Analytical and Machine Learning based Frameworks for Synthesis of Planar, Spherical and Spatial Mechanisms

A Dissertation Presented

by

Shashank Sharma

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the degree of

Doctor of Philosophy

in

Mechanical Engineering

Stony Brook University

August 2020

Stony brook University

The Graduate School

Shashank Sharma

We, the dissertation committee for the above candidate for the

Doctor of Philosophy degree, hereby recommend

acceptance of this dissertation.

Dr. Anurag Purwar - Dissertation Advisor
Professor, Mechanical Engineering

Dr. Qiaode Jeffrey Ge - Chairperson of Defense
Professor, Mechanical Engineering

Dr. Nilanjan Chakraborty
Professor, Mechanical Engineering

Dr. Jahangir Rastegar
Professor, Mechanical Engineering

Dr. Xianfeng David Gu
Professor, Computer Science

This dissertation is accepted by the Graduate School

Eric Wertheimer

Dean of the Graduate School

Abstract of the Dissertation

Analytical and Machine Learning based Frameworks for Synthesis of Planar, Spherical and Spatial Mechanisms

by

Shashank Sharma

Doctor of Philosophy

in

Mechanical Engineering

Stony Brook University

2020

During the product design phase, an engineer uses kinematic synthesis to create multiple concepts. Conventionally, kinematic synthesis has been studied by researchers for the purpose of synthesizing a pure motion or path. However, practical design problems are usually messy and constitute a mix of motion and path constraints. Also, these methods have been limited to synthesizing only one type of mechanism among planar, spherical, and spatial mechanisms.This research attempt to unify path and motion synthesis within a single general framework and enhance the concept generation process by producing planar, spherical, and spatial mechanisms.

In this work, both analytical and machine learning techniques are explored to solve the Alt-Burmester problem for planar and spatial mechanisms. Also, new algorithms have been proposed for unifying the analysis and motion synthesis of planar, spherical, and spatial mechanisms.

Dedication

I dedicate this dissertation to the loving memory of Amma and Bauji, my late grandparents. They introduced me to the scientific way of thinking and taught me to persevere in the face of adversary.

Acknowledgements

I would like to thank my parents, Mr. Shashi Kant and Mrs. Pammi Dixit, for always being there for me through thick and thin. I am incredibly grateful for the ethics and values instilled in me by them. Their unwavering trust in my abilities has motivated me to push my limits.

I am grateful to Professor Anurag Purwar, my advisor, for providing me the immense opportunity to work with him. He has always given me the freedom to shape my path and encouraged me to focus on research. His support and guidance have made this work possible.

I express my gratitude to Professor Q. Jeffrey Ge, Professor Nilanjan Chakraborty, Professor Jahangir Rastegar, and Professor X. David Gu for being a part of my dissertation committee.

This work has been financially supported by The 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 document are those of the authors and do not represent those of the funding agencies.

Contents

Tables

Figures

Figure

Journal Publications

(1) Sharma S., Purwar A. “A Machine Learning Approach to Solving the Alt-Burmester Problem for Synthesis of Defect-free Spatial Mechanisms”, Manuscript under preparation

(2) Sharma S., Purwar A. “Unified Design of Spatial, Spherical and Planar Mechanisms for the Motion Synthesis Problem”, ASME. J. Mechanisms Robotics. JMR-20-1314, Under Review

(3) Sharma, S., and Purwar, A. (May 26, 2020). “Using a Point-Line-Plane Representation for Unified Simulation of Planar and Spherical Mechanisms.” ASME. J. Comput. Inf. Sci. Eng. December 2020; 20(6): 061002. doi: 10.1115/1.4046817

(4) Sharma, S., Purwar, A., and Jeffrey Ge, Q. (December 17, 2018). “A Motion Synthesis Approach to Solving Alt-Burmester Problem by Exploiting Fourier Descriptor Relationship Between Path and Orientation Data.” ASME. J. Mechanisms Robotics. February 2019; 11(1): 011016. doi: 10.1115/1.4042054

(5) Sharma, S., Purwar, A., and Jeffrey Ge, Q. (October 18, 2018). “An Optimal Parametrization Scheme for Path Generation Using Fourier Descriptors for Four-Bar Mechanism Synthesis.” ASME. J. Comput. Inf. Sci. Eng. March 2019; 19(1): 014501. doi: 10.1115/1.4041566

Conference Publications

(1) Sharma, Shashank, and Purwar, Anurag. "Path Synthesis of Defect-free Spatial 5-SS Mechanisms using Machine Learning", Proceedings of the ASME 2020 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, IDETC2020-22731.

(2) Sharma, Shashank, and Purwar, Anurag. "Unified Motion Synthesis of Spatial Seven-bar Platform Mechanisms and Planar Four-bar Mechanisms", Proceedings of the ASME 2020 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, IDETC2020-22718.

(3) Sharma, Shashank, and Purwar, Anurag. "Using a Point-Line-Plane Representation for Unified Simulation of Planar and Spherical Mechanisms." Proceedings of the ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Volume 5A: 43rd Mechanisms and Robotics Conference. Anaheim, California, USA. August 18–21, 2019. V05AT07A030. ASME. doi: 10.1115/DETC2019-98194

(4) Sharma, Shashank, Purwar, Anurag, and Ge, Q. Jeffrey. "A Motion Synthesis Approach to Solving Alt-Burmester Problems by Exploiting Fourier Descriptor Relationship Between Path and Orientation Data." Proceedings of the ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Volume 5B: 42nd Mechanisms

and Robotics Conference. Quebec City, Quebec, Canada. August 26–29, 2018. V05BT07A077. ASME. doi: 10.1115/DETC2018-85567

(5) Sharma, Shashank, Purwar, Anurag, and Ge, Q. Jeffrey. “Optimal Non-Uniform Parametrization for Fourier Descriptor Based Path Synthesis of Four Bar Mechanisms.” Proceedings of the ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Volume 5A: 42nd Mechanisms and Robotics Conference. Quebec City, Quebec, Canada. August 26–29, 2018. V05AT07A033. ASME. doi: 10.1115/DETC2018-85568

Chapter 1

Introduction

Maker culture is a new trend built upon DIY and hacker movement which encourages the creation of new devices and tinkering with the old. It spans electronics, robotics, 3-D printing, metalworking, and woodworking. The rise of the maker culture is closely associated with the rise of hackerspaces, Fab Labs, and other "makerspaces" where like-minded individuals share ideas, skills, and tools. University campuses have especially been a hotbed for makerspaces. With the rise of cheap 3-D printing technologies and low-cost sensors, actuators, and micro-controllers, manufacturing tools have become accessible to many.

Thus, with the recent maker-movement and democratization of manufacturing capability, the creation of new devices is within everyone's grasp. However, there is still one missing piece. There is a lack of machine design tools for mechanism synthesis which can be used by creators. Unfortunately, design theory for even the simplest of mechanism i.e. a four-bar is too complex for the uninitiated. This chasm needs to be filled for real innovation to thrive.

During concept generation, a designer breaks down a task into multiple constraints and criteria. Mechanisms satisfying the constraints are deemed feasible and the best mechanism is chosen based on the specified criteria. The type of constraints a designer can impose for a mechanism design problem are extremely diverse. The constraints could include the motion of a moving link, path traced by a point on a moving link, relationship between the input and output links, position of grounded joints, lengths of moving links, etc. Also, the space of solution conceptual designs can involve planar, spherical, and spatial mechanisms.

This dissertation aims to improve the state-of-art in the field of kinematic mechanism analysis and synthesis. New path and motion synthesis algorithms have been proposed which can generate a diverse and more accurate family of concepts. We also generalize the path and motion problems as the Alt-Burmester problem and propose analytical and machine learning based algorithms to synthesize planar and spatial mechanisms respectively. A real-time unified approach to simulate planar, spherical, and spatial mechanisms with a variety of joint types has also been proposed.

Chapter 2 presents an improved approach for path synthesis of planar mechanisms. Fourier descriptor based path synthesis algorithms for generation of planar four-bar mechanisms require assigning time parameter values to the given points along the path. An improper selection of time parameters leads to poor fitting of the given path and sub-optimal four-bar mechanisms while also ignoring a host of mechanisms that could be potentially generated otherwise. A common approach taken is to use uniform time parameter values, which does not take into account the unique harmonic properties of the coupler path. In this chapter, we are presenting a non-uniform parametrization scheme in conjunction with an objective function that provides a better fit, leverages the harmonics of the four-bar coupler, and allows imposing additional user-specified constraints.

In Chapter 3, an analytical algorithm is proposed to solve a new class of problems, called Alt-Burmester problems, for planar mechanisms. This chapter presents a generalized framework to solve m-pose, n-path-points mixed synthesis problems, known as the Alt-Burmester problems, using a task-driven motion synthesis approach. We aim to unify the path and motion synthesis problems into an approximate mixed synthesis framework. Fourier descriptors are used to establish a closed-form relationship between the path and orientation data. This relationship is then exploited to formulate mixed synthesis problems into pure motion synthesis ones. We use an efficient algebraic fitting based motion synthesis algorithm to enable simultaneous type and dimensional synthesis of planar four-bar linkages.

Chapter 4 proposes a framework for unified analysis of planar, spherical and spatial mechanisms. This chapter presents a geometric constraints driven approach to unified kinematic simulation of n-bar planar and spherical linkage mechanisms consisting of both revolute and prismatic joints. Generalized constraint equations using point, line and plane coordinates have been proposed which unify simulation of planar and spherical linkages and are demonstrably scalable to spatial mechanisms. As opposed to some of the existing approaches, which seek to derive loop-closure equations for each type of mechanism separately, we have shown that the simulation can be made simpler and more efficient by using a unified version of the geometric constraints on joints and links. This is facilitated using homogeneous coordinates and constraints on geometric primitives, such as point, line, and plane. Furthermore, the approach enables simpler programming, real-time computation, and ability to handle any type of planar and spherical mechanism. This work facilitates creation of practical and intuitive design tools for mechanism designers.

In Chapter 5, a novel motion synthesis algorithm is proposed which simultaneously synthesizes planar, spherical and spatial mechanisms. There exist many approaches in literature that solve the kinematic motion synthesis problem independently for planar, spherical, or spatial mechanisms. While it is well-known that planar and spherical mechanisms can be regarded as special cases of spatial mechanisms, most synthesis methods break down when presented with degenerate data. A unified approach to spatial, spherical, and planar mechanism synthesis has been elusive. In this chapter, we present a novel method to generate planar four-bar, spherical four-bar, and spatial 5-SS mechanisms using a unified algorithm and in the process enable a unified treatment of the Burmester problem. For a spatial motion problem where all the poses lie on a sphere or a plane, we show that there exist 3-∞ spherical-spherical (SS) solution dyads or planar-spherical (PS) solution dyads, respectively. We also show that for a spatial five pose problem where all poses lie on a sphere or plane, there exists a 2-∞ solution space of spherical RR dyads or planar RR dyads. This multiplicity of solutions are intelligently constrained to find unique dyads on a characteristic-sphere or -plane representing spherical or planar four-bar mechanism, respectively. The proposed approach removes the limitations of existing spatial synthesis algorithms which fail to synthesize possible mechanisms when the poses lie on a sphere or a plane. The algorithm is efficient and real-time in nature. Several examples are presented to validate the proposed algorithm.

In Chapter 6, we propose a machine learning based approach to solve the path synthesis problem for spatial mechanisms. The synthesis of spatial mechanisms for defect-free path generation has not received a lot of attention. In this chapter, 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.

Finally, Chapter 7 proposes a machine learning based approach to solve the Alt-Burmester problem for spatial mechanisms. An m-pose, n-path-points kinematic synthesis problem is known as the Alt-Burmester problem. It is the generalization of well-studied motion and path synthesis problems and unifies them into a single framework. In this chapter, we present a machine learning based algorithm to synthesize defect-free spatial 5-SS platform mechanisms, wherein a moving platform is connected to a fixed base via five SS (spherical-spherical) dyads. While a lot of work has been done in the synthesis of planar and to some extent spherical mechanisms, generating mechanisms, which are free of circuit, branch, and order defects has proven to be a difficult task. This task is even more difficult for spatial mechanisms, which can consist of a large number of circuits and branches. Combining this with the Alt-Burmester problem, solving such problems using a purely analytical approach becomes even more unmanageable. In this chapter, we present a novel machine learning algorithm based on a Variational Auto-Encoder (VAE) architecture, which helps capture the relationship between path and orientation properties of the motion of the 5-SS mechanisms. This helps in reformulating the mixed synthesis problem into a pure motion synthesis problem. This approach is scalable to any single degree of freedom spatial mechanism. First, we create a database consisting of possible 5-SS linkage's coupler motions. These motions are calculated using an iterative Newton-Raphson algorithm based solver. Subsequently, a data cleanup, normalization, balancing, augmentation, and masking pipeline is established using curvature and torsion of path-curves and quaternion based orientation data. Then, a Variational Autoencoder based machine-learning model is trained to capture the relationship between path and orientation data. This helps is reformulating the mixed synthesis problem into a pure motion synthesis problem. Finally, K-means clustering is used to find a multiplicity of defect-free 5-SS mechanisms, and examples are presented. The outlined approach can be scaled to any single degree-of-freedom spatial mechanism synthesis.

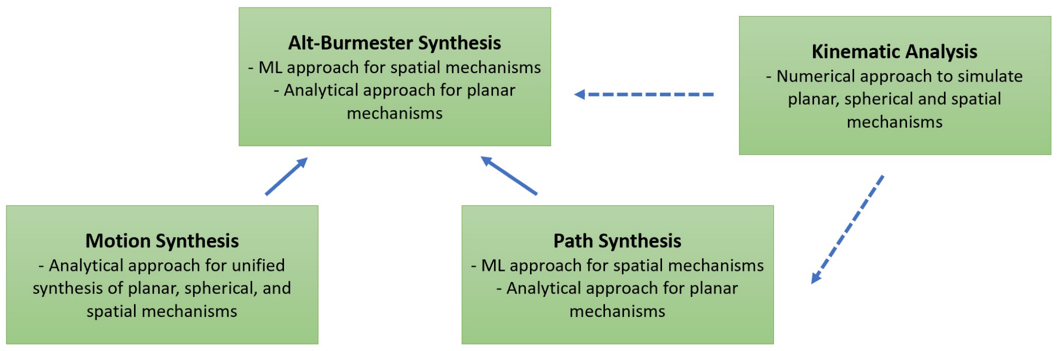

Alt-Burmester Synthesis
- ML approach for spatial mechanisms
- Analytical approach for planar mechanisms
Kinematic Analysis
- Numerical approach to simulate planar, spherical and spatial mechanisms
Motion Synthesis
- Analytical approach for unified synthesis of planar, spherical, and spatial mechanisms
Path Synthesis
- ML approach for spatial mechanisms
- Analytical approach for planar mechanisms

Figure 1.1: Dissertation Contributions

Thus, in this dissertation, we propose multiple algorithms which pushes the state-of-art in the field of kinematic analysis and synthesis of closed-loop single degree of freedom mechanisms. The summary of major contributions of this dissertation have been visualized in Fig. 1.1.

Chapter 2

An Optimal Parametrization Scheme for Path Generation using Fourier Descriptors for Four-bar Mechanism Synthesis

2.1 Introduction

This chapter is concerned with the path generation problem, wherein a path is usually given as a sequence of discrete points in R^2 and the goal is to find dimensions of a planar four-bar mechanism such that a point on the coupler of the mechanism traces the given points as close as possible [1]. Optimization-based techniques attempt to minimize an objective function and find mechanisms, which best approximate a curve generated by using Fourier series [2]. This curve, called the task curve requires point data on the prescribed path **and** associated time parameter values. Discrete Fourier transform (DFT) has been used to calculate the Fourier coefficients or Fourier descriptors (FDs) of the task curve from the prescribed path. FDs have been frequently used in computational shape analysis to create the objective function [3, 4, 5, 6, 7, 8, 9]. This chapter focuses on solving the path generation problem using a FD based technique.

McGarva and Mullineux [5] studied the inherent dependency of FDs of a task curve on time parametrization and concluded that different parametrization leads to different FDs. In previous studies, this limitation has been ignored and the parametrization has been assumed to be uniform. This results in a task curve which might be sub-optimal for use in mechanism synthesis.

Fig. 2.1 demonstrates an instance, where using a non-uniform parametrization yields a better mechanism than a uniform parametrization. A test mechanism is taken and ten arbitrary points are sampled on its coupler curve to generate the input data. Once the task curve is calculated, a four-bar coupler curve is fitted to synthesize a mechanism for each case. Comparing the resulting mechanisms with the input shows that non-uniform parametrization fits the points more accurately and better matches the first mechanism. Although, in Computer Aided Design (CAD), finding parametrization for a given sequence of points in the context of curve interpolation is a common problem, here we have the additional burden of ensuring that the parametrization is compatible with the properties of the coupler curves of planar four-bar mechanisms. In the example presented later on, we will show that the optimized task curve matches the harmonic contents of the coupler curve better than an arbitrary choice of parameters. Finding this optimum non-uniform parametrization serves as the motivation of this chapter.

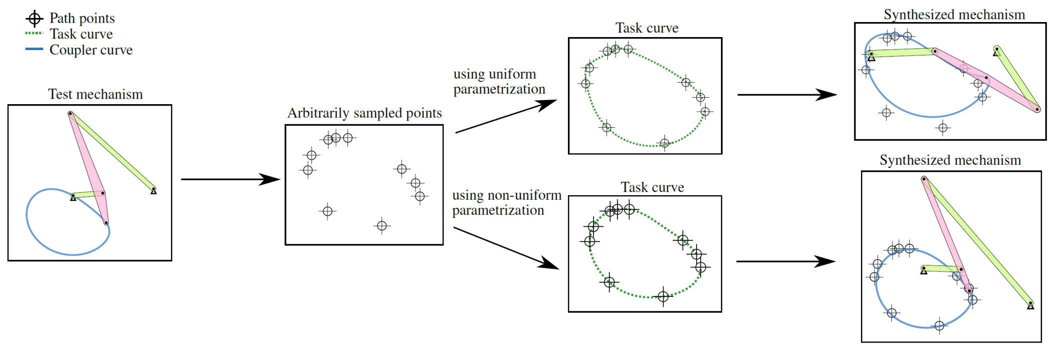


Figure 2.1: Path generation of two four-bar mechanisms; one using uniform parametrization while the other using optimal non-uniform parametrization

Recently, Li et al. [10] proposed an approach to eliminate time dependency using arc length parametrization. It is well-known in CAD community that finding an explicit closed-form expression for arc-length parametrization is impossible. As a result, they numerically **guess** the non-uniform parametrization using the distance between input points. Geometrically, the formulated parametrization is close to being time-independent. But this approach forces the coupler point to move at a constant speed along the task curves and eliminates a host of other possible four-bar mechanisms.

In this chapter, a novel methodology which calculates the optimal parametrization for a sequence of points has been proposed. For a four-bar coupler curve, the magnitude of its higher order harmonics has been observed to be negligible by Freudenstein [3] and Li et al. [11]. This property is used to search for an optimal parametrization to generate a task curve with low magnitude higher order harmonics. Coupler speed criteria for enhanced control over the task curve has also been incorporated. A new cost function is proposed that combines the cost of fitting to the given data points, the cost of low magnitude higher order harmonic, and the penalty for larger speed ratios. Nelder-Mead optimization is used to compute two critical state space parameters that minimize the cost function and provide optimized time parameters. Thereafter, we use the algorithm presented by Wu et al. [7] to synthesize a four-bar mechanism from the optimal task path. This algorithm fits task curve FDs to coupler curve FDs using a four-dimensional search space instead of the conventional ten-dimensional search space.

The rest of this chapter is organized as follows. Section 2.2 reviews an existing FD based path generation approach, which supplements the proposed algorithm for four-bar mechanism synthesis. Section 2.3 introduces a family of non-uniform parametrization and a methodology for finding the optimal one among them. Section 2.4 presents an example, which illustrates the effectiveness of the proposed approach.

2.2 Review of Fourier descriptor based path generation

This section provides a brief overview of an existing FD based path generation algorithm proposed by Wu et al. [7], which is used in conjunction with the improved task curve generation method as discussed in Sec. 2.3 for four-bar synthesis.

In this method, input path points are used to calculate a task curve described by a trigonometric polynomial curve with an open interval and is represented as

$$z(t) = \sum_{k=-p}^{p} T_k e^{ik\omega_o t} \qquad \forall \quad t \in [0, t_{max}], \tag{2.1}$$

where $z(t) = x(t) + iy(t)$ denotes the point coordinates in complex form at time t, k are the frequency indices, T_k are the FDs, ω_o is the angular velocity of crank, and $[0, t_{max}]$ is the time interval over which the curve is defined. It has been shown that a task curve with $p = 5$ captures the four-bar coupler path very accurately and is deemed sufficient for practical implementation [11]. Taking $\omega_0 = 2\pi$ and $t_{max} \in (0, 1]$ represents the possible closed and open task paths. The FDs are calculated by solving the least square fitting problem with the objective function as

$$\Delta = \sum_{i=1}^{n} \left\| z(t_i) - \sum_{k=-p}^{p} T_k e^{ik\omega_o t_i} \right\|^2, \tag{2.2}$$

where n is the number of input points. If $n < (2p + 1)$, then the highest $2\lceil \frac{2p+1-n}{2} \rceil$ harmonics are specified as zero to find a unique solution. Here, $\lceil ... \rceil$ represents the ceiling function. Finding the domain $[0, t_{max}]$ for the open task path is a one-dimensional optimization problem for minimum error measure defined in Eq. (2.2). Once a time parametrization is chosen or assumed, the task curve can be calculated.

A four-bar mechanism can be represented by the design parameters x_0, y_0, l_1, l_2, l_3, l_4, r, α, θ_1, and initial crank angle ϕ_0 as displayed in Fig. 2.2.

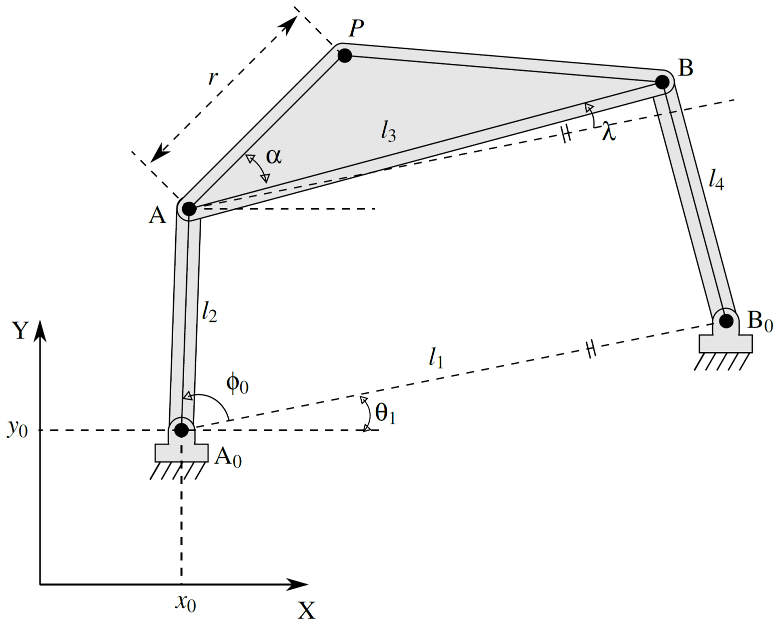


Figure 2.2: A planar four-bar mechanism showing dimensional parameters

The analytic equation of coupler point P can be given as

$$P = A_0 + l_2 e^{i\theta_1} e^{i\phi} + r e^{i\alpha} e^{i\theta_1} e^{i\lambda}, \tag{2.3}$$

where λ is the coupler angle as shown in Fig. 2.2 and A_0 is the complex form of the fixed pivot given by (x_0, y_0). A closed form expression for λ using loop-closure criteria is available in literature [7], and can be approximated using FDs as

$$e^{i\lambda} = \sum_{k=-p}^{p} C_k e^{ik\phi}, \tag{2.4}$$

where C_k are coupler angle FDs whose value depends upon the link lengths l_1, l_2, l_3 and l_4, and $\phi = \phi_0 + \omega_0 t$.

The coupler path can also be expressed as a Fourier series given in Eq. (2.5) where P_k are coupler path FDs.

$$P = \sum_{k=-p}^{p} P_k e^{ik\omega t}. \tag{2.5}$$

By substituting Eq (2.4) into Eq. (2.3) and collecting coefficients of $e^{ik\omega t}$ to form Eq (2.5), we can express the P_k as

$$P_0 = re^{i\alpha}e^{i\theta_1}C_0 + A_0, \tag{2.6}$$

$$P_1 = re^{i\alpha}e^{i\theta_1}C_1e^{i\phi_0} + l_2e^{i(\theta_1+\phi_0)}, \text{and} \tag{2.7}$$

$$P_k = re^{i\alpha}e^{i\theta_1}C_ke^{ik\phi_0}|_{k\neq 0,1}. \tag{2.8}$$

The coupler path can now be fitted to the task curve to calculate the four-bar design parameters using Eq. (2.1) and Eq. (2.5). Equating T_k to P_k leads to a system of equations with ten unknowns given as following

$$S = \left\{ l_2, \frac{l_2}{l_1}, \frac{l_3}{l_1}, \frac{l_4}{l_1}, x_0, y_0, \theta_1, \phi_0, \mathbb{C}, \mathbb{S} \right\}, \tag{2.9}$$

where $\mathbb{C} = r\cos(\alpha+\theta_1)$ and $\mathbb{S} = r\sin(\alpha+\theta_1)$. Equation (2.8) depends on six design variables $\left\{\frac{l_2}{l_1}, \frac{l_3}{l_1}, \frac{l_4}{l_1}, \phi_0, \mathbb{C}, \mathbb{S}\right\}$, while the remaining four variables $\{l_2, x_0, y_0, \theta_1\}$ exist independently in Eq. (2.6) and (2.7). Wu et al. [7] show that the six-dimensional design space in Eq (2.8) can be further reduced to a four-dimensional space of $\left\{\frac{l_2}{l_1}, \frac{l_3}{l_1}, \frac{l_4}{l_1}, \phi_0\right\}$ by analytically minimizing the following objective function

$$I = \sum_{k\neq 0,1} |C_kre^{i(\alpha+\theta_1+k\phi_0)} - T_k|^2. \tag{2.10}$$

This objective function is obtained by summing the squared difference of P_k given in Eq. (2.8) and T_k.

The direct search method has been used by Wu et al. [7] to solve this optimization problem. In summary, matching task path FDs to coupler path FDs using a four-dimensional search space forms the core of this approach.

2.3 Optimum Parametrization

The task curve calculation is inherently associated with the time parametrization used. However, there are an infinite ways to select a non-uniform parametrization. To make the problem tractable and facilitate the selection of a single non-uniform parametrization, a chord-length based parametrization scheme for n-points sequence is defined as

$$t_1 = 0, \tag{2.11}$$

$$t_k = t_{max} \left(\frac{\sum_{i=2}^{k} |z_i - z_{i-1}|^\alpha}{\sum_{i=2}^{n} |z_i - z_{i-1}|^\alpha} \right). \tag{2.12}$$

Here, t_k represents the time parameter associated with the k^{th} path point, t_{max} is the interval domain as defined in Eq. (2.1), z_i represents the coordinates of i^{th} point and $\alpha \in \mathbb{R}$ is the parametrization control variable (PCV). Varying the control variable α generates multiple parametrizations. In this scheme, when $\alpha = 0$ and $\alpha = 1$, we obtain uniform and arc length parametrizations, respectively. Physically, uniform parametrization results in a task curve where the coupler takes equal amount of time to pass through each target point. Similarly, the arc length parametrization approximates constant speed motion of the coupler. By varying α, one can generate different parametrizations for the calculation of the task curves, which in turn, could provide a range of mechanism design solutions and also facilitates selection of an optimal parametrization, leading to mechanisms that provide better fit with the input data.

To measure the quality of the task curve, we define a cost function as following

$$C_t = C_f + C_h + C_s \tag{2.13}$$

where C_f is the cost attributed to FD fitting error, C_h is the cost due to higher order harmonic content, and C_s is the cost due to enforced speed criteria on the task curve.

If the path (z_i), PCV (α), and time domain (t_{max}) are known, the FDs of an open task curve can be mean square fitted as shown in Eq. 2.2. The square-root of the residual in the fitting process is normalized to define C_f in the cost function and is given as

$$C_f = \frac{1}{n} \sqrt{\sum_{i=1}^{n} \left\| z_i - \sum_{k=-p}^{p} T_k e^{ik\omega_o t_i} \right\|^2}. \tag{2.14}$$

Here, n represents the total path points, z_i are the path coordinates, T_k are the task curve FDs and t_i is the time parameter attached to i^{th} point. This term ensures that the selected parametrization and domain accurately represent the original point data.

The second term in the cost function, namely harmonic cost, is motivated by the observation from Li et al. [11] that the magnitude of higher order harmonics are negligible for four-bars. As a result, task curves whose higher order harmonics have minimal magnitude would be better prospective curves for the path synthesis process. To enforce this harmonic criterion, a weighted harmonic magnitude based metric is defined as

$$C_h = \frac{1}{2p+1} \sum_{k=-p}^{p} (|k|+1)^{\beta} \left\| T_k \right\|. \tag{2.15}$$

Here, p is the maximum number of harmonics being considered, $\beta \geq 1$ is a constant and T_k are the task curve FDs. The term $(|k|+1)^{\beta}$ adds larger weight to the higher order harmonics.

In practical scenarios, large coupler speed changes can lead to large induced forces on the links which could compromise their rigidity and render the kinematic analysis useless. In contrast, a uniform speed motion of the coupler can also be undesirable in some instances, such as when designing a quick-return mechanism. Thus, enforcing a speed-based criteria over the task curve gives users more control in design. To select a task curve with desirable speed properties, speed cost (C_s) is defined using quadratic penalty function as follows

$$C_s = w \left[max(0, S_{r,min} - S_r)^2 + max(0, S_r - S_{r,max})^2 \right], \tag{2.16}$$

where S_r represents ratio of task curve's maximum to minimum speed. $S_{r,min}$ and $S_{r,max}$ are the minimum and maximum speed ratio bounds enforced by the designer and w is the penalty weight imposed. In case there are no speed restrictions, $S_{r,min}$ and $S_{r,max}$ can be set to 1 and infinity, respectively. $w = 10^3$ has been taken in the implementation. The expression for task curve speed can be calculated by differentiating Eq. (2.1) which gives

$$s(t) = \left\| \sum_{k=-p}^{p} ik\omega_o T_k e^{ik\omega_o t} \right\|. \tag{2.17}$$

Maximum and minimum speeds can be calculated numerically by sampling a large number of points over the task curve.

Thus, an optimal task curve can be calculated by minimizing the total cost (C_t) as given in Eq. (2.13). The search space is two dimensional with α and t_{max} as the state variables. Thereafter, the optimal chord-length based time parametrization can be calculated using Eq. (2.11) and Eq. (2.12). Nelder-Mead optimization has been used for searching the state space. With the task curve known, a four-bar mechanism can be generated as discussed in Sec. 2.2. The complete Fourier based path synthesis algorithm using optimum parametrization has been summarized in the Algorithm 1.

```
Algorithm 1: Path generation using Fourier descriptor based approach with optimal parametrization
  Input: Set of path points
1 Search for optimum $\alpha$ and $t_{max}$ by minimizing $C_t$ given in Eq. (2.13).
2 Calculate $\left\{\frac{l_2}{l_1}, \frac{l_3}{l_1}, \frac{l_4}{l_1}, \phi_0, \mathbb{C}, \mathbb{S}\right\}$ by minimizing $I$ given in Eq. (2.10)
3 Calculate $\{l_2, x_0, y_0, \theta_1\}$ using Eq. (2.6) and Eq. (2.7) to synthesize a
    four-bar mechanism.
  Output: Four-bar design parameters
```

2.4 Example

This example demonstrates the improvement made using the proposed methodology. A 12-point trajectory is taken as the input and is given in

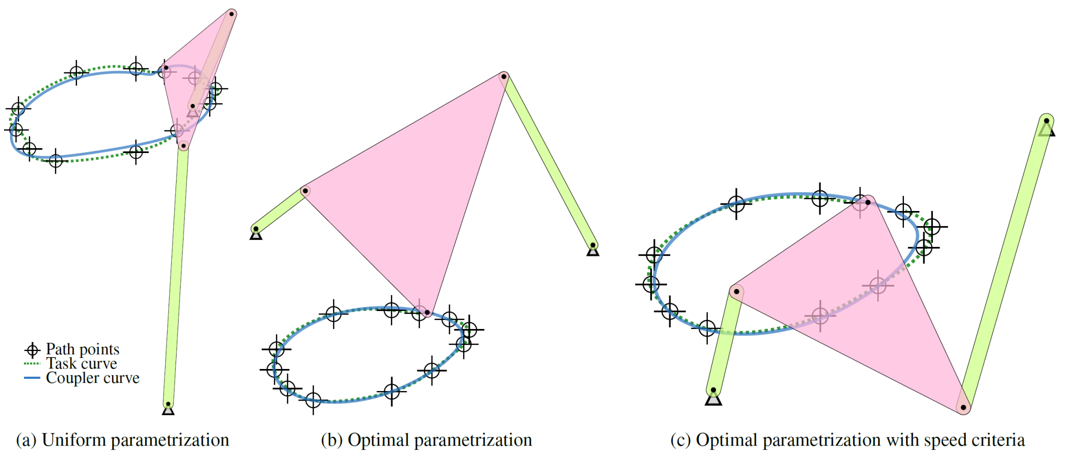

Path points
Task curve
Coupler curve
(a) Uniform parametrization
(b) Optimal parametrization
(c) Optimal parametrization with speed criteria

Figure 2.3: Synthesized solutions using different parametrizations

Table 2.1. In this example, we use $\omega_o = 2\pi$ rad/s (Eq. 2.2) and $\beta = 2$ (Eq. 2.15).

Table 2.1: Input point data

No.	Coordinate (x, y)	No.	Coordinate (x, y)	No.	Coordinate (x, y)
1	$0.000, -1.000$	5	$-2.866, -2.118$	9	$0.246, -2.135$
2	$-0.550, -0.942$	6	$-2.608, -2.488$	10	$0.876, -1.615$
3	$-1.696, -1.018$	7	$-2.098, -2.720$	11	$0.986, -1.329$
4	$-2.821, -1.715$	8	$-0.546, -2.551$	12	$0.593, -1.123$

Table 2.2: Task curve Fourier Descriptors

	Parametrization		
Descriptor	Uniform	Optimal	Optimal with speed criteria
-5	$0.035 - 0.051i$	$-0.001 + 0.013i$	$-0.003 + 0.017i$
-4	$-0.021 + 0.033i$	$-0.034 - 0.013i$	$-0.011 - 0.007i$
-3	$0.008 + 0.037i$	$-0.022 - 0.014i$	$-0.021 - 0.016i$
-2	$-0.015 + 0.086i$	$-0.009 - 0.018i$	$-0.003 - 0.036i$
-1	$0.420 - 0.455i$	$0.412 - 0.290i$	$0.378 - 0.261i$
0	$-0.822 - 1.696i$	$-0.960 - 1.763i$	$-0.970 - 1.764i$
1	$0.352 + 1.334i$	$0.715 + 1.167i$	$0.719 + 1.148i$
2	$0.074 - 0.204i$	$-0.005 - 0.072i$	$-0.015 - 0.065i$
3	$0.054 - 0.051i$	$-0.053 - 0.008i$	$-0.072 + 0.013i$
4	$-0.051 - 0.077i$	$-0.045 + 0.002i$	$-0.021 - 0.002i$
5	$-0.019 + 0.042i$	$-0.002 - 0.010i$	$0.003 - 0.032i$

Table 2.3: Coupler curve Fourier Descriptors

	Parametrization		
Descriptor	Uniform	Optimal	Optimal with speed criteria
-5	$0.011 + 0.006i$	$-0.001 + 0.000i$	$-0.001 + 0.000i$
-4	$0.008 + 0.019i$	$0.000 + 0.001i$	$-0.001 + 0.000i$
-3	$0.049 - 0.019i$	$-0.006 - 0.011i$	$-0.007 - 0.014i$
-2	$0.005 + 0.104i$	$-0.005 - 0.020i$	$-0.001 - 0.031i$
-1	$0.430 - 0.453i$	$0.412 - 0.291i$	$0.376 - 0.262i$
0	$-0.822 - 1.696i$	$-0.960 - 1.763i$	$-0.970 - 1.764i$
1	$0.352 + 1.334i$	$0.715 + 1.167i$	$0.719 + 1.148i$
2	$0.055 - 0.170i$	$0.004 - 0.074i$	$-0.006 - 0.077i$
3	$0.000 - 0.034i$	$-0.006 + 0.006i$	$-0.004 + 0.007i$
4	$-0.006 - 0.025i$	$0.003 - 0.000i$	$0.003 - 0.001i$
5	$-0.005 - 0.012i$	$-0.001 - 0.000i$	-0.001 - 0.000i

Table 2.4: Synthesized mechanism parameters

	Parametrization		
Variable	Uniform	Optimal	Optimal with speed criteria
l_1	10.020	9.271	10.140
l_2	1.921	2.085	1.464
l_3	6.719	6.520	3.994
l_4	5.423	5.160	8.139
x_0	-6.464	-5.560	11.951
y_0	1.659	0.931	-2.386
θ_1	-0.346	-0.191	0.859
r	12.191	11.519	10.194
α	0.119	0.001	0.661
ϕ_0	0.774	0.764	3.218

First, we use uniform parametrization to create a task curve. The task curve is calculated to have $t_{max} = 0.8770$ and FDs as given in Table 2.2. The task curve fitting error, as defined in Eq (2.2), is found to be $\Delta = 0.3509$. From this task curve, a mechanism is synthesized to find the four-bar design parameters. Coupler curve FDs and computed mechanism parameters are given in Table 2.3 and Table 2.4, respectively. Coupler curve fitting error as defined in Eq. (2.10), is calculated to be $I = 0.0228$. The synthesized mechanism can be viewed in Fig. 2.3a. Next, mechanism synthesis is accomplished using optimal parametrization. The task curve is calculated to have $t_{max} = 0.9336$, $\alpha = 0.5823$ and FDs as given in Table 2.2. The task curve fitting error is observed to be $\Delta = 0.0591$, which is better than the previous case. Subsequently, a mechanism is synthesized and the coupler curve FDs and the solution mechanism parameters are given in Table 2.3 and Table 2.4, respectively. The coupler curve fitting error is found to be $I = 0.0067$, which is also less than the uniform parametrization case. The synthesized mechanism can be viewed in Fig. 2.3b. While using non-uniform parametrization enables the reduction of fitting error Δ, this example demonstrates that a task curve with low magnitude higher order harmonics decreases I and leads to a better task-coupler curve matching. Fig. 2.4 displays the comparison of weighted FDs, given as $T_{w,k} = (|k| + 1)^2 \|T_k\|$, for uniform and optimal parametrization. These figures shows that for optimal parametrization case, the magnitude of the higher order harmonics is less compared to the uniform case for both the task and the coupler curves.

Finally, mechanism synthesis involving speed criteria is carried out. Speed ratio for the task curve calculated using uniform parametrization is observed to be 8.22. Let us assume that the user desires to constrain S_r, such that $1 \leq S_r \leq 2$. After applying the speed criteria, the task curve is calculated to have $t_{max} = 0.9234$, $\alpha = 0.7885$ and FDs as given in Table 2.2. The task curve fitting error is observed to be $\Delta = 0.1813$. The S_r of the generated task curve is 2. Comparison of task curve speeds has been done in Fig 2.5. It can be observed that the new task curve has reduced the speed ratio. Synthesized mechanism design parameters are given in Table 2.4. The coupler curve fitting error is observed to be $I = 0.0444$ and the solution is displayed in Fig. 2.3c.

2.5 Conclusion

In this chapter, a non-uniform parametrization scheme has been proposed for the task curve calculation from a given sequence of path points. A novel methodology to find the optimal parametrization based on fitting accuracy, the harmonic properties of four-bar coupler path, and user imposed speed criteria has been demonstrated. Synthesis of a more accurate four-bar mechanism for path generation has been shown using an example. The proposed approach improves upon the existing FD based path generation algorithm for mechanism synthesis.

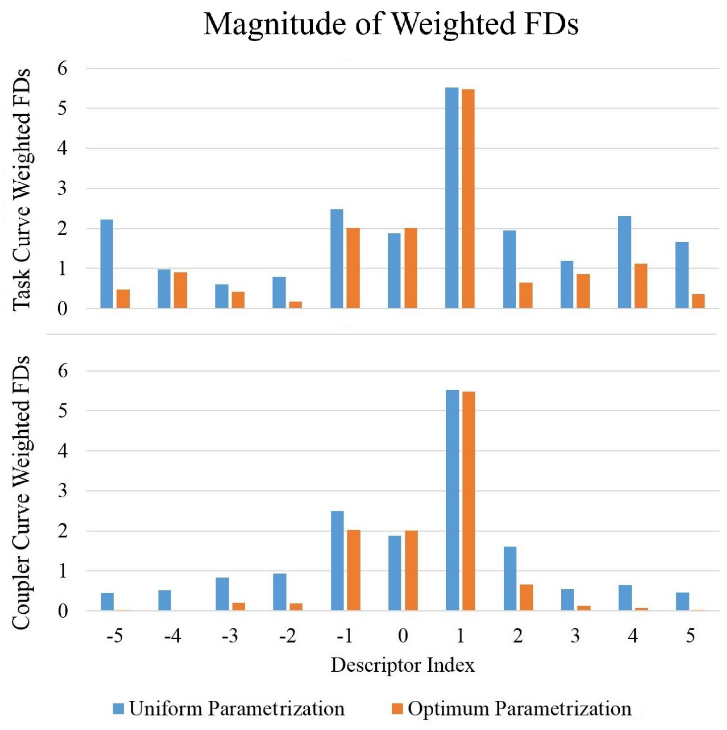


Figure 2.4: Comparison of task curve and coupler curve weighted FDs for uniform and optimal parametrization

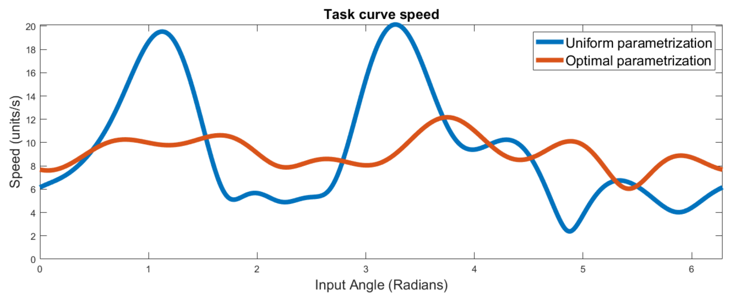


Figure 2.5: Comparison of task curve speeds obtained with and without speed criteria

Chapter 3

A Motion Synthesis Approach to Solving Alt-Burmester Problem by exploiting Fourier Descriptor Relationship between Path and Orientation Data

3.1 Introduction

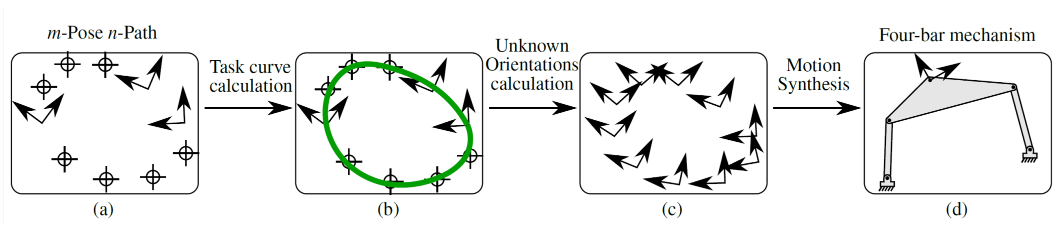


Figure 3.1: An Overview of our Approach to the Alt-Burmester Problems: (a) specify m-pose, n-path points; (b) a task curve is fit through the $m + n$ path points using Fourier series; (c) use the harmonic content of the path data to find the **missing** orientations at the n-path points; and (d) finally, compute both type and dimensions of planar four-bar linkages.

Conventionally, mechanism synthesis problems have been categorized and studied independently as path, motion, and function synthesis problems [12]. Path synthesis problems specify only path-point coordinates (x_i, y_i), while motion synthesis problems specify pose constraints (x_i, y_i, ζ_i), where (x_i, y_i) are the coordinates of the path-points or the origin of a moving frame attached to a given pose, while ζ_i is the orientation of the moving frame. In function synthesis, only input-output angle pairs (θ_i, ψ_i) are specified. Unfortunately, most of the real world problems do not conform to such a rigid categorization – many practical problems provide a mixture of path, motion and function synthesis requirements. However, a synthesis approach which seamlessly incorporates all the three conventional synthesis problems has been elusive. As a result, machine designers often have to compromise on design specifications. In this chapter, the focus is on the synthesis of planar four-bar mechanisms for a hybrid of path and motion synthesis problems. This problem formulation, which consolidates both path and orientation data, has been termed as mixed synthesis in this chapter.

Murray's group termed the combined path and motion problems as the Alt-Burmester problems [13] named after Alt's [14] and Burmester's [15] work on path and motion generation, respectively. Brake et al. [16] discuss the dimensionality of solution sets for a variety of path-point and pose combinations. However, a finite number of solutions exist only for a subset of possible m-pose, n-path point synthesis problem. For example, there exist finite solutions for nine path-points and for five poses independently. We define such problems to be fully constrained problems. For a lesser number of path-points or poses, usually an infinite number of solutions are obtained. Subsequently, the authors explore only fully-constrained or under-constrained problem sets where up to nine constraints can be used to find four-bar mechanism parameters. This ignores the vast majority of over-constrained problems in m-pose, n-path point mixed synthesis family of problems, where exact solutions are not possible and only approximate, albeit useful solutions, can still be obtained. This is reflective of real-world design problems, which usually impose a large number of often challenging constraints.

A graphical approach has been presented by Zimmerman [17] to solve the mixed path, motion and function problem using sketching tools built in modern Computer-Aided Design (CAD) softwares. The proposed methodology can conveniently solve under-constrained and fully-constrained mixed synthesis problems and generate four-bar mechanisms. However, this methodology is unable to solve generalized m pose, n path-point synthesis problems, which may be over-constrained. This study does state the possibility of including prismatic joints in the synthesized mechanism. However, it is not the focus of study and details on synthesizing mechanical dyads with at least one prismatic joint are not included.

Motion synthesis turns out to be a mathematically less complex problem than path synthesis as each dyad can be generated independently effectively halving the number of unknowns. Typically, path synthesis problems involve solving a nonlinear system of equations. We have recently presented a generalized framework for solving motion synthesis problems using an efficient algorithm that involves solving a linear system of equations using singular value decomposition [18, 19, 20, 21, 22]. The algorithm produces multiple solutions and can compute both the type and dimensions of the four-bar mechanisms. The algorithm produces results in real-time and is thus amenable to its implementation in interactive computational design tools [18].

In this chapter, we are presenting an approach to solve the Alt-Burmester problem by reducing it to a pure motion synthesis problem so that the aforementioned algorithm can be leveraged. In a planar four-bar linkage, the path of a coupler point is inextricably tied to the orientation of the coupler. This coupling can be revealed by analyzing and relating the harmonic content of the path and orientation data. First, an analytical relationship between the orientation- and path-data is obtained using the harmonic breakdown of the loop closure equation. Then, this relation is used to reformulate the mixed synthesis problem into a motion synthesis problem by attaching compatible orientations to input path points and consequently turning them into poses. The Fourier approximation based analytical approach proposed in this chapter can handle almost all possible variations of path points or poses. Once, the problem has been converted into a pure motion synthesis problem, we repurpose our algebraic fitting approach in [21, 22] to solve for four-bar linkages. Figure 3.1 provides an overview of this approach.

We note that here mixed synthesis does **not** refer to the mixed exact-approximate path or motion synthesis, wherein we have a set of precision and approximate constraints. Our definition of **mixed** refers to a mixture of path-point and pose constraints.

This chapter's original contributions are in 1) the formulation of a Fourier descriptor based closed-form relationship between coupler orientations and path, 2) the novel use of this relationship to solve the generalized m-pose, n-path mixed synthesis problem, and 3) the incorporation of task-driven algebraic fitting based motion synthesis within the mixed synthesis algorithm for synthesis.

Rest of the chapter is organized as follows. Section 3.2 calculates a new path-orientation formulation from existing four-bar loop closure Fourier decomposition. Section 3.3 discusses the use of path-orientation relationship to reformulate mixed synthesis into motion synthesis problem. Section 3.4 reviews algebraic fitting based motion synthesis algorithm. Section 3.5 proposes a new algorithm to solve mixed synthesis problem and finally in section 3.6, we present a few examples to demonstrate the efficacy of the proposed approach.

3.2 Fourier Descriptors based relations

Use of Fourier descriptors is abundant in the domain of mechanism synthesis. It has been used for planar four-bar mechanism synthesis using optimization routines [23, 4, 24, 6, 11], atlas-based search algorithms [25, 26], and machine learning approach [8]. Fourier descriptors have also been used to synthesize spherical and spatial mechanism [27]. A class of single degree of freedom open-loop mechanisms termed as planar coupled serial chain mechanisms [28, 29] have also been generated with the help of Fourier descriptors.

In this section, we are interested in exploring the relationship between the coupler path and coupler orientation to establish a closed-form relationship between them. This would give us a framework for dealing with both pose and path constraints simultaneously. Path and motion synthesis formulations, which use Fourier decomposition of four-bar closure equation [24, 6, 11] are used as a starting point here. In [11], Li et al. presented a decomposition of the design space of four-bar mechanisms by using Fourier descriptors in the context of planar motion approximation. Harmonic decomposition of four-bar loop closure equation has been analyzed to independently fit rotational and translational Fourier descriptors and synthesize motion.

A four-bar mechanism is represented by its design parameters x_0, y_0, l_1, l_2, l_3, l_4, r, θ_1, and α as displayed in Fig. 3.2. These parameters are constant for a given four-bar mechanism. Coupler angle λ represents the varying orientation of coupler link with respect to fixed link at any given instant. Point P is the location of the coupler point in the global frame. which is also a variable. Coupler orientation ζ refers to the orientation of a moving frame attached to the coupler point, while δ is the constant angle at which moving frame is attached to coupler with respect to the coupler link line AB. All of these design parameters are unknown before a mechanism has been synthesized. Our goal is to find an explicit closed-form relationship between coupler path and orientation which forms the heart of our mixed synthesis algorithm.

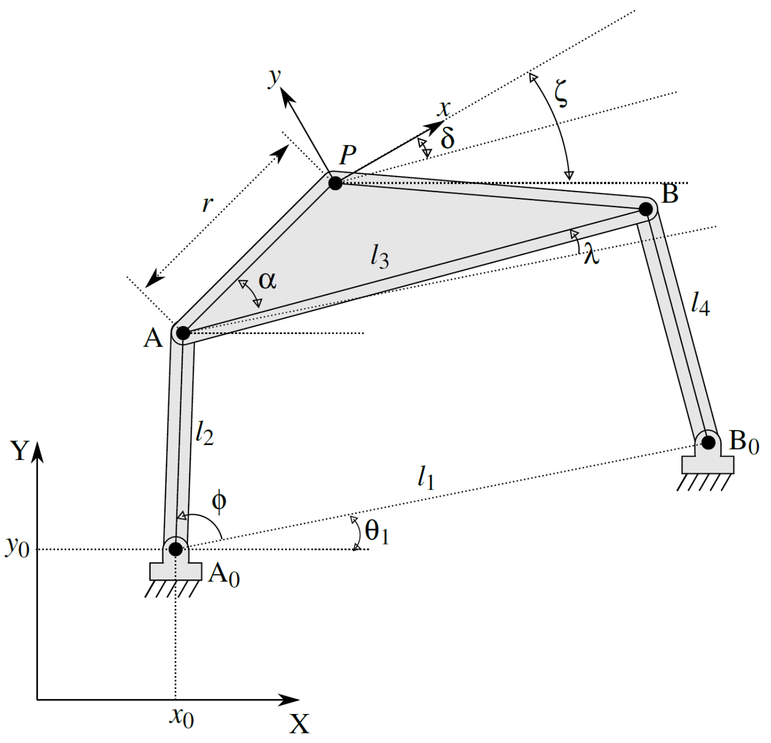


Figure 3.2: Visualization of parameters describing a four-bar mechanism

3.2.1 Coupler angle

The Fourier series representation of the coupler angle λ for a four bar mechanism is given as

$$e^{j\lambda} = \sum_{k=-\infty}^{\infty} C_k e^{jk\phi} = \sum_{k=-\infty}^{\infty} C_k e^{jk\omega t} e^{jk\phi_0}, \tag{3.1}$$

where C_k are the harmonic descriptors of coupler angle, ϕ the crank angle, ϕ_0 the initial crank angle, and ω is the constant angular speed of the input link.

3.2.2 Coupler path

The analytical equation which defines the path of coupler point P for a four bar mechanism is given by

$$P = A_0 + l_2 e^{j\theta_1} e^{j\phi} + r e^{j\alpha} e^{j\theta_1} e^{j\lambda}, \tag{3.2}$$

where A_0 is the complex form of the position of input link fixed pivot, l_2 is the length of input link, θ_1 is the angle of fixed link, and r and α are the coupler parameters. Being a periodic function, it can also be represented as a Fourier series:

$$\mathbf{P} = \sum_{k=-\infty}^{\infty} P_k e^{jk\omega t}. \tag{3.3}$$

Substituting (3.1) into (3.2) and then equating resulting (3.2) and (3.3), we get harmonic descriptors P_k for the path as following

$$P_0 = C_0 r e^{j(\alpha+\theta_1)} + (jy_0 + x_0); \qquad k = 0, \tag{3.4}$$

$$P_1 = C_1 e^{j\phi_0} r e^{j(\alpha+\theta_1)} + l_2 e^{j\theta_1} e^{j\phi_0}; \qquad k = 1, \tag{3.5}$$

$$P_k = C_k e^{jk\phi_0} r e^{j(\alpha+\theta_1)}; \qquad k \neq 0, 1. \tag{3.6}$$

3.2.3 Coupler orientation

The orientation (ζ) at the coupler point for a four-bar mechanism can be defined as

$$\zeta = \delta + \lambda + \theta_1 = \arg(e^{j(\delta+\lambda+\theta_1)}), \tag{3.7}$$

where δ is the fixed angle at which moving frame is attached to coupler with respect to $\theta_1 + \lambda$. As λ varies periodically while δ and θ_1 remain constant, the orientation can be decomposed harmonically as

$$e^{j(\delta+\lambda+\theta_1)} = \sum_{k=-\infty}^{\infty} C_k^* e^{jk\omega t}, \tag{3.8}$$

where C_k^* are the harmonic descriptors for orientation and obtained as

$$C_k^* = C_k e^{j(\delta+\theta_1)} e^{jk\phi_0} \tag{3.9}$$

by substituting for $e^{j\lambda}$ from Eq. (3.1) in Eq. (3.8).

3.2.4 Path-orientation relation

With the above relations, it is now possible to find explicit closed form relations between the Fourier descriptors of coupler path and coupler orientation data. Using Eqns. (3.4), (3.5), (3.6), and (3.9), relationship between the harmonic descriptors of path (P_k) and orientation (C_k^*) is found to be

$$C_0^* = (P_0 + z_2)z_1, \tag{3.10}$$

$$C_1^* = (P_1 + z_3)z_1, \tag{3.11}$$

$$C_k^* = P_k z_1, \tag{3.12}$$

where

$$z_1 = \frac{e^{j(\delta-\alpha)}}{r}, \tag{3.13}$$

$$z_2 = -(x_0 + jy_0), \tag{3.14}$$

$$z_3 = -(l_2 e^{j\theta_1} e^{j\phi_0}). \tag{3.15}$$

Using the above relationship, the orientation at coupler point can be defined exclusively using path harmonic descriptors as follows

$$e^{j\zeta(t)} = z_1 \left(z_2 + z_3 e^{j\omega t} + \sum_{k=-\infty}^{\infty} P_k e^{jk\omega t} \right). \tag{3.16}$$

Subsequently, using Eq. (3.16) for n path points, the system of equation describing orientation at each path point turns out to be

$$\begin{bmatrix} e^{j\zeta_1} \\ e^{j\zeta_2} \\ \vdots \\ e^{j\zeta_n} \end{bmatrix} = \begin{bmatrix} 1 & e^{j\omega t_1} & \sum_{k=p}^{-p} P_k e^{jk\omega t_1} \\ 1 & e^{j\omega t_2} & \sum_{k=p}^{-p} P_k e^{jk\omega t_2} \\ \vdots & \vdots & \vdots \\ 1 & e^{j\omega t_n} & \sum_{k=p}^{-p} P_k e^{jk\omega t_n} \end{bmatrix} \begin{bmatrix} z_1 z_2 \\ z_1 z_3 \\ z_1 \end{bmatrix}. \tag{3.17}$$

Thus, the orientations at different points of a four-bar coupler path are dependent on path descriptors and three complex variables z_1, z_2, and z_3, which are termed as Mixed Synthesis Parameters (MSP). The MSP are dependent on four-bar mechanism design parameters according to Eqns. (3.13), (3.14), and (3.15). The Eq. (3.17) is the key to the mixed synthesis formulation. It will help us find orientation information for path points as discussed in the next section.

3.3 Calculating unknown orientations

The aim of this section is to reformulate m-pose, n-path point mixed synthesis problems into an $m+n$- pose motion synthesis problems. To enable that, generation of orientation data for n path points and converting them to n poses is required. Eq. (3.17) will be used to accomplish this objective.

For a m-pose, n-path synthesis problem, a task path described by a trigonometric polynomial curve with an open interval is calculated and represented as

$$z(t) = \sum_{k=-p}^{p} T_k e^{ik\omega t} \qquad \forall \quad t \in [0, t_{max}], t_{max} < 1, \tag{3.18}$$

where $z(t) = x(t) + iy(t)$ denotes the point coordinates in complex form at time t, k are the frequency indices, T_k are the task curve Fourier descriptors, ω is the angular velocity of crank, and $[0, t_{max}]$ is the time interval over which the curve is defined. The T_k can be calculated by least square minimization of

$$\Delta = \sum_{i=1}^{n} \left\| z(t_i) - \sum_{k=-p}^{p} T_k e^{ik\omega t_i} \right\|^2, \tag{3.19}$$

where Δ is the fitting error measure and $z(t_i)$ are the complex-valued point data at time t_i. Analytically solving the minimization problem gives a linear system of equation as follows

$$\Omega \mathbb{X} = \mathbb{Y}, \tag{3.20}$$

where

$$\mathbb{X} = [\ldots, \underset{m\rightarrow}{T_m}, \ldots]^T, \tag{3.21}$$

$$\Omega = \begin{array}{c} \begin{bmatrix} & \cdots & \\ \vdots & \sum_{i=0}^{n} e^{i(k-m)\theta_i} & \vdots \\ & \cdots & \end{bmatrix} \downarrow m, \\ k\rightarrow \end{array} \tag{3.22}$$

$$\mathbb{Y} = [\ldots, \underset{m\rightarrow}{\sum_{i=0}^{n} z(t_i)e^{-im\theta_i}}, \ldots]^T. \tag{3.23}$$

Here, k and m vary from $-p$ to p which denote the column and row index of an element in the matrix. Thus, $\mathbb{X}$ and $\mathbb{Y}$ are m-dimensional vectors while Ω is a $m \times n$ dimensional matrix. LU decomposition can be used to solve the above system. More details can be found in the work done by Wu et al. [24]. In [30], we have proposed a method to calculate optimal time parametrization for task curve for Fourier descriptor fitting of the path data. In our implementation, task curves are represented using up to eleven descriptors i.e. $p \in [-5, 5]$. If $m + n < 11$, we use a lesser number of descriptors to generate a unique task curve.

The reasoning behind using a task curve with low higher order harmonic content is supported in literature [11, 3], which says that the magnitude of high harmonics for coupler path of a four-bar mechanism has an insignificant impact. Thus, the fitted task path is a good prospective four-bar coupler curve and the task curve descriptors T_k can be equated to coupler path descriptors P_k.

The intention now is to find the MSP i.e. $\{z_1, z_2, z_3\}$ using available orientation data and subsequently generate unknown orientations. We define the system of equation given by Eq. (3.17) as fully constrained if all the MSP can be calculated exactly. For a fully-constrained MSP computation problem, three poses are required to calculate the MSP directly from Eq. (3.17). Physically, this condition makes perfect sense as the user might know orientations at the initial position, final position and an additional intermediate location while a sequence of path points might be given in addition.

For under-constrained MSP computation problem, there are only one or two poses given. As a result, additional constraints are required to uniquely calculate the MSP. The MSP are dependent on four-bar mechanism parameters according to Eqns. (3.13), (3.14), and (3.15). These equations can be used to generate additional constraints which are called Mixed Constraints (MIC). The three possible MIC are

(1) Specify coupler parameters i.e. $\{r, \alpha, \delta\} \rightarrow z_1$

(2) Specify actuating fixed pivot i.e. $\{x_0, y_0\} \rightarrow z_2$

(3) Specify scale of input link, orientation of fixed pivot line, and initial angle i.e. $\{l_2, \theta_1, \phi_0\} \rightarrow z_3$

Thus, if two poses are input by the user, one MIC is required to fully define the system of equations in Eq. (3.17). If only one pose is specified by user, two MIC are required to solve the problem. Two pose problem is fairly common when only the first and last orientations are important, such as in pick-and-place operations. The MIC also mirror practical user-specified constraints, such as selection of the location of the fixed pivot where an actuator might be situated. In another case, there might be a restriction on coupler link dimensions. Thus, the MIC represent a set of practical design constraints.

It is important to note that a pure path synthesis problem cannot be restructured into a motion synthesis problem without fully defining all three MSP. However, constraining all MSP simultaneously makes the synthesis less useful as most of the mechanism parameters are then fixed.

For over-constrained MSP computation problems, the number of poses specified is more than three. In this case, a least square solution to Eq. (3.17) can be calculated using complex Singular Value Decomposition (SVD). Real

SVD solvers, which are more easily available, can also be used by reducing the complex system of equation in Eq. (3.17) into an equivalent real system of equation in accordance with [31]. The K_1 formulation presented in [31] has been used in our implementation. According to the formulation, a complex system of equation

$$(A + iB)(x + iy) = b + ic \tag{3.24}$$

can be written as a real system of equation

$$\begin{bmatrix} A & -B \\ B & A \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ c \end{bmatrix} \tag{3.25}$$

Finding least square solution to this equivalent real system of equations gives the solution to original complex problem and values of MSP can easily be calculated in over-constrained cases.

Once the values of MSP z_1, z_2, z_3 are calculated using m poses, orientations at n path points can be found out by simple matrix multiplication using the system of equation in Eq. (3.17). As a result, n path points and m poses are converted to $m + n$ poses. The motion synthesis algorithm can now be used to calculate dyads. A review of algebraic fitting based motion synthesis algorithm is discussed in next section.

3.4 Motion synthesis algorithm

Now that the mixed synthesis problem has been reformulated as motion synthesis problem, solution mechanisms can be achieved by calculating the dyads. Algebraic fitting based motion synthesis algorithm [18, 19, 20, 21] has been used in our implementation. In this approach, a planar four-bar linkage is split open in two dyads and each dyad is computed independently thus reducing significant computational burden. Moreover, this approach enables us to carry out simultaneous type and dimensional synthesis of four-bar linkages, i.e. it takes into consideration the possibility of both revolute and prismatic joints. Another benefit of the approach is its fast and efficient computation.

First, using kinematic mapping [32], each of the user-defined pose $\{x, y, \zeta\}$ is mapped to quaternion space defined by a four-dimensional vector $\mathbf{Z} = \{Z_1, Z_2, Z_3, Z_4\}$ called planar quaternions [33]. This space is also termed as the Image Space of planar kinematics [32]. This mapping is defined by

$$\begin{aligned} Z_1 &= \frac{1}{2}(x\cos\frac{\zeta}{2} + y\sin\frac{\zeta}{2}), & (3.26)\\ Z_2 &= \frac{1}{2}(-x\sin\frac{\zeta}{2} + y\cos\frac{\zeta}{2}), & (3.27)\\ Z_3 &= \sin\frac{\zeta}{2}, & (3.28)\\ Z_4 &= \cos\frac{\zeta}{2}. & (3.29) \end{aligned}$$

The geometric constraints of all types of dyads can be represented by a single algebraic equation as following

$$\begin{aligned} &q_1(Z_1^2 + Z_2^2) + q_2(Z_1Z_3 - Z_2Z_4) + q_3(Z_2Z_3 + Z_1Z_4)\\ &\quad +q_4(Z_1Z_3 + Z_2Z_4) + q_5(Z_2Z_3 - Z_1Z_4) + q_6Z_3Z_4\\ &\quad +q_7(Z_3^2 - Z_4^2) + q_8(Z_3^2 + Z_4^2) = 0, \qquad (3.30) \end{aligned}$$

where $q_i(i = 1, 2, \cdot, 8)$ are the homogeneous coefficients of the manifold surface represented by the above equation. In [21], we call this as a generalized (G-) manifold, which is capable of representing all types of mechanical dyads. For every pose, one such linear equation with unknowns as q_i is obtained. Assembling all the G-manifold equations for all the poses results in the following over-constrained homogeneous linear system on equation

$$Aq = 0, \qquad (3.31)$$

where

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & \cdots & \cdots & \cdots & A_{18} \\ A_{21} & A_{22} & A_{23} & A_{24} & \cdots & \cdots & \cdots & A_{28} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ A_{n1} & A_{n2} & A_{n3} & A_{n4} & \cdots & \cdots & \cdots & A_{n8} \end{bmatrix}, \tag{3.32}$$

and

$$q = \begin{bmatrix} q_1 & q_2 & \cdots & q_8 \end{bmatrix}^T \tag{3.33}$$

The elements of each row of the matrix A are given as

$$A_{i1} = Z_{i1}^2 + Z_{i2}^2, \tag{3.34}$$

$$A_{i2} = Z_{i1}Z_{i3} - Z_{i2}Z_{i4}, \tag{3.35}$$

$$A_{i3} = Z_{i2}Z_{i3} + Z_{i1}Z_{i4}, \tag{3.36}$$

$$A_{i4} = Z_{i1}Z_{i3} + Z_{i2}Z_{i4}, \tag{3.37}$$

$$A_{i5} = Z_{i2}Z_{i3} - Z_{i1}Z_{i4}, \tag{3.38}$$

$$A_{i6} = Z_{i3}Z_{i4}, \tag{3.39}$$

$$A_{i7} = Z_{i3}^2 - Z_{i4}^2, \tag{3.40}$$

$$A_{i8} = Z_{i3}^2 + Z_{i4}^2, \tag{3.41}$$

where i is the pose index ranging from $i = (1, 2, \cdots, n)$. The least square solution to this homogeneous system of equation can be found out using the singular value decomposition of the coefficient matrix A [34]. The right singular vectors corresponding to the smallest singular values are candidate solutions for the minimization problem. The subspace spanned by the three smallest singular value right-vectors represents a family of possible dyad solutions. However, for these dyads to make physical sense, the following extra constraints are required to be satisfied

$$\begin{aligned} q_1q_6 + q_2q_5 - q_3q_4 &= 0 \\ 2q_1q_7 - q_2q_4 - q_3q_5 &= 0 \end{aligned} \tag{3.42}$$

An analytical solution to the above reduces to a quartic equation, which can give zero, two, or four real dyad solutions. Complex solutions do not represent a physical dyad. Combining any of the two dyads results in a four-bar mechanism. For further details, see [21]. As a result, the path synthesis problem is solved and prospective solutions are generated. Using this motion synthesis algorithm also enables us to simultaneously carry out type and dimensional synthesis. However, that is not the focus of this work.

The above methodology for motion computation works for five or more poses when the system of equation is fully-constrained or over-constrained. To handle under-constrained cases, additional Motion Synthesis constraints (MOC) also called geometric constraints outlined in [19] are used. We note that the constraints being discussed here are the ones required to define the motion computation problem, which are different from the constraints discussed earlier in the context of MSP computation. We ensure that the context would make it clear which constraints are being discussed.

3.5 Unified synthesis algorithm

A unified synthesis algorithm to solve the Alt-Burmester problems has been summarized in Algorithm 2. It states that when the synthesis problem has zero poses, the Fourier descriptor based path synthesis algorithm as described by Wu et al. [24] is used. For all the other cases, mixed synthesis approach using Eq. (3.17) can be used to solve for four-bar mechanisms. It must be noted that except for the case where there are no poses, the synthesis calculates both type and dimensions.

A key advantage of the methodology outlined is that it can handle motion, path and mixed synthesis problems seamlessly. Various permutations of $(0, 1, \cdots, m)$ poses and $(0, 1, \cdots, n)$ path point problems are presented in Ta-

Algorithm 2: Algorithm for Unified Motion, Path and Mixed Synthesis

```
  Input: Path points and Poses
1 if n(Pose)=0 then
2     Calculate $T_k$ using Eq.(3.19)
3     Calculate the four-bar mechanism by solving a minimization
        problem in a four dimensional subspace as described by Wu et
        al. [24]
4 else
5     Calculate $T_k$ using Eq.(3.19)
6     Calculate MSP using Eq.(3.17)
7     Calculate the singular vectors using Eq.(3.31)
8     Calculate the dyads using Eq.(3.42)
9 end
  Output: Synthesized mechanism
```

ble 3.1. The legends in the table are MOC = Motion Synthesis constraint [19], MIC = Mixed Synthesis Constraint, FD = Fully Defined, and X=trivial or undefined. The * refers to conditions where a Fourier task curve with just four points needs to be fitted and would have unsymmetrical descriptors.

Table 3.1: Various Possibilities for Unified Motion, Path, and Mixed Synthesis Problem

		Path Points					
		0	1	2	3	4	n
Poses	0	X	X	X	X	FD*	FD
	1	X	X	X	2 MIC*	2 MIC	2 MIC
	2	X	X	1 MIC*	1 MIC	1 MIC	1 MIC
	3	2 MOC	1 MOC*	FD	FD	FD	FD
	4	1 MOC	FD	FD	FD	FD	FD
	5	FD	FD	FD	FD	FD	FD
	m	FD	FD	FD	FD	FD	FD

Motion Synthesis constraints can be used to specify the position of fixed or moving pivots using line or point constraints or any other compatible geometric constraint; see [19] for details. Mixed Synthesis Constraints, described earlier, involve constraints on actuating pivot, coupler dimensions, and other mechanism parameters. Fully Defined entails that no extra constraints are needed to exactly or least square solve the mixed synthesis Eq. (3.17). If either one of the MSP computation problem or Motion synthesis problem is under-constrained, the mixed synthesis problem is defined to be under-constrained. These under-constrained cases in Table 3.1 require additional constraints to be solved. In the table, zero-pose (or, pure-path synthesis) problems are solved using Wu et al. [24]. In that case, when four or more path-points are specified, the problem is fully defined and is non-trivial; however for $n = 4$, we can only calculate unsymmetrical descriptors. When only one pose is given, then two MIC are required to calculate all the MSP; with two poses, one MIC is required; and for three poses, no additional MIC are needed. However, for three poses, at least two path points need to be specified to obtain five poses needed for the motion synthesis algorithm. Burmester [15] showed that one needs five poses to solve a motion generation problem uniquely. In [18], we have extended Burmester problem to show that one can specify not only five poses, but a combination of pose and other geometric constraints to have unique mechanism design solutions. Therefore, for one path-point with three poses, we need one MOC and for zero path point, we need two MOC to get a total of five constraints. However, zero path-point problem reduces to a pure motion generation problem. We will illustrate some of these permutations and combinations in the examples next.

3.6 Examples

In this section, we present some examples to illustrate the effectiveness of the proposed algorithm. First example aims to validate the approach by extracting path-points and poses from a known mechanism. Second example solves mixed synthesis problem with fully-constrained MSP computation involving three poses and five path points. Third and fourth examples deal with under-constraint MSP computation and motion synthesis cases and require additional mixed- and motion-constraints, respectively. Demonstrating valid results from each of these cases proves the robustness of proposed algorithm. It also demonstrates the flexibility of the algorithm and its ability to incorporate various constraints. In the Figures 3.3, 3.4, 3.5, 3.6, 3.7 and 3.8, the blue and red curves denote the coupler curve in two possible assembly modes.

3.6.1 Example 1: Reverse Engineering a Mechanism

To validate the proposed mixed synthesis algorithm, points and poses from a known planar four-bar mechanism are taken and then our algorithm is used to synthesize mechanisms. Ideally, we should get the exact same mechanism. However, a similar mechanism is also acceptable since approximations occur at various steps – from task curve generation to algebraic fitting of the pose data.

A sample mechanism displayed in Fig. 3.3 is used to generate seven path-points and five poses. The mechanism has been defined using the position of its fixed pivots, moving pivots and coupler coordinates as shown in Table 3.2.

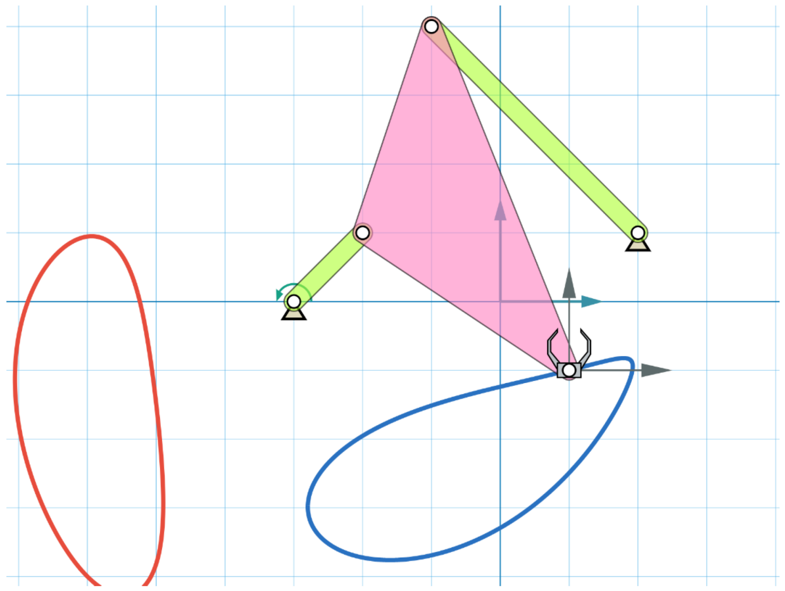


Figure 3.3: Example 1: Known target mechanism

Table 3.2: Example 1: Sample mechanism design parameters as shown in Fig. 3.3

Point	X	Y
Input link fixed pivot	-3.0	0.0
Input link moving pivot	-2.0	1.0
Output link fixed pivot	2.0	1.0
Output link moving pivot	-1.0	4.0
Coupler point	1.0	-1.0

The arbitrarily sampled poses and path points are listed in Table 3.3. This is a fully defined problem and all the MIC can be computed without requiring any additional information. These constraints are used as input to mixed synthesis algorithm. Four solution dyads are output as listed in Table 3.4. This also allows us to reverse-engineer a known mechanism since there are six planar four-bars that can satisfy the given constraints. Figure 3.4 shows a four-bar obtained by assembling dyads 1 and 4. It is observed that the mechanism generated is very similar to the original mechanism. This approximate result is due to the best-fitted low harmonic task curve following the original coupler curve closely but not exactly. The average path error measured by calculating the deviation of input points from final path using the Euclidean distance is 0.0694 units for the displayed configuration. The maximum angular deviation among all the given poses is for the pose 3 as 0.0736 rad.

Table 3.3: Example 1: Input data

No.	Type of Data	x	y	ζ (rad)
1	Point	0.350	−1.160	
2	Point	−0.410	−1.340	
3	Pose	−1.585	−1.737	5.853
4	Point	−2.110	−2.030	
5	Point	−2.800	−2.970	
6	Pose	−2.216	−3.665	5.896
7	Point	−0.420	−3.500	
8	Point	0.910	−2.580	
9	Pose	1.520	−1.832	0.351
10	Pose	1.912	−1.036	0.385
11	Point	1.560	−0.860	
12	Pose	1.000	−1.000	0.000

Table 3.4: Example 1: Output dyad data

Dyad	Fixed Pivot	Moving Pivot	Coupler Point
1	−2.793, −0.554	−2.056, 0.799	0.350, −1.160
2	−0.172, 6.978	1.340, 8.356	0.350, −1.160
3	−18.181, 11.280	−4.226, 6.257	0.350, −1.160
4	1.625, 0.824	0.081, 3.477	0.350, −1.160

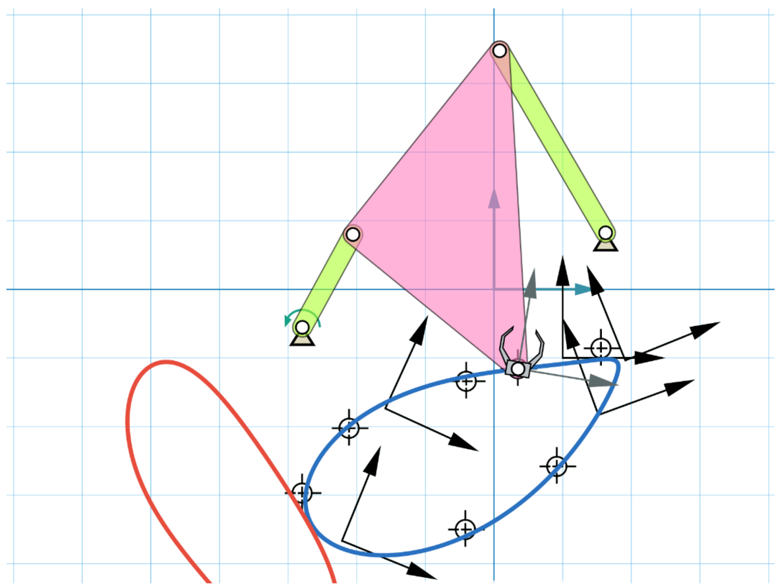


Figure 3.4: Example 1: Mechanism generated using mixed synthesis algorithm

In this example, the input data consisted of five poses and seven path points. Greater than three input poses over-constraints the MSP computation due to which the path-orientation relationship is satisfied using SVD. Thus, this example demonstrates mixed synthesis with over-constrained MSP computation.

3.6.2 Example 2: Mixed synthesis with fully-constrained MSP computation

In this example, the input data consists of three poses and five path points which fully constrains the MSP computation problem. The data input to mixed synthesis algorithm is given in Table 3.5. The two dyads generated as output have been shown in Table 3.6. The final mechanism has been displayed in Fig 3.5. It can be observed that a good match has been established with the constraints. Average path error is 0.0226 units while the maximum angular deviation for poses is 0.0106 rad for second pose. Note that in this case, the path-orientation relationship has an exact solution, i.e. MSP are uniquely determined using SVD.

Table 3.5: Example 2: Input data

No.	Type of Data	x	y	ζ (rad)
1	Pose	-5.263	1.441	0.161
2	Point	-3.810	1.690	
3	Point	-2.890	1.590	
4	Point	-2.010	1.120	
5	Pose	-1.416	0.789	5.919
6	Point	-0.200	0.490	
7	Point	1.040	0.600	
8	Pose	2.206	1.203	0.405

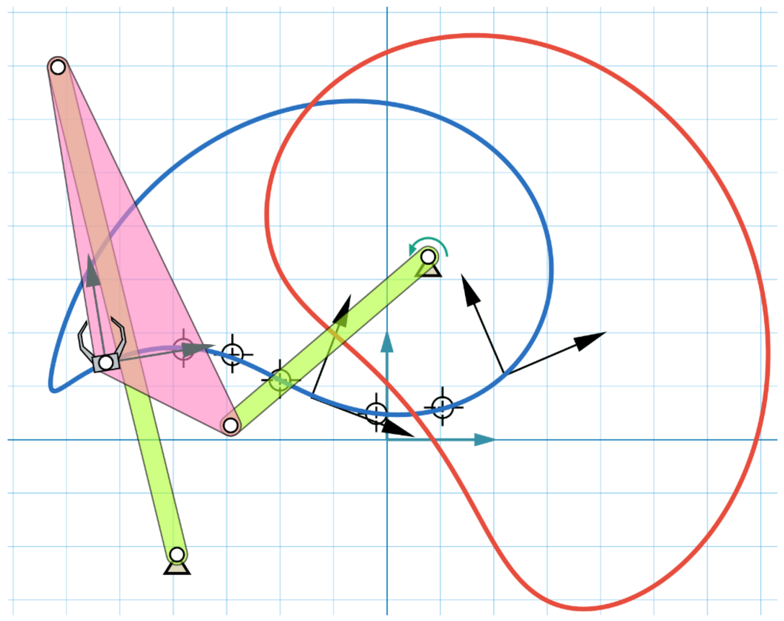


Figure 3.5: Example 2: Mixed synthesis with fully-constrained MSP computation for three poses and five path points

Table 3.6: Example 2: Output dyad data

Dyad	Fixed Pivot	Moving Pivot	Coupler Point
1	$0.771, 3.424$	$-2.927, 0.266$	$-5.263, 1.441$
2	$-3.931, -2.151$	$-6.160, 6.975$	$-5.263, 1.441$

One of the major advantages of mixed synthesis is the additional flexibility it imparts to users while specifying inputs and generating good solutions.

Using a pure motion synthesis algorithm, the user would have to input all the data as poses even if the problem demanded otherwise. This would lead to an over-constrained motion problem, which when solved using existing kinematic mapping based algebraic fitting approach [18, 19, 20, 21] usually produces poor solutions. A comparable motion synthesis problem for the same path points, but with orientations also given is displayed in Fig 3.6. It can be observed that the solution provides a poor fit to the given constraints. This happens because the orientation provided are not compatible with the motion of coupler of the planar-four linkages.

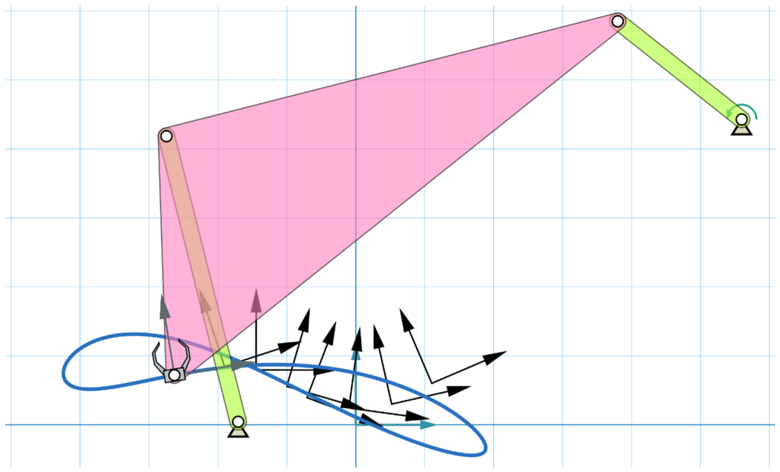


Figure 3.6: Example 2: Over-constrained motion synthesis for eight poses produces a poor solution.

3.6.3 Example 3: Mixed synthesis with under-constrained MSP computation using mixed constraints

This example shows mixed synthesis with under-constrained MSP computation problem where two poses and four path points are specified in the input. Lesser than three input poses makes MSP computation problem under-constrained. To solve for MSP, an additional mixed constraint is required which could specify any of z_1, z_2, z_3. The constraint data input to mixed synthesis algorithm is shown in Table 3.7. A MIC is used to specify z_2 by defining the preferred location of a fixed joint at point $(1, 3)$. The four dyads generated as output are shown in Table 3.8. One of the final mechanisms is displayed in Fig 3.7 using dyads 3 and 4. It can be observed that the generated mechanism closely satisfies path and mixed constraints. The average path error is 0.0948 units while the maximum angular deviation for poses is 0.0389 rad for the second pose in the displayed mechanism. Note that in this case, the path-orientation relationship has an infinite solutions and the use of MIC restricts the solution space to a unique solution to the MSP.

Table 3.7: Example 3: Input data

No.	Type of Data	x	y	ζ (rad)
1	Pose	4.962	-0.514	0.134
2	Point	3.850	-1.480	
3	Point	1.920	-0.740	
4	Point	0.850	0.760	
5	Point	3.360	1.650	
6	Pose	4.900	1.178	0.510

Table 3.8: Example 3: Output dyad data

Dyad	Fixed Pivot	Moving Pivot	Coupler Point
1	$8.705, 10.738$	$8.705, 10.738$	$4.962, -0.514$
2	$-7.961, 8.082$	$-3.450, 6.381$	$4.962, -0.514$
3	$0.973, 3.179$	$1.004, -0.295$	$4.962, -0.514$
4	$4.748, 0.792$	$6.635, -0.004$	$4.962, -0.514$

3.6.4 Example 4: Mixed synthesis with under-constrained motion synthesis using motion constraints

This example shows mixed synthesis with under-constrained motion synthesis problem where three poses and one path point is specified in the input. Motion synthesis is under-constraint because the total pose and path con-

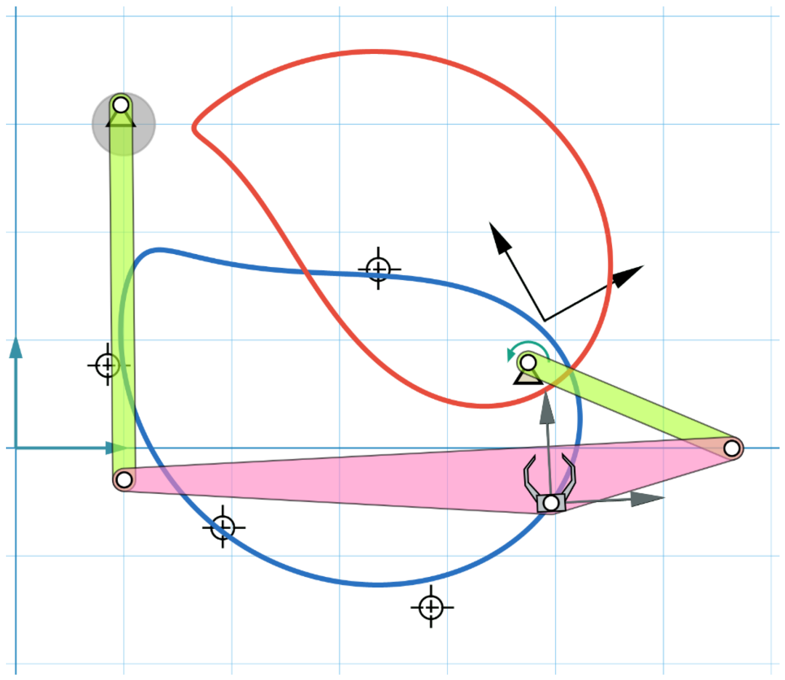


Figure 3.7: Example 3: Under-constrained mixed synthesis for two poses and four path points using additional mixed constraint

straints are just four. Even though three poses specified can be used to calculate the MSP, an additional motion constraint is required to solve the motion synthesis problem. The data input to mixed synthesis algorithm is given in Table 3.9. A line constraint is used as MOC in the example presented. The line segment is defined by its end points $(-4, 1)$ and $(1, 4)$. The four dyads generated as output are shown in Table 3.10. One of the final mechanisms is displayed in Fig 3.8 using dyads 1 and 2. It can be observed that the generated mechanism closely satisfies path constraints. The average path error is 0.0026 units while the maximum angular deviation for poses is 0.0005 rad for second pose in the displayed mechanism. Also, both the fixed pivots fall on the line constraint specified. Thus, the synthesis problem is successfully solved. Note that in this case, it is not the path-orientation relationship that is under-defined but the algebraic fitting algorithm which requires at-least five poses to be fully defined.

Table 3.9: Example 4: Input data

No.	Type of Data	x	y	ζ (rad)
1	Pose	-2.018	-1.391	0.146
2	Pose	0.288	-1.115	0.287
3	Point	1.700	0.360	
4	Pose	2.895	1.253	1.487

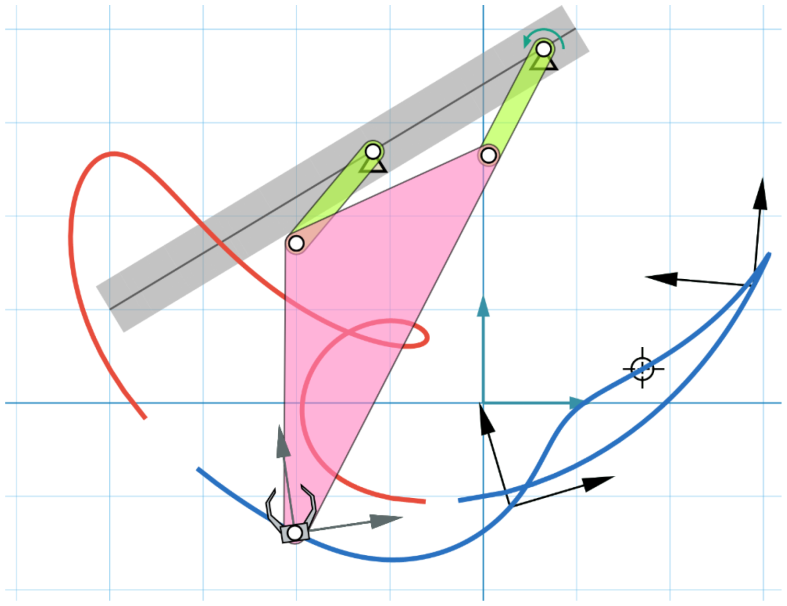


Figure 3.8: Example 4: Under-constrained mixed synthesis for three poses and one path points using additional motion constraint

Table 3.10: Example 4: Output dyad data

Dyad	Fixed Pivot	Moving Pivot	Coupler Point
1	$0.647, 3.788$	$0.058, 2.651$	$-2.018, -1.391$
2	$-1.181, 2.692$	$-2.002, 1.708$	$-2.018, -1.391$
3	$-3.322, 1.407$	$-4.111, 3.061$	$-2.018, -1.391$
4	$-3.322, 1.407$	$-4.111, 3.061$	$-2.018, -1.391$

3.7 Conclusion

In this chapter, we have presented a generalized m pose, n path-point mixed synthesis approach for four-bar mechanisms. Original contributions of this chapter include the closed-form relationship between coupler orientation and coupler path and exploiting this relationship to present a novel framework for solving the mixed synthesis problem. Another novel feature is the use of task-driven motion synthesis algorithm within the framework to keep the computation cost at minimum and perform simultaneous type and dimensional synthesis. A few examples were presented to demonstrate the effectiveness of the approach.

Chapter 4

Using a Point-Line-Plane Representation for Unified Simulation of Planar, Spherical, and Spatial Mechanisms

4.1 Introduction

Kinematic simulation of a mechanism requires calculation of the position and orientation of all of its constituent links during its entire range of motion. Simulation methodologies can be broadly classified into three categories: graphical, analytical and numerical [35]. The graphical analysis method is based on dyadic decomposition, i.e., identification of four-bar loops in mechanisms [36]. Although this approach is prominently used in simulation packages like Linkages [37] and PMKS [38], its limitations are well known [39]; e.g., they are unable to handle complex mechanisms like a double butterfly mechanism. Analytical methods involve solving a loop closure constraint-based system of non-linear equations [40]. Most analytical methods use the Polynomial continuation method [41, 42], elimination method or Grobner bases [43] to solve the simulation problem. These methods are able to find all the possible assembly configurations of a given mechanism. However, they are not general in nature since the motion equations need to be derived for each type of mechanism separately. As a result, these approaches cannot be used to simulate n-bar planar or spherical mechanisms without manually deriving equations on a case by case basis.

Numerical simulation methods are iterative in nature and can handle extremely complex mechanism [44, 45]. They use numerical methods like the Newton-Raphson method to solve the system of non-linear equations for one solution only instead of all possible ones [46]. These methods accept the mechanism joints and link information as inputs. Subsequently, the algorithm repeatedly solves the finite displacement problem, i.e., the input link is iteratively moved with finite displacement and consequently, positions of remaining links are calculated. As a result, the entire range of motion for the specified mechanism is calculated.

Hernández and Petuya have proposed a **geometrical-iterative method** which performs better than Newton-Rhapson method [47]. However, the approach is limited to n-bar planar mechanism with revolute joints only. Radhakrishnan and Campbell [48] have created a computational tool for planar mechanism, which carries out position analysis of planar mechanisms using a geometrically iterative algorithm. However, due to the use of dyadic decomposition, it shares the limitations of graphical methods and is limited to the planar mechanisms.

Commercial CAD software like Autodesk Inventor, Solidworks, ADAMS, etc. solve differential-algebraic equations numerically to provide multi-body simulation capability [49, 50, 51, 52]. However, their use is more prominent during detail design stage rather than the conceptual design stage. Creation of feature-based assembly of planar and spherical mechanisms and initializing constraints on these systems is a nontrivial task. Changing the type of joints or the number of links for a mechanism is also more involved than carrying out the same operation on purely kinematic simulators like PMKS [38]. Additionally, their solvers model the motion problem as a set of coupled differential and algebraic equations. This type of model is more suited for dynamic simulations rather than kinematic simulations, which involves purely algebraic constraints.

Also, the algebraic equations for commercial softwares are modeled using reference point representation which leads to more number of constraints when compared to other representations. Thus, use of these softwares for concept design is not ideal. SAM and GIM are two more packages which supports n-bar simulation for planar linkages with both revolute and prismatic joints [53, 54]. Furlong et al. [55] have demonstrated a virtual reality environment for simulating spherical four-bar mechanisms and in the academic domain, SPHINX, ISIS and OSIRIS are softwares which enable the analysis and synthesis of spherical mechanisms [56, 57, 58].

However, currently there are no approaches which unify n-bar planar and spherical mechanism analysis and can be demonstrated to more complex linkage systems. The proposed approach hopes to bridge this gap and augment the capability of pure kinematic design systems like MotionGen [18]. In this chapter, planar-and spherical-mechanisms are represented as a collection of geometric constraints spanned by points, lines, and planes. The geometric mechanism representation enables the design of unified constraint equations which are easily programmed. As a result, a simple generalized real-time framework for mechanism simulation is achieved.

Our previous work on mechanism synthesis problems includes the creation of geometric constraint equations for four-bar mechanisms with revolute or prismatic joints [21, 59, 60, 61, 62]. In [63], we have demonstrated an algebraic fitting based approach in the space of planar quaternions to simulate planar four-bar linkages. However, that approach does not scale for more complex planar or spherical linkages. In this chapter, we show that by using homogeneous coordinates, we can derive unified geometric constraint equations for both planar and spherical linkages, which simplifies the simulation without resorting to calculations for individual types of mechanisms. The rigidity constraints imposed by the links are modeled as simple geometric constraints using points, lines and planes. Once the mechanism is specified, the solver proceeds with iteratively perturbing the input and solving the constraints for other links. To the best of authors' knowledge, this work is the first attempt at using a point-line-plane mechanism representation and presenting unified geometric constraints for simulation.

The major intellectual contributions of this chapter can be summarized as 1) presenting generalized constraint equations for planar and spherical mechanisms using point-line-plane representation and 2) enabling real-time simulation of n-bar planar or spherical mechanisms.

Rest of the chapter is organized as follows. Section 4.2 discusses the representation and constraints required to describe the motion of a general planar and spherical mechanism. Section 4.3 demonstrates the algorithm required to simulate a mechanism using the iterative numerical approach. Finally, in Section 4.4, we present a few examples to demonstrate the use of proposed algorithm.

4.2 Mechanism Representation and Constraints

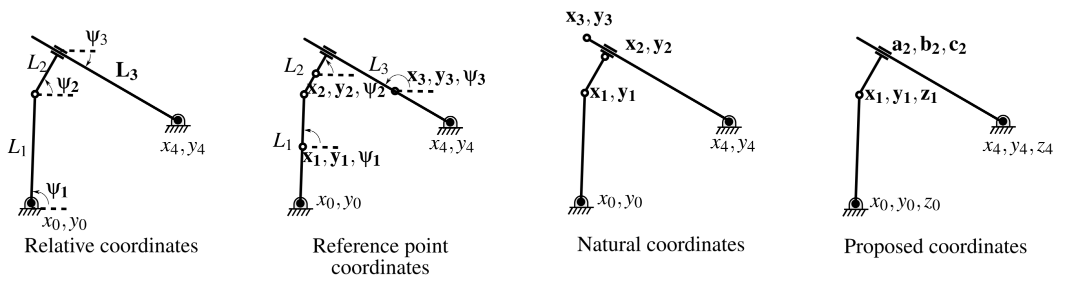


Figure 4.1: Different types of mechanism representations

Selection of an apt mechanism representation and constraints is important as it has a profound effect on algorithm's simplicity and efficiency. Conventionally, a multi-body system has been specified using multiple representations namely: relative coordinates, reference point coordinates, and natural coordinates [44]. Relative coordinates are based on parameters specifying one link relative to another; reference point coordinates are based on specifying absolute position of each link independently; while the natural coordinates are based on each link being specified by two points. Using relative coordinates enables a scalable representation while reference point coordinates tend to be more computationally efficient. Natural coordinates provide a compromise between the two approaches in terms of simplicity and efficiency. Most commercial softwares use reference point coordinate representation which usually leads to maximum number of constraint equations and subsequently high computation time.

Figure 4.1 shows an RRPR (R: Revolute, P: Prismatic) four-bar mechanism and its specification using different representations. For the relative coordinate representation, there are three unknown coordinates i.e. ψ_1, ψ_2, L_3. For the reference point coordinate representation, the mechanism has nine unknown variables i.e. location and orientation of each link x_i, y_i, ψ_i. Similarly, for the natural coordinate representation, there are six unknown variables namely $x_1, y_1, x_2, y_2, x_3, y_3$. Since the four-bar mechanisms are a single degree of freedom mechanisms, each of the representation requires two, eight and five constraint equations, respectively to fully define the motion. In this chapter, we derive unified constraint equations for all types of planar and spherical linkages consisting of both revolute and prismatic joints. We use homogeneous coordinates to write geometric constraints on points, lines, and planes. For example, our representation for the shown RRPR mechanism will require using four unknown point and line coordinates i.e. x_1, y_1, a_2, b_2 since we can set homogenizing factors $z_1 = 1$ and $c_2 = 1$ without any loss in generality. Such a representation would keep the number of unknown variables smaller while also enabling construction of simpler geometric constraint equations. Computational efficiency aside, this representation also naturally maps to the geometric constraints, which for the shown RRPR mechanism are a circle-constraint on the moving pivot and a line-constraint on the fixed pivot of the RP dyad. It is well known that the time complexity of multidimensional Newton-Raphson method, which is used in this chapter, is at-least $O(n^2)$ for a single iteration where n is the number of unknowns in state variable [45]. This is a direct consequence of a dense Jacobian matrix having n^2 elements that need to be calculated after every iteration. Thus, more unknowns result in needing to calculate larger Jacobian matrices which is computationally expensive.

Planar mechanisms can be uniquely specified using their joint and link data. A joint can be prismatic or revolute, which naturally associates with points and lines, respectively. We use homogeneous coordinates to represent both points and lines. Thus, a point P is given by homogeneous coordinates (x, y, z) whose Affine coordinates are given as $(x/z, y/z)$, while a line L is also represented using homogeneous coordinates (a, b, c), where equation of the line passing through the point P in the projective plane is given by $ax+by+cz = 0$. Depending on the constraint being expressed, this line can be fixed or floating in the plane. A link can be represented by a subset of joints. The link can be binary, ternary or n-ary depending on the number of joints it contains. An example six-bar planar mechanism is displayed in Fig. 4.2. Its joints are represented as points and lines while its links are defined as a group of joints as shown in Table 4.1.

Similarly, spherical mechanisms can also consist of revolute and prismatic joints. A spherical prismatic joint constrains the link movement along a circular arc instead of a line. We represent a spherical revolute joint as a point P in terms of its homogeneous coordinates (x, y, z, w) with respect to the

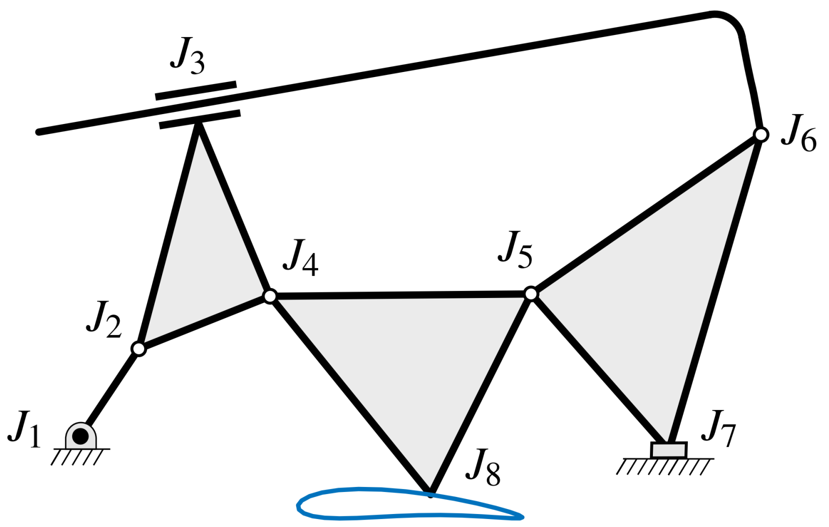


Figure 4.2: Planar Stephenson II six-bar linkage

center of the unit sphere such that its Affine coordinates are $(x/w, y/w, z/w)$. A spherical prismatic joint is defined as a plane $Pl : (a, b, c, 0)$ passing through the center of the sphere and is given by the equation $ax + by + cz = 0$. The intersection of the plane and the unit sphere defines the great circle along which the constituent links are constrained to move for a spherical prismatic joint. Similar to the lines for planar mechanisms, this plane can be fixed or moving depending on the geometric constraint being expressed. An example RRPR spherical mechanism is displayed in Fig. 4.3. Its joints are represented as points and planes while its links are defined as a group of joints as shown in Table 4.2.

During the motion, a mechanism is subjected to a set of constraints imposed by the rigidity of its links. Thus, to simulate a mechanism, these constraint equations need to be formulated. For planar and spherical mechanisms, modeling three constraint equations are sufficient for simulation. We propose a unique constraint equation for each of the binary links RR, RP, PR, and PP. But, we will see that all of these constraints can be expressed in a single equation. Any link with more than two joints can easily be reduced to

Table 4.1: Joint and Link data for Stephenson II linkage using Affine Coordinates

Joint	Type	Coordinates
$J_{1,input}$	Revolute	0, -1
J_2	Revolute	1, .5
J_3	Prismatic	-0.17, 0.98, -4.28
J_4	Revolute	3.25, 1.4
J_5	Revolute	7.72, 1.44
J_6	Revolute	11.66, 4.17
J_7	Prismatic	0, 1, 1.24
J_8	Coupler point	6, -2

Link	Constituent joints
L_1	J_1, J_2
L_2	J_2, J_3, J_4
L_3	J_3, J_6
L_4	J_4, J_5, J_8
L_5	J_5, J_6, J_7
$L_{6,ground}$	J_1, J_7

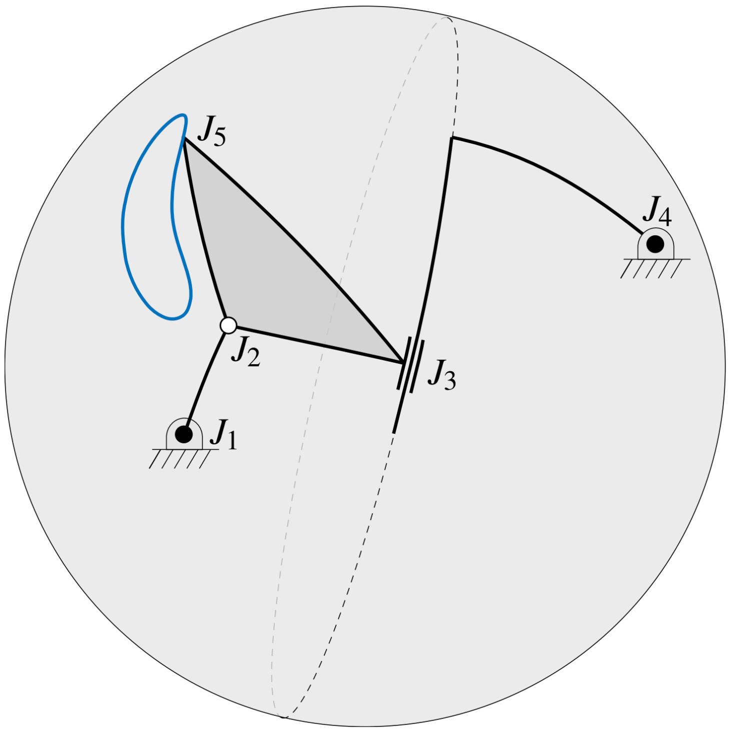


Figure 4.3: Spherical RRPR four-bar linkage

an equivalent collection of binary links. For example, a ternary link can be treated as three binary links. Thus, these constraints can successfully be used to enforce the rigidity of any link in a general mechanism. Figures 4.4 and 4.5 show different planar and spherical binary links, which are building blocks for any planar and spherical mechanism and are discussed below.

The first general constraint enforces the rigidity of a spherical binary

Table 4.2: Joint and Link data for Spherical RRPR linkage using Affine Coordinates; the coordinates are given with respect to the fixed coordinate frame located at the center of the reference sphere.

Joint	Type	Coordinates
$J_{1,input}$	Revolute	0.94, 0.24, 0.24
J_2	Revolute	0.80, 0.27, 0.53
J_3	Prismatic	0.68, -0.68, 0.26
J_4	Revolute	-0.38, 0.76, 0.53
J_5	Coupler point	0.50, -0.21, 0.84

Link	Constituent joints
L_1	J_1, J_2
L_2	J_2, J_3, J_5
L_3	J_3, J_4
$L_{4,ground}$	J_1, J_4

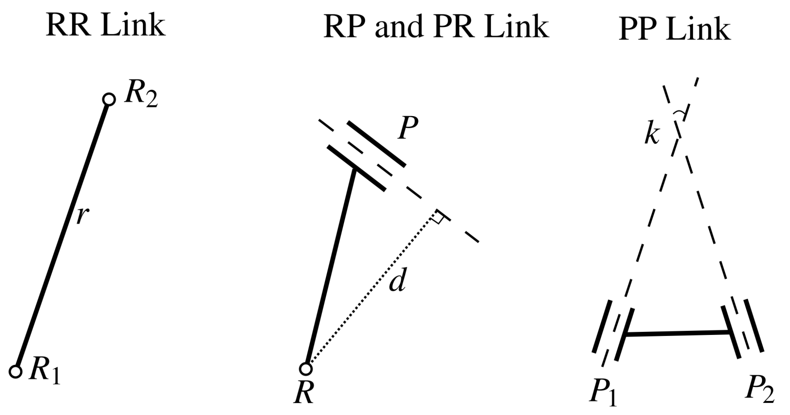


Figure 4.4: Types of binary planar links

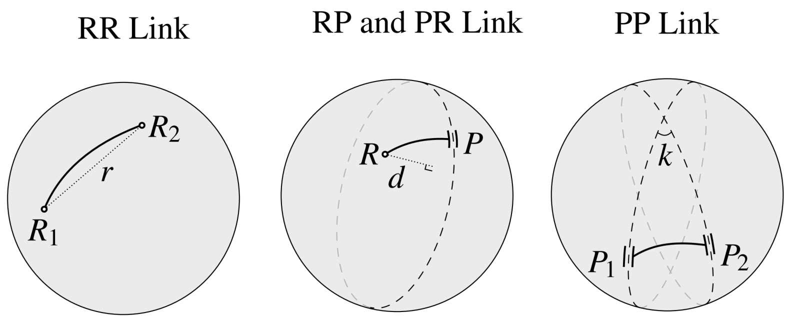


Figure 4.5: Types of binary spherical links

link with two revolute joints represented by two homogeneous point coordinates of the fixed point (a_1, a_2, a_3, a_4) and floating point (b_1, b_2, b_3, b_4), where a_4 and b_4 are homogenizing factors. The RR link imposes the constraint that the distance between two points remains constant, i.e., $dist(R_1, R_2) = r$ in

Figures 4.4 and 4.5. The constraint equation is given as

$$\begin{aligned} C_{S,RR} : 2a_1b_1 + 2a_2b_2 + 2a_3b_3 + a_0b_4 = \\ a_4\left(\frac{b_1^2 + b_2^2 + b_3^2}{b_4}\right), \end{aligned} \tag{4.1}$$

where a_0 is given as

$$a_0 = a_4r^2 - \frac{a_1^2 + a_2^2 + a_3^2}{a_4}. \tag{4.2}$$

Here, r is the radius of the sphere formed by the RR link with the center given by (a_1, a_2, a_3, a_4). When the z-coordinate is set to zero, the constraint equation degenerates into the one for a planar RR link. The constraint equation for a planar RR link represented by points (a_1, a_2, a_4) and (b_1, b_2, b_4) is thus given as

$$C_{P,RR} : 2a_1b_1 + 2a_2b_2 + a_0b_4 = a_4\left(\frac{b_1^2 + b_2^2}{b_4}\right), \tag{4.3}$$

where a_0 is given as

$$a_0 = a_4r^2 - \frac{a_1^2 + a_2^2}{a_4}. \tag{4.4}$$

The second general constraint enforces the rigidity of a binary link with one prismatic and one revolute joint represented by a homogeneous point and a plane given by (a_1, a_2, a_3, a_4) and (L_1, L_2, L_3, L_4), respectively. An RP or PR link imposes the constraint that the distance between a point and a line (planar case) or a point and a plane (spherical case) is constant, i.e., $dist(R, P) = d$ in Figures 4.4 and 4.5. RP and PR links are inversions of each other and are expressed by the same constraint. The general constraint equation for a spherical RP link is given as

$$C_{S,RP} : a_1L_1 + a_2L_2 + a_3L_3 + a_4L_4 = da_4\sqrt{L_1^2 + L_2^2 + L_3^2}, \tag{4.5}$$

where d is the signed perpendicular distance between the revolute joint and prismatic joint. For spherical linkages, the prismatic plane always passes through the origin, i.e. $L_4 = 0$. Thus, the spherical RP link constraint equation reduces to

$$C_{S,RP} : a_1 L_1 + a_2 L_2 + a_3 L_3 = d a_4 \sqrt{L_1^2 + L_2^2 + L_3^2}. \tag{4.6}$$

When the z-coordinates is set to zero, the general constraint equation degenerates into a planar case. Thus, constraint equation for a planar RP or PR link represented by a point (a_1, a_2, a_4) and a line (L_1, L_2, L_4) is given as

$$C_{P,RP} : a_1 L_1 + a_2 L_2 + a_4 L_4 = d a_4 \sqrt{L_1^2 + L_2^2}. \tag{4.7}$$

When the perpendicular distance d becomes zero, the equation describes a line passing through a point, i.e. constraint equation of PR or RP links. This is usually the case with the PR links where the moving joint point is constrained to be on the fixed line of the prismatic joint and in case of the RP link where the moving line is constrained to pass through the point of the fixed joint.

Finally, the third constraint enforces the rigidity of a spherical binary link with two prismatic joints represented as $(L_1, L_2, L_3, 0)$ and $(M_1, M_2, M_3, 0)$. For the PP link, the angle between two lines (planar case) or two planes (spherical case) remains constant, i.e., $angle(P_1, P_2) = \cos^{-1}(k)$ in Figures 4.4 and 4.5.The constraint equation is given as

$$C_{S,PP} : L_1 M_1 + L_2 M_2 + L_3 M_3 = k \sqrt{L_1^2 + L_2^2 + L_3^2} \sqrt{M_1^2 + M_2^2 + M_3^2}, \tag{4.8}$$

where k represents the cosine of angle between two prismatic joints. Similarly, for planar PP binary link represented by two lines (L_1, L_2, L_4) and (M_1, M_2, M_4), the constraint equation degenerates to

$$C_{P,PP} : L_1 M_1 + L_2 M_2 = k \sqrt{L_1^2 + L_2^2} \sqrt{M_1^2 + M_2^2}. \tag{4.9}$$

When the two prismatic joints on a binary link are defined as two parallel lines, a degree of freedom is added to the mechanism. This situation is impractical and will not been considered further in this chapter.

It can be seen that the Eqns. 4.3, 4.6, 4.7, 4.8, 4.9 are all degenerate case of the Eq. 4.1. In the projective plane for the planar geometric constraints, the lines and points are dual to each other; thus, their meanings can be interchanged without changing the underlying structure of the equations. In the projective three-space for the spherical constraints, the points and planes are dual to each other and thus their meanings can be interchanged. Thus, the Eq. 4.1 is the single equation that unifies all the geometric constraints associated with all types of links for both planar and spherical mechanisms. This facilitates creation of the following metrics for computation: 1) distance between two points in space, 2) perpendicular distance between a point and a plane, and 3) angle between two planes.

For links with prismatic joints, the line or plane coordinates are homogeneous in nature, i.e. multiplying a non-zero scalar λ to prismatic coordinates (L_1, L_2, L_3) does not change the coordinates. Thus, the magnitude of this vector can be fixed to unity without losing generality and another constraint can be written as

$$C_P : \quad L_1^2 + L_2^2 + L_3^2 - 1 = 0. \tag{4.10}$$

For spherical mechanisms, an additional geometric constraint is imposed on the joints due to the spherical nature of the motion. It is assumed that all the revolute joints move on the unit sphere which leads to the constraint

$$C_{S,R} : \quad a_1^2 + a_2^2 + a_3^2 - 1 = 0, \tag{4.11}$$

where (a_1, a_2, a_3) are the coordinates of any revolute joint on a spherical mechanism. Thus, the rigidity constraints described in Eqs. (4.1), (4.3), (4.6), (4.7), (4.8), (4.9), (4.10) and (4.11) are sufficient to uniquely determine the unknown coordinates of a n-bar planar or spherical mechanism. This concludes our discussion on representation and constraints for a generalized planar or spherical mechanism.

4.3 Solving Constraint Equations

In this section, we discuss the algorithmic steps required to solve the kinematic simulation problem. The general approach is to iteratively perturb the input links by a finite displacement and find the new position of the mechanism.

4.3.1 Input link perturbation

The simulation process involves iteratively perturbing the input link by a finite displacement. Depending on the actuating joint being revolute or prismatic, the displacement could be translation or rotational in nature. In this chapter, we restrict ourselves to consider actuation at the fixed joints. The relations governing the motion of input link are derived in this subsection.

For a perturbed RR link with the actuating fixed joint (x_1, y_1) and moving joint (x_2, y_2), the new coordinates of moving revolute joint can be given as

$$\begin{bmatrix} X_2 \\ Y_2 \\ 1 \end{bmatrix} = [\mathbf{T}]^{-1}[\mathbf{R}][\mathbf{T}] \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}, \tag{4.12}$$

where

$$[\mathbf{R_x}] = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \text{and} \quad [\mathbf{T}] = \begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -y_1 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.13}$$

In the above equation, (X_2, Y_2) represent the moving joint after perturbation and θ is the angle through which the input link is perturbed.

For a perturbed RP link with the actuating fixed joint (x_1, y_1) and moving joint (a, b, c), the new coordinates of moving line representing the prismatic joint can be given as

$$\begin{bmatrix} A \\ B \\ C \end{bmatrix} = ([\mathbf{T}]^{-\mathbf{1}}[\mathbf{R}][\mathbf{T}])^{-\mathbf{T}} \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \tag{4.14}$$

where (A, B, C) are the moving line coordinates after perturbation, and $\mathbf{T}$ and $\mathbf{R_x}$ are the translation and rotation matrices as described in Eqs. (4.13).

For a planar mechanism with the actuation being at prismatic joint, input link perturbation causes translation of other joints on the input link. For a perturbed PR link with the actuating joint (a, b, c) and moving joint (x, y), the new coordinates of translating revolute joint can be given as

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{b}{\sqrt{a^2+b^2}}d \\ \frac{-a}{\sqrt{a^2+b^2}}d \\ 0 \end{bmatrix} + \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{4.15}$$

where (X, Y) are the moving joint coordinates after perturbation and d is the distance through which the prismatic joint is moved along the fixed line. It can be seen that in Eq. (4.15) the actuating line coordinate c doesn't effect the new position of moving joint coordinates as new position only depends on direction cosines.

For a perturbed PP link with the actuating fixed joint (a_1, b_1, c_1) and moving joint (a_2, b_2, c_2), the new coordinates of translating prismatic joint can be given as

$$\begin{bmatrix} A_2 \\ B_2 \\ C_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{a_1b_2-a_2b_1}{\sqrt{a_1^2+b_1^2}}d \end{bmatrix} + \begin{bmatrix} a_2 \\ b_2 \\ c_2 \end{bmatrix} \tag{4.16}$$

where (A_2, B_2, C_2) are the moving prismatic joint coordinates after perturbation and d is the distance through which the input link has been perturbed.

Similarly, relationships determining the values of perturbed joints for spherical mechanisms can also be calculated. For spherical mechanisms with a fixed revolute actuating joint, the moving joints rotate around the axis passing through the actuation joint and the centre of sphere. The transformation matrix which rotates spherical link around an axis passing through the centre of sphere $(0, 0, 0)$ and an arbitrary point on surface of the sphere (l, m, n) is given by

$$[\mathbf{R}]_{(l,m,n)} = [\mathbf{R_x}]^{-\mathbf{1}}[\mathbf{R_y}]^{-\mathbf{1}}[\mathbf{R_z}][\mathbf{R_y}][\mathbf{R_x}] \tag{4.17}$$

$$[\mathbf{R_x}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{n}{\sqrt{m^2+n^2}} & \frac{-m}{\sqrt{m^2+n^2}} \\ 0 & \frac{m}{\sqrt{m^2+n^2}} & \frac{n}{\sqrt{m^2+n^2}} \end{bmatrix} \tag{4.18}$$

$$[\mathbf{R_y}] = \begin{bmatrix} \sqrt{m^2+n^2} & 0 & -l \\ 0 & 1 & 0 \\ l & 0 & \sqrt{m^2+n^2} \end{bmatrix} \tag{4.19}$$

$$[\mathbf{R_z}] = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.20}$$

where $\mathbf{R_x}$, $\mathbf{R_y}$, $\mathbf{R_z}$ are the rotation matrix around x,y and z axis and θ is the angle by which the link is rotated around the axis.

Using Eq. (4.17), the new coordinates of the moving joints of a perturbed spherical RR link can be given as

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = [\mathbf{R}]_{(x_1,y_1,z_1)} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} \tag{4.21}$$

where (x_1, y_1, z_1) are the fixed joint coordinates, (x_2, y_2, z_2) are the moving joint coordinates before perturbation and (X_2, Y_2, Z_2) are the moving joint coordinates after perturbation.

For a spherical RP link with a fixed revolute joint, the coordinates of moving prismatic joint can be given as

$$\begin{bmatrix} A \\ B \\ C \end{bmatrix} = [\mathbf{R}]_{(x,y,z)} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{4.22}$$

where (x, y, z) are the fixed joint coordinates, (a, b, c) are the moving joint coordinates before perturbation, (A, B, C) are the moving joint coordinates after perturbation, and as described above.

When the actuation joint is prismatic in nature, the moving joints translate on the intersection of a parallel plane and the unit sphere. This motion can also be characterized as rotation around an axis which passes through the centre of the sphere and 'pole' of the prismatic joint. The poles of a great circle are defined as intersection of two circles perpendicular to the initial circle. If a spherical prismatic joint is defined as a plane (a, b, c), its pole coordinates are also given as (a, b, c). Thus, for a spherical RP link with fixed prismatic joint, the coordinates of moving revolute joint can be given as rotation i.e.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = [\mathbf{R}]_{(a,b,c)} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{4.23}$$

where (a, b, c) are the fixed prismatic joint coordinates, (x, y, z) are the moving revolute joint coordinates before perturbation and (X, Y, Z) are the moving revolute joint coordinates after perturbation.

For a spherical PP link with a fixed prismatic joint, the coordinates of a moving prismatic joint can be given as

$$\begin{bmatrix} A_2 \\ B_2 \\ C_2 \end{bmatrix} = [\mathbf{R}]_{(a_1,b_1,c_1)} \begin{bmatrix} a_2 \\ b_2 \\ c_2 \end{bmatrix} \tag{4.24}$$

where (a_1, b_1, c_1) are the coordinates of fixed prismatic joint, (a_2, b_2, c_2) are moving prismatic joint coordinates before perturbation and (A_2, B_2, C_2) are moving prismatic joint coordinates after perturbation.

With these expressions, we can successfully calculate the location of input link after imparting it a discrete perturbation. The next step is to find the coordinates of all the other unknown joint coordinates which are compatible with the rigidity constraints imposed on the mechanism during simulation.

4.3.2 Numerical nonlinear system of equation solving

For any multi-body system, the position problem is always based on solving a system of constraint equations. This set of equations can be represented as

$$\mathbf{\Phi}(\mathbf{q}) = 0 \tag{4.25}$$

where $\mathbf{q}$ is the state vector which consists of all the unknown coordinates. The well-known Newton-Raphson method can be used to solve this nonlinear system of equation. It is featured by quadratic convergence in the neighborhood of the solution. Since the input link is perturbed by a small finite displacement, the previous state of mechanism serves as a good initial approximation. The number of constraint equations should be equal to or greater than the number of unknowns for this approach to work. For planar and spherical mechanisms, it is always possible to satisfy this criterion using the constraints outlined in section.

The iterative algorithm followed can be defined as

$$\mathbf{q}_{i+1} = \mathbf{q}_i - [\boldsymbol{J}^{-1}(\mathbf{q}_i)]\mathbf{\Phi}(\mathbf{q}_i) \tag{4.26}$$

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

$$[\boldsymbol{J}(\mathbf{q})] = \begin{bmatrix} \frac{\partial\phi_1}{\partial q_1} & \frac{\partial\phi_1}{\partial q_2} & \cdots & \frac{\partial\phi_1}{\partial q_n} \\ \frac{\partial\phi_2}{\partial q_1} & \frac{\partial\phi_2}{\partial q_2} & \cdots & \frac{\partial\phi_2}{\partial q_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial\phi_m}{\partial q_1} & \frac{\partial\phi_m}{\partial q_2} & \cdots & \frac{\partial\phi_m}{\partial q_n} \end{bmatrix} \tag{4.27}$$

where m is the number of constraints and n is the number of unknown coordinates. Thus to calculate the Jacobian matrix, relations describing the first order partial derivatives of constraint equations are required.

The first order partial derivatives for spherical RR constraint given in Eq. (4.1) can be given as follows

$$\frac{\partial C_{RR}}{\partial a_1} = 2(a_1 - b_1) \tag{4.28}$$

$$\frac{\partial C_{RR}}{\partial a_2} = 2(a_2 - b_2) \tag{4.29}$$

$$\frac{\partial C_{RR}}{\partial a_3} = 2(a_3 - b_3) \tag{4.30}$$

Here, the homogeneous point coordinate a_0 and b_4 has been assumed as unity without loss in generality. For planar RR link, only $\frac{\partial C_{RR}}{\partial a_1}$ and $\frac{\partial C_{RR}}{\partial a_2}$ exists.

The first order partial derivatives for spherical RP constraint given in Eq. (4.6) can be given as follows

$$\frac{\partial C_{S,RP}}{\partial a_1} = L_1, \frac{\partial C_{S,RP}}{\partial a_2} = L_2, \frac{\partial C_{S,RP}}{\partial a_3} = L_3 \tag{4.31}$$

$$\frac{\partial C_{S,RP}}{\partial L_1} = a_1 - \frac{dL_1}{\sqrt{L_1^2 + L_2^2 + L_3^2}} \tag{4.32}$$

$$\frac{\partial C_{S,RP}}{\partial L_2} = a_2 - \frac{dL_2}{\sqrt{L_1^2 + L_2^2 + L_3^2}} \tag{4.33}$$

$$\frac{\partial C_{S,RP}}{\partial L_3} = a_3 - \frac{dL_3}{\sqrt{L_1^2 + L_2^2 + L_3^2}} \tag{4.34}$$

Here, the homogeneous point coordinate a_0 has been assumed as unity without loss in generality. For planar RR constraint given in Eq. (4.7), the first order differentials are

$$\frac{\partial C_{P,RP}}{\partial a_1} = L_1, \frac{\partial C_{P,RP}}{\partial a_2} = L_2, \tag{4.35}$$

$$\frac{\partial C_{P,RP}}{\partial L_1} = a_1 - \frac{dL_1}{\sqrt{L_1^2 + L_2^2}}, \tag{4.36}$$

$$\frac{\partial C_{P,RP}}{\partial L_2} = a_2 - \frac{dL_2}{\sqrt{L_1^2 + L_2^2}}, \tag{4.37}$$

$$\frac{\partial C_{P,RP}}{\partial L_4} = 1 \tag{4.38}$$

The first order partial derivatives for spherical PP constraint given in Eq. (4.8) can be given as follows

$$\frac{\partial C_{S,PP}}{\partial L_1} = M_1 - \frac{kL_1\sqrt{M_1^2 + M_2^2 + M_3^2}}{L_1^2 + L_2^2 + L_3^2} \tag{4.39}$$

$$\frac{\partial C_{S,PP}}{\partial L_2} = M_2 - \frac{kL_2\sqrt{M_1^2 + M_2^2 + M_3^2}}{L_1^2 + L_2^2 + L_3^2} \tag{4.40}$$

$$\frac{\partial C_{S,PP}}{\partial L_3} = M_3 - \frac{kL_3\sqrt{M_1^2 + M_2^2 + M_3^2}}{L_1^2 + L_2^2 + L_3^2} \tag{4.41}$$

These equations degenerate to planar case when $L_3 = 0$ and $M_3 = 0$ and $\frac{\partial C_{P,PP}}{\partial L_4} = 0$

The first order partial derivatives for homogeneous Prismatic joint constraint given in Eq. (4.10) can be given as follows

$$\frac{\partial C_P}{\partial L_1} = 2L_1, \quad \frac{\partial C_P}{\partial L_2} = 2L_2, \quad \frac{\partial C_P}{\partial L_3} = 2L_3 \tag{4.42}$$

The first order partial derivatives for unit circle revolute joint constraint given in Eq. (4.11) can be given as follows

$$\frac{\partial C_{S,R}}{\partial a_1} = 2a_1, \quad \frac{\partial C_{S,R}}{\partial a_2} = 2a_2, \quad \frac{\partial C_{S,R}}{\partial a_3} = 2a_3 \tag{4.43}$$

To automate the calculation of residual vector $\mathbf{\Phi}(\mathbf{q}_i)$ and the Jacobian matrix $[\boldsymbol{J}(\mathbf{q}_i)]$, the constraints are handled in a sequential manner. While creating the residual vector in our implementation, first the rigidity constraints for each link are calculated and then the constraints for joints are calculated.

Similarly, the Jacobian matrix is created in a column-first manner i.e. all the partial differential equations with respect to an unknown state variable are calculated before progressing to the next variable. The outlined method is just one way of calculating $\mathbf{\Phi}(\mathbf{q}_i)$ and $[\boldsymbol{J}(\mathbf{q}_i)]$ since their values are independent of the sequence adopted to calculate each element.

Thus, using the constraint equations and their first order partial derivatives, it is possible to solve iteratively for the solution using Newton-Raphson method. The iterations are continued until a solution within desired accuracy is calculated.

For some input link perturbations, the Newton-Raphson method might fail to converge even after many iterations. In these instances, there does not exist a mechanism state which fulfills all the constraint equations. As a result, this input perturbation is outside the possible limits of motion of mechanism.

Thus, by iteratively perturbing the input link and solving the constraints for other joint coordinates, we are able to simulate any general planar or spherical mechanism. Numerous techniques exist that can improve the convergence and efficiency of the Newton-Raphson method. However, the basic method suffices to achieve real-time simulation. The complete algorithm has been described in Algorithm 3.

4.4 Examples

This section presents sample examples to demonstrate the use of proposed algorithm for mechanism simulation. The simulation has been carried out in MATLAB on a PC running Core i5-7300 at 2.6GHz with 8GB RAM. The simulation is carried out within seconds for residual value of $1.0e-8$. Each closed-loop output curve is made up of 180 points while open-loop curves have less than 180. Each point corresponds to $\frac{2\pi}{180}$ radian or units input perturba-

Algorithm 3: Algorithm for planar and spherical mechanism simulation

```
   Input: Initial mechanism configuration
1  Calculate initial rigidity metric for each link
2  for range of input link motion do
3      Perturb input link
4      for max iterations do
5          Calculate the constraint residual
6          if residual $\leq \epsilon$ then
7              Solution found
8              break
9          end
10         Calculate the Jacobian matrix
11         Calculate predicted unknown joint coordinates
12     end
13     if solution found then
14         Store as subsequent mechanism state
15     else
16         Motion limit reached
17 end
18 Animate the range of motion
   Output: Mechanism simulation
```

tions.

4.4.1 Speed comparison with a commercial software

To compare the speed of commercially available CAD systems and proposed algorithm, a planar four-bar crank rocker mechanism with revolute joints is modeled and simulated. Autodesk Inventor 2020 with educational license is used as reference commercial CAD system. Simulation is performed for 180 time-steps with one degree input link perturbation for each time-step. The simulation takes 5s to complete Inventor while it finishes in 1.4s when the proposed algorithm is used. Thus, even for a simple mechanism, there is a significant speed difference between the commercial solvers and proposed methodology.

4.4.2 Planar Stephenson-II linkage

A planar Stephenson-II six-bar linkage is simulated in this example. This linkage does not have a four-bar linkage and proves to be challenging to simulate using dyadic decomposition based approaches. However, our approach handles these non-dyadic mechanisms without issues.

The six-bar mechanism is displayed in Fig. 4.2 and its joint and link data is given in Table 4.1. The mechanism has J_1, J_7 as the fixed joints, J_2 as the perturbed joint and J_3, J_4, J_5, J_6, J_8 as the unknown joints defining the 11-dimensional state vector. The mechanism consists of ten rigidity constraint equations and one homogeneous coordinate equation for prismatic joint. The simulation algorithm successfully solves these constraints and plots the trajectory of the coupler point J_8 as shown in Fig. 4.2. The run-time of this simulation was 3.79s.

4.4.3 Planar Modified Theo Jansen linkage

In this example, a planar modified Theo Jansen linkage with one of its revolute joints replaced by a floating prismatic joints is simulated. The eight-bar mechanism is displayed in Fig. 4.6 and its joint and link data is given in Table 4.3. J_1, J_5 are the fixed joints, J_2 is the perturbed joint and the state vector consists coordinates of J_3, J_4, J_6, J_7, J_8. This results in a 11-dimensional state vector. Ten rigidity constraint equations for links and one homogeneous coordinate equation for prismatic joint are available for this mechanism. The simulation algorithm plots the trajectory of the coupler point J_8 as shown in Fig. 4.6. Note, the length of stride for this modified mechanism is larger than that of the conventional Theo Jansen mechanism which has revolute joints only. As a result, this mechanism is a prospective candidate for walking robots. The run-time of this simulation was 3.81s.

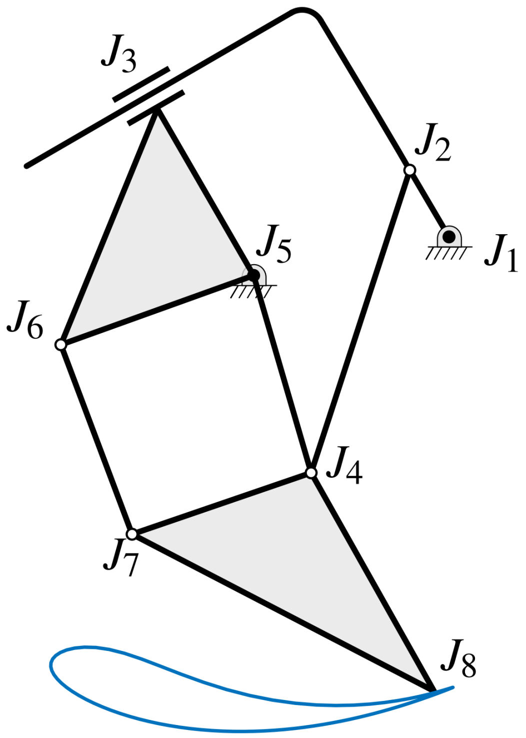


Figure 4.6: Planar modified Theo Jansen with floating prismatic joint

Table 4.3: Joint and Link data for Modified Theo Jansen linkage

Joint	Coordinates
$J_{1,input}$	2.77, 2.31
J_2	2.17, 3.33
J_3	-0.50, 0.87, -4.80
J_4	.66, -1.3
J_5	-.22, 1.72
J_6	-3.17, .66
J_7	-2.08, -2.24
J_8	2.54, -4.64

Link	Constituent joints
L_1	J_1, J_2
L_2	J_2, J_3
L_3	J_2, J_4
L_4	J_3, J_5, J_6
L_5	J_5, J_4
L_6	J_6, J_7
L_7	J_4, J_7, J_8
$L_{8,ground}$	J_1, J_5

4.4.4 Spherical RRPR mechanism

This example presents the simulation of a spherical RRPR mechanism which is the spherical analog of the Whitworth quick-return mechanism. The four-bar mechanism is displayed in Fig. 4.3 and it's joint and link data is given in Table 4.2. The mechanism has J_1, J_4 as the fixed joints, J_2 as the perturbed joint and J_3, J_5 as the unknown joints defining the 6-dimensional state vector. The mechanism consists of four rigidity constraint equations for output and coupler links which can be described using Eq. (4.1) and Eq. (4.6). Also, one unit sphere equation for revolute joint using Eq. (4.11) and one homogeneous coordinate equation for prismatic joint using Eq. (4.10) can be written. Once the simulation is completed, the trajectory of coupler point J_5 can be plotted as shown in Fig. 4.3. The run-time of this simulation was 1.49s.

4.4.5 Spherical Watt-I linkage

In this example, a spherical Watt-I six-bar linkage with prismatic input joint is simulated. Spherical Watt I type linkages have been used to design door hinges for spatial movement. The six-bar mechanism is shown in Fig. 4.7 and its link and joint data is given in Table 4.4. From the data, its known that J_1, J_6 are the fixed joints, J_2, J_3 are the perturbed joints and J_4, J_5, J_7, J_8

are the unknown joints representing the 12-dimensional state vector. The mechanism can be described using eight rigidity constraints for links and four unit circle constraints. Perturbing the input link along the input prismatic joint results in the motion of coupler point J_8 as shown in Fig. 4.7. The run-time of this simulation was 3.33s.

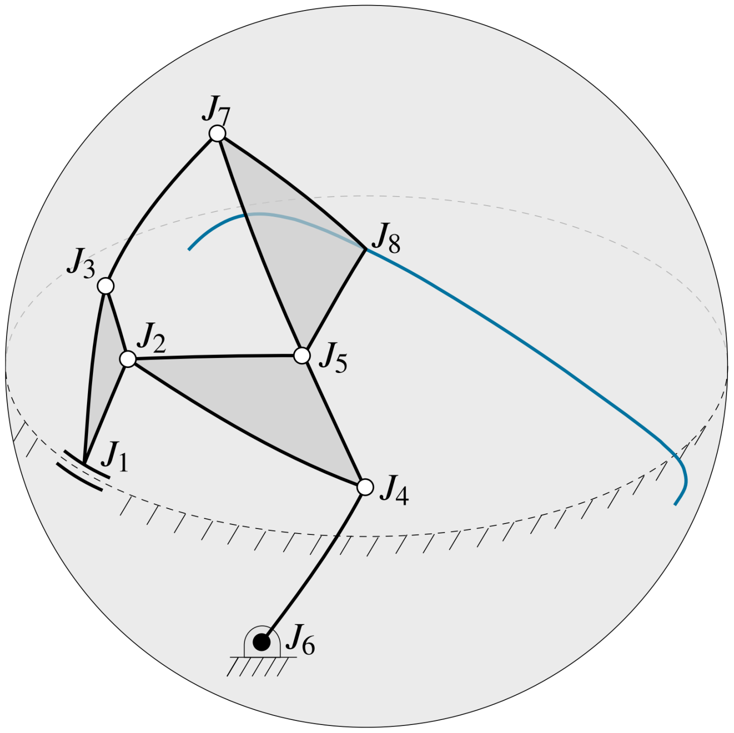


Figure 4.7: Spherical Watt I six-bar linkage

4.4.6 Spatial 5-SS Mechanism

In this section, we demonstrate the scalability of proposed algorithm to spatial mechanisms by simulating a 5-SS platform linkage. A 5-SS mechanism consists of five binary spherical-spherical (SS) links connected to a floating coupler link on one end and the ground link on the other [64]. Although using the Grüebler criterion [36], the mobility of this mechanism is six, five of the rotational degrees for each binary link are redundant and therefore, 5-SS platform linkage is a one degree-of-freedom mechanism.

Table 4.4: Joint and Link data for Spherical RRPR linkage

Joint	Coordinates
$J_{1,input}$	0, 0, 1
J_2	0.93, 0, 0.37
J_3	0.85, -0.17, 0.51
J_4	0.70, 0.70, 0.14
J_5	0.73, 0.49, 0.49
J_6	0.81, 0.41, -0.41
J_7	0.48, -0.10, 0.87
J_8	0.49, 0.49, 0.73

Link	Constituent joints
L_1	J_1, J_2, J_3
L_2	J_2, J_4, J_5
L_3	J_4, J_6
L_4	J_3, J_7
L_5	J_5, J_7, J_8
$L_{6,ground}$	J_1, J_6

An example 5-SS mechanism is displayed in Fig. 4.8 and its joint and link data is given in Table. 4.5. Since the input link perturbation approaches outlined in Section 4.3.1 only cover revolute and prismatic joints, a new approach is needed for spherical input links. We attach a linear actuator between J_1 and J_7 to actuate the mechanism [64]. This results in the mechanism having J_1, J_2, J_3, J_4, J_5 as fixed joints and $J_6, J_7, J_8, J_9, J_{10}, J_{11}$ as the unknown joints defining the 18-dimensional state vector. Five rigidity constraints for binary links, twelve independent rigidity constraints for coupler and one additional constraint for input actuator length is available for this mechanism. All these constraints are modelled using Eq. (4.1) and the simulation is carried out to generate a spatial trajectory of coupler point J_{11}. The input prismatic link is perturbed by .01 units and run-time of this simulation was 0.28s.

4.5 Conclusion

In this chapter, we have presented unified equations for motion simulation of planar and spherical $n-$bar mechanisms and an efficient algorithm for computation to enable real-time, interactive simulation. The approach is general and uses simple geometric primitives, such as point, line, and planes to represent the constraints inherent in mechanisms. A 5-SS mechanism is simulated to demonstrate the scalability of the proposed approach to spatial

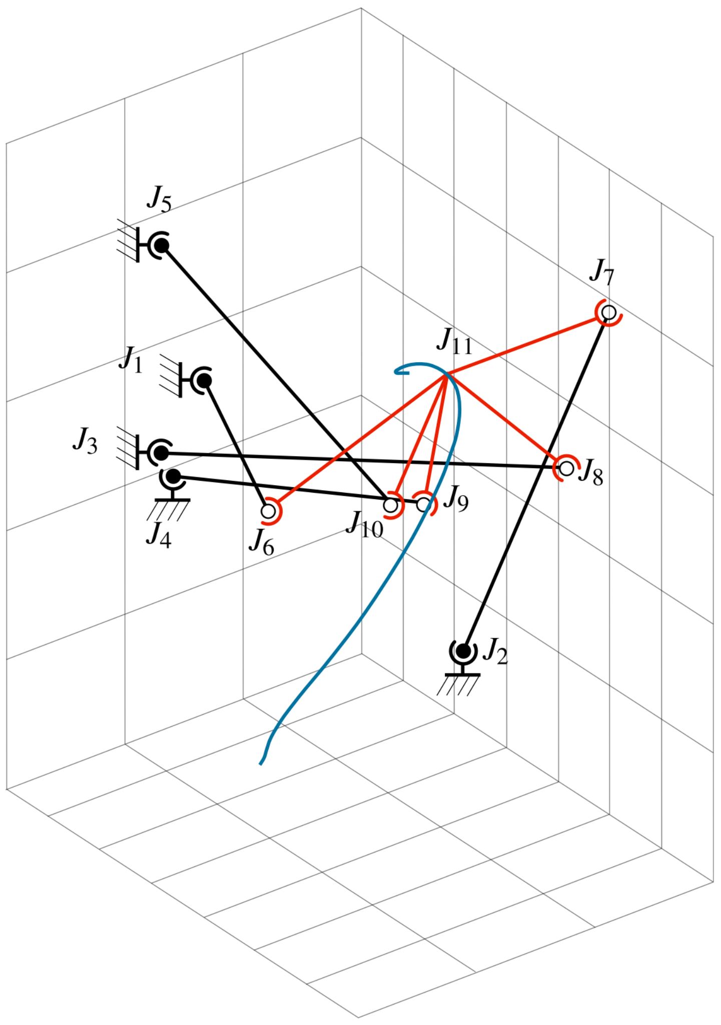


Figure 4.8: Spatial 5-SS platform linkage

mechanisms. Once the mechanism is simulated and the path of coupler point determined, velocity and acceleration curves can easily be determined using numerical differentiation. Future research would involve finding appropriate representation and rigidity constraints for cylindrical and helical joints to further unify spatial synthesis.

Table 4.5: Joint and Link data for Spatial 5-SS platform linkage

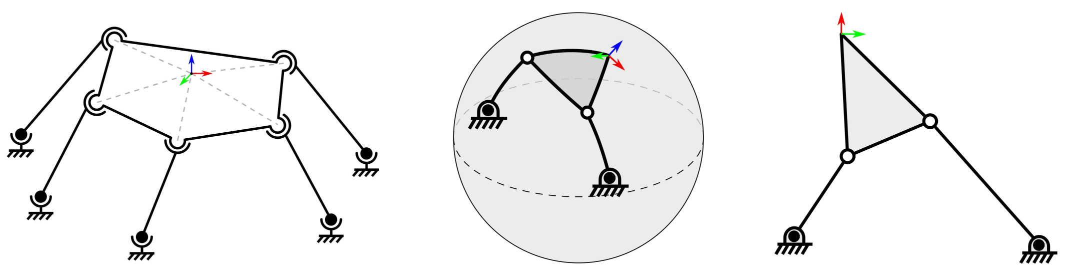
Joint	Coordinates
J_1	-7.2, -5.29, 7.87
J_2	5.72, -0.71, -8.26
J_3	-8.75, -4.60, 5.49
J_4	-10.00, -8.72, 6.09
J_5	-3.88, 6.61, 8.91
J_6	-6.61, -9.99, 3.93
J_7	8.19, -9.05, 6.07
J_8	7.84, -5.73, -0.62
J_9	4.15, -0.48, -2.04
J_{10}	-0.97, -9.04, 1.92
J_{11}	2.20, -7.17, 5.36

Link	Constituent joints
L_1	J_1, J_6
L_2	J_2, J_7
L_3	J_3, J_8
L_4	J_4, J_9
L_5	J_5, J_{10}
L_6	J_6, J_7, J_8, J_9, J_{10}, J_{11}
$L_{7,ground}$	J_1, J_2, J_3, J_4, J_5

Chapter 5

Unified Design of Spatial, Spherical and Planar Mechanisms for the Motion Synthesis Problem

5.1 Introduction



(a) Spatial 5-SS platform linkage

(b) Spherical Four-Bar Linkage

(c) Planar Four-Bar Linkage

Figure 5.1: Kinematic diagrams of the family of mechanisms that can be synthesized using proposed algorithm

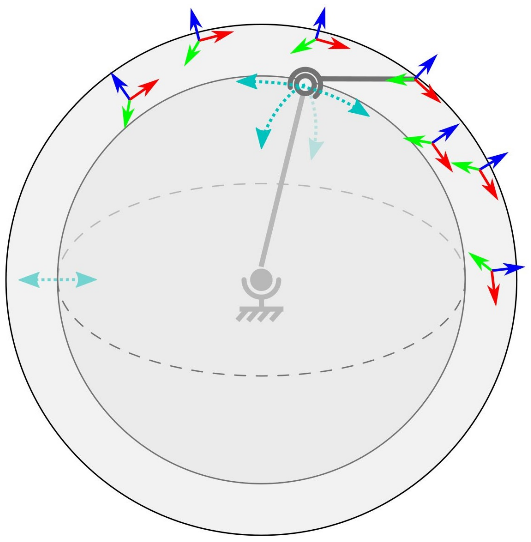
Determining a kinematic mechanism, specified by its joint types and pattern of their interconnection, and link dimensions, such that one of its links passes through several specified poses is defined as the motion synthesis problem [12, 65]. This problem has a rich literature dealing with the synthesis of planar, spherical, and spatial mechanisms.

Of all types of mechanisms, synthesis and analysis of planar mechanisms have been well investigated using analytical, graphical or optimization based approaches. Burmester [15] first solved the planar four-bar motion synthesis problem for five poses. Ravani and Roth [66] used a kinematic mapping based approach for motion synthesis for the first time. Larochelle [67] calculated planar RR dyads using constraint manifold projection. Bawab et al. [68] synthesized crank driven four-bar linkages using optimization theory. Holte et al. [69] proposed a motion generation approach that incorporated both exact and approximate pose constraints. Al-Widyan et al. [70] introduced a robust motion synthesis algorithm that better handled algorithmic singularities. Brunnthaler et al. [71] also used kinematic mapping to generate a univariate quartic and solve for four-bar mechanisms. Bourrelle et al. [72] used a graphical approach to calculate both RR and PR dyads for the five pose Burmester problem. Larochelle [73] proposed an analytical approach that can handle exact and approximate poses to produce planar RR dyads. Ge et al. [21] proposed an algebraic fitting based unified type and dimensional synthesis for planar mechanisms. Purwar et al. [18] created a mobile application to implement a real-time motion synthesis algorithm which unifies pose constraints and pivot location constraints. Deshpande and Purwar [19] further extended this approach to nonlinear constraints using a Lagrange multiplier method. Recently, they have also developed a machine learning based approach to synthesize defect-free mechanisms [74, 75].

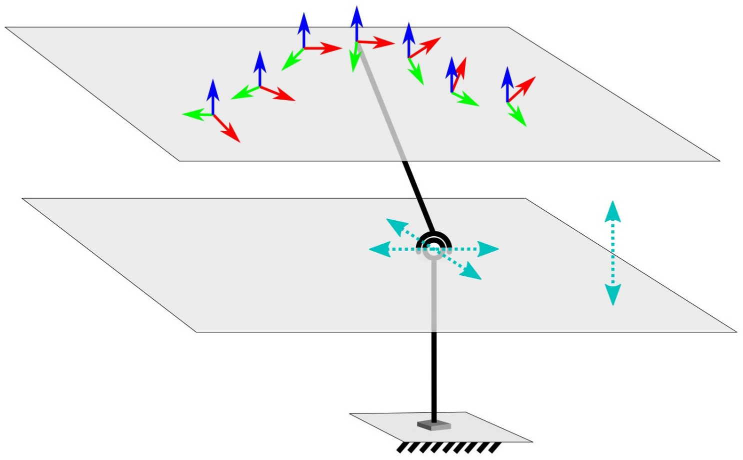
Spherical mechanism synthesis has been also conducted using kinematic mapping based analytical approaches or optimization algorithms. Chiang presented an in-depth review on the kinematic analysis and synthesis of spherical mechanisms [76]. Bodduluri and McCarthy [77] carry out finite position synthesis of spherical mechanisms by minimizing the normal distance in the image space. Lin [78] uses homotopy methods to generate spherical four-bar mechanisms for motion and path generation. Ruth and McCarthy [79] describe SphinxPC, a computer-aided design software system for spherical four-bar linkage synthesis. Brunnthaler et al. [80] used kinematic mapping to synthesize spherical four-bars. Zhuang et al. [81] have used an adaptive genetic algorithm to synthesize spherical four-bars. Li et al. [60] solve a more general n-discrete pose problem using kinematic mapping and synthesize spherical four-bar mechanisms.

For spatial mechanisms synthesis, several algorithms have used kinematic mapping and numerical homotopy based approaches. Innocenti [82] first extended the notion of geometric constraints to the construction of 5-SS spatial platforms using spherical constraints and termed it the Spatial Burmester problem. It has been shown that a 5-SS mechanism can pass exactly through seven spatial poses. Liao and McCarthy [64] improved upon Innocenti's work and formulated a methodology for singularity analysis. Plecnik and McCarthy [83] used the 5-SS platform as a steering linkage. Li et al. [84] carry out both type and dimensional synthesis of platform linkages using an algebraic fitting approach to unify the spherical and planar constraints. Ge et al. [85, 86] improved upon this algebraic fitting approach and incorporated constraints on pivot locations.

In the above classic approaches, planar, spherical, and spatial mechanism synthesis are considered as separate design problems. An ideal approach would be one that can synthesize all types of mechanisms using a unified framework. This serves as the motivation for this chapter. Ge et al. [87] have proposed an approach that unifies the synthesis of planar and spherical dyads. Ge et al. [88] also proposed a methodology to synthesize spatial RR dyads using an intersection of SS dyads. It is well known that there exist planar and spherical RR dyads for the five pose planar and spherical Burmester problems even when the poses are represented as spatial poses [15, 76]. However, the approach by Ge et al. is unable to generate RR dyads for the five spatial pose problem and requires at least seven poses.



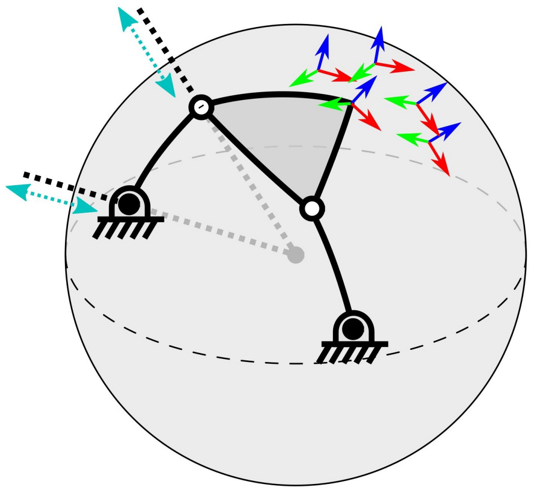
(a) Family of SS dyads



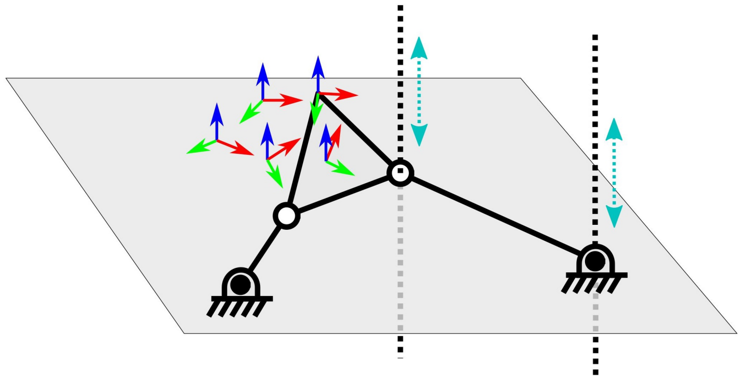
(b) Family of PS dyads

Figure 5.2: Multiplicity in dyad solutions which satisfy seven spatial poses lying on a sphere or plane

In this chapter, we present a novel algebraic fitting based algorithm to unify spatial, spherical, and planar mechanism synthesis. We use a dual quaternion representation to describe the pose data. The algorithm aims to generate Spherical-Spherical (SS) and Revolute-Revolute (RR) dyads that can be put together to form a spatial 5-SS, spherical four-bar, or planar four-bar mechanism. Degenerate form of the SS dyad, i.e., the Planar-Spherical (PS) dyad is also included in the framework. Kinematic diagrams of these



(a) Family of Spherical RR dyads



(b) Family of Planar RR dyads

Figure 5.3: Multiplicity in dyad solutions which satisfy five spatial poses lying on a sphere or plane

mechanisms have been displayed in Fig. 5.1. First, an existing methodology to generate SS dyads for seven or more spatial poses is reviewed. However, this method fails when all spatial poses lie on a sphere or a plane. We show that there exists at least 3-∞ SS/PS dyad solutions for this special case and an approach is proposed to find this solution space. Also, the coordinates of the characteristic-sphere or -plane on which the spatial poses lie are determined. For the spherical case, we generate a family of SS dyads that have their fixed pivot (FP) located at the center of the characteristic sphere while their moving pivot (MP) is free. Similarly, for the planar case, we get a family of PS dyads that have their FP in the same direction as the normal of the characteristic plane and MP coordinates are free. This 3-∞ multiplicity in dyadic solutions is visualized in Fig. 5.2 by the three cyan arrows denoting free MP coordinates.

Next, we tackle the spatial five-pose problem to find if a RR dyad exists. There exist $2-\infty$ SS/PS dyads due to two less constraints than the seven pose case. It is shown that at least 2-∞ RR dyad solutions can exist when the five spatial poses lie on a sphere or plane. For a spherical RR dyad, its FP and MP can move along the axis connecting to the center of the characteristic sphere without any effect on kinematics of motion. Similarly, for a planar RR dyad, its FP and MP can move perpendicular to its characteristic plane. Constraints are systematically placed on this solution space to extract the RR dyad existing on the characteristic sphere or plane. This 2-∞ multiplicity in dyadic solutions is visualized in Fig. 5.3 by the two cyan arrows denoting free MP coordinates. An approach to isolate distinct RR dyads from the 2-∞ solutions is proposed.

The chapter also enhances the robustness of existing approaches by reducing possible algorithmic singularities, i.e., cases where a solution kinematic mechanism exists, but the algorithm fails to find it. According to the Grubler's criterion for spatial mechanisms, planar and spherical linkages are over-constrained and thus prove challenging to incorporate in spatial motion synthesis algorithms. However, they can now be synthesized using the proposed algorithm in a unified manner. The algorithm is also able to find all possible existing dyads, i.e., up to 20 SS dyads, up to 6 spherical RR dyads, and up to 4 planar RR dyads.

The original contributions of this chapter are in 1) unification of planar, spherical, and spatial motion synthesis problem into a single framework, 2) developing insight into the dimensionality of dyadic solution space and using it to isolate planar and spherical RR dyads, and 3) enhancing the robustness by addressing algorithmic singularities in existing motion synthesis algorithms for cases where spatial poses lie on a plane or sphere.

The remaining chapter is organized as follows. Section 2 discusses the representation of spatial poses using dual quaternions. Section 3 reviews the unified constraint equation for plane and sphere constraints. Section 4 demonstrates the existing algorithm to carry out motion synthesis using seven spatial poses. Section 5 discusses the nature of solution space when all poses lie on a sphere or a plane. Section 6 presents a systematic methodology to isolate RR dyads from the solution space representing SS and PS dyads. Section 7 illustrates some examples before making concluding remarks in the last section.

5.2 Spatial Displacement Representation

In this section, we review the relations used to express spatial Poses in the form of dual quaternions [32, 33]. Let a moving rigid body in space be denoted by a coordinate frame M attached to it. A point on the moving body with respect to M can be represented using homogeneous coordinates $\mathbf{c} = (c_1, c_2, c_3, c_4)$. The same point is defined as $\mathbf{C} = (C_1, C_2, C_3, C_4)$ in fixed frame F. The point coordinate transformation from moving frame M to fixed frame F is given as

$$\mathbf{C} = \begin{bmatrix} \mathbf{R} & \mathbf{d} \\ 0 & 1 \end{bmatrix} \mathbf{c} \tag{5.1}$$

where $\mathbf{R}$ is the rotation matrix that describes the orientation of M relative to F and $\mathbf{d} = (d_1, d_2, d_3)$ is the vector from origin of F to M. The pose orientation can be described using the Euler-Rodrigues parameters which involves rotation axis and angle. The axis is represented by a unit vector $\mathbf{s} = (s_x, s_y, s_z)$ and rotation angle θ.

A spatial pose can be represented using a unit dual quaternion $\mathbf{Q} = (\mathbf{q}, \mathbf{g})$ where the real part is quaternion $\mathbf{q} = (q_1, q_2, q_3, q_4)$ and dual part is quaternion $\mathbf{g} = (g_1, g_2, g_3, g_4)$ [32]. It satisfies the relations

$$q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1, \tag{5.2}$$

$$q_1 g_1 + q_2 g_2 + q_3 g_3 + q_4 g_4 = 0. \tag{5.3}$$

The dual quaternion $\mathbf{Q}$ can be calculated as follows

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} s_x sin(\theta/2) \\ s_y sin(\theta/2) \\ s_z sin(\theta/2) \\ cos(\theta/2) \end{bmatrix}, \tag{5.4}$$

$$\begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -d_3 & d_2 & d_1 \\ d_3 & 0 & -d_1 & d_2 \\ -d_2 & d_1 & 0 & d_3 \\ -d_1 & -d_2 & -d_3 & 0 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix}. \tag{5.5}$$

The relationships between dual quaternion $\mathbf{Q}$ and the rotation matrix $\mathbf{R}$ and displacement vector $\mathbf{d}$ can be given as

$$\mathbf{R} = \begin{bmatrix} q_4^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_4q_3) & 2(q_1q_3 + q_4q_2) \\ 2(q_1q_2 + q_4q_3) & q_4^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_4q_1) \\ 2(q_1q_3 - q_4q_2) & 2(q_2q_3 + q_4q_1) & q_4^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{5.6}$$

$$\mathbf{d} = -2 \begin{bmatrix} g_4q_1 - g_1q_4 + g_2q_3 - g_3q_2 \\ g_4q_2 - g_2q_4 + g_3q_1 - g_1q_3 \\ g_4q_3 - g_3q_4 + g_1q_2 - g_2q_1 \end{bmatrix} \tag{5.7}$$

The use of dual quaternion $\mathbf{Q}$ leads to a compact constraint equation which is derived in the next section.

5.3 Unified Representation of Spherical and Planar Constraints

In this section, we review the unified constraint relation derived in [86]. Some of the equations have been modified for clarity. For a spatial SS dyad, its coupler point $C = (C_1, C_2, C_3, C_4)$ is geometrically constrained on a sphere whose radius and center are represented by homogeneous coordinates $A = (A_0, A_1, A_2, A_3, A_4)$. This constraint can be given as

$$2A_1C_1 + 2A_2C_2 + 2A_3C_3 + A_0C_4 = A_4\left(\frac{C_1^2 + C_2^2 + C_3^2}{C_4}\right), \tag{5.8}$$

$$A_4^2r^2 - A_0A_4 = A_1^2 + A_2^2 + A_3^2, \tag{5.9}$$

where r is the radius of the sphere formed by SS dyad. When $A_4 = 0$, the geometric constraint equation represents a plane described by a PS dyad. Equation (5.8) consists of seven independent parameters that specify a unique dyad since A_i and C_i are homogeneous in nature.

The final dyadic constraint equation is obtained by substituting the fixed frame coupler point ($\mathbf{C}$) in geometric constraint equation Eq (5.8) with moving coordinate ($\mathbf{c}$) according to transformation relationships in Eq (5.1). On collecting similar terms, we can restructure constraint equation as follows

$$\sum_{i=0}^{16} K_iP_i = 0 \tag{5.10}$$

where constraint space coefficients (CSCs) $(K_0, K_1, ...K_{16})$ are given as

$$\begin{aligned}
K_0 &= 1, \\
K_1 &= 2(q_1^2 - q_2^2 - q_3^2 + q_4^2), \\
K_2 &= 4(q_1q_2 + q_3q_4), \\
K_3 &= 4(q_1q_3 - q_2q_4), \\
K_4 &= 4(g_4q_1 + g_3q_2 - g_2q_3 - g_1q_4), \\
K_5 &= 4(q_1q_2 - q_3q_4), \\
K_6 &= 2(-q_1^2 + q_2^2 - q_3^2 + q_4^2), \\
K_7 &= 4(q_2q_3 + q_1q_4), \\
K_8 &= 4(-g_3q_1 + g_4q_2 + g_1q_3 - g_2q_4), \\
K_9 &= 4(q_1q_3 + q_2q_4), \\
K_{10} &= 4(q_2q_3 - q_1q_4), \\
K_{11} &= 2(-q_1^2 - q_2^2 + q_3^2 + q_4^2), \\
K_{12} &= 4(g_2q_1 - g_1q_2 + g_4q_3 - g_3q_4), \\
K_{13} &= 4(-g_4q_1 + g_3q_2 - g_2q_3 + g_1q_4), \\
K_{14} &= 4(-g_3q_1 - g_4q_2 + g_1q_3 + g_2q_4), \\
K_{15} &= 4(g_2q_1 - g_1q_2 - g_4q_3 + g_3q_4), \\
K_{16} &= 4(-g_1^2 - g_2^2 - g_3^2 - g_4^2),
\end{aligned} \tag{5.11}$$

and the constraint space parameters (CSPs) $(P_0, P_1, ...P_{16})$ are given as

$$\begin{gathered}
P_0 = A_0c_4 - A_4c_4\left(\frac{c_1^2}{c_4^2} + \frac{c_2^2}{c_4^2} + \frac{c_3^2}{c_4^2}\right), \\
P_1 = A_1c_1, \quad P_2 = A_2c_1, \quad P_3 = A_3c_1, \quad P_4 = A_4c_1, \\
P_5 = A_1c_2, \quad P_6 = A_2c_2, \quad P_7 = A_3c_2, \quad P_8 = A_4c_2, \\
P_9 = A_1c_3, \quad P_{10} = A_2c_3, \quad P_{11} = A_3c_3, \quad P_{12} = A_4c_3, \\
P_{13} = A_1c_4, \quad P_{14} = A_2c_4, \quad P_{15} = A_3c_4, \quad P_{16} = A_4c_4.
\end{gathered} \tag{5.12}$$

Equation (5.10) is referred to as the Constraint equation (C-equation) and it consists of 17 homogeneous CSPs P_i. $A_i(i = 0 \dots 4)$ and $c_j(j = 1 \dots 4)$ are referred to as the Dyadic parameters. The CSPs P_i are subjected to nine additional bi-linear constraints defined as

$$\begin{aligned}
\frac{P_1}{P_{13}} = \frac{P_2}{P_{14}} = \frac{P_3}{P_{15}} = \frac{P_4}{P_{16}} &= \lambda_1 = \frac{c_1}{c_4}, \\
\frac{P_5}{P_{13}} = \frac{P_6}{P_{14}} = \frac{P_7}{P_{15}} = \frac{P_8}{P_{16}} &= \lambda_2 = \frac{c_2}{c_4}, \\
\frac{P_9}{P_{13}} = \frac{P_{10}}{P_{14}} = \frac{P_{11}}{P_{15}} = \frac{P_{12}}{P_{16}} &= \lambda_3 = \frac{c_3}{c_4},
\end{aligned} \tag{5.13}$$

or

$$\begin{aligned}
\frac{P_1}{P_4} = \frac{P_5}{P_8} = \frac{P_9}{P_{12}} = \frac{P_{13}}{P_{16}} &= \mu_1 = \frac{A_1}{A_4}, \\
\frac{P_2}{P_4} = \frac{P_6}{P_8} = \frac{P_{10}}{P_{12}} = \frac{P_{14}}{P_{16}} &= \mu_2 = \frac{A_2}{A_4}, \\
\frac{P_3}{P_4} = \frac{P_7}{P_8} = \frac{P_{11}}{P_{12}} = \frac{P_{15}}{P_{16}} &= \mu_3 = \frac{A_3}{A_4},
\end{aligned} \tag{5.14}$$

where, $\lambda_i(= c_i/c_4)$ represents the MP coordinate in the moving frame and $\mu_i(= A_i/A_4)$ represents the FP in the fixed frame. There exist alternative ways of writing the bi-linear constraints which are equivalent to Eq. (5.13) or Eq. (5.14),

These spherical and planar constraints can also represent other spatial linkages. The geometric constraint for a Universal-Spherical (TS) link and an RRS open chain with intersecting axes for the RR joints is a sphere. The constraint for an RRS with parallel axis, a Prismatic-Revolute-Spherical, and a Revolute-Prismatic-Spherical open chain is a plane.

Next, we discuss the algorithm which uses Eq (5.10) and Eq (5.13) to synthesize platform mechanisms constrained by spherical or planar geometric constraints.

5.4 Dyad Calculation for Seven or more General Spatial Poses

This section reviews the work done by Ge et al. [86, 87, 88] with minor modifications. The motion synthesis algorithm takes a set of spatial poses as input. To get unique solutions, at least seven spatial poses are required by the algorithm. For less than seven poses, an infinite number of spherical and planar constraints exist. The computation is carried out in three distinct steps and the result is a set of unique dyads defined by their homogeneous coordinates A_i and c_i.

First, the pose constraints are enforced using Eq (5.10). Then, the bilinear constraints are imposed using Eq (5.13) to get unique values of the CSPs P_i. Lastly, these values are mapped to their respective dyadic parameters A_i and c_i.

5.4.1 Applying Pose Constraints

First, the CSCs (K_i) for each spatial pose are calculated using Eq (5.10). For an n-pose problem, these values can be consolidated into a linear system of equations as follows

$$\begin{bmatrix} K_{1,0} & K_{1,1} & \cdots & K_{1,16} \\ K_{2,0} & K_{2,1} & \cdots & K_{2,16} \\ \vdots & \vdots & \ddots & \vdots \\ K_{n,0} & K_{i,1} & \cdots & K_{n,16} \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ \vdots \\ P_{16} \end{bmatrix} = [\mathbf{K}]\mathbf{P} = 0 \qquad (5.15)$$

where $[\mathbf{K}]$ represents the coefficient matrix. The null-space (solution subspace) of this system of equation can be analyzed using Singular Value Decomposition (SVD); i.e., $[\mathbf{K}]$ is factored into

$$[\mathbf{K}] = [\mathbf{U}][\mathbf{W}][\mathbf{V}]^T \qquad (5.16)$$

where $[\mathbf{U}]$ is an $n \times n$ orthonormal matrix whose columns represent the left singular vectors of $[\mathbf{K}]$; $[\mathbf{W}]$ is the $n \times 17$ diagonal matrix whose elements are square root of eigenvalues of $[\mathbf{K}][\mathbf{K}]^T$; and $[\mathbf{V}]^T$ is the 17×17 orthonormal matrix whose columns represent the right singular vectors. The null-space is the vector subspace spanned by the right singular vectors corresponding to singular values having negligible magnitude.

Since it is well known that a spatial SS dyad can exactly pass through a maximum of seven poses, ten right singular vectors corresponding to singular values having the least magnitude are selected. Basically, we apply seven constraints on a 17-dimensional solution space to get a ten-dimensional solution subspace. When more than seven poses are inputted, the span of these ten singular vectors represent the solution space in the least square sense. Thus, the solution dyad $\mathbf{P}$ can be given as a linear combination of ten singular vectors and the ten-dimensional solution subspace is denoted as $[\mathbf{V}]$, i.e.,

$$\mathbf{P} = \sum_{i=1}^{10} \alpha_i \mathbf{v_i} = \begin{bmatrix} \mathbf{v_1} & \mathbf{v_2} & \cdots & \mathbf{v_{10}} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{10} \end{bmatrix} = [\mathbf{V}]\alpha. \tag{5.17}$$

5.4.2 Applying Bi-linear Constraints

Next, we apply the nine bi-linear constraints on the solution space as described in Eq (5.13). A naive approach would be to solve the system of non-linear equation using numerical homotopy methods directly to find solutions. However, such an approach is computationally intensive and not real-time. To enforce the bi-linear constraints in a more efficient manner, $[\mathbf{V}]$ in Eq (5.17) is broken down into smaller system of equations as follows

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} = \begin{bmatrix} v_{1,1} & \cdots & v_{10,1} \\ v_{1,2} & \cdots & v_{10,2} \\ v_{1,3} & \cdots & v_{10,3} \\ v_{1,4} & \cdots & v_{10,4} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{10} \end{bmatrix} = W_1\alpha \tag{5.18}$$

$$\begin{bmatrix} P_5 \\ P_6 \\ P_7 \\ P_8 \end{bmatrix} = \begin{bmatrix} v_{1,5} & \cdots & v_{10,5} \\ v_{1,6} & \cdots & v_{10,6} \\ v_{1,7} & \cdots & v_{10,7} \\ v_{1,8} & \cdots & v_{10,8} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{10} \end{bmatrix} = W_2\alpha \tag{5.19}$$

$$\begin{bmatrix} P_9 \\ P_{10} \\ P_{11} \\ P_{12} \end{bmatrix} = \begin{bmatrix} v_{1,9} & \cdots & v_{10,9} \\ v_{1,10} & \cdots & v_{10,10} \\ v_{1,11} & \cdots & v_{10,11} \\ v_{1,12} & \cdots & v_{10,12} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{10} \end{bmatrix} = W_3\alpha \tag{5.20}$$

$$\begin{bmatrix} P_{13} \\ P_{14} \\ P_{15} \\ P_{16} \end{bmatrix} = \begin{bmatrix} v_{1,13} & \cdots & v_{10,13} \\ v_{1,14} & \cdots & v_{10,14} \\ v_{1,15} & \cdots & v_{10,15} \\ v_{1,16} & \cdots & v_{10,16} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{10} \end{bmatrix} = W_4\alpha \tag{5.21}$$

The bi-linear constraints in terms of MP coordinate as mentioned in Eq.(5.13) can now be imposed as follows

$$W_1\alpha = \lambda_1 W_4\alpha \tag{5.22}$$

$$W_2\alpha = \lambda_2 W_4\alpha \tag{5.23}$$

$$W_3\alpha = \lambda_3 W_4\alpha \tag{5.24}$$

This generalized eigenvalue problem can be written as

$$\begin{bmatrix} W_1 - \lambda_1 W_4 \\ W_2 - \lambda_2 W_4 \\ W_3 - \lambda_3 W_4 \end{bmatrix}_{12\times 10} \alpha_{10\times 1} = [\mathbf{W}]\alpha = 0. \tag{5.25}$$

As we can note, Eq. (5.25) is a system of twelve equations with ten unknowns. For this system of equations to have a non-trivial solution, the matrix $[\mathbf{W}]$ needs to be rank reduced, i.e., at least have a rank of nine. If $[\mathbf{W}]$ has a rank of nine, determinant of all 10×10 square sub-matrices of $[\mathbf{W}]$ has the value zero. To find the solutions λ_i, the determinant of any three 10×10 sub-matrices of $[\mathbf{W}]$ is constrained to be zero. This results in three non-linear polynomial equations in variables $\lambda_1, \lambda_2, \lambda_3$ which can be solved using a polynomial homotopy based solver, such as Bertini [89] or a symbolic solver, such as Mathematica [90] or Maple [91] to get all possible solutions. The degree of the three non-linear polynomial constraints can be up to ten. For the spatial Burmester problem, it has been shown that there exists a univariate polynomial equation of twentieth order whose roots represent the twenty solution dyads [82]. Once the set of eigenvalues $(\lambda_1, \lambda_2, \lambda_3)$ are known, they can be substituted back in Eq. (5.25) for upto 20 distinct system of equations. The nullspace i.e. the rightmost singular vector can be calculated using SVD which determines the solution vectors α. Finally, the solution CSPs P_i can be calculated using Eq. (5.17).

However, when the spatial poses lie on a sphere or a plane, the algorithm fails to find distinct solution dyads. In the next section, we discuss how to extract the solution space for this degenerate case.

5.4.3 Finding Dyadic Parameters

Once the CSPs are known, the dyadic parameters finally can be calculated. For a Planar geometric constraint, $A_4 = 0$ due to which $P_4, P_8, P_{12},$ and $P_{16} = 0$, while for spherical geometric constraints, $A_4 \neq 0$, i.e. $P_4, P_8, P_{12},$ and P_{16} are nonzero.

If the constraint is spherical, its dyadic parameters represents the fixed pivot coordinates (A_i) and moving pivot coordinates (c_i) and are described using the following inverse relationships

$$A_0 : A_1 : A_2 : A_3 : A_4 = P_0 + \frac{P_4^2}{P_{16}} + \frac{P_8^2}{P_{16}} + \frac{P_{12}^2}{P_{16}} : P_{13} : P_{14} : P_{15} : P_{16} \tag{5.26}$$

$$c_1 : c_2 : c_3 : c_4 = P_4 : P_8 : P_{12} : P_{16} \tag{5.27}$$

Also, the radius of sphere can be calculated as

$$r = \sqrt{\frac{P_{13}^2}{P_{16}^2} + \frac{P_{14}^2}{P_{16}^2} + \frac{P_{15}^2}{P_{16}^2} + \frac{P_4^2}{P_{16}^2} + \frac{P_8^2}{P_{16}^2} + \frac{P_{12}^2}{P_{16}^2} + \frac{P_0}{P_{16}}} \tag{5.28}$$

If the constraint is planar, its dyadic parameters represents the plane normal vector (A_i) and moving pivot coordinates (c_i)

$$\begin{aligned} A_0 : A_1 : A_2 : A_3 &= P_0/2 : P_1 : P_2 : P_3 \\ &= P_0/2 : P_5 : P_6 : P_7 \\ &= P_0/2 : P_9 : P_{10} : P_{11} \\ &= P_0/2 : P_{13} : P_{14} : P_{15} \end{aligned} \tag{5.29}$$

$$\begin{aligned} c_1 : c_2 : c_3 : c_4 &= P_1 : P_5 : P_9 : P_{13} \\ &= P_2 : P_6 : P_{10} : P_{14} \\ &= P_3 : P_7 : P_{11} : P_{15} \end{aligned} \tag{5.30}$$

Thus, we obtain up to 20 unique SS/PS dyads which can satisfy the initial spatial pose problem. Picking any five of the available 20 can lead up to 15,504 unique solutions of one degree-of-freedom mechanisms.

5.5 Dyad Calculation for Seven or more Spatial Poses Lying on a Sphere or Plane

In this section, we explore a special case of the spatial motion synthesis problem where all the poses lie on a sphere or plane. The algorithm outlined by Ge et. al. [86] fails to generate any solutions in this scenario.

For a seven or more spherical/planar pose problem, we can easily impose the pose constraints using nullspace analysis of Eq (5.15). However, when applying the bi-linear constraints using Eq (5.25), we observe (when doing numerical experiments) that all determinants of 10×10 sub-matrices have zero magnitudes and are thus trivially satisfied. This entails that any values of $\lambda_1, \lambda_2, \lambda_3$ can be selected and a solution dyad can be found. Since λ_i represents the coupler coordinates in the moving frame, the three coupler coordinates are free variables which mean there exists a 3-∞ solution space of spherical or planar constraints.

We also observe that the FP locations remain fixed for this family of solutions. We call the sphere or plane on which these special spatial poses lie as their characteristic-sphere or -plane, respectively. For the spherical case, the FP is located at the center of the characteristic sphere while for the planar case, the FP represents the normal vector of the characteristic plane. As a result, the solutions are essentially a set of concentric spherical constraints or parallel plane constraints. This family of solutions has been visualized in Fig. 5.2 where the cyan arrows represent the free MP coordinates.

Within each of the 3-∞ family of solutions, there exists a special solution dyad which has a zero-length coupler, i.e., it has $(c_1, c_2, c_3 = 0)$. This dyad has the same radius/intercept and FP as the characteristic sphere/plane. Existence of the zero-coupler length dyad can be used as an elegant way to check if a set of spatial poses lie on a sphere/plane. According to Eq. (5.12), the zero length coupler dyad has its CSPs $(P_1, \cdots, P_{12} = 0)$ due to zero coupler coordinates. To check for its existence, $[\mathbf{V}]$ in Eq (5.17) is broken down into smaller system of equations as follows

$$\begin{bmatrix} P_1 \\ \vdots \\ P_{12} \end{bmatrix} = \begin{bmatrix} v_{1,1} & \cdots & v_{10,1} \\ \vdots & & \vdots \\ v_{1,12} & \cdots & v_{10,12} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{10} \end{bmatrix} = 0 \tag{5.31}$$

If the null-space of Eq (5.31) is one dimensional, i.e., a non-trivial null-space exists, the poses lie on a sphere/plane and there exist 3-∞ solution dyads. Once the α_i are known, they can be back substituted in Eq. (5.17) to find CSPs P_i for zero-length coupler dyad and the A_i for FP coordinates. In this way, the parameters of the characteristic sphere or plane can be calculated. However, we have yet to determine the four-dimensional solution subspace for CSPs.

Alternative to Eq. (5.14), there also exist an equivalent set of bi-linear constraints in terms of FP as follows

$$\begin{aligned} \frac{P_2}{P_1} = \frac{P_6}{P_5} = \frac{P_{10}}{P_9} = \frac{P_{14}}{P_{13}} = \frac{A_2}{A_1} \\ \frac{P_3}{P_1} = \frac{P_7}{P_5} = \frac{P_{11}}{P_9} = \frac{P_{15}}{P_{13}} = \frac{A_3}{A_1} \\ \frac{P_4}{P_1} = \frac{P_8}{P_5} = \frac{P_{12}}{P_9} = \frac{P_{16}}{P_{13}} = \frac{A_4}{A_1} \end{aligned} \tag{5.32}$$

Imposing these constraints results in an alternative system of equations to

Eq. (5.25) as follows

$$Z_1 = \begin{bmatrix} v_{1,1} & \cdots & v_{10,1} \\ v_{1,5} & \cdots & v_{10,5} \\ v_{1,9} & \cdots & v_{10,9} \\ v_{1,13} & \cdots & v_{10,13} \end{bmatrix}, Z_2 = \begin{bmatrix} v_{1,2} & \cdots & v_{10,2} \\ v_{1,6} & \cdots & v_{10,6} \\ v_{1,10} & \cdots & v_{10,10} \\ v_{1,14} & \cdots & v_{10,14} \end{bmatrix}, \tag{5.33}$$

$$Z_3 = \begin{bmatrix} v_{1,3} & \cdots & v_{10,3} \\ v_{1,7} & \cdots & v_{10,7} \\ v_{1,11} & \cdots & v_{10,11} \\ v_{1,15} & \cdots & v_{10,15} \end{bmatrix}, Z_4 = \begin{bmatrix} v_{1,4} & \cdots & v_{10,4} \\ v_{1,8} & \cdots & v_{10,8} \\ v_{1,12} & \cdots & v_{10,12} \\ v_{1,16} & \cdots & v_{10,16} \end{bmatrix}, \tag{5.34}$$

$$\begin{bmatrix} Z_2 - (A_2/A_1)Z_1 \\ Z_3 - (A_3/A_1)Z_1 \\ Z_4 - (A_4/A_1)Z_1 \end{bmatrix}_{12\times 10} \alpha_{10\times 1} = [\mathbf{Z}]\alpha = 0. \tag{5.35}$$

where A_1, A_2, A_3, A_4 represents the homogeneous FP coordinates.

To calculate the basis vectors of the 3-∞ solution space, we back-substitute FP coordinates into Eq. (5.35) and calculate $[\mathbf{Z}]$. Due to the $3-\infty$ solution, the rank of $[\mathbf{Z}]$ is six and there exist four singular vectors α. We concatenate them as $[\alpha]$ matrix of dimension 10×4 and back-substitute in Eq. (5.17) to find four-dimensional CSPs that represent the basis solution subspace. Since CSPs are homogeneous in nature, their 4-D subspace maps to 3-∞ solution SS or PS dyads. This completes the calculation of the subspase representing 3-∞ solution dyads. The complete algorithm for seven or more spatial pose synthesis has been visualized in Fig. 5.4.

Thus, we can analytically find the 3-∞ spherical or planar constraints to motion synthesis problem for seven or more spatial poses that lie on a sphere/plane. Next, we solve the motion generation problem involving five spatial poses, which lie on a sphere or plane.

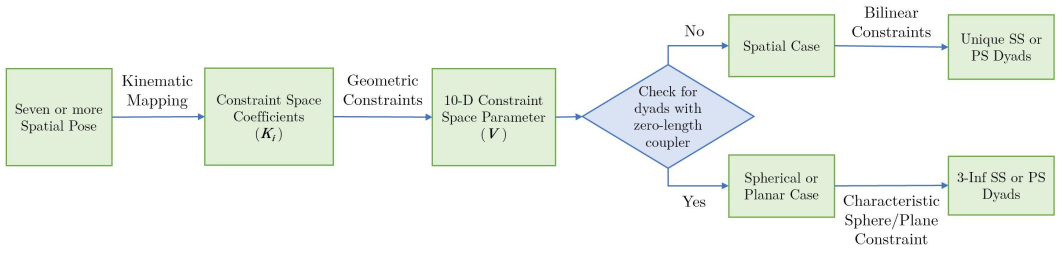
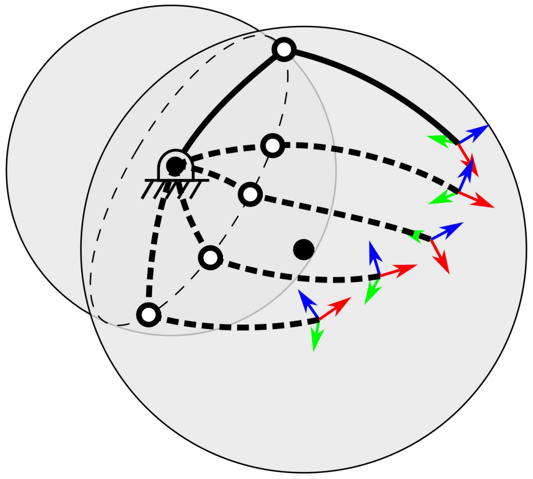

Seven or more Spatial Pose
Kinematic Mapping
Constraint Space Coefficients ($\boldsymbol{K}_i$)
Geometric Constraints
10-D Constraint Space Parameter ($\boldsymbol{V}$)
Check for dyads with zero-length coupler
No
Yes
Spatial Case
Bilinear Constraints
Unique SS or PS Dyads
Spherical or Planar Case
Characteristic Sphere/Plane Constraint
3-Inf SS or PS Dyads

Figure 5.4: A step by step visualization of proposed algorithm to solve motion synthesis problem for seven or more spatial poses

5.6 Dyad Calculation for Five Spatial Poses

It is well known that a spherical or planar four-bar RRRR mechanism can pass exactly through a maximum of five poses. The MP of their solution RR dyads is constrained to lie on a circle. For planar mechanisms, there exist up to four circle constraints while for spherical mechanisms, there exist up to six circle constraints for the five pose problem. Thus, to synthesize planar and spherical mechanisms, we need to calculate these circle constraints.

In previous sections, we discussed the generation of spherical/plane constraints that represent SS/PS dyads. A spatial circle constraint, representing an RR dyad, can be calculated as a sphere-sphere intersection or a plane-sphere intersection. For spherical mechanisms, the RR dyad can be represented as an intersection of the characteristic sphere of poses and a sphere whose center lies on the surface of the characteristic sphere. Similarly, for planar mechanisms, an RR dyad can be represented as an intersection of the characteristic plane of poses and a sphere whose center lies on the surface of the characteristic plane. These cases have been visualized in Fig. 5.5. Note, that the MP coordinates of the two distinct intersecting primitives need to be equal (and consequently lie on a circle) for the intersection to be a valid RR dyad. In this section, we first find the characteristic sphere/plane and then find the unique spherical constraints that are located on the characteristic sphere/plane by imposing appropriate constraints on the solution space.


(a) Spherical RR dyad

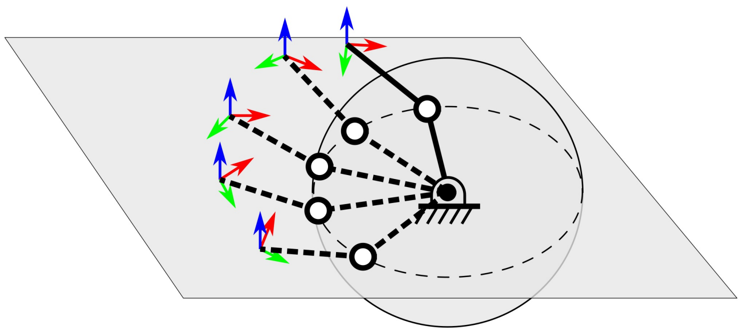

(b) Planar RR dyads

Figure 5.5: Representing a circle constraint as a sphere-sphere intersection or a plane-sphere intersection

First, the five pose constraints are applied using null-space analysis on Eq (5.15) and resulting in a 12-dimensional solution space [**V1**] as follows

$$\mathbf{P} = \sum_{i=1}^{12} \alpha_i v1_i = \begin{bmatrix} v1_1 & v1_2 & \cdots & v1_{12} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{12} \end{bmatrix} = [\mathbf{V1}]\alpha. \tag{5.36}$$

This is in contrast to the 10-dimensional solution space [**V**] as seen in Eq. (5.17).

Next, we check if the five spatial poses lie on a sphere/plane by finding their characteristic-sphere or -plane. We follow the same approach of calculating the dyad with zero-length coupler as outlined in Sec. 5.5 by finding the null-space of the following system of equations

$$\begin{bmatrix} P_1 \\ \vdots \\ P_{12} \end{bmatrix} = \begin{bmatrix} v1_{1,1} & \cdots & v1_{12,1} \\ \vdots & & \vdots \\ v1_{1,12} & \cdots & v1_{12,12} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{12} \end{bmatrix} = 0 \tag{5.37}$$

If a null-space exists, the radius/intercept and FP coordinates of the characteristic sphere/plane can be calculated by back-substituting the values of α_i in Eq. (5.36). In this case, there exists a possibility of existence of RR dyads.

If a null-space does not exist for Eq. (5.37), the five poses problem only have SS/PS dyads and no RR dyads. Since the problem has two fewer pose constraints than seven poses, there exist at least 2-∞ SS or PS dyads. These dyads can be calculated by enforcing bilinear constraints using Eq. (5.25) with the following changed dimensionality

$$\begin{bmatrix} W_1 - \lambda_1 W_4 \\ W_2 - \lambda_2 W_4 \\ W_3 - \lambda_3 W_4 \end{bmatrix}_{12\times 12} \alpha_{12\times 1} = [\mathbf{W1}]\alpha = 0 \tag{5.38}$$

where W_1, W_2, W_3, W_4 submatrices are calculated from $[\mathbf{V1}]$ in Eq. (5.36). The $[\mathbf{W1}]$ matrix, in this case, has a dimension of 12×12. Constraining the determinant of this 12×12 matrix to zero results in a single constraint on variables $(\lambda_1, \lambda_2, \lambda_3)$. We can plug in two free variables and the third can be calculated to find a solution λ_i .Thus, 2-∞ solution SS dyads can be calculated for five general spatial poses.

In the previous section, we discussed the existence of the 3-∞ solution space when the spatial poses lie on a sphere or plane. However, when a planar or spherical RR dyad exists, an additional 2-∞ solution space exists. It is well known that for spherical RR dyad, the FP and MP can be located anywhere on the axis that passes through the center of the characteristic sphere. Similarly, for a planar RR dyad, moving the FP and MP coordinates perpendicular to the characteristic plane does not affect the motion of the coupler point. This family of solutions has been visualized in Fig. 5.3, where the cyan arrows represent the free directions in which FP and MP coordinates are free to move while satisfying the pose constraints.

Out of these 2-∞ spherical solutions, our aim is to isolate the spherical constraint whose FP and MP lies on the characteristic sphere/plane. We know that both FP and MP have 1-∞ solution multiplicity each. Since $\lambda_1, \lambda_2, \lambda_3$ represent MP coordinates in Eq. (5.38), we can constrain 1-∞ solutions by fixing $\lambda_3 = 1$ in matrix $[\mathbf{W1}]$. Due to fixing a free variable, Eq. (5.38) now contains 1-∞ solution and thus its rank should be 10. To find this solution space, we set the determinant of two 11×11 sub-matrices of $[\mathbf{W1}]$ to zero and find the solution set for λ_1, λ_2. We observe that there exist upto six solutions for spherical case and four solutions for planar case which is in agreement with common knowledge for five pose synthesis. λ_1 and λ_2 are back-substituted in Eq. (5.38) and all two-dimensional null-spaces of $[\mathbf{W1}]$ are calculated as sets of α_i. These two-dimensional α_i are back-substituted in Eq. (5.36) to get two-dimensional solution spaces for CSPs P_i.

Randomly sampling a single dyad from each of the solution spaces gives us all the unique RR dyads denoted by their FP and MP coordinates. The axis of rotation for spherical RR dyad is the line joining FP and center of the characteristic circle. For a planar RR dyad, the axis of rotation is the line passing through FP in the direction perpendicular to the characteristic plane. However, these unique dyads don't lie on the characteristic sphere/surface since we have randomly sampled them. Since the equation of characteristic sphere/plane and the axis of rotation is known, it is fairly trivial to find the required FP. To find the MP coordinates on the characteristic sphere/plane, the coordinates need to be transformed from moving frame to fixed frame before they can be projected onto the characteristic sphere/plane. Thus, we can successfully find planar RR dyads using the spatial motion synthesis algorithm. The complete algorithm for five spatial pose synthesis is displayed in Fig. 5.6.

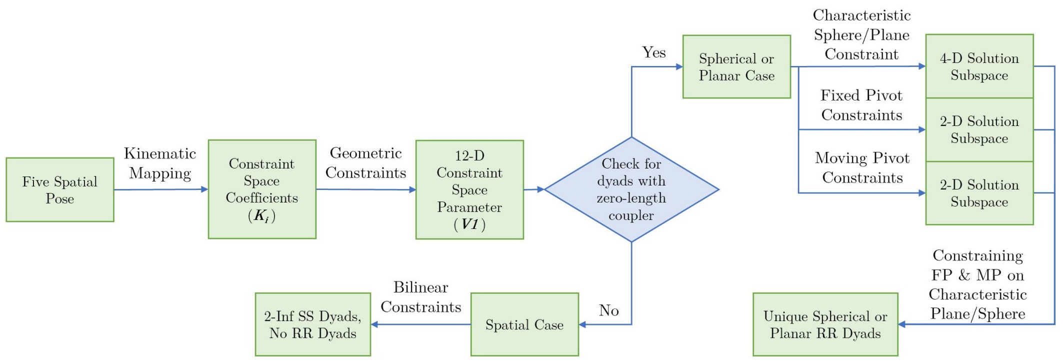

Five Spatial Pose
Kinematic Mapping
Constraint Space Coefficients (K_i)
Geometric Constraints
12-D Constraint Space Parameter (VI)
Check for dyads with zero-length coupler
Yes
Spherical or Planar Case
Characteristic Sphere/Plane Constraint
4-D Solution Subspace
Fixed Pivot Constraints
2-D Solution Subspace
Moving Pivot Constraints
2-D Solution Subspace
Constraining FP & MP on Characteristic Plane/Sphere
Unique Spherical or Planar RR Dyads
No
Spatial Case
Bilinear Constraints
2-Inf SS Dyads, No RR Dyads

Figure 5.6: A step by step visualization of proposed algorithm to solve motion synthesis problem five spatial poses

5.7 Examples

5.7.1 Example 1: Motion synthesis for seven general poses

To test our algorithm for accuracy we put it to test using the example from Innocenti's paper [82]. The pose quaternions are given in Table 5.1 and the twenty output dyads are given in Table 5.2, which match with the original results. The input poses are shown in Fig 5.7. A 5-SS mechanism is created using the smallest radius spheres and its geometric model is shown in Fig. 5.8. As can be observed from Table 5.2, the output from the proposed algorithm agrees up to four digits after decimal with results from Innocenti's paper.

5.7.2 Example 2: Motion synthesis for seven spherical poses

To validate the multiplicity of solutions in case where spatial poses lie on a sphere, we choose seven poses as shown in Fig. 5.9a and given in Table 5.3.

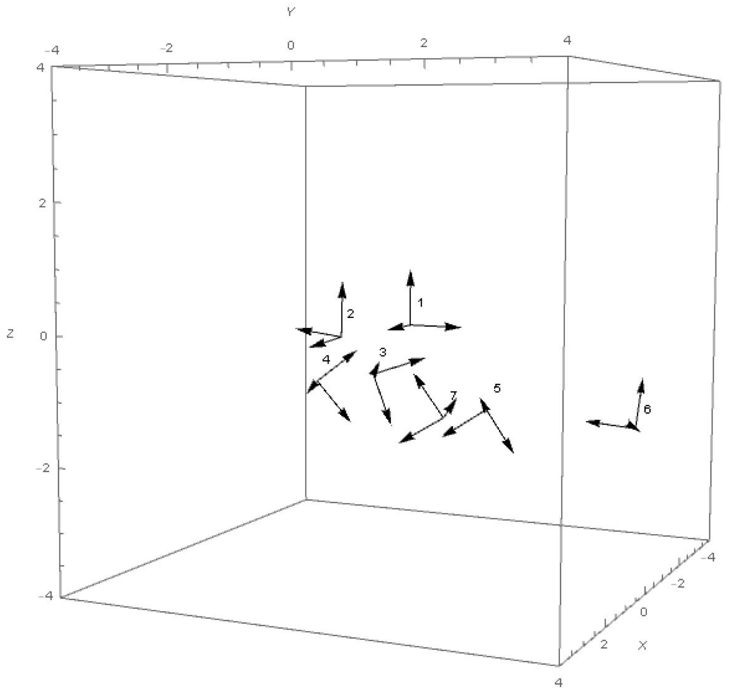


Figure 5.7: Input spatial poses for example 1

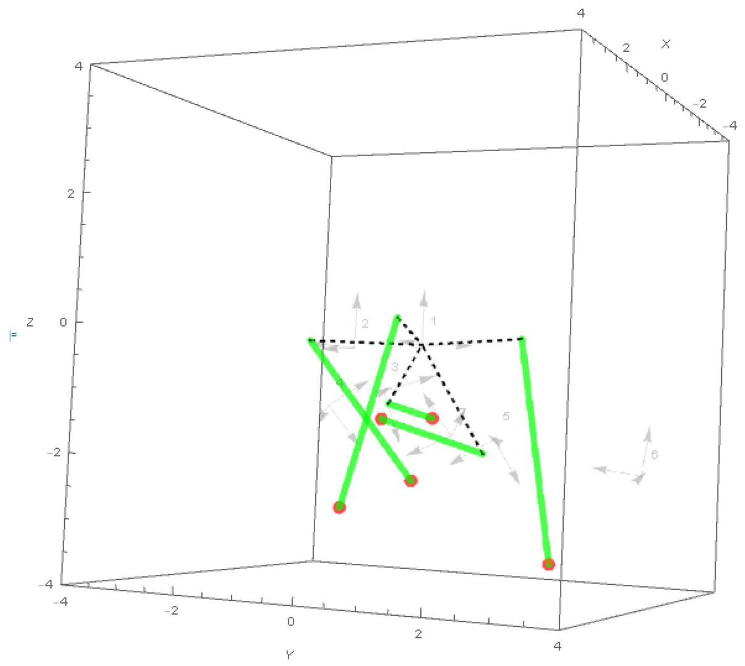


Figure 5.8: Five of the total twenty output spherical constraints for example 1. The green links have their FP located at their orange ends while the other end is the MP. The dotted lines represent the floating link.

Table 5.1: Seven Input Poses in Dual Quaternion representation for Example 1

q_1	q_2	q_3	q_4	g_1	g_2	g_3	g_4
0	0	0	1	0	0	0	0
-0.0651026	0.00725456	-0.75133	0.656667	0.607675	0.136295	-0.0644337	-0.0149826
-0.298121	-0.59615	0.526405	0.527851	-0.285992	-0.0998713	-0.379727	0.104369
-0.376624	0.859028	0.339364	0.0711109	0.138837	0.163292	-0.449864	0.90963
-0.582787	-0.346858	0.734635	0.018995	0.0351985	0.906884	0.447598	0.329131
0.00431816	-0.0844728	-0.561614	0.823064	-2.33102	0.1092	-0.690907	-0.448
0.821544	0.147805	0.372509	0.405532	0.205892	-0.485491	-0.539083	0.255028

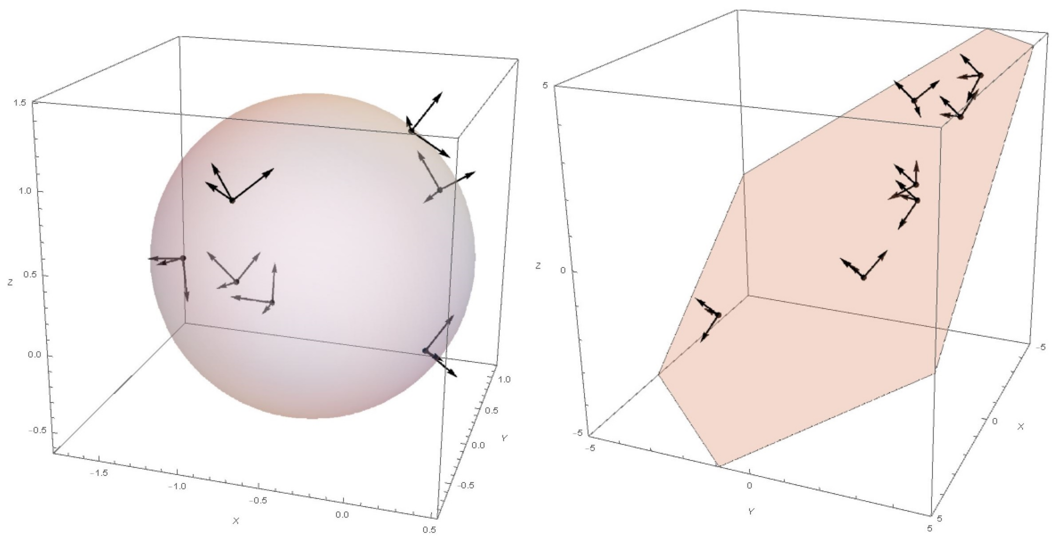
Table 5.2: Synthesized Dyads for example 1

Dyad	A_1	A_2	A_3	A_4	C_1	C_2	C_3	C_4	r
1	-1.4532	-0.4131	-1.1781	1	1.3606	0.0849	-1.0281	1	2.8615
2	-0.3764	-0.2694	-2.255	1	2.1434	-0.9266	-0.1025	1	3.3786
3	-0.4049	-0.8841	-1.2398	1	1.6293	1.8376	-1.7463	1	3.4354
4	0.8106	-0.9742	-2.7163	1	2.3575	0.664	0.2455	1	3.7214
5	0.2611	2.4586	-3.4243	1	-1.5559	1.1521	0.2328	1	4.2874
6	0.9736	2.9071	-3.0424	1	-0.9779	1.0619	0.4361	1	4.3947
7	-0.1484	2.679	-0.4008	1	0.161	-0.6354	2.4776	1	4.4007
8	1.2796	0.716	-1.2142	1	-0.6459	4.1423	0.9059	1	4.4657
9	-3.4211	-0.2941	1.4625	1	0.7026	-0.3223	-0.6563	1	4.6363
10	-3.8199	-3.7259	4.3852	1	0.8758	2.4775	2.7771	1	7.9447
11	-2.5613	-4.1576	-8.7597	1	-0.3029	0.2047	-0.9218	1	9.25
12	-4.0709	-2.56	-3.6963	1	-1.3247	-7.2043	5.0687	1	10.2926
13	0.8993	-0.907	0.1315	1	3.4589	3.3523	-9.6642	1	10.984
14	-7.8388	-0.0889	9.6483	1	0.3671	-0.5878	-0.9344	1	13.4007
15	-7.7369	-9.6348	-10.4396	1	-0.4714	1.5845	-1.9814	1	15.8177
16	0.073	-0.5606	0.2413	1	-5.3932	3.2032	25.0842	1	25.7141
17	-3.2431	-34.9651	-7.2184	1	5.4831	-5.0918	14.7174	1	38.0754
18	-48.9413	-37.5477	-43.9745	1	-0.0682	5.1441	-4.516	1	75.9482
19	-7.9651	2.5179	-4.8166	1	51.361	26.9419	-62.1902	1	86.0688
20	75.4808	37.5827	-87.3605	1	-43.3276	-113.601	-109.994	1	193.608

As discussed in the chapter, we can calculate a four dimensional output space which is given in Table 5.4. The characteristic sphere for these poses is centred at $(-0.512653, 0.166042, 0.402181)$ and has a radius of 1. The four-dimensional solution space represents a family of spherical constraints which are coincident with the characteristic sphere.

Table 5.3: Input Spherical Poses in Dual Quaternion representation for example 2

q_1	q_2	q_3	q_4	g_1	g_2	g_3	g_4
0.0359951	0.595485	0.624179	0.504483	0.100504	0.191117	0.198061	-0.477817
-0.863559	-0.298669	-0.404926	0.0331426	-0.131388	0.157084	0.171486	0.0873328
-0.386719	0.256693	-0.25578	-0.848018	0.272862	-0.0203728	-0.628229	0.0588879
-0.00189387	0.302044	0.16847	0.938288	-0.136238	0.121647	0.58056	-0.143674
-0.762556	0.141124	0.630785	0.0265215	0.0877535	0.391824	0.0457285	-0.649417
0.165336	-0.883582	-0.393837	0.191934	-0.346006	-0.134437	0.347322	0.391852
0.0313552	0.853597	0.304762	-0.421319	0.388445	0.0337678	-0.516785	-0.276495



(a) Seven poses lying on a sphere (b) Seven poses lying on a plane

Figure 5.9: Input spatial poses for example 2 and 3

Table 5.4: Synthesized CSP Solution space for example 2

P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8
0.315904	0.329053	-0.106576	-0.258145	-0.641863	-0.169279	0.0548276	0.132801	0.330202
-0.0264274	-0.199656	0.0646664	0.156632	0.389457	-0.374148	0.121182	0.293523	0.729827
-0.740367	0.171908	-0.0556789	-0.134863	-0.33533	-0.0877932	0.0284352	0.0688747	0.171253
-0.314513	-0.0573885	0.0185875	0.0450219	0.111944	0.068077	-0.0220494	-0.0534071	-0.132794
P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}	
-0.0380594	0.012327	0.029858	0.0742401	-0.156662	0.0507411	0.122903	0.305591	
0.014512	-0.00470027	-0.0113848	-0.0283077	-0.0282476	0.00914906	0.0221605	0.0551008	
0.148797	-0.0481936	-0.116733	-0.290249	0.149594	-0.0484517	-0.117358	-0.291804	
0.17701	-0.0573316	-0.138866	-0.345283	-0.351882	0.11397	0.276055	0.686394	

5.7.3 Example 3: Motion synthesis for seven planar poses

Similar to the previous example, in this example, we validate the multiplicity of solutions in cases where spatial poses lie on a plane. The seven input poses as shown in Fig. 5.9b and given in Table 5.5. As discussed in the chapter, we can calculate a four-dimensional output space which is given in Table 5.6. The characteristic plane has the normal vector as (0.513483,-0.48064,0.710859) and the intercept as 0.549628. It can be observed that the solution space is a family of plane constraints which are parallel to the characteristic plane.

Table 5.5: Input Planar Poses in Dual Quaternion representation for example 3

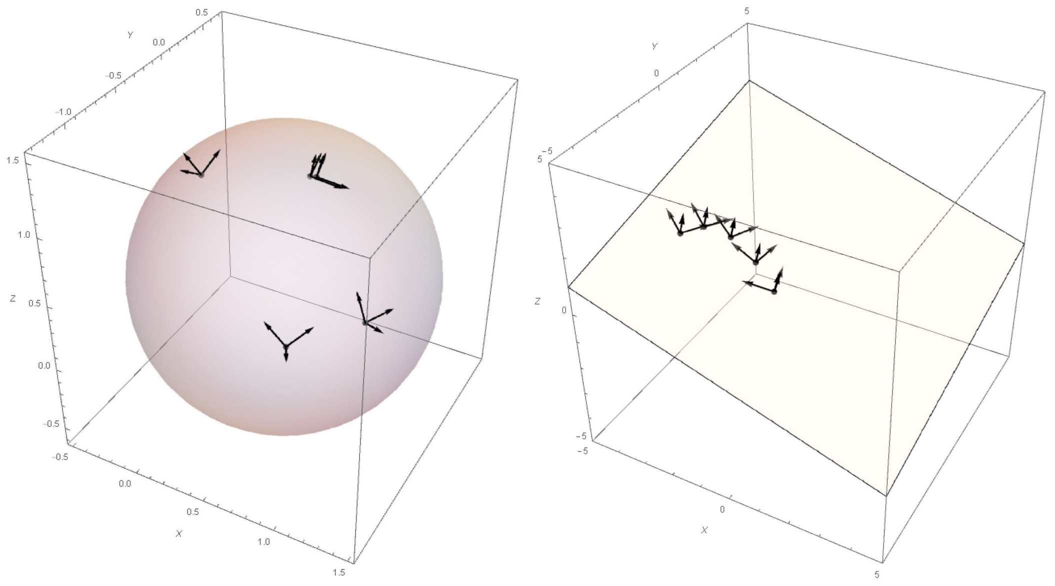
q_1	q_2	q_3	q_4	g_1	g_2	g_3	g_4
0.355749	-0.134214	0.854875	0.353013	0.980586	-0.376976	-0.51362	0.112298
0.314093	0.214281	0.201594	0.902657	-1.91212	1.4711	0.697185	0.16042
0.255202	-0.281855	0.92174	-0.0763208	1.39753	0.435009	-0.272715	-0.227058
0.119561	-0.360937	0.812314	-0.44224	1.38602	-0.460968	-0.403417	0.0099337
0.109936	-0.363984	0.800286	-0.46365	-1.96301	0.554419	0.893871	0.642179
0.33642	-0.177179	0.891969	0.244581	1.46594	1.92175	0.0213357	-0.702059
0.0836475	-0.370909	0.765112	-0.519647	2.95867	0.534235	-0.570964	-0.745735

Table 5.6: Synthesized CSP Solution space for example 3

P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8
-0.286692	0.44113	-0.412914	0.610694	0	0.0852415	-0.0797893	0.118007	0
0.630011	0.25607	-0.239692	0.3545	0	-0.12697	0.118848	-0.175775	0
-0.0506416	-0.00602674	0.00564125	-0.00834332	0	-0.471286	0.441141	-0.652441	0
0.597709	-0.0588312	0.0550682	-0.081445	0	0.134788	-0.126166	0.186598	0
P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}	
0.147883	-0.138424	0.204727	0	0.13514	-0.126496	0.187086	0	
-0.239448	0.224133	-0.331488	0	-0.141365	0.132323	-0.195703	0	
0.107174	-0.100319	0.14837	0	0.171337	-0.160378	0.237197	0	
0.0556503	-0.0520908	0.0770414	0	0.380452	-0.356117	0.526692	0	

5.7.4 Example 4: Motion synthesis for five spherical poses which fall on a circle constraint

To verify the correct working of the proposed algorithm, we use spherical poses from an example used by Zhuang et al. [81] and transform them to an arbitrary spatial sphere. The poses are translated by $(.5, -.3, .4)$ and rotated by Euler angles $(15°, 15°, 15°)$. The input poses are given in Table 5.7 and have been plotted in Fig 5.10a. We obtain six solutions representing each spherical RR dyad as given in Table 5.10. Each of the six solution dyads is shown in Fig 5.12. As can be observed from Table 5.10, the output from the proposed algorithm agrees up to four digits after decimal with results from the paper by Zhuang et. al. [81]



(a) Five poses lying on a sphere (b) Five poses lying on a plane

Figure 5.10: Input spatial poses for example 4 and 5

Table 5.7: Input Spatial Poses in Dual Quaternion representation for example 4

q_1	q_2	q_3	q_4	g_1	g_2	g_3	g_4
0.111411	0.145194	0.111411	0.976777	0.271041	-0.207793	0.736754	-0.0840615
0.616384	-0.049129	0.433949	0.655245	0.0839802	-0.391689	0.538847	-0.465229
0.537275	0.248421	0.428057	0.682928	0.18105	-0.370636	0.620746	-0.396696
0.115335	0.131981	0.0830842	0.981008	0.272383	-0.202523	0.737001	-0.0671956
0.0409948	-0.261604	0.605156	0.750779	0.0184403	-0.276204	0.466294	-0.473098

Table 5.8: Synthesized six spherical RR dyadic parameters for example 4

A_1	A_2	A_3	A_4	C_1	C_2	C_3	C_4	r
0.579085	0.695201	0.457631	1	0.56608	0.470253	-0.234306	1	0.7277
0.661773	-1.00196	1.0936	1	0.586939	-0.471541	1.38133	1	0.608065
0.73826	0.0664105	-0.499431	1	0.530103	-0.589928	-0.556575	1	0.690923
0.752583	-0.38842	-0.563527	1	0.219313	-0.871137	-0.371374	1	0.744523
0.80072	0.0237179	1.29709	1	1.32461	-0.0332188	0.898845	1	0.660533
1.18109	0.426381	0.492128	1	1.15123	0.206906	-0.164748	1	0.693215

5.7.5 Example 5: Motion synthesis for five planar poses which fall on a circle constraint

For this example, we use planar poses from an example used by Ge et al. [21] and transform them to an arbitrary spatial plane. The poses are translated by $(.2, -.3, .2)$ and rotated by Euler angles $(15°, 15°, 15°)$. The poses

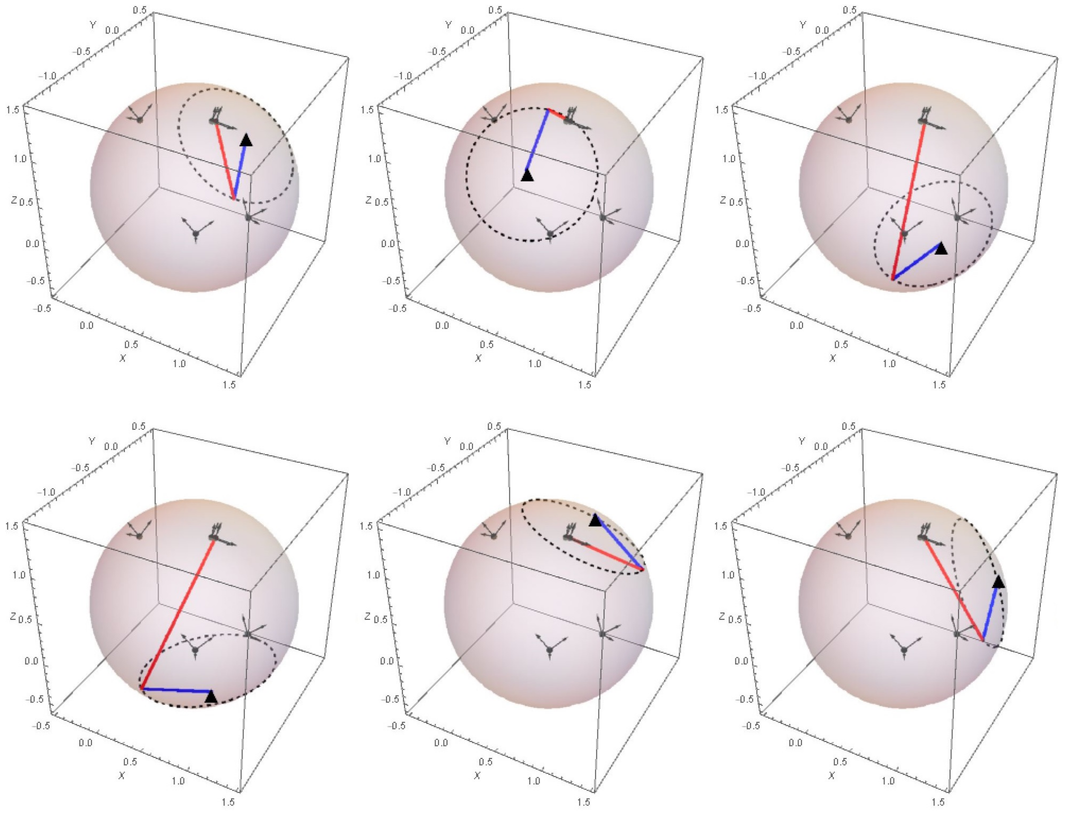


Figure 5.11: Set of six RR dyad solutions for example 4. The red link is the floating link while the blue link has one end fixed (FP) and the other end moving(MP).

are given in Table 5.9 and have been plotted in Fig 5.10b. We obtain four sets of solution spaces representing each planar RR dyad as given in Table 5.10. Each of the four solution dyads is shown in Fig 5.12. As can be observed from Table 5.10, the output from the proposed algorithm agrees upto four digits after decimal with results from the paper by Ge et al. [21]

Table 5.9: Input Spatial Poses in Dual Quaternion representation for example 5

q_1	q_2	q_3	q_4	g_1	g_2	g_3	g_4
0.18182	0.0208603	0.779558	0.598996	-0.694984	-0.283932	0.177855	-0.010623
0.173037	0.0595978	0.63196	0.753083	-0.811229	0.0912475	0.256593	-0.0361472
0.162258	0.084653	0.513016	0.838642	-1.18682	0.385687	0.358205	-0.0284309
0.157636	0.0929757	0.468714	0.864184	-1.61222	0.444565	0.435748	0.00991535
0.158956	0.0907013	0.481082	0.85736	-1.93825	0.297065	0.472565	0.0627621

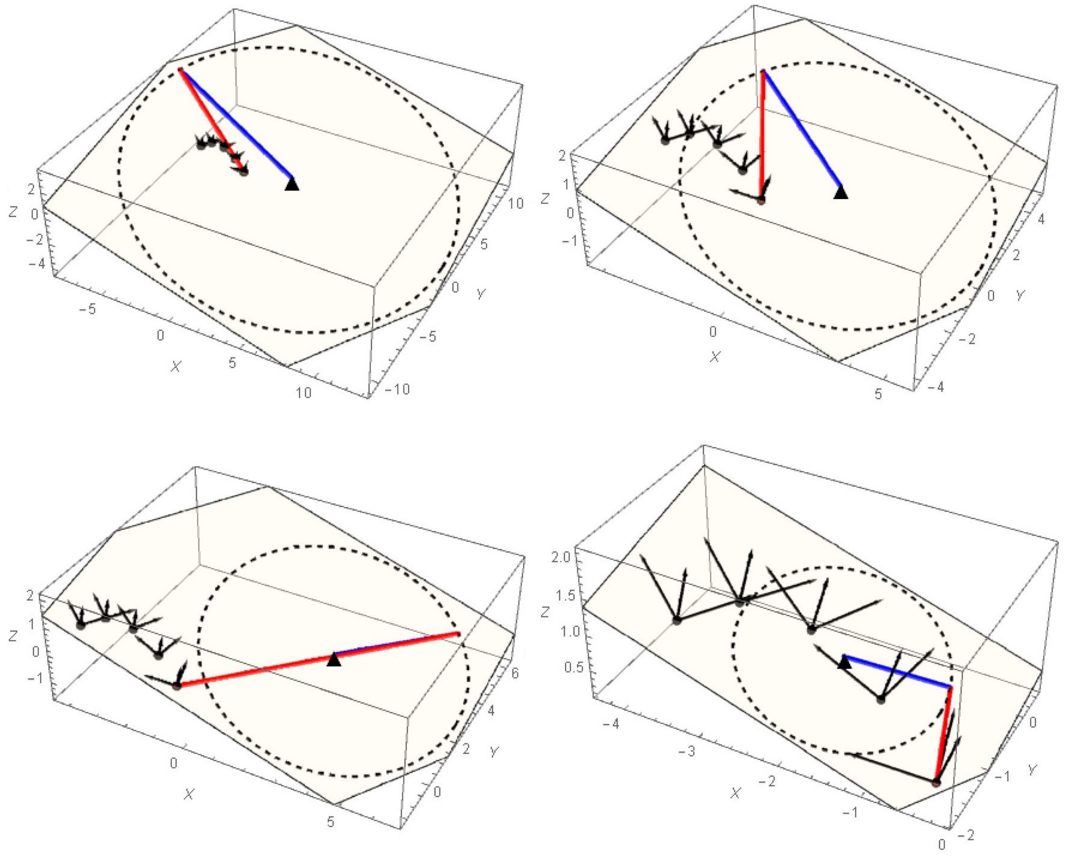


Figure 5.12: Set of four RR dyad solutions for example 5. The red link is the floating link while the blue link has one end fixed (FP) and the other end moving(MP).

Table 5.10: Synthesized four planar RR dyadic parameters for example 5

A_1	A_2	A_3	A_4	c_1	c_2	c_3	c_4	r
2.35097	0.0675858	-0.438144	1.0000	6.45577	6.50408	0	1.0000	11.199
1.20255	0.348633	-0.0026419	1.0000	4.58995	1.34007	0	1.0000	4.6712
2.73061	2.27873	-0.124526	1.0000	5.03163	-7.40889	0	1.0000	4.06469
-1.87116	-0.754965	0.794402	1.0000	1.24	0.1	0	1.0000	1.23773

5.8 Conclusion

In this chapter, we have proposed a methodology to unify the spatial, spherical, and planar motion synthesis problems. We use the algebraic fitting algorithm and intelligently constrain the solution spaces to find all possible RR dyads for spherical and planar four-bar mechanisms and SS dyads for 5-SS spatial mechanisms. The algorithm also provides deeper insights in the nature of solution space generated while performing spatial synthesis of mechanisms.

Chapter 6

Path Synthesis of Defect-free Spatial 5-SS Mechanisms using Machine Learning

6.1 Introduction

Path synthesis problem is the determination of the dimensions of a kinematic mechanism to guide one of its links through many specified points [12, 65]. Extensive research has been done to solve the path generation problem for planer mechanisms. Analytical methods for synthesis include algebraic methods [92, 93, 94, 59], complex number methods [95] and displacement matrix methods [1]. Optimization-based techniques attempt to minimize an objective function and find mechanisms, which best approximate a curve [2, 96, 4, 6, 61, 74]. Atlas-based approaches explore the use of curve invariants like Fourier descriptors to intelligently form and search a database of coupler curves [27, 97].

A substantial literature on the synthesis of Spherical mechanisms is also available [76, 98, 99]. 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 [100, 101]. Ananthasuresh and Kramer solve the synthesis of the RSCR spatial mechanism using the Generalized Reduced Gradient method of optimization [102]. Jiménez et al. outline a generalized constraint based optimization technique [103]. Sun et al. use an atlas-based approach which uses the Fourier series to compare curves and synthesize RCCC spatial mechanism [104]. In this chapter, 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. 6.1.

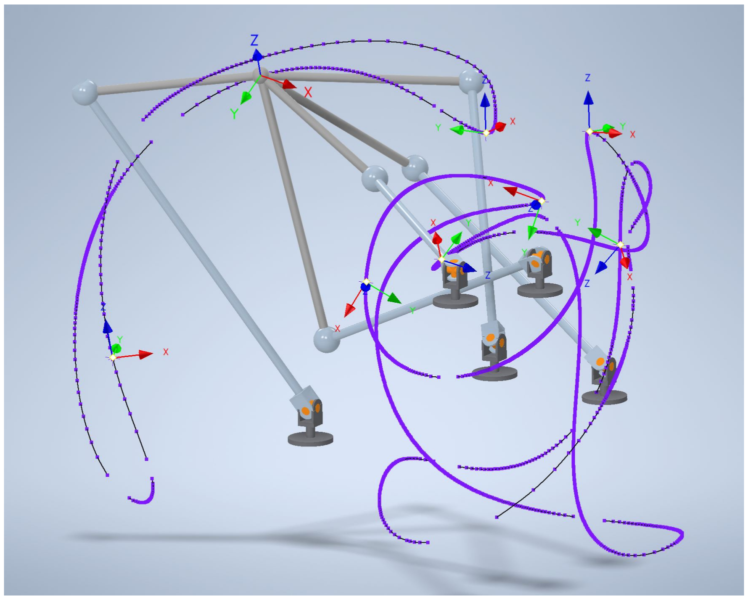


Figure 6.1: Sample 5-SS mechanism

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[105]. Roth and Freudenstein have discussed the occurrence of defects in mechanism synthesis for path generation using numerical methods [106]. Wampler et al. show that there exist many defective mechanisms for the nine-point path synthesis problem [107]. These defects tend to be more prominent in spatial mechanisms when compared to planar mechanisms. Thus, to synthesize practical spatial mechanisms, we consider path synthesis for defect-free mechanisms.

In this chapter, 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 chapter 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.

6.2 Data Generation

For a machine learning based approach to work, a large amount of data is required. In this chapter, 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 [108] given in Eq. (6.1).

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

$$C_{SS} : 2a_1c_1 + 2a_2c_2 + 2a_3c_3 + a_0c_4 = a_4\left(\frac{c_1^2 + c_2^2 + c_3^2}{c_4}\right), \tag{6.1}$$

where a_0 is given as

$$a_0 = a_4 r^2 - \frac{a_1^2 + a_2^2 + a_3^2}{a_4}. \tag{6.2}$$

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 [109]. The length of the actuator imposes an additional constraint on the motion and can be defined using Eq (6.1) 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

$$\mathbf{\Phi}(\mathbf{q}) = 0 \tag{6.3}$$

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

$$\mathbf{q}_{i+1} = \mathbf{q}_i - [\boldsymbol{J}^{-1}(\mathbf{q}_i)]\boldsymbol{\Phi}(\mathbf{q}_i) \tag{6.4}$$

where $\mathbf{q}_i$ is the state vector at i^{th} iteration, $\boldsymbol{\Phi}(\mathbf{q}_i)$ 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

$$[\boldsymbol{J}(\mathbf{q})] = \begin{bmatrix} \frac{\partial \phi_1}{\partial q_1} & \frac{\partial \phi_1}{\partial q_2} & \cdots & \frac{\partial \phi_1}{\partial q_{16}} \\ \frac{\partial \phi_2}{\partial q_1} & \frac{\partial \phi_2}{\partial q_2} & \cdots & \frac{\partial \phi_2}{\partial q_{16}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial \phi_{16}}{\partial q_1} & \frac{\partial \phi_{16}}{\partial q_2} & \cdots & \frac{\partial \phi_{16}}{\partial q_{16}} \end{bmatrix}. \tag{6.5}$$

To calculate the Jacobian matrix, relations describing the first order partial derivatives of constraint equations are required. For an SS dyad described in Eq (6.1), the first order partial derivatives can be given as follows

$$\frac{\partial C_{SS}}{\partial a_1} = 2(a_1 - c_1) \tag{6.6}$$

$$\frac{\partial C_{SS}}{\partial a_2} = 2(a_2 - c_2) \tag{6.7}$$

$$\frac{\partial C_{SS}}{\partial a_3} = 2(a_3 - c_3) \tag{6.8}$$

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.

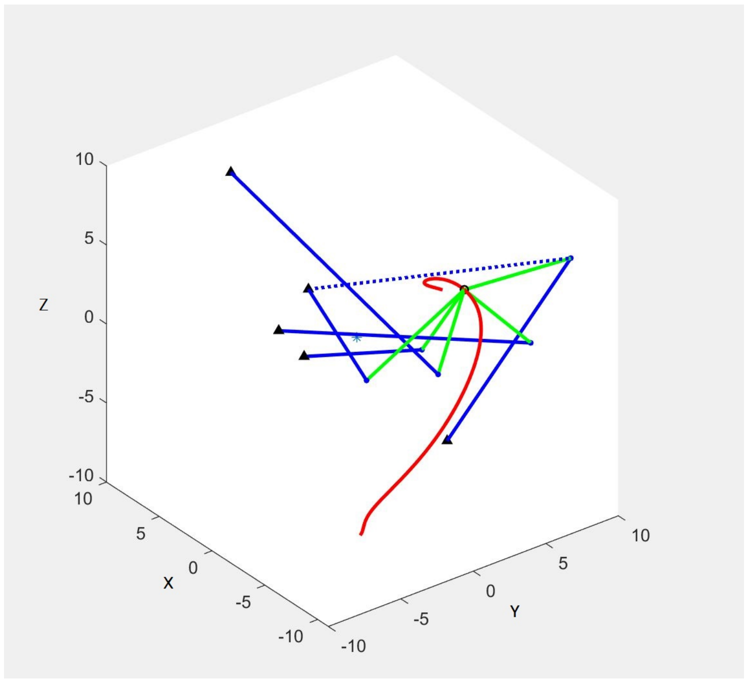


Figure 6.2: A 5-SS mechanism simulation where black triangles are the fixed pivots, blue solid lines are the SS dyads, green lines are the floating coupler link, red curve is the coupler path, and dotted blue line is the linear actuator.

For this chapter, we generated a data set of 7,500 defect-free coupler paths using arbitrarily selected 5-SS mechanisms. Fig. 6.2 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.

6.3 Data Preprocessing

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

6.3.1 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. 6.3. 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.

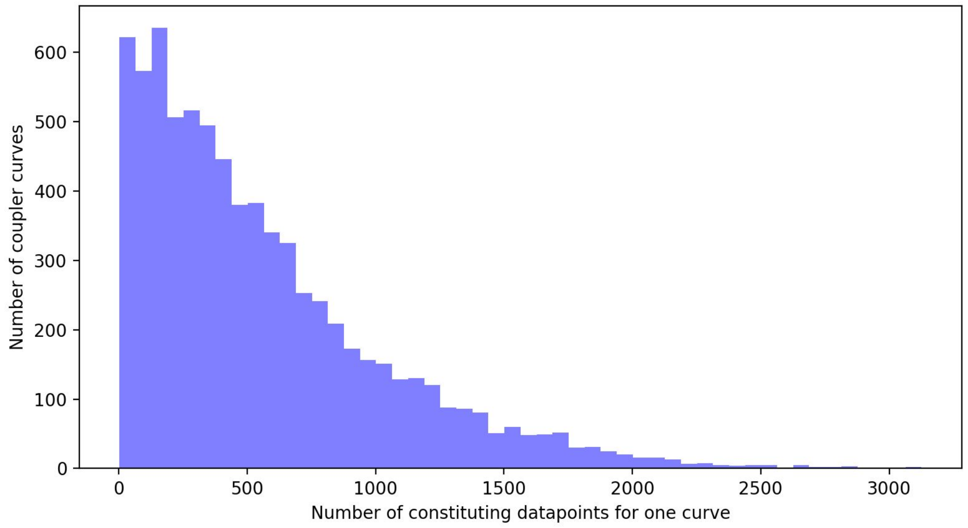


Figure 6.3: Histogram showing number of data points in each path curve included in database

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. 6.4.

6.3.2 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

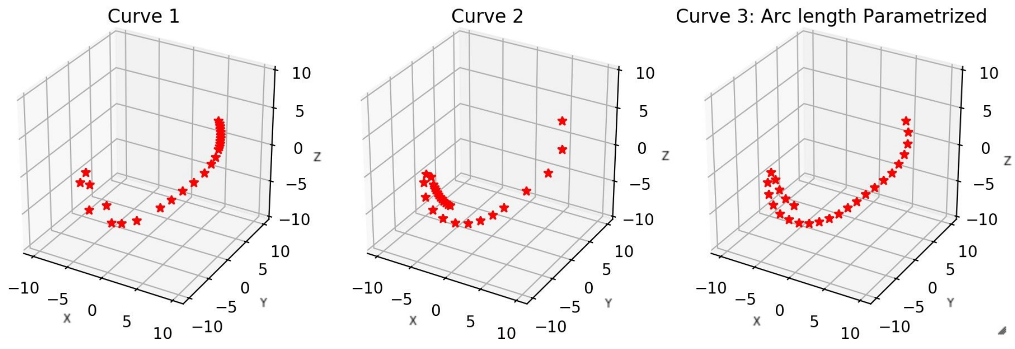


Figure 6.4: Curve 1 and Curve 2 represents the same geometric curve with different time parametrization. They share the same unique arc length parametrization as shown in Curve 3

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. 6.5.

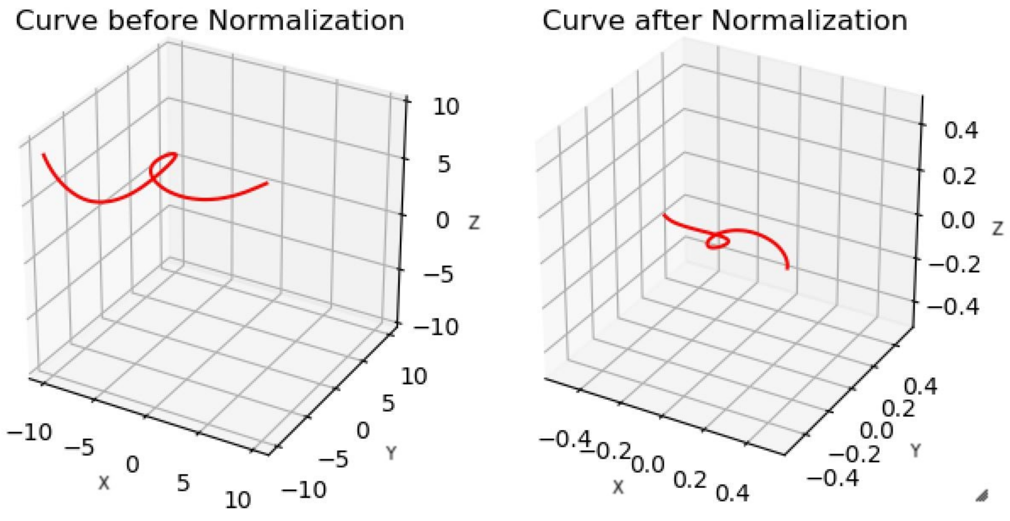


Figure 6.5: Before and after normalizing a path curve

6.3.3 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

$$\kappa = \frac{||\mathbf{r}'(t) \times \mathbf{r}''(t)||}{||\mathbf{r}'(t)||^3}, \tag{6.9}$$

$$\tau = \frac{(\mathbf{r}'(t) \times \mathbf{r}''(t)) \cdot \mathbf{r}'''(t)}{||\mathbf{r}'(t) \times \mathbf{r}''(t)||^2}, \tag{6.10}$$

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

$$z = \frac{X - \mu}{\sigma} \tag{6.11}$$

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. 6.6. Filtering out the outliers results in a clean database containing 7,200 coupler paths.

6.3.4 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,

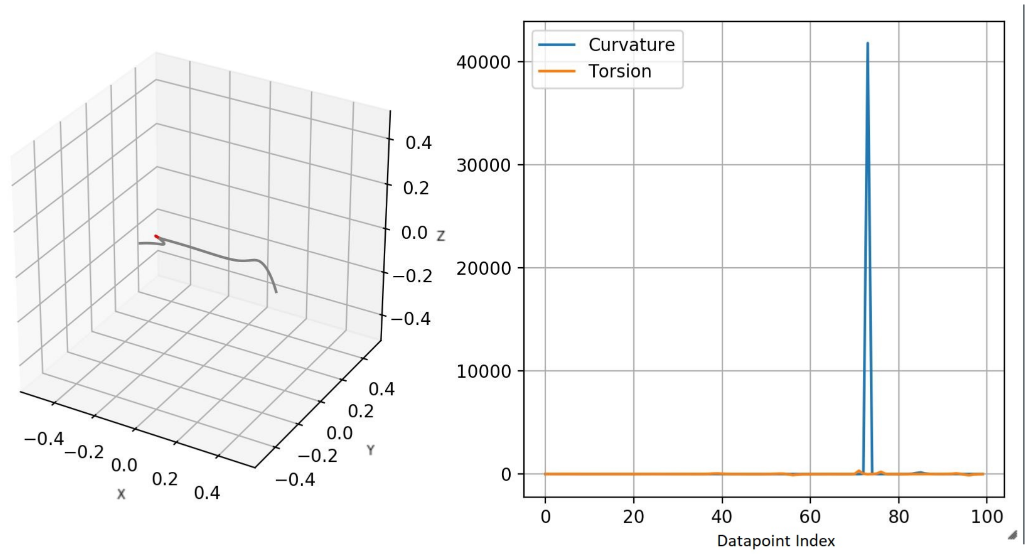


Figure 6.6: An outlier path curve with high curvature. The branch jump occurs at the red dot denoted on curve

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

$$\delta = \frac{||\kappa_2(s) - \kappa_1(s)||_2 + w||\tau_2(s) - \tau_1(s)||_2}{n} \tag{6.12}$$

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 chapter. 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. 6.7 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. 6.8. Under-sampling similar curves lead to a balanced data set containing 5,021 coupler paths.

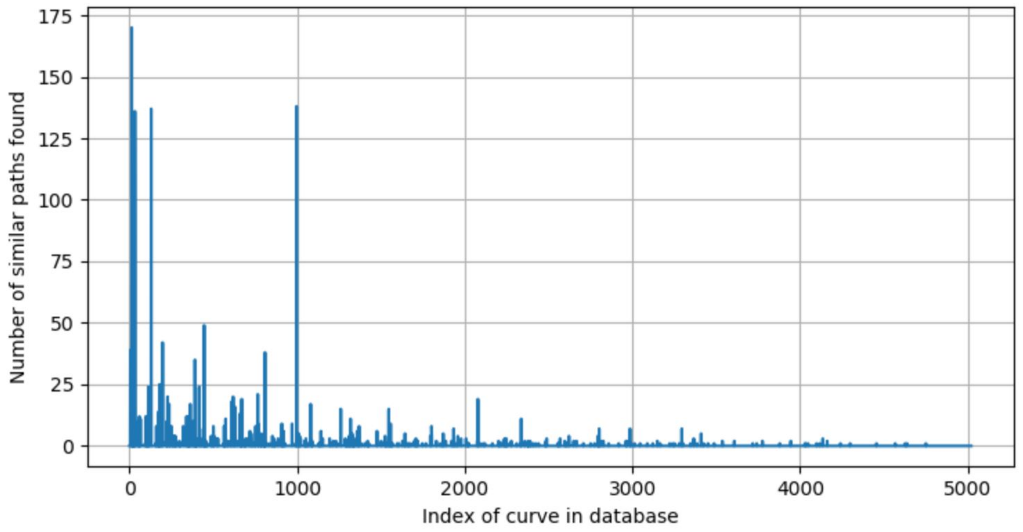


Figure 6.7: Bar graph showing the number of similar curves found for each curve.

6.3.5 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.

6.3.6 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.

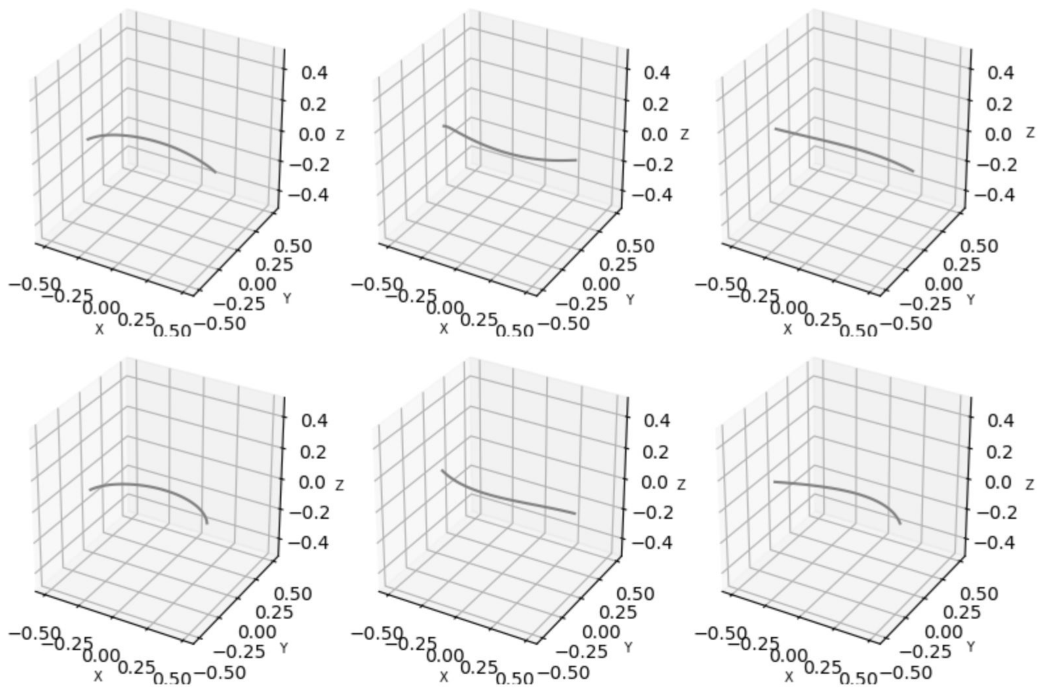


Figure 6.8: Most common variety of curves in the database

6.4 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.

6.4.1 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. 6.9, 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.

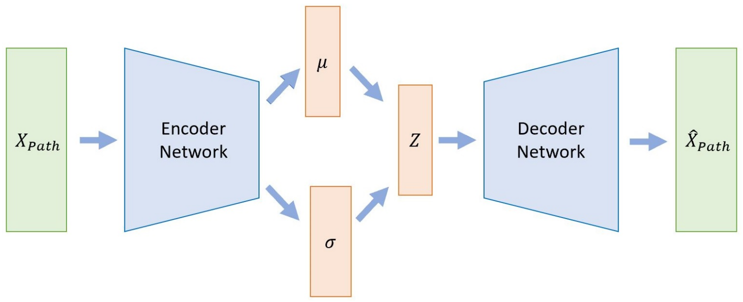


Figure 6.9: Architecture of a Variational Autoencoder

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

$$Loss = RL + KL \tag{6.13}$$

$$RL = \sqrt{\sum_{i=1}^{n} d_i^2} \tag{6.14}$$

$$d_i = \sqrt{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2 + (\hat{z}_i - z_i)^2} \tag{6.15}$$

$$KL = \sum_{i=1}^{k} \sigma_i^{2\text{‘}} + \mu_i^2 - log(\sigma_i) - 1 \tag{6.16}$$

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 6.1. Each VAE is trained for 3000 epochs with a batch size of 256 using Adam (adaptive moment estimation) optimizer.

Table 6.1: 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

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. 6.10.

The image processing community has successfully used convolution layers for feature extraction to enhance the performance of image classification algorithms [110]. 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.

For spatial curves, the local geometry of a curve segment is heavily dependent on previous and next curve segments due to the continuity constraints.

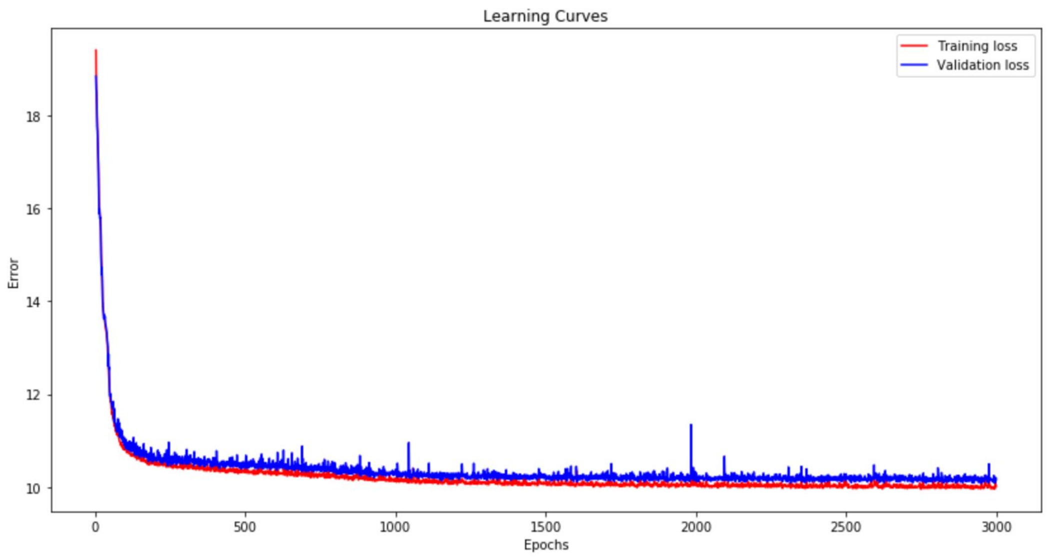


Figure 6.10: Training losses for the fully connected VAE

This is especially true in the case of interpolation curves like B-splines where each control point has local effect [111]. 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. 6.11. We find that this model performs even better and has a training loss of 9.9034 and a validation loss of 10.0311.

Some sample outputs generated using VAE-COV-H8-Z30 are shown in Fig 6.12 where an input curve (red) generates five trajectories (gray) sampled from the underlying distribution.

6.4.2 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,

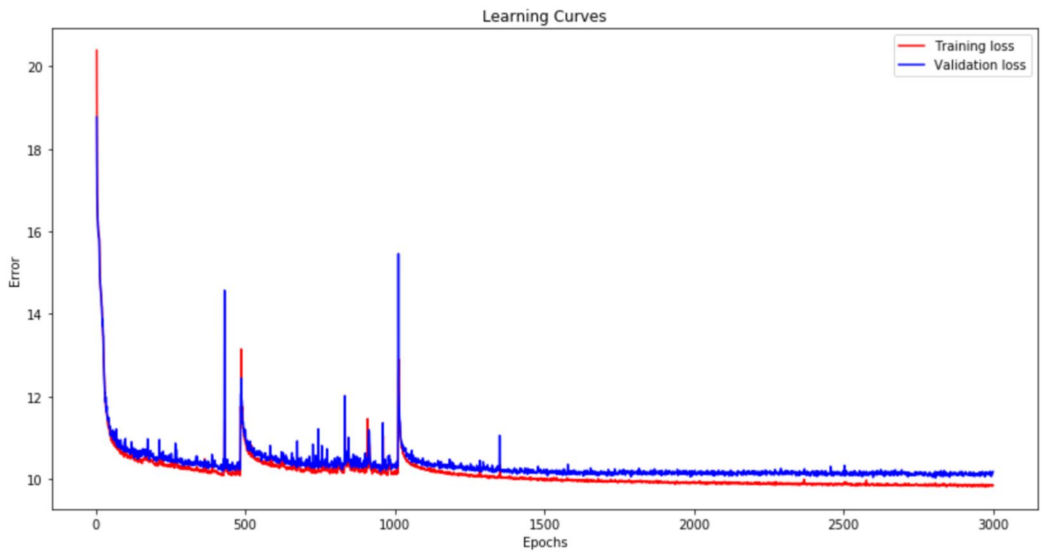


Figure 6.11: Training losses for the Convolutional VAE

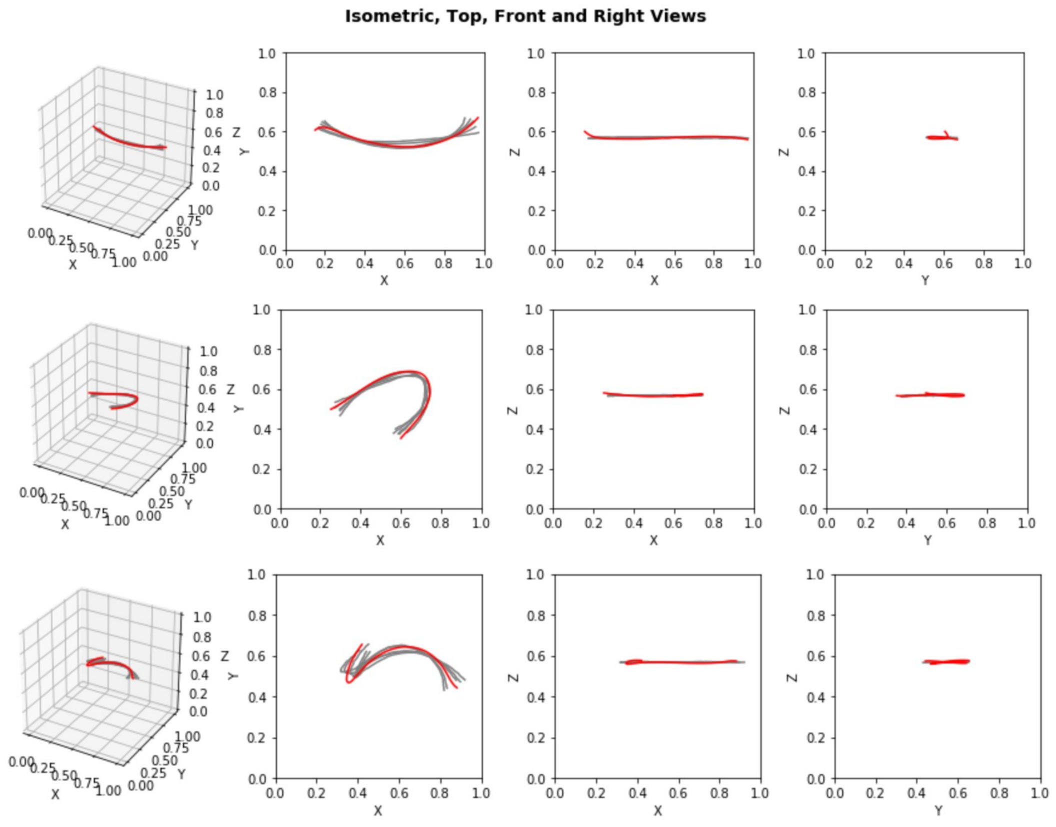


Figure 6.12: Comparing X (red curve) and $\hat{X}$ (grey curves)

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.

6.4.3 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.

6.5 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. 6.13 and Fig. 6.14. 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.

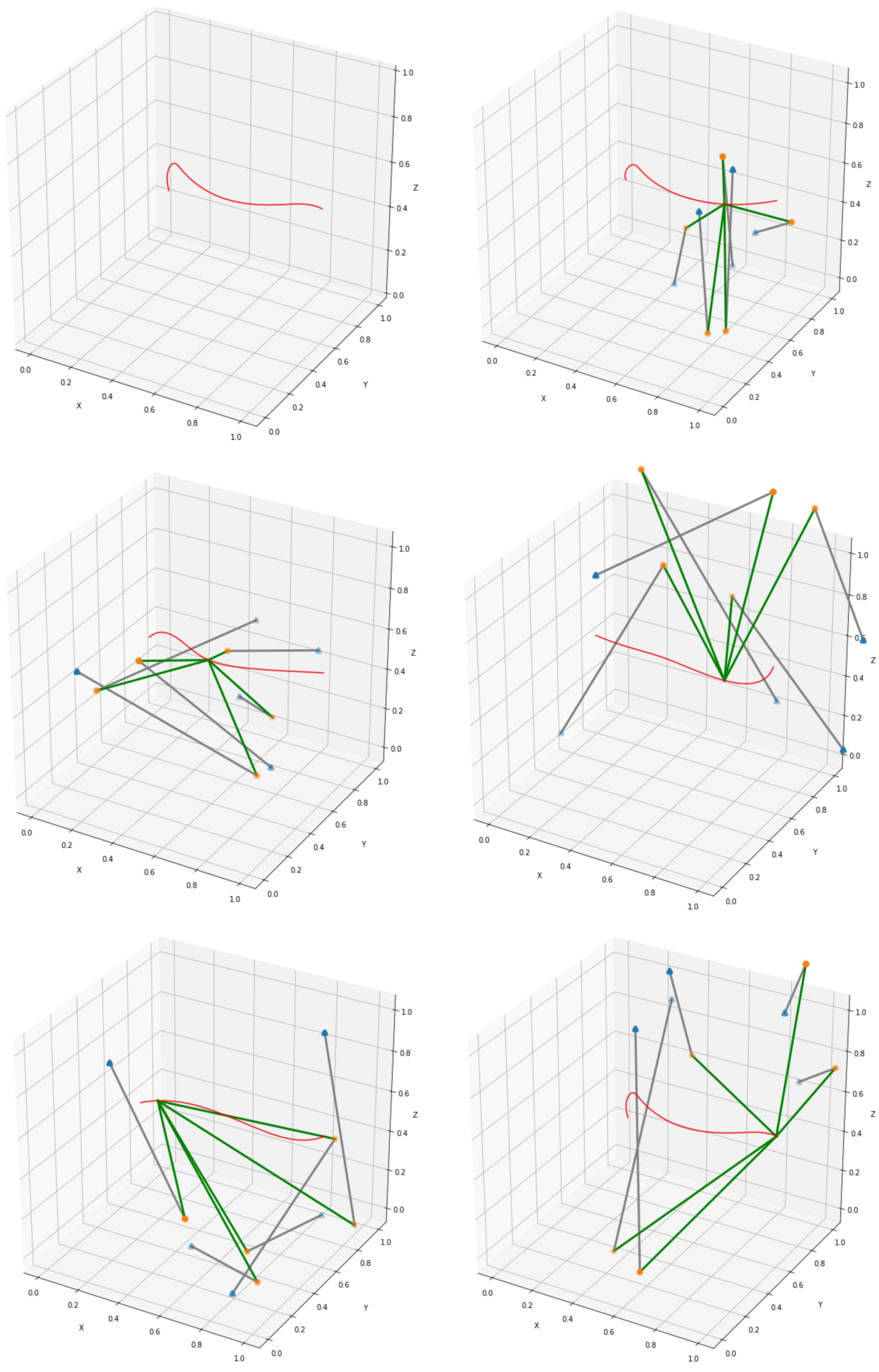


Figure 6.13: Example 1

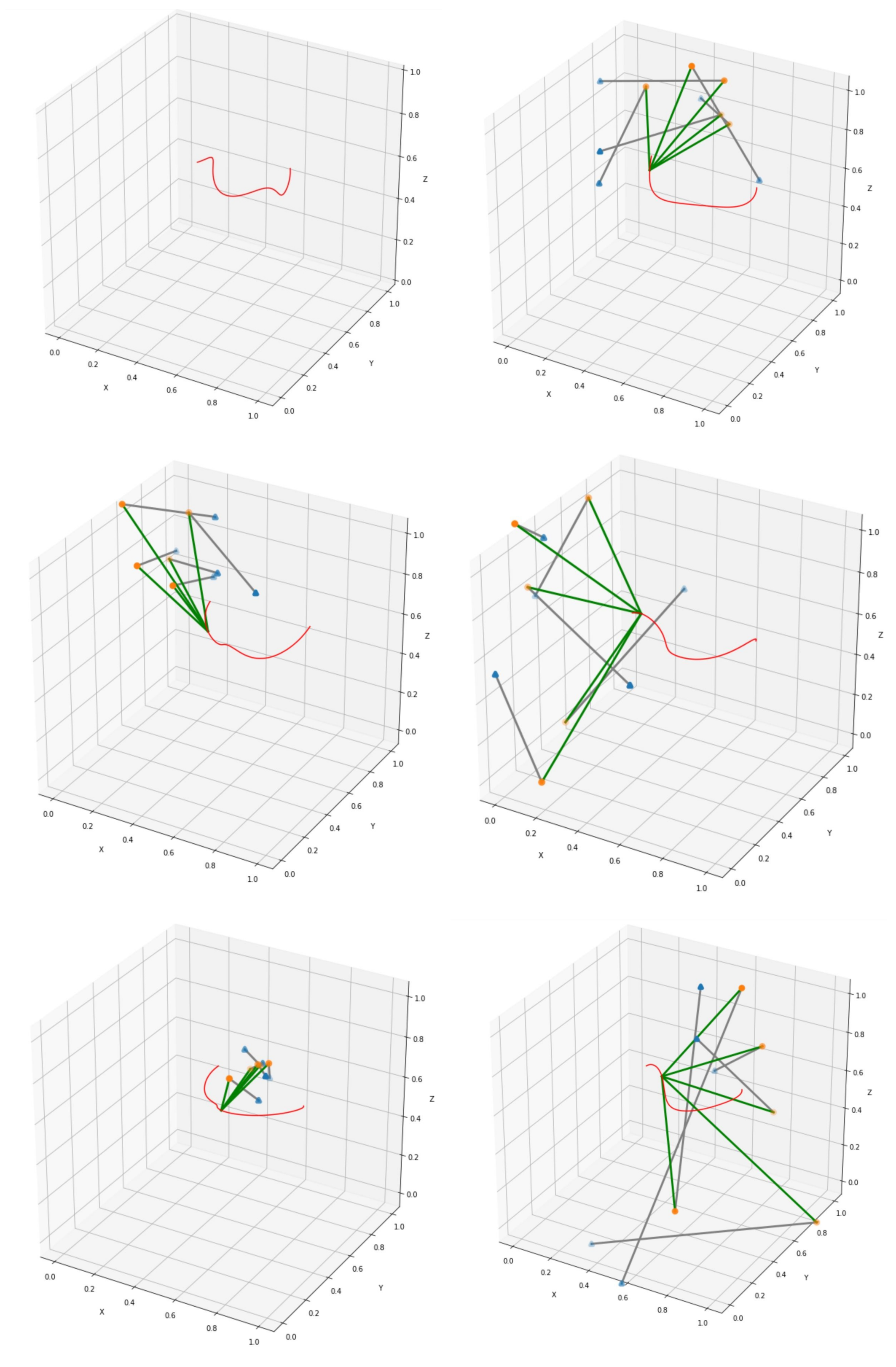


Figure 6.14: Example 2

6.6 Conclusion

Thus, in this chapter, 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.

Chapter 7

A Machine Learning Approach to Solving the Alt-Burmester Problem for Synthesis of Defect-free Spatial Mechanisms

7.1 Introduction

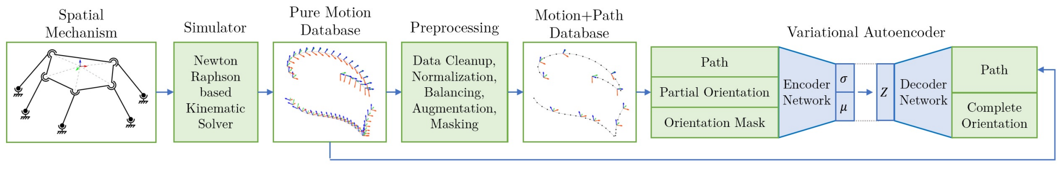
Kinematic synthesis of mechanisms has been conventionally classified and studied as path, motion and function generation problems [1, 65]. Path synthesis problems considers path-point coordinates (x_i, y_i, z_i) as input constraints while motion synthesis problems involves pose constraints $(x_i, y_i, z_i, \alpha_i, \beta_i, \gamma_i)$ where (x_i, y_i, z_i) are the location coordinates of moving frame attached to the coupler link and $(\alpha_i, \beta_i, \gamma_i)$ represent the moving frame orientations expressed as Euler angles. The function synthesis problem aims to generate a mechanism to satisfy input-out angle pairs (θ_i, ζ_i)

Extensive research has been done to solve the path generation problem for planer, spherical, and spatial mechanisms. Analytical methods for synthesis include algebraic methods [14, 92, 93, 94, 59], complex number methods [95] and displacement matrix methods [1]. Optimization-based techniques attempt to minimize an objective function and find mechanisms, which best approximate a curve [2, 96, 4, 6, 61, 74]. Atlas-based approaches explore the use of curve invariants like Fourier descriptors to intelligently form and search a database of coupler curves [27, 97]. Neural network based approaches have also been proposed for planar path synthesis [8, 112]. Chiang presents an exhaustive review of the kinematics of spherical mechanisms [76]. Atlas-based methods exist for path generation of spherical mechanism [98, 99]. Premkumar et al. have proposed an optimization based solution for the synthesis of the RRSC and RRSS spatial mechanisms [100, 101]. Ananthasuresh and Kramer solve the synthesis of the RSCR spatial mechanism using the Generalized Reduced Gradient method of optimization [102]. Jiménez et al. outline a generalized constraint based optimization technique [103]. Sun et al. use an atlas-based approach which uses the Fourier series to compare curves and synthesize RCCC spatial mechanism [104].

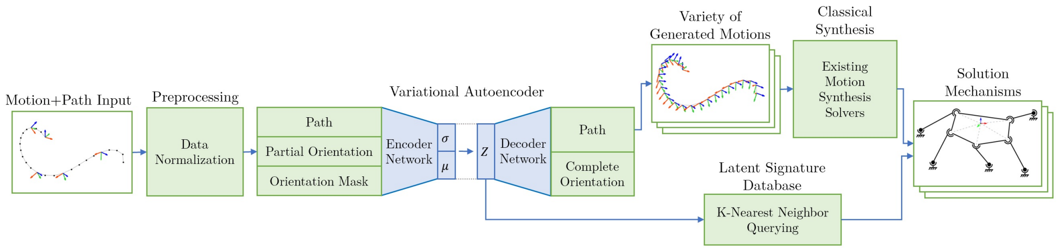
Similarly, the motion synthesis of planer, spherical, and spatial mechanisms is also a well-researched problem. Kinematic mapping and algebraic fitting based algorithms have been used for type and dimensional synthesis of planar mechanisms [15, 66, 113, 22, 21]. Generalizations of these approaches using analytical, homotopy or optimization based algorithms have been proposed for both spherical mechanisms [77, 78, 79, 80, 81, 60] and spatial mechanisms [82, 64, 83, 84].

Unfortunately, the theoretical demarcation between path and motion synthesis does not play well with real-world problems. Many practical problems require a designer to satisfy a mixture of both path and pose constraints. Using path synthesis or motion synthesis algorithms in these situations lead to a sub-optimal solution which cannot satisfy all the product design specifications.

Tong et al. [13] solved the Alt-Burmester problems named after Alt's [14] and Burmester's [15] for planar four-bar mechanisms. Brake et al. [16] proposed a homotopy based approach to solve this problem, also for planar four-bar mechanisms. Zimmerman [17] presented a geometric approach to find four-bar solutions for the mixed synthesis problem. In our previous work [62], we proposed a motion synthesis approach to solve Alt-Burmester problems



(a) Training Pipeline



(b) Inference Pipeline

Figure 7.1: Algorithm Schematic

using Fourier descriptor relationships. However, this approach requires that we painstakingly discover the connection between the properties of the path and the motion, which may be attempted only for simpler mechanisms, such as planar four-bar. Therefore, the literature available on Alt-Burmester problems is fairly recent and restricted in scope to studying planar mechanisms. In this chapter, we propose a machine learning based approach to solve the Alt-Burmester problems for spatial mechanisms. Although the chapter focuses on the synthesis of 5-SS mechanisms, the approach is general enough to handle any spatial mechanism.

A 5-SS platform linkage consists of a rigid floating body supported by five legs, each with a spherical joint on both ends. A kinematic view of a 5-SS mechanism is shown in Fig. 7.2. The motion of coupler point for a sample 5-SS mechanism has been displayed in Fig. 7.3.

As can be observed in Fig. 7.3, spatial 5-SS mechanism coupler curves tend to have multiple branches and circuits. We can see two circuits and each circuit has many branches separated by singularity points. Chase and

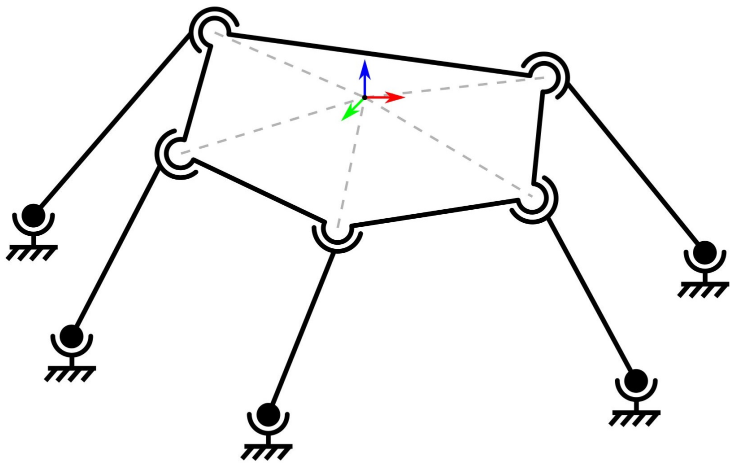


Figure 7.2: Kinematic diagram for a 5-SS mechanism

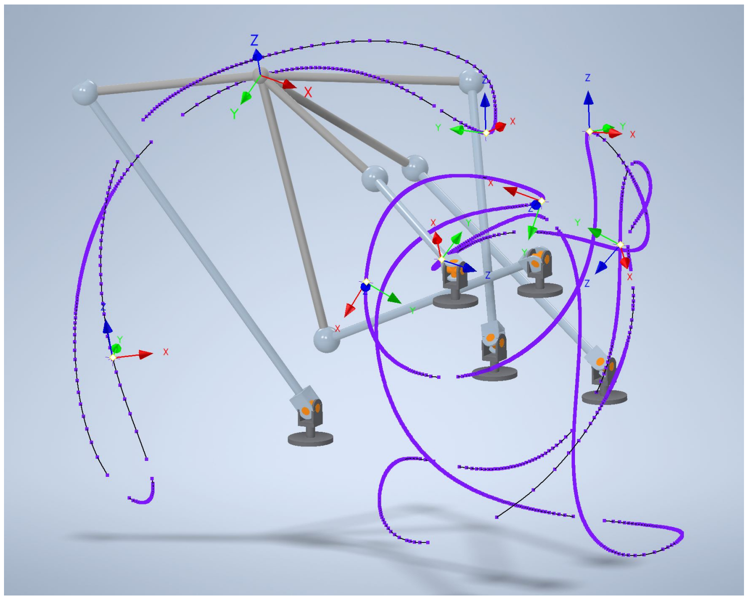


Figure 7.3: Motion of coupler point of a sample 5-SS mechanism

Mirth discuss in great detail the challenges faced in synthesizing practical one degree-of-freedom mechanisms due to circuit and branch defects[105]. Roth and Freudenstein have discussed the occurrence of defects in mechanism synthesis for path generation using numerical methods [106]. Wampler et al. show that there exist many defective mechanisms for the nine-point path synthesis problem [107]. These defects tend to be more prominent in spatial mechanisms when compared to planar mechanisms. Thus, to synthesize practical spatial mechanisms, we consider the synthesis of defect-free mechanisms.

The field of computer vision has spearheaded the use of machine learning technologies to solve challenging real-life problems [114, 115]. One of these problems has been Image inpainting which aims to restore damaged paintings and photographs [116]. Use of generative models like Variational Autoencoders (VAEs) [117] and Generative Adversarial Networks (GANs) [118] have proved to be extremely successful in adding missing information to corrupted images [119, 120]. We use these data augmentation techniques to reformulate the Alt-Burmester problem into a motion synthesis problem. A VAE is trained to take the path and pose data as input, add missing orientations, and return pure motion constraints as output.

In this chapter, we use semi-supervised machine learning algorithms to synthesize mechanisms. First, the coupler motion data is generated using Newton-Raphson based kinematic solver. Then this data is normalized, pruned, and augmented using curvature and torsion of path-curves and quaternion based orientation data. After that, we use a VAE to learn the underlying relationship between path and orientation data. Once the training phase is completed, the VAE is utilized to reformulate the Alt-Burmester problem into a motion synthesis problem by augmenting missing orientation data. A variety of plausible motion trajectory signatures, that fall in the family of defect-free 5SS coupler motions, can be generated. These latent signatures are finally looked up in a hierarchical database, created using the K-means clustering algorithm, to find solution mechanisms. Alternatively, classical motion synthesis algorithms can also be used at this stage to generate mechanisms. The algorithm proposed in this chapter has been visualized in Fig. 7.1.

Rest of the chapter is organized as follows: Section 2 presents the numerical approach to generate 5-SS coupler motions; Section 3 discusses the methodology devised to make data more conducive to learning; Section 4 uses semi-supervised machine learning tools to calculate multiple solution mechanisms and finally, Section 5 shows examples solving the spatial Alt-Burmester problem.

7.2 Data Generation

For a machine learning based approach to work, a large amount of data is required. In this chapter, this data would include 5-SS mechanisms with their coupler motion trajectory. This is achieved by creating a Newton-Raphson method based solver which uses the general constraint equations proposed in our previous work [121] given in Eq. (7.1).

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

$$C_{SS} : 2a_1c_1 + 2a_2c_2 + 2a_3c_3 + a_0c_4 = a_4\left(\frac{c_1^2 + c_2^2 + c_3^2}{c_4}\right), \tag{7.1}$$

where a_0 is given as

$$a_0 = a_4r^2 - \frac{a_1^2 + a_2^2 + a_3^2}{a_4}. \tag{7.2}$$

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 [64]. The length of the actuator imposes an additional constraint on the motion and can be defined using Eq (7.1) 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 a circuit defect occurs.

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

$$\mathbf{\Phi}(\mathbf{q}) = 0 \tag{7.3}$$

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

$$\mathbf{q}_{i+1} = \mathbf{q}_i - [\boldsymbol{J}^{-1}(\mathbf{q}_i)]\mathbf{\Phi}(\mathbf{q}_i) \tag{7.4}$$

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

$$[\boldsymbol{J}(\mathbf{q})] = \begin{bmatrix} \frac{\partial\phi_1}{\partial q_1} & \frac{\partial\phi_1}{\partial q_2} & \cdots & \frac{\partial\phi_1}{\partial q_{18}} \\ \frac{\partial\phi_2}{\partial q_1} & \frac{\partial\phi_2}{\partial q_2} & \cdots & \frac{\partial\phi_2}{\partial q_{18}} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial\phi_{18}}{\partial q_1} & \frac{\partial\phi_{18}}{\partial q_2} & \cdots & \frac{\partial\phi_{18}}{\partial q_{18}} \end{bmatrix}. \tag{7.5}$$

To calculate the Jacobian matrix, relations describing the first order partial derivatives of constraint equations are required. For an SS dyad described in Eq (7.1), the first order partial derivatives can be given as follows

$$\frac{\partial C_{SS}}{\partial a_1} = 2(a_1 - c_1) \tag{7.6}$$

$$\frac{\partial C_{SS}}{\partial a_2} = 2(a_2 - c_2) \tag{7.7}$$

$$\frac{\partial C_{SS}}{\partial a_3} = 2(a_3 - c_3) \tag{7.8}$$

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 moving pivot coordinates, we can simulate a 5-SS mechanism and extract the path traced by the six points $P = (P_x, P_y, P_z, 1)$ on the coupler. 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.

To find the coupler orientation, the displacement matrix D_f can be calculated using the six points on coupler as

$$D_f = P_f P_i^+ = P_f P_i^*(P_i P_i^*)^{-1} \tag{7.9}$$

where $P_i = [P_{i,1}^T, P_{i,2}^T, \cdots, P_{i,6}^T]$ is the matrix representing initial coupler coordinates and $P_f = [P_1^T, P_2^T, \cdots, P_6^T]$ represents the coupler coordinates after

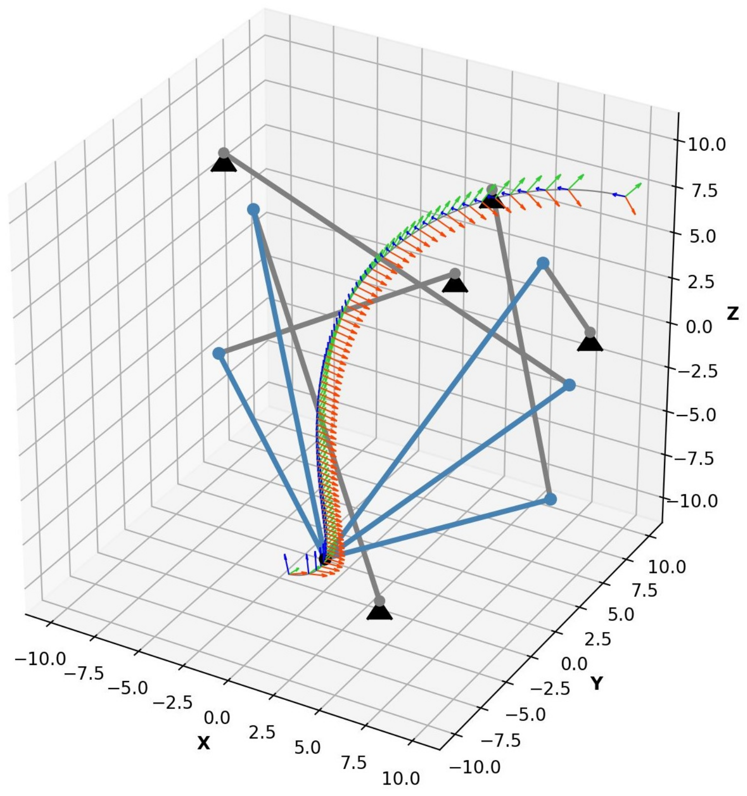


Figure 7.4: A 5-SS mechanism simulation where black triangles are the fixed pivots, gray lines are the SS dyads, light blue lines are the floating coupler, and the red, green blue arrows represent the moving frame attached to the coupler.

displacement. The 3×3 rotation matrix R is isolated from the 4×4 displacement matrix D_f and stored in unit quaternion form. Thus, a spatial coupler motion is represented by a 7-D curve consisting of three location coordinates and four quaternion coordinates.

For this chapter, we generated a data set of 10,000 defect-free coupler motions using arbitrarily selected 5-SS mechanisms. Fig. 7.4 shows one of the simulated mechanisms. This database represents a family of motions 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.

7.3 Data Preprocessing

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

7.3.1 Data Cleanup

A spatial coupler motion is defined as an array of n 7-D data points. In the data set of 10,000 motion curves, we observe that n ranges from 1 to 1,562 as can be seen in Fig. 7.5. 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 20 data points are removed resulting in a data set of 9,472 curves.

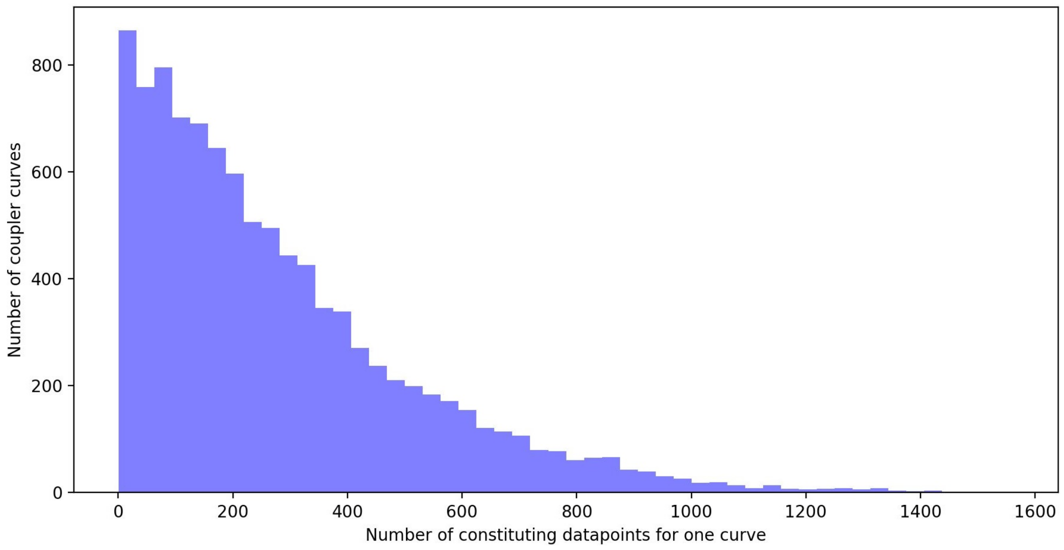


Figure 7.5: Histogram showing number of data points in each motion curve included in database

When the solver is simulating a 5SS-mechanism, as outlined in Section 7.2, it may jump from one branch to another due to inherent limitations of numerical methods. These incorrect motions are characterized by a C^1 discontinuity at the point where the branch jump occurs. Example of such curves have been shown in Fig. 7.6.

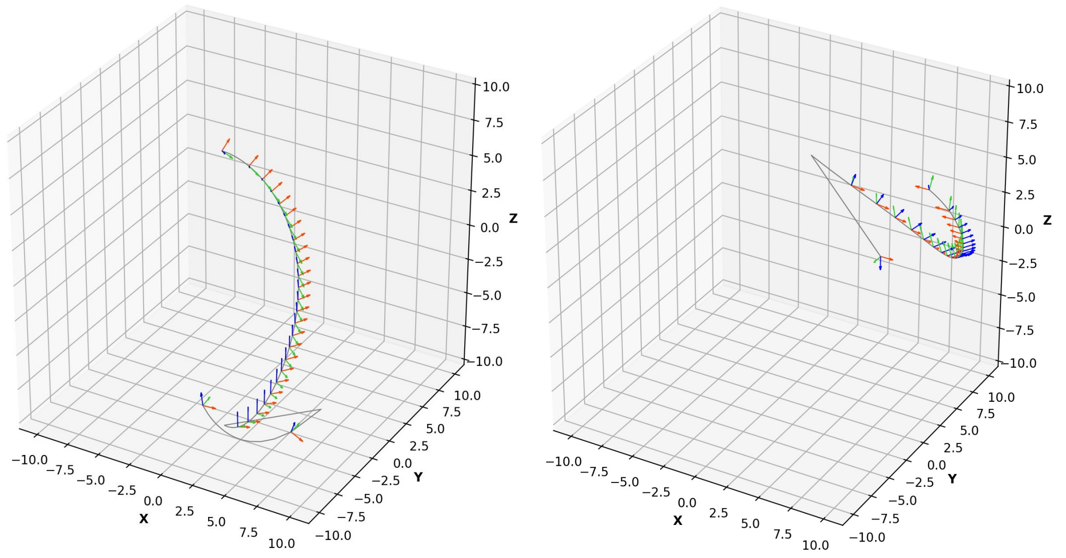


Figure 7.6: Coupler curves with C_1 discontinuity caused due to a branch jump.

Thus, to isolate the incorrect curves, the first order differential is calculated and spikes in its magnitude are observed. The Z-score metric, also called the standard score is used to characterize these spikes. A Z-score indicates how many standard deviations an element is from the mean and is given as

$$z = \frac{X - \mu}{\sigma} \tag{7.10}$$

where μ is the mean and σ is the standard deviation. In our study, an outlier is defined as any curve having $Z_{max} > 15$. Filtering out the outliers results in a clean database containing 8,688 coupler curves.

7.3.2 Data Normalization

The remaining coupler motions are fitted with an interpolation curve. Fourth-order B-spline [111] interpolation is used for path data while spherical linear quaternion interpolation (Slerp) [122] is used for orientation data. 25 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. 7.7.

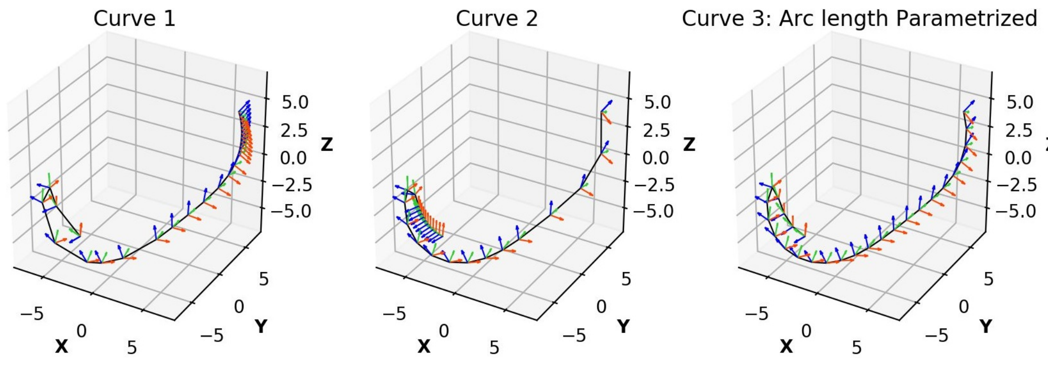


Figure 7.7: Curve 1 and Curve 2 represents the same geometric curve with different time parametrization. They share the same unique arc length parametrization as shown in Curve 3

Creating a curve representation which is translation, rotation, and scaling invariant is desirable when comparing curves. For the path data, 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 path data 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 path data is scaled to unit arc-length. The orientation data is normalized by rotating the moving frame such that it is aligned to the fixed frame axis at the start of motion. The effect of normalization on a sample coupler motion has been demonstrated in Fig. 7.8.

7.3.3 Data Balancing

The database in its present form is unbalanced i.e. it has more samples of coupler curves 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,

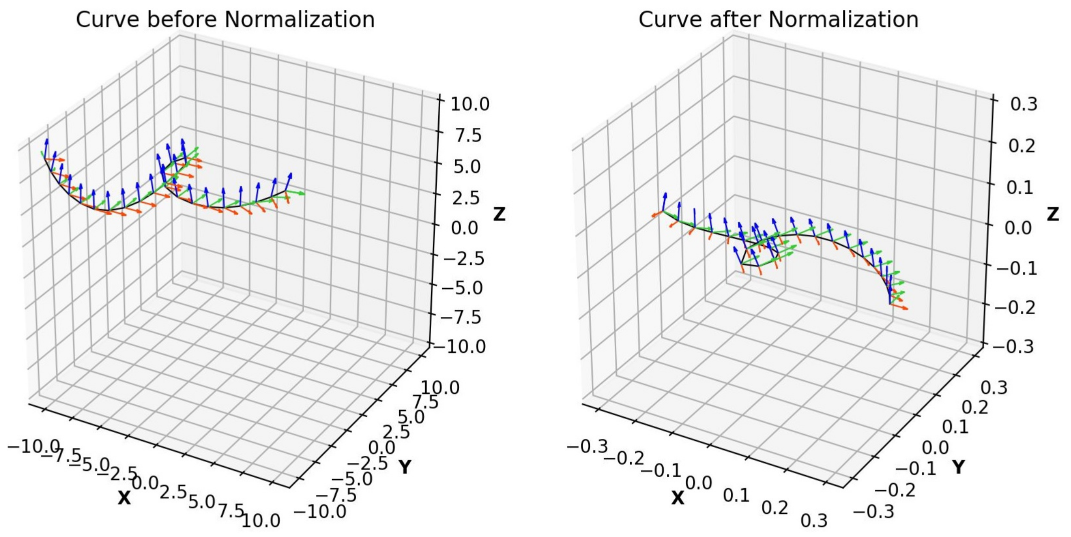


Figure 7.8: Before and after normalizing a motion curve

a limited number of diverse motions 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 [123]. Thus, a good metric to compare the similarity of two curves are these Curvature–Torsion Descriptors. 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

$$\kappa = \frac{||\mathbf{r}'(t) \times \mathbf{r}''(t)||}{||\mathbf{r}'(t)||^3}, \tag{7.11}$$

$$\tau = \frac{(\mathbf{r}'(t) \times \mathbf{r}''(t)) \cdot \mathbf{r}'''(t)}{||\mathbf{r}'(t) \times \mathbf{r}''(t)||^2}, \tag{7.12}$$

where $\mathbf{r}(t)$ is the curve. The curvature is always positive while the torsion can be negative.

For the path data, we calculate the curvature and torsion while we use the quaternion data directly to compare curves. We define the similarity score (δ) as a weighted sum of P2 norms of difference between path curvature (κ), path torsion (τ), and orientation quaternion (Q) of two motions which is given as

$$\delta = \frac{||\kappa_2 - \kappa_1||_2 + w_1||\tau_2 - \tau_1||_2 + w_2||Q_2 - Q_1||_2}{n} \tag{7.13}$$

where w_1 and w_2 are 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 $w_1 = .1$ in this chapter. Weight $w_2 = 1$ so that equal importance is given to both path data and orientation data. We select the similarity metric threshold such that if $\delta < 0.25$, the two curves are similar and one of them is dropped from the database. It can be observed in Fig. 7.9 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. 7.10. Under-sampling similar curves lead to a balanced data set containing 5,222 coupler paths.

7.3.4 Data Augmentation

In kinematics, it is known that if a curve is a valid coupler motion, its mirrored curve is also a valid motion. For the machine learning algorithm to gain this domain knowledge, coupler motions mirrored across xy,yz, and zx planes are added to the database. Thus, this step is to encourage the model to be invariant to mirror operations. Since reflecting the moving coordinate frame converts it from a right-handed system to a left-handed system, we attach a new right-handed frame to the coupler and recalculate the orientation data over the motion. After this step, our database contains 20,888 motion curves.

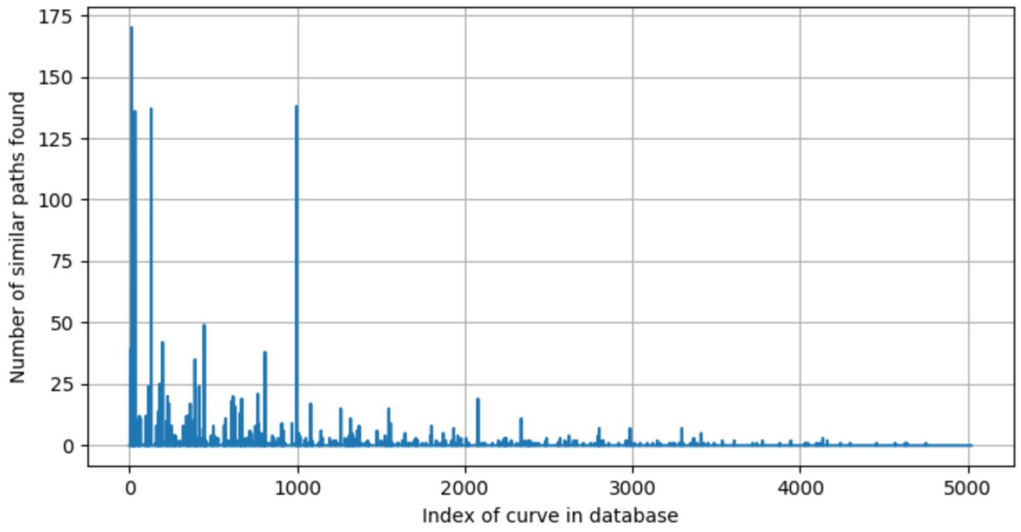


Figure 7.9: Bar graph showing the number of similar curves found for each curve.

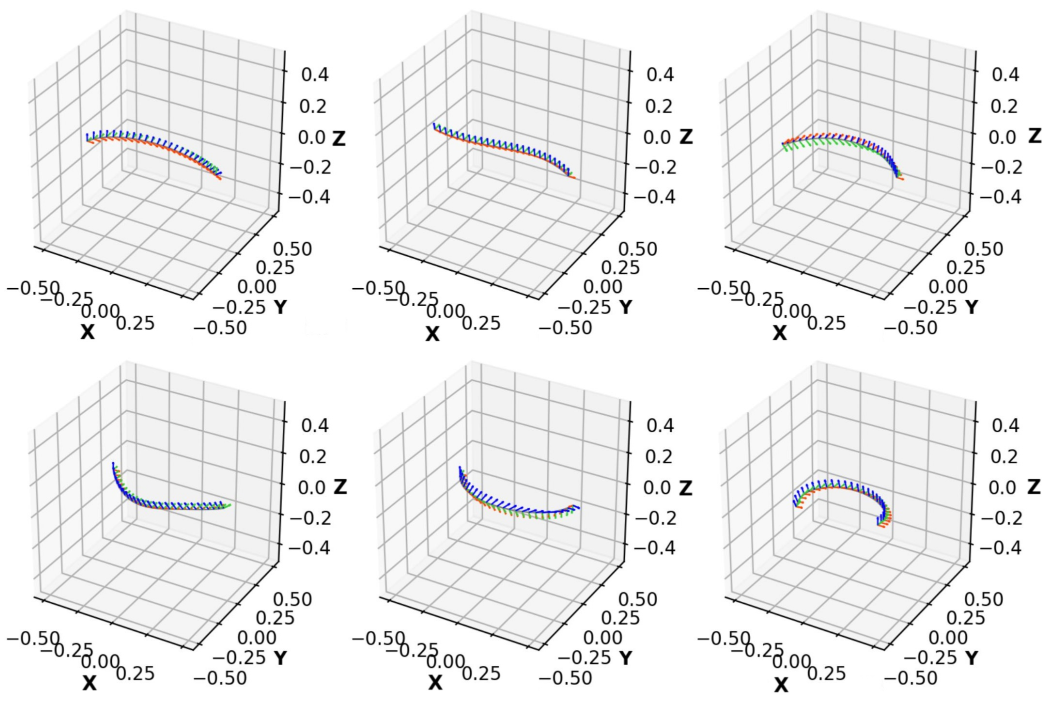


Figure 7.10: Most common variety of motions in the database

7.3.5 Data Masking

For the machine learning model to learn the underlying kinematic relationship between path-points and orientations, we synthetically mask the coupler motion database to create a path+pose constraint database. To create a masked motion from a n-point coupler motion, m-points $(0 \leq m \leq n)$ are randomly selecting and the orientations of these m points are set to zero. Also, a 1D binary mask is created representing the locations where the information is missing. An example of unmasked and masked motion curves are shown in Fig. 7.11. The mask value is 0 at m-points and 1 at the remaining points. This operation increases the database size to 417,760 masked motion curves.

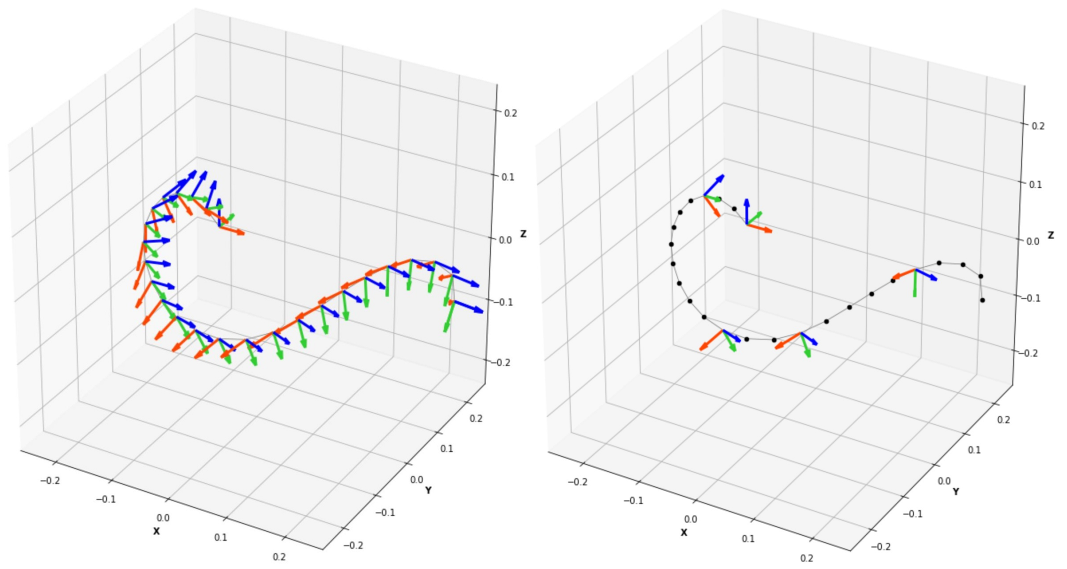


Figure 7.11: An unmasked and masked 5SS coupler motion curve

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. The magnitude of the added noise is up to 3% the maximum magnitude of motion curve coordinates.

This concludes the data preprocessing pipeline which generates two databases that are used to train the machine learning models. First is the database containing 7-D pure coupler motion curves defined by three location coordinates and four orientation coordinates. Second is the database containing 8-D masked coupler motion curves defined by three location coordinates, four orientation coordinates, and one mask coordinate.

7.4 Machine Learning based Synthesis

Now that the database has been processed it can be used to train a machine learning model.

7.4.1 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 and motion constraints. Also, it should provide a low dimensional signature to the coupler curve 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 partial motion curves (X_{mix}) and predicts complete motion curves (X_{mot}) by approximating the underlying distribution of observed data. As can be seen in Fig. 7.12, 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 unmasked motion 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.

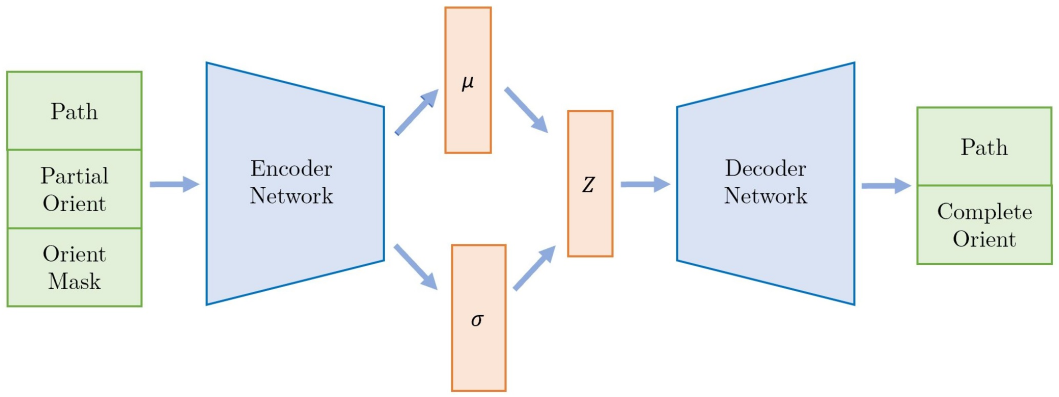


Figure 7.12: Architecture of a Variational Autoencoder

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

$$Loss = RL + KL \tag{7.14}$$

$$RL = ||\hat{X}_{mot} - X_{mot}||_2 \tag{7.15}$$

$$KL = \sum_{i=1}^{k} \sigma_i^{2\text{‘}} + \mu_i^2 - log(\sigma_i) - 1 \tag{7.16}$$

where the latent space is k dimensional and the reconstruction loss is the Euclidean norm, represented as $||.||_2$ operator, of the difference between generated motion ($\hat{X}_{mot}$) and true motion (X_{mot}).

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 7.1. Each VAE is trained for 1000 epochs with a batch size of 256 using Adam (adaptive moment estimation) optimizer.

Table 7.1: 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)	32.2734	32.4453
VAE-FC-H1-Z30	(100)	30	(100)	32.2415	32.2879
VAE-FC-H1-Z60	(100)	60	(100)	32.0597	32.3151
VAE-FC-H2-Z15	(150,75)	15	(75,150)	28.1138	28.8488
VAE-FC-H2-Z30	(150,75)	30	(75,150)	28.6563	29.2358
VAE-FC-H2-Z60	(150,75)	60	(75,150)	28.1563	28.8108
VAE-FC-H3-Z15	(200,100,60)	15	(60,100,200)	25.2833	25.8237
VAE-FC-H3-Z30	(200,100,60)	30	(60,100,200)	25.5534	26.0324
VAE-FC-H3-Z60	(200,100,60)	60	(60,100,200)	26.2158	26.6811
VAE-FC-H4-Z15	(200,150,100,60)	15	(60,100,150,200)	25.5023	25.8424
VAE-FC-H4-Z30	(200,150,100,60)	30	(60,100,150,200)	26.3350	26.6101
VAE-FC-H4-Z60	(200,150,100,60)	60	(60,100,150,200)	28.2903	28.3822

We notice that VAE-FC-H3-Z15 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 15 nodes. The training curves of VAE-FC-H3-Z15 can be seen in Fig. 7.13.

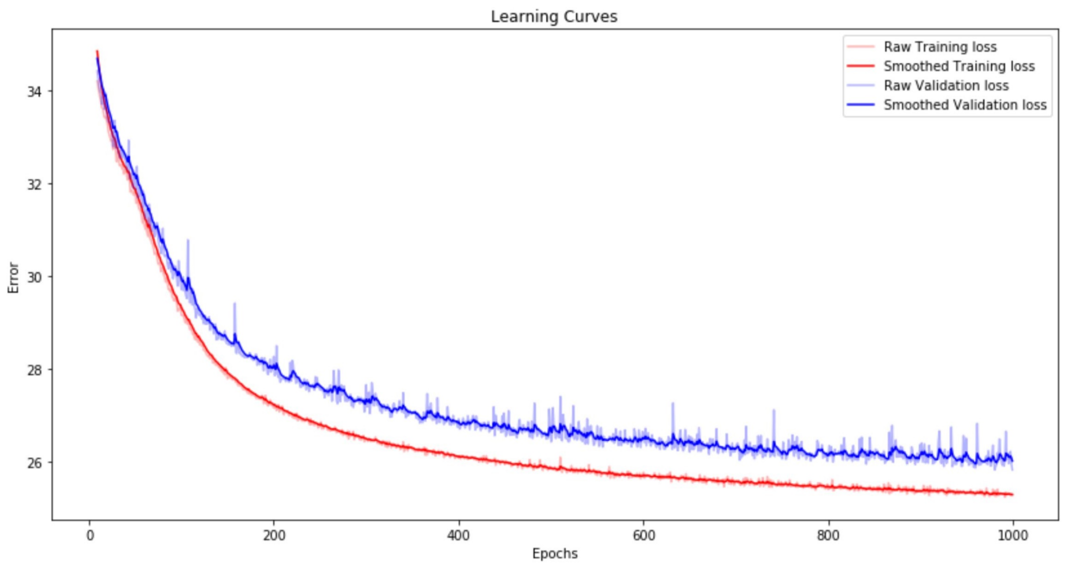
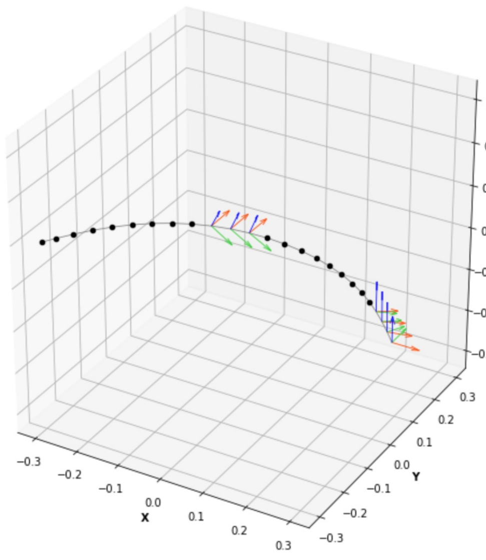
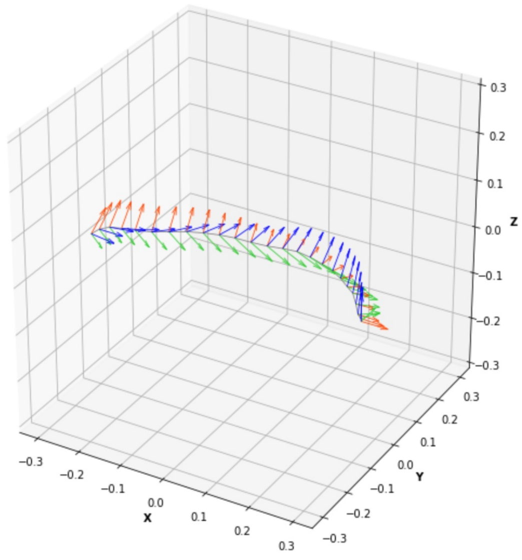


Figure 7.13: Training losses for the fully connected VAE

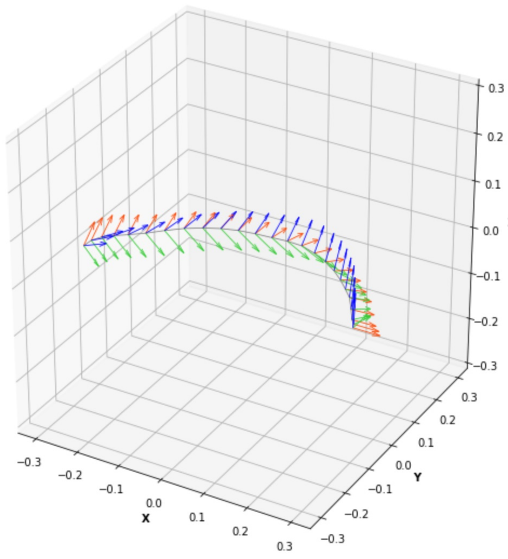
Some sample outputs generated using VAE-FC-H3-Z15 are shown in Fig. 7.14 where an input constraints (top-left) generates three motion trajectories sampled from the underlying distribution. Notice how the VAE can generate variations in both path and orientation data using its understanding of family of 5-SS mechanism motions.



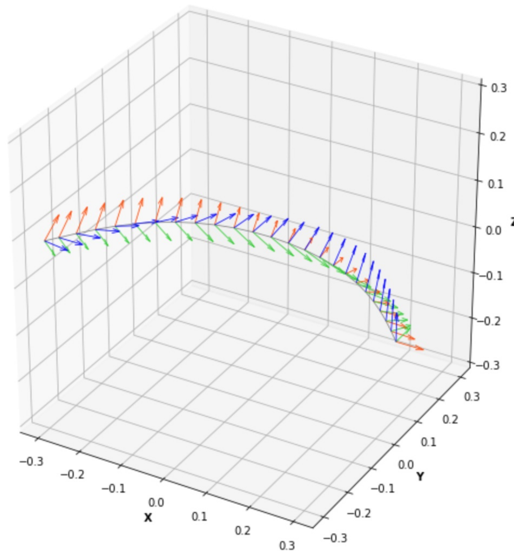
(a) Input Masked Motion



(b) Generated Motion 1



(c) Generated Motion 2



(d) Generated Motion 3

Figure 7.14: Motion Trajectories generated using VAE for a Alt-Burmester input constraint

7.4.2 Using Classical Motion Generation

Once the model is trained, it is capable of augmenting any Alt-Burmester problem into a motion synthesis problem. Existing kinematic mapping based algebraic fitting algorithms can now be used to generate 5-SS mechanisms [82, 64]. However, for a general spatial mechanism, a motion synthesis approach might not exist in literature. In that case, we use a hierarchical database based approach to find solution mechanisms efficiently.

7.4.3 Creating a Hierarchical Database

One of the most powerful features of the VAE is its ability to provide a compact representation of each coupler motion using its latent vector. We use this compact signature for fast and efficient database search and clustering algorithms. Thus, once the training is completed, the encoder module is used to generate latent vector signatures of each masked motion in the database denoted by the μ vector. Then, these signatures are clustered into 500 groups using the K-Means Clustering algorithm. The distance metric used is the Euclidean distance. As a result, we get 500 cluster centers subdividing the original dataset of 400k+ coupler constraint curves.

7.4.4 Mechanism synthesis for User Inputted Trajectory

When the user inputs a curve consisting of path-points and poses, 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.

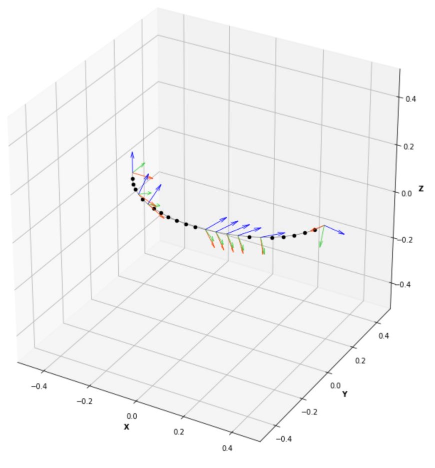
7.5 Examples of Mechanism Synthesis for Alt-Burmester Problem

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 motion available and provide it as a solution.

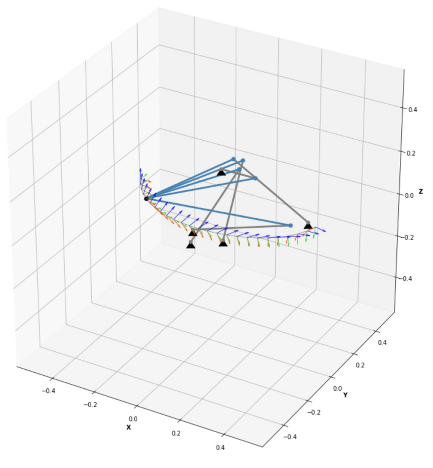
The input trajectory is shown in the first plot in Fig. 7.15 and Fig. 7.16. The other plots show a prospective 5-SS solution that closely matches the target constraints. More mechanisms can be generated by sampling additional latent vectors from the VAE.

7.6 Conclusion

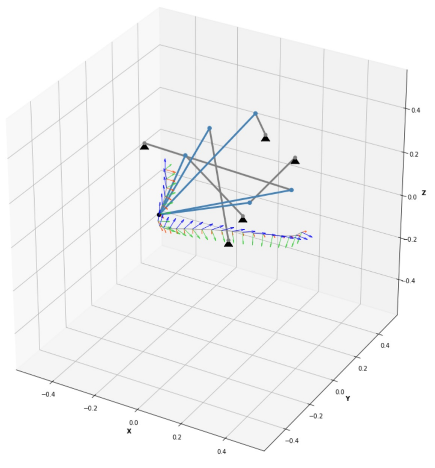
Thus, in this chapter, we have discussed a complete pipeline including mechanism simulation, data preprocessing, and machine learning model creation for the defect-free synthesis of spatial 5-SS mechanism. To generate coupler motion data, we use a geometric constraint based numerical approach which uses the Newton-Raphson method. Then, the data is pre-processed and masked for the purpose of robust learning. Finally, semi-supervised machine learning techniques of VAE and K-mean clustering are used to efficiently find solution mechanisms. The algorithm is general enough to solve the Alt-Burmester problem for any spatial mechanism and generate multiple solutions.



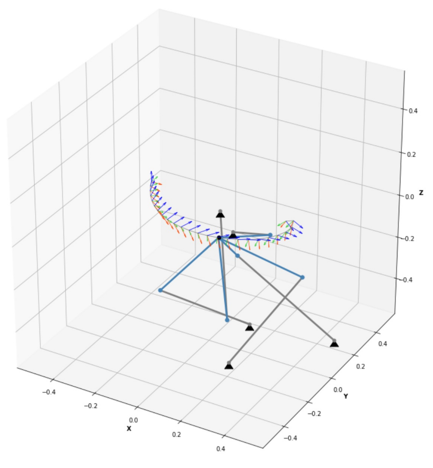
(a) Input Alt-Burmester Problem



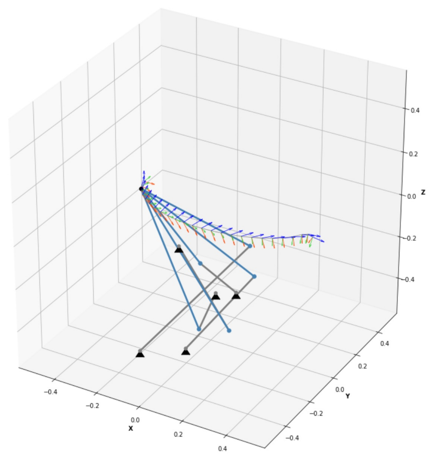
(b) Synthesized Mechanism 1



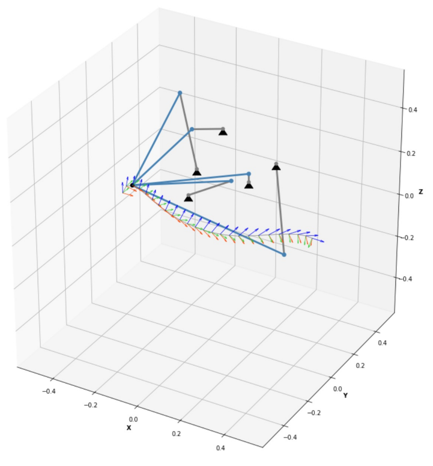
(c) Synthesized Mechanism 2



(d) Synthesized Mechanism 3

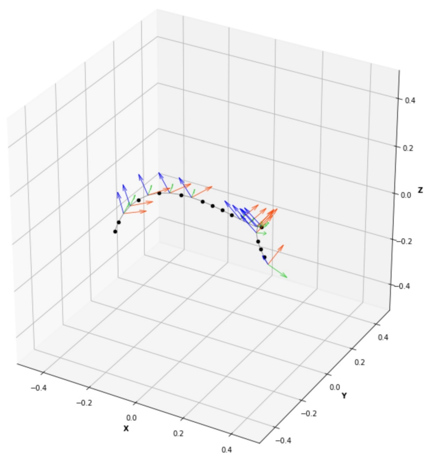


(e) Synthesized Mechanism 4

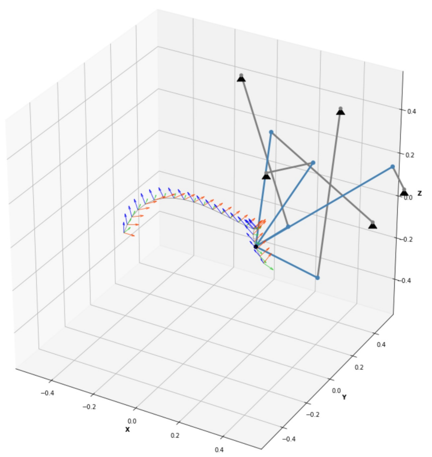


(f) Synthesized Mechanism 5

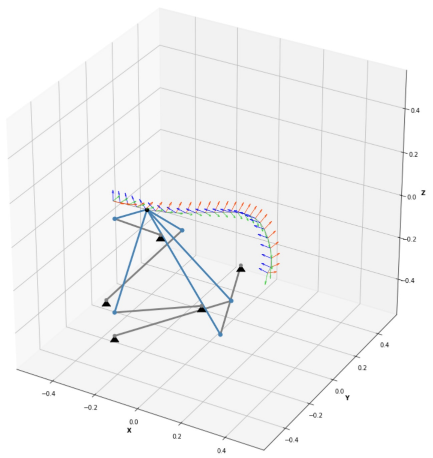
Figure 7.15: Example 1



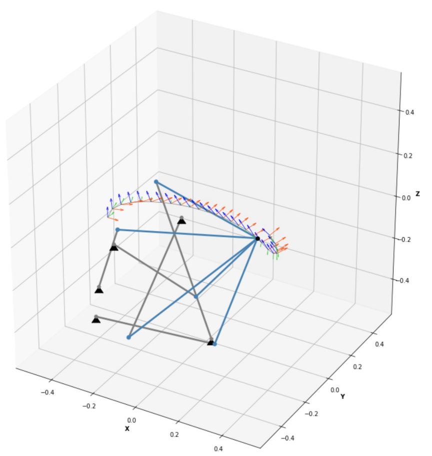
(a) Input Alt-Burmester Problem



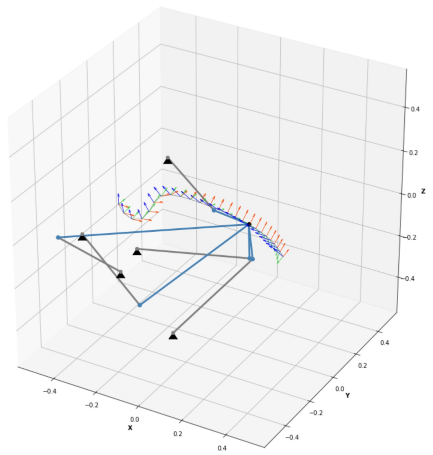
(b) Synthesized Mechanism 1



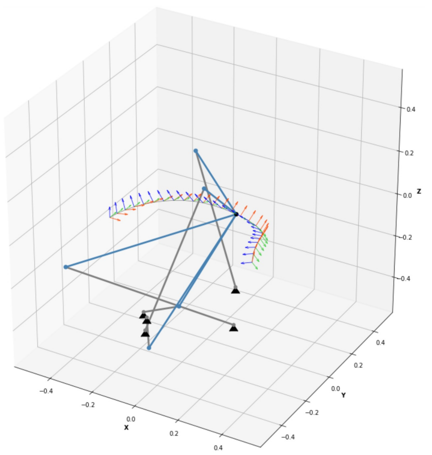
(c) Synthesized Mechanism 2



(d) Synthesized Mechanism 3



(e) Synthesized Mechanism 4



(f) Synthesized Mechanism 5

Figure 7.16: Example 2

Chapter 8

Conclusion

This dissertation claims the following original contributions

- A Fourier based mixed synthesis approach which unifies path and motion synthesis for planar mechanisms.
- An non-uniform parameterization scheme to optimize Fourier parameters for planar path synthesis.
- A point-line-plane representation for unified real-time simulation of planar, spherical, and spatial mechanisms.
- A kinematic synthesis based approach for the unified motion synthesis of planar, spherical, and spatial mechanisms.
- A machine learning based approach to synthesize spatial mechanisms for the Alt-Burmester problem and Path Synthesis problem.

Future work could include an extension of the mixed synthesis framework to include function generation. This would unify the three conventional methodologies into a unified framework. Path and motion synthesis for more complex mechanisms like the six-bar or the eight-bar can also be explored. Also, the use of machine learning to make mechanism synthesis less computationally-intensive and more real-time is worth investigating.

Bibliography

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

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

[3] Freudenstein, F., 1959, "Harmonic Analysis of Crank-and-Rocker Mechanisms With Application", ASME J. Appl. Mech, **26**, p. 673–675.

[4] 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.

[5] Mcgarva, J. and Mullineux, G., 1993, "Harmonic Representation of Closed Curves", Applied Mathematical Modelling, **17(4)**, pp. 213–218.

[6] 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.

[7] Wu, J., Ge, Q. J., Gao, F., and Guo, W. Z., 2010, "On the extension of a fourier descriptor based method for four-bar linkage synthesis for generation of open and closed paths", 2010 ASME Mechanisms and Robotics Conference, dETC2010–29028.

[8] Vasiliu, A. and Yannou, B., 2001, "Dimensional synthesis of planar mechanisms using neural networks: application to path generator linkages", Mechanism and Machine Theory, **36(2)**, pp. 299–310.

[9] Li, X., Zhong, X., and Ge, Q., 2015, "Parametrization-Independent Non-Uniform Fourier Approach to Path Synthesis of Four-Bar Mechanism", Proceedings of the 14th IFToMM World Congress, pp. 440–448.

[10] Li, X. Y. and Chen, P., 2017, "A Parametrization-Invariant Fourier Approach to Planar Linkage Synthesis for Path Generation", Mathematical Problems in Engineering, **2017**.

[11] Li, X., Wu, J., and Ge, Q. J., 2016, "A Fourier Descriptor-Based Approach to Design Space Decomposition for Planar Motion Approximation", ASME Journal of Mechanisms and Robotics, **8(6)**, pp. 064501–064501–5, 10.1115/1.4033528.

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

[13] Tong, Y., Myszka, D. H., and Murray, A. P., 2013, "Four-Bar Linkage Synthesis for a Combination of Motion and Path-Point Generation", 10.1115/DETC2013-12969.

[14] Alt, H., 1923, "Uber die Erzeugung gegebener ebener Kurven mit Hilfe des Gelenkvierecks", ZAMM, **3(1)**, pp. 13–19.

[15] Burmester, L., 1886, Lehrbuch der Kinematik, Verlag Von Arthur Felix, Leipzig, Germany.

[16] Brake, D. A., Hauenstein, J. D., Murray, A. P., Myszka, D. H., and Wampler, C. W., 2016, "The Complete Solution of Alt–Burmester Synthesis Problems for Four-Bar Linkages", ASME Journal of Mechanisms and Robotics, **8(4)**, pp. 041018–041018–8, 10.1115/1.4033251.

[17] Zimmerman, A. R. I., 2018, "Planar Linkage Synthesis for Mixed Motion, Path, and Function Generation Using Poles and Rotation Angles", ASME Journal of Mechanisms and Robotics, **10(2)**, pp. 025004–025004–8, 10.1115/1.4039064.

[18] 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 Journal of Mechanisms and Robotics, 10.1115/1.4035899.

[19] 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)**, p. 061005, 10.1115/1.4037801.

[20] 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 Journal of Mechanisms and Robotics, **8(6)**, pp. 061003–061003–8, 10.1115/1.4033434.

[21] 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.

[22] Ge, Q. J., Zhao, P., Purwar, A., and Li, X., 2012, "A Novel Approach to Algebraic Fitting of a Pencil of Quadrics for Planar 4R Motion Synthesis", ASME Journal of Computing and Information Science in Engineering, **12**, pp. 041003–041003.

[23] Chu, J. and Cao, W.-q., 1993, "Synthesis of coupler curves of planar four-bar linkages through fast fourier transform", Chin. J. Mech. Eng, **29(5)**, pp. 117–122.

[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", Journal of Mechanisms and Robotics-Transactions of the Asme, **3(3)**.

[25] 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.

[26] Yue, C., Su, H.-J., and Ge, Q. J., 2012, "A hybrid computer-aided linkage design system for tracing open and closed planar curves", Computer-Aided Design.

[27] 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)**.

[28] Krovi, V., Ananthasuresh, G., and Kumar, V., 2002, "Kinematic and kinetostatic synthesis of planar coupled serial chain mechanisms", Journal of Mechanical Design, **124(2)**, pp. 301–312.

[29] Nie, X. and Krovi, V., 2005, "Fourier methods for kinematic synthesis of coupled serial chain mechanisms", Journal of Mechanical Design, **127(2)**, pp. 232–241.

[30] Sharma, S. and Purwar, A., "Optimal non-uniform parameterization scheme for fourier descriptor based path synthesis of four bar mechanisms", ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference,, American Society of Mechanical Engineers, volume 2018, dETC2018-85567.

[31] Day, D. and Heroux, M. A., 2001, "Solving complex valued linear systems via equivalent real formulations", SIAM Journal on Scientific Computing, **23(2)**, p. 480–498.

[32] Bottema, O. and Roth, B., 1979, Theoretical Kinematics, Dover Publication Inc., New York.

[33] McCarthy, J. M., 1990, Introduction to Theoretical Kinematics, The MIT Press, Cambridge, MA.

[34] Golub, G. and Van Loan, C., 1996, Matrix Computations, Johns Hopkins Univ Press, Baltimore, MD.

[35] Norton, R., 2011, Design of Machinery: An Introduction To The Synthesis and Analysis of Mechanisms and Machines, McGraw Hill, 5th edition.

[36] Erdman, A. G. and Sandor, G. N., 1991, Mechanism Design: Analysis and Synthesis, volume 1, Prentice-Hall, Englewood Cliffs, NJ, 2nd edition.

[37] Rector, D., "Linkage", URL http://blog.rectorsquid.com/linkage-mechanism-designer-and-simulator/.

[38] Campbell, M., "Planar Mechanism Kinematic Simulator", URL http://design.engr.oregonstate.edu/pmksintro.html.

[39] Waldron, K. and Sreenivasan, S., 1996, "A study of the solvability of the position problem for multi-circuit mechanisms by way of example of the double butterfly linkage", Journal of Mechanical design, **118(3)**, pp. 390–395.

[40] Nielsen, J. and Roth, B., 1999, "On the Kinematic Analysis of Robotic Mechanisms", The International Journal of Robotics Research, **18(12)**, pp. 1147–1160, doi:10.1177/02783649922067771, URL https://doi.org/10.1177/02783649922067771.

[41] Wampler, C. W., 1999, "Solving the kinematics of planar mechanisms", Journal of Mechanical Design, **121(3)**, pp. 387–391.

[42] Nielsen, J. and Roth, B., 1999, "Solving the input/output problem for planar mechanisms", Journal of Mechanical Design, **121(2)**, pp. 206–211.

[43] Raghavan, M. and Roth, B., 1995, "Solving polynomial systems for the kinematic analysis and synthesis of mechanisms and robot manipulators", Journal of Mechanical Design, **117(B)**, pp. 71–79.

[44] De Jalon, J. G. and Bayo, E., 2012, Kinematic and dynamic simulation of multibody systems: the real-time challenge, Springer Science & Business Media.

[45] Nikravesh, P. E., 1988, Computer-aided analysis of mechanical systems, volume 186, Prentice-hall Englewood Cliffs, NJ.

[46] Kreyszig, E., 2007, Advanced engineering mathematics, John Wiley & Sons.

[47] Hernández, A. and Petuya, V., 2004, "Position analysis of planar mechanisms with R-pairs using a geometrical–iterative method", Mechanism and machine theory, **39(2)**, pp. 133–152.

[48] Radhakrishnan, P. and Campbell, M. I., 2012, "An Automated Kinematic Analysis Tool for Computationally Synthesizing Planar Mechanisms", ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers, pp. 1553–1562.

[49] Inc., A., 2016, "Autodesk Inventor", URL http://www.autodesk.com/products/inventor/overview.

[50] Systems, D., "SolidWorks, http://www.solidworks.com/", URL http://www.solidworks.com/.

[51] MSCsoftware, 2015, "Adams/Solver Help - DOC10902, https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=DOC10902&actp=RSS", .

[52] Software, M., "Adams, http://www.mscsoftware.com/product/adams", URL http://www.mscsoftware.com/product/adams.

[53] Artas Engineering, "SAM (Synthesis and Analysis of Mechanisms)", URL http://www.artas.nl/en.

[54] Petuya, V., Macho, E., Altuzarra, O., and Pinto, C., 2011, "Educational Software Tools for the Kinematic Analysis of Mechanisms", Comp. Appl. Eng. Education, **6(4)**, pp. 261–266.

[55] Furlong, T. J., Vance, J. M., and Larochelle, P. M., 1999, "Spherical mechanism synthesis in virtual reality", ASME Journal of Mechanical Design, **121(4)**, pp. 515–520.

[56] Larochelle, P., Dooley, J., Murray, A., and McCarthy, J. M., 1993, "SPHINX: Software for Synthesizing Spherical 4R Mechanisms", Proc. of the 1993 NSF Design and Manufacturing Systems Conference, volume 1, pp. 607–611.

[57] Ruth, D. and McCarthy, J., 1997, "Sphinxpc: An implementation of four position synthesis for planar and spherical 4r linkages", ASME Design Engineering Technical Conferences.

[58] Tse, D. and Larochelle, P., 1999, "Osiris: a new generation spherical and spatial mechanism cad program", Florida Conference on Recent Advancements in Robotics.

[59] 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.

[60] Li, X., Zhao, P., Purwar, A., and Ge, Q., 2018, "A Unified Approach to Exact and Approximate Motion Synthesis of Spherical Four-Bar Linkages Via Kinematic Mapping", ASME Journal of Mechanisms and Robotics, **10(1)**, p. 011003.

[61] 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.

[62] 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 Journal of Mechanisms and Robotics, **11(1)**, p. 011016.

[63] Li, X., Ge, X., Purwar, A., and Ge, Q. J., 2015, "A Unified Algorithm for Analysis and Simulation of Planar Four-Bar Motions Defined With R- and P-Joints", ASME Journal of Mechanisms and Robotics, **7(1)**, pp. 011014–011014–7, 10.1115/1.4029295.

[64] Liao, Q. and McCarthy, J. M., 1997, "On the Seven Position Synthesis of a 5-SS Platform Linkage ", ASME Journal of Mechanical Design, **123(1)**, pp. 74–79, doi:10.1115/1.1330269, URL https://doi.org/10.1115/1.1330269.

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

[66] Ravani, B. and Roth, B., 1983, "Motion Synthesis Using Kinematic Mappings", Journal of Mechanisms Transmissions and Automation in Design-Transactions of the Asme, **105(3)**, pp. 460–467.

[67] Larochelle, P., 1996, "Synthesis of planar RR dyads by constraint manifold projection", ASME Design Engineering Technical Conferences, ASME, Irvine, CA.

[68] Bawab, S., Sabada, S., Srinivasan, U., Kinzel, G. L., and Waldron, K. J., 1997, "Automatic synthesis of crank driven four-bar mechanisms for two, three, or four-position motion generation", Journal of Mechanical Design, **119(2)**, pp. 225–231.

[69] Holte, J. E., Chase, T. R., and Erdman, A. G., 2000, "Mixed exact-approximate position synthesis of planar mechanisms", Journal of Mechanical Design, **122(3)**, pp. 278–286.

[70] Al-Widyan, K., Cervantes-Sanchez, J., and Angeles, J., 2002, "A Numerically Robust Algorithm to Solve the Five-Pose Burmester Problem", ASME Paper No. DETC2002/MECH-34270.

[71] Brunnthaler, K., Pfurner, M., and Husty, M., 2006, "Synthesis of Planar Four-Bar Mechanisms", Transactions of CSME, **30(2)**, pp. 297–313.

[72] Bourrelle, J., Chen, C., Caro, S., and Angeles, J., 2007, "Graphical user interface to solve the burmester problem", IFToMM World Congress, pp. 1–8.

[73] Larochelle, P., 2015, "Synthesis of Planar Mechanisms for Pick and Place Tasks With Guiding Positions", ASME Journal of Mechanisms and Robotics, **7(3)**.

[74] 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.

[75] Deshpande, S. and Purwar, A., 2019, "Computational Creativity via Assisted Variational Synthesis of Mechanisms Using Deep Generative Models", ASME Journal of Mechanical Design, **141(12)**, doi:10.1115/1.4044396, URL https://doi.org/10.1115/1.4044396, 121402.

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

[77] Bodduluri, R. M. C. and McCarthy, J. M., 1992, "Finite position synthesis using image curve of a spherical four-bar motion", ASME J. of Mechanical Design, **114(1)**.

[78] Lin, C.-C., 1998, "Complete solution of the five-position synthesis for spherical four-bar mechanisms", Journal of marine science and Technology, **6(1)**, pp. 17–27.

[79] Ruth, D. and McCarthy, J., 1999, "The design of spherical 4R linkages for four specified orientations", Mechanism and Machine Theory, **34(5)**, pp. 677 – 692, doi:https://doi.org/10.1016/S0094-114X(98)00048-2, URL http://www.sciencedirect.com/science/article/pii/S0094114X98000482.

[80] Brunnthaler, K., Schrocker, H., and Husty, M., 2006, Synthesis of spherical four-bar mechanisms using spherical kinematic mapping, Advances in Robot Kinematics, Springer, Netherlands.

[81] Zhuang, Y., Zhang, Y., and Duan, X., 2015, "Complete real solution of the five-orientation motion generation problem for a spherical four-bar

linkage", Chinese Journal of Mechanical Engineering, **28(2)**, pp. 258–266, doi:10.3901/CJME.2015.0105.003, URL https://doi.org/10.3901/CJME.2015.0105.003.

[82] Innocenti, C., 1995, "Polynomial Solution of the Spatial Burmester Problem", ASME Journal of Mechanical Design, **117(1)**, pp. 64–68, doi:10.1115/1.2826118, URL https://doi.org/10.1115/1.2826118.

[83] Plecnik, M. M. and McCarthy, J. M., 2012, "Design of a 5-SS spatial steering linkage", ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers Digital Collection, pp. 725–735.

[84] Li, X., Ge, Q. J., and Gao, F., 2014, "A Unified Algorithm for Geometric Design of Platform Linkages With Spherical and Plane Constraints", volume Volume 5B: 38th Mechanisms and Robotics Conference of International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, doi:10.1115/DETC2014-35218, URL https://doi.org/10.1115/DETC2014-35218, v05BT08A101.

[85] Ge, X., Ge, Q. J., and Gao, F., 2015, "A Novel Algorithm for Solving Design Equations for Synthesizing Platform Linkages", volume Volume 5C: 39th Mechanisms and Robotics Conference of International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, doi:10.1115/DETC2015-47981, URL https://doi.org/10.1115/DETC2015-47981, v05CT08A049.

[86] Ge, X., Purwar, A., and Ge, Q. J., 2017, "Finite Position Synthesis of 5-SS Platform Linkages Including Partially Specified Joint Locations", volume Volume 5B: 41st Mechanisms and Robotics Conference of International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, doi:10.1115/DETC2017-67809, URL https://doi.org/10.1115/DETC2017-67809, v05BT08A074.

[87] Ge, Q., Ge, X., Purwar, A., and Li, X., 2015, "A null-space analysis method for solving bilinear equations in kinematic synthesis of planar and spherical dyads", Proceedings of the 14th IFToMM World Congress, DOI, volume 10.

[88] Ge, X., Purwar, A., and Ge, Q. J., 2016, "From 5-SS Platform Linkage to Four-Revolute Jointed Planar, Spherical and Bennett Mechanisms", volume Volume 5B: 40th Mechanisms and Robotics Conference of International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, doi:10.1115/

DETC2016-60574, URL https://doi.org/10.1115/DETC2016-60574, v05BT07A081.

[89] Bates, D. J., Hauenstein, J. D., Sommese, A. J., and Wampler, C. W., "Bertini: Software for Numerical Algebraic Geometry", Available at bertini.nd.edu with permanent doi: dx.doi.org/10.7274/R0H41PB5.

[90] Wolfram Research, Inc., "Mathematica, Version 12.1", URL https://www.wolfram.com/mathematica, champaign, IL, 2020.

[91] Maplesoft, a division of Waterloo Maple Inc., "Maple 2020", URL https://www.maplesoft.com/products/maple/, waterloo, Ontario, 2020.

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

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

[94] 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.

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

[96] 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.

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

[98] 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.

[99] 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.

[100] 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.

[101] 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.

[102] 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.

[103] 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.

[104] 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.

[105] 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.

[106] 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.

[107] 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.

[108] 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.

[109] Liao, Q. and McCarthy, J. M., 1997, "On the Seven Position Synthesis of a 5-SS Platform Linkage ", ASME Journal of Mechanical Design, **123(1)**, pp. 74–79, doi:10.1115/1.1330269, URL https://doi.org/10.1115/1.1330269.

[110] 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.

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

[112] 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", Mechanism and Machine Theory, **44(6)**, pp. 1132–1143.

[113] Hayes, M. and Zsombor-Murrary, P., 2004, "Towards Integrated Type and Dimensional Synthesis of Mechanisms for Rigid Body Guidance", Proceedings of the CSME Forum 2004, London, ON, Canada, pp. 53–61.

[114] Sebe, N., Cohen, I., Garg, A., and Huang, T., 2005, Machine Learning in Computer Vision, Computational Imaging and Vision, Springer Netherlands, URL https://books.google.com/books?id=lemw2Rhr_PEC.

[115] Hemanth, D. and Estrela, V., 2017, Deep Learning for Image Processing Applications, Advances in Parallel Computing, IOS Press, URL https://books.google.com/books?id=vsFVDwAAQBAJ.

[116] Bertalmio, M., Sapiro, G., Caselles, V., and Ballester, C., 2000, "Image Inpainting", Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, ACM Press/Addison-Wesley Publishing Co., USA, SIGGRAPH '00, p. 417–424, doi:10.1145/344779.344972, URL https://doi.org/10.1145/344779.344972.

[117] Kingma, D. P. and Welling, M., 2014, "Auto-Encoding Variational Bayes", Computing Research Repository, **abs/1312.6114**.

[118] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., 2014, "Generative Adversarial Nets", Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds., Advances in Neural Information Processing Systems 27, Curran Associates, Inc., pp. 2672–2680, URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

[119] Yeh, R. A., Chen, C., Yian Lim, T., Schwing, A. G., Hasegawa-Johnson, M., and Do, M. N., 2017, "Semantic Image Inpainting With Deep Generative Models", The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[120] Ivanov, O., Figurnov, M., and Vetrov, D., 2018, "Variational autoencoder with arbitrary conditioning", arXiv preprint arXiv:1806.02382.

[121] Sharma, S. and Purwar, A., 2020, "Using a Point-Line-Plane Representation for Unified Simulation of Planar and Spherical Mechanisms", Journal of Computing and Information Science in Engineering, **20(6)**, doi: 10.1115/1.4046817, URL https://doi.org/10.1115/1.4046817, 061002.

[122] Dam, E. B., Koch, M., and Lillholm, M., 1998, Quaternions, interpolation and animation, volume 2, Citeseer.

[123] Kühnel, W., 2015, Differential Geometry, Student Mathematical Library, American Mathematical Society, URL https://books.google.com/books?id=qNBYCwAAQBAJ.