An Image-Based Approach to Variational Path Synthesis of Linkages

Shrinath Deshpande
Computer-Aided Design and Innovation Lab,
Department of Mechanical Engineering,
Stony Brook University,
Stony Brook, NY 11794-2300
e-mail: shrinath.deshpande@stonybrook.edu

Anurag Purwar[1]
Computer-Aided Design and Innovation Lab,
Department of Mechanical Engineering,
Stony Brook University,
Stony Brook, NY 11794-2300
e-mail: anurag.purwar@stonybrook.edu

This paper brings together computer vision, mechanism synthesis, and machine learning to create an image-based variational path synthesis approach for linkage mechanisms. An image-based approach is particularly amenable to mechanism synthesis when the input from mechanism designers is deliberately imprecise or inherently uncertain due to the nature of the problem. In addition, it also lends itself naturally to the creation of a unified approach to mechanism synthesis for different types of mechanisms, since for example, images are formed from a collection of pixels, which themselves could be generated from a four-bar or six-bar. Path synthesis problems have generally been solved for a set of precision points on the intended path such that the designed mechanism passes through those points. This approach usually leads to a small set of over-fitted solutions to particular precision points. However, most kinematic synthesis problems are concept generation problems, where a designer cares more about generating a large number of plausible solutions, which could reach given precision points only approximately. This paper models the input curve as a probability distribution of image pixels and employs a probabilistic generative model to capture the inherent uncertainty in the input. In addition, it gives feedback on the input quality and provides corrections for a more conducive input. The image representation allows for capturing local spatial correlations, which plays an important role in finding a variety of solutions with similar semantics as the input curve. This approach is also conducive to implementation for pressure-sensitive touch-based design interfaces, where the input is not a zero-thickness curve, but the sweep of a small patch on the finger. [DOI: 10.1115/1.4048422]

Keywords: deep generative models, path synthesis, planar mechanisms, machine learning, deep learning

1 Introduction

Kinematic synthesis of mechanisms dates back to centuries. However, a systematic treatment of the synthesis process has been largely missing from the literature. True synthesis of mechanisms requires that both the type of the mechanism, which refers to the number of joints and links and their pattern of interconnection, and the dimensions of the links and locations of the joints, be determined exclusively from the given input. Modern literature in mechanism design and analysis draws from volumes of text by Sandor and Erdman [1] and McCarthy and Soh [2]. The other important texts in this area are written by Suh and Radcliffe [3], Hartenberg and Denavit [4], Lohse [5], and Hunt [6].

This paper is concerned with the path synthesis, where the objective is to move the coupler along a given path or through a set of prescribed points. Generally, this problem is solved via optimization, wherein a measure of error between the given path and the synthesized path is minimized and dimensions are obtained in the process. To make this mathematically easier, the input path is represented by a set of precision points. This way of setting up the problem heavily relies on the placement of precision points. However, in many practical cases, the precision points do not possess the level of significance bestowed by the computational methods. Possibly, the motivation behind proposing the precision point approach is rooted in its tractability for digital computation. However, it imposes an artificial constraint on problem specification, which further limits the solution possibilities.

More recently, we have proposed an algebraic fitting approach using kinematic mapping for simultaneous type and dimension synthesis of mechanisms, see Deshpande and Purwar [7] and Ge et al. [8]. We have also developed a cross-platform application that implements our synthesis and simulation algorithms for planar linkages [9]. In Refs. [10,11], we have addressed the inherent uncertainty in the user input and intelligently manage it to obtain a variety of solutions. This paper presents a novel approach that models the input path as a discrete probability distribution. This image-based representation allows us to employ recent advances in computer vision and pattern recognition. The outcome is a probability distribution of conducive input possibilities, which efficiently enumerates alternative paths that highly correlate with the raw uncertain input. Also, our deep generative model suppresses the infeasible portion of the input image and modifies it into a more conducive input, thereby proving feedback on the input. The approach is independent of the algebraic complexity of the coupler curve (CC) and thus enables extension to different types of mechanisms. Deep learning techniques can learn arbitrarily complex spatial signals making it a scalable approach to learning all possible coupler curves.

Modeling the input path as a signal is a well-established idea in theoretical kinematics. Nolle and Hunt [12] proposed an optimization-based analytical method for coupler curve generation of planar four-bar mechanisms. Ullah and Kota [13] presented an approach using simulated annealing, which did not require simultaneous matching of all characteristics of the path and produced near-globally optimized solutions. More recently, Wu et al. [14] have presented a method using finite Fourier series for path generation of four-bar linkages. Buskiewicz et al. [15] used the curvature of the coupler curve as the shape descriptor for path synthesis using genetic algorithms. Khan et al. [16] have presented an approach where an artificial neural network learns the mapping between Fourier coefficients of the coupler curve and mechanism parameters.

[1]Corresponding author.

Manuscript received March 11, 2020; final manuscript received August 9, 2020; published online October 13, 2020. Assoc. Editor: Bin He.

In the above approaches, the input path is assumed as a one-dimensional input by parameterizing the input path as a function of time. It is advantageous in cases where the sequential nature of the path is important. In addition, these approaches work only when the given curve shape is traceable by a mechanism. They provide little to no guidance on the direction a user should take if acceptable results are not produced. Our proposed approach models the input as a 2D spatial signal which allows capturing non-sequential spatial correlations. Besides, classical approaches generally work for closed curves only, which leaves out a large possibility of solutions. In contrast, the proposed approach works for open as well as closed curves.

In the classical path generation approach, a Fourier series approximation of input path is computed. Then, an optimization routine returns a linkage that closely approximates this task curve. This procedure narrows down the search space aiming to obtain a single accurate solution. On the contrary, our proposed approach takes in a raw, and often imprecise input and computes a distribution of plausible inputs that capture the spatial correlations. This is enabled by first creating a standard form of all coupler curves and representing them as images, which are further fed to a deep generative machine learning model, called Variational AutoEncoder (VAE). The VAE generates sampling of new curves by using a probability distribution of latent features, which are learned by the model over time. Then, these plausible samples are used to find nearest neighbors in a database of four-bar and six-bar linkages. The result is a large set of diverse solutions in contrast to a single optimal solution. Figure 1 shows an overview of this approach. In the succeeding sections, we will present details of each of the components shown in the figure.

To the best of authors' knowledge, this is the first attempt to create (1) an image-based representation of input path which captures spatial correlations in the input path and provides a probabilistic estimate, (2) representation of coupler curves of different algebraic order in a unified way, and (3) convolutional neural network-based state-of-the-art deep generative models that learn joint probability distribution of coupler curve images to generate variational inputs and provide the user feedback on the quality of input.

The rest of this paper is organized as follows. Section 2 presents the details of image-based representation of coupler curves. Section 3 reviews the theory of convolutional neural networks. It also presents the motivation behind using the deep generative models along with the details of its architecture. Section 5 presents the applications of VAE for image-based path generation problems.

2 Representing Coupler Curves As An Image

It is well-known that the coupler curves of linkage mechanisms are algebraic curves of degree 6 or more. In the case of the path generation problem, the task is to find a CC that best fits a set of precision points. Thus, for all practical purposes, CC is represented as a discrete set of N points in a plane given by,

$$\mathrm{CC} = \{x_i, y_i\}_{i=1}^{N}$$

A common approach is to treat this sequence of points as a one-dimensional temporal signal with parameter t given by

$$\mathrm{CC} = \{x(t), y(t)\}$$

This formulation allows for applying various signal processing techniques like Fourier Analysis. While this type of modeling has many advantages, which are evident by the success in path generation, it fails to capture spatial correlations that are not implicit in temporal correlations.

The proposed image-based representation allows us to capture those correlations in exchange for losing temporal correlations between the input sequences. However, it can be argued that the ordering is embedded in the spatial distribution of pixels since unlike typical computer vision applications, such as facial recognition, which have a true 2D set of pixels for images, a curve is fundamentally a one-dimensional entity only. In our previous work, we have used standard VAE, which is trained on a dataset where each data sample is a sequence of cartesian points. In that case, the input must be a fixed dimensional vector which requires each sequence to have the same length. This makes it incompatible to work with unordered point clouds with a varying number of points. In contrast to this, the image-based representation provides a way to convert this information to a fixed size spatial matrix which is invariant to the order and number of points in the sequence. Thus, an image-based representation offers solutions for the problems where the order of input points is not important and thus should not bias the synthesis process. To summarize, the image-based task does not need a specific ordering of points, and thus allows for capturing all possible input orders in a unified way.

2.1 Standard Form and Pixel Mapping.

In this paper, an image represents a probabilistic coupler curve, where the intensity of a pixel represents the likelihood of containing a coupler curve point. First, each coupler curve is normalized with respect to scale, position, and orientation.

Normalization. First, the path points are subtracted by the mean $(\bar{x}, \bar{y})$, which results in a translated curve. The translated curve is divided by the root-mean-squared variance in Cartesian X- and Y-directions. To make the representation invariant to orientation, the curve is rotated such that its principal component axes are aligned with the coordinate axes. Figure 2 depicts the normalization procedure for two couple paths.

The principal component axes are eigenvectors of the covariance matrix C of a coupler path comprising m two-dimensional

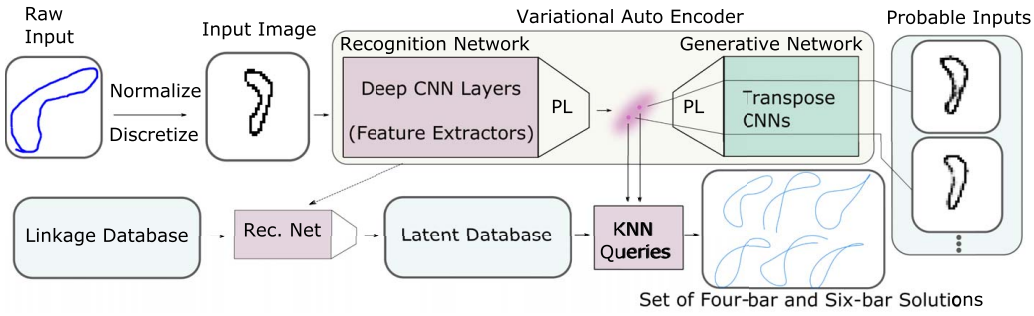

Fig. 1 Raw input is normalized and discretized into an image. This image is passed through the recognition network of a CNN-based trained VAE which computes a probability distribution of latent features describing conducive variations of the raw input. Samples from this distribution are queried to find nearest neighbors in a dataset of four-bar and six-bar linkages. The linkages corresponding to the nearest neighbors have coupler curves similar to the input.

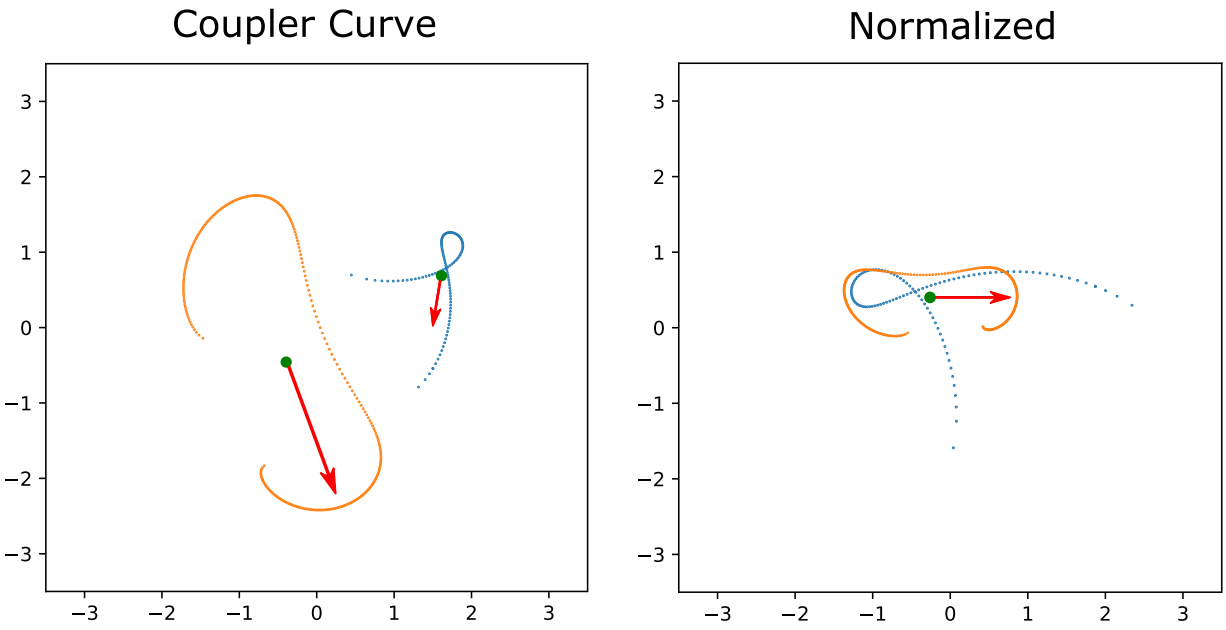

Fig. 2 Each curve is normalized for position, scale, and orientation

points $\{x_j, y_j\}_{j=1}^{m}$, where C is given by,

$$C = \begin{bmatrix} C_{xx} & C_{xy} \\ C_{yx} & C_{yy} \end{bmatrix}, \quad \text{where} \tag{1}$$

$$C_{xx} = \frac{1}{m}\sum_{i=1}^{m}(x_i - \bar{x})(x_i - \bar{x}) \tag{2}$$

$$C_{yy} = \frac{1}{m}\sum_{i=1}^{m}(y_i - \bar{y})(y_i - \bar{y}) \tag{3}$$

$$C_{xy} = C_{yx} = \frac{1}{m}\sum_{i=1}^{m}(x_i - \bar{x})(y_i - \bar{y}) \tag{4}$$

Discretization. The normalized planar curve is now discretized into a 2D image. The two-dimensional Cartesian space is linearly discretized into $S-1$ partitions along each axis, where S corresponds to a side of the image. The partitions along X axis is done as follows: x coordinates that lie between $-\infty$ and $-X_{lim}$ are binned into the first partition. The next partition includes x ranging from $(-X_{lim}, -X_{lim}+X_{step}]$ and so on up to the last partition $(X_{lim}, +\infty)$. Here, X_{lim} and X_{step} are discretization parameters, which control the range and resolution, respectively. Figure 3 depicts the discretization of a normalized curve. It can be noted that the extreme coordinates play an important role in deciding resolution and range of pixels.

To estimate X_{lim}, a statistical study is conducted on a dataset of 6818 linkages consisting of four-bar (2188), Stephenson I (2754), and Stephenson III (1876) six-bar linkages. The coupler curve for each linkage is stored as a sequence of points sampled per one

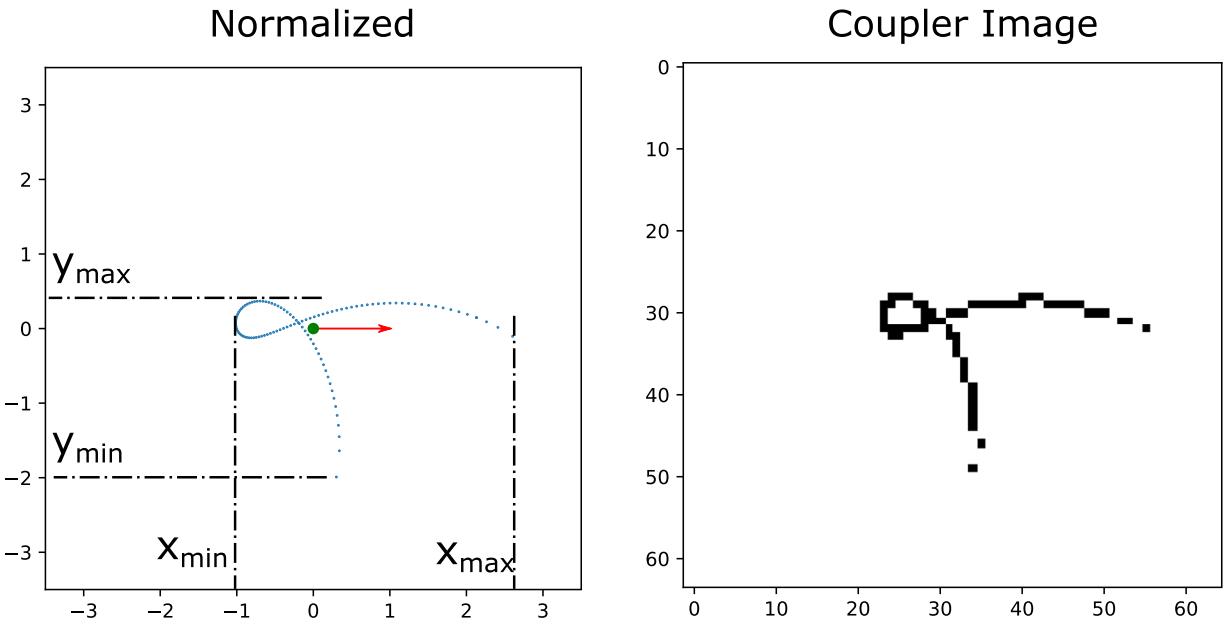

Fig. 3 The extreme coordinates (x_{min}, x_{max}, y_{min}, y_{max}) of a normalized coupler curve are shown. If a point lies in a Cartesian span of pixels, it is given 1.0 probability.

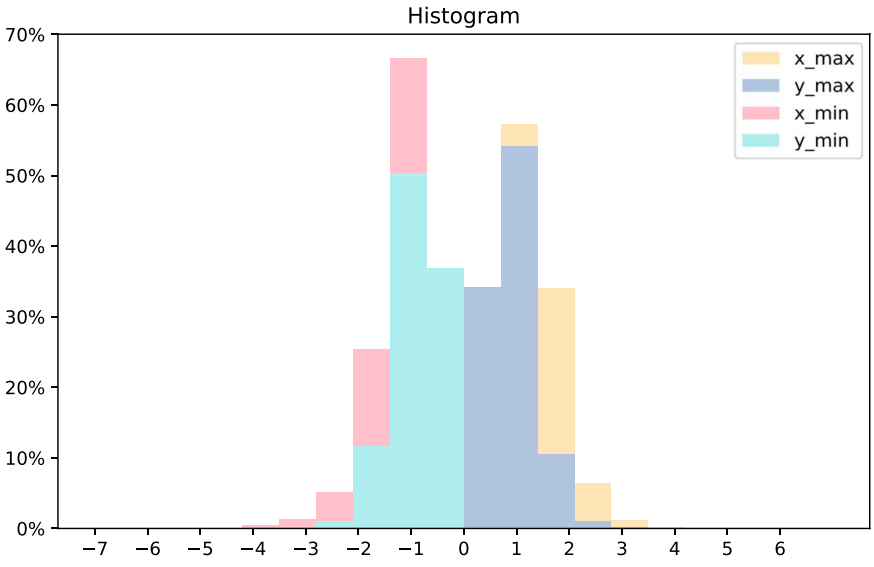

Fig. 4 A histogram of extreme coordinates (x_{min}, x_{max}, y_{min}, y_{max}) of coupler curves. It can be seen that less than 1% of the normalized coupler curves have a point with an absolute coordinate larger than 3.5 units.

degree change in the crank angle. The number of points in coupler curves is different for different mechanisms. Figure 4 shows the histogram of extreme coordinates corresponding to the normalized coupler curves from the database. It can be seen that more than 99% of linkages are contained by a square of a side 7 unit and its center at the origin. Thus, X_{lim} is set as $7/2 = 3.5$. For an image with $S \times S$ pixels, X_{step} is given by

$$X_{step} = \frac{2X_{lim}}{S-1}$$

Here, each pixel of an image represents a probability of containing at least one point of the coupler curve. In this paper, we choose $S = 64$, which provides sufficient resolution to capture the variation in coupler curves. It should be noted that the discretization process causes some loss of information. However, in many cases, the input is inherently vague. Thus, it acts as a buffer and prevents the further process from over-emphasizing the numerical precision. One can also notice that image-based representation removes the order of points in the curve. However, it grants spatial correlations between neighboring points, which helps in capturing interesting spatial patterns. In Sec. 3, we review some of the techniques that are used to learn and recognize such spatial patterns.

3 A Review of Deep Learning and Generative Models

Capturing patterns in the images is a well-researched topic, see Forsyth and Ponce [17]. The computer vision literature has witnessed a vast amount of research ranging from classical computer vision approaches to deep convolutional networks. Classical computer vision approaches use hand-crafted feature extractors for capturing spatial correlations, whereas deep convolution networks learn these feature extractors from the training data. Subsection 3.1 presents a brief overview of convolution neural networks.

3.1 Convolutional Neural Networks. Convolutional neural networks (CNNs) are constructed to capture the spatial structure of the input [18]. CNNs consist of a set of learnable filters. In the 2D case, these filters can be represented by 2D matrices, whose coefficients are updated at each step of the training process. At the time of inference, each CNN filter is convolved around the image. A high score is reported for a strong pattern match between the filter and the input. The high score gets passed on to the next layer through max pool operation.

In simple words, the output of each layer in CNN is an agglomeration of scores for various learned filters used during the convolution. This enables CNN to learn low-level spatial features like an edge in the initial layers. As the layers go deep, the composition of simple features constitutes highly complex spatial structures depending upon the training data.

Convolution Operation. Convolutional layers use convolution operation between two entities. In the continuous case, the convolution of two functions f and g is given by

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\,d\tau = \int_{-\infty}^{\infty} f(t-\tau)g(\tau)\,d\tau \tag{5}$$

In the discrete case, it is replaced by the sum

$$(f * g)(n) = \sum_{m=-\infty}^{\infty} f(m)g(n-m) = \sum_{m=-\infty}^{\infty} f(n-m)g(m) \tag{6}$$

Here, g is called a kernel function. In 2D discrete case, g can be represented via a 2D matrix which has support on $\{(-M, -N), \ldots, (M, N)\}$. Then, convolution is given by

$$(f * g)(x, y) = \sum_{m=-M}^{M} \sum_{n=-N}^{N} f(x-n, y-m)g(m, n) \tag{7}$$

Equation (7) has a very simple geometric interpretation as shown in Fig. 5. A 2D filter is slid along the image with a stride. Before each sliding action, a sum is conducted and stored into an output tensor at a location governed by the number of strides. The final outcome of such operations is a 2D matrix. This entire process is repeated for k number of filters resulting in k number of 2D matrices as shown in the figure.In CNN, the learning parameters are the coefficients of kernel matrices. These coefficients are updated via the Backpropagation algorithm in a stochastic gradient descent method of optimization. For more details, please see Refs. [19,20].

3.2 Deep Generative and Recognition Models. In this section, we present a latent variable generative model that learns an approximate distribution of possible coupler curves observed

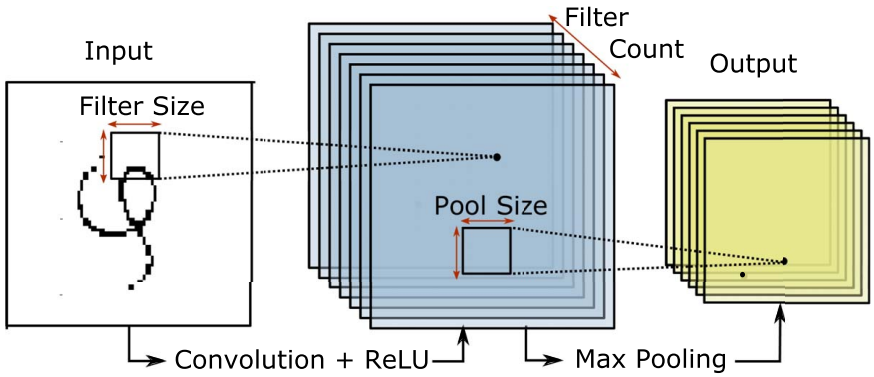

Fig. 5 A filter is a 2D matrix of whose coefficients are learnable parameters of the convolution network. During convolution, each such filter is convolved on the input followed by nonlinear ReLU activation resulting in *k* such 2D matrices as shown by the middle entity. Next, the output is max pooled to formulate the output to be passed on to the next layer.

from a dataset. In the latent variable generative model theory, it is assumed that the observed data are generated via a natural process that involves hidden variables known as latent variables. A companion to a generative model is a recognition model, which can infer the latent variable corresponding to the observed coupler curve. Then the task is to learn the joint distribution of observed and latent variables.

Learning such a distribution facilitates an understanding of the kind of curves a linkage can or cannot generate. This is based on the assumption that the database contains enough representative curves. If such a generative model and inference model exist or can be learned, it can be employed to perform a host of tasks such as input denoising, input recognition, and generation of curves similar to the input but more likely to be generated by a linkage. In this work, we have used VAE [21] as our generative modeling framework which can be used to train such a generative-inference model pair efficiently.

3.3 Variational AutoEncoders. VAE [21] is a machine learning model, which learns to approximate the true distribution of an observed data X. In this work, VAE is trained to learn the distribution of coupler curves in the form of images. Thus, here X is an $S \times S$ image, where S refers to the number of pixels along each of the two dimensions. Figure 6 shows a sample architecture of VAE. It consists of a Recognition Model, which infers a probability distribution of latent variables for a given observed variable X, whereas the Generative Model generates an image using a sample drawn from latent variable space. In this figure, the recognition model consists of multiple stacks of convolutional, ReLU, and Max Pool layers. At the end of the final stack, a fully connected layer outputs two vectors μ and σ, respectively. These two vectors correspond to parameters of multivariate Gaussian distribution. The generative model takes a sample vector z from the Gaussian distribution and outputs an image $\hat{X}$ given by

$$pr(z) = \mathcal{N}(0, 1) \tag{8}$$

$$\hat{X} = G(z; \theta_g) \tag{9}$$

where θ_g are weight parameters of the generative model. The task for inference model is to infer salient attributes z based on observed X, which can be expressed by conditional probability $pr(z|X)$

$$pr(z|X) = \frac{pr(X|z)pr(z)}{pr(X)} \tag{10}$$

where the abbreviation $pr(A)$ represents probability of variable A. Unfortunately, computing probability of X (i.e., $pr(X)$) is usually intractable because it involves computing the integral $\int pr(X|z)pr(z)\mathrm{d}z$. However, we can apply variational inference [22] to estimate the conditional probability distribution $pr(z|X)$. We approximate $pr(z|X)$ by a tractable distribution $q(z|X)$, which we define such that it can be computed by a neural network Q. The prior distribution of z is assumed to have a tractable form such as multivariate Gaussian with mean 0 and variance 1.

$$\mu, \sigma = Q(X; \theta_e) \tag{11}$$

$$q(z|X) = \mathcal{N}(\mu, \sigma) \tag{12}$$

Here, $\mathcal{N}(\mu, \sigma)$ is a multivariate Gaussian distribution function with mean μ and variance σ. Now, we want to find parameters of Recognition Model $Q(X)$ that predicts the distribution $q(z|X)$ such that it is very similar to $pr(z|X)$. Then, we can use it to perform approximate inference of the intractable distribution.

The objective is to find parameters θ_g, θ_e of $G(z)$ and $Q(X)$, respectively, such that our model generates samples as close as

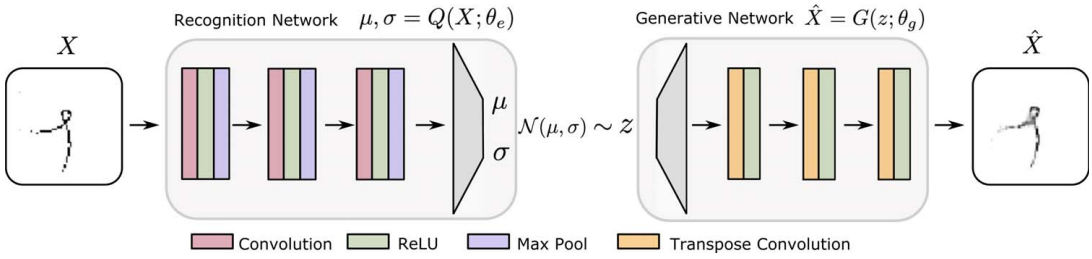

Fig. 6 Recognition model encodes the observed data *X* into a multivariate distribution of *z*. Generative model takes samples from this distribution to generate output $\hat{X}$, the entire process is called as an approximate marginal inference of *X*.

true observed distribution and distribution $q(z|X)$ is close to true distribution $p(z)$. This is achieved by training the neural network models for maximizing the expectation lower bound (ELBo) of the marginal likelihood, which is given by

$$L_{ELBo}^{(X^i)} = \mathbb{E}_{Q(z|X^i;\theta_e)}(\log(p(X^i|z))) - D_{KL}(Q(z|X^i;\theta_e)\|p(z)) \quad (13)$$

Here, the first term on the right-hand side represents reconstruction likelihood and the second term is called Kullback–Leibler divergence (KL divergence) [23], which forces the inferred distribution of z for the dataset to be similar to the assumed prior distribution $p(z)$. Since we assume that $p(z)$ is a multivariate Gaussian distribution with mean 0 and variance 1, the lower bound of marginal likelihood becomes,

$$L_{ELBo}^{(X^i)} = -L_{\text{Cross Entropy}}(\hat{X}, X) - \left(\sum_{i}^{k} \sigma_i^2 + \mu_i^2 - \log(\sigma_i) - 1\right) \quad (14)$$

where $L_{\text{Cross Entropy}}$ between $\hat{X}$ and X is given by

$$L_{\text{Cross Entropy}} = -X\log(\hat{X}) + (X-1)\log(1-\hat{X}) \quad (15)$$

The training objective is given by

$$\arg\min_{\theta_e,\theta_g}(-L_{ELBo}^{(X^i)}) \quad (16)$$

For further details on VAE, please see Ref. [21]. Once, entire VAE is trained, the recognition model and generative model can be used separately or together depending upon the application.

4 Data Generation

One of the advantages of mechanism synthesis domain over other domains such as the domain of natural images or natural language processing is the availability of clean synthetic data. We exploit this advantage to create a carefully tuned dataset to improve the prediction performance of the deep learning models. One can raise an issue that the machine learning theory is built upon the underlying assumption that the data should be identical, independently distributed (i.i.d.). However, we argue that the goal of our application is not to capture true underlying distribution, which is skewed towards more commonly observed circular paths but to capture the representation capacity of planar linkages in terms of the variety of tasks they can accomplish.

In light of this objective, we formulate a data generation scheme that allows only sufficiently novel samples to be added into the dataset. Figure 7 depicts the data generation process. The process starts by picking a linkage topology. Random values for linkage parameters are sampled from a uniform distribution such that the ratio between the maximum and minimum link length does not exceed 30 and the crank can rotate at least 30 deg. The coupler curve is calculated using forward kinematics and is passed for novelty check. The novelty check can either be done by comparing all other curves already present in the dataset or by using a VAE that is trained on that dataset. Both of the approaches work well, but the latter approach is more efficient when the size of the dataset grows larger.

It should be noted that this way of generative data is not imperative to the proposed approach. However, we observed performance improvements after incorporating this scheme. This scheme can be applied to other data-driven tasks as long as the data generated is guaranteed to be error-free. In this case, we use forward kinematic solvers to compute the coupler curve X, which is guaranteed to be noiseless.

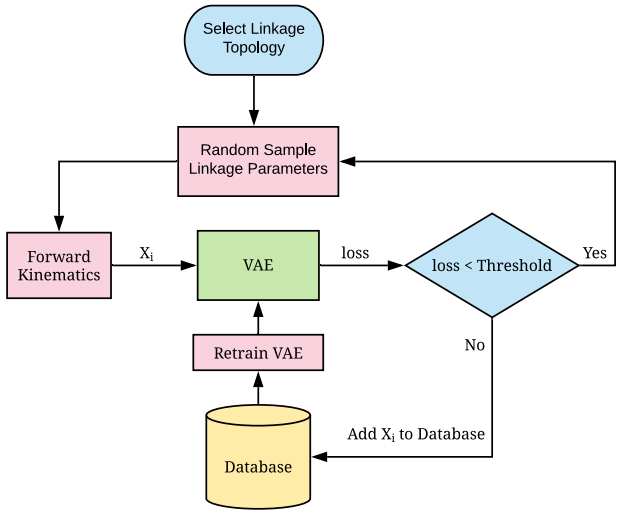

Fig. 7 The first step is to select the topology. A sample is drawn from the uniform distribution of the linkage parameters. This sample is used to test the performance of VAE. If the loss is higher than a threshold, the sample is added to the dataset. This prevents the data imbalance and improves the performance of the neural network on rare but legitimate input.

5 Variational AutoEncoder for Coupler Images

This section presents the architecture and training details of the VAE used for the coupler image recognition and generation task. The training goal is to maximize the variational lower bound L_{ELBo} given in Eq. (13) by changing the weights of recognition and generative models via stochastic gradient descent. Maximizing the variational lower bound increases the likelihood of a given image to be the image of an actual coupler curve. Higher the likelihood, the more accurate the pixel intensities would get.

5.1 Architecture. The input to the encoder is a 64×64 image in pixel size. This input is fed to three stacks of convolutional-MaxPooling layers. The output of the convolutional layer is first passed through ReLU activation before it can be max pooled. Max pooling layer passes the highest activation in the filter window to the next layer. The hyper-parameters for each layer are given in Table 1. After three such layers, the output is flattened into a single vector, which then is connected to a single fully connected layer which outputs two vectors representing μ and $\log\sigma$ of 50 dimensions each. Vector z is obtained by sampling a multivariate Gaussian distribution with mean μ and standard deviation σ. This vector is passed to the generative model as an input.

Table 1 Recognition network architecture

Layer	Filter count	Filter size	Stride	Output shape	Activation
Convolution 1	32	(11, 11)	(1,1)	(64, 64, 32)	ReLU
Max Pool 1	–	(2,2)	(2,2)	(32, 32, 32)	–
Convolution 2	64	(5, 5)	(1,1)	(32, 32, 32)	ReLU
Max Pool 2	–	(2,2)	(2,2)	(16, 16, 64)	–
Convolution 3	128	(3, 3)	(1,1)	(16, 16, 128)	ReLU
Max Pool 3	–	(2,2)	(2,2)	(8, 8, 128)	–
Flatten 1	–	–	–	8192	–
Fully connected	–	–	–	100	–
Split	–	–	–	(2, 50)	(–, exponential)

Table 2 Generative network architecture

Layer	Filter count	Filter size	Stride	Output shape	Activation
Fully connected	–	–	–	1024	ReLU
Reshape	–	–	–	(8, 8, 16)	–
Transpose convolution 1	128	(3, 3)	(2,2)	(16, 16, 128)	ReLU
Transpose convolution 2	64	(5, 5)	(2,2)	(32, 32, 64)	ReLU
Transpose convolution 3	32	(11, 11)	(2,2)	(64, 64, 32)	ReLU
Transpose convolution 3	1	(3, 3)	(1,1)	(64, 64, 1)	Sigmoid

The generative model architecture is composed of transpose-convolution layers to finally output a tensor similar to that of an image. The architecture and hyper-parameters are given in Table 2.

5.2 Training. Figure 8 shows improvement in reconstructions of coupler image samples from the test set as the training progresses. It can be seen that the pixels with a higher probability of containing a coupler point are assigned a higher probability and vice versa. Once the network is trained sufficiently, the encoder learns to capture spatial correlations and returns a probability distribution in latent feature space. Each sample from this distribution has a high probability of being a valid coupler image. This feature is used to obtain feasible representations of raw input and is demonstrated in Sec. 5.3.

5.3 Raw Input Recognition and Reconstruction. Raw user input is defined as an image with a different distribution of pixel intensities than the ones the VAE has learned while training. If such an input image is fed to VAE, the recognition network tries to find the learned patterns in the input. The patterns with high correlation with learned patterns will receive a high score that will be passed through max-pooling layers and eventually reflect in the obtained latent distribution. If samples from that distribution are passed through the generative model, it will produce coupler curve-like images that share similar spatial attributes. Figure 9 shows examples where VAE takes in raw inputs and computes feasible inputs that are used for downstream tasks for path generation. It can be seen that VAE shifts the probability assignments on the pixels of the raw input. Figure 9 highlights this effect on various raw inputs. This is used to provide feedback on the user input. Figure 10 depicts the outcome of VAE construction for two different inputs. The input shown on the top can be closely approximated by a four-bar mechanism, whereas the other input is highly unlikely for a four-bar or six-bar to interpolate. Since the VAE is trained on coupler curves of four-bar and six-bar linkages, it can reconstruct the original input with high accuracy. On the other hand, second input is highly modified by VAE to change it into a more conducive input. The difference between the modification done by VAE is reflected in the percentage change in the histogram of pixel intensities. For the first image, almost 70% of the pixels retain their original 100% intensity, whereas that number is a mere 17% for the second input.

5.4 Nearest Latent Neighbors for Variational Path Synthesis. In Sec. 5.3, it was shown that the recognition network captures the spatial correlations that highly correspond to that of coupler curve images. To take advantage of this fact, the database is remapped into a 50-dimensional space where each mechanism is represented by the latent vector obtained using the recognition model. Since the latent vector is obtained from the coupler image, the latent database can consist of any type of linkage with any number of links and joints. Synthesis is performed as a search for neighbors in this database. At the time of synthesis, raw input is passed through the recognition model to find a distribution of possible latent vectors. The linkages that correspond to the K-Nearest neighbors to the mean of this distribution are returned as potential solutions. Figure 11 presents some of the solutions obtained for

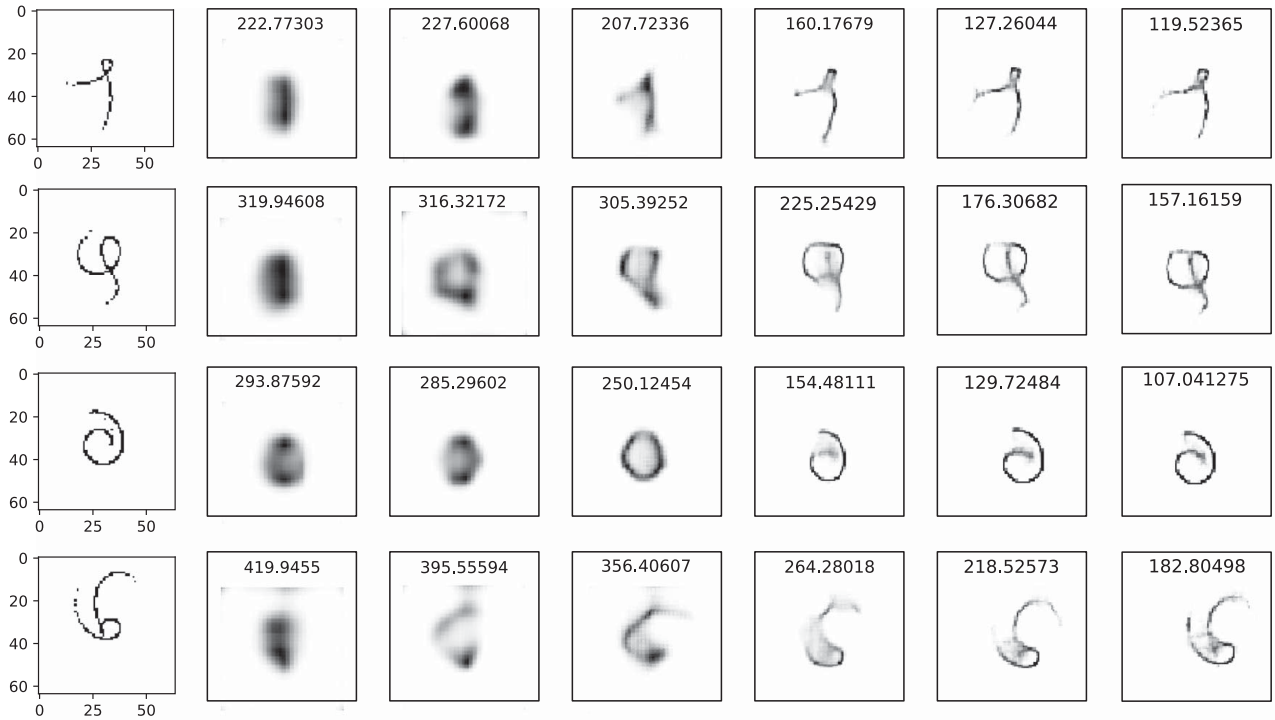

Fig. 8 The training progress is reflected in the reconstruction quality of test coupler curve images. The corresponding variational lower bound $L^{x'}_{ELBo}$ is displayed above each reconstructed image. It can be seen that the probability assignment by VAE for each pixel improves over the training.

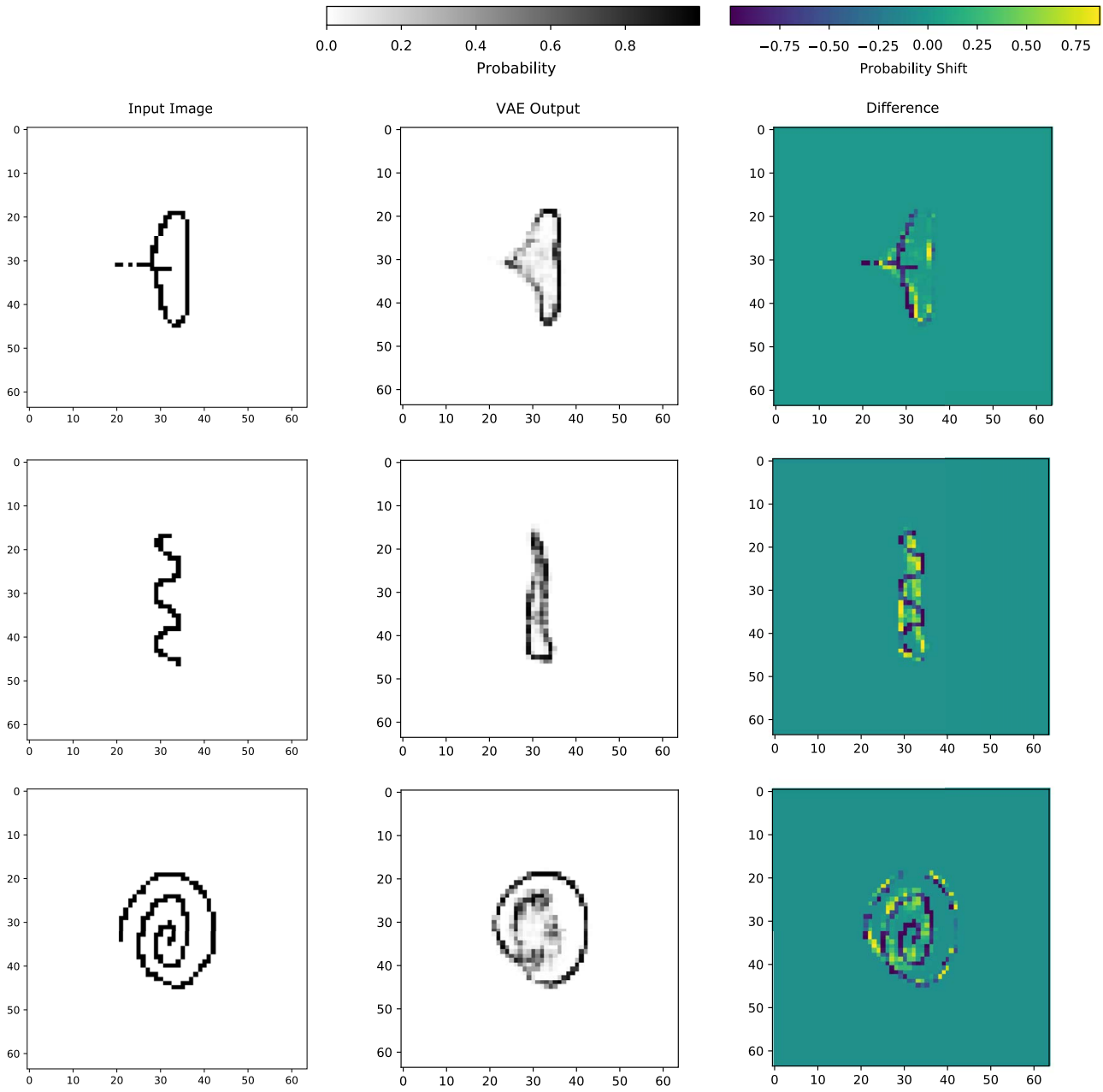

Fig. 9 Raw input image is passed through VAE which provides a feasible input image. The shift in the probabilities is highlighted in the rightmost images. The shift shown in the dark indicates that the portion of raw input is highly unlikely for linkage from the dataset to interpolate. This provides visual and intuitive feedback to the user on the input.

the nearest neighbor query on raw user input. These solutions represent good candidates for further fine-tuning. The next subsection presents the effect of different latent dimensions on various factors such as training loss, k-nearest neighbors, and varying levels of noise in the input.

5.5 Effect of Latent Dimensions on Latent Neighbors. This case study compares the performance of VAE with 50 (L_{50}), 24 (L_{24}), and 10 (L_{10}) dimensional latent spaces. Figures 12, 13, and 14 show L_{ELBo}, reconstruction, and KL-divergence losses on the held-out samples for the three VAEs. It can be seen in Figs. 13 and 14 that the higher latent dimension achieves better reconstruction, while suffering from higher KL divergence loss. Higher KL divergence indicates a sparse distribution of latent variables. While it is true that the visual similarity between two images constructed from two points in the latent space is proportional to their distance, the variation in similarity is not uniform in all directions of the latent space. Figure 15 presents a comparative study. It can be seen in the last example presented in Fig. 15 that L_{50} yields a more visually similar neighbor than L_{24} and L_{10}. We suspect that the high dimensionality of latent space makes it easy for the encoder to push the latent embeddings of visually dissimilar coupler images apart from each other. Figure 16 depicts the effect of input corruption, where a valid coupler curve is incrementally corrupted by a high-frequency and low-frequency noise respectively. It can be seen that the perturbations can transform the input into another possible coupler curve, which is reflected in the closest neighbor returned by VAEs. Figure 17 presents three different metrics by which nearest neighbors are calculated. It can be seen that the closest neighbor is the same regardless of the choice of metric. However, the next neighbors can be different if a different metric

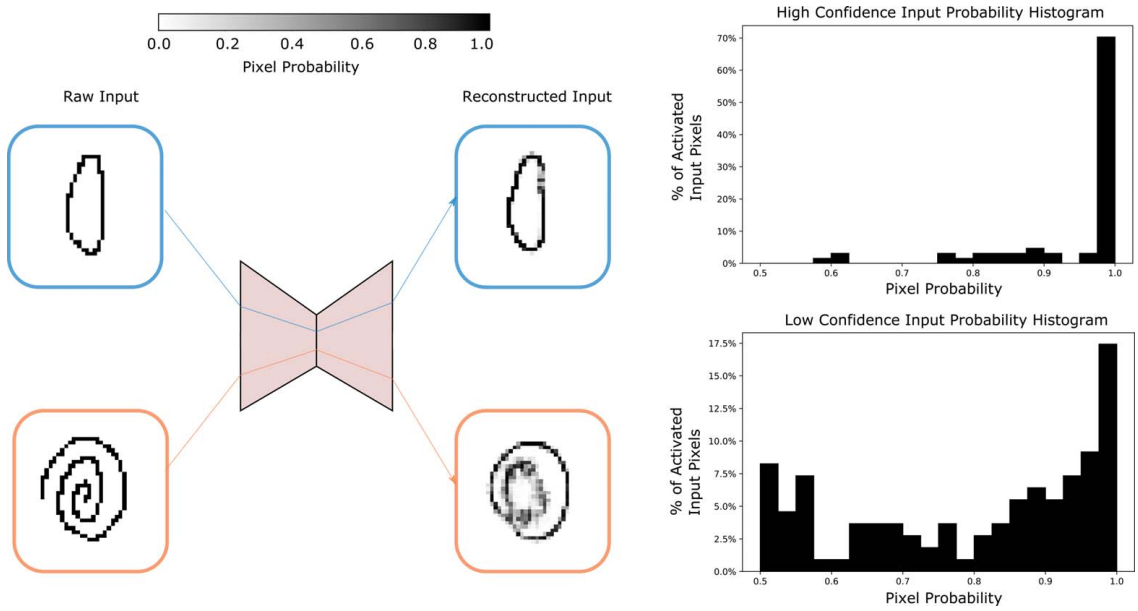

Fig. 10 The raw input on the top is close to the true distribution of coupler curves, thus VAE retains almost 70% of the highest intensities as depicted by the histogram on the top. For the second input, VAE overrides user assignments on the input pixels by a large number. This is due to the highly spiral nature of the input. It can be also seen in Fig. 9 that the inner portion of the input is highly penalized.

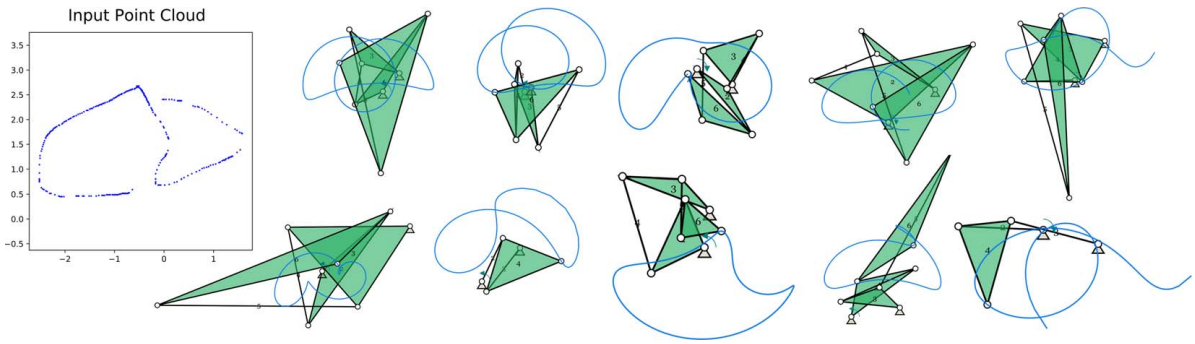

Fig. 11 The linkages corresponding to the K nearest neighbor in latent space for the given raw input are depicted. It can be seen that the linkages exhibit a wide variety in terms of their coupler path; thereby capturing input uncertainty.

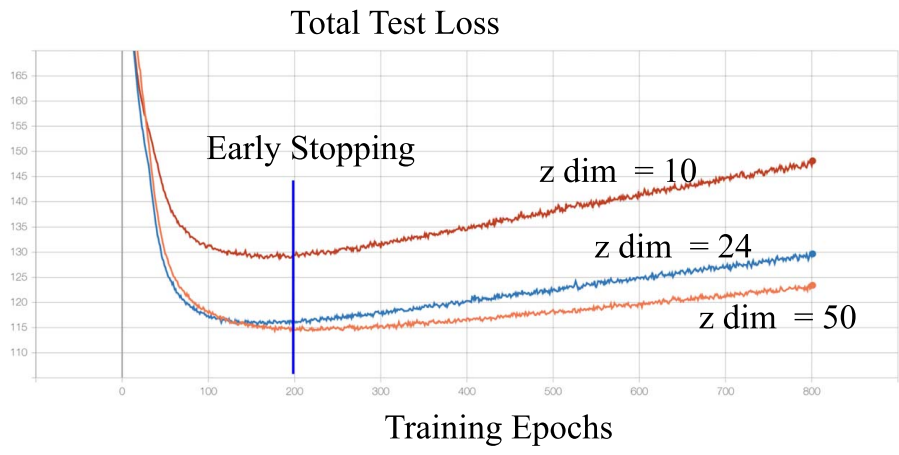

Fig. 12 Test loss for three different VAEs with latent dimensions 50, 24, and 10, respectively. The figure depicts the training for 800 epochs to demonstrate that after 200 epochs, the model starts overfitting to the training data. Thus, the training is stopped after 200 epochs to prevent overfitting. This technique is called Early Stopping. It can be seen that VAE with higher dimensional latent space achieves higher Variational Lower Bound (L_{ELBo}).

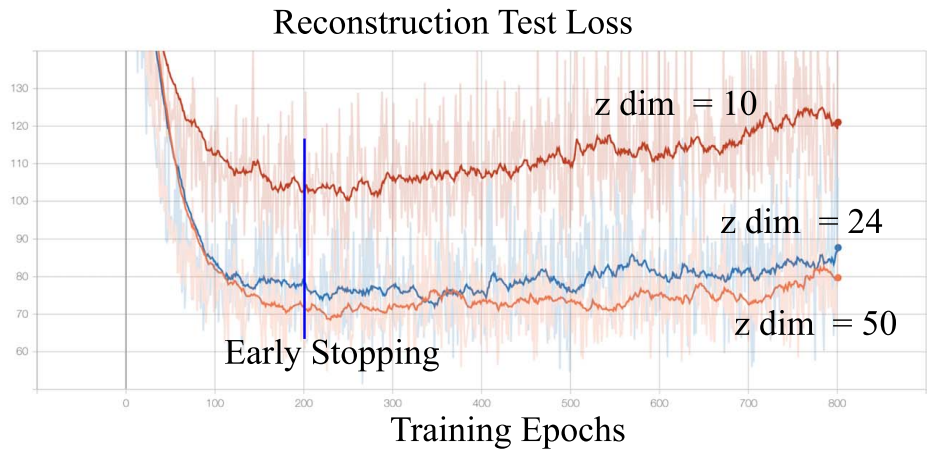

Fig. 13 Reconstruction losses for the three VAEs with latent dimensions 50, 24, and 10. It can be seen that higher dimensionality of latent space achieves better reconstruction loss.

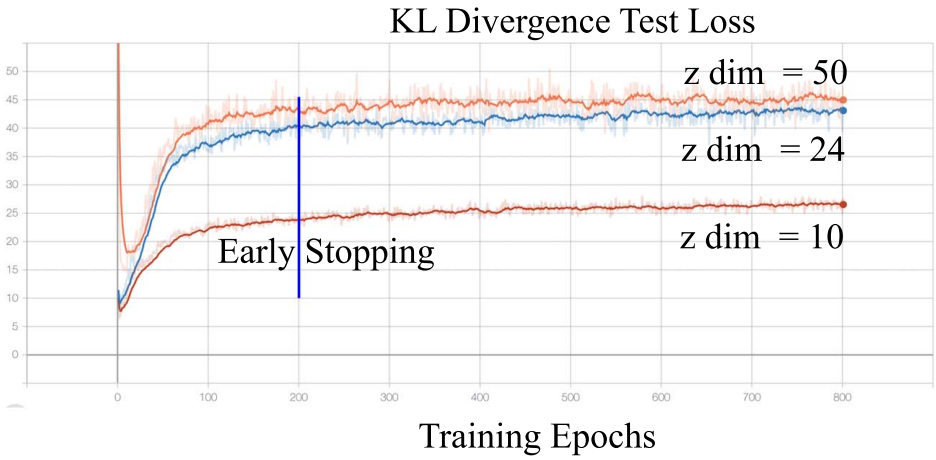

Fig. 14 Reconstruction losses for the three VAEs with latent dimensions 50, 24, and 10. It can be seen that higher dimensionality incurs a higher KL-divergence loss.

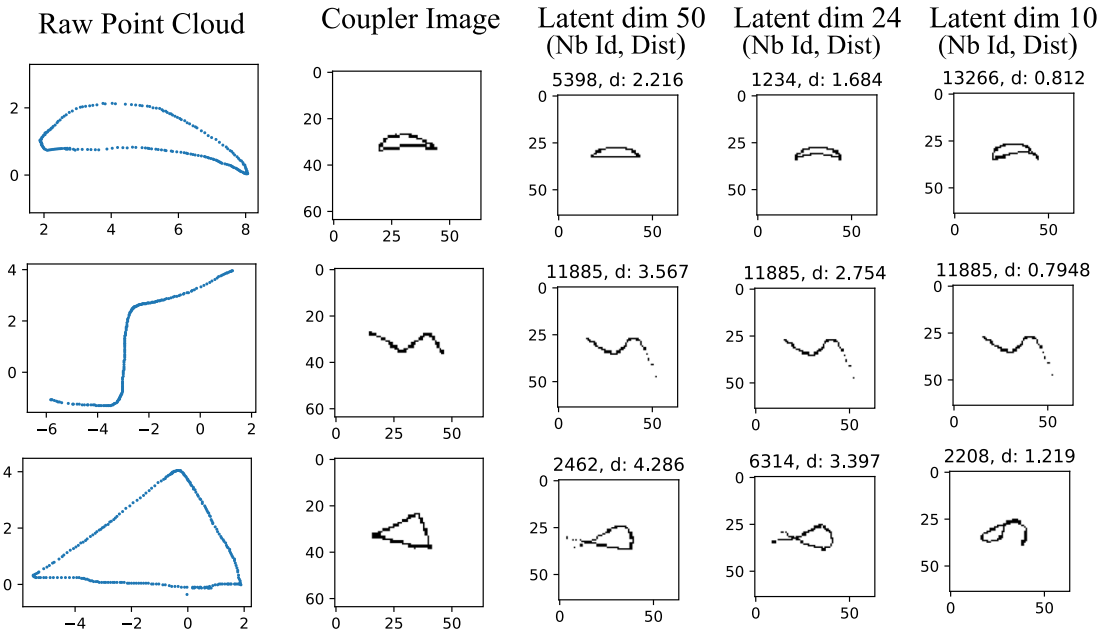

Fig. 15 Comparison between the latent spaces when an image is queried for the nearest neighbor. The performance degrades with lower dimensionality. This can be attributed to the denser latent embeddings, which forces dissimilar images to move closer. In the figure, Nb Id indicates the unique identifier for each data point from the dataset, and Dist represents the Euclidean distance between the query point and the neighbor.

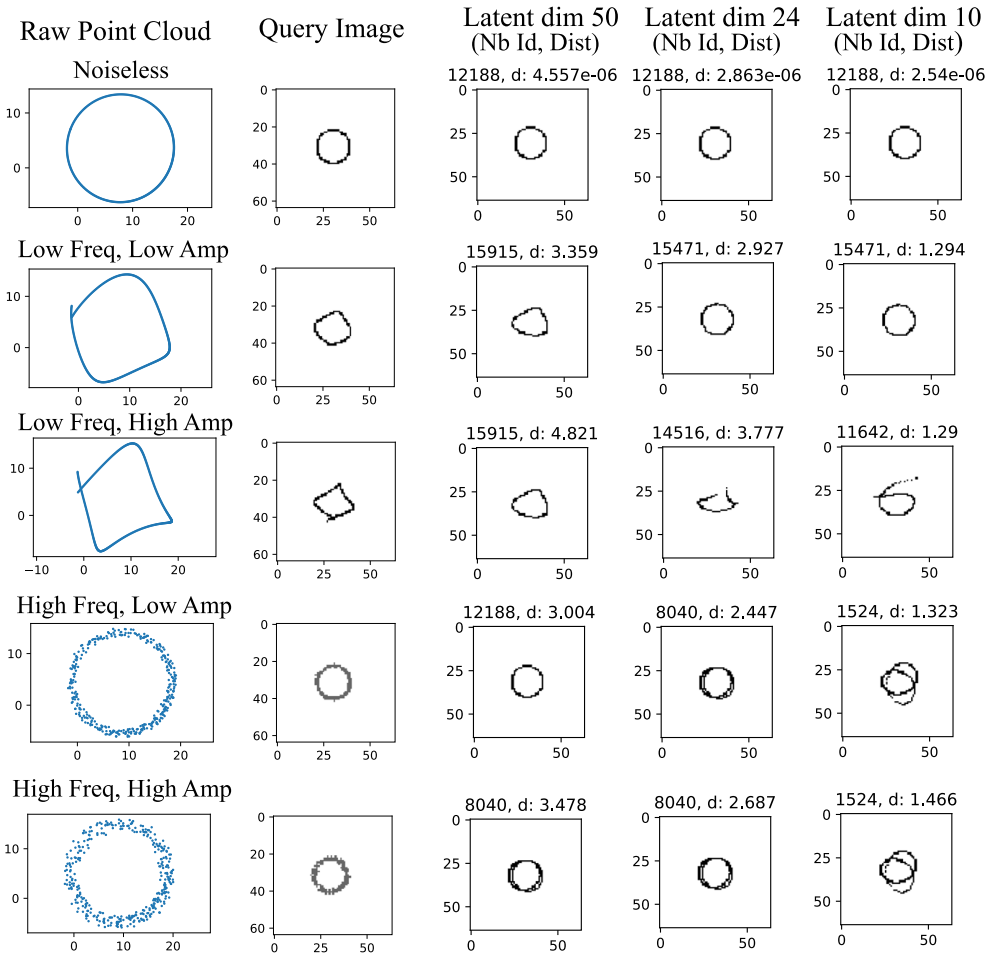

Fig. 16 Comparison between the latent spaces when a known coupler curve is corrupted with varying degrees of noise and queried for the nearest neighbor. Lower dimensional latent space is relatively less robust to the noise. We believe it happens because a small error in recognition can lead to larger variations as the latent space becomes more compact.

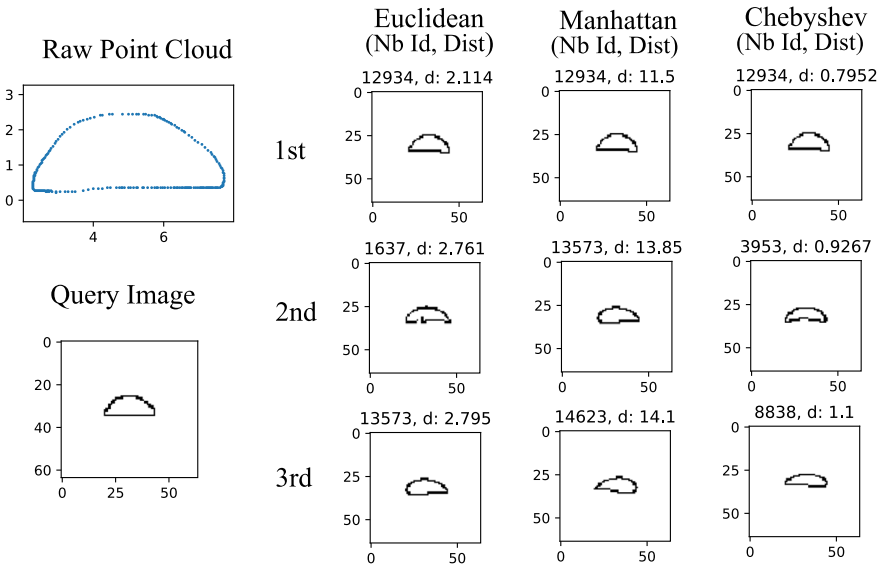

Fig. 17 The effect of different metrics to find three-nearest neighbors in 50-dimensional latent spaces. All of the three metrics yield the same closest nearest neighbor (Nb Id 12934). However, the next two neighbors are different for each metric. Incidentally, Manhattan distance is defined as the distance between two points in the latent space measured along axes at right angles, while the Chebyshev distance between two points is defined as the largest of the difference along any coordinate dimension.

is used. The choice for an appropriate metric remains an open question that demands further investigation. Overall, 50-dimensional latent space performs better for recognition tasks.

Conclusion

The paper presents a novel image-based representation schema for probabilistic modeling of coupler curves. It is combined with deep convolutional VAE to capture spatial correlations. VAE is used to generate a variety of conducive inputs that efficiently represent the inherent uncertainty of the raw user input. In addition, these generated inputs provide feedback on the user input by manipulating the probability distribution of input pixels. The purpose is to obtain a diverse set of linkages with similar coupler curves instead of finding a single optimal solution for an inherently uncertain input. The approach is general enough to be extended to spatial linkages for which there are even fewer synthesis methods available.

Acknowledgment

This work has been financially supported by The National Science Foundation under a research grant # CMMI-1563413 to Stony Brook University. All findings and results presented in this paper are those of the authors and do not represent those of the funding agencies.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request. The authors attest that all data for this study are included in the paper.

References

[1] Sandor, G. N., and Erdman, A. G.1997, *Advanced Mechanism Design: Analysis and Synthesis*, Vol. 2, Prentice-Hall, Englewood Cliffs, NJ.
[2] McCarthy, J. M., and Soh, G. S., 2010, *Geometric Design of Linkages*, Vol. 11, Springer, New York.
[3] Suh, C. H., and Radcliffe, C. W., 1978, *Kinematics and Mechanism Design*, John Wiley and Sons, New York.
[4] Hartenberg, R. S., and Denavit, J., 1964, *Kinematic Synthesis of Linkages*, McGraw-Hill, New York.
[5] Lohse, P., 2013, *Getriebesynthese: Bewegungsablufe Ebener Koppelmechanismen*, Springer-Verlag, Berlin.
[6] Hunt, K., 1978, *Kinematic Geometry of Mechanisms*, Clarendon Press, Oxford.
[7] Deshpande, S., and Purwar, A., 2017, "A Task Driven Approach to Optimal Synthesis of Planar Four-bar Linkages for Extended Burmester Problem," ASME. J. Mech. Rob., **9**(6), p. 061005.
[8] Ge, Q. J., Purwar, A., Zhao, P., and Deshpande, S., 2016, "A Task Driven Approach to Unified Synthesis of Planar Four-Bar Linkages Using Algebraic Fitting of a Pencil of G-Manifolds," ASME J. Comput. Inform. Sci. Eng., **17**(3), p. 031011.
[9] Purwar, A., Deshpande, S., and Ge, Q. J., 2017, "MotionGen: Interactive Design and Editing of Planar Four-Bar Motions Via a Unified Framework for Generating Pose- and Geometric-Constraints," ASME. J. Mech. Rob., **9**(2), p. 024504.
[10] 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.
[11] Deshpande, S., and Purwar, A., 2019, "A Machine Learning Approach to Kinematic Synthesis of Defect-Free Planar Four-Bar Linkages," ASME J. Comput. Infor. Sci. Engin., **19**(2), p. 021004.
[12] Nolle, H., and Hunt, K. H., 1971, "Optimum Synthesis of Planar Linkages to Generate Coupler Curves," J. Mech., **6**(3), p. 267.
[13] Ullah, I., and Kota, S., 1997, "Optimal Synthesis of Mechanisms for Path Generation Using Fourier Descriptors and Global Search Methods," ASME J. Mech. Des., **119**(4), pp. 504–510.
[14] Wu, J., Ge, Q. J., Gao, F., and Guo, W. Z., 2011, "On the Extension of a Fourier Descriptor Based Method for Planar Four-Bar Linkage Synthesis for Generation of Open and Closed Paths," ASME J. Mech. Robot., **3**(3), p. 031002.
[15] Buskiewicz, J., Starosta, R., and Walczak, T., 2009, "On the Application of the Curve Curvature in Path Synthesis," Mech. Mach. Theory., **44**(6), pp. 1223–1239.
[16] 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.
[17] Forsyth, D. A., and Ponce, J., 2002, *Computer Vision: a Modern Approach*, Prentice Hall Professional Technical Reference, New York City.
[18] Krizhevsky, A., Sutskever, I., and Hinton, G. E., 2012, "Imagenet Classification With Deep Convolutional Neural Networks," Adv. Neural inform. Process. Syst., pp. 1097–1105.
[19] Bottou, L., 2010, "Large-Scale Machine Learning With Stochastic Gradient Descent," Proceedings of COMPSTAT'2010, Paris.
[20] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., 1986, "Learning Representations by Back-Propagating Errors," Nature, **323**(6088), pp. 533–536.
[21] Kingma, D. P., and Welling, M., 2014, "Auto-Encoding Variational Bayes," Comput. Res. Repository, **abs/1312.6114**.
[22] Blei, D. M., Kucukelbir, A., and McAuliffe, J. D., 2017, "Variational Inference: A Review for Statisticians," J. Am. Stat. Assoc., **112**(518), pp. 859–877.
[23] Kullback, S., and Leibler, R. A., 1951, "On Information and Sufficiency," Ann. Math. Stat., **22**(1), pp. 79–86.