SPEND BITS WHERE THE PATTERN BENDS: CORNER-BASED CODING OF SEWING-PATTERN BOUNDARIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Learned models increasingly read and write sewing patterns, and every panel passes through them as a coded boundary that pattern software must accept back. Current boundary codes cut the panel into equal pieces and give each the same number of bits. A straight seam then costs as much as an armhole, and the corner between them gets rounded off. We observe that most of the error sits in a few short curves, and that the corners and straight seams that make up most of a panel are charged like curves. To this end, we propose CornerMCQ, a boundary code that spends bits only where the boundary bends and needs no neural network. Specifically, it stores the detected corners on a vertex grid, gives the straight runs between them no curve bits, and describes each remaining curve relative to its chord with a few small codebooks fitted once. On a public corpus and a private industrial archive, CornerMCQ is more accurate than Bézier fitting, PCA, and a learned vector quantizer at equal bits and several times more accurate than the same codebooks on equal pieces; on the public corpus, with fewer bits than the equal-piece code, three panels in four stay within two millimetres of the reference boundary everywhere, against one panel in a hundred.

1 INTRODUCTION

Sewing patterns are the working documents of garment manufacturing: flat panels whose outer boundaries are cut from fabric, graded across sizes, and exchanged between pattern-design systems as CAD files. Learned models increasingly read and write such patterns (Korosteleva & Sorkine-Hornung, 2023; He et al., 2024; Nakayama et al., 2025; Weng et al., 2026), and every panel passes through them as a coded outer boundary. We study this code: how to turn one closed panel boundary into a short sequence of discrete symbols from which pattern software gets the boundary back.

Coding a panel boundary is demanding for three reasons. First, its information is unevenly spread: a corner is a construction point that must stay where it is, a straight seam carries almost none, and a few curves carry most of it. Second, the output must come back usable: a closed polyline in millimeters that a CAD parser accepts is necessary, and the corners must still be in place and the straight seams straight. Third, the cost should follow the geometry: a rectangle should not pay as much as an armhole. Fixed-rate codes address neither the first nor the third requirement. A quantizer that resamples every boundary to the same number of positions and spends the same number of indices on each stretch pays for straight edges, smooths corners that fall between supports, and leaves ripples that a pattern maker has to redraw; global bases such as PCA or elliptic Fourier descriptors (Jolliffe, 2002; Kuhl & Giardina, 1982) allocate in the same way. Section 5.4 measures the second for every codec. Classical shape coding places vertices at dominant points and codes the pieces between them (Ramer, 1972; Teh & Chin, 1989; Schuster & Katsaggelos, 1998; Katsaggelos et al., 1998), but with a generic parametric fit that knows nothing about the corpus. Learned tokenizers know the corpus but quantize fixed latent grids (van den Oord et al., 2017; Lee et al., 2022), and SVG language models emit grid coordinates, not curve shapes (Yang et al., 2025; Xing et al., 2026).

Our observation is that the two halves combine. Once the corners are placed and the straight runs are marked, what remains of a panel is a handful of curved segments. A corpus of patterns repeats the same few curve shapes—armholes, necklines, hems—at different positions, orientations, and

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.

sizes, and a small codebook can learn them. We build CornerMCQ on this observation (Figure 1). A deterministic, corner-adaptive partition places explicit vertices at corners and marks straight runs, which cost zero curve bits. Every remaining curved segment is mapped into its chord frame, which removes position, orientation, and length. A few sequential residual K-means codebooks, fitted on the curved segments of a training corpus, code what is left. The code spends bits only on curvature, and its rate grows with the number of curves rather than with the boundary length. Codebooks are fitted in seconds without a neural network and occupy well under a megabyte; because every level refines the previous one, one fit yields a whole ladder of rates.

We evaluate on two domains. The public GarmentCodeData corpus (Korosteleva et al., 2024) makes every number reproducible; a private archive of industrial production patterns, reported in aggregate only, shows that the results hold on real patterns. All design decisions were made on GarmentCode validation, and every operating point was frozen before one evaluation on each held-out test split (Section 4). On GarmentCode, at 397 bits per panel CornerMCQ reaches 0.292 mm held-out Chamfer, against 1.625 mm for the same kind of residual codebooks on fixed supports and 1.907 mm for PCA, both at 480 bits. On the industrial archive it reaches 0.303 mm at 322 bits, against 1.173 and 1.359 mm. With the partition and the bits per segment held fixed, the codebooks beat a Bézier coder, PCA with scalar quantization, and a tuned learned residual VQ-VAE at 16 of 16 rate points on GarmentCode and 16 of 16 on the industrial archive. With the coder held fixed, a uniform partition costs 13× and 3×. Codebooks fitted on the public corpus code held-out industrial panels with a 2–4% loss on the finest partition and still beat every control fitted on industrial data at 15 of 16 points. A common 512-position rendering of all 9 audited codecs passes the write–parse–remeasure audit on every held-out panel of both domains with 0.0 mm readback.

In summary, our main contributions are:

- We present CornerMCQ, a corner-explicit, variable-rate boundary code: residual K-means codebooks on geometry-defined chord-frame segments, with an explicit bit ledger and a self-delimiting token sequence.
- We run a controlled comparison on public GarmentCode data and a private industrial archive in which every control shares all components but one: the same partition and bits with Bézier, PCA+SQ, and a tuned learned residual VQ-VAE; the same quantizer on fixed supports; and partition ablations.
- We apply a CAD return audit identically to every method. It separates parser acceptance, coordinate readback, end-to-end error, and polygon validity.

2 Related Work

Sewing-pattern representations. This line generates or edits complete sewing patterns. GarmentCode and GarmentCodeData define parametric pattern programs and a large public corpus of them (Korosteleva & Sorkine-Hornung, 2023; Korosteleva et al., 2024). NeuralTailor and SewFormer regress straight and Bézier edge parameters (Korosteleva & Lee, 2022; Liu et al., 2023), DressCode quantizes them uniformly into tokens (He et al., 2024), GarmentDiffusion diffuses them (Li et al., 2025), and SewingLDM, AIpparel and GarmentGPT compress them into quantized latents (Liu et al., 2025; Nakayama et al., 2025; Weng et al., 2026); ChatGarment and Design2GarmentCode generate GarmentCode programs instead (Bian et al., 2025; Zhou et al., 2025). These systems produce whole garments and assume program- or Bézier-level edges. CornerMCQ addresses the complementary step: encoding an arbitrary supplied outer boundary, such as a digitized or exported pattern piece, at a controlled bit budget, with explicit corners and a verified CAD return.

Discrete tokens for shapes, vector graphics, and CAD. Vector quantization turns continuous signals into index sequences for generative models (Gray, 1984; van den Oord et al., 2017; Razavi et al., 2019; Esser et al., 2021; Yu et al., 2022). Multi-stage residual quantization, in which each codebook encodes the error left by the previous ones, dates back to speech coding (Juang & Gray, 1982), underlies additive and product quantizers (Jégou et al., 2011; Babenko & Lempitsky, 2014), and drives SoundStream, EnCodec and RQ-VAE (Zeghidour et al., 2022; Défossez et al., 2023; Lee et al., 2022). For geometry, Sketch-RNN, DeepSVG, IconShop and StrokeNUWA tokenize pen offsets, SVG commands or strokes (Ha & Eck, 2018; Carlier et al., 2020; Reddy et al., 2021; Wu

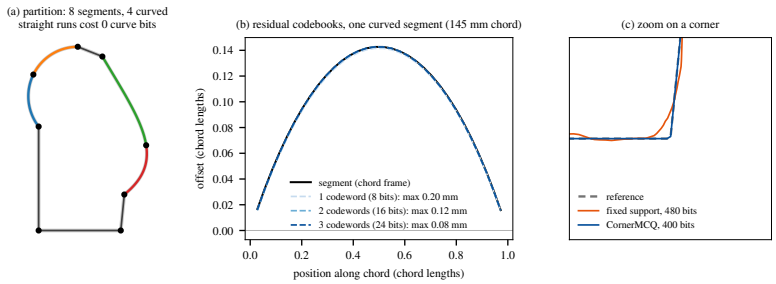


Figure 1: CornerMCQ on a GarmentCode validation panel (selected by a fixed rule: median fixed-support error among panels with at least three curved and two straight segments). (a) Corner-adaptive Coarse partition: transmitted vertices (dots), straight runs (gray), curved segments (color). (b) One curved segment in its chord frame and its reconstruction from one to three residual codewords (8 bits each); the legend gives the largest deviation in millimeters on this 145 mm chord. (c) Zoom on a corner: the fixed-support quantizer at 480 bits rounds it and ripples along the straight edge; CornerMCQ at 400 bits keeps both.

et al., 2023; Tang et al., 2024), OmniSVG and HiVG emit variable-length command and coordinate sequences (Yang et al., 2025; Xing et al., 2026), DeepCAD, Vitruvion, SkexGen, HNC-CAD and Text2CAD emit parametric sketches and construction sequences (Willis et al., 2021; Ganin et al., 2021; Wu et al., 2021; Seff et al., 2022; Xu et al., 2022; 2023; Khan et al., 2024). These tokenizers quantize fixed grids of latent positions or coordinates and are judged by what a generator produces from them; none reports the rate–distortion behaviour of re-encoding a given boundary, and none audits whether a decoded contour survives a CAD exchange file. CornerMCQ keeps the residual codebooks but moves them onto geometry-defined supports whose number varies per shape, and treats CAD return as a measured outcome.

Classical contour coding. Dominant-point detection and polygonal approximation (Ramer, 1972; Douglas & Peucker, 1973; Teh & Chin, 1989), piecewise cubic curve fitting (Plass & Stone, 1983; Schneider, 1990), and the rate–distortion-optimal vertex-based and B-spline shape coders of MPEG-4 (Katsaggelos et al., 1998; Schuster & Katsaggelos, 1998; Meier et al., 2000) place bits at shape-dependent vertices and treat straight runs cheaply. Global descriptors such as PCA and elliptic Fourier coefficients (Jolliffe, 2002; Kuhl & Giardina, 1982) give fixed rates but smear corners. These coders describe each piece with a generic parametric fit that knows nothing about the corpus. CornerMCQ inherits the vertex-based view and replaces the per-segment fit with residual codebooks learned from a corpus of pattern curves; its Bézier and PCA+SQ controls are the classical counterparts on the same partition.

3 Method

Given one closed panel boundary, CornerMCQ produces a variable-length but self-delimiting code: a corner-adaptive partition places the vertices, chord-frame residual codebooks code the curved segments between them, and a bit ledger and a CAD audit account for what the code costs and what comes back (Figure 1). Codebooks are fitted once per corpus on training segments and then frozen.

Input. A panel boundary is a closed polyline, arc-length resampled to N=512 samples $P = (p_0, \ldots, p_{N-1})$ in millimeters (Section 4).

3.1 Corner-adaptive partition: vertices where the boundary bends

Corners. For each sample, the turning angle $\alpha_i = \angle\big(p_i - p_{i-w},\, p_{i+w} - p_i\big)$ is computed with w=3 and cyclic indices. Corners are the local maxima over $\pm w$ with $|\alpha_i| > \theta$=20° (cf. Teh & Chin, 1989). Each corner is refined to the intersection of two lines, fitted by total least squares to the w samples on either side; the refinement is kept only if it lies within half a sample spacing of the detected sample.

A boundary with fewer than two corners uses its first and farthest samples. No GarmentCode panel needs this fallback, and fewer than 1% of industrial panels do (Appendix B).

Straight and curved runs. Between consecutive corners, a run is *straight* if every interior sample lies within τ_s=0.3 mm of the chord. A straight run is represented by its two endpoints only and costs no curve bits. Any other run is *curved*, and may be split at the sample of maximum chord deviation while that deviation exceeds τ_c (in the spirit of Ramer, 1972; Douglas & Peucker, 1973). We report four partitions that differ only in this split rule: Corner (τ_c=∞, split at corners only), Coarse (τ_c=25 mm), Mid (τ_c=10 mm), and Fine (split while an endpoint-constrained cubic Bézier misses the run by more than 1 mm). Splitting stops at depth 8 or at runs of fewer than six samples. θ, τ_s, the vertex precision, and the 1 mm Bézier tolerance come from an earlier prototype's sweep on 500 training panels; the two deviation thresholds were fixed a priori; none was tuned on validation (Appendix B).

Vertices. The frame is three float32 values (96 bits): the origin of the panel bounding box and a single extent E, its longer side. Segment endpoints are rounded to the isotropic grid of step $E/(2^{12}-1)$ and charged 12 bits per coordinate. A refined corner can lie just outside the box; its index is then transmitted unclipped, at a cost given with the ledger below. The partition is computed on the unquantized boundary. Chord frames, residuals, and the codeword search use the quantized vertices that the decoder receives (Appendix B).

3.2 Chord-frame residual codebooks: coding what is left

Chord frame. A curved segment from vertex A to vertex B is described in its chord frame. Let $\ell = \|B-A\|$ be the chord length, $e_x = (B-A)/\ell$ the chord direction and $e_y = (-e_{x,2}, e_{x,1})$ its counter-clockwise normal. The run is resampled to M=32 arc-length-uniform interior points $q_1, \ldots, q_M$, and its descriptor is

$$r_k = \left(\frac{(q_k - A)^\top e_x}{\ell} - \frac{k}{M+1}, \frac{(q_k - A)^\top e_y}{\ell} \right), \qquad k = 1, \ldots, M, \qquad r = (r_1, \ldots, r_M) \in \mathbb{R}^{2M}, \tag{1}$$

where $k/(M+1)$ is the position the k-th point would have on the chord itself, so a straight run maps to $r = 0$. The chord frame removes position, orientation, and length, so an armhole, a neckline, or a hem curve from any panel lands in one shared space; what remains is shape.

Codebooks. On all training segments we fit L residual codebooks $C_1, \ldots, C_L$ of K=256 codewords each, one level at a time. C_λ is the K-means solution (Lloyd, 1982) on the residuals left by the previous levels, with seeded sample initialization, at most 40 Lloyd iterations, and dead centroids reseeded at high-error rows. Encoding is greedy over levels,

$$i_\lambda = \arg\min_k \left\| r^{(\lambda-1)} - C_\lambda[k] \right\|_2, \qquad r^{(\lambda)} = r^{(\lambda-1)} - C_\lambda[i_\lambda], \qquad r^{(0)} = r, \qquad \lambda = 1, \ldots, L, \tag{2}$$

where $C_\lambda[k]$ is the k-th codeword of level λ and $r^{(\lambda)}$ the residual after that level. The decoder sums the selected codewords $\sum_\lambda C_\lambda[i_\lambda]$ and inverts Eq. (1). Because each level is fitted on the residual of the previous ones, the first λ levels of an L-level fit are the λ-level fit: one fit yields the whole rate ladder, and a code can be truncated to lower precision without re-encoding. The fit minimizes squared error in chord-normalized units; errors are reported in millimeters.

3.3 Rate, state, and token sequence

Bit ledger and state. For a panel with S segments, C of them curved, the bit cost without entropy coding is

$$R = 96 + 8 + S(2 \cdot 12 + 1) + CL \log_2 K, \tag{3}$$

where the frame takes 96 bits and the segment count 8 bits, each of the S segments carries a start vertex of two 12-bit coordinates and a one-bit type, and each of the C curved segments carries L indices of $\log_2 K$ bits. The ledger charges 12 bits per coordinate; charging 13, so that the field also holds the refined corners outside the box, would add $2S$ bits per panel, 397 to 416 at Coarse L=2 on GarmentCode and 322 to 334 on the industrial archive, still below 480 (Appendix B); all rates here use the 12-bit ledger. The reusable state is $L \cdot K \cdot 2M$ float32 values, 196,608 bytes (0.19 MiB) for L=3. It is fitted once per domain, shared by every panel, and not part of the per-panel payload; Appendix H lists the state of every codec.

Token sequence. Read left to right, the code is a self-delimiting token sequence: one token for the segment count, then one record per segment. A record holds the two coordinates and the type of the segment's start vertex and, for a curved segment only, its L codebook indices. A segment ends at the start vertex of the next record, and the last segment closes on the first. The vocabulary has 2^{12} coordinate symbols, a few more for indices just outside the box (Appendix B), two type symbols and one symbol per codeword of each level. This sequence is the interface a sequence model would consume; we evaluate the code, not a model trained on it.

3.4 Controls that share every component but one

Same partition, different coder. To isolate the curve coder, three controls use the same partition, the same quantized vertices, and exactly the same bits per curved segment, and differ only in how a curved segment is coded. An endpoint-constrained least-squares cubic Bézier (Plass & Stone, 1983; Schneider, 1990) scalar-quantizes its two inner control points either on the panel grid or in the chord frame. PCA+SQ takes principal components of r with uniform scalar quantization of the coefficients. A learned residual VQ-VAE on r uses an MLP or 1-D convolutional encoder/decoder and EMA residual codebooks (van den Oord et al., 2017; Lee et al., 2022; Zeghidour et al., 2022).

Two further controls change the support. The fixed-support residual multi-codebook quantizer applies the same kind of sequential K-means codebooks to eight fixed quarter-length single-coordinate blocks of the 512-sample boundary (48 indices at 384 shape bits). Whole-boundary PCA uses 8-bit coefficients (Jolliffe, 2002). Within CornerMCQ's own pipeline, a uniform partition into S equal arc-length segments (no corners, no straight runs) and a variant without zero-bit straight runs ablate the partition.

3.5 CAD return contract: write, parse, remeasure

Audit. A reconstruction is only useful if pattern software can open it. We therefore adopt a write–parse–remeasure audit and apply it identically to every method. The decoded boundary is rendered at 512 positions, starting at the reference's first sample and placed at the panel's original position; neither is transmitted, and both are the same for every codec. The rendering is written as an ASCII DXF R2010 file in millimeters containing one closed LWPOLYLINE entity with 513 coordinate records (explicit repeated endpoint), one anonymous TEXT entity, and no bulges. It is read back by a locked raw group-code parser and by the external ezdxf library (Moitzi), which must agree coordinate for coordinate.

The audit reports four separate outcomes: parser acceptance, decoded-versus-parsed coordinate discrepancy, end-to-end error between the original and the parsed boundary, and simple-polygon validity. Acceptance does not certify accuracy and a valid polygon does not certify acceptance, so none of the four is folded into another. A uniform rendering does not pass through CornerMCQ's vertices in general, so Appendix K also audits the native decode: a variable-length polyline that contains every transmitted vertex.

4 Experimental protocol: two corpora, one frozen release per domain

Corpora. We use two domains. The first is the public GarmentCodeData corpus (Korosteleva et al., 2024): 11,757 outer panel boundaries resampled to 512 arc-length positions, split by duplicate component into 9,405 training, 1,176 validation, and 1,176 test panels. The second is a private archive of industrial production patterns: 16,227 outer panel boundaries, prepared and split in the same way into 12,981, 1,623, and 1,623 panels. The archive cannot be released, so we report aggregate statistics only and show no industrial geometry (Appendix A). All codebooks, bases, and networks are fitted on the training split of the domain they are evaluated on, except in the transfer experiment of Section 5.3.

Development on validation. Every choice—partitions, codebook sizes, baseline variants, and the learned-model grid—was made on GarmentCode validation. The industrial domain came second; nothing of CornerMCQ was re-tuned for it, and only the baseline variants and learned configurations were selected again, on industrial validation, by the same rules.

	GarmentCode held-out test (1,176 panels)					Industrial held-out test (1,623 panels)				
Codec	Bits	Chamfer	Curve	Worst	$\leq$2 mm	Bits	Chamfer	Curve	Worst	$\leq$2 mm
Whole-boundary PCA	480	1.907	1.550	11.86	0.00	480	1.359	1.107	7.61	0.07
Fixed-support codebooks, L=6	480	1.625	1.311	9.83	0.01	480	1.173	0.964	6.88	0.14
Fixed-support codebooks, L=12	864	0.983	0.728	6.43	0.19	864	0.685	0.526	4.39	0.35
CornerMCQ Corner, L=3	349	0.519	0.311	2.57	0.63	297	0.655	0.487	2.88	0.60
CornerMCQ Coarse, L=2	397	0.292	0.121	1.67	0.75	322	0.303	0.180	1.36	0.83
CornerMCQ Coarse, L=3	428	0.248	0.083	1.46	0.82	353	0.230	0.114	1.09	0.90
CornerMCQ Mid, L=3	507	0.206	0.048	1.29	0.88	431	0.161	0.059	0.85	0.96
CornerMCQ Fine, L=4	794	0.174	0.035	1.21	0.90	761	0.121	0.039	0.76	0.99

Table 1: Adapting the support, on the held-out split of both domains. Bits: mean total bits per panel on the 12-bit ledger of Eq. (3), frame included (Appendix B gives the 13-bit alternative). Chamfer: 512-sample symmetric Chamfer. Curve: mean symmetric point-to-curve distance. Worst: mean per-panel worst point-to-curve distance. $\leq$2 mm: fraction of panels whose worst point is within 2 mm. Distances in mm; seeded rows are five-seed means (validation and SD in Appendix C).

Frozen held-out release. We then froze 79 operating points, their model files, the metrics, the paired comparisons, and three hypotheses under a written protocol, and evaluated them once on the GarmentCode test split with a single-use release script (Appendix A). A second protocol was frozen and released once on the industrial test split. Both test splits had been used once before, for an earlier fixed-support variant, and the GarmentCode split once more in an exploratory analysis of it (Appendix A); no decision about CornerMCQ was made on test data. Every held-out number comes from these two runs, and every outcome is reported (Nosek et al., 2018).

Metrics. Chamfer distance on 512 samples is the metric of prior boundary-codec work, but it confounds geometry with where samples fall along the curve; for segment-based decoders we align the evaluation resampling to the reference start sample (not transmitted, identical for all segment coders). Our primary metric is therefore phase-free: the symmetric mean distance from reference samples to the decoded polyline and from decoded samples to the reference polyline (*curve distance*). It is measured on each codec's own decoded polyline. For CornerMCQ and the same-partition controls this is the vertices plus the decoded curve points, densified to 2,048 samples that contain every vertex. Chamfer, polygon validity, and the CAD audit use a common 512-position rendering of every decode. Because a pattern is only as good as its worst seam, we also report the per-panel worst point-to-curve distance and the fraction of panels whose worst point lies within 1 or 2 mm, simple-polygon validity, and the mean bits per panel of Eq. (3), frame included. Seeded methods report the mean over five fitting seeds; paired differences use a percentile bootstrap over duplicate components (10,000 replicates) and are reported as 95% intervals.

5 Results

5.1 Support: corner-adaptive against fixed and whole-boundary codes

GarmentCode. At a lower rate than the two 480-bit codes, CornerMCQ's error is several times lower (Table 1). On the held-out split, CornerMCQ with the Coarse partition and two codewords per curve spends 397 bits per panel, fewer than the 480 bits of the fixed-support quantizer and of whole-boundary PCA, and reaches 0.292 mm Chamfer against 1.625 and 1.907 mm, 5.6$\times$ and 6.5$\times$ less. The worst point of a panel falls from 9.83 mm to 1.67 mm on average (5.9$\times$), and the share of panels whose worst point stays within 2 mm rises from 0.01 to 0.75. Paired over duplicate components, the Chamfer advantage is 1.333 mm over the fixed-support codec and 1.614 mm over PCA, with bootstrap lower bounds of 1.255 and 1.548 mm. Simple-polygon validity is 0.945 for CornerMCQ and 0.799 for the fixed-support decode; the references themselves are at 0.944.

Industrial archive. The same picture holds on real patterns. CornerMCQ Coarse with two codewords spends 322 bits and reaches 0.303 mm Chamfer; the two 480-bit codecs reach 1.173 and 1.359 mm, 3.9$\times$ and 4.5$\times$ more. The share of panels whose worst point stays within 2 mm rises from 0.14 to 0.83, and the paired Chamfer advantage over the fixed-support codec is 0.870 mm, with a bootstrap lower bound of 0.807 mm.

Partition	Bits/ curve	Total bits	Held-out mean point-to-curve distance (mm) Bézier	PCA+SQ	Learned RVQ-VAE	CornerMCQ	Gain	Best control − ours Interval (mm)
Corner	16	331	1.879	3.496	1.305±0.096	0.491±0.025	62%	[0.702, 0.942]
	24	349	1.105	1.658	1.224±0.207	0.311±0.015	72%	[0.724, 0.868]
	32	367	0.987	1.373	1.173±0.335	0.236±0.012	76%	[0.677, 0.831]
	48	403	0.971	0.718	0.910±0.315	0.171±0.010	76%	[0.402, 0.728]
Coarse	16	397	0.489	0.216	0.176±0.006	0.121±0.001	31%	[0.051, 0.060]
	24	428	0.243	0.211	0.133±0.004	0.083±0.001	37%	[0.046, 0.053]
	32	459	0.209	0.209	0.113±0.010	0.065±0.000	43%	[0.045, 0.052]
	48	520	0.197	0.109	0.078±0.006	0.047±0.000	39%	[0.028, 0.032]
Mid	16	464	0.251	0.113	0.122±0.010	0.067±0.001	41%	[0.042, 0.050]
	24	507	0.119	0.084	0.103±0.014	0.048±0.000	43%	[0.033, 0.039]
	32	550	0.100	0.081	0.084±0.003	0.039±0.000	52%	[0.039, 0.046]
	48	637	0.096	0.068	0.058±0.003	0.030±0.000	49%	[0.026, 0.030]
Fine	16	680	0.141	0.076	0.083±0.006	0.054±0.002	29%	[0.017, 0.027]
	24	737	0.097	0.060	0.064±0.004	0.040±0.001	33%	[0.017, 0.023]
	32	794	0.081	0.053	0.053±0.003	0.035±0.001	34%	[0.015, 0.020]
	48	908	0.039	0.039	0.045±0.004	0.030±0.000	22%	[0.008, 0.010]

Table 2: Same partition, same vertices, same bits per curved segment; only the curve coder differs. Held-out test, five-seed mean ± SD for seeded coders; baseline variants and learned configurations were selected on validation (Appendix F). Total bits: held-out mean per panel. Gain: relative reduction against the best control at that point. Interval: paired component bootstrap of the best control minus CornerMCQ on this metric. Validation values and the paired statistics for Chamfer and worst-point error are in Appendix D.

Rate ladder. The four partitions trace one frontier that the whole-boundary codecs never approach (Figure 2). CornerMCQ with the Corner partition and two codewords, at 331 bits, already has a lower mean curve distance (0.4915 mm) than the fixed-support codec at 864 bits (0.7277 mm). Error saturates with more codewords: on Fine, three to six codewords move the mean curve distance only from 0.0403 to 0.0304 mm, because the partition and the reference sampling then set the error.

5.2 Coder against partition: same-partition controls and ablations

Same partition, three other coders. With the partition, the quantized vertices, and the bits per curved segment held fixed, the codebooks beat every alternative coder (Table 2). CornerMCQ has the lowest held-out mean curve distance at all 16 of 16 matched points, by 22–76% against the best control at each point. The paired bootstrap interval against every control excludes zero at all 48 of 48 comparisons on this metric and at 48 of 48 on Chamfer; for the worst point of a panel it excludes zero at 46 of 48, the other 2 being on Fine at 32 and 48 bits per curve (Tables 2 and 7). On the industrial archive it has the lowest held-out mean curve distance at 16 of 16 points, by 13–69% (Table 8, Appendix D). The strongest control varies: learned at seven points, PCA+SQ at seven, Bézier at two.

Stability and cost. Across five fitting seeds the held-out curve distance of CornerMCQ varies by at most 0.0249 mm, against 0.3348 mm for the learned control. Fitting three levels of codebooks takes 4.2 s on one CPU thread and the state is 196,608 bytes, against 6 MiB for the fixed-support codec; encoding a panel takes 3.0 ms and decoding 0.22 ms (Appendix H).

Ablations. The partition carries most of the gain (validation only, Tables 13 and 14). Replacing the corner-adaptive partition by a uniform arc-length partition with the same coder and a comparable rate raises the mean curve distance 13× on GarmentCode and 3× on the industrial archive. On GarmentCode, uniform S=8 with three codewords reaches 1.134 mm curve distance (1.677 mm Chamfer) at 480 bits; Coarse reaches 0.088 mm (0.268 mm) at 437 bits. Coding straight runs as curves instead of giving them zero bits costs 568 bits per panel instead of 473 and buys 0.004 mm (0.0670 versus 0.0712 mm). A panel-scaled frame instead of the chord frame raises the mean curve

Codec (one seed)	GarmentCode held-out test (1,176 panels)						Industrial held-out test (1,623 panels)					
	Raw	ezdxf	Readb.	Recon.	E2E	Valid	Raw	ezdxf	Readb.	Recon.	E2E	Valid
Whole-boundary PCA (480 b)	1176	1176	0.0	1.907	1.907	0.861	1623	1623	0.0	1.359	1.359	0.992
Fixed-support, L=6 (480 b)	1176	1176	0.0	1.622	1.622	0.798	1623	1623	0.0	1.175	1.175	0.920
Mid Bézier, 24 b/curve	1176	1176	0.0	0.313	0.313	0.943	1623	1623	0.0	0.302	0.302	0.996
Mid PCA+SQ, 24 b/curve	1176	1176	0.0	0.279	0.279	0.945	1623	1623	0.0	0.278	0.278	0.996
Mid learned, 24 b/curve	1176	1176	0.0	0.243	0.243	0.944	1623	1623	0.0	0.203	0.203	0.994
CornerMCQ Coarse, L=2	1176	1176	0.0	0.291	0.291	0.946	1623	1623	0.0	0.305	0.305	0.996
CornerMCQ Coarse, L=3	1176	1176	0.0	0.247	0.247	0.944	1623	1623	0.0	0.230	0.230	0.996
CornerMCQ Mid, L=3	1176	1176	0.0	0.206	0.206	0.944	1623	1623	0.0	0.161	0.161	0.996
CornerMCQ Fine, L=4	1176	1176	0.0	0.176	0.176	0.945	1623	1623	0.0	0.119	0.119	0.995

Table 3: CAD return audit on the held-out split of both domains, identical for every codec. Each row audits one model (the first fitting seed, or the deterministic fit), so seeded rows differ in the third decimal from the five-seed means of Table 1. Raw / ezdxf: panels accepted by the locked group-code parser and by ezdxf. Readb.: maximum decoded-versus-parsed discrepancy. Recon. / E2E: Chamfer between the original and the decoded / parsed boundary. Valid: simple-polygon validity. Distances in mm; validation audits in Appendix J.

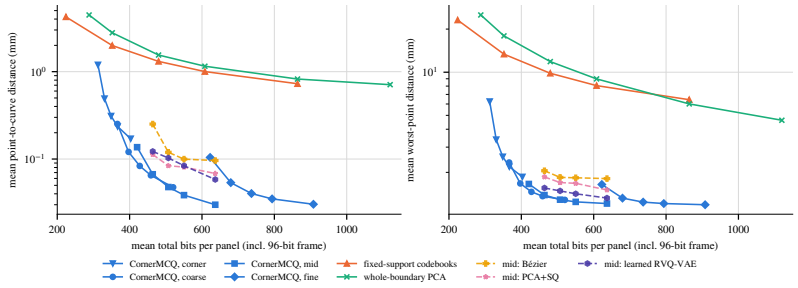


Figure 2: GarmentCode held-out rate–distortion (log scales). Left: mean point-to-curve distance; right: mean per-panel worst-point distance. Solid: CornerMCQ on four partitions (1–6 codewords per curve), fixed-support codebooks (2–12 levels), and whole-boundary PCA. Dashed: the same-partition controls on Mid. Validation and industrial counterparts: Figures 3 and 4 (Appendix C).

distance from 0.0712 to 0.0945 mm at the same rate, and resampling each segment to 16 or 64 points instead of 32 changes it by at most 0.002 mm.

5.3 Transfer: public codebooks on industrial panels

The codebooks released for GarmentCode code industrial panels as they are, with nothing fitted on industrial data. On the held-out industrial panels and the Fine partition they lose 2–4% in mean curve distance against codebooks fitted on the industrial training split. The loss grows with segment length: 24–32% on Mid, 39–52% on Coarse, and 57–94% on Corner. Even so, the transferred codebooks beat every control fitted on industrial data at 15 of 16 points, and every control transferred in the same way at 16 of 16 (Table 15, Appendix I). The opposite direction behaves alike on validation (Appendix I).

5.4 CAD return: every audited codec on every held-out panel

A common 512-position rendering of every audited operating point passes the identical write–parse–remeasure audit on all 1,176 GarmentCode and 1,623 industrial held-out panels: 9 codecs per domain, including all three controls and both whole-boundary baselines. Both parsers accept every file, and the decoded-versus-parsed coordinate discrepancy is exactly 0.0 mm. The end-to-end error of each audited model therefore equals its own reconstruction error: columns Recon. and E2E of Table 3 agree to within 2×10^{-7} mm. CornerMCQ's native decode passes as well (Table 17; validation only, added after the held-out release). Both parsers accept this variable-length polyline (134 records on average instead of 513) on every validation panel and return every transmitted vertex exactly; the common rendering lies 0.49 mm from a vertex on average.

5.5 Failure case and held-out consistency

Where it fails. The precommitted qualitative rule (median, 90th-percentile and worst CornerMCQ panel on the test split) exposes the failure mode (Figure 5, Appendix E). On the median panel CornerMCQ is 30× more accurate than the fixed-support codec while spending 130 fewer bits; on the worst panel it is worse: 1.44 mm mean and 5.14 mm worst-point error at 253 bits, against 0.36 and 2.42 mm for the fixed-support codec at 480. That panel has few detected corners, so its 253-bit code is far below the mean rate. For the industrial archive, where no geometry may be shown, the same rule gives a worst panel coded with 448 bits that misses by 7.11 mm at its worst point, against 11.22 mm for the fixed-support codec at 480 bits.

Validation against held-out. Held-out and validation numbers are close on both domains, and the ranking of the four coders is the same on both splits at 16 of 16 matched points on GarmentCode and 14 of 16 on the industrial archive (Appendix D), and the three preregistered hypotheses of Appendix A held on both domains.

6 Limitations

We code one closed outer boundary per panel. Notches, drill holes, seam allowances, grain lines, and inter-panel relations are CAD entities that the writer does not emit. Neither corpus contains digitization noise; hand-digitized patterns, with noise and near-collinear vertices, may stress the corner detector, and we have no such data. Under synthetic jitter (Appendix K), accuracy degrades slowly but the rate does not hold (437 to 520 bits at σ=0.1 mm). The partition thresholds were set on the public corpus; other sampling densities or noise levels may need their own.

The evaluation is phase-free; the code is not. Rotation leaves the partition unchanged up to a few threshold ties, but the chord frame is handed: a reversed traversal raises the Coarse L=3 curve distance by 6% on GarmentCode and 20% on the industrial archive, and a clockwise rule on the input removes the loss without being part of the released encoder (Appendix K). Where the detector finds fewer than two corners, the partition also depends on the start sample (6,428 of 6,492 industrial panel–partition pairs survive a cyclic shift unchanged; Appendix B). Nothing guarantees a minimum rate: a panel with few detected corners gets a short code, and the worst held-out panel is of this kind (5.14 mm at 253 bits, Section 5.5); a rate floor or a bound on segment length would address this, and neither is implemented.

The mean worst point of a panel stays around a millimeter even at the highest rates (Appendix C); what remains is set by where the partition places vertices relative to a 512-sample reference that itself clips sharp tips, not by the curve coder. Every error in this paper is measured against that cleaned, resampled reference; the error against the original CAD boundary is not quantified. The main audit's 512-position rendering, used so that codecs are comparable, cuts CornerMCQ's corners; the native polyline that keeps them was audited on validation only (Appendix K). The learned controls were selected on validation, which favours them. Finally, we evaluate the code as a representation and do not train a sequence model on it; whether its tokens help a generative model remains open.

7 Conclusion

CornerMCQ codes a sewing-pattern boundary as explicit corner vertices, straight runs that cost no curve bits, and a few residual codewords per curved segment. On the GarmentCode held-out split it reaches 0.292 mm Chamfer at 397 bits per panel, against 1.625 and 1.907 mm for the fixed-support codebooks and PCA at 480 bits; on a private industrial archive, 0.303 mm at 322 bits against 1.173 and 1.359 mm. With the partition and the bits held fixed, its codebooks beat a Bézier coder, PCA with scalar quantization, and a tuned learned quantizer at every matched rate point on both domains, and codebooks fitted on the public corpus lose only 2–4% on the finest partition of the industrial panels. Every audited codec comes back through a CAD file unchanged; the differences between them are set by the allocation of bits, not by the exchange format.

AI USE STATEMENT

We used generative AI tools to assist with manuscript writing and language polishing. All AI-assisted text was carefully reviewed and revised by the authors. LLM-generated code was verified and tested for correctness by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses the publicly released GarmentCodeData corpus of synthetic, program-generated sewing patterns and a private archive of industrial production patterns, used with the permission of its owner; the archive cannot be shared publicly. Only aggregate statistics of the archive are reported: no pattern geometry, identifier, or file is shown or released. The work involves no human subjects or personal data. We are not aware of harmful applications specific to a boundary codec.

REPRODUCIBILITY STATEMENT

Section 3 specifies the partition, the chord-frame codebooks, the bit ledger and the CAD audit. Section 4 and Appendix A give the corpora, the splits, the one-time held-out release and every frozen setting; Appendix F gives the controls and their tuning, and Appendix K the analyses added after the release. The GarmentCodeData results use the public corpus; the industrial archive cannot be shared, so its results are reported as aggregate statistics only.

REFERENCES

Artem Babenko and Victor Lempitsky. Additive quantization for extreme vector compression. In *Proc. CVPR*, 2014.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2924–2934, 2025. doi: 10.1109/CVPR52734.2025.00278.

Alexandre Carlier, Martin Danelljan, Alexandre Alahi, and Radu Timofte. DeepSVG: A hierarchical generative network for vector graphics animation. In *Proc. NeurIPS*, 2020.

Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *Transactions on Machine Learning Research*, 2023.

David H. Douglas and Thomas K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica*, 10(2):112–122, 1973. doi: 10.3138/FM57-6770-U75U-7727.

Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12868–12878, 2021. doi: 10.1109/CVPR46437.2021.01268.

Yaroslav Ganin, Sergey Bartunov, Yujia Li, Ethan Keller, and Stefano Saliceti. Computer-aided design as language. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 5885–5897, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/2e92962c0b6996add9517e4242ea9bdc-Abstract.html.

Robert M. Gray. Vector quantization. *IEEE ASSP Magazine*, 1(2):4–29, 1984. doi: 10.1109/MASSP.1984.1162229.

David Ha and Douglas Eck. A neural representation of sketch drawings. In *Proc. ICLR*, 2018.

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*, 43(4):1–13, 2024. doi: 10.1145/3658147.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer, New York, NY, 2 edition, 2002. doi: 10.1007/b98835.

Biing-Hwang Juang and A. Gray. Multiple stage vector quantization for speech coding. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 7, pp. 597–600, 1982. doi: 10.1109/ICASSP.1982.1171604.

Aggelos K. Katsaggelos, Lisimachos P. Kondi, Fabian W. Meier, Jörn Ostermann, and Guido M. Schuster. MPEG-4 and rate-distortion-based shape-coding techniques. *Proceedings of the IEEE*, 86(6):1126–1154, 1998. doi: 10.1109/5.687833.

Mohammad Sadil Khan, Sankalp Sinha, Talha Uddin Sheikh, Didier Stricker, Sk Aziz Ali, and Muhammad Zeshan Afzal. Text2CAD: Generating sequential CAD designs from beginner-to-expert level text prompts. In *Proc. NeurIPS*, volume 37, pp. 7552–7579, 2024.

Maria Korosteleva and Sung-Hee Lee. NeuralTailor: Reconstructing sewing pattern structures from 3D point clouds of garments. *ACM Transactions on Graphics*, 2022. doi: 10.1145/3528223.3530179.

Maria Korosteleva and Olga Sorkine-Hornung. GarmentCode: Programming parametric sewing patterns. *ACM Transactions on Graphics*, 42(6):199:1–199:15, 2023. doi: 10.1145/3618351.

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 *Proc. ECCV*, 2024.

Frank P. Kuhl and Charles R. Giardina. Elliptic Fourier features of a closed contour. *Computer Graphics and Image Processing*, 18(3):236–258, 1982.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proc. CVPR*, 2022.

Xinyu Li, Qi Yao, and Yuanda Wang. GarmentDiffusion: 3D garment sewing pattern generation with multimodal diffusion transformers. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1458–1466, 2025. doi: 10.24963/ijcai.2025/163.

Lijuan Liu, Xiangyu Xu, Zhijie Lin, Jiabin Liang, and Shuicheng Yan. Towards garment sewing pattern reconstruction from a single image. *ACM Transactions on Graphics*, 42(6):1–15, 2023. doi: 10.1145/3618319.

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 *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17640–17650, 2025. doi: 10.1109/ICCV51701.2025.01639.

Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28 (2):129–137, 1982. doi: 10.1109/TIT.1982.1056489.

Fabian W. Meier, Guido M. Schuster, and Aggelos K. Katsaggelos. A mathematical model for shape coding with B-splines. *Signal Processing: Image Communication*, 15(7–8):685–701, 2000. doi: 10.1016/S0923-5965(99)00045-4.

Manfred Moitzi. ezdxf: Python interface to DXF. https://github.com/mozman/ezdxf. Python package, version 1.4.2 used in this work.

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 *Proc. CVPR*, pp. 8138–8149, 2025.

Brian A. Nosek, Charles R. Ebersole, Alexander C. DeHaven, and David T. Mellor. The preregistration revolution. *Proceedings of the National Academy of Sciences*, 115(11):2600–2606, 2018.

Michael Plass and Maureen Stone. Curve-fitting with piecewise parametric cubics. In *Proceedings of the 10th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '83)*, pp. 229–239, 1983. doi: 10.1145/800059.801153.

Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1(3):244–256, 1972. doi: 10.1016/S0146-664X(72)80017-0.

Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with VQ-VAE-2. In *Proc. NeurIPS*, 2019.

Pradyumna Reddy, Michaël Gharbi, Michal Lukac, and Niloy J. Mitra. Im2Vec: Synthesizing vector graphics without vector supervision. In *Proc. CVPR*, 2021.

Philip J. Schneider. An algorithm for automatically fitting digitized curves. In Andrew S. Glassner (ed.), *Graphics Gems*, pp. 612–626. Academic Press, 1990. doi: 10.1016/B978-0-08-050753-8.50132-7.

Guido M. Schuster and Aggelos K. Katsaggelos. An optimal polygonal boundary encoding scheme in the rate distortion sense. *IEEE Transactions on Image Processing*, 7(1):13–26, 1998. doi: 10.1109/83.650847.

Ari Seff, Wenda Zhou, Nick Richardson, and Ryan P. Adams. Vitruvion: A generative model of parametric CAD sketches. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=Ow1C7s3UcY.

Zecheng Tang, Chenfei Wu, Zekai Zhang, Minheng Ni, Shengming Yin, Yu Liu, Zhengyuan Yang, Lijuan Wang, Zicheng Liu, Juntao Li, and Nan Duan. StrokeNUWA—tokenizing strokes for vector graphic synthesis. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *PMLR*, pp. 47830–47845, 2024. URL https://proceedings.mlr.press/v235/tang24h.html.

Cho-Huak Teh and Roland T. Chin. On the detection of dominant points on digital curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8):859–872, 1989. doi: 10.1109/34.31447.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Proc. NeurIPS*, 2017.

Fangsheng Weng, Junhao Chen, Xiang Li, Jie Qin, Hanzhong Guo, Shaochun Hao, and Xiaoguang Han. GarmentGPT: Compositional garment pattern generation via discrete latent tokenization. In *Proc. ICLR*, 2026.

Karl D. D. Willis, Pradeep Kumar Jayaraman, Joseph G. Lambourne, Hang Chu, and Yewen Pu. Engineering sketch generation for computer-aided design. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 2105–2114, 2021. doi: 10.1109/CVPRW53098.2021.00239.

Ronghuan Wu, Wanchao Su, Kede Ma, and Jing Liao. IconShop: Text-guided vector icon synthesis with autoregressive transformers. *ACM Transactions on Graphics*, 42(6):1–14, 2023. doi: 10.1145/3618364.

Rundi Wu, Chang Xiao, and Changxi Zheng. DeepCAD: A deep generative network for computer-aided design models. In *Proc. ICCV*, 2021.

Ximing Xing, Ziteng Xue, Zhenxi Li, Weicong Liang, Linqing Wang, Zhantao Yang, Tiankai Hang, Zijin Yin, Qinglin Lu, Chunyu Wang, and Qian Yu. Hierarchical SVG tokenization: Learning compact visual programs for scalable vector graphics modeling. *arXiv preprint arXiv:2604.05072*, 2026.

Xiang Xu, Karl D. D. Willis, Joseph G. Lambourne, Chin-Yi Cheng, Pradeep Kumar Jayaraman, and Yasutaka Furukawa. SkexGen: Autoregressive generation of CAD construction sequences with disentangled codebooks. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *PMLR*, pp. 24698–24724, 2022. URL https://proceedings.mlr.press/v162/xu22k.html.

Xiang Xu, Pradeep Kumar Jayaraman, Joseph G. Lambourne, Karl D. D. Willis, and Yasutaka Furukawa. Hierarchical neural coding for controllable CAD model generation. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *PMLR*, pp. 38443–38461, 2023. URL https://proceedings.mlr.press/v202/xu23f.html.

Yiying Yang, Wei Cheng, Sijin Chen, Xianfang Zeng, Fukun Yin, Jiaxu Zhang, Liao Wang, Gang Yu, Xingjun Ma, and Yu-Gang Jiang. OmniSVG: A unified scalable vector graphics generation model. In *Proc. NeurIPS*, 2025.

Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved VQGAN. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=pfNyExj7z2.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. SoundStream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30, 2022.

Feng Zhou, Ruiyang Liu, Chen Liu, Gaofeng He, Yong-Lu Li, Xiaogang Jin, and Huamin Wang. Design2GarmentCode: Turning design concepts to tangible garments through program synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23712–23722, 2025. doi: 10.1109/CVPR52734.2025.02208.