# PARAMETRIC SWEEP SEARCH FOR PARALLEL ROBOT WORKSPACE DETERMINATION

by

**CHE ZULKHAIRI ABDULLAH**

A thesis submitted to
The University of Birmingham
for the degree of
**DOCTOR OF PHILOSOPHY**

School of Mechanical Engineering

Collage of Engineering and Physical Sciences

The University of Birmingham
September 2013

# SYNOPSIS

The research presented in this thesis aims to augment the conventional kinematic-based parallel robot workspace determination into an interactive 3D visual system by highlighting the design issue clearly and providing important design information to the user in real-time. The conventional iteration involves heavy computation, high resolution data processing and multiple technical skills set which usually reduce the design option into amending an existing design most related to the requirement. The thesis presents a 3D simulation system that is open and modular and allows for algorithm and technical functional extensions.

Following an extensive literature survey on key aspects relating to parallel design development, parallel robot singularity, and search for workspace, five main phases of experimental works were undertaken to develop a strategic search and analysis of parallel robot's workspace.

Phase 1 involves the development of a 3-dimensional simulation system based on Python in order to search for workspace and singularity. The 3-dimensional motions are based on Kinematic, and should integrate easily with various algorithms. The simulation system produces a draft quality result, which is scalable to higher resolutions. Phase 2 involves the development of geometric singularity and Grassmann singularity real-time test. The code should work effortlessly when user redefines the architecture or geometry of the robot. Phase 3 involves the development of Boolean Parametric sweep search strategy that

provides an analysis and validation method for the system. The system has to consider various factors such as limit, edge, resolution, and travel direction. This phase involves the integration of 3D and 4D interpolation and extrapolation strategies including Trilinear, L-system fractal, Simplex, adjacency tree and Marching Cube. Phase 4 involves the development of a dynamic grid that is easily integrated into 2D and 3D databases. The haptic engine finds position and orientation information based on this dynamic grid. The grid is tested for a number of concepts such as Simplex, Binary tree and L-system fractal. It was found that the parametric sweep based on Boolean control is applicable in real-time dynamic grid system, where the direction, region and constraint are configurable by either the user or automatically by the system. Phase 5 involves the development of a controller for the Python simulation. This is a multiple Degree-of-Freedom device with two-way motor control for all axes, which is fitted with sensors to provide pre-processed value for distance and orientation on all axes. The controller is a haptic device that provides the user with force-feedback sensation while user operates and manipulates the device. The haptic device provides (i) control for the Python simulation, (ii) processed data to a Matlab and Solid Works system, and (iii) singularity-free control of an associated physical robot. The proposed system is finally verified against a number of applications.

# ACKNOWLEDGMENTS

First of all, I would like to express my deepest gratitude to my supervisor Dr Mozafar Saadat, in the School of Mechanical Engineering, University of Birmingham for his academic supervision, guidance and encouragement over the course of the research.

I would like to acknowledge the support from Yayasan Telekom Malaysia and Multimedia University Cyberjaya, Malaysia for providing the opportunity, time and scholarship which enabled this research to be carried out.

I would also like to express my heartfelt gratitude to my family (Rodiah Khalid, Aisya Imya and Aileen Fatima Imya) for their understanding and support in providing endless encouragement and motivation.

Lastly, I would like to thank my friend Mr Hamid Rakhodaie, Mr Ali Tayebisadrabadi and Mr Nik Farid Che Zainal and colleagues for all their assistance and support, and Mr Hisham bin Jabar for reading through my thesis.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# Chapter 1: INTRODUCTION

## 1.1 Background to the project

Parallel robot contribution in industry can be seen, amongst others, in medical, rehabilitation, machining, and assembly fields. It is also used to provide motion for platform simulators. The characteristics of parallel robot is identified as being rigid, accurate and high speed, where Song explains that due to manufacturing and assembly issues, the robot's error has to be reduced and compensated to achieve those qualities of being highly accurate [1-3].

Parallel robot commonly has a small workspace, and since it allows for parametric scaling to increase the workspace, it would require the increment of the robot's size too. Therefore, cost and work cell's size and obstruction within the workspace have to be considered early during the design stage.

Parallel robots are much stiffer, have more position accuracy and are capable of higher speeds compared to serial robots. It is demonstrated that the geometrical errors of a parallel robots are averaged out while those of serial robots accumulate [1,2]. Although parallel robots exhibit higher Force/Torque loads with high accuracy and less errors, their errors due to manufacturing and assembly have to be reduced and compensated to achieve those qualities [2,3].

The traditional methods for calibration require mobility restraining and dependency on error compensation or reduction algorithms, which limit the capabilities of parallel robot.

On the other hand, integrating sensors or redundant measurement devices into a parallel robot system is not a straightforward task and has to be considered during the design stage. An example of a developed parallel robot measuring device is described as being consisted of linear variable differential transformer, a biaxial inclinometer, and a rotary sensor that improves the measurement quality which is capable of performing simultaneous measurements of position and orientation data in one cycle [4,5].

There are various singularities within a general non-singular workspace or path. These include the issue with consistent acceleration, and the number of actuators which may operate at the limit and the optimization or interpolation problem with numerical approximation of the path, edge and position [6,7]. Poor convergence of numerical methods for solving 4n-dimensional two point boundary value problem (TPBVP) for an n-degree-of-freedom DOF manipulator has been demonstrated. It is also shown that numerical systems cannot quickly solve and produce results for such a computation [4]. Monte Carlo method has been used to estimate the scatter plot of a workspace, while a switching function has been used in situations where path parameters change has difficulty finding an optimum solution. Numerical methods fail with the existences of arcs in the path, where there are also control problems with non-constrained paths [6-8]. Furthermore, there are problems with the limitation of using different dimensional data and algorithm in solving time-optimal paths. The 3D solutions found using numerical method has to be mapped onto 2D methods for detail analysis like obstacle avoidance. This is also true for path planning, velocity and position planning, and other formulations which are not generally available in 3D [5].

Parallel robot path planning has to consider various other factors. This path planning is different to game design or CAD 3D path planning, where a parallel robot as a closed-loop system has to consider kinematic, geometric constraint, Grassmann error, force, velocity and speed error as its feature [9]. Qin explains that parallel robot path planning has to consider geometric constraint, the $C_{space}$ and $C_{obstacle}$ which are the space occupied by the obstacle [8]. Even when a Bezier spline formulation can be performed in 3D to validate parallel robot's path, it has to consider all those other elements for a closed-loop structure to move safely and perform its tasks [10].

Parallel robot's 3D problem is usually remapped into 2D so that traditional and proven methods can be used to solve this problem, and then remapped again into 3D. Roy demonstrates 2D slicing methods which utilizes the 2D mapped formulation to get a 3D representation of the solution. The method is referred to as $C_{space}$ mapping algorithms for 2D sliced workspace [11]. A 4 Cspace transformation for determining the 3D representation is also suggested which is based on point, line, circle and the finite dimensions of the robot [12].

The issue with parallel robot analysis is the amount of information, design complexity, and closed-loop structure where relationship between each component has to be acknowledged. Here a primary objective is to reduce the problem into a small and simple 2D problem, where the focus is on the issue itself rather than the whole 3D space. Methods such as slide-step have been introduced in order to rapidly establish the maximum reach or the edge, and to rapidly find the end pose or the target. In order to improve trust in a questionable region, a random generated test data has been suggested as a sub goal [6].

The force-feasible C-space for a path, where the best path on a mesh is found, has been suggested through avoiding singularity loci space as the travel is done based on constant orientation method. The other problems related to interpolation are listed as certain factor may not be considered by the system such as the orientation compliance or the gradient angle changes between the two poses from start to finish position [7].

There is a significant difference when the parametric sweep search for non-singularity path is done using serial or parallel search method. An example of experimentation with parallel search in a C-space consists of a seed generator placed at any node with certain criteria for branching out to form the parallel search [8]. In parallel robot design activity the literature shows the ambiguity of having singularity within a workspace, where there is an issue of data complexity and multiple solutions with varied result.

In this thesis a simulation system is developed to help visualize the parallel robot design problems with a focus on the workspace search problem. The system is also capable of providing a quick and draft solution before the designer proceeds with the final design validation using kinematic based numerical methods. The system extends the traditional parametric sweep method by exploring into various aspects of grid type and technical parameters.

The parallel robot configuration is constrained by the parameters that define its geometry including coordinates of the joint on the base, coordinates of the joint on the travelling plate, and the actuators' length. In the research presented in this thesis the chosen geometry follows an existing similar physical robot at the laboratory. A similar geometry has been developed for the numerical system based on Solid Work and Matlab.

Investigation has included development of multiple systems of Python simulation, numerical system using Solid Work and Matlab software tools, a physical robot, and the control between all three systems. The combinatorial classes for the chosen structure is one class of 6-6 ($|^6$) and two classes of 6-3 ($/\backslash^3$) and (/|\/\|) [13].

## 1.2 Aims and objectives

The overall aim of the project is to develop an assistive tool for the design of parallel robot with a specific focus on rapid identification of workspace boundaries and singularities through a 3D simulation system. The result is achieved through real-time control and manipulation of proposed haptic controller devices.

The specific objectives necessary to achieve the project aim were;

1. To develop a 3D simulation system that display parallel robot's CAD model that conforms to a kinematic model.
2. To develop parametric sweep search methods that search for parallel robot's workspace
3. To develop a Grassmann validation system that checks for Grassmann-related errors within the workspace region.
4. To develop a haptic controller that provides force-feedback to user that helps identify singularity errors within a 3D space.

## 1.3 Thesis layout

The thesis is organised into 10 main Chapters. Chapter 1 gives a brief introduction to the project including background to the work, research aims and objectives. Chapter 2 provides a detailed literature review focussing on parallel robot’s design issue, challenges in parallel robot design, singularity, workspace, haptic controller and example applications.

Chapter 3 details the simulation system development, kinematic model and workspace numerical calculations. Chapter 4 discusses the Grassmann algebra and a weighted ranking system to identify Grassmann problem. Chapter 5 is a detailed review and experimental methods on parametric sweep, which includes condition test theory, various parametric sweep methods and data population concepts. Chapter 6 investigates Boolean control for parametric sweep, and includes many case studies. Chapter 7 discusses robot path planning issue and includes definition of search region, path planning methods and numerical validation. Chapter 8 discusses Haptic controller development, which includes controller kinematics, controller structure design and validation, sensor development and two case studies. Comprehensive results and discussion of the example applications are presented in Chapter 9. This includes applications on rehabilitation and machining. Chapter 10 provides conclusions, summary, contributions of the thesis and a suggested future work.

# Chapter 2: LITERATURE REVIEW

## 2.1 Challenges in Parallel Robot simulation design

There are several challenges in the development of parallel robot simulation system, namely the large variations of parallel robot architecture. The closed loop structure has different linkages type, therefore simulation strategy has to consider a new kinematic system when the architecture is changed.

Ben-Horin has developed a parallel robot visualization system based on Grassmann-Cayley algebra (GCA), which is simplified into GCA brackets. A bracket is a vector, and the two basic operations used in the GCA are the join and meet operators. Join operator is the union of two vector spaces and the meet operator is the intersection of two vector spaces. Ben-Horin only considers a limited number of different Gough-Stewart Platform (GSP) for the software development [14].

Ben-Horin system is the foundation idea for the development of the 3D Python simulation system, called **PyPKM**. Ben-Horin has simplified the concept for designing a parallel robot into bracket and vector space. The visualization allows user to construct visual image of the parallel robot limit and ability while it is being developed early during the design stage. Previously, the design stage begins with analysis and review of the project brief, which leads to literature study and review of similar previous design regarding the robot's task. Then, highly accurate yet costly and timely production of Solid Works model begins, and leads to various Solid Works analysis and simulation. The control method is then developed by using Matlab. The initial design stage stops prematurely, thereby leading into unexpected results or wrong design. Therefore, helper software like Singulab is an important contribution for parallel robot development. However, these systems are demanding on programming skill and complex techniques [14].

Dash highlights the many researches on finding and optimizing the different geometry configuration available for parallel robot design. According to Dash, the main issue is to get the most suitable topology for the given task. Dash developed SEMORS-PKM simulation software which optimizes the topology [15].

Xi developed a Design for Reconfigurability (DfR) system which is based on Axiomatic Design theory. Xi's system consists of modular configurations of 6-DOF, 5-DOF, 4-DOF and 3-DOF parallel robots which are planned for an attached, detached and partial condition. The simulation is focusing on emergency behaviour when any part breaks down and become partially detached as in space applications [16].

## 2.2 Parallel Robot's Singularity

Stewart Platform may reach singularity when the platform is within the same plane of one of the leg and when the platform rotates $90^O$ about an axis perpendicular to it. Lazarevic mentions that singularity is a position where the mechanism loses controls. Some of the known methods for identifying singularity condition includes a) by monitoring the condition number of the Jacobian matrix, b) when all six lines associated to links intersected one line and c) singular configuration is obtained by rotating the mobile platform around the vertical axis by an angle of +-pi/2. The three methods mentioned here describe the Pencil line method which is explained as Grassmann ranking system or Grassmann-Cayley method in Chapter 4 [17].

The ranking for singularity is generally classified as a) check for collision point, where an axis can be formed, by checking the plane orientation, b) check for planar orientation, where a plane may become coplanar with a pencil line, side vector or another planar vector and c) check for infinity condition, where a coplanar condition may develop an axis which may promote instantaneous rotation axis (IRA) condition.

Rojas distance-based formulation shall be examined to understand the integration with the Python system to have a quick reverse-engineer solution when a task has singularity by assisting user in improving the currently investigated structure. The distance-based adjustment allow for new joints position (new architecture), which create new problem with configuration singularity [18]. Chablat devise a distance formulation for solving structure with different joint configuration or specifically different joint types [19]. The Python 3D simulation system is highly reconfigurable, where user can change the geometry, linkage relationship type, constraint and platform’s shape. However, to get the best geometry for a given task would require extensive research on geometry optimisation and its relationship to path, workspace and singularity. Due to the complexity of the task, this part of the research shall be regarded as future works.

## 2.3 Parallel Robot’s workspace

The parallel robots are closed loop mechanisms that exhibit higher accuracy and stiffness compared to their serial counterparts. A critical problem with parallel robots is the existence of singularity points in the workspace due to the limitation of joints motion and orientation of the moving platform. Typically, square, spherical or hemispherical grid systems are used to determine the workspace for parallel robots [1,2]. Merlet presented various workspaces such as constant orientation or translation workspace, which are application specific search methods and require low-to-medium computational time Merlet’s other workspace types includes maximal, inclusive, total orientation, dextrous and reduced total orientation which requires orientation check for a specific task or region, or other advanced searches.  These are generally computationally complex processes and require additional attribute or constraints such as Grassmann vector, stiffness, and cost factor [20]. The traditional geometrical grid search is not suitable for extendible and complex search [21]. Bandyopadhyay claims that the search for orientation and constant position workspaces is difficult to perform but is important in terms of their application [22]. The search strategy is a method of moving the end-effectors along a path following a strategy like cubic grid inside an envelope, then for the system to perform data analysis at the test positions placed along the path. This is a test to validate singularity and kinematic value. Yu explains that the common method of using kinematics to relate unknown kinematic parameters P to the known information of the manipulator M is prone to error, due to the difficulty in getting a closed loop form for the kinematic solutions [23]. Parametric sweep is a square grid systems which are used to determine the workspace for parallel robots [1].

## 2.4 Haptic Controller

There are many numerical researches on Parallel mechanism, which leads to investigations on kinematic, singularity, optimization, path and many more. Zefran wrote that numerical system failed at certain condition like detection of at least one actuator that can still move within a singularity condition [5]. Not much visualization work has been developed to aid designer to design and verify their Parallel mechanism design. Horin develop a Parallel mechanism simulation system called Singulab based on Grassmann-Cayley algebra [14]. The Grassmann-Cayley method is based on linear and planar element position and orientation check for coplanar condition Typically, Parallel mechanism simulation is done using commercial software like Matlab, Solid Work and ADAM [21]. The simulation is either specific to a problem or reconfigurable for various geometries.

The interactive singularity analysis is based on visualization alone, which brings the issue of human sensory limit. Barnett-Cowan research shows that human perception ability is limited for orientation and position data in 3 dimensional spaces, and additional information such as touch, light and sound improves the probability for motion cueing and human sensory ability [24]. A Parallel mechanism is a complex structure with multiple moving components like joint's position, stroke changes and travelling plate and visualization of this structure in 3 dimensions has other issues like culling and hidden faces.

Uchiyama develop a 6DOF haptic controller, where the topology is a delta mechanism at the base, and serially connected gimbals is placed on the next layer, and the end effectors is a 1DOF rotary joint. Delta mechanism has high speed and compact footprint, however, the kinematic is complex and it is a low stiffness structure [21]. Kim lists the requirement for haptic controller as a low inertia, high stiffness, large force, simple kinematic, dual-drive system, low-friction and low weight. The Delta mechanism can be improved as a haptic controller when the travelling plate's diameter is reduced, where the kinematic performance is now improved [25].  Okamura explains that Da Vincci and Phantom haptic system require complex control structure in order to get the haptic force works properly in a surgical task. The problem is also related to the workspace type related to compliance (end-effectors orientation) and stiffness (due to Grassmann error and sudden changes to DOF) [26].

Zhang use 6D Roydomm system to calibrate a Stewart platform, where the problem is with the detection and tracking of the travelling plate's position and orientation that needs to be free from any measuring device attachment, weight and collision with the components [27]. Colton highlights the drift and data combination problem with orientation sensor, where low-pass and high-pass filter should be able to produce accurate result [28]. Perl describe the development of 6DOF tracking system which demonstrate the Madgwick algorithm for IMU [29]. Streng explains that the primary element in 3D haptic API development is the collision detection, since this detection is an iterative process and creates delay in haptic rendering [30]. H3D API does not provide collision detection, which is handled by Python. H3D provide the function for force, spring effect, and timer and viscosity effects. Paneels discuss about the development of rapid prototyping toolkit for haptic visualization, since the product cost has been lowered when Novint Falcon becomes publicly available. Paneels mentions that the main focus of those rapid prototyping kit is the definition for surface or collision, where the parameters will create virtual sensation that describe smooth, bump, hard surfaces, etc. [31]. Ruffaldi and Ayache advices on standard test for haptic [29,30] which Ayache describe the key factor for haptic engine which are gradient forces and edge extractions. Two primary haptic renderer which are god-object and Ruspini renderer address the relationship between end-effectors (user) and the contact surface. H3D renderer surface parameters are stiffness and damping.

Korobeynikov literature shows that Stewart Platform design begin with CAD/CAE software, and later evaluated using existing library system for the CAD/CAE system or done by hand [34]. Brezina use linear block-by-block State Space in Lab view to minimize computation cost for control of Stewart Platform [35]. Dongsu describe that 6DOF flight simulator uncertain parameters can be divided into two groups, namely the constant and the time-varying. The constant is known and being part of the architecture. The time-varying is a form of System Dynamics (SD), and need to be simplified as State Machine. Supervisory control is required when the system runs as Discrete Event System (DES) with prepared path. Dongsu further adds that adaptive control for non-linear system can be used to identify constant uncertain parameters. Adaptive control with feedback should address the dynamic changes to the search for operating region according the search state [36]. Zhu develop fuzzy support vector machine control to remove the nonlinear, uncertainty characteristics and external disturbance of parallel robot. According to Zhu, sliding mode can adapt to system disturbance, while fuzzy logic is sensitive to disturbance and neural network has congenital effect of getting into local minimum easily. Zhu Support Vector Machine (SVM) intends to reduce the dependence on user experience in finding dynamic solution. This would help the Master-slave relationship quickly find an optimum state for the sliding mode [37].

The research has developed an experimental haptic engine that performs in 3D space. The experimental haptic engine considers factors like parallel robot geometric singularity and Grassmann singularity in real-time. And, it use fast pre-processed weighted value for cost-factor when dealing with path and trajectory optimization. The iteration process which seems to slow down haptic has been replaced with a faster 3D directional and region-scoping method. The uses of 9 Degree-of-Freedom (DOF) Inertial Moment Unit (IMU) and RGB camera for tracking orientation and position has helped to improve the haptic interaction for a large dimension device, that allows for obstructive view, higher dimension of linear and angular travel for a haptic controller.

There are various researches done on robotic rehabilitation and their aim is to reproduce accurate motion for a variety of condition and constraint, reduce the workload and man-hour for a therapist intervention and the ability to access recovery performances by measuring force and motion patterns. However, robotic element is prompt to various singularity conditions even within the geometric-safe region. Therefore, path planner has to consider other singularity factor like Grassmann error, specific workspace error and stiffness error during a motion.

## 2.5 Limb rehabilitation strategy

Therapist intervention should be minimized by introducing elements like motion recording and optimization mode, haptic force-feedback and reduction of the workload and function. Where usually, a few therapist is required to handle a patient, a robotic system should perform with only one therapist that focus on training the robot to perform the rehabilitation accurately, rather than performing the actual rehabilitation himself. The robotic system should be able to track and record patient and therapist's performance. With optimization and certain improvement algorithm, the robotic system should be able to propose a new motion scheme. Therapist's workload shall be reduced, for the next sessions. Current system like GaitMaster5 (GM5) is a specific-task robot system which focuses on walking and stair-climbing activities. GM5 moves patient's foot involuntary which might not be within his comfort or even allowable motion region, therefore patient expectedly exert forces on the footpad to discourage the robot from continuing with the selected trajectory. And, according to Yano, the robotic system reduces the therapist workload and improves the rehabilitation quality, by removing some heavy workload with the robotic system capabilities and repeatability [38]. Marchal reports that certain rehabilitation motion is not within the robotic capacity, therefore Marchal use Virtual Reality environment to substitute for certain motion sensation, like reproducing effect via height, orientation and velocity variations [39].

Rutger ankle with haptic interface provides a force-feedback to motivate, encourage or force the limb to follow a certain routine. Furthermore, Cioi concludes that visualization with rehabilitation games activities for the Rutger system help improve the process by providing something for patient to focus and engage themselves and remove the anxiety of having an ankle strapped to a robotic device. Rutger ankle provides resistive force to encourage patient to try harder to achieve the objective of the games, which is a strength exercise. While a flexibility exercise, focus on repetitive motion near their limits of motion[40]. IIT High Performance Ankle Rehabilitation robot requires actuation redundancy to reduce the possibility of getting into singularity region. A condition where an error within a non-singularity region which is detected by Grassmann could be catastrophic is being identified especially when in an instantaneous-rotation-axis (IRA), a condition where the system suddenly loss its stiffness and introduce an unwanted degree-of-freedom (DOF). This may result with the system itself collapsing.

The Gwangju Institute of Science and Technology (GIST)'s rehabilitation robot provide therapist with the ability of reconfiguring the robot to satisfy certain desired position, orientation and path. Lum's comparative study between conventional and robotic movement therapy shows the improvement in the areas of strength and reach exercise. The strength component is attributed to the constrained mode, where patient is expected to exert force to achieve the given goals. The strength component is directly related to improved reach gains and, robotic system does not have fatigue condition for durability during repetitive procedures [41].

The requirement for human intervention has been mentioned by various researchers for various stages in the rehabilitation procedures. Robotic elements has been successful in reproducing stereotyped movement and repetitive movement with sensor-recorded performance data [7]. Syrseloudisa mentions that Stewart Platform is not suitable for ankle rehabilitation due to its rotation characteristic around the vertical pivot strut which contradict with ankle movement [42]. Sui also identified a few ankle rehabilitation system[43]. Belda-Lois research on the subject of a patient interaction with Lokomat, which is based on impedance-control due to the possibility of combined contact-free position and contact-force control, shows that patient will eventually produce active torque that is exerted into the system. This extra interaction will cause a change in the position deviation, or shall be restrained by the robot's closed-loop system. Either way, the system or patient will be having disadvantages and may cause discomfort. Belda-Lois develop a patient-Lokomat-patient interaction algorithm based on studies done by Hogan regarding the interaction of manipulator and its environment, and the result shows that patient intention and the robot motivation is not always synchronizes and similar, therefore an adaptation between the two is necessary [44]. Patton discusses the differences between two adaptive-training strategy of whether patient benefits more from a force that enhance the errors or forces that reduce the errors, which Kahn suggested that error-reduction does not add any benefit[45], [46]. Reikensmeyer acknowledge that a condition known as after-effects which is caused by the removal of force that redirect or direct a movement from the robot system cause movement deviation, and the patient correction behaviour includes the increment of impedance to reduce trajectory error. Patient normally improves their ability to estimate and reduce the after-effects sensation after each new session. [47]. Kahn and Reikensmeyer explain the fundamental concept to assistive robotic rehabilitation which is primarily based on ARM or MIME. Patient initiates the movement, while ARM guide help complete the movement along a smooth trajectory. MIME guide copies mirror image of the movement of patient's unimpaired limb [46, 47]. Oldewurtel  studies on impedance control include visual inspection of the path lines, which include the calculation of supportive force for arbitrary ADL inside a force-field, adaptive distance and the allowable freedom of motion and movement ahead of the goal [45]. Having access to the 3-dimensional path lines curvature and its attribute help the therapist to evaluate patient's performance and the goal. The Python simulation system provides various analysis facilities for assisting this path lines related task. It allows adjustment, placement of planar slice analysis tool, path planning activities, path optimization activities and integration with haptic engine.

## 2.6 Cutting path strategy based on Bezier

Bezier provide robust control point curve construction and control system which is useful in 2D or 3D design. Bezier has simple visual representation and the control points are related directly to its function. Bezier provide adequate rule for complex curve formation and subdivision [49].

Amato used divide and conquer approach to produce curve segments with optimal running time of$0(nlogn + k)$. This combined with Bezier should help parallel robot's to manage obstacle and surface mesh interaction [50]. Simas use Bezier patch representing a convex shape that help guide the end-effectors in a scallop height algorithm while ensuring that the end-effectors is parallel and equally spaced for each passes. In this research it is planned to demonstrate the numerical and the simulation capability by replicating the method. This include the definition of a surface patch using Bezier, then reproduce the end-effectors and surface patch interaction, and validate the path using numerical system, by checking its performance in terms of velocity and force. Simas's method of using Bezier produce parallel and equally spaced potential path for the end-effectors.

# Chapter 3: KINEMATIC MODEL

## 3.1 Introduction

This chapter discusses the development of CAD models that follows the motion defined by kinematic rules. The CAD model is developed using Python and it can have various geometries and configurations. For the purpose of this research, a hybrid and a hexapod configuration is chosen to allow for numerical validation using existing physical robot and numerical system developed within the research group.

The haptic system has been proven to work with the Python simulation as a standalone system, with the Numerical system as connected system, and with the physical robot as a direct controller or control-validation by the Numerical system.

## 3.2 Kinematic model

Inverse kinematic and dynamic in control strategy is an important aspect of PKM development, and studies have been done based on various configurations [84-87]. The experiment is based on physical robot which is controlled via Solid Work CAD model and Matlab numerical simulation. The numerical result and the Python simulation is then compared for accuracy. The kinematic presented here is based on a robot structure that is defined as a 6-UPU-3-UPR structure [51]. The motions can be described by a transformation matrix that consists of six independent variables which are linear and rotational motions in XYZ.

Figure 3-1 explains the kinematics of the structure. Equation 1 gives the centre points $E_{xyz}$and $o_{xyz}$ based on homogeneous linear and rotational transformation matrix:

$$[T]\begin{bmatrix} c\emptyset c\psi & -c\emptyset s\psi & s\emptyset & 0 \\ c\Theta s\psi + c\psi s\Theta s\emptyset & c\Theta c\psi - s\emptyset s\psi & -c\emptyset s\Theta & 0 \\ s\Theta s\psi - c\Theta c\psi s\emptyset & c\psi s\Theta + c\Theta s\emptyset s\psi & c\Theta c\emptyset & 0 \\ L & M & N & 1 \end{bmatrix} \tag{1.a}$$

$$L = s\psi((mc\Theta\theta + ns\theta\Theta) + c\psi(lc\emptyset - s\emptyset(nc\theta\Theta - ms\theta\Theta) \tag{1.b}$$

$$M = c\psi(mc\theta\Theta + ns\theta\Theta) - s\psi(lc\emptyset - s\emptyset(nc\theta\Theta - ms\theta\Theta) \tag{1.c}$$

$$N = ls\emptyset + c\emptyset(nc\emptyset\Theta - ms\Theta\emptyset) \tag{1.d}$$

Where,$\theta$,$\emptyset$,ψ represent the rotational components and l, m and n are linear motions in X, Y, Z.

Equation 2 gives the second position after the platform completed its motion.

$$A_{i1} \times T_H = A_{i2} \qquad \text{For i=1...6} \tag{2}$$

Where, $T_H$ is transformation matrix for platform A and $A_{i2}$ is second position of joints after the actuator's motion.

Equation 3 gives Platform E's joint position.

$$E_{j1} \times {T_O^E}^{-1} \times T_T \times T_H \times T_O^E = E_{j2} \qquad \text{For j=1...3} \tag{3}$$

Where,$E_{j1}$ and $E_{j2}$ are the initial and second positions of the joints on the platform E.

$T_O^E$ is the transformation matrix.

Equation 4 and 5 gives the actuator's length.

$$l_{Hi} = A_{i2} - B_i \quad \text{For i=1...6} \tag{4}$$

$$l_{Tj} = E_{j2} - A_{j2} \text{For j=1...3} \tag{5}$$

Where,$l_{Hi}$ and $l_{Tj}$ are the position vector of the actuators.

The degrees of freedom for each joint can be identified by $U_X = (1 \quad 0 \quad 0 \quad 1)$ and $U_Y = (0 \quad 1 \quad 0 \quad 1)$ for their initial positions.

$$u_X = U_X \times T_H \text{ (6.a)}$$

$$u_Y = U_Y \times T_H \qquad \text{(6.b)}$$

$$u_{2X} = U_X \times T_T \times T_H \qquad \text{(6.c)}$$

$$u_{2Y} = U_Y \times T_T \times T_H \qquad \text{(6.d)}$$

Where, $u_X$ and $u_Y$ are joints coordinates for platform A, and $u_{2X}$,$u_{2Y}$ are joint coordinates for platform E:

$$\alpha_{Ai} = cos^{-1}\left(\frac{u_X.L_{Hi}}{|L_{Hi}|}\right) \qquad \text{For i=1...6 (7)}$$

$$\beta_{Ai} = cos^{-1}\left(\frac{u_y.L_{Hi}}{|L_{Hi}|}\right) \qquad \text{For i= 1...6} \tag{8}$$

$$\alpha_{Ej} = cos^{-1}\left(\frac{u_{2X}.L_{Tj}}{|L_{Ti}|}\right) \qquad \text{For j=1...3} \tag{9}$$

$$\beta_{Ej} = cos^{-1}\left(\frac{u_{2y}.L_{Tj}}{|L_{Tj}|}\right) \qquad \text{For j=1...3} \tag{10}$$

Where $\alpha_{Ai}$ and $\beta_{Ai}$ are angles between the actuators of the hexapod and the platform A in X and Y directions respectively.

Where $\alpha_{Ei}$ and $\beta_{Ei}$ are the angles between the tripod actuators and platform E.

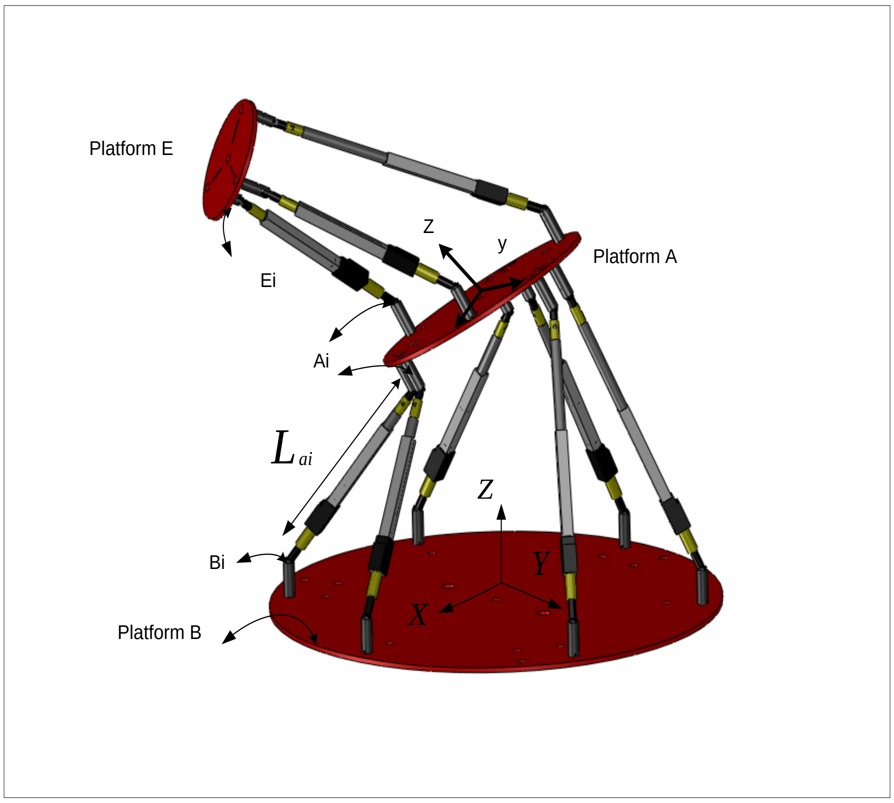


Figure 3-1 Schematic of hybrid parallel robot

## 3.3 Workspace calculations based on inverse kinematic

Workspace calculations are based on a development of inverse kinematic formulation [51]. Equation 11 gives the end effectors' position and size.

$$L_{Hi} = A_i \times T_B^A - B_i \qquad i \in \{1 \dots 6\} \quad (11)$$

Where, $L_{Hi}$ is position vector of actuators connecting platform B to A, $A_i$ is the initial position of joint on platform A and $B_i$ is the position of joints on the platform B.

Equation 12.a and 12.b gives the platform E's transformation matrix.

$$T_B^E = T_A^E \times T_B^A \quad (12.a)$$

$$T_A^E = T_n \times R_{\theta_T} \times R_{\emptyset_T} \quad (12.b)$$

Where, $T_B^E$ is transformation matrix of platform E related to platform B, $T_B^E$ is transformation matrix of platform E to B and $T_B^A$ is transformation matrix of platform A to B.

Equation 13 gives the length of the actuators.

$$L_{Tj} = E_j \times T_B^E - A_j \times T_B^A \quad j \in \{1 \dots 3\} \qquad (13)$$

Where, $L_{Ti}$ is position vector of the each actuator connecting platform A and E, $E_j$ is initial joint position on platform E and $A_j$ is joint positions on top of platform A. The work volume search is done and programmed using Solid Work and Matlab.

## 3.4 Conclusion

Kinematic analysis provides position and orientation for a single point in 3D space. Kinematic for parallel robot development considers the transformation matrices which consider angle and stroke changes. Kinematic modelling does not do any singularity validation, and will not highlight any issues regarding geometric constraint limitations. It is known that kinematic methods have difficulty in getting a closed loop form for kinematic solutions. The Python simulation motion is based on kinematics, with large buffer for probable errors. Some example or case study shown in this research does not elaborate the kinematic aspect of it, since transformation matrices varied and many. Chapter 3 gives an overall overview about kinematic and how it is being used in this research.

# Chapter 4: GRASSMANN ALGEBRA

## 4.1 Introduction

This chapter discusses the probable existences of Grassmann error inside a safe workspace region due to a specific condition or pose. Grassmann has the advantage of checking for co-planar position or orientation, while the geometric check does not. Grassmann algebra is a method that checks for angle differences between large set of vector lines, where the vector lines represent parallel robot’s legs or edges. A co-planar condition and instantaneous-rotation-axis (IRA) condition is detrimental to the robot structure. Therefore, Grassmann singularity test is an essential element in workspace search. Previous chapter discuss about kinematic, which provide probable position and orientation for the end-effectors centre-point position in 3D space.

## 4.2 Grassmann theory

A Python 3D simulation system has been developed based on Grassmann pencil-line terminology, where it’s kinematic has been validated using numerical system, and the basic singularity check is based on structural geometric and Grassmann validation. The full 3D system is based on Grassmann algebra. Previously, Ben-Horin has developed Parallel robot simulation and analysis system based on Grassmann-Cayley bracket concept [47, 48]. This research extends the concept further by integrating the Grassmann singularity test-condition within a safe region as the base system. Then, the research moves forward into complex problems like parametric sweep search, extended geometric and fractal sweep search and path planning [51].

Grassmann error rule includes – a) lines for any elements meet at infinity, b) lines for any elements has a difference angles equals to 0 or 180 degrees and c) a virtual axis is formed where opposing force may produce instantaneous rotation that produces unwanted degree-of-freedom. If total weighted, $\omega$ T for all pairs is found, then use weighted sum to find non-singularity or singularity position inside or outside the workspace during application. Interpolation test for the Python path, by using scatter distribution of test position for example by doing 3 units of 1D-interpolation or 'Trilinear' and distribute the scatter plot by using 3D random generator.

The condition test theory is based on identifying the geometrical constraint, then checking the quality of the non-singular pose by checking for Grassmann error and various workspace requirement errors like compliance and trajectory. First, it checks the geometry error, since design issue for parallel robot particularly Stewart Platform is identified as joint angle, leg stroke, leg location, kinematic complexity, limited workspace and singularity, and then the system check for Grassmann condition, where line geometry is used to analyze, validates and displays the Grassmann singularity condition [54].

The parallel robot geometry definition is based on a circle divided into variable six point's position following Charters method which allows for reconfigurable geometry automatic creation and easy numerical validation [55].

Kinematic calculation provide the end-effectors position which need to confirmed by checking the geometric limit, which is calculated by checking the leg's stroke limit and angle limit (Eq.14).

Where angle 1, 2 = Vector angle for leg between 2 poses,

$$angle_n = x_n i + y_n j + z_n kn \in \{1,2\} \tag{14.a}$$

$$p = x_1 i + x_2 i \tag{14.b}$$

$$q = y_1 j + y_2 j \tag{14.c}$$

$$r = z_1 k + z_2 k \tag{14.d}$$

$$|S_j| = \sqrt{p^2 + q^2 + r^2} j \in \{1 \dots 3\} \tag{14.e}$$

Where, $pose1 = (x_{1,}y_1, z_1)$ and $pose2 = (x_{2,}y_2, z_2)$. Moreover $S_i$ is stroke size of the actuators connecting platform A and E (Tripod).

The Boolean condition for Grassmann (Eq.15) which check for coplanar, line-line join, line-line meet, plane-line meet and point-line join is presented here.

$$G_{GN} = \sum_{Iteration=0}^{n} \left( f(pos, limit, error) \wedge limit \rightarrow error_{a,b,c,d,e} \right) \tag{15.a}$$

Where $Error_{\mathrm{a,b,c,d,e}}$ represent the Grassmann error condition described in Equation 6.d. And, $\mathrm{G_{GN}}$ is a set of Grassmann error for a given set of iteration [4]. The iteration is created by drawing pencil line using the structure's line and edge. The limit is the threshold value for error state.

The formulation is adapted from a Wikipedia website entitled ‘Plucker coordinates’ [56]. For a case where x =($x_1$, $x_2$, $x_3$) and y=($y_1$, $y_2$, $y_3$)for line L, therefore displacement along L is found as scalar multiple of d=y-x. The point being displaced, known as moment, $m = x.y$.

x =($x_1$, $x_2$, $x_3$) , y= ($y_1$, $y_2$, $y_3$) for line L. (15.b)

Where m is the moment for this point displaced from its origin.

$$m' = \begin{bmatrix} x'_i & y'_i \\ x'_j & y'_j \end{bmatrix} = \begin{bmatrix} x_i & y_i \\ x_j & y_j \end{bmatrix} \begin{bmatrix} \lambda_{00} & \lambda_{01} \\ \lambda_{10} & \lambda_{11} \end{bmatrix} \quad (15.c)$$

Where m’ are the linear combinations of the columns of m, for some 2x2 non-singular matrix ∧. Grassmann error, $error_{a,b,c,d}$ condition is described here as

a) coplanar condition, when d.m’+m.d’=0,

b) line-line join condition, when$(m.d')_{x_0} + (dxd') = 0$,

c) line-line meets condition,

When$(x_0{:}\,x) = (d.m'{:}\,m \times m)$,

d) plane-line meets when given a plane with equation $0 = a^0 x_0 + a.x$. The point of intersection is given as$(x_0{:}\,x) = (a.d{:}\,a \times m - a_0 d)$,

e) point-line join condition, when $0 = (y.m)x_0 + (y \times d - y_0 m).x$

Where displacement, d=(y-x) and Plucker coordinates= (d: m) [56].

Therefore, the total weighted value found here is the evaluation of end-effectors position for Grassmann error and geometry error.

Singularities of the hexapod need to be considered to overcome platform position for desired one.

$$H = \begin{bmatrix} S_1 & \dots & S_6 \\ S_1 \times q_1 & \dots & S_6 \times q_6 \end{bmatrix} \tag{16}$$

Where, $H$ is kinematic relation of the hexapod structure.

$$q_i = O_B A_i - O_B O_A i \in \{1 \dots 6\} \tag{17}$$

Where, $S_i$ is unit vector of each actuator attaching platform A and B. moreover, $O_B$ is centre of platform B , $O_A$ is centre of platform A and $A_i$ is position of joints on platform A, that value could be calculated for each motion.

The singularities can be identified while value of determination of the matrix H is zero.

## 4.3 Weighted value ranking based on Grassmann algebra

The weighted value is a collection of process starting with the establishment of a grid, then populating the grid with a strategy, then check for singularities. The result is then processed for Grassmann and other ranking criteria, for example nearest distance to centre-point or pre-defined path. This ranking criteria, grid and grid population method is extendible. The concept is presented here in Equation 18, 19 and 20.

$$angle_1 = \left(\underline{x}i_1 + \underline{y}j_1 + \underline{z}k_1\right) \tag{18.a}$$

$$angle_2 = \left(\underline{x}i_2 + \underline{y}j_2 + \underline{z}k_2\right) \tag{18.b}$$

Where angle 1, 2 = Vector angle for leg between 2 poses

$$p = \underline{x}i_1 + \underline{x}i_2 \quad (18.c)$$

$$q = \underline{y}j_1 + \underline{y}j_2 \quad (18.d)$$

$$r = \underline{z}k_1 + \underline{z}k_2 \quad (18.e)$$

$$|a, b, c, d| = \sqrt{p^2 + q^2 + r^2} \quad (18.f)$$

Where |a, b, c| = legs stroke or vector magnitude, |d| = distance to the plate's centre.

$$angT[0-3, center] = \cos^{-1}(\bar{a}.\bar{b})/(|a|.|b|) \quad (18.g)$$

$$angH[0-6, center] = \cos^{-1}(\bar{a}.\bar{b})/(|a|.|b|) \quad (18.h)$$

Where AngT represent tripod angle and AngH represent hexapod angle,

$$f(pos, limit, error) = pos \wedge limit \rightarrow error \quad (18.i)$$

Where pos = end-effectors position x, y and z, limit is the search limit, and error is the stroke, angle and collision error.

$$\omega_a = \sum_{n=0}^{3,center}(\text{angT}[n] - d) > 0 \quad (19.a)$$

$$\omega_b = \sum_{n=0}^{6,center}(\text{angH}[n] - d) > 0 \quad (19.b)$$

$$\omega_c = \sum_{n=0}^{3,center}(|a[n], b[n], c[n]| - e) > 0 \quad (19.c)$$

$$\omega_d = \sum_{n=0}^{6,center}(|a[n], b[n], c[n]| - e) > 0 \quad (19.d)$$

$$\omega_e = \sum_{n=0}^{x} (|d[G_n]| \cap |d[G_{n+1}]| \neq 0) \qquad (19.e)$$

Where $\omega_a$ = weighted value for tripod angle if f ($\omega_a$ )>0, $\omega_b$ = weighted value for hexapod angle if f ($\omega_b$)>0,

$\omega_c$ = weighted value for Tripod Stroke if f ($\omega_c$)>0, $\omega_d$ = weighted value for Hexapod Stroke if f ($\omega_d$)>0,

$\omega_e$ = weighted value for distance to the plate's centre and at least one data for each group, $G_n$ must be selected and d is the threshold value.

$$angG[o_1, o_2] = \cos^{-1}(\bar{a}.\bar{b})/(|a|.|b|) \qquad (20)$$

Where angG [$o_1$, $o_2$] = angle differences between various pair-wise comparison of planar sides, planar plane and line vector set n, $\sigma_n$ for any pair of $o_1$ and $o_2$. Weighted f, $\omega_f$ for $\sigma_n$ = angG [$o_1$, $o_2$] if Coplanar OR meets at infinity, given by the Grassmann rules. The large sets of comparison-wise check for non-coplanar condition has to consider non-redundant loop and Boolean validation method for all pairs $\sigma_n$. Boolean validation where $\sigma_0 \cap \sigma_1 \neq 0$ and $\sigma_n \in$ (comparable pair-wise set $\sigma_c$), and $\sigma_c$ must produce Grassmann errors. Figure 4-1 displays an example of Python rendering of a Grassmann coplanar indicator, which is based on the Grassmann rules.

Grassmann checking for error is based on the vector line that represents a bracket. This is a representation of angle, distance and collision between any vector line and bracket [57]. Referring to Figure 4-1, a bracket can be defined by the pencil line between 'w1v6v1w1'. The pencil line for this bracket is those that go across the plane created between 'w1v6w1' and ‘w1w6v6’. Pencil line ‘w1v1’ and ‘w6v6’ represent the vector line for the actuator. Pencil line ‘w1w6’ and ‘v1v6’ represent the vector line for the platform edge. There are various other combinations which have been designed for the probability ranking evaluation [54, 55]. This Grassmann check is structured in a few levels of detail. The level of detail performs different set of Grassmann pair-wise comparison check [60]. Each level will increase the checking complexity and the amount of members that is being compared and checked. When a region has been inspected for level 1(lowest probability level), and qualify as Grassmann error, the Boolean algebra will validate this for next level checks. As the checking become more complex and the result demonstrate higher probability for Grassmann to occur, then this pose shall be considered as higher probability Grassmann. Probability here is used to describe the condition factor for Grassmann errors to fully execute it. Grassmann error is a condition where the vector line for any pair of bracket suggests that an error may happens if the criteria are fully met. The Grassmann error pose is a platform’s pose that may produce an instantaneous rotation axis (IRA) which either add or remove a degree-of-freedom (DOF) from the system [61]. When the IRA angle is within an angle where an external force or the structural weight could collapse the system, then the Grassmann error is concluded as the highest rank. Grassmann error is a “guessing value” for the probability of IRA to form and produce the collapse condition [20].

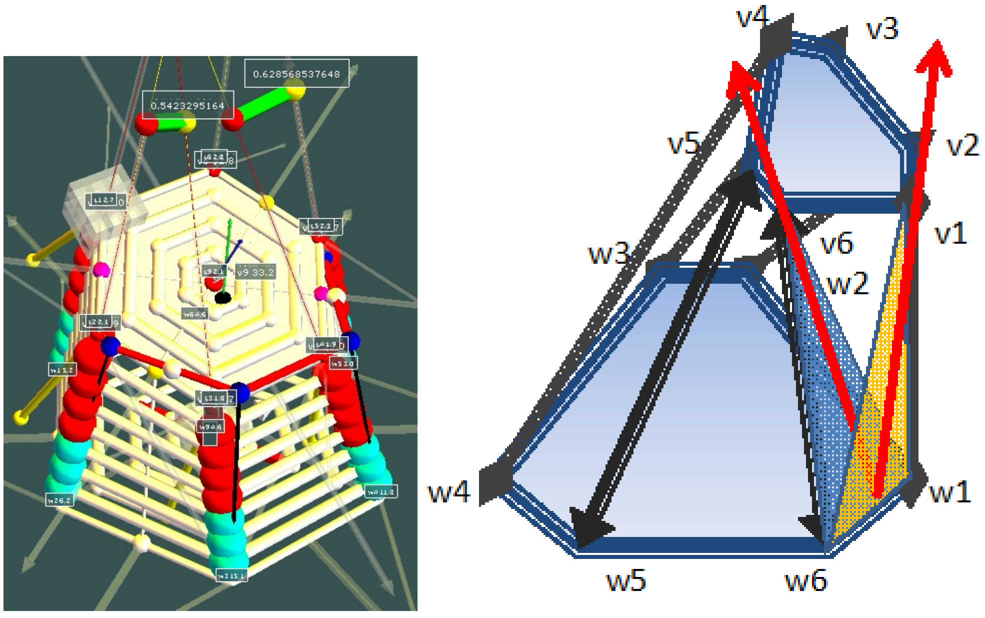


Figure 4-1 An example of Python rendering of the Grassmann coplanar indicator based on vector bracket

### 4.4 Results for Grassmann probability experiment

Figure 4-2 demonstrates an example Grassmann condition. The probability percentages for Grassmann to occur are recorded as between 27-31%. Grassmann condition may become detrimental to the structure if there is any external forces acting on the opposite side of the weak region or when the co-planar conditions allow for instant development of a new degree-of-freedom (DOF) which is also known as instantaneous rotational axis (IRA) condition. The Grassmann ranking or probability system examines and proposes the probability for a singularity error to occur at that particular pose. If certain condition is met, than the pose can become fully singular.

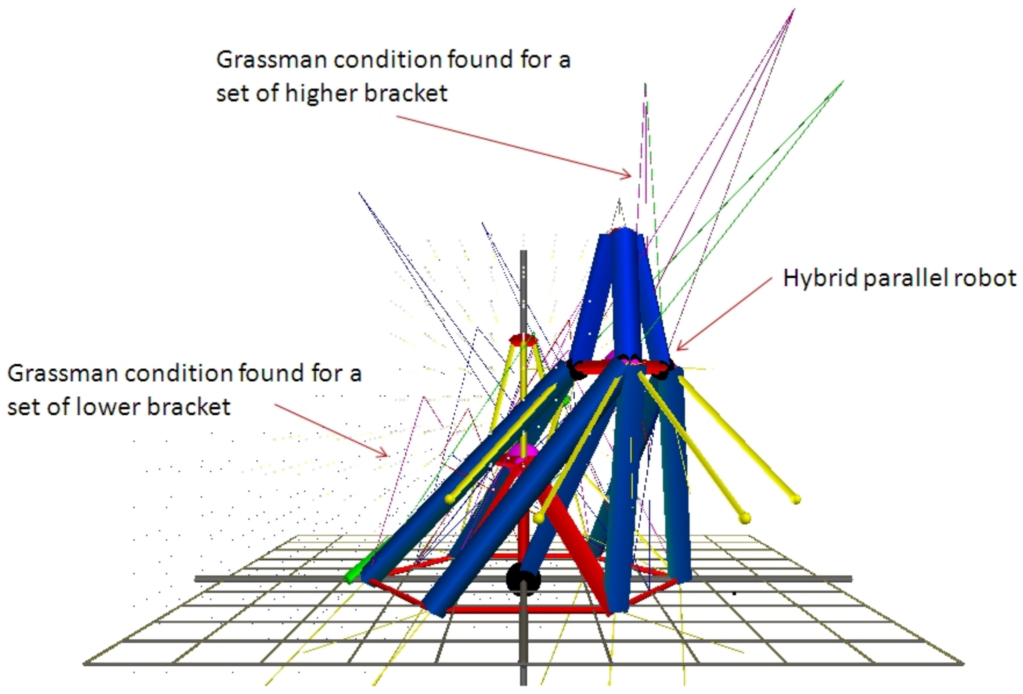

Grassman condition found for a set of higher bracket
Hybrid parallel robot
Grassman condition found for a set of lower bracket

Figure 4-2 Grassmann condition found within workspace region

## 4.5 Conclusion

Grassmann algebra or Grassmann-Cayley method is a bracket collection of vector sets. A comparison for angle differences between selected vector sets signal the probability for Grassmann singularity, which may result with system collapse, loss of stiffness, and the IRA issue (instantaneous rotation axis). A weighted ranking system provides visual information for user. Grassmann condition may lead to a problem when external forces, or the loads exceed certain limit, therefore the research propose a probability system rather than a confirmed error for a Grassmann singular pose.

# Chapter 5: PARAMETRIC SWEEP SEARCH

## 5.1 Introduction

This chapter discusses the parametric sweep search, which is a parallel robot's end-effectors path which checks for singularities. The search path for a parametric sweep is usually a geometric shape like cube, rectangle and spherical. This chapter explores and experiment with various data population strategy and rules for the sweeping motion. The Python simulation allows for any type of data population, control and validation. The previous chapter on Grassmann algebra solved the singularity errors found within an established workspace [52].

## 5.2 Introduction to parametric sweep

This chapter addresses the issue of the establishment of test positions for verification of non-singularity position for the end-effectors, where the result is shown as its workspace. This research is based on a successful development of a 3d-Python visualization system. The system can perform parametric search based on general test such as cubic and spherical sweep, and also advanced test such as strategic sweep with configurable dimension and direction in the form of controllable test planes. This plane is indeed a slice of information within the workspace.  We examine various types of L-System random generators for populating the plane with test positions. The 2D L-System is positioned and orientated to cover the assumed space covered by the workspace based on rotation around centre-point, parallel arrangement to form a cubic grid or interpolation done between various slices. Interpolation method creates relationship between random generated positions to produce 3-Dimensional parametric sweep positions [62]. Other 2D to 3D method studied for this research includes loft, extrusion and quaternion methods [63]. Other advanced methods mentioned in this research includes spiral 3D, Hilbert 3D and Marching Cube 3D [55, 56]. The workspace validation is done using weighted ranking based on Grassmann and is further extrapolated by using Simplex and Trilinear to verify the condition for ‘going into’ and’ leaving’ of a position condition. Finally, we showcase an early experimental result to demonstrate automated 3D fractal search for workspace methods. This 3D fractal system shall be further examined to develop a fine control of its trajectory, dimension and dynamic region.

## 5.3 Condition test theory

The condition test theory is based on identifying the geometrical constraint, then checking the quality of the non-singular pose by checking for Grassmann error and various workspace requirement errors like compliance and trajectory. The Parallel robot geometry definition is based on a circle divided into variable six point’s position following Charters method which allows for reconfigurable geometry automatic creation and easy numerical validation [55].

Grassmann error is defined for any given set of lattices that are formed by pencil line created using the structure’s line and edge. The limit is the threshold value for error state [66].

Based on the extensor definition for Grassmann-Cayley algebra (GCA) [49, 54, 55] which is given here as

$$P = V(u_1, u_2, \dots u_k) = u_1 V u_2 V \dots V u_k \tag{21}$$

Where the vector subspace U or $\bar{P}$ , in a 4-dimensional vector space V has the extensor of step 1,2 and 3 that correspond to points, lines and planes [53].

The Trilinear interpolation formula produces a series of reference plane which is shifted away from the test points in x, y and z-axis [69]. Then, the system performs 2D linear or polynomial interpolation, and merges or interpolates all results from all axes to get the 3D result. The grid test position is the data populated on a slice of plane, or a 3D grid system. The populated test position presented is an example of L-system fractal generated data.

$$pos[0] = \big((x-g),(y-g),(z-g)\big) \tag{22.a}$$

$$pos[1] = \big(x,(y-g),(z-g)\big) \tag{22.b}$$

$$pos[2] = \big((x-g),y,(z-g)\big) \tag{22.c}$$

$$pos[3] = \big((x-g),(y-g),z\big) \tag{22.d}$$

$$pos[4] = (x,(y-g),z) \tag{22.e}$$

$$pos[5] = \big((x-g),y,z\big) \tag{22.f}$$

$$pos[6] = \big(x,y,(z-g)\big) \tag{22.g}$$

$$pos[7] = (x,y,z) \tag{22.h}$$

$$G_T = \textstyle\sum_{pos=0,pos+1}^{7}(x,y,z) \tag{22.i}$$

$G_T$ represent Trilinear extrapolation around a coordinate (x, y, and z) [70]. Where the end-effectors position= (x, y, z)

$$G_{MC} = \sum_{pos=0,pos+1}^{8}(x,y,z,\alpha,\beta,\gamma) \in [MC_{1\ to\ 8}] \tag{23}$$

Where $G_{MC}$ represent Marching Cube extrapolation orientation check for 8 different configurations, MC $_{1 \text{ to } 8}$ and the end-effectors position and orientation is given by x, y, z, α, β and γ

The grid test position is presented by the data populated on a slice of plane, or a 3D grid system. The populated test position presented is an L-system fractal generated data.

$$G_{total\ wieghted} = \sum_{grid's\ test\ position\ for\ slice\ n}^{\max value} f(pos, limit, error) \wedge G_{GN} \vee G_T | G_{MC} \tag{24}$$

The $G_{\text{total weighted}}$ provide an overall evaluation for any test point or point on a parametric grid [51]. This is a general test to assist the algorithm for path planning and workspace definition. A detail inspection of each test point, collection of test points or path is available using slice and interval analysis. The next step is to define the parametric sweep.

## 5.4 Parametric Sweep theory

The parametric sweep activity is a step by step method to establish collections of test positions, where the end-effectors shall be directed to travel through these positions and tested for singularity. To optimize this operation, we need to consider the travel path, the test point's position effectiveness and method for controlling the search activity.

Typically, this grid is based on geometrical shape like cube, sphere and cone, which is then populated with test position to form a rigid grid. This section demonstrate a few method such as Hilbert as an efficient grid system, Marching cube for orientation check, extrapolation by using Simplex and Marching Cube, and Ulam Spiral and L-system fractal random generator with few arrangement and data population method [54-58]. Cohen develop a Voxel sweep method that uses a union-find data structure which keep the connection as one mesh, while Marching cube works like a Trilinear interpolation of the grid surrounding a point in space [75].

Boolean algebra control the path traversal for a rigid geometrical grid based on method A, for a cubic parametric sweep.

$$\text{Method A (Cubic)}, V_A = \left(\sum_{z=0}^{max} \sum_{y=0}^{max} \sum_{x=0}^{max} z + y + x\right) \qquad (25.a)$$

Boolean algebra typical method for various sweep method for a rigid geometric envelope is given below, in method B to method F.

$$\text{Method B (Prime Factor or LS (Serial sweep))}, V_B = \sum_{\text{i,i+1 for slice n(n+1)}}^{n}([f(ax^2 + bx + c) \wedge f(pos, limit, error)] = 1) \quad (25.b)$$

$$\text{Method C(Prime Factor or LS (Parallel sweep))}, V_C = \sum_{\text{i,slice i+1}}^{\text{element in slice i}}([f(ax^2 + bx + c) \wedge f(pos, limit, error)] = 1) \quad (25.c)$$

$$\text{Method D(Prime Factor or LS (Radial sweep))}, V_C = \sum_{\text{i in radial element}}^{\text{element in slice i}}([f(ax^2 + bx + c) \wedge f(pos, limit, error)] = 1) \quad (25.d)$$

$$\text{Method E (Prime Factor or LS (Optimized serial sweep))}, V_D = \sum_{\text{i,i+1 for slice n(n+1)}}^{n \wedge f(pos,limit,error)}([f(ax^2 + bx + c) \wedge G_T] = 1) \quad (25.e)$$

$$\text{Method F(Prime Factor or LS (Optimized parallel sweep), ))}, V_E = \sum\_(\text{i, slice i} + 1)\text{^}(\text{element in slice i} \wedge f(pos, limit, error)\ ) \in ([f(ax\text{^}2 + bx + c) \wedge G\_T\ ] = 1) \ (25.f)$$

There are variety of arrangement possible to improve the effectiveness of each fractal L-system, namely vase lathe method where a 2-D planes is rotated around an axis to form 3D mesh, extrusion (ternary numbers [66]) to form another dimension by using weighted ranking value and other attributes, interpolation between points placed on a given set of planes from any axis, and quaternion method.

Equations 25.f, 25.g, 25.h, 25.i and 25.j are based on quaternion, where $q = x + yi + zj + \omega k$ which is a four tuples for values x, y, z and $\boldsymbol{\omega}$[66]. For two quaternion numbers $q_1$ and $q_2$, then $q_1 + q_2 = (x_1 + x_2) + (y_1 + y_2) \times i + (z_1 + z_2) \times j + (\omega_1 + \omega_2) \times k$

There are various Quaternion or Quad algebra methods, and the published example formulation is given below.

$$Method\ G(Quaternion_{Jiang}), V_G = q_1.q_2 = (x_1x_2 + y_1y_2 + z_1z_2 + \omega_1\omega_2) + (x_1y_2 + y_1x_2 + z_1\omega_2 + \omega_1z_2) \times i + (x_1z_2 + z_1x_2 + y_1\omega_2 + \omega_1y_2) \times j + (x_1\omega_2 + \omega_1x_2 + y_1z_2 + z_1y_2) \times k \qquad (25.g)$$

$$q^2 = (x^2 + y^2 + z^2 + \omega^2) + 2 \times (xy + z\omega) + 2 \times (xz + y\omega) \times j + 2 \times (x\omega + yz) \times k \qquad (25.h)$$

Where $V_G$ = Jiang's Quaternion method to create the 3D grid system.

$$Method\ H(Quaternion_{Qu}), V_H = q_1.q_2 = (x_1x_2 - y_1\omega_2 - z_1z_2 + \omega_1y_2) + (x_1y_2 + y_1x_2 - z_1\omega_2 - \omega_1z_2) + (x_1z_2 + y_1y_2 + z_1x_2 - \omega_1\omega_2) + (x_1\omega_2 + y_1z_2 + z_1y_2 + \omega_1x_2)(25.i)$$

$$q^2 = (x^2 - 2y\omega^2 - z^2) + 2 \times (xy - z\omega) + 2 \times (xz + \omega^2) \times j + 2 \times (x\omega + yz) \times k(5.j)$$

$$Method\ H(Extrusion), V_H = z_{x,y} = c_{x,y} = xi + yj + zk(25.k)$$

Where $V_H$ refers to the extrusion method where a parameter like weighted ranking provide the extrusion z-value for any x and y – coordinate [76]. At x and y coordinate, we can derive z coordinate based on parameter c. The extrusion method is also known as ternary algebra (Eq. 25.1, 25.m, 25.n and 25.o).

This ternary algebra equation is following Cheng's definition [66].

$$t = x_i + y_j + z_k \qquad (25.l)$$

Where x, y and z are real numbers, while i, j and k are imaginary units. Based on two ternary numbers $t_1$ and $t_2$ [77],

$$t_1 + t_2 = (x_1 + x_2)i + (y_1 + y_2)j + (z_1 + z_2)k, \quad (25.m)$$

$$t_1.t_2 = (x_1x_2 - y_1z_2 - z_1y_2)i + (x_1y_2 + y_1x_2 - z_1z_2)j + (x_1z_2 + y_1y_2 + z_1x_2)k(25.n)$$

Get ternary number t

$$t^2 = (x^2 - 2yz)i + (2xy - z^2)j + (2xz + y^2)k \quad (25.o)$$

Method I (Random Generator) generates random 3D Mandelbrot test points [78].

$$xy = xz = yz = random \times pi2 \quad (25.p)$$

$$sxy = \sin(xy); sxz = \sin(xz); syz = \sin(yz) \quad (25.q)$$

$$cxy = \cos(xy)\,; cxz = \cos(xz)\,; cyz = \cos(yz)(25.r)$$

Where pi2= math.pi X 2

The 3D rotation around centre of the plane is given in Equation 25.s, 25.t and 25.u

$$x_0 = x \times cxy - y \times sxy; y = x \times sxy + y \times cxy \quad (25.s)$$

$$x_0 = x \times cxz - z \times sxz; z = x \times sxz + z \times cxz \quad (25.t)$$

$$y_0 = y \times cyz - z \times syz; z = y \times syz + z \times cyz \quad (25.u)$$

Method J (3D graph)

Now, that we have the strategy for placing, orientating and building the 3D shape, we need a way to populate the collection of planes or slices with test points. The end-effectors will travel through this test points to determine workspace, path etc. This is achieved by using 3D adjacency graph [79, 80, 81].

Method K (3D branching tree)

This method is based on the original finding by Verhoeff, where a comparison between ternary mitre joint and binary tree mitre joint is compared for randomness effects that copy the natural form of a real tree. Verhoeff continues with other variants by differentiating the cross-section profile, bevel angle, and reduction scale factor. The fractal dimension is based on $= \frac{\log N}{\log f}$, where N=parity, and f=scale-down factor. When D>3, the system may have self-intersection [82].

## 5.5 Parametric sweep modelling

The search for workspace, validation of path etc. are usually associated with the motion of the end-effectors or platform E and the secondary motion by platform A for a hybrid system. This motion design criteria is measurable, covers the required space, can be analysed, and is repeatable. This motion has been given a specific term known as parametric sweep modelling or discretization, which is a search for workspace following a grid structure shaped as primitive geometric objects such as square or hemisphere.

Standard parametric sweep for workspace search can be categorized into constant orientation, orientation workspace, constant position, maximal workspace, inclusive workspace and total orientation workspace. Constant orientation workspace is a condition where the platform E (end-effectors) and platform ‘A’ (middle travelling plate) is positioned towards a fixed orientation, and parametric sweep process the region to find the non-singular positions. Bonev demonstrate constant orientation strategy for finding singularity loci, where a platform is expected to have singularity with a fixed orientation and the orientation itself is a singularity, such as reaching a coplanar state [83]. Constant position workspace for finding singularity loci is a strategy associated with coplanar condition, where the platform can develop a stiffness problem. The singularity loci describe the workspace where this coplanar happens for any unit of test parameters. Each test parameters represent a collection of pencil line, edge and plane which is comparable to form the coplanar condition. Orientation workspace is defined as all possible orientation for any fixed position, as an offset from which a spherical shape is formed and the sphere aids user in defining the best path and motion gradient's trajectory. The orientation workspace is then compared with Marching cube method where similar strategy is implemented, but the orientation check is done using marching cube's orientation face. This method allows for building spatial relationship and neighbourhood analysis. The cubic parametric sweep system is comparable to Troyanov's growing algorithm formulation [84]. The seed distribution towards a region follows fractal random generator concept.

Maximal workspace or reachable workspace is described as all reachable position for end-effectors with one fixed orientation. The maximal workspace will benefit pose-to-pose planning where user can plan the gradient changes between two different set of maximal workspace to generate the optimum path based on adjacency graph. Inclusive workspace is a form of maximal workspace; however the orientations angles are set within a range.

The Python system can be used to solve all 6 types of workspace strategy by modifying the Boolean rule which has the attributes like envelope, edge, travel strategy, travel limit strategy, interpolation between different datasets and redundancy check. For example, in case 2, a constant orientation workspace generates few datasets, and the system generate optimum path between the datasets via an interpolation process.

## 5.6 Basic Sweep theory

Chablat suggested a method whereby, geometrical shapes such as a cube are used to envelop the robot with adequate edges for maximum stroke. The geometrical shape is subdivided into paths with nodes for recursive checks for, angle, stroke length, and collision errors of the robot's end-effecter [85]. This raw search method is also called 'parametric sweep' by Shah [62]. Standard parametric sweep for workspace search can be categorized into different variants depending on the search method, sweep method and result filtering. Breadth-first search is a systematic search, of brute-type, that perform exhaustive search of all space without first knowing the possible result. This search assumed the maximum possible plane or leg extension, then defines the outer edge border, and start searching through all search space within this border. This method is applied in basic search using the inverted cone search, cubic grid search space and the spherical / hemispherical search space. A minimum search space is defined by the Platform's A centre-to Base Plate's centreline only, while an optimized search is defined by all leg. Depth-first search with back-tracking is an exhaustive search of one region (shape), when found solution, or not, then increase the depth limit without going into infinite search. However, this method should continue the search from its last position when new depth is established. Binary search has two branch trees, where the first branch represents a valid region, while the other represents a non-valid (error) region within an established parametric sweep search region.  Binary value for left branch=0, and for the right branch=1. It is recommended to use a balanced tree, like AVL for the search strategy. A heuristic rule adds guides for where to search within a known region or an established workspace grid. This knowledge improves decision-making process about a specific search problem, suitable for path planning purposes. Best first search is an improvement to depth-first searches, by validating which node to search first, and keep the other node and do search on them when no solution is found. This could be a slice along any axis in the parametric sweep region.

The search method has to follow a strategy which is related to user requirement. This strategy aspect is discussed here as workspace type, which are general, constant orientation, singularity loci, constant position, orientation workspace and maximal workspace. Generally a search method is related to 2 dimensional searches of databases for example a tree search. A visual method like Voxel or spiral system provides 2D or 3D visual representation of a database, where the path between nodes is defined as the lattice patterns.

## 5.7 Advanced Parametric Sweep search based on L-system

This section discussed the various strategies for parametric sweep discretization or iteration method to find workspace for parallel robot. Parametric sweep is an iteration process, where the subject is arranged to move along a given path which is usually in a form of grid, and Hrishi extended this to include recursive nearest neighbour to improve the search performance [86]. Typical parametric sweep method is based on the establishment of a cubic, spherical or conical grid system with regular spacing between each unit, and this would require additional control to reduce redundancy and to limit the travel outside a useful range. Lara-Molina research on search optimization discuss about the issue of common method of performing local search based on gradients and Hessian which examine subsequent value and compare with the relative optima, will eventually converge into a local optima [87].

This study extends the concept of primitive shape envelope by introducing L-system fractal, Spiral, Hilbert 3D and Marching Cube 3D method. Marching Cube's Lorenson method which is a box triangulation method is optimized to produce a minimal pattern producing 8 different configurations check for the end-effectors [87]. Marching Cube is a method where position and orientation check is done effectively as an extrapolation strategy for any one point. Si develops variation to Hilbert using quadratic Hamiltonian to produce helix lattices by performing spin-coupling that rotate the angle to form the helix shape [88].

L-system is a formal grammar based plant growth system proposed by Lindenmayer. L-system is parallel system that accommodates interaction with the environment. Some variations to L-system are Stochastic grammars which gives probability for the occurrences, Context sensitive grammar which is a combination of context-free and sensitive checks for the occurrences before and after it and Parametric grammars allows for definition for dimension[88]. L-system fractal is a random generated fractal or natural grid system which is either 2D or 3D, and if in 2D forms, and then it would require a strategy for generating the 3D grid system. Cheng demonstrated few methods for fractal generator 3d transformation by using Jiang's quaternion, Qu's quaternion, Hamilton's quaternion and bi-complex numbers. Cheng concludes that ternary algebra produce better result compared to Quad algebra method. Ternary algebra method is described as providing more intuitive control of spacing, region definition and direction, and it is faster. Mandelbrot’s sets using Jiang’s quaternion a flat 3D visual and Mandelbrot’s sets using Qu’s quaternion produce a square sweep-lathe effect with noises. Mandelbrot’s sets using Hamilton quaternion produce a cylindrical sweep-lathe effect with some noises which could be the fractal tree branching out, while Mandelbrot’s sets using bi-complex numbers produce a clean cylindrical sweep-lathe visual.  The value of $\boldsymbol{\omega}$=0.022 give better shape for all experiments.  The method is also known as Ternary algebra [66, 90]. Other method includes sweep loft on all 3-axes, extrusion on any axes and interpolation between collections of strategically placed planes. Rosa describe the quaternion method for transforming Julia fractals into 3D system [90]. Séquin has extended the 3D Hamiltonian grid concept into 2D manifold, which enable the 3D system to be presented as 2D database which can be integrated with common 2D methods for database search, path planner and data optimization [91].

Fractal has this attributes of having similar basic shape and recursive definition which is a form of grid, with an added advantage of being non-rigid in its shape. The advantage is for the cost of finding a solution, where L-system methods allow for fractal neighbourhood growth with directional and dimensional factors. A geometrical parametric sweep would be costly and difficult to determine the correct dimension size for each expansion or branching out as the system grows. An L-system method is looking at the problem from a graph point of view which is practical and efficient when dealing with sweep approach.

Interpolation for additional test of any single point is done using Trilinear 3D and Simplex 3D. The weighted ranking for any non-singular position is given by Grassmann factor, which checks for Grassmann-related errors like instantaneous rotational axis, plane or edge’s vector meets at infinity or any pair of edge or plane’s vectors is coplanar.

Ulam Spiral is a prime factor spiral generator, and its polar pattern is useful for the parametric sweep activity. There are few variants to this prime factor spiral like Vogel spiral, Fibonacci sums and direct rasterization. Ulam Spiral and its variant is generated and strategically positioned to compare with the other various parametric sweep methods.

## 5.8 Boolean control for Parametric Sweep search

Boolean Logic concept is based on the input from leg stroke limits, joint angles and ranking value that feed into a deductive logic system to find singular and non-singular positions. Vector operations like direction, cross, dot product and closest distance provide input for checking a pose for singularity. The system checks for performance comparison between AND, NAND and NOR logics for mixed logic condition. The condition helps define the parametric search parameters and is specific to particular applications [92].

Boolean optimization or simplification has the capability of removing redundancies and simplifying combinational circuits. However, this method assumes that any point away from the central points that is not connected to the existing network is singular for that particular condition. The search for special workspace on a surface mesh is based on minimum distance, and checks for duplicate results between the robot workspace and the mesh coordinates. Interval analysis algorithm adjusts the parameter for the search range and resolution, and improves performance speed [93]. Slicing operation allows the study of any slices which would improve system performance. For example, a slice analysis of a selected subset of constant orientation workspace helps to understand the effect of orientation as a predefined path that passes along these slices [94]. A Boolean algorithm provides the search strategy with controls for search behaviour, search limit, search direction and redundant search reduction. Jeroen developed a Boolean simplification system by deploying graph system, where the graph relationship greatly reduces the requirement for additional description to some elements, due to Boolean effective terms [93]. Boolean system is flexible and universal in addressing many algorithm functions integration with other methods. Boolean make use of simple notation in providing iteration and flow control for much strategy used in the Python simulation.

Boolean algebra has been used extensively and successfully in controlling the system behaviour. A low resolution and small region investigation can be performed in real-time, while a higher resolution and larger region is still a lot faster than numerical system.

## 5.9 Test point population theory

The parametric sweep next strategy is to populate the search region with test points and direct the end-effectors to cycle through the test points, where search validation is done on each point. A typical method is to populate in a rigid grid manners such as rectangle or circular grid. This method is essential in estimating the maximum search region, but it is not economical to perform high resolution search based on this method.

Table 1 3D grid system comparison (Low Resolution)

| Type | Test 1: Found workspace / Test Points | Average Time 1 |
|---|---|---|
| Cubic 3D | 323 / 3179 (10%) | 1.1e+03s |
| Cubic 3D (at large stroke, more than 20cm.) | 22 / 1156 (2%) | 3e+02s |
| Spherical 3D | 819 / 1200 (68%) | 9.2e+02s |
| Hilbert 3D | 267/ 2000(13%) | 9.7e+02s |
| Marching cube 3D (Cubic Grid) | 1196/ 1440 (12 %) | 5.7e+03s |
| Marching cube 3D (Hilbert Grid) | 748 / 2178 (34%) | 2.9e+03s |

**Table 1** shows that parametric sweep type cubic and Hilbert 3D (which is a form of L-system) is less effective compared to spherical method. The Hilbert 3D grid formation is harder to control in terms of direction and edge limit, therefore its effectiveness is low. The cubic sweep has similar problem and not effective especially at the large stroke (where its efficiency is only 2%). Marching Cube is an orientation check which is repeated 7 times at each test point; therefore the efficiency is far less than the found valid points. Hilbert 3D combine with Marching Cube yield an efficiency of 34% compared to cubic efficiency which is only 12%.The next step is to find alternative ways for populating the assumed workspace region, by looking into fractal random generator system.

## 5.10 Test point population based on fractal theory

This section intends to explore other ways of populating the grid or region effectively, while considering the region growth, direction, multiple scales and resolution and its effectiveness. Various L-system fractals are tested, and some is presented here to demonstrate the result. There exist various types of L-system fractal system, each with its own advantage, practical uses and advantages. Integrating the L-system in this workspace analysis is rather strange if compared to the regular standard dimension and rigid grid usually found in parallel robot workspace search based on cubic, spherical and conical. Spiral system with search range and sub-search using L-system tree can benefit from the search scope. All L-system random generators help interpolate and extrapolate between known workspace positions. Random fractal generator based on various L-system algorithm produce a variety of 2D shape. However, it is difficult to direct the region growth, the region shape and sub-region resolution.

Table 2 L-system 2D to 3D Hamilton's Quaternion comparison

| Type 1D planar | Test 1: Found workspace / Test Points (Percentage %) | Test 2: Found workspace / Test Points (Percentage %) | Average Time |
|---|---|---|---|
| Snowflake | 305/6096 (5%) | 498/6096 (8%) | 7.4e+02s |
| Hexaflake | 294/8192 (3.5 %) | 362/8192 (4.4%) | 9e+02s |
| Spiral fractal | 459/6144 (7.4%) | 428 /6144 (7%) | 8.8e+02s |
| Vicsek fractal | 436/6145 (7%) | 485/6145 (8%) | 8.6e+02s |

**Table 1** show that the 2D fractal parametric search is less efficient by 10% compared to other methods. Comparing Table **1** and Table 2, it shows that fractal and cubic is far less efficient compared to spherical parametric sweep. Hilbert grid has no redundant visit for any test point, however due to lack of control for direction and growth, it become less suitable for workspace search task.

This experiment demonstrate that the L-system as a random generator can be implemented as 2D or 3D parametric grid, however it is difficult to control its direction, interval analysis aspect like variable resolution, region definition and scale, and the growth of this grid when the search grow. Spiral system like Ulam spiral proves to be more controllable and provide various opportunities for improving the cubic and spherical parametric sweep method. The next step is to define the test plane or slice analysis planes. The test point is populated and arranged based on the parametric sweep method.

## 5.11 Parametric Sweep methodology

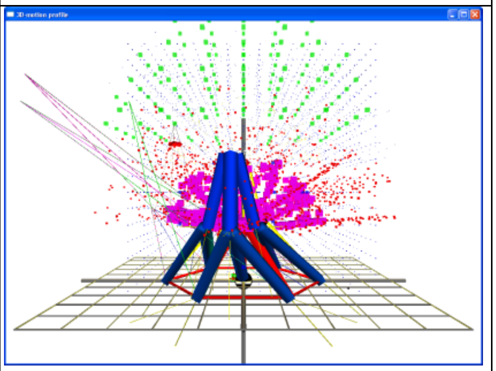
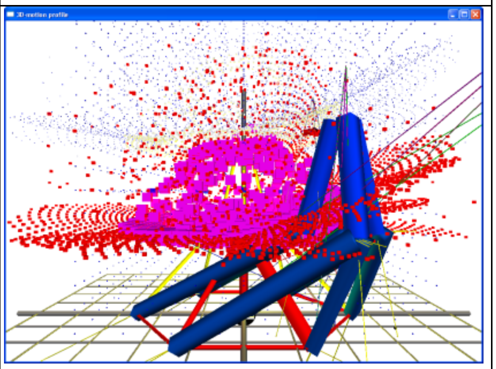
This section discusses the strategy for defining the test plane or the slice analysis plane. This shall define how the 2D or 3D test points are effectively placed for variety of task, such as search for workspace, path planning and optimization.

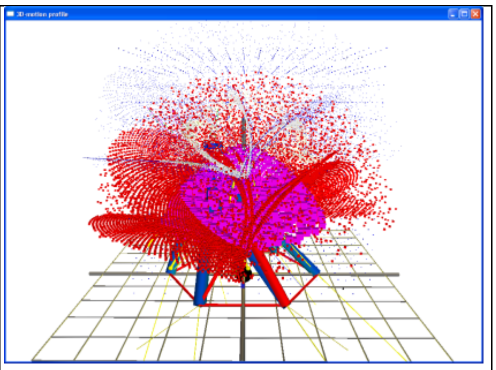
## 5.12 Parametric sweep random fractal generator

This section shall focus on Mandelbrot 3D shape, which is being randomly generated. Basic optimization including maximum distance from fixed origin point, and points below the fixed bottom plate is removed from the search set.

Table 3 demonstrates the special attributes for using fractal as test point's generator, which produce random data.

Table 3 Mandelbrot 3D random generated test points.

| Test | Test 1: Found workspace / Test Points /Average Time | Image |
|---|---|---|
| Low- resolution 1 | 419 / 2973( 14%) 8.5e+02s |  |
| Medium – resolution 2 | 1765 / 11708 (15 %) 3.7e+03s |  |

| | | |
|---|---|---|
| High – resolution 3 | No data |  |

## 5.13 Parametric sweep type Hilbert 3D

Parametric sweep Hilbert 3d is a cubic-based grid with the exceptions that no repeat visit is allowed during the iteration process. Parametric sweep should produce efficient grid, and a grid that can be structured as 2D-databse data system. Hilbert is a form of a quad tree. 3d space-filling curve quad tree system forms an advanced visual search concept. The system utilizes Hilbert curve to optimize the search strategy. 3-bit Gray - 3d Hilbert curve Order is presented below (Refer Figure 5-1)[95].

$\begin{matrix} [0,0,0] & [0,0,1] & [0,1,1] & [0,1,0] \\ [1,1,0] & [1,1,1] & [1,0,1] & [1,0,0] \end{matrix}$, Developing a control for Hilbert, by introducing learning mode where the change of direction is now based on weighted value. However, due to the characteristic of Hilbert angle flips that should remain the same to maintain the overall shape, the control is only limited to distance away from origin. Si has proved that Hilbert can be transformed into a helix shape which is an advantage in defining the workspace compared to a cubic-pattern lattice [96]. Helix and cubic Hilbert 3D has great potential, however due to the complexity of developing this into a manifold system, this part of the research is planned for future works. For this research, special focus is given on

Hamiltonian path, which works with similar rule that allows a single visit per node. Hamiltonian path can be developed using various L-system fractal, and extrusion strategy. This shall be discussed in the next section.

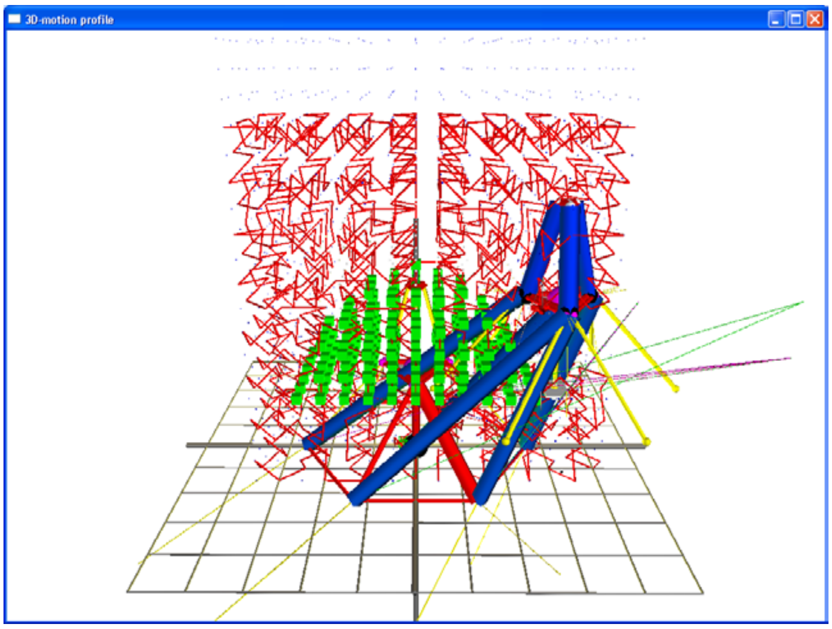


Figure 5-1 Parametric sweep type Hilbert's 3D.

## 5.14 Parametric sweep grid extrapolation

Parametric sweep's grid extrapolation means the strategy for improving the grid quality, by effectively introduce additional points close to the found workspace points. The extrapolation method experimented here includes Trilinear and Simplex. Simplex, Marching cube, and the L-system Fractal 2d-to-3D loft generator need Euler rotation matrices to correctly positioned them in the manner that user specified.

## 5.15 Parametric sweep Euler-convention

Parametric sweep's strategic grid involves the act of defining the plane, position the plane and distribute random test point on the plane, and interpolate test points between planes to form a volumetric test points. Bonev mentions that Euler rotation matrices can be in any of this 12 different conventions [83] . Where p is a vector coordinate in fixed frame and p' is the same vector in the rotated body frame based on orthogonal rotation matrix R, and p=Rp'. (XYZ, XZY, YXZ, YZX, ZXY, ZYX, XYX, XZX, YXY, YZY, ZXZ, and ZYZ.) Therefore, variety of result is obtainable when different conventions are used. This is an important aspect when building 3D quadratic fractal.

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \sin\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \tag{26.a}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{26.b}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{26.c}$$

$$R = R_z(\omega)R_y(\theta)R_x(\varphi) =$$

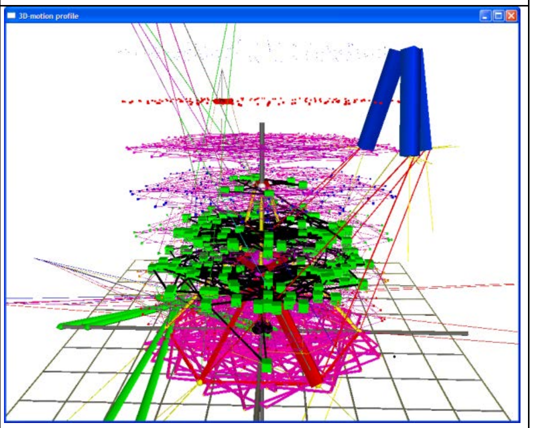
$$\begin{bmatrix} \cos\theta\cos\omega & \sin\varphi\sin\theta\cos\omega - \cos\varphi\sin\omega & \sin\varphi\sin\omega + \cos\varphi\sin\theta\cos\omega \\ \cos\theta\sin\omega & \cos\varphi\cos\omega + \sin\varphi\sin\theta\sin\omega & \cos\varphi\sin\theta\sin\omega - \sin\theta\cos\omega \\ -\sin\theta & \sin\varphi\cos\theta & \cos\varphi\cos\theta \end{bmatrix} \tag{26.d}$$

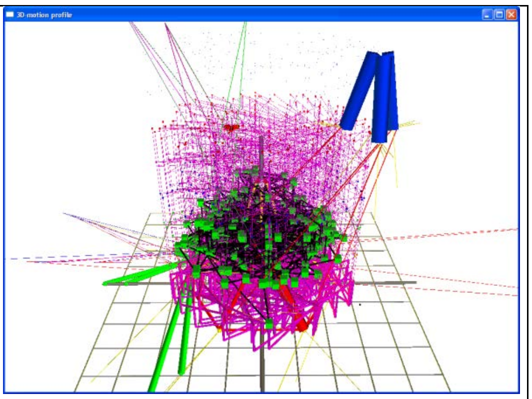
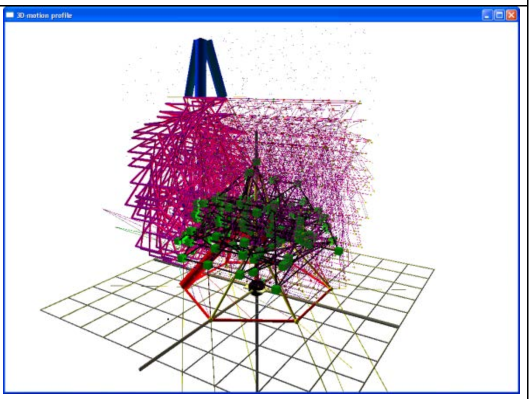
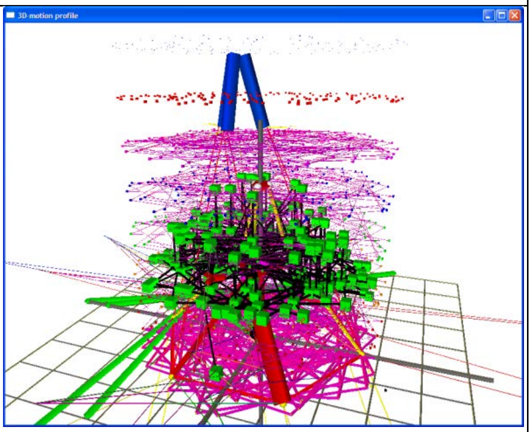
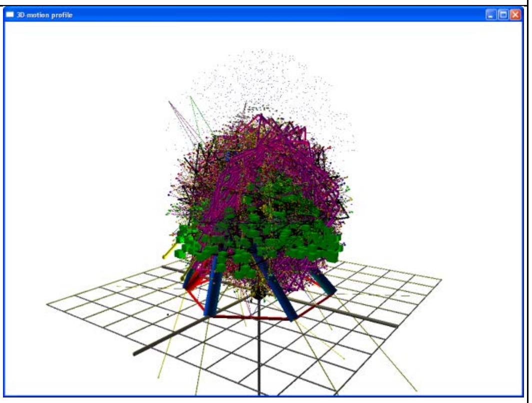
## 5.16 Parametric sweep spiral grid

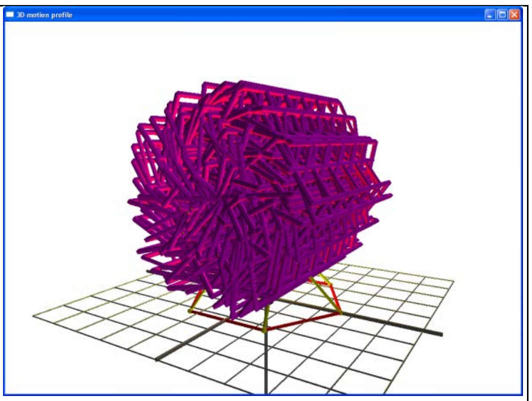
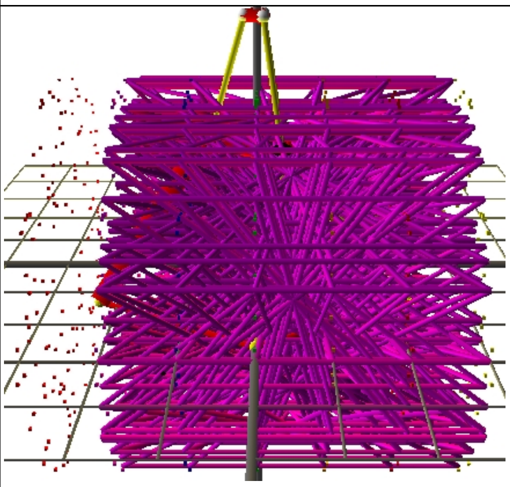
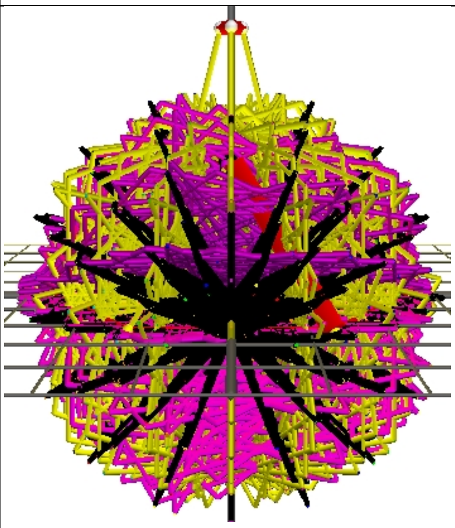
Parametric sweep's spiral grid method includes Helix and Spiral format. The fractal system examined in this section is the Alum Spiral, following the better performance provided by helix based search. Different arrangement strategy for L-system type Snowflake to form 3D grid based on various strategy is presented in Table 4. The test is based on Ulam spiral grid with various strategies to explore its potential.

The strategy for optimization is based on radius shrinkage and expansion for each slice. Therefore along the slices, the Ulam spiral search grid's radius will try to fit into smallest searchable radius while considering all possible test points outside the range. Herschel graph is the smallest possible polyhedral that does not have a Hamiltonian cycle.

Table 4 3D grid formation strategy based on 2D plane

| Arrangement type | Test 1 : Found workspace / Points (Average Time) (Percentage %) | Image |
|---|---|---|
| Serial (Horizontal) | 175 / 834 (6e+02s) |  |

| | | |
|---|---|---|
| Serial (Vertical) | No data |  |
| Parallel (Horizontal) | 175 / 828 (6e+02s) |  |
| Parallel (Vertical) | No data |  |
| Radial Grid | No data |  |

| | | |
|---|---|---|
| (B-Complex) Extrusion (Horizontal) | 309 / 834 (7.8e+02s) |  |
| (B-Complex) Extrusion (Vertical) | 349 / 834 (8.1e+02s) |  |
| Hamilton's quaternion (24 slices) | No data (2.4e+03s) |  |

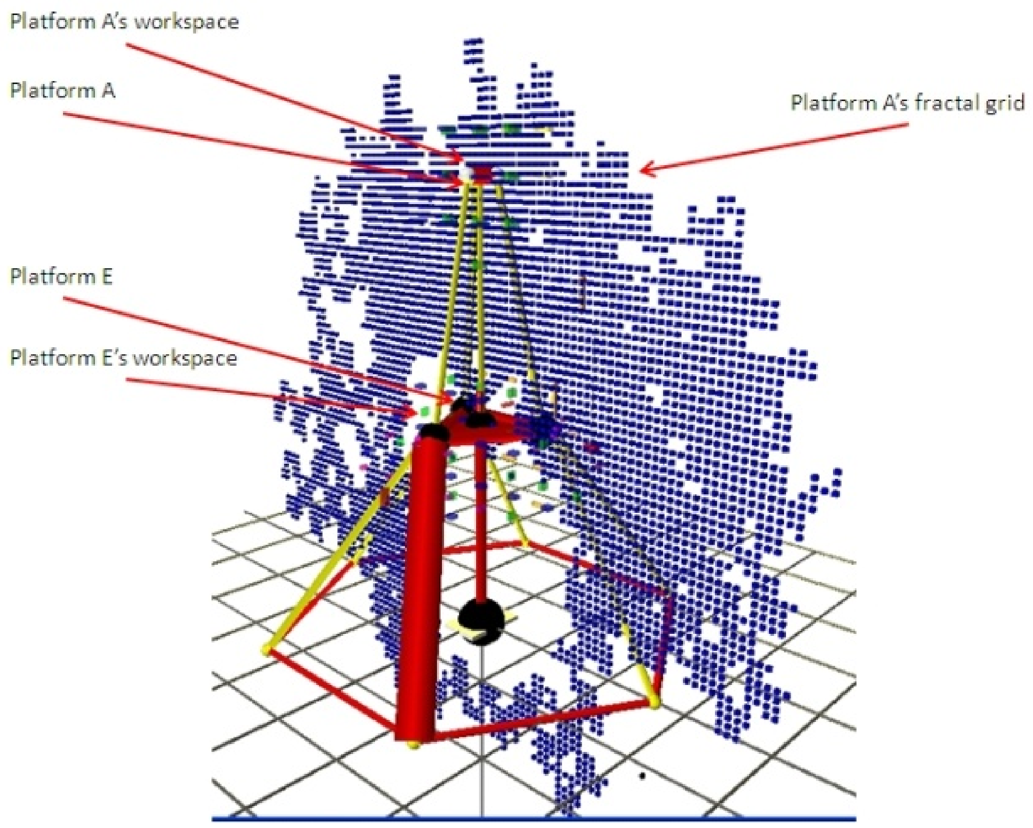
## 5.17 Parametric sweep 2D based on turtle-cursor method

Turtle-cursor method is also known as Lindenmayer system which is a recursive systems that leads to self-similarity [88]. A parametric L-system is defined as $G = (V, \omega, P)$, where V (the alphabet) is a set of variable of symbols, ω = (start, axiom) which is a string of symbols from V defining the initial condition, and P is a set of production rules which define the method for replacing the initial conditions with combinations of constants and other variables. A context-free L-system's production rule does not consider its neighbors. A context-sensitive system considers its neighbors. A deterministic context-free L-system which has only one production-rule for each symbol is called a DOL-system, while if each variations of production rule are chosen due to probability per iteration; this is called stochastic L-system [97].

$$Cursor_{pos} = \sum_{c=0,ws(c+1)}^{ws(max)} (Fractal(type)) \wedge \left((ws > limit) \rightarrow Cursor_{pos=}ws(c)\right) \quad (27)$$

Where Cursor$_{pos}$= Fractal generator's cursor position, ws=workspace pre-generated based on cubic parametric sweep Fractal (type) is either type Levy Dragon, Koch Snowflake, Levy C and Hilbert. Fi 5gure-2 demonstrate a planar slice fractal grid based on this parameter 'FX', 'X', 'X-YF-', 'Y', '-YF-X', where F = Forward motion, + = turn Right, - = turn Left and X,Y represent the 2D axis of X and Y.

Example production rules used in the Python simulation is given here, for example to draw a Levy Dragon pattern, the rule is = (1, 4, 16, 'FX', 'X', 'X+YF+', 'Y', '-FX-Y'). To draw a Koch snowflake, the production rule is = (1, 6, 6, 'F++F++F', 'F', 'F-F++F-F', '', '').



Fi 5gure-2 Example Python rendering of a 2D-fractal pattern

The running time for 2D fractal growth search is n sec, and the 3D mode is still being developed to remove redundant or repeat visit of any test point, and to improve the 3-dimensional growth or branching out algorithm. Extending this concept by performing slicing at n-interval along an axis produces a 3D representation of the fractal 2D growth. The relationship between populated test points can be regarded as adjacency graph elements.

## 5.18 Parametric sweep grid strategic distribution

The strategic distribution has to be formula-based, configurable and can be made random. This way, the operation can cater for different task, and adjust its growth and region based on the task requirement. Usually, the formulation produces high density small-region distribution of test points within a certain organic shape as its outline.

White and Nylander's formula for the "nth power" of the 3D vector to build a 3D fractal is based on Hamilton quaternion of complex numbers [98]. White method for Mandelbrot 3D is defined in Eq. 28a And NY lander method for Mandelbulb 3D is shown in Eq. 28b

$$(x, y, z)^n = \rho^2(\cos(2\theta)\cos(2\varphi), \sin(2\varphi)\cos(2\varphi), -\sin(2\varphi)) \quad (28.a)$$

Where $\rho = \sqrt{(x^2 + y^2 + z^2)},\ \theta = arctan(y/x), \varphi = arcsin(z/\rho)$

$$(x, y, z)^n = \rho^n(\cos(n\theta)\cos(n\varphi), \sin(n\varphi)\cos(n\varphi), -\sin(n\varphi)) \quad (28.b)$$

Where $(\rho, \varphi, \theta)^n = (\rho^n, n\varphi, n\theta)$

## 5.19 Parametric sweep grid based large quaternion

Polyhedra grid formulation for n=-3, -2, -1, 0, 1, 2 and 3 is given in Eq.10. Polyhedra grid based on either rhombic triacontahedron (RT) or rhombic dodecahedron (RD) [99]. Figure 5-3 demonstrate a range between -2 to 2 for n, which can be further extended to build large branches.

$$(x, y, z)^{-2} = \left(a \times (x^2 - y^2), (-2 \times x \times y \times a), \left(-2z \times r_{xy}\right)\right)/r^2 \quad (29.a)$$

$$(x, y, z)^{-1} = \big((x), (-y), (-z)\big)/r^2 \quad (29.b)$$

$$(x, y, z)^{0} = \big((1), (0), (0)\big) \quad (29.c)$$

$$(x, y, z)^{1} = \big((x), (y), (z)\big) \quad (29.d)$$

$$(x, y, z)^{2} = \Big(a \times (x^2 - y^2), (2 \times x \times y \times a), \big(2 \times z \times r_{xy}\big)\Big) \quad (29.e)$$

Where$a = 1 - z^2/r_{xy}^2$

Where$a = 1 - 3 \times z^2/r_{xy}^2$, after which we must use Monte Carlo to solve the rendering equation.

This is formulated using the pseudo-code in Eq.30 to demonstrate an example for large quaternion. Example Python rendering of the large quaternion based on Eq.30 is given in Figure **5-4**

$$\mathrm{a} = 1 + (\mathrm{z}^8 - 28 \times \mathrm{z}^6 \times \mathrm{rxy}^2 + 70 \times \mathrm{z}^4 \times \mathrm{rxy}^4 - 28 \times \mathrm{z}^2 \times \mathrm{rxy}^6) \quad (30.a)$$

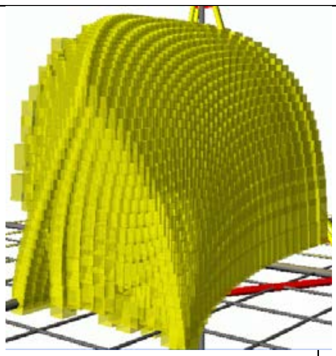
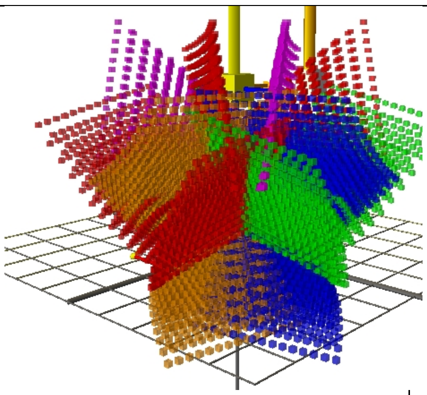
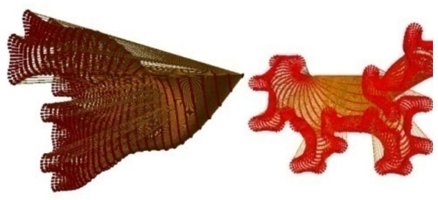
$$dx = a \times (x^8 - 28 \times x^6 \times y^2 + 70 \times x^4 \times y^4 \times -28 \times x^2 \times y^6) \quad (30.b)$$

$$dy = -8 \times a \times x \times y \times (x^6 - 7 \times x^4 \times y^2 + 7 \times x^2 \times y^4 - y^6) \quad (30.c)$$

$$dz = 8 \times z \times rxy \times (z^2 - rxy^2) \times (z^4 - 6 \times x^2 \times rxy^2 + rxy^4) \text{ , where rxy } = \sqrt{(\mathrm{x} * \mathrm{x} + \mathrm{y} * \mathrm{y})} \quad (30.d)$$

Z-axis in 3d or 4d fractal can be defined as level definition, which produces branching out. The outcome produces an organic shape that follows a given direction, with a tendency to branch out to test for out of scope’s data distribution. The system appears flatten and all axis will eventually force test points’ distribution towards the given direction. Example 4D rendering is shown in Figure 5-5

Table 5 Comparison between Quaternion sweep systems

| | | |
|---|---|---|
|  Figure 5-3 Quadratic 3D Mandelbulb Set |  Figure 5-4 Large quaternion |  Figure 5-5 4D fractal grid |

## 5.20 Conclusion

Various sweep theories are presented in this chapter, with a basic geometrical square shape and complex methods like L-system fractal generator and quadric algebra. The sweep theory integrated in the 3d Python simulation manages to solve various types of workspace search methods. The sweep method presented is based on geometric constraint and Grassmann check, and has been validated for kinematic accuracy. The Python simulation is able to solve many workspace types which include constant orientation workspace, constant position workspace and the blending between collections of workspace datasets.

Fractal distance away from the start point can be estimated, and the spawn direction can be controlled, however, in a 3D environment, this proves to be complex and computational intensive process. While propagating and distributing test points in 3D space, the fractal set has the tendency of distributing away from the learning data. Therefore, redundant test has to be controlled which require an additional process. Generally escape time algorithm is utilized to control the iteration process; however, it is common to see large amount of non-useful iteration has been performed before the process catch the exception.

The developed research demonstrates the advancement in parametric sweep search which is useful in determining workspace, analysis of workspace, path planning, singularity determination and interaction with external elements like contact surface and obstacle. The traditional geometric sweep based on cubic and spherical can solve the search problem, and some has been examined for various workspace typologies. Then, the research advances into L-system fractals to examine the possibility of using fractal random generator for similar purposes. This is further improved to become 3D and 4D fractal system based on various interpolation method like extrusion, axial rotation and algebraic. The development of full 3D random fractal generator is based on Verhoeff works, where the concept is related to parallel robot.

The modified pattern for L-System has successfully identified the workspace region based on a robust training data and produce adjacency graph. The adjacency graph provides the opportunity for various relationship establishments between the test data. This extrusion, quaternion and sweeping method to build a 3D data population can be easily optimized, prepare for path planning and surface mesh interaction.

# Chapter 6: BOOLEAN CONTROL FOR PARAMETRIC SWEEP SEARCH

## 6.1 Introduction

This chapter discusses Boolean algebra and its application in solving parallel robot's design problem. The Boolean algebra provides control for the parametric sweep strategy mentioned in the previous chapter. This control strategy is necessary since the parametric sweep search method is lacking in control. Without control, the sweep search have to assume a larger than necessary search region, and has to sweep through all region.

## 6.2 The Boolean algebra for fractal parametric sweep

Learning from Alum spiral path it can be seen that the spiral effectively increases the search area without having to search outside the workspace region in each search cycle. Boolean gate is required here to control the search expansion, and instruct the L-system fractal generator to explore the space. The experiment is performed on a slice of data, where y-axis is fixed at position=0. Boolean algorithm shall decide either to move the cursor forward, to the left or to the right and render a test point at that position. Upon, which the weighted ranking for distance to a known workspace position is used to evaluate this directional changes. The random generated fractal is then directed towards this favourable direction. Growing a fractal cluster while removing the chaotic factor needs Monte Carlo method [100]. Topology straightening formula restrict the boundary for the fractal growth and branching [101]. Key objective here is to straighten and direct the fractal growth axis towards a given trajectory. And, to limit the growth within a given boundary that satisfy the task's objective. Examples Python rendering are given in Figure 6-1, Figure 6-2 and Figure 6-3

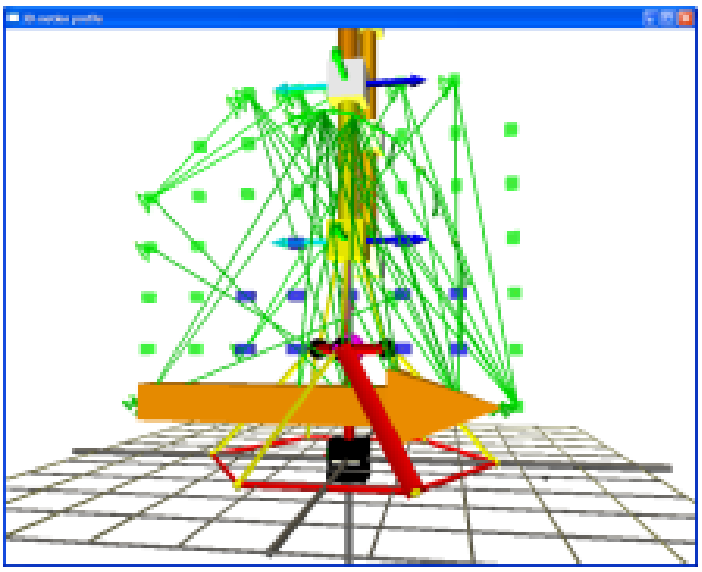


Figure 6-1 Example of 3D-fractal parametric sweep development history

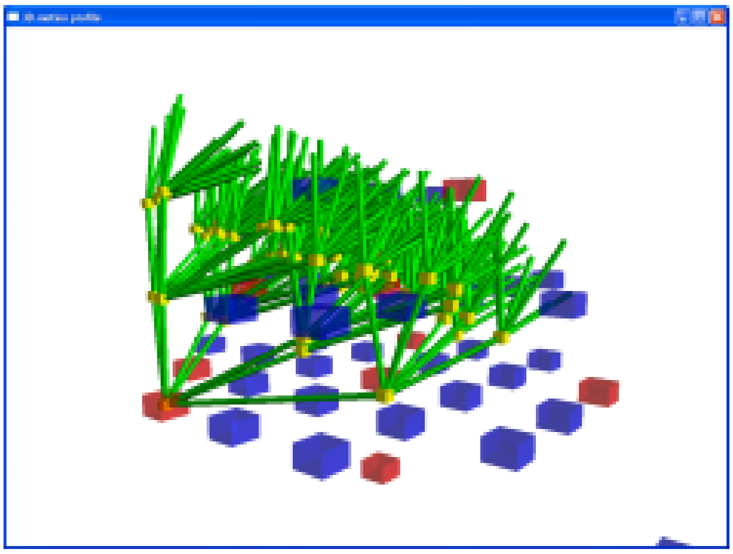


Figure 6-2 Example of 3D 3D-fractal parametric sweep development history

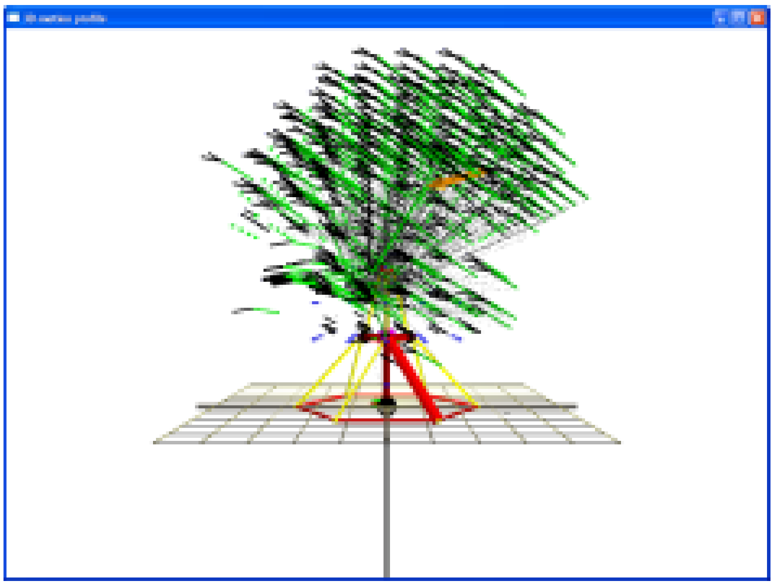


Figure 6-3 Example of 3D 3D-fractal parametric sweep development history

Verhoeff successfully produced an easy to reproduce and quite natural-like 3D tree-structure based on simple rule. Firstly, Verhoeff start with a rectangular polygon as the seed. Then, generate the first pair of branch (left and right direction, where angle is between $60^O$ to $120^O$), by performing a cut (using functions like square, parallelogram, rhombus and rectangle to produce the basic seed shape), ridge (using functions like horizontal and slanted to perform the ridge cutting through the seed volume, usually at the top end), and roof (using functions like asymmetric, symmetric and congruent for the cutting action itself to produce random effects). The 3D binary tree development shall be based on Verhoeff findings and experimental results [82].

In this section, a methodology for real time identification of various singularities for various types of workspace for parallel robots is proposed. Python 3D simulation software has been developed to position the moving platform of the robot's CAD model through pre-defined rules in order to solve specific problems including singularity identification, obstacle avoidance, and path planning. The system is designed to identify the moving platform’s best possible pose. Boolean logic is used to identify valid path trajectory through parametric sweep search method. Joint constraints are checked to validate the platforms’ positions using the actuators’ stroke length, their angles, and any possible collisions. Solutions for the desired pose are then obtained, based on line collision and mesh model algorithms,. The path, position and workspace data are verified against a kinematic model of the robot, developed in Solid works and Matlab software tools. The Python system offers fast solution in designing new parallel robot configurations and geometries through local and global optimization of the search area

## 6.3 Boolean Logic for search control and validation strategy

Boolean method is based on the logic controls of the search and validation strategy to determine singular and non-singular position and orientation workspace. Boolean logic is the control structure that defines the search parameters; search criteria and analysis. Search parameters include upper and lower bound and parametric sweep shape. Search criteria include search technique and filter.

Table 6 is based on ‘case 1’ to demonstrate the Python system capabilities, where the end-effectors are directed to follow a path, while the system performs workspace analysis for the Platform A to determine its best path. The process completion time is resolution-dependent, and interval and slicing allows detail analysis of interesting region. The computation time shown here is faster than numerical method for finding a workspace. Case 1, 2, 3 and 4 is not replicated in numerical method due to few constraints.

Table 6 Parametric sweep performance for Case 1

| Type | Travel limit (x, y axis) | Resolution | Time (seconds) |
|---|---|---|---|
| Python Test 1 | 40 cm | 1 unit | 34000 |
| Python Test 2 | 40 cm | 5 unit | 1100 |
| Python Test 3 | 40 cm + z – axis(2 slices) | 5 unit | 2000 |
| Python Test 3 | 40 cm (x, y & z – axis) | 5 unit | 17000 |
| Numerical Test | Not implemented | Not implemented | Not implemented |

Analysis is performed on the dataset based on a general structure, constrained structure (by obstacle type and shape), interval analysis and data slicing analysis. The case studies presented here is based on literature search results for various typical and specific workspaces. Reference is made to Figure 3-1in the following descriptions: Case 1 is a special-case for a hybrid robot system, where the search is for platform E's workspace when platform A is following a pre-defined 3D path. Case 1 optimizes platform E's travel while ensuring minimum trajectory changes during travel.  Case 2 is a search for obstacle-free workspace when a set of primitive objects are placed within a known workspace. This obstacle is user-defined, where the dimension, position and accuracy can be configurable and can be used for obstacle-free path planning. Case 3 is an end effectors' search for position on a 3D mesh surface. Interval analysis can add a higher resolution search on a specific path on the 3D mesh surface. Case 4 are optimized search strategies, where Boolean analysis is performed on selected information only, which is either a slice or a set.

## 6.4 Case studies

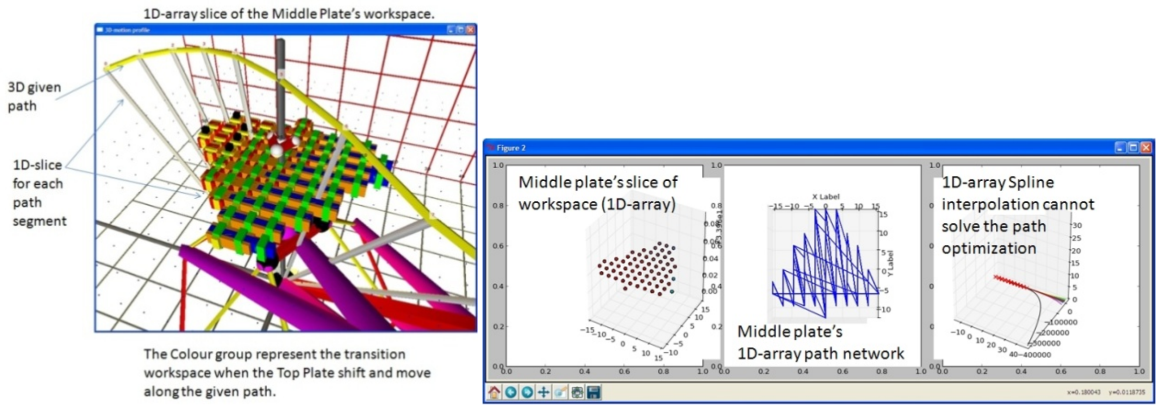
### 6.4.1 Case 1: Boolean search for platform A's path when platform E's path is known.

Weighted f, $\omega_f$ for $\sigma_n$ = angG [$o_1$, $o_2$] if Coplanar OR meets at infinity, given by the Grassmann rules. The large sets of comparison-wise check for non-coplanar condition has to consider non-redundant loop and Boolean validation method for all pairs $\sigma_n$. Boolean validation where $\sigma_0 \cap \sigma_1 \neq 0$ and $\sigma_n \in$ (comparable pair-wise set $\sigma_c$), and $\sigma_c$ must produce Grassmann errors.

Table 7 A random-generated 3D grids for case 2

| Random generator Types | Time (seconds) | Grassmann weighted value $\omega_f$ |
|---|---|---|
| 3-axis x 1-plane arrangement (case 2) | 880 | 17280 |
| 3-axis x 4-plane arrangement (case 2) | 4400 | 83008 |
| Trilinear arrangement (eq.3) (case 2) | 700 | 12684 |

The criteria for selecting each set's representation is based on weighted rank, where the rank itself is based on either a) K-Means cluster centroid or b) combination weighted rank for distance between each set's representation, distance to center and Grassmann error value. 2D slice features allow for 2D check and interpolation to optimize Platform A's path when Platform E's path is known and fixed. Python rendering of the case study is presented in Figure 6-4(a), and the Spline interpolation is shown in Figure 6-4(b).



1D-array slice of the Middle Plate's workspace.
3D given path
1D-slice for each path segment
The Colour group represent the transition workspace when the Top Plate shift and move along the given path.
Figure 2
Middle plate's slice of workspace (1D-array)
Middle plate's 1D-array path network
1D-array Spline interpolation cannot solve the path optimization
X Label
Y Label

Figure 6-4(a) Middle Travelling Plate's list of possible path when Top Travelling Plate moves along the defined path and 12(b) Spline interpolation

Spline interpolation cannot solve the problem with weighted rank value. The interpolation may derive best fit curve but this unlikely to be the best solution. A weighted system need to consider factors like distance to base's center-point, gradient angle between travel and distance between sets of node in the path segment, where the objective is to reduce platform A motion to a minimum or to pose the structure in the best stiffness orientation, by aligning both platform E and A together.

Adjacency graph strategy to solve the 2D problem slice in Figure 6-5(b) for platform A is shown in Figure 6-5(a). Adjacency graph can be adopted to work with various methods like k-means clustering, path algorithm, search algorithm and weighted rank system. The adjacency graph also work with different slices or dataset of information, by creating linkage as an interpolation between the sets.

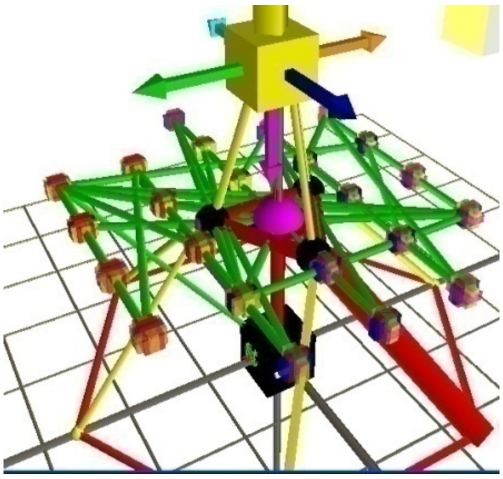
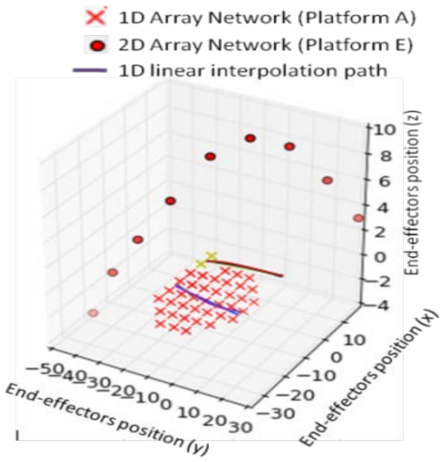


1D Array Network (Platform A)
2D Array Network (Platform E)
1D linear interpolation path
End-effectors position (z)
End-effectors position (x)
End-effectors position (y)

Figure 6-5(a) Adjacency graph strategy to solve the 3D problem and 6-5(b) 1D array network (A 2D solution)

### 6.4.2 Case 2: Boolean method for data slicing analysis

Slicing analysis operation is a data interpolation between points. Here, a ‘constant orientation slice’ checks a path that goes past each slice of information. The interpolation between points along the path and the slices give extra information regarding the path trajectory. There is a collections of possible points that connect the two data slices at the given ‘constant orientation workspace’, as shown in Figure 6-6(a). This is a 2D curve spline interpolation working on 1D-array data. Figure 6-6(a) is provided as an example of Python rendering for 3D interpolation between slices of datasets. Figure 6-6(b) demonstrates the problem with data interpolation, where other algorithm like Trilinear is required to add extra dimension to the interpolation.

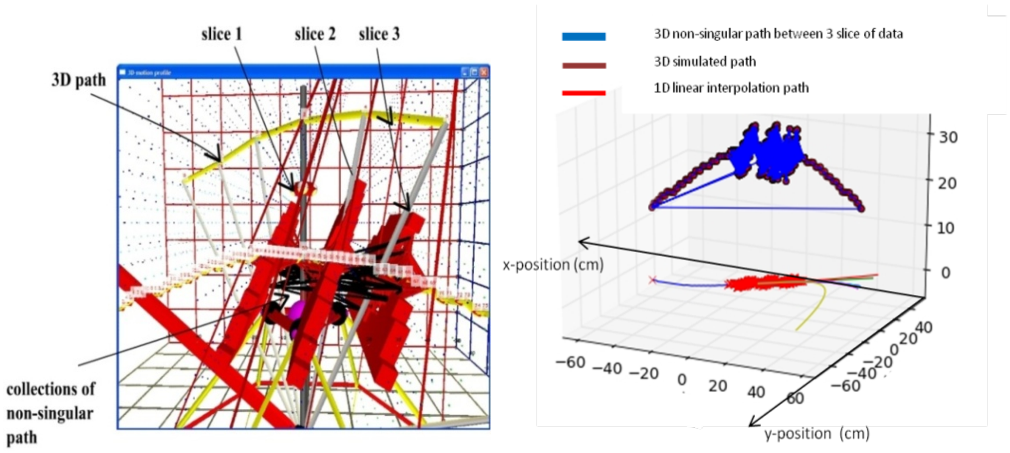


Figure 6-6(a) Interpolation between slices and 6-6(b) 1D path between the 3 data slices.

The 2D path interpolation based on 1D linear interpolation does not produce relevant result following similar argument of not being able to utilize the weighted value which is related to gradient changes between path’s node, distance of path’s node to the given path and distance between each node itself. The example for this 1D linear interpolation is given in Figure 6-4 and Figure 6-5. L-system random generator strategy for 3D interpolation is another method for extending the data slicing analysis. In this method quadratic algebra or ternary algorithm is used to form the adjacency graph between the random populated test points distributed on the collection of test planes. Gill use connected planar graph, and check for intersection between connected links to form the interpolation between slices. Then consider the slice orientation and rotation direction to ensure proper linkages is formed. Boolean logic is used here to control the 3D graph formation [102]

### 6.4.3 Case 3: Boolean method for search whilst avoiding obstacle

This case study demonstrates a method of using Boolean operation to find non-singular positions outside the obstacles placed within the non-singular workspace of the robot system. Two spherical obstacles placed at two different coordinates. The formulation for the collection of detectors shaped to form the spherical-shaped obstacles is based on standard sphere formula. Figure 6-7 shows an example case for obstacle avoidances or collision detections, where the case demonstrates the avoidances of two obstacles placed within the Work volume region. The obstacle (developed using equation 1) can be described as pick and place object, obstacle or working plane. Haptic interaction shall include force feedback interaction with the obstacle while user navigate or collide with the obstacle. This haptic interaction shall be based on Boolean logic operating on n-D Simplex algorithm. 2D optimization by using D* and A* for obstacle avoidance and path planning within a workspace is a useful to validate the various condition. Liu develop a Boolean method for collision avoidance inside Configuration space (Cspace), which is simplified here to demonstrate the idea $B_j(X) = \{X|h(X) > 0\} for\ j = 1,2..n$ , where X is a point in space (x,y,z), h(X)= 0 denote boundary, Bj represent Boolean method for dataset avoiding obstacle j, for a multiple obstacles scene [103]

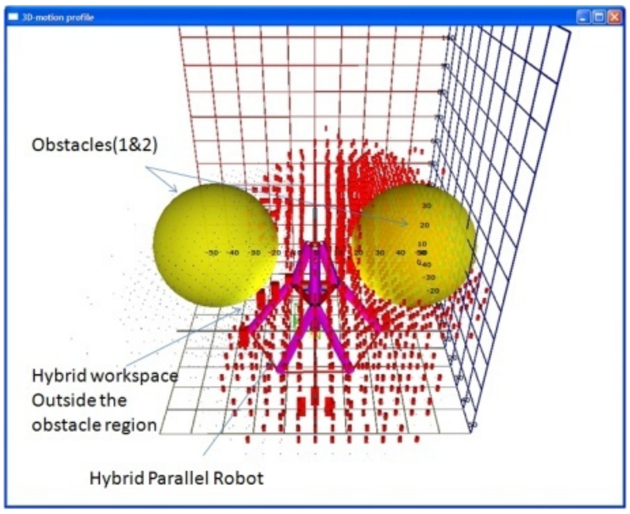


Figure 6-7 Example Python rendering of a work volume outside the two obstacles volumetric region

### 6.4.4 Case 4: Boolean method for search on a surface mesh

This is a study of platform E's path for non-singular condition following a surface mesh placed within the robot workspace. The search objective is to find points on the mesh that have non-singular positions. The search for non-singular points on the mesh surface is generalized. Figure **6-8**(a) and 6-8(b) shows the plot for non-singular points on the mesh. There is no limit to the number of meshes, locations and the shape variations placed within the workspace. A 1D array of the results of regression analysis produces lists of paths. The simulation then continues to perform a parametric sweep to find the best possible path on the mesh surface based on the slicing of sets of planes. Python example result for 2D univariate interpolation is shown in Figure 6-9. Adapting Liu's Boolean algorithm, to ensure the distance between end-effectors and mesh(target) is always equal to 0, therefore the equation now become $B_j(X) = \{X|h(X) = 0\} for\ j = 1,2..n$ , where X is a point in space (x,y,z), h(X)= 0 denote boundary, Bj represent Boolean method for dataset avoiding obstacle j, for a multiple obstacles scene. Liu's method does not require the analysis of the Cspace[103]. Pavic's research on continuous Boolean operation on surface mesh use both polygonal and volumetric data. The method requires extraction of the geometric mesh, which can be done by executing algorithm like Marching cube, simplex, dual-contouring and manifold topology. Pavic introduce cuboids to provide blocking and simplify the volumetric problem, then define the inside and outside mesh surface criteria based on Boolean logic. However, when using this method, there exists a problem with clipping during sharp corner and hole [104].

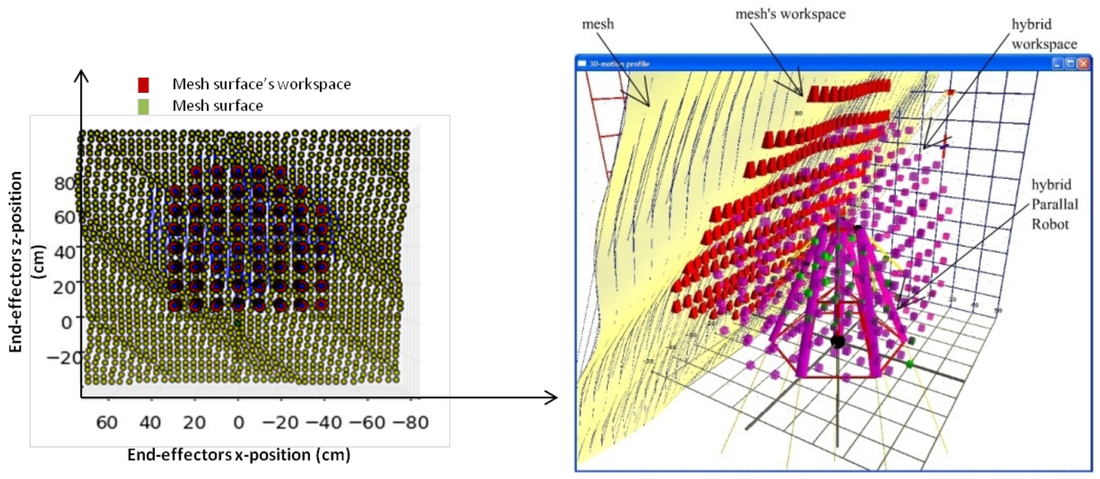


Figure 6-8(a) Boolean method search avoiding obstacles and 6-8(b) red dots represent the non-singularity points on the surface mesh

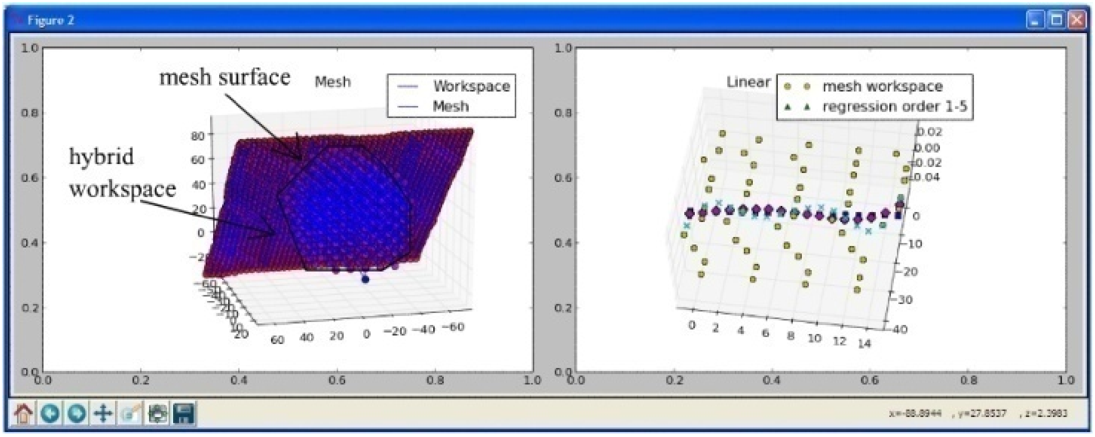


Figure 6-9 A univariate interpolations to find path on the mesh surface

### 6.4.5 Case 5: Boolean method for interval analysis

Interval analysis is a method for detail inspection of a set of data. The objective of this experiment is to demonstrate the detail analysis by subdivision of the path, and performing analysis for a given range along the path. Line segment is an operation done on a set of points to solve problems like analyzing the path accuracy. Interval analysis defines a suitable lower and upper bound during a parametric sweep search. The objective here is to reduce redundancy, unnecessary search outside the bound. Interpolation produces a relationship between sets of grid for a specific node. Figure 6-10 demonstrate an example for interval analysis operation on collections of points along a path, where interval analysis allows detail inspection of a union of points on that path.

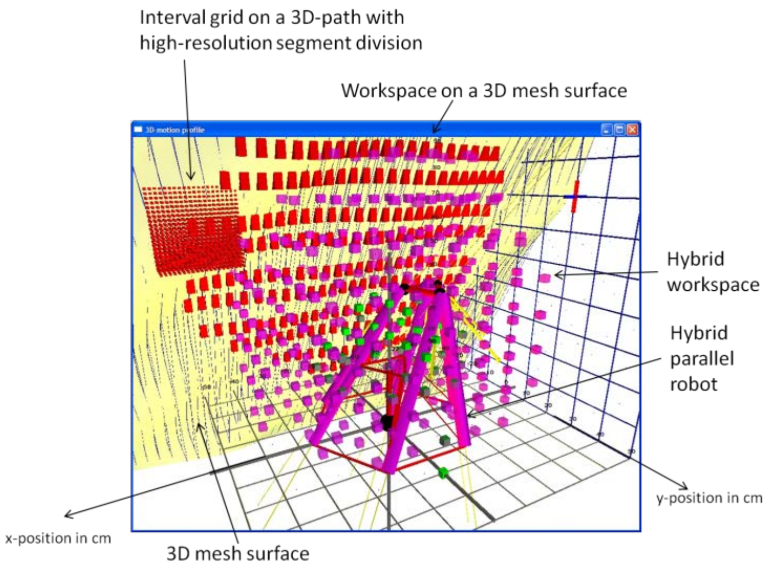


Figure 6-10 An ‘interval analysis’ system for the mesh surface workspace

Interval analysis is planned to be used as local and global search size definition and real-time path planning tool. The upper-bound and lower-bound search range defines the start and end node for a subset of a path, and new local parametric sweep grid is prepared. Gallego demonstrate the positive advantages of using adjacency graph or adjacency matrix in solving complex data [105]. Pigot uses n-simplicial topology which can be described as adjacency graph that allows for remapping between types of simplexes. Here, the migration between 1-simplex (an interval) to 2-simplex (a face) and 3-simplex (a volume) is allowable. Boolean provide useful and compact algebra to control the seeding and branching control of the simplex tree [106]. Following this idea, an interval analysis method has been developed, which quickly migrates between simplexes when either the system or the user needs it.

### 6.4.6 Case 6: Boolean method for quadratic interpolation edge determination

Quadratic or quaternion interpolation builds an instance of a parametric sweep grid between any set of slices or planes. The final pattern depends on formulation and the extrusion method. Table 4 displays an example Python rendering based on horizontal ternary interpolation using a given L-system pattern. The result is an adjacency graph which is a sweep travelling path for the end-effectors.

The difficulty in handling complex adjacency graph produced by the quadratic interpolation includes large number of linkages and large number of nodes. The Boolean algebra has to estimate each slice's parameters, thus reducing the number of network. This could lead to a combination of search methods in a single run [107]. Based on Bezier interpolation, the system can reduce the slice size and arrange its position and orientation effectively.  Figure 9-15,Figure 9-16 and Figure 9-17 illustrate this concept, where a path is given, and the system is expected to conduct an efficient parametric sweep following this path.

### 6.4.7 Case 7: Boolean method for L-system fractal random growth pattern determination

L-system is a formal grammar system with dynamic interaction with its environment (Refer Figure 6-11). Boolean logic is implemented to drive the fractal growth position, direction and limit. Boolean control the growth rule by directing the fractal towards the training set data. This training set data is a collection of test points generated by draft workspace cubic parametric sweep search. For generating the fractal, we use context sensitive grammars. Therefore Boolean logic can provide the ‘if-else’ scenario controls. Ikbal produced the development of scaling factor into L-System grammar to follow Iterated Function System (IFS) [108]. Merell discussed the requirement for constraints in fractal or procedural rendering, where we chose to implement the geometric constraints type algebraic in the L-System [88]. Boolean has to control few methods in L-System which are – a) ‘F’ move forwards d steps and draw a line, b) ‘f’ move forwards without drawing a line, c) ‘+’ turn right, d) ‘-‘ turn left and e) ‘|’ which means turn away from the current trajectory [109]. Dubois developed anticipatory algorithms that help control the L-System chaotic behavior, by introducing saturation factor by having incursion over the solutions. Dubois use Boolean table as a recursive flip-flop memory table that helps reduce the system complexity especially for a one-to-many relationship [110].

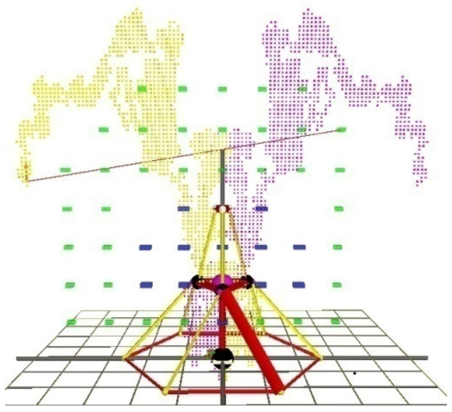


Figure 6-11 Example Python rendering of L-system fractal in automatic search mode

### 6.4.8 Case 8: Boolean method for determining Grassmann search behaviour

Boolean simple logic is used here to provide control for Grassmann error condition which includes coplanar condition, line-line join condition, and line-line meets condition, plane-line meets when given a plane and point-line join condition. Details regarding this operation can be found in section 5.4. Boolean logic controls various conditions and parameters namely the type of search, the parametric sweep method, the $C_{space}$ size and the analysis method.

### 6.4.9 Case 9: Boolean method for finding singularity loci

Tatsuya develop a graph-based control based on Boolean Network (BN), which pose a problem with the control method [111]. The parametric sweep search has been constraints with quick edge detection and stop travelling along a singular region. This is a difficult control method to solve, since this region is not known or there exists some non-singular position or orientation in the singularity region. And, then, there is also a large variation of workspace type to be considered. The application to control the sweep search is limited to a combinations of AND and OR which check for geometry singularity and all 5 Grassmann constraints. The Python simulation reduces the complexity by focusing on a smaller $C_{space}$ and performing coarse-resolution parametric sweep before the fine-resolution operation. There is various search methods, which can be employed by the user to optimize the loci search. The loci condition is equivalent to Grassmann singularity [53], which has been implemented as a real-time operation. The loci search optimization has to consider other factors including the parametric sweep type, data population method, and the interpolation between the test nodes. The singularity loci conditions need the moving platform to be rotated at an angle, where the Grassmann singularity becomes a risk. The search parameter would set the moving platform angle, perform the parametric sweep and follow the Boolean rules. This iteration continues with different rotation angles for the moving platform until the iteration reaches the rotation limit. The discussion on two fixed constant orientation blending is given in section 7.4.

## 6.5 Conclusion

Boolean provides logic controls of the search and validation strategy to determine singular and non-singular position and orientation workspace. Boolean logic is the control structure that defines the search parameters; search criteria and analysis. Search parameters include upper and lower bound and parametric sweep shape. Boolean produces criteria for search technique and filter. Boolean can limit the region, direction of search and the search behaviour. The system is scalable and modifiable; therefore a region can be extended or reduced. The research follows the concept demonstrated by Merlet, which is then extended by many other researchers.

Boolean algebra controls the 3D binary tree method by directing the growth and new seeding where appropriate. This way, Boolean provide limit for growth, direction of growth and scale the growth to satisfy the 3D sweeping parameters.

Various Boolean methods are presented in this chapter as case studies to demonstrate the concept. Boolean provide the necessary control for path planning, surface mesh interaction, slicing and interval analysis. Boolean has successfully limits the search space and control the search direction.

# Chapter 7: PATH PLANNING

## 7.1 Introduction

Parallel robot is a closed-loop system with translation (x, y and z) and Euler angles (α, β, γ). The amount of Degree-of-Freedom and its workspace is not limited to any combination. However, there is a risk of existence of singularity in the final design. This section discusses the path planning aspect of the parallel robot design. Path planning requires the parametric sweep result, whose validation using was explained in the previous chapters.

## 7.2 Definition for search region

C space is a configuration space where the robot can move without going into a singularity condition. C space for Hexapod Robot is basically a hemispherical shape and can be represented as special Euclidean group$SE(3) = R^3 SO(3)$. A $C_{free}$space is a space where the robot can move while avoiding obstacle. Qin explains the uses of $C_{free}$ and $C_{obstacle}$ to reduce the requirement for C-space search, since C-space search is computationally intensive. Qin describes the potential of using any representation that would help improve the robot’s motion-related task when C-space is replaced by another method. Glavina first explain the concept in 1994, where it is hoped that a new method would solve the problem by focusing on the problem at lower Degree-of-Freedom, reduced number of nodes in graph and to help improve A* search by reducing the system complexity. Qin highlights the issue of identifying the maximum reachable pose, which may not be within the $C_{unique}$

workspace. Qin uses discretization heuristic to control the resolution, which is also known as interval analysis. Heuristic rule ensure that critical region is being given higher resolution, in order to improve overall system accuracy, while not losing speed [8,79].

Glavina suggested that C space depends on the number of actuators, however neighbouring node search grid is limited to 3D or 4D space. The graph nodes have to be limited in order to generate enough test data within the acceptable resolution for a wide range of 3D space. A* which is based on Euclidean distance to an end pose is not sufficient in this type of C space. Glavina primary objective is to divide the problems into a small and simple 2D problem, where the focus is on the issue itself rather than the whole 3D space. Glavina uses method like slide-step in order to quickly establish the maximum reach or the edge, and to quickly find the end pose or the target. To improve trust in a questionable region, Glavina introduces a sub goal which is a form of random generated test data [112].

Bohigas explore the force-feasible C-space for a path, where A* is then used to find best path on a mesh generated by first avoiding singularity loci space, since the travel is done based on constant orientation method. Some of the issue addressed by Bohigas includes the problem with interpolation, where certain factor may not be considered by the system like the orientation compliance or the gradient angle changes between the two poses from start to finish position [7].

## 7.3 Path planning strategy for parametric sweep search

There is a significant difference when the parametric sweep search for non-singularity path is done using serial or parallel search method. Qin experiments with parallel search in a C-space, where a seed generator can be placed at any node, with certain criteria for branching out to form the parallel search [8]. This section shall explore the serial, parallel and spiral sweep method while adapting ideas from ‘Gift wrapping algorithm’ and ‘Jarvis march’ to quickly identify workspace’s edge, $C_{free}$ and $C_{obstacle}$. Those method and few others are well known in convex-hull quick search method, such as Graham Scan. The methods are adapted for 3D sweeping strategy to quickly find path, workspace and singularity for parallel robot application.

Adjacency graph could be categorized into unbalanced, AVL balanced tree, serial and parallel, ternary and binary tree. An AVL tree (developed by Brad Appleton) is a binary tree, and as a balanced tree, the difference between the height of the left and right tree is never more than one. To balance an AVL tree, every time there is a new insertion into the graph, a rotation function is required. The rotation can be either single or double rotations, with corresponding ‘left’ and ‘right’ versions. An AVL criterion has to keep track of the AVL state by checking the height difference and rotate the graph when necessary. The advantage of using AVL is described as being fast 0(log n) time in both the average and worst cases for data lookup, insertion and deletion [113].

The Python simulation integrates various form of adjacency graph in simplifying the order of chaos in the result, especially in making senses of the relationship with other weighted value like cost-factor.

Parallel robot path planning is an essential factor in ensuring that the robot can perform the given task safely, accurately, timely and cost effectively. As an example to illustrate the concept, a robot is given a task of performing a soldering work on a block. The robot must be able to position its end-effectors and move along the given cutting path. The robot must be able to follow the given path while its based is fixed to a given position or mobile, moving along another path. Therefore, path planning has to consider many general and specific factors that relate to these tasks. A Python 3D simulation system has been developed based on Grassmann pencil-line terminology, where it's kinematic has been validated using numerical system, and the basic singularity check is based on structural geometric and Grassmann validation. The full 3D system allows for 3D experiment, analysis and data visualization. Various 3D path planning and analysis method is shown in this research to demonstrate the system ability. The experimental result is an ad hoc study of the problem, which produces result quickly at low resolution. User is expected to perform highly accurate and numerical analysis following the result in order to improve its accuracy and validity.

Bhattacharya mentions the requirement for estimation algorithm in solving path planning, which leads to heavy computation time [81,82]. Ur-Rehman explains that one import aspect of path planner is the placement of the work piece inside the robot's workspace. For a mobile or reposition-enabled robot, the path placement has to consider the optimization of the robot's base, too [116].

## 7.4 Experiment setup for two different scenario

### 7.4.1 Setup 1: Path strategy between two Constant Orientation Workspace

The first experiment setup is shown in Figure 7-1, which is based on constant orientation workspace data for two different pose, where start pose =coordinate (-20, 16.1763, -18.2472) and end pose = coordinate (10, 18.2663, 21.8673), where their orientation axis are + and – degrees along x-axis. The red cubic sets represent the constant orientation workspace for start node, and the yellow cubic represent the end node. Two extreme positions have been selected as the start and end point, where a direct straightforward path between them is not possible, when any of the two orientation angle is applied to the end-effectors.

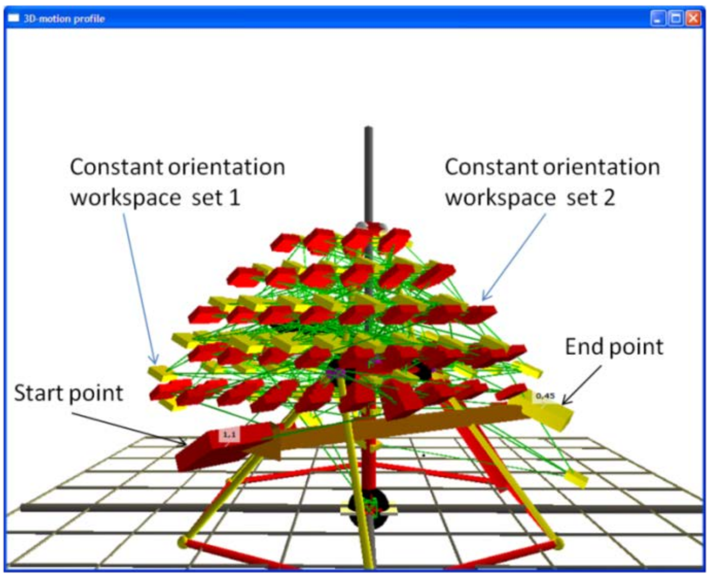

3D-motion profile
Constant orientation workspace set 1
Constant orientation workspace set 2
End point
Start point
0,45
1,1

Figure 7-1 Path between [0][45] to [1][1]

### 7.4.2 Setup 2: Path strategy for platform A when platform E is moving and following a path

The second experiment, shown in Figure 7-2, is based on a hybrid robot, where platform A is a hexapod with 6 legs structure and platform E is a tripod with 3 legs structure. This setup is a search for best path for platform A when platform E needs to follow a specific path. The criteria shall consider these parameters like gradient changes towards the target, gradient changes between nodes, distance from each node, distance from the centre position and singularity rank value. Figure 7-3 demonstrates the nearest distance to centre-point result for platform A's workspace data.

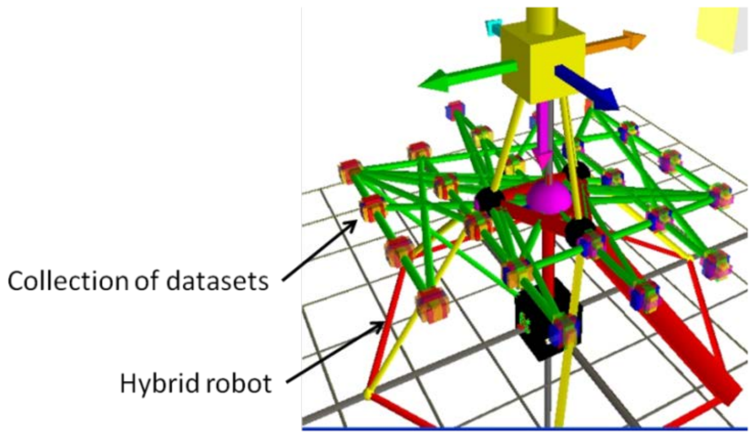


Figure 7-2 Collection of datasets representing Platform A's workspace

## 7.5 Path planning with slice analysis for trajectory control

Path planning is related to allowable end-effectors’ motion when there is no singularity in the path, where the path could be fixed or modifiable. Path planning has to consider various factors like distance between node, cost to move and reach the next node, sub-singularity elements like stiffness factor, Grassmann, vibration, trajectory changes and velocity. Bonnemains found that elastic deformations have higher influence on the x-axis for his experiment on cutting a block of an aluminium block. Bonnemains added that Matlab ODE15’s solver is noisy [117]. Rossi propose that no robot can move with discontinuous movement and velocity, and irregular curve should be avoided in performing path planning. Rossi explains that polynomial planner has the disadvantage of many issues, for example when the polynomial increases, the trajectory becomes not natural for the manipulator and the polynomial depends on the given points, when any points is changed, the polynomial has to be recomputed. According to Rossi, to solve the single high order polynomial disadvantage, the path is broken into segment of low order polynomials. Rossi comments that a trajectory planning method has been developed at the Di.M.E. at the University of Naples ‘‘Federico II’’, where the planner considers another constraints that is the trajectory tangent [118]. There are various paths planning strategy in 2 dimensional applications, but not many in 3 dimensional spaces. Path planning is also associated with task, therefore the path strategy has to consider new factors like constant position, constant orientation, continuous surface contour, obstacle within the path space, cutting and machining theory, force required to perform cutting and many more.

Ata use the term trajectory planning, and describe the constraints for planning as system constraints which is imposed by the robot's geometry and task constraints given by the task. Ata define the planning problem as the difficulty in calculating feasible trajectories from a given task (could also be a 'pose-to-pose' problem) while maximising the robot's capabilities. A time-trajectory planning is either in joint space or Cartesian space. At joint space, this is specified to each joint and actuator. A Cartesian space is not generated at joint space; therefore there exist possibilities for geometric singularity at joint space. Trajectory planning has to consider reasonable time for the manipulator to increase and decrease its stroke length [119].

Ata explains that the common trajectory planning methods are a) polynomials in time, cubic polynomial and splines in time, b) linear interpolation with smoothing and linear interpolation with parabolic blends, and c) optimal controls like shooting method [119].Lou mentions that motion planning can be a kinematic-based or dynamic based. Kinematic based planning is limited to kinematic constraints only. Dynamic based consider both kinematic and dynamic constraints, which has the objectives of maximizing structure stiffness and to minimizes actuation forces while working within the limit of workspace and singularity [120]. Chen discuss the problem with Stewart Platform reaching a singularity manifold, that leads to the leg forces sudden increment moving towards the allowable limits. Chen develops a cost functions and constraints which consider minimum actuating forces, optimum time and energy efficiency. And, Chen added that a weighting coefficient is required in order to validate the singularity possible condition. The parameters are leg length constraint, leg linear velocity, leg linear acceleration, actuating force constraint and it should also consider leg's collision. Chen mapped pseudo cost function based on the weighted penalty functions , which is then mapped into the function of the spline parameters [9]. Bhattacharya mentions the requirement for estimation algorithm in solving path planning, which leads to heavy computation time [81,82].

Ur-Rehman explains that one import aspect of path planner is the placement of the work piece inside the robot's workspace. For a mobile or reposition-enabled robot, the path placement has to consider the optimization of the robot's base, too [116].

Various paths planning method is demonstrated with examples of the 3D Python simulation capabilities for flexible integration with various methods to solve design problems. The research addresses the issues of using 2D interpolation in solving the 3D problem in Parallel robot design. Such problem includes the issue with 3D path planning and 3D obstacle position. The methods demonstrated are experimental ideas, and many of the condition or case study presented is not validated using other method, since it is difficult to implement such method in Solid Work or Matlab environment.

## 7.6 Experiment with various path planning methodologies

Various paths planning method is demonstrated to provide reader with examples of the 3D Python simulation capabilities for easy integration with various method to solve design problems.

### 7.6.1 Method (1) k-means clustering

K-means cluster is a strategy which partition the n datasets into k sets $(kn \leq n) S = \{S_1, S_2, .., S_K\}$ to minimize the within-cluster sum of squares

$$min = \sum_{i=1}^{k} \sum_{x_j \in S_i} \left\| x_j - \mu_i \right\|^2, \text{ for a given set of data(x}_1\text{, x2... xn)}, \quad [32]$$

Where each dataset is a form of d-dimensional real vector.

Figure **7-3** and Figure **7-4** demonstrate an example where k-means cluster path planning optimizes platform A path when platform E is following a fixed path. To demonstrate and clearly visualize the effect, data slicing is used to parametric sweep and plot only a list of layers along an axis. This way, the k-means path planning and the dataset relationship are clearly shown.

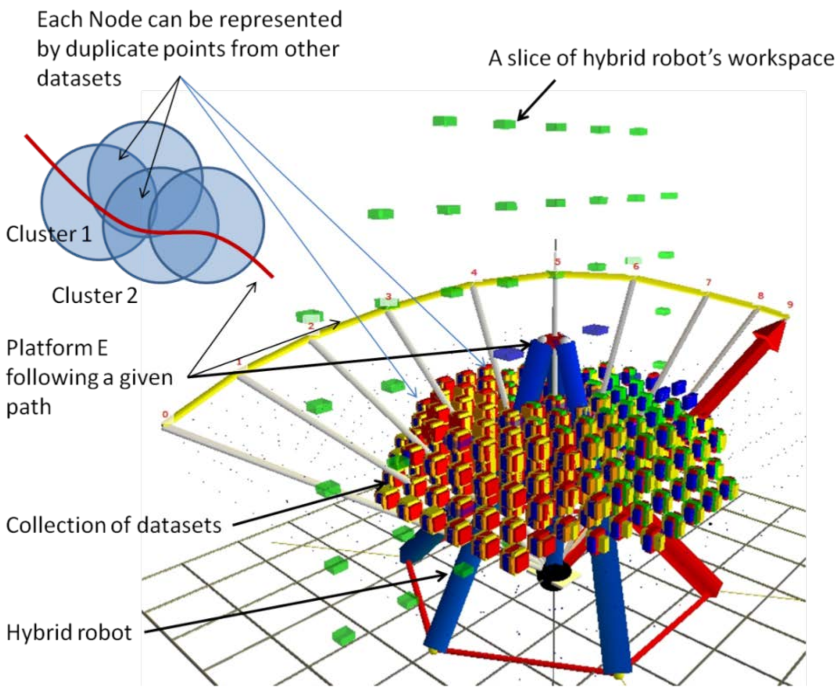

Each Node can be represented by duplicate points from other datasets
A slice of hybrid robot's workspace
Cluster 1
Cluster 2
Platform E following a given path
Collection of datasets
Hybrid robot

Figure 7-3 Adjacency graph network

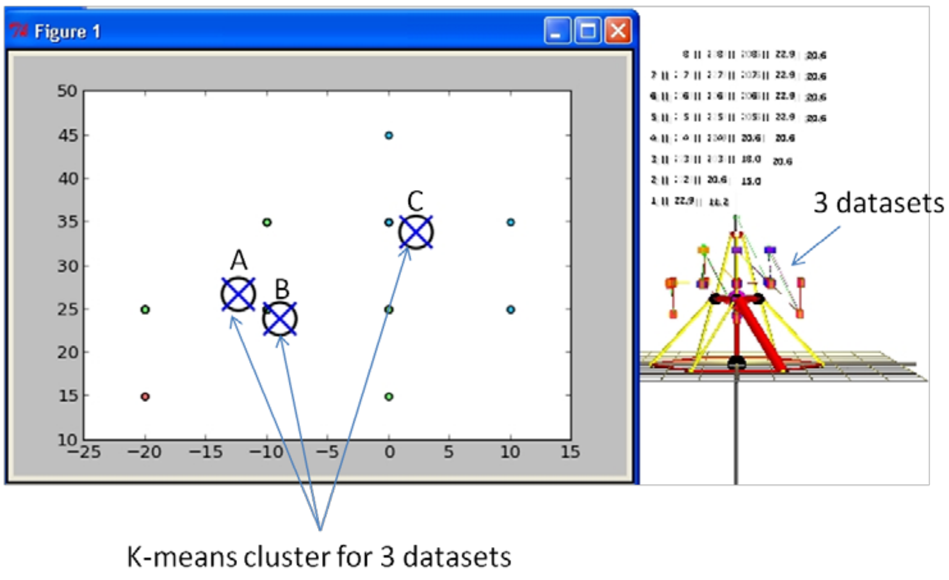


Figure 7-4 K-means clustering example (point A, B and C) for 3 datasets representing workspace for platform A

### 7.6.2 Method (2) following a given target

Assuming that similar trajectories for both platforms A and E produce stiffer pose, this experiment looks into path planning based on following a target, where both platforms A and E attempt to maintain heading or trajectory direction towards the given target. This given target can be a moving object, static or a series of points representing a path. This is similar to a compliance workspace search, where the end-effectors try to maintain its trajectory following a given path or mesh. The trajectory can also be maintained at a minimal gradient changes [20]. This method is seen be similar with visual servoing method if it is integrated with a minimal gradient changes, where the lowest angle differences is maintained [121].

### 7.6.3 Method (3) A* D* 3d path planning

Carsten describe the D* 3D path planning method which is based on interpolation-based cost approximation. This method allows for optimal straight line path via interpolation of its current pose, and validating its trajectory towards the target[122].

D* path planning for obstacle avoidance within a known workspace is given here as

$$g(s_f) = [g(s_1) + g(s_0) - g(s_1).t].(1-u) + [g(s_2) + g(s_3) - g(s_2).t].u$$

$$rhs_{sf}(s) = C.\sqrt{1+t^2 + u^2} + [g(s_1) + g(s_0) - g(s_1).t].(1-u) + [g(s_2) + g(s_3) - g(s_2).t].u$$

(33)

Where g($s_0$),g($s_1$), g($s_2$), and g($s_3$) is the path cost of any point on the face of the cubic 3d grid unit, C is the traversal cost of the Voxel on both f and s, and the cost of a path from s through $s_{fs}$is given as $rhs_{sf}$(s).

### 7.6.4 Method (4) Voxel-planner based on Binary tree

3D Voxel-based planner is based on nearest neighbourhood interpolation, which we attempt to replicate and extend the concept for parallel robot path optimization, by using Marching cube model. The methodology for seeding the 3D marching cube path is based on Verhoeff 3D binary tree development, which replaces the basic geometry seed generator object with marching cube, and the ridge functions is replaced by the 10-variations of marching cube orientation seeding element [123]. Figure 7-5 displays the 10-

variations trajectory indicator and seeding element where new object is to be generated to build a path tree.

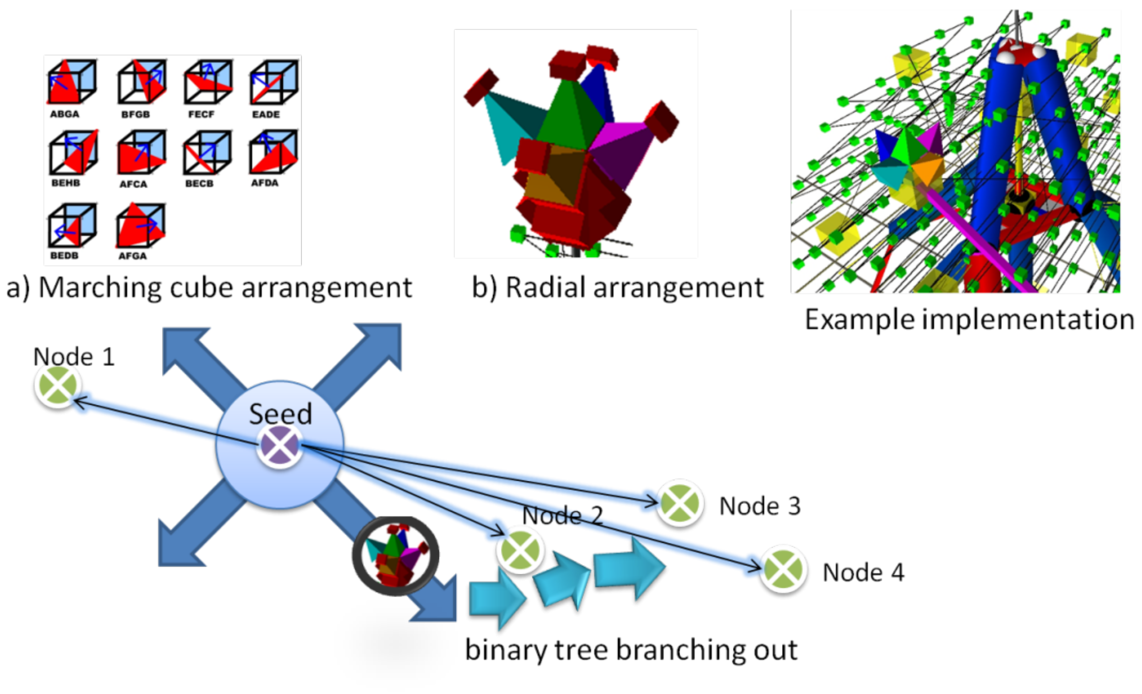


Figure 7-5 A marching cube 10-variations with trajectory indicator for seeding new branch

Figure 7-6 displays the seed which generate binary tree throughout the workspace. The seed branching out towards the low-resolution cubic parametric sweep search result. The sweep search travels between the low-resolution and the high resolution workspace. The cubic parametric sweep is parallel, regular-spacing between lines and rigid.

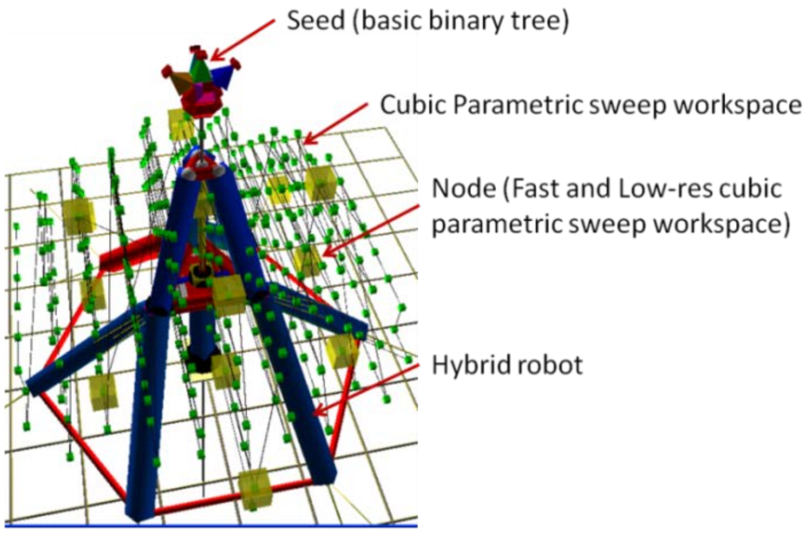

Seed (basic binary tree)
Cubic Parametric sweep workspace
Node (Fast and Low-res cubic parametric sweep workspace)
Hybrid robot

Figure 7-6 Seed for binary tree production placed within the workspace

Figure 7-7 displays the calibration for fabricating the binary tree result or workspace shape. Changes to the binary tree sensitivity parameters which comprised of distance and angle between low-resolution and high-resolution test data will result with different final shape. Figure 7-8 displays the result of a 3D binary tree workspace. This 3D tree generator is based on Verhoeff solutions.

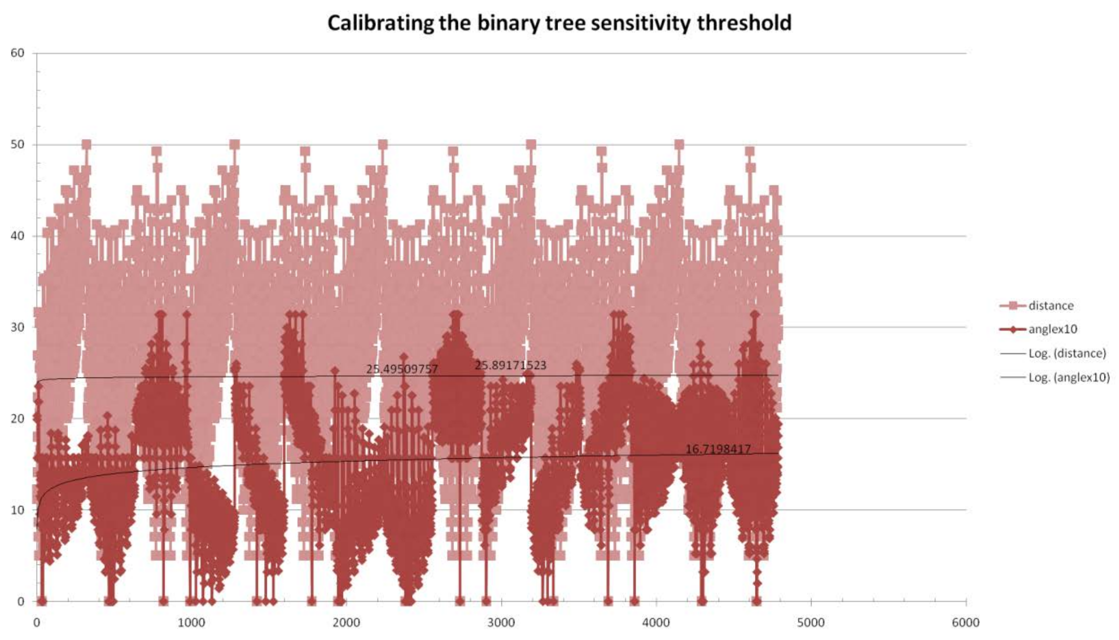


Figure 7-7 Calibration parameters for binary tree

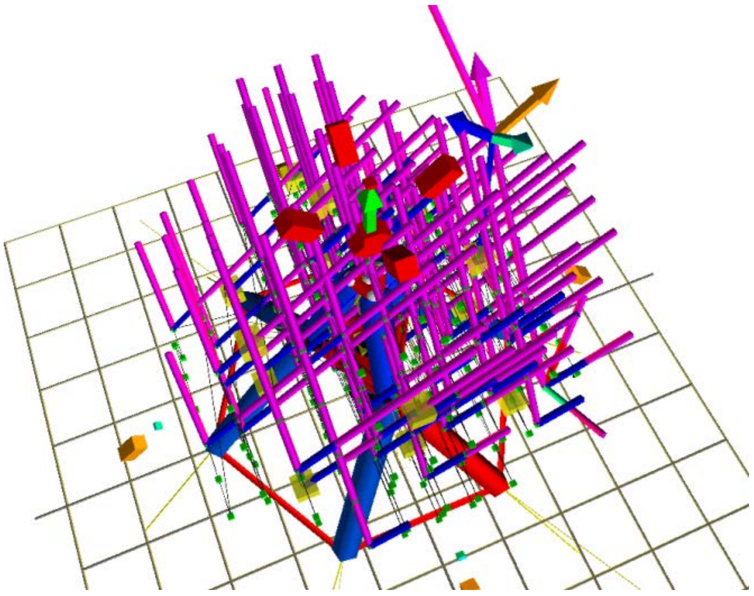


Figure 7-8 Example Python simulation rendering of a 3D binary tree

### 7.6.5 Method (5) Connecting two constant orientation workspace dataset

For a known start and end pose, where the pose's orientation is known, two sets of parametric sweep to find the constant orientation workspace for both start and end nodes were developed. Then, both adjacency graphs were connected to build the final path, connecting the optimum points on both dataset and interpolate when there is no data available. Li researched on connecting two graphs in 3D which is for narrow passage planning method combining Randomized Star Builder (RSB) and uniform sampling to extend the local tree and build the connection between two datasets [124]. Nieuwenhuisen extended the Rapidly-exploring Random Trees (RRT) with Boolean algebra to reduce the graph complexity thereby improving system performance [125]. Figure 7-1 demonstrates an example of connecting two sets of constant orientation workspace graph, and further discussion regarding the method is given in the 3D sweeping section. Figure 7-9 display the sequential orientation changes between start and end position.

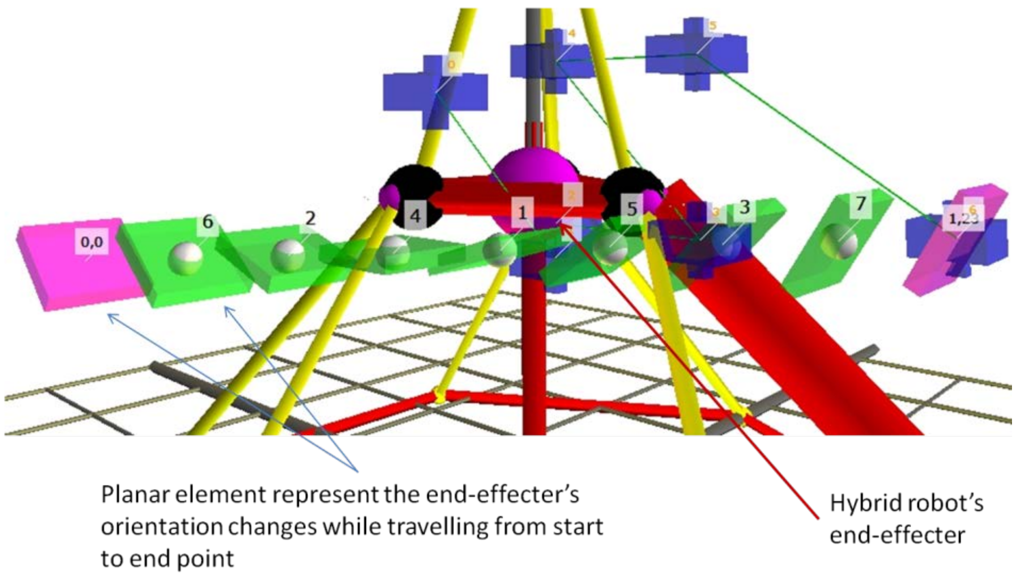

0,0
6
2
4
1
5
3
7
1,23
Planar element represent the end-effecter's orientation changes while travelling from start to end point
Hybrid robot's end-effecter

Figure 7-9 A constant orientation pose from start point [0,0] to end point [1,23]

### 7.6.6 Method (6) 3D Ternary Interpolation

Interpolation-based path planner by using L-system random generator, with specific Voxel referred to as Ulam spiral is populated onto a collections of slice planes. The ternary or quadratic relationship between the slices produce 3D interpolation data, which can be ranked, based on nearest to straight-line path, gradient changes and singularity value is shown in Figure 7-10.

Ternary algorithm for interpolation between random populations of dataset on collections of plane's slices is given below following Jin's method. According to Qu, a ternary number, $t = xi + yj + zk$ , where x, y and z are real numbers while I,j and k are imaginary units. And, a ternary maps is described as, $t \to t^{m} + c\ (t, c \in T)$, where T= ternary number, C=complex number system, m = exponential number [49].

Ulam Spiral or prime spiral formulation is in 2D, and the extrusion or ternary method produce the 3D effects for Ulam Spiral. This is then used for path planning, distribution of test points, dataset interpolation or extrapolation and producing adjacency graph network. General Ulam or prime spiral formula is given here as$f(n) = 4n^{2} + bn + c$, where b and c is an integer constant, and n is a set of numbers [73].

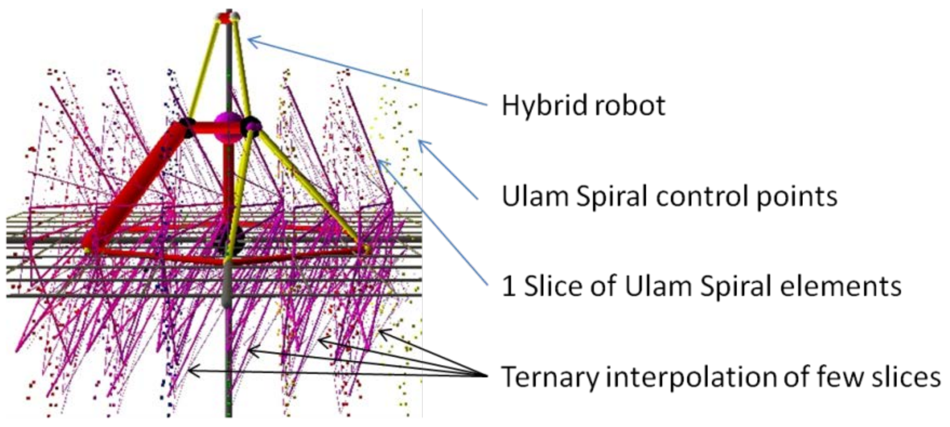

Hybrid robot
Ulam Spiral control points
1 Slice of Ulam Spiral elements
Ternary interpolation of few slices

Figure 7-10 Ternary interpolation

Ternary interpolation along a path is done by using slice analysis. Each slice is a planar element with user-defined or automatically builds position and orientation to satisfy the path criteria [126]. The criteria for placement along a path includes divide and conquer to build a 3D Bezier curve, where a midpoint calculation is performed between each node to build a new node to be further subdivided, using 3D general midpoint formula $midpoint = \big((x + x) \div 2, (y + y) \div 2, (z + z) \div 2\big)$. This Bezier curve construction can be done after the system has optimized and run the selection process based on criteria for best possible path. Use de Casteljau Subdivision Algorithm which is an ordering system for subdividing the best possible path's dataset into an AVL tree graph [127]. This best possible path is derived from a set of ternary graph network data. The criteria for selections include nearest distance to the path, gradient angle between nodes and target, and singularity rank value. Amato use clustering method to prioritize or seclude the important region away from the whole dataset. The objective here is to populate the space with reasonable amount of test data, and use effective way to measure and define parameters to seclude the important region for the final processing which are the best possible path and then the Bezier curve construction [50].

The ternary slices shall then be placed along the generated 3D Bezier curve with the control point perpendicular to the Bezier curve is proposed as the best candidate for placing the slices. The trajectory is derived from the Bezier curve forward angle toward the next node. However, each slice is user reconfigurable and ternary interpolation can be generated to satisfy user requirement.

Figure 7-11 demonstrates an example for a collection of 2D planar slices placed along a given path. The planar slices shown as dotted green can be placed using various algorithm to optimizes and satisfy user requirements. Interpolation of various methods can be performed here to create graphs between the planar slices.

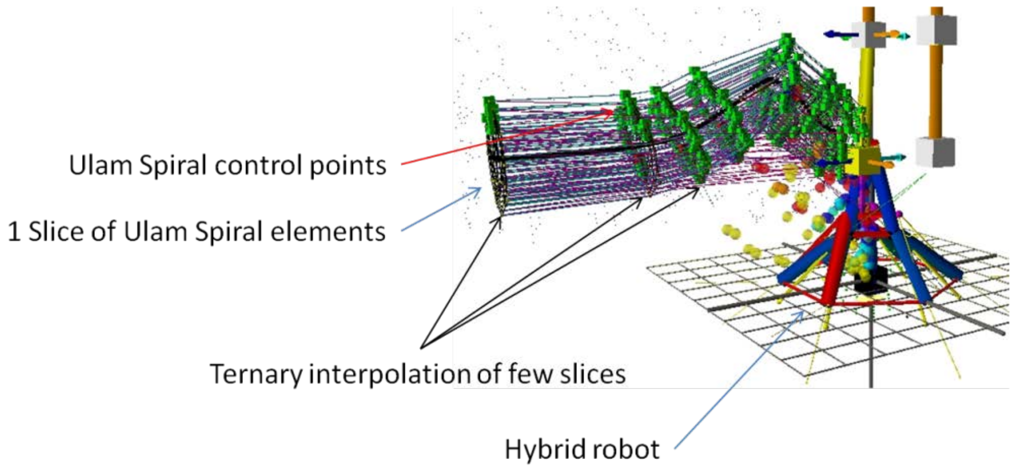

Ulam Spiral control points
1 Slice of Ulam Spiral elements
Ternary interpolation of few slices
Hybrid robot

Figure 7-11 An example Ternary interpolation for a collection of planar slices.

### 7.6.7 Method (7) 3D sweeping

The typical sweep is also known as parametric sweep based on geometrical and rigid grid like cubic, conical or spherical. Shah commented that the parametric sweep is lacking the initial condition estimation, and the problem with the definition for sweeping range since the region may change depending on the workspace search typology [86].

3D sweeping based on mesh surface grid or patch in the form of Bezier or Coons patch is a special case for dynamic grid system. This grid patch is driven by de Boors control points which is a generalized form of de Casteljau's algorithm which help find C(u) on segment u, where C is a subdivision of a segment between point A and point B. The control point parameters like scale, position and orientation, and subdivision method can generate various shapes with parallel path line and equally spaced and arranged test points. The resulting shape serves as the grid system for the parametric sweep search [49]. A basic cubic parametric sweep is demonstrated in Figure 7-12(a), and the cutting block is shown in Figure 7-12(b).

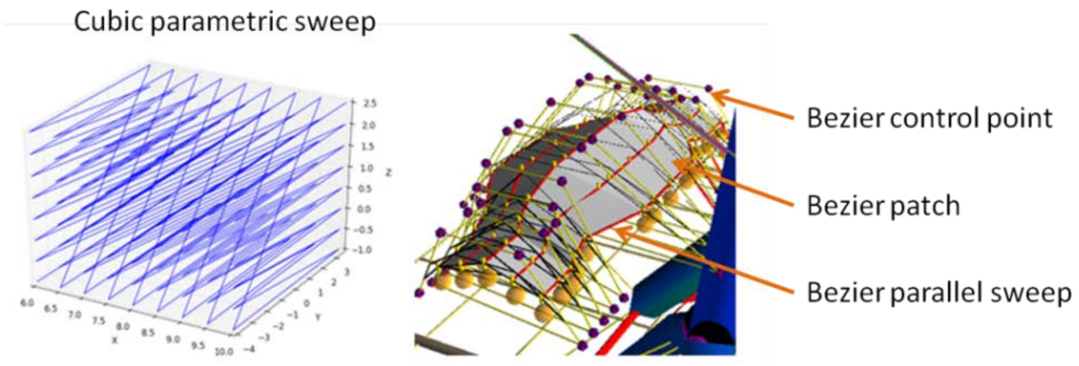


Figure 7-12(a) Cubic parametric sweep and (b) patch grid

This method is a modification from Jarvis march which looks for edges of a set of data. The 3D sweep shall consider these parameters including gradient changes towards the target, gradient changes between nodes, distance from each node, distance from the centre position and singularity rank value.

The characteristic for the graph plot in Figure 7-12 demonstrate the parametric cubic sweeping pattern, therefore the plot resemble the layer-by-layer and end-to-end sweeping effects.  The criteria are plotted against the first stage's result. First stage is the initiation method where two sets of workspace are found by using constant orientation workspace search. Figure 7-13 demonstrates a complete graph method where each node is linked by an edge. Figure 7-14 display the second approach to handling adjacency graph. There are various other methods for handling the graph network, which is related to the search strategy.  Table 8 describe some of the search strategy which could be employed in solving the path problem.

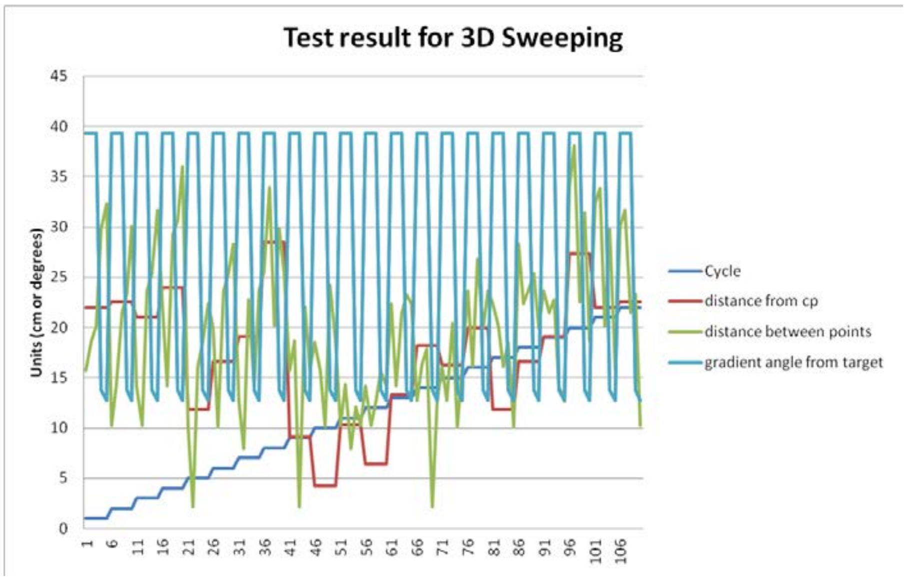


Figure 7-13 Test result for 3D sweeping method based on Adjacency complete graph

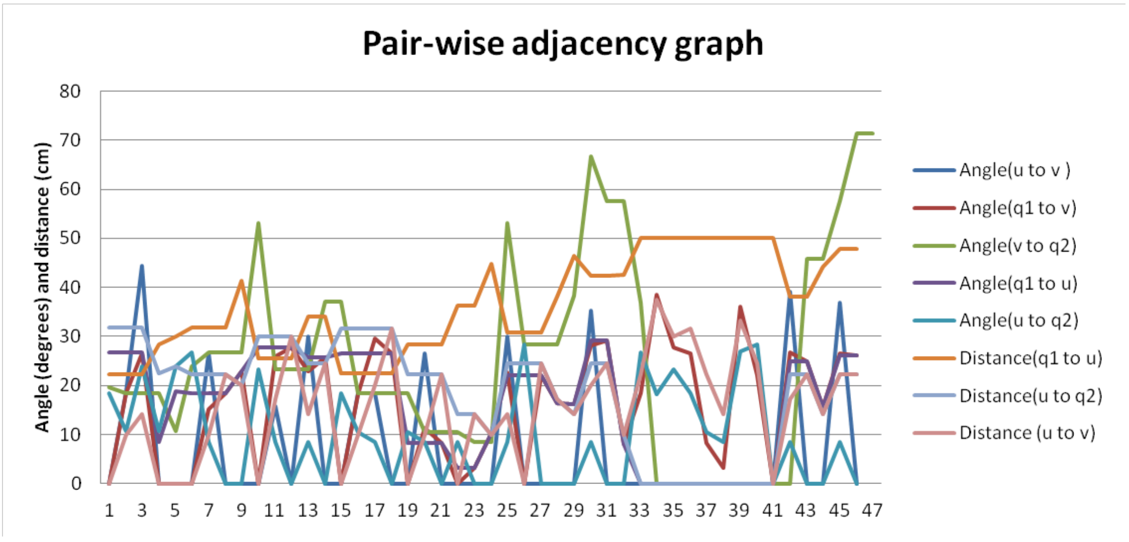


Figure 7-14 Pair-wise adjacency graph

Figure 7-5 displays another way of analysing the adjacency graph. This method put emphasizes on the minimum angle differences between start (q1) to node (u or v) and towards the endpoint (q2). The optimum condition is when the node's angle is a minimum and it is near to the straight line path.

Table 8 gives the general differences between the paths methods presented in this chapter. The 3D sweep and Voxel-based planner shows a lot of potential and possibilities in solving 3D Parallel and Hybrid robot workspace, path and singularity problem. Further investigation into the various search algorithms should be able to improve and optimize the adjacency graph network used extensively in majority of the listed path methodologies presented here.

Table 8 Path comparison

| Method | Time | Characteristic | Advantage | Disadvantage |
|---|---|---|---|---|
| (1) K-means clustering | 20s | Simple and quick way of solving path definition for a collection of datasets. | Fast 2D and 3D system. | Does not consider other factors like weighted ranking. |
| (2) Following a given path | 1e+04s (1511 Test points) | Interpolate within the given region, while considering Beziers's path profile to estimate the intervention node. | Known region | Robot's base position and orientation, geometry may restrict the robot from following the given path. |
| (3) A* D* | No data | Best for solving 2D obstacle problem. | Fast and flexible method for solving 2D obstacle. | 3D integration is difficult |
| (4) Voxel-based planner | 77s (291 Test points) | Similar strategy to no. (7) However this is a more balanced and stable approach. | A simple tree with balanced distribution. An open system that allow various formulation for graph network, search strategy and search | The calibration for threshold value plays important role, which has direct effect on the resulting grid's shape. |

| | | | | |
|---|---|---|---|---|
| | | | dimension. | |
| (5) Connecting two constant orientation workspace dataset | 2.3e+03s (296 Test points) | Connecting separate workspaces | Interpolation or network graph to predicts the transition or the missing links. | Assumption has to consider various factors like constant target orientation. |
| (6) 3D ternary interpolation | 5.5e+02s (57 Test points) | Various data population and data slices' position and orientation opportunity. | The L-system has wide variety of patterns, and it is scalable. | The random generator produce less effective test position and it is hard to control its direction and distribution. |
| (7) 3D sweeping | 49s (296 Test points) | True 3D random generator and learning mode for building 3D parametric sweep network grid. Based on convex-hull quick search method. | The system should be able to build grid coverage of similar workspace region faster and it is a scalable and manageable system. | Complex to develop and integrate with the system |

## 7.7 Conclusion

In this chapter path planning, which starts with the definition for smaller test region or Cspace, was presented. Then, the significant of having a different envelope shapes for the allowable region for the parametric sweep search was discussed. The experimental setup consider two difficult aspect which are constant orientation workspace blending and the test for estimating the middle plate or platform A’s position and path when platform E has to follow a pre-defined path. The simulation result provides optimization method which reduces noisy data which is apparent in Matlab ODE15’s solver.

The chapter highlights the problem with the usages of 2D path planner for parallel robot problem solving. The system is proven to perform the path planning in 3 dimensions and most analysis is done in 3 dimensions, too.

AVL Binary graph approach has been tested for adjacency graph and 3D binary tree development. The AVL binary tree which is a balanced tree simplifies the data processing and post-processing analysis. The geometric parametric sweep is a rigid system and this is limited to invalid initial estimation and unknown search limit. The non-geometric parametric sweep systems like ternary L-system and Bezier patch has the advantage of being specific and focus. The 3D binary search tree concept is based on Glavina approach which reduces a complex workspace problem into smaller and manageable region. The 3D binary tree sweep method is open-system, and the example search is based on breadth-first search algorithm. The adjacency graph construction display the flexibility of the system, where any search algorithm can be integrated while the fundamental concept remains similar. The search algorithm produces different interpolation and data population strategy, which use different focus and priority.

The 3D Python simulation has demonstrated a few paths planning strategies in 3-dimensional space. The Python simulation can be integrated with other path algorithm which utilizes any of the data population and test strategy. Boolean provide control for the system flow and help reduce the problem’s complexity.

# Chapter 8: HAPTIC CONTROLLER DEVELOPMENT

## 8.1 Introduction

This chapter addresses the development of a haptic controller system, which consists of a structure, control system and a haptic engine. The structure has to be universal for various types of parallel robot typologies and topologies. Therefore, the structure itself needs to be optimized to cater for the different constraint and number of degree-of-freedom specific for a design. On the other hand, user prefers a single contact point to represent the travelling plate. This control should be transferrable to other element in the geometry like joint's position and other travelling plate. The haptic controller is PPP and RRR structure where the RRR is placed on the z-axis vertical travel of the PPP base. The haptic control system is integrated into a 3D Python simulation system. User move or manipulate the haptic controller end-effectors and orientation and position sensor check this pose and check for its kinematic and singular state, where the servo-motor would control the motion when singularity is found in that path. The Python simulation checked for non-singular condition and passed the coordinate and orientation value to the Matlab and Solid Work system. Matlab validates and then control a physical Hybrid Robot.  The haptic controller can have a direct control of the Hybrid Robot, or it can confirm the pose by checking this against the Matlab verification. The control features include the interpolation and extrapolation strategy to determine the optimize path. The interpolation strategy uses L-system triangle system like Sierpsinki. The extrapolation strategy use Trilinear and Simplex 3D to solve the problem.

## 8.2 The haptic controller kinematics

The haptic controller is a simple kinematic device with 3-axis Cartesian at the first layer and 3-axis Spherical RRR unit at the second layer. Bonev explains that Cartesian structure has similar accuracy with different position and stroke, therefore there is no optimal design parameters for Cartesian design [128]. The Rotary unit is coupled to the Cartesian axis at one axis. The design objective is the development of a 6 degree-of-freedom structure that would allow two-way motor feedback, and provides the sensation or feeling of operating a single plane in space, where the intuition of moving the linear and rotary axis is coupled properly.

Dash wrote that leg symmetry is an advantage, due to uniform force distributions. Less number of legs reduces the risk for interferences. It is recommended that the actuators are arranged symmetrically among the limbs. Actuator’s gearing and servo motor contributes towards weight, leg interference and bulkier size [15].

Gregorio developed an algorithm that determines all assembly classes for a SP-PS-RS architecture [129].According to Herve, there are many design variations; however the design objective is to reduce the moving of masses or limbs, to place majority of loads (including servo motor and bracket) at the bottommost layer and the range of motions. The structure for two-ways servo control should also consider gravity effects and the servo motor ability to provide braking [19]. Following Gosselin design, the RRR rotary unit design has 2 coplanar condition and the axis for all plane should intersect at the centre of the end-effectors’ rotation axis [131].

Being a haptic device, the actuator must allow for user-exerted force to move it, and it should be able to move by itself. The linear motion links the joint at the base and the travelling plate. The linear motion has to be gear-driven, and the gearing should allow for external exerted forces to have effect on its motion. The motor should have adequate braking power to stop the external forces from having any effect on the system, when the system reaches singularity point. In this case, it is concluded that the best method is the rack and pinion gearing, which allows two way feedbacks.

## 8.3 Kinematic formulation for the haptic controller

Kinematic describes the rotation and linear transformation that create robot's motion. Equation 34 gives the desired motion.

$$[T] = [T_r] \times \left([R_\theta] \times [R_\emptyset] \times [R_\varphi]\right) \tag{34}$$

Where,$T_r$,$R_\theta, R_\emptyset$ and $R_\varphi$ are the translation and rotational matrices around X,Y and Z. The coordinate is fixed to the fixed robot's base.

$$[R_\theta] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\Theta & sin\Theta & 0 \\ 0 & -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{35.a}$$

$$[R_\emptyset] = \begin{bmatrix} cos\emptyset & 0 & -sin\emptyset & 0 \\ 0 & 1 & 0 & 0 \\ s\emptyset & 0 & cos\emptyset & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{35.b}$$

$$[R_\varphi] = \begin{bmatrix} cos\psi & sin\psi & 0 & 0 \\ -sin\psi & cos\psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{35.c}$$

Equation 36 gives the whole transformation matrices.

$$[T] = \begin{bmatrix} c\phi c\psi & -c\phi s\psi & s\phi & 0 \\ c\Theta s\psi + c\psi s\Theta s\phi & c\Theta c\psi - s\phi s\psi & -c\phi s\Theta & 0 \\ s\Theta s\psi - c\Theta c\psi s\phi & c\psi s\Theta + c\Theta s\phi s\psi & c\Theta c\phi & 0 \\ l & m & n & 1 \end{bmatrix} \tag{36}$$

Where,$\theta$,$\emptyset$,ψ are defined rotational components and l, m and n are linear motions in X, Y, Z direction. And, s and c represents sine and cosine function.

Since, the haptic structure is based on simple rotation and linear transformation matrices; it works with various parallel robots' geometry and configuration.

## 8.4 The Haptic structure design

The experimental 6 degree-of-freedom haptic controller is shown in Figure **8-1**.

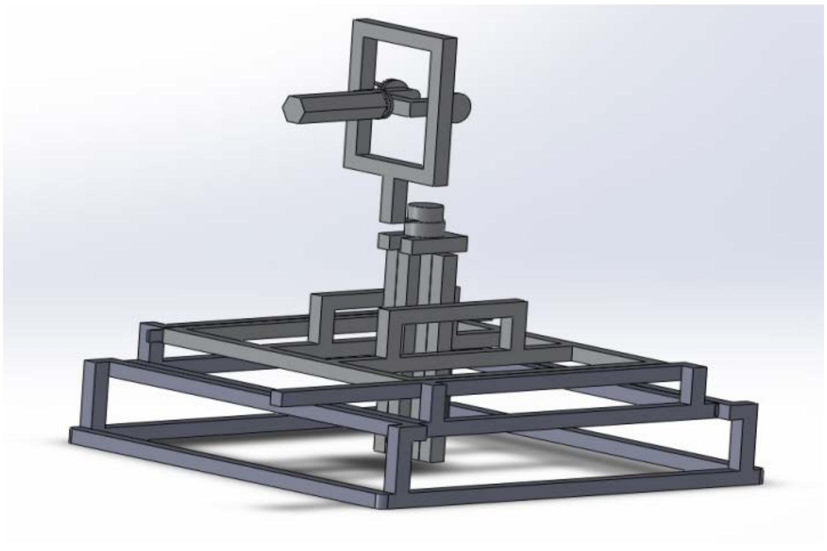
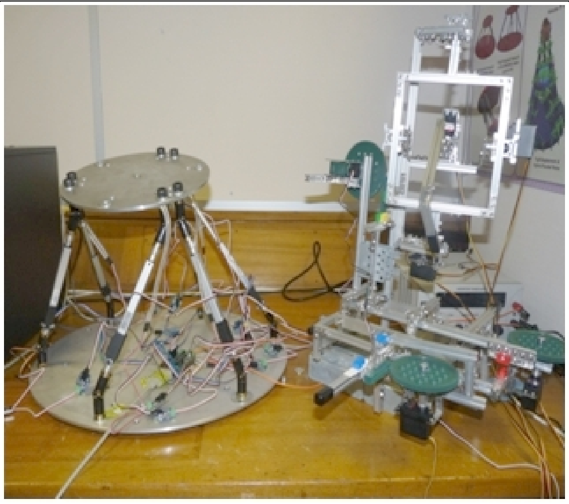
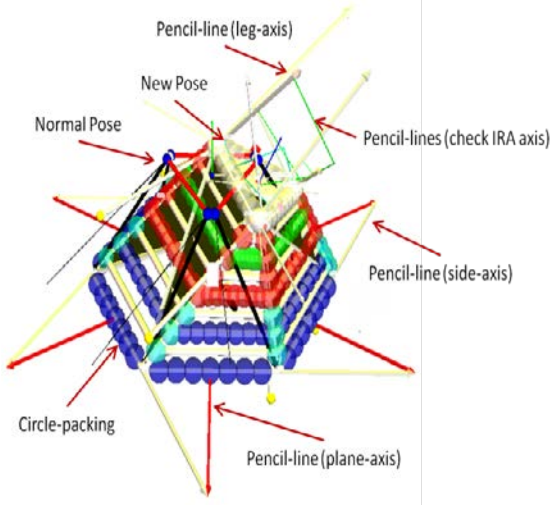


Figure 8-1 Model of the Haptic Controller (PPPRRR)

This PPPRRR configuration is chosen due to the intention of controlling a single point in 3D space. Similar method to this arrangement is the Agile-Eye , however the construction is complex and involves curvy extrusion [132]. The rotary joints are placed in the centre of the linear translation axis. The rotary axis is similar for all axes. Both the rotary and linear axes are extendible and can be limited by software or physical constraint.

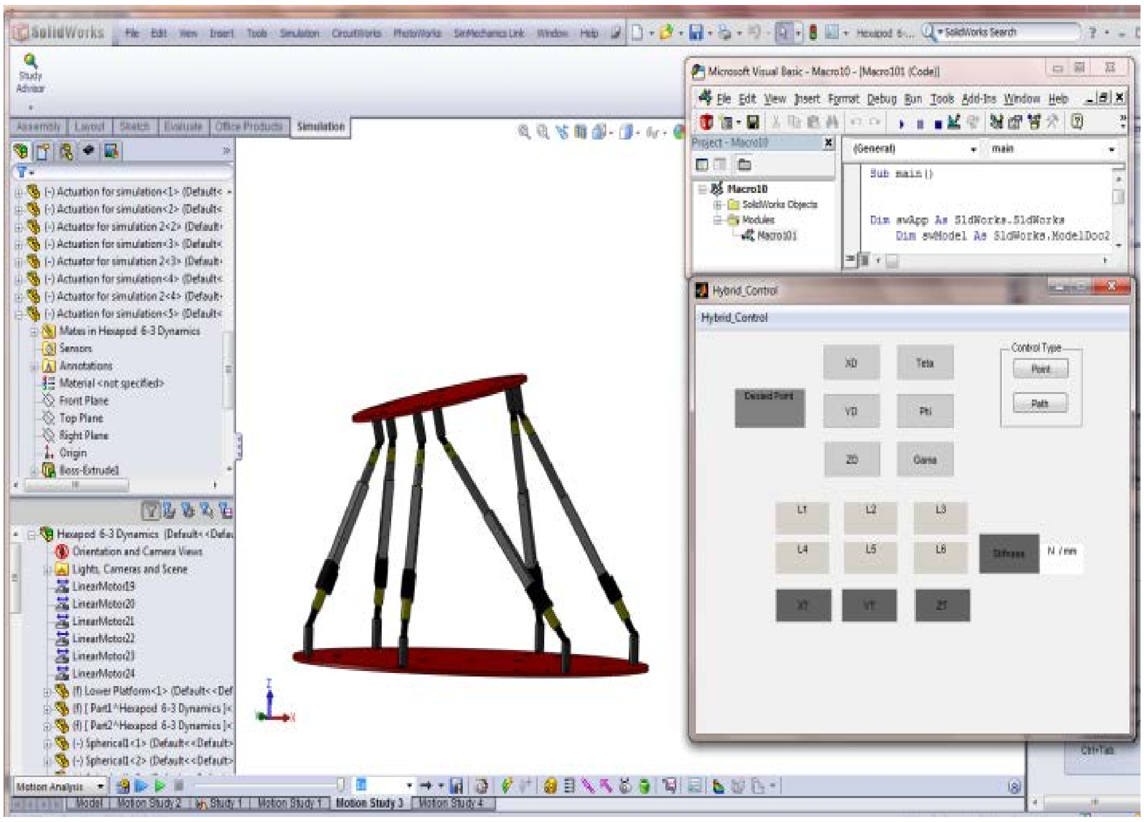
The haptic controller is composed of extrusion beams, rack and pinion gear, and servo motor, IMU sensor for orientation sensing and RGB camera for position sensing. The haptic Controller is linked to the 3D Python simulation via the orientation and position sensor. The 3D Python simulation can control the haptic controller's end-effectors. The haptic controller shown in Figure 8-2can either link to the Matlab Solid Work numerical system, or control the physical robot (shown Figure 8-2) directly with singularity check performed by the 3D Python system.

Table 9 Python controller and the robot simulator

| | |
|---|---|
|  Figure 8-2(a) the physical robot(left) and (b) the haptic controller(right) |  Pencil-line (leg-axis)<br>New Pose<br>Normal Pose<br>Pencil-lines (check IRA axis)<br>Pencil-line (side-axis)<br>Circle-packing<br>Pencil-line (plane-axis) Figure 8-3 Modality guide (3D Python simulation) |

## 8.5 Control and Validation

Haptic data is sent to the control program (MATLAB program) to calculate the size of the actuators for parallel robot. An associated CAD model simulates the given motions. A snapshot of the Solid Work system is shown in Figure 8-4. The result shows that the position of the numerical ends effectors matches the haptic data (refer Figure 8-6).



```
Sub main()


Dim swApp As SldWorks.SldWorks
    Dim swModel As SldWorks.ModelDoc2
```

Figure 8-4 Matlab as a numerical system that validates the haptic engine experiment

The PPP RRR topology fulfils the geometry selection criteria because it provides natural feedback that user experiences when controlling the travelling plate, the topology provide simple kinematic for solving the end-effectors position and orientation and the structure allow for reconfiguration of each parts without the need for major modifications. Furthermore, identification and tracking of position and orientation of end effectors, and many components is achievable by using camera based sensor. The design objective is to develop a structure that allows for large linear travel and large rotational angles for all axes. The linear and rotary travel can be limited later by adjusting the limiting component or by software control. The structure should be easily extendible to reduce or enlarge the workspace depending on user requirements or the robot workspace dimension or shape.

User interference or blocking of sensor's view should be compensated by software. The research attempt to develop a plane-based control rather than single-point based interaction of haptic control.

Here, the position provided for the end-effectors by Python simulation is $P = (36 \quad 70 \quad 955 \quad 5 \quad 0)$. The MATLAB program, which is based on the developed kinematic map of the mechanism, reproduces the motion. Figure 8-5 shows a close match in X, Y, and Z directions between the results obtained by the proposed methodology and those obtained using the theoretically-based numerical method. Numerical method verified that the Python kinematic position is accurate and valid.Figure 8-6 displays the position's validation result, when compared with a Numerical system.

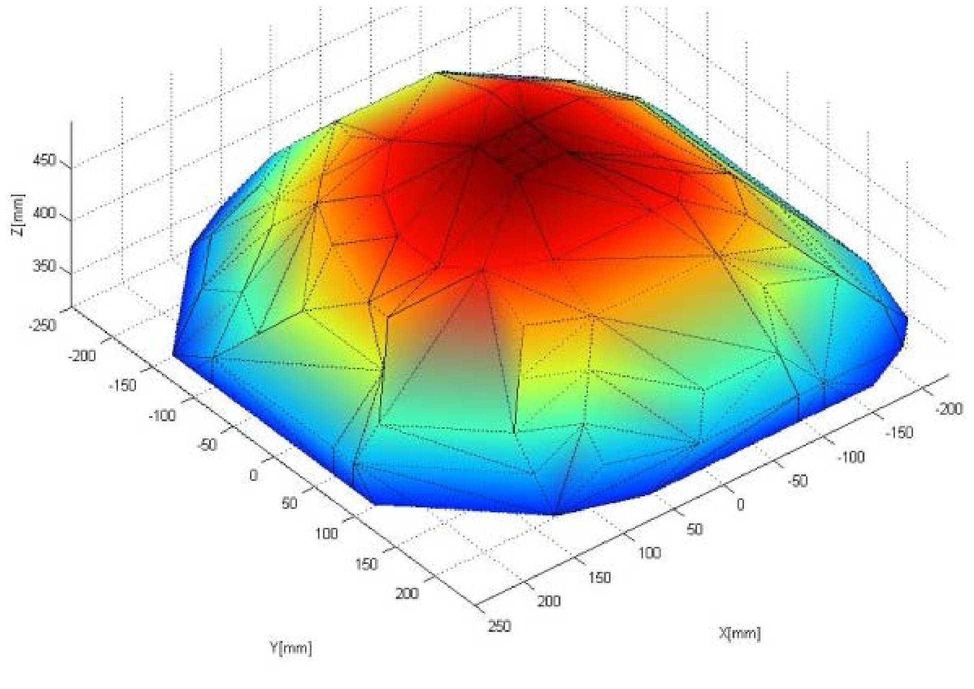


Figure 8-5 Numerical workspace based on cubic parametric sweep

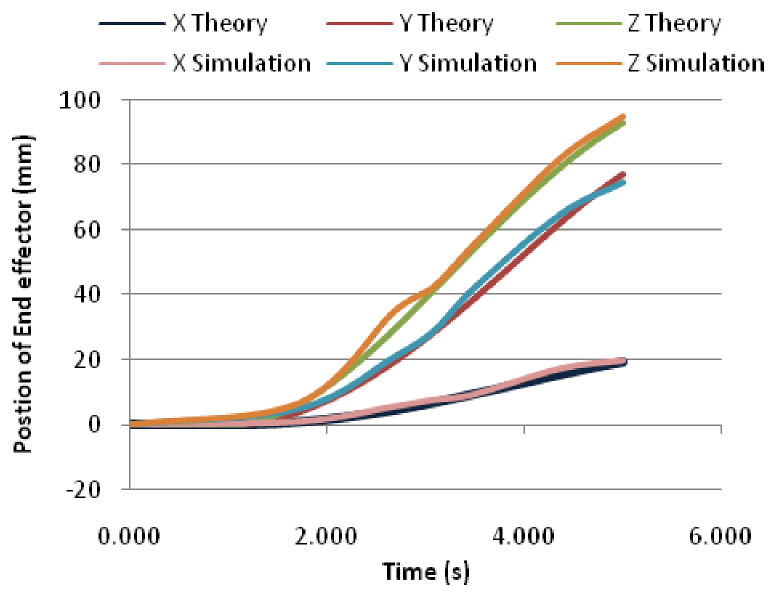
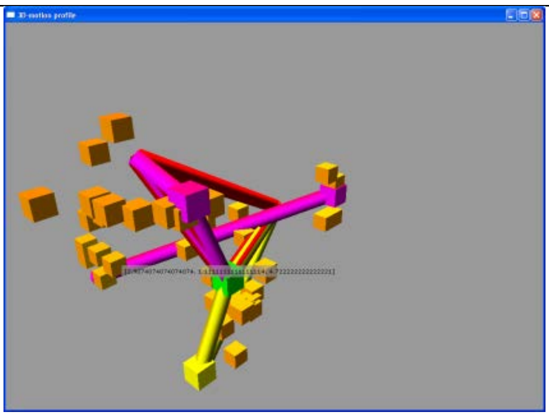
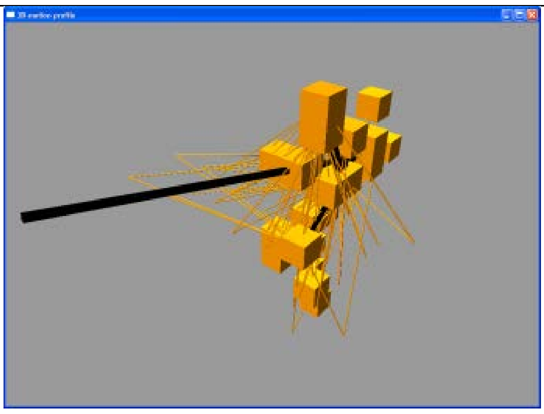


Figure 8-6 End effectors tracking position

## 8.6 Haptic interaction method using Simplex

Downhill simplex or Nelder-Mead method places an active moving simplex inside a design space, where it expands contracts and reflects (flip) around a point (optimal point). This process continues until it reaches a specified error tolerance value. This method requires the establishment of a guide or initial path, for the system start its data sampling distribution. First, produce Simplex seed at starting point, and it will grow to estimate the next best pose. The best pose criteria are based on weighted ranking of geometric singularity and Grassmann error. Additional criteria is cost-factor, gradient trajectory changes and time. Additional criteria from outside sources (Numerical methods) include stiffness and force. Figure 8-7(a, b) demonstrates Simplex result for different setting. (Refer Table 10)

Table 10 Comparison between different settings for the objective function

| | |
|---|---|
|  Figure 8-7(a) Example Python rendering of simplex data sampling where search direction is moving towards a minima<br><br>Based on objective functions(abs(args[0] * args[0] * args[0] * 5 –args[1] * args[1] * 7 + math.sqrt(abs(args[0])) – 118)) |  Figure 8-7(b) Contracted simplex (concentrated towards the center)<br><br>Based on objective functions (abs(args[0] * args[0] * args[0] * 9 + args[1] * args[1] * 9 + args[2] * args[2] * 9 + math.sqrt(abs(args[0])) – 118)) |

## 8.7 Haptic interaction method using Voxel

3D Voxel-based planner is based on nearest neighbourhood interpolation, which is used to replicate and extend the concept for parallel robot path optimization by using Marching cube model (Refer Figure 8-8). The methodology for seeding the 3D marching cube path is based on Verhoeff 3D binary tree development, which replaces the basic geometry seed generator object with marching cube, and the ridge functions is replaced by the 10-variations of marching cube orientation seeding element.

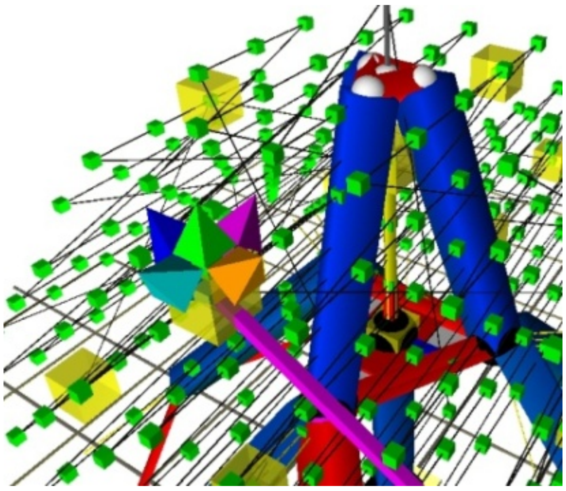


Figure 8-8 Example Python rendering of 3D Voxel-based planner

## 8.8 Haptic interaction method using ternary and binary tree

The two major types of fractal tree are, namely, binary tree and ternary tree. Binary tree produces more natural-like shape and distribution of test points over a wider range of region, with some opportunity for directing the tree back towards the seed. The turn angle is between 60° to 120°.While ternary generator produce a forward distribution only (Refer Figure 8-9). The experiment on ternary and binary tree is a huge concept; therefore the works done on this subject is just to demonstrate the idea. The Python demonstration has produce reasonable result, which provide opportunity for various new research on this subject in the area of parallel robot workspace research.

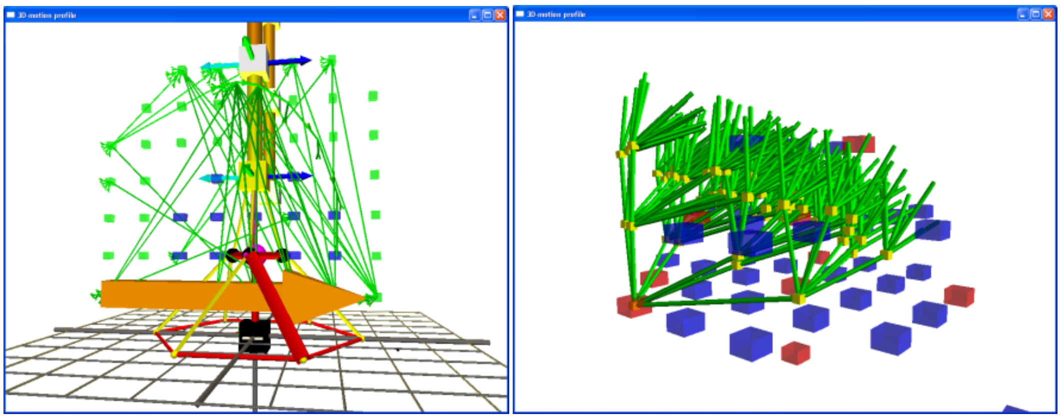


Figure 8-9 Example Python rendering of ternary tree

## 8.9 Sensors for detecting position and orientation

Sensors are required for measuring the orientation and position of the haptic controller's end-effectors. The sensor must not obscure the existing structure, and is best placed away from the structure. Phidget 3-axis Accelerometer and X-IO 9DOF IMU is used to measure, calibrate the orientation data. RGB Camera is used to detect and track 4 dominant colour markers. Detail discussion on measuring, calibration and integration is given in the next section.

The calibration for structure is essential in ensuring that the position and orientation data is not being affected by structural bending or alignment error on any axis. The servo motor position, the pinion position on the linear axis, and the joint or contact position for ball-bearing units or structural joints has effect on the structure deformation or bending, especially when the upper-structure load distribution is focusing on a region. The motor force has effect on structural deformation during dynamic motion. The PPPRRR structure has its own problem, where the RRR rotary units and the vertical linear axis unit is the load to be distributed along the x and y-linear axis.

The sensor for position and orientation has to consider many factors like user interaction with the device, itself. User approach towards controlling the haptic device would introduce interference (blocking the camera view) or produce shadow-cast on the structure and marker. The vision tracking sensor (RGB camera) can track four markers at one time, however, the camera sensitivity towards certain colour marker together with the effect of shadow, environmental light changes has dire effect on the sensor readings. The X-IMU 9 Degree-of-Freedom (9 DOF) sensor has its own problem, where without the embedded Madgwick special calibration code hidden inside the microprocessor, there is problem with the Pitch, Roll and Yaw value, where the value is only useful within a range, and not within the gimbals lock position when the system suddenly flips direction.

## 8.10 The experimental setup

Human has the ability to sense in other ways than visual alone. The haptic sensation highlights the singularity position in 3D space. The experiment with object collision with user is a common evaluation for haptic system. Two spherical shapes are placed inside the workspace, and user is given the task of finding the usefulness of the workspace minus the occupied area by these two objects. This task is not suitable for a 2-dimensional input device, and difficult for vision to memorize the state and the 3d visual properties like direction, curvature and location in space. Spring theory is included to provide the linkage condition between the end-effectors and the two spherical shapes. Three methods have been implemented for the Python system, which are Downhill Simplex, Voxel planner and L-system slicing method [69, 75, 134].

## 8.11 The IMU sensor for detecting orientation

Accelerometer and IMU are known to exhibit drift, which is a noise factor where the sensor data accumulates over time, even at stationary state. This drift can be reduced to a minimum by continuous check for sensor value changes of a defined threshold. The Python system records 3 datasets, then check for value changes, if exist, and then use the final set for controlling the simulation. Figure 8-10 display the range errors for each quaternion matrices value represented by a2[0,0], b2[0,1], c2[0,2], d2[1,0], e2[1,1], f2[1,2], g2[2,0], h2[2,1] and i2[2,2]. The range of error is acceptable therefore this is not compensated when calibrating the haptic structure. The drift value has been effectively reduced by active checks for value changes during data acquisition.

Each yaw, pitch and roll value needs to be combined to provide the final orientation to the travelling plate. Kinematic use the value individually, therefore there 2 ways of producing the 3-dimensional plate orientation. The transformation matrices or kinematic produces 3 by 3 matrices which is then multiplied to the current vector coordinate to get the next pose vector coordinate. This method would require kinematic validation for all sensor inputs, which is an extra process. The kinematic validation can be part of the calibration process, this way; the real-time 3D simulation can skip the kinematic validation for continuous position and orientation data acquisition. This method requires complimentary Filter, which are parts of the drift error correction filter (Refer Figure 8-10). The Complimentary filter performs two different functions here, first by reducing drift error, and secondly it provides orientation data by using accelerometer and gyroscope data to get the orientation data. Complimentary filter is a combination of high pass (and low pass filter and is described in this formulation, where low-pass filter use accelerometer data, high pass filter use the gyroscope data (Refer [10]).

$$y[n] = (1-\propto) \times y[n-1] + (1-\propto) \times (x[n] - x[n-1]) \qquad (37)$$

$$y[n] = (1-\propto) \times x[n] + \propto \times\, y[n-1] \qquad (38)$$

$$\text{Where} \propto = \tau \times (\tau + dt), dt = 1/sensor\_frequency \qquad (39)$$

$$angle = (1-\propto) \times (angle + gyroscope \times dt) + \propto \times\, accelerometer \qquad (40)$$

Where x[n] is the raw gyroscope value, Y[n] is the processed value, n is the dataset index value,∝ is a time-constant, and τ is the desired time-constant and *sensor frequency* is the sampling frequency. Figure 8-11(where Yaw (angle) and Gyro (y) represent the IMU data output) demonstrate the data delay during acquisition. Therefore, timing for collection of dataset is required to synchronise the output and use these again for Complimentary filter to derive the Roll and Pitch value [28].

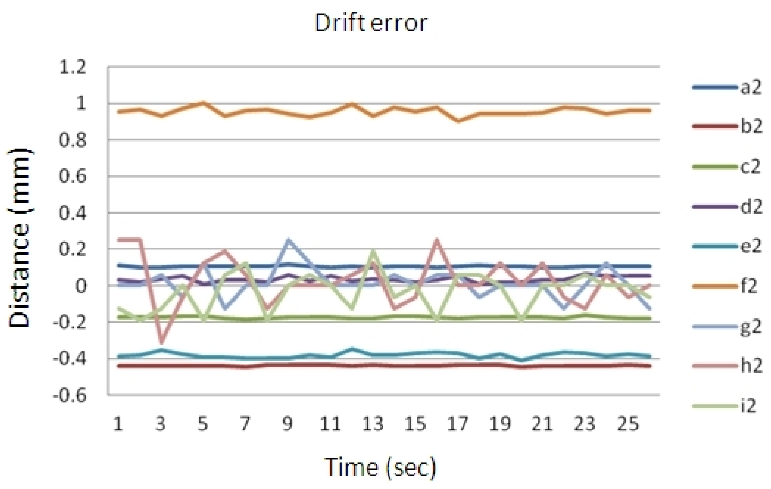


Figure 8-10 Drift errors at Normal position

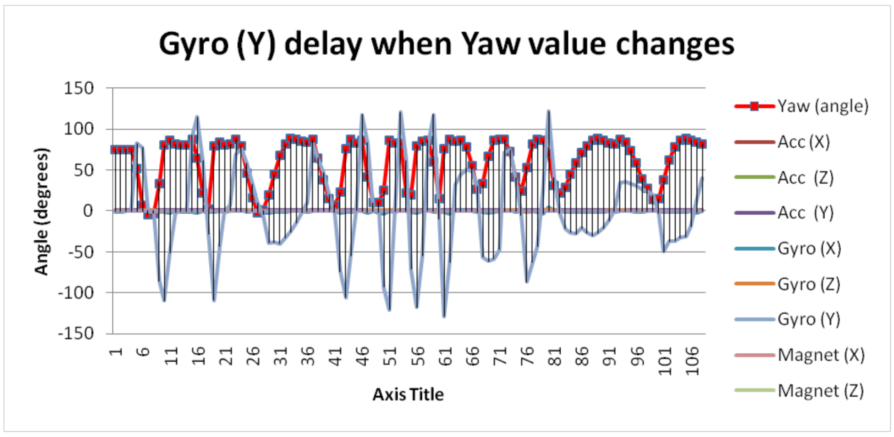


Figure 8-11 The Gyro (Y) delays

## 8.12 Yaw North Compass noisy output

The accelerometer and IMU provide raw data, which requires computation to derive the yaw, pitch and yaw value. Where pitch is the North compass forward direction tilt, Roll is the side roll of the plane's body, and Yaw is the North compass direction. Complimentary filter provide pitch and roll, and combination of filter like Kalman or Complimentary reduce the drift errors [134]. The Yaw north Compass is a difficult value to derive. The research follows Madgwick algorithm to derive all Yaw, Pitch and Roll [29]. However, some of the computation is not disclosed for public uses, therefore the derivatives does not provide a reliable values. Figure 8-12 demonstrate the output from the 9DOF IMU based on Madgwick algorithm (where the values represent their place inside the matrices, where m00[0,0], m01[0,1], m02[0,2], m10[1,0], m11[1,1], m12[1,2], m20[2,0], m21[2,1] and m22[2,2]), where the pitch angles varied from -20 to 160 degrees, and the yaw and roll value is fixed. The Figure 8-12 displays the data consistencies and moo, m10 and m20 is a structure error instead. The values for m00, m10 and m20 should remain fixed for this test.

The errors could come from the servo stepping intervals, where each stop and new motion produces a structure jitter.

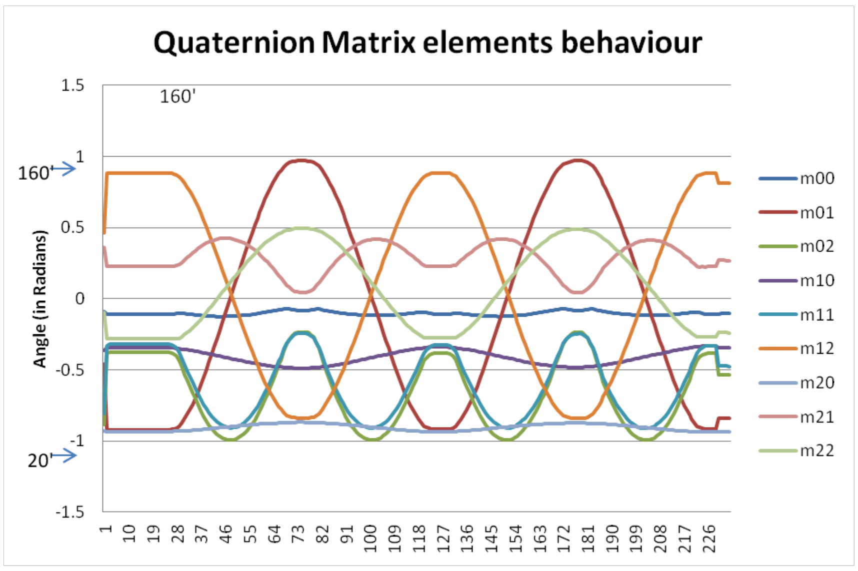


Figure 8-12 Quaternion elements result for Pitch angle between 20 to 160 degrees

Figure 8-12 (where the values represent their place inside the matrices i.e. m00[0,0], m01[0,1], m02[0,2], m10[1,0], m11[1,1], m12[1,2], m20[2,0], m21[2,1] and m22[2,2]), displays the problem with the IMU, where although each Roll, Pitch and Yaw produce accurate values, but when combined, one change will affect other values. In Figure 8-13, a Yaw changes produce different range for Pitch value 0 degrees and 170 degrees. The sudden changes seen at index 81 and 231 are Gimbals Lock, a condition where the system flips side due to the uses of positive and negative angles in the mathematics.

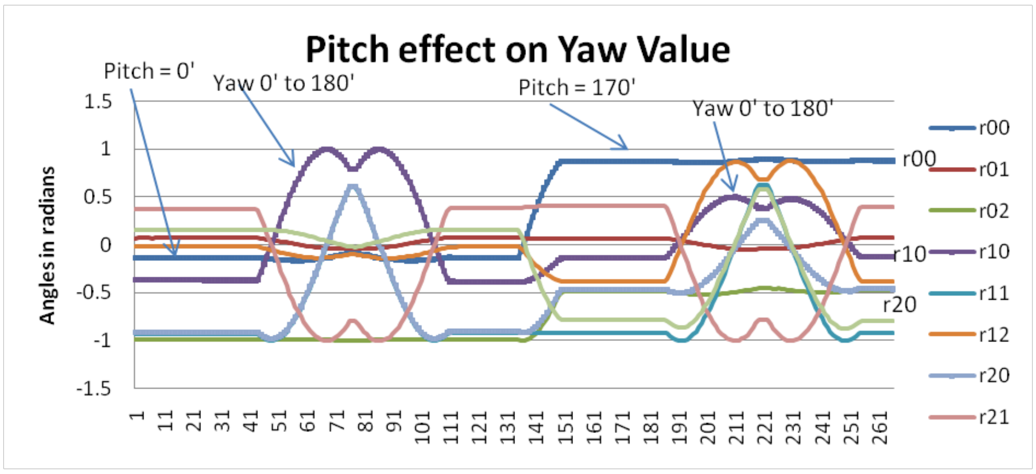
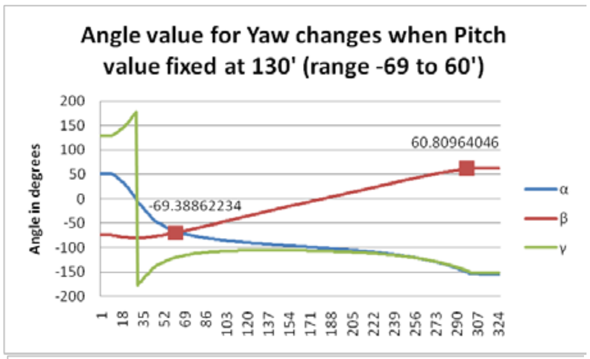


Figure 8-13 A studies on combination effects for a Yaw Value test

Figure 8-14(a, b) (where α, β and γ represent rotation axis) demonstrates the problem when the 9DOF IMU is posed at a certain orientation, where the orientation has Pitch and Roll value. Therefore, the outcome from the post-processed data from the 9DOF IMU is not useful for sensing the end-effectors orientation. The IMU data is coming from a C# pipe which is timed to push a set of data to Python. The Quaternion Yaw and distance calculation is done on C# therefore reducing the burden on Python. Python perform complimentary filter processing on the Accelerometer, Gyroscope data. The data set consists of Accelerometer, Gyroscope, Magnetometer, and Quaternion Yaw angle (Eq.41) and distance data.

$$\text{Quaternion Yaw} = -\operatorname{atan}\left(2 \times (q1 \times q3 + q0 \times q2)\right) / \sqrt{1 - (2 \times q1 \times q3 + 2 \times q0 \times q2)^2} \qquad (41)$$

Where Quaternion Yaw is the north Compass direction, q0, q1, q2 and q3 is Madgwick Quaternion output.



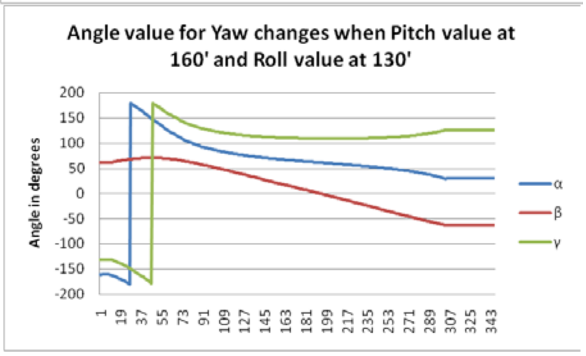


Figure 8-14(a) Yaw Value for a fixed Pitch value, and (b) Yaw value for a fixed Pitch at 160’ and Roll at 130’

## 8.13 RGB camera for detecting linear translation

The haptic structure is a free-motion structure. A physical attachment of distance sensor would limit the motion, or would provide further complication to the structural design. Kinect is first investigated. However, due to its embedded filter algorithm which prefers detection of limb or large blobs, but not small diameter colour marker, is not used to detect and track the marker or blob. RGB camera detection and tracking has no marker limit, and satisfy the objective of having a flexible sensing system that works in changing lighting condition where the hue, saturation and object brightness will change over time. Figure 8-15(where peak1,2 and 3 represent the dominant colour, and rr, gg and bb represent Red, Green and Blue RGB Colour, and polynomial represent the average for each colour) display the colour peak (dominant colour) change its value during the platform motion and user interactions, when casted shadow and structural blocking of camera view. Also, marker sensing should be based on colour cluster. K-means histogram definition and primary colour identifications.

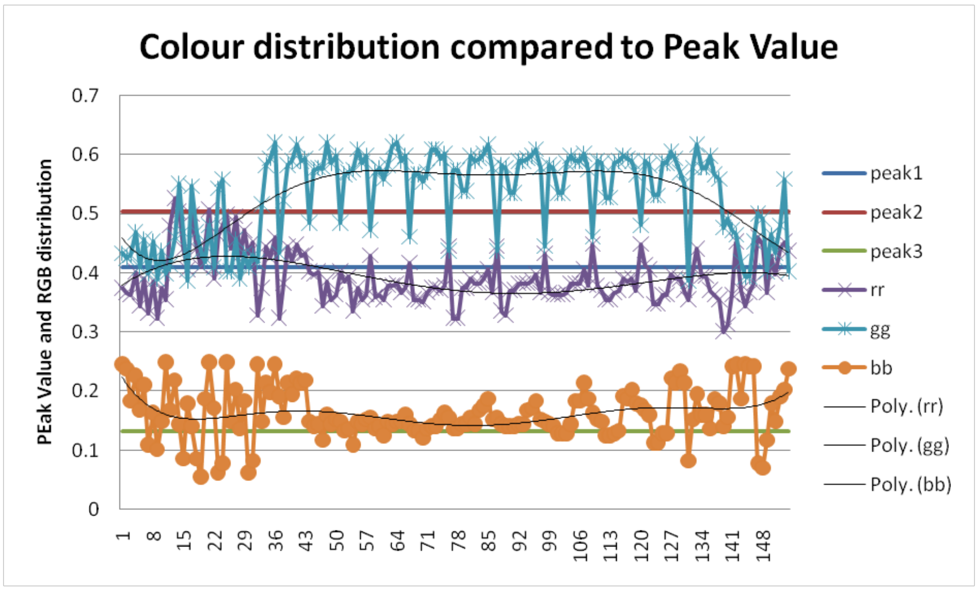


Figure 8-15 Colour distribution and peak value

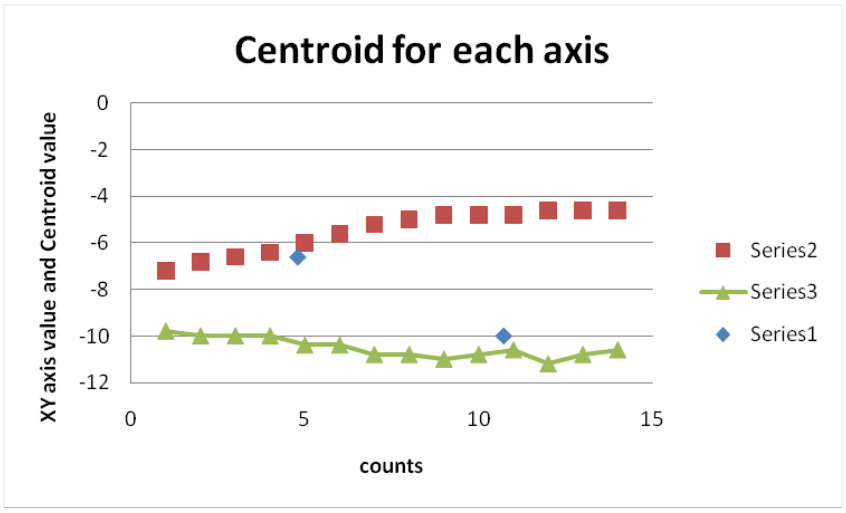


Figure 8-16 Dominant colour Centroids within each Cluster

RGB camera’s detection and tracking can be implemented in Python simply by using ready-to-use package like OpenCv and SimpleCV [136, 137]. However, this would require import of various packages with limited availability to a certain platform. And, Python package may contradict valid version useful for other components required to run the whole 3D simulation. The primary Python Visual package has problems loading both openCV and SimpleCV for a real-time runs. The simulation make use simple Python Video Capture package which is low memory and very limited in function, but adequate for Python PIL package to provide image processing. The calculation for detection and tracking is done using Python NumPy package.

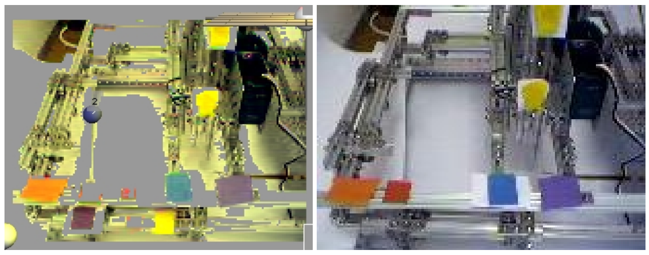
The strategy is to load low resolution image captured at slow intervals, where the image is given convolution filter. Bicubic convolution produces good contrast and faster processing. Counts for dominant colour cluster for Antialias is 362 and 361, bilinear counts are 351 and 408, and Bicubic counts are 161 and 405. The next step is to define the histogram or cluster for the primary colour which is not in the dark colour region or clear colour region. The low and high threshold for removing the background is defined as low value in the range of 10, 35 and 60 RGB value, and high range as 200, 230 and 250 RGB value. The system then seeks for peak or cluster colour regions centroid positions (Refer Figure 8-16 where Series 2 and 3 represent Colour cluster and series 1 represent the centroid position). This centroid position defines the linear vector distance value for the 4 markers position in space. The checks for centroids need to be optimised to reduce unnecessary check for non-related distance. The comparison strategy control makes use of Boolean gate to keep the range of centroids to a minimum 4. The Boolean gate run check for the centroid belongs to the required range and then check if the cluster represent horizontal or vertical axis.

Based on K-means algorithm we can find the centroid, and placed related data into its centroid class $c_{i,}$ then repeat for the predefined threshold to eliminate background from the targeted markers. (Refer Figure 8-16)

$$Marker(i) = \sum_{i=1}^{k} \sum_{x_j \epsilon c_i} \left(x_j - |\mu_i\right)^2 \qquad (42)$$

Figure 8-17(a) (where the left image represents the post-processed acquired image with background noise, and on the right is the test platform with colour markers) displays the background problem. This can be seen in Figure 8-17(b) (where Red, Green and Blue represent RGB Colour value) as index number 1, 4,5,6,7 and 8. While a mix problems is identified in Figure 8-17(b) at index number 2, 10 and 11. The dominant colour cluster is found in Figure 8-17(b) which is index number 3 and 9. The Boolean gate reduces the background problem by scoping the histogram range differences (where Red-Green-Blue RGB value is more than a X threshold) and the Unit found inside each dominant colour cluster should be lower than a Y threshold to qualify itself as a marker.

The distance between centroids is calculated based on the vector elements representing each horizontal x and y-axis and vertical elements representing z-axis1 and z-axis2. The Python PIL system is slower than SimpleCV and OpenCV, but it has the advantage of reducing the requirement for various Python packages. The strategy for improving the processing speed is by acquiring low resolution image and reducing the bit size as thumbnail.



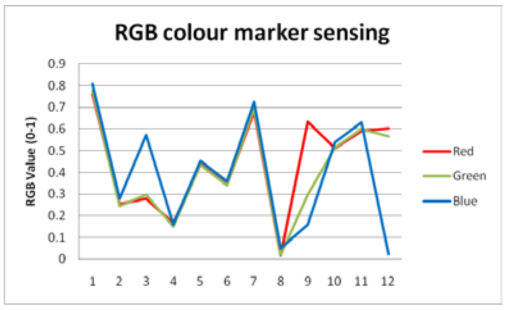


Figure 8-17(a) Background noise in the RGB data acquisition, and (b) Noise due to marker colour range is within the background colour range

Red peak data shows that markers yellow, red and green can accommodate the peak range, however for blue marker, the peak value rarely within range (refer Figure 8-18).

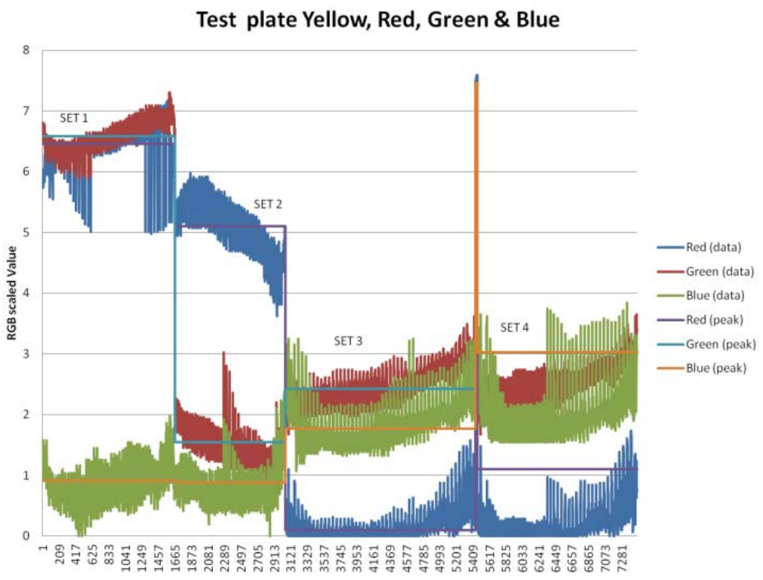


Figure 8-18 Test plate for marker colour Yellow, Red, Green and Blue

## 8.14 Haptic structure calibration

The haptic orientation sensing is based on IMU sensor, and the sensor is measures the structure error included into the orientation result. The objective is to reduce this structure error and compensate for the errors, or put forward a strategy to reduce its impact.

The haptic structure is based on both horizontal planar x and y axes. This forms the base platform and should remain flat while the y-axis plane which is placed on top of the x-axis travels along the rail. The servo motor should provide smooth travel and the start and stop should not collide with other structure. The control should provide a buffer space of 1 mm before the system reach its limit or hit a structure. The vertical linear z-axis should provide vertical travel with no structural collision and ability to sustain consistent contact with the supporting column. The structure should be able to support the 3RRR weigh when it change orientation. The vertical structure should remain fixed to its rail along its travel axis. The top 3RRR components should remain flat when it’s rotate on the z-axis vertical part. The Pitch component is placed inside a bracket that allows Roll. And the bracket is placed on Yaw North compass component that is attached to the vertical z-axis component.

Figure 8-19 (where X1, Y1 and X2, Y2 represent the displacement error detected at slow and fast servo-motor speed for x-axis test) demonstrates linear x-axis translation error, which range between -1.8 to 0.8 degrees. X and Y represent high motor speed, and X2 and Y2 is slow motor speed.

Figure 8-20 (where X1, Y1 and X2, Y2 represent the displacement error detected at slow and fast servo-motor speed for z-axis test) displays the effort required for the servo motor to push and bring down the 3RRR components, and having to move within the allowed frame. In order for the servo motor and the vertical rail to work, it is required for a minimum buffer where the structure can change its position and orientation on the constrained axis in order for it to work properly as a force-feedback structure. This component is the hardest to design, and to find the optimum condition where user can exert force to it, and it can drive itself upward s and downwards consistently.

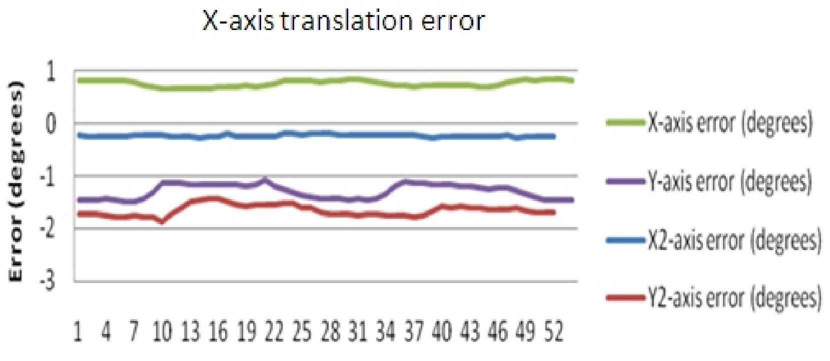


Figure 8-19 Test result for linear X-axis error

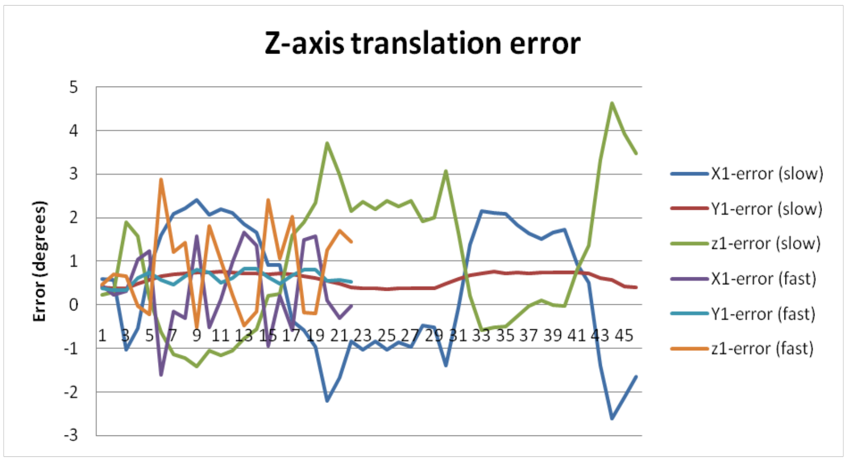


Figure 8-20 Test result for linear Z-axis error

## 8.15 Case studies for validating the Haptic interaction

This section discusses the procedure for validations of the haptic interaction. The objectives are to reduce the risk of errors like unstable probe, vector direction and fake state. The case studies demonstrate the concept for the implementation of the Python methods of using Boolean control for complex algorithm.

### 8.15.1 Case 1: Two spherical shape in the workspace

This haptic experiment demonstrates the haptic simulation where two spherical objects is placed inside the Robot's workspace. The end-effectors should not travel inside the objects, but may slide through its surface contacts (Refer Figure 8-21, where two yellow spherical shapes are shown, and the Red dots represent the effective workspace outside the obstacle region). The mesh definition is based on Simplex, parametric grid or Grassmann. Simplex builds a dynamic grid that contracts and expands when attempting to detect the nearest distance points between the end-effectors and the mesh. Parametric grid is based on any grid structure that is placed strategically on the mesh surface, where closest distance calculations determine the object and obstacle relationship. The Grassmann concept is based on the calculations of coplanar; line meet and line join operation.

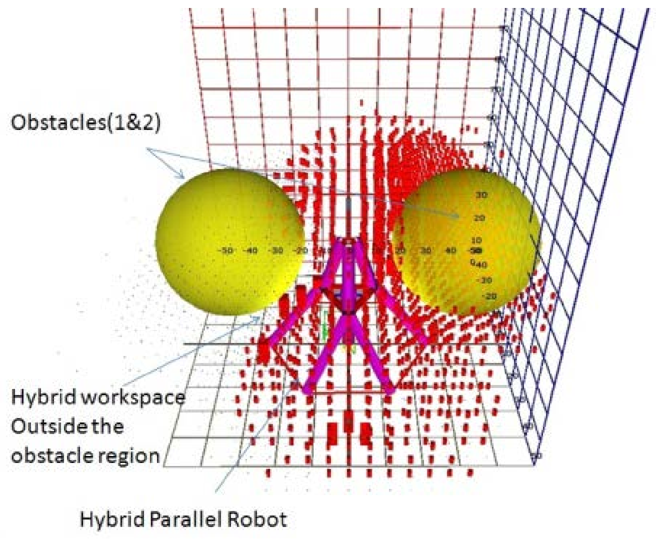


Figure 8-21 Example Python rendering of an obstacle region

### 8.15.2 Case 2: Simplex mesh in the workspace

This haptic experiment demonstrates the haptic simulation where a mesh surface is placed in the workspace, where three surface-haptic algorithms shall be tested here. The key issue is identified as gradient force and edge extractions. The next issue is both side of the interaction, which are the probe and the object, where range, detection and sensitivity could produce false sensation (Refer Figure 8-22). The mesh is user-defined, and can be strategically placed in the 3D space. There is no limit to the amount of mesh within the work area. Relationship between meshes can be defined by using 3D interpolation, which produces adjacency graph. This setup can be used for determining compliance workspace, which ensures that the end-effector is always facing the mesh surface as it travels across the mesh. Compliance map can be defined with different resolution, and different algorithm for checking the nearest distance between the subject and the target.

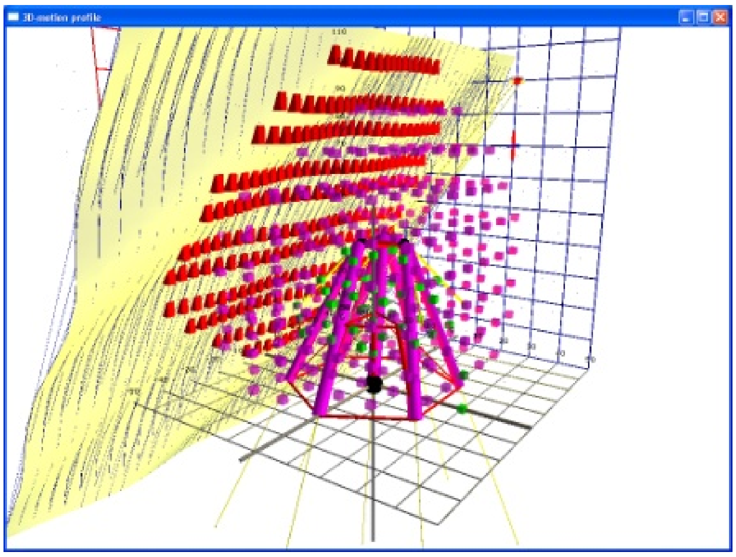


Figure 8-22 Example Python rendering of haptic sensation simulation

## 8.16 Conclusion

This chapter discusses the development of a haptic controller, which is a force-feedback device that helps the user to identify singularity regions in 3D space. The haptic structure is selected based on simple kinematic, shared three rotational axes centre-points and simple three translational axes. The structure has been tested for the control of an existing physical hybrid robot in the laboratory, and with an existing numerical system developed using Solid Work and Matlab. The haptic structure is proven successful in performing the above-mentioned task.

Next, the development of 3D haptic engine, which allows for force-feedback correction of 3D poses of the end-effectors, was presented. The haptic engine is planned for static pose and dynamic pose which produces 3D path. A few 3D haptic engines have been tested, for example the Downhill Simplex, 3D Voxel and ternary tree [138, 139, 140, 77, 141]. The ternary and binary tree method is easier to control and command. The result shows satisfactory confirmation of the concept, where the region, direction and search pattern can be defined by user.  The engine which produces adjacency tree can then be integrated with 3D path planner to drive the haptic controller.

The IMU sensor proves useful in the development of the haptic controller, by providing the angle data. However, IMU is also complex and produces too much noise which increases over time. The RGB camera data with clustering technique produce fast and reliable multiple tracked point data.

Haptic interaction case study has been presented and discussed in this chapter, where simplex mesh drives the mesh surface interaction with the end-effectors. The simulation shows a good confirmation of mesh detection. The force-feedback interaction is limited to braking, acceleration and completely.

# Chapter 9: APPLICATION OF PARALLEL ROBOT

## 9.1 Introduction

This chapter presents and discusses three different application scenarios, including control strategy based on Bezier method, path planning based on dynamic force and velocity, and Path’s slicing analysis as a therapist’s intervention tool in robotic ankle rehabilitation. The applications demonstrate the Python simulation system capabilities. The applications have been developed for validation purposes, where a numerical system checks and validates the results. They also demonstrate some features from the Python simulation such as smaller $C_{space}$, Grassmann algebra, Boolean control and fractal data population methods.

Bezier curve has features for approximation, fine control and parameterisation, which let user modify the path to achieve an objective. The search for control points to approximate the given path, also produce control points and error-correction scheme for that particular formulated path. A few concepts integrated for the Python simulation has been tested and validated using numerical system. The method like Bezier, ternary and 3D iteration like parametric sweep reduce the complex problem into smaller and focus region, where lower resolution test is applicable.

## 9.2 Application 1: Control strategy based on Bezier method

This section deals with the investigation of the end effectors' tracking position for a 9-DOF hybrid parallel robot. The structure contains 6-DOF and 3-DOF parallel robots connecting serially to each other. In the proposed method, the best configuration of the robot to reach a desired position in the workspace has been found to match accordingly to the developed stiffness and inverse dynamic of the system. The proposed developed network allows the robot to consider all the possibilities and takes into account the velocity and stiffness of motion profile.

### 9.2.1 Bezier method for producing a ternary extrusion

The Beziers's control points produce predictable curve, which is aligned to form a ternary extrusion of 3D space, and the n-order is dependent on the final shape and design complexity. A typical formulation for a Bezier segment with 4 control points is shown here as:

$$P(u,v) = \big(x(u,v), y(u,v)\big) = \sum_{i,j=0}^{3} B_i(u)\, B_j(v) P_{ij} \qquad (54)$$

Where, $B_i(u) = \binom{3}{i} u^i (1-u)^{3-i}$ is the $i$th cubic Bernstein polynomial [141]. Furthermore, the control points can be arranged and orientated to produce various surface meshes, and the concept is suitable for parallel robot's parametric sweep method for exploring workspace [142][86].

De Casteljau's algorithm define the control point and subdivide the segment to formulate point C(u) which is situated between a segment which is a line from A to B. Farin discuss another method for creating a curve mesh surface by using Coons patch, and an example formulation to generate a bilinear blended Coons patch is given here as:

$$x(u,v) =$$
$$(1-u)\times(0,v) + ux(1,v) + (1-v)\times(u,0) + vx(u,1) -$$
$$\begin{bmatrix} 1-u & u \end{bmatrix} \begin{bmatrix} x(0,0) & x(0,1) \\ x(1,0) & x(1,1) \end{bmatrix} \begin{bmatrix} 1-v \\ v \end{bmatrix} \quad (55)$$

where the boundary curves is arbitrary and can be listed as x(u,0),x(u,1),x(0,v) and x(1,v) [143].

Shardt explained the method for subdividing or splitting a connected region to ensure the bezulation algorithm rejected line segments can be partitioned to form a smaller region. Bezulation is a process similar to triangulation, except that cubic Bezier patches is used to produce a region, where self-intersecting condition can be handled by splitting at the intersection points. This extraction method can produce control points, when only a cloud data of the final mesh is available for the simulation processing [144].

Figure 9-1 demonstrate the end-effectors path characteristic which is parallel and equally spaced and aligned. The cubic parametric sweep can then follow the path to simulate the search for workspace, singularity and end-effectors action.

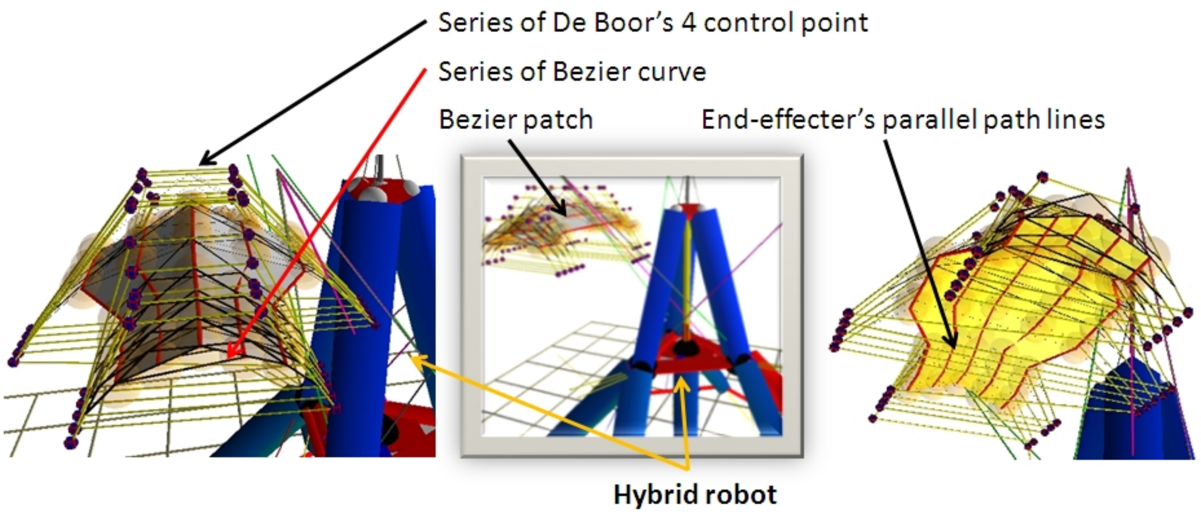

Series of De Boor's 4 control point
Series of Bezier curve
Bezier patch
End-effecter's parallel path lines
Hybrid robot

Figure 9-1 Bezier patch with simulated end-effectors path segment is shown as yellow node.

Figure 9-2 demonstrates a case study, where a block of material is placed as a target, and then a Bezier patch is placed inside the material. This Bezier patch is then used as marker for the end-effecter's motion. The Bezier parallel line becomes the cubic parametric sweep path for the end-effecter. Figure 9-2 shows the end-effecter's action and the resulting shape.

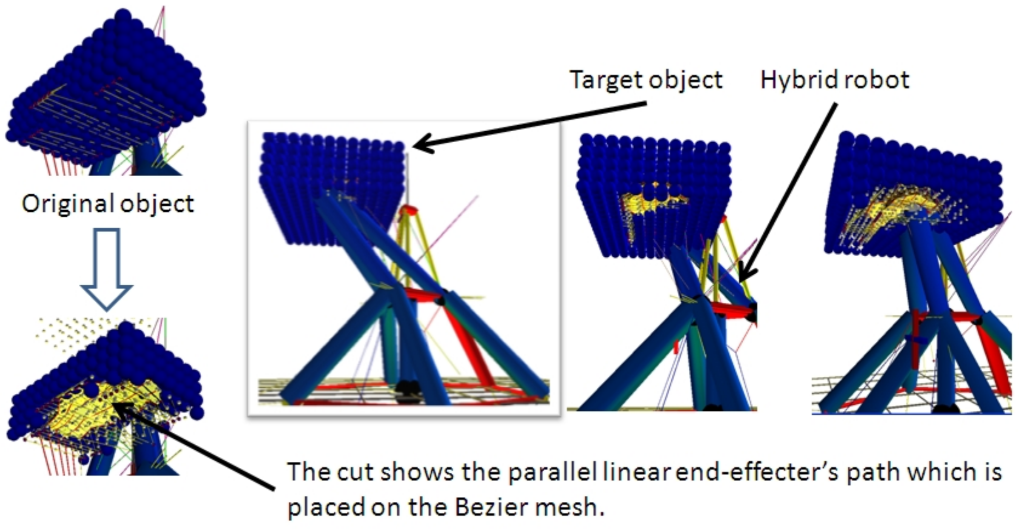

Target object
Hybrid robot
Original object
The cut shows the parallel linear end-effecter's path which is placed on the Bezier mesh.

Figure 9-2 A demonstration of the cutting action following the Bezier path line

Figure 9-3 displays the Bezier patch data processed using numerical system, which validates the Python simulation path.

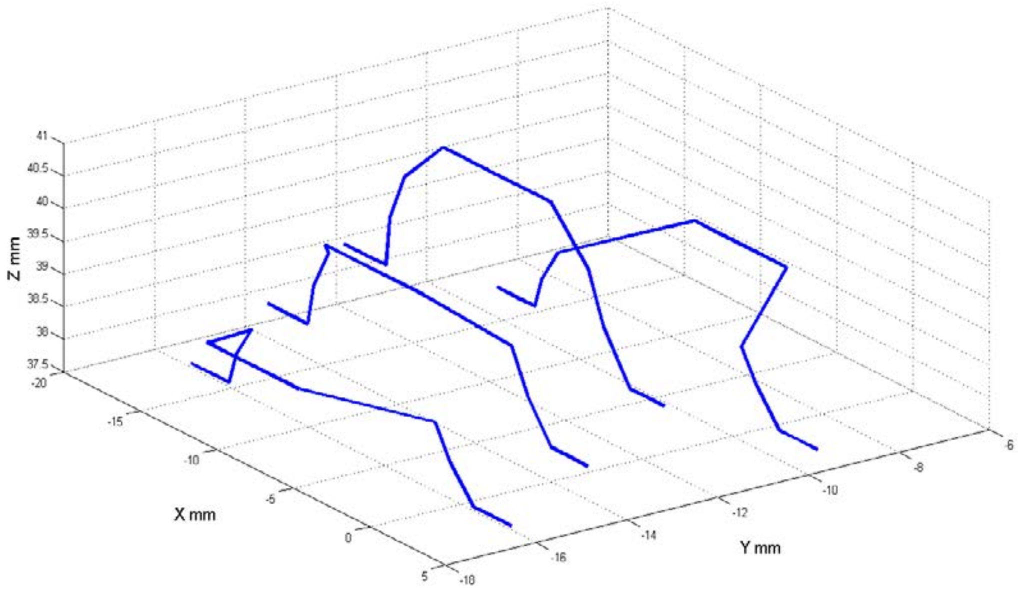


Figure 9-3 The Bezier patch data

### 9.2.2 Simulation result

Juri shows the potential of applying the path planning method for a machining task. However, no experiment has been conducted due to various scale, operating system and data format issues. Experimental result from a similar cutting task is shown in Figure 9-4[145].

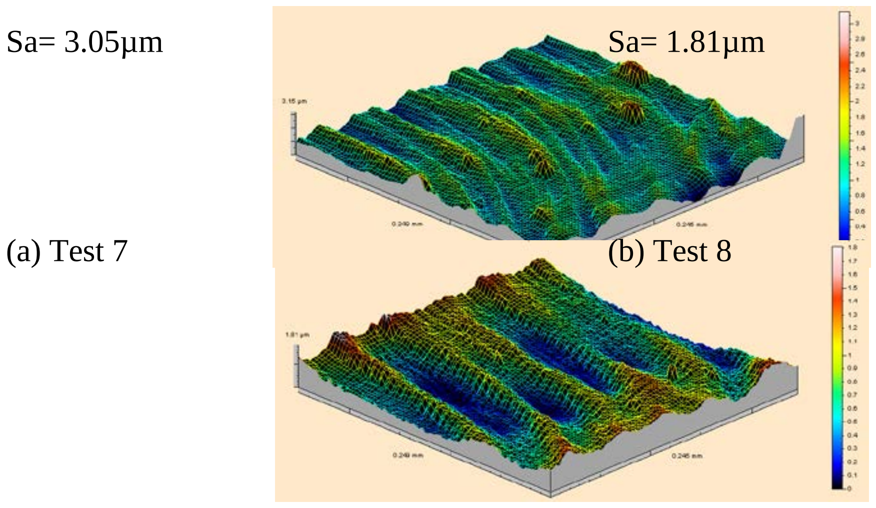


Figure 9-4 3D topography plot of surface in (a) Test 7 vs (b) Test 8 **[145]**

## 9.3 Application 2: Path planning based on dynamic force and velocity

This section presents the optimized path planning of hybrid parallel robot for ankle rehabilitation based on force vector which is described in the previous section. The focus of this experiment is on replicating the motion path provided by lower-limb rehabilitation program. Force vectors driven path is then compared with Python simulation result for validation purposes [51].

## 9.4 Application 3: Path's slicing analysis as a therapist's intervention tool in ankle robotic rehabilitation

The assisted limb rehabilitation process is commonly associated with advanced control of the affected limb which is in the form of robotic assistance and human interference. Robotic element is only expected to be able to reproduce the motion suitable for large variations of patient's condition within a reasonable accuracy and stiffness. Therapist interferences or fine-control is in the format of planar elements (like pelvis linkage) or joints (like knee), which relates to trajectory or orientation adjustments. The rehabilitation process has to consider the patient's ability, limit and motion constraint which form those two factors. The parameters for controlling all this is associated with kinematic, that defines the behaviour and characteristic of the lower limb. The 3D Python simulation system allows for this fine-tuning in the form of slice analysis and interval analysis. The Hybrid and Hexapod robot design which is a stationary foot Orthoses system is linked to a haptic controller that runs on Python's haptic engine. This haptic controller provides therapist with force-feedback sensation to aid the motion correction and adjustment process to allow for a balanced operational consideration mainly for these two factors, which are robot's stiffness factor and patient's motion constraints.

### 9.4.1 Experiment setup for the limb rehabilitation project

The experimental setup for this research is based on a motion signature of healthy participants, which were recorded as a skeletal model with joint characteristics and ranges including links' displacement information and kinematics with force distribution by the Vicon cameras in the gait laboratory of the West Midland Rehabilitation Centre, Birmingham, United Kingdom. Identification and measurement of leg segmental motion characteristics performed by 30 participants which were selected by an advertisement in the University Of Birmingham, including 15 males and 15 females, with the mean age of 27.05±7.32 years old, mean height of 168.16±14.32 cm and mean body mass of 68± 7.43 kg. The recorded path is the source data to the whole experimental setup. This data is post-processed using Python 3D simulation system.

Firstly, the system needs to perform research for non-singular position which leads to a general workspace. Cubic parametric sweep and spherical parametric sweep produce the base data, that contribute to the Learning mode system. In a learning mode, the 3D or 4D fractal (an L-system with random generated test point) populates the strategic slices (plane placed at strategic position on or near the target, which is the Vicon path for the purpose of this experiment) to form an interpolation or extrapolation for a range of nodes. The node could be test point, a limb or a joint, which should remain within the robot's workspace.

The 'data slicing analysis' allow therapist to perform local analysis while inspecting option for trajectory, plane (axis) changes to where the ankle is acting upon. A 'slice' is a plane or collection of planes, with strategy for placing, orientating and spacing them. The spacing, orientation and placing of slice's strategy is based on 3D- curve algorithm which allows fine-control of each segment curvature parameters. The slice or the plane in the most basic form is a simple 2D cubic parametric sweep, and the advanced format is L-system fractal in 3D and 4D shape. The spacing, orientation and placing of slices is therapist-configurable, therefore this allow for typical operation and handling of limb usually found in limb rehabilitation exercise. Jim mentions that lower limb rotation is dependent on angle, and EMG activity demonstrates that patient has angle limitation which is specific due to some muscle activity, mechanical properties or motion constraint [146]. Belda-Lois explain the rehabilitation model known as Carr and Shepherd motor relearning method which focus on movement components which is not achievable and functional tasks that try to correct the problem. This correction is specific to a problem, therefore machine learning or automation for robotic rehabilitation has limited scope, which otherwise would require therapist intervention [44].

### 9.4.2 Theory for therapist intervention

The focus is about therapist intervention related to path lines, which means that a therapist can redefine the path, adjust the patient's trajectory and ensure that both couple system synchronize following the rehabilitation task objectives. The therapist is expected to focus on non-achievable movement which is related to a functional task, where the system or patient may have difficulty in achieving the goals. Therefore, the main aspect of this research is about the path lines, and how the system can assist therapist by giving them the facility to monitor, adjust and optimizes the path lines.

The first issue to be considered here is the subdivision aspect of the given path. The control point used in this simulation is an approximation by reducing noise and drift especially during dynamic motion recording [147]. This control point is critical to both the system (for interpolation, curvature control and optimization purposes) and therapist, where therapist need to be able to control the slicing plane, therefore allowing therapist's intervention to fine-tune and control the curve.

The intervention point can be derived from the Bezier control point, where a Bezier curve is generated based on the result from the best possible interpolated point to form a comparative path against the given path. Bezier curve construction is based on 3D midpoint segmentation, and the order is based on de Casteljau Subdivision Algorithm that follows a balanced AVL tree [114, 149]. Simas studies on Bezier curve construction forms a matrix arrangement which correspond to an end-effectors position for welding task. Simas claims that Bezier curve or surface is suitable for planning an end-effectors parallel positioning which reduce the erosion while doing welding task [142]. Furthermore, Shardta explains the advantage of using Bezier compared to Nurbs, where Bezier piecewise cubic polynomial segments and tensor product patches conform well. When Bezier segment passes through its two end control points, a Bezier patch must have passes through its four corners [144].

Bezier curve construction based on de Casteljau plans to find C (u), where u is within the range of [0, 1]. Therefore the subdivision ratio is given as u: 1-u and a typical ratio are 0.4. Bezier curve can be written in Bernstein form as $B(t) = \sum_{i=0}^{n} \beta_i b_{i,n} n(t)$. Where b is a Bernstein basis polynomial, and can be written as $b_{i,n}(t) = \binom{n}{i}(1-t)^{n-i}t^i$. Then, use recurrence to find the curve at $t_o$ which is described here as $\beta_i^0 = \beta_i$ , where I =0,..,n

$$\beta_i^j = \beta_i^{j-1}(1-t_0) + \beta_{i+1}^{j-1}(t_0), \text{ where i=0, .. , n-j and j=1, .., n} \qquad (56.a)$$

$B(t_0) = \beta_0^{(n)}$, which is the evaluation of B at point $t_0$ in n steps.

For a 3D Bezier curve with n+1 control points $P_i$

$$B(t) = \sum_{i=0}^{n} P_i b_{i,n}(t), t \in [0,1], \text{ where } P_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \qquad 56.b)$$

Then, the Bezier is split into components, which are

$$B_1(t) = \sum_{i=0}^{n} x_i b_{i,n}(t), t \in [0,1] \qquad (56.c)$$

$$B_2(t) = \sum_{i=0}^{n} y_i b_{i,n}(t), t \in [0,1] \qquad (56.d)$$

$$B_3(t) = \sum_{i=0}^{n} z_i b_{i,n}(t), t \in [0,1] \qquad (56.e)$$

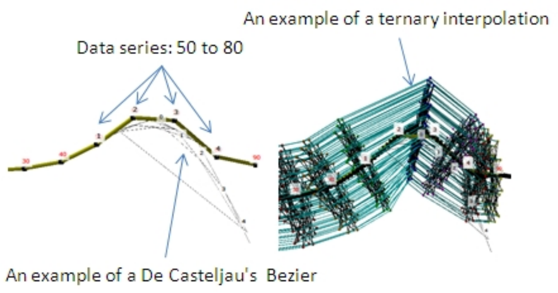
Then, use De Casteljau's algorithm to evaluate $B_1(t)$,$B_2(t)$ and $B_3(t)$[149].The De Casteljau's subdivision is an iterative process; therefore a control is required to stop the subdivision, which is based on distance error between the control point and the lines. Bezier curve and Bezier path is suitable for parametric sweep search, due to its characteristic such equal spacing between line, linear path arrangement and rigid format. 5-n should be able to solve for a continuous and complex curve construction where axis changes is dominant. Typical error for Bezier construction includes cusp and collinear.

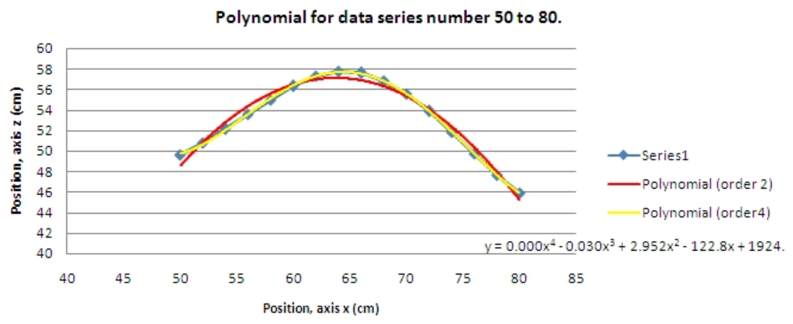
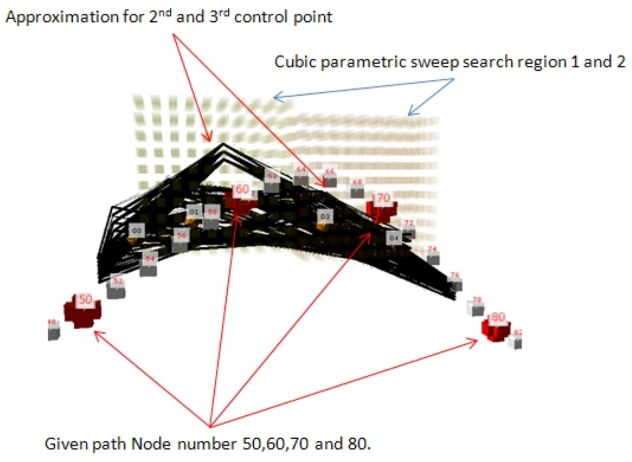
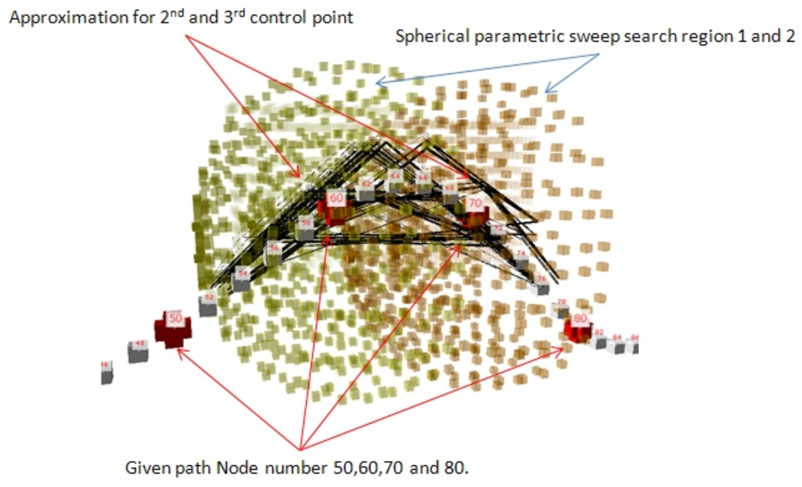
Sohel described a combination of chain code (CC) and run length code (RLC) to derive Bezier control points from a target, which is an image. This iteration process stops when the distance has reached an approximated boundary edge threshold value [150].Ahmad wrote approximation method such as recursive subdivision (RS) and parabolic approximation (PA) when the Bezier curve is flattened. The error factor includes curve flatness and wrong approximation of control points where each control points has direct effect on the final shape. However, parabolic approximation produces smooth curve outcome compared to RS method which produces a small deviation to the trajectory while the iteration process move around the approximated points[151].

Bhuiyan developed an approximate and learning strategy to find the control points for a Bezier curve. Out of the 4 standard control points, where one is associated with the start point, and the other with the end, the strategy for robust approximation has to consider the location for the two other control points. Bhuiyan strategy can be adapted following the 3d sweep or parametric sweep search with selection criteria for the curvature’s accuracy compared to the given curve [152].

Table 11gives approaches of finding and estimating a given path, and formulate a way to control and manipulate the path. Considering Bhuiyan method, a parametric sweep method test for an estimated 3d region was conducted, where the control points may fall inside. Due to the infinite possibility for cubic grid orientation and size, a spherical parametric sweep is the best geometric grid for this task. Using this method, we can perform analysis for a collection of segment that makes a complete path. Therefore, providing therapist with the ability to control and modify the path.

Table 11 the approach for finding and estimating the given path

| Description | Image |
|---|---|
| 2D interpolation test. Using the given path, conduct tests to find ways for re-generating the given curve. The test is performed on a subset of the given path, at data series number 50 to 80. |   An example of a ternary interpolation<br>Data series: 50 to 80<br>An example of a De Casteljau's Bezier Figure 9-5 Example dataset for series 50 to 80 |
| A) Result from Microsoft Excel using Polynomial interpolation order | |

| | |
|---|---|
| number 2 and 4. |  Figure 9-6 Polynomial result (order 2 and 4) |
| B) Result from cubic parametric sweep search for 4 Bezier control points. |   Approximation for 2nd and 3rd control point<br>Cubic parametric sweep search region 1 and 2<br>Given path Node number 50,60,70 and 80. Figure 9-7 Cubic parametric sweep |
| C) Result from spherical parametric sweep search for 4 Bezier control points. |   Approximation for 2nd and 3rd control point<br>Spherical parametric sweep search region 1 and 2<br>Given path Node number 50,60,70 and 80. Figure 9-8 Spherical parametric sweep |

### 9.4.3 Geometric Brownian motion

Geometric Brownian motion is used to approximate the Bezier control point and interpolate the path in order to optimize its performance. Geometric Brownian motion is also used to generate the adjacency graph. Geometric Brownian motion is a continuous-time Gaussian process, with a covariance function like this

$$E[B_H(t)TB_H(s)] = \frac{1}{2}(|t|^{2H} + |s|^{2H} - |t-s|^{2H}) \quad (57)$$

Where h is a real number in (0,1), which is also called the Hurst index. This process $B_H$ (f) is for [0, 7], t is time. Dieker explains that there is a few method for simulating the Brownian motion namely Hosking method (which generates Xn+1 given Xn,…, X0 recursively) , Cholesky method (which is based on the decomposition of the covariance matrix), Davies and Harte method (which rely on finding a `square root' of the covariance matrix), approximate method like stochastic, and numerous other methods. [153]

Table 12 illustrates the differences between the Brownian motion, Ternary algebra and 3D sweeping method which are extrusion or interpolation method to distribute test points and build adjacency graph. With the graph available, we can proceed with path planning and optimization effort. Table 13 demonstrates the flexibility with the ternary system, where various L-system algorithms can be integrated therefore providing different adjacency graph result.

Table 12 Information regarding various extrusion methods for path optimization

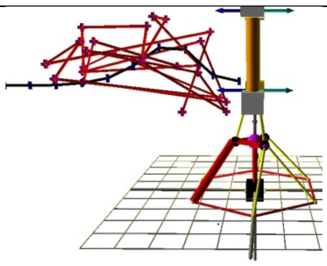
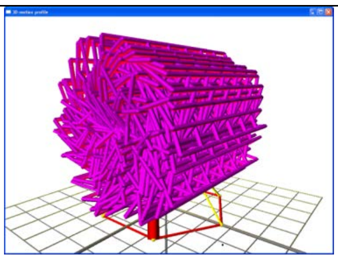
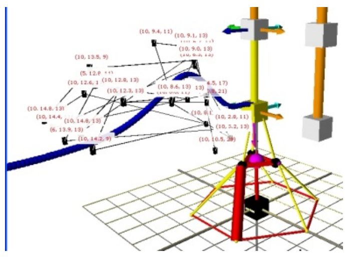
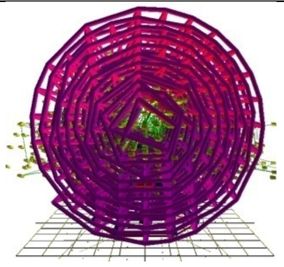
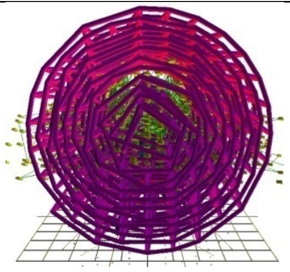
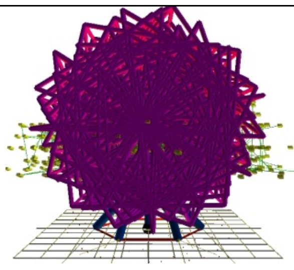
| Extrusion type | Characteristics | Image |
|---|---|---|
| A) Geometric Brownian motion. | A flexible fractal system with no clear definition for edge and direction. This method is more natural compared to the pre-defined space for L-system. |  Figure 9-9 Brownian |
| b) Ternary algebra. | Quadric algebra ternary based on various L-system fractal algorithms, and spiral produces more relevant result. |  Figure 9-10 Ulam spiral or Voxel |
| c) 3D sweeping. | Serial or multi-parallel process based on the established method for hull edge, where a seed branch out towards a learning data. |  (10, 9.4, 11)<br>(10, 9.1, 13)<br>(10, 9.0, 13)<br>(10, 13.5, 9)<br>(5, 12.8<br>(10, 12.6, 1<br>(10, 12.8, 13)<br>(10, 12.3, 13)<br>(10, 8.6, 13)<br>4.5, 17)<br>(10, 14.8, 13)<br>(10, 14.4,<br>(10, 14.8, 13)<br>(6, 13.9, 13)<br>(10, 14.2, 9)<br>(10, 2.8, 11)<br>(10, 3.2, 13)<br>(10, 10.5, 28) Figure 9-11 3D sweeping |

Table 13 Comparison for ternary algorithm variations

| Ternary variation | Result | Image |
|---|---|---|
| A) Simple spiral. $theta = r \times 2 \times pi$ $x = \cos(theta) \times r$ $y = -\sin(theta) \times r$ | Accuracy=43% (427 / 973). Time = 22s. |  Figure 9-12 Simple spiral |
| B)Sack Spiral. $x = \cos(\sqrt{i} \times 2 \times pi) \times \sqrt{i}$ $y = \sin(\sqrt{i} \times 2 \times pi) \times \sqrt{i}$ | Accuracy= 57% (555/ 973). Time = 22s . |  Figure 9-13 Sack spiral |
| C) Vogel spiral. $theta = i \times 2 \times pi \div (pi \times pi)$ $x = \cos(theta) \times r$ $\sin(theta) \times r$ | Accuracy= 62% (604 / 973). Time = 22s. |  Figure 9-14 Vogel spiral |

With Bezier curve carefully selected, important control points are then selected and slices are placed. Each slice is populated using L-system fractal generator. In this case Ulam Spiral is used to effectively populate and create Graph network as a 3D extrapolation to test the path, and then interpolate between the slices to form a path for the travelling plates.

Ulam Spiral is a prime factor spiral generator, and its polar pattern is useful for the parametric sweep activity. There are few variants to this prime factor spiral like Vogel spiral, Fibonacci sums and direct rasterization. Ulam Spiral and its variant is generated and strategically positioned to demonstrate the system capability and its integration with numerical system. The formulation for Ulam Spiral's Vogel 2D is given here in (58), the circular or spherical sweep is more effective and relevance to parallel robot design.

$$(x, y) = \sum_{i=0,i+1}^{n} \left( (\cos theta) \times r, (-\sin theta) \times r \right) \quad (58)$$

Where theta=r x pi, and $r = \sqrt{i}$

Then, a strategy to generate a Graph network for the travelling plate to sweep through or search the region is used. This is done using the ternary algebra. This ternary algebra equation follows Cheng's definition [66]. Ternary algebra produces the 3D L-system pattern. Spiral or Voxel parametric sweep is more efficient compared to cubic sweep, due to the characteristics of the parallel robot's structure, where if the structure is simplified into a serial connection between base to center point or actuator, then a translation or rotation of the end-effectors will produce a spherical or hemispherical matrix. Cheng's formulation is given below in (Eq.59).

$$t = x_i + y_j + z_k \quad (59.a)$$

Where x,y and z are real numbers, while i,j and k are imaginary units. Based on two ternary numbers $t_1$ and $t_2$,

$$t_1 + t_2 = (x_1 + x_2)i + (y_1 + y_2)j + (z_1 + z_2)k, \quad (59.b)$$

$$t_1.t_2 = (x_1x_2 - y_1z_2 - z_1y_2)i + (x_1y_2 + y_1x_2 - z_1z_2)j + (x_1z_2 + y_1y_2 + z_1x_2)k \quad (59.c)$$

Get ternary number t

$$t^2 = (x^2 - 2yz)i + (2xy - z^2)j + (2xz + y^2)k \quad (59.d)$$

$$x_0 = x \times cxy - y \times sxy; y = x \times sxy + y \times cxy \quad (59.e)$$

$$x_0 = x \times cxz - z \times sxz; z = x \times sxz + z \times cxz \quad (59.f)$$

$$y_0 = y \times cyz - z \times syz; z = y \times syz + z \times cyz \quad (59.g)$$

Nylander's formula for the "*n*th power" of the 3D vector to build a 3D fractal is based on Hamilton quaternion of complex numbers [98].Nylander method for Mandelbulb 3D is shown in Eq.60. This method is also used in generating 3D L-system shape as shown in Table 12.

$$(x, y, z)^n = \rho^n(\cos(n\theta)\cos(n\varphi), \sin(n\varphi)\cos(n\varphi), -\sin(n\varphi)) \quad (60)$$

Where $(\rho, \varphi, \theta)^n = (\rho^n, n\varphi, n\theta)$

There are various ways of connecting the test points populated on a plane to be linked to form an adjacency graph network. The adjacency graph has additional parameters such as cost, Grassmann and path-distance data. The generated lines represent a list of possible path connection between the start and end planar slices. Ulam Spiral produce the rotating Voxel effect which make it compact and efficient graph.

The adjacency graph is modifiable based on therapist chosen criteria. The criteria are related to preferences for trajectory, target-following criteria, gradient changes, local or global slicing analysis for any node or range of segments along the path.

The first step is to identify the various methods to 3D interpolation to develop the adjacency graph. This interpolation is performed on one type of L-system which is known as Ulam Spiral. The advantage of using spiral or helix shape or pattern allows for optimum end-effectors travel, path conformation and reduce unnecessary travel outside the potential region.

Table 13 demonstrates the method used to extrapolate and interpolate the region that forms within the given path. This way we can provide therapist with interaction ability, and the system has knowledge over the given path's variation.

The end-effectors shall follow the interpolated test points, and shall be checked against the geometry error, kinematic transformation and Grassmann error. The strategy for Grassmann is explained here in the format of weighted ranking for any non-singular position. This is given by the Grassmann factor, which checks for Grassmann-related errors like instantaneous rotational axis, plane or edge's vector meets at infinity or any pair of edge or plane's vectors is coplanar. The method for finding the non-singular position is based on parametric sweep to find the workspace allowable by the geometry limit and Grassmann check to reduce stiffness error.

The Ulam Spiral or Helix method provide optimum opportunity for end-effectors travel without having to go into irrelevant region, like when using cubic parametric sweep. Another advantage of using helix or spiral path is the path's ability to grow radials as the search expands.

### 9.4.4 Adjacency graph

Carsten used D* to solve 3D path planning based on interpolation on faces within a given region of a test point[122]. The Python simulation make use of Jarvis March 3D, where the algorithm use GGN, distance to End-effectors' centre point and angle between selected nodes as the attribute to find the best path. Robert demonstrate the effective uses of Jarvis March in 2D to find convex hull [154]. Adjacency graph is an efficient ways of handling tree-like structure, which is useful in handling the workspace and path data.

### 9.4.5 Optimizing the Hybrid robot effort to follow the given PATH

The collection of possible paths is interpolated to optimize the results. The simulation uses shortest distance to platform A's centre-point, whose coordinates must be identical to those of the next node in the queue to form platform A's path. The 1D-array interpolation cannot solve the problem without additional values, whereas a 2D-array would help find the optimized path. The additional attribute to make the 2D-array is obtained from stiffness factor, Grassmann vector factor, and minimum distance to the centre-point of each corresponding node.

### 9.4.6 Optimizing the given path

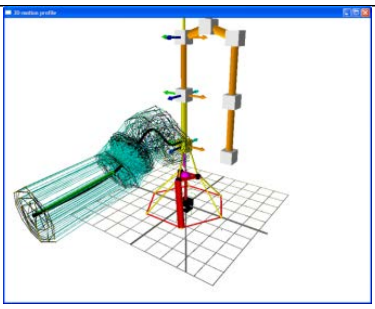
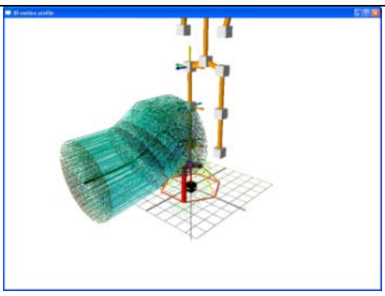
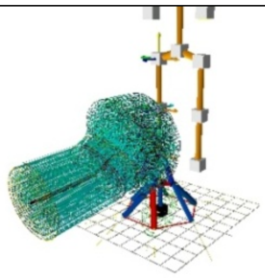
Iterative processes such as recursive subdivision (RS) suffer from flatness distributed throughout the path approximation process. Parabolic reduces the error, while losing some detail. If this temporal problem is transferred to the robot’s motion or the limb rehabilitation system, then the RS system’s flatness characteristics may produce negative effects.

### 9.4.7 The adjacency graph variations

The variation depends on the complexity of the adjacency graph, which connects the test points found on each planes (slices). The adjacency graph has direct effect on the travel path or parametric sweep behaviour. Table 12 demonstrates the interval analysis action performed at selected region, where an increased number of network graph is plotted for that interesting region. Table 13 demonstrates scale and dimension changes achievable with the system.

Table 14 demonstrates examples from possible ternary variations, and how they affect the system performance. Similar and large diameter system has the advantage of being within range of the given path, and produces less Grassmann error condition, while the accuracy remains similar for the three designs.

Table 14 Comparison for ternary interpolation

| Interpolation variation | Result | Image |
|---|---|---|
| Similar small diameter. | Accuracy= 77.8% (324 / 416).<br>Time = 8.7e+03s.<br>$G_{GN}$ ranking = 1141 | <br>Figure 9-15 Small diameter interpolation |
| Similar large diameter. | Accuracy= 77.8% (162 / 208).<br>Time = 4e+03s .<br>$G_{GN}$ ranking = 572 | <br>Figure 9-16 Large diameter interpolation |
| Varied diameter. | Accuracy= 76.6% (478 / 624).<br>Time = 2.6e+04s.<br>$G_{GN}$ ranking = 1556 | <br>Figure 9-17 Varied diameter interpolation |

## 9.4.8 Experiment results

Figure 9-18 illustrates the Python simulation iteration based on cubic parametric sweep to estimate the de Boors 4 control points' position. The de Boors control point is a stable numerical ways to find a point on a B-spline curve given a $u$ in the domain. The de Boor control point is generalized as $p(u) = N_{i,p}(u)p_i = p_i$. Figure 9-19 and75 display the approximation for the control points found using the cubic parametric sweep search.

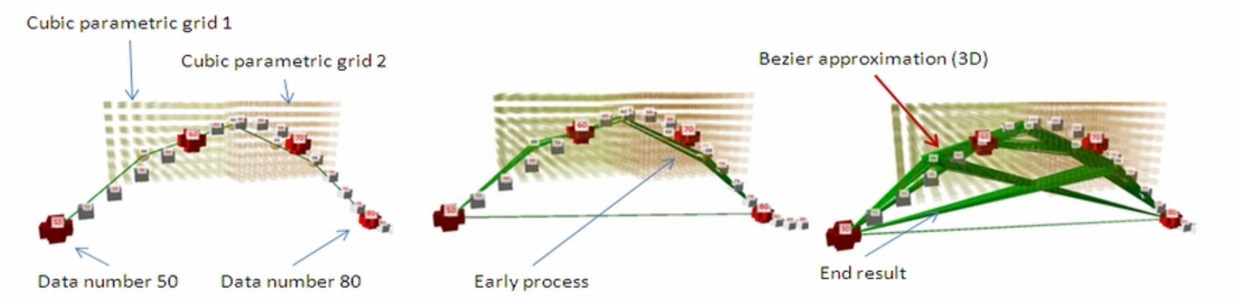


Figure 9-18 Plot for Bezier approximation for data series between data 50 to 80.

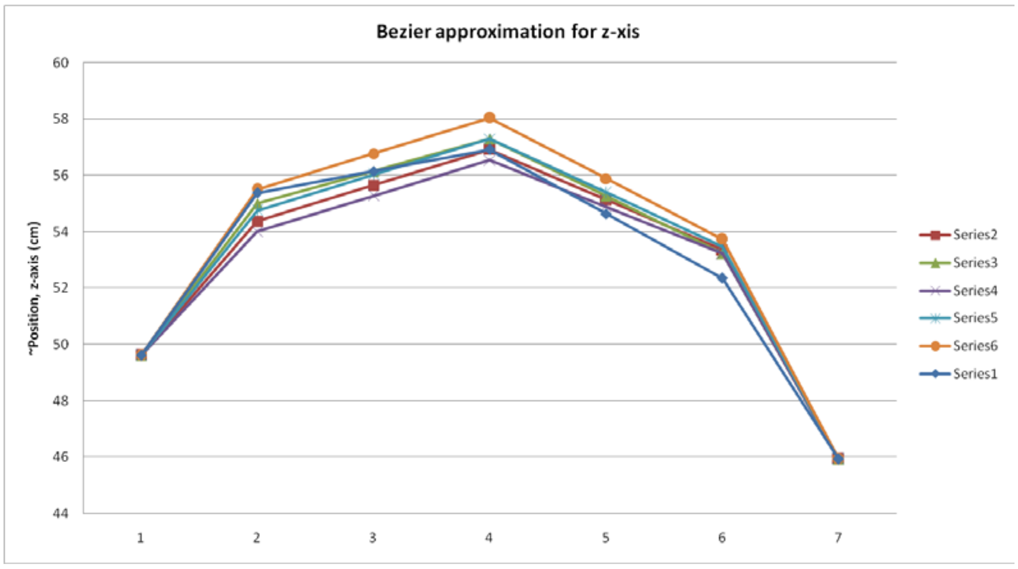


Figure 9-19 Plot for Bezier approximation for z-axis

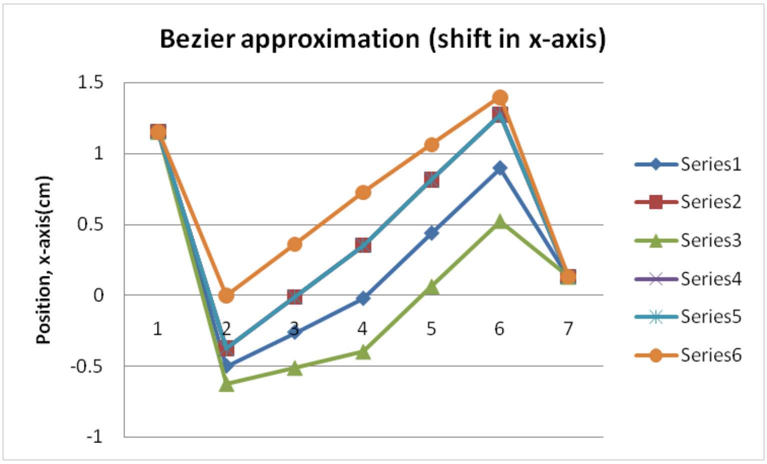


Figure 9-20 Plot for Bezier approximation on x-axis

The accuracy for iteration and parabolic approach has an effect on the usability in rehabilitation. The parametric search for control points provide therapist with the options for optimizing the path and explore new possibilities especially when the path is experiencing structural or motion dynamic problems. The quadric algebra ternary approach is given to illustrate another approach in interpolating the path, and this approach is directly related to the Bezier curve method, which provides the intervention position along the path.

## 9.5 Conclusion

The objective of this chapter is to demonstrate two lower limb rehabilitation applications and one machining application which has been tested and validated by a numerical system. The 3D Beziers's line and surface patch method integrate a cubic parametric sweep search which produces a cutting profile for the parallel robot's end-effectors Boolean control produce quick test result. The system is highly configurable and flexible, where user can define the parallel robot's base and workspace compliance strategy.

The next application is the dynamic force and velocity control which utilizes Grassmann singularity check and geometric constraint check. The result shows that the Python simulation, numerical system and the acquired training data confirm each other well.

The last application is the curve fine-control which is useful in lower limb rehabilitation which allows therapist with the ability of teaching the parallel robot to perform the rehabilitations process correctly. The strategy is based on various 3D test-data population method which plans to improve the identification of smaller test region. The adjacency graph is user-configurable and have many different optimization methods suited for different task. The spiral data population strategy with various data relationship strategy produces best adjacency graph suited for this therapist intervention concept. The advantage of using this ternary interpolation method is that, it allows for data slicing at any position and axis, and the adjacency graph can be rebuilt to network back the whole system. This way therapist can manipulate each node by performing orientation and position adjustment as though therapist is holding a patient's limb. GGN, Grassmann weighted rank and non-geometric singularity condition data is positioned accordingly, and adjacency graph optimizes the path in 2D axes. The numerical result demonstrates a similar trend in Python simulation, therefore validating its result.

# Chapter 10: CONCLUSIONS

## 10.1 Project aims and objectives

The overall aim of the research presented in this thesis is to develop an assistive tool for the design of parallel robot with a specific focus on rapid identification of workspace boundaries and singularities through a 3D simulation system. The output is a draft quality result to achieve real-time control and manipulation effects.

A detailed breakdown of the objectives necessary to achieve the project aim is as follows:

1. Perform a comprehensive literature review to determine the current design and development method for parallel robot.
2. Analyse previous parallel robot simulation system which considers the search for workspace as one of its function.
3. Investigate parallel robot’s workspace search method.
4. Develop a 3D simulation system that performs a parallel robot’s workspace search with considerations for kinematic, geometric constraint and Grassmann singularity.
5. Develop Boolean algebra for flow control and optimization.
6. Explore the potential of extending the geometric grid related to the search envelope parameter.

7. Extends the 3D simulation system ability, by investigating into workspace analysis and path planning.
8. Develop a haptic controller.
9. Determine the haptic functions by integrating with the 3D simulation system.
10. Evaluate and calibrate the system by linking the 3D simulation, haptic controller and a Numerical system developed using Solid Work and Matlab.
11. Evaluate the system for two different applications which are cutting path and limb rehabilitation’s motion planning.

## 10.2 Summary

Highly accurate approach to designing parallel robot is the best method in solving any design issue. However, this approach limits the designer’s ability to explore and try different design parameters, performing test scenario and being creative in developing the robot. The heavy computation and resources required to validate a design usually force user to copy existing design, and try to improve an existing design by introducing some changes. Even then, it is difficult to perform comparison studies, after changes have been done to the existing design. It is costly to build many models and test them in highly accurate system. The Python simulation system fills in this gap, where it allows creative work in finding the best design for a series of task. The simulation system allows integration of various algorithms, employ basic geometric and Grassmann check for singularity suitable for real-time operation and it has variety of methods in defining the search for workspace. Since, many of the previously uncontrollable factors is now user-configurable, therefore user has the flexibility of operating at multiple resolution and test complexity. This allows rapid prototyping of parallel robot with ranges of task definition.

A summary of the five major phases of the research are as follows:

### 10.2.1 Phase 1

*Development of a 3-dimensional simulation system that is able to demonstrate parallel robot's motion, singularity, workspace and path planning.*

- Parallel robot design process is an iteration process involving the determination of topology, required motion (path), available or useful workspace, robot's dimension and its stiffness.
- Parallel robot design activity is prone to mistake, and a redesign activity is heavy computation, timely and costly.
- There are only a few choices for parallel robot rapid development software, for example work done Singulab and SEMORS-PKM. Other than that, there is a variety of specific problem-solver simulation which is very restrictive and limited.
- Parallel robot design produces high number of data, or large matrices which is difficult for user to trace, track and monitor the dynamic changes. Visualization software should be able to assist this effort.
- The 3D Python visualization system provides 3-dimensional view of the dynamic motion, and allow for modification to various simulation and robot's parameters.

The result is usually real-time for low-quality and draft simulation, but require less than Numerical simulation running time when completing parametric search and optimization tasks.

### 10.2.2 Phase 2

*Development of a 3-dimensional system that is able to detect singularity for a real-time application. The system should be able to cater for changes to the geometry and configuration. IT should be object-oriented, modular and open for integration with various algorithms.*

- Parallel robot workspace is not entirely safe from error.
- Typical errors which are specific to a requirement or applications which is found inside the workspace are Grassmann, stiffness, velocity and trajectory's compliance.
- The Python simulation uses the Grassmann to demonstrate few example of singularity search where co-planar and instantaneous rotation axis (IRA) is found inside a known workspace. The issue is not limited to workspace, but also in path planning, dataset blending and analysis process.
- The probability percentages for Grassmann to occur are recorded as between 27-31%. Grassmann condition may become detrimental to the structure if there is any external forces acting on the opposite side of the weak region or when the co-planar conditions allow for instant development of a new degree-of-freedom (DOF) which is also known as instantaneous rotational axis (IRA) condition.

- The Python simulation system manage to complete Merlett's collection of workspace which are constant orientation workspace, orientation workspace, maximal workspace, inclusive orientation workspace, total orientation workspace, dextrous workspace, reduced total orientation workspace, and singularity loci.
- The Python simulation can perform various strategic operations on the workspace data like cross-section analysis by doing slice analysis, multi-resolution operation by using interval analysis, small $C_{space}$ operation by doing local operation and data blending between different datasets.

### 10.2.3 Phase 3

*Development of a Boolean Parametric sweep search strategy to be employed in solving various issues regarding parallel robot design activity*

- Traditional parametric sweep search does not limit the search space, and usually does not have optimization features. Common geometrical shape boundary like cube and sphere limits the scope of workspace research.
- Boolean algebra provides simple operation for controlling the search operation, and the flow-control operation. Boolean algebra also provides simple operation for assessing the database in 2-dimensional and 3-dimensional ways.
- The Python simulation provides weighted ranking system which utilizes parameters like cost, efficiency and error. The Grassmann probability studies highlight the co-planar and IRA problem. Numerical system and most traditional work usually process the Parallel workspace and path using 2-dimensional

interpolation and extrapolation algorithm. The Trilinear interpolation which, is performed at later stage is computationally costly.

### 10.2.4 Phase 4

*Development of a 3-dimensional parametric sweep search strategy that is based on search algorithm. Boolean algebra provides the control strategy to optimize the operation.*

- Traditional parametric sweep is limited to geometrical shape envelope like cube, sphere, cylinder and collections of previous-mentioned shapes.
- The search space $C_{space}$ for traditional method is not optimized, not suitable for path planning, not suitable for real-time operation and the search path is redundant.
- The Python simulation provides Boolean algebra control for limiting the search scope, region and direction. Therefore, the fractal growth method can be directed towards an economical search.
- Boolean control for the search resolution has improved the parametric sweep, where the test for a 40cm wide region has reduced the test completion time from 3400 seconds down to 1100 seconds.
- The Python simulation system was experimented with various 2-dimensional and 3-dimensional methods including Trilinear, L-system fractal, Simplex, Adjacency tree, Voxel, Brownian motion and Marching cube. The experimental test and result does not claim the method is better than the traditional approach. However, it proposes various new approaches in handling Parallel robot data.

- The experimental method demonstrated that the method is transferable for performing other tasks like definition for surface mesh, obstacle avoidance and haptic algorithm.
- The experiments show that the key features, which are not found in traditional cubic sweep, are the expansion, contraction, seeding, direction and pattern generation found in tree, L-system and n-D interpolation methods. These features allow for optimized and small work region (Cspace) and ready integration with algorithm integration. The fundamental object-oriented and modular engine allows for various database systems to be integrated within.
- The Parametric cubic sweep was able to find 10% valid workspace from the overall test data population. In contrast the polar coordinate system can find 68% valid workspace. Cubic-like parametric sweep based on Hilbert and Marching Cube integration with cube and Hilbert produce between 12 to 34% valid non-singular points.
- Type 1D L-System planar grid system is not efficient, where the search can only find between 3.5 to 7.4% valid non-singular points.
- However, when the L-System is interpolated and arranged strategically, total valid non-singular counts have improved and fallen into the range of 20.9% to 41.8%.

### 10.2.5 Phase 5

*Development of a haptic controller that links with the 3D Python simulation, the 3D Numerical system (Solid Works and Matlab), and provides singularity-free control for a physical parallel robot.*

- Typical simulator software does not link to numerical validation software.
- Typical simulator software does not provide control of a physical robot, where the design and control is combined in one unit.
- The PPPRRR configurations serve the purpose of having an easy reconfigurable structure and a single centre-point for all rotary and linear axes, providing capability to cater for various geometries and configurations.
- The designer has difficulty in visualizing a large dataset and matrices for parallel robot system, where a typical hexapod would have 6 linear actuators, 12 joints and end-effectors in a 3-dimensional spatio-temporal system.
- The development of the physical robot is an experimental work in validating the control strategy for a physical robot. The concept has been proven valid and practical, and has been validated by the numerical system.
- The haptic engine is an experiment with restricted search space or $C_{space}$ that extends the Boolean algebra and grid method in Phases 3 and 4 into a restricted search region with weighted ranking system. The probability ranking system is linked to the force-feedback systems which provide braking, acceleration or change of motor direction to the haptic controller.
- The haptic controller can control the 3D Python simulation directly, where it can validate geometric constraint and Grassmann error in real-time. The graphical user interface provides control for end-effectors' motion, any joint's motion, and geometric reconfiguration. It also selects the analysis and algorithm type and places slicing planes in 3D space.
- The haptic controller can control an existing experimental physical robot directly or via Matlab programming, where the system can validate the position and

orientation. The haptic controller has the options of using Simplex, parametric sweep or Grassmann as its haptic engine.

## 10.3 Contributions of the research

1. Development of a new highly configurable and customizable parallel robot design assistive system. The system has collections of useful fundamental core objects, which are modular and extendible. This object-oriented approach allows for various algorithms for varied tasks to be integrated in the system. The database system is open and controlled by simple Boolean algebra.

2. Development of a series of extended parametric sweep methods including L-system, quadric algebra and binary tree. Boolean algebra provides control to constraint the search region, search direction and logical response to a user-defined or system-defined pattern. Slicing and interval analysis together with 2D, 3D and 4D interpolations allow for complex operations such as dataset blending.

3. Development of a weighted ranking Grassmann validation system that checks for Grassmann-related errors within the workspace region. This weighted ranking system gives probability percentage for co-planar and IRA conditions, which may contribute to structural singularities.

4. Development of a haptic controller that helps identifies singularity errors within a 3D space. The haptic controller is a simple kinematic structure that works with various parallel robot’s geometries and configurations. The device adds a new modality and guides the robot the designer in identifying problems involving structures with high-degrees of freedom with large matrices. The integration of nD interpolation method based on Simplex, binary tree, L-system and quadric algebra has been developed into an open and modular haptic engine.

## 10.4 Future Work

Parallel robot development activity involves high resolution, high computation time, and high skilled workers in various simulation and modelling fields. The best approach for robot development is to conduct all computation at highest resolution, however this approach leads to slow result generation and a mistake at any design iteration will require user to restart the expensive process. This approach limit user ability in exploring and designing different design parameters, performing test scenario and being creative in developing the robot. The heavy computation and resources required to validate a design usually force user to use prior design, and try to improve an existing design by introducing some changes. It is costly to build many different models and test them in a highly accurate system. The Python simulation system fills in this gap, where it allows creative work for various design tasks. The simulation system allows integration of various algorithms suited for various task. It employs geometric constraint and Grassmann check for singularity which is suitable for real-time operation. It has variety of methods in defining the search for workspace. Boolean algebra allows for many of the previously uncontrollable factors to become user-configurable. This allows for rapid prototyping of parallel robot with varied ranges of task definition.  A number of further extensions of the system as future work are suggested below:

- There are various research opportunities in expanding the weighted ranking parameters by adding elements like stiffness, force, velocity and voltage.
- The system can be expanded to be able to import CAD model, scale and position the CAD model in the robot's environment. Then, the employ surface meshing algorithm to enable collision, path planning on surface mesh and surface mesh interaction.
- Following Faugire's GSP classification and Ben-Horin's vector-space method, further study on this aspect could lead to auto-reconfigurable geometry per allowed assembly positions per classes, where the system could predict best geometric changes solution when the system reach singularity [14,15].

# REFERENCES

[1] J. P. Conti, C. M. Clinton, G. Zhang, and A. J. Wavering, “Workspace variation of a hexapod machine tool,” 1998.

[2] P. Merlet and P. Prisme, “Designing a Parallel. Manipulator for a Specific Workspace,” 1995.

[3] Song, J.Mou, J.-I King, C., “Error Modeling and Compensation for Parallel Kinematic Machines,” Springer, 1999, pp. 171–187.

[4] Z. Shiller and S. Dubowsky, “On computing the global time-optimal motions of robotic manipulators in the presence of obstacles,” *Robot. Autom. IEEE Trans.*, vol. 7, no. 6, pp. 785–797, 1991.

[5] M. Zefran, “Review of the literature on time-optimal control of robotic manipulators,” *Tech. Reports CIS Univ. Pa.*, 1994.

[6] Glavina, B., "Solving findpath by combination of goal-directed and randomized search," Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on , vol., no., pp.1718,1723 vol.3, 13-18 May 1990

[7] O. Bohigas, M. Manubens, and L. Ros, “Planning singularity-free force-feasible paths on the Stewart platform,” in *Latest Advances in Robot Kinematics*, Springer, 2012, pp. 245–252.

[8] C. Qin and D. Henrich, “Path planning for industrial robot arms-A parallel randomized approach “,” in *Proc. of the Int. Symp. on Intelligent Robotic Systems (SIRS 96), Lissabon, Portugal*, 1996, pp. 65–72.

[9] Chun-Ta Chen and Te-Tan Liao (2010). On the Optimal Singularity-Free Trajectory Planning of Parallel Robot Manipulators, Advances in Robot Manipulators, Ernest Hall (Ed.), ISBN: 978-953-307-070-4

[10] N. Zoso and C. Gosselin, “Point-to-point motion planning of a parallel 3-DOF underactuated cable-suspended robot,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 2325–2330.

[11] Debanik Roy (2012). Spatial Path Planning of Static Robots Using Configuration Space Metrics, Serial and Parallel Robot Manipulators - Kinematics, Dynamics, Control and Optimization, Dr. Serdar Kucuk (Ed.), ISBN: 978-953-51-0437-7.

[12] D. Roy, “Study on the Configuration Space Based Algorithmic Path Planning of Industrial Robots in an Unstructured Congested Three-Dimensional Space: An Approach Using Visibility Map,” *J. Intell. Robot. Syst.*, vol. 43, no. 2–4, pp. 111–145, Sep. 2005.

[13] J. C. Faugire and D. Lazard, “Combinatorial Classes of Parallel Manipulators,” in *Mech. Mach. Theory*, vol. 30, pp. 765–776.

[14] P. Ben-Horin, M. Shoham, S. Caro, D. Chablat, and P. Wenger, “SinguLab–A Graphical User Interface for the Singularity Analysis of Parallel Robots Based on Grassmann–Cayley Algebra,” *Adv. Robot Kinemat. Anal. Des.*, pp. 49–58, 2008.

[15] Anjan Kumar Dash, I-Ming Chen, Song Huat Yeo, and Guilin Yang. Int. J. Computer Integrated Manufacturing 18(7):615-634 (2005), Nanyang Technological University.

[16] F. Xi, A. Ross, and S. Lang, “Exploring a re-configurable parallel robot for space applications,” in *Proceedings of the 6thInternational Symposium on Artificial Intelligence and Robotics & Automation in Space: I-SAIRAS, Canadian Space Agency, St-Hubert, Quebec, Canada*, 2001.

[17] Z. Lazarevic, 1997, “Feasibility of a Stewart platform with fixed actuators as a platform for CABG surgery device,” Master’s Thesis, Columbia University, Department of Bioengineering.
[18] N. Rojas, J. Borràs Sol, and F. Thomas Arroyo, “A Distance-Based Formulation of the Octahedral Manipulator Kinematics,” 2010.
[19] D. Chablat, “Joint space and workspace analysis of a two-DOF closed-chain manipulator,” *Romansy 18 Robot Des. Dyn. Control*, pp. 81–90, 2010.
[20] J. -P. Merlet, *Parallel Robots*, vol. Volume 128. 2006.
[21] M. Uchiyama, Y. Tsumaki, and W.-K. Yoon, “Design of a compact 6-dof haptic device to use parallel mechanisms,” in *Robotics Research*, Springer, 2007, pp. 145–162.
[22] S. Bandyopadhyay and A. Ghosal, “Geometric characterization and parametric representation of the singularity manifold of a 6–6 Stewart platform manipulator,” *Mech. Mach. Theory*, vol. 41, no. 11, pp. 1377–1400, 2006.
[23] D. Yu and J. Han, “Kinematic calibration of parallel robots,” in *Mechatronics and Automation*, 2005, vol. 11, pp. 521–525.
[24] M. F. Barnett-Cowan, “Multisensory spatial perception: sex and neurological differences,” York Universtity Toronto, Ontario, PhD Thesis, 2009.
[25] H. S. Kim, “Mechanism Design of Haptic Devices,” *Adv. Haptics InTech*, pp. 283–297, 2010.
[26] A. M. Okamura, “Haptic feedback in robot-assisted minimally invasive surgery:,” *Curr. Opin. Urol.*, vol. 19, no. 1, pp. 102–107, Jan. 2009.
[27] Yong Zhang and Feng Gao, “A calibration test of Stewart platform,” UK, 2007, pp. 15–17.
[28] S. Colton and F. R. C. Mentor, “The balance filter,” *Present. Mass. Inst. Technol.*, 2007.
[29] S. O. Madgwick, “An efficient orientation filter for inertial and inertial/magnetic sensor arrays,” Report x-io and University of Bristol (UK), 2010.
[30] B. T. Streng, “Mechanical linkage design for haptic rehabilitation and development of fine motor skills,” 2009.
[31] S. A. Panëels, J. C. Roberts, and P. J. Rodgers, “HITPROTO: a tool for the rapid prototyping of haptic interactions for haptic data visualization,” in *Haptics Symposium, 2010 IEEE*, 2010, pp. 261–268.
[32] E. Ruffaldi, D. Morris, T. Edmunds, F. Barbagli, and D. K. Pai, “Standardized evaluation of haptic rendering systems,” *Haptic Interfaces Virtual Environ. Teleoperator Syst.*, vol. 2006 14th Symposium, pp. 225–232.
[33] J. M.-H. D. Scapel and N. Ayache, “Representation, shape, topology and evolution of deformable surfaces. Application to 3D medical image segmentation,” 2000.
[34] Alexander V. Korobeynikov, Vadim E. Turlapov, “Modeling and Evaluating of the Stewart Platform.,” in *International Conference Graphicon 2005*, Department of Computational Mathematics and Cybernetics, Nizhny Novgorod State University after N.I.Lobachevski, 2005.
[35] L. Brezina, O. Andrs, T. Brezina, “NI LabView — Matlab SimMechanics Stewart platform design,” in *Applied and Computational Mechanics 2*, 2008, pp. 235–242.
[36] Wu Dongsu, Gu Hongbin, “Adaptive Sliding Control of Six-DOF Flight Simulator Motion Platform,” *Chin. J. Aeronaut.*, vol. 20, no. 5, pp. 425–433, 2007.
[37] Dequan Zhu, Tao Mei, Lei Sun, “Fuzzy Support Vector Machines Controlfor 6-DOF Parallel Robot,” *J. Comput.*, vol. 6, no. 9, 2011.

[38] H. Yano, S. Tamefusa, N. Tanaka, H. Saito, and H. Iwata, “Interactive gait rehabilitation system with a locomotion interface for training patients to climb stairs,” *Presence Teleoperators Virtual Environ.*, vol. 21, no. 1, pp. 16–30, 2012.
[39] M. Marchal, A. Lécuyer, G. Cirio, L. Bonnet, and M. Emily, “Walking up and down in immersive virtual worlds: Novel interactive techniques based on visual feedback,” in *3D User Interfaces (3DUI), 2010 IEEE Symposium on*, 2010, pp. 19–26.
[40] D. Cioi, A. Kale, G. Burdea, J. Engsberg, W. Janes, and S. Ross, “Ankle control and strength training for children with cerebral palsy using the Rutgers Ankle CP,” 2011, pp. 1–6.
[41] P. S. Lum, C. G. Burgar, P. C. Shor, M. Majmundar, and M. Van der Loos, “Robot-assisted movement training compared with conventional therapy techniques for the rehabilitation of upper-limb motor function after stroke,” *Arch. Phys. Med. Rehabil.*, vol. 83, no. 7, pp. 952–959, Jul. 2002.
[42] C. E. Syrseloudis, I. Z. Emiris, T. Lilas, and A. Maglara, “Design of a simple and modular 2-DOF ankle physiotherapy device relying on a hybrid serial-parallel robotic architecture,” *Appl. Bionics Biomech.*, vol. 8, no. 1, pp. 101–114, 2011.
[43] P. Sui, L. Yao, J. S. Dai, and H. Wang, “Development and Key Issues of the Ankle Rehabilitation Robots.” 13th World Congress in Mechanism and Machine Science, Guanajuato, México, 19-25 June, 2011
[44] J.-M. Belda-Lois, S. Mena-del Horno, I. Bermejo-Bosch, J. C. Moreno, J. L. Pons, D. Farina, M. Iosa, M. Molinari, F. Tamburella, and A. Ramos, “Rehabilitation of gait after stroke: a review towards a top-down approach,” *J. Neuroengineering Rehabil.*, vol. 8, no. 1, p. 66, 2011.
[45] J. L. Patton, M. E. Stoykov, M. Kovic, and F. A. Mussa-Ivaldi, “Evaluation of robotic training forces that either enhance or reduce error in chronic hemiparetic stroke survivors,” *Exp. Brain Res.*, vol. 168, no. 3, pp. 368–383, Oct. 2005.
[46] L. E. Kahn, P. S. Lum, and D. J. Reinkensmeyer, “Selection of robotic therapy algorithms for the upper extremity in chronic stroke: Insights from MIME and ARM Guide results,” *Kaist Daejeon Repub. Korea*, pp. 208–210, 2003.
[47] D. J. Reinkensmeyer, J. L. Emken, and S. C. Cramer, “Robotics, motor learning, and neurologic recovery,” *Annu. Rev. Biomed. Eng.*, vol. 6, no. 1, pp. 497–525, Aug. 2004.
[48] F. Oldewurtel, M. Mihelj, T. Nef, and R. Riener, “Patient-cooperative control strategies for coordinated functional arm movements,” 2007, pp. 2527–2534.
[49] C. Kaspar, “Using Bezier Curves for Geometric Transformations,” Fall 2009.
[50] N. M. Amato, M. T. Goodrich, and E. A. Ramos, “Computing the arrangement of curve segments: Divide-and-conquer algorithms via sampling,” in *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, 2000, pp. 705–706.
[51] Che Zulkhairi Abdullah, M. Saadat, and H. Rakhodaei, “A methodology for Workspace Identification of Parallel Robots Using Parametric Sweep Search Method,” in *Proceedings of the ASME 2013 37th DETC2013 Conference on Mechanisms and Robotics*, Oregon, USA, 2013.
[52] P. Ben-Horin and M. Shoham, “Application of Grassmann - Cayley Algebra to Geometrical Interpretation of Parallel Robot Singularities,” *J Robot. Res*, pp. 127–141, 2009.
[53] S. Amine, M. T. Masouleh, S. Caro, P. Wenger, and C. M. Gosselin, “Singularity Analysis of the 4-RUU Parallel Manipulator using Grassmann-Cayley Algebra,” *Trans. Can. Soc. Mech. Eng.*, vol. 35, no. 5, pp. 515–528, 2011.

[54] Ay, S, Vatandas, O.E, Hacioglu, A., “The effect of radius of joint location on workspace analysis of the 6-6 Stewart Platform Mechanism,” presented at the RAST ’09. 4th International, 2009, pp. 728 – 731.
[55] Charters, “how to find the geometry 3-118-1-PB.pdf,” *Math.--Ind. Case Stud. J.*, vol. 1, 2009.
[56] Creative Commons Attribution-ShareAlike License, “Plücker coordinates,” *Wikipedia*, 28-Feb-2013. .
[57] N. Karcanias and J. Leventides, “Grassmann invariants, matrix pencils, and linear system properties,” *Linear Algebra Its Appl.*, vol. 241, pp. 705–731, 1996.
[58] C. Geiss and S. Geiss, “An introduction to probability theory,” *Lect. Notes*, vol. 60, 2004.
[59] R. B. Ash, *Basic probability theory*. Mineola, N.Y.: Dover Publications, 2008.
[60] J.-M. Chang, *Classification on the Grassmannians: Theory and Applications*. ProQuest, 2008.
[61] E. Staffetti and F. Thomas, “Analysis of rigid body interactions for compliant motion tasks using the Grassmann-Cayley algebra,” in *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, 2000, vol. 3, pp. 2325–2332.
[62] H. Shah, M. S. Narayanan, and V. N. Krovi, “CAD-enhanced workspace optimization for parallel manipulators: A case study,” in *Automation Science and Engineering (CASE), 2010 IEEE Conference on*, 2010, pp. 21–24.
[63] A. Rockwood and P. Chambers, *Interactive curves and surfaces: a multimedia tutorial on CAGD*. San Francisco, Calif.: Morgan Kaufmann Publishers, 1996.
[64] “Hilbert Curves in More (or fewer) than Two Dimensions.” [Online]. Available: http://www.tiac.net/~sw/2008/10/Hilbert/. [Accessed: 30-May-2013].
[65] T. S. Newman and H. Yi, “A survey of the marching cubes algorithm,” *Comput. Graph.*, vol. 30, no. 5, pp. 854–879, Oct. 2006.
[66] J. Cheng and J. Tan, “Generalization of 3D Mandelbrot and Julia sets,” *J. Zhejiang Univ. Sci.*, vol. 8, no. 1, pp. 134–141, Jan. 2007.
[67] O. Faugeras and T. Papadopoulo, “Grassmann–Cayley algebra for modelling systems of cameras and the algebraic equations of the manifold of trifocal tensors,” *Philos. Trans. R. Soc. Lond. Ser. Math. Phys. Eng. Sci.*, vol. 356, no. 1740, pp. 1123–1152, 1998.
[68] F. Jourdan, “Quadric modeling in a Grassmann-Cayley algebra setting,” in *Information Visualisation, 2005. Proceedings. Ninth International Conference on*, 2005, pp. 860–865.
[69] P. Hemingway, “n-Simplex interpolation,” Technical Report HPL-2002-320, 2002.
[70] B. Domonkos and B. Csébfalvi, “DC-splines: Revisiting the trilinear interpolation on the body-centered cubic lattice,” 2010.
[71] F. Gao and L. Han, “Implementing the Nelder-Mead simplex algorithm with adaptive parameters,” *Comput. Optim. Appl.*, vol. 51, no. 1, pp. 259–277, May 2010.
[72] A. Lopes and K. Brodlie, “Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing,” *Vis. Comput. Graph. IEEE Trans.*, vol. 9, no. 1, pp. 16–29, 2003.
[73] C. Gribble, “Primes in Spirals,” THE UNIVERSITY OF ARIZONA, 2010.
[74] P. Prusinkiewicz and A. Lindenmayer, *Graphical modeling using L-systems*. Springer, 1990.

[75] I. Cohen and D. Gordon, “The Voxel-Sweep: A Boundary-based Algorithm for Object Segmentation and Connected-Components Detection.,” in *VMV*, 2004, pp. 405–411.

[76] N. Srebro and T. Jaakkola, “Weighted low-rank approximations,” in *ICML*, 2003, vol. 3, pp. 720–727.

[77] A. A. Obiniyi, E. E. Absalom, and K. Adako, “Arithmetic Logic Design with Color-Coded Ternary for Ternary Computing,” *Int. J. Comput. Appl. 0975 – 8887*, vol. 26, no. 11, Jul. 2011.

[78] A. Douady and J. H. Hubbard, *Exploring the Mandelbrot set. The Orsay notes*. Citeseer. Université Paris Sud, Orsay, France; Cornell University, Ithaca, NY, USA. 1981-82

[79] J. Orchard and T. Möller, “Accelerated splatting using a 3d adjacency data structure,” in *Graphics Interface*, 2001, vol. 1, pp. 191–200.

[80] D. Holten, “Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data,” *Vis. Comput. Graph. IEEE Trans.*, vol. 12, no. 5, pp. 741–748, 2006.

[81] P. V. Sander, D. Nehab, E. Chlamtac, and H. Hoppe, “Efficient traversal of mesh edges using adjacency primitives,” in *ACM Transactions on Graphics (TOG)*, 2008, vol. 27, p. 144.

[82] Verhoeff, T (Tom); Verhoeff, K, “Mitered fractal trees: constructions and properties,” in *Proceedings of Bridges Towson: Mathematics, Music, Art, Architecture, Culture*, 2012.

[83] I. A. Bonev and C. M. Gosselin, “Singularity loci of planar parallel manipulators with revolute joints,” in *Proc. 2nd Workshop on Computational Kinematics*, 2001, pp. 291–299.

[84] Troyanov,M. Note sur le probleme de prolegomenes. Prolegomenes. April 1995

[85] D. Chablat, “Joint space and workspace analysis of a two-DOF closed-chain manipulator,” *Romansy 18 Robot Des. Dyn. Control*, pp. 81–90, 2010.

[86] H. Shah, M. S. Narayanan, and V. N. Krovi, “CAD-enhanced workspace optimization for parallel manipulators: A case study,” in *Automation Science and Engineering (CASE), 2010 IEEE Conference on*, 2010, pp. 816–821.

[87] F. A. Lara-Molina, J. M. Rosário, and D. Dumur, “Multi-Objective Design of Parallel Manipulator Using Global Indices,” *Open Mech. Eng. J.*, vol. 4, no. 1, pp. 37–47, 2010.

[88] P. Merrell and D. Manocha, “Model synthesis: A general procedural modeling algorithm,” *Vis. Comput. Graph. IEEE Trans.*, vol. 17, no. 6, pp. 715–728, 2011.

[89] L. Jiang, *Image processing and computing in structural biology*. S.l.: s.n., 2009.

[90] A. Rosa, “Methods and applications to display quaternion Julia sets,” *Electron. J. Differ. Equations Control Process. St Petersburg*, vol. 4, pp. 1–22, 2005.

[91] C. H. Séquin, “Symmetrical Hamiltonian manifolds on regular 3D and 4D polytopes,” *Coxeter Day Banff Can.*, pp. 463–472, 2005.

[92] M. Tannous, S. Caro, and A. Goldsztejn, “Sensitivity Analysis of Parallel Manipulators Using a Fixed Point Interval Iteration Method”, 13th World Congress in Mechanism and Machine Science, Guanajuato, M´exico, 19-25 June, 2011.

[93] J. J. Keiren, M. A. Reniers, and T. A. Willemse, “Structural Analysis of Boolean Equation Systems,” *ACM Trans Comput Log TOCL*, vol. 13, no. 1, p. 8, 2012.

[94] H. Shah, M. S. Narayanan, and V. N. Krovi, “CAD-enhanced workspace optimization for parallel manipulators: A case study,” in *Automation Science and Engineering (CASE)*, pp. 21–24.

[95] cortesi, “Portrait of the Hilbert curve,” *Portrait of the Hilbert curve*, 03-Aug-2012. [Online]. Available: http://corte.si/posts/code/hilbert/portrait/index.html.

[96] T. Si and Y. Yu, “Anyonic loops in three-dimensional spin liquid and chiral spin liquid,” *Nucl. Phys. B*, vol. 803, no. 3, pp. 428–449, Nov. 2008.

[97] “L-system - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/L-system. [Accessed: 24-Aug-2013].

[98] J. Barrallo, “Expanding the Mandelbrot Set into Higher Dimensions,” presented at the Bridges 2010: Mathematics, Music, Art, Architecture, Culture, Pécs, Hungary, 2010.

[99] S. Kabai, “Investigation of Polyhedral Rings and Clusters with the Help of Physical Models and  Wolfram Mathematica,” presented at the Bridges 2010: Mathematics, Music, Art, Architecture, Culture, Pécs, Hungary, 2010.

[100] G. Paul and H. E. Stanley, “Fractal Dimension of 3-Blocks in 4d, 5d, and 6d Percolation Systems,” *ArXiv Prepr. Cond-Mat0210345*, 2002.

[101] W. Jung, “Homeomorphisms on Edges of the Mandelbrot Set,” Universitätsbibliothek, 2002.

[102] G. Barequet and A. Vaxman, “Nonlinear interpolation between slices,” in *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, 2007, pp. 97–107.

[103] H. Liu, Y. Wang, and Q. Tao, “A realistic method for real-time obstacle avoidance without the Calculation of Cspace Obstacles.”.J.Comput.Sci & Technol. Vol.20, No.6, page 774-787. Nov. 2005.

[104] D. Pavić, M. Campen, and L. Kobbelt, *Comput. Graph. Forum*, vol. 29, pp. 75–87, 2010.

[105] M. A. Gallego, J. D. Fernández, M. A. Martínez-Prieto, and P. de la Fuente, *Rdf visualization using a three-dimensional adjacency matrix*. 4th International Semantic Search Workshop (SemSearch 2011), 2011.

[106] S. Pigot, “Topological models for 3d spatial information systems,” in *AUTOCARTO-CONFERENCE-*, 1991, vol. 6, pp. 368–368.

[107] Y. Choi and C. S. Rim, “Circuit partitioning by quadratic Boolean programming for reconfigurable circuit boards,” in *Custom Integrated Circuits, 1999. Proceedings of the IEEE 1999*, 1999, pp. 571–574.

[108] I. Zammouri and B. Ayeb, “Fractal shapes description with parametric L-systems and turtle algebra,” *World Acad. Sci. Eng. Technol.*, vol. 34, 2007.

[109] P. Prusinkiewicz, “Graphical applications of L-systems,” in *Proceedings of graphics interface*, 1986, vol. 86, pp. 247–253.

[110] D. M. Dubois, “Incursive and hyperincursive systems, fractal machine and anticipatory logic,” 2001, vol. 573, pp. 437–451.

[111] T. Akutsu, M. Hayashida, W.-K. Ching, and M. K. Ng, “Control of Boolean networks: hardness results and algorithms for tree structured networks,” *J. Theor. Biol.*, vol. 244, no. 4, pp. 670–679, 2007.

[112] B. Glavina, “Solving findpath by combination of goal-directed and randomized search,” in *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, 1990, pp. 1718–1723.

[113] “OOPWeb.com - AVL Trees: Tutorial and C++ Implementation by Brad Appleton.” [Online]. Available: http://oopweb.com/Algorithms/Documents/AvlTrees/VolumeFrames.html. [Accessed: 03-Jul-2013].

[114] S. Bhattacharya, H. Hatwal, and A. Ghosh, “An on-line parameter estimation scheme for generalized stewart platform type parallel manipulators,” *Mech. Mach. Theory*, vol. 32, no. 1, pp. 79–89, 1997.

[115] S. Bhattacharya, H. Hatwal, and A. Ghosh, “Comparison of an exact and an approximate method of singularity avoidance in platform type parallel manipulators,” *Mech. Mach. Theory*, vol. 33, no. 7, pp. 965–974, 1998.

[116] R. Ur-Rehman, S. Caro, D. Chablat, and P. Wenger, “Multi-objective path placement optimization of parallel kinematics machines based on energy consumption, shaking forces and maximum actuator torques: Application to the Orthoglide,” *Mech. Mach. Theory*, vol. 45, no. 8, pp. 1125–1141, 2010.

[117] T. Bonnemains, H. Chanal, B. C. Bouzgarrou, and P. Ray, “Dynamic model of an overconstrained PKM with compliances: The Tripteor X7,” *Robot. Comput.-Integr. Manuf.*, vol. 29, no. 1, pp. 180–191, Feb. 2013.

[118] C. Rossi and S. Savino, “Robot trajectory planning by assigning positions and tangential velocities,” *Robot. Comput.-Integr. Manuf.*, vol. 29, no. 1, pp. 139–156, Feb. 2013.

[119] A. A. Ata, “Optimal trajectory planning of manipulators: a review,” *J. Eng. Sci. Technol.*, vol. 2, no. 1, pp. 32–54, 2007.

[120] Y. Lou, F. Feng, and M. Y. Wang, “Trajectory planning and control of parallel manipulators,” in *Control and Automation, 2009. ICCA 2009. IEEE International Conference on*, 2009, pp. 1013–1018.

[121] M. Callegari, G. Palmieri, M.-C. Palpacelli: "Cartesian space visual control of a translating parallel manipulator", Proc. 19th AIMeTA Congress of Theoretical and Applied Mechanics, Ancona, Italy, September 14 -17, 2009.

[122] J. Carsten, D. Ferguson, and A. Stentz, “3d field d: Improved path planning and replanning in three dimensions,” in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, 2006, pp. 3381–3386.

[123] “Mitered fractal trees: constructions and properties.” .

[124] D. Li, Q. Li, N. Cheng, and J. Song, “Extended RRT-based path planning for flying robots in complex 3D environments with narrow passages,” in *Automation Science and Engineering (CASE), 2012 IEEE International Conference on*, 2012, pp. 1173–1178.

[125] D. Nieuwenhuisen, J. van den Berg, and M. Overmars, “Efficient path planning in changing environments,” in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 2007, pp. 3295–3301.

[126] P. Bouboulis, V. Drakopoulos, S. Theodoridis, Image Compression Using Affine Fractal Interpolation Surfaces on Rectangular Lattices, Fractals, Vol. 14, No. 4 (2006)

[127] G. Rote, “Extension of Geometric Filtering Techniques to Higher-Degree Parametric Curves: Curve Intersection by the Subdivision-Supercomposition Method,” Technical report, Freie Universität Berlin, Institute of Computer Science, 2008. ACS Technical Report No.: ACS-TR-361503-01, 2008.

[128] S. Briot and I. Bonev, “Are parallel robots more accurate than serial robots?,” *CSME Trans.*, vol. 31, no. 4, pp. 445–456, 2007.

[129] R. Di Gregorio, “Forward position analysis of the SP-PS-RS architectures,” *Int. J. Robot. Autom.*, vol. 21, no. 4, p. 295, 2006.
[130] J. M. Herve, “Uncoupled actuation of pan-tilt wrists,” *IEEE Trans. Robot.*, vol. 22, no. 1, pp. 56–64, Feb. 2006.
[131] X. Kong and C. M. Gosselin, “Type Synthesis of Three-Degree-of-Freedom Spherical Parallel Manipulators,” *Int. J. Robot. Res.*, vol. 23, no. 3, pp. 237–245, Mar. 2004.
[132] C. M. Gosselin, E. St Pierre, and M. Gagne, “On the development of the agile eye,” *Robot. Autom. Mag. IEEE*, vol. 3, no. 4, pp. 29–37, 1996.
[133] “Lsystems in Python,” 18-May-2013. [Online]. Available: http://www.4dsolutions.net/ocn/lsystems.html. [Accessed: 18-May-2013].
[134] R. Faragher, “Understanding the Basis of the Kalman Filter Via a Simple and Intuitive Derivation [Lecture Notes],” *IEEE Signal Process. Mag.*, vol. 29, no. 5, pp. 128–132, Sep. 2012.
[135] G. R. Bradski and A. Kaehler, *Learning OpenCV [computer vision with the OpenCV library]*. Farnham: O’Reilly, 2008.
[136] K. Demaagd, A. Oliver, and N. Oostendorp, *Practical computer vision with SimpleVC / Kurt Demaagd, ... [et al.]*. Sebastopol, Calif.: O’Reilly, 2012.
[137] W. T. Vetterling, *Numerical recipes example book (C)*. Cambridge; New York: Cambridge University Press, 1992.
[138] I. Cohen and D. Gordon, “The Voxel-Sweep: A Boundary-based Algorithm for Object Segmentation and Connected-Components Detection.,” 2004, pp. 405–411.
[139] P.-Y. Chiang and C.-C. J. Kuo, “Voxel-based shape decomposition for feature-preserving 3D thumbnail creation,” *J. Vis. Commun. Image Represent.*, vol. 23, no. 1, pp. 1–11, Jan. 2012.
[140] J. P. Arpasi, “A brief introduction to ternary logic,” *7th Novemb.*, 2003.
[141] H. Prautzsch, W. Boehm, and M. Paluszny, *Bézier and B-spline techniques*. Springer, 2002.
[142] H. Simas, A. Dias, and R. Guenther, “A scallop-height based algorithm to compute parallel paths on parametric surfaces,” *ABCM Symp. Ser. Mechatronics*, vol. 3, pp. 326–335, 2008.
[143] G. Farin and D. Hansford, “Discrete coons patches,” *Comput. Aided Geom. Des.*, vol. 16, no. 7, pp. 691–700, 1999.
[144] O. Shardt and J. C. Bowman, “Surface parameterization of nonsimply connected planar Bézier regions,” *Comput.-Aided Des.*, vol. 44, no. 5, pp. 484.e1–484.e10, May 2012.
[145] Saedon, Juri Bin, “Micromilling of hardened (62 HRC) AISI D2 cold work tool steel,” Ph.D, University of Birmingham, 2012.
[146] R. Jim, T. Dominic, S. Selfe, and A. Cunningham, “A Biomechanical Investigation of a Single-Limb Squat: Implications for Lower Extremity Rehabilitation Exercise,” *J. Athl. Train.*, vol. 2008;43(5):477–482, pp. 477–482, 2008.
[147] C. Granata, P. BIDAUD, R. Ady, and J. Salini, “A personal robot integrating a physically-based human motion tracking and analysis,” in *Proc. 16th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, University of Technology, Sydney, Australi*, pp. 1–12.
[148] “Finding a Point on a Bézier Curve: De Casteljau’s Algorithm,” 09-Jul-2013. [Online]. Available:

http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/spline/Bezier/de-casteljau.html. [Accessed: 09-Jul-2013].

[149] G. E. Farin and D. Hansford, *The Essentials of Cagd*. Taylor & Francis, 2000.

[150] F. A. Sohel, G. C. Karmakar, and L. S. Dooley, “A generic shape descriptor using Bezier curves,” 2005, pp. 95–100 Vol. 2.

[151] A. L. Ahmad, “Approximation of a Bézier Curve with a Minimal Number of Line Segments,” University of South Alabama, 2001.

[152] M. A.-A. Bhuiyan and H. Hama, “An Accurate Method for Finding the Control Points of Bezier Curves,” *MEMOIRS-Fac. Eng. OSAKA CITY Univ.*, vol. 38, pp. 175–182, 1997.

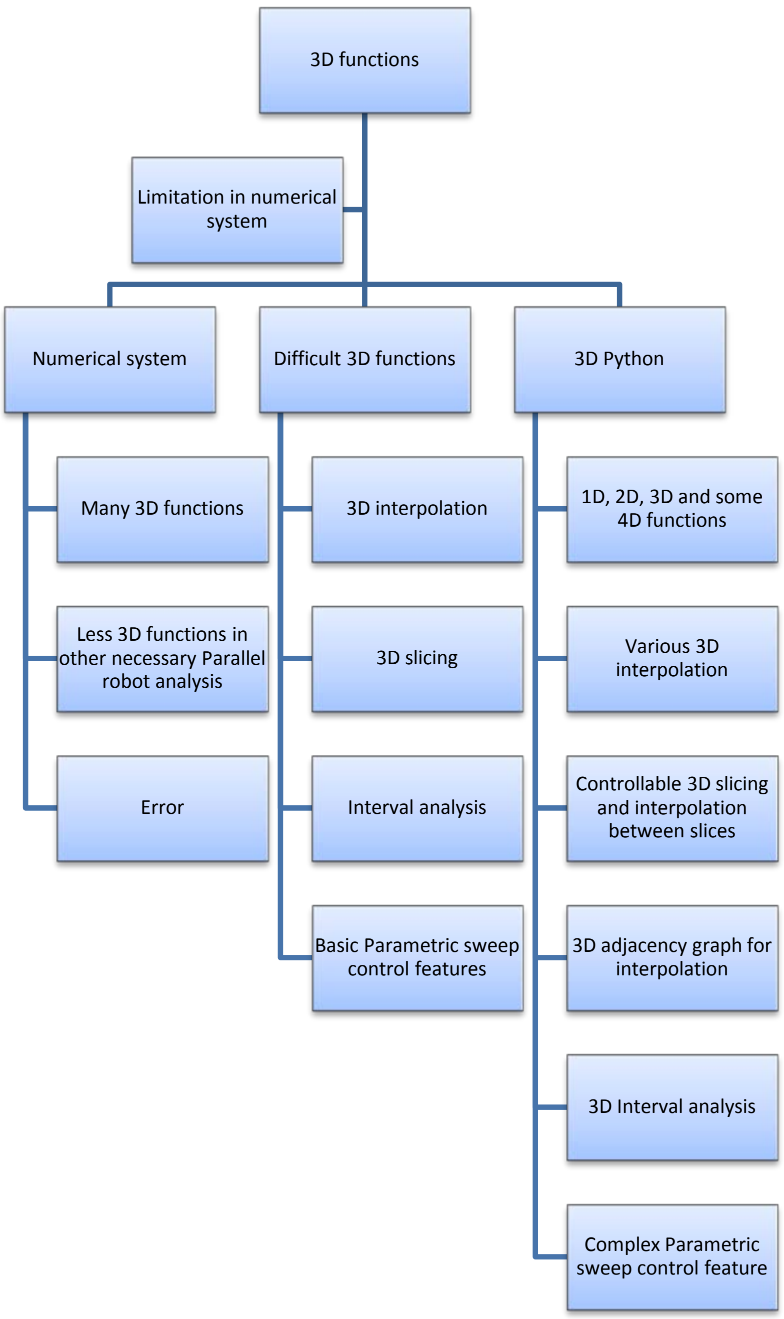
[153] T. Dieker, “Simulation of fractional Brownian motion,” *MSc Theses Univ. Twente Amst. Neth.*, 2004.

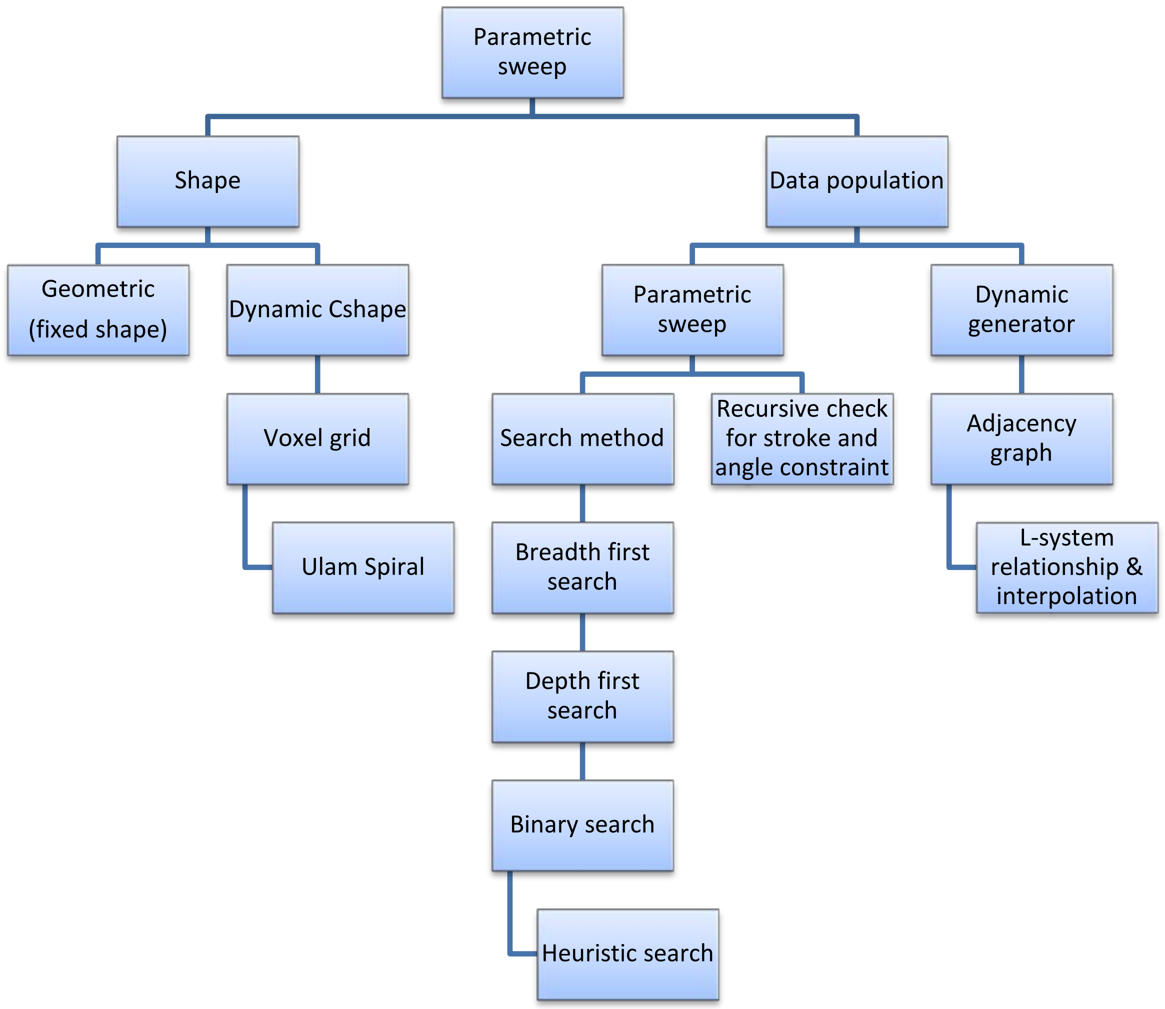
[154] R. Sedgewick and K. Wayne, “Geometric Algorithms.” Princeton University, Algorithms and Data Structures Fall-2007.

[155] P. Ben-Horin and M. Shoham, “Singularity of Gough-Stewart platforms with collinear joints,” in *12th IFToMM World Congress*, 2007, pp. 743–748.

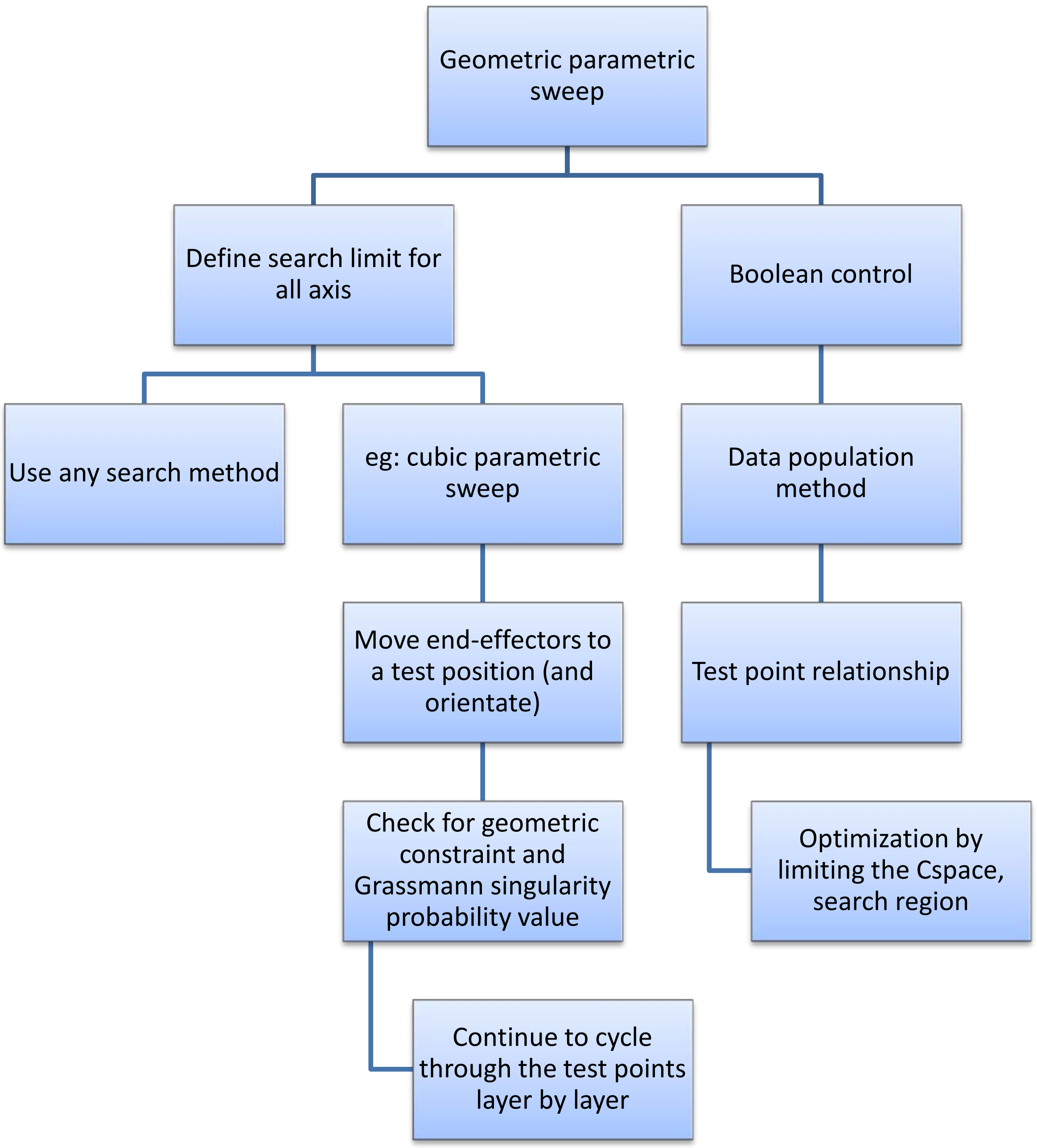
# APPENDICES

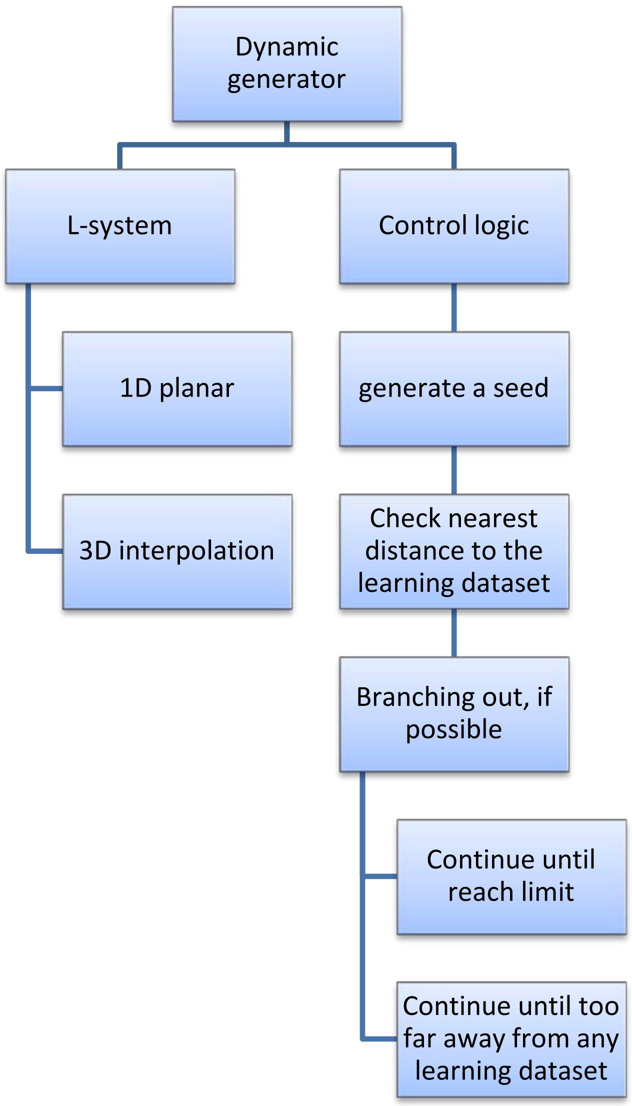
## Software flowchart

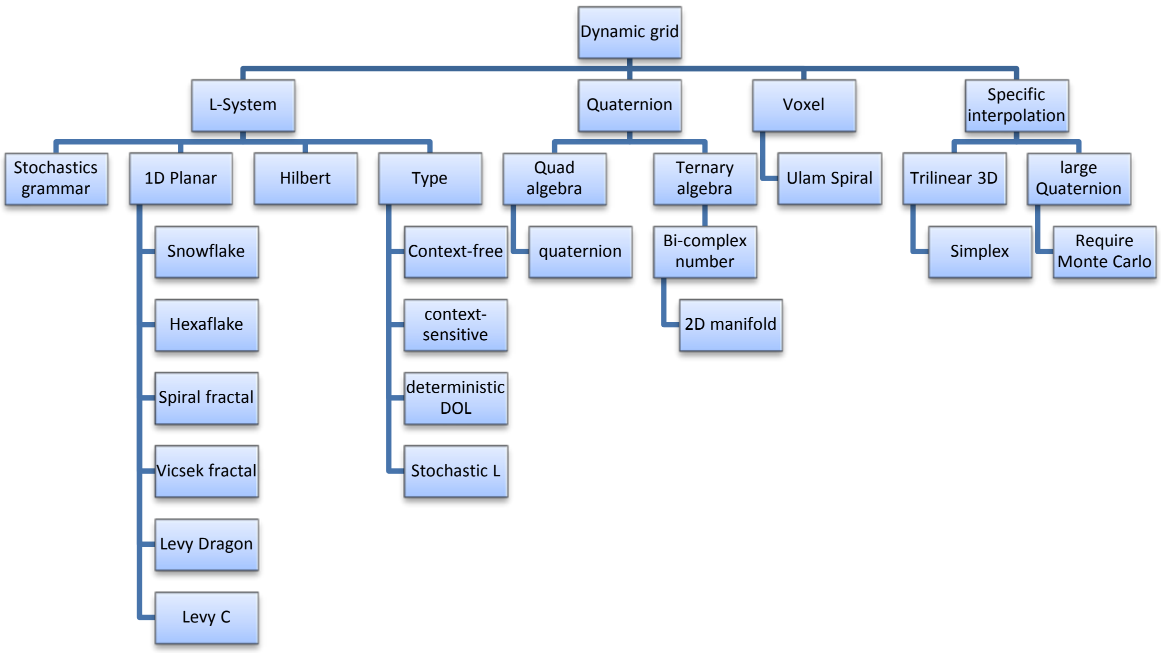

3D functions
Limitation in numerical system
Numerical system
Difficult 3D functions
3D Python
Many 3D functions
Less 3D functions in other necessary Parallel robot analysis
Error
3D interpolation
3D slicing
Interval analysis
Basic Parametric sweep control features
1D, 2D, 3D and some 4D functions
Various 3D interpolation
Controllable 3D slicing and interpolation between slices
3D adjacency graph for interpolation
3D Interval analysis
Complex Parametric sweep control feature

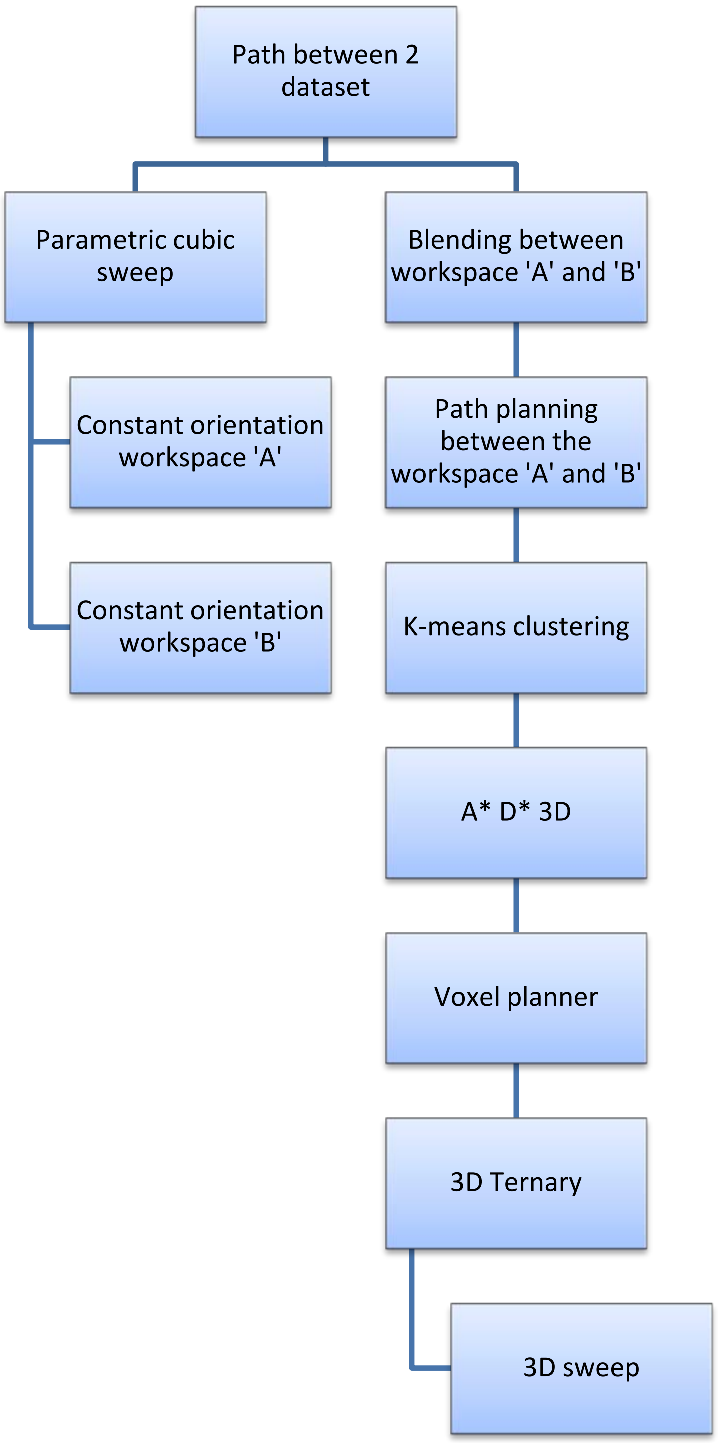


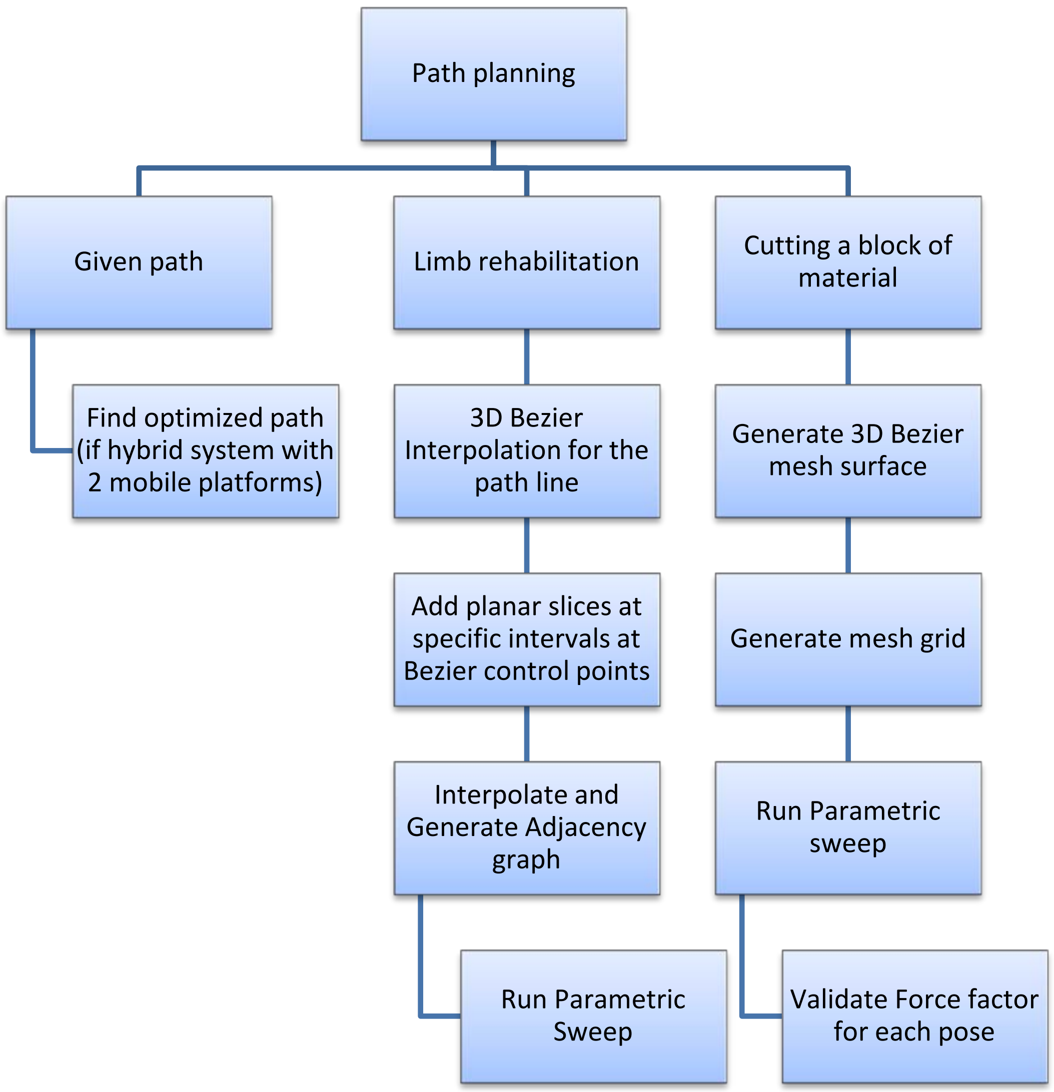
Parametric sweep
Shape
Data population
Geometric (fixed shape)
Dynamic Cshape
Parametric sweep
Dynamic generator
Voxel grid
Search method
Recursive check for stroke and angle constraint
Adjacency graph
Ulam Spiral
Breadth first search
L-system relationship & interpolation
Depth first search
Binary search
Heuristic search

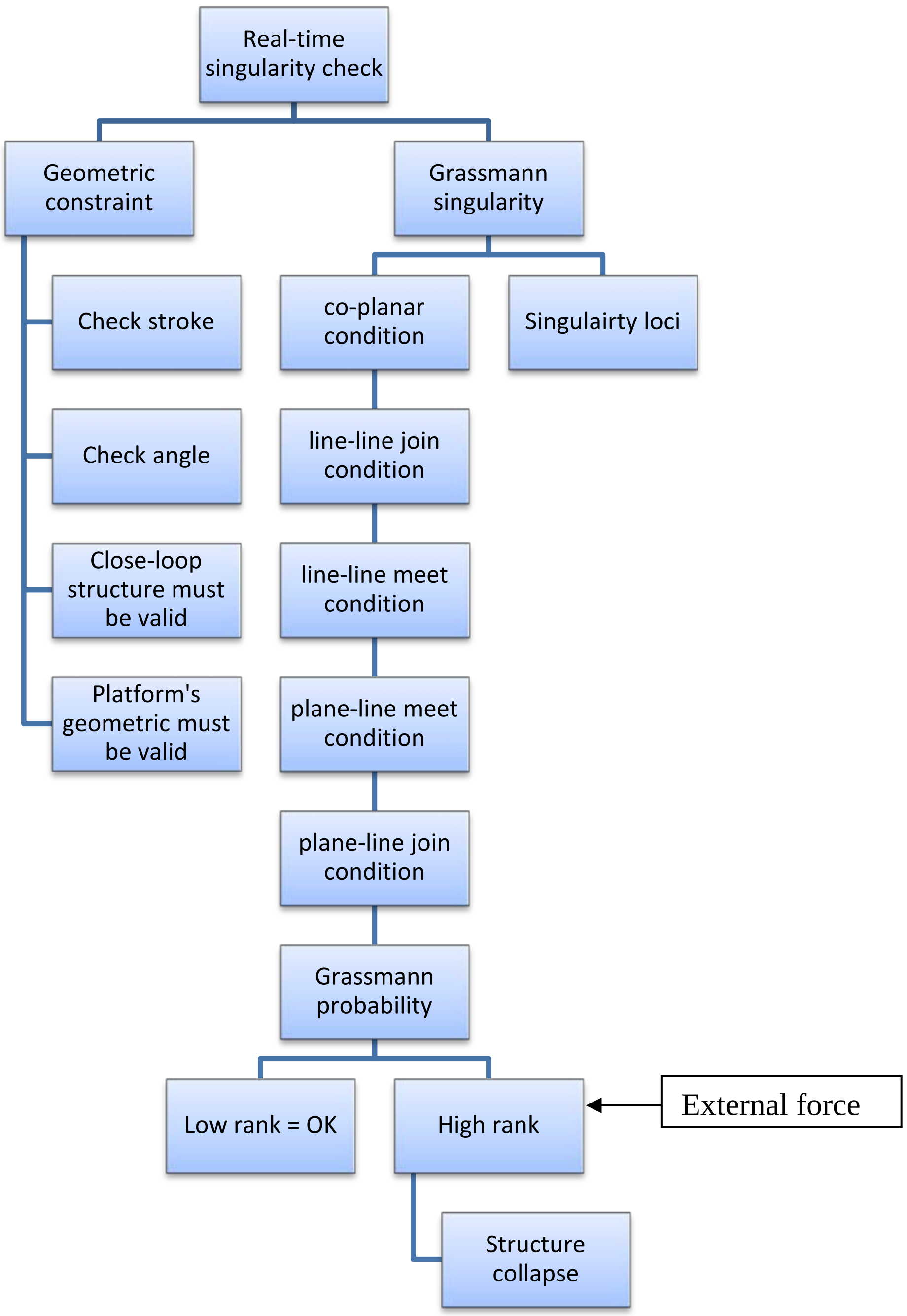

Geometric parametric sweep
Define search limit for all axis
Boolean control
Use any search method
eg: cubic parametric sweep
Data population method
Move end-effectors to a test position (and orientate)
Test point relationship
Check for geometric constraint and Grassmann singularity probability value
Optimization by limiting the Cspace, search region
Continue to cycle through the test points layer by layer

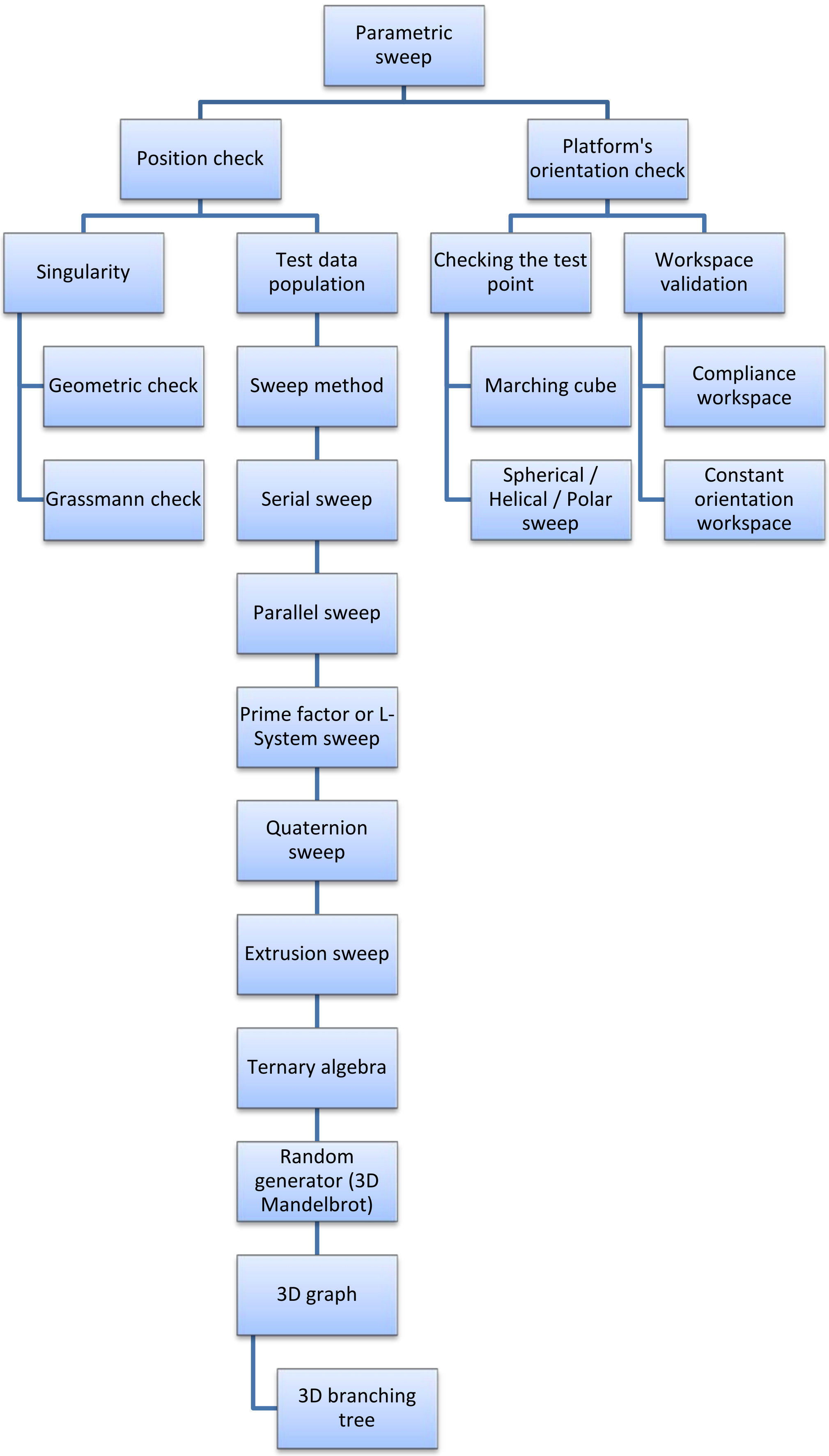

Dynamic generator
L-system
1D planar
3D interpolation
Control logic
generate a seed
Check nearest distance to the learning dataset
Branching out, if possible
Continue until reach limit
Continue until too far away from any learning dataset

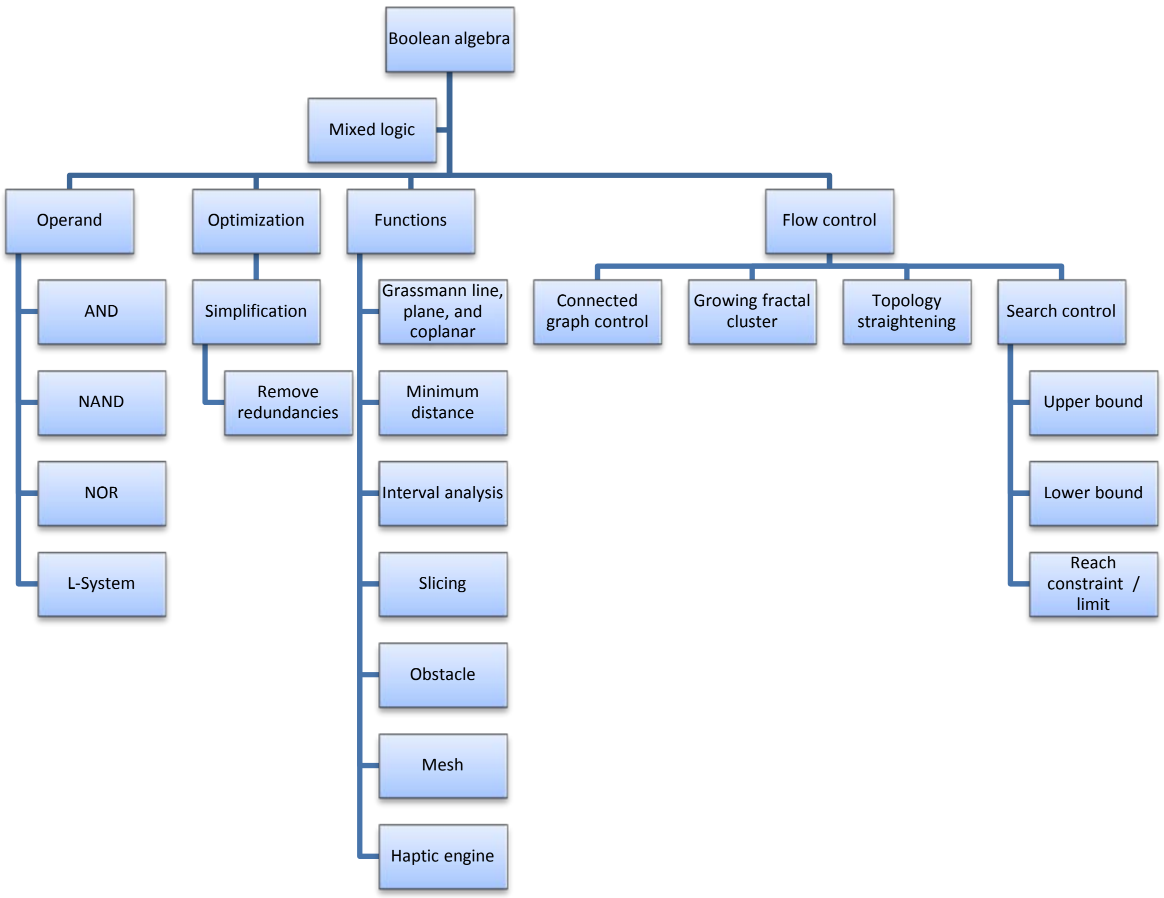

Dynamic grid
L-System
Quaternion
Voxel
Specific interpolation
Stochastics grammar
1D Planar
Hilbert
Type
Quad algebra
Ternary algebra
Ulam Spiral
Trilinear 3D
large Quaternion
Snowflake
Hexaflake
Spiral fractal
Vicsek fractal
Levy Dragon
Levy C
Context-free
context-sensitive
deterministic DOL
Stochastic L
quaternion
Bi-complex number
2D manifold
Simplex
Require Monte Carlo

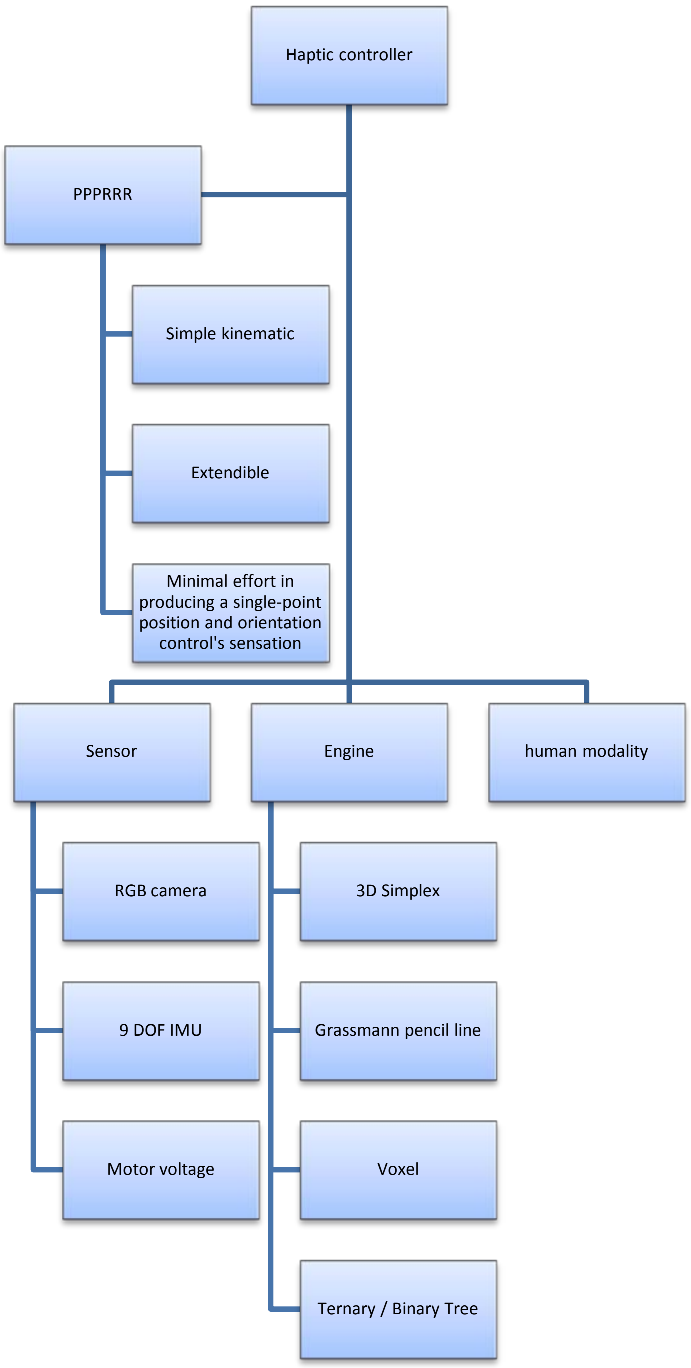

Path between 2 dataset
Parametric cubic sweep
Blending between workspace 'A' and 'B'
Constant orientation workspace 'A'
Path planning between the workspace 'A' and 'B'
Constant orientation workspace 'B'
K-means clustering
A* D* 3D
Voxel planner
3D Ternary
3D sweep


Path planning
Given path
Limb rehabilitation
Cutting a block of material
Find optimized path (if hybrid system with 2 mobile platforms)
3D Bezier Interpolation for the path line
Generate 3D Bezier mesh surface
Add planar slices at specific intervals at Bezier control points
Generate mesh grid
Interpolate and Generate Adjacency graph
Run Parametric sweep
Run Parametric Sweep
Validate Force factor for each pose


Real-time singularity check
Geometric constraint
Grassmann singularity
Check stroke
Check angle
Close-loop structure must be valid
Platform's geometric must be valid
co-planar condition
Singulairty loci
line-line join condition
line-line meet condition
plane-line meet condition
plane-line join condition
Grassmann probability
Low rank = OK
High rank
External force
Structure collapse


Parametric sweep
Position check
Platform's orientation check
Singularity
Test data population
Checking the test point
Workspace validation
Geometric check
Grassmann check
Sweep method
Serial sweep
Parallel sweep
Prime factor or L-System sweep
Quaternion sweep
Extrusion sweep
Ternary algebra
Random generator (3D Mandelbrot)
3D graph
3D branching tree
Marching cube
Spherical / Helical / Polar sweep
Compliance workspace
Constant orientation workspace


Boolean algebra
Mixed logic
Operand
AND
NAND
NOR
L-System
Optimization
Simplification
Remove redundancies
Functions
Grassmann line, plane, and coplanar
Minimum distance
Interval analysis
Slicing
Obstacle
Mesh
Haptic engine
Flow control
Connected graph control
Growing fractal cluster
Topology straightening
Search control
Upper bound
Lower bound
Reach constraint / limit


Haptic controller
PPPRRR
Simple kinematic
Extendible
Minimal effort in producing a single-point position and orientation control's sensation
Sensor
Engine
human modality
RGB camera
9 DOF IMU
Motor voltage
3D Simplex
Grassmann pencil line
Voxel
Ternary / Binary Tree