathcal{M}_{\mathcal{P}}) - e_i(\mathcal{M}_{\mathcal{T}}))^2 \tag{47}$$

where $e_i(\cdot)$ denotes i-th edge length. Furthermore, we introduce a temporal consistency loss defined as follows, which promotes temporal smoothness across frames:

$$\mathcal{L}_{\text{tc}} = k_{\text{tc}} \frac{1}{W-1} \sum_{t=s}^{s+W-2} \left(\|\delta\alpha_t\|^2 + \|\delta uv_t\|^2 \right). \tag{48}$$

where $\delta\alpha_t = \Delta\alpha_{t+1} - \Delta\alpha_t, \quad \delta uv_t = \Delta uv_{t+1} - \Delta uv_t$.

5.4 OneFit

OneFit (see Figure 5.3) leverages PolyFit to efficiently simulate garment deformations using self-supervised learning. The template garment $\mathcal{T}$ is divided into patches

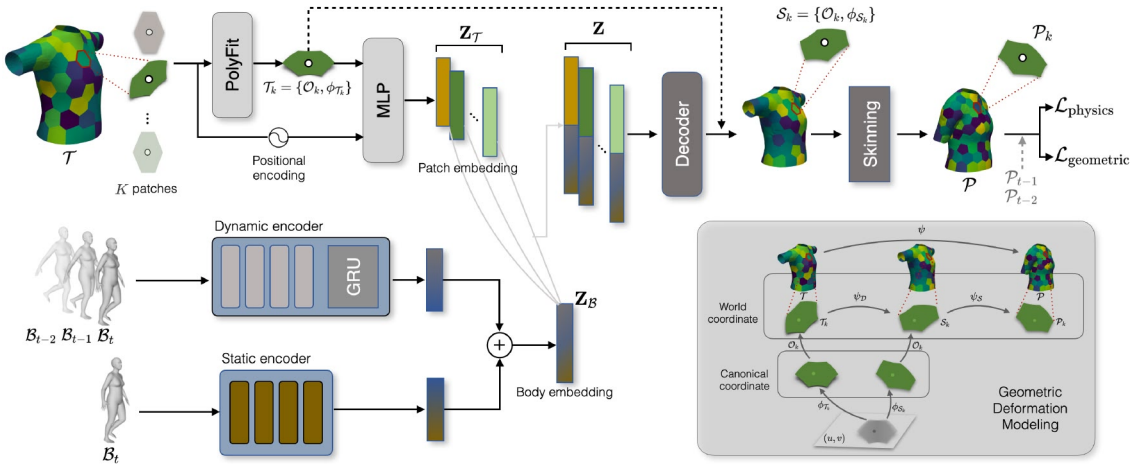


FIGURE 5.3 : **OneFit.** It deforms $\mathcal{T}$ isometrically to obtain $\mathcal{P}$ posed on body $\mathcal{B}_t$ by forcing patch boundary consistency, avoiding collisions and maintaining physical equilibria.

using Valette and Chassery [2004] which are passed into PolyFit to obtain a smooth patch representation, $\mathcal{T}_k := \{\mathcal{O}_k, \phi_{\mathcal{T}_k}(u, v)\}$.

A *garment patch embedding*, $\mathbf{Z}_{\mathcal{T}_k}$, is generated by passing $\mathcal{T}_k$ along with its positional encoding into the encoder, an MLP with skip connections. The positional encoding, as described in Mildenhall et al. [2020], is applied to each patch to incorporate its center position and its relative offsets from body joints.

A *body embedding*, $\mathbf{Z}_{\mathcal{B}}$ is obtained as a concatenation of dynamic and static encoding. To describe joint orientation relative to the parent joint, we follow Bertiche et al. [2022] and adopt 6D descriptors Zhou et al. [2019] concatenated with a unit vector with the unposed direction of gravity. This allows to alleviate the discontinuities in the rotation space presented in axis-angle representation. For the structure of the static and dynamic encoder, we adhere to the framework in Bertiche et al. [2022]. The global body pose, $\mathcal{B}(\beta, \theta, \vec{v})$ encapsulates the body shape (β), the current body pose (θ), and the global velocity of the root joint ($\vec{v}$).

Given $\mathcal{B}(\beta, \theta, \vec{v})$ and $\mathcal{T}$, the network first computes the garment patch and body embeddings, $\mathbf{Z}_{\mathcal{T}_k}$ and $\mathbf{Z}_{\mathcal{B}}$ respectively. They are then concatenated and fed into a decoder as $\mathbf{Z} = \text{concatenate}(\mathbf{Z}_{\mathcal{T}_k}, \mathbf{Z}_{\mathcal{B}})$ to predict the patch deformations, $\mathcal{S}_k := \{\mathcal{O}_k, \phi_{\mathcal{S}_k}(u, v)\}$. The garment deformations are learned by enforcing the physical equilibrium of forces and geometric consistency of template and deformed surface patches posed on the desired body after skinning. This enables a self-supervised, mesh-agnostic, garment-agnostic learning of the deformations.

Geometric Deformation Modeling. As seen in Figure 5.3(b)), $\mathcal{T}_k$ is deformed to $\mathcal{S}_k$. Upon skinning with $\psi_{\mathcal{S}_k}$, we obtain $\mathcal{P}_k$ posed on body $\mathcal{B}_t$. We impose patch deformations to be isometric and enforce the preservation of their first fundamental form in terms of local metric tensors at $\mathcal{T}_k$, $\mathbf{g}_{\mathcal{T}_k} = \mathbf{J}_{\phi_{\mathcal{T}_k}}^{\top} \mathbf{J}_{\phi_{\mathcal{T}_k}}$ and $\mathcal{P}_k$, $\mathbf{g}_{\mathcal{P}_k} = \mathbf{J}_{\phi_{\mathcal{S}_k}}^{\top} \mathbf{J}_{\psi_{\mathcal{S}_k}}^{\top} \mathbf{J}_{\psi_{\mathcal{S}_k}} \mathbf{J}_{\phi_{\mathcal{S}_k}}$. $\mathbf{J}_{\phi_{\mathcal{T}_k}}$ and $\mathbf{J}_{\phi_{\mathcal{S}_k}}$ can be expressed analytically from the parametric representation obtained in PolyFit. $\mathbf{J}_{\psi_{\mathcal{S}_k}}$ can be analytically calculated from the LBS skinning function Lin et al. [2022].

We impose geometric restrictions on the patch boundaries to maintain consistency. Like Chen et al. [2024], we allow local stretchings to avoid collisions. We impose following losses:

1) Collision. It penalizes penetration between the body and the garment. For each point, it is given by

$$\mathcal{L}_{\text{collision}} = k_{\text{c}} \sum_{\text{points}} d_c^2, \tag{49}$$

where $d_c = \max(\epsilon - d(x), 0)$ quantifies the degree of interpenetration. $d(x)$ is the signed distance between the garment vertex and the body surface, and ϵ is a small positive constant introduced to enhance stability.

2) Isometry. To preserve geodesics between the template and draped garment, it enforces metric tensor similarity. It is computed as

$$\mathcal{L}_{\text{iso}} = k_{\text{i}} \frac{1}{KM} \sum_{\mathcal{T}_k \in \mathcal{T}} \sum_{\mathbf{x} \in \mathcal{T}_k} |k_{\text{ext}} \mathbf{g}_{\mathcal{T}_k}(\mathbf{x}) - \mathbf{g}_{\mathcal{P}_k}(\mathbf{x})| \tag{50}$$

M is the number of points in each patch and K is total number of patches. The inextensibility factor k_{ext} is defined similarly to (45). We first allow network to stabilize and then enforce inextensibility.

3) Boundary. It enforces the connectivity between adjacent patches and is defined as follows:

$$\mathcal{L}_{\text{boundary}} = \frac{1}{M_b} \sum_{(i,j) \in \mathcal{B}} \sum_{\text{points}} k_{\text{b}} \|\mathbf{x}_i - \mathbf{x}_j\|^2 + k_{\text{bn}} \left(1 - \cos(\theta_n)\right)^2 \tag{51}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ denote boundary points on the adjacent patch of index i and j, M_b denotes the total number of adjacent points between all pairs of patches. $\cos(\theta_n)$ represents the cosine similarity between the normals of the n-th pair of adjacent points. It penalizes deviations from perfect parallelism, thus promoting smoother

transitions at the boundaries. Overall, the geometric losses are given by

$$\mathcal{L}_{\text{geometric}} = \mathcal{L}_{\text{iso}} + \mathcal{L}_{\text{collision}} + \mathcal{L}_{\text{boundary}} + \mathcal{L}_{\text{mesh\_inext}} \tag{52}$$

$\mathcal{L}_{\text{mesh\_inext}}$, defined in Eq. (47), and $\mathcal{L}_{\text{iso}}$ impose the geodesic preservation constraints at zeroth and first order respectively. It allows the garment deformations to be isometric while taking local body-garment collisions into account.

Physics-based deformation modeling. They incorporate effect of inertia and gravitational forces. Their implementation is similar to Chen et al. [2024] except they are defined on points instead of mesh vertices.

1) Gravity. It incorporates gravity by minimizing the potential energy of the garment, given by

$$\mathcal{L}_{\text{gravity}} = \sum_{\text{vertices}} -mg^{\top}\mathbf{x}, \tag{53}$$

where m is the particle mass and g is the gravitational acceleration.

2) Inertia. It incorporates the inertia loss as proposed in Santesteban et al. [2022b]. It is given by

$$\mathcal{L}_{\text{inertia}} = \sum_{\text{vertices}} \frac{1}{2\Delta t^2} m(\mathbf{x}^{[t]} - \mathbf{x}^{[t-1]} - \Delta t v^{[t-1]})^2, \tag{54}$$

where Δt is the simulation time step, $\mathbf{x}^{[t]}$ and $\mathbf{x}^{[t-1]}$ specify the particle's position at times t and $t-1$, respectively.

Overall, physics-based losses are

$$\mathcal{L}_{\text{physics}} = \mathcal{L}_{\text{inertia}} + \mathcal{L}_{\text{gravity}} \tag{55}$$

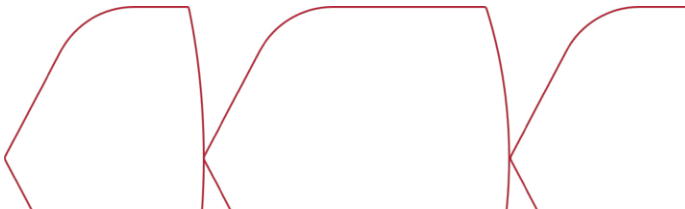
Together, the losses are given by

$$\mathcal{L} = \mathcal{L}_{\text{physics}} + \mathcal{L}_{\text{geometric}} \tag{56}$$

5.5 Experiments

5.5.1 PolyFit

We trained PolyFit on point clouds sampled from regular explicit functions (4-jets, trigonometric, Gaussian and Bessels) and on patches sampled from garment meshes



in CLOTH3D Bertiche et al. [2020] dataset. The training time is about 2 hours.

Training dataset. To support the training of the rotation correction block in PolyFit, we created a dataset consisting of point cloud patches, generated by combining four families of functions, including jet, trigonometric, Gaussian and Bessel. The four families of functions are:

1) 4-jet: $f(u,v) = \sum_{i=0}^{4} \sum_{j=0}^{i} \alpha_{i-j,j} u^{i-j} v^{j}$

2) Trigonometric: $T(u,v) = \alpha \cos\left(\theta \sqrt{u^2+v^2}\right)$

3) Gaussian: $G(u,v) = \alpha \exp\left(-\frac{(u-u_0)^2+(v-v_0)^2}{2\sigma^2}\right)$

4) Bessel: $B(u,v) = \alpha J_0\left(k\sqrt{(u-u_0)^2+(v-v_0)^2}\right)$

where $\alpha \in [-0.5, 0.5]$, $\theta \in [\pi, 2\pi]$, $\sigma \in [0.5, 1]$ and $k = 5$. Here, J_0 denotes the Bessel function of the first kind of order 0. Using $(u,v) \in [-1,1]$, we sum the outputs from the four functions and train the PolyFit in a supervised way, by minimizing the height discrepancies between the original and the fitted surface points. We further add patches extracted from CLOTH3D Bertiche et al. [2020] training dataset. The garment meshes are first subdivided four times to achieve a dense mesh. ACVD Valette and Chassery [2004] is applied to the refined mesh, clustering the vertices into k patches according to the surface area. Specifically, the number of patches is given by $\max\left(100, \min\left(400, \lfloor \frac{A}{0.008} \rfloor\right)\right)$, where A denotes the area of the mesh. We extracted 100k patches and computed ground truth normals from their corresponding meshes.

Training details. The batch size is set to 512 and the learning rate is set to 0.001. For every patch, we perform a preprocessing step including normalization, basis extraction and coordinate frame transformation, as depicted in Ben-Shabat and Gould [2020]. Figure 5.4 illustrates the benefit of using STN correction module, which refines the orientation of the given input point cloud and promotes a near-bijective height-graph parameterization before n-jet fitting.

Note that DeepFit Ben-Shabat and Gould [2020] and NJF Aigerman et al. [2022] are not directly comparable for our setting. DeepFit performs point-wise jet fitting to estimate local differentials and does not provide a compact patch-level state that can be driven as control variables for surface deformation. NJF presumes a training set of source–target maps and a global latent code to learn piecewise-linear mesh mappings, supervision that is unavailable in our scenario.

We therefore compare PolyFit against AtlasNet Groueix et al. [2018], a learned multi-chart surface generator, for patch fitting. AtlasNet encodes an input point cloud

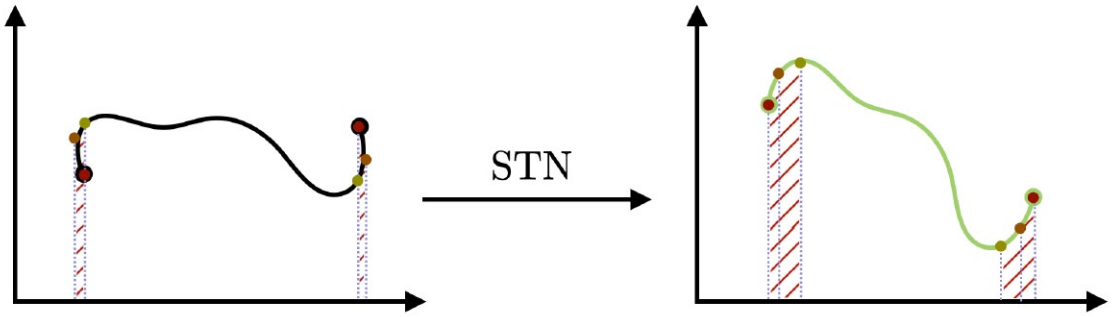


FIGURE 5.4 : Effect of STN canonicalization. It promotes a near-bijective parameterization (1-D section shown).

with PointNet Qi et al. [2017a] and decodes a latent code through K chart decoders (MLPs), each mapping a 2D parametric domain to a 3D patch; the union of all K patches forms the reconstruction. We train the autoencoder variant of AtlasNet with $K \in \{5, 25, 125\}$ on 5,000 CLOTH3D garments covering diverse types, and evaluate on six garment templates from Santesteban et al. [2022b]. For evaluation, we follow the AtlasNet protocol and compute the symmetric Chamfer distance between the reconstruction (concatenating points from all K charts) and 10,000 points uniformly sampled on the ground-truth template. For both methods, the point clouds are normalized before computing the metric. We observe that varying K yields only minor differences in Chamfer distance, so we report the average across K in Table 5.1 and provide the full table in Table 5.2 of the supplement. As summarized in Table 5.1, PolyFit consistently attains lower Chamfer distance, demonstrating accurate fitting with analytic, patch-wise representation.

	Tshirt	Dress	Tank	Top	Shorts	Pants
AtlasNet	0.517	1.070	0.962	0.464	1.509	0.938
PolyFit (Ours)	**0.229**	**0.168**	**0.268**	**0.092**	**0.372**	**0.237**

TABLE 5.1 : Chamfer Distance (multiplied by 10^3) for patch fitting on six garment templates.

Comparison with AtlasNet. We report per-template Chamfer distances for AtlasNet and PolyFit across $K \in \{5, 25, 125\}$ learned charts. For training, we use square (patch) as template type, the number of sampled points is set to 10,000. The point clouds are normalized before computing the metric. As seen in Table 5.2,

varying K leads to only minor CD changes (the total point budget is fixed), while PolyFit attains consistently lower errors on all templates.

	Tshirt	Dress	Tank	Top	Shorts	Pants
AtlasNet ($K = 5$)	0.531	1.039	0.939	0.481	1.508	0.963
AtlasNet ($K = 25$)	0.490	1.060	0.942	0.420	1.471	0.992
AtlasNet ($K = 125$)	0.531	1.111	1.005	0.490	1.547	0.858
PolyFit (Ours)	**0.229**	**0.168**	**0.268**	**0.092**	**0.372**	**0.237**

TABLE 5.2 : Chamfer Distance (multiplied by 10^3) for patch fitting on six garment templates.

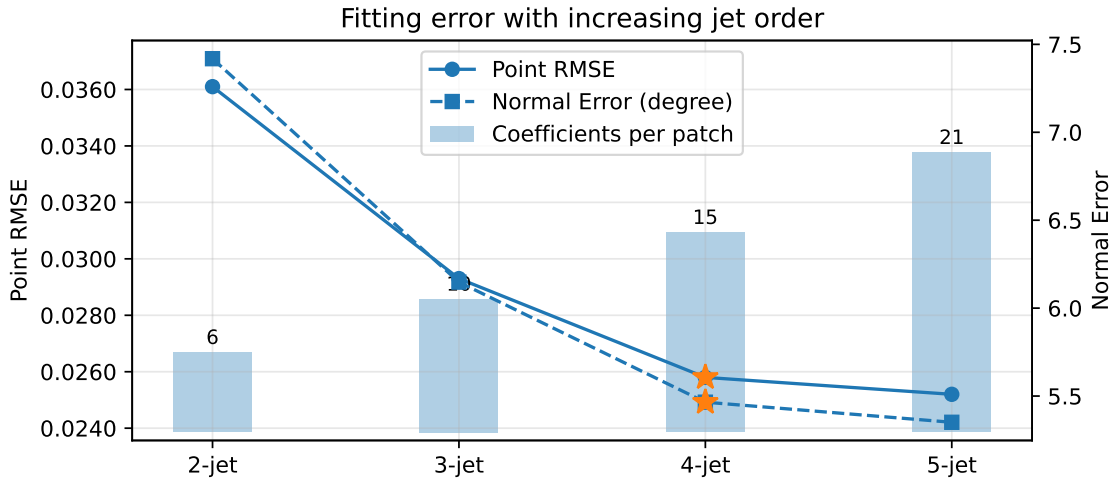


FIGURE 5.5 : Fitting error with respect to jet order n on CLOTH3D patches.

Ablation on Jet Order n. To evaluate the fitting performance of PolyFit, we use garment patches from the CLOTH3D validation dataset Bertiche et al. [2020]. We compute its performance from metrics including height RMSE and normal loss, measured in degrees. Figure 5.5 shows the performance of n-jet fitting on the CLOTH3D dataset. This shows that the 4-jet function is capable of fitting point clouds from garment patches effectively. Therefore, we fix $n = 4$, as this setting has been shown to achieve high accuracy on garments with reasonable computational complexity.

Table 5.3 indicates that the STN module noticeably enhances the model's fitting accuracy as it re-orients patches to improve their bijectivity, which leads to better jet-fitting.

Additional ablation studies. We evaluate PolyFit on patches sampled from the CLOTH3D validation split and compare it with PointNet Qi et al. [2017a] and

	Height RMSE	Normal Diff (°)
with	0.0201	5.274
w/o	0.0259	5.465

TABLE 5.3 : PolyFit fitting metric, with and w/o STN.

DGCNN Wang et al. [2019], two point-based networks that we adapt to regress jet coefficients directly from point clouds. As summarized in Table 5.4, PolyFit delivers lower geometric error (height RMSE and normal difference) and shorter per-patch inference time than these alternatives.

Model	Height RMSE	Normal Diff (°)	Time (ms)
PointNet Qi et al. [2017a]	0.0309	6.936	0.0754
DGCNN Wang et al. [2019]	0.0290	6.406	0.0625
PolyFit	**0.0201**	**5.274**	**0.0481**

TABLE 5.4 : Comparison with point-based backbones on CLOTH3D validation patches. We report height RMSE, normal-angle error (°), and per-patch inference time (ms).

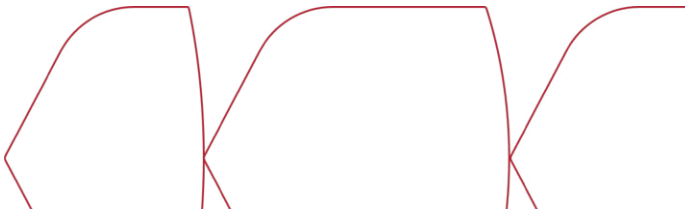
We further ablate the family of parametric functions used for training. Table 5.5 shows that increasing the diversity of parametric functions and augmenting the training set with patches extracted from garment meshes both yield additional accuracy gains.

Function used for training	Height RMSE	Normal Diff (°)
Gaussians only	0.0248	5.485
4 Families	0.0239	5.423
4 Families + garment patches	0.0201	5.317

TABLE 5.5 : Study on different training data for PolyFit.

5.5.2 PolySfT

Implementation details We adopt an **adaptive sliding-window optimization strategy** with a window size of W. Within each window, optimization continues until either the loss fails to improve for a preset number of consecutive iterations (referred to as the **patience threshold**) relative to the current minimum, or the number of iterations exceeds a certain period (referred to as the **frame period**). Once either condition is met, we shift the window forward by one frame and initialize the new



frame's parameters using those from the previous frame. This method promotes temporal consistency and maximizes optimization efficiency. We employ adaptive window optimization with W=3, patience threshold of 100 iterations, and frame period of 500. Loss coefficients are set to k_{mi}=0.1 for $\mathcal{L}_{\text{mesh\_inext}}$ and k_{tc}=0.05 for $\mathcal{L}_{\text{tc}}$. We optimize using Adam with learning rates of 10^{-3} for displacement $(\Delta u, \Delta v)$, 10^{-2} for jet coefficients $\Delta\alpha$, and 10^{-4} for rigid transformation $(\mathbf{R}_c, \mathbf{T}_c)$.

We evaluate PolySfT on two real datasets: Kinect-Paper (193 images with ground truth) Varol et al. [2012] and Paper-Bend (71 images without ground truth) Salzmann et al. [2007]. Table 5.6 compares quantitative results on Kinect-Paper against traditional SfT (SfT) Bartoli et al. [2015] and supervised SfT methods Fuentes-Jimenez et al. [2021, 2022]; PolySfT attains consistently lower errors than these baselines. We did not compute results for ϕ-SfT Kairanda et al. [2022] due to prohibitive runtime and memory requirements on long sequences. Stotko et al. [2024] is designed to work with square meshes only and is therefore not applicable to the Kinect-Paper template. Selected reconstructed frames are shown in Figure 5.8. The code for TD-SfT Fuentes-Jimenez et al. [2021] is not publicly available, so we cannot show visual results. Additional qualitative results on Paper-Bend are shown in Figure 5.6 of the supplement.

Figure 5.6 presents the qualitative results on the Paper-Bend and Kinect-Paper datasets. Renderings of the reconstructed meshes (second column) closely match the input images (first column), resulting in low per-pixel RGB error maps (third column). Figure 5.7 shows a comparison with SOTA methods on the synthetic dataset provided by Kairanda et al. [2022].

	SfT [8]	**DeepSfT** [53]	**TD-SfT** [52]	**PolySfT**
RMSE (mm)	6.17	6.97	3.37	**2.59**

Table 5.6 : Reconstruction results on **Kinect-Paper dataset**. Methods compared: SfT Bartoli et al. [2015], DeepSfT Fuentes-Jimenez et al. [2022], TD-SfT Fuentes-Jimenez et al. [2021], and our PolySfT. Depth RMSE is reported in mm.

In addition, we evaluated PolySfT on synthetic dataset provided by Kairanda et al. [2022], which comprises four sequences (S1-S4) of cloth deformations, each containing between 45 and 50 frames. We compute the 3D error e_{3D} and the average per-vertex angular error e_n in degrees, following the definition given in the supplement of Kairanda et al. [2022]. Table 5.7 shows that our method outperforms PGSfT and

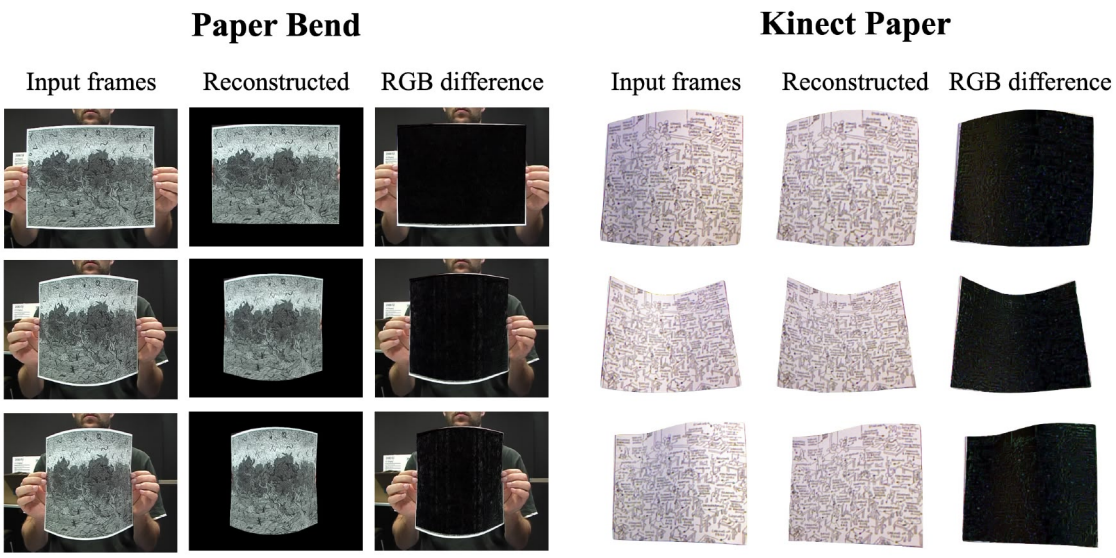


FIGURE 5.6 : Additional reconstruction results on Paper-Bend and Kinect-Paper.

	S1		S2		S3		S4	
	e_{3D}	e_n	e_{3D}	e_n	e_{3D}	e_n	e_{3D}	e_n
PGSfT Stotko et al. [2024]	0.0298	7.780	0.0448	8.770	0.0823	21.058	0.0919	6.885
ϕ-SfT Kairanda et al. [2022]	0.0420	11.860	**0.0230**	10.620	0.0330	**9.120**	0.0050	2.610
SfT Bartoli et al. [2015]	0.0328	7.275	0.0483	7.683	0.0481	14.607	0.0232	5.165
PolySfT	**0.0234**	**6.337**	0.0298	**4.815**	**0.0266**	10.222	**0.0026**	**0.475**

TABLE 5.7 : Results on the ϕ-SfT synthetic dataset, comparing e_{3D} and e_n errors across methods for sequences S1 to S4.†

SfT on all sequences. It outperforms ϕ-SfT on S1 and S4, and achieves comparative results on S2 and S3 sequences. Visual comparisons with state-of-the-art methods are shown in Figure 5.7.

We report wall-clock per-frame optimization time averaged over entire sequences. On an NVIDIA V100 GPU, PolySfT runs at ∼10s/frame. This is ∼270× faster than ϕ-SfT (∼2,800s/frame) and about 2× slower than PGSfT (∼5s/frame). Despite this gap to PGSfT, PolySfT achieves accurate reconstructions while remaining substantially faster and far more memory-efficient than ϕ-SfT.

Stability test. We assess PolySfT's stability by running the optimization process for many more iterations than usual. Figure 5.9 displays the reconstructed meshes for two scenes with different motion patterns at 50 iterations, 300 iterations (the average evaluation point), and extended runs at 1000 and 5000 iterations. The results

demonstrate that the mesh reliably tracks the intended motion, with only minimal changes beyond the typical iteration threshold. Moreover, initializing from previous frames provides a robust starting point for the current frame. Experiments are conducted on a single NVIDIA V100 GPU.

5.5.3 OneFit

Training details In the Dynamic encoder, different from Bertiche et al. [2022], the Gated Recurrent Unit (GRU) layers are initialized with random hidden states. The body feature extractor is implemented using a five-layer multilayer perceptron (MLP) with LeakyReLU activation between the layers. Each layer contains 256 nodes, with the exception of the final layer.

The decoder consists of four fully connected layers, each with dimensions of 512, 512, 512, and 256, respectively. This is followed by three prediction heads for jet coefficients, translation and scale, each implemented as three fully connected layers with dimensions 128 and 64, ending with a final output layer.

Finally, to maximize parallel computation on GPUs, the batch size for each garment is dynamically determined based on the number of patches using the following equation: $bs = \frac{20{,}000}{\text{number of patches}}$.

We trained OneFit on a set of 6 standard garment templates (tshirt, dress, pants, shorts, long-sleeve top and tank) used in Santesteban et al. [2022b]. We utilize the human motion sequences from the AMASS dataset (60 seq., 10K poses) Mahmood et al. [2019]. We then validate the resulting models on unseen garment meshes from CLOTH3D Bertiche et al. [2020], which are preprocessed as described later in this section.

We set the adaptive batch size according to the number of patches of the garment. The learning rate begins at 10^{-3} for the first 10 epochs and then reduces to 10^{-4} . We set $k_b = 5e3$, $k_{mi} = 2$, $k_g = 1$, $k_c = 1$, and $k_i = 0.5$. These parameters are fixed for all garments across all experiments.

Dataset preprocessing. CLOTH3D garments are posed with legs slightly apart and fitted on different body shapes, differing from the standard SMPL T-pose used in our framework. We therefore preprocess each garment as follows:

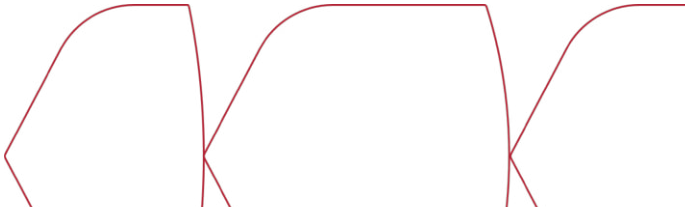
1. Query the closest body vertex for each garment vertex

†. Best and second-best results are in bold and with underline, respectively.

2. Apply displacement based on the shape difference between original and average SMPL bodies
3. Perform one iteration of Laplacian smoothing to remove local artifacts
4. For loose garments (dresses, skirts), apply only shape correction without pose alignment

We compare OneFit with state-of-the-art self-supervised methods: GAPS Chen et al. [2024], SNUG Santesteban et al. [2022b], NCS Bertiche et al. [2022] and HOOD Grigorev et al. [2023]. Except for HOOD, all these methods train mesh-specific models of a single garment. HOOD trains a mesh-based model, but can train a unified model for multiple garments. OneFit trains a mesh-independent model and can learn either a single or a multiple garment network. Furthermore, an existing model can be finetuned to a specific garment, avoiding from-scratch training. Being mesh-independent, OneFit can also generalize to various mesh resolutions with a similar inference time, as shown in Figure 5.11. SNUG requires post-processing to remove garment-body collision artifacts. GAPS learns a body-specific model and thus requires no post-processing. OneFit does not require collision post-processing when the garments and bodies fall within the training distribution. However, for unseen garments, e.g., draping a full-sleeve top from a model trained on a half-sleeve one, some collision artifacts may appear, which can be removed with collision post-processing.

OneFit as a single garment model. We test its generalization capabilities. Figure 5.10 shows the results of OneFit trained on a Tank top. While it drapes well on the trained garment, it generalizes well to the garments of similar style without a post-processing. We also test the generalization capabilities of OneFit towards garment inter-class variations. Figure 5.12 (top) shows results of OneFit trained on a dress and tested on various garments. Since it learns garment deformations from small patches, it basically learns localized garment-body interactions which are generally extensible to various garments. Hence, we see a decent drape on tshirt and tank tops. The only artifacts that appear over these garment are due to collisions. Since the network is learned on a dress which does not have arms, it is not trained to be aware of the garment-body interactions in this region which makes the collision artifacts inevitable. A simple post-processing can remove these artifacts. The interesting results in Figure 5.12 (top) are with pants and shorts which are tightly wrapped to the body as opposed to dress. Besides the collision artifacts, some deformation



artifacts are also visible within the area between the legs. Since dress is a loose garment, the network does not witness tightly-bound garment-body interactions between the legs and produces artifacts.

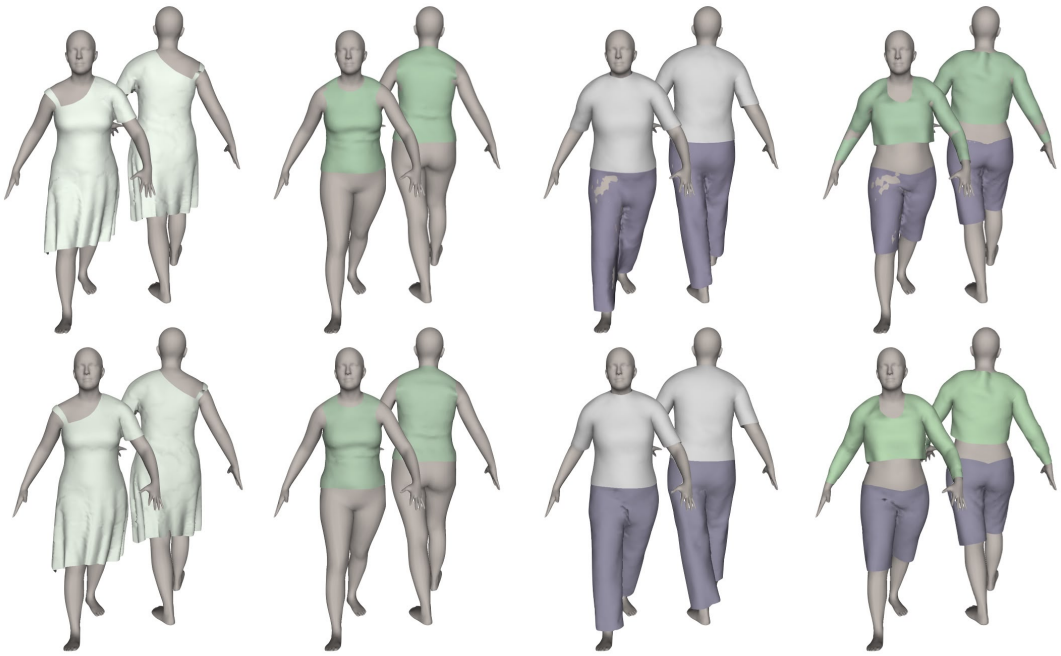


FIGURE 5.12 : Top: OneFit trained using Dress. Bottom: OneFit trained using a collection of 6 garments.

Loose garments are known to be challenging for most garment draping methods. Figure 5.13 shows that OneFit trained on dress is close to GAPS, the best performing method in this case. All other methods yield noticeable artifacts.

OneFit as a multiple garment model. We trained OneFit jointly on all six garments: tshirt, dress, pants, shorts, long-sleeve top and tank top in order to cover a wide range of body-garment interactions. Table 5.8 shows that the ε_c (% of vertices under collisions) has drastically reduced as compared to OneFit trained only on dress. We also see that training on multiple garments improves the generalizability of OneFit. Figure 5.12 (bottom) shows better garment drapings on pants and shorts; which demonstrated deformation artifacts in Figure 5.12 (top) under a single garment OneFit. Figure 5.14 shows that OneFit is on par with GAPS, the best performing

†. Poses may differ slightly across methods due to differences in the SMPL implementations used by each method.

Model	T-shirt	Dress	Tank	Top	Shorts	Pants
OneFit (Dress)	0.330	0.840	2.834	10.033	6.271	2.389
OneFit (6 garments)	0.422	0.756	0.481	1.592	1.749	1.194

TABLE 5.8 : ε_c for various garments. Training on multiple garments improves OneFit's generalizability without requiring any post-processing.

Model	ε_c	Training time
OneFit (6 garments)	2.397	8h
OneFit (6 garments) + finetuning	1.982	1h
OneFit (jumpsuit)	1.845	3h

TABLE 5.9 : Fine-tuning vs training OneFit on jumpsuit.

method in this case. Figure 5.15 demonstrates the garment-agnostic nature and capability of OneFit to handle extreme poses, including a high-kick dress and a cobra-pose tank-top.

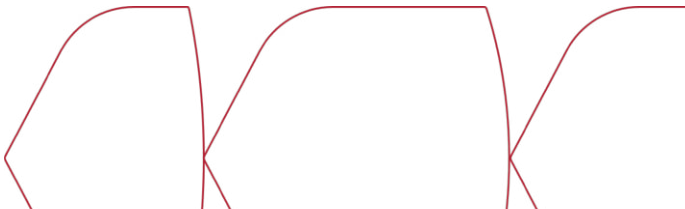
Finetuning OneFit. Once learned, OneFit can be finetuned to a new garment. Table 5.9 compares OneFit trained on multiple garments to drape a new garment, jumpsuit. Almost 2.5% vertices are observed to be under collision which are brought down to less than 2% by finetuning this model on jumpsuit for an hour. Training OneFit from scratch achieves a similar performance with 3× more computation.

Ablation study. Table 5.10 shows ablation study on various losses, using a tank top as the test garment. Besides ε_c, we also compute ε_a and ε_e reporting average per-point % area and edge length change. Losses $\mathcal{L}_{\text{mesh\_inext}}$ and $\mathcal{L}_{\text{iso}}$ control the stretchability of garment through zeroth-order (point-based) and first-order (normal-based) metrics; they are both required to minimize size variations while avoiding collisions. $\mathcal{L}_{\text{col}}$ reduces the amount of body-garment collisions.

Model	ε_e	ε_a	ε_c
OneFit	7.828	13.020	0.227
no $\mathcal{L}_{\text{mesh\_inext}}$	13.175	24.011	0.263
no $\mathcal{L}_{\text{iso}}$	7.739	12.760	1.641
no $\mathcal{L}_{\text{col}}$	8.004	13.373	0.387

TABLE 5.10 : Ablation of OneFit training losses.

Timing Comparison. Table 5.11 reports the timing performance. The mesh specific methods take less time to train but cannot generalize to different topologies. SNUG



	SNUG	HOOD	GAPS	OneFit
Train	2 h	10 h	2-6 h	2-8 h
Runtime	32.4 ms	125.5 ms	5.12 ms	0.48 ms

TABLE 5.11 : Timing performance.

takes 2 hours to converge for tight garments with less than 10k vertices. GAPS takes 2 hours in the same setting and up to 6 hours for looser garments like dresses. HOOD reports ∼10h. OneFit takes 8 hours for training a multiple garment model on 4 NVIDIA A100 GPUs. For runtime, we evaluate on a 2,175-frame CMU sequence on an Quadro RTX 6000. HOOD takes the longest runtime. SNUG takes less inference time but is slower than GAPS due to additional collision post-processing. OneFit achieves the fastest runtime; the optional post-processing step (required only while draping garments out of training set) adds only ∼3-4ms per frame.

Limitations. Each patch in PolyFit is encoded as a single-valued height function; extreme wrinkles, large bulges, or self-occlusions may violate this assumption. We plan to adopt adaptive, curvature/visibility-aware partitioning and enable higher-order jets on demand to preserve fine details. Additionally, seam artifacts at patch boundaries may occur under large deformations and are mitigated via a lightweight Laplacian smoothing applied along the boundaries.

5.6 Conclusions

We introduced PolyFit, a patch-based representation that deforms surfaces via a compact set of jet coefficients. We demonstrated its utility in two applications: PolySfT, a test-time optimization that estimates jet coefficients and local uv shifts so that differentiable renderings match the input images, and OneFit, a self-supervised, mesh- and garment-agnostic neural garment simulation model that generalizes across resolutions and garment types. These results highlight the promise of polynomial, patch-wise representations for efficient deformation modeling/learning.

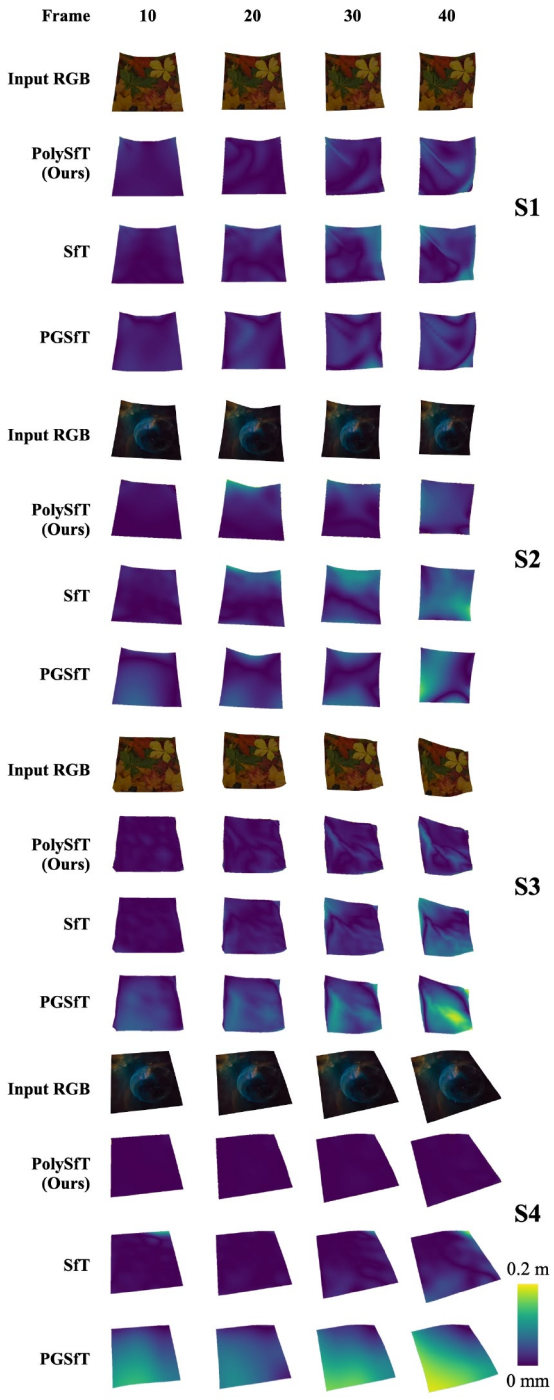


FIGURE 5.7 : Error map comparison with SOTA methods on ϕ-SfT Synthetic Dataset, showing frames 10, 20, 30, and 40 from left to right.

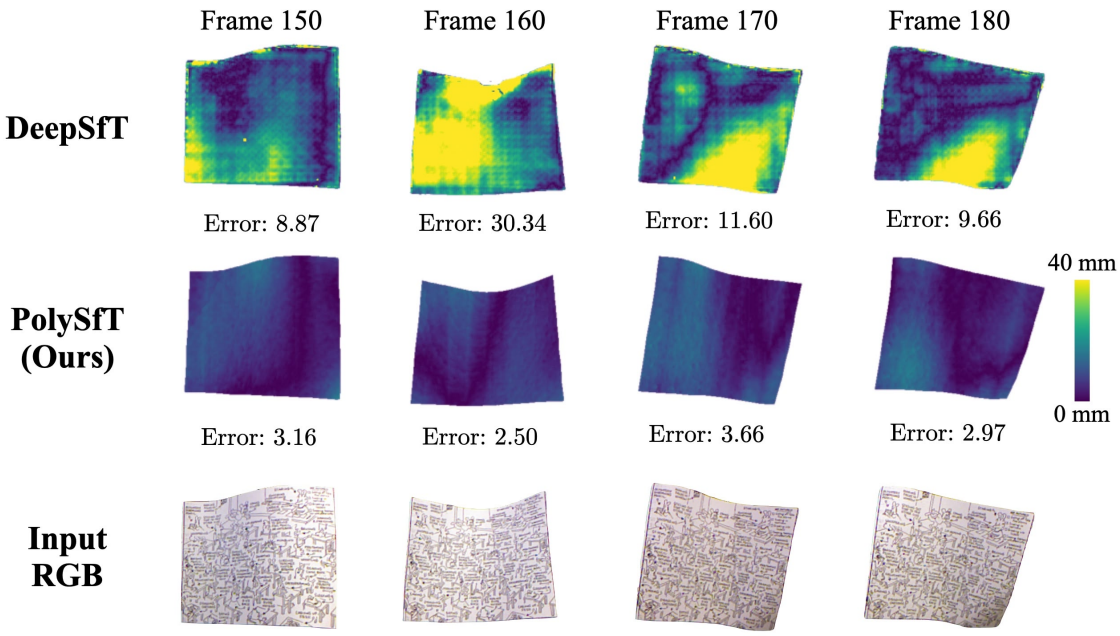


FIGURE 5.8 : Error map comparison between DeepSfT and Ours on example frames from the Kinect-Paper dataset.

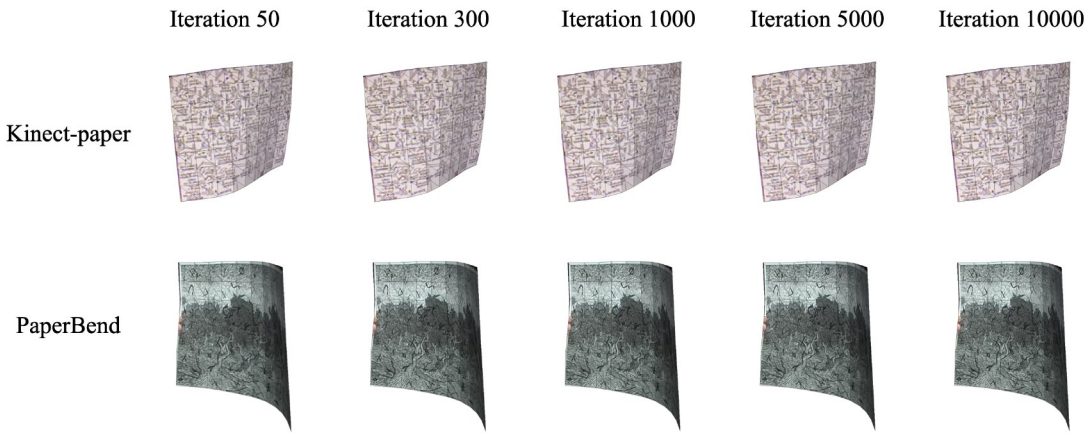


FIGURE 5.9 : Stability test for Kinect-Paper and Paper-Bend. We show the reconstructed mesh at various iterations.

FIGURE 5.10 : Single garment OneFit under garment intra-class variations. Trained on a Tank top (in green), OneFit is able to drape tank tops of different styles.

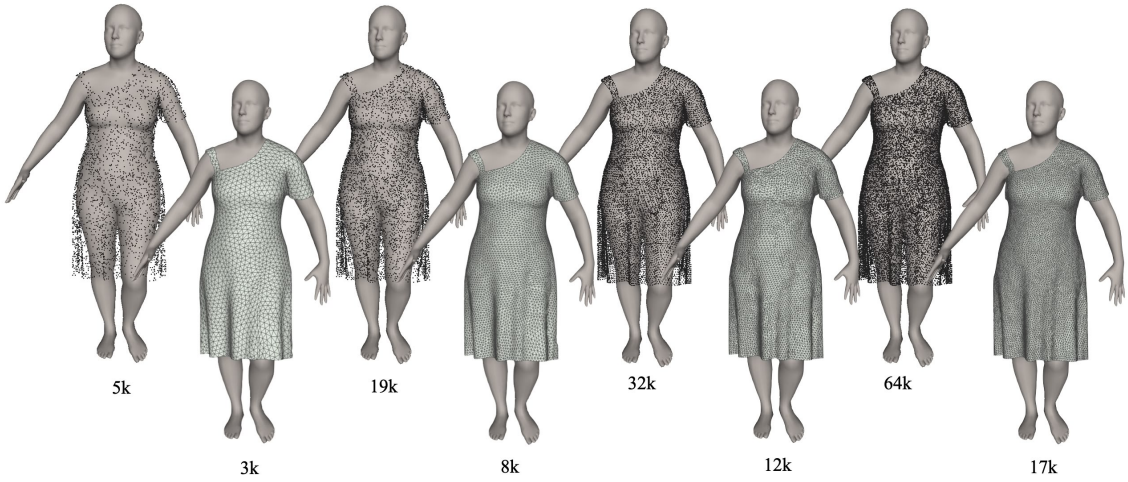


FIGURE 5.11 : **OneFit** drapings with different mesh resolutions obtained within a similar inference time.

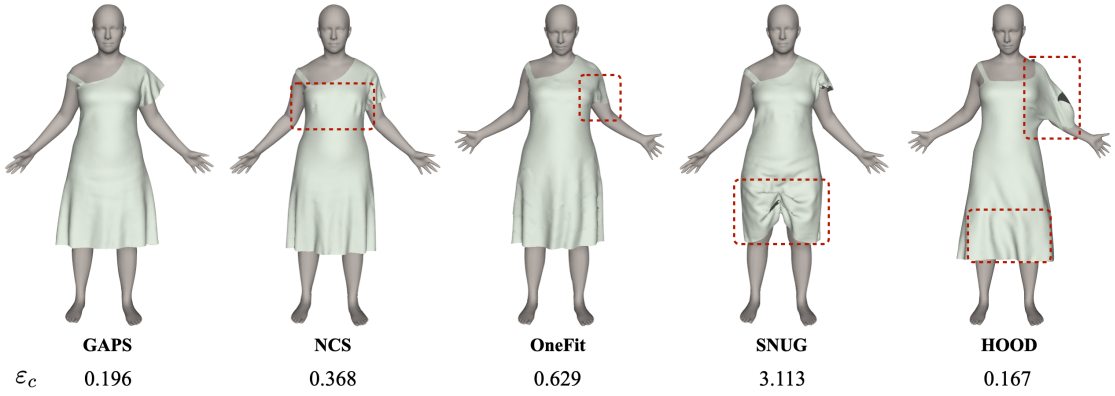


FIGURE 5.13 : SOTA comparison for OneFit trained on dress. The results on GAPS are reported after post-processing.†

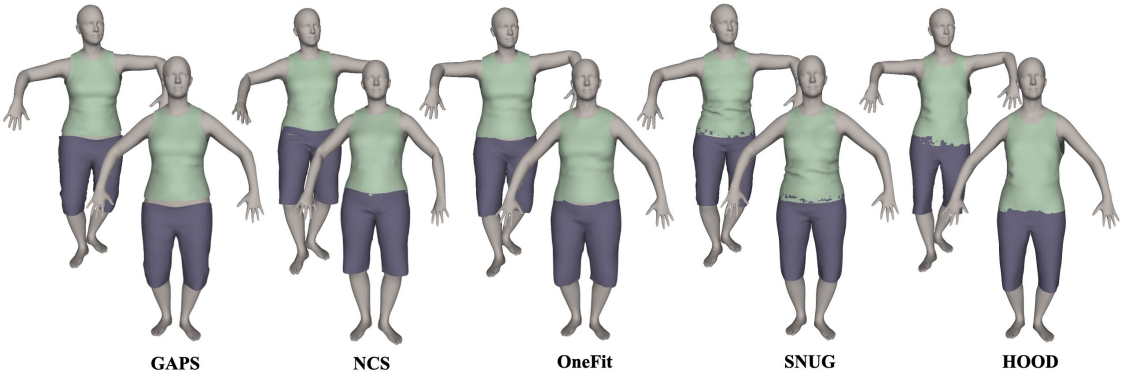


FIGURE 5.14 : SOTA comparison on tight garments. The results on GAPS are reported after post-processing.†

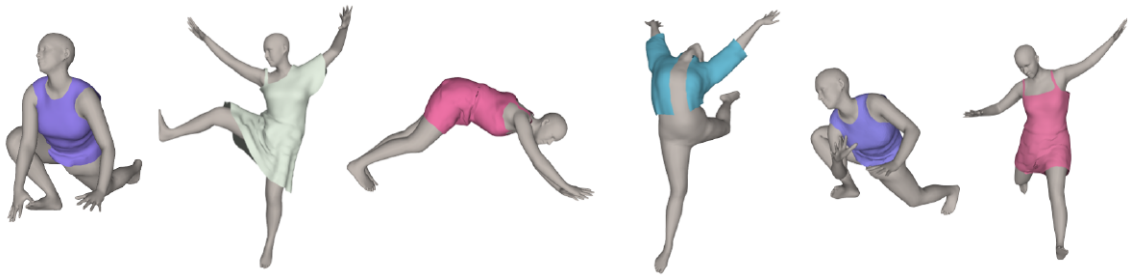
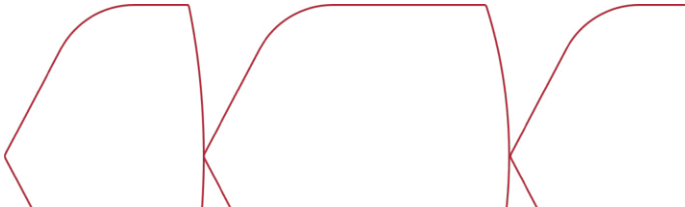


FIGURE 5.15 : Draping results on challenging poses with diverse garments.



CENTRALE LYON

LIRIS

CHAPTER 6

FNOpt: Resolution-Agnostic, Self-Supervised Cloth Simulation using Meta-Optimization with Fourier Neural Operators

Summary

We present FNOpt, a self-supervised cloth simulation framework that formulates time integration as an optimization problem and trains a resolution-agnostic neural optimizer parameterized by a Fourier neural operator (FNO). Prior neural simulators often rely on extensive ground truth data or sacrifice fine-scale detail, and generalize poorly across resolutions and motion patterns. In contrast, FNOpt learns to simulate physically plausible cloth dynamics and achieves stable and accurate rollouts across diverse mesh resolutions and motion patterns without retraining. Trained only on a coarse grid with physics-based losses, FNOpt generalizes to finer resolutions, capturing fine-scale wrinkles and preserving rollout stability. Extensive evaluations on a benchmark cloth simulation dataset demonstrate that FNOpt outperforms prior learning-based approaches in out-of-distribution settings in both accuracy and robustness. These results position FNO-based meta-optimization as a compelling alternative to previous neural simulators for cloth; thus reducing the need for curated data and improving cross-resolution reliability.

6.1 Introduction

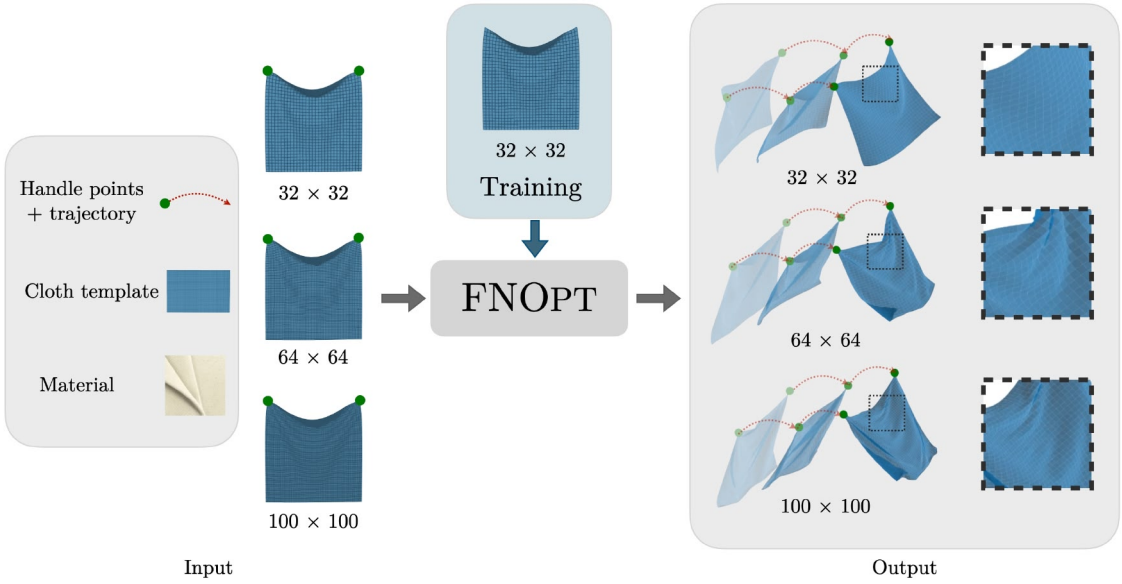


FIGURE 6.1 : Being trained on 32×32 mesh, FNOPT generalizes to various mesh resolutions, cloth templates, motion speeds, handle points and trajectories without retraining.

In the previous chapter, we explored PolyFit, an analytical surface representation based on local n-jet functions, and introduced its application OneFit, a self-supervised and mesh-agnostic framework that achieves temporally coherent garment draping by operating within a compact functional space. In this chapter, we take a step further by exploring operator learning to bridge the "resolution gap" and decouple simulation fidelity from training discretization.

Traditional physics-based cloth simulation approaches model the cloth dynamics by discretizing the classical time-varying partial differential equation of motion into an ordinary differential equation, and apply various numerical integration methods for simulation. Since the cloth deformation is stiff, traditional solvers require expensive computation for either small time steps in explicit methods Breen et al. [1994] or additional techniques for implicit ones Baraff and Witkin [1998] in order to avoid numerical instability. This hinders real-time applications such as realistic animations in computer-aided engineering. To overcome the computational limitations of traditional physics-based cloth simulators, data-driven methods Pfaff et al. [2021], Libao et al. [2023a] have emerged, where neural networks are trained on ground truth trajectories to predict cloth dynamics efficiently. However, these supervised

approaches require large amounts of curated training data and struggle to generalize beyond the distribution of motions and mesh resolutions seen during training. Modern self-supervised methods Santesteban et al. [2022b], Stotko et al. [2024] address the data burden by recasting implicit numerical integrators as optimization problems, enabling training through physics-based losses (cf. Sec. 2.2.3). While these methods reduce data requirements, their rollouts often suffer from stability issues. A recent development Wandel et al. [2025] demonstrates that training a neural optimizer via meta-learning enables adaptive iterative updates with improved precision. However, this approach still struggles to generalize across different mesh resolutions.

Recently, neural operators such as Li et al. [2020, 2021] are introduced that directly learn mappings between function spaces. Unlike conventional neural networks that can only interpolate between the learned spaces, these operators allow a genuine evaluation across continuous spatial domains, leading to a stronger generalization across various mesh resolutions. An important work in this direction is FNO Li et al. [2021], which performs a computationally efficient super-evaluation by transforming the computation onto a Fourier domain.

In this work, we present FNOpt, a novel self-supervised cloth simulation framework that combines meta-learning capabilities of Wandel et al. [2025] and efficient super-evaluation capabilities of FNOs. Therefore, the proposed framework enables a generalized meta-learning of various cloth motions across a diverse range of mesh resolutions. The super-evaluation capability of FNOs incorporated in cloth simulation allows a trained model on a lower resolution to perform a zero-shot generalization on a higher resolution (see Figure 6.1), capturing finer details that may not necessarily be present on the training data at lower resolution. Our experiments demonstrate a superior generalization and accuracy of FNOpt across a wide range of mesh resolutions, templates and boundary conditions compared to state-of-the-art methodologies in both supervised and self-supervised domains.

6.2 Simulating Cloth via FNOpt

6.2.1 Mathematical Foundation

Our goal is to simulate cloth dynamics, which can be modeled by the time-varying partial differential equation (PDE) of motion Eq. (2).

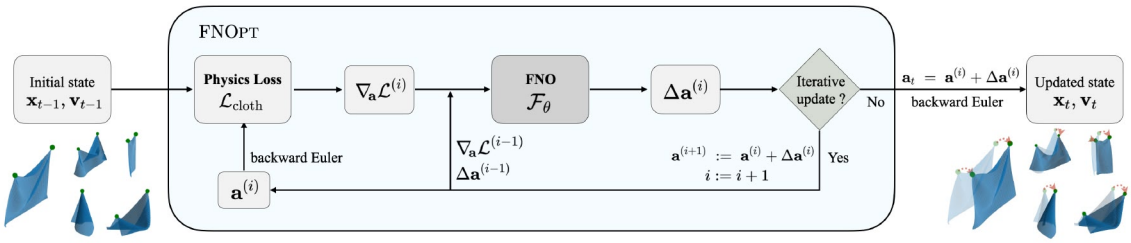


FIGURE 6.2 : FNOPT pipeline. At each simulation time step t, an inner optimization loop uses an FNO-based optimizer to predict updates $\Delta\mathbf{a}^{(i)}$ from physics-based loss gradients and state information. After N iterations, backward Euler advances the state to $(\mathbf{x}_t, \mathbf{v}_t)$. Superscript i indicates the inner iteration index.

By defining the state $u(p,t) := \left(x(p,t), \dfrac{\partial x(p,t)}{\partial t}\right)^\top$, we can reformulate cloth dynamics to the PDE problem

$$\frac{\partial u}{\partial t} = \mathcal{R}(u,t), \qquad \text{in } \Omega \times (0,\infty), \tag{57}$$

$$u = g, \qquad \text{in } \partial\Omega \times (0,\infty), \tag{58}$$

$$u = a, \qquad \text{in } \bar{\Omega} \times \{0\}, \tag{59}$$

where $\mathcal{R}$ is a possibly nonlinear partial differential operator, g is a known boundary condition on the boundary $\partial\Omega$ describing the handle points and their trajectories specified as input, and $a = u(\cdot, 0)$, is the initial condition describing position and velocity of points within the closed domain, $\bar{\Omega}$. Such dynamical systems can be solved by training a model u_θ to minimize the residual errors of Equations (57) to (59). In order to train it in a discretization-invariant way, we employ an FNO, $\mathcal{F}_\theta$ Li et al. [2021, 2024b], which maps from the initial condition a to the state $u = \mathcal{F}_\theta(a)$. In contrast to standard neural networks, this choice of architecture allows us to learn a mapping $a \mapsto u$ between function spaces, which is the key to generalize to perform super-evaluation on higher-resolution cloth to get finer details.

6.2.2 Training FNO

Because the dynamics in Equations (57) to (59) evolves over time, we must include the current time t as an input to the neural operator $\mathcal{F}_\theta$ so that it can produce at that time the corresponding cloth state $u(\cdot, t) = \mathcal{F}_\theta(a, t)$. However, this requires to train the network within a fixed time interval $(0, T]$ for some choice of large T,

which prevents simulation for out-of-distribution $t > T$. Therefore, as suggested by Brandstetter et al. [2022], we instead train an autoregressive neural operator $\mathcal{F}_\theta$ that maps current state $u(\cdot, t)$ to next state $u(\cdot, t + \Delta t)$ for initial condition $u(\cdot, 0) = a$, allowing us to simulate up to any time step.

As mentioned in Section 6.2.1, we must train the neural operator by minimizing the residual errors of Equations (57) to (59). Since the initial and boundary conditions associated with each sequence of cloth motion are known, the computation of the residual errors for Equations (58) and (59) is straightforward. However, the physics prior in Equation (57) requires an expensive computation of high-order derivatives of the network. To simplify the computation, we discretize the PDE in Equation (57) by using backward Euler method. Thus, we write

$$\mathbf{M}\frac{\mathbf{x}_{t+1} - \mathbf{x}_t - \Delta t \mathbf{v}_t}{\Delta t^2} + \frac{\partial \mathcal{L}_{\text{int}}}{\partial \mathbf{x}} = \mathbf{f}_{\text{ext}}\left(\mathbf{x}_{t+1}, \frac{\mathbf{x}_{t+1} - \mathbf{x}_t}{\Delta t}\right), \tag{60}$$

where we have incorporated the damping term into external force $\mathbf{f}_{\text{ext}}$, $\mathbf{x}$ and $\mathbf{v}$ are the positions and velocities, respectively. Consistent with the optimization-based time integration detailed in Section 2.2.3, we recast the solution of this system as minimizing the following loss function:

$$\mathcal{L}_{\text{cloth}} = \mathcal{L}_{\text{int}} + \mathcal{L}_{\text{ext}} + \mathcal{L}_{\text{inertia}}. \tag{61}$$

The first term $\mathcal{L}_{\text{int}}(\mathbf{x}_t)$ corresponds to the internal potential energies of cloth, including stretching, shearing, and bending. The second term $\mathcal{L}_{\text{ext}}(\mathbf{x}_t)$ refers to the external forces such as gravity and wind. The last term $\mathcal{L}_{\text{inertia}}$ imposes the inertia constraint (cf. Section 3.4), which makes wrinkles and dynamic behavior appear. We refer to Stotko et al. [2024] for more details on the loss terms.

6.2.3 Training Neural Optimizer via Meta-learning

Using the loss functions formalized above, we train a neural cloth simulator that predicts the acceleration $\mathbf{a}_{t+1}$ from the current position $\mathbf{x}_t$ and velocity $\mathbf{v}_t$, given some boundary conditions. We use the training cycle proposed by Stotko et al. [2024], where the data pool is initialized with random cloth states and then progressively augmented with the model's own predictions. This technique allows us to train the model in a self-supervised manner without using any data generated

from traditional physics-based simulator (PBS). However, the predictions can be unstable due to error accumulation over frames when simulating long sequences. Hence, we follow Wandel et al. [2025] to train a neural optimizer via meta-learning technique. Concretely, rather than predicting the acceleration directly, we initialize the prediction as $\mathbf{a}_{t+1}^{(0)} = 0$ and iteratively optimize it using the following update rule

$$\mathbf{a}_{t+1}^{(i+1)} = \mathbf{a}_{t+1}^{(i)} + \Delta\mathbf{a}_{t+1}^{(i)}, \qquad \text{for } i = 0, ..., N-1, \tag{62}$$

where N is the number of iterations used for each time step $t+1$, and the update direction $\Delta\mathbf{a}_{t+1}^{(i)}$ is given by the neural operator $\mathcal{F}_\theta$. The resulting acceleration is then used to evaluate the next cloth state using backward Euler method as

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \Delta t \mathbf{a}_{t+1}, \tag{63}$$

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \Delta t \mathbf{v}_{t+1}. \tag{64}$$

Combining all features introduced so far, our FNOpt framework follows the pipeline in Figure 6.2.

6.2.4 Inference

At inference time, given initial position and velocity of the cloth as well as boundary conditions including handle points and their trajectories, FNOpt performs an autoregressive rollout. For each time step, we apply the learned optimizer $\mathcal{F}_{\theta^\star}$ for a fixed number N of inner updates to minimize the objective in Equation (61). Note that the same trained $\mathcal{F}_{\theta^\star}$ is used across all experiments, no fine-tuning is required.

Optionally, to handle self-collision between cloth, we can augment the objective with a repulsive term, $\mathcal{L}_{\text{rep}}$ Lee et al. [2023] at inference time, which reduces interpenetration without retraining:

$$\mathcal{L}_{\text{rep}} = \lambda_{\text{rep}} \sum_{i=1}^{N} \sum_{j \in \mathcal{A}_i} -\log\big(\|\mathbf{x}_i - \mathbf{x}_j\|^2\big).$$

Where $\mathcal{A}_i \coloneqq \{\, j \nsim i : \|\mathbf{x}_i - \mathbf{x}_j\| < \delta \,\}$. Here $j \nsim i$ denotes non-adjacent vertices $(i, j) \notin E$ for edge set E. We set δ to the local grid spacing; see Section 6.3.3 for ablations.

Algorithm 2 FNOPT Simulation Rollout (Single Time Step)

Input: Current state $(\mathbf{x}_{t-1}, \mathbf{v}_{t-1})$, boundary conditions g (handle trajectories), trained FNO optimizer F_θ, number of inner iterations N, time step Δt
Output: Next state $(\mathbf{x}_t, \mathbf{v}_t)$

▷ Initialization
$\mathbf{a}_t^{(0)} \leftarrow \mathbf{0}$ ▷ Initialize acceleration as zero
▷ Inner Meta-Optimization Loop
for $i = 0$ **to** $N - 1$ **do**
  $\mathbf{v}_t^{(i)} \leftarrow \mathbf{v}_{t-1} + \Delta t \mathbf{a}_t^{(i)}$ ▷ Compute intermediate velocity
  $\mathbf{x}_t^{(i)} \leftarrow \mathbf{x}_{t-1} + \Delta t \mathbf{v}_t^{(i)}$ ▷ Compute intermediate position
  $\mathcal{L}_{cloth}^{(i)} \leftarrow \mathcal{L}_{int} + \mathcal{L}_{ext} + \mathcal{L}_{inertia}$
  ▷ Compute physics-based loss
  $\nabla_{\mathbf{a}} \mathcal{L}^{(i)} \leftarrow \frac{\partial \mathcal{L}_{cloth}^{(i)}}{\partial \mathbf{a}_t^{(i)}}$ ▷ Calculate gradients
  $\Delta \mathbf{a}_t^{(i)} \leftarrow F_\theta(\nabla_{\mathbf{a}} \mathcal{L}^{(i)}, \nabla_{\mathbf{a}} \mathcal{L}^{(i-1)}, \Delta \mathbf{a}_t^{(i-1)})$ ▷ Predict update using FNO
  $\mathbf{a}_t^{(i+1)} \leftarrow \mathbf{a}_t^{(i)} + \Delta \mathbf{a}_t^{(i)}$ ▷ Update acceleration
end for
▷ State Update via Backward Euler
$\mathbf{a}_t \leftarrow \mathbf{a}_t^{(N)}$ ▷ Final optimized acceleration
$\mathbf{v}_t \leftarrow \mathbf{v}_{t-1} + \Delta t \mathbf{a}_t$ ▷ Update velocity
$\mathbf{x}_t \leftarrow \mathbf{x}_{t-1} + \Delta t \mathbf{v}_t$ ▷ Update position
return $(\mathbf{x}_t, \mathbf{v}_t)$

6.3 Experiments

We use the official implementation provided in the NeuralOperator library Kossaifi et al. [2024] which includes the FNO. Regarding the network hyper-parameters, we use 4 Fourier layers, with the number of Fourier modes set to 8 for both spatial dimensions (height and width). The number of hidden channels for intermediate Fourier layers, the lifting layer, the projection layer are set to 64, 256, 64, respectively. We train only at 32×32 resolution, with data pool of 1000 data points updated progressively during the self-supervised training. We set the batch size to 10 and learning rate to 10^{-3} and train the network for 100 epochs, which takes 8 hours on an NVIDIA H100 GPU. For experiments, we use the dataset from Libao et al. [2023a] which contains trajectories of a square cloth generated by ARCSim given sequences of handle point trajectory, and run the evaluation on NVIDIA A100 GPU. We use the evaluation set which contains 11 trajectories for translation and 4 trajectories for rotation run at 60 fps, which serves as ground truth. We selected various baselines

for comparison: 1) the self-supervised cloth model used in PGSfT Stotko et al. [2024]; 2) Metamizer Wandel et al. [2025], the neural optimizer baseline with iterative refinement; and 3) MeshGraphNetRP (MGNRP) Libao et al. [2023a], the supervised baseline trained using PBS data. We use the official checkpoints released by the respective authors, except for Metamizer, which is re-trained exclusively with the cloth data pool to ensure a fair comparison. The physics-based losses in Equation (61) are different from ARCSim, which leads to different meaning of material coefficients in our setting. Hence, we apply grid search to find the corresponding material for PGSfT, Metamizer and FNOpt on the simulated sequences. Unless otherwise noted, the repulsive term is disabled. We refer to Section 6.3.3 for experiments with the repulsive loss.

6.3.1 SOTA Comparison

Training resolution. We first evaluate FNOpt at the training resolution of 32×32. We measure the chamfer distance (e_{CD}) between the ground truth and the predicted point clouds, each of which is uniformly sampled 10^4 points from the corresponding mesh. As shown in Table 6.1, compared to other self-supervised methods, FNOpt significantly outperforms PGSfT and surpasses Metamizer in most of the sequences. MGNRP is trained with supervision on domain-matched ground truth trajectories at the same resolution and therefore attains the lowest e_{CD} in-domain. However, as we will show in later sections, this advantage does not transfer to out-of-domain settings, making it less scalable.

Figure 6.3 shows the evaluated shapes and their Euclidean distance to the PBS ground truth sequence `xy_v2`. PGSfT suffers from noticeable drift over long horizons resulting in high global errors. In addition, it fails to recover high-frequency wrinkles, leading to unnaturally stiff cloth behavior. Metamizer and FNOpt perform quite well at training, showing good dynamics and wrinkles. Although achieving the best quantitative results, MGNRP exhibits subtle but noticeable temporal jitters (visible in supplementary video). Figure 6.5 (top) plots the per-frame chamfer distance e_{CD} on `xy_v2`; PGSfT produces significantly higher error than the other methods, showing limited dynamic fidelity and a lack of fine-scale detail.

Super-evaluation. To show our generalizability in multi-resolution, we perform simulation with higher resolutions 64×64 and 100×100. Although using graph-based

network that can deal with different mesh topologies, MGNRP fails to simulate at higher resolutions. PGSfT's U-Net implementation supports inference only at the training resolution, cross-resolution results are unavailable. Metamizer also uses U-Net but their implementation can be run on different resolutions. As shown in Table 6.1, Metamizer is still able to get decent results at 64×64 but struggles at 100×100. Thanks to its FNO architecture, FNOpt can perform super-evaluation automatically at both 64×64 and 100×100. Since the wrinkles and details are simulated more precisely in these cases, it improves the average chamfer distance to 4.8 and 4.7 respectively, bridging the gap with respect to supervised method MGNRP. We show comparison for super-evaluation of sequence `xy_v2` in Figure 6.4. Metamizer exhibits large errors and temporal instabilities on a finer 64×64 mesh, and diverges at 100×100. MGNRP also results in divergence at both 64×64 and 100×100 resolutions. In contrast, FNOpt produces visually plausible cloth with realistic wrinkles and maintains stable, accurate rollouts across all tested resolutions, demonstrating superior fidelity and resolution-agnostic generalization (more details in supplementary video). Furthermore, the per-frame chamfer distance in the lower figure of Figure 6.5 demonstrates our stability when simulating higher resolutions.

To complement the Chamfer distance analysis, we also report the 3D error e_{3D} (defined in Sidhu et al. [2020]) in Table 6.2, computed using the known vertex correspondences after re-meshing the ground truth. Both metrics show broadly consistent trends across methods, though some variations in relative rankings can be observed.

Interpretation of the results. We analyze why the different approaches attain varying levels of motion realism, fine-scale detail, and temporal stability. PGSfT Stotko et al. [2024] is trained in a fully self-supervised fashion. Its one-shot architecture, however, cannot correct accumulated errors, leading to noticeable drift and a loss of high-frequency wrinkles in long rollouts. MGNRP Libao et al. [2023a] is trained in a supervised fashion on pre-computed ground truth trajectories, achieving high accuracy for motions, velocities, and mesh resolutions seen during training. It can handle irregular meshes due to its use of the MeshGraphNets architecture. However, the model extrapolates poorly, and its performance degrades significantly when tested on finer resolutions or faster motions outside the training distribution. Metamizer Wandel et al. [2025] improves visual fidelity and wrinkle detail over PGSfT by

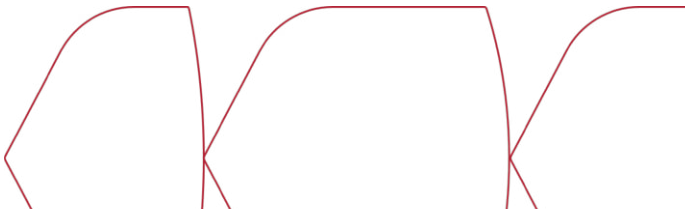
employing an iterative, meta-learned optimizer. Nevertheless, its performance drops sharply on finer meshes, and rollouts exhibit temporal instabilities. FNOpt produces stable, high-fidelity rollouts across all tested resolutions. We attribute this robustness to the resolution-agnostic FNO backbone, which supports zero-shot inference on previously unseen mesh resolutions, and to the meta-learned optimizer that optimizes per-step acceleration with only a few iterations.

Runtime performance. All timings were measured on an NVIDIA A100 GPU with a 1024-vertex mesh. With the default setting of 10 optimizer iterations, FNOpt runs at 61 ms/frame, and at 33 ms/frame with 5 iterations at the expense of a modest loss in accuracy. Enabling the repulsive term incurs an additional 3 ms/frame. In comparison, the supervised baseline MGNRP completes a frame in 40 ms using the authors' released model, while PGSfT, which requires only a single feed-forward pass, achieves 7 ms/frame but fails to capture fine-scale wrinkles. Metamizer with iterative refinement runs at 63 ms/frame. The offline finite-element solver ARCSim requires 404 ms/frame. Overall, FNOpt offers a compelling trade-off: its per-frame runtime is about 1.5× that of the supervised baseline while delivering substantially higher accuracy on cross-resolution rollouts, and is roughly 6× faster than the PBS.

6.3.2 Boundary Condition Generalization

Beyond scaling across resolutions, FNOpt remains robust under changes to boundary conditions (e.g. handle speed and placement, mesh sizes, etc), in contrast to baseline methods that struggle in such settings.

Speed generalization. We evaluate the speed generalization ability of FNOpt and compare with the supervised approach MGNRP. We create new motion sequences by accelerating the original ones using interpolation. We test speed factors $\alpha \in \{1.2, 1.5, 2\}$. The results are shown in Figure 6.6. Both methods are able to handle slight speed increase, as shown in 1.2× with no noticeable artifacts. When speed factor goes to 1.5×, MGNRP shows overstretching artifacts around upper corners. With 2× speed, severe artifacts are shown and MGNRP simulation becomes unstable. These behaviors happen because the accelerated sequences are not present in the training set, preventing MGNRP from generalizing to out-of-distribution motions. On the other hand, FNOpt remains stable at all tested motion speeds without



Sequence name	MGNRP [93]			PGSfT [160]	Metamizer [172]			FNOpt [Ours]		
	res32	res64	res100	res32	res32	res64	res100	res32	res64	res100
xy_v2	**4.8**	–	–	25.3	7.1	33.6	–	7.1	**5.2**	**4.2**
xy_v2_opp	**5.0**	–	–	25.2	7.7	14.7	12.9	6.5	**5.3**	**4.8**
yz_v2	**2.6**	–	–	10.7	11.6	3.6	77.2	4.9	**3.4**	**5.5**
yz_v2_opp	**3.0**	–	–	13.5	4.9	4.2	–	4.3	**3.2**	**3.6**
xz_v2	**2.7**	–	–	22.5	6.5	**6.6**	–	8.9	**6.6**	**3.9**
xyz_v2	**5.4**	–	–	21.9	6.3	**4.9**	29.9	6.4	**4.9**	**4.3**
xyz_v2_opp	**4.3**	–	–	22.7	6.4	4.9	–	6.4	**4.1**	**5.2**
xyz_v3	**3.5**	–	–	21.4	7.2	10.2	–	6.4	**4.7**	**3.7**
xyz_v3_opp	**3.0**	–	–	20.6	29.2	12.1	–	6.1	**4.8**	**4.6**
xyz_v4	**3.7**	–	–	18.9	7.2	4.8	21.2	6.7	**3.7**	**3.5**
xyz_v4_opp	**3.3**	–	–	19.0	5.8	6.2	–	7.4	**4.2**	**4.7**
rot_h0	**5.4**	–	–	17.5	8.2	21.1	–	6.9	**5.2**	**6.2**
rot_h0_opp	**5.7**	–	–	17.5	9.3	6.8	–	7.6	**6.6**	**5.9**
rot_h1	**5.1**	8.3	–	18.6	7.8	7.6	–	7.6	**4.5**	**6.0**
rot_h1_opp	**5.3**	9.4	–	16.7	8.3	7.7	–	7.5	**5.7**	**6.2**
Avg $\pm$ σ	**4.2** $\pm$1.1	–	–	19.5 $\pm$3.9	8.9 $\pm$5.6	10.0 $\pm$7.8	–	6.7 $\pm$1.1	**4.8** $\pm$1.0	**4.7** $\pm$0.9

TABLE 6.1 : Comparison of the chamfer distance e_{CD} (scaled by 10^3) at different resolutions for each motion sequence. Methods compared: MGNRP Libao et al. [2023a], PGSfT Stotko et al. [2024], Metamizer Wandel et al. [2025], and ours. The best and second-best results are shown in **bold** and underline, respectively.

artifacts around handles.

Flexible handle placement. We evaluate 5 different configurations of the handle points: *Diagonal*, *Mid-Edge*, *Corner-Center*, *Single Mid-Edge* and *Single Center*. *Diagonal* puts the two handles on the opposite corners. For *Mid-Edge*, we fix the handles on two midpoints of the opposite sides of the mesh. *Corner-Center* uses one corner and the center of the cloth as handles. Each of the two settings *Single Mid-Edge* and *Single Center* uses only a single handle point on the midpoint of the upper edge and the center of the cloth, respectively.

The qualitative comparison is illustrated in Figure 6.7. We can see that FNOpt supports arbitrary handle placements as boundary conditions, yielding stable rollouts with preserved fine-scale wrinkles without retraining or further fine-tuning. MGNRP only supports two-handle configuration with both handles at the top corners, and fails to simulate these 5 configurations. Metamizer utilizes the same formulation of self-supervised neural optimizer as FNOpt, but yields unstable simulation.

Sequence name	MGNRP [93]			PGSfT [160]	Metamizer [172]			FNOpt[Ours]		
	res32	res64	res100	res32	res32	res64	res100	res32	res64	res100
xy_v2	**11.0**	–	–	24.1	15.5	27.8	–	15.4	**12.7**	**11.0**
xy_v2_opp	**11.8**	–	–	23.2	16.2	17.9	16.5	14.5	**12.0**	**10.5**
yz_v2	**9.7**	–	–	20.1	17.6	12.1	34.3	15.0	**11.3**	**13.1**
yz_v2_opp	**8.6**	–	–	21.4	13.8	12.8	–	13.5	**10.3**	**10.9**
xz_v2	**7.9**	–	–	21.8	13.2	**11.9**	–	15.2	13.3	**9.9**
xyz_v2	**10.7**	–	–	21.7	14.7	**11.4**	18.9	14.6	12.0	**10.5**
xyz_v2_opp	**10.6**	–	–	22.7	15.4	11.5	–	14.7	**10.8**	**11.4**
xyz_v3	**9.1**	–	–	22.7	14.5	15.3	–	15.2	**12.2**	**10.6**
xyz_v3_opp	**9.7**	–	–	22.1	23.5	20.7	–	14.3	**11.8**	**10.9**
xyz_v4	**9.4**	–	–	21.9	14.8	11.4	22.1	14.4	**10.7**	**9.9**
xyz_v4_opp	**9.0**	–	–	21.4	13.4	13.3	–	15.2	**11.2**	**11.2**
rot_h0	**12.5**	–	–	23.5	17.0	22.4	–	16.1	**14.3**	**15.1**
rot_h0_opp	**12.8**	–	–	22.9	18.3	**14.8**	–	17.1	15.8	**14.8**
rot_h1	**11.4**	15.4	–	23.8	16.8	14.9	–	16.8	**13.9**	**14.6**
rot_h1_opp	**11.9**	16.4	–	23.2	17.8	**15.0**	–	17.0	15.8	**14.7**
Avg $\pm$ σ	**10.4** $\pm$1.4	–	–	22.4 $\pm$1.0	16.2 $\pm$2.5	15.5 $\pm$4.6	–	15.3 $\pm$1.0	**12.6** $\pm$1.7	**11.9** $\pm$1.9

TABLE 6.2 : Comparison of the 3D error e_{3D} ($\times 10^2$; lower is better) at different resolutions for each motion sequence. Methods compared: MGNRP Libao et al. [2023a], PGSfT Stotko et al. [2024], Metamizer Wandel et al. [2025], and ours. Within each resolution group, the best and second-best results are shown in **bold** and underline, respectively.

Non-square meshes. Trained on square grid of the cloth, FNOPT is able to perform rollouts on grids with non-square aspect ratios. We demonstrate rollouts on six different mesh sizes across two motion sequences in Figure 6.8. Without any retraining or parameter tuning, the results indicate that the learned optimizer transfers across domain shapes, preserving fine-scale wrinkles and long-horizon stability.

6.3.3 Ablation Studies

Iterative Update. The iterative update discussed in Section 6.2.3 is crucial for our framework. The improvement is already demonstrated in Metamizer results (see Table 6.1, Figures 6.3 and 6.5), in which the iterative update is employed to achieve higher precision compared to PGSfT. We further train a baseline model using PGSfT formulation that directly predicts the acceleration $\mathbf{a}_{t+1}$ in a single forward pass, without using iterative update in Equation (62). This baseline is trained in a supervised fashion using the same loss in Equation (61), and uses FNO backbone rather than U-Net. As shown in Table 6.4, this approach yields significantly less accurate rollouts, and fails to capture high-frequency details such as wrinkles.

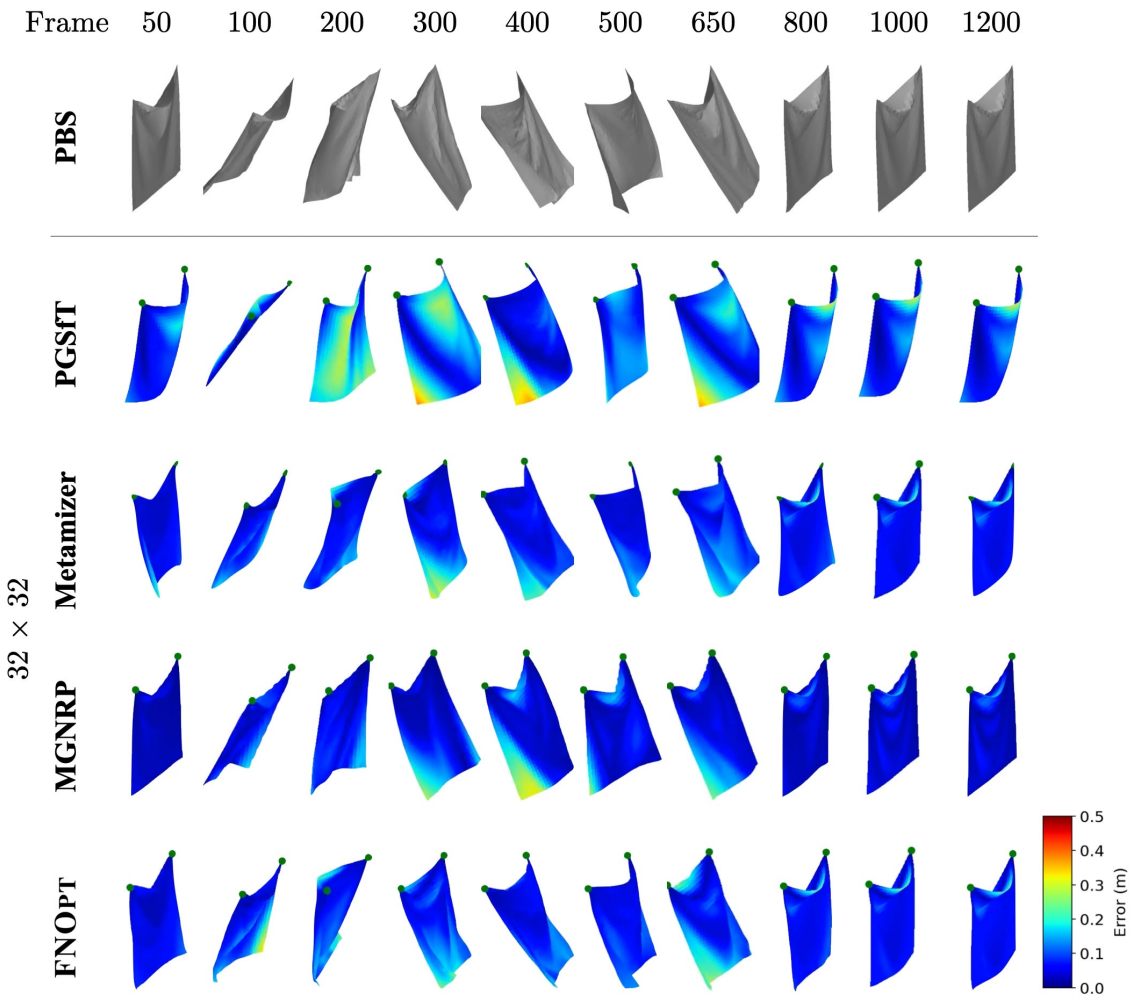


FIGURE 6.3 : Vertex-wise error maps on the `xy_v2` sequence on training resolution. Colors indicate the Euclidean distance between rollout and PBS.

Neural Optimizer. To assess the effectiveness of the neural optimizer formulation in FNOPT framework, we replace the trainable neural optimizer by a classical optimizer using a step size and direction computed from the gradients with respect to the losses in Equation (61). These baselines retain the same optimization framework but lack any data-driven adaptation of step size or search direction. For each method, we perform an independent learning-rate sweep and report the best-performing accuracy in Table 6.4. Full table can be found in Table 6.6. All experiments were run on a 64×64 mesh.

We first evaluate vanilla gradient descent (GD) Cauchy [1847] and Adam Kingma and Ba [2014], two first-order methods. Even with a carefully tuned learning rate,

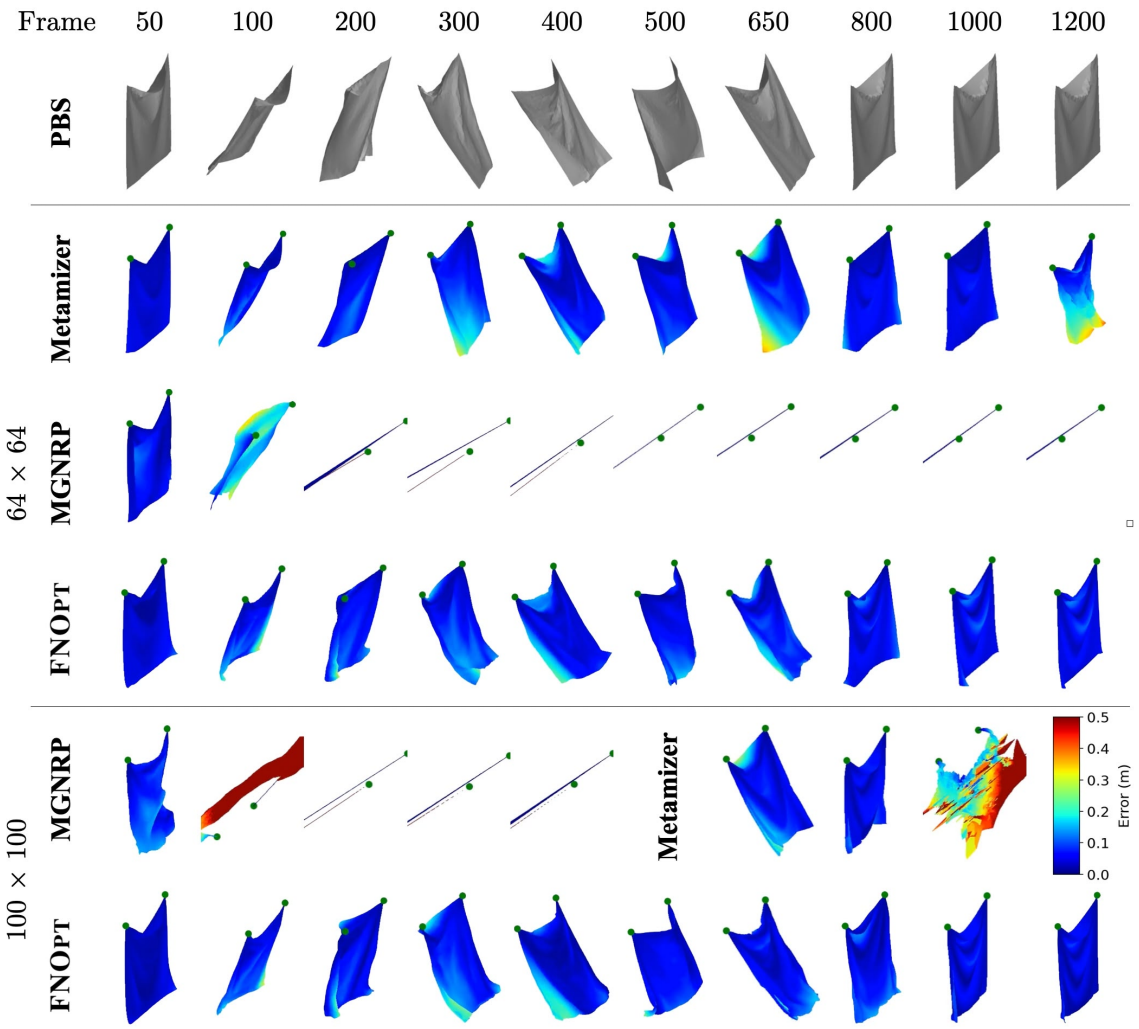


FIGURE 6.4 : Vertex-wise error maps on finer resolutions on the `xy_v2` sequence.

GD fails to significantly reduce the cloth loss $\mathcal{L}_{\text{cloth}}$ within N=500 iterations per time step, and the resulting rollouts exhibit noticeable artifacts. Adam benefits from adaptive step sizes and eventually converges provided that a large number of inner iterations ($\sim$500) is allowed. We use its standard hyperparameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Despite converging more reliably than GD, Adam remains more than 30$\times$ slower than FNOPT (see Figure 6.9) and produces higher errors in both e_{CD} and e_{3D}.

We additionally compare with limited-memory BFGS (L-BFGS), a quasi-Newton method. We adopt the standard two-loop recursion to compute search directions, a fixed step size to ensure a fair comparison of gradient evaluations with first-order

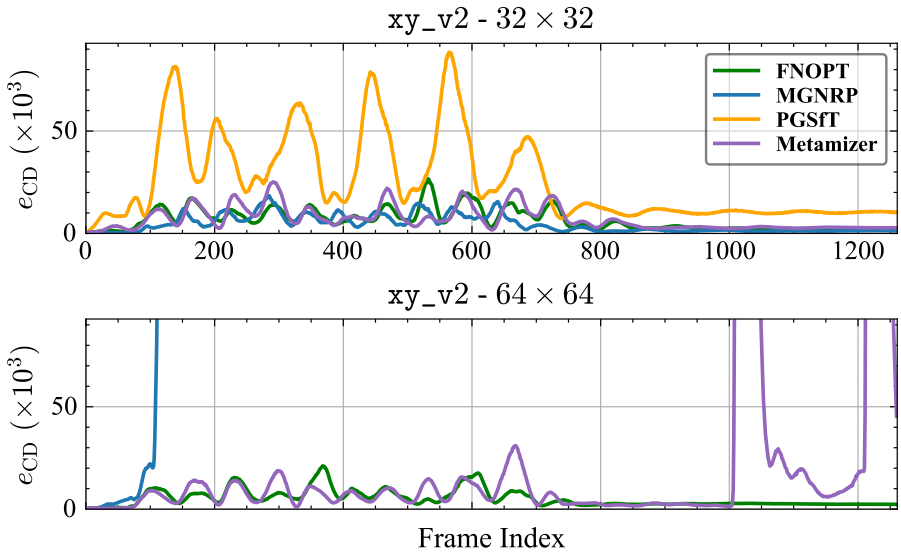


FIGURE 6.5 : Per-frame e_{CD} ($\times 10^3$) on `xy_v2` sequence. All models are trained at 32×32. FNOPT achieves second lowest error on 32×32 resolution and generalizes to the finer 64×64 resolution.

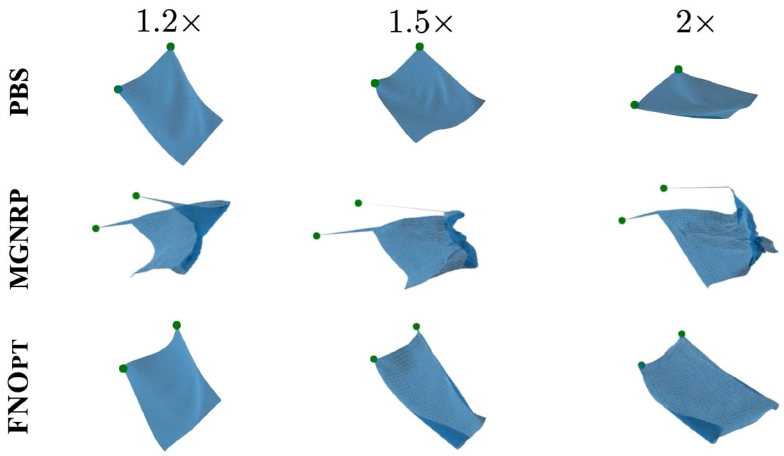


FIGURE 6.6 : Generalization to speeds on a representative frame of the `xy_v2` rollout. MGNRP exhibits overstretching artifacts near the handles while FNOPT remains stable.

baselines, and a history size of m=5; we refer to this baseline as L-BFGS (fixed step, no line search). This variant has $O(N^2)$ time complexity because of iteration over history, where N is the number of iterations per time step, but it turns out to converge in fewer iterations than GD and Adam and achieves the most accurate results with 100 iterations per time step. Yet, this is still 10× slower than ours. Curiously, we evaluate its performance on a finer 100×100 mesh with 70, 100, 120,

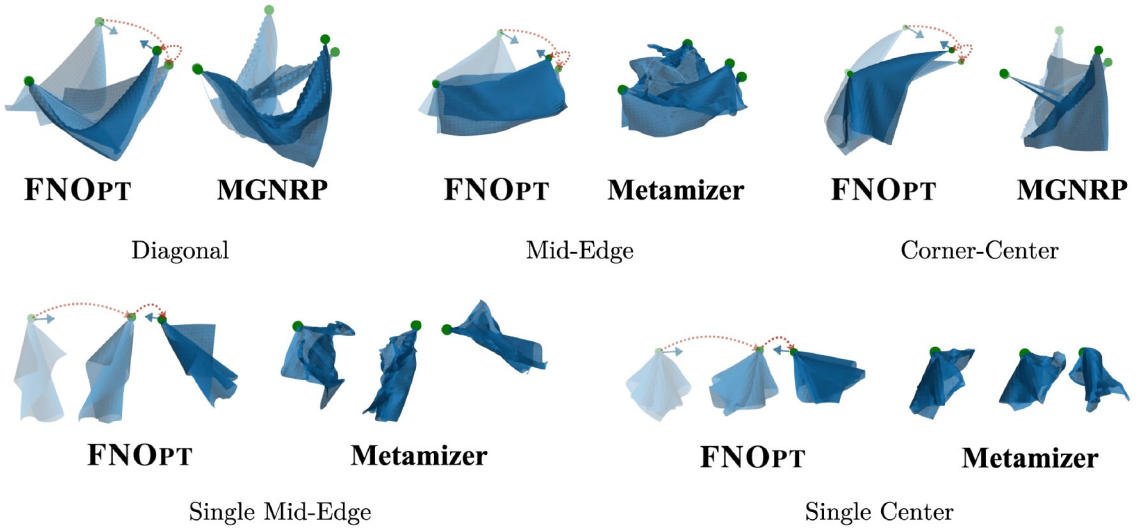


FIGURE 6.7 : Qualitative comparison of generalization to various handle configurations. The top row shows configurations with two control points. The bottom row contains single-handle scenarios. FNOPT consistently produces plausible and stable cloth dynamics across all settings. Green points denote control handles. Red dashed arrows denote handle trajectories. Blue arrows indicate acceleration directions. Repulsive loss is activated.

150 iterations per time step and record the e_{CD} in Table 6.3. We find that it achieves the lowest metric at around 120 iterations per time step, and remains at low value for higher N. FNOPT achieves on par performance with only 10 iterations per time step.

Finally, with only ten iterations per time step, FNOPT achieves low error and stable, high-fidelity rollouts while being at least an order of magnitude faster than all classical optimizers. Figure 6.10 compares of e_{CD} and $\mathcal{L}_{\text{cloth}}$ across classical optimizers and our method as a function of N. Unlike classical optimizers, which require at least 100 iterations to achieve low error, FNOPT reaches both low e_{CD} and $\mathcal{L}_{\text{cloth}}$ within just 10 iterations.

Figure 6.9 shows the runtime comparison between different optimizers, evaluated on a node equipped with an AMD EPYC 7543 CPU and an NVIDIA A100 GPU. This highlights the advantage of a learned, resolution-agnostic optimizer over classical update rules.

FNO Architecture. As discussed in Section 6.2.1, we adopt FNO as the backbone in our framework because its global spectral kernels capture long-range interactions and naturally support resolution-agnostic inference, producing realistic wrinkles and

Sequence name	L-BFGS (step size = 1)				Ours
	70	100	120	150	10
xy_v2	73.8	7.9	5.3	7.3	4.2
xy_v2_opp	74.9	7.3	5.0	7.2	4.8
yz_v2	44.4	7.5	3.2	2.4	5.5
yz_v2_opp	40.9	7.3	3.3	2.4	3.6
xz_v2	40.6	4.9	5.7	8.1	3.9
xyz_v2	91.1	6.6	4.0	5.0	4.3
xyz_v2_opp	87.7	6.4	3.9	5.2	5.2
xyz_v3	85.7	6.7	3.8	5.1	3.7
xyz_v3_opp	88.1	6.0	3.8	5.3	4.6
xyz_v4	90.5	6.9	3.8	5.1	3.5
xyz_v4_opp	89.5	6.9	3.8	4.8	4.7
rot_h0	27.1	5.1	5.3	8.0	6.2
rot_h0_opp	27.3	5.0	5.6	7.9	5.9
rot_h1	26.9	4.9	5.4	7.8	6.0
rot_h1_opp	26.8	5.1	5.8	7.9	6.2
Avg	58.3	6.1	4.7	5.8	4.7
σ	±25.2	±1.1	±0.9	±1.7	±0.9

TABLE 6.3 : Ablation on a 100×100 mesh comparing e_{CD} ($\times 10^3$) for L-BFGS and ours under varying *numbers of iterations* per time step.

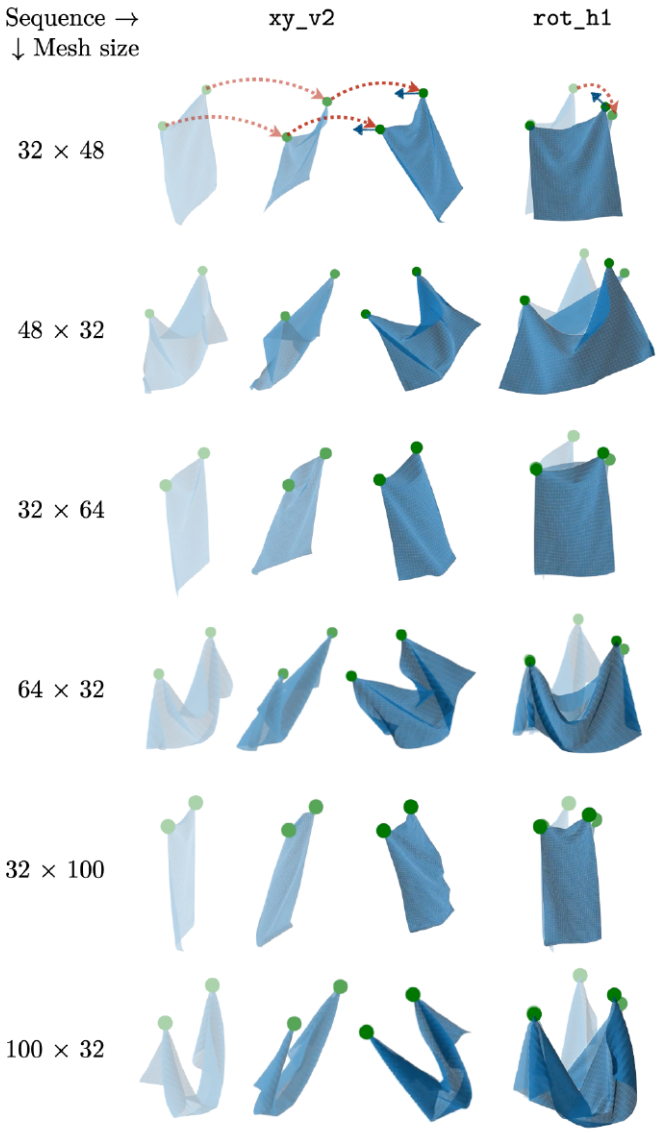


FIGURE 6.8 : Generalization to non-square cloth meshes.

faithful high-resolution details. We have compared FNO with the U-Net backbone used by Metamizer in Section 6.3.1 and shown a clear advantage in both accuracy and visual quality over various resolution. We further show the benefit of using FNO over MeshGraphNets Pfaff et al. [2021], a popular graph-based simulator in cloth and fluid dynamics. Note that we keep the same formulation as the FNOPT framework and only change the network architecture. We refer to this variant as MGN. For hyper-parameters, we keep the default number of message-passing steps (15); the hidden size and the number of layers for the encoder, graph-net blocks, and

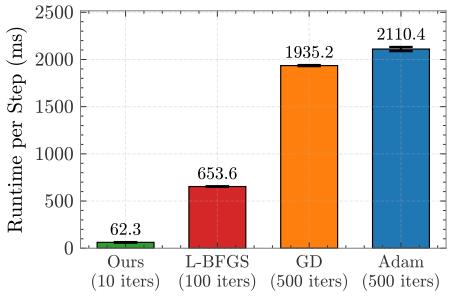


FIGURE 6.9 : Runtime comparison across different optimizers. Each bar shows per-step runtime (ms); for every optimizer we use the minimum number of iterations that achieves convergence of $\mathcal{L}_{\text{cloth}}$ (except GD, which did not converge). Results on 64×64 resolution.

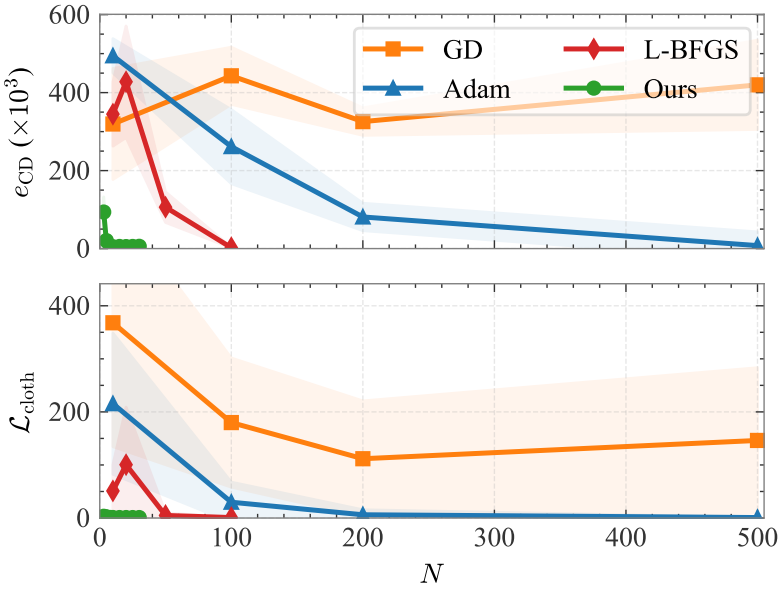


FIGURE 6.10 : Comparison of e_{CD} and $\mathcal{L}_{\text{cloth}}$ across classical optimizers and our method, versus the number of iterations per time step. Results on 64×64 resolution.

decoder are set to 128 and 3, respectively. We train the network for 100 epochs, which takes around 40 hours on an NVIDIA H100 GPU. We evaluate on the evaluation sequences of the datasets from Libao et al. [2023a]; results are shown in Figure 6.11. The MGN variant exhibits visible high-frequency oscillations already at the training resolution and diverges rapidly on 64×64 meshes. Despite using identical loss objectives, iteration counts, and training strategy, the MGN version exhibits visible high-frequency oscillations even at the training resolution and diverges rapidly on 64×64 meshes, see Figure 6.11.

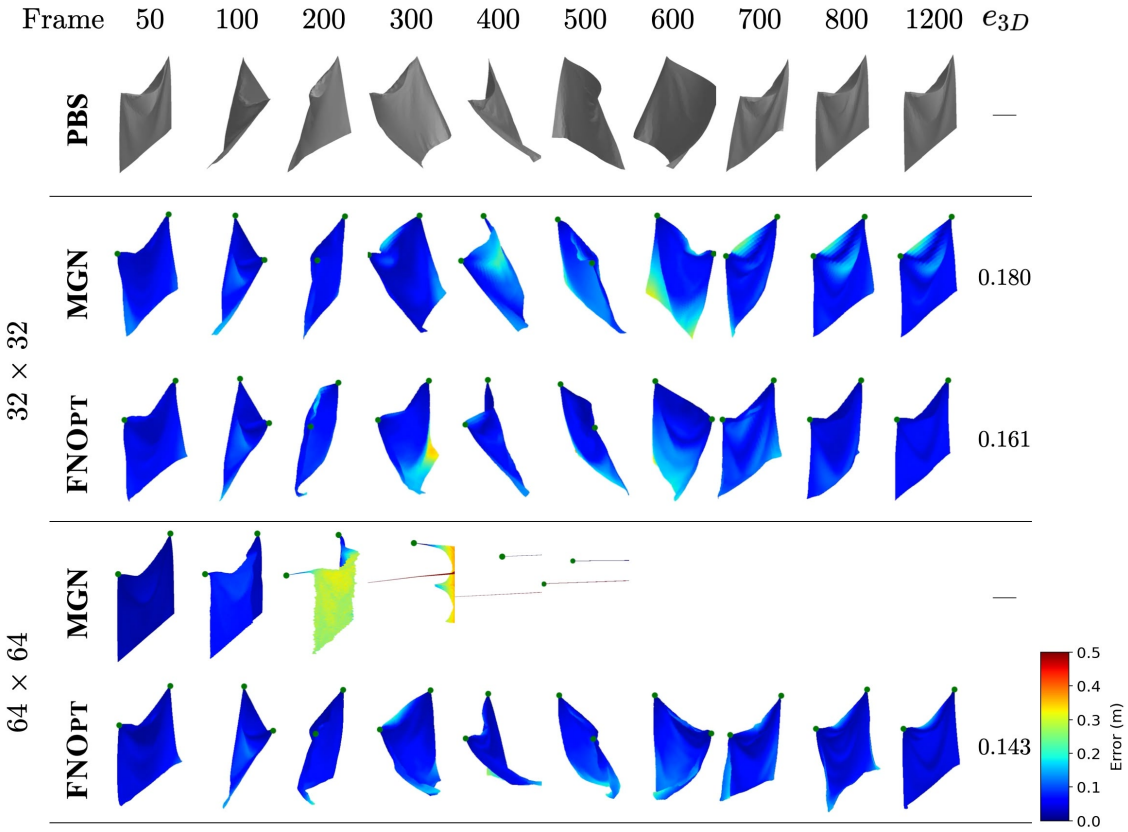


FIGURE 6.11 : Vertex-wise error maps on the `rot_h0` sequence, compared to MGN. Colors indicate the Euclidean distance between rollout and the PBS.

We hypothesize that the instability arises from the autoregressive rollouts: with a fixed number of message-passing steps, MeshGraphNets can propagate information only within a limited neighborhood, so local errors are re-fed into the next step, propagate and amplify over time. Since the learned optimizer relies on accurate per-vertex gradients, even small per-step misestimations may accumulate and drive the system into unstable regimes. By contrast, the global spectral kernels of FNOs

couple every vertex to the entire cloth surface, yielding smoother, more consistent updates and preventing long-horizon error growth.

Super-evaluation. We emphasize that FNOpt framework allows for super-evaluation that is trained on a lower resolution and generalizes directly to higher resolution, capturing finer details that were not presented in the low-resolution training data. This property is essential for physics simulation, especially when simulating cloth with folds and wrinkles. We clarify that such property is different from interpolation, which estimates high-resolution rollouts from lower-resolution prediction. To evaluate, we estimate the 64×64 rollouts by bilinearly interpolating the predicted 32×32 accelerations given by FNOpt. As can be seen in Figure 6.12, the resulting interpolation recovers fewer fine-scale wrinkles due to information loss during upsampling, showcasing the necessity of *super-evaluation* at the target resolution.

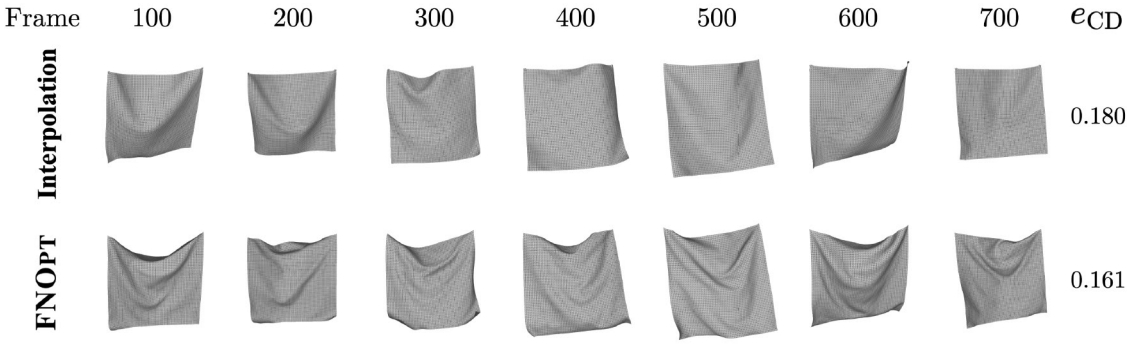


Figure 6.12 : Visual comparison of 64×64 rollouts using bilinear interpolation (upper row) and ours (lower row).

Method	$e_{CD} \downarrow$
FNOpt (Ours)	4.8 $\pm$0.99
No iterative update	24.7 $\pm$3.22
GD (lr=0.01, 500 iters)	419.9 $\pm$114.4
Adam (lr=0.1, 500 iters)	7.7 $\pm$2.6
Interpolation to hi-res	7.0 $\pm$1.27

Table 6.4 : Ablation study at 64×64 resolution. We report e_{CD} over all evaluation sequences. Chamfer distances are multiplied by 10^3 for readability.

Number of iterations per time step. To analyze the trade-off between iteration count and performance, we evaluate our method using different value of N (iterations per time step) ranging from 3 to 30. Figure 6.13 in supplementary plots the mean accuracy and runtime per frame curves at three resolutions across the tested values of N. We first observe that the runtime is quasi-linear with respect to the number of iterations. Meanwhile, increasing the number of iterations from 5 to 10 yields a significant accuracy gain. However, increasing to 20 iterations offers no further improvement while doubling the computational cost. The same trend is reflected in the e_{3D} error. See Table 6.7 in supplementary for the quantitative results of runtimes and Chamfer distances at each iteration count and resolution.

Table 6.8 reports the per-sequence accuracy for 5, 10, and 20 iterations per time step across three mesh resolutions. These results suggest that approximately 10 iterations are sufficient for the learned optimizer to converge across all tested mesh resolutions. We therefore adopt $N = 10$ as the default to balance accuracy and efficiency. Runtimes are similar across resolutions, indicating that FNOpt maintains efficient inference across resolutions.

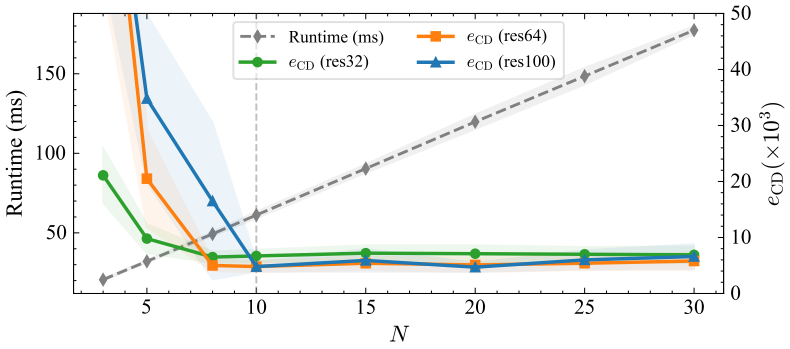


FIGURE 6.13 : FNOpt runtime (dashed gray line, left axis) and Chamfer distance e_{CD} (solid colored lines, right axis) as functions of the number of iterations per time step. Green, orange, and blue curves correspond to mesh resolutions of 32×32, 64×64, and 100×100, respectively. Shaded bands denote one standard deviation over test sequences.

Repulsive loss To alleviate self-collision, we applied a repulsive loss that penalizes non-adjacent cloth vertices when they are in close proximity. To assess its effectiveness, we consider two challenging scenarios with severe self-collision: a) Corner-center handle placement with motion sequence `rot_h1`, and b) Single Mid-Edge with motion sequence `xy_v2`. Table 6.5 reports the effect of the repulsive loss: the left column shows the average repulsive loss over the sequence; the middle column reports

	Corner-Center			Single Mid-Edge		
	$\mathcal{L}_{\text{rep}}(\times 10^3) \downarrow$	% (0.1) ↓	% (0.02) ↓	$\mathcal{L}_{\text{rep}}(\times 10^3) \downarrow$	% (0.1) ↓	% (0.02) ↓
with $\mathcal{L}_{\text{rep}}$	13.12 ± 4.37	0.00	0.07	0.11±0.26	0.00	0.00
no $\mathcal{L}_{\text{rep}}$	598.53 ± 141.29	97.95	98.08	508.79 ± 182.38	97.60	97.74

TABLE 6.5 : Ablation study on the repulsive loss. We report average repulsive loss (scaled by 10^3 for readability) and the percentage of frames with loss exceeding thresholds (0.02, 0.1).

the proportions of frames with repulsive loss higher than 0.1, and the right column uses a stricter threshold of 0.02. Figure 6.14 further provides the framewise loss curves together with representative qualitative results, illustrating that the repulsive loss substantially reduces self-collision.

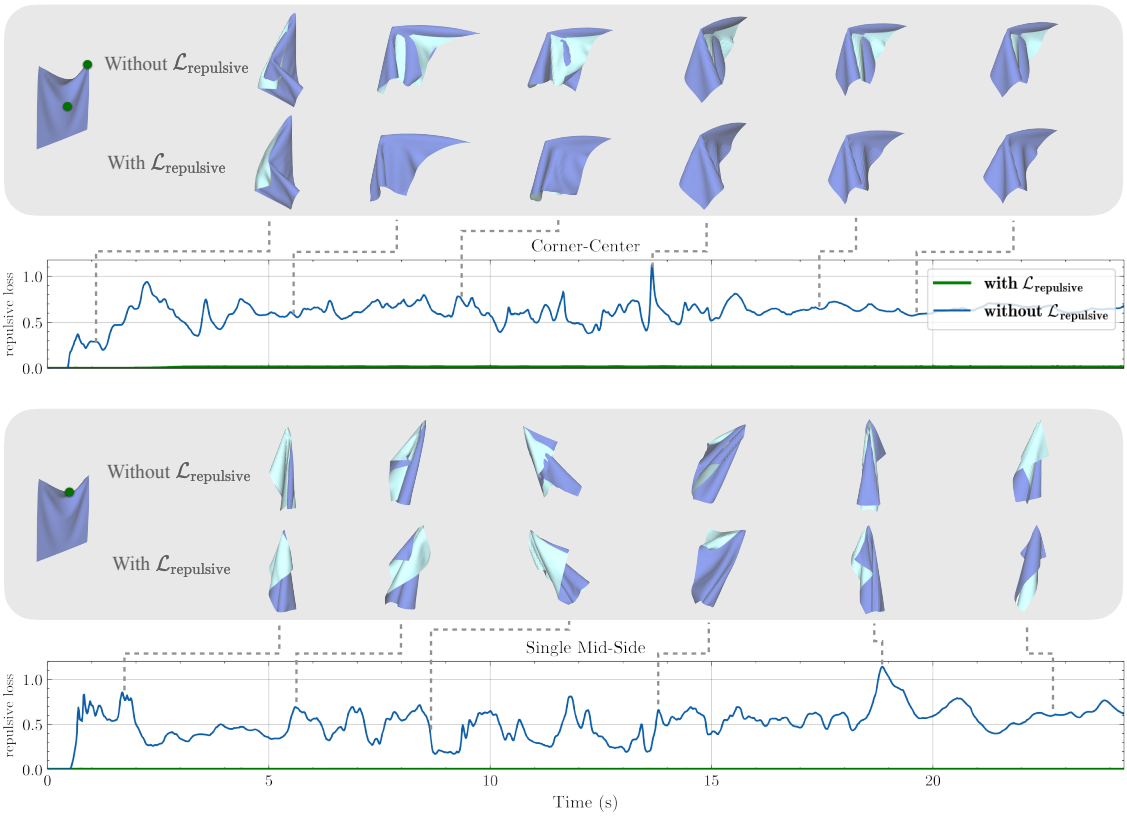


FIGURE 6.14 : Framewise repulsive loss with and without $\mathcal{L}_{\text{rep}}$. A subset of frames is shown for visual comparison: results without $\mathcal{L}_{\text{rep}}$ (left) versus with $\mathcal{L}_{\text{rep}}$ (right).

6.4 Simulation under Varying Material Coefficients

FNOpt can simulate cloth with different stretching, shearing and bending coefficients. We visualize the results under varying stretching and bending coefficients in Figure 6.15. The simulations are performed at 100×100 resolution with 10 iterations per time step.

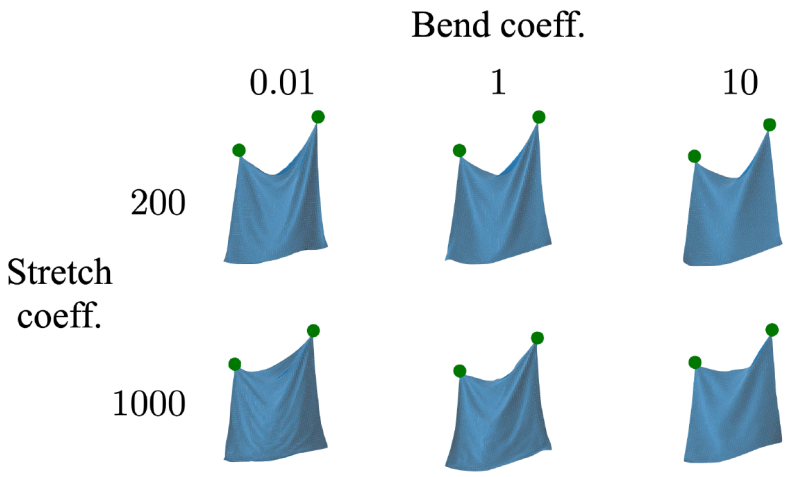


Figure 6.15 : Simulation results of selected frame with different stretching and bending coefficients.

Limitations While FNOpt scales across mesh resolutions, sizes and handle configurations, the current implementation is designed for rectangular grids. It is therefore not intended to operate directly on irregular or unstructured meshes. Additionally, the FNO hyperparameters in our implementation were selected manually without systematic optimization.

6.5 Conclusion

We presented FNOpt, a novel resolution-agnostic cloth simulation framework that learns an optimizer from FNOs to simulate cloth dynamics. Extensive experiments demonstrate that FNOpt achieves stable, accurate and efficient cloth dynamics in comparison with previous cloth simulators. Moreover, by leveraging the discretization-invariant capabilities of neural operators, it exhibits significant generalizability across a wide range of mesh resolutions and boundary conditions.

Sequence name	GD (lr = 0.01)				Adam (lr = 0.1)				L-BFGS (step size = 1)				Ours
	10	100	200	500	10	100	200	500	10	20	50	100	10
xy_v2	390.2	492.8	313.3	463.0	479.1	338.1	96.8	9.4	405.1	580.2	120.2	3.1	5.2
xy_v2_opp	389.4	499.6	316.3	466.8	480.0	320.4	115.6	7.7	400.3	579.5	119.3	3.2	5.3
yz_v2	491.4	434.4	317.0	493.2	550.8	314.7	125.8	11.7	380.1	443.0	99.0	2.3	3.4
yz_v2_opp	130.0	247.5	213.0	401.5	454.7	282.6	105.5	8.0	270.0	438.3	89.7	2.4	3.2
xz_v2	282.9	417.9	364.0	475.9	527.1	224.8	5.8	6.8	435.0	553.7	126.0	2.5	6.6
xyz_v2	540.6	507.0	352.6	568.5	561.7	377.6	101.4	13.9	450.1	511.1	150.6	1.9	4.9
xyz_v2_opp	248.7	366.9	309.1	435.1	453.7	304.1	116.5	10.6	350.6	500.5	140.8	2.5	4.1
xyz_v3	251.2	367.6	309.3	433.8	448.4	327.7	98.6	5.1	350.8	501.3	134.9	2.5	4.7
xyz_v3_opp	535.0	505.7	353.0	566.4	585.4	355.0	107.6	5.4	430.2	505.5	147.7	2.8	4.8
xyz_v4	536.0	504.7	354.8	566.9	537.7	291.8	91.1	5.2	424.2	508.1	145.2	2.0	3.7
xyz_v4_opp	251.3	367.2	313.3	433.9	446.6	327.7	81.3	5.4	363.1	499.8	132.0	3.1	4.2
rot_h0	180.5	483.9	341.7	247.7	473.6	116.6	46.9	6.4	225.4	204.3	47.8	1.6	5.2
rot_h0_opp	193.2	486.5	344.0	250.3	471.5	147.0	51.6	5.9	228.5	206.3	46.8	1.6	6.6
rot_h1	180.6	481.4	340.2	248.7	463.2	91.8	42.8	8.2	223.4	186.1	47.1	1.7	4.5
rot_h1_opp	192.0	485.2	342.6	246.7	482.2	103.7	29.8	5.4	235.7	199.3	46.2	1.7	5.7
Avg	319.5	443.1	325.6	419.9	494.4	261.5	81.1	7.7	344.8	427.8	106.2	2.3	4.8
σ	±142.4	±73.3	±35.2	±114.4	±44.0	±95.0	±35.2	±2.6	±82.1	±143.1	±39.2	±0.5	±1.0

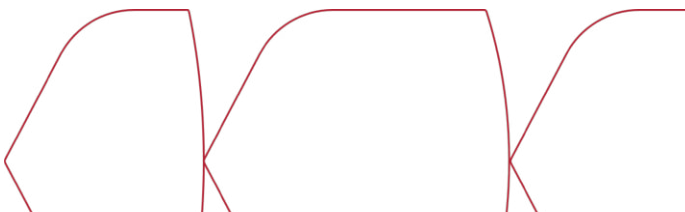
TABLE 6.6 : Ablation study on a 64×64 mesh comparing e_{CD} ($\times 10^3$; lower is better) for different optimizers (GD, Adam, L-BFGS and ours) under varying *numbers of iterations* per time-step.

N	Runtime (ms)			$e_{\text{CD}} \downarrow$		
	res32	res64	res100	res32	res64	res100
3	20.6 ±0.7	20.6 ±1.1	21.9 ±0.9	21.1 ±4.9	93.7 ±38.9	93.1 ±35.9
5	32.1 ±0.7	32.2 ±1.1	32.4 ±0.4	9.8 ±2.6	20.5 ±8.7	34.8 ±15.0
8	49.3 ±1.3	52.3 ±2.2	50.8 ±4.9	6.5 ±1.0	5.0 ±1.0	16.5 ±14.0
10	61.2 ±2.1	62.3 ±2.3	61.7 ±1.3	6.7 ±1.1	4.8 ±1.0	4.7 ±0.9
15	90.4 ±2.7	91.7 ±3.2	90.8 ±1.3	7.2 ±1.4	5.4 ±1.2	5.9 ±2.0
20	119.8 ±4.5	119.3 ±2.2	120.4 ±3.6	7.1 ±1.3	5.1 ±1.3	4.7 ±0.8
25	148.5 ±5.2	147.3 ±1.8	149.6 ±1.8	7.0 ±1.3	5.4 ±1.2	6.0 ±1.7
30	177.4 ±2.5	177.5 ±3.5	178.7 ±3.6	6.9 ±1.5	5.8 ±1.6	6.6 ±2.2

TABLE 6.7 : Ablation study with $N \in \{3, 5, 8, 10, 15, 20, 25, 30\}$ iterations per time-step across three mesh resolutions. We report the per-frame runtime (ms) and e_{CD} ($\times 10^3$).

Sequence name	32×32			64×64			100×100		
	5 iters	10 iters	20 iters	5 iters	10 iters	20 iters	5 iters	10 iters	20 iters
xy_v2	6.2 (16.1)	7.0 (15.2)	7.6 (15.5)	19.2 (24.4)	5.2 (12.7)	5.2 (12.4)	64.5 (46.5)	4.2 (11.0)	4.8 (10.5)
xy_v2_opp	5.9 (14.7)	6.5 (14.5)	7.7 (14.9)	24.7 (24.0)	5.3 (12.0)	5.6 (12.1)	32.8 (30.5)	4.8 (10.5)	5.5 (10.9)
yz_v2	11.3 (21.2)	4.9 (15.0)	4.0 (13.9)	8.3 (17.6)	3.4 (11.3)	3.3 (10.8)	19.9 (24.3)	5.5 (13.1)	3.4 (10.8)
yz_v2_opp	11.0 (19.4)	4.3 (13.5)	5.2 (13.8)	6.4 (15.1)	3.2 (10.3)	2.9 (09.8)	38.7 (27.7)	3.6 (10.9)	3.6 (09.5)
xz_v2	16.6 (21.6)	8.9 (15.2)	8.6 (14.9)	21.3 (22.7)	6.6 (13.3)	6.1 (10.7)	24.3 (23.6)	3.9 (9.9)	5.4 (10.3)
xyz_v2	10.0 (18.5)	6.4 (14.6)	6.4 (14.1)	14.0 (20.5)	4.9 (12.0)	4.3 (10.5)	44.4 (31.0)	4.3 (10.5)	4.0 (8.6)
xyz_v2_opp	9.1 (17.7)	6.4 (14.7)	5.3 (13.1)	16.5 (18.6)	4.1 (10.8)	4.5 (12.7)	27.4 (27.0)	5.2 (11.4)	4.4 (10.9)
xyz_v3	12.0 (19.4)	6.4 (15.2)	5.7 (13.2)	27.6 (24.4)	4.7 (12.2)	4.4 (12.1)	41.1 (31.8)	3.7 (10.6)	4.3 (10.7)
xyz_v3_opp	12.1 (19.6)	6.1 (14.3)	6.9 (14.8)	18.8 (21.3)	4.8 (11.8)	4.5 (10.7)	27.4 (23.8)	4.6 (10.9)	5.0 (10.0)
xyz_v4	6.5 (15.8)	6.7 (14.4)	8.3 (15.5)	21.4 (22.7)	3.7 (10.7)	4.8 (11.9)	29.5 (29.4)	3.5 (9.9)	4.0 (10.0)
xyz_v4_opp	8.7 (17.5)	7.4 (15.2)	7.2 (14.7)	25.3 (22.9)	4.2 (11.2)	4.0 (10.9)	31.1 (31.8)	4.7 (11.2)	4.1 (10.5)
rot_h0	9.1 (20.4)	6.9 (16.1)	7.7 (16.8)	14.3 (22.1)	5.2 (14.3)	7.5 (15.8)	18.3 (26.3)	6.2 (15.1)	4.6 (12.1)
rot_h0_opp	8.5 (19.6)	7.6 (17.1)	9.5 (18.3)	33.3 (22.3)	6.6 (15.8)	6.8 (14.2)	16.7 (25.0)	5.9 (14.8)	5.8 (13.3)
rot_h1	10.2 (21.0)	7.6 (16.8)	8.4 (17.5)	40.8 (31.1)	4.5 (13.9)	6.2 (14.5)	70.4 (54.3)	6.0 (14.6)	5.1 (13.2)
rot_h1_opp	9.2 (21.1)	7.5 (17.0)	8.7 (17.9)	16.3 (25.9)	5.7 (15.8)	7.1 (15.7)	36.1 (34.4)	6.2 (14.7)	6.0 (13.5)
Avg	9.8 (0.189)	6.7 (0.153)	7.2 (0.153)	20.5 (0.224)	4.8 (0.126)	5.1 (0.124)	34.8 (0.312)	4.7 (0.119)	4.7 (0.110)
σ	±2.6 (±0.021)	±1.1 (±0.010)	±1.5 (±0.016)	±8.7 (±0.036)	±1.0 (±0.017)	±1.3 (±0.019)	±15.0 (±0.083)	±0.9 (±0.019)	±0.8 (±0.014)

TABLE 6.8 : Per-sequence e_{CD} ($\times 10^3$) and e_{3D} error ($\times 10^2$; values in parentheses) at 5, 10, and 20 iteration budgets, evaluated on three mesh resolutions.



CENTRALE LYON

LIRIS

CHAPTER 7

Conclusion

7.1 Summary of contributions

This thesis has examined the problem of efficient and physically plausible digital garment simulation from three complementary perspectives: enforcing continuous physical constraints on discrete surfaces, developing compact and differentiable surface representations, and learning resolution-agnostic dynamics operators. Together, these contributions aim to bridge the inherent gap between the continuous nature of cloth mechanics and the discrete structures used in digital modeling and learning systems.

We first introduced GAPS, a geometry-aware and physics-based self-supervised draping framework that extends existing neural garment simulators with differential geometric constraints. By enforcing inextensibility through locally computed covariance-based measures, GAPS preserves local distances and areas while relaxing constraints only when collisions require it. This yields stable and realistic draping without the heavy post-processing typically used to resolve body–cloth intersections. Furthermore, the proposed RBF-based garment skinning enhances robustness across garment types, particularly for loose garments such as skirts and dresses, where conventional skinning strategies struggle.

Building on the observation that many challenges in garment simulation arise from the discrete and irregular nature of mesh representations, we next proposed PolyFit, a continuous and differentiable patch-based surface representation. PolyFit models local geometry using n-jet functions, providing explicit access to first- and higher-order derivatives while substantially reducing the dimensionality of the surface representation. This continuous formulation enables a variety of downstream applications. In PolySfT, it allows efficient monocular surface reconstruction. In OneFit,

it enables garment-agnostic deformation learning directly in the jet-function space rather than predicting per-vertex displacements.

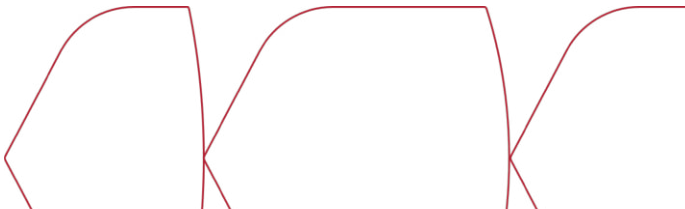
Finally, we explored the learning of simulation dynamics in a resolution-agnostic manner through FNOpt. Rather than predicting cloth states directly on a fixed discretization, FNOpt learns an optimization operator in the spectral domain using Fourier Neural Operators. Although losses are computed on discretized meshes, the learned operator acts in function space, allowing simulations trained on coarse meshes to generalize to finer ones without retraining. This formulation brings neural simulation closer to continuous PDE-based reasoning and offers a principled pathway to overcoming discretization-dependent behavior.

Taken together, these contributions illustrate a coherent strategy toward closing the discretization gap in garment simulation: *(i)* enforcing continuous physical constraints on discrete surfaces, *(ii)* representing surfaces in compact and analytically differentiable forms, and *(iii)* learning operators that act in continuous function spaces rather than discrete mesh domains. By addressing these three aspects, this thesis advances the robustness, fidelity, and generalization capability of neural garment modeling, and lays the foundation for future work on resolution-invariant simulation methods.

7.2 Future perspectives

While the works presented in this thesis have addressed specific challenges in deformation of cloth, the core principles, namely geometric constraints, compact representations, and operator learning, hold potential well beyond the scope of this dissertation. We envision several broad avenues for future exploration.

Generalization to Other Deformable Domains. Although the experiments in this thesis are conducted in the context of garment simulation, the underlying methods are applicable to a broader class of deformable systems. For instance, these techniques could be adapted to biomedical simulation, such as modeling skin, muscle, or organ deformation during surgical planning, as well as to soft robotics, where accurate real-time prediction of deformable body dynamics is essential for control. More broadly, inflatable structures and other thin-shell systems in engineering share similar mechanical properties with cloth, such as large deformations, contact, and near-inextensibility, making them natural candidates for the approaches proposed in



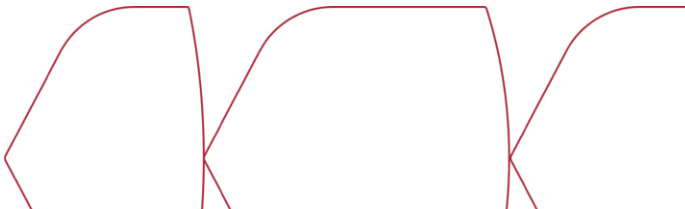
this thesis.

Bridging Forward and Inverse Problems. The methods developed in this thesis have been primarily demonstrated on the forward problem of deformation simulation. However, their differentiable formulations naturally lend themselves to inverse problems. In PolySfT, we showed that the PolyFit representation can be leveraged for monocular 3D surface reconstruction by optimizing jet coefficients to match image observations. A promising direction would be to extend this principle using the learned dynamics of FNOpt: by coupling a differentiable, resolution-agnostic forward simulator with rendering or sensing models, one could jointly estimate material parameters, external forces, and boundary conditions from visual observations alone. Such a pipeline would enable applications in robotic manipulation, where a robot needs to infer the physical state and material properties of a deformable object to plan grasping and folding actions, as well as in visual servoing, where real-time deformation predictions guide closed-loop control of the robot's interaction with soft objects.

Scalability and Real-World Deployment. Several practical challenges remain before neural cloth simulation can be deployed at scale in production environments. First, all methods presented in this thesis operate on single-layer garments, whereas real-world clothing often involves multiple superimposed layers with complex inter-layer frictional coupling and sliding Santesteban et al. [2022a], Grigorev et al. [2024]. Extending the framework to multi-layer settings will require collision models capable of handling simultaneous contacts between garment layers. Second, the proposed simulation methods rely on predefined skinning weights to ensure stable garment initialization. Although GAPS improves skinning robustness for loose garments, this dependence inherently constrains expressiveness for garments whose motion deviates significantly from body-driven kinematics; removing skinning entirely, as attempted in Grigorev et al. [2023], Li et al. [2024a], often leads to instability. Third, the spectral formulation of FNOpt currently assumes grid-structured inputs, precluding direct application to irregular garment meshes. Exploring neural operators designed for arbitrary geometries Li et al. [2023b] would extend resolution-agnostic learning to unstructured domains.

We believe that the contributions proposed in this thesis offer a promising step toward simulation systems that are not only accurate and efficient, but also general-

purpose and resolution-agnostic. We hope this work will encourage continued efforts toward closing the gap between continuous physical reality and discrete computational models.



References

Noam Aigerman, Kunal Gupta, Vladimir G Kim, Siddhartha Chaudhuri, Jun Saito, and Thibault Groueix. Neural jacobian fields: Learning intrinsic mappings of arbitrary meshes. In *SIGGRAPH*, 2022.

Thiemo Alldieck, Marcus Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll. Video based reconstruction of 3d people models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8387–8397, 2018.

Thiemo Alldieck, Marcus Magnor, Bharat Lal Bhatnagar, Christian Theobalt, and Gerard Pons-Moll. Learning to reconstruct people in clothing from a single rgb camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1175–1186, 2019a.

Thiemo Alldieck, Gerard Pons-Moll, Christian Theobalt, and Marcus Magnor. Tex2shape: Detailed full human body geometry from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2293–2303, 2019b.

Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. In *ACM Siggraph 2005 Papers*, pages 408–416. 2005.

Matthieu Armando, Laurence Boissieux, Edmond Boyer, Jean-Sébastien Franco, Martin Humenberger, Christophe Legras, Vincent Leroy, Mathieu Marsot, Julien Pansiot, Sergi Pujades, et al. 4dhumanoutfit: a multi-subject 4d dataset of human motion sequences in varying outfits exhibiting large displacements. *Computer Vision and Image Understanding*, 237:103836, 2023.

David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, page 43–54, 1998.

Adrien Bartoli, Yan Gérard, Francois Chadebecq, Toby Collins, and Daniel Pizarro. Shape-from-template. *IEEE TPAMI*, 37(10):2099–2118, 2015.

Jan Bednarik, Shaifali Parashar, Erhan Gundogdu, Mathieu Salzmann, and Pascal Fua. Shape reconstruction by learning differentiable surface representations. In *CVPR*, 2020.

Jan Bednarik, Vladimir G. Kim, Siddhartha Chaudhuri, Shaifali Parashar, Mathieu Salzmann, Pascal Fua, and Noam Aigerman. Temporally-Coherent Surface Reconstruction via Metric-Consistent Atlases. In *ICCV*, 2021.

Yizhak Ben-Shabat and Stephen Gould. Deepfit: 3d surface fitting via neural network weighted least squares. In *ECCV*, 2020.

Jan Bender and Daniel Bayer. Parallel Simulation of Inextensible Cloth. In *Workshop in Virtual Reality Interactions and Physical Simulation*, 2008.

Hugo Bertiche, Meysam Madadi, and Sergio Escalera. Cloth3d: Clothed 3d humans. In *ECCV*, 2020.

Hugo Bertiche, Meysam Madadi, and Sergio Escalera. Pbns: Physically based neural simulation for unsupervised garment pose space deformation. *ACM TOG*, 40(6), 2021.

Hugo Bertiche, Meysam Madadi, and Sergio Escalera. Neural cloth simulation. *ACM TOG*, 41(6), 2022.

Bharat Lal Bhatnagar, Garvita Tiwari, Christian Theobalt, and Gerard Pons-Moll. Multi-garment net: Learning to dress 3d people from images. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE, Oct 2019.

Michael J Black, Priyanka Patel, Joachim Tesch, and Jinlong Yang. Bedlam: A synthetic dataset of bodies exhibiting detailed lifelike animated motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8726–8737, 2023.

Boris Bonev, Thorsten Kurth, Christian Hundt, Jaideep Pathak, Maximilian Baust, Karthik Kashinath, and Anima Anandkumar. Spherical fourier neural operators: Learning stable dynamics on the sphere. In *International conference on machine learning*, pages 2806–2823. PMLR, 2023.

F.L. Bookstein. Principal warps: thin-plate splines and the decomposition of deformations. *IEEE TPAMI*, 11(6):567–585, 1989.

Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, and Bruno Lévy. *Polygon Mesh Processing*. CRC Press, 2010. ISBN 978-1568814261.

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. Projective dynamics: Fusing constraint projections for fast simulation. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 787–797. 2023.

Johannes Brandstetter, Daniel Worrall, and Max Welling. Message passing neural pde solvers. *arXiv preprint arXiv:2202.03376*, 2022.

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

David E Breen, Donald H House, and Michael J Wozny. Predicting the drape of woven cloth using interacting particles. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 365–372, 1994.

Romain Brégier, Guénolé Fiche, Laura Bravo-Sánchez, Thomas Lucas, Matthieu Armando, Philippe Weinzaepfel, Grégory Rogez, and Fabien Baradel. Human mesh modeling for anny body. *arXiv preprint arXiv:2511.03589*, 2025.

Robert Bridson, Ronald Fedkiw, and John Anderson. Robust treatment of collisions, contact and friction for cloth animation. *SIGGRAPH*, pages 594–663, 2002.

Robert Bridson, Sebastian Marino, and Ronald Fedkiw. Simulation of clothing with folds and wrinkles. In *ACM SIGGRAPH 2005 Courses*, pages 3–es. 2005.

Michel Carignan, Ying Yang, Nadia Magnenat Thalmann, and Daniel Thalmann. Dressing animated synthetic actors with complex deformable clothes. In *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '92, page 99–104, New York, NY, USA, 1992. Association for Computing Machinery. ISBN 0897914791. doi: 10.1145/133994.134017. URL https://doi.org/10.1145/133994.134017.

David Casillas-Perez, Daniel Pizarro, David Fuentes-Jimenez, Manuel Mazo, and Adrien Bartoli. Equiareal shape-from-template. *Journal of Mathematical Imaging and Vision*, 61(5):607–626, 2019.

David Casillas-Perez, Daniel Pizarro, David Fuentes-Jimenez, Manuel Mazo, and Adrien Bartoli. The isowarp: the template-based visual geometry of isometric surfaces. *International Journal of Computer Vision*, 129(7):2194–2222, 2021.

Augustin Cauchy. Méthode générale pour la résolution des systèmes d'équations simultanées. *Comp. Rend. Sci. Paris*, 25:536–538, 1847.

F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design*, 22(2):121–146, 2005.

Ruochen Chen, Liming Chen, and Shaifali Parashar. GAPS: Geometry-aware, physics-based, self-supervised neural garment draping. In *International Conference on 3D Vision (3DV)*, 2024.

Ruochen Chen, Thuy Tran, and Shaifali Parashar. Patch-based representation and learning for efficient deformation modeling. In *International Conference on 3D Vision (3DV)*, 2026.

Xu Chen, Yufeng Zheng, Michael J Black, Otmar Hilliges, and Andreas Geiger. Snarf: Differentiable forward skinning for animating non-rigid neural implicit shapes. In *International Conference on Computer Vision (ICCV)*, 2021.

Xu Chen, Tianjian Jiang, Jie Song, Jinlong Yang, Michael J Black, Andreas Geiger, and Otmar Hilliges. gdna: Towards generative detailed neural avatars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20427–20437, 2022.

Ruochen Chen*, Thuy Tran*, and Shaifali Parashar. FNOpt: Resolution-agnostic, self-supervised cloth simulation using meta-optimization with fourier neural operators. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2026.

Ajad Chhatkuli, Daniel Pizarro, Adrien Bartoli, and Toby Collins. A stable analytical framework for isometric shape-from-template by surface integration. *IEEE TPAMI*, 39(5):833–850, 2016.

Abouzar Choubineh, Jie Chen, David A. Wood, Frans Coenen, and Fei Ma. Fourier neural operator for fluid flow in small-shape 2d simulated porous media dataset.

Algorithms, 16(1), 2023. ISSN 1999-4893. doi: 10.3390/a16010024. URL https://www.mdpi.com/1999-4893/16/1/24.

Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016.

Gabriel Cirio, Jorge Lopez-Moreno, David Miraut, and Miguel A. Otaduy. Yarn-level simulation of woven cloth. *ACM TOG*, 33(6), 2014.

CLO Virtual Fashion. Marvelous Designer. https://www.marvelousdesigner.com, 2018.

Enric Corona, Albert Pumarola, Guillem Alenyà, Gerard Pons-Moll, and Francesc Moreno-Noguer. Smplicit: Topology-aware generative model for clothed people. In *CVPR*, 2021.

Sagnik Das, Ke Ma, Zhixin Shu, and Dimitris Samaras. Learning an isometric surface parameterization for texture unwrapping. In *ECCV*, 2022.

Luca De Luigi, Ren Li, Benoit Guillard, Mathieu Salzmann, and Pascal Fua. DrapeNet: Garment Generation and Self-Supervised Draping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Zhantao Deng, Jan Bednařík, Mathieu Salzmann, and Pascal Fua. Better patch stitching for parametric surface reconstruction. In *3DV*, 2020.

Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. Learning elementary structures for 3d shape generation and matching. In *Advances in Neural Information Processing Systems*, pages 7433–7443, 2019.

Nicolas Donati, Abhishek Sharma, and Maks Ovsjanikov. Deep Geometric Functional Maps: Robust Feature Learning for Shape Correspondence. In *CVPR*, 2020.

Jeffrey W Eischen, Shigan Deng, and Timothy G Clapp. Finite-element modeling and control of flexible fabric parts. *IEEE Computer Graphics and Applications*, 16 (5):71–80, 1996.

Olaf Etzmuß, Michael Keckeisen, and Wolfgang Straßer. A fast finite element solution for cloth modelling. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, pages 244–251. IEEE, 2003.

Gerald E Farin. *Curves and surfaces for CAGD: a practical guide*. Morgan Kaufmann, 2002.

David Fuentes-Jimenez, Daniel Pizarro, David Casillas-Perez, Toby Collins, and Adrien Bartoli. Texture-generic deep shape-from-template. *IEEE Access*, 9:75211–75230, 2021. doi: 10.1109/ACCESS.2021.3082011.

David Fuentes-Jimenez, Daniel Pizarro, David Casillas-Pérez, Toby Collins, and Adrien Bartoli. Deep shape-from-template: Single-image quasi-isometric deformable registration and reconstruction. *Image and Vision Computing*, 127:104531, 2022. ISSN 0262-8856. doi: https://doi.org/10.1016/j.imavis.2022.104531. URL https://www.sciencedirect.com/science/article/pii/S0262885622001603.

L Gan, NG Ly, and GP Steven. A study of fabric deformation using nonlinear finite elements. *Textile research journal*, 65(11):660–668, 1995.

Lin Gao, Jie Yang, Bo-Tao Zhang, Jia-Mu Sun, Yu-Jie Yuan, Hongbo Fu, and Yu-Kun Lai. Mesh-based gaussian splatting for real-time large-scale deformation. *arXiv preprint arXiv:2402.04796*, 2024.

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

Artur Grigorev, Bernhard Thomaszewski, Michael J Black, and Otmar Hilliges. HOOD: Hierarchical graphs for generalized modelling of clothing dynamics. In *CVPR*, 2023.

Artur Grigorev, Giorgio Becherini, Michael Black, Otmar Hilliges, and Bernhard Thomaszewski. Contourcraft: Learning to resolve intersections in neural multi-garment simulations. In *ACM SIGGRAPH 2024 conference papers*, pages 1–10, 2024.

Eitan Grinspun, Anil N. Hirani, Mathieu Desbrun, and Peter Schröder. Discrete shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium*

on Computer Animation, SCA '03, page 62–67, Goslar, DEU, 2003. Eurographics Association. ISBN 1581136595.

Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *CVPR*, 2018.

Peng Guan, Loretta Reiss, David A. Hirshberg, Alexander Weiss, and Michael J. Black. Drape: Dressing any person. *ACM TOG*, 31(4), 2012.

Benoit Guillard, Federico Stella, and Pascal Fua. Meshudf: Fast and differentiable meshing of unsigned distance field networks. In *European Conference on Computer Vision*, 2022.

Erhan Gundogdu, Victor Constantin, Shaifali Parashar, Amrollah Seifoddini, Minh Dang, Mathieu Salzmann, and Pascal Fua. Garnet++: Improving Fast and Accurate Static 3D Cloth Draping by Curvature Loss. *IEEE TPAMI*, 44(1):181–195, 2020.

Zhongkai Hao, Chengyang Ying, Zhengyi Wang, Hang Su, Yinpeng Dong, Songming Liu, Ze Cheng, Jun Zhu, and Jian Song. Gnot: A general neural operator transformer for operator learning. *ArXiv*, abs/2302.14376, 2023. URL https://api.semanticscholar.org/CorpusID:257232579.

Nazim Haouchine and Stephane Cotin. Template-based monocular 3d recovery of elastic shapes using lagrangian multipliers. In *CVPR*, 2017.

Mohamed Hassan, Vasileios Choutas, Dimitrios Tzionas, and Michael J Black. Resolving 3d human pose ambiguities with 3d scene constraints. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2282–2292, 2019.

Zhu Heming, Cao Yu, Jin Hang, Chen Weikai, Du Dong, Wang Zhangye, Cui Shuguang, and Han Xiaoguang. Deep fashion3d: A dataset and benchmark for 3d garment reconstruction from single images. In *Computer Vision – ECCV 2020*, pages 512–530. Springer International Publishing, 2020. ISBN 978-3-030-58452-8.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, page 2017–2025, Cambridge, MA, USA, 2015. MIT Press.

Navami Kairanda, Edith Tretschk, Mohamed Elgharib, Christian Theobalt, and Vladislav Golyanik. ϕ-sft: Shape-from-template with a physics-based deformation model. In *CVPR*, 2022.

Navami Kairanda, Marc Habermann, Shanthika Naik, Christian Theobalt, and Vladislav Golyanik. Thin-shell-sft: Fine-grained monocular non-rigid 3d surface tracking with neural deformation fields. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 11373–11383, 2025.

Jonathan M. Kaldor, Doug L. James, and Steve Marschner. Simulating knitted cloth at the yarn level. In *ACM SIGGRAPH 2008 Papers*, SIGGRAPH '08, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781450301121. doi: 10.1145/1399504.1360664. URL https://doi.org/10.1145/1399504.1360664.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42 (4):139–1, 2023.

Mohammad S. Khorrami, Pawan Goyal, Jaber R. Mianroodi, Bob Svendsen, Peter Benner, and Dierk Raabe. A physics-encoded fourier neural operator approach for surrogate modeling of divergence-free stress fields in solids, 2025. URL https://arxiv.org/abs/2408.15408.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL https://api.semanticscholar.org/CorpusID:6628106.

James T Klosowski, Martin Held, Joseph SB Mitchell, Henry Sowizral, and Karel Zikan. Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE transactions on Visualization and Computer Graphics*, 4(1):21–36, 1998.

Maria Korosteleva, Timur Levent Kesdogan, Fabian Kemper, Stephan Wenninger, Jasmin Koller, Yuhan Zhang, Mario Botsch, and Olga Sorkine-Hornung. GarmentCodeData: A dataset of 3D made-to-measure garments with sewing patterns. In *Computer Vision – ECCV 2024*, 2024.

Jean Kossaifi, Nikola Kovachki, Zongyi Li, David Pitt, Miguel Liu-Schiaffini, Robert Joseph George, Boris Bonev, Kamyar Azizzadenesheli, Julius Berner, and Anima Anandkumar. A library for learning neural operators, 2024.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

Thorsten Kurth, Shashank Subramanian, Peter Harrington, Jaideep Pathak, Morteza Mardani, David Hall, Andrea Miele, Karthik Kashinath, and Anima Anandkumar. Fourcastnet: Accelerating global high-resolution weather forecasting using adaptive fourier neural operators. In *Proceedings of the Platform for Advanced Scientific Computing Conference*, PASC '23, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701900. doi: 10.1145/3592979.3593412. URL https://doi.org/10.1145/3592979.3593412.

Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020.

Jose Lamarca, Shaifali Parashar, Adrien Bartoli, and JMM Montiel. Defslam: Tracking and mapping of deforming scenes from monocular sequences. *IEEE Transactions on robotics*, 37(1):291–303, 2020.

Dohae Lee, Hyun Kang, and In-Kwon Lee. Clothcombo: Modeling inter-cloth interaction for draping multi-layered clothes. *ACM Transactions on Graphics (TOG)*, 42:1 – 13, 2023. URL https://api.semanticscholar.org/CorpusID:258041126.

John M. Lee. *Introduction to Smooth Manifolds*. Springer, 2003. ISBN 0-387-95448-1.

Thomas Lewiner, Hélio Lopes, Antônio Wilson Vieira, and Geovan Tavares. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of graphics tools*, 8(2):1–15, 2003.

Peizhuo Li, Tuanfeng Y Wang, Timur Levent Kesdogan, Duygu Ceylan, and Olga Sorkine-Hornung. Neural garment dynamics via manifold-aware transformers. In *Computer Graphics Forum*, volume 43, page e15028. Wiley Online Library, 2024a.

Ren Li, Benoît Guillard, and Pascal Fua. Isp: Multi-layered garment draping with implicit sewing patterns. In *Advances in Neural Information Processing Systems*, 2023a.

Zhijie Li, Wenhui Peng, Zelong Yuan, and Jianchun Wang. Fourier neural operator approach to large eddy simulation of three-dimensional turbulence. *Theoretical and Applied Mechanics Letters*, 12(6):100389, 2022. ISSN 2095-0349. doi: https://doi.org/10.1016/j.taml.2022.100389. URL https://www.sciencedirect.com/science/article/pii/S2095034922000691.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations, 2021. URL https://arxiv.org/abs/2010.08895.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-informed neural operator for large-scale 3D PDEs. In *Advances in Neural Information Processing Systems (NeurIPS)*, December 2023b.

Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *ACM/IMS Journal of Data Science*, 1(3):1–27, 2024b.

Zhouyingcheng Liao, Sinan Wang, and Taku Komura. Senc: Handling self-collision in neural cloth simulation. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part IX*, page 385–402, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-3-031-72672-9. doi: 10.1007/978-3-031-72673-6_21. URL https://doi.org/10.1007/978-3-031-72673-6_21.

Emmanuel Ian Libao, Myeongjin Lee, Sumin Kim, and Sung-Hee Lee. Meshgraphnetrp: Improving generalization of gnn-based cloth simulation. In *Proceedings of the 16th ACM SIGGRAPH Conference on Motion, Interaction*

and Games, MIG '23, New York, NY, USA, 2023a. Association for Computing Machinery. ISBN 9798400703935. doi: 10.1145/3623264.3624441. URL https://doi.org/10.1145/3623264.3624441.

Emmanuel Ian Libao, Myeongjin Lee, Sumin Kim, and Sung-Hee Lee. Meshgraphnetrp: Improving generalization of gnn-based cloth simulation. In *Proceedings of the 16th ACM SIGGRAPH Conference on Motion, Interaction and Games*, MIG '23, New York, NY, USA, 2023b. Association for Computing Machinery. ISBN 9798400703935. doi: 10.1145/3623264.3624441. URL https://doi.org/10.1145/3623264.3624441.

Siyou Lin, Hongwen Zhang, Zerong Zheng, Ruizhi Shao, and Yebin Liu. Learning implicit templates for point-based clothed human modeling. In *ECCV*, 2022.

Tiantian Liu, Adam W. Bargteil, James F. O'Brien, and Ladislav Kavan. Fast simulation of mass-spring systems. *ACM TOG*, 32(6):1–7, 2013.

Xiaoxiao Long, Cheng Lin, Lingjie Liu, Yuan Liu, Peng Wang, Christian Theobalt, Taku Komura, and Wenping Wang. Neuraludf: Learning unsigned distance fields for multi-view reconstruction of surfaces with arbitrary topologies. In *CVPR*, 2023.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015.

Weng Fei Low and Gim Hee Lee. Minimal neural atlas: Parameterizing complex surfaces with minimal charts and distortion. In *European Conference on Computer Vision*, pages 465–481. Springer, 2022.

Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

Sara Luengo-Sanchez, David Fuentes-Jimenez, Cristina Losada-Gutierrez, Daniel Pizarro, and Adrien Bartoli. Weakly-supervised deep shape-from-template. *IEEE Access*, 13:22868–22892, 2025.

Zorah Lähner, Daniel Cremers, and Tony Tung. DeepWrinkles: Accurate and Realistic Clothing Modeling. In *ECCV*, 2018.

Q. Ma, J. Yang, M. J. Black, and S. Tang. Neural point-based shape modeling of humans in challenging clothing. In *2022 International Conference on 3D Vision (3DV)*, pages 679–689, Los Alamitos, CA, USA, sep 2022. IEEE Computer Society. doi: 10.1109/3DV57658.2022.00078. URL https://doi.ieeecomputersociety.org/10.1109/3DV57658.2022.00078.

Qianli Ma, Jinlong Yang, Anurag Ranjan, Sergi Pujades, Gerard Pons-Moll, Siyu Tang, and Michael J. Black. Learning to dress 3D people in generative clothing. In *Computer Vision and Pattern Recognition (CVPR)*, 2020.

Qianli Ma, Shunsuke Saito, Jinlong Yang, Siyu Tang, and Michael J. Black. SCALE: Modeling clothed humans with a surface codec of articulated local elements. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2021a.

Qianli Ma, Jinlong Yang, Siyu Tang, and Michael J. Black. The power of points for modeling humans in clothing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021b.

Miles Macklin, Matthias Müller, and Nuttapong Chentanez. XPBD: Position-Based Simulation of Compliant Constrained Dynamics. In *Proceedings of the 9th International Conference on Motion in Games*, 2016.

Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *ICCV*, 2019.

Abed Malti and Cédric Herzet. Elastic shape-from-template with spatially sparse deforming forces. In *CVPR*, 2017.

Abed Malti, Richard Hartley, Adrien Bartoli, and Jae-Hak Kim. Monocular template-based 3d reconstruction of extensible surfaces with local linear elasticity. In *CVPR*, 2013.

Abed Malti, Adrien Bartoli, and Richard Hartley. A linear least-squares solution to elastic shape-from-template. In *CVPR*, 2015.

Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus Gross. Example-based elastic materials. In *SIGGRAPH*, 2011.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *Journal of Visual Communication and Image Representation*, 18 (2):109–118, 2007.

Rahul Narain, Armin Samii, and James F. O'Brien. Adaptive anisotropic remeshing for cloth simulation. *ACM TOG*, 31(6):147:1–10, 2012.

Andrew Nealen, Matthias Mueller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically Based Deformable Models in Computer Graphics. *Comput. Graph. Forum*, 25(4), 2006.

Nvcloth. Nvcloth, 2018. https://docs.nvidia.com/gameworks/content/gameworkslibrary/physx/nvCloth/index.html.

NVIDIA Flex. NVIDIA Flex. https://developer.nvidia.com/flex, 2018.

Mohamed Omran, Christoph Lassner, Gerard Pons-Moll, Peter Gehler, and Bernt Schiele. Neural body fitting: Unifying deep learning and model based human pose and shape estimation. In *3DV*, pages 484–494. IEEE, 2018.

Optitex. Optitex. https://optitex.com/, 2018.

Ahmed AA Osman, Timo Bolkart, and Michael J Black. Star: Sparse trained articulated human body regressor. In *European Conference on Computer Vision*, pages 598–613. Springer, 2020.

Ahmed AA Osman, Timo Bolkart, Dimitrios Tzionas, and Michael J Black. SUPR: A sparse unified part-based human representation. In *European Conference on Computer Vision (ECCV)*, 2022.

Pablo Palafox, Aljaž Božič, Justus Thies, Matthias Nießner, and Angela Dai. Npms: Neural parametric models for 3d deformable shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12695–12705, 2021.

Xiaoyu Pan, Jiaming Mai, Xinwei Jiang, Dongxue Tang, Jingxiang Li, Tianjia Shao, Kun Zhou, Xiaogang Jin, and Dinesh Manocha. Predicting loose-fitting garment deformations using bone-driven motion networks. In *SIGGRAPH*, 2022.

Shaifali Parashar and Adrien Bartoli. 3dvfx: 3d video editing using non-rigid structure-from-motion. In *Eurographics*, 2019.

Shaifali Parashar, Daniel Pizarro, Adrien Bartoli, and Toby Collins. As-rigid-as-possible volumetric shape-from-template. In *ICCV*, 2015.

Shaifali Parashar, Daniel Pizarro, and Adrien Bartoli. Local deformable 3d reconstruction with cartan's connections. *IEEE TPAMI*, 42(12):3011–3026, 2020a.

Shaifali Parashar, Mathieu Salzmann, and Pascal Fua. Local Non-Rigid Structure-From-Motion from Diffeomorphic Mappings. In *CVPR*, 2020b.

Shaifali Parashar, Adrien Bartoli, and Daniel Pizarro. Robust Isometric Non-Rigid Structure-From-Motion. *IEEE TPAMI*, 44(10):6409–6423, 2021.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019.

Chaitanya Patel, Zhouyingcheng Liao, and Gerard Pons-Moll. TailorNet: Predicting Clothing in 3D as a Function of Human Pose, Shape and Garment Style. In *CVPR*, 2020.

Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, Pedram Hassanzadeh, Karthik Kashinath, and Animashree Anandkumar. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

Mark Pauly, Markus Gross, and P. Leif Kobbelt. Efficient simplification of point-sampled surfaces. In *IEEE Visualization*, 2002.

Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10975–10985, 2019.

Mathieu Perriollat, Richard Hartley, and Adrien Bartoli. Monocular Template-Based Reconstruction of Inextensible Surfaces. In *BMVC*, 2008.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *ICLR*, 2021.

L. Piegl. On nurbs: a survey. *IEEE Computer Graphics and Applications*, 11(1): 55–71, 1991.

Gerard Pons-Moll, Sergi Pujades, Sonny Hu, and Michael J Black. Clothcap: Seamless 4d clothing capture and retargeting. *ACM Transactions on Graphics (ToG)*, 36 (4):1–15, 2017.

Xavier Provot. Deformation constraints in a mass-spring model to describe rigid cloth behaviour. In *Graphics interface*, 1995.

Xavier Provot. Collision and self-collision handling in cloth model dedicated to design garments. In *Computer Animation and Simulation'97: Proceedings of the Eurographics Workshop in Budapest, Hungary, September 2–3, 1997*, pages 177–189. Springer, 1997.

Albert Pumarola, Jordi Sanchez-Riera, Gary Choi, Alberto Sanfeliu, and Francesc Moreno-Noguer. 3dpeople: Modeling the geometry of dressed humans. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2242–2251, 2019.

Abhinanda R Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J Black. Babel: Bodies, action and behavior with english labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 722–731, 2021.

Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *CVPR*, 2017a.

Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NeurIPS*, 2017b.

Elad Richardson, Matan Sela, and Ron Kimmel. 3d face reconstruction by learning from synthetic data. 2016.

Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *arXiv preprint arXiv:2201.02610*, 2022.

Shunsuke Saito, Jinlong Yang, Qianli Ma, and Michael J. Black. SCANimate: Weakly supervised learning of skinned clothed avatar networks. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2021a.

Shunsuke Saito, Jinlong Yang, Qianli Ma, and Michael J Black. Scanimate: Weakly supervised learning of skinned clothed avatar networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2886–2897, 2021b.

Mathieu Salzmann, Richard Hartley, and Pascal Fua. Convex optimization for deformable surface 3-d tracking. In *ICCV*, 2007.

Mathieu Salzmann, Francesc Moreno-Noguer, Vincent Lepetit, and Pascal Fua. Closed-form solution to non-rigid 3d surface registration. In *ECCV*, 2008.

Igor Santesteban, Miguel A. Otaduy, and Dan Casas. Learning-Based Animation of Clothing for Virtual Try-On. *Comput. Graph. Forum*, 38(2):355–366, 2019.

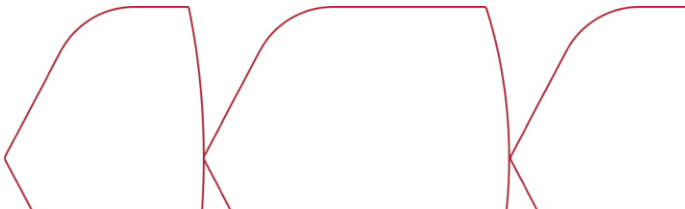
Igor Santesteban, Nils Thuerey, Miguel A Otaduy, and Dan Casas. Self-Supervised Collision Handling via Generative 3D Garment Models for Virtual Try-On. 2021.

Igor Santesteban, Miguel Otaduy, Nils Thuerey, and Dan Casas. Ulnef: Untangled layered neural fields for mix-and-match virtual try-on. In *NeurIPS*, 2022a.

Igor Santesteban, Miguel A Otaduy, and Dan Casas. SNUG: Self-Supervised Neural Dynamic Garments. In *CVPR*, 2022b.

Yidi Shao, Chen Change Loy, and Bo Dai. Towards multi-layered 3d garments animation. In *ICCV*, 2023.

Vikramjit Sidhu, Edgar Tretschk, Vladislav Golyanik, Antonio Agudo, and Christian Theobalt. Neural dense non-rigid structure from motion with latent space constraints. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI*, page 204–222, Berlin, Heidelberg, 2020. Springer-Verlag. ISBN 978-3-030-58516-7. doi: 10.1007/978-3-030-58517-4_13. URL https://doi.org/10.1007/978-3-030-58517-4_13.



Soltani, Huang, Wu, Kulkarni, and Tenenbaum. Synthesizing 3d shapes via modeling multi-view depth maps and silhouettes with deep generative networks. In *CVPR*, 2017.

Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, volume 4, pages 109–116, 2007.

David Stotko and Reinhard Klein. Saft: Shape and appearance of fabrics from template via differentiable physical simulations from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 27660–27670, 2025.

David Stotko, Nils Wandel, and Reinhard Klein. Physics-guided shape-from-template: Monocular video perception through neural surrogate models. In *CVPR*, 2024.

Richard Szeliski and David Tonnesen. Surface modeling with oriented particle systems. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 185–194, 1992.

Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 205–214, 1987.

Garvita Tiwari, Bharat Lal Bhatnagar, Tony Tung, and Gerard Pons-Moll. SIZER: A Dataset and Model for Parsing 3D Clothing and Learning Size Sensitive 3D Clothing. In *ECCV*, 2020.

Garvita Tiwari, Nikolaos Sarafianos, Tony Tung, and Gerard Pons-Moll. Neural-gif: Neural generalized implicit functions for animating people in clothing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11708–11718, 2021.

Lokender Tiwari and Brojeshwar Bhowmick. Garsim: Particle based neural garment simulator. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 4472–4481, January 2023.

Lokender Tiwari, Brojeshwar Bhowmick, and Sanjana Sinha. Gensim: Unsupervised generic garment simulator. In *Proceedings of the IEEE/CVF Conference on*

Computer Vision and Pattern Recognition (CVPR) Workshops, pages 4169–4178, June 2023.

Thuy Tran, Ruochen Chen, and Shaifali Parashar. Image-guided shape-from-template using mesh inextensibility constraints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Carsten Stoll, and Christian Theobalt. Patchnets: Patch-based generalizable deep implicit 3d shape representations. In *European Conference on Computer Vision*, pages 293–309. Springer, 2020.

Sébastien Valette and Jean-Marc Chassery. Approximated centroidal voronoi diagrams for uniform polygonal mesh coarsening. *Comput. Graph. Forum*, 23(3): 381–389, 2004.

Aydin Varol, Appu Shaji, Mathieu Salzmann, and Pascal Fua. Monocular 3d reconstruction of locally textured surfaces. *IEEE TPAMI*, 34(6):1118–1130, 2012.

Joanna Waczyńska, Piotr Borycki, Sławomir Tadeja, Jacek Tabor, and Przemysław Spurek. Games: Mesh-based adapting and modification of gaussian splatting. *arXiv preprint arXiv:2402.01459*, 2024.

Nils Wandel, Stefan Schulz, and Reinhard Klein. Metamizer: a versatile neural optimizer for fast and accurate physics simulations. In *International Conference on Learning Representations (ICLR)*, 2025.

Huamin Wang, James F O'Brien, and Ravi Ramamoorthi. Data-driven elastic models for cloth: modeling and measurement. *ACM transactions on graphics (TOG)*, 30 (4):1–12, 2011.

Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *NeurIPS*, 2021.

Tuanfeng Y. Wang, Tianjia Shao, Kai Fu, and Niloy J. Mitra. Learning an intrinsic garment space for interactive authoring of garment animation. *ACM TOG*, 38(6), 2019.

Wenbo Wang, Hsuan-I Ho, Chen Guo, Boxiang Rong, Artur Grigorev, Jie Song, Juan Jose Zarate, and Otmar Hilliges. 4d-dress: A 4d dataset of real-world human clothing with semantic annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 550–560, 2024.

Joe Warren and Henrik Weimer. *Subdivision methods for geometric design: A constructive approach*. Elsevier, 2001.

Gerald Wempner, Demosthenes Talaslidis, and J Petrolito. Mechanics of solids and shells: theories and approximations. *Appl. Mech. Rev.*, 56(5):B68–B68, 2003.

Gege Wen, Zongyi Li, Qirui Long, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally Benson. Real-time high-resolution co2 geological storage prediction using nested fourier neural operators. *Energy & Environmental Science*, 16, 01 2023. doi: 10.1039/D2EE04204E.

Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J Black. Icon: Implicit clothed humans obtained from normals. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13286–13296. IEEE, 2022.

Sirui Xu, Zhengyuan Li, Yu-Xiong Wang, and Liang-Yan Gui. Interdiff: Generating 3d human-object interactions with physics-informed diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14928–14940, 2023.

Lei Yang, Yongqing Liang, Xin Li, Congyi Zhang, Guying Lin, Alla Sheffer, Scott Schaefer, John Keyser, and Wenping Wang. Neural parametric surfaces for shape modeling, 2023.

Yan Yang, Angela F. Gao, Jorge C. Castellanos, Zachary E. Ross, Kamyar Azizzadenesheli, and Robert W. Clayton. Seismic wave propagation and inversion with neural operators. *The Seismic Record*, 1(3):126–134, 11 2021. ISSN 2694-4006. doi: 10.1785/0320210026. URL https://doi.org/10.1785/0320210026.

Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 206–215, 2018.

Rui Yu, Chris Russell, Neill DF Campbell, and Lourdes Agapito. Direct, dense, and deformable: Template-based non-rigid 3d reconstruction from rgb video. In *Proceedings of the IEEE international conference on computer vision*, pages 918–926, 2015.

Ilya Zakharkin, Kirill Mazur, Artur Grigorev, and Victor Lempitsky. Point-based modeling of human clothing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14718–14727, October 2021.

Chao Zhang, Sergi Pujades, Michael J Black, and Gerard Pons-Moll. Detailed, accurate, human shape estimation from clothed 3d scan sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4191–4200, 2017.

Hongwen Zhang, Yating Tian, Yuxiang Zhang, Mengcheng Li, Liang An, Zhenan Sun, and Yebin Liu. Pymaf-x: Towards well-aligned full-body model regression from monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):12287–12303, 2023.

Meng Zhang, Tuanfeng Wang, Duygu Ceylan, and Niloy J Mitra. Deep detail enhancement for any garment. In *Computer Graphics Forum*, volume 40, pages 399–411. Wiley Online Library, 2021a.

Meng Zhang, Tuanfeng Y. Wang, Duygu Ceylan, and Niloy J. Mitra. Dynamic neural garments. *ACM TOG*, 40(6), 2021b.

Tingtao Zhou, Xuan Wan, Daniel Zhengyu Huang, Zongyi Li, Zhiwei Peng, Anima Anandkumar, John F. Brady, Paul W. Sternberg, and Chiara Daraio. Ai-aided geometric design of anti-infection catheters. *Science Advances*, 10(1):eadj1741, 2024. doi: 10.1126/sciadv.adj1741. URL https://www.science.org/doi/abs/10.1126/sciadv.adj1741.

Yi Zhou, Connelly Barnes, Lu Jingwan, Yang Jimei, and Li Hao. On the continuity of rotation representations in neural networks. In *CVPR*, 2019.

Xingxing Zou, Xintong Han, and Waikeung Wong. Cloth4d: A dataset for clothed human reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12847–12857, 2023.