ASME Journal of Computing and Information Science in Engineering
Online journal at:
https://asmedigitalcollection.asme.org/computingengineering

Deep Learning-Driven Design of Robot Mechanisms

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

Nilanjan Chakraborty
Interacting Robotic Systems Lab,
Department of Mechanical Engineering,
Stony Brook University,
Stony Brook, NY 11794

In this paper, we discuss the convergence of recent advances in deep neural networks (DNNs) with the design of robotic mechanisms, which entails the conceptualization of the design problem as a learning problem from the space of design specifications to a parameterization of the space of mechanisms. We identify three key inter-related problems that are at the forefront of using the versatility of DNNs in solving mechanism design problems. The first problem is that of representation of mechanisms and their design specifications, where the representation challenges arise primarily from the non-Euclidean nature of the data. The second problem is that of developing a mapping from the space of design specifications to the mechanisms where, ideally, we would like to synthesize both type and dimensions of the mechanism for a wide variety of design specifications including path synthesis, motion synthesis, constraints on pivot locations, etc. The third problem is that of designing the neural network architecture for end-to-end training and generation of multiple candidate mechanisms for a given design specification. We also present a brief overview of the state-of-the-art on each of these problems and identify questions of potential interest to the research community. [DOI: 10.1115/1.4062542]

Keywords: deep learning, kinematics, mechanisms, robotics, machine learning, neural networks

1 Introduction

Neural networks (NNs) are well known as universal function approximators for a class of functions defined with the Euclidean space as their domain [1,2]. Classically, for approximating functions whose domain is not Euclidean [3], handcrafted feature vectors were used to map the data to a Euclidean space and then NNs could be used to learn or approximate the function. Consequently, the performance of NNs were limited by the human designers' ability to capture or map the relevant features. The feature vectors are an Euclidean embedding of the data so that the relevant aspects of the data required for learning are captured. The modern revolution in deep learning (DL) came from the insight that the feature vectors should be learned from data as well, and different neural network architectures were designed for different data types that enabled this feature extraction or embedding of the data in a Euclidean space [4–7]. Examples include convolution neural networks (CNNs) [8] for image data, graph neural networks (GNNs) [9] for data obtained from systems most naturally modeled as a graph, and transformer networks [10] for sequential data, which have been used for learning large language models. Use of deep neural networks (DNNs) in engineering design is relatively new; a recent review on this topic is provided in Ref. [11] and in a special issue on Machine Learning in Engineering Design by the *ASME Journal of Mechanical Design* [12].

NNs have proven to be an interesting alternative in engineering design problems characterized by difficulty in formulating analytical representations, imprecise input, and large solution space. DNNs' advantages over classical NNs have been that they allow one to automate the process of feature generation and provide a larger representation and mapping capacity. Let us consider an example of generating a four-bar mechanism for path synthesis problems. For designing a neural network to solve this problem, there are various ways of representing the input path that has been explored in the past, e.g., Fourier descriptors of the curve and wavelet descriptors of the curve. These are called different feature vectors for the path and even changing the number of Fourier descriptors or wavelet descriptors used changes the feature vectors. The feature vector choice is made by the human designer a priori and there is no clarity on which feature vector is preferable, if any. The choice of a feature vector is usually a trial-and-error process, and there is seldom any universally good choice (at least, such a choice has not been discovered yet). The use of DNNs allows the learning of the appropriate feature vector. The user can input the path as a list of points or an image and using a training dataset, the neural network will learn the features as well as the mapping from the features to the output space of mechanism parameterization. However, the cost to be paid for such flexibility is more training data and computational time. The paradigm of learning features instead of hand-coding them (which was also prevalent in computer vision and natural language processing (NLP) literature) has worked well across different application areas including computer vision and NLP. So the hope is, and preliminary results do support this, that the use of DNNs will make the learned models produce better quality mechanisms that follow the given input paths as closely as possible.

The design of robotic mechanisms from a given set of design specifications can also be formulated as a learning problem from the space of specifications to a parameterization of the space of mechanisms. Classically, design of mechanisms has been a

[1]Corresponding author.

Manuscript received February 1, 2023; final manuscript received April 28, 2023; published online June 5, 2023. Assoc. Editor: Satyandra K. Gupta.

two-step process of type and dimensional synthesis with scant attention paid to the type synthesis. As a result, the synthesis problems have been reduced to primarily computing dimensional parameters after selecting a type [13–15]. This approach has shown to be ineffective in creating practical mechanism design solutions from the given task [16]. In the context of machine learning (ML), the space of design specifications, which can be quite varied and application specific, is non-Euclidean in general. Examples of design specifications can be a desired planar curve (planar path synthesis problem), a subset of the group of rigid body configurations or $SE(3)$[2] (mechanism synthesis for a given workspace, which can be useful in exoskeleton robotic device synthesis), etc. [17]. Furthermore, there can be other geometric and kinematics constraints and objectives like the location of some of the pivots [18], moving envelope of mechanisms, relative link length ratios, defect-free generation [19], etc. The current DL literature does not deal with this kind of data that arise in mechanism design, and consequently there is no known way of neural network architecture design to embed the design specifications into a Euclidean space.

Within the mechanism design literature, CNNs have been used to embed design specifications given as planar curves into a Euclidean space, by converting the curve to a 2D image [20]. In the past, Fourier descriptors [21–26] and wavelets [27–31] have also been used to represent coupler curves. However, creating neural network architectures that can embed design specifications into a Euclidean space or learn features of the design task is a largely unexplored research challenge. More recently, it has been shown that geometric deep learning models could capture non-Euclidean domains more efficiently and effectively such as in graphs, trees, manifolds, and networks [32,33]. The joint representation of the design space along with the representation of known mechanisms that produces a design can lead to novel encoder–decoder architectures or deep generative networks that can facilitate novel mechanism synthesis [34]. This is analogous to the use of CNNs with variational autoencoder (VAE) [35,36] or generative adversarial network (GAN) [37] for DL-based image synthesis from known images of object categories. Moreover, it is well known that for simple one-degree-of-freedom linkage mechanisms, such as planar four-bar, Fourier descriptors of the path and the orientation are coupled, which can be exploited to synthesize mechanisms for motion generation problems [38,39]. An open question is whether such relationships can be captured by a NN by only providing a large amount of good synthetic simulation data? If so, this could lead to a breakthrough in developing artificial intelligence (AI) systems, which can predict critical kinematic properties of robot mechanisms. Enforcing a structure on the latent layers in VAEs and GANs could enable design synthesis under multiple constraints while helping such AI systems to be explainable and interpretable. In summary, there are great opportunities for researchers to (1) embed existing knowledge to create effective representations of mechanisms (type and dimension) and (2) use domain-specific algorithms in conjunction with ML models to process synthetically produced motion data from simulations or from physical motion capture. The outcomes could be component-based reusable unified NN models with sufficient granularity so that they can be applied in diverse synthesis applications and truly integrate machine learning with robotic mechanism design.

The rest of this paper discusses specific research questions and challenges, which could lead to new areas of investigation in deep learning of robot mechanisms. The foremost of these is the issue of representation of mechanisms and their properties. While the mechanisms themselves are defined by their structure, their properties are defined by the motion generated by a floating link of mechanisms. The space of design specifications primarily begins by defining desirable properties, but can also include structural parameters. We look at the representation aspects in Sec. 2. Next question is of the mapping from the design specifications to mechanisms. The goal is to generate a large number of conceptual design solutions, which satisfy design specifications while allowing users an intuitive way to explore the design space. The mapping should ideally generate both type and dimensions of single- and multi-degree-of-freedom mechanisms. This naturally leads to the questions of (1) representation of mechanism parameters, which encode both the type and dimensions, and (2) generation of a dataset for training DNNs with good quality of synthetic motion data for a variety of mechanisms. We discuss these issues in Sec. 3. Related to the question of mapping from design specifications to the mechanisms is the design of DNN architectures which would enable the mapping. While the recent research in DNNs has proposed a variety of generative frameworks, such as VAE, GAN, and their variants, it is unclear as to how they should be leveraged in mechanism synthesis! In Sec. 4, we discuss a few recent work using VAEs and possible directions for future efforts in designing DNNs for mechanism synthesis.

2 Representation of Mechanisms and Their Properties

Linkage mechanisms are a collection of rigid bodies connected via joints; for example, the most ubiquitous of them is a planar four-bar mechanism consisting of four links connected by prismatic or revolute joints. Their topology is defined by the pattern of interconnection of their links and joints, while the length of the links and the location of joints define the dimensional parameters of the mechanism. Kinematic synthesis of such mechanisms is typically concerned with finding their topology and dimensions given design specifications, such as path of a point of interest on a floating link or the complete motion of the floating link as well as additional constraints. There have been several efforts in recent years to design NNs, which learn the mapping from the space of design specifications to the mechanism properties (type and dimensions). However, these efforts have largely been limited to a path or motion as the input and dimensional parameters as the output. Even then, several fundamental questions have remained unanswered, such as the choice of best representation for the path or the motion specifications for a given problem. Typically, one of these representations would serve as the input vectors for the DNNs. Fourier descriptors, wavelets, Cartesian coordinates, and curvature integral have remained the dominant choice for representing the path of a point; however, the choices made have been arbitrary without adequate justification. More recently, it was proposed that to leverage the learning capabilities of CNN, a path could be embedded in an image of fixed size, which would be independent of the algebraic complexity of the curve [20]. This area of research requires a careful investigation of different representations of the coupler curves and their comparative advantages and disadvantages. A key question related to their representation is if there could be a representation, which acts as a normalizer for the NN-based mechanism synthesis? In other words, could we use a unifying representation, which is invariant of these base representations and is insensitive to the generated mechanism solutions. This would be similar to the idea of a deep CNN extracting features from an input automatically.

While most work has focused on the path synthesis of planar mechanisms, their extension to the motion generation problem is even harder. That begets how the path and orientation together should be represented? Since they are unit incompatible, how to combine them in a single input vector for a NN is unclear! While it is well known that the path and orientation of the coupler of single-degree-of-freedom mechanisms cannot be decoupled, a straight-forward extension of the current approach would be to find Fourier or wavelet descriptors, Cartesian and curvature integral of path and orientation data independently and flatten the input vector to include both of them.

A third related question is how these representations should be extended for spherical and spatial mechanism motions, where the

[2] $SE(3)$ is the special Euclidean group of dimension 3, which is the group of all rigid body positions and orientations.

paths are in R^3 and motions in $SE(3)$. This debate is not limited just to the path and motion properties of mechanisms; it gets even more difficult when one needs to incorporate other geometric and kinematic constraints, such as location of the pivots, transmission angle, range of motion, or dimensional constraints. These design specifications form a non-Euclidean space for a NN and there is little literature on this.

It is heartening that alternative representations for the properties of a mechanism, such as path and motion of a mechanism, have been proposed. On the other hand, representation of the structure of mechanisms for NNs has been largely relegated to using an output vector of few joint positions. This may be fine for synthesizing dimensional parameters of a mechanism if the type of the mechanism chosen is accurate; however, it has been shown that a pre-selection of the type of mechanism often leads to no or poor solutions. Thus, there is a need to encode the type and dimensional information of a mechanism together for its representation in NNs. Kinematicians are familiar with the seminal work of Tsai [40], who modeled mechanisms as graphs and represented them using an adjacency- or incidence-matrix. While such representations may be useful for enumeration and evaluation of mechanisms, they form a non-Euclidean space of mechanism parameters, which makes it difficult to directly employ and interpret them in DNNs. What may be needed is a structured coupling of the type and dimensional parameters, which can be learnt by DNNs. We discuss such a possible representation in the next section in the context of motion synthesis using a DNN.

3 Mapping From the Design Specifications to Mechanisms

The ultimate goal is to learn a mapping from a diverse and often conflicting set of design specifications (including path or motion properties and additional constraints) to mechanism structure (type and dimensions). This mapping should yield a large set of conceptual designs satisfying specifications. Existing literature on using neural networks for mechanism synthesis is largely focused on the path generation problem, wherein a path is provided as an input to a NN using one of the aforementioned representations and the NN is trained to learn the mapping from the path information to the dimensional parameters of a mechanism. While these early efforts have produced desirable outcomes, the problem of synthesis is defined too narrowly and the results are far and few. Generally, these approaches yield one or a few mechanisms based on a specific task requirement, which is often a computational convenience rather than true representation of users' requirements. For example, a given path or motion is converted into a precision point or pose problem with discrete data points, which can be embedded in a Euclidean space and a direct mapping is achieved using a NN. Moreover, typically the user will have to make an artificial choice about the type of mechanism and the goal will be limited to finding dimensional parameters. If the goal is to find a large set of feasible mechanism building blocks without assuming their types while accounting for amorphous, incomplete, and imprecise input, then the traditional or aforementioned NN-based synthesis approaches fall short.

It has been shown recently that the classical two-step paradigm of mechanism design, wherein the topology or type of a mechanism is first picked and then its dimensions are computed, is a flawed approach, which often yields no or highly suboptimal solution. Fruits of such endeavor will be sweet only if a true data-driven approach is incorporated in synthesis—one which includes simultaneous type and dimensional synthesis. However, the early work in this direction have focused on creating unified design equations for a class of mechanisms and then solving a low-degree polynomial to generate a limited number of solutions, which mostly produce defective mechanisms. There have also been several attempts to incorporate additional design specifications, such as location of pivots, but their extension to the more complex constraints or even higher-order mechanisms using a pure analytical approach has proven to be much more difficult.

However, a great advantage of unified design equations is that the choice of joint types for dyad synthesis is totally task driven, i.e., it does not require user specification for the joint types. For example, a dyad of a planar four-bar mechanism can be represented by a linear equation with nine parameters (irrespective of the type of joints).

$$A_0P_0 + A_1P_1 + A_2P_2 + A_3P_3 + \cdots + A_9P_9 = 0 \tag{1}$$

where the parameters A_i ($i = 0, 1, 2, \ldots, 9$) are defined by given input poses and the parameters P_i ($i = 0, 1, 2, \ldots, 9$) are intermediate design parameters defined in terms of mechanism parameters that encodes the type of joints as well as the position of the joints [16–18,41]. Regardless of the type of dyads, the parameters P_i must satisfy the following two constraints:

$$P_1P_6 + P_2P_5 - P_3P_4 = 0, \quad 2P_1P_7 - P_2P_4 - P_3P_5 = 0$$

Hence, each dyad can be represented by a parameter vector P, which is a possible redundant representation, where the actual linkage parameters can be obtained from these parameters (in spite of the redundancy). The pattern of zeros in the parameter vector gives the type of the dyad. In this way, a four-bar linkage can be represented by a pair of P vectors. Thus, using P as a representation for the mechanism, we have a unified representation of the mechanism, i.e., the output layer of a NN, which are not the naive design output variables like the type of joints or position of joints or even a matrix combining the two. However, a set of P vectors encodes all the information about the mechanism and the design output variables can be easily computed from this representation. This is a promising direction for exploration, which could bring together recent advances in simultaneous type and dimensional synthesis with the NNs and computational shape analysis to effectively solve the motion generation problem. This approach could be extended to a class of mechanisms for which unified design equations can be created; however, it is a non-trivial exercise in itself.

4 Design of Deep Neural Network Architecture

Another set of key challenges that arise in forming the end-to-end system is the decision on architecture of the deep neural network, the design of the training algorithm, and obtaining the data to train the network for a particular task. The existing NN architectures for mechanism synthesis have been simple mapping networks from a path description to mechanism parameters. These have been limited to mostly planar four-bar mechanisms.

In the recent years, DL-based generative frameworks based on VAEs, GANs, and deep reinforcement learning have proven to be quite effective in generating a variety of synthetic data, such as handwritten digits, images, and videos. They have also been used in other fields notably autonomous driving, molecule design, anomaly detection, text classification, recommendation system, and many more. While autoencoders, a simpler variation of a VAE, can only compress the input data into a latent space, a VAE can also generate new data by sampling in the latent space. These generative models can be used for learning joint probability distribution of various mechanism parameters and their interdependence. With that, we can ask the following probabilistic inference questions: (1) what are the infeasible aspects of the input and how to modify them to make a more conducive input, (2) given a path or a motion task, how likely it is that a particular type of mechanism can perform the task, and (3) for a given task, what is the distribution of mechanism parameters with similar coupler motions (or, paths)?

A VAE-based approach could accept deliberately imprecise or inherently uncertain input from users and compute a distribution of conducive inputs and feed them to a synthesis solver to obtain a large variety of acceptable solution concepts. Having learnt the probability distribution of the coupler curves of mechanisms, the

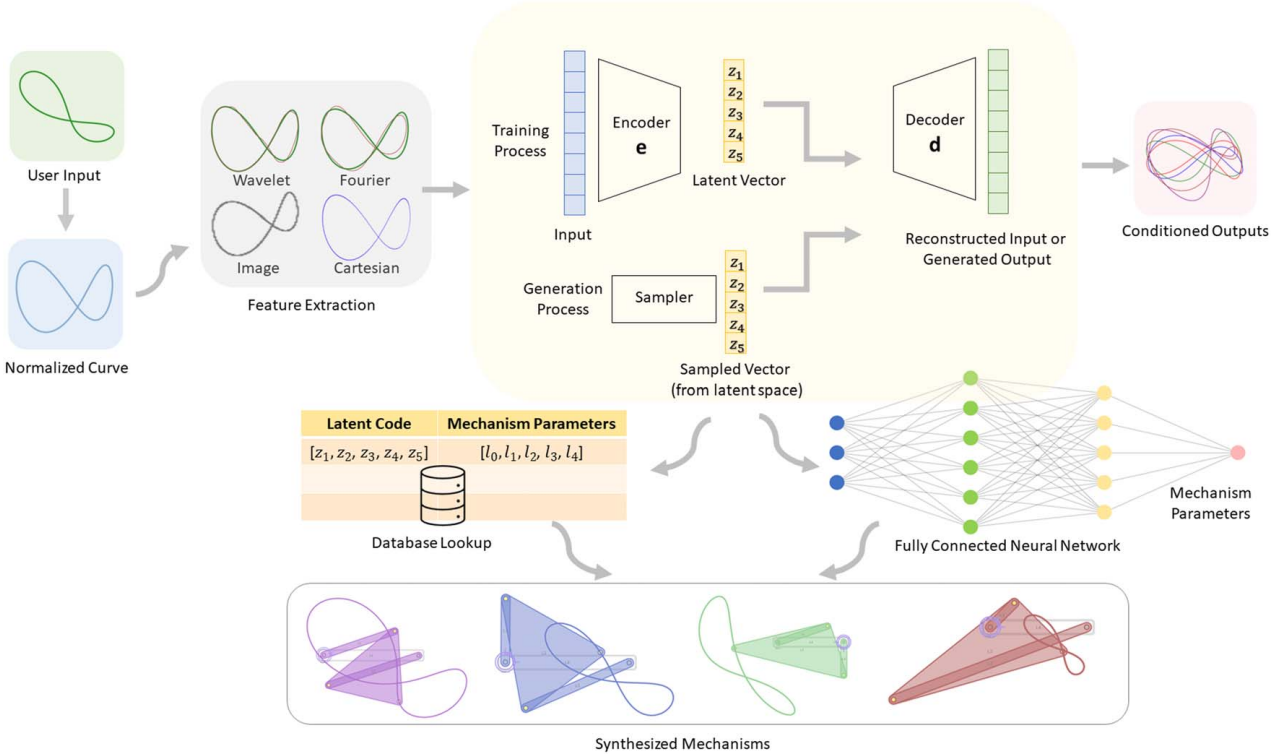

Fig. 1 The raw user input is represented using one of the common Euclidean-based representations, such as Fourier and wavelet descriptors, or Cartesian coordinates of the input path, or an image of the input, which is passed through an encoder network e. The encoder captures the salient information in the form of multivariate distribution of latent features. Random samples from this distribution are fed to the decoder e to generate paths with a high likelihood of producing good solutions. These random samples can be queried from a database of mechanisms or fed to a fully connected NN to get multiple candidate mechanisms.

encoder part of the VAE predicts a 2D Gaussian distribution of latent features for each of the inputs. By drawing random samples in the vicinity of the computed latent code, the generator network of the VAE outputs the samples with closed paths that resemble the original input path. While a traditional VAE would simply generate outputs similar to the input, our synthesis goal requires that a VAE be augmented with another mapping to generate mechanism design parameters. Two potential approaches for this are to (1) map latent codes to mechanism parameters by a kNN search and a database query or (2) train another fully connected DNN to map from the latent code to mechanism parameters. While in the former approach, generated mechanisms could be tightly controlled, such as Grashof-only or of a certain complexity, in the latter approach, the capacity to represent a mechanism is completely built-in a NN, which could lead to more efficient and generalized query. Figure 1 shows an overview of this approach where a sequence of discrete points is the input to a trained VAE and the output are various mechanisms.

MotionGen Pro [17,42], a web-based app available,[3] has implemented a VAE for path generation with a database and produces up to 30 four-bar and six-bar mechanisms for a user-sketched path. The training data were generated from a unified simulation algorithm presented in Ref. [42]. Figure 2 shows a few mechanism concepts produced by the MotionGen for a sit-to-stand motion path of hip joint.

While these VAE-based approaches have resulted in successfully generating a number of mechanism design solutions of specific type, they have still been restricted to approximating users' task into a set of points embedded in a Euclidean space for the path generation problem. Several other questions as well remain to be answered: (1) how can this be scaled to other mechanisms without having to train separate NNs for each type of mechanisms, (2) how can motion generation problem be solved using such an approach, (3) how can additional constraints be handled, and (4) how can users visualize and filter out a large number of solutions? There are several variants of VAEs available, such as conditional-VAE (C-VAE), which enable conditioning input vectors on some random variables. A C-VAE could be used to condition the given task on, say, location of joints, which could enable users to directly synthesize mechanisms with specific properties.

The basic premise of the deep learning-driven robot mechanisms is that we want to create a probabilistic distribution of output motions of a variety of mechanisms including planar, spherical, and spatial ones. We are fortunate that we can algorithmically assemble different types of mechanisms and perform forward kinematics to get the output motions. So far, the NN literature has largely investigated a mapping between simple planar four-bar mechanisms and their paths. It is of course, but a prudent choice, to begin with something simple like planar four-bar mechanisms for which several simulation algorithms exist, which can create a large dataset of mechanisms. Their extension to the spherical and spatial multi-chained mechanisms, such as those found in exoskeleton devices for the different human applications, is hardly a straight-forward extension; see, for example, exoskeleton mechanisms discussed in Refs. [43–45]. This requires that new computationally efficient algorithms be developed, which can provide high-quality synthetic data associating mechanism designs with their output motions. Good quality data used in training a VAE or GAN or one of its variants will be amenable to exploration in latent space. Once we have learned this distribution, then the next question is of querying. While the user provides a raw input, a kNN or related search in the latent space of a generative network could find a plethora of design solutions, which can be further

[3]http://www.motiongen.io

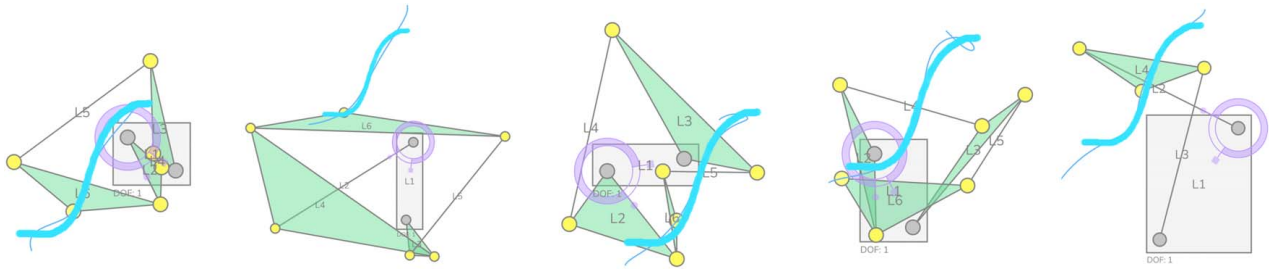

Fig. 2 User input (path) is sketched using a mouse and the MotionGen produces various four-bar and six-bar mechanisms with a fully rotatable link by sampling in the latent space of a trained VAE

subjected to constraints on type, dimension, as well as range of motion. This is the basic idea, but a fundamental question is how to systematically explore the latent space, which requires that we have some understanding of the manifold structure of the latent space [46]. With that, it may be possible to impose the constraints ab initio in the design process, which has not been done before.

5 New Approaches

Apart from the ML approaches discussed so far in representation of the specification space, mapping from set of specifications to mechanisms, and DNN models, recently several other generative models have taken the world by storm. Diffusion models are a class of generative models and have primarily found applications in image synthesis [47,48]. The key idea is to iteratively add Gaussian noise to an input image to convert it into noise and also simultaneously learn the inverse conditional distribution function from the noise to an image iteratively. Within the image synthesis literature, diffusion models have been known to produce more photo-realistic images than GANs. Although the use of diffusion models has not been explored thus far in mechanism design, they can be potentially used in mechanism design problems where the use of GANs have been explored.

Transformer networks have been proposed originally for various learning tasks on sequential data, a hallmark of language processing applications [49,50]. However, more recently they have also found applications in image processing tasks and tasks involving multi-modal input data (e.g., text and image). There are two key strengths of transformer networks. First, in principle, transformer networks allow one to encode global structure in the input, if present, which is usually referred to as long range dependencies or context in the NLP tasks. Second, they allow a common encoding (or embedding) of multimodal input data, which allows one to answer queries or give output for inputs in any modality. From the perspective of mechanism design, this raises interesting possibilities for embedding design constraints. For example, it may provide a way for stating qualitative constraints like "make the coupler shorter than the fixed link." Furthermore, it can also provide ways to give constraints on locations of a fixed pivot by marking an area on an image. To develop such capabilities, the training dataset also has to be generated appropriately, based on the types of constraints and modes of specification that may be of interest to designers. The generation of such training sets itself provides an interesting opportunity and offers various possibilities of how designers can interact with a design software.

Physics-informed machine learning (PIML) approaches try to generate solutions for complex multi-scale systems by combining observational data with the mathematical models underpinning the physics of the system [51,52]. This field of research has gained momentum in many application areas including prediction of complex systems like atmospheric air-flow, design of functional materials, etc. PIML approaches can potentially lead to formulation and study of new problems in mechanism design that to the best of our knowledge are currently not well studied. For example, for given design tasks, there is very little in terms of principled approaches for synthesizing multi-link mechanisms with both kinematics and dynamics constraints as well as rigid and flexible components. In such scenarios, the physics could be modeled by (partial) differential equations governing the dynamics of motion along with algebraic constraints modeling the kinematic constraints. It is within the realm of possibility to formalize such problems as learning problems. Apart from architecture design and modeling challenges, a key challenge that arises here is in generating training data sets, which has to be curated carefully and possibly with a community-driven effort, with data sets coming from multiple researchers with their own application interest and expertise.

6 Conclusions

The formulation of the robotic mechanism design problem as a learning problem has immense potential of introducing deep learning techniques as a new tool to the design engineers toolbox. DNN-based approaches have the potential of solving problems in motion synthesis and path synthesis along with other scenario-specific constraints on the mechanism in a unified framework. However, there are key technical challenges that need to be overcome for DNN-based synthesis techniques to realize their potential. The key problems that we identified are that of representation of the design specifications and the mechanisms, development of the mapping from the design specification space to the mechanism parameterization that allows both type and dimensional synthesis, and design of the DNN architecture to learn the aforementioned mapping. We summarize key questions and approaches below:

(1) The first area of investigation pertains to the representation of mechanisms, including their type, dimensions, and properties, as well as input specifications related to path, motion, pivots, and link length, among others. Specifically, there are several questions that need to be addressed:
   (a) How can one embed these properties and specifications in a Euclidean space, and can DNNs assist in the creation of invariant representations that do not require feature engineering?
   (b) What is the best way to develop a unified representation that combines inputs that are incompatible, such as path and orientation data?
   (c) How can these representations be extended from 2D to 3D spatial data for spherical and spatial mechanisms?
(2) The second area of investigation focuses on NN-based mapping from the design space to mechanisms. This involves addressing the following questions:
   (a) How can a unified representation of the type and dimensional parameters of mechanisms be developed that can be learned by NNs? Can traditional representations, such as adjacency matrices, and newer techniques, such as GNNs, be useful?
   (b) What is the best way to leverage generative models to create a diverse range of mechanism design concepts?

(c) How can conflicting design constraints be accommodated in synthesis and can multimodal models be useful?

(3) The third area of investigation pertains to DNN architecture and training data. Here are some of the questions that need to be answered:
   (a) What is the best way to generate high-quality training data from simulations?
   (b) Which DNN architecture is most appropriate, and can end-to-end architectures be developed that meet design specification requirements?
   (c) Can a single DNN serve the diverse needs of generating mechanism design concepts?
   (d) How can the approach be scaled up to more complex mechanisms, such as spherical and spatial mechanisms?

The extant literature provides evidence that DNN-based techniques have the potential to solve challenging mechanism design problems. However, the state-of-the-art has only scratched the surface of the research questions that can be formulated, and we have identified some key questions that we believe are central to the progress of adapting DNN-based approaches to robotic mechanism design.

Acknowledgment

This work was supported in part by NSF STTR phase 2 award #2126882 and NSF CMMI award #1853454. All opinions and conclusions presented in this paper are those of authors only and not of funding agencies.

Conflict of Interest

The first author of this publication has research support from an NSF STTR phase 2 award (#2126882) as the Co-PI and also holds stock in Mechanismic Inc. The research findings included in this publication 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.

Data Availability Statement

No data, models, or code were generated or used for this paper.

References

[1] Hornik, K., Stinchcombe, M. B., and White, H. L., 1989, "Multilayer Feedforward Networks Are Universal Approximators," Neural Netw., **2**(5), pp. 359–366.
[2] Hornik, K., 1991, "Approximation Capabilities of Multilayer Feedforward Networks," Neural Netw., **4**(2), pp. 251–257.
[3] Kratsios, A., and Bilokopytov, E., 2020, "Non-Euclidean Universal Approximation," Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS'20, Virtual, Curran Associates Inc.
[4] Schulz, H., and Behnke, S., 2012, "Deep Learning," KI - Künstliche Intelligenz, **26**(4), pp. 357–363.
[5] Bengio, Y., Courville, A., and Vincent, P., 2013, "Representation Learning: A Review and New Perspectives," IEEE Trans. Pattern Anal. Mach. Intell., **35**(8), pp. 1798–1828.
[6] LeCun, Y., Bengio, Y., and Hinton, G., 2015, "Deep Learning," Nature, **521**(7553), pp. 436–444.
[7] Zhou, D.-X., 2019, "Universality of Deep Convolutional Neural Networks," Appl. Comput. Harmonic Anal., **48**(2), pp. 787–794.
[8] Krizhevsky, A., Sutskever, I., and Hinton, G. E., 2017, "Imagenet Classification With Deep Convolutional Neural Networks," Commun. ACM, **60**(6), pp. 84–90.
[9] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G., 2009, "The Graph Neural Network Model," IEEE Trans. Neural Netw., **20**(1), pp. 61–80.
[10] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I., 2017, "Attention is All You Need," Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Long Beach, CA, Dec. 4–9, Curran Associates Inc., pp. 6000–6010.
[11] Regenwetter, L., Nobari, A. H., and Ahmed, F., 2022, "Deep Generative Models in Engineering Design: A Review," ASME J. Mech. Des., **144**(7), p. 071704.
[12] Panchal, J. H., Fuge, M., Liu, Y., Missoum, S., and Tucker, C., 2019, "Machine Learning for Engineering Design," ASME J. Mech. Des., **141**(11), p. 110301.
[13] 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.
[14] 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.
[15] 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.
[16] Ge, Q., Purwar, A., Zhao, P., and Deshpande, S., 2017, "A Task-Driven Approach to Unified Synthesis of Planar Four-Bar Linkages Using Algebraic Fitting of a Pencil of G-Manifolds," ASME J. Comput. Inf. Sci. Eng., **17**(3), p. 031011.
[17] 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.
[18] 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.
[19] Deshpande, S., and Purwar, A., 2019, "A Machine Learning Approach to Kinematic Synthesis of Defect-Free Planar Four-Bar Linkages," ASME J. Comput. Inf. Sci. Eng., **19**(2), p. 021004.
[20] 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.
[21] Mcgarva, J., and Mullineux, G., 1993, "Harmonic Representation of Closed Curves," Appl. Math. Model., **17**(4), pp. 213–218.
[22] Mcgarva, J. R., 1994, "Rapid Search and Selection of Path Generating Mechanisms From a Library," Mech. Mach. Theory, **29**(2), pp. 223–235.
[23] 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.
[24] 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. Rob., **3**(3), p. 031002.
[25] Li, X., Wu, J., and Ge, Q. J., 2016, "A Fourier Descriptor-Based Approach to Design Space Decomposition for Planar Motion Approximation," ASME J. Mech. Rob., **8**(6), p. 064501.
[26] Sharma, S., Purwar, A., and Ge, Q. J., 2019, "An Optimal Parametrization Scheme for Path Generation Using Fourier Descriptors for Four-Bar Mechanism Synthesis," ASME J. Comput. Inf. Sci. Eng., **19**(1), p. 014501.
[27] Chuang, G. C.-H., and Kuo, C.-C. J., 1996, "Wavelet Descriptor of Planar Curves: Theory and Applications," IEEE Trans. Image Process., **5**(1), pp. 56–70.
[28] Nabout, A. A., and Tibken, B., 2005, "Wavelet Descriptors for Object Recognition Using Mexican Hat Function," IFAC Proc., **38**(1), pp. 1107–1112.
[29] Liu, W., Sun, J., Zhang, B., and Chu, J., 2018, "Wavelet Feature Parameters Representations of Open Planar Curves," Appl. Math. Modell., **57**, pp. 614–624.
[30] Sun, J., Liu, W., and Chu, J., 2015, "Dimensional Synthesis of Open Path Generator of Four-Bar Mechanisms Using the Haar Wavelet," ASME J. Mech. Des., **137**(8), p. 082303.
[31] Osowski, S., and Nghia, D. D., 2002, "Fourier and Wavelet Descriptors for Shape Recognition Using Neural Networks—A Comparative Study," Pattern Recogn., **35**(9), pp. 1949–1957.
[32] Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P., 2017, "Geometric Deep Learning: Going Beyond Euclidean Data," IEEE Signal Process. Mag., **34**(4), pp. 18–42.
[33] Cao, W., Yan, Z., He, Z., and He, Z., 2020, "A Comprehensive Survey on Geometric Deep Learning," IEEE Access, **8**, pp. 35929–35949.
[34] 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.
[35] Kingma, D. P., and Welling, M., 2014, "Auto-Encoding Variational Bayes," Computing Research Repository, abs/1312.6114.
[36] Kingma, D. P., and Welling, M., 2019, "An Introduction to Variational Autoencoders," Found. Trends® Mach Learn., **12**(4), pp. 307–392.
[37] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., and Bharath, A. A., 2018, "Generative Adversarial Networks: An Overview," IEEE Signal Process. Mag., **35**(1), pp. 53–65.
[38] Sharma, S., Purwar, A., and Ge, Q. J., 2019, "A Motion Synthesis Approach to Solving Alt-burmester Problem by Exploiting Fourier Descriptor Relationship Between Path and Orientation Data," ASME J. Mech. Rob., **11**(1), p. 011016.
[39] Sharma, S., and Purwar, A., 2022, "A Machine Learning Approach to Solve the Alt-burmester Problem for Synthesis of Defect-Free Spatial Mechanisms.," J. Comput. Inf. Sci. Eng., **22**(2), p. 021003.
[40] Tsai, L., 2001, *Mechanism Design: Enumeration of Kinematic Structures According to Function*, CRC Press LLC, Boca Raton, FL.
[41] Zhao, P., Li, X., Purwar, A., and Ge, Q. J., 2016, "A Task-Driven Unified Synthesis of Planar Four-Bar and Six-Bar Linkages With R- and P-Joints for Five-Position Realization," ASME J. Mech. Rob., **8**(6), p. 061003.
[42] Sharma, S., and Purwar, A., 2020, "Using a Point-Line-Plane Representation for Unified Simulation of Planar and Spherical Mechanisms," ASME J. Comput. Inf. Sci. Eng., **20**(6), p. 061002.

[43] Ophaswongse, C., and Agrawal, S. K., 2020, "Optimal Design of a Novel 3-dof Orientational Parallel Mechanism for Pelvic Assistance on a Wheelchair: An Approach Based on Kinematic Geometry and Screw Theory," IEEE Rob. Autom. Lett., **5**(2), pp. 3315–3322.
[44] Murray, R. C., Ophaswongse, C., Park, J. H., and Agrawal, S. K., 2020, "Characterizing Torso Stiffness in Female Adolescents With and Without Scoliosis," IEEE Rob. Autom. Lett., **5**(2), pp. 1634–1641.
[45] Ophaswongse, C., Lent, V., and Agrawal, S. K., 2022, "Kinematic Validation of a Robotic Exoskeleton for Assisting Seated Pelvic Movements by Wheelchair Users With Trunk Impairments," 2022 9th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob), Seoul, South Korea, Aug. 21–24, pp. 1–6.
[46] Oring, A., Yakhini, Z., and Hel-Or, Y., 2020, "Faithful Autoencoder Interpolation by Shaping the Latent Space," CoRR, abs/2008.01487.
[47] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S., 2015, *Deep Unsupervised Learning Using Nonequilibrium Thermodynamics*, PMLR, pp. 2256–2265.
[48] Dhariwal, P., and Nichol, A., 2021, "Diffusion Models Beat GANs on Image Synthesis," Adv. Neural Inform. Process. Syst., **34**, pp. 8780–8794.
[49] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, U., and Polosukhin, I., 2017, "Attention Is All You Need," Adv. Neural Inform. Process. Syst., Long Beach, CA.
[50] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J., 2019, "Transformers: State-of-the-Art Natural Language Processing," Conference on Empirical Methods in Natural Language Processing, Hong Kong, China.
[51] Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L., 2021, "Physics-Informed Machine Learning," Nat. Rev. Phys., **3**(6), pp. 422–440.
[52] Raissi, M., Perdikaris, P., and Karniadakis, G. E., 2019, "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations," J. Comput. Phys., **378**, pp. 686–707.