SARTORAI: MULTIMODAL GARMENT PATTERN GENERATION AND EDITING WITH TOKENIZED GEOMETRY

Anonymous authors
Paper under double-blind review



Figure 1: **SartorAI** is a two-stage framework that combines a VLM with a flow model to generate sewing patterns aligned with the target body, given text or image inputs.

ABSTRACT

Garments and their sewing patterns play an important role in fashion, gaming, film, and virtual reality. However, designing garments that can be sewn together still requires considerable expertise, making it difficult to meet the growing demand for personalized clothing. To bridge this gap, we propose **SartorAI**, a framework for garment generation and editing. Rather than representing garment panels as sets of parametric edges, we model their boundaries as arbitrary closed 2D curves and define stitching relationships as connections between boundary points. This representation avoids dependence on predefined panel-edge parameterizations and stitching schemes, which are commonly used in existing works. Our key insight is that panel shapes and their 3D layout provide strong cues about how the panels should be sewn together. We therefore separate panel geometry generation from stitching prediction, using a vision-language model (VLM) to generate panel geometry and a flow model to predict stitching conditioned on that geometry. To improve geometric precision, we encode panel geometry into discrete tokens using residual vector-quantized variational autoencoders (RQ-VAEs) and train the VLM to predict these tokens. This preserves the VLM's symbolic reasoning capabilities and interactivity while alleviating the precision limitations of autoregressive floating-point prediction. To train **SartorAI**, we construct a dataset of 100k garments, each draped over human bodies with diverse shapes and rendered with realistic PBR textures. Trained on this dataset, **SartorAI** achieves state-of-the-art performance in garment sewing pattern generation and editing.

1 INTRODUCTION

High-quality garment assets support a wide range of applications in fashion design, gaming, film, and virtual reality. Sewing patterns provide a practical representation of these garments by specifying 2D panel shapes and their stitching relationships, supporting both physical garment construction and digital cloth simulation. However, translating a desired design into a sewing pattern requires specialized knowledge and substantial manual effort. Natural language and reference images offer complementary, intuitive ways to communicate design intent: language expresses desired styles, requirements, and modifications, while images convey garment appearance and visual details. Generating sewing patterns from images or text and refining them through language could give both

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

individual users and professional designers a more accessible and efficient way to create and customize garments for virtual and physical use.

Recent approaches to sewing pattern generation broadly follow two paradigms. **Diffusion- and Flow-based** methods can synthesize detailed garment geometry Liu et al. (2025); Li et al. (2025); Nakayama et al. (2026), but lack semantic understanding of garment construction and interactivity. **VLM-based methods**, by contrast, adapt pretrained VLMs for multimodal generation and instruction-based interaction Bian et al. (2025); Nakayama et al. (2025); Pham et al. (2026). Despite these capabilities, accurately predicting the many continuous parameters of sewing patterns remains challenging, and their outputs are often constrained by predefined garment parameterizations or explicit edge-based representations of panel shapes and stitching. GarmentGPT Weng et al. (2026) alleviates the numerical precision issue by quantizing individual boundary edges using an RQ-VAE and training a VLM to predict the corresponding codebook indices, but still relies on predefined edge parameterizations.

We introduce **SartorAI**, which decouples geometry generation from topology prediction: a VLM generates panel geometry, and a flow model predicts stitching topology conditioned on that geometry. Our key insight is that stitching prediction becomes relatively straightforward once panel shapes and their 3D layout are given. We aim to use these geometric cues to model stitching accurately without explicitly decomposing panel boundaries into edges. For geometry, we treat each panel as a whole and train an RQ-VAE to encode its closed boundary into discrete tokens and decode a 2D occupancy field, from which the boundary is recovered. This representation requires only **16** shape tokens per panel, removing the need for explicit edge parameterizations while reducing error accumulation and improving efficiency during autoregressive generation. For topology, we discretize stitching relationships as adjacency between sampled boundary points. Inspired by recent work on mesh generation Wang et al. (2026a), we train a VAE that encodes these relationships into one latent vector per point and decodes the resulting latent set into pointwise embeddings, whose pairwise spacetime distances Shen et al. (2024) determine stitching connectivity. We then train a flow model to generate these topology latents conditioned on panel geometry.

To support these tasks, we construct **GCD-Verse** from the GCD pipeline Korosteleva et al. (2024). It contains approximately **100k** garment instances fitted to diverse body shapes and rendered with realistic PBR textures. The dataset provides structural and contextual descriptions, body prompts, and validated editing pairs with natural-language instructions. These annotations connect geometric supervision with the visual and semantic conditions used at inference.

Our main contributions are as follows:

- We introduce **SartorAI**, a multimodal framework that combines autoregressive next-token prediction of tokenized panel geometry with geometry-conditioned flow modeling of point-level stitching relationships. This formulation represents complete panel shapes and their connectivity without predefined panel-edge parameterizations or stitching schemes.
- We construct **GCD-Verse**, a dataset of approximately 100k garments fitted to diverse body shapes and rendered with realistic PBR textures. It provides text annotations covering garment structure, usage scenarios, and body measurements, as well as instruction-based editing pairs, to support multimodal generation and editing.
- Experiments demonstrate that **SartorAI** achieves state-of-the-art performance in text-conditioned garment generation, image-conditioned garment generation, and instruction-based garment editing.

2 Related Work

2.1 Garment Representations

Digital garments are represented in different forms depending on the target application. Image-based representations capture visual appearance and are widely used for virtual try-on through image synthesis Kim et al. (2024). Mesh-based representations support garment deformation, animation, and learned physical simulation Patel et al. (2020); Santesteban et al. (2022), while implicit surfaces enable flexible shape modeling, reconstruction, and draping Corona et al. (2021); De Luigi et al. (2023). Appearance and surface geometry alone, however, do not explicitly describe how fabric

pieces are cut and sewn. Sewing patterns instead encode two-dimensional panel shapes and their assembly relationships Korosteleva et al. (2024); Korosteleva & Sorkine-Hornung (2023). They closely match industrial cutting and sewing workflows and provide rest-state geometry and seam constraints that, together with material properties and body geometry, support physics-based assembly and draping Korosteleva et al. (2024); Korosteleva & Sorkine-Hornung (2023). Therefore, we adopt sewing patterns, a more compact and lower-level representation, to accomplish generation and editing tasks.

2.2 Language Models for 3D Generation and CAD

Language-driven 3D modeling explores both program-based construction and direct geometric generation. In the former paradigm, Text2CAD Khan et al. (2024) and CAD-MLLM Xu et al. (2024) predict parametric CAD sequences from text or multimodal inputs, while CAD-Recode Rukhovich et al. (2025) and Articraft Zhou et al. (2026) generate executable programs for CAD reconstruction and articulated asset creation, respectively. A complementary paradigm represents 3D geometry directly as discrete token sequences: LLaMA-Mesh Wang et al. (2024) and ShapeLLM-Omni Ye et al. (2025) fine-tune pretrained language models for autoregressive geometric generation. Existing language-based sewing pattern methods Weng et al. (2026); Bian et al. (2025); Nakayama et al. (2025) have predominantly followed the former paradigm, predicting structured garment code or parameters. In contrast, inspired by geometric tokenization, we discretize garment panel geometry itself into tokens and use a VLM to directly predict them.

2.3 Sewing Pattern Generation

Existing sewing pattern generation methods can be broadly grouped into **optimization-based** and **learning-based** approaches. Optimization-based methods optimize parameterized panel geometry through geometric objectives or differentiable cloth simulation to match target garment shapes Yu et al. (2024); Chen et al. (2025); Li et al. (2024). Learning-based methods instead learn mappings from visual observations or semantic conditions to sewing patterns. NeuralTailor Korosteleva & Lee (2022) predicts patterns from garment point clouds with a recurrent architecture, while SewFormer Liu et al. (2023) uses a hierarchical Transformer for single-image prediction. ReWeaver Li et al. (2026b) further explores topology-aware pattern prediction from sparse multi-view images. Moving toward semantic control, DressCode He et al. (2024), ChatGarment Bian et al. (2025), AIpparel Nakayama et al. (2025), GarmentGPT Weng et al. (2026), and SwiftTailor Pham et al. (2026) employ autoregressive models or VLMs for conditional pattern generation and, in some cases, instruction-based editing. In parallel, SewingLDM Liu et al. (2025), GarmageNet Li et al. (2025), and Garment Particles Nakayama et al. (2026) adopt diffusion or flow-based models for conditional pattern generation. **SartorAI** combines language-driven geometry generation with flow-based topology modeling: a VLM generates and edits tokenized whole-panel geometry, while a separate geometry-conditioned flow model predicts point-level stitching relationships.

3 METHOD

3.1 Overview

We present **SartorAI**, a framework for text- and image-conditioned garment generation and editing. Formally, we represent a sewing pattern as $\mathcal{P} = (\mathcal{G}, \mathcal{T})$, where $\mathcal{G}$ denotes its geometry and $\mathcal{T}$ denotes its topology. The input condition $\mathcal{C}$ can be an image, a textual description, or, for editing, source panels accompanied by an editing instruction.

We represent $\mathcal{G}$ as a *boxmesh*: an initial configuration of 2D panels translated and rotated into 3D space around the body for cloth simulation. We model only the boundary of each panel, so $\mathcal{G}$ can be viewed geometrically as a collection of closed curves in 3D space. The region enclosed by each panel boundary can be triangulated to obtain a mesh for simulation.

The topology $\mathcal{T}$ specifies stitching relationships. To model $\mathcal{T}$, we discretize the panel boundaries in $\mathcal{G}$ into a set of points and define $\mathcal{T}$ as a set of undirected edges connecting distinct points in this set. We impose no additional restrictions on stitch connectivity.

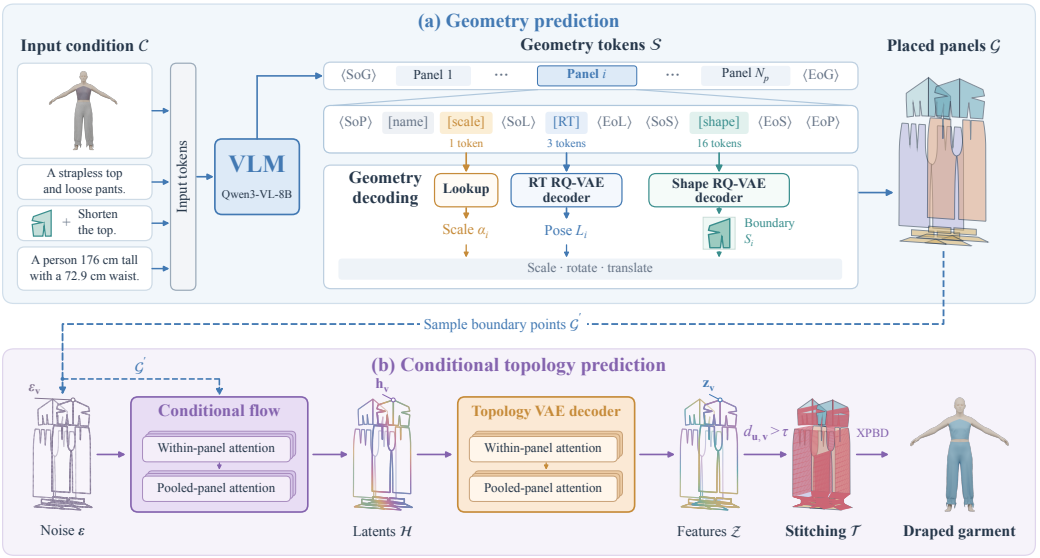

Figure 2: **Overview of SartorAI. (a)** Geometry tokens predicted by a VLM are decoded into placed panels. **(b)** A geometry-conditioned flow model generates topology latents, which the topology VAE decoder maps to pointwise features for stitching prediction via spacetime distances. XPBD simulates the assembled garment.

As illustrated in Figure 2, we use a VLM to predict $\mathcal{G}$ and a separate flow model Lipman et al. (2022) to predict $\mathcal{T}$ conditioned on $\mathcal{G}$. Our key insight is that the panel shapes and their placement in 3D space provide sufficient information to infer stitching relationships, making stitch prediction relatively straightforward once the geometry is established. Accordingly, we model geometry generation and conditional topology generation separately, using the following approximate factorization:

$$p(\mathcal{P} \mid \mathcal{C}) \approx p(\mathcal{G} \mid \mathcal{C})\, p(\mathcal{T} \mid \mathcal{G}). \tag{1}$$

We employ two residual-quantized VAEs (RQ-VAEs) Lee et al. (2022) to discretize panel geometry: one encodes sampled 2D panel boundaries into shape tokens, while the other encodes rotations and translations (RT) into layout tokens. The VLM predicts these tokens, together with panel scale tokens, conditioned on $\mathcal{C}$. A shape decoder maps the shape tokens to a continuous 2D occupancy field, which is used to recover the panel boundary. An RT decoder reconstructs the corresponding rotation and translation from the layout tokens. The decoded panel boundaries, together with the predicted scales and transformations, define the 3D geometry $\mathcal{G}$. Following Nexus Wang et al. (2026a), we train a VAE to encode the stitching relationships $\mathcal{T}$ into a latent vector at each boundary point, conditioned on the geometry $\mathcal{G}$. The decoder maps these latents to pointwise features. We infer stitching relationships using spacetime distance, a pairwise compatibility measure defined as the difference between squared distances in two learned feature subspaces. After training the VAE, we freeze it and train the flow model to generate the pointwise latents conditioned on $\mathcal{G}$.

3.2 Geometry Prediction

The garment geometry $\mathcal{G}$ consists of N_p panels, $\mathcal{G} = \{P_i\}_{i=1}^{N_p}$, each described by its shape, rotation, translation, and scale. Each panel shape is represented by a closed 2D boundary normalized to a canonical square domain. The rotation, translation, and scale together place the corresponding 2D panel in 3D space around the body.

Quantization. We use separate RQ-VAEs to encode each panel's shape S_i and RT L_i parameters and represent its scale α_i with a discrete token. We extend the VLM vocabulary with tokens representing the discrete scale values and the codebook indices of each RQ-VAE, assigning a learnable embedding to each new token.

For each panel, we select one of ten discrete scale factors to maximize the area of the scaled and centered panel while keeping it within the canonical square domain. To represent the panel's 3D

layout, we encode its RT parameters using the same RQ-VAE architecture as GarmentGPT Weng et al. (2026).

For each normalized panel shape S_i, we sample 256 points along its closed boundary. Inspired by 3DShape2VecSet Zhang et al. (2023), we map these points to embeddings using a positional embedding function and aggregate them into four latent vectors through cross-attention with four learnable queries. We then use an RQ-VAE to quantize each latent vector through four stages of residual vector quantization, yielding four discrete tokens per latent vector. During decoding, we retrieve the codewords corresponding to the discrete tokens from the codebook and sum the codewords from the four quantization stages for each latent vector, recovering four quantized latent vectors. Conditioned on these latent vectors, an implicit decoder predicts occupancy values at 2D query points, and we extract the panel boundary from the predicted occupancy field.

VLM Backbone. We build on Qwen3-VL-8B Bai et al. (2025) and fine-tune the entire language model, updating all its parameters as well as the embeddings of the newly added tokens, while keeping the visual module frozen. For image- and text-conditioned generation, the input is a garment image or a text prompt, respectively. For editing, we condition the model on the token sequence of the source garment panels and an editing instruction. Across all tasks, the model is trained to autoregressively predict the token sequence of the target garment panels.

Tokenization. We serialize the garment geometry $\mathcal{G}$ into a token sequence $\mathcal{S}$ by sorting panels according to their 3D positions, prioritizing x, then y, and finally z. Four pairs of structural tokens delimit the garment, individual panels, and their RT and shape token groups. The garment sequence begins with $\langle\mathrm{SoG}\rangle$, contains the ordered panel sequences, and ends with $\langle\mathrm{EoG}\rangle$. Each panel sequence records its name and discrete scale token, followed by the RT and shape token groups:

$$\langle\mathrm{SoP}\rangle\ [\text{name}]\ [\text{scale}]\ \langle\mathrm{SoL}\rangle\ [\mathrm{RT}]\ \langle\mathrm{EoL}\rangle\ \langle\mathrm{SoS}\rangle\ [\text{shape}]\ \langle\mathrm{EoS}\rangle\ \langle\mathrm{EoP}\rangle. \tag{2}$$

Each bracketed field denotes a token or a sequence of tokens. Within the shape token group, we concatenate tokens from successive residual quantization stages, keeping the four latent positions in the same order at every stage.

Loss Function. We first train the two RQ-VAEs separately, using reconstruction and residual quantization commitment losses. For the RT RQ-VAE, we write $L_i = (\mathbf{t}_i, \mathbf{r}_i)$, where $\mathbf{t}_i$ is the translation and $\mathbf{r}_i$ contains the three Euler angles. The objective combines mean squared error (MSE) for translation and rotation with the commitment loss:

$$\mathcal{L}_{\mathrm{RT}} = \mathrm{MSE}(\hat{\mathbf{t}}_i, \mathbf{t}_i) + \mathrm{MSE}(\hat{\mathbf{r}}_i, \mathbf{r}_i) + \beta_{\mathrm{RT}}\mathcal{L}^{\mathrm{RT}}_{\mathrm{commit}}, \tag{3}$$

where hats denote reconstructed parameters. For the shape RQ-VAE, we supervise the predicted occupancy values using binary labels indicating whether a query point lies inside the region enclosed by S_i. We use weighted binary cross-entropy (BCE) to account for the imbalance between occupied and unoccupied query points, giving the objective:

$$\mathcal{L}_{\mathrm{shape}} = \mathcal{L}_{\mathrm{BCE}} + \beta_{\mathrm{shape}}\mathcal{L}^{\mathrm{shape}}_{\mathrm{commit}}. \tag{4}$$

With both RQ-VAEs fixed, we then fine-tune the VLM to predict the target sequence $\mathcal{S} = (s_1, \ldots, s_M)$ conditioned on $\mathcal{C}$, using autoregressive cross-entropy:

$$\mathcal{L}_{\mathrm{VLM}} = -\frac{1}{M}\sum_{j=1}^{M} \log p(s_j \mid s_{<j}, \mathcal{C}). \tag{5}$$

3.3 Topology Prediction

Given the predicted panels $\{P_i\}_{i=1}^{N_p}$, we apply each panel's scale, rotation, and translation to its closed 2D boundary and discretize the transformed curves into a set of 3D points, denoted by $\mathcal{G}' = \{\mathbf{v}_j\}_{j=1}^{N_v} \subset \mathbb{R}^3$. Each sampled point retains its panel membership and cyclic boundary order. For topology prediction, we condition on this discrete representation to approximate the distribution of stitching relationships: $p(\mathcal{T} \mid \mathcal{G}) \approx p(\mathcal{T} \mid \mathcal{G}')$.

Following Nexus Wang et al. (2026a), we represent the points in $\mathcal{G}'$ by a set of embeddings $\mathcal{Z} = \{\mathbf{z}_{\mathbf{v}}\}_{\mathbf{v}\in\mathcal{G}'}$, where $\mathbf{z}_{\mathbf{v}} \in \mathbb{R}^c$. We use the spacetime distance introduced in Shen et al. (2024) to infer stitching relationships between pairs of points. Specifically, we split each embedding into spatial

and temporal components, $\mathbf{z_v} = [\mathbf{s_v}; \mathbf{t_v}]$, where $[\cdot;\cdot]$ denotes concatenation and $\mathbf{s_v}, \mathbf{t_v} \in \mathbb{R}^{c/2}$. The spacetime distance between two points $\mathbf{u}, \mathbf{v} \in \mathcal{G}'$ is defined as

$$d_{\mathbf{u},\mathbf{v}} = \|\mathbf{s_u} - \mathbf{s_v}\|_2^2 - \|\mathbf{t_u} - \mathbf{t_v}\|_2^2 . \tag{6}$$

We connect two distinct points with an edge when $d_{\mathbf{u},\mathbf{v}} > \tau$, where τ is a threshold, indicating that the points should be stitched together. Our ablation study in Section 5.4 shows that using spacetime distance yields more accurate stitching reconstruction than using Euclidean distance or cosine similarity.

Topology VAE. To learn this topology representation, we train a VAE Kingma & Welling (2013) that encodes $\mathcal{T}$ conditioned on $\mathcal{G}'$ into per-point latent vectors $\mathcal{H} = \{\mathbf{h_v}\}_{\mathbf{v}\in\mathcal{G}'}$. Our encoder interleaves GraphSAGE convolutions Hamilton et al. (2017) over boundary adjacency and stitching edges with self-attention within each panel. Attention over pooled panel features then exchanges information across panels. This design balances fine-grained pointwise feature encoding with computational efficiency while fully exploiting the structural prior provided by panel topology. The encoder predicts a diagonal Gaussian posterior $q(\mathcal{H} \mid \mathcal{G}', \mathcal{T})$. A transformer decoder maps sampled latents to $\mathcal{Z}$ through attention within and across panels. For reconstruction, we supervise all unordered pairs of distinct points in $\mathcal{G}'$ with binary labels $y_{\mathbf{u},\mathbf{v}}$, where $y_{\mathbf{u},\mathbf{v}} = 1$ indicates a stitching edge in $\mathcal{T}$ and 0 otherwise. Following Nexus Wang et al. (2026a), we partition these pairs into true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) according to their ground-truth labels and predicted edges. We average the binary cross-entropy losses within each group before averaging across the four groups:

$$\mathcal{L}_{\text{stitch}} = \frac{1}{4} \sum_{g\in\{\text{TP,TN,FP,FN}\}} \mathbb{E}_{(\mathbf{u},\mathbf{v})\in g}[\text{BCE}(d_{\mathbf{u},\mathbf{v}}, y_{\mathbf{u},\mathbf{v}})] , \tag{7}$$

where BCE takes the continuous spacetime distance $d_{\mathbf{u},\mathbf{v}}$ as its logit input and $y_{\mathbf{u},\mathbf{v}}$ as the target label. We combine this reconstruction loss with KL regularization to jointly train the VAE:

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}_{\text{stitch}} + \beta_{\text{KL}}\mathcal{L}_{\text{KL}}, \tag{8}$$

where $\mathcal{L}_{\text{KL}}$ is the KL divergence from the posterior to a standard normal prior, averaged over points and latent dimensions, and β_{KL} controls the strength of this regularization.

Conditional Flow. To generate topology latents for a given geometry $\mathcal{G}'$, we freeze the VAE and learn $p(\mathcal{H} \mid \mathcal{G}')$ through conditional flow matching Lipman et al. (2022). During training, we sample pointwise latents from the VAE posterior and denote their vectorized form by $\mathbf{x}_1 \in \mathbb{R}^D$. We construct a linear path between Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and these latents as $\mathbf{x}_t = (1-t)\boldsymbol{\epsilon} + t\mathbf{x}_1$, where $t \sim \mathcal{U}(0, 1)$. A velocity predictor v_θ, conditioned on $\mathcal{G}'$, is trained to approximate the path velocity $\mathbf{x}_1 - \boldsymbol{\epsilon}$ by minimizing

$$\mathcal{L}_{\text{flow}}(\theta) = \mathbb{E}_{t,\mathbf{x}_1,\boldsymbol{\epsilon}}\left[\frac{1}{D}\left\|v_\theta(\mathbf{x}_t, t, \mathcal{G}') - (\mathbf{x}_1 - \boldsymbol{\epsilon})\right\|_2^2\right] . \tag{9}$$

We parameterize v_θ with a transformer that takes $\mathbf{x}_t$, point coordinates, and timestep embeddings as input. Its attention layers operate on individual points within each panel and on pooled features across panels, with 3D RoPE Su et al. (2024) incorporating point positions.

At inference, we initialize the latent state with Gaussian noise and integrate the learned velocity field using DPM-Solver++ Lu et al. (2025). The frozen VAE decoder then maps the generated latents to $\mathcal{Z}$, and we apply the spacetime criterion to recover the stitching relationships $\mathcal{T}$.

4 DATASET

Building on GCD Korosteleva et al. (2024), we construct GCD-Verse, comprising approximately 100k garment instances with diverse body shapes and appearances, split 9:1 into training and test sets. We sample garment design parameters and one of GCD's 5,000 body shapes, construct sewing patterns from the corresponding measurements, and drape the garments using XPBD Macklin et al. (2016). We exclude outfits with insufficient body coverage and extend the design space to separate upper and lower garments without waistbands. For these two-piece outfits, waist constraints prevent

the lower garment from slipping, while staged simulation Wang et al. (2026b) first drapes the lower garment and then uses it as an additional collider for the upper garment.

For appearance generation, we select 37K high-quality OpenVTON images Li et al. (2026a) and remove their backgrounds using SAM2 Ravi et al. (2025). Rule-based descriptions derived from the sewing patterns are matched to these images using CLIP similarity Radford et al. (2021). NI-Tex Shan et al. (2026) generates PBR textures conditioned on the matched images, and Blender renders the garments with randomized camera viewpoints, target positions, and framing scales near the canonical frontal view.

Conditioned on rendered images and rule-based descriptions, GPT-5.5 produces detailed structural prompts and context prompts describing usage scenarios; body prompts summarize the associated measurements. We construct editing pairs by changing one semantic attribute of a source pattern while keeping the body fixed. Only targets passing geometric validation and physical simulation are retained, together with natural-language instructions describing the changes. Further details are provided in Appendix B.

5 EXPERIMENTS

5.1 Implementation Details

The shape RQ-VAE encodes 256 boundary samples into four 384-dimensional latents, each quantized through four residual stages with a shared codebook of 4,096 entries; the RT RQ-VAE uses three residual stages with a shared codebook of 1,024 entries. After training the two RQ-VAEs independently, we freeze them and fine-tune the language model of Qwen3-VL-8B Bai et al. (2025), including the added token embeddings, for 20 epochs while keeping its visual module frozen. During VLM training, we sample editing, image-conditioned generation, and generation conditioned on simple descriptions, detailed descriptions, or occasion prompts with relative weights of $2 : 6 : 2 : 6 : 2$, respectively. Task assignments are resampled at each epoch. For topology, we first train a VAE with 64-dimensional latents at each boundary point, then freeze it to train a conditional flow transformer with a hidden dimension of 2,048, 16 within-panel attention blocks, and two blocks over pooled panel features. We use the flow checkpoint obtained after approximately 700 epochs of training. All models are trained on eight NVIDIA H200 GPUs. Further architectural and optimization details are provided in Appendix D.

5.2 Metrics

Following Nakayama et al. (2025), we report **Panel Accuracy** as the percentage of panels that are correctly predicted and successfully parsed. All other metrics are computed only on these panels. **Rot L2** and **Transl L2** measure the L_2 distances between the predicted and ground-truth rotations and translations, respectively. Since our representation does not explicitly model individual panel edges, we follow Li et al. (2026b) and evaluate shape reconstruction using **Panel IoU** and **Panel CD**. We first align each predicted panel with its ground-truth counterpart, then compute the intersection over union (IoU) of the enclosed panel regions and the Chamfer distance (CD) between their boundary curves. IoU is the ratio of the intersection area to the union area of the two regions. Transl L2 and Panel CD are reported in centimeters. Since our representation does not explicitly model stitching edges, we introduce **Stitch F1** to evaluate stitching prediction. We uniformly sample each closed panel boundary in normalized arc length. For predicted and ground-truth panels with matching semantic names, we align their boundary starting points using contour geometry and adopt a consistent traversal direction. A predicted point pair matches a ground-truth pair if they connect the same panels and the circular arc-length errors at both endpoints are below a prescribed tolerance (2% of each panel's perimeter). We compute the F-score between the predicted point pairs and the ground-truth matches as the Stitch F1.

5.3 Comparison with Baselines

We compare SartorAI with AIpparel Nakayama et al. (2025) and ChatGarment Bian et al. (2025), both initialized from their publicly released checkpoints and further trained on **GCD-Verse**. The comparison covers image-conditioned generation, text-conditioned generation from simple descrip-

tions, detailed descriptions, and occasion prompts, as well as instruction-guided editing. Figure 4 presents qualitative comparisons on GCD-Verse, while Figure 5 shows results on in-the-wild photographs from OpenVTON Li et al. (2026a). Figure 3 presents qualitative comparisons for instruction-guided editing.

Table 1 shows that ChatGarment achieves the lowest rotation and translation errors. Its generation pipeline provides a strong layout prior: it first predicts a YAML configuration and then constructs sewing patterns from this configuration and body measurements using predefined rules. SartorAI directly predicts panel layouts without this procedural prior, resulting in higher layout errors than ChatGarment, but achieves the best results on all remaining metrics. The qualitative comparisons, particularly the in-the-wild examples in Figure 5, further illustrate SartorAI's closer alignment with the structural details of the input garments.

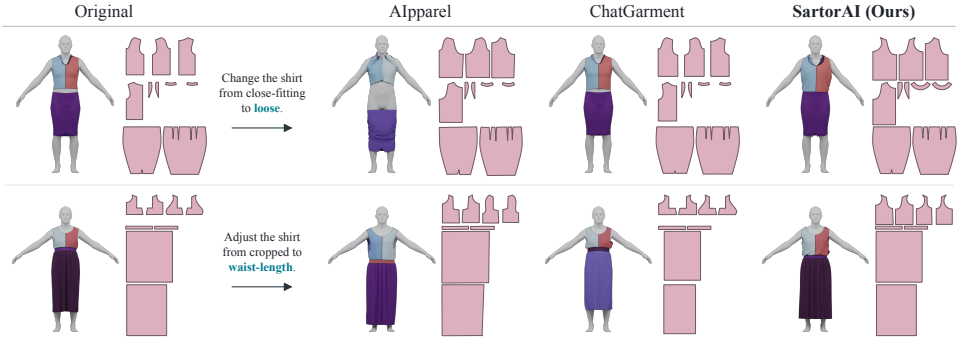

Figure 3: **Instruction-guided editing.** Each row shows a source garment, an edit instruction, and the compared methods' simulated garments and 2D sewing patterns.

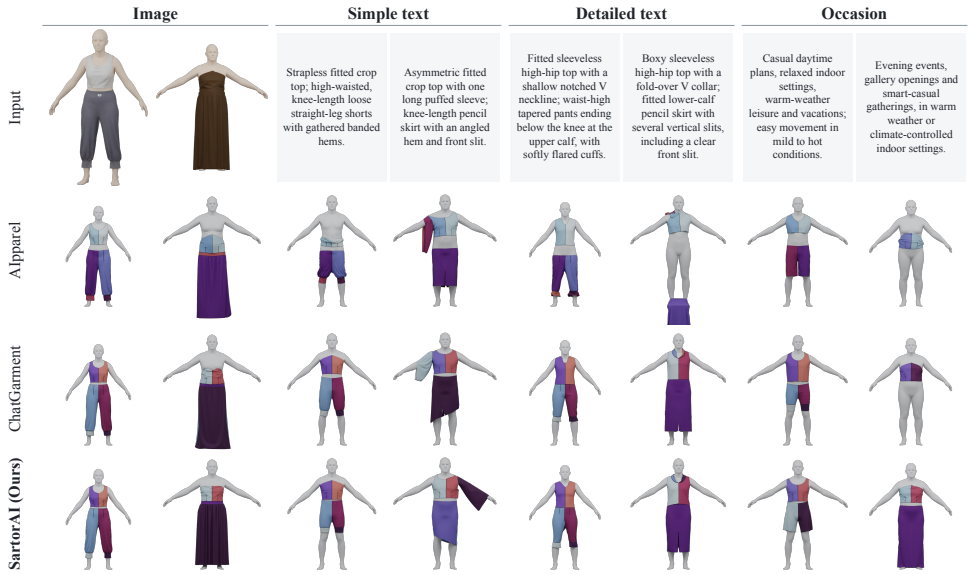

Figure 4: **Qualitative comparisons on GCD-Verse.** Columns cover image, simple-text, detailed-text, and occasion conditioning. Rows show the inputs and the final simulated garments from AIpparel, ChatGarment, and SartorAI. Text prompts are abridged for display.

5.4 Ablation Studies

Number of residual quantizers. We investigate the number of residual quantization stages Q in our shape RQ-VAE by comparing $Q = 2$, $Q = 3$, and $Q = 4$. As shown in Table 2, increasing Q improves reconstruction quality. With $Q = 4$, the model achieves a panel IoU of 99.70% and

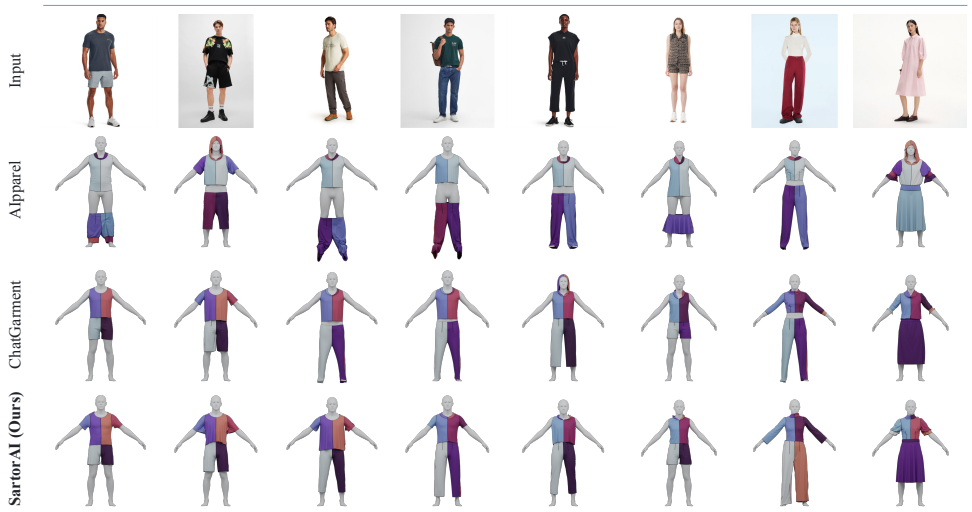

Figure 5: **Generation from in-the-wild images.** Top: reference photographs from OpenVTON. Below: simulated outputs. Each column uses a body estimated by GPT-5.5.

Table 1: **Quantitative comparison on GCD-Verse.**

Method	Panel Acc. ↑	Rot L2 ↓	Transl L2 ↓	Panel IoU ↑	Panel CD ↓	Stitch F1 ↑
ChatGarment	87.52	**0.0192**	**5.91**	0.740	1.893	0.871
AIpparel	86.70	0.0254	6.82	0.736	2.027	0.830
SartorAI (ours)	**88.47**	0.0248	6.38	**0.814**	**1.824**	**0.896**

an occupancy prediction accuracy of 99.88%, supporting our choice of four stages for high-fidelity panel reconstruction.

Table 2: **Effect of residual quantizer count in the shape RQ-VAE.** IoU measures panel overlap; Acc. measures occupancy prediction accuracy.

Metric	$Q = 2$	$Q = 3$	$Q = 4$
Total loss ($\times 10^2$) ↓	2.13	1.27	**1.15**
IoU (%) ↑	99.38	99.54	**99.70**
Acc. (%) ↑	99.23	99.72	**99.88**

Table 3: **Pairwise criteria for topology VAE reconstruction.** Metrics use exact matching of reconstructed and ground-truth stitching point pairs.

Metric	Euclidean	Cosine	Spacetime
F1-score ↑	0.8626	0.9307	**0.9999**
Precision ↑	0.8891	0.8780	**1.0000**
Recall ↑	0.8376	0.9901	**0.9999**

Pairwise criterion for topology reconstruction. We investigate how the pairwise criterion affects stitching reconstruction in the topology VAE by comparing Euclidean distance, cosine similarity, and spacetime distance on the decoded pointwise embeddings. As shown in Table 3, the spacetime criterion achieves near-perfect reconstruction, with an F1-score of 0.9999. These results support our choice of spacetime distance for accurate reconstruction of pointwise stitching relationships.

6 CONCLUSION

We present **SartorAI**, a framework that combines a vision-language model with a flow model for purely geometric modeling of sewing patterns. SartorAI achieves state-of-the-art performance in image-conditioned generation, text-conditioned generation, and interactive editing on **GCD-Verse**, our synthetic garment dataset. It also generalizes well to real-world images, benefiting from the rich garment textures in GCD-Verse and the visual understanding capabilities of the pretrained VLM.

AI USE STATEMENT

In this work, we used generative AI tools for synthetic data generation and annotation, assistance with method implementation and debugging, feedback on training and evaluation procedures, interpretation of experimental results, and Chinese-to-English translation. Specifically, GPT-5.5 was used to generate detailed garment descriptions and usage-context annotations from rendered images and rule-based descriptions. GPT-5.5 also provided approximate body-size priors for selecting proxy bodies in the in-the-wild qualitative comparisons. The dataset construction and annotation procedures are described in Appendix B. For writing, we used GPT5.5 to help with paper translation and proofreading, and to adjust the formatting. All AI translation and proofreading work underwent human secondary verification.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Siyuan Bian, Chenghao Xu, Yuliang Xiu, Artur Grigorev, Zhen Liu, Cewu Lu, Michael J Black, and Yao Feng. Chatgarment: Garment estimation, generation and editing via large language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2924–2934. IEEE, 2025.

Hsiao-yu Chen, Egor Larionov, Ladislav Kavan, Gene Lin, Doug Roble, Olga Sorkine-Hornung, and Tuur Stuyck. Dress anyone: Automatic physically-based garment pattern refitting. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 8(4):1–17, 2025.

Enric Corona, Albert Pumarola, Guillem Alenya, Gerard Pons-Moll, and Francesc Moreno-Noguer. Smplicit: Topology-aware generative model for clothed people. In *2021 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11870–11880. IEEE, 2021.

Luca De Luigi, Ren Li, Benoit Guillard, Mathieu Salzmann, and Pascal Fua. Drapenet: Garment generation and self-supervised draping. In *2023 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 1451–1460. IEEE, 2023.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Kai He, Kaixin Yao, Qixuan Zhang, Jingyi Yu, Lingjie Liu, and Lan Xu. Dresscode: Autoregressively sewing and generating garments from text guidance. *ACM Transactions on Graphics (TOG)*, 43(4):1–13, 2024.

Mohammad S Khan, Sankalp Sinha, Talha U Sheikh, Didier Stricker, Sk A Ali, and Muhammad Z Afzal. Text2cad: Generating sequential cad designs from beginner-to-expert level text prompts. *Advances in Neural Information Processing Systems*, 37:7552–7579, 2024.

Jeongho Kim, Gyojung Gu, Minho Park, Sunghyun Park, and Jaegul Choo. Stable viton: Learning semantic correspondence with latent diffusion model for virtual try-on. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8176–8185. IEEE, 2024.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Maria Korosteleva and Sung-Hee Lee. Neuraltailor: Reconstructing sewing pattern structures from 3d point clouds of garments. *ACM Transactions on Graphics (TOG)*, 41(4):1–16, 2022.

Maria Korosteleva and Olga Sorkine-Hornung. Garmentcode: Programming parametric sewing patterns. *ACM Transactions on Graphics (TOG)*, 42(6):1–15, 2023.

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 *European Conference on Computer Vision*, pp. 110–127. Springer, 2024.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11513–11522. IEEE, 2022.

Jin Li, Tao Chen, Kai Wen, Siqi Yin, Shuai Jiang, Weijie Wang, Jingwen Luo, and Chenhui Wu. Openvton-bench: A large-scale high-resolution benchmark for controllable virtual try-on evaluation. *arXiv preprint arXiv:2601.22725*, 2026a.

Ming Li, Hui Shan, Kai Zheng, Chentao Shen, Siyu Liu, Yanwei Fu, Zhen Chen, and Xiangru Huang. Reweaver: Towards simulation-ready and topology-accurate garment reconstruction. *arXiv preprint arXiv:2601.16672*, 2026b.

Siran Li, Ruiyang Liu, Chen Liu, Zhendong Wang, Gaofeng He, Yong-Lu Li, Xiaogang Jin, and Huamin Wang. Garmagenet: A multimodal generative framework for sewing pattern design and generic garment modeling. *ACM Transactions on Graphics (TOG)*, 44(6):1–23, 2025.

Yifei Li, Hsiao-yu Chen, Egor Larionov, Nikolaos Sarafianos, Wojciech Matusik, and Tuur Stuyck. Diffavatar: Simulation-ready garment optimization with differentiable simulation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4368–4378. IEEE, 2024.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Lijuan Liu, Xiangyu Xu, Zhijie Lin, Jiabin Liang, and Shuicheng Yan. Towards garment sewing pattern reconstruction from a single image. *ACM Transactions on Graphics (TOG)*, 42(6):1–15, 2023.

Shengqi Liu, Yuhao Cheng, Zhuo Chen, Xingyu Ren, Wenhan Zhu, Lincheng Li, Mengxiao Bi, Xiaokang Yang, and Yichao Yan. Multimodal latent diffusion model for complex sewing pattern generation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17640–17650. IEEE, 2025.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 22 (4):730–751, 2025.

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*, pp. 49–54, 2016.

Kiyohiro Nakayama, Jan Ackermann, Timur Levent Kesdogan, Yang Zheng, Maria Korosteleva, Olga Sorkine-Hornung, Leonidas J Guibas, Guandao Yang, and Gordon Wetzstein. Aipparel: A multimodal foundation model for digital garments. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8138–8149. IEEE, 2025.

Kiyohiro Nakayama, I-Chao Shen, Ruofan Liu, Yiming Wang, Gordon Wetzstein, and Takeo Igarashi. Garment particles: a 2d–3d symmetric garment representation for generation and editing. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–12, 2026.

Chaitanya Patel, Zhouyingcheng Liao, and Gerard Pons-Moll. Tailornet: Predicting clothing in 3d as a function of human pose, shape and garment style. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 7363–7373. IEEE, 2020.

Phuc Pham, Uy Dieu Tran, Binh-Son Hua, and Phong Nguyen. Swifttailor: Efficient 3d garment generation with geometry image representation. *arXiv preprint arXiv:2603.19053*, 2026.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. In *International Conference on Learning Representations*, volume 2025, pp. 28085–28128, 2025.

Danila Rukhovich, Elona Dupont, Dimitrios Mallis, Kseniya Cherenkova, Anis Kacem, and Djamila Aouada. Cad-recode: Reverse engineering cad code from point clouds. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9801–9811. IEEE, 2025.

Igor Santesteban, Miguel A Otaduy, and Dan Casas. Snug: Self-supervised neural dynamic garments. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8130–8140. IEEE, 2022.

Hui Shan, Ming Li, Haitao Yang, Kai Zheng, Sizhe Zheng, Yanwei Fu, and Xiangru Huang. Nitex: Non-isometric image-based garment texture generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19884–19893, 2026.

Tianchang Shen, Zhaoshuo Li, Marc Law, Matan Atzmon, Sanja Fidler, James Lucas, Jun Gao, and Nicholas Sharp. Spacemesh: A continuous representation for learning manifold surface meshes. In *SIGGRAPH Asia 2024 conference papers*, pp. 1–11, 2024.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Hanxiao Wang, Ying-Tian Liu, Yuan-Chen Guo, Qi-Yuan Feng, Zi-Xin Zou, Ding Liang, Biao Zhang, and Yan-Pei Cao. Nexus: Native mesh generation with diffusion. *ACM Transactions on Graphics (TOG)*, 45(4):1–14, 2026a.

Yuanhao Wang, Johanna Karras, Yingwei Li, and Ira Kemelmacher-Shlizerman. Fit: A large-scale dataset for fit-aware virtual try-on. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–11, 2026b.

Zhengyi Wang, Jonathan Lorraine, Yikai Wang, Hang Su, Jun Zhu, Sanja Fidler, and Xiaohui Zeng. Llama-mesh: Unifying 3d mesh generation with language models. *arXiv preprint arXiv:2411.09595*, 2024.

Fangsheng Weng, Junhao Chen, Xiang Li, Jie Qin, Hanzhong Guo, Xiaoguang Han, et al. Garmentgpt: Compositional garment pattern generation via discrete latent tokenization. In *The Fourteenth International Conference on Learning Representations*, 2026.

Jingwei Xu, Chenyu Wang, Zibo Zhao, Wen Liu, Yi Ma, and Shenghua Gao. Cad-mllm: Unifying multimodality-conditioned cad generation with mllm. *arXiv preprint arXiv:2411.04954*, 2024.

Junliang Ye, Zhengyi Wang, Ruowen Zhao, Shenghao Xie, and Jun Zhu. Shapellm-omni: A native multimodal llm for 3d generation and understanding. *arXiv preprint arXiv:2506.01853*, 2025.

Boyang Yu, Frederic Cordier, and Hyewon Seo. Inverse garment and pattern modeling with a differentiable simulator. In *Computer Graphics Forum*, volume 43, pp. e15249. Wiley Online Library, 2024.

Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions On Graphics (TOG)*, 42(4):1–16, 2023.

Matt Zhou, Ruining Li, Xiaoyang Lyu, Zhaomou Song, Zhening Huang, Chuanxia Zheng, Christian Rupprecht, Andrea Vedaldi, and Shangzhe Wu. Articraft: An agentic system for scalable articulated 3d asset generation. *arXiv preprint arXiv:2605.15187*, 2026.

APPENDIX CONTENTS

A More Results **14**

B Details on GCD-Verse **14**

B.1 Garment Geometry and Body Variation 14

B.2 Texture Generation and Rendering 14

B.3 Text and Body Annotations 16

B.4 Instruction-Based Editing Pairs 17

B.5 Dataset Partition 18

C Details on Simulation **19**

C.1 Simulation for GCD-Verse 19

C.2 Simulation of Predicted Sewing Patterns 20

D More Implementation Details on SartorAI **21**

D.1 Geometric Preprocessing 21

D.2 RT RQ-VAE 21

D.3 Shape RQ-VAE 22

D.4 VLM 22

D.5 Topology VAE 22

D.6 Conditional Flow 23

E Limitations and Future Work **23**

F Discussion **24**

A More Results

Figure 6 presents additional qualitative results from SartorAI for image- and text-conditioned sewing pattern generation. The image inputs include GCD-Verse renderings and in-the-wild photographs, while the text inputs range from simple garment descriptions to detailed specifications and occasion prompts. Each input is paired with a garment obtained by physically simulating the generated sewing pattern. Colors distinguish pattern panels for visualization. More editing examples are shown in Figure 8.

B Details on GCD-Verse

GCD-Verse contains approximately 100k garment instances, pairing sewing patterns with body measurements, simulated garments, textured renderings, and text annotations for multimodal generation and editing. This section expands on the construction of these modalities, the generation of editing pairs, and the dataset partition. The simulation procedure is described separately in Appendix C. Figure 7 shows examples of the resulting garments.

B.1 Garment Geometry and Body Variation

Procedural design space. We build on the garment programs and body collection of GCD Korosteleva et al. (2024); Korosteleva & Sorkine-Hornung (2023). A design is specified by categorical choices, such as garment components and neckline types, together with continuous parameters controlling dimensions and proportions. The available components include fitted and loose tops, several skirt constructions, trousers, waistbands, sleeves, collars, and hoods. Skirt variations include circle, asymmetric, paneled, pencil, tiered, and godet constructions. Parameters are conditional: sleeve dimensions apply only when sleeves are present, and godet parameters apply only to the corresponding skirt construction. Sampling these choices changes both panel geometry and the number and connectivity of panels.

Fitting to different bodies. For each sampled design, we select a body from the 5,000 shapes provided by GCD and construct the pattern using that body's measurements. Thus, body variation enters the pattern construction itself, rather than being applied as a uniform scaling of a garment fitted to a single reference body. The saved measurements include the quantities required by the procedural constructors as well as the six measurements used in our body descriptions: height, bust circumference, waist circumference, hip circumference, leg length, and arm length. The body collection supplies both straight-leg and separated-leg poses; trouser designs use the latter during construction and simulation. These controlled poses provide a consistent initial arrangement for draping while allowing variation in body proportions.

Geometric and physical filtering. We check parameter compatibility before constructing a pattern and reject constructions with self-intersecting panel boundaries or excessive total garment length. Short upper-only garments are excluded, while sufficiently long shirt constructions can form one-piece outfits. Our assembly procedure also supports separate upper and lower garments without a connecting waistband. After construction, the panels are triangulated, assembled, and simulated on the associated body. Patterns that fail meshing or simulation checks are removed, including cases that time out, fail to settle, or exceed the configured intersection thresholds. This filtering determines the retained data; a valid parameter combination alone is insufficient for inclusion. The garment and body measurements, original sewing pattern, and draped mesh remain associated throughout subsequent texturing and annotation.

B.2 Texture Generation and Rendering

Reference image bank. To give the simulated garments varied appearances, we build a reference bank from the selected OpenVTON images Li et al. (2026a) described in the main paper. We organize the bank into five garment categories: tops, skirts, trousers, shorts, and dresses. SAM2 Ravi et al. (2025) removes the background from each reference garment, producing an RGBA cutout. The cutouts provide appearance references for texture synthesis; the sewing patterns and simulated meshes remain the source of geometric supervision.

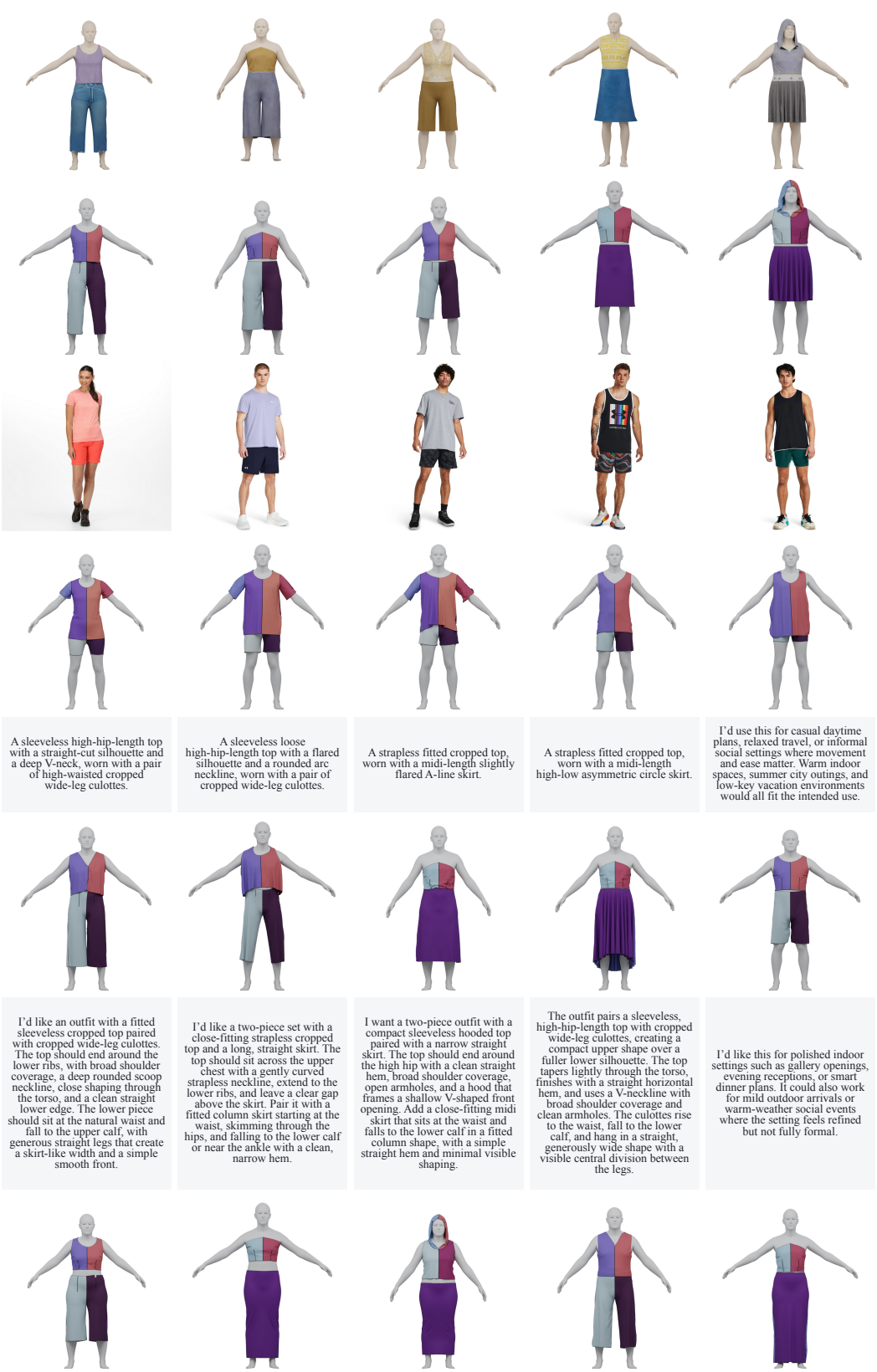

Figure 6: Additional qualitative results from SartorAI.



Figure 7: **Examples from GCD-Verse.** The dataset combines variations in panel construction, body proportions, and rendered appearance. These are saved dataset renderings of procedurally constructed and physically simulated garments. Their different colors and surface patterns come from the texture-generation stage; the sewing patterns specify geometry and stitching.

Matching garments to reference images. We derive a short description of each simulated garment from its design parameters, using the rules described below. When upper and lower components are processed separately, each receives its own description and reference matches; otherwise, we match the garment as a whole. We encode the descriptions and reference images with CLIP ViT-B/32 Radford et al. (2021), normalize their feature vectors, and score text–image compatibility by their inner product. Matching is restricted to the corresponding garment category so that, for example, a trouser component retrieves trouser references.

To distribute the appearance references across the dataset, our matching procedure combines CLIP similarity with limits on reference reuse across the query set.

Generating surface appearance. Conditioned on a reference cutout and the simulated mesh, NI-Tex Shan et al. (2026) generates multiview appearance maps. The geometry conditioning consists of rendered normal and position maps, which keep the texture generation tied to the existing garment surface. The generated maps are upsampled, baked into a UV atlas, and completed by texture inpainting. The exported assets include the textured mesh, albedo, and metallic and roughness maps. Upper and lower garments are textured independently when the outfit is split into two components, allowing their appearance references to differ.

Rendering variation. We render the clothed body in Blender at 1024×1024 resolution. The dataset image is a randomized orthographic view near the canonical frontal direction. The camera azimuth and elevation vary within $\pm 15^\circ$ and $\pm 8^\circ$, respectively; the target position is perturbed by up to 2.5% of the scene extent along each axis, and the framing scale varies by $\pm 4\%$. We also perturb light positions, intensities, and sizes and vary the environment illumination. For the final cloth shader, we retain the generated albedo while setting metallicity to zero and roughness to 0.7. These rendering choices provide visual variation without changing the pattern or the physical material parameters used for draping. The resulting image is shared by image-conditioned training and the image-assisted annotation procedure.

B.3 Text and Body Annotations

Following the general annotation strategy of AIpparel Nakayama et al. (2025), we combine procedural design information with rendered observations. We retain a short rule-based description and

Table 4: **Complete text annotations for the garment in Figure 7(a).** The rule-based description and body measurements are derived deterministically; the detailed pattern and context descriptions are generated from the rendering and rule-based text.

Rule-based description. A sleeveless loose high-hip-length top with a flared silhouette and a V-neckline, worn with a pair of knee-length straight-leg shorts with banded ruffled flounce hems.
Detailed pattern description. A loose sleeveless top forms a flared, boxy upper silhouette, widening through the sides and ending around the high hip with a slightly uneven-looking lower edge. The neckline cuts into a clear V, while the armholes are deep and open, giving the side panels a hanging, draped shape. Below it, straight-leg shorts extend to just above the knees and finish with banded flounce hems that flare outward, with small pointed edges visible near the outer sides.
Context description. This outfit would fit relaxed daytime settings such as casual meetups, resort walks, informal lunches, or low-key travel activities. It feels most plausible in warm conditions or climate-controlled indoor spaces where ease of movement and an unfussy mood are appropriate.
Body description. A person 181.0 cm tall with a 115.0 cm bust, 104.0 cm waist, 115.0 cm hips, 90.3 cm leg length, and 57.8 cm arm length.

generate two complementary language annotations: a detailed pattern description and a context description. A separate, deterministic body description supplies numerical measurements. Keeping these fields distinct allows the same garment to support requests at different levels of specificity.

Rule-based descriptions. The rule system reads the saved design parameters and body measurements and describes only the active components. Categorical choices provide terms such as V-neckline, hood, pencil skirt, and godet skirt. Continuous parameters are converted into qualitative descriptions of fit, silhouette, and length. Where the interpretation depends on body proportions, the rules use body landmarks rather than a single fixed metric threshold. For example, top length is interpreted relative to the waist and hip levels, and a sleeve's effective length accounts for the depth of its armhole. The rules also distinguish the left and right sides when asymmetric construction is enabled.

Individual phrases are composed into a description of the complete garment or outfit. For two-piece outfits, we preserve descriptions for the upper and lower components as well as the combined description. This provides a common semantic description for reference-image retrieval and for caption generation.

Detailed pattern descriptions. We provide GPT-5.5 with one rendered garment image and its rule-based description. The rule-based text supplies a prior over the garment's components and construction, while the image provides evidence about the draped silhouette, visible proportions, and relative lengths. We instruct the annotator to describe the cut and structure, including informative details of the neckline, sleeves, waist, hems, and component relationships when supported by the inputs.

Context descriptions. The same annotation request also produces a separate context field describing plausible occasions, environments, or weather conditions for the outfit. This field expresses a use scenario without listing garment types or construction features. It therefore supports a less explicit form of conditioning than a direct pattern description.

Body descriptions and task inputs. We convert the six body measurements into a fixed English sentence, reporting each quantity in centimeters to one decimal place. These values come directly from the saved body measurements and are not estimated by the language model from the rendering. The short description, detailed description, and context can each serve as a text condition, with the body sentence optionally appended to specify the intended wearer. Image-conditioned examples use the saved garment rendering. For editing, the condition instead contains the source garment representation and the edit instruction, with the target garment as supervision. These task variants reuse the same underlying records and do not increase the number of distinct source garments. Table 4 gives a complete example of the four text fields associated with one garment.

B.4 Instruction-Based Editing Pairs

We construct editing pairs by applying one eligible semantic operation to a source design while keeping its body measurements fixed. The ten operation families in Table 5 are selected according to the active construction, with numerical edits restricted to valid ranges and required to produce

Table 5: **Semantic editing operations.** Each accepted pair applies one eligible operation with the source body held fixed. Numerical changes are restricted to valid parameter ranges and checked for a geometric effect.

Attribute	Operation and applicability
Top fit	Adjust torso width for a compatible top.
Top length	Lengthen or shorten a compatible top.
Neckline shape	Switch between supported front-neckline constructions.
Hood presence	Add or remove a two-panel hood at a compatible neckline.
Collar presence	Add or remove a fold-over collar.
Sleeve presence	Switch between sleeved and sleeveless construction.
Sleeve length	Lengthen or shorten existing sleeves.
Skirt length	Adjust the active skirt construction's length.
Godet count	Change the number of inserts in a compatible godet skirt.
Trouser length	Lengthen or shorten existing trousers or shorts.

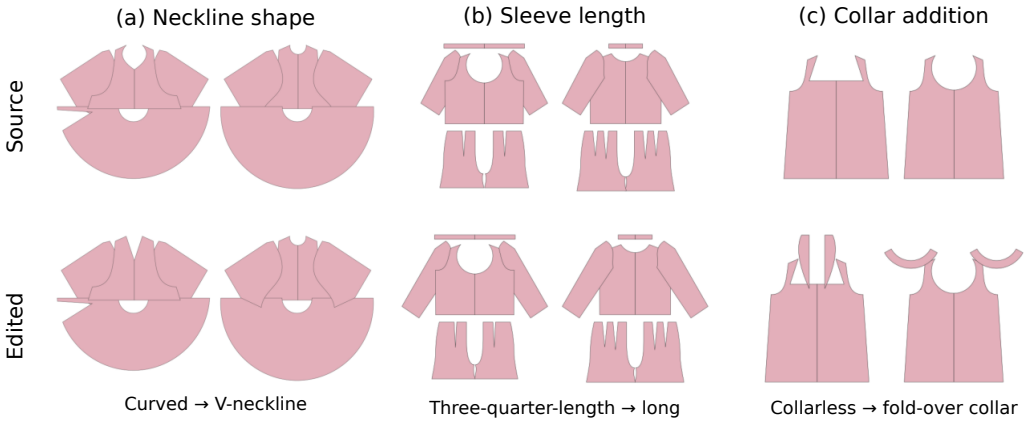

Figure 8: **Editing examples from GCD-Verse.** Source patterns are shown above their edited targets. Changing the neckline or sleeve length modifies existing panels, whereas adding a collar introduces new panels. The labels summarize the edits; each dataset pair stores a full instruction, such as "Change both sleeves from three-quarter-length to long."

a meaningful geometric or semantic change. An edit may affect multiple panels or seams through construction dependencies. To preserve unaffected regions, we compare the unedited and edited reconstructions and, where edge indexing remains compatible, copy unaffected panels from the saved source specification, including their geometry and placement.

For each pair, we generate a template-based instruction specifying the edited feature and its transition from the source state to the target state, along with a rule-based target description. Figure 8 illustrates edits that modify existing panels or introduce new ones.

B.5 Dataset Partition

Figure 9 summarizes the structural complexity of the source patterns. Garments contain between 4 and 35 panels, with a median of 10. The number of boundary edges ranges from 24 to 214, with a median of 80, while the number of paired seams ranges from 6 to 88, with a median of 26.

The retained garments also span body proportions and description lengths. Body height ranges from 147 to 196 cm, and waist circumference from 49.2 to 127.0 cm. The median detailed pattern description contains 75 words, compared with 37 words for the context description, with words counted by whitespace separation.

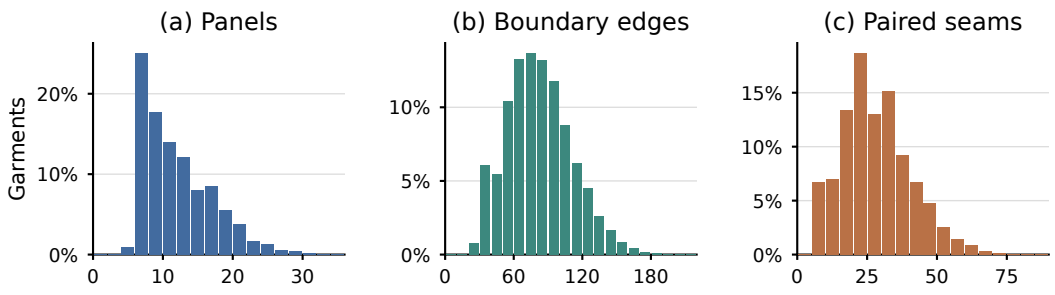

Figure 9: **Structural statistics of GCD-Verse.** Distributions of panel counts, parametric boundary-edge counts, and paired-seam counts per source garment. Each vertical axis reports the percentage of garments.

C Details on Simulation

C.1 Simulation for GCD-Verse

We build on the simulation pipeline released with GCD Korosteleva et al. (2024), which triangulates sewing patterns, assembles panels by merging corresponding seam vertices, and drapes the resulting cloth onto a body mesh using XPBD Macklin et al. (2016). We retain its cloth material model, body collision filtering, and reference-based collision handling. Our main changes concern the composition and support of separate upper and lower garments, as illustrated in Figure 10.

Separate upper and lower garments. In the released GCD composition code, adding a lower garment beneath an upper garment creates a seam between their adjoining boundaries, including when no waistband is present. For GCD-Verse, we omit this joining seam when the waistband is absent, allowing the two garments to remain structurally separate. Their internal panel seams are preserved, and their initial placements still follow the body-dependent pattern construction. Garments connected through a waistband retain the original assembly procedure.

Persistent waist support. Separating the garments removes the support that the upper garment would otherwise provide to the lower garment. We constrain the upper boundary of the lower garment using the waist height obtained from the body measurements. The underlying GCD attachment mechanism applies a compliant, one-sided constraint:

$$c_{\text{waist}}(\mathbf{x}) = \mathbf{e}_y^\top \mathbf{x} - h_{\text{waist}} \geq 0, \tag{10}$$

where $\mathbf{x}$ is an attached mesh vertex, $\mathbf{e}_y$ is the upward unit vector, and h_{waist} is the target waist height. The solver applies a restoring correction when a vertex falls below this height, while allowing lateral motion and motion above it. GCD already provides waist, collar, and strapless-top attachments, but releases them after 400 frames under its default configuration. We keep the waist attachment active throughout simulation and release the other attachments after 400 frames.

Staged draping. For outfits containing separate upper and lower garments, we follow the staged draping strategy of FIT Wang et al. (2026b). We use the pattern's panel labels to identify the lower garment and simulate it first. We then hold the draped lower mesh fixed and add it as a collision object when simulating the upper garment against the body. The resulting meshes are combined while preserving the separation between garments. This is a sequential approximation: the lower garment does not deform in response to the upper garment during the second stage. Other garments are simulated as a single assembly.

Solver settings and acceptance. We use a nominal mesh spacing of $1\,\text{cm}$ and advance each stage at 60 frames per second with 10 substeps per frame. Gravity is disabled for the first 10 frames, and each stage is limited to 2,400 frames. After the attachment-release period, a stage is considered static when fewer than 1.5% of its vertices move by more than $0.03\,\text{cm}$ in per-frame ℓ_1 displacement. Meshing failures, timeouts, failure to settle, and excessive intersections cause rejection; a failure in either stage rejects the outfit.

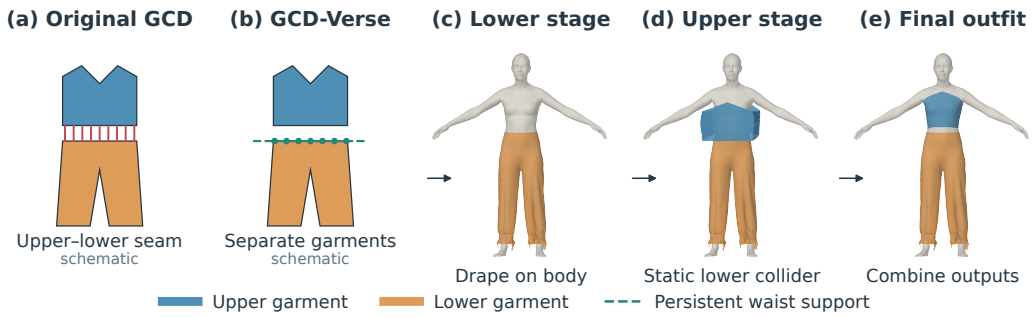

Figure 10: **Simulation of separate upper and lower garments.** (a–b) Schematics of composition without a waistband: GCD joins the garments, whereas GCD-Verse keeps them separate and retains waist support. (c) The lower garment is draped first. (d) Its final mesh serves as a static collider while the upper garment is simulated. (e) The stage outputs form the final outfit. Views (c–e) are reconstructed from saved initial and final meshes; view (d) shows the upper garment before its simulation.

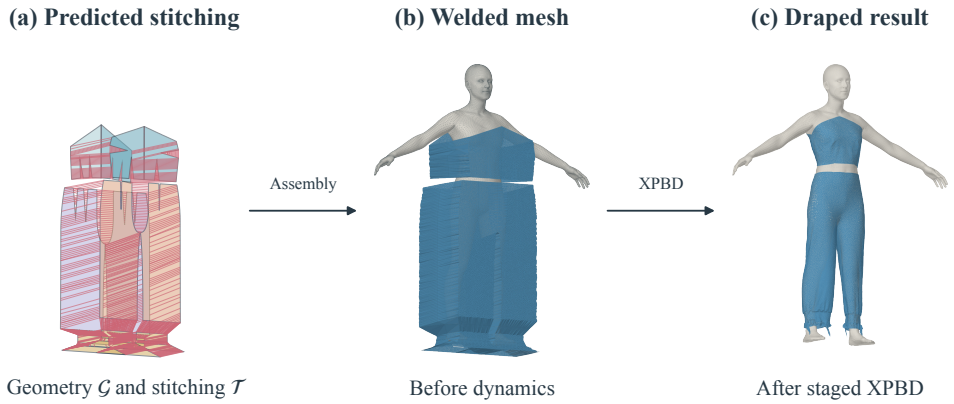

Figure 11: **Simulation of predicted sewing patterns.** (a) Decoded panels in their predicted 3D layout, with predicted point-level stitch connections shown in red. (b) Triangulated and welded cloth before dynamics. (c) Draped result after staged XPBD simulation. All three views use the same saved prediction.

C.2 Simulation of Predicted Sewing Patterns

The output of **SartorAI** consists of decoded panel geometry $\mathcal{G}$ and point-level stitching relationships $\mathcal{T}$. We convert these predictions into a triangulated cloth mesh using the predicted panel names and an independently supplied body mesh and measurements (Figure 11).

Mesh assembly. We retain the dense decoded panel contours and use GCD's constrained Delaunay triangulation to mesh their interiors in local 2D coordinates at a nominal spacing of $1\,\mathrm{cm}$. Paired seam boundaries are sampled with matching vertex counts. The predicted layout places the panels in 3D, and corresponding seam vertices are welded to assemble the cloth. The planar panel meshes provide its reference lengths.

Support constraints. Predicted panels lack the attachment labels supplied by procedural pattern construction. We infer them from panel names, boundary orientation, and predicted connectivity. Upward-facing runs on waistband panels, or on the uppermost skirt or trouser panels when no waistband exists, provide waist-support candidates. Shoulder connections between front and back torso panels distinguish supported tops from strapless tops: we select collar boundaries in the former case

and free upper boundaries in the latter. The selected boundary locations are mapped by arc length to the resampled mesh vertices, with constraint targets determined by the supplied body measurements. We apply the one-sided waist constraint in Equation 10 throughout simulation, while collar and strapless-top attachments are released after 400 frames.

Staged simulation. Panel names also determine whether staged simulation applies. When upper and lower garments are both present without a waistband, we simulate them separately and discard predicted cross-stage stitch pairs before welding, following the two-piece convention of GCD-Verse. The lower garment is draped first and held fixed as an additional collider while the upper garment is simulated against it and the body. Other garments are simulated as a single assembly. All stages use the XPBD backend described in Appendix C.1.

D More Implementation Details on SartorAI

We provide the network architectures, preprocessing conventions, and training settings for the geometry tokenizers and topology models. All training is performed on NVIDIA H200 GPUs using PyTorch. The two geometry RQ-VAEs are trained independently, and the topology VAE is trained before the conditional flow model. Once trained, the geometry tokenizers are fixed; the topology VAE is likewise frozen during flow training.

D.1 Geometric Preprocessing

Panel coordinates. We express the original panel coordinates and translations in centimeters and represent rotations by three Euler angles in degrees. For shape encoding, each panel is centered and scaled into the canonical square $[-1, 1]^2$. The ten candidate scale factors are $\{0.004, 0.006, 0.009, 0.0135, 0.02, 0.03, 0.045, 0.0675, 0.1, 0.15\}$. We choose the largest factor that fits the panel within the square, with a small boundary margin. Centering the panel changes its local origin, so we adjust its translation to preserve the original placement in 3D. With this convention, a normalized boundary point $\mathbf{u} = (u_x, u_y)$ is placed in 3D as

$$\mathbf{v} = R(\mathbf{r}_i) \begin{bmatrix} u_x/\alpha_i & u_y/\alpha_i & 0 \end{bmatrix}^\top + \mathbf{t}_i, \tag{11}$$

where $R(\mathbf{r}_i) = R_z(r_{i,z})R_y(r_{i,y})R_x(r_{i,x})$ is the rotation matrix associated with the panel's Euler angles. Thus, α_i denotes the normalization scale, whose inverse restores physical dimensions. Panel sequences follow the position-based ordering described in Section 3.2.

Topology coordinates. For topology prediction, we represent each placed panel by 256 points in cyclic boundary order. We center the complete garment at the midpoint of its 3D bounding box and divide all coordinates by half the longest box extent. This common transformation places the garment within $[-1, 1]^3$ while preserving relative panel positions and proportions. Panel membership is retained throughout the topology models. In a batch containing garments with different numbers of panels, attention across panels and pairwise stitching supervision are restricted to points from the same garment.

D.2 RT RQ-VAE

The RT encoder takes the six-dimensional vector $[\mathbf{t}_i; \mathbf{r}_i]$ and projects it to a hidden dimension of 512. Four residual MLP blocks then map it to a 256-dimensional latent vector through a final linear projection. Each residual block contains two linear layers with layer normalization and GELU activations. The decoder mirrors this architecture, mapping the quantized latent back to three translation coordinates and three Euler angles.

We use three residual quantization stages with a shared codebook of 1,024 entries, each of dimension 256. Each panel therefore has three RT tokens. At each stage, the selected codeword approximates the residual left by the preceding stages; their sum forms the quantized latent used by the decoder. The codebook is updated by an exponential moving average (EMA) with decay 0.99. We set $\beta_{\mathrm{RT}} = 0.25$ in Equation 3, with equal weights for the translation and rotation MSE terms.

D.3 SHAPE RQ-VAE

Boundary encoding and quantization. The encoder processes 256 samples from each normalized panel boundary using a feature dimension of 384. For each 2D coordinate, we apply sine and cosine functions at frequencies $2^k\pi$, $k = 0, \ldots, 11$, independently to both coordinate components. This gives 48 Fourier features, which are concatenated with the two original coordinates and linearly projected to 384 dimensions. Four learnable queries aggregate the boundary features through cross-attention. The latent encoder contains four blocks combining cross-attention to the boundary features with self-attention among the four latent vectors. Cross-attention uses one head of dimension 384, while latent self-attention uses eight heads of dimension 64 and projects its output back to 384 dimensions.

Each of the four latent vectors is quantized through four residual stages using a shared codebook with 4,096 entries of dimension 384. This produces 16 shape tokens per panel, serialized by quantization stage as described in the main text. The shape codebook is separate from the RT codebook and uses an EMA decay of 0.9. We set $\beta_{\text{shape}} = 0.25$ in Equation 4 and use straight-through gradients for the encoder.

Implicit decoding. For each latent position, the decoder sums the four selected codewords and refines the resulting four quantized vectors with 16 self-attention blocks. The refined vectors are concatenated into a 1,536-dimensional conditioning vector for the occupancy decoder. The decoder is a residual MLP with six blocks of width 512. It takes Fourier-embedded 2D query coordinates as input, and the conditioning vector provides learned scale and shift parameters for the normalized hidden features. A final linear layer predicts one occupancy logit per query point.

During training, we supervise 8,192 query points per panel, covering the canonical domain, the panel interior, and regions near its boundary. The occupancy objective uses weighted BCE with logits, with the positive-class weight capped at 10 to limit the effect of class imbalance. At inference, we evaluate the occupancy field on a 512×512 grid and extract the contour at occupancy probability 0.5. We retain the largest enclosed contour and orient it counterclockwise. The recovered boundary is placed in 3D using Equation 11 and resampled uniformly by arc length to obtain the 256 points used for topology prediction. We also retain the dense contour for the subsequent meshing and simulation procedure in Appendix C.

D.4 VLM

Backbone and training examples. We initialize from Qwen3-VL-8B-Instruct Bai et al. (2025) and extend its vocabulary with 5,138 tokens: 4,096 shape codes, 1,024 RT codes, ten scale values, and eight structural delimiters. The added input embeddings and output-projection rows are initialized from $\mathcal{N}(0, 0.02^2)$. Images are resized while preserving their aspect ratio, with a maximum pixel budget of 768^2. Inputs follow the backbone's chat template.

Garments are shuffled without replacement each epoch, with task assignments and instruction variants resampled per garment. Body measurements are appended with probability 0.5 for the three text-generation tasks. Editing inputs contain the source garment tokens and the requested change. Only the complete target garment sequence and the end-of-sequence token receive autoregressive cross-entropy supervision; conditioning inputs are excluded.

Optimization and decoding. We train on eight H200 GPUs with 16 garments per GPU, giving a global batch size of 128. We use fused AdamW with weight decay 0.01 and a global gradient-norm clipping threshold of 1.0. Our training consists of two stages. In Stage 1, we train for 1 epoch and only update the newly added embeddings and the weights of the LM head. In Stage 2, we train for 11 epochs and update all weights except those of the vision module. We use bfloat16, gradient checkpointing, scaled dot-product attention, DeepSpeed ZeRO-2, and random seed 42. Generation uses greedy decoding with at most 2,048 new tokens.

D.5 TOPOLOGY VAE

Encoder and decoder. The topology VAE uses a hidden dimension of 512, eight attention heads, and feed-forward layers of width 2,048. The encoder first projects the 3D boundary coordinates to hidden features. It then interleaves four GraphSAGE layers with four transformer blocks that attend

within individual panels. GraphSAGE uses mean aggregation over an undirected graph containing cyclic boundary adjacency and ground-truth stitching edges, with residual updates, layer normalization, and GELU activations. These two edge types support feature propagation; only stitching edges serve as positive targets for topology reconstruction.

After the local blocks, we average the point features within each panel to obtain one panel feature. Two transformer blocks exchange information among these pooled features, and the resulting panel features are broadcast and added to the corresponding point features. Separate linear heads predict the mean and log variance of a 64-dimensional diagonal Gaussian posterior at each point. We sample the latents using the reparameterization trick.

The decoder projects each 64-dimensional latent to width 512 and applies four within-panel transformer blocks followed by two blocks over pooled panel features. Its final projection produces a 512-dimensional feature $\mathbf{z_v}$, so $c = 512$ in Equation 6. The spatial and temporal components each have 256 dimensions. The decoder does not require stitching edges as input.

Reconstruction and regularization. We use $\tau = 0$ for the spacetime criterion. During training, this threshold determines the predicted edge labels used to form the TP, TN, FP, and FN groups, while BCE is evaluated on the continuous logits as in Equation 7. All unordered pairs of distinct points within each garment are supervised, including pairs within the same panel, which can represent darts. Pairwise losses are evaluated in chunks to avoid materializing the complete pairwise feature tensor. We linearly increase the KL coefficient to $\beta_{\mathrm{KL}} = 10^{-4}$ over the first 1,000 optimization steps and keep it fixed thereafter. The KL term is averaged over points and latent dimensions, following Equation 8.

D.6 Conditional Flow

Velocity predictor. The flow model operates on the 64-dimensional topology latents at each boundary point. We concatenate the interpolated latent, its three normalized coordinates, and a 256-dimensional timestep embedding, then project the result to a hidden dimension of 2,048. The timestep embedding is obtained from sinusoidal features followed by an MLP. The transformer contains 16 blocks that attend within panels and two blocks that attend over mean-pooled panel features, using 16 attention heads and feed-forward layers of width 8,192. As in the VAE, the updated panel features are broadcast back to their constituent points.

Timestep embeddings also modulate the attention and feed-forward branches through scale, shift, and residual-gating parameters. We apply RMS normalization to attention queries and keys and use 3D RoPE to incorporate positions. Within-panel attention uses individual point coordinates; attention over pooled panel features uses the mean coordinates of each panel.

Training and sampling. At each training step, we sample target latents from the frozen VAE posterior and independently draw Gaussian noise of the same shape. One timestep is sampled uniformly for each garment and shared by all its boundary points. We form the interpolated latents and optimize the velocity MSE in Equation 9. An EMA of the flow parameters, with decay 0.9999, is maintained for inference.

We use the second-order multistep DPM-Solver++ sampler with 20 updates. The sampled latents are passed through the frozen VAE decoder, and point pairs with positive spacetime logits are classified as stitching edges.

E Limitations and Future Work

Despite promising results, our work remains limited by the diversity and realism of the training data. Expanding the dataset to include more varied human poses, textured human bodies, more realistic garment textures, and a broader range of garment categories would be a valuable next step. We expect greater data diversity to improve the model's generalization and generation quality.

Several directions also warrant further investigation. Our framework could be extended beyond garments to a broader range of sewn objects, including plush toys, bags, shoes, and suitcases. Another direction is to incorporate panel manufacturability and production costs into pattern design and generation, accounting for practical cutting and assembly constraints. Finally, integrating advances in

AI agents and frontier language models could enable more capable and interactive workflows for sewing pattern design and generation.

F DISCUSSION

Commercial potential. By turning natural-language instructions and reference images into editable sewing patterns, **SartorAI** could lower the barrier to initial pattern creation and help designers explore and refine alternatives. Potential applications include personalized clothing, product development for small fashion brands, and garment assets for games and films. More efficient digital iteration may shorten development cycles and reduce early sampling costs, although these benefits require evaluation in practical design workflows.

Bias and inclusivity. Uneven coverage of body shapes, garment styles, and cultural contexts may lead to differences in how well the system serves different users. Pretrained VLMs and automatically generated annotations may also carry gender stereotypes or culturally specific assumptions about what constitutes appropriate attire. Evaluating performance across body shapes and cultural contexts, together with feedback from affected users, would help identify such biases and support more inclusive design.

Privacy of body data. Personalized applications may process user-provided body photographs and measurements, which can reveal personal information about identity, appearance, and body characteristics. Deployment should clearly communicate how these data are collected and used, minimize their collection and retention, and protect them through access controls and safeguards for transmission and storage. Reusing these inputs for model training or other purposes should require separate user consent, and users should have a means to delete their stored inputs.