Proceedings of the ASME 2020
International Design Engineering Technical Conferences
and Computers and Information in Engineering Conference
IDETC/CIE2020
August 16-19, 2020, St. Louis, MO, USA

IDETC2020-22731

PATH SYNTHESIS OF DEFECT-FREE SPATIAL 5-SS MECHANISMS USING MACHINE LEARNING

Shashank Sharma, Anurag Purwar[*]
Computer-Aided Design and Innovation Lab
Department of Mechanical Engineering
Stony Brook University
Stony Brook, New York, 11794-2300

ABSTRACT

The synthesis of spatial mechanisms for defect-free path generation has not received a lot of attention. In this paper, we focus on the synthesis of 5-SS mechanisms and use a machine learning based approach. First, we create a coupler path database using a solver based on the iterative Newton-Raphson optimization algorithm. Subsequently, a data cleanup, normalization, balancing, and augmentation pipeline is established based on intrinsic properties of space curves namely curvature and torsion. Finally, we use an unsupervised learning algorithm based on Variational Autoencoder combined with K-means clustering to find a multiplicity of defect-free 5-SS mechanisms and examples are presented.

1. INTRODUCTION

Path synthesis problem is the determination of the dimensions of a kinematic mechanism to guide one of its links through many specified points [1, 2]. Extensive research has been done to solve the path generation problem for planer mechanisms. Analytical methods for synthesis include algebraic methods [3, 4, 5, 6], complex number methods [7] and displacement matrix methods [8]. Optimization-based techniques attempt to minimize an objective function and find mechanisms, which best approximate a curve [9,10,11,12,13,14]. Atlas-based approaches explore the use of curve invariants like Fourier descriptors to intelligently form and search a database of coupler curves [15, 16].

A substantial literature on the synthesis of Spherical mechanisms is also available [17, 18, 19]. However, exploring the path synthesis problem for spatial one degree-of-freedom mechanisms has been relatively limited. Premkumar et al. have proposed an optimization based solution for the synthesis of the RRSC and RRSS spatial mechanisms [20, 21]. Ananthasuresh and Kramer solve the synthesis of the RSCR spatial mechanism using the Generalized Reduced Gradient method of optimization [22]. Jiménez et al. outline a generalized constraint based optimization technique [23]. Sun et al. use an atlas-based approach which uses the Fourier series to compare curves and synthesize RCCC spatial mechanism [24]. In this paper, we focus on the path synthesis of spatial 5-SS mechanisms that haven't been explored before. A 5-SS mechanism has been displayed in Fig. 1.

As can be observed in the figure, spatial 5-SS mechanism coupler paths tend to have multiple branches and circuits. We can see two circuits and each circuit has many branches separated by singularity points. Chase and Mirth discuss in great detail the challenges faced in synthesizing practical one degree-of-freedom mechanisms due to circuit and branch defects [25]. Roth and Freudenstein have discussed the occurrence of defects in mechanism synthesis for path generation using numerical methods [26]. Wampler et al. show that there exist many defective mechanisms for the nine-point path synthesis problem [27]. These defects tend to be more prominent in spatial mechanisms when com-

[*] Address all correspondence to this author at anurag.purwar@stonybrook.edu

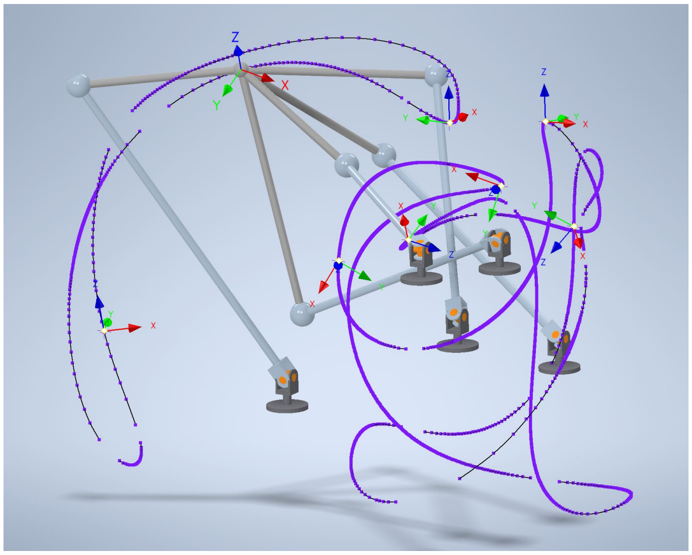

FIGURE 1: Sample 5-SS mechanism

pared to planar mechanisms. Thus, to synthesize practical spatial mechanisms, we consider path synthesis for defect-free mechanisms.

In this paper, we use unsupervised machine learning algorithms to synthesize mechanisms. First, the relevant data is generated using Newton-Raphson based kinematic solver. Then the data is normalized, pruned, and augmented using intrinsic space curve properties like curvature and torsion. After that, we use Variational Autoencoder to generate multiple plausible trajectory signatures that fall in the family of defect-free 5SS coupler curves. These signatures are finally looked up in a hierarchical database created using the K-means clustering algorithm.

Rest of the paper is organized as follows: Section 2 presents the numerical approach to generate 5-SS coupler curves; Section 3 discusses the methodology devised to improve data quality; Section 4 uses unsupervised machine learning tools to calculate multiple solution mechanisms and finally, Section 5 shows two examples solving the path synthesis problem.

2. DATA GENERATION

For a machine learning based approach to work, a large amount of data is required. In this paper, this data would include 5-SS mechanisms with their coupler point trajectory. This is achieved by creating a Newton-Raphson method based solver which uses the general constraint equations proposed in our previous work [28] given in Eq. (1).

During a spatial motion, a 5-SS spatial mechanism is subjected to a set of constraints imposed by the rigidity of its links. The general constraint enforces the rigidity of a binary link with two spherical joints represented by two homogeneous point coordinates of the fixed point (a_1,a_2,a_3,a_4) and floating point (c_1,c_2,c_3,c_4), where a_4 and c_4 are homogenizing factors. The constraint equation is given as

$$C_{SS}: 2a_1c_1+2a_2c_2+2a_3c_3+a_0c_4 = a_4\left(\frac{c_1^2+c_2^2+c_3^2}{c_4}\right), \quad (1)$$

where a_0 is given as

$$a_0 = a_4r^2 - \frac{a_1^2+a_2^2+a_3^2}{a_4}. \quad (2)$$

Here, r is the radius of the sphere formed by the SS link with the center given by (a_1,a_2,a_3,a_4).

A spatial 5-SS mechanism is subjected to seventeen independent rigidity constraints. These include the five constraints for the SS-dyads and twelve constraints for the coupler link. During a simulation, there exist eighteen unknown parameters and fifteen known parameters. The (x,y,z) coordinates of the five fixed pivots are the known parameters while the (x,y,z) of the five moving pivots and the coupler point are the unknowns. This results in one degree of freedom motion.

To actuate the 5-SS mechanism, a linear actuator placed between the fixed pivot of the first dyad and moving pivot of the second dyad. Liao and McCarthy also use the same actuation scheme in their paper on seven pose synthesis of 5-SS linkages [29]. The length of the actuator imposes an additional constraint on the motion and can be defined using Eq (1) as a spherical constraint with a changing radius. Now, to simulate the mechanism, the input actuator is iteratively perturbed by a finite displacement and the new position of the mechanism is calculated until the algorithm fails to converge. Newton-Raphson algorithm fails to converge at singular configurations and these configurations occur at the extreme of each defect-free trajectory where circuit defect occurs.

For a 5-SS mechanism, a system of eighteen unknowns and eighteen constraint equations can be formed and is represented as

$$\mathbf{\Phi}(\mathbf{q}) = 0 \quad (3)$$

where $\mathbf{q}$ is the state vector that consists of the unknown coordinates. The well-known Newton-Raphson method can be used to solve this nonlinear system of equations and get a unique solution. Since the linear actuator is perturbed by a small finite displacement, the previous state of mechanism serves as a good initial approximation.

The iterative simulation algorithm followed can be defined as

$$\mathbf{q}_{i+1} = \mathbf{q}_i - [\boldsymbol{J}^{-1}(\mathbf{q}_i)]\mathbf{\Phi}(\mathbf{q}_i) \quad (4)$$

where $\mathbf{q}_i$ is the state vector at i^{th} iteration, $\mathbf{\Phi}(\mathbf{q}_i)$ is the vector of residuals at $\mathbf{q} = \mathbf{q}_i$, and $[\boldsymbol{J}^{-1}(\mathbf{q}_i)]$ is the inverse of Jacobian matrix evaluated at $\mathbf{q} = \mathbf{q}_i$. The Jacobian matrix is of the following form

$$[\boldsymbol{J}(\mathbf{q})] = \begin{bmatrix} \frac{\partial \phi_1}{\partial q_1} & \frac{\partial \phi_1}{\partial q_2} & \cdots & \frac{\partial \phi_1}{\partial q_{16}} \\ \frac{\partial \phi_2}{\partial q_1} & \frac{\partial \phi_2}{\partial q_2} & \cdots & \frac{\partial \phi_2}{\partial q_{16}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial \phi_{16}}{\partial q_1} & \frac{\partial \phi_{16}}{\partial q_2} & \cdots & \frac{\partial \phi_{16}}{\partial q_{16}} \end{bmatrix}. \tag{5}$$

To calculate the Jacobian matrix, relations describing the first order partial derivatives of constraint equations are required. For an SS dyad described in Eq (1), the first order partial derivatives can be given as follows

$$\frac{\partial C_{SS}}{\partial a_1} = 2(a_1 - c_1) \tag{6}$$

$$\frac{\partial C_{SS}}{\partial a_2} = 2(a_2 - c_2) \tag{7}$$

$$\frac{\partial C_{SS}}{\partial a_3} = 2(a_3 - c_3) \tag{8}$$

Here, the homogeneous point coordinate a_4 and c_4 has been assumed as unity without loss in generality.

Thus, by iteratively perturbing the input actuator and solving the constraints for other moving pivot coordinates, we can simulate a 5-SS mechanism and extract the path traced by coupler point. There does exist an accuracy-storage trade-off for the simulation process. The accuracy of the path increases with decrease perturbation magnitude. However, this results in sampling more points on the path and thus needs more storage.

For this paper, we generated a data set of 7,500 defect-free coupler paths using arbitrarily selected 5-SS mechanisms. Fig. 2 shows one of the simulated mechanisms. This database represents a family of paths a general 5-SS mechanism can achieve. In the next section, we discuss the methodology used to refine this data set for machine learning purposes.

3. DATA PREPROCESSING

Before the generated data can be used for machine learning, the data needs to be normalized, cleaned and augmented.

3.1 Normalizing the number of constituent points in each path

A spatial coupler curve is defined as an array of n 3-D data points. In the data set of 7,500 path curves, we observe that n ranges from 2 to 3,126 as can be seen in Fig. 3. Since curves with a very low number of data points do not capture its geometry well, we choose to ignore them. Thus, curves made of less than ten data points are removed resulting in a data set of 7,408 curves.

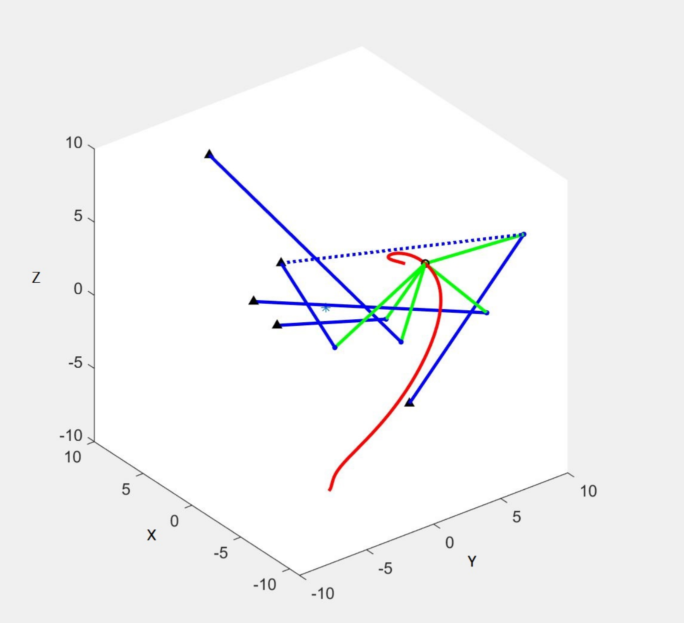

FIGURE 2: A 5-SS mechanism simulation where black triangles are the fixed pivots, blue solid lines are the SS dyads, green lines are the floating coupler link, red curve is the coupler path, and dotted blue line is the linear actuator.

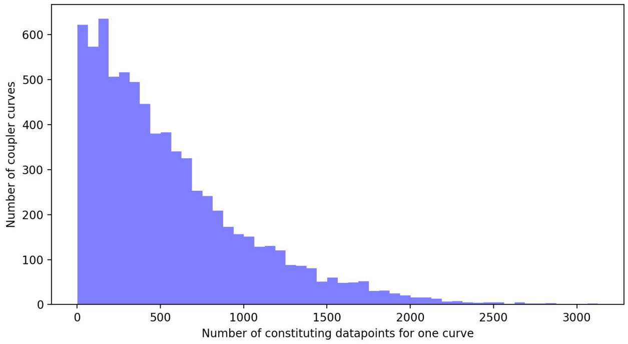

FIGURE 3: Histogram showing number of data points in each path curve included in database

The remaining curves are fitted with a fourth-order B-spline interpolation curve. Then, 100 data points are uniformly sampled on each curve leading to an arc-length based parametrization. The benefit of using this arc-length parametrization is that it allows a unique coupler curve representation which is time-invariant. This property is desirable since it makes comparing two curves with a similar trajectory but different time parametrization much easier as has been demonstrated in Fig. 4.

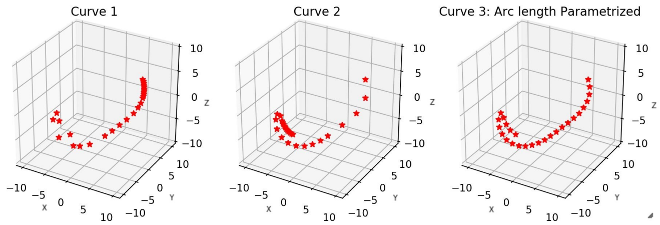

FIGURE 4: Curve 1 and Curve 2 represents the same geometric curve with different time parametrization. They share the same unique arc length parametrization as shown in Curve 3

3.2 Normalizing the location, orientation, and scale of paths

Creating a curve representation which is translation, rotation and scaling invariant is desirable. First, the mean $(\bar{x}, \bar{y}, \bar{z})$ of the curve is calculated and it is translated to origin. Next, the principal axes of the curve are rotated to align with x, y, z axes. The principal component axes are the eigenvectors of the covariance matrix of the point cloud that defines the curve. Also, the curves are scaled to unit arc-length. The effect of normalization on a sample path curve has been demonstrated in Fig. 5.

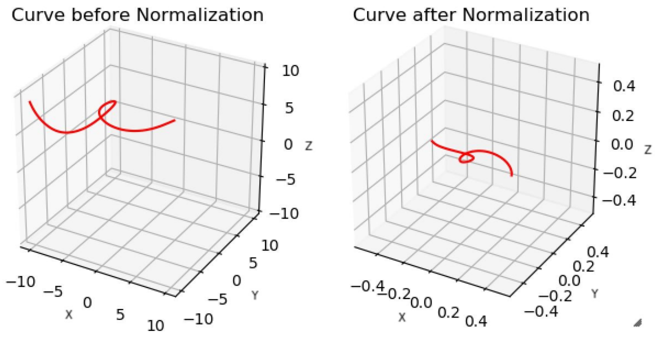

FIGURE 5: Before and after normalizing a path curve

3.3 Incorrect Path Cleanup

When the solver is simulating a 5SS-mechanism, it may jump from one branch to another due to inherent limitations of numerical methods. Due to the discontinuity at points where branch jump happens, the invalid coupler paths have extremely high curvature or torsion. For a spatial curve, Curvature is a scalar measurement of the magnitude of the bending of the curve within the osculating plane at a point as the point moves along the curve. Torsion is a scalar measurement of the amount that the curve bends out of the osculating plane at a point as the point moves along the curve. The curvature and torsion can be calculated as follows

$$\kappa = \frac{||\mathbf{r}'(t) \times \mathbf{r}''(t)||}{||\mathbf{r}'(t)||^3}, \tag{9}$$

$$\tau = \frac{(\mathbf{r}'(t) \times \mathbf{r}''(t)) \cdot \mathbf{r}'''(t)}{||\mathbf{r}'(t) \times \mathbf{r}''(t)||^2}, \tag{10}$$

where $\mathbf{r}(t)$ is the curve.

To isolate the incorrect paths, the Z-score metric, also called the standard score is used. A Z-score indicates how many standard deviations an element is from the mean and is given as

$$z = \frac{X - \mu}{\sigma} \tag{11}$$

where μ is the mean and σ is the standard deviation. We calculate three Z-score of maximum curvature $(Z_{\kappa,max})$, maximum torsion $(Z_{\tau,max})$ and minimum torsion $(Z_{\tau,min})$. In our study, an outlier is defined as any curve having $Z_{\kappa,max} > 1.5$ or $Z_{\tau,max} > 3$ or $Z_{\tau,min} < -3$. An example of an outlier has been shown in Fig. 6. Filtering out the outliers results in a clean database containing 7,200 coupler paths.

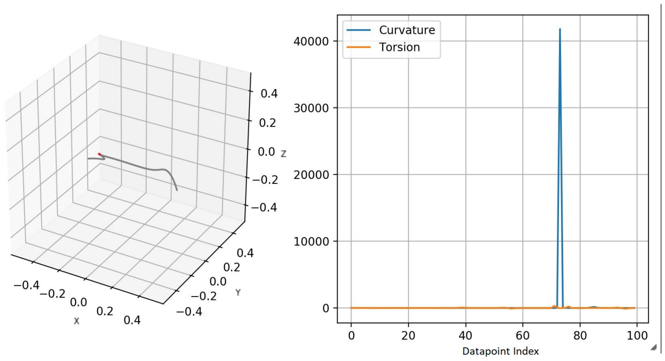

FIGURE 6: An outlier path curve with high curvature. The branch jump occurs at the red dot denoted on curve

3.4 Coupler path diversity balancing

The database in its present form is unbalanced i.e. it has more samples of coupler paths which are more probable while lesser samples of other more diverse examples. This leads to the algorithm not learning well since it comes across the more probable examples most of the time. To overcome this bias, a limited number of diverse paths are selected from the complete database by under sampling similar curves.

According to the fundamental theorem of space curves in differential geometry, every regular curve in three-dimensional space, with non-zero curvature, has its shape completely determined by its curvature and torsion. Thus, a good metric to compare the similarity of two curves are these Curvature–Torsion Descriptors. The curvature is always positive while the torsion can be negative. We define the similarity score (δ) as a weighted sum of P2 norms of difference between curvature and torsion of two curves which is given as

$$\delta = \frac{||\kappa_2(s) - \kappa_1(s)||_2 + w||\tau_2(s) - \tau_1(s)||_2}{n} \tag{12}$$

where w is the weight and n is the total number of constituent points in each curve. Since the numerical calculation of torsion can end up being somewhat inaccurate, we set the weight at $w = .1$ in this paper. We select the similarity metric threshold such that if $\delta < .065$, the two curves are similar and one of them is dropped from the database. It can be observed in Fig. 7 that some curves occur up to 170 times in the database. On further exploring, we find that the common curves represent simple arcs and line trajectories and their reflections as seen in Fig. 8. Undersampling similar curves lead to a balanced data set containing 5,021 coupler paths.

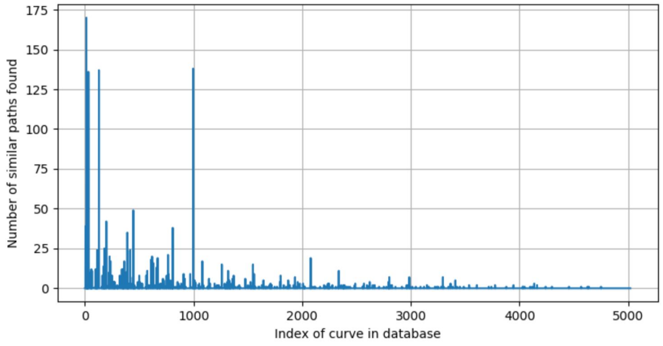

FIGURE 7: Bar graph showing the number of similar curves found for each curve.

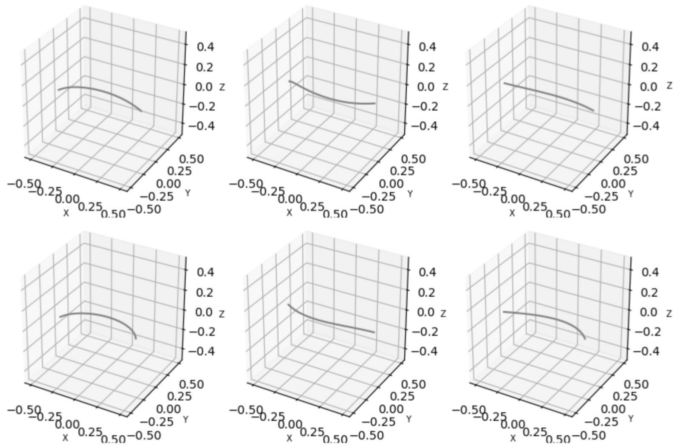

FIGURE 8: Most common variety of curves in the database

3.5 Adding mirrored paths

In kinematics, it is known that if a path is a valid coupler path, its mirrored curve is also a valid path. For the machine learning algorithm to gain this domain knowledge, coupler paths mirrored across xy,yz and zx planes are added to the database. Thus, this step to encourage the model to be invariant to mirror operations. After this step, our database contains 20,084 curves.

3.6 Adding noise to paths

Finally, some Gaussian noise is added to all the curves. This acts as a regularizer to the machine learning algorithm, encourages robust learning, and avoids overfitting.

4. MACHINE LEARNING BASED PATH SYNTHESIS

Now that the database has been normalized, cleaned, and augmented, it can be used to train a machine learning model.

4.1 Training using Variational Autoencoder

The goal of our machine learning model is to learn the distribution behind the family of defect-free 5-SS coupler curves. It should be able to generate multiple plausible trajectories that fall in this family and is similar to the user-inputted path. Also, it should provide a low dimensional signature to the coupler path which can easily be compared to other curves as a similarity metric.

To achieve this, we use a Variational Autoencoder (VAE) which is a type of generative neural network. It trains on coupler path (X_{path}) and approximates the underlying distribution of observed data. As can be seen in Fig. 9, it uses the encoder model to find the latent distributions defined as a multivariate Gaussian distribution defined by mean vector μ and standard deviation vector σ. A latent vector z can then be sampled from this distribution and used to generate similar path trajectories using the decoder model. The encoder is represented as $q_\theta(z|X)$ where θ are the encoder weights and biases while a decoder is represented as $p_\phi(X|z)$ where ϕ denotes decoder weights and biases.

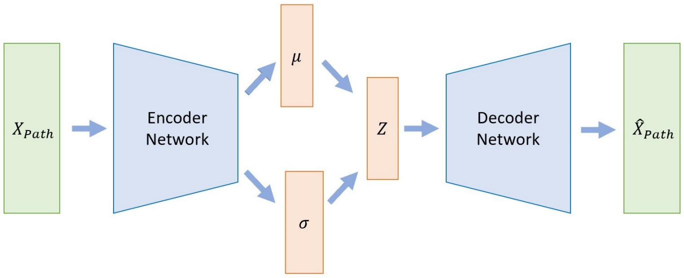

FIGURE 9: Architecture of a Variational Autoencoder

The loss function used to train the VAE is defined as sum of reconstruction loss and KL divergence which is given as

$$Loss = RL + KL \tag{13}$$

$$RL = \sqrt{\sum_{i=1}^{n} d_i^2} \tag{14}$$

$$d_i = \sqrt{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2 + (\hat{z}_i - z_i)^2} \tag{15}$$

$$KL = \sum_{i=1}^{k} \sigma_i^{2}\text{‘} + \mu_i^2 - log(\sigma_i) - 1 \tag{16}$$

where the latent space is k dimensional, each path consists of n points and the reconstruction loss is the Eucledian norm.

Multiple VAEs with different depths and bottlenecks were tested to find the best architecture. The capacity of a network increases with increasing depths and it can describe a much more complex function. However, due to the problem of vanishing gradient, deep networks tend to be harder to train. Thus, there exists an optimal depth that balances complexity and trainability. Similarly, the narrower the bottleneck layer, the better is the dimensionality reduction. However, reducing the width too much can lead to loss of excessive information. Networks with depth=(1,2,3,4) and bottleneck layer width=(15,30,60) were tested and the results are given in Table 1. Each VAE is trained for 3000 epochs with a batch size of 256 using Adam (adaptive moment estimation) optimizer.

We notice that VAE-FC-H3-Z30 performs the best. It contains three hidden fully-connected layers consisting of 200, 100 and 60 nodes each, ReLU activation function after each layer and the bottleneck layer z contains 30 nodes. The training curves of VAE-FC-H3-Z30 can be seen in Fig. 10.

The image processing community has successfully used convolution layers for feature extraction to enhance the performance of image classification algorithms [30]. One of the reasons for this success is that the convolution layers conserve the locality of information while the fully connected layers lose this spatial information. Another advantage of a convolution layer is weight sharing which leads to reduced memory requirements.

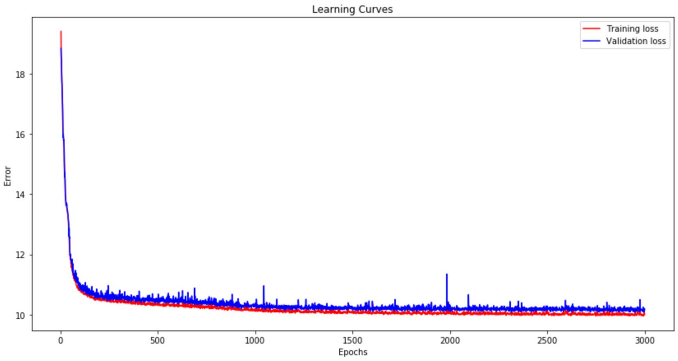

FIGURE 10: Training losses for the fully connected VAE

For spatial curves, the local geometry of a curve segment is heavily dependent on previous and next curve segments due to the continuity constraints. This is especially true in the case of interpolation curves like B-splines where each control point has local effect [31]. Since using convolution layers make sense, they are augmented to the VAE-FC-H3-Z30 model architecture and their effectiveness is empirically tested. The VAE-COV-H8-Z30 has five 1D-convolution layers followed by three fully connected layers. Each convolution layer has a kernel=2, stride=2 and filters=(5,10,15,20,25). The fully connected layer size is (200,100,60) and the bottleneck layer size is 30. 1d-deconvolution is used for the decoder. The learning curves for this model are shown in Fig. 11. We find that this model performs even better and has a training loss of 9.9034 and a validation loss of 10.0311.

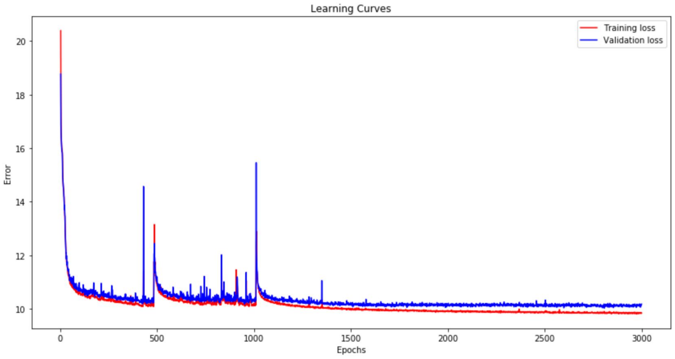

FIGURE 11: Training losses for the Convolutional VAE

Some sample outputs generated using VAE-COV-H8-Z30 are shown in Fig 12 where an input curve (red) generates five trajectories (gray) sampled from the underlying distribution.

TABLE 1: VAE model architectures that were tested and their performances

Name	Encoder Arch.	Latent (z) dim.	Decoder Arch.	Training loss	Validation loss
VAE-FC-H1-Z15	(100)	15	(100)	10.5323	10.5466
VAE-FC-H1-Z30	(100)	30	(100)	10.8516	10.8248
VAE-FC-H1-Z60	(100)	60	(100)	10.8739	10.8573
VAE-FC-H2-Z15	(150,75)	15	(75,150)	10.2167	10.2773
VAE-FC-H2-Z30	(150,75)	30	(75,150)	10.2189	10.2589
VAE-FC-H2-Z60	(150,75)	60	(75,150)	10.2577	10.2853
VAE-FC-H3-Z15	(200,100,60)	15	(60,100,200)	10.0575	10.0999
VAE-FC-H3-Z30	(200,100,60)	30	(60,100,200)	9.9742	10.0760
VAE-FC-H3-Z60	(200,100,60)	60	(60,100,200)	10.0260	10.0821
VAE-FC-H4-Z15	(200,150,100,60)	15	(60,100,150,200)	10.0175	10.1155
VAE-FC-H4-Z30	(200,150,100,60)	30	(60,100,150,200)	10.1348	10.1911
VAE-FC-H4-Z60	(200,150,100,60)	60	(60,100,150,200)	10.2031	10.2924

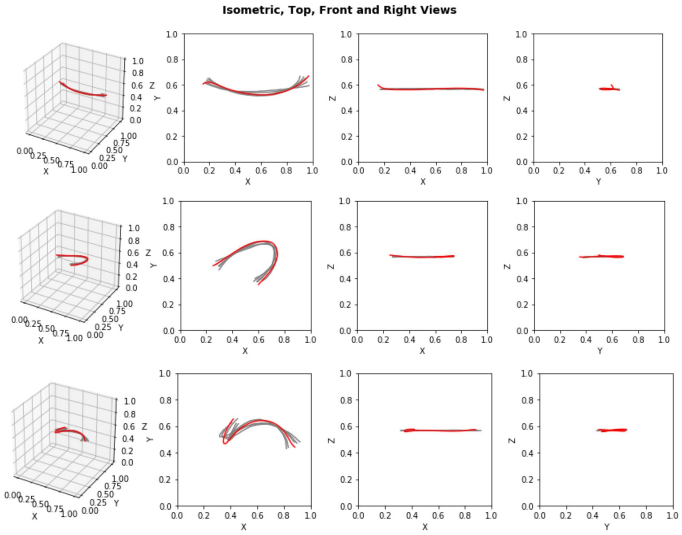

FIGURE 12: Comparing X (red curve) and $\hat{X}$ (grey curves)

4.2 Creating a Hierarchical Database

Once the training is completed, the recognition model is used to generate signatures of coupler curves in the database denoted by the μ vector. Then, these signatures are clustered into 500 groups using K-Means Clustering. The distance metric used is the Euclidean distance. As a result, we get 500 cluster centers subdividing the original dataset of 20k+ coupler curves.

4.3 Mechanism synthesis for User Inputted Trajectory

When the user inputs a curve X_{path}, it's run through the encoder network of VAE to find the μ and σ vectors. Multiple z vectors are then sampled from the latent distribution which denotes a family of feasible curve signatures. These curve signatures are then compared to each of the cluster centers using the P2 norm error metric. Once a center is selected, the best available mechanism within the cluster can be returned to the user as a feasible solution. Thus, the user can find multiple defect-free solution mechanisms.

5. EXAMPLES OF PATH SYNTHESIS FOR SPATIAL CURVE

In this section, we provide two examples of our algorithm in action. In the examples, we input a spatial trajectory. The trajectory is then processed by the encoder of our VAE resulting in a 30-dimensional Gaussian distribution specified by μ and σ. We sample five latent vectors z from this distribution and look up the closest cluster centers in our database. In the cluster, we find the best approximation of the coupler path available and provide it as a solution.

The input trajectory is shown in the first plot in Fig. 13 and Fig. 14. The other plots show a prospective 5-SS solution that closely matches the target path. More mechanisms can be generated by sampling additional latent vectors from the VAE.

6. CONCLUSION

Thus, in this paper, we have discussed a complete pipeline including data generation, data cleanup, and machine learning model creation for defect-free path synthesis of spatial 5-SS mechanism. To generate coupler path data, we use a geometric constraint based numerical approach which uses Newton-Raphson optimization. Then, the data is pre-processed using intrinsic curve properties including curvature and torsion. Finally, unsupervised machine learning techniques of VAE and K-mean clustering are used to efficiently find solution mechanisms.

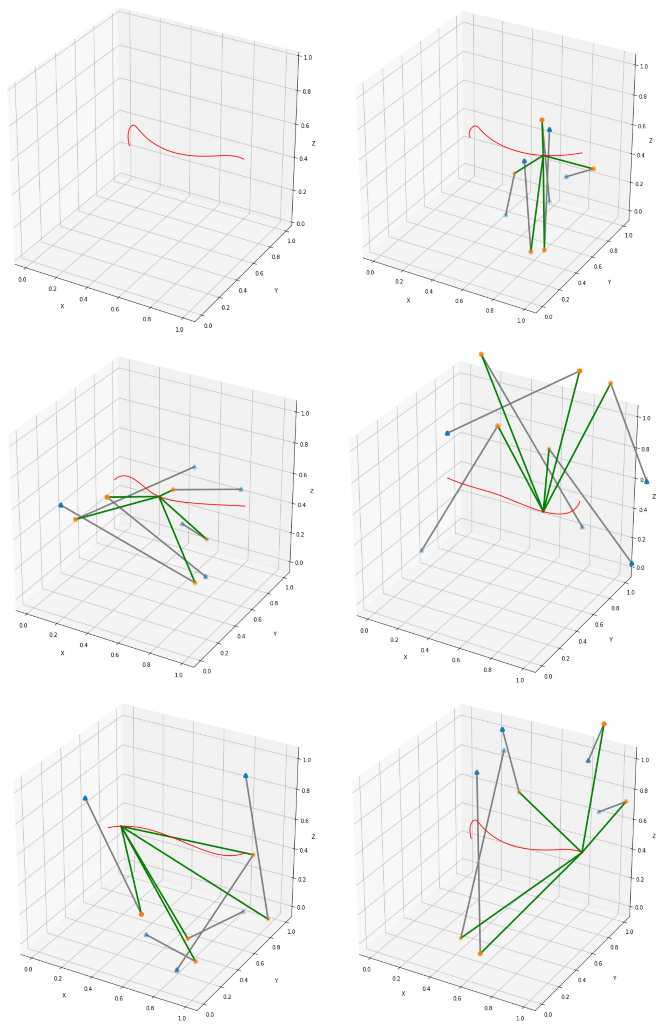

FIGURE 13: Example 1

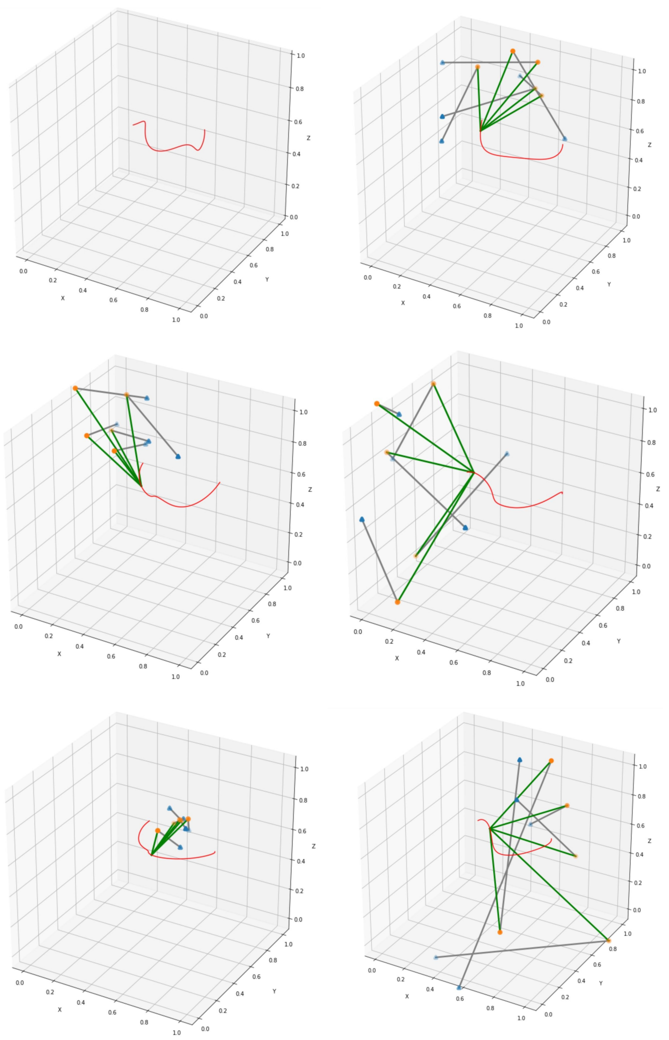

FIGURE 14: Example 2

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.

REFERENCES

[1] Erdman, A. G. and Sandor, G. N., 1997, Mechanism Design: Analysis and Synthesis, Prentice Hall, NJ, 3rd edition.

[2] McCarthy, J. M. and Soh, G. S., 2010, Geometric design of linkages, volume 11, Springer.

[3] Freudenstein, F., 1954, “An analytical approach to the design of four-link mechanisms”, Trans. ASME, **76(3)**, pp. 483–492.

[4] Hartenberg, R. S. and Denavit, J., 1964, Kinematic Synthesis of Linkages, McGraw-Hill, New York.

[5] Blechschmidt, J. L. and Uicker, J. J., 1986, “Linkage Synthesis Using Algebraic-Curves”, Journal of Mechanisms Transmissions and Automation in Design-Transactions of the Asme, **108(4)**, pp. 543–548.

[6] Deshpande, S. and Purwar, A., 2017, “A Task-Driven Approach to Optimal Synthesis of Planar Four-Bar Linkages for Extended Burmester Problem”, ASME Journal of Mechanisms and Robotics, **9(6)**, doi:10.1115/1.4037801, URL https://doi.org/10.1115/1.4037801, 061005.

[7] Suh, C. H. and Radcliffe, C. W., 1978, Kinematics and Mechanism Design, John Wiley and Sons, New York.

[8] Erdman, A. G. and Sandor, G. N., 1991, Advanced Mechanism Design: Analysis and Synthesis, volume 2, Prentice-Hall, Englewood Cliffs, NJ, 2nd edition.

[9] Nolle, H. and Hunt, K. H., 1971, “Optimum Synthesis of Planar Linkages to Generate Coupler Curves”, Journal of Mechanisms, **6(3)**, p. 267.

[10] Sancibrian, R., Viadero, F., Garcia, P., and Fernandez, A., 2004, “Gradient-based optimization of path synthesis problems in planar mechanisms”, Mechanism and Machine Theory, **39(8)**, pp. 839–856.

[11] Ullah, I. and Kota, S., 1997, “Optimal synthesis of mechanisms for path generation using Fourier descriptors and global search methods”, ASME Journal of Mechanical Design, **119(4)**, pp. 504–510.

[12] Wu, J., Ge, Q. J., and Gao, F., 2009, “An Efficient Method for Synthesizing Crank-Rocker Mechanisms for Generating Low Harmonic Curves”, ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, volume 2009, dETC2009-87140.

[13] 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 Journal of Computing and Information Science in Engineering, **19(1)**, p. 014501.

[14] Deshpande, S. and Purwar, A., 2019, “A Machine Learning Approach to Kinematic Synthesis of Defect-Free Planar Four-Bar Linkages”, ASME Journal of Computing and Information Science in Engineering, **19(2)**, pp. 021004–021004–10, 10.1115/1.4042325.

[15] Chu, J. K. and Sun, J. W., 2010, “A New Approach to Dimension Synthesis of Spatial Four-Bar Linkage Through Numerical Atlas Method”, Journal of Mechanisms and Robotics-Transactions of the Asme, **2(4)**.

[16] Wandling Sr, G. R., 2000, “Synthesis of mechanisms for function, path, and motion generation using invariant characterization, storage and search methods”, Ph. d. thesis.

[17] Chiang, C. H., 2000, Kinematics of spherical mechanisms, Krieger Pub., Malabar, FL, 00039067 C.H. Chiang. ill. ; 23 cm. Includes bibliographical references and index.

[18] Chu, J. and Sun, J., 2010, “Numerical atlas method for path generation of spherical four-bar mechanism”, Mechanism and Machine Theory, **45(6)**, pp. 867 – 879, doi: https://doi.org/10.1016/j.mechmachtheory.2009.12.005, URL http://www.sciencedirect.com/science/article/pii/S0094114X09002286.

[19] Mullineux, G., 2011, “Atlas of spherical four-bar mechanisms”, Mechanism and Machine Theory, **46(11)**, pp. 1811 – 1823, doi:https://doi.org/10.1016/j.mechmachtheory.2011.06.001, URL http://www.sciencedirect.com/science/article/pii/S0094114X11001121.

[20] Premkumar, P., Dhall, S. R., and Kramer, S. N., 1988, “Selective Precision Synthesis of the Spatial Slider Crank Mechanism for Path and Function Generation”, ASME Journal of Mechanisms, Transmissions, and Automation in Design, **110(3)**, pp. 295–302, doi:10.1115/1.3267461, URL https://doi.org/10.1115/1.3267461.

[21] Premkumar, P. and Kramer, S. N., 1989, “Position, Velocity, and Acceleration Synthesis of the RRSS Spatial Path-Generating Mechanism Using the Selective Precision Synthesis Method”, ASME Journal of Mechanisms, Transmissions, and Automation in Design, **111(1)**, pp. 54–58, doi:10.1115/1.3258971, URL https://doi.org/10.1115/1.3258971.

[22] Ananthasuresh, G. K. and Kramer, S. N., 1994, “Analysis and Optimal Synthesis of the RSCR Spatial Mechanisms”, ASME Journal of Mechanical Design, **116(1)**, pp. 174–181, doi:10.1115/1.2919342, URL https://doi.org/10.1115/1.2919342.

[23] Jiménez, J., Álvarez, G., Cardenal, J., and Cuadrado, J., 1997, “A simple and general method for kinematic synthesis of spatial mechanisms”, Mechanism and Machine Theory, **32(3)**, pp. 323 – 341, doi:https://doi.org/10.1016/S0094-114X(96)00017-1, URL http://www.sciencedirect.com/science/article/pii/S0094114X96000171.

[24] Sun, J. W., Mu, D. Q., and Chu, J. K., 2012, “Fourier series method for path generation of RCCC mechanism”, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, **226(3)**, pp. 816–827, doi:10.1177/0954406211416176, URL https://doi.org/10.1177/0954406211416176.

[25] Chase, T. and Mirth, J., 1993, “Circuits and Branches of Single-Degree-of-Freedom Planar Linkages.”, ASME Journal of Mechanical Design, **115(2)**, p. 223–230.

[26] Roth, B. and Freudenstein, F., 1963, “Synthesis of Path-Generating Mechanisms by Numerical Methods”, ASME Journal of Engineering for Industry, **85(3)**, pp. 298–304, doi:10.1115/1.3669870, URL https://doi.org/10.1115/1.3669870.

[27] Wampler, C. W., Morgan, A. P., and Sommese, A. J., 1992, “Complete Solution of the Nine-Point Path Synthesis Problem for Four-Bar Linkages”, ASME Journal of Mechanical Design, **114(1)**, pp. 153–159, doi:10.1115/1.2916909, URL https://doi.org/10.1115/1.2916909.

[28] Sharma, S. and Purwar, A., 2019, “Using a Point-Line-Plane Representation for Unified Simulation of Planar and Spherical Mechanisms”, volume Volume 5A: 43rd Mechanisms and Robotics Conference of *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, doi: 10.1115/DETC2019-98194, URL https://doi.org/10.1115/DETC2019-98194, v05AT07A030.

[29] Liao, Q. and McCarthy, J. M., 1997, “On the Seven Position Synthesis of a 5-SS Platform Linkage ”, ASME Jour-

nal of Mechanical Design, **123(1)**, pp. 74–79, doi:10.1115/1.1330269, URL https://doi.org/10.1115/1.1330269.

[30] Rawat, W. and Wang, Z., 2017, “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review”, Neural Computation, **29(9)**, pp. 2352–2449, doi:10.1162/neco_a_00990, URL https://doi.org/10.1162/neco_a_00990, pMID: 28599112.

[31] Piegl, L. and Tiller, W., 1995, The NURBS Book, Springer, Berlin.