Path Planning and Assembly Mode-Changes of 6-DOF Stewart-Gough-type Parallel Manipulators

Wesley Au[a], Hoam Chung[b], Chao Chen[b,*]

[a] *Centre for Autonomous Systems, University of Technology Sydney, 81 Broadway, Ultimo, New South Wales, Australia 2007*
[b] *Department of Mechanical Engineering, Monash University, Wellington & Blackburn Road, Clayton, Victoria, Australia 3800*

Abstract

The Stewart-Gough platform (SGP) is a six degree-of-freedom (DOF) parallel manipulator whose reachable workspace is complex due to its closed-loop configuration and six DOF outputs. As such, methods of path planning that involve storing the entire reachable workspace in memory at high resolutions are not feasible due to this six-dimensional workspace. In addition, *complete* path planning algorithms struggle in higher dimensional applications without significant customisations. As a result, many workspace analysis algorithms and path planning schemes use iterative techniques, particularly when tracking the manipulator's many direct kinematic solutions. The aim of this paper is to present the viability of singularity-free path planning in the Stewart-Gough platform's 6-dimensional workspace on modern-day computing systems by demonstrating its assembly mode-changing capability. The entire workspace volume is found using flood-fill algorithms with smooth and singularity-free trajectories generated within this known workspace. Workspace volume analysis was also performed with results comparable to other works.

1. Introduction

The Stewart-Gough platform [1, 2] (SGP) is a six degree-of-freedom (DOF) parallel manipulator with a six-dimensional (6D) workspace. Workspace analysis and path planning methods that involve storing the entire reachable workspace into memory at high resolutions is not feasible in most 6D systems due to memory constraints. As a result, path planning for this manipulator is not a straight-forward process. The assembly modes, direct kinematics and singularities of the SGP have been investigated in [3, 4, 5, 6, 7] and others. Due to the number of direct kinematics (DK) solutions it exhibits, up to 40 in its most general configuration [7], the configuration space must be constructed carefully to avoid issues with solution-finding and tracking in both trajectory generation and manipulator actuation.

Currently, the most viable method for path planning for the SGP is in the task space, or C-space constructed from task-space variables. Then, the direct kinematic solutions are tracked using iterative and approximation methods during joint space control. This is also typically useful when root-finding capabilities are limited or non-existent on embedded systems. A common method to track these solutions in path tracking is the Newton-Raphson algorithm [8]. Other iterative methods have also been developed specifically for use on parallel manipulator root-tracking [9]. These methods are efficient and useful in real-time path tracking, but always require an initial condition to operate. In [10], straight-line trajectories with numerically-calculated root-tracking was demonstrated. In [11], path planning for the 6-6 SGP is performed in the task or configuration space using short, linear paths to connect two configurations. These linear paths are generated using a local optimisation approach with a divide-and-conquer strategy. While successful, the performance of this path planning algorithm was never stated and the amount of workspace that could be utilised was never determined, nor was the maximal reachable workspace. Other path planners involve reconstructing singularity clouds in the configuration space as an obstacle, then generating a path using line geometry to avoid them [12].

[*] Corresponding author
Email addresses: wesley.au@uts.edu.au (Wesley Au), hoam.chung@monash.edu (Hoam Chung), chao.chen@monash.edu (Chao Chen)

There are many types of discretisation methods that exist for representing the C-space, with the scheme chosen dependent on the application. For workspace volume analysis, iterative techniques and algorithms are commonly used. A numerical algorithm was developed for finding the maximal singularity-free workspace in both positional and orientation space [13, 14], but it is based on the workspace's known geometry [15]. This process was also used for finding the volume of the reachable position workspace [16]. However, none of the above works ever mentioned performing any kinematic path planning on their results. Their work on workspace volume calculations however, can serve as a useful comparative guide to the accuracy of volumetric calculations from the algorithms in this work, albeit limited to fixed-orientation and symmetrical SGPs. A generalised numerical method for finding the singularity sets for the SGP is given in [17], using an adaptive discretisation technique. In [18], the workspace volume of various SGP geometries were calculated and compared to each other, also using an iterative process, in which they are also able to find the total orientation and total inclusive workspaces. A complete analysis of the reachable workspace of the 6-6 SGP is given in [19], with the use of kinematic mapping.

Where path planning is concerned, a more systematic approach to discretisation can be adopted with many of these schemes described in [20]. Beyond the simplistic discretisation technique of dividing the C-space into evenly-spaced intervals along each variable, adaptive discretisation techniques such as octree have been used successfully in 3D cases [21] where path planning is applied within the C-space. These structured discretisation techniques also allow flood-filling algorithms to be used, with common applications in computer graphics such as to identify connected regions bounded by a closed surface in a binary image [22]. Many of these algorithms can be modified and applied to higher dimensional space, where various numerical computational packages have successfully implemented them[1]. Assuming the volume of each discretised spatial element is known and the units are homogeneous, calculation of the volume of the space is a simple exercise of counting the connected elements in the data structure. Structured discretisation techniques also allow efficient path-finding algorithms to be used. Dijkstra's algorithm and its derivatives such as A* are commonly used in this setting, but as parallel manipulators grow in DOF, the time it takes for a graph-based search to execute grows exponentially. In recent years, rapidly exploring random tree algorithms and in particular, the probabilistic roadmap algorithm (PRM) were found to be efficient methods for path finding in robotics in higher dimensional workspaces [24, 25].

As the DOF and complexity of a parallel manipulator increases, the number of working modes and assembly modes also increases. Working modes exist when a single point in the task space represents multiple points in the joint space, or having multiple inverse kinematic (IK) solutions. In general, task space path planning often results in a small reachable workspace due to problems relating to end effector velocities at the workspace boundary (serial singularities) and the inverse kinematic solutions at those points. Some researchers have looked into methods for finding enhancing the reachable workspace by overcoming the serial singularities in various ways [26, 27]. Other works aimed to find the best working modes to perform tasks [28], such as minimising actuator energy [29]. The advantage of task space path planning is that for parallel manipulators in general, planning in the task space variables is easier to solve kinematically, but unfortunately have limited workspace potential as aforementioned. Joint space path planning can overcome this issue as it avoids the serial singularity problem, but introduces another significant challenge with the possible existence of multiple assembly modes.

Assembly modes exist when a single point in the joint space represents multiple points in the task space, or having multiple direct kinematic (DK) solutions. In other words, one joint posture can lead to many different postures of the end effector [30]. Generally, DK solutions are much harder to solve than IK solutions and are non-analytic when more than four DK solutions exist [31]. Further, the manipulator's workspace can exhibit one or more *aspect*, where each aspect is defined as the maximal reachable workspace that is singularity-free in the joint space [21, 32]. Trajectory planning between assembly modes is a very challenging problem for parallel manipulators, particularly for higher DOF manipulators due to a complicated parallel singularity profile. In a singular configuration, the control is lost at the end effector whereby the joint actuators are locked but the platform remains free to move, resulting in an increased DOF. Therefore a singularity-free trajectory between assembly modes cannot exist if they are located on different aspects. For example, the 2-DOF 5R manipulator only has two DK solutions and hence only two assembly modes exist. Because each assembly mode belongings to a different aspect, a singularity-free path between these assembly

[1]The MATLAB functions `bwconncomp()` and `imfill()` implement a flood-fill algorithm based on morphological reconstruction [23] which is compatible in spaces greater than three dimensions.

modes cannot exist [27]. However, this is not the case for higher DOF parallel manipulators [33], where a higher number of DK solutions is possible.

Planar Stewart Gough-type manipulators have had particular interest from researchers in this field as they have only two aspects [32, 34], but have up to six DK solutions. This means multiple assembly modes can exist within the same aspect, and hence have the capability of transitioning between assembly modes while remaining singularity-free [33]. Many of these methods utilised the reduced C-space, which is generally easier to formulate than the full C-space. Particularly for high-DOF parallel manipulators, it is more time-efficient to generate the reduced C-space, then utilise efficient path planning techniques for overcoming the problems with using this space. This has been widely studied, where different path planning methods to achieve this were explored [35, 36, 37, 38, 39]. In previous works [38, 41], the path planning problem was simplified on this type of manipulator using the global workspace roadmap, which is a graphical representation of the reduced C-space. This resulted in efficient path planning in that the existence of a singularity-free path can be pre-determined before path planning algorithms are applied. However, this scheme has only been validated up to three DOF. With the SGP exhibiting in excess of 16 DK solutions, constructing a C-space that connects all assembly modes in one manifold is a very challenging task, and to the best knowledge of the authors, has never been done.

By actively finding connections between assembly modes in the reduced C-space, it is perceived that the workspace is being enlarged (or enhanced) and the negative effects of using this space can be mitigated to a degree. This is because not all configurations are mapped uniquely in the reduced C-space. In the full C-space space, it becomes a trivial task as all DK solutions are mapped one-to-one and fully defined, but generating this space, particularly for higher DOF parallel manipulators can be very challenging and wasteful in computational resources depending on the base variables chosen. Some of these problems can be alleviated with mapping techniques such as kinematic mapping [40]. However, it is seen that for the 6-DOF Stewart-Gough platform used in the following examples, the full C-space can be very easily obtained.

The aim of this work is to evaluate the workspace and perform path planning for an assembly mode-change on the 3-3 Stewart-Gough platform, utilising its full 6-DOF workspace. The first task is to demonstrate the feasibility of flood-filling algorithms to identify the overall reachable workspace in six dimensions, and to calculate the volume of the total connected C-space based on this application. Comparisons to other works [16] will be drawn to verify these results.

For the path planning portion of this work, the PRM algorithm is used in a six-dimensional environment to determine a trajectory to perform a singularity-free assembly mode change. It is shown that the choice of variables in constructing the C-space can significantly reduce the complexity of path planning, as is shown in the assembly mode-change example that follows. These aims are ratified in an example in Section 5, and the work is an extension of [42]. In all examples, it was shown that these methods are viable on standard desktop systems, where it must be feasible in memory utilisation and computational time, although what is deemed to be feasible time is dependent on the individual.

The path planning process consists of four main steps, which is highlighted in Figure 1:

1. Discretise workspace into cells
2. Evaluate workspace data for each cell, applying any constraints to the manipulator
3. Use a flood-fill algorithm to find connected regions of reachable workspace for each aspect
4. Given two points in task space, plan a path in the connected reachable workspace using PRM algorithm

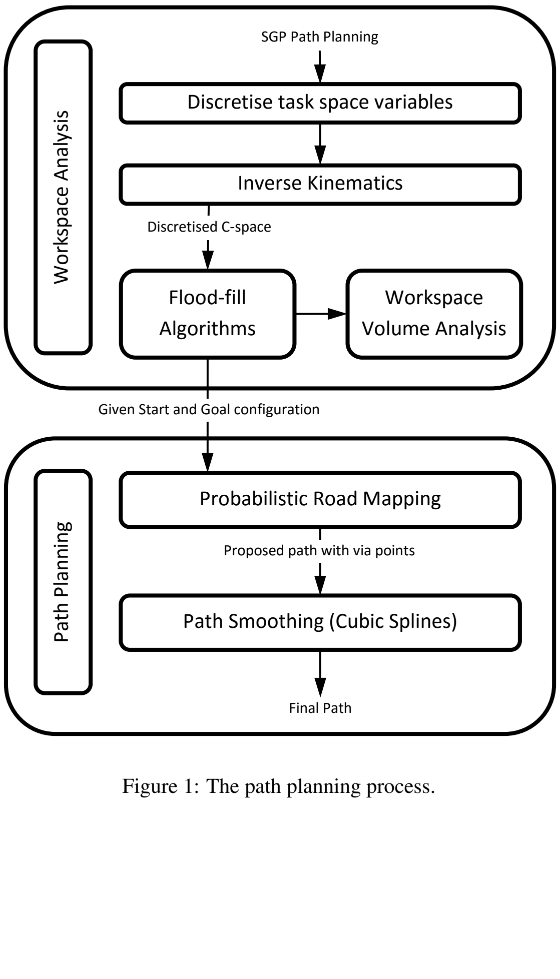
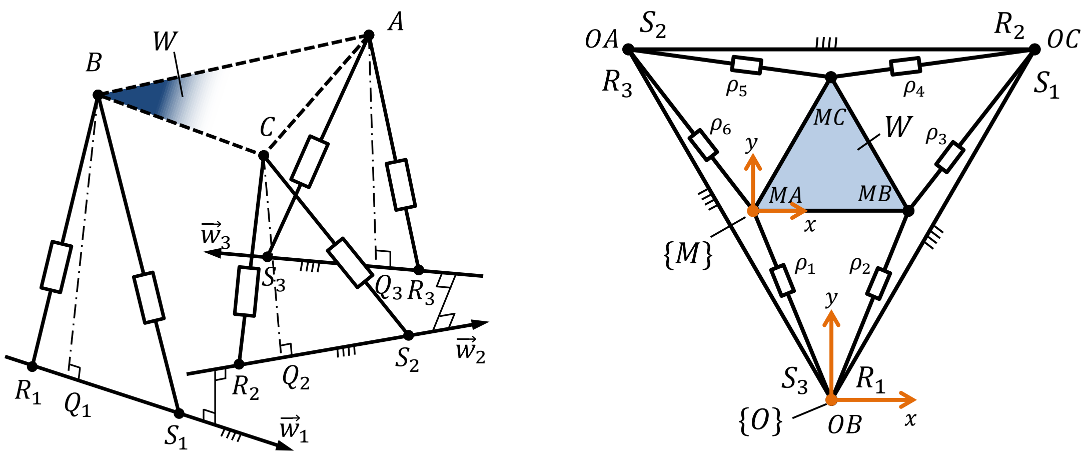

SGP Path Planning
Workspace Analysis
Discretise task space variables
Inverse Kinematics
Discretised C-space
Flood-fill Algorithms
Workspace Volume Analysis
Given Start and Goal configuration
Path Planning
Probabilistic Road Mapping
Proposed path with via points
Path Smoothing (Cubic Splines)
Final Path

Figure 1: The path planning process.



(a) Generalised 6-3 SGP

(b) z−planar view of the symmetrical 3-3 SGP with base frame $\{O\}$ and moving frame $\{M\}$.

Figure 2: The Stewart-Gough platform and the 6-3 and 3-3 configurations.

The structure of this paper follows the logical sequence of steps as laid out by the path planning algorithm process; after the definition of the 6-3 SGP in Section 2, workspace analysis is discussed in Section 3 where discretisation (steps 1. and 2.), the configuration space and flood-fill algorithms (step 3.) are discussed. Then follows Section 4 which discusses path planning techniques used (step 4.) and finally, Section 5 lists some numerical examples of the uses of the path planning algorithm as discussed in the overview of this work.

2. The Stewart-Gough Platform

In this work, examples are shown for the 3-3 Stewart-Gough platform, but the kinematics are derived from the 6-3 configuration to maintain generality with application for the 6-3 SGP. In the 6-3 configuration, the manipulator consists of a planar base defined by points $R_{1,2,3}$ and $S_{1,2,3}$ and a moving platform W. The base frame $\{O\}$ is attached to the base and frame $\{M\}$ is attached to the moving frame, at which the end effector is measured from. The moving frame is actuated with six actuated legs in an $S\underline{P}S$ configuration, connected between R and S on the base, to points $P1, 2, 3$ on the moving platform (Figure 2a). The output platform W is capable of a 3-axis translation and 3-axis rotation output for a total of 6-DOF. Let $\mathbf{q}$ be a six-dimensional vector consisting of ρ_i that describes the extension of the i-th leg (distance between R or S and P) and let $\mathbf{x}$ be a six-dimensional vector describing the configuration of frame $\{M\}$; the first three variables (x, y, z) defines the translation relative to frame $\{O\}$ and the last three variables (α, β, γ) describes the orientation of frame, using the $Z-Y-X$ Euler angle scheme. The overall reachable workspace is restricted due to the box-constraint applied to $\mathbf{q}$.

2.1. Symmetric 3-3 SGP Parameters

The parameters of the 6-3 SGP were chosen such that it represents a symmetric 3-3 configuration, where S_3 and R_1, S_1 and R_2 and S_2 and R_3 are paired accordingly (Figure 2b). From now on, any mention of the 3-3 SGP implies the symmetric property. In this configuration, the moving platform is still capable of the full 6-DOF movement, but its kinematics is simplified. In the SGP's most general configuration, there exists up to 40 direct kinematic (DK) solutions [7], but for both the 6-3 and 3-3 SGP, this is reduced to 16 DK solutions [43].

Just four parameters can be used to define the mechanism's dimensions with symmetry: $t_{1,2,3,4}$. The base passive joint definitions are

$$R_1 = S_3 = OB = [0, 0, 0] \qquad R_2 = S_1 = OC = [t_1, t_2, 0] \qquad R_3 = S_2 = OA = [-t_1, t_2, 0] \tag{1}$$

and the passive joints on the moving platform W relative to $\{M\}$ are

$$MA = [t_3, t_4, 0] \qquad MB = [0,0,0] \qquad MC = [2t_3, 0, 0] \tag{2}$$

The inverse kinematics (actuator lengths) is solved by

$$\overline{MA\ OB} = \rho_1 = \left\|{}^{O}MA\right\| \tag{3}$$
$$\overline{MB\ OB} = \rho_2 = \left\|{}^{O}MB\right\| \tag{4}$$
$$\overline{MB\ OC} = \rho_3 = \left\|{}^{O}MB - OC\right\| \tag{5}$$
$$\overline{MC\ OC} = \rho_4 = \left\|{}^{O}MC - OC\right\| \tag{6}$$
$$\overline{MC\ OA} = \rho_5 = \left\|{}^{O}MC - OA\right\| \tag{7}$$
$$\overline{MA\ OA} = \rho_6 = \left\|{}^{O}MA - OA\right\|, \tag{8}$$

where actuated system variables $\mathbf{q}$ are ρ_1 to ρ_6. The output co-ordinates $\mathbf{x}$ are measured at frame ${}^{O}M(x, y, z, \alpha, \beta, \gamma)$.

2.2. Jacobian Analysis

The singularity profiles can be determined by utilising the well-known equation for Jacobian analysis

$$\mathbf{A}\dot{\mathbf{q}} = \mathbf{B}\dot{\mathbf{x}} \tag{9}$$

where it is observed that

$$\begin{aligned}\det \mathbf{A} &= f(\rho_1, \rho_2, \rho_3, \rho_4, \rho_5, \rho_6)\\ &= f(\mathbf{q})\\ &= \rho_1\rho_2\rho_3\rho_4\rho_5\rho_6\end{aligned} \tag{10}$$

and

$$\begin{aligned}\det \mathbf{B} &= g(x, y, z, \alpha, \beta, \gamma)\\ &= g(\mathbf{x}),\end{aligned} \tag{11}$$

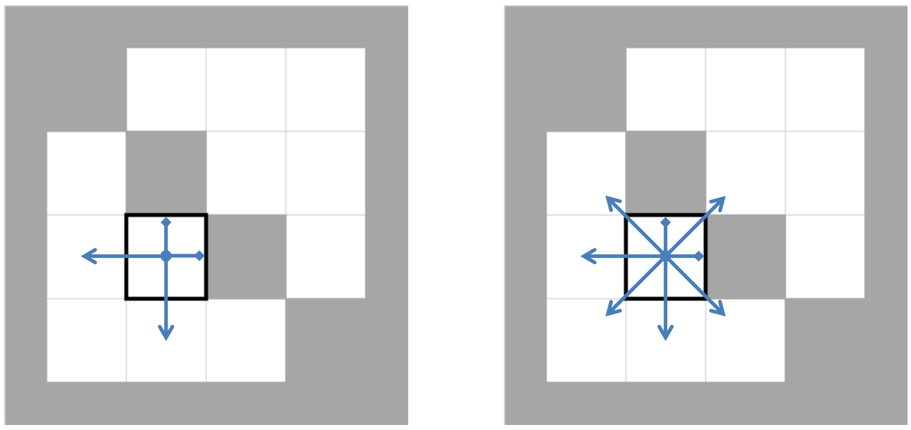
where $\mathbf{B}$ is a 6×6 matrix comprised of many non-linear elements. These elements are shown in Appendix B.

The serial Jacobian $\mathbf{A}$ as shown in Equation (10) indicates that provided linear actuator displacements remain greater than zero, a practical SGP will never encounter a serial singularity. The parallel Jacobian matrix expressed in Equation (11) is non-linear, but depends on task space variables only. This feature implies that the maximum reachable workspace in the task space for this mechanism can only be defined by parallel singularities, if the condition $\rho_{1,\ldots,6} > 0$ is maintained.

3. Workspace Analysis

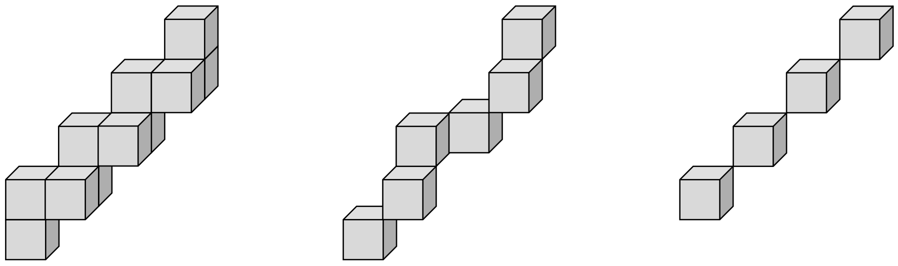
The n_d-dimensional C-space in the path planning process is uniformly discretised along each variable. Each element in this discretised space is identified as a *cell*, which represents a unique configuration of the manipulator. The property of a cell is determined by evaluating the kinematic properties of the centre of the cell. It is only during path planning analysis that the cell may be further discretised for accuracy in regard to singularities. This method of discretisation was chosen because it allows easy implementation of searching algorithms. The minimal-connectivity scheme was implemented where adjacent cells can only be identified as connected through adjacent faces. This restricts path planning between fully defined features of a cell, rather than infinitesimally small points such as vertices and edges (in spaces greater than 2 dimensions). A two dimensional case is shown in Figure 3, where minimal connectivity restricts connections between defined edges of the cell. Shaded regions indicate obstacles, or in the case of a manipulator, unreachable or singularity configurations. It can be seen in Figure 3b that ideally, the entire workspace should be split into two connected regions. A minimal 4-connectivity scheme maintains this separation, but higher connectivities do not. The connectivity schemes in three dimensions are shown in Figure 4, where the minimal connectivity scheme is 6-connected [22].

Because the Stewart-Gough platform is a 6-DOF mechanism, care must be taken not to use a grid that is too fine as memory usage exponentially grows.



(a) 4-connectivity (b) 8-connectivity

Figure 3: Obstacle definitions with 4 and 8-connectivity in 2D.



(a) 6-connectivity (b) 18-connectivity (c) 26-connectivity

Figure 4: 6-, 18- and 26-connected paths in 3D discretised space.

3.1. Constructing the Configuration Space

Path planning in general occurs in the C-space where all possible configurations of the manipulator are encoded in a generalised set of co-ordinates. The boundary of the configuration space represents the parallel singularities or any other physical constraints where a path cannot be planned.

A general SGP is configured in a 6-6 arrangement [3] (six groups of passive spherical joints on the base and six under the moving platform) and it is known to have up to 40 direct kinematic (DK) solutions [7] - these solutions are called assembly modes [44]. It was generally assumed that these assembly modes were separated by parallel singularities but has since been proven incorrect [33]. However, in deriving the C-space from the joint space variables (known here as the C_θ-space), it is very difficult to identify the connectivity between so many assembly modes to construct one continuous configuration space that is singularity-free. The goal is to always generate a C-space where all points represent a unique, fully-defined configuration of the manipulator, but generating it from the joint space variables in this case is not an efficient way to achieve this.

The SGP has only one inverse kinematic solution in its task space, where one point in this space represents a single solution of the manipulator (one-to-one mapping). Because of this property, the task space variables can be implicitly used as the full C-space for path planning. However in doing this, when a path is generated and actuated, DK solution tracking is required because the output platform configuration is difficult to measure in practice. Because task space variables $\mathbf{x}$ were used as the basis variables for the full C-space, from now on it will be defined as the C_x-space, or the full configuration space generated from the task space variables.

3.1.1. Representation of the Discretised C-space

During construction of the discretised C_x-space, each cell is assigned an identifier to indicate which aspect it belongs to, based on the C_x-space variables it defines (centre of the cell). Where adjacent cells differ in aspects, it is likely that there is a singular region hidden between the cell centres due to the resolution of discretisation. In general, it cannot be guaranteed that all configurations that can exist inside the cell are singularity free or belong the the same aspect. However in path planning, only linear paths between cell centres are used. In order to ensure singularity-free path planning, any time a cell is connected during path planning, singularity checks are carried out at a high resolution along these linear paths to ensure a singularity does not exist. If singularities exist, then the C_x-space can be updated so that these cells must remain separated.

3.2. Data Evaluation

Because the workspace is in six dimensions, data should only be generated and stored if necessary to avoid memory overflows. In the interests of path planning, the most important thing to evaluate and store is the information on Jacobians, i.e. the result of evaluating Equation (11). If the sign of this equation is evaluated, then connected regions of the same sign indicates the maximal reachable workspace for that *aspect* [34].

An important observation to note is that Equation (11) depends only on the task space variables, due to the SGP exhibiting only one IK solution. This saves a significant amount of time as the joint space variables do not need to be evaluated at the same time. This also means that the task space variables do not need to be stored, saving memory. Barring the extra variables required to process the entire workspace, the approximate memory usage per grid size for storage of an 8-byte double-precision matrix when stored in memory is calculated using the following equation:

$$\text{Memory (bytes)} = 8 \times n^6, \tag{12}$$

where n is the number of discretised elements along each dimension. Even when each axis is discretised into 35 elements, the amount of memory needed to store a double-precision matrix approaches 15 GB. Due to overhead in storing data structures or other sub-procedures utilising other parts of memory, it is highly unlikely that this level of discretisation can be achieved ($n = 21$ in following examples).

3.3. Finding the Connected Workspace - Flood-filling

There are many types of flood-filling algorithms that can be used to identify connected regions of similar properties. These can range from simple scanned lines along a particular variable to more complex tree type fill searches and image morphology methods [23]. While image morphology is a very useful tool within the image processing community, it does not share the same attention in the robotics community, particularly in workspace analysis where its application is quite similar (aspect separation). It's not to say, though, that these algorithms have not been used in robotics before. For lower DOF manipulators, simple flood filling algorithms such as depth or breadth first searches are sufficient. Where higher dimension workspaces need to be analysed, there is the need to use more efficient algorithms.

In previous works [38], the rotary disk search was used for sorting and stitching multiple kinematic solutions together via a simple continuity function. This was a weighted depth-first search that favoured traversing away from the known boundaries of workspaces to improve the accuracy of solution-tracking. Relative performance of this algorithm was sufficient, particularly in handling up to 6 DK solutions in a three dimensional space. For manipulators that have only one solution in the kinematics, such as the 6-DOF SGP with one inverse kinematic solution, the solution-matching component of the RDS algorithm becomes redundant and hinders performance due to overhead associated with maintaining a data structures within the algorithm. In this case, it is better to find an algorithm that is optimised for purely finding the connected workspace within known boundaries, and thus the need to find alternative flood-fill algorithms.

A flood fill algorithm, also known as a seed fill algorithm is typically used in computer graphics [22]. Paint programs are a common application of this, where a user may wish to fill a drawn, fully enclosed shape with a uniform colour. These algorithms work on discrete datasets such as binary images and up to 3-dimensional spaces [45], but are shown to be effective and efficient if they can be adapted to higher dimensional spaces. There are many types of flood fill algorithms [22], which all have their advantages and disadvantages. The flood fill algorithm used in the experiments were implemented using MATLAB's built-in function `bwconncomp()` in MATLAB's Image Processing Toolbox and is capable of analysing n_d-dimensional images.

3.4. *Workspace Volume*

Because the entire reachable workspace is discretised and flood-filled to find all connected and singularity-free regions in that space, this information can be easily used to determine the volume of the reachable workspace in up to six dimensions. However, because this 6-dimensional space contains both position and orientation co-ordinates, a single value to represent the volume of this space is not feasible, i.e counting the number of cells for instance. Typically, the volume of the positional workspace is of only concern, hence the 6-dimensional space should be condensed to 3-dimensional by considering the positional space only. This can be done in the discretised space; for each point p in the position space (x, y, z), if the platform contains any valid points in the orientation space (α, β, γ), then point p is considered for volume calculation. It is assumed that the volume of the reachable workspace can only be found if the actuator lengths are constrained; if not, then the reachable workspace volume may become infinite.

4. Path Planning

4.1. *Probabilistic Road Map Algorithm*

As parallel manipulators grow in DOF, the time it takes for a graph-based search to execute grows exponentially. In recent years, the PRM has been the go-to method for path finding in robotics as it performs relatively well in higher dimensional workspaces [24, 25]. The PRM algorithm is a sampling-based path planning scheme which samples random points within a connected set of points and builds a tree based on collision-free straight-line trajectories between these sampled points. Sampling-based path planners have the advantage of speed over *complete* algorithms in that weaker notions of completeness are tolerated (resolution and probabilistic completeness) [20]. Because the workspace is in six dimensions for the SGP, complete algorithms will inherently take very long to complete simply due to time complexity and the sheer amount of data needed to process. Sampling-based planners are iteration-based and in general, the probability of finding a solution converges to one as the number of samples approaches infinity. As such, it was shown that PRM planners work well in solving difficult motion planning problems, with detailed analysis of the algorithm shown in [25].

Path costing, or heuristics are regularly used to help the path planner generate better-conditioned paths in the workspace in addition to obstacle avoidance [46]. Although feasible paths are normally found in a matter of milliseconds in smooth and well-conditioned data sets, the execution of the PRM algorithm is generally allowed to continue to smooth out the path.

This algorithm has a time complexity of $O(n \log n)$.

4.1.1. *Heuristics*

In order to generate paths that actively avoid singularities, a cost function is added in the PRM algorithm during tree generation. The simplest way to cost each linear path is to associate a cost based on distance from a boundary cell for each cell within the connected workspace, also known as adding a heuristic function to the planner. Heuristics is regularly used to help the path planner to generate better-conditioned paths in the workspace in addition to obstacle avoidance [46]. While this has no implication on the time taken to find a valid path in the connected space, with continuing iterations beyond the path-existence termination condition, the path found will eventually converge to a balanced solution regarding path cost and path length. It is observed that this generally smooths trajectories generated in the task space.

4.1.2. *Process*

Given an initial node, goal node and a finite discretised space, the algorithm begins by creating a straight-line path between any open nodes to any existing nodes. In this case, it simply checks if there is a collision-free[2] path between the initial node and goal node. If a collision-free path exists, then the path is complete; otherwise, create a new node at a randomly sampled point in the discretised space. Linear paths are then created from this new node to all existing nodes (initial and goal nodes) to check for collision-free connectivity. If there is a valid connection, then this node is marked as connected on a connectivity matrix, which can include heuristic or weighting data. The connectivity

[2]For manipulator path planning, collision-free implies singularity-free.

matrix is then evaluated to check for connectivity between the initial and goal node. If there is no connection, then the process loops again.

In checking for initial and goal node connectivity, there are three possible conditions to check:

- A path does not exist; sample next random point in the discretised space
- A path exists between the initial and goal node; algorithm finishes
- A path exists between the initial and goal node; algorithm continues to find a better solution.

The final condition allows the algorithm to continue to sample random points. As more points are sampled in the discretised space, the solution will converge to a shortest path solution. If weighting is factored into the connectivity matrix, then the quality of the solution depends on the cost criteria of the solutions found in the connectivity/weighting matrix. For example, if paths are weighted to avoid boundaries of workspace patches, then the lowest-scoring solution will contain the path that is optimised in both path length and boundary avoidance.

The algorithm used to find the connectivity between the initial and goal nodes is currently implemented by MATLAB using Dijkstra's algorithm in the *Bioinformatics Toolbox*, called `graphshortestpath()`. This function analyses the connectivity matrix with weighting data given and returns the list of nodes sampled in the discretised space as a lowest-cost collision-free (singularity-free) path. The pseudo-code is shown in Algorithm 1.

4.1.3. Algorithm Definition

In general, node $N \equiv$ cell C. Let W be the set of all connected nodes in the reachable C-space. In the context of piecewise paths between an initial and goal node, $N_i \in W$ and $N_g \in W$ respectively, let $P \subset W$ be an ordered set of nodes where the first node in the set is N_i, the last node in the set is N_g and nodes in between represent the via nodes in the piecewise path. Therefore between nodes $N_j \in P$ and $N_{j+1} \in P$ where j is the j-th node in P, exists a fully connected linear path, comprised of an ordered set of cells $L_j \subset W$ where adjacent cells $C_k \in L_j$ and $C_{k+1} \in L_j$ is also adjacent in the C-space.

Inputs

W: set of all connected nodes in the reachable C-space
N_0: initial node, $N_0 \in W$
N_1: goal node, $N_1 \in W$
n: number of nodes in W to sample

Algorithm Variables

N_s: set of nodes sampled
R: $(n \times n)$ connectivity and weighting matrix

Sub-Functions

`Path`(N_0, N_1): returns an ordered set of cells that represents a linear, fully connected path between nodes N_0 and N_1
`Cost`$(\{N_0, ..., N_1\})$: returns the cost of the ordered set of cells representing a linear fully connected path
`RandomNode`(W): returns a random node in set W
`graphshortestpath`(N_0, N_1, R): MATLAB-implemented function that evaluates the connectivity matrix R for connectivity between nodes N_0 and N_1. It returns an ordered set of nodes that represents a piecewise, fully connected path from N_0 to N_1 of lowest cost. Returns $\emptyset$ if no path exists.

4.2. Path Smoothing

The side effect of discretisation is that for any linear path in space, it is made up of a series of small discretised steps linking the start and end points together. While this kind of trajectory is typically acceptable for very fine grids (for example, 200 steps between 0 to 2π for a rotational joint), when discretisation is very coarse, such as for the case of the SGP task space, linear paths become very jagged where each step can almost be considered discontinuous due to the large step size. This has flow-on effects when the trajectory is mapped to other spaces, such as mapping from task to joint space. This can cause problems where both task and joint space variables are needed to calculate

Algorithm 1 PRM Search

```
function PRM($W, N_0, N_1, n$)
    $N_s \leftarrow \{N_0, N_1\}$
    if Path($N_0$, $N_1$) $\neq \emptyset$ then return $N_s$
    end if
    for $i \leftarrow 2, n$ do
        $N_i \leftarrow$ RandomNode($W \setminus N$)
        for $j \leftarrow 0, i-1$ do
            if Path($N_j$, $N_i$) $\neq \emptyset$ then
                $R(i, j) \leftarrow$ Cost(Path($N_j$, $N_i$))
            end if
        end for
        $N_s \leftarrow N_s \cup N_i$
    end for
    return graphshortestpath($N_0$, $N_1$, $R$)
end function
```

Jacobians for workspace analysis. To alleviate this problem, some level of post-processing is needed to improve the quality of the results.

4.2.1. Interpolating Between Cells

A simple, finely discretised linear spline is used between each regularly discretised point on the original trajectory. For example, if a trajectory requires the manipulator to move from 0 to 0.5 along a dimension in task space, then a linear trajectory that moves from 0 to 0.5 along s steps is created, where s is typically chosen to balance path planning performance and accuracy needed to calculate Jacobians. This means that for a trajectory of length c cells, the total number of points along the post-processed path n_p is

$$n_p = s(c-1) - 1. \tag{13}$$

In the numerical experiments, chosen $s = 10$ was chosen, which provides enough points for the final path to be considered smooth in both task and joint spaces. This should also alleviate physical issues in actuator movement as via points are heavily condensed along the proposed trajectory.

4.2.2. Cubic Splines

Continuing the path smoothing process from Section 4.2.1, for a single trajectory along a single dimension in task space, an n_s number of points can be sampled along this path. Remember that for a path of length c-cells, the total number of discretised points along the entire path for a single dimension is given by Equation (13).

The process of cubic-splining each trajectory for each dimension is as follows.

1. Sample n points along the entire trajectory after initial smoothing in Section 4.2.1.
2. Solve for the cubic coefficients, given conditions of the via points.
3. Regenerate the path based on cubic spline coefficients, discretised in d-points for each cubic spline.

The new number of points along each dimension is

$$n_{p-cubic} = d \times n_{cubics} + 1. \tag{14}$$

From this point on, the independent time variable is now arbitrary (given as *Time Step* in all trajectory figures). This arbitrary time step unit can be scaled to fit any length of time as necessary to fit the constraints of the manipulator, such as its dynamics.

There is an inherent danger of introducing cubic splines if the path is ill-conditioned, ie. close to singularity. Due to the continuous nature of the splines, any segment of path that was originally close to singularity, the final solution may force the path to cross this singularity order to satisfy the continuity constrains of these splines. This results in an invalid path which cannot be used for singularity-free movement of the end-effector. Increasing the resolution of the proposed splined path (i.e. increasing the via points) does not alleviate the problem, but rather exacerbates the issue of path smoothness. Because cubic splines were intended to smooth out the original discretised path, a low number of via points are sampled from the original path. The number of via points varies due to path length, but in general through trial and error, 9 via points (excluding start and end points) was deemed suitable for path-smoothing.

4.2.3. Checking for Path Validity

Although via points are set by the PRM, the path may be perturbed from the original set path between these via points by the path smoothing process. To check for singularities, the final path is discretised to a high resolution and each point is checked for sign changes in the determinant of the Jacobians. At this stage, if the path smoothing process has introduced any singularities, then the path is discarded and the path is re-calculated by the PRM.

5. Example: 3-3 Symmetric SGP

The following examples utilise the 6-3 Stewart-Gough platform, configured in a symmetric 3-3 arrangement. From now on, all references of the 3-3 SGP refer to its symmetric configuration. Referencing Equations (1) to (2) and Figure 2b, the t-parameters used in the following examples are

$$t_1 = \frac{1}{\sqrt[4]{3}} \qquad t_2 = \sqrt[4]{3} \qquad t_3 = \frac{3}{5\sqrt[4]{3}} \qquad t_4 = \frac{3\sqrt[4]{3}}{5}. \tag{15}$$

For all workspace analysis examples, the following joint space (leg lengths $\rho_{1..6}$) are box-constrained to

$$0.917823 < \mathbf{q} < 2.134458. \tag{16}$$

For all path planning examples, the task space is discretised along each spatial dimension in 21 intervals between

$$x = [-3, 3] \qquad y = [-3, 3] \qquad z = [0, 2] \qquad \alpha = [-\frac{\pi}{2}, \frac{\pi}{2}] \qquad \beta = [-\frac{\pi}{2}, \frac{\pi}{2}] \qquad \gamma = [-\frac{\pi}{2}, \frac{\pi}{2}]. \tag{17}$$

The computer system used in all experiments, unless otherwise stated, utilised an Intel i7 2600K processor at 4.5 GHz with 16 GB of RAM, running MATLAB 2012b with Windows 7 x64.

5.1. Volume Analysis of Position Workspace with Constant Orientation

In order to analyse the workspace for path planning, the physical boundaries of the workspace should be determined by finding all singular points in a defined workspace. The task space variables are box-constrained to

$$x \in [-3, 3] \qquad y \in [-3, 3] \qquad z \in [0, 2] \tag{18}$$

and the end effector orientation set constant to

$$\{M\}(\alpha, \beta, \gamma) = [0, 0, 0]. \tag{19}$$

The results of the path planning process are shown in Table 1, and are separated into two columns - unfiltered and filtered. The unfiltered workspace represents the entire workspace that is of the same aspect. Due to discretisation of the workspace, it is inevitable that the workspace around the boundary will become fragmented, hence the filtered workspace column represents the usable volume of the workspace that the path planner can reach and utilise, where fragments of workspace no longer reachable by the path planner are removed. The filtered workspace will return smaller volume than the unfiltered workspace as a result.

Comparatively, the results of the workspace volume generated by [16] were iteratively calculated, with the final result deemed accurate once convergence was achieved. With the same parameter setup, the volume calculated in their work was 2.758013 units, calculated in 7 seconds[3].

[3]With a convergence of $\epsilon = 10^{-5}$

Cells per dimension	Unfiltered		Filtered		Time taken (s)
	Vol (unit^3)	Cell count	Vol (unit^3)	Cell count	
11	2.736000	38	2.376000	33	0.18
21	2.799000	311	2.763000	307	0.27
51	2.707776	4701	2.702016	4691	1.97
101	2.710872	37651	2.709864	37637	9.98
201	2.712906	301434	2.712609	301401	69.50
501	2.712505	4709210	2.712474	4709156	1079.14
801	2.712406	19288219	2.712369	19287955	4479.34

Table 1: Volume of the reachable workspace as using the path planning algorithm with constant orientation $(\alpha, \beta, \gamma) = [0, 0, 0]$. Results of the 501 and 801-discretised space were generated on the Monash Sun Grid cluster to cope with the additional computational requirements.

It is observed that while the value of the volume of the workspace seems to have converged to 3 decimal places, any additional refinement of discretisation does not bring the value significantly closer to 2.758 units. However, the results are feasible as it truly represents the volume of workspace that is *useful*, in that the path planner can utilise this entire volume for trajectory planning. Hence the result of the 201-discretised workspace can be deemed appropriate as it returns a relatively accurate value for the volume in a reasonable amount of time.

Reasons for the discrepancy between the calculated result and the result in [16] could be attributed to the unreachable regions of workspace due to discretisation of the workspace. The workspace manifold is shown in Fig 5. It must be noted that Gosselin's algorithm is able to find the volume of the reachable workspace without box-constraining any task space variables. This is not the case here, hence the task space variables must be box-constrained.

As shown in Figure 5b where the features on the underside of the reachable workspace show seven tapered regions. While the algorithm used in [16] is able to refine its discretisation to handle this profile, due to the nature of the fixed discretisation scheme used, this cannot be accounted for and hence information can be lost, resulting in a smaller volume calculated.

5.2. *Volume Analysis of Position Workspace with Bounded Orientation*

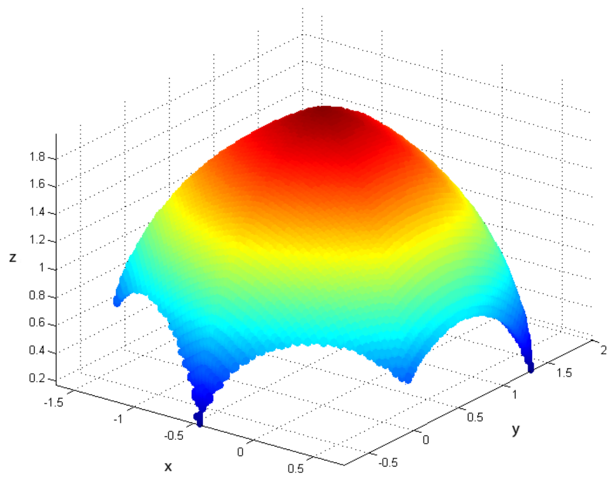
The volume of the reachable positional workspace is calculated by condensing the orientation workspace into a single point. For a point in the reachable position workspace, if any orientation of the platform exists for this position, then this point in position space is valid. This will give the volume of the position workspace with the orientation bounded. The algorithm is broken down as follows:

1. Define the entire workspace as $P^3 \times O^3$ where P is the position space and O is the orientation space. Every point in P-space has an O^3 workspace.
2. Choose a point p in P-space.
3. If its associated O^3-space contains any valid points (any valid orientations), then point p in the P-space is a valid point, i.e. a valid point in the position space.
4. Repeat steps 2 and 3 until all points are sampled in the $P-$space.

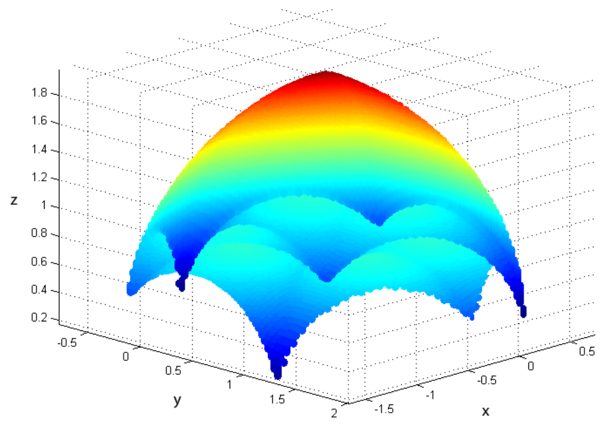
This process can be time consuming, especially if a precise measurement of volume is required. Although efforts were made to keep memory usage to a minimum but maintain the parallelisation of the code, this inevitably results in longer code execution time. To justify a self-imposed time constraint in waiting for a reasonable result, high-performance computing should be used such as a cluster grid.

Table 2 shows the workspace volume of the 3-3 SGP whose task space is box-constrained to

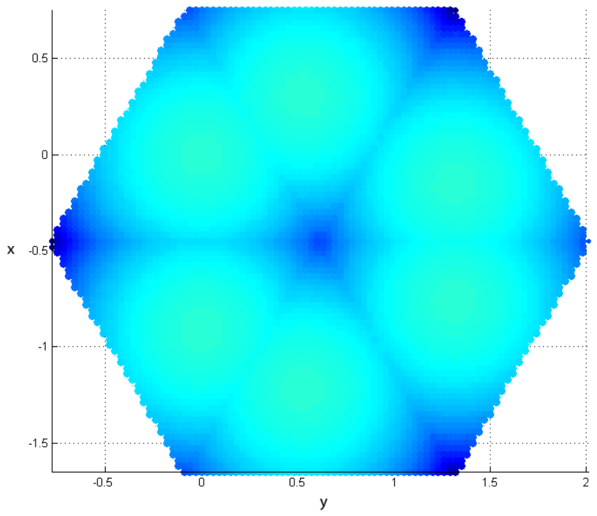
$$\begin{aligned} x &= [-3, 3] & y &= [-3, 3] & z &= [0, 2] \\ \alpha &= [-\frac{\pi}{6}, \frac{\pi}{6}] & \beta &= [-\frac{\pi}{6}, \frac{\pi}{6}] & \gamma &= [-\frac{\pi}{6}, \frac{\pi}{6}], \end{aligned} \tag{20}$$



(a) Outer profile



(b) Bottom profile



(c) z-plane bottom view

Figure 5: Reachable workspace of the 3-3 SGP.

and the actuator lengths are constrained as defined in Equation (16). All results were generated on the Monash Sun Grid cluster so that processing times can be compared between all discretisation levels; discretisations 11 to 26 can be processed with a high-performance desktop.

As observed in Table 2, the volume of the workspace has difficulty in converging due to a coarse discretisation grid, but in fact the calculated volume is rather consistent between all resolutions of discretisation. The volume fluctuates by 3.81% in the negative aspect and 0.96% in the positive aspect. With consistent results in the three-dimensional volume and based on convergence of the 21-discretised space in the two-dimensional volume, it is evident that the 21-discretised space strikes the right balance between volume accuracy and algorithm execution time.

Figs 6 show the total reachable workspace profile of the 51-discretised workspace. The observed workspace is quite smooth, but still contains recesses underneath the workspace as was observed in the orientation-constrained case (Figure 5c). In this case, there are only two visible recesses.

Cells per	Negative Aspect		Positive Aspect		Time taken
dimension	Vol (unit^3)	Cell count	Vol (unit^3)	Cell count	
11	1.944000	27	5.544000	77	14s
16	2.816000	132	5.482667	257	2m 12s
21	2.925000	325	5.517000	613	10m 58s
26	2.884608	626	5.488128	1191	39m 51s
31	2.874667	1078	5.464000	2049	1h 47m
41	2.919375	2595	5.510250	4898	9h 19m
51	2.926080	5080	5.512896	9571	1d 12h

Table 2: Volume of the reachable workspace as using the path planning algorithm, with the orientation of the platform box-constrained.

	Total Cells
Positive	85,766,121
Negative	73,089,637
Total Usable	85,700,522
Unreachable	65,599
Entire Workspace	85,766,121

Table 3: Cell counts of the workspace analysis and generation process.

5.3. Path Planning - General Path

5.3.1. Workspace Analysis

After processing the entire workspace that consists of $21^6 = 85,766,121$ cells in the C-space, Table 3 shows the cell counts for the positive and negative aspects, as well as other totals. In total, there are 65,599 cells not reachable by the path planner, which accounts for only 0.0765% of all cells in the defined workspace. This can be attributed to noise introduced by discretisation.

The flood fill algorithm searches for connected cells associated with the positive and negative aspects in the entire task space - this defines the maximum reachable workspace. A minimum of two sets of connected cells was expected, one for each aspect in positive and negative. It was also anticipated that there will be multiple connected sets of smaller workspaces for the positive and negative aspects as seen for other parallel mechanisms such as the 3-R$\underline{P}$R and 3-RRR [38, 39]. However for the parameters given for the 3-3 SGP, there seem to be *only two connected workspaces*. Other similar parameters were tried and all cases came up with only two connected workspaces, one for the positive and one for the negative aspect. It can be concluded from this observation that there is evidence to support the hypothesis that the SGP's configuration space is always made up of two aspects.

The time taken to evaluate the Jacobian for all cells is 352 seconds and time taken to use the flood fill algorithm to find all workspaces is 166 seconds for a total pre-processing time of 518 seconds.

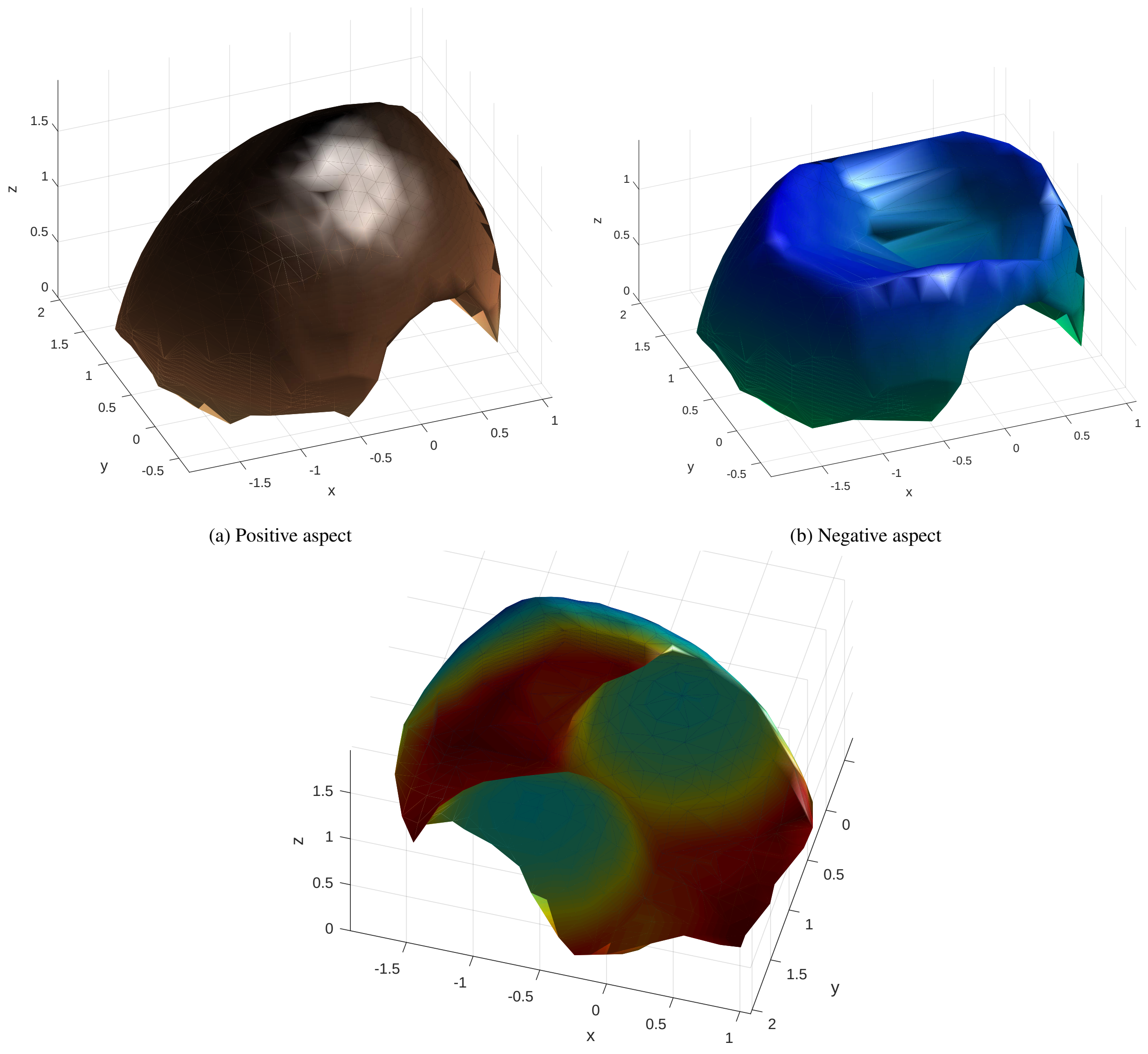
5.3.2. Results

The proposed configurations in task space to plan a path between in $(x, y, z, \alpha, \beta, \gamma)$ co-ordinates ($Z - Y - X$ Euler angles) is:

$$p_1 = \begin{bmatrix} -0.3 & 1.2 & 1.4 & -27^\circ & 9^\circ & -9^\circ \end{bmatrix} \qquad p_2 = \begin{bmatrix} -0.6 & 0.6 & 0.9 & -9^\circ & -54^\circ & 0^\circ \end{bmatrix} \tag{21}$$

The PRM planner completed in 158 iterations with 1 via point. A path was found in 0.004 s, and converged to the best solution in 6.826 s.

Fig 7 show that the path generated is successful in moving from point p_1 and p_2 in the task space. The trajectory is smoothed via cubic splines where 11 interpolation points were used, 2 end points and 9 via points. The start and end points for each cubic are located at every 10th unit of time. While the 21-interval discretisation is quite coarse, the cubic splines has successfully smoothed out the trajectory for the Stewart-Gough platform without introducing any singularities. The singularity profile along the generated path is shown in Figure 7d, where it remains positive (for the positive aspect) throughout the trajectory. The numerical scale in the figure is arbitrary.



(a) Positive aspect

(b) Negative aspect

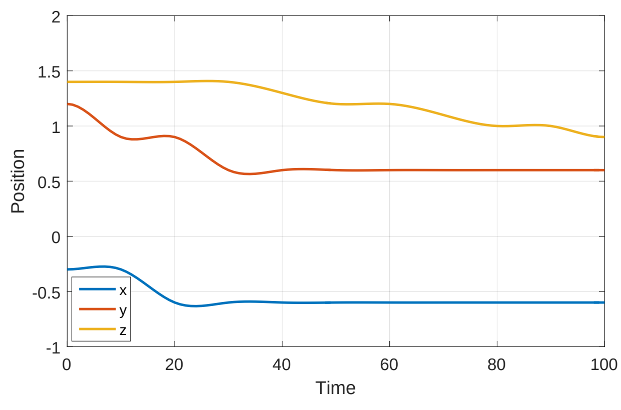
(c) View underneath the workspace, showing recesses

Figure 6: Total reachable workspace of the 3-3 SGP. Note that both positive and negative aspect workspaces share the same bottom profile.

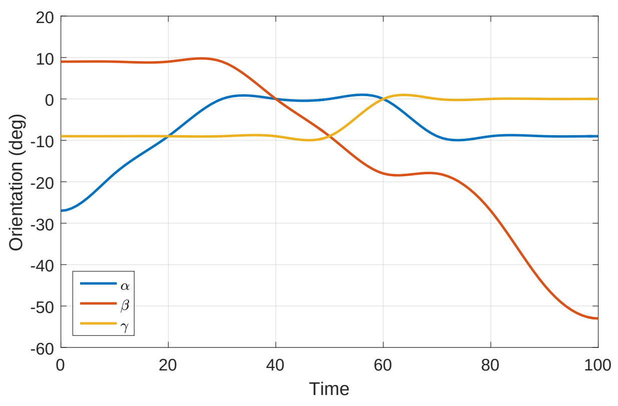
The typical time taken for the PRM to find a valid path was almost instantaneous in this example, in 0.004 seconds. However the algorithm was allowed to improve upon the path found, and the best path was found on the 219th iteration with the PRM stopping at 6.826 seconds. The workspace data is stored entirely in just under 3 GB of memory.

5.4. Path Planning - Assembly Mode Changes

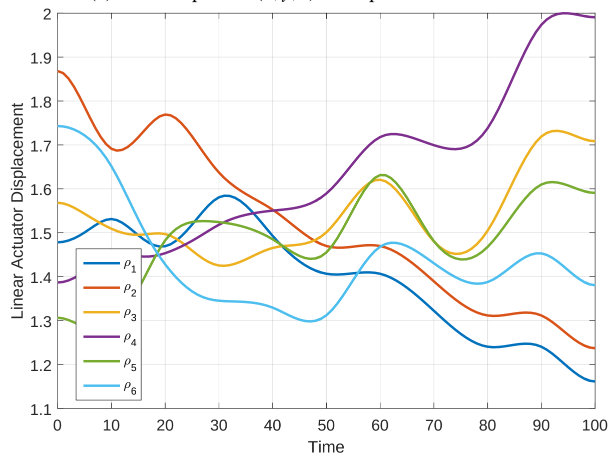
An assembly mode change occurs when both starting and ending configurations share the same joint variables $\mathbf{q}$, but lie in different points in the discretised $C_{\mathbf{x}}$-space. To begin, a starting configuration $\mathbf{x}_0$ in $C_{\mathbf{x}}$ is chosen and its joint variables $\mathbf{q}$ using inverse kinematics are calculated. Then by using direct kinematics, the other solutions for $\mathbf{q}$ (up to 7) are found and mapped back into the task space variables $\mathbf{x}$. Any solutions for $\mathbf{x}$ that lie beyond the defined boundaries of task space are automatically discarded. Finally, the remaining solutions of $\mathbf{x}$ are mapped back into the discretised space of $C_{\mathbf{x}}$ and ranked, based on the difference of the configuration of the platform given by the solution,



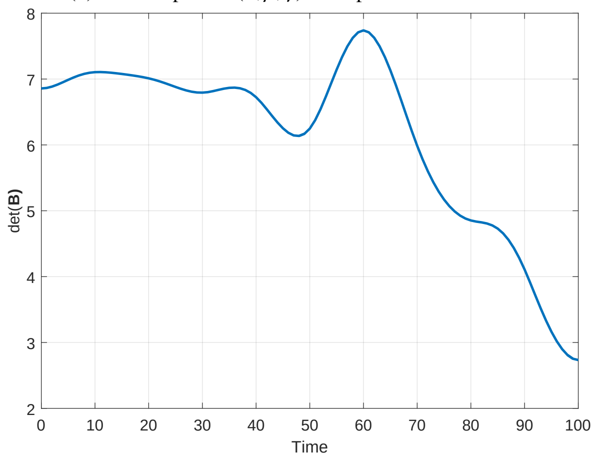
(a) Planned path in (x, y, z) task space co-ordinates



(b) Planned path in (α, β, γ) task space co-ordinates



(c) Planned path in joint space co-ordinates



(d) Parallel singularity profile of the proposed path

Figure 7: Path planning results between two different positions p_1 and p_2 in the configuration space.

with the closest configuration of the platform defined in the discretised $C_{\mathbf{x}}$-space:

$$\Delta_n = \sqrt{(P_{c1} - P_{n1})^2 + (P_{c2} - P_{n2})^2 + (P_{c3} - P_{n3})^2} \tag{22}$$

An assembly mode change is proposed from p_{x1} in the task space *negative* aspect (angle in degrees)

$$p_{x1} = \begin{bmatrix} 0.30 & 0.90 & 0.40 & -90° & -45° & 45° \end{bmatrix}. \tag{23}$$

Its associated point in joint space is

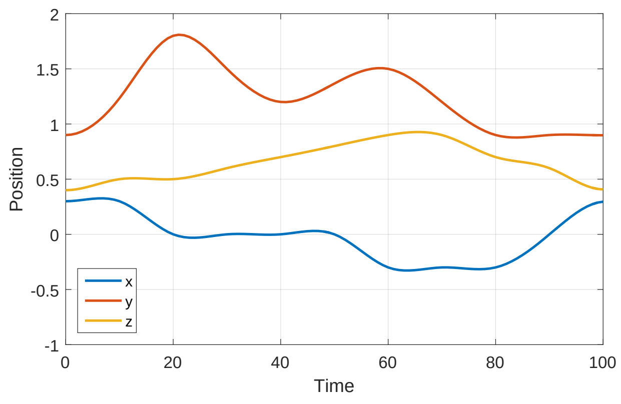
$$p_{q1} = \begin{bmatrix} 2.00 & 1.21 & 1.03 & 1.12 & 1.56 & 1.17 \end{bmatrix}. \tag{24}$$

According to this joint space configuration, there are three more task space solutions in the negative aspect (not including the original point p_{x1}). Ranked on the criteria based on Equation (22), the proposed task space co-ordinates to perform this assembly mode change is

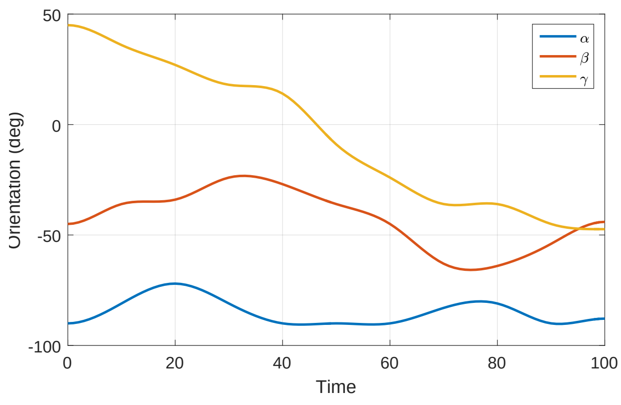
$$p_{x2} = \begin{bmatrix} 0.30 & 0.90 & 0.41 & -87.6° & -43.9° & -47.6° \end{bmatrix}, \tag{25}$$

where its nearest-neighbour cell defined in the discretised $C_{\mathbf{x}}$-space is

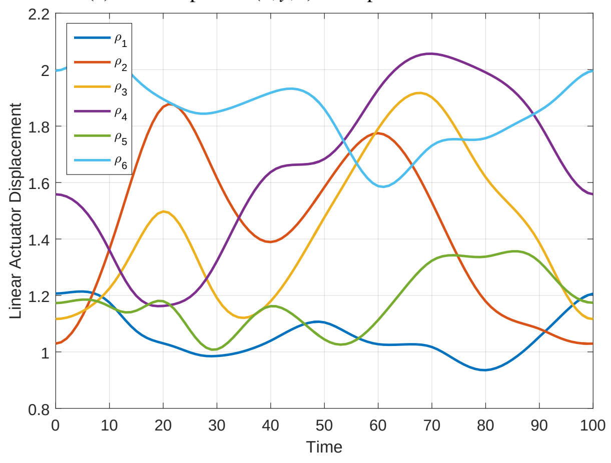
$$p_{c2} = \begin{bmatrix} 0.30 & 0.90 & 0.40 & -90° & -45° & -45° \end{bmatrix}. \tag{26}$$



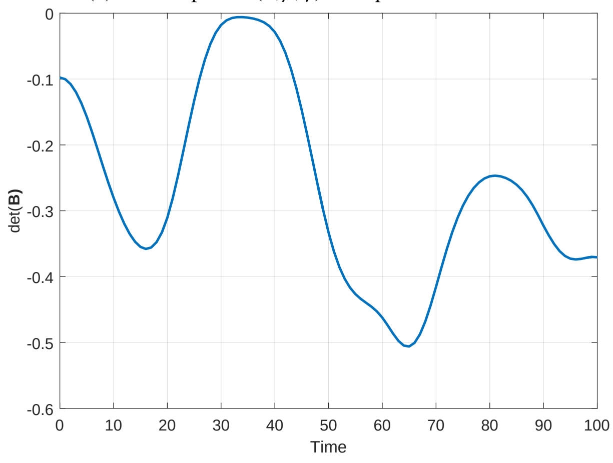
(a) Planned path in (x, y, z) task space co-ordinates



(b) Planned path in (α, β, γ) task space co-ordinates



(c) Planned path in joint space co-ordinates



(d) Parallel singularity profile of the proposed path. Numerical scale in the y-axis is arbitrary.

Figure 8: Path planning results between assembly modes p_{x1} and p_{x2} in the configuration space.

Therefore the entire path in the task space, or $C_{\mathbf{x}}$-space, will consist of the following points

$$p_{x1} \to p_{c2} \to p_{x2}, \tag{27}$$

where the path between the starting configuration p_{x1} and the discretised point in $C_{\mathbf{x}}$-space p_{c2} is found using PRM, and the path between p_{c2} and p_{x2} is a linear path to the goal configuration.

Figures 8a and 8b show the results of the assembly mode-change trajectory, after the path has been smoothed using cubic splines.

5.4.1. Results

The PRM planner completed in 345 iterations with 3 via points. A path was found in 5.220 s, but converged to the best solution of 30.470 s.

Utilising path smoothing via cubic splines (8 cubics, 1 cubic every 10th point), it is observed that in the joint space path in Figure 8c, both start and end points are exactly the same. Figures 8a and 8b also show differing start and end points in the task space. This shows that the proposed trajectory planned in the task space has successfully linked two assembly modes together. The singularity profile of the proposed trajectory in Figure 8d remains negative and free of singularities, even after path smoothing using cubic splines, indicating a valid path between these two configurations.

It is observed that the end effector pose between start and end points are usually close in position, but differ greatly in orientation in at least one rotational parameter. Also, the initial and final poses for assembly mode changes relatively closer to singularity cells than in general path planning.

5.5. Comments on Results

5.5.1. Aspects

Based on the workspace analysis in Section 5.1 and 5.2, there is evidence that the entire 3-3 SGP workspace is made up of only two aspects: one positive and one negative parallel Jacobian. In the example shown in Section 5.1, the orientation was set constant and joint space box-constrained such that the manipulator was forced into positive aspect. As a result, only one large connected workspace in the positive aspect was defined as seen in Figures 5a - 5c. Also, as seen in the workspace analysis example in Section 5.2, with the same joint space constraints and orientation space box-constrained, two fully connected workspaces of differing aspects were observed (Figure 6).

Another interesting observation is that in the numerical experiments conducted, where multiple direct kinematic solutions lie within the bounded region of workspace in the same aspect, they are always connected.

5.5.2. Discretisation Resolution

The resolution at which workspace analysis and path planning occurs, depends on the system resources available on a given computing environment. It is likely to be memory-constrained rather than processing power-constrained. Based on numerical experiments, the optimum resolution for resources available for a typical desktop system in today's age (2015) is 21 elements in each dimension. The approximate memory usage in this scenario is around 3 GB with around a 10 minute pre-processing time.

Although the discretisation resolution can be improved, this results in exponential growth in both memory usage and time. However, the discretisation can be thought of as obtaining an initial picture, in that it can be first used to determine path feasibility. After a path is verified, then the coarsely discretised path can be improved in a post-processing routine to smooth out and improve the accuracy of the desired path. This process is described in Section 4.2. In this case, it was rare that a path generated was not feasible after the smoothing process. Currently, trajectories where singularities introduced by path smoothing are re-calculated by the PRM or discarded.

5.5.3. Effects of Costing

All cells defined in the $C_{\mathbf{x}}$-space are costed based on cell distance from the nearest boundary cell. By doing this, as the PRM is iterated it will converge to the lowest-cost path. Ideally, the lowest-cost path will maintain a safe distance from the boundary cells which represent the parallel singularities. In Figure 9, the effect of this path costing is observed for the same starting and end points in the $C_{\mathbf{x}}$-space. By removing the cost function, any singularity avoidance mechanisms in the PRM are turned off, and results in a shortest-distance path based on cell count. In this case, the path may track closely to a parallel singularity in order to get the shortest distance path, as indicated by its low parallel Jacobian determinant det(B) relative to Path 1. The path with costing (Path 1) is longer, but is in a better position to complete a singularity-free path based on det(B), if errors introduced by discretisation or path-smoothing are prevalent.

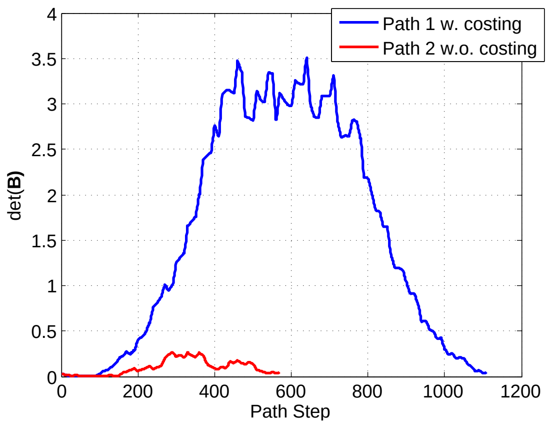


Figure 9: Parallel singularity analysis of a costed versus non-costed PRM-planned path.

6. Conclusion

The workspace analysis and path planning algorithm on the 3-3 symmetric Stewart-Gough platform were proposed and demonstrated. The kinematics were derived from the 6-3 Stewart-Gough platform, and so this work is also compatible for this platform with a number of parameter changes because the path planning and flood-fill algorithms (PRM and flood fill) were not dependent on kinematics. The full 6-DOF capability of this mechanism was fully utilised in all examples, and results for workspace volume were verifiable from previous works. Volumetric calculations for the unconstrained orientation space can be extended because of generality of the algorithm, and calculated results are consistent between different resolutions of discretisation as well as being comparable to previous works. Based on workspace analysis for the current configuration of the 3-3 SGP, there is evidence that this platform configuration exhibits only two aspects in the workspace.

In the path planning aspect of this work, it was shown by example that the 3-3 SGP is capable of assembly mode changes. All trajectories generated were accurate, singularity-free and time performance was feasible. The kinematic path planning for the Stewart-Gough platform is not difficult as long as the configuration space is derived from the task space variables, and so in future work, there is scope to adapt this work to the general 6-DOF SGP. The main challenge is solving the direct kinematic solutions for assembly mode changes. Another challenge to investigate in future work is to determine which assembly modes can be connected, given the natural constrains of the SGP such as actuator limits.

Although the workspace was in six dimensions due to its 6-DOF output, the entire analysis and path planning process can be performed on a standard desktop computer. While the path planning resolution was limited by memory size, the discretisation effect was minimised by an effective path smoothing technique. Results showed that cubic splining introduced minimal errors relating to the parallel singularity. After the workspace was analysed, the path planning was performed with feasible results returned almost instantaneously, but the PRM was allowed to continue to generate better results.

Significant performance gains could be made by using a faster programming language or utilising an adaptive discretisation scheme such as octree, which also has the added benefit of reducing memory usage.

Acknowledgements

This work was supported in part by the Engineering Small Funding Scheme 2016 at Monash University.

Appendix A. 6-3 SGP Kinematics

Referencing Figure 2a, the variable parameters that define the manipulator are:

$$OA = (OA_x, OA_y) \quad OB = (0,0) \quad OC = (OC_x, OC_y) \quad MA = (0,0) \quad MB = (MB_x, 0) \quad MC = (MC_x, MC_y), \tag{A.1}$$

where $\{O\}$ is the base frame and $\{M\}$ is the moving (output) frame. Because this is a symmetric configuration, the parameters should be set such that this property is achieved. These parameters are described in Section 5.

The output parameters $\mathbf{x}$ that describe the location of frame $\{M\}$ with respect to frame $\{O\}$ are

$$\mathbf{x} = \mathbf{x}(x, y, z, \alpha, \beta, \gamma). \tag{A.2}$$

where rotational parameters α, β and γ describe the rotations of $\{M\}$ based on $Z - Y - X$ Euler angles.

Given the position of the output frame $\{M\}(\mathbf{x})$, following the method of inverse kinematics in Section 5, the result of Equations 3 to 8 (i.e. actuated leg lengths) in terms of the parameters OB, OC, MB and MC written in the form of $\mathbf{F}(\mathbf{q}, \mathbf{x})$ where $\mathbf{q}$ and $\mathbf{x}$ are joint and task space variables respectively,

$$F_1(\mathbf{q}, \mathbf{x}) = z^2 + (OB_x - x)^2 + (OB_y - y)^2 - {\rho_1}^2 = 0 \tag{A.3}$$

$$\begin{aligned} F_2(\mathbf{q}, \mathbf{x}) = {} & (z - MB_x s_\beta + MB_y c_\beta s_\gamma)^2 + (y - OB_y + MB_y(c_\alpha c_\gamma + s_\alpha s_\beta s_\gamma) + MB_x c_\beta s_\alpha)^2 + \\ & (OB_x - x + MB_y(c_\gamma s_\alpha - c_\alpha s_\beta s_\gamma) - MB_x c_\alpha c_\beta)^2 - {\rho_2}^2 = 0 \end{aligned} \tag{A.4}$$

$$F_3(\mathbf{q},\mathbf{x}) = (z - MB_x s_\beta + MB_y c_\beta s_\gamma)^2 + (y - OC_y + MB_y(c_\alpha c_\gamma + s_\alpha s_\beta s_\gamma) + MB_x c_\beta s_\alpha)^2 + (OC_x - x + MB_y(c_\gamma s_\alpha - c_\alpha s_\beta s_\gamma) - MB_x c_\alpha c_\beta)^2 - {\rho_3}^2 = 0 \tag{A.5}$$

$$F_4(\mathbf{q},\mathbf{x}) = (z - MC_x s_\beta + MC_y c_\beta s_\gamma)^2 + (y - OC_y + MC_y(c_\alpha c_\gamma + s_\alpha s_\beta s_\gamma) + MC_x c_\beta s_\alpha)^2 + (OC_x - x + MC_y(c_\gamma s_\alpha - c_\alpha s_\beta s_\gamma) - MC_x c_\alpha c_\beta)^2 - {\rho_4}^2 = 0 \tag{A.6}$$

$$F_5(\mathbf{q},\mathbf{x}) = (z - MC_x s_\beta + MC_y c_\beta s_\gamma)^2 + (y - OA_y + MC_y(c_\alpha c_\gamma + s_\alpha s_\beta s_\gamma) + MC_x c_\beta s_\alpha)^2 + (OA_x - x + MC_y(c_\gamma s_\alpha - c_\alpha s_\beta s_\gamma) - MC_x c_\alpha c_\beta)^2 - {\rho_5}^2 = 0 \tag{A.7}$$

$$F_6(\mathbf{q},\mathbf{x}) = z^2 + (OA_x - x)^2 + (OA_y - y)^2 - {\rho_6}^2 = 0. \tag{A.8}$$

The direct kinematics are solved using a 16th order univariate polynomial [43].

Appendix B. Jacobians

By taking $\frac{d\mathbf{F}}{dt}$, the time derivative of Equations A.3 to A.8, the following relation is obtained:

$$\mathbf{A}\dot{\mathbf{q}} = \mathbf{B}\dot{\mathbf{x}} \tag{B.1}$$

where

$$\mathbf{A} = \frac{\partial \mathbf{F}}{\partial \mathbf{q}} = \begin{bmatrix} \rho_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \rho_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \rho_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \rho_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & \rho_6 \end{bmatrix} \qquad \mathbf{B} = \frac{\partial \mathbf{F}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial\rho_1}{\partial x} & \frac{\partial\rho_1}{\partial y} & \frac{\partial\rho_1}{\partial z} & \frac{\partial\rho_1}{\partial \alpha} & \frac{\partial\rho_1}{\partial \beta} & \frac{\partial\rho_1}{\partial \gamma} \\ \frac{\partial\rho_2}{\partial x} & \frac{\partial\rho_2}{\partial y} & z & 0 & 0 & 0 \\ \frac{\partial\rho_3}{\partial x} & \frac{\partial\rho_3}{\partial y} & z & 0 & 0 & 0 \\ \frac{\partial\rho_4}{\partial x} & \frac{\partial\rho_4}{\partial y} & \frac{\partial\rho_4}{\partial z} & \frac{\partial\rho_4}{\partial \alpha} & \frac{\partial\rho_4}{\partial \beta} & \frac{\partial\rho_4}{\partial \gamma} \\ \frac{\partial\rho_5}{\partial x} & \frac{\partial\rho_5}{\partial y} & \frac{\partial\rho_5}{\partial z} & \frac{\partial\rho_5}{\partial \alpha} & \frac{\partial\rho_5}{\partial \beta} & \frac{\partial\rho_5}{\partial \gamma} \\ \frac{\partial\rho_6}{\partial x} & \frac{\partial\rho_6}{\partial y} & \frac{\partial\rho_6}{\partial z} & \frac{\partial\rho_6}{\partial \alpha} & \frac{\partial\rho_6}{\partial \beta} & \frac{\partial\rho_6}{\partial \gamma} \end{bmatrix} \tag{B.2}$$

where

$$\frac{\partial\rho_1}{\partial x} = x - OA_x + MC_x c_\alpha c_\beta - MC_y c_\gamma s_\alpha + MC_y c_\alpha s_\beta s_\gamma \qquad \frac{\partial\rho_2}{\partial x} = x - OA_x$$

$$\frac{\partial\rho_3}{\partial x} = x - OB_x \qquad \frac{\partial\rho_4}{\partial x} = x - OB_x + MB_x c_\alpha c_\beta - MB_y c_\gamma s_\alpha + MB_y c_\alpha s_\beta s_\gamma$$

$$\frac{\partial\rho_5}{\partial x} = x - OC_x + MB_x c_\alpha c_\beta - MB_y c_\gamma s_\alpha + MB_y c_\alpha s_\beta s_\gamma \qquad \frac{\partial\rho_6}{\partial x} = x - OC_x + MC_x c_\alpha c_\beta - MC_y c_\gamma s_\alpha + MC_y c_\alpha s_\beta s_\gamma$$

$$\frac{\partial\rho_1}{\partial y} = y - OA_y + MC_y c_\alpha c_\gamma + MC_x c_\beta s_\alpha + MC_y s_\alpha s_\beta s_\gamma \qquad \frac{\partial\rho_2}{\partial y} = y - OA_y$$

$$\frac{\partial\rho_3}{\partial y} = y - OB_y \qquad \frac{\partial\rho_4}{\partial y} = y - OB_y + MB_y c_\alpha c_\gamma + MB_x c_\beta s_\alpha + MB_y s_\alpha s_\beta s_\gamma$$

$$\frac{\partial\rho_5}{\partial y} = y - OC_y + MB_y c_\alpha c_\gamma + MB_x c_\beta s_\alpha + MB_y s_\alpha s_\beta s_\gamma \qquad \frac{\partial\rho_6}{\partial y} = y - OC_y + MC_y c_\alpha c_\gamma + MC_x c_\beta s_\alpha + MC_y s_\alpha s_\beta s_\gamma$$

$$\frac{\partial\rho_1}{\partial z} = z - MC_x s_\beta + MC_y c_\beta s_\gamma \qquad \frac{\partial\rho_4}{\partial z} = z - MB_x s_\beta + MB_y c_\beta s_\gamma$$

$$\frac{\partial\rho_5}{\partial z} = z - MB_x s_\beta + MB_y c_\beta s_\gamma \qquad \frac{\partial\rho_6}{\partial z} = z - MC_x s_\beta + MC_y c_\beta s_\gamma$$

$$\frac{\partial\rho_1}{\partial \alpha} = MC_y OA_x c_\alpha c_\gamma - MC_x OA_y c_\alpha c_\beta + MC_x OA_x c_\beta s_\alpha + MC_y OA_y c_\gamma s_\alpha + MC_x y c_\alpha c_\beta - MC_y x c_\alpha c_\gamma - MC_x x c_\beta s_\alpha - MC_y y c_\gamma s_\alpha - MC_y OA_y c_\alpha s_\beta s_\gamma + MC_y OA_x s_\alpha s_\beta s_\gamma + MC_y y c_\alpha s_\beta s_\gamma - MC_y x s_\alpha s_\beta s_\gamma$$

$$\begin{aligned}\frac{\partial\rho_4}{\partial\alpha} &= MB_yOB_xc_\alpha c_\gamma - MB_xOB_yc_\alpha c_\beta + MB_xOB_xc_\beta s_\alpha + MB_yOB_yc_\gamma s_\alpha + MB_xyc_\alpha c_\beta - MB_yxc_\alpha c_\gamma - MB_xxc_\beta s_\alpha - \\ &MB_yyc_\gamma s_\alpha - MB_yOB_yc_\alpha s_\beta s_\gamma + MB_yOB_xs_\alpha s_\beta s_\gamma + MB_yyc_\alpha s_\beta s_\gamma - MB_yxs_\alpha s_\beta s_\gamma\end{aligned}$$

$$\begin{aligned}\frac{\partial\rho_5}{\partial\alpha} &= MB_yOC_xc_\alpha c_\gamma - MB_xOC_yc_\alpha c_\beta + MB_xOC_xc_\beta s_\alpha + MB_yOC_yc_\gamma s_\alpha + MB_xyc_\alpha c_\beta - MB_yxc_\alpha c_\gamma - MB_xxc_\beta s_\alpha - \\ &MB_yyc_\gamma s_\alpha - MB_yOC_yc_\alpha s_\beta s_\gamma + MB_yOC_xs_\alpha s_\beta s_\gamma + MB_yyc_\alpha s_\beta s_\gamma - MB_yxs_\alpha s_\beta s_\gamma\end{aligned}$$

$$\begin{aligned}\frac{\partial\rho_6}{\partial\alpha} &= MC_yOC_xc_\alpha c_\gamma - MC_xOC_yc_\alpha c_\beta + MC_xOC_xc_\beta s_\alpha + MC_yOC_yc_\gamma s_\alpha + MC_xyc_\alpha c_\beta - MC_yxc_\alpha c_\gamma - MC_xxc_\beta s_\alpha - \\ &MC_yyc_\gamma s_\alpha - MC_yOC_yc_\alpha s_\beta s_\gamma + MC_yOC_xs_\alpha s_\beta s_\gamma + MC_yyc_\alpha s_\beta s_\gamma - MC_yxs_\alpha s_\beta s_\gamma\end{aligned}$$

$$\begin{aligned}\frac{\partial\rho_1}{\partial\beta} &= MC_xOA_xc_\alpha s_\beta - MC_xzc_\beta + MC_xOA_ys_\alpha s_\beta - MC_xxc_\alpha s_\beta - MC_xys_\alpha s_\beta - MC_yzs_\beta s_\gamma - MC_yOA_xc_\alpha c_\beta s_\gamma - \\ &MC_yOA_yc_\beta s_\alpha s_\gamma + MC_yxc_\alpha c_\beta s_\gamma + MC_yyc_\beta s_\alpha s_\gamma\end{aligned}$$

$$\begin{aligned}\frac{\partial\rho_4}{\partial\beta} &= MB_xOB_xc_\alpha s_\beta - MB_xzc_\beta + MB_xOB_ys_\alpha s_\beta - MB_xxc_\alpha s_\beta - MB_xys_\alpha s_\beta - MB_yzs_\beta s_\gamma - MB_yOB_xc_\alpha c_\beta s_\gamma - \\ &MB_yOB_yc_\beta s_\alpha s_\gamma + MB_yxc_\alpha c_\beta s_\gamma + MB_yyc_\beta s_\alpha s_\gamma\end{aligned}$$

$$\begin{aligned}\frac{\partial\rho_5}{\partial\beta} &= MB_xOC_xc_\alpha s_\beta - MB_xzc_\beta + MB_xOC_ys_\alpha s_\beta - MB_xxc_\alpha s_\beta - MB_xys_\alpha s_\beta - MB_yzs_\beta s_\gamma - MB_yOC_xc_\alpha c_\beta s_\gamma - \\ &MB_yOC_yc_\beta s_\alpha s_\gamma + MB_yxc_\alpha c_\beta s_\gamma + MB_yyc_\beta s_\alpha s_\gamma\end{aligned}$$

$$\begin{aligned}\frac{\partial\rho_6}{\partial\beta} &= MC_xOC_xc_\alpha s_\beta - MC_xzc_\beta + MC_xOC_ys_\alpha s_\beta - MC_xxc_\alpha s_\beta - MC_xys_\alpha s_\beta - MC_yzs_\beta s_\gamma - MC_yOC_xc_\alpha c_\beta s_\gamma - \\ &MC_yOC_yc_\beta s_\alpha s_\gamma + MC_yxc_\alpha c_\beta s_\gamma + MC_yyc_\beta s_\alpha s_\gamma\end{aligned}$$

$$\frac{\partial\rho_1}{\partial\gamma} = MC_y(OA_yc_\alpha s_\gamma - OA_xs_\alpha s_\gamma + zc_\beta c_\gamma - yc_\alpha s_\gamma + xs_\alpha s_\gamma - OA_xc_\alpha c_\gamma s_\beta - OA_yc_\gamma s_\alpha s_\beta + xc_\alpha c_\gamma s_\beta + yc_\gamma s_\alpha s_\beta)$$

$$\frac{\partial\rho_4}{\partial\gamma} = MB_y(OB_yc_\alpha s_\gamma - OB_xs_\alpha s_\gamma + zc_\beta c_\gamma - yc_\alpha s_\gamma + xs_\alpha s_\gamma - OB_xc_\alpha c_\gamma s_\beta - OB_yc_\gamma s_\alpha s_\beta + xc_\alpha c_\gamma s_\beta + yc_\gamma s_\alpha s_\beta)$$

$$\frac{\partial\rho_5}{\partial\gamma} = MB_y(OC_yc_\alpha s_\gamma - OC_xs_\alpha s_\gamma + zc_\beta c_\gamma - yc_\alpha s_\gamma + xs_\alpha s_\gamma - OC_xc_\alpha c_\gamma s_\beta - OC_yc_\gamma s_\alpha s_\beta + xc_\alpha c_\gamma s_\beta + yc_\gamma s_\alpha s_\beta)$$

$$\frac{\partial\rho_6}{\partial\gamma} = MC_y(OC_yc_\alpha s_\gamma - OC_xs_\alpha s_\gamma + zc_\beta c_\gamma - yc_\alpha s_\gamma + xs_\alpha s_\gamma - OC_xc_\alpha c_\gamma s_\beta - OC_yc_\gamma s_\alpha s_\beta + xc_\alpha c_\gamma s_\beta + yc_\gamma s_\alpha s_\beta).$$

[1] V. E. Gough, Contribution to discussion of papers on research in Automobile Stability, Control and Tyre performance, Proc. Auto Div Inst. Mech. Eng (1956) 392–394.

[2] D. Stewart, A Platform with Six Degrees of Freedom, Proc. Institution of Mechanical Engineers 180 (15) (1965) 371–386.

[3] E. Fichter, A Stewart Platform-Based Manipulator: General Theory and Practical Construction, The International Journal of Robotics Research 5 (2) (1986) 157–182.

[4] J.-P. Merlet, Singular Configurations of Parallel Manipulators and Grassmann Geometry, The International Journal of Robotics Research 8 (5) (1989) 45–56.

[5] P. Nanua, K. J. Waldron, V. Murthy, Direct Kinematic Solution of a Stewart Platform, IEEE Transactions on Robotics and Automation 6 (4) (1990) 438–444.

[6] K. H. Hunt, E. Primrose, Assembly Configurations of Some In-Parallel-Actuated Manipulators, Mechanism and Machine Theory 28 (1) (1993) 31–42.

[7] M. L. Husty, An algorithm for solving the direct kinematics of the general Stewart-Gough platforms, Mechanism and Machine Theory 31 (4) (1996) 365–380.

[8] K. Liu, F. L. Lewis, M. Fitzgerald, Solution of Nonlinear Kinematics of a Parallel-Link Constrained Stewart Platform Manipulator, Circuits Systems Signal Process 13 (1994) 167–183.

[9] X. Zhao, S. Peng, A successive approximation algorithm for the direct position analysis of parallel manipulators, Mechanism and Machine Theory 35 (2000) 1095–1101.

[10] C. C. Nguyen, S. S. Antrazi, J.-Y. Park, X.-L. Zhou, Trajectory Planning and Control of a Stewart-Platform-Based End-Effector with Passive Compliance for Part Assembly, Journal of Intelligent and Robotic Systems 6 (1992) 263–281.

[11] B. Dasgupta, T. S. Mruthyunjaya, Singularity-free path planning for the Stewart Platform manipulator, Mechanism and Machine Theory 33 (1998) 711–725.

[12] A. K. Dash, I.-M. Chen, S. H. Yeo, G. Yang, Singularity-Free Path Planning of Parallel Manipulators Using Clustering Algorithm and Line Geometry, in: Proceedings of the 2003 IEEE International Conference on Robotics & Autonomation, Taipei, Taiwan, 2003, pp. 761–766.

[13] H. Li, C. M. Gosselin, M. J. Richard, Determination of the maximal singularity-free zones in the six-dimensional workspace of the general Gough-Stewart platform, Mechanism and Machine Theory 42 (2007) 497–511.

[14] Q. Jiang, C. M. Gosselin, Evaluation and Representation of the Theoretical Orientation Workspace of the Gough-Stewart Platform, Journal of Mechanisms and Robotics 1 (2009) 1–9.

[15] C. Gosselin, Determination of the Workspace of 6-DOF Parallel Manipulators, Journal of Mechanical Design 112 (1990) 331–336.

[16] Q. Jiang, C. M. Gosselin, The Maximal Singularity-Free Workspace of the Gough-Stewart Platform for a given Orientation, Journal of Mechanical Design 130 (2008) 1–8.

[17] O. Bohigas, D. Zlatanov, L. Ros, M. Manubens, J. M. Porta, A General Method for the Numerical Computation of Manipulator Singularity Sets, IEEE Transactions on Robotics 30 (2) (2014) 340–351.

[18] J. P. Merlet, Determination of 6D Workspaces of Gough-Type Parallel Manipulator and Comparison between Different Geometries, Journal of Mechanisms and Robotics 18 (1999) 902–916.

[19] F. Pernkopf, M. L. Husty, Workspace analysis of Stewart-Gough-type parallel manpulators, Journal of Mechanical Engineering Science 220 (7) (2006) 1019–1032.

[20] S. M. LaValle, Planning Algorithms, Cambridge, 2006.

[21] P. Wenger, D. Chablat, Uniqueness Domains in the Workspace of Parallel Manipulators, in: IFAC-SYROCO, Vol. 2, Nantes, 1997, pp. 431–436.

[22] A. Kaufman, Volume Graphics, London: Springer, 2000.

[23] P. Soille, Morphological Image Analysis, Springer-Verlag, 1999.

[24] L. E. Kavraki, P. Svestka, J.-C. Latombe, M. H. Overmars, Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces, IEEE Transactions on Robotics and Autonomation 12 (4) (1996) 566–580.

[25] D. Hsu, J.-C. Latombe, H. Kurniawati, On the Probabilistic Foundations of Probabilistic Roadmap Planning, The International Journal of Robotics Research 25 (7) (2006) 627–643.

[26] W. Au, H. Chung, C. Chen, Path Planning of Planar Parallel Mechanisms Using Global Workspace Road Maps, in: ASME 2012 International Design Engineering Technical Conferences and CIE, Chicago IL, USA, 2012.

[27] E. Macho, O. Altuzarra, C. Pinto, A. Hernandez, Workspaces associated to assembly modes of the 5R planar parallel manipulator, Robotica 26 (2008) 395–403.

[28] J. A. Pámanes, P. Wenger, J. L. Zapata, Motion Planning of Redundant Manipulators for Specified Trajectory Tasks, in: J. Lenarčič, F. Thomas (Eds.), Advances in Robot Kinematics, Kluwer Academic Publishers, 2002, pp. 203–212.

[29] S. Kucuk, Energy minimization for 3-RRR fully planar parallel manipulator using particle swarm optimization, Mechanism and Machine Theory 62 (2012) 129–149.

[30] D. Chablat, P. Wenger, Working modes and aspects in fully parallel manipulators, in: Proceedings of the 1998 IEEE ICRA, Leuven, Belgium, 1998, pp. 1964–1969.

[31] J. J. Craig, Introduction to Robotics Mechanics and Control, 3rd Edition, Prentice Hall, 2005.

[32] D. Chablat, G. Moroz, P. Wenger, Uniqueness domains and non singular assembly mode changing trajectories, in: 2011 IEEE International Conference on Robotics and Automation, Shanghai International Conference Center, 2011, pp. 3946–3951.

[33] C. Innocenti, V. Parenti-Castelli, Singularity-Free Evolution From One Configuration to Another in Serial and Fully-Parallel Manipulators, Journal of Mechanical Design 120 (1998) 73–79.

[34] M. Coste, A Simple Proof That Generic 3-RPR Manipulators Have Two Aspects, Journal of Mechanisms and Robotics 4 (2012) 1–6.

[35] M. Zein, P. Wenger, D. Chablat, Non-singular assembly-mode changing motions for 3-R$\underline{\text{P}}$R parallel manipulators, Mechanism and Machine Theory 43 (2008) 480–490.

[36] E. Macho, V. Petuya, O. Altzarra, A. Hernandez, Planning Nonsingular Transitions Between Solutions of the Direct Kinematic Problem From the Joint Space, Journal of Mechanisms and Robotics 4 (2012) 041005–1–9.

[37] E. Macho, O. Alturazza, C. Pinto, A. Hernandez, Transitions between Multiple Solutions of the Direct Kinematic Problem, in: J. Lenarčič, P. Wenger (Eds.), Advances in Robot Kinematics: Analysis and Design, Springer Science+Business Media B.V. 2008, 2008, pp. 301–310.

[38] W. Au, H. Chung, C. Chen, Generation of the Global Workspace Roadmap of the 3-R$\underline{\text{P}}$R using rotary disk search, Mechanism and Machine Theory 78 (2014) 248–262.

[39] W. Au, H. Chung, C. Chen, Path Planning of the 3-R$\underline{\text{P}}$R Using Global Workspace Road Maps, in: 3rd IFToMM International Symposium on Robotics and Mechatronics, Singapore, 2013.

[40] Y. Chen, O. Bottema, B. Roth, Rational rotation functions and the special points of rational algebraic motions in the plane, Mechanism and Machine Theory 17 (1982) 335–348.

[41] W. Au, H. Chung, C. Chen, Path planning for assembly mode changes for the 3-RRR using global workspace roadmaps, in: Proceedings of the Australasian Conference on Robotics and Automation 2014, Melbourne, Australia, 2014.

[42] W. Au, H. Chung, C. Chen, Path planning of the Stewart-Gough Platform in Enhanced Workspace, in: The 14th IFToMM World Congress, Taipei, Taiwan, 2015.

[43] C. Innocenti, V. Parenti-Castelli, Direct Position Analysis of the Stewart Platform Mechanism, Mechanism and Machine Theory 25 (6) (1990) 611–621.

[44] J. P. Merlet, Parallel Robots, 2nd Edition, Springer, 2006.

[45] L. Feng, S. H. Soon, An effective 3D seed fill algorithm, Computer & Graphics 22 (5) (1998) 641–644.

[46] H. Abdellatif, B. Heimann, A novel multiple-heuristic approach for singularity-free motion planning of spatial parallel manipulators, Robotica 26 (2008) 679–689.