

A Machine Learning Approach to Kinematic Synthesis of Defect-Free Planar Four-Bar Linkages

Shrinath Deshpande, Anurag Purwar*
Computer-Aided Design and Innovation Lab
Department of Mechanical Engineering
Stony Brook University
Stony Brook, New York, 11794-2300

The past forty years of research in mechanism synthesis has witnessed an unprecedented volume of work in formulating and solving planar four-bar linkage synthesis problems. However, finding practical and useful mechanisms for the motion synthesis problem has proven to be elusive, as a large majority of mechanisms turn out to be defective with respect to their assembly modes. Most methods formulate the problem as a discrete precision position problem, which inherently ignores the continuity information in the input, resulting in linkages with branch-, circuit- and order-defects. In this paper, we bring together diverse fields of pattern recognition, machine learning, artificial neural network, and computational kinematics to present a novel approach that solves this problem both efficiently and effectively. At the heart of this approach lies an objective function that compares the motion as a whole thereby capturing designer's intent. In contrast to widely used structural error or loop-closure equation based error functions which convolute the optimization by considering shape, size, position, and orientation simultaneously, this objective function computes motion difference in a form, which is invariant to similarity transformations. We employ auto-encoder neural networks to create a compact and clustered database of invariant motions of known linkages. The query is raised in the database for nearest neighbors, which are either solutions or good initial conditions for fast local optimization techniques. In spite of highly nonlinear parameters space, our approach discovers a wide pool of defect-free solutions very quickly. We show that by employing proven machine learning techniques, this work could have far-reaching consequences to creating a multitude of useful and creative conceptual design solutions for mechanism synthesis problems, which go beyond planar four-bar linkages.

keywords: *Machine Learning, Artificial Neural Network (ANN), Path Synthesis, Motion Synthesis, Planar Four-bar linkage, Circuit- Order-, and Branch-Defect*

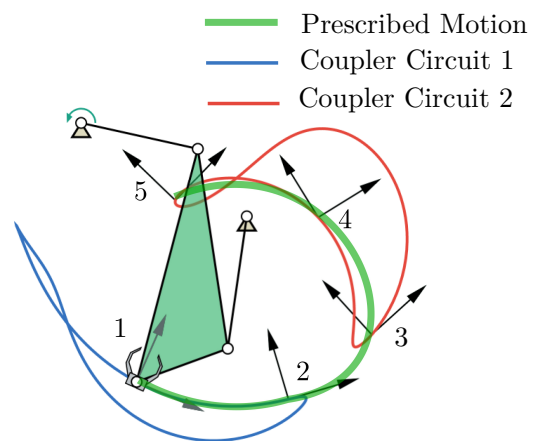


Fig. 1: The mechanism synthesis problem for the prescribed motion. The four-bar mechanism obtained using the precision position approach suffers from circuit defect, as no coupler circuit passes through all precision positions.

1 Introduction

The classical mechanism synthesis problem deals with computing type and dimensions of a linkage system for performing specific tasks, which are categorized as the path, motion and function generation. Path generation task aims to synthesize a mechanism which can guide a point along a prescribed path. On the other hand, the objective of motion generation task is to synthesize a mechanism which can guide a rigid body along a prescribed motion. The prescribed path is a time sequence of positions, whereas the prescribed motion is a time sequence of positions and orientations. For an exhaustive reference, see books from McCarthy and Soh [1], Sandor and Erdman [2], Hunt [3], Hartenberg and Denavit [4], Suh and Radcliffe [5], and Lohse [6]. Various approaches have been proposed in the literature with aim of finding acceptable solutions for path and motion generation problem. In the majority of the methods, the first step is choosing a mechanism type and then calculating the de-

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

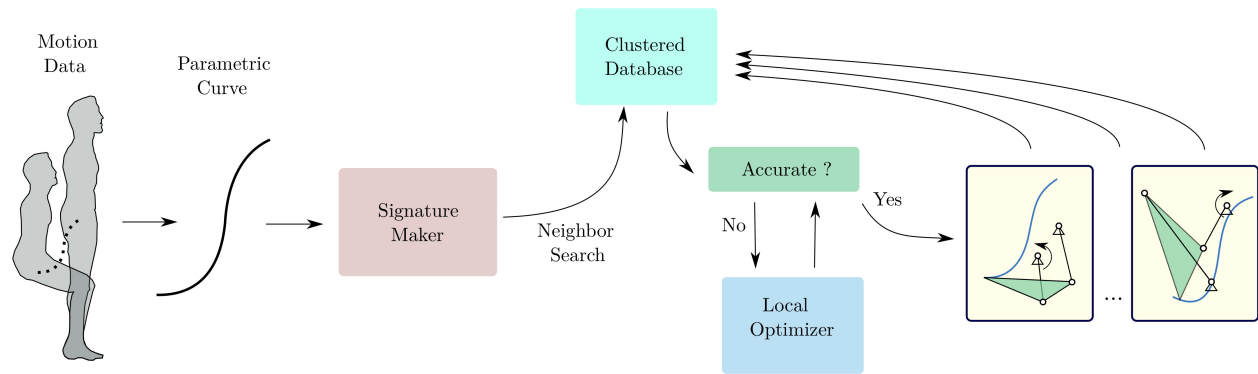


Fig. 2: The Machine Learning approach begins by creating an invariant signature for the path and the motion data, which facilitates a compact and hierarchical clustered database and an auto-encoder Neural Network trained to elicit good, defect-free solutions or subjected to local, fast optimization. The results are astonishingly good, defect-free concept design solutions for input problems.

sign parameters of chosen mechanism type. The first step is called as Type Synthesis, and the later step is called as Dimension Synthesis. This paper presents an approach to Dimension Synthesis of planar four-bar linkages, but the approach is general enough to be applied on any planar linkage.

1.1 Motivation

The majority of mechanism synthesis methods are based on the precision point approach. This approach is a clever approximation trick to solve the abovementioned tasks, where the task is discretized into precision points. These points instead of the actual task, are treated as interpolation or approximation task for mechanism synthesis. However, this way of formulation loses important information about the actual continuous task and can lead to solutions with the order, circuit and branch defects. The defect in a mechanism renders itself useless for its intended application. The defects in the single degree of freedom mechanisms are thoroughly discussed by Chase and Mirth [7].

Let's see an example of this in a motion generation task. The objective is to synthesize a four-bar mechanism that can perform the prescribed motion. Figure 1 shows the prescribed motion and its discretized precision positions. In the case of motion generation problem, a four-bar can go through at most five precision positions and it is called as classical Burmester [8] problem. For this type of problem, only finite solutions exist and in this case only one solution is obtained as shown in the Fig. 1. It can be seen that although coupler of the four-bar passes exactly through five precision positions, it can not do it without changing the circuit. A circuit represents an assembly mode in which the mechanism is put together. This phenomenon is called circuit defect in the linkage, which makes the linkage useless for the prescribed task. To deal with it, a naive approach is to tweak precision positions in a brute-force way within the tolerance until a solution is found. Even if the defect-free solution is found, the coupler motion in-between the precision points may go through

undesired position and orientation. This happens because the method discards the functional aspect of continuous motion and turns it into a vacuous interpolation problem. This tells us that just finding a defect-free linkage with respect to the precision positions is not enough to qualify as a useful mechanism.

Thus, we present a perceptive way of synthesizing defect free linkages for motion and path generation problems. To capture the complete prescribed task, we represent it as a continuous parametric curve. The objective is to find a linkage, that has a coupler motion or path compatible with the task. We developed the compatibility measure in a way that is invariant to similarity transformations. Next step is to conduct a search in the space of linkage parameters to find linkages with coupler motions compatible with the prescribed task. Although global search methods can be applied for finding solutions, we employ an efficiently clustered database and Powell's local search method to come up with a variety of different solutions. The motivation behind using clustered database is rooted in the broader objective of machine design. Mechanism synthesis is a part of the conceptual design phase, which is one of the earliest stages of product design cycle of a machine. Thus, the synthesis method should be prolific in terms of concept generation to 1) realize the potential of attainable design possibilities, and 2) have the agility to adapt a design to evolving requirements. The method only deals with the coupler curves and is indifferent to the linkage type. Thus, it is readily scalable to any type of planar linkage. In section 6.2, we present the obtained solutions for the prescribed motion presented in the Fig. 1.

1.2 Literature Survey and Overview

Instead of brute-force search within tolerance regions, some approaches apply separate constraints to take care of the defects and form an optimization problem of Non-Differentiable objective function. It can be solved by meta-heuristic algorithms like Differential-evolution(DE), Particle

Swarm Optimization (PSO), Cuckoo Search (CS). Cabrera et al. [9] used Genetic Algorithm as the tool for optimization in mechanism synthesis. Sardashti et al. [10] used PSO towards the defect-free synthesis of four-bar linkage with joint clearance for path generation problem. Ebrahimi and Payvandy [11] presented an application of Imperialist Competitive Algorithm (ICA) for synthesizing path generating four-bars having desired workspace limits. Bulatovic [12] used Cuckoo Search for solving the problem of optimum synthesis of a six-bar double dwell linkage. The above methods can produce many defect-free solutions and are scalable to any type of linkages. However, these methods tend to be slow and inefficient as they start from scratch for every instance with no guarantee of converging to an acceptable solution.

Path synthesis methods based on Fourier analysis do take the continuity information of coupler path into account. However, most of them are defined only for closed loop curves. Ullah and Kota [13] have presented an invariant approach towards representation and synthesis of closed loop paths through shape optimization. They use a combination of global and local search methods for optimizing Fourier Deviant function to compute the dimensions of planar four-bar linkages without an initial guess. Wu et al. [14] presented a method based on finite Fourier series for open and closed path generation of four-bar mechanisms. In the case of motion generation, Li et al. [15] have developed a Fourier descriptor-based approach for approximate motion generation. Buśkiewicz et al. [16] used curvature of the coupler curve for path synthesis using Genetic Algorithms. Khan et al. [17] presented an approach where an artificial neural network is used for mapping between Fourier coefficients corresponding to a coupler path and corresponding linkage parameters.

Instead of the global search, an alternative approach is to start from a good initial guess based on an atlas and use local search methods. We adopt this approach and combine with our novel formulation to generate a diverse set of concept solutions. McGarva [18] took the earliest approach towards creating a library for coupler trajectories based on the harmonic analysis. Wandling [19] has presented an atlas-based approach, where coupler paths and motions are stored in terms of Fourier Transforms. Input motion is searched for neighbors based on Euclidean distances of Fourier Transforms. Yue et al. [20] presented a similar approach of path generation using P-Type Fourier Descriptor applicable for open curves. In their approach, a task curve is transformed into normalized Fourier coefficients and queried for the nearest neighbor search. The best match is returned as the solution to the input. Chu et al. [21] presented an atlas-based method for synthesizing spatial four-bar linkages for function generation problems, where orientation data is stored in terms of Fourier descriptors.

The above methods generate data based on uniform sampling in the linkage parameter space. Given highly nonlinear mapping between linkage parameters and coupler trajectory, this way of sampling leads to a nonuniform sampling of trajectory space, which causes under-representation of possible motions. We address this issue by employing log-normal dis-

tribution in the linkage parameter space to generate the data samples. Then, we perform compact clustering of the data using machine learning techniques. A hierarchy is created in the database by means of clustering, where the top level comprises of data points called cluster centers, which are representative of the cluster points in lower levels. Wandling [19] and Yu et al. [20] have built libraries with all possible coupler curves, where one curve is broken down into many segments for creating data for partial curves. In contrast to this, we need to store only one curve that represents all the segments in it. This is because our representation facilitates part-to-whole matching. None of the other previous methods facilitate this partial matching of open motion curve into another open or closed curve, which significantly contributes to providing a large number of solutions, and reduces the data requirement even further.

The synthesis routine starts by creating a continuous parametric representation of a prescribed path or motion. We employ pattern recognition and computational shape analysis to create a similarity transform-independent signature for a parametric path and motion. A query representing invariant signature¹ of the prescribed path or motion is raised for k nearest neighbors among cluster centers in the database. These k neighbors, if needed, are subjected to fine-tuning by local optimization to obtain a set of defect-free solutions. The objective function that drives the synthesis process computes a distance measure of dissimilarity between the task and the coupler motion or path generated by current linkage parameters. This distance measure of dissimilarity inherently requires continuity of motion, thus ensuring that the output mechanism is defect-free throughout the task. Figure 2 illustrates an overview of our method, which is codified in Algorithm 1.

The original contributions in the paper are as follows: 1) a perceptive problem formulation for path and motion generation, that solves the issues associated with precision point approach, 2) we exploit the nonlinear nature of the relationship between the linkage parameters and coupler motions to create a sensitive, wide-ranging, compact, and efficient database with hierarchical clustering, and 3) an algorithm for partial matching of motions and paths which significantly improves the synthesis of open prescribed tasks.

Rest of this paper is organized as follows. Section 2 presents the computation of motion and path signatures. Section 3 is comprised of evaluation criterion for signatures based on the shape similarity, which leads to the formulation of error function for optimization. Section 4 discusses the nature of objective function via sensitivity analysis at a singularity. Section 5 presents the database generation and clustering using auto-encoders for efficient sampling and query operations. Finally, two case studies are presented in section 6 to illustrate the efficiency and efficacy of the method.

¹path or motion signature

Algorithm 1: Planar Linkage Synthesis

Input : Task Motion $\{x_i, y_i, \theta_i\}_{i=1}^N$ or Path $\{x_i, y_i\}_{i=1}^N$
Output: Linkage Parameters l_1, l_2, \dots

```

1 signature = calculateSignature(Input);
2 distances = [];
3 for centerPoint in clusterCenters do
4     distances.push(getDistance(signature,
        centerPoint))
5 end
6 kNeighbors = getNeighbors(distances, k) for
    neighbor in kNeighbors do
7     if threshold < neighbor.distance then
8         return neighbor.LinkParameters
9     else
10        return Optimize(neighbor.LinkParameters)
11    end
12 end

```

2 Signatures of Coupler Path and Motion

Focus of the paper is on a novel method for mechanism synthesis that takes a parametric motion ($x : x(t), y : y(t), \theta : \theta(t)$) or path ($x : x(t), y : y(t)$) as the input, and returns defect-free linkages that produce similar motion or path. The input is transformed into a representation, termed as a signature, which is invariant to similarity operations, viz. reflection, rotation, translation, and scaling. Signatures for path and motion are termed as path signature and motion signature, respectively. For calculating path signature, we use formulations developed by Cui et.al [22].

Consider a motion given in parametric form as, $x : x(t), y : y(t), \theta : \theta(t)$. Where, $\theta(t)$ is the change in orientation along the path with respect to initial orientation. It should be noted that $\theta(t)$ is a continuous curve with domain $(-\infty, \infty)$ in contrast to conventional domain i.e. $[-\pi, \pi]$.

Curvature $\kappa(t)$ of the path ($x : x(t), y : y(t)$) and its integral $K(t)$ is given by,

$$\kappa(t) = \frac{\ddot{y}(t)\dot{x}(t) - \ddot{x}(t)\dot{y}(t)}{(\dot{x}^2(t) + \dot{y}^2(t))^{\frac{3}{2}}}, \quad (1)$$

$$K(t) = \int_0^t |\kappa(t)| dt, \quad (2)$$

where $\dot{x}(t), \ddot{x}(t)$ are first and second order derivative with respect to parameter t . As an example, the parametric motion could be a B-spline motion as shown in Fig 3. We compute $\kappa(t)$ and $K(t)$ along the direction of t , using Eq. (1) and Eq. (2) respectively. Figure 4 shows computed curvature and its unsigned integral of the coupler path shown in Fig. 3. It can be seen that the curvature is small at the start, increases as the curve bends along the path and drops once again as the path straightens out. It is obvious that curvature plot will reverse if the direction of parameterizations reverses, while the integral is a monotonically increasing function.

Now, we re-sample the curvature at equal intervals of $K(t)$, which is equivalent to plotting κ vs K in Fig 5(a). This

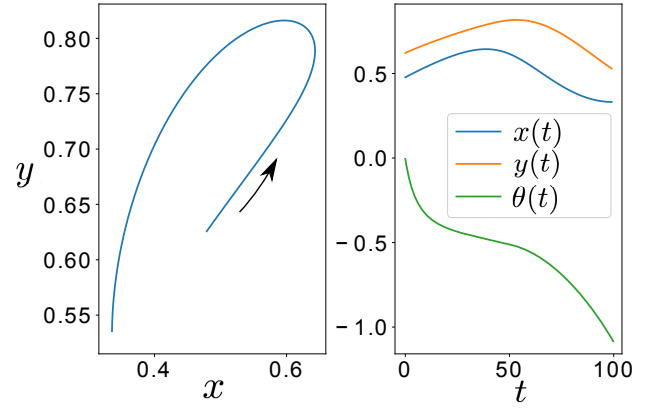


Fig. 3: a) The path of the input motion along with direction of parametrization. b) Motion components $x(t), y(t), \theta(t)$ are plotted against parameter t .

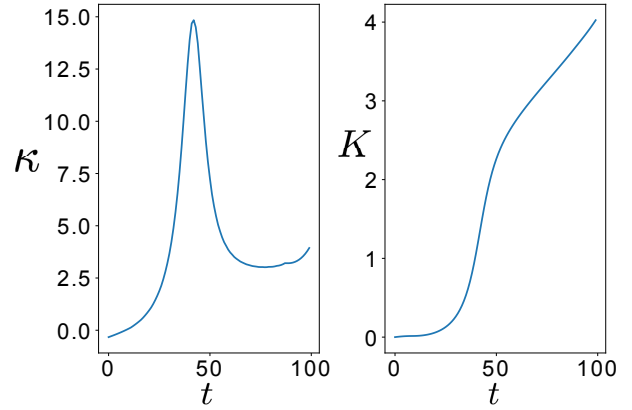


Fig. 4: Curvature and its unsigned integral for the path shown in Fig. 3

is done by finding the parameter values of t where K changes uniformly. For practical purposes, we find an array of the parameter t such that K increments by 0.1. For each value of t in that array, we compute $\kappa(K)$ and $\theta(K)$ and store it as the path and motion signatures respectively. We note that there is a one-to-one mapping between t and K . Although it seems natural to use dual-quaternions for representing motion and finding its signature, it couples orientation with the path in a way, such that the signatures formed no longer remain invariant to similarity transformation.

These signatures are invariant under similarity transformations; for proof see [22]. We know that curvature changes inversely to scale of the curve, so when it is integrated along the scaled curve, the scale factor cancels itself out. Reflection operation produces flipped path signature, but motion signature remains invariant. Figure 5 shows path and motion signature obtained for the motion depicted in Fig. 3. It is important to note, that the signature depends on the direction of parameter t . The procedure of signature calculation presented in this section is given in Algorithm 2.

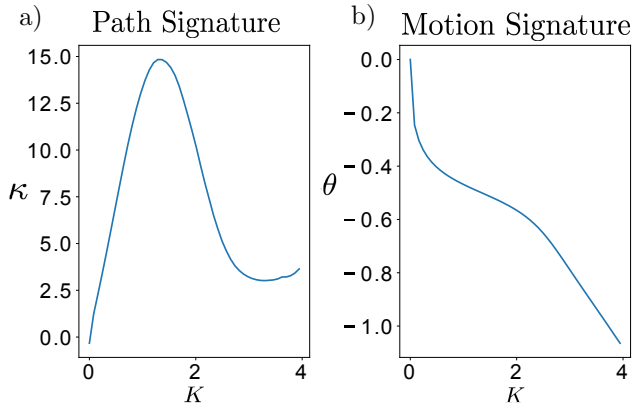


Fig. 5: Path and motion signatures of the motion shown in Fig. 3

Algorithm 2: Calculate Invariant Signatures

Input : Twice Differentiable Parametric Representation of Motion
 $(x : x(t), y : y(t), \theta : \theta(t))$

Output: signature //discretized signal in form of an array

```

1  $\kappa(t) = \text{ComputeCurvature}(x, y)$  using Eq. 1
2  $K(t) = \text{IntergrateCumulatively}(\kappa(t))$  using Eq. 2
3  $\text{motionSignature} = []$ 
4  $\text{pathSignature} = []$ 
5 for  $i = 0 \rightarrow \max(K)$  do
6    $\text{tmp} = (\text{value of } t \text{ corresponding to which } K \text{ has value } i)$ 
7    $i = i + 0.1$ 
8    $\text{motionSignature.push}(\theta(\text{tmp}))$ 
9    $\text{pathSignature.push}(\kappa(\text{tmp}))$ 
10 end
11 return  $\text{PathSignature}, \text{MotionSignature}$ 

```

3 Signature Matching and Error Function

The signatures obtained in previous steps contain important information about the shape of the trajectory. In this section, we formulate functions that evaluate the similarity between two trajectories based on their signatures. These distance functions can be used as error metric, which can be minimized using optimization methods.

3.1 Partial Matching of Path Signatures

When a path query is raised, it can be very useful to know whether this path matches with a part of a path from the database. This subsection presents a method for determining this partial similarity.

Let us consider two coupler paths; namely *Part* and *Whole* as shown in Fig. 6. Let p and W be their signatures respectively, where W completely contains p as shown in Fig. 7. The orientation information shown in Fig. 6 is ignored for path matching. It will be used later for matching of motion signatures. The partial matching works as follows:

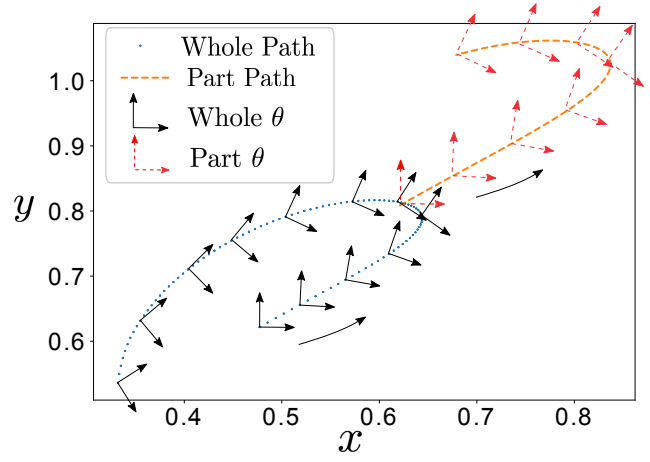


Fig. 6: *Part* path is formed by trimming whole path followed by translation and scaling. Arrows indicate the increasing direction of parameter t .

1. p and W are expressed in terms of arrays and W must contain more points than p .
2. p is slid with offset index j along W .
3. For each offset j , we compute normalized cross-correlation [23] function given by,

$$Cn(j, p, W) = \left| \sum_i^{p_{sp}} \frac{(W(i+j) - \bar{W}(j:j+p_{sp}))(p(i) - \bar{p})}{\sqrt{\sum_i^{p_{sp}} (W(i+j) - \bar{W}_{p_{sp}})^2 \sum_i^{p_{sp}} (p(i) - \bar{p})^2}} \right|, \quad (3)$$

where $Cn(j, p, W)$ is the normalized cross-correlation value when p is matched against W at j^{th} index; p_{sp} is length of the array p and $\bar{W}(j:j+p_{sp})$ is mean of the values of array W between index range of $(j, j+p_{sp})$.

Here, p acts as a template that tries to find the best match against W while sliding over it along j . Domain of $Cn(j, p, W)$ is $[0, 1]$, where 1 represents the complete embedding² of p inside W .

The maximum score of the matching $Cn_{max}(p, W)$ represents similarity of the template in W , and offset index j at which maximum occurs is the start point for matching. As we know that the signature reverses with reversal of the direction of t , we compute the correlation along both directions and select the best matching score, offset index and the matching direction of sampling. Figure 8 depicts normalized cross correlation function over the sliding domain j for *Part* and *Whole* curves.

3.2 Partial Matching of Motion Signatures

This section presents how a template motion can be checked against other motion for potential matching. Consider *Part* and *Whole* motions shown in Fig. 6. Let p and W be the motion signatures of the *Part* and *Whole* motions, respectively as shown in Fig. 9. Similar to partial matching

²*Part* is identical to a portion of *Whole*.

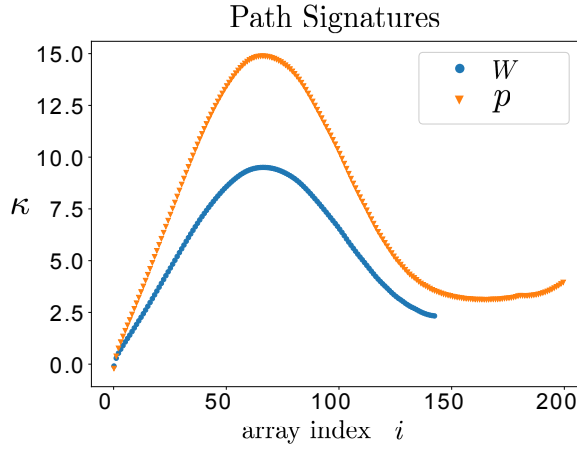


Fig. 7: Path Signatures of part p and whole W from Fig. 6. The array index i corresponds to the index location for the array of the K . The domain of path signature is scale-invariant but the range still has a scaling factor, which is taken care of by normalized cross-correlation.

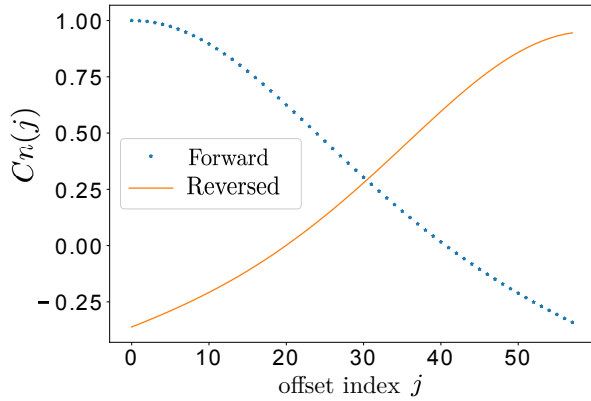


Fig. 8: Normalized cross correlation of the signatures computed along each direction is shown. It can be seen that exact match is found at $j = 0$.

of path signatures, the cross-correlation function is given by,

$$E(j, p, W) = \sum_i^{p_{sp}} ((W(i+j) - \bar{W}(j:j+p_{sp})) - (p(i) - \bar{p}))^2, \quad (4)$$

where $E(j, p, W)$ is the dissimilarity value when template p is matched to W at j^{th} index. Here p tries to find the best match against W while sliding over it. Similar to path signature, motion signature is dependent on the direction of t . Thus, we compute the dissimilarity for both directions and choose whichever is the least i.e. $E_{min}(p, W)$. Figure 10 depicts dissimilarity function over the sliding domain j . In this case, as shown in Fig. 10, we find that the first point is the matching point, which is consistent with the fact that we have essentially sliced the whole motion to obtain the part motion.

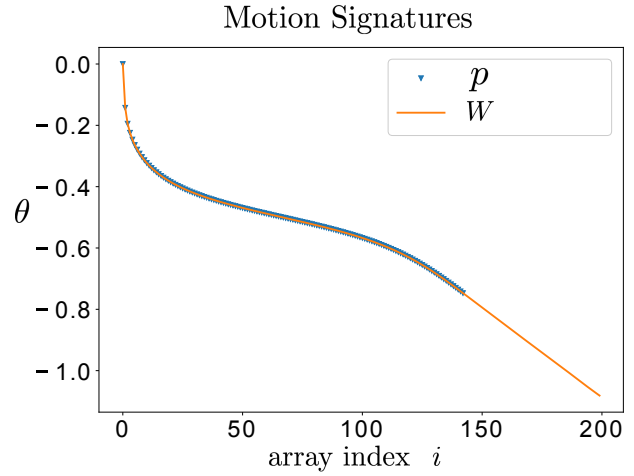


Fig. 9: Motion Signatures of the trajectories shown in Fig. 6. The domain as well as range of motion signature is invariant to similarity transformation.

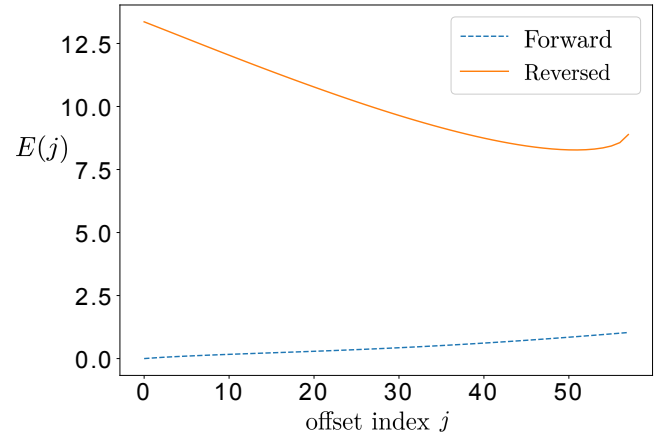


Fig. 10: Dissimilarity function of two motion signatures along both directions. It can be seen that exact match is found at $j = 0$, where the template is fully embedded inside the other motion.

3.3 Objective Function for Synthesis

The functions in Eq. (3) and (4) presented in the sections 3.1, 3.2 can be used as the error measure for path and motion synthesis of any planar linkage, where the objective is to find a linkage that produces a motion whose part or whole corresponds to the target motion (or path). Thus we can formulate the path synthesis problem as,

$$\arg \min_{l, W_i} (1 - Cn_{max}(p, W_i)), \quad (5)$$

where l is the vector of linkage parameters for particular planar linkage, p is signature of task path taken as the template and $\{W_i\}_{i=0}^s$ is the signature set of all s coupler paths generated by the linkage corresponding to l . In case of four-

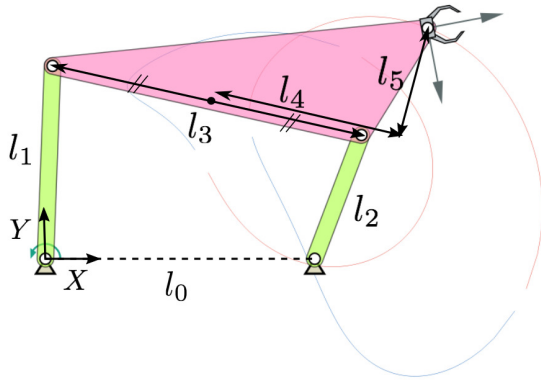


Fig. 11: Parametric representation of four-bar linkage with all revolute joints. We set $l_0 = 1$ and one fixed joint at the origin of the global frame along with making the fixed length of four-bar parallel to the x-axis.

bar, $l: l_1, l_2, l_3, l_4, l_5$, where l_i is link ratio of i^{th} link shown in Fig 11.

Similarly, we can formulate motion synthesis problem as,

$$\arg \min_{l, W_i} (E_{min}(p, W_i)). \quad (6)$$

Here, E is the Dissimilarity function from Eq. (4) while p and $\{W_i\}_{i=0}^s$ are motion signatures instead of path signatures.

Objective function evaluation step consists of calculation of coupler motion or path and finding its dissimilarity score. It is important to note that representation obtained in Section 2 reduces the number of parameters in optimization. This optimization problem can be solved using search methods which do not require gradient computation. We can employ global optimization methods such as differential evolution at the start, and local optimization methods link Powell's method towards the end for faster convergence [13].

Considering highly nonlinear nature of the problem, finding a good initial guess proves to be daunting. Also, as we mentioned in the motivation, it is important to generate a diverse set of solutions. This requires a diverse and large number of good initial guesses. Thus, we exploit machine learning techniques to create a database for finding many good initial guesses or the solution itself. Section 5 presents the details on the approach.

4 Sensitivity Analysis of Signatures

Due to the complex relationship between parameter space and generated motion, small changes in linkage parameters can produce large and discontinuous structural changes in the generated motions. For example, a small change in crank length (l_1) can open a previously closed coupler path. Most of the methods based on Fourier descriptors cannot

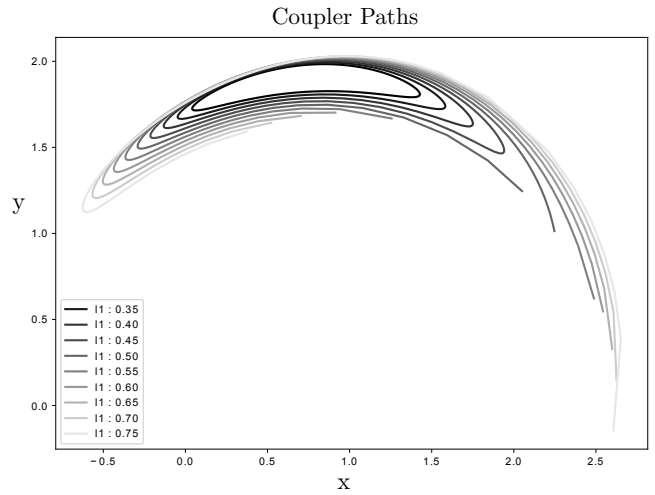


Fig. 12: Coupler Motions of the four-bar linkage with variation of parameters l_1 and l_2 . It can be seen that motion topology changes from close-loop Grashof to open loop Triple-Rocker.

capture the continuity at such singular locations, which adversely affect the optimization process. In contrast to this behavior, the signatures derived in Sec. 2 have a smooth transition at these singular locations due to shape similarity between closed and just opened curve or motion.

To illustrate this via example, we perform sensitivity analysis in the vicinity of a singularity as follows: A four-bar with link ratios ($l_1 : 0.55, l_2 : 1, l_3 : 1.5, l_4 : 1, l_5 : 1$) is subjected to gradual change in parameters l_1 and l_3 by the amount $(-0.2, 0.2)$ in steps of 0.01. The link ratios are chosen such that, small changes in some parameters lead to the topological change in the coupler curve. Error function between motions of new and initial four-bar are calculated using Eq. (6). Figure 12 shows coupler motion of some of the four-bars while Fig. 14 depicts their motion signatures. Please note that these two figures show the effect of changing only one parameter l_1 . It can be seen from Fig. 12 that there exists a discontinuity in the topology of coupler curves even though their shapes have a continuous shift. Our method captures this continuity, which is shown by error function evaluations depicted in Fig. 13, where it is visible that surface is well behaved in the singularity region. This error function accounts for changes to both the parameters l_1 and l_3 .

5 Clustered Database of Planar Linkages

Having an invariant representation facilitating partial matching greatly reduces data required to sample all possible types of shapes of coupler motion. We have built a database of planar four-bar linkages with revolute joints as an example, but the approach is same for any planar motion generating mechanism. We generate this database comprising of 40,000 linkages while taking following aspects into consideration.

1. Sampling should maximize the uniformity of its distribution over the space of four-bar coupler motions.

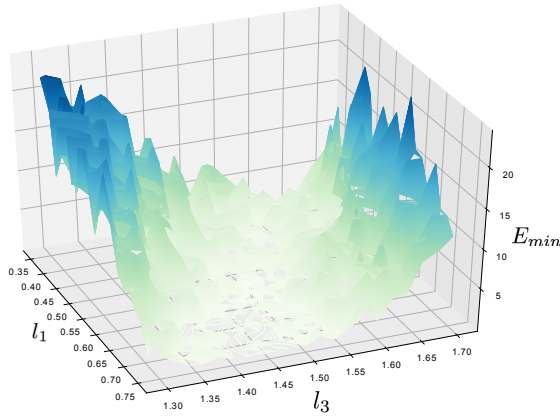


Fig. 13: Distance (E_{min}) from Eq. (4) as the parameters l_1 and l_3 are varied. Although open loop breaks at $l_1:0.55$, $l_3:1.5$, there are no spikes of error function in the region near singularity, as the shape is very similar between the two topologies.

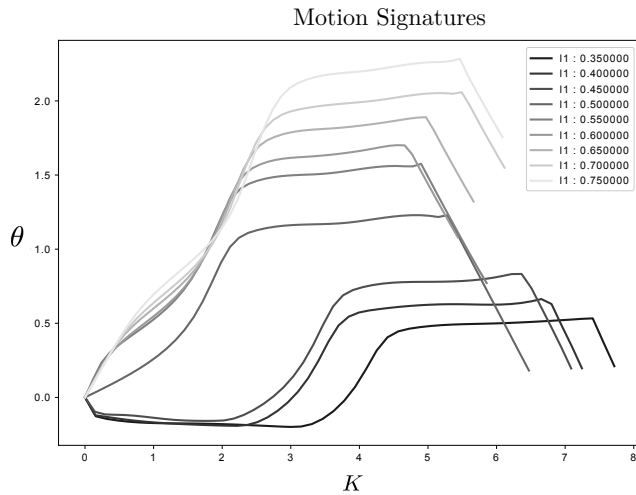


Fig. 14: Motion signatures obtained by steps given in 2. Although topology difference is even more evident in this representation, it also signifies the similarity pattern between them.

2. Data generation should be parallelized.
3. It should be scalable to higher-order linkages.

Figure 11 represents parametric representation of four-bar linkage with parameters $(l_1, l_2, l_3, l_4, l_5)$. As mapping between four-bar linkage parameter space and coupler motion space is highly nonlinear, uniform distribution over linkage parameter space does not necessarily mean uniform sampling over coupler motion space. Thus, an efficient approach would be to sample more in the regions where sensitivity is maximum. We have observed that whenever the link ratios of four-bar linkage are close to 1, the sensitivity of shape of a coupler motion is higher than otherwise. Thus, we have chosen Log Normal probability distribution ($\mu = 0, \sigma = 0.6$) for selecting the link ratios : (l_1, l_2, l_3) as shown in Fig. 15,

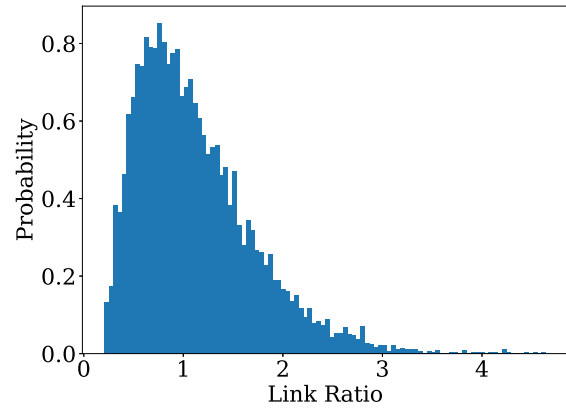


Fig. 15: Probability Distribution function used in random sampling for parameters l_1, l_2 and l_3 .

and Normal Distribution ($\mu = 0, \sigma = 2$) for (l_4, l_5) .

We use machine learning techniques such as clustering and data-compression using auto-encoder neural networks to come up with good initial guesses for local optimization. First, We cluster the database using hierarchical clustering algorithm. Then, we find representative data point in each cluster called cluster center and form their set. This set of cluster centers represent a diverse group of linkages. When a query is raised in the database, the first step is to search for neighbors in the set of cluster centers. This often yields a diverse set of neighbors and is used as the set of independent initial guesses for local optimization.

5.1 Dimensionality Reduction using Auto-Encoders

Each data point in the database consists of discrete signature, which is kept to be of 100 float digits. In order to have efficient query operations, we perform hierarchical Clustering; a method that summarizes and creates a hierarchy in the database. Clustering in higher dimensions suffers from *Curse of Dimensionality* [24], thus we first perform dimensionality reduction using Auto-encoder Neural Networks. Auto-encoder is a powerful mapping model, which learns to encode the input data in very compact representation and can reconstruct the input with minimal error; performing much better than Principal Component Analysis [25]. This nonlinear mapping by auto-encoder can greatly improve the representation of data for clustering [26]. Figure 16 shows a Neural Network architecture similar to the one we designed for the task. Our architecture consists of 100 neurons in input and output layer, while five hidden layers have (80, 50, 10, 50, 80) neurons respectively. Each neuron in the hidden layer is activated by Rectified Linear Unit (*ReLU*) activation function. In i^{th} hidden layer, $d^{(i-1)}$ dimensional vector output of the previous layer $h(i-1)$ is fed as input to produce $d^{(i)}$ dimensional output h_i . Input-output relationship of a layer is given by,

$$h_i = \text{ReLU}(W_i h_{i-1} + b_i), \quad (7)$$

$$\text{ReLU}(x) = \max(0, x), \quad (8)$$

where W_i is weight matrix with dimensions $(d^i, d^{(i-1)})$ and b_i is $d^{(i)}$ dimensional bias vector of i^{th} layer, which are computed in the process of training. Auto-encoders are trained to reconstruct the input. In this way each layer encodes the input, which is sufficient for the next layers to reconstruct the output. The objective of training is to find out the set of weights and biases that minimizes the error loss given by,

$$\arg \min_{W, b} \sum_{i=0}^N \|X_i - \tilde{X}_i\|^2, \quad (9)$$

where X_i is input, \tilde{X}_i is reconstructed output and N is number of training examples.

Once a network is trained, the output of the bottleneck layer (h_{ib}) represents the compressed feature space (Z). As bottleneck layer has 10 neurons and input is a 100-dimensional vector, it is evident that information is compressed by a factor of 10, while achieving 95% reconstruction accuracy as the result of training. Standard clustering algorithms are performed on this latent³ space for better clustering [26]. We use Agglomerative Clustering, a method of hierarchical clustering, which is an approach to partitioning clustering for identifying groups in the dataset. Ward [27] criterion is used for clustering, which minimizes the variance of the clusters being merged. The distance metric used for clustering is the Euclidean distance in the latent space. Although the more accurate distance metric is distance function discussed in sec. 3, it is very expensive to calculate it for the entire database. Signatures with $O(m)$ points take $O(m \log m)$ time for each comparison and there are $O(N^2)$ number of comparisons to be made for the database of N points.

Now, when the user raises a query, we use the distance function from sec. 3 for finding k nearest neighbors among 1500 cluster centers. If a cluster center is not sufficiently close, we descend into its corresponding cluster to find the closest data point. Motion with highest similarity score is returned along with its corresponding linkage parameters. If required, the parameters are fine-tuned to match the query using local optimization methods. Computationally, on a 2.4 GHz Core i5 MacBook Pro with 8 GB memory, every query takes 23 seconds on average to find the sorted list of nearest neighbors among cluster centers.

6 Case Studies

This section presents two case studies presenting the effectiveness of our approach for path and motion synthesis applications.

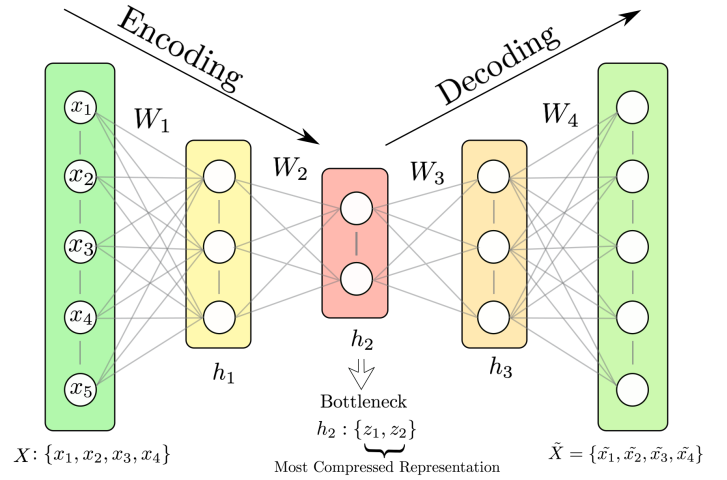


Fig. 16: A small-scale version of the Auto-Encoder. This network takes 5-dimensional input in the input layer. At each encoder layer, the input is compressed into a vector of lesser dimensions, lowest at the bottleneck layer.

6.1 Path Generation

In the design phase of a rehabilitation device that assists people to stand from sitting position, it is required to generate linkages that can execute a sit-to-stand (STS) trajectory of the hip joint as shown in Fig. 17. Table 1 presents the discretized path data. As our approach requires parametric representation of path, we first fit a cubic B-Spline with cord length parametrization through path data points to generate parametric curve shown in Fig. 17. We compute its path signature by the steps mentioned in Algorithm 2 and raise the query for nearest neighbors among 1500 cluster centers of our database. The distance metric for finding neighbors among cluster centers is $1 - Cn_{max}$ in Eq. (5). Table 2 tabulates the link ratios corresponding to obtained nine nearest neighbors. Next step is to compute actual parameters according to position, scale, and orientation of the path. It is done by comparing analogous points found by offset index j in Eq. 3. Figure 18 shows first eight four-bar mechanisms corresponding to nearest signatures to path signature of input. It can be clearly seen that these linkages generate highly accurate paths for the sit to stand activity. It is important to note that every solution is a result of partial matching of coupler paths, and otherwise would be very hard to search using other atlas-based approaches that only have the whole-to-whole matching facility.

6.2 Motion Generation

The task is to find a pool of linkage systems that can perform snow-shoveling with a motion shown in Fig. 19. The motion data is tabulated in Table 3 to which we fit a B-spline with cord length parametrization in order to get the parametric representation of motion. The task can also be treated as

³compressed output of bottleneck layer.

Table 1: Case Study 1 : Path Data

Point	x	y	Point	x	y
1	-7.81	-9.65	10	0.28	-1.31
2	-6.42	-9.81	11	0.64	1.07
3	-5.14	-9.62	12	0.98	2.73
4	-3.72	-8.99	13	1.47	4.30
5	-2.62	-8.14	14	2.73	6.58
6	-1.75	-7.13	15	3.46	7.41
7	-0.91	-5.67	16	4.07	7.95
8	-0.32	-4.10	17	4.70	8.41
9	-0.02	-2.92	18	5.32	8.76

Table 2: Linkage Parameters of Nine Nearest Neighbor Paths

linkage	l_1	l_2	l_3	l_4	l_5	$1 - Cn_{max}$
1	0.79	2.78	1.70	1.35	-0.71	0.0011
2	1.59	1.28	0.96	-1.74	-1.13	0.0015
3	0.99	0.71	1.66	-1.07	-0.85	0.0016
4	0.51	0.48	1.11	-0.02	-0.15	0.0017
5	0.93	0.75	2.18	-1.65	0.96	0.0018
6	1.29	1.98	1.02	-1.83	-1.33	0.0019
7	0.63	1.42	1.03	-1.90	-0.38	0.0020
8	0.66	0.85	0.84	-1.40	-0.13	0.0021
9	1.81	0.55	1.08	0.14	-0.04	0.0022

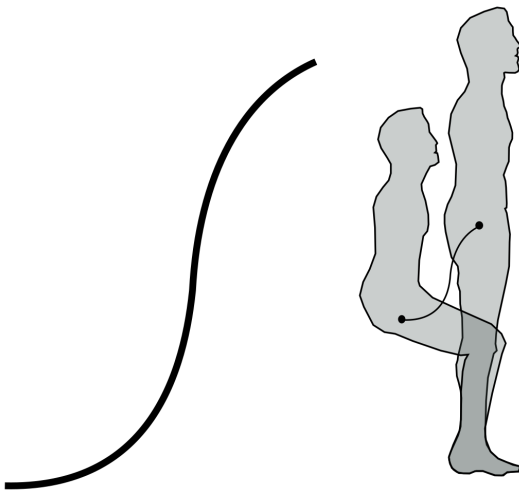


Fig. 17: Case Study 1: Path traced by hip joint during Sit-to-Stand Motion.

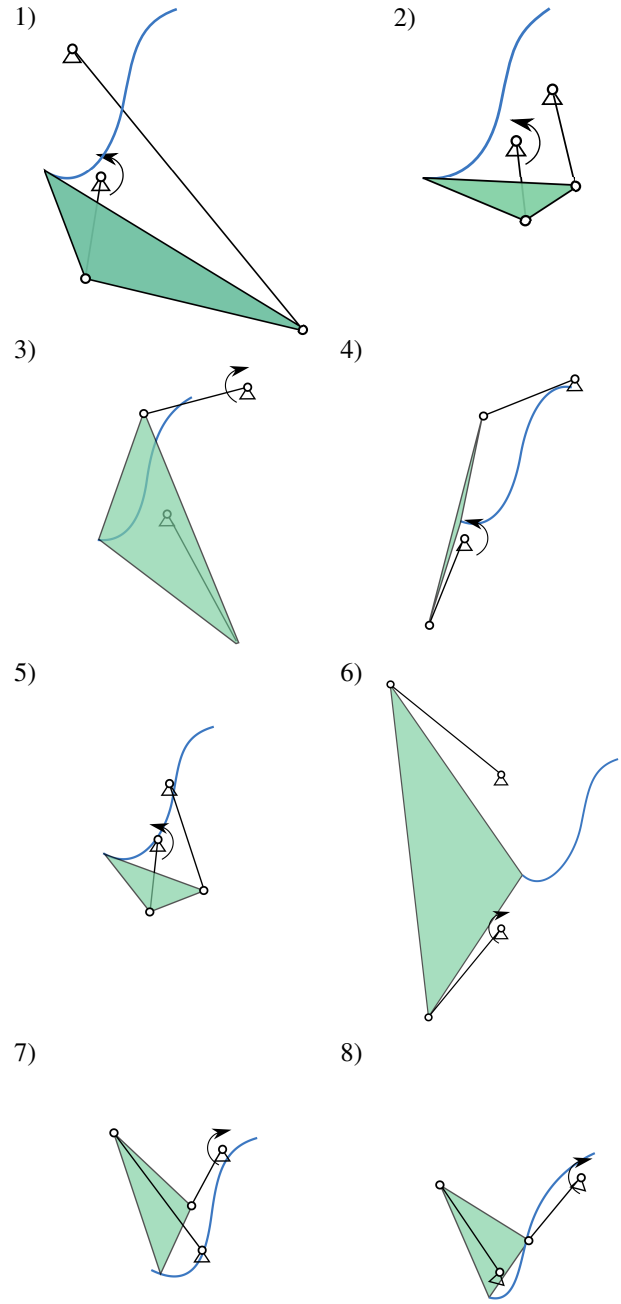


Fig. 18: First eight linkages in the table 2 and their resultant coupler paths.

a finite position motion generation and solved for valid solutions. We try with our real-time computational methods of algebraic fitting [28], [29] but obtained solutions suffer from circuit defect, which is not surprising as those methods do not account for the continuity of input positions. Also, It is obvious that the coupler should not drop the snow during its entire motion except at the end. Although, the prescribed motion entails this information, precision point approach cannot capture it.

Now we employ the approach presented in this paper.

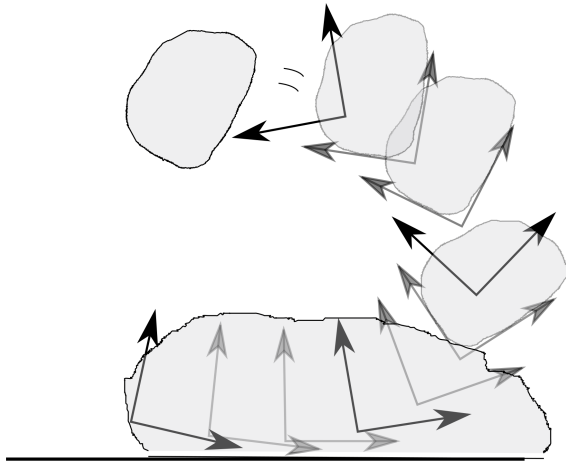


Fig. 19: Case Study 2: User specified motion necessary for the snow shoveling task.

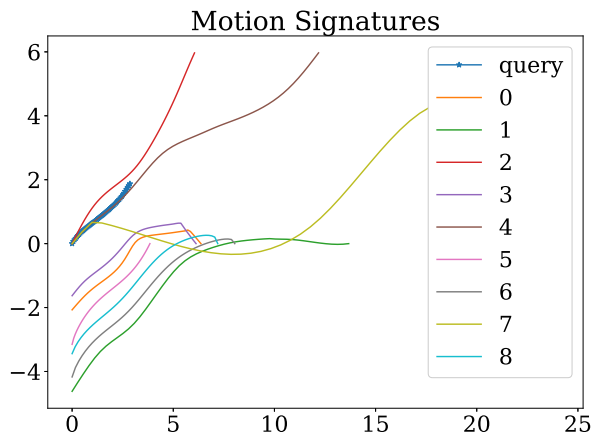


Fig. 20: Case Study 2 - Query Result: Motion signatures in the dataset with highest similarity.

The first step is to calculate motion signature of the task motion using steps mentioned in Algorithm 2. For that we follow the steps given in section 2 to obtain the motion signature depicted in Fig. 20. Next, we raise the signature query for nearest signatures among cluster centers of the database. Fig. 20 shows nine nearest neighbor signatures along with the task signature. Table 4 presents the linkage parameters corresponding to the nearest neighbors along with their distance score from the task. Coupler motions of these linkages have a part, which matches with the shape of input motion query. Actual scaling and orientation of the linkage can be found out easily by comparing analogous points, which are given by the offset index j that corresponds to minimum distance (E_{min}) in Eq. (4). Figure 21 depict the solutions obtained after scaling and orienting the linkage to match required motion. All of these linkages satisfactorily perform the input task without any defect. As ground or fixed pivot

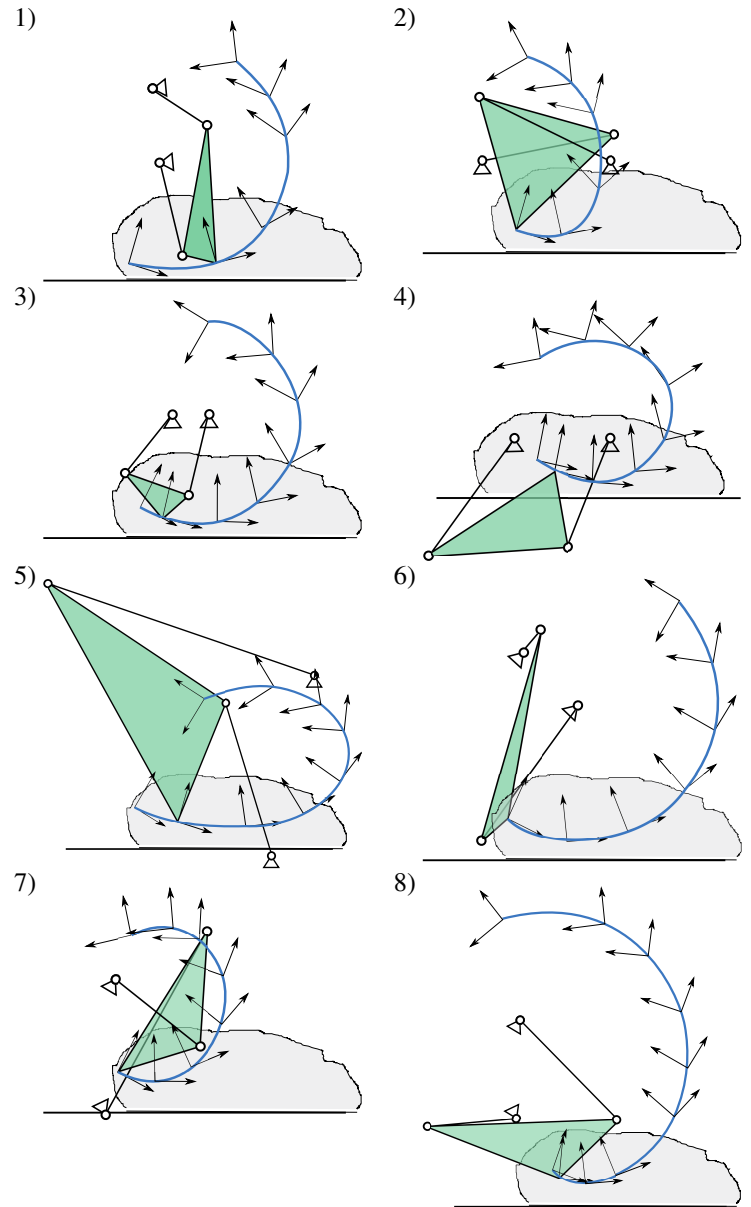


Fig. 21: Case Study 2: First eight linkages in the table 4 and their resultant coupler motions.

locations should lie above the ground, all solutions except the 4th solution are suitable for the task. Although Fig. 21 shows that 5th and 7th solution may slightly interfere with the ground, it can be rectified by lifting the mechanism slightly up or making small changes in coupler dimensions. In light of these results, we can say that this approach produces a large variety of solutions, which otherwise would be very hard to find using precision point approach.

Conclusion

The methods based on precision point approach do not capture continuity of the task. This causes the solutions to have the branch, circuit and order defects. Also, the formulation fails to detect undesired properties of the coupler

Table 3: Case Study 2: Pose Data

Pose	x	y	θ	Pose	x	y	θ
1	0.03	0.07	6.06	6	1.39	0.47	0.73
2	0.38	0.01	6.18	7	1.36	0.77	1.03
3	0.71	-0.00	0.04	8	1.18	1.06	1.36
4	1.02	0.06	0.22	9	0.88	1.27	1.74
5	1.26	0.22	0.46				

Table 4: Case Study 2: Linkage Parameters corresponding to Nine Nearest Neighbor Motions

linkage	l_1	l_2	l_3	l_4	l_5	E_{min}
1	1.28	0.88	1.77	-1.32	1.79	0.0186
2	1.05	1.14	1.09	0.35	-0.60	0.0362
3	2.06	2.28	1.84	-1.71	0.51	0.0378
4	1.52	1.22	1.46	0.05	0.12	0.0402
5	1.12	0.99	0.57	0.05	0.68	0.0467
6	1.58	0.88	1.17	0.08	-0.22	0.0481
7	2.17	0.38	2.87	-3.51	1.39	0.0578
8	1.55	0.79	0.85	-0.80	-0.52	0.0585
9	0.91	1.40	1.93	-0.93	-0.83	0.0605

motion in the region between precision points. Thus, we present a perceptive problem formulation, by considering the entire prescribed task. We solve the formulation by employing machine-learning techniques and generate a large number of defect-free conceptual designs. The approach is highly data-efficient due to similarity invariant representation and partial matching. Sensitivity analysis indicates that complexity of the objective function is well behaved at the singular locations. The hierarchically clustered database provides efficient query search. Finally, the effectiveness of the presented approach is showcased by two case studies. Every solution presented in the examples section is a result of part-to-whole matching. The other atlas-based approaches facilitate only whole-to-whole matching, hence they would need a very large amount of data to find these results.

Although the partial matching metric is more accurate, it is expensive in terms of computation cost. Thus, we use Euclidean metric in the latent space of compressed data for hierarchical clustering of the database. The problem formulation is invariant with respect to translation, orientation, and scaling. Hence, constraints like geometric restrictions on pivots have to be addressed after finding feasible solutions for the task. The approach is general enough to be extended to higher order linkage systems for which there are even fewer methods available for synthesizing defect-free solutions. However, the database size increases exponen-

tially with the number of links in the mechanisms. A potential solution to this problem could be to use learning based methods, where the pattern is learned instead of storing all of the information. Overall, the method provides a holistic approach towards the prescribed path and motion synthesis and encourages artificial intelligence techniques to make an impact.

Acknowledgements

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

References

- [1] McCarthy, J. M. and Soh, G. S., 2010, Geometric design of linkages, volume 11, Springer.
- [2] Sandor, G. N. and Erdman, A. G., 1997, Advanced Mechanism Design: Analysis and Synthesis Vol. 2, Prentice-Hall, Englewood Cliffs, NJ.
- [3] Hunt, K., 1978, Kinematic Geometry of Mechanisms, Clarendon Press, Oxford.
- [4] Hartenberg, R. S. and Denavit, J., 1964, Kinematic Synthesis of Linkages, McGraw-Hill, New York.
- [5] Suh, C. H. and Radcliffe, C. W., 1978, Kinematics and Mechanism Design, John Wiley and Sons, New York.
- [6] Lohse, P., 2013, Getriebesynthese: Bewegungsabläufe ebener Koppelmechanismen, Springer-Verlag.
- [7] Chase, T. and Mirth, J., 1993, "Circuits and Branches of Single-Degree-of-Freedom Planar Linkages.", ASME Journal of Mechanical Design, **115**(2), p. 223230.
- [8] Burmester, L., 1886, Lehrbuch der Kinematik, Verlag Von Arthur Felix, Leipzig, Germany.
- [9] Cabrera, J. A., Simon, A., and Prado, M., 2002, "Optimal synthesis of mechanisms with genetic algorithms", Mechanism and Machine Theory, **37**(10), pp. 1165–1177.
- [10] Sardashti, A., Daniali, H. M., and Varedi, S. M., 2013, "Optimal free-defect synthesis of four-bar linkage with joint clearance using PSO algorithm", Meccanica, **48**(7), pp. 1681–1693.
- [11] Ebrahimi, S. and Payvandy, P., 2015, "Efficient constrained synthesis of path generating four-bar mechanisms based on the heuristic optimization algorithms", Mechanism and Machine Theory, **85**, pp. 189–204.
- [12] Bulatovic, R. R., Dordevic, S. R., and Dordevic, V. S., 2013, "Cuckoo Search algorithm: A metaheuristic approach to solving the problem of optimum synthesis of a six-bar double dwell linkage", Mechanism and Machine Theory, **61**, pp. 1–13.
- [13] Ullah, I. and Kota, S., 1997, "Optimal synthesis of mechanisms for path generation using Fourier descriptors and global search methods", Journal of Mechanical Design, **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", *Journal of Mechanisms and Robotics-Transactions of the Asme*, **3**(3).
- [15] Li, X., Wu, J., and Ge, Q. J., 2016, "A Fourier Descriptor-Based Approach to Design Space Decomposition for Planar Motion Approximation", *Journal of Mechanisms and Robotics*, **8**(6), pp. 064501–064501–5, 10.1115/1.4033528.
- [16] Buskiewicz, J., Starosta, R., and Walczak, T., 2009, "On the application of the curve curvature in path synthesis", *Mechanism and Machine Theory*, **44**(6), pp. 1223–1239.
- [17] Khan, N., Ullah, I., and Al-Grafi, M., 2015, "Dimensional synthesis of mechanical linkages using artificial neural networks and Fourier descriptors", *Mechanical Sciences*, **6**(1), pp. 29–34.
- [18] Mcgarva, J. R., 1994, "Rapid Search and Selection of Path Generating Mechanisms from a Library", *Mechanism and Machine Theory*, **29**(2), pp. 223–235.
- [19] Wandling Sr, G. R., 2000, "Synthesis of mechanisms for function, path, and motion generation using invariant characterization, storage and search methods", Ph. d. thesis.
- [20] Yue, C., Su, H.-J., and Ge, Q., 2011, "Path Generator via the Type-P Fourier Descriptor for Open Curves", *Proceedings of 13th World Congress in Mechanism and Machine Science*.
- [21] 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).
- [22] Cui, M., Femiani, J., Hu, J., Wonka, P., and Razdan, A., 2009, "Curve matching for open 2D curves", *Pattern Recognition Letters*, **30**(1), pp. 1–10.
- [23] Lewis, J. P., 1995, Fast normalized cross-correlation, volume 10 of *Vision interface*.
- [24] Marimont, R. B. and Shapiro, M. B., 1979, "Nearest Neighbor Searches and the Curse of Dimensionality", *Journal of the Institute of Mathematics and Its Applications*, **24**(1), pp. 59–70.
- [25] Hinton, G. E. and Salakhutdinov, R. R., 2006, "Reducing the dimensionality of data with neural networks", *Science*, **313**(5786), pp. 504–507.
- [26] Song, C., Liu, F., Huang, Y., Wang, L., and Tan, T., "Auto-encoder based data clustering", *Iberoamerican Congress on Pattern Recognition*, Springer, pp. 117–124.
- [27] Ward, J. H., 1963, "Hierarchical Grouping to Optimize an Objective Function", *Journal of the American Statistical Association*, **58**(301), pp. 236–244.
- [28] 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 Journal of Computing and Information Science in Engineering*, 10.1115/1.4035528.
- [29] 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*, 10.1115/1.4037801.

List of Tables

1	Case Study 1 : Path Data	10
2	Linkage Parameters of Nine Nearest Neighbor Paths	10
3	Case Study 2: Pose Data	12
4	Case Study 2: Linkage Parameters corresponding to Nine Nearest Neighbor Motions	12

List of Figures

1	The mechanism synthesis problem for the prescribed motion. The four-bar mechanism obtained using the precision position approach suffers from circuit defect, as no coupler circuit passes through all precision positions.	1
2	The Machine Learning approach begins by creating an invariant signature for the path and the motion data, which facilitates a compact and hierarchical clustered database and an auto-encoder Neural Network trained to elicit good, defect-free solutions or subjected to local, fast optimization. The results are astonishingly good, defect-free concept design solutions for input problems.	2
3	a) The path of the input motion along with direction of parametrization. b) Motion components $x(t), y(t), \theta(t)$ are plotted against parameter t	4
4	Curvature and its unsigned integral for the path shown in Fig. 3	4
5	Path and motion signatures of the motion shown in Fig. 3	5
6	Part path is formed by trimming whole path followed by translation and scaling. Arrows indicate the increasing direction of parameter t	5
7	Path Signatures of part p and whole W from Fig. 6. The array index i corresponds to the index location for the array of the K . The domain of path signature is scale-invariant but the range still has a scaling factor, which is taken care of by normalized cross-correlation.	6
8	Normalized cross correlation of the signatures computed along each direction is shown. It can be seen that exact match is found at $j = 0$	6
9	Motion Signatures of the trajectories shown in Fig. 6. The domain as well as range of motion signature is invariant to similarity transformation.	6
10	Dissimilarity function of two motion signatures along both directions. It can be seen that exact match is found at $j = 0$, where the template is fully embedded inside the other motion.	6

11	Parametric representation of four-bar linkage with all revolute joints. We set $l_0 = 1$ and one fixed joint at the origin of the global frame along with making the fixed length of four-bar parallel to the x-axis.	7
12	Coupler Motions of the four-bar linkage with variation of parameters l_1 and l_2 . It can be seen that motion topology changes from close-loop Grashof to open loop Triple-Rocker.	7
13	Distance (E_{min}) from Eq. (4) as the parameters l_1 and l_3 are varied. Although open loop breaks at $l_1:0.55, l_3:1.5$, there are no spikes of error function in the region near singularity, as the shape is very similar between the two topologies.	8
14	Motion signatures obtained by steps given in 2. Although topology difference is even more evident in this representation, it also signifies the similarity pattern between them.	8
15	Probability Distribution function used in random sampling for parameters l_1, l_2 and l_3	8
16	A small-scale version of the Auto-Encoder. This network takes 5-dimensional input in the input layer. At each encoder layer, the input is compressed into a vector of lesser dimensions, lowest at the bottleneck layer.	9
17	Case Study 1: Path traced by hip joint during Sit-to-Stand Motion.	10
18	First eight linkages in the table 2 and their resultant coupler paths.	10
19	Case Study 2: User specified motion necessary for the snow shoveling task.	11
20	Case Study 2 - Query Result: Motion signatures in the dataset with highest similarity.	11
21	Case Study 2: First eight linkages in the table 4 and their resultant coupler motions.	11