Appendices: MechaFormer: Sequence Learning for Kinematic Mechanism Design Automation

Diana Bolanos, Mohammadmehdi Ataei, Pradeep Kumar Jayaraman

1 Implementation Details

Data Preprocessing. The raw mechanism dataset[1] undergoes several preprocessing steps to create a suitable representation for sequence learning. Each mechanism in the dataset is first normalized to a canonical coordinate frame through a sequence of transformations that place the two ground (fixed) joints at standardized positions: the first at the origin $(0, 0)$ and the second at $(1, 0)$. This normalization eliminates the infinite variations due to translation, rotation, and scaling while preserving the essential kinematic relationships. For the target curves, we employ cubic B-spline fitting with a fixed number of 64 control points, providing a compact yet expressive representation that captures curve shapes while maintaining a consistent input dimension for the neural network. The continuous joint coordinates are discretized into 200 uniform bins spanning the range $[-10, 10]$, transforming the regression problem into a multi-class classification task that the Transformer can handle more effectively.

Domain-Specific Language. The DSL serializes each mechanism into a structured token sequence following a strict grammar. Each sequence begins with a start-of-sequence token, followed by the mechanism type declaration (e.g., `MECH_TYPE: RRRR`), then a `POINTS:` marker, and finally the quantized coordinates of each free joint. Each joint is represented as `P_i X: BIN_j Y: BIN_k`, where i indexes the joint and j, k are the bin indices for the x and y coordinates respectively. This structured format ensures that the model learns both the syntax and semantics of valid mechanism descriptions while maintaining interpretability.

Model Architecture and Training. MechaFormer employs a standard Transformer encoder-decoder architecture. The encoder takes as input a sequence of 64 B-spline control points (each point represented as 2D coordinates) and processes them through six self-attention layers to build a contextualized representation of the target curve. The decoder autoregressively generates the DSL token sequence, starting from a start-of-sequence token and producing one token at a time through six masked self-attention layers with cross-attention to the encoder output. Each token prediction is made from a vocabulary of 232 tokens (including special tokens, DSL structure tokens, mechanism type tokens, and 200 coordinate bins). The architecture incorporates several modern improvements including flash attention for computational efficiency, RMSNorm for training stability, gated linear units (GLU) with Swish activation in the feedforward layers, rotary positional embeddings for better length generalization, and QK normalization in attention layers. Table 1 summarizes all hyperparameters used in our experiments. Training is performed using distributed data parallel (DDP) across 8× NVIDIA A100 GPUs. The training takes about one hour to complete. The

[1] https://www.kaggle.com/datasets/purwarlab/four-six-and-eight-bar-mechanisms-with-curves

complete training and inference code are made publicly available to facilitate reproducibility and future research.

Hyperparameter	**Value**
Model Architecture	
Hidden dimension (d_{model})	256
Attention heads	8
Encoder layers	6
Decoder layers	6
Total parameters	$\sim$19M
Training	
Optimizer	Adam
Learning rate	1×10^{-4}
Weight decay	1×10^{-5}
Batch size (per GPU)	256
Total batch size	2048
Epochs	30
LR schedule	ReduceLROnPlateau
LR reduction factor	0.5
LR patience	3 epochs
Gradient clipping	1.0
Loss Function	
Loss type	Cross-entropy
Padding token ignored	✓
Data Processing	
B-spline control points	64
B-spline degree	3
Coordinate range	$[-10, 10]$
Coordinate bins	200
Min. instances per type	20,000
Train/validation split	90/10
Architecture Features	
Flash attention	✓
RMSNorm	✓
GLU feedforward	✓
Rotary position embeddings	✓
QK normalization	✓
Swish activation	✓
No bias in feedforward	✓

Table 1: Hyperparameters used for training MechaFormer.

2 Coordinate Discretization Ablation

To determine the optimal discretization granularity for joint coordinates, we conducted an ablation study varying the number of bins B used to quantize the continuous coordinate space $[-10, 10]$. We evaluated three bin sizes: $B \in \{50, 200, 2000\}$, specifically chosen to represent coarse, medium, and fine discretization levels respectively. These values span two orders of magnitude to comprehensively assess how quantization resolution influences model accuracy.

The bin size directly influences model accuracy through two competing effects. With $B = 50$ (bin width = 0.4 units), the coarse quantization introduces substantial discretization error—each predicted coordinate can only take one of 50 possible values, limiting the precision with which joint positions can be specified. This manifests as poor reconstruction accuracy (DTW = 8.0628) since the model cannot place joints with sufficient precision to accurately trace the target curves.

At the opposite extreme, $B = 2000$ (bin width = 0.01 units) provides high spatial resolution but paradoxically yields the worst performance (DTW = 12.2427). This degradation occurs because fine-grained discretization creates a sparse, high-dimensional output space where each of the 2000 bins appears infrequently in the training data. The model struggles to learn robust patterns across this sparse categorical distribution, leading to poor generalization despite the theoretical capability for precise coordinate specification.

The optimal configuration at $B = 200$ (bin width = 0.1 units) balances these competing factors. It provides sufficient spatial resolution for accurate joint placement while maintaining a learnable output distribution where each bin appears frequently enough in the training data for the model to learn meaningful patterns. This finding guided our choice of $B = 200$ for all experiments reported in the main paper, achieving a median DTW of 3.0900 that represents nearly 3× improvement over either extreme.

Bin Size	**DTW Median**
50	8.0628
200	**3.0900**
2000	12.2427

Table 2: DTW median values for different bin sizes. We run these experiments with Best @ $k = 1$ and 10 different samples.

3 Temperature Sampling

The purpose of this study was to identify an appropriate sampling temperature T for generating mechanisms across experiments. Since temperature influences the stochasticity of the model, selecting a suitable value ensures both quality and diversity in generated outputs while avoiding performance degradation due to overly deterministic or overly random behavior. We evaluated the average best normalized DTW between the predicted and target trajectories across different temperature values $T \in \{0.001, 0.1, 0.5, 1.0\}$ and sampling counts $k \in \{1, 2, 4, 8\}$. As shown in Table 3, $T = 0.1$ achieved the best performance when $k = 1$. Furthermore, as k increased, $T = 0.1$ remained competitive, with DTW scores comparable to other temperatures. Based on this balance between quality and consistency, we selected $T = 0.1$ as the default temperature for all experiments.

Temperature	k = 1	k = 2	k = 4	k = 8
T = 0.001	6.8652	6.8652	6.8652	6.8652
T = 0.1	**6.8316**	5.3570	4.2911	3.6301
T = 0.5	7.5084	**5.2166**	**3.8533**	**3.1371**
T = 1.0	9.6690	6.3901	4.5011	3.4234

Table 3: Average best normalized DTW for different values of sampling temperature T and number of samples k. Lower is better.

4 Distributions

Figure 1 shows the distribution of DTW scores for different sampling counts k, plotted as a boxplot on a linear scale. Each box illustrates the interquartile range (IQR), with the central line indicating the median DTW value for each k. We chose to highlight median values in our main text over means due to the presence of large outliers, which can significantly skew the average and misrepresent the typical performance. The boxplot makes this effect clear, particularly at lower k values where a few poorly performing samples inflate the upper range. The median provides a more robust summary statistic under these conditions, capturing the central tendency of the distribution more accurately.

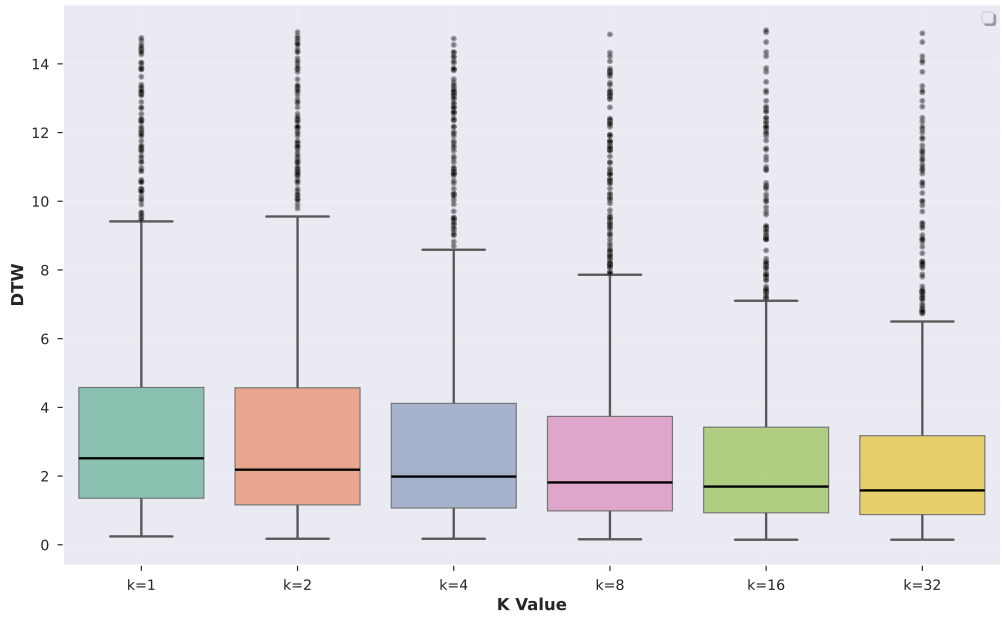

Figure 1: DTW distribution across different values of k. Each box shows the interquartile range with the median marked. Extreme outliers are present and motivate comparisons across medians.

5 L-BFGS-B

In this study, we use the best outcomes from the Best @ $k = 32$ study, and search through the joint spaces for each of the free joints. This approach does not search through different topologies, as these are non-differentiable values. We initiate this problem using the following optimization statement:

$$
\begin{aligned}
\min_{\mathbf{x}} \quad & \mathrm{DTW}(\mathcal{N}(\mathcal{X}_I), \mathcal{N}(\mathcal{X}_C(\mathbf{x}))) \\
\text{subject to} \quad & x_i \in [x_i^0 - \delta_i, x_i^0 + \delta_i], \quad i \in \{1, \ldots, n\} \\
& \mathbf{j}'_{g_1} = (0, 0) \\
& \mathbf{j}'_{g_2} = (1, 0) \\
\text{where} \quad & \delta_i = \max(0.5, 0.5|x_i^0|), \\
& \mathcal{N}(\mathcal{X}) = \frac{\mathcal{X}_I - \mu_I}{\sigma_{RMS_I}}
\end{aligned}
$$

as defined by: $\mathbf{x}$ is the vector of mechanism coordinates to be optimized, $\mathcal{X}_I$ is the input trajectory curve, $\mathcal{X}_C(\mathbf{x})$ is the coupler trajectory for coordinates $\mathbf{x}$, $\mathcal{N}(\cdot)$ is the normalization function, μ is the mean of the input trajectory points, σ_{RMS_I} is the RMS variance of the input trajectory curve, x_i^0 are the initial coordinate values, δ_i are the bounds for each coordinate, and $\mathbf{j}'_{g_1}$ and $\mathbf{j}'_{g_2}$ are fixed ground points at $(0, 0)$ and $(1, 0)$ respectively.

This optimization routine is solved using the L-BFGS-B algorithm with the following parameters:

$$
\begin{aligned}
\texttt{maxiter} &= 50 \\
\texttt{maxfun} &= 100 \\
\texttt{ftol} &= 10^{-6} \\
\texttt{gtol} &= 10^{-6} \\
\texttt{eps} &= 10^{-3}
\end{aligned}
$$

6 Examples

Figure 2 presents nine representative examples of generated mechanisms along with their corresponding target (input) and generated (output) trajectories. Each subplot displays a unique mechanism sample, visualizing the coupler path traced by the mechanism in relation to the desired curve. The DTW score is annotated in each plot, providing a quantitative measure of trajectory alignment. We deliberately selected a range of samples with varying performance levels to highlight the diversity in accuracy: from high-performing mechanisms with low DTW values (green) to low-performing ones with large trajectory mismatches (red). Figure 3 shows 100 examples of different curves in the dataset used to train our model.

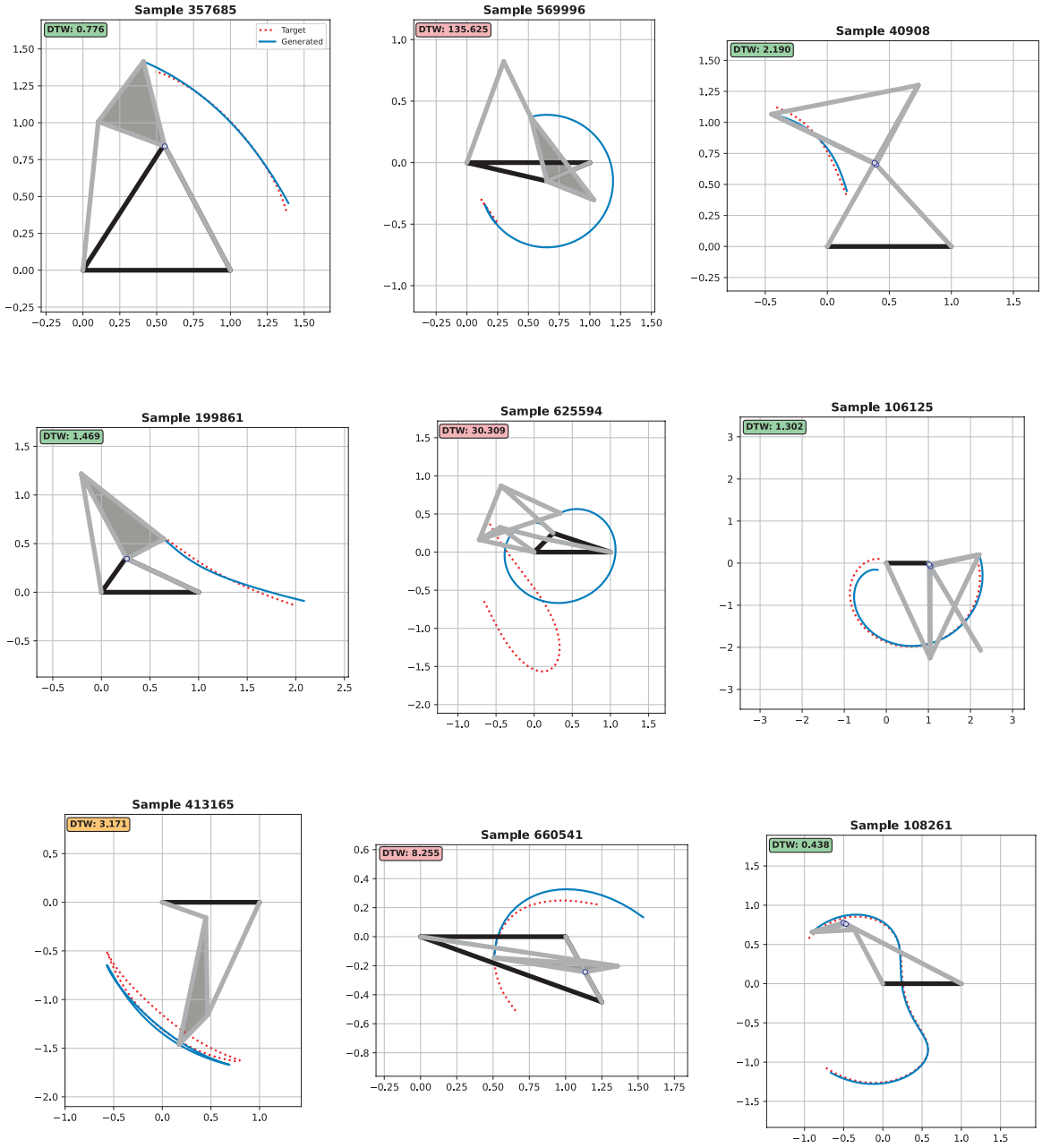

Figure 2: Nine sample mechanisms and their output trajectories in addition to the input curve. High, medium, and low accuracy performing outcomes are represented. Prismatic joints are denoted with a gray circle.

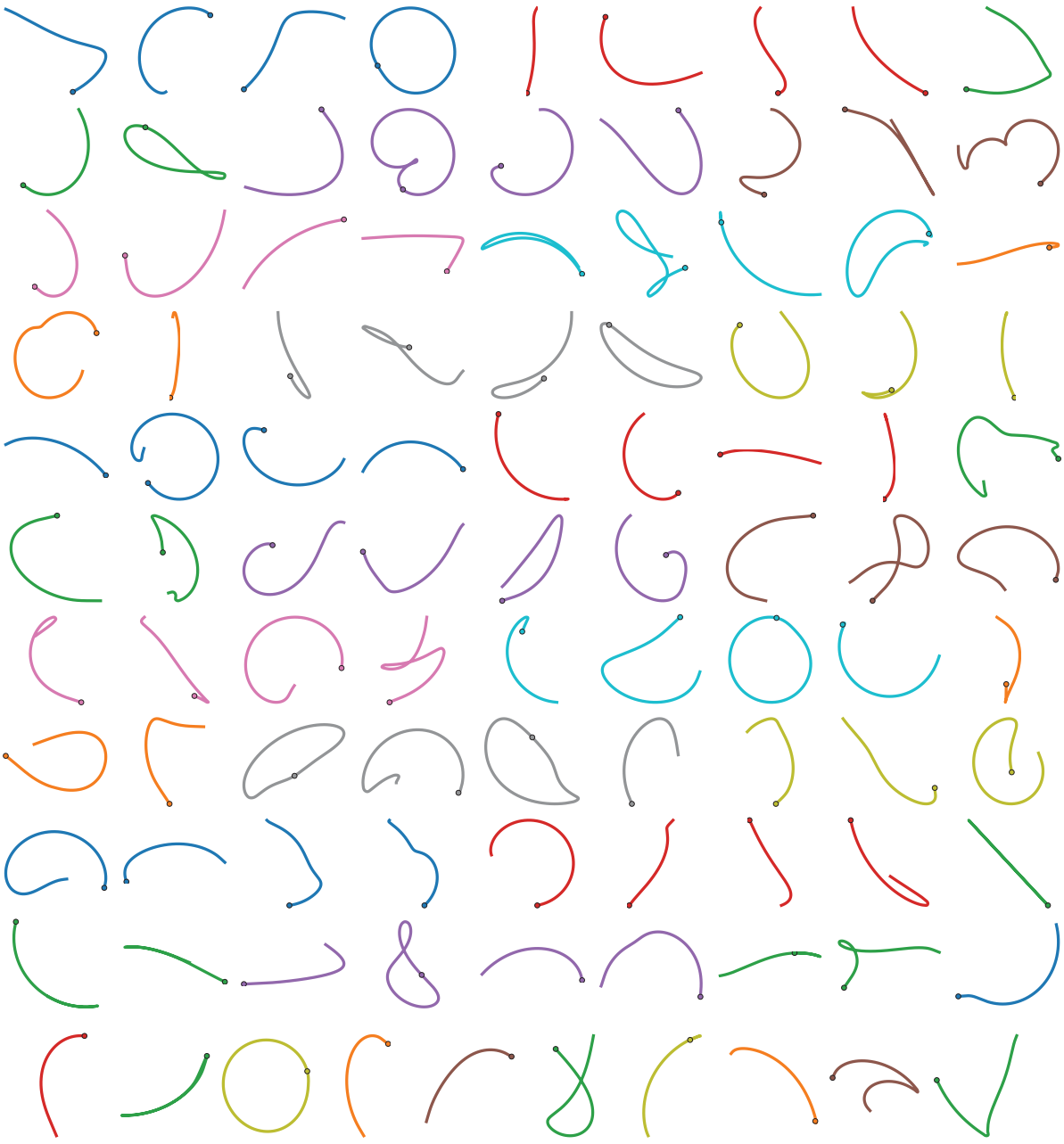

Figure 3: 100 sample curves used in the dataset.