ASME Journal of Mechanisms and Robotics
Online journal at:
https://asmedigitalcollection.asme.org/mechanismsrobotics

Path Generative Model Based on Conditional β-Variational Auto Encoder for Four-Bar Mechanism Design

Anar Nurizada
Computer-Aided Design and Innovation Lab,
Department of Mechanical Engineering,
Stony Brook University,
Stony Brook, NY 11794
e-mail: anar.nurizada@stonybrook.edu

Zhijie Lyu
Computer-Aided Design and Innovation Lab,
Department of Mechanical Engineering,
Stony Brook University,
Stony Brook, NY 11794
e-mail: zhijie.lyu@stonybrook.edu

Anurag Purwar[1]
Computer-Aided Design and Innovation Lab,
Department of Mechanical Engineering,
Stony Brook University,
Stony Brook, NY 11794
e-mail: anurag.purwar@stonybrook.edu

This article introduces a novel methodology based on conditional β-variational autoencoder (cβ-VAE) architecture to generate diverse types of planar four-bar mechanisms for a given coupler curve. Central to our contribution is the novel integration of cross- and self-attention layers within the VAE framework, facilitating an encoding and decoding process that captures the complex interdependencies of mechanism parameters and associated coupler curves. We propose a unified representation scheme for four-bar mechanisms with both revolute and prismatic joints, utilizing a consistent set of joints to describe each mechanism type. To support and validate our methodology, we have compiled an extensive dataset featuring both open and closed coupler curves of the aforementioned mechanism types. Furthermore, the article introduces three metrics aimed at quantifying the efficacy of our model, alongside an innovative algorithm designed to enhance the predictive outcomes by identifying and computing cognate mechanisms. [DOI: 10.1115/1.4067169]

Keywords: planar four-bar linkage, path synthesis, conditional variational autoencoder, neural networks, machine learning, deep learning, application of machine learning, artificial intelligence, mechanism synthesis and analysis, theoretical and computational kinematics

1 Introduction

Linkage synthesis for path, function, and motion generation in mechanical engineering has been extensively studied. Traditional approaches discretize curves or motions into points or poses, either directly specified or sampled from continuous motions. In path synthesis, linkage parameters are determined to ensure the coupler interpolates through these precision points, achieving exact solutions for up to nine points [1]. For more than nine points, approximate solutions using optimization methods become necessary, which typically pose challenges in terms of accuracy and efficiency.

Despite their widespread application, optimization-based methods exhibit notable limitations such as slow optimization, dependence on initial conditions, and lack of guaranteed outcomes. These methods only ensure the coupler passes through finite precision points, often failing to capture the overall shape of the given path. The structural error objective function, commonly used, is ineffective for practical solutions as it mischaracterizes the design problem, requiring simultaneous optimization of the coupler curve's shape, size, orientation, and position. In addition, these approaches usually produce just one mechanism at high computational expense, limiting mechanism designers who often seek multiple potential solutions at early stages of machine design.

An alternative approach involves synthesizing linkages capable of tracing continuous paths, addressing challenges in solving the overdetermined problem of linkage coupler-curve synthesis, where the coefficients of the coupler curve equation exceeds the available design parameters [2]. More complex linkages, beyond simple four-bar mechanisms, enable intricate motions but are harder to design. Advanced optimization algorithms have been employed, but challenges in synthesis remain significant.

Researchers have applied machine learning techniques to model the relationship between mechanism parameters and coupler curves. Typically, a dataset of mechanisms and associated coupler curves is used to train an artificial neural network (ANN) to map various coupler curve representations to corresponding mechanisms. A key advantage is the near-instantaneous design of new mechanisms once the model is trained. However, they are known to produce a singular solution of a specific type, such as planar four-bar with revolute joints only. A key problem in conceptual design of mechanisms is generation of novel, diverse, and unique solutions without presupposing the type of mechanisms. The recent advances in generative modeling framework are poised to facilitate these goals.

This article presents a novel combination of conditional and β-variational autoencoder (β-VAE), enabling the synthesis of multiple mechanisms that approximate input coupler curves while allowing users to choose the mechanism type. The goal is not precision position synthesis or exact curve generation, but novel and

[1]Corresponding author.

Contributed by the Mechanisms and Robotics Committee of ASME for publication in the Journal of Mechanisms and Robotics. Manuscript received March 2, 2024; final manuscript received November 8, 2024; published online December 12, 2024. Assoc. Editor: Ashis G. Banerjee.

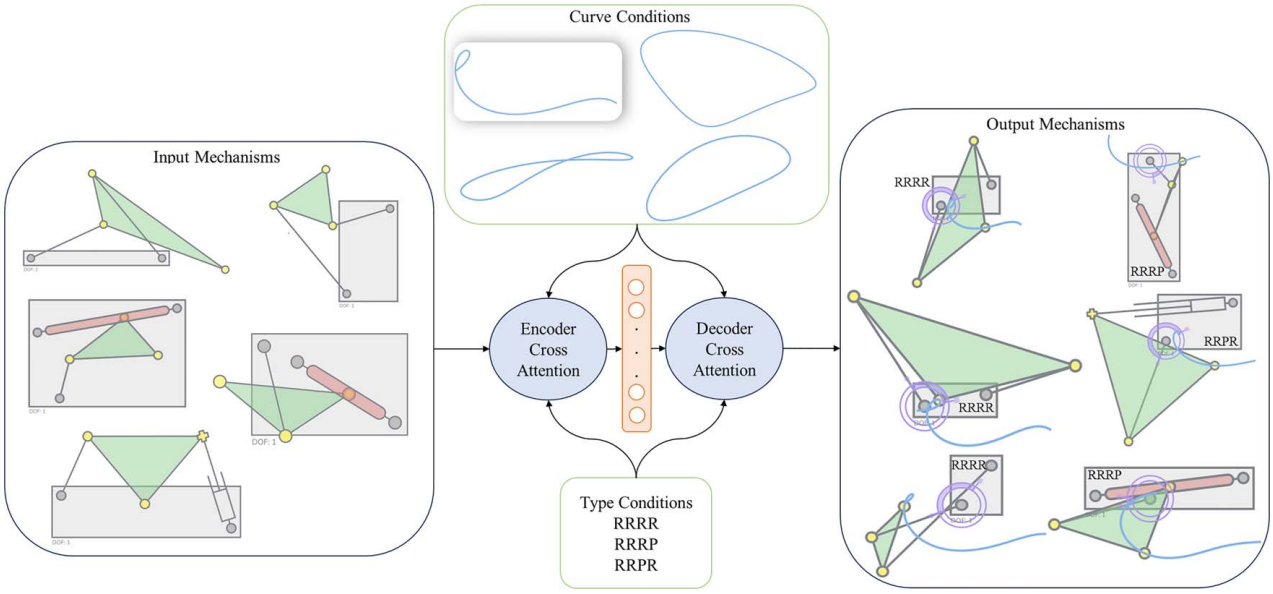

Fig. 1 This overview figure illustrates our proposed methodology. In the training phase, the model processes mechanisms by incorporating them with their respective coupler curves and types, which serve as conditions, into an encoder. This process generates a latent vector. This vector, when reintegrated with the coupler curves and mechanism types (conditions), enables the decoder to recreate the original mechanism accurately. After the model is fully trained, it has the capability to navigate the latent space, allowing it to generate a variety of mechanisms tailored to specific coupler curves and mechanism types. For example, when provided with a particular curve (highlighted in the figure) and different mechanism types, the model can produce suitable mechanisms, as demonstrated in the output section of the figure.

diverse concept generation of perceptually similar curves. Our approach utilizes a unified deep generative neural network (DGNN) to predict mechanism parameters.

We introduce a deep generative model integrating conditional embeddings tailored to the desired mechanism type and coupler curve. Figure 1 shows an overview of our proposed model. The model processes mechanisms represented by Cartesian coordinates of joints, coupled with their coupler curve and mechanism type as conditions. This creates a latent vector representing the input data, merged with initial conditions for reconstruction by the decoder. After training, the decoder generates diverse mechanisms from random latent space samples and specified conditions.

To assess and quantify the efficacy of our model, we propose three hierarchical metrics. The hierarchy signifies that certain measurements take precedence over others. The primary and most crucial measurement is the reconstruction quality of the predicted coupler curves concerning the input (desired) coupler curve. The second most significant measurement is the novelty of the predicted mechanisms, followed by the diversity of the generated mechanisms. We define a set of mechanism solutions to be novel if they are sufficiently different from each other and diverse if there is adequate representation of novel mechanisms from each type. If a more critical measurement falls below a satisfactory threshold, the significance of the other measurements diminishes.

Owing to computational constraints, we have prepared a dataset consisting of three types of four-bar mechanisms. Specifically, our model is applicable to four-bar mechanisms featuring (1) entirely revolute (R) joints, denoted as RRRR mechanisms, (2) slider-crank mechanisms incorporating one prismatic joint (P), referred to as RRRP mechanisms, and (3) inverted slider-crank mechanisms, which also have one prismatic joint and are denoted as RRPR mechanisms. However, the methodology presented is scalable to include other types of four-bar and higher-order mechanisms.

The original contributions of this work are in (1) development of a novel methodology based on a conditional β-VAE model with cross-attention and self-attention layers for synthesizing multiple mechanisms of different types for a given coupler curve, (2) creating a unified representation for different types of four-bar mechanisms by representing them using a fixed number of coordinates, (3) generation of an extensive dataset comprising open and closed curves of three types of planar four-bar mechanisms, and (4) introduction of three metrics for quantitative assessment of the predicted outcomes.

The rest of this article is organized as follows: Sec. 2 reviews machine learning applications to mechanism path synthesis. Section 3 introduces essential variational autoencoder concepts necessary for our proposed method and details our model's architecture, highlighting the innovative aspects that distinguish our approach from existing methodologies. In Sec. 4, we outline the metrics used to evaluate our results. Section 5 explains how we built the dataset for training the conditional-β-VAE and describes a unifying algorithm for computing cognates of the predicted mechanisms. Finally, Sec. 6 presents the results of applying our trained model to various coupler curves.

2 Review of Machine Learning Approach to Kinematic Synthesis

It is interesting that it has been only 30 years since Hoskins and Kramer [3] first investigated the synthesis of mechanical linkages for user-specified curves using ANNs. Focusing on four-bar linkages, they utilized a radial basis function ANN to retrieve parameters based on user-specified curves. The resulting design is optimized numerically, demonstrating the effectiveness of ANNs and optimization in inverse modeling of complex systems. Independent of the use of ANNs for mechanical linkage synthesis, researchers have also explored alternative representation of input coupler curves, such as Fourier descriptors (FDs) [4] and wavelets [5] and using them as a key to search for mechanisms in a catalogue [6,7]. Using these representations as feature vectors, there was a concerted effort to design neural networks to learn the mapping between these representations and mechanism parameters. Almost a decade passed since Hoskins and Kramber's work before Vasiliu and Yannou [8] represented the coupler curves using Fourier descriptors and trained a fully connected neural network

(FCNN) to approximate the relationship between input descriptors and mechanism parameters. Galan-Marin et al. [9] presented an approach using a wavelet-based neural network to streamline the design space for optimal crank-rocker mechanisms in path generation, yielding superior performance over Fourier-based networks.

Erkaya and Uzmay [10] analyzed a planar slider-crank mechanism with clearances using a multilayered neural network. Joint clearances are modeled as massless virtual links, and a genetic algorithm optimizes design parameters to minimize deviations. The effects of clearances on kinematics and transmission quality are investigated for crank-pin and piston-pin centers. Ahmadi et al. [11] introduce a game-theoretic approach for multi-objective optimal synthesis of four-bar mechanisms, treating tracking error as the leader and transmission angle deviation as the follower. Using a group method of data handling-type neural network, the approach demonstrates efficacy in transforming multi-objective synthesis into a single-objective synthesis in two case studies on four-bar mechanism synthesis.

Khan et al. [12] proposed a method for dimensional synthesis of mechanisms tracing given paths, using Fourier descriptors of cumulative angular deviation instead of points on the curve and training an artificial neural network to learn the relationship with mechanism dimensions. While effective, the current FD-based methods have limitations. They require a mathematically defined, parametric representation of the path. However, different parameterizations for the same path yield distinct Fourier descriptors, introducing variability in path identity or tags. Li and Chen [13] addressed this problem by proposing a parametrization-invariant method based on Fourier analysis to eliminate the influence of different parameterizations.

Yim et al. [14] introduced a neural network-based approach to simultaneously determine mechanism topology and end-effector location, using a unified spring-connected rigid block model for diverse mechanisms. In a subsequent work [15], they extended this approach to synthesize spatial linkage mechanisms.

Path synthesis is a one-to-many problem where an input coupler curve can be approximated using many different mechanisms. According to the Roberts-Chebyshev theorem [16], for every curve traceable by planar four-bar linkages, there exist three linkages, called cognate linkages, that describe the same curve. Predominantly, the previous research utilizing ANNs has been limited in that they produce only one solution for a given coupler curve, overlooking alternative solutions. The arrival of deep generative models like VAE [17] and generative adversarial networks [18] are poised to transform the field of mechanism design by modeling the synthesis problem more naturally and generating a variety of solutions, hitherto not possible. Taking advantage of such models, Deshpande and Purwar [19,20] introduced an image-based variational path synthesis approach particularly suited for scenarios where imprecise or uncertain input is provided by mechanism designers. The proposed approach, modeling input curves as probability distributions of image pixels, first creates a database of four- and six-bar mechanisms and then trains a VAE that maps the image representation of a desired coupler curve to its latent representation. Performing the k-nearest neighbor (k-NN) [21] in the latent space of the VAE, enables the generation of a variety of solutions with similar semantics as the input curve, making it adaptable to pressure-sensitive touch-based design interfaces.

Sharma and Purwar [22] extend this approach to synthesize defect-free single-degree-of-freedom spatial mechanisms for the Alt–Burmester problem by utilizing a VAE to capture the relationship between path and orientation properties. Nurizada et al. [23] further expand upon this framework by incorporating not only all types of four-, six-, and eight-bar linkages but also certain four-bars with prismatic joints, broadening the scope of synthesized mechanisms and enhancing the flexibility of the generative approach.

Chen [24] presented a novel linkage synthesis approach employing multiple deep neural networks to solve the inverse problem of linkage parameters for desired trajectory curves, using Fourier descriptors and a multisolution distribution evaluation strategy. Kapsalyamov et al. [25] introduced a DGNN to optimize dimensions and parameters for a Stephenson III six-bar linkage-based gait exoskeleton, addressing challenges in obtaining linkage parameters for simultaneous knee and ankle joint trajectories.

Vermeer et al. [26] demonstrated the synthesis of kinematic mechanisms using feature-based reinforcement learning (RL) to design straight-line mechanisms. Later, Fogelson et al. [27] used a deep reinforcement learning algorithm for generating up to 16-bar linkages with revolute joints only for path synthesis.

More recently, Nurizada and Purwar [28] explored the representation and synthesis of coupler curves in planar mechanisms using a deep neural network, comparing four common representations: Fourier descriptors, wavelets, point coordinates, and images. They used a VAE to provide a unified representation of coupler curves, regardless of the input representation, and trained a second FCNN to learn the mapping between the latent representation of coupler curves and mechanism parameters.

While this article focuses on machine learning approaches to kinematic synthesis, it is worth noting that machine learning has been applied to other domains within mechanical engineering. Raina et al. [29] proposed an RL-based truss design approach that learns from sequential human design decisions. They train an autoencoder to map trusses to a design embedding and predict possible next design states using a supervised transition network. Puentes et al. [30] enhanced this approach by learning action-sequence heuristics, while Raina et al. [31] introduced a goal-based reinforcement learning agent for further improvement. For a review of deep generative models applied to other engineering design problems, we recommend the review article by Regenwetter et al. [32].

To summarize our findings, we have compiled the aforementioned methods related to kinematic mechanism synthesis in Table 1. Most neural network (NN)-based methods [3,8,9,11–15,25,33,34] produce only a single solution approximating the desired coupler curve and sometimes rely on optimization techniques to refine these preliminary results. In addition, these methods are restricted to specific types of mechanisms, limiting their generalizability and applicability across various designs. Reinforcement-based models [26,27] take a long time to produce a single output. While advanced methods [19,20,22,24,28], utilizing generative models show promise, they are typically limited to the mechanisms present in the dataset and cannot be fully considered generative models. Some approaches involve multiple steps or even multiple networks. Purwar and Chakraborty [36] emphasized the importance of using deep neural networks to address the challenges in representing non-Euclidean mechanism data, mapping design specifications to mechanisms, and developing neural network architectures for generating diverse candidate mechanisms.

Our method, by contrast, addresses these challenges with a single model that handles two user-defined conditions: the desired coupler curve and the mechanism type. This allows it to generate multiple distinct mechanisms that trace the same input coupler curve. Trained on three types of four-bar mechanisms, the model also shows promising potential for generalizing to higher-order mechanisms and even entirely different types not present in the training data. Unlike traditional optimization-based methods, which are often computationally expensive and time consuming due to iterative searches over large parameter spaces, the conditional β-VAE model offers a significant speed advantage. Once trained, it can generate a large set of valid mechanism designs in seconds, which can then be further refined or optimized. This rapid generation enables a more efficient exploration of the design space. In addition, the integration of cross- and self-attention layers within the framework enhances the model's ability to focus on relevant features in both the input and latent spaces, improving both the quality and diversity of the generated mechanisms. Cross-attention effectively integrates condition-based information with mechanism parameters. Together, these attention mechanisms lead to more accurate and versatile synthesis compared to traditional methods, resulting in better alignment between the generated mechanisms and the desired performance criteria.

Table 1 Machine learning methods for kinematic mechanism synthesis

Reference	Mechanisms	Curve descriptor	Method	Comments
Hoskins and Kramer [3]	Crank-rocker	Curvature power spectrum	ANN + Optim	Capable of generating only one outcome and is constrained to a specific mechanism type
Vasiliu and Yannou [8]	Crank-rocker	FD	ANN	
Xie and Chen [33]	4-bars	FD	ANN	
Erkaya and Uzmay [10]	Slider-crank	CC	ANN + Optim	
Galan-Marin et al. [9]	Crank-rocker	Wavelets	ANN	
Khan et al.) [12]	4-bars	FD	ANN + Optim	
Ahamdi et al. [11]	4-bars	CC	ANN	
Li and Chen [13]	4-bars	FD	ANN	
Mo et al. [34]	Crank-rocker	FD	ANN	
Yim et al. [14]	4-bars	FD	ANN	
Kapsalyamov et al. [25]	6-bars	CC	DGNN	
Yim et al. [15]	Spatial	FD	ANN	
Chen [24]	4-, 6-bars	FD	Multiple ANNs	Utilizes multiple neural networks, suggesting potential for consolidating into a more unified framework
Yu et al. [35]	4-bars			
Vermeer et al. [26]	4-bars	CC	RL	Takes a considerable amount of time to produce a single result
Fogelson et al. [27]	4- to 16-bars	CC	GCP-HOLO	Initial results are suboptimal
Deshpande and Purwar [19]	4-, 6-bars	Image	VAE + k-NN	These approaches are constrained to the mechanisms present in the dataset, limiting their versatility
Sharma and Purwar [22]	Spatial	CC		
Nurizada et al. [23]	4-, 6-, 8-bars	Image		
Deshpande and Purwar [20]	4-bars	CC	VAE + c-VAE	Involving two separate neural networks, these approaches tend to yield similar mechanisms
Nurizada and Purwar [28]	4-bars	FD, wavelets, CC, images	VAE + MLP	

Note: FD, Fourier descriptors; CC: Cartesian coordinates; ANN, artificial neural network; RL, reinforcement learning; (c)VAE, (conditional) variational autoencoder; MLP, multilayer perceptron; k-NN, k-nearest neighbor; DGNN, deep generative neural network.

3 Variational Autoencoder, Conditional Variational Autoencoder, and Conditional β-Variational Autoencoder

The variational autoencoder (VAE) [17] is a generative model composed of an encoder that learns a probabilistic mapping from the input space to a latent space and a decoder, which maps an element of the latent space, also called latent vector, to the output space. Samples from the latent space, which has a Gaussian structure, when passed through the decoder generate new data. In the context of mechanism design, the input can be the coupler curves, which can be embedded in a latent space of much smaller dimension, thus serving as a feature or key. The outputs are also curves similar to the input, which are obtained through decoding. However, this model alone is insufficient to generate new mechanisms as it merely works as a reproducer of the coupler curves.

The loss function for a VAE is composed of two main terms: the reconstruction term and the regularization term. The reconstruction term measures the model's ability to reconstruct the input data from sampled latent vectors. It is defined as the negative log likelihood of the data given the latent representation. Mathematically, this term is represented as $-\mathbb{E}_{q(z|x)}[\log p(x|z)]$, where $q(z|x)$ is the probability distribution of latent vectors given the input data x and $p(x|z)$ is the probability distribution of reconstructing x from a sampled latent vector z.

The regularization term ensures that the latent space follows a specific distribution, typically a standard normal distribution. The Kullback–Leibler (KL) [37] divergence given by $\mathrm{KL}(q(z|x)\|p(z))$ measures the difference between the distribution $q(z|x)$ and the chosen prior distribution $p(z)$.

The regularization term penalizes deviations from the chosen prior distribution, guiding the model to learn a more structured and interpretable latent space.

The overall loss function for a regular VAE is the sum of the reconstruction term and the regularization term given by

$$\mathcal{L}_{\mathrm{VAE}} = -\mathbb{E}_{q(z|x)}[\log p(x|z)] + \mathrm{KL}(q(z|x)\|p(z)) \tag{1}$$

The reconstruction error is crucial in guiding the learning process of the VAE. Minimizing this error helps the model learn a meaningful representation in the latent space and generate outputs that resemble the original input data. The regularization term in the VAE objective also ensures that the latent space is continuous and smooth, promoting better generalization. The optimization process involves adjusting the model parameters to minimize this combined loss, resulting in an effective generative model with a structured latent space.

To provide a trade-off between reconstruction accuracy and regularization, β-VAE [38] has been proposed, which can be seen as an extension of the standard VAE with the introduction of a hyperparameter β. The loss function for β-VAE is a modification of the VAE loss, where the regularization term is scaled by the parameter β as follows:

$$\mathcal{L}_{\beta-\mathrm{VAE}} = -\mathbb{E}_{q(z|x)}[\log p(x|z)] + \beta \cdot \mathrm{KL}(q(z|x)\|p(z)) \tag{2}$$

In β-VAE, the β parameter controls the importance of the regularization term, allowing users to tune the trade-off between reconstruction fidelity and latent space representation. Higher β values encourage more disentangled representations of the latent space [38], but its effect on the generation process is still an open question in the machine learning (ML) research.

Conditional VAE (cVAE) [39] is another variation of the VAE designed for conditional generation tasks, allowing the model to generate data conditioned on additional information. The loss function for conditional VAE is an extension of the VAE loss, introducing an additional term for conditional information:

$$\mathcal{L}_{\mathrm{cVAE}} = -\mathbb{E}_{q(z|x,c)}[\log p(x|z, c)] + \mathrm{KL}(q(z|x, c)\|p(z)) \tag{3}$$

where c represents the conditional information. The reconstruction term measures the negative log likelihood of the data given the latent representation and conditional information, while the regularization term ensures a well-behaved latent space. In this article, the conditions will be the coupler curve and the type of mechanism, while the output will be a vector representation of mechanism parameters given as joint coordinates.

In this work, a combination of conditional and β-variational autoencoder architectures is employed, providing a robust framework for conditional generation tasks while incorporating probabilistic latent representations. The β-VAE component is fundamental for

learning a structured latent space, allowing for the generation of diverse samples. Simultaneously, the conditional aspect of the model, as implemented in cVAE, facilitates the generation of data conditioned on additional information, contributing to the model's adaptability in various scenarios.

The loss function utilized in the training process is a composite of components from both VAE and cVAE frameworks. Specifically, the mean-squared error (MSE) loss is employed as a reconstruction term, measuring the fidelity of the generated outputs in terms of joint locations to the ground truth values. The MSE loss ensures that the model optimally reconstructs the input, aligning with the primary objective of generating accurate and realistic joint configurations for the given mechanism.

3.1 Neural Network Architecture of Cβ-VAE. The architecture of our proposed pipeline, as depicted in Fig. 2, diverges from traditional cVAE methods, such as the one introduced by Ref. [39], which typically concatenate input and condition vectors directly before encoding. Instead, our model incorporates cross- and self-attention blocks to enhance the model's capability to process and integrate information more effectively.

The conceptual foundation for our use of attention mechanisms is rooted in their successful application across various domains, beginning with natural language processing. The landmark implementation by Ref. [40] demonstrated the potential of attention to overcome the limitations of recurrent neural networks (RNNs), leading to its rapid adoption for tasks including text classification [41], image captioning [42], and speech recognition [43]. Attention mechanisms, particularly through the advent of the transformer model [44], have revolutionized the field by showing that models can achieve state-of-the-art results without the need for RNNs, thus avoiding issues like difficulty in parallelization. Our model takes advantage of these insights, employing multihead attention [45] to capture a wide array of dependencies and relationships.

Central to our approach is the strategic use of self- and cross-attention mechanisms depicted in the left side of Fig. 3. Self-attention allows the model to evaluate the relevance of different elements within the same input vector, while cross-attention enables it to identify correlations between distinct vectors. Our model commences with the processing of type and curve conditions through separate multilayer perceptrons (MLPs) into a cross-attention mechanism. This step not only merges these conditions into a unified condition but also facilitates the learning of intricate relationships between the sequences, which are then synthesized with a latent vector through further cross-attention and self-attention processing.

The calculation of attention weights, shown in the right side of Fig. 3, involves projecting the first input vector into a query vector $\mathbf{Q}$ and the second vector into key $\mathbf{K}$ and value $\mathbf{V}$ vectors, all via separate MLPs. Attention scores are computed using $\text{softmax}\left(\frac{\mathbf{Q}\cdot\mathbf{K}^T}{\sqrt{d_k}}\right)$ [46], where d_k is determined by dividing the length of the input sequence by the number of heads. The weighted sum of values is subsequently computed as the product of the attention scores and $\mathbf{V}$ that encapsulates the attention-directed interactions between inputs.

Our pipeline further refines the integration of input characteristics and conditions. Initially, the characteristics of the input mechanism, such as type and coupler curve conditions, are standardized through an MLP into representations $\mathbf{y_t}$ and $\mathbf{y_c}$, respectively. These are then combined into a unified condition $\bar{\mathbf{y}}$ using a cross-attention block. Concurrently, the input mechanism's vector representation is enlarged through another MLP to align with the combined condition vector's dimensions.

Following this, a cross-attention block acts as an encoder, producing a latent representation vector $\mathbf{z}$ that encapsulates the input

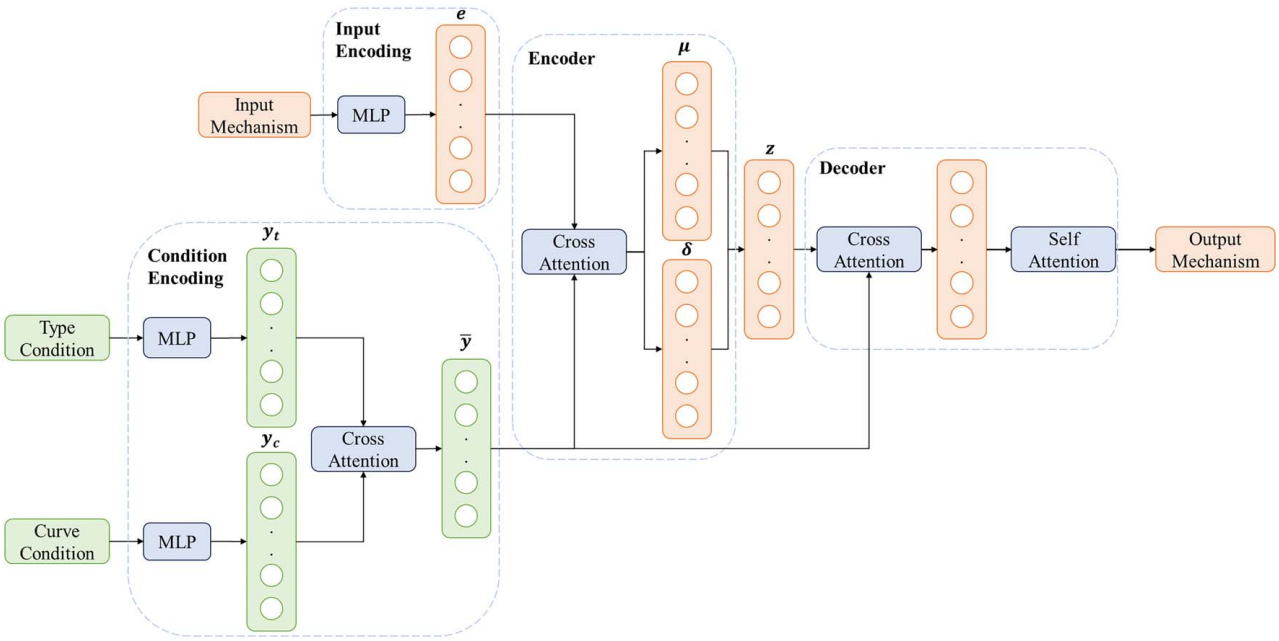

Fig. 2 The proposed conditional β-VAE model: Initially, the input mechanism's type and coupler curve conditions undergo processing through a MLP. This step ensures that their representations, denoted as $\mathbf{y_t}$ and $\mathbf{y_c}$ respectively, are standardized to equal sizes. These representations are then merged into a combined condition $\bar{\mathbf{y}}$ utilizing a cross-attention block. Meanwhile, the input mechanism vector, depicted as an array of joint locations, undergoes transformation into a larger encoded vector e via another MLP to align its dimensions with the combined condition vector. Subsequently, both vectors are fed into a cross-attention block functioning as an encoder for the variational autoencoder, resulting in the generation of a latent representation vector z for the input. This latent representation is then merged with $\bar{\mathbf{y}}$ before being inputted into a sequence of self-attention blocks, facilitating the prediction of an output mechanism closely resembling the input mechanism. During inference, the model leverages the trained conditional-β-VAE to randomly sample within the latent space. By ensuring that the conditions—output mechanism joints meet specified criteria, the model can generate different mechanisms that trace desired coupler paths while adhering to desired mechanism types.

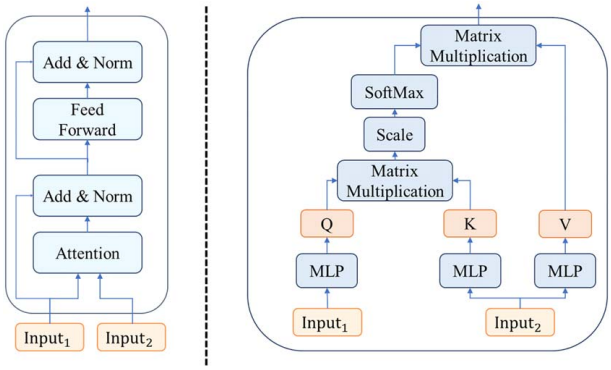

Fig. 3 Left: The attention block architecture used in the calculation of attention between two distinct input sequences. It computes attention weights, which are then added to the first input, normalized, and supplied to a feed-forward block. The resultant vector is then added residually to the original normalized vector before undergoing another normalization step. Right: The attention mechanism, shown on the right, involves mapping one input sequence to a query vector Q through an MLP, while the second input sequence is mapped to key K and value V vectors using separate MLPs. The query and key vectors undergo multiplication, scaling, softmax activation, and subsequent multiplication with the value matrix, generating weighted values that capture attention-weighted information from the input sequence(s). Self-attention blocks operate with identical input vectors, whereas cross-attention blocks accommodate different input vectors.

mechanism's characteristics. This vector is then enriched with the unified condition $\bar{\mathbf{y}}$ and processed through self-attention blocks to predict an output mechanism closely mirroring the input.

During the inference stage, the model harnesses the trained conditional-β-VAE to randomly sample within the latent space. By ensuring that the condition, i.e., output mechanism joints adhere to specified criteria, the model can generate various mechanisms that trace desired coupler paths while maintaining desired mechanism types.

In the training process, we begin by flattening 32×32 pixel images of the coupler curve into a 1024-dimensional vector. Simultaneously, the other condition, indicating the input mechanism's type, is introduced to the model as a single number: 0 for an RRRR mechanism, 1 for an RRRP mechanism, and 2 for an RRPR mechanism. This single-number representation undergoes a transformation into a larger vector, aligning its dimension with that of the image condition vector through a simple fully connected layer.

With both conditions now harmonized in size, we leverage the cross-attention mechanism, where the first input corresponds to the type condition and the second input corresponds to the curve condition. The output of this cross-attention mechanism forms the combined condition vector. Concurrently, the input vector representing the joint locations of the input mechanism also undergoes a transformation, aligning its dimension with that of the combined condition vector, achieved via a straightforward linear layer.

This transformed input vector is then fed into the encoder, constituting a separate cross-attention mechanism. Refer to Table 2 for a breakdown of each layer and module in the cβ-VAE architecture. The output of the encoder signifies the latent representation of the input. By combining this latent vector with the combined condition once again, we supply the information to the decoder, which is composed of six self-attention layers. The output of this comprehensive pipeline constitutes the reconstructed joint locations. During the inference phase, random sampling in the latent space of the trained conditional-β-VAE becomes feasible. Given the conditions, i.e., output mechanism joints satisfying both conditions, the model can generate diverse mechanisms that trace the desired coupler path and belong to the desired mechanism type.

The inner workings of the attention blocks are described in Table 3. The block consists of several key components to enable effective feature transformation and extraction. Initially, layer normalization is applied to the input sequence. Subsequently, a multihead self-attention mechanism operates on the input, allowing the model to focus on different parts of the sequence simultaneously. Following this, a linear projection is performed, and a rectified linear unit (ReLU) [47] activation function is applied to introduce nonlinearity.

To prevent overfitting and enhance generalization, a dropout layer [48] is incorporated. Another linear projection is then employed. Finally, layer normalization [49] is applied again to ensure stable and consistent output sequences. Each step in this block contributes to the overall ability of the model to capture complex relationships within the input data, facilitating effective cross-attention between different conditions in the context of the larger architecture.

The number of the self-attention blocks is a hyperparameter, and it was decided to introduce these blocks only in the decoder. Introducing the self-attention blocks in any other parts of the model would not affect the reconstruction accuracy, increase its complexity, and make the training time longer. The number of self-attention blocks in the decoder was decided to be 6, as increasing it any further did not affect the reconstruction quality.

The proposed model employs an extensive architecture, comprising a total of 49, 387, 528 parameters, and occupies 565.4MB of storage space. This complexity allows the model to capture intricate patterns and relationships within the data. All models were trained for 40 epochs on an Nvidia L40 GPU, with each epoch taking approximately 7 min to run. The learning rate was identified as a crucial parameter, with a rate of 0.0001 yielding the best results. Higher rates, such as 0.001, disrupted training, while lower rates, such as 0.00001, significantly slowed progress. The training process involved batches of 2048 samples, providing a balance between computational efficiency and model update quality. During inference, predicting 2048 mechanisms for a single-input coupler curve takes 17.2 ms $\pm$ 1.71 ms.

Table 2 Conditional-β-VAE architecture

Layer/module	Input dim	Output dim	Description
Type condition encoder	1	1024	Linear layer for label encoding
Dual condition cross-attention block	2×1024	1024	Cross-transformer block for conditioning
Joint vector encoder	8	1024	Linear layer for joints encoding
Encoder cross-attention block	2×1024	1024	Cross-transformer block for encoder attention
Mean vector calculation	1024	1024	Linear layer for calculating mean
Logvar vector calculation	1024	1024	Linear layer for calculating logvar
Reparameterization trick	2×1024	1024	Reparameterization trick for latent vector calculation
Decoder cross-attention block	2×1024	1024	Cross-transformer block for decoder attention
6 Decoder self-attention blocks	1024	1024	List of self-transformer blocks for decoder
Joint predictor	1024	8	Linear layer for joint prediction

Table 3 Attention block architecture

Layer/module	Input dim	Output dim	Description
Layer normalization 1	1024	1024	Applies layer normalization to the input sequence
Multihead attention	1024	1024	Multihead attention mechanism
Linear layer 1	1024	1024	Projection
ReLU activation	1024	1024	ReLU activation
Dropout layer	1024	1024	Dropout
Linear layer 2	1024	1024	Output projection
Layer normalization 2	1024	1024	Applies layer normalization to the output sequence

4 Model Effectiveness Criteria, Unified Cognates Generation

Three distinct criteria are employed to measure the performance of our models. The foremost criterion, emphasizing its significance, is the quality of reconstruction of predicted coupler curves when compared to the desired ones. Dynamic time warping (DTW) [50] is employed as a metric to assess the similarity between two curves represented as temporal sequences, forming the foundation for the computation of the initial criterion. The DTW allows for the comparison of sequences with different lengths by aligning corresponding points in a manner that minimizes the total distance. In the context of two sequences represented by discrete sets of points $S_1 = \{s_{1,1}, s_{1,2}, \ldots, s_{1,m}\}$ and $S_2 = \{s_{2,1}, s_{2,2}, \ldots, s_{2,n}\}$, the DTW distance $D(S_1, S_2)$ is defined as follows:

$$D(S_1, S_2) = \min_{\phi} \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} d(s_{1,i}, s_{2,j})^2 \cdot \phi(i, j)}$$

where ϕ is the alignment path, a warping function that maps indices of the sequences to each other, allowing for flexible matching between points. DTW accommodates variations in speed and time distortions, making it suitable for comparing sequences in scenarios where temporal alignment is crucial. In our approach, the primary goal is to precisely predict mechanisms that closely resemble the desired temporal sequences. Therefore, assessing the predictive performance of our model involves measuring the DTW distance between the predicted and desired temporal sequences. We use an openly accessible library tslearn [50] to calculate the DTW between two curves. Computational experiments show that a DTW value below 2 indicates satisfactory reconstruction, values between 2 and 3 are deemed moderate, and values exceeding 3 are considered indicative of poor reconstruction.

Once the initial metric of aligning the generated coupler curve with the desired one is accomplished, our focus turns to evaluating the novelty and diversity of the generated mechanisms. We define that the space of generated mechanisms is novel if they are sufficiently different from one another within a given type, while diversity measures a well-distributed representation of each type of mechanism. Given a desired coupler curve, a predetermined number of random mechanisms is generated and categorized into three groups based on their type. The novelty metric is defined by the sum of the L^2 norm of the difference between the joint coordinates of mechanisms within each type, allowing for a comparison of dissimilarity. A larger difference indicates greater dissimilarity, signifying the generation of novel mechanisms rather than repetitions of similar ones. If the difference is small, it implies redundancy, leading to the exclusion of duplicated mechanisms. This analysis is conducted on normalized versions of the mechanisms to mitigate any bias that could arise from size discrepancies, ensuring a fair comparison.

Diversity is quantified by computing the relative ratio of novel mechanisms for each type generated. Essentially, we aim to ascertain if the model excels in generating a particular type of mechanism, while potentially neglecting other types. An evenly distributed ratio of novel mechanisms for each type suggests diversity in the generated set of solutions. This threefold evaluation, encompassing approximation quality, novelty, and diversity, provides a comprehensive understanding of the model's strengths and limitations in generating an array of innovative mechanisms. This evaluation is presented in Algorithm 1.

Algorithm 1 Evaluating Mechanisms for Reconstruction Quality, Novelty, and Diversity

Input: Desired coupler curve $C_{desired} = \{(x_i, y_i)\}_{i=1}^{M}$, N generated mechanisms with their coupler curves $\{C_{gen}^{j}\}_{j=1}^{N}$, types $\{T_j\}_{j=1}^{N}$, and joint vectors $\{J_{gen}^{j}\}_{j=1}^{N}$
Output: Mechanisms that meet all criteria (m_s); diversity ratio (d_r)
Define: DTW threshold (t_{DTW}); novelty threshold ($t_{novelty}$)
Initialize: $m_s = \{RRRR : [], RRRP : [], RRPR : []\}$
for $j \leftarrow 1$ **to** N **do**
  if $DTW(C_{desired}, C_{gen}^{j}) < t_{DTW}$ **then**
    $m_s[T_j].add(J_{gen}^{j})$
  end
end
for type **in** m_s **do**
  for $i \leftarrow 0$ **to** $len(m_s[\text{type}]) - 2$ **do**
    for $j \leftarrow i + 1$ **to** $len(m_s[type]) - 1$ **do**
      if $L^2(m_s[type][i].joints, m_s[type][j].joints) < t_{novelty}$ **then**
        remove $m_s[type][j]$
      end
    end
  end
end
$d_r = \{len(m_s[RRRR]) : len(m_s[RRRP]) : len(m_s[RRPR])\}$
return m_s, d_r

5 Dataset Compilation

The engineering design community faces a significant challenge in leveraging data-driven methods, particularly deep learning, due to the limited availability of large, publicly accessible datasets. Nobari et al. [51] introduced LINKS, a dataset containing a hundred million planar linkage mechanisms designed to support data-driven research in kinematic design. The dataset features mechanisms with at least one fully rotatable link, which generate only closed coupler curves, and excludes prismatic joints. Nurizada and Purwar [28] also provide access to their dataset; however, it comprises only four-bar mechanisms with revolute joints.

To address these challenges, we decided to make our own dataset consisting of closed and open curved planar four-bar mechanisms of the following three commonly used types: RRRR, RRRP, and RRPR. The conversion of mechanisms with prismatic joints into ones with only revolute joints greatly simplifies the task of dataset construction. We represent a planar four-bar mechanism as a sequence of five joints. A four-bar mechanism comprises fixed and moving joints, along with a coupler point, together represented as a tensor $\mathbf{c} = (j_0, j_1, j_2, j_3, j_4)$, where $j_i = (x_i, y_i)$. To generate the possible locations for the joints, we define a 2D grid with values ranging between -3 and 3 in increments of 1, i.e., define a 7×7 grid. Then by fixing the j_0 at (0, 0) for all mechanisms, we vary the position of the other joints and the coupler point on the grid generating various possible mechanisms. Each potential mechanism underwent two tests before inclusion in the dataset. Any mechanism with two or more joints occupying the same grid spot was rejected to prevent the creation of a structure. Additionally, mechanisms were subjected to a second condition based on the minimum angular displacement of the actuation link, as small values lead to an unacceptably small coupler curve. Consequently, a minimum displacement of 20 deg was established as the threshold for inclusion. Figure 4 shows prototypical mechanisms of each type in our dataset.

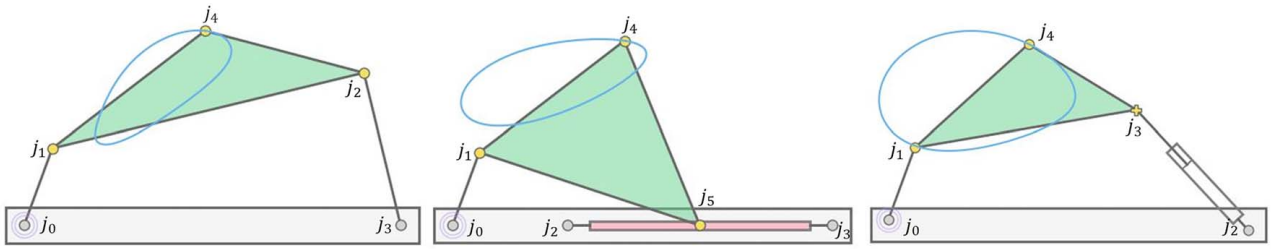

Fig. 4 From left to right: Depiction of RRRR, RRRP, and RRPR mechanisms illustrating coupler curves. Circles denote free and fixed revolute joints. Notably, j_3 in the RRPR mechanism is symbolized as a welded joint, marked by a cross. Furthermore, j_5 in the RRRP mechanism is set as the midpoint between j_2 and j_3. All mechanisms are actuated through j_0 visually highlighted by two concentric circles.

A headless server instance of MotionGen[2], which implements the unified simulation algorithm presented in Ref. [52], was used to simulate all the mechanisms. Generated coupler curves along with the corresponding locations of joints were stored in the dataset. The dataset was randomly divided, allocating 80% to the training set and the remaining 20% to the testing set. It is important to point out that other types of four-bar and higher-order mechanisms can also be created following the procedure described earlier. The current dataset is accessible[3], which we expect to augment with other types of mechanisms in the future.

5.1 Dataset Size. The maximum number of four-bar mechanisms generated through the grid-based approach can be determined given the grid dimension (N). With one fixed joint j_0 positioned at (0, 0), there are only four variable joints. Keeping in mind that no grid point can hold two joints simultaneously, the maximum number of mechanisms are given by $\prod_{i=1}^{4}(N^2 - i)$. Choosing N as 7 yields a maximum of 4, 669, 920 mechanisms.

The dataset comprises different types of mechanisms, each denoted by its joint configuration for a total of 4, 346, 626 RRRR mechanisms, 4, 211, 930 RRRP mechanisms, and 4, 619, 098 RRPR mechanisms. Difference in the generated mechanisms stems from the fact that although the grid is the same for producing all possible joint combinations, certain combinations do not pass the two aforementioned tests. Creating the whole dataset for each mechanism type took approximately 18 h on an Intel Core(TM) i7 − 9750H processor with 32 GB primary memory.

Training our model on datasets created by keeping a lower value of N resulted in overfitting [53], causing the model to practically memorize everything. Overfitting can negatively impact the model's ability to generalize to new data, and in this case, maintaining a higher value of N helps mitigate this issue. However, while a smaller value of N led to rapid overfitting, a larger value leads to a combinatorial explosion of the dataset. Therefore, a good compromise was achieved by setting N to 7.

5.2 Normalization Process. Each coupler curve underwent normalization for translation, scale, and rotation invariance; see Ref. [28] for details. Initially, the curve's path is adjusted by subtracting the mean $(\bar{x}, \bar{y})$. Following this adjustment, the path is scaled by dividing through the Cartesian X and Y directions' root-mean-squared variance. To eliminate dependence on the path's initial orientation, it is rotated to align its principal components with the coordinate system's axes. No normalization was applied to mirrored mechanisms, treating them as distinct entities in the latent space. The normalized curve is finally embedded inside a 32×32 gray-scale image and used as a representation for the original coupler curve.

[2]http://www.motiongen.io
[3]https://www.kaggle.com/datasets/purwarlab/four-bar-coupler-curves

5.3 A Unified Approach to Creating Cognates. As we will see later, while our conditional-β-VAE model generates a multitude of mechanism design concepts, there is an opportunity to create their cognates to expand on the solution space. We have recently proposed a method for unified simulation of planar N-bar mechanisms [52], which converts mechanisms with one or more prismatic joints into an all revolute jointed mechanism. This method led to a single simulation algorithm at the expense of controllable approximation inaccuracy. We now show that this technique can also help us create cognates of RRPR and RRRP mechanisms without having to resort to different methods.

For the RRRP mechanism, the slot and slot joint relationship can be considered as a PR-dyad, as shown in Fig. 5. In the conversion phase, a third joint is added to the slot link, and the slot itself is removed. This third joint is also called the *far joint* because it is far from the workspace of the mechanism so that the slot joint travels only on the arc that is close to the straight line defined by the two end joints. Furthermore, the far joint is fixed to the ground because the slot link shares two joints with the ground link. As a result, we can find an RRRR subgraph in this kinematic structure, where one of the fixed joints is the far joint. Additionally, such property can be used in the cognate generation phase.

For the RRPR mechanism, the conversion is done similarly, but it has a welded joint as shown in the left-most mechanism of Fig. 6. The combination of the piston and the welded joint can be replaced by a slot and two slot joints, as shown in the middle of Fig. 6. As a result, we can now create a far joint for the mechanism. Additionally, we find that when these two joints connect with the far joint and are part of the moving link, they form one rigid body. Together with the ground link and two moving binary links, we can find an

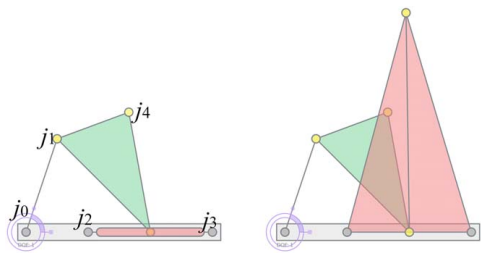

Fig. 5 This figure shows the conversion of the RRRP mechanism into an approximate RRRR mechanism. The slot link is fixed to the ground and is converted into a triad. Additionally, a binary link connects the far joint and the slot joint to represent the slot's linear constraint. For the drawing purposes, the far joint (top right) is shown closer to the origin than when the kinematic structure is simulated in reality. Furthermore, this far joint is considered fixed to the ground because its associated link shares two joints with the ground link. The location of the far joint determines the quality of approximation. In the sample generation phase, the slot joint is placed at the midpoint of line joining two slot joints.

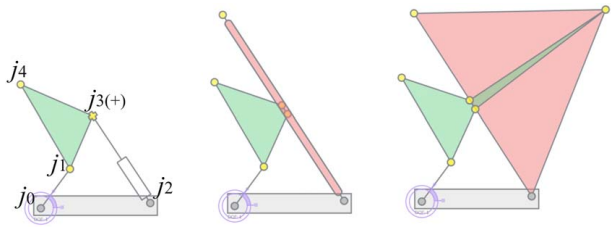

Fig. 6 This image shows the conversion of the RRPR mechanism. Notably, the welded joint is considered as two slot joints in the same slot. The far joint is drawn much closer to the origin in the figure to show its existence. When the far joint is far from the origin, two slot joints" trajectories are close to two straight curves in the slot. Furthermore, the two slot joints and the far joint form a triangle together, and their relative positions with respect to the coupler point are unchanged throughout the linkage's motion. Additionally, j_3 is denoted with a "+" sign to indicate that it is a welded joint, i.e., its two corresponding links are rigidly connected.

RRRR subgraph inside the kinematic structure. This RRRR subgraph is formed by the following five joints: the actuator ground joint, the actuated joint, the other ground joint, the far joint, and the coupler point.

5.4 Cognate Generation. The cognate generation of RRRR planar linkages is based on several geometric similarities inside a bent Cayley diagram [54]. These cognates can generate the same coupler curve, while they have different dimensional parameters from each other.

We will first look at the cognate generation process for RRRR mechanism, which uses four similar triangles and three parallelograms as shown in Fig. 7. As a result, we can see three four-bar linkage graphs in the Cayley diagram. They are constructed by $(j_0, j_1, j_2, j_3, j_4)$, $(j_0, p_1, p_5, p_2, j_4)$, and $(j_2, p_3, p_5, p_4, j_4)$. These three four-bar linkages produce the same coupler curve.

The P-joint conversion presented earlier converts the slot into a combination of revolute joints, in which an RRRR subgraph exists. Specifically, a revolute joint is used to approximate the straight-line constraint, which is also called a far joint because the line constraint is represented with a large circular arc. For RRRP, j_3 in Fig. 7 is the joint in the slot (or slot joint), and j_2 is the far joint. This means that $j_2 j_3$ is theoretically an infinitely long link. Then, it is easy to deduce that $\triangle j_4 p_3 p_4$ has one side that is infinite in length, which is not practical for design. Furthermore, the binary link $p_2 p_5$ is also infinitely long, which can be seen as another

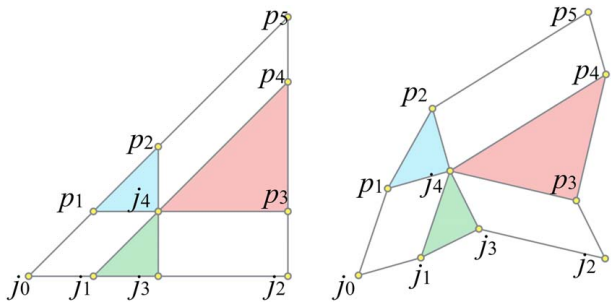

Fig. 7 The left and right images show the diagram and when this diagram is bent, respectively. The three triangles in the two images represent the moving triad of the RRRR planar linkage. Three triangles, namely, $\triangle j_1 j_3 j_4$, $\triangle p_1 j_4 p_2$, and $\triangle j_4 p_3 p_4$, are similar. Additionally, $\triangle j_0 j_2 p_5$ is also similar to the three triangles. Furthermore, there are three parallelograms in the diagram, which are $j_0 j_1 j_4 p_1$, $j_3 j_2 p_3 j_4$, and $p_2 j_4 p_4 p_5$. These geometric property holds even when the diagram is bent, as shown in the right image.

P-joint, and we can convert R-joint back to P-joint to form an RRRP mechanism. Therefore, for one RRRP mechanism formed by $(j_0, j_1, j_2, j_3, j_4)$, we can always find another cognate formed by $(j_0, p_1, p_5, p_2, j_4)$.

However, when we apply the same conversion method to RRPR, no additional practical RRPR cognate exists. Our tests show that the results are either (1) the other fixed joint (j_2 or p_5) coincides with j_0 or (2) the other fixed joint is too far away from j_0 to be a practical design. The reason is that during the P-joint conversion, j_3 is infinitely far away from both j_1 and j_4, creating an extremely skewed triangle. Then, the binary links like $j_3 j_4$, $j_3 j_2$, $j_4 p_3$, $p_3 p_4$, and $p_2 p_5$ are infinitely long. Rotating the binary link $j_2 j_3$ slightly can result in the position j_2 and/or j_5 infinitely far from j_0, or they coincide with j_0 in some extreme cases. Additionally, when the two fixed joints coincide, the mechanism is in a singular configuration, i.e., the mechanism can move even if the slot joint is not sliding in the slot. These two results are consistent with Ref. [55].

6 Results and Discussion

Upon completion of the training, the decoder can be used to generate new mechanisms under the guidance of user-specified conditions. Users can provide a coupler curve and solicit the generation of n mechanisms approximating the provided curve, regardless of their respective types or specify both the coupler curve and the type of desired mechanisms. In this scenario, the model conducts the sampling of n random vectors within its latent space, along with an optional type conditional vector characterized by values of 0, 1, and 2, representing distinct mechanism types. Consequently, mechanisms of various types are produced. Thus, users retain the option to generate only specific types of mechanisms by configuring type conditions in accordance with their preferences.

However, it is imperative to note that the model occasionally yields suboptimal outcomes, resulting in unacceptable deviation between the resulting coupler curve and the desired input. An ablation study, particularly focused on the β value selection during the training process, was undertaken. In our investigation, we considered six different β values: 1, 25, 50, 75, 90, and 100. The comparison of results and the determination of the optimal β value are facilitated by the calculation of three proposed metrics: reconstruction quality, novelty, and diversity.

To ascertain the reconstruction accuracy of the trained models, we randomly select 1000 coupler curves from the testing dataset and generate 100 mechanisms of various types for each coupler curve. By calculating the average DTW values between the input coupler curves and those of the generated mechanisms, we find certain trends in the results, as illustrated in Table 4. This table encapsulates the mean DTW value across all results, along with its median value for a specific β value.

Computational experiments indicate that a DTW value below 2 signifies a satisfactory reconstruction. Consequently, the table also includes the percentage of generated mechanisms with a DTW value below 2, relative to the total number of generated mechanisms. A consistent pattern emerges across all presented results: the model with a β value of 1 produces subpar outcomes, achieving

Table 4 Average DTW and percentage acceptable for 1000 coupler curves from the testing dataset and 100 randomly generated mechanisms for each coupler curve

	$\beta = 1$	$\beta = 25$	$\beta = 50$	$\beta = 75$	$\beta = 90$	$\beta = 100$
Mean DTW	6.462	4.047	4.131	4.283	4.298	4.470
Median DTW	5.850	2.441	2.594	2.724	2.739	2.864
Percentage of satisfactory Mechanisms (DTW < 2)	12.7	45.1	43.2	41.7	41.5	41.2

Table 5 Average number of novel mechanisms of each type per 100 randomly generated mechanisms from 1000 coupler curves for varying novelty threshold and β values

Novelty threshold ↓		$\beta = 1$	$\beta = 25$	$\beta = 50$	$\beta = 75$	$\beta = 90$	$\beta = 100$
0.1	RRRR	3	6	5	4	3	3
	RRRP	2	5	4	3	2	1
	RRPR	3	5	4	3	2	3
	Total	*8*	*16*	*13*	*10*	*7*	*7*
0.2	RRRR	5	5	4	3	3	2
	RRRP	4	5	3	2	2	1
	RRPR	5	5	4	3	2	2
	Total	*14*	*15*	*11*	*8*	*7*	*5*
0.3	RRRR	4	4	3	2	2	2
	RRRP	3	3	2	2	1	1
	RRPR	4	3	3	2	2	2
	Total	*11*	*10*	*8*	*6*	*5*	*5*
0.4	RRRR	3	3	2	2	2	2
	RRRP	3	2	1	1	1	1
	RRPR	3	3	2	2	2	2
	Total	*9*	*8*	*5*	*5*	*5*	*5*
0.5	RRRR	2	2	2	2	2	1
	RRRP	2	2	1	1	1	1
	RRPR	2	2	2	2	1	1
	Total	*6*	*6*	*5*	*5*	*4*	*3*

a satisfactory results in only every nine mechanisms. Subsequently, the results notably improve with a β value of 25, where approximately every second approximation yields a satisfactory result. However, as the β value increases beyond 25, the results progressively deteriorate. It is imperative to underscore that increasing the β value beyond 100 adversely affects the training process, potentially accounting for the diminishing quality of results as the value increases.

After evaluating the quality of the curve generation, our subsequent focus is on comparing the novelty and diversity of the generated results. Using the same coupler curves employed for calculating reconstruction accuracy, we compute the mean square difference between the respective joints of all mechanisms within each mechanism type. If the difference between two mechanisms falls below a specified threshold, one of them is randomly discarded until the differences between all mechanisms surpass the threshold. We experiment with threshold values of 0.1, 0.2, 0.3, 0.4, and 0.5.

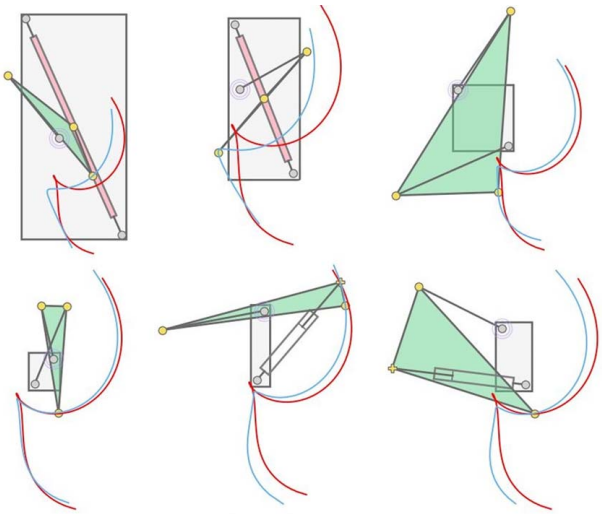

Fig. 8 The input (desired) open coupler curve and six mechanisms (two of each type) with generated coupler curves

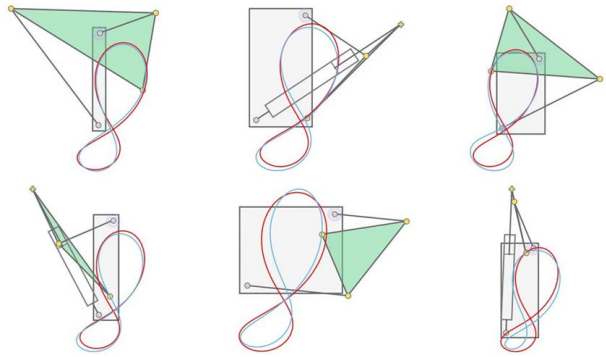

Fig. 9 The input (desired) closed coupler curve and six mechanisms (three each of type RRRR and RRPR) with generated coupler curves

A higher threshold imposes a stricter constraint on novelty, resulting in fewer novel mechanisms across all trained models. Results are presented in Table 5.

With a low threshold of 0.1, the model trained with a β value of 25 produces the most diverse results, generating 16 distinct mechanisms approximating the input coupler curve. As the β value increases, the number of novel mechanisms decreases. A significant increase in the threshold substantially reduces the number of generated novel mechanisms. For instance, at a threshold value of 0.5, the number of novel mechanisms generated for $\beta = 25$ is only 6. With a threshold set to 1 (not in the table), only one novel mechanism

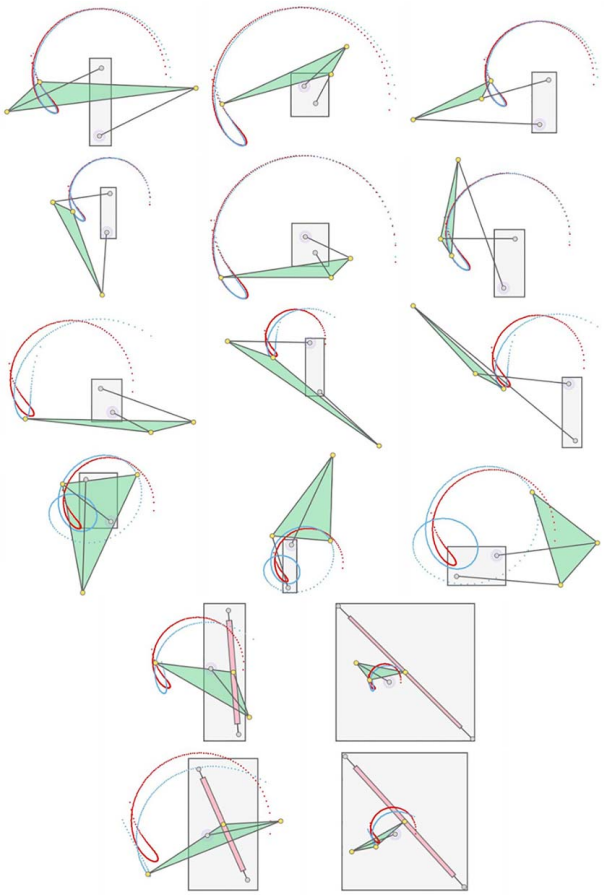

Fig. 10 Left column shows mechanisms generated using our c-VAE model with cognates drawn in the second and third column. There is only cognate for each of the RRRP mechanisms in the last two rows.

remains. It is important to note that, regardless of the threshold and β value, all trained models consistently provide diverse results, indicating that none of the mechanism types are underrepresented.

By combining these findings, we conclude that the model trained with a β value of 25 yields the most balanced results, achieving sufficiently accurate reconstruction with novel and diverse mechanisms. Further refinement of our results is achieved by computing cognates for the generated mechanisms, providing two additional mechanisms for each RRRR mechanism and one additional mechanism for the RRRP type. Figures 8 and 9 display a few selected example results generated by the decoder for a threshold value of 0.5 ($\beta = 25$). It can be seen from these figures that while the desired curves are not exactly matched, an optimization routine could easily use these as initial solutions to find a better fit. Moreover, there is a good diverse set of solutions, which can help a designer choose the one that best fits their performance requirements, such as transmission angle, relative size of links, location of pivots, and load.

Figure 10 shows another example where the left-most column shows mechanisms generated by our model, while their cognates are shown in the second and third columns. The results reveal similarities between mechanisms in the second and third rows. This occurrence is attributed to our model generating two coupler mechanisms. Specifically, the middle cognate mechanism in the second row resembles the first (generated) mechanism in the third row, while the first (generated) mechanism in the second row is similar to the second cognate mechanism in the third row. This is a result of the dataset generation process, where no constraints were imposed to ensure the exclusion of the cognate mechanisms in the dataset. As mentioned earlier, our cognate generation algorithm produces only one useful mechanism for RRRP mechanisms; thus, for every RRRP mechanism generated, we only show a single computed cognate linkage.

7 Conclusions and Future Work

The exploration of linkage synthesis for path, function, and motion generation in mechanical engineering has revealed significant challenges with traditional approaches. These traditional methods, while widely applied, suffer from limitations such as slow optimization processes, dependence on initial conditions, and lack of guaranteed outcomes, often failing to accurately capture the desired trajectory of a given path. The introduction of machine learning techniques, especially artificial neural networks, has offered a promising alternative, allowing for the use of continuous paths as synthesis data and enabling the rapid generation of diverse mechanism designs.

This work advances the field by introducing a novel deep generative model that combines conditional and β-VAEs, offering a more efficient and versatile approach to linkage synthesis. By integrating conditional embeddings tailored to specific mechanism types and coupler curves, our model not only accommodates a broader range of design requirements but also facilitates the generation of multiple, diverse solutions. This approach represents a significant shift from existing methods, moving toward a more intuitive and user-friendly design process that prioritizes novel concept generation over precise path matching.

The proposed model has been rigorously evaluated using a hierarchical set of metrics focusing on the reconstruction quality, novelty, and diversity of the generated mechanisms. The generated mechanisms exhibit diverse types and effectively approximate the specified input coupler curves. We presented six different models trained with varying β values, with the results indicating that the model trained with $\beta = 25$ outperformed others. The results underscore the model's ability to produce a wide array of novel and diverse mechanisms, demonstrating its potential to significantly enhance the conceptual design stage in mechanism design.

Furthermore, the inclusion of a unified algorithm for computing cognates represents an additional innovation, expanding the utility and efficiency of the model by generating multiple cognate solutions for each predicted mechanism. This feature, along with the development of an extensive dataset and the introduction of quantitative assessment metrics, underscores the comprehensive nature of this work.

The development of this novel methodology not only addresses the limitations of traditional methods but also opens new avenues for research and application in mechanism design. Future research directions may explore extending the proposed method to different types of linkage systems, including other types of four-bar mechanisms, as well as higher-order mechanisms with more links and joints as well as spherical and spatial mechanisms.

Acknowledgment

This publication has research support from a National Science Foundation STTR Phase II award (#2126882) to Co-PI Purwar who also holds stocks in Mechanismic Inc. The research findings included in this publication may or may not necessarily relate to the interests of Mechanismic Inc. The terms of this arrangement have been reviewed and approved by Stony Brook University in accordance with its policy on objectivity in research.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The data and information that support the findings of this article are freely available.[4]

Nomenclature

Abbreviations

ANN	=	artificial neural network
cβ-VAE	=	conditional β variational autoencoder
CC	=	Cartesian points
CNN	=	convolutional neural network
DGNN	=	deep generative neural network
DNN	=	deep neural network
DTW	=	dynamic time warping
FCNN	=	fully connected neural networks
GMDH	=	group method of data handling
k-NN	=	k-Nearest neighbor
KL	=	Kullback–Leibler
ML	=	machine learning
MLNN	=	multilayered neural network
MLP	=	multilayer perceptron
MSE	=	mean square error
NN	=	neural network
ReLU	=	rectified linear unit
RNN	=	recurrent neural networks
VAE	=	variational autoencoder

Symbols

$\bar{\mathbf{y}}$	=	combined condition representation
P	=	prismatic
R	=	revolute
$\mathbf{K}$	=	key vector
$\mathbf{Q}$	=	query vector
$\mathbf{V}$	=	value vector

[4]See Note 3.

$\mathbf{y_c}$ = coupler curve condition representation
$\mathbf{y_t}$ = type condition representation
β = variable representing beta value

References

[1] Wampler, C. W., Morgan, A. P., and Sommese, A. J., 1992, "Complete Solution of the Nine-Point Path Synthesis Problem for Four-Bar Linkages," ASME. J. Mech. Des., **114**(1), pp. 153–159.
[2] Bai, S., and Angeles, J., 2015, "Coupler-Curve Synthesis of Four-Bar Linkages Via a Novel Formulation," Mech. Mach. Theory, **94**(1), pp. 177–187.
[3] Hoskins, J., and Kramer, G., 1993, "Synthesis of Mechanical Linkages Using Artificial Neural Networks and Optimization," International Conference on Neural Networks (ICNN), San Francisco, CA, Mar. 28–Apr. 1.
[4] Zahn, C. T., and Roskies, R. Z., 1972, "Fourier Descriptors for Plane Closed Curves," IEEE Trans. Comput., **C-21**(3), pp. 269–281.
[5] Chuang, G.-H., and Kuo, C.-C., 1996, "Wavelet Descriptor of Planar Curves: Theory and Applications," IEEE Trans. Image Process., **5**(1), pp. 56–70.
[6] Mcgarva, J. R., 1994, "Rapid Search and Selection of Path Generating Mechanisms From a Library," Mech. Mach. Theory, **29**(2), pp. 223–235.
[7] Mcgarva, J., and Mullineux, G., 1993, "Harmonic Representation of Closed Curves," Appl. Math. Model., **17**(4), pp. 213–218.
[8] Vasiliu, A., and Yannou, B., 2001, "Dimensional Synthesis of Planar Mechanisms Using Neural Networks: Application to Path Generator Linkages," Mech. Mach. Theory, **36**(2), pp. 299–310.
[9] Galan-Marin, G., Alonso, F. J., and Del Castillo, J. M., 2009, "Shape Optimization for Path Synthesis of Crank-Rocker Mechanisms Using a Wavelet-Based Neural Network," Mech. Mach. Theory, **44**(6), pp. 1132–1143.
[10] Erkaya, S., and Uzmay, I., 2008, "Optimization of Transmission Angle for Slider-Crank Mechanism With Joint Clearances," Struct. Multidiscipl. Optim., **37**(5), pp. 493–508.
[11] Ahmadi, B., Nariman-zadeh, N., and Jamali, A., 2016, "Path Synthesis of Four-Bar Mechanisms Using Synergy of Polynomial Neural Network and Stackelberg Game Theory," Eng. Optim., **49**(6), pp. 932–947.
[12] Khan, N., Ullah, I., and Al-Grafi, M., 2015, "Dimensional Synthesis of Mechanical Linkages Using Artificial Neural Networks and Fourier Descriptors," Mech. Sci., **6**(1), pp. 29–34.
[13] Li, X., and Chen, P., 2017, "A Parametrization-Invariant Fourier Approach to Planar Linkage Synthesis for Path Generation," Math. Probl. Eng., **2017**(1), pp. 1–16.
[14] Yim, N. H., Lee, J., Kim, J., and Kim, Y. Y., 2021, "Big Data Approach for the Simultaneous Determination of the Topology and End-Effector Location of a Planar Linkage Mechanism," Mech. Mach. Theory, **163**(1), p. 104375.
[15] Yim, N. H., Ryu, J., and Kim, Y. Y., 2023, "Big Data Approach for Synthesizing a Spatial Linkage Mechanism," International Conference on Robotics and Automation (ICRA), London, UK, May 29–June 2, pp. 7433–7439.
[16] Verstraten, E., 2012, *Cognate Linkages the Roberts–Chebyshev Theorem*, Springer, Netherlands, pp. 505–519.
[17] Kingma, D. P., and Welling, M., 2014, "Auto-Encoding Variational Bayes," Computing Research Repository, abs/1312.6114.
[18] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., 2014, "Generative Adversarial Networks," Conference on Neural Information Processing Systems, Montreal, Canada, Dec. 8 –13.
[19] Deshpande, S., and Purwar, A., 2020, "An Image-Based Approach to Variational Path Synthesis of Linkages," ASME J. Comput. Inf. Sci. Eng., **21**(2), p. 021005.
[20] Deshpande, S., and Purwar, A., 2019, "Computational Creativity Via Assisted Variational Synthesis of Mechanisms Using Deep Generative Models," ASME. J. Mech. Des., **141**(12), p. 121402.
[21] Taunk, K., De, S., Verma, S., and Swetapadma, A., 2019, "A Brief Review of Nearest Neighbor Algorithm for Learning and Classification," International Conference on Intelligent Computing and Control Systems (ICCS), Madurai, India, May 15–17.
[22] Sharma, S., and Purwar, A., 2022, "A Machine Learning Approach to Solve the Alt-Burmester Problem for Synthesis of Defect-Free Spatial Mechanisms," ASME J. Comput. Inf. Sci. Eng., **22**(2), p. 021003.
[23] Nurizada, A., Dhaipule, R., Lyu, Z., and Purwar, A., 2025, "A Dataset of 3M Single-DOF Planar 4-, 6-, and 8-bar Linkage Mechanisms With Open and Closed Coupler Curves for Machine Learning-Driven Path Synthesis," ASME J. Mech. Des., **147**(4), p. 041701.
[24] Chen, C.-H., 2023, "Application of Multiple Deep Neural Networks to Multi-Solution Synthesis of Linkage Mechanisms," Machines, **11**(11), p. 1018.
[25] Kapsalyamov, A., Hussain, S., Brown, N. A., Goecke, R., Hayat, M., and Jamwal, P. K., 2023, "Synthesis of a Six-Bar Mechanism for Generating Knee and Ankle Motion Trajectories Using Deep Generative Neural Network," Eng. Appl. Artif. Intell., **117**(1), p. 105500.
[26] Vermeer, K., Kuppens, R., and Herder, J., 2018, "Kinematic Synthesis Using Reinforcement Learning," ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE), Quebec City, Quebec, Canada, Aug. 26–29.
[27] Fogelson, M. B., Tucker, C., and Cagan, J., 2023, "GCP-HOLO: Generating High-Order Linkage Graphs for Path Synthesis," ASME J. Mech. Des., **145**(7), p. 073303.
[28] Nurizada, A., and Purwar, A., 2024, "An Invariant Representation of Coupler Curves Using a Variational AutoEncoder: Application to Path Synthesis of Four-Bar Mechanisms," ASME J. Comput. Inf. Sci. Eng., **24**(1), p. 011008.
[29] Raina, A., McComb, C., and Cagan, J., 2019, "Learning to Design From Humans: Imitating Human Designers Through Deep Learning," ASME J. Mech. Des., **141**(11), p. 111102.
[30] Puentes, L., Raina, A., Cagan, J., and McComb, C., 2020, "Modeling a Strategic Human Engineering Design Process: Human-Inspired Heuristic Guidance Through Learned Visual Design Agents," 16th International Design Conference, Virtual, Oct. 26–29, pp. 355–364.
[31] Raina, A., Cagan, J., and McComb, C., 2022, "Design Strategy Network: A Deep Hierarchical Framework to Represent Generative Design Strategies in Complex Action Spaces," ASME J. Mech. Des., **144**(2), p. 021402.
[32] Regenwetter, L., Nobari, A. H., and Ahmed, F., 2022, "Deep Generative Models in Engineering Design: A ReviewJ," ASME J. Mech. Des., **144**(7), p. 071704.
[33] Xie, J., and Chen, Y., 2007, "Application Back Propagation Neural Network to Synthesis of Whole Cycle Motion Generation Mechanism," 12th IFToMM World Congress, Besançon, France, June 17–21, pp. 17–21.
[34] Mo, X., Ge, W., Zhao, D., and Zhang, Y., 2019, *Path Synthesis of Crank-Rocker Mechanism Using Fourier Descriptors Based Neural Network*, Springer, Singapore, p. 32.
[35] Yu, S.-C., Chang, Y., and Lee, J.-J., 2022, "A Generative Model for Path Synthesis of Four-Bar Linkages Via Uniform Sampling Dataset," Proc. Inst. Mech. Eng., Part C: J. Mech. Eng. Sci., **237**(4), pp. 811–829.
[36] Purwar, A., and Chakraborty, N., 2023, "Deep Learning-Driven Design of Robot Mechanisms," ASME J. Comput. Inf. Sci. Eng., **23**(6), p. 060811.
[37] Kullback, S., and Leibler, R. A., 1951, "On Information and Sufficiency," Ann. Math. Stat., **22**(1), pp. 79–86.
[38] Irina, H., Matthey, L., Pal, A., Christopher, B., Xavier, G., Matthew, B., Shakir, M., and Alexander, L., 2016, "beta-VAE: Learning Basic Visual Concepts With a Constrained Variational Framework," 5th International Conference on Learning Representations, Toulon, France, Apr. 24–26.
[39] Sohn, K., Lee, H., and Yan, X., 2015, "earning Structured Output Represen-tation using Deep Conditional Generative Models," 29th Conference on Neural Information Processing Systems, Montreal, Canada, Dec. 7–12.
[40] Bahdanau, D., Cho, K., and Bengio, Y., 2014, "Neural Machine Translation by Jointly Learning to Align and Translate," 3rd International Conference on Learning Representations, San Diego, CA, May 7–9.
[41] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E., 2016, "Hierarchical Attention Networks for Document Classification," Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, June 12–17, pp. 1480–1489.
[42] Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., and Zhang, L., 2018, "Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering," IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, June 18–22.
[43] Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., and Bengio, Y., 2015, "Attention-Based Models for Speech Recognition," 28th Conference on Neural Information Processing Systems (NeurIPS 2015), Montreal, Canada, Dec. 7–12.
[44] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I., 2017, "Attention Is All You Need," 31st Conference on Neural Information Processing Systems (NeurIPS 2017), Long Beach, CA, Dec. 4–9.
[45] Cordonnier, J., Loukas, A., and Jaggi, M., 2020, "Multi-Head Attention: Collaborate Instead of Concatenate," 9th International Conference on Learning Representations (ICLR 2021), Virtual Conference, May 3–7.
[46] Pearce, T., Brintrup, A., and Zhu, J., 2021, "Understanding Softmax Confidence and Uncertainty," 37th Conference on Uncertainty in Artificial Intelligence, Virtual Conference, July 27–30.
[47] Agarap, A. F., 2018, "Deep Learning using Rectified Linear Units (ReLU)," CoRR.
[48] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., 2014, "Dropout: A Simple Way to Prevent Neural Networks From Overfitting," J. Mach. Learn. Res., **15**(1), pp. 1929–1958.
[49] Ba, J. L., Kiros, J. R., and Hinton, G. E., 2016, "Layer Normalization."
[50] Tavenard, R., Faouzi, J., Vandewiele, G., Divo, F., Androz, G., Holtz, C., Payne, M., Yurchak, R., Rußwurm, M., Kolar, K., and Woods, E., 2020, "Tslearn, A Machine Learning Toolkit for Time Series Data," J. Mach. Learn. Res., **21**(118), pp. 1–6.
[51] Nobari, A. H., Srivastava, A., Gutfreund, D., and Ahmed, F., 2022, "LINKS: A Dataset of a Hundred Million Planar Linkage Mechanisms for Data-Driven Kinematic Design," International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, St. Louis, MO, Aug. 14–17.
[52] Lyu, Z., Purwar, A., and Liao, W., 2024, "A Unified Real-Time Motion Generation Algorithm for Approximate Position Analysis of Planar N-Bar Mechanisms," ASME. J. Mech. Des., **146**(6), p. 063302.
[53] Ying, X., 2019, "An Overview of Overfitting and Its Solutions," J. Phys.: Conf. Ser., **1168**(1), p. 022022.
[54] Sëylemez, E., 2023, *Four-Bar and Slider-Crank Cognates*, Springer Nature, Switzerland, pp. 265–285.
[55] Dijksman, E. A., and Smals, A. T., 2000, "How to Exchange Centric Inverted Slider Cranks With λ-Formed Four-Bar Linkages," Mech. Mach. Theory, **35**(2), pp. 305–327.