Generation of the Global Workspace Roadmap of the 3-R$\underline{P}$R Using Rotary Disk Search

W. Au, H. Chung, C. Chen*

Monash University (Clayton Campus), Department of Mechanical Engineering
Wellington Road, Clayton, Victoria 3800, Australia

Abstract

Path planning for parallel manipulators in the configuration space can be a challenging task due to the existence of multiple direct kinematic solutions. Hence the aim of this paper is to define a generalised hierarchical path planning scheme for trajectory generation between two configurations in the configuration space for manipulators that exhibit more than one solution in their direct kinematics. This process is applied to the 3-R$\underline{P}$R mechanism, constrained to a 2-DOF system by setting active joint parameter ρ_1 to a constant. The overall reachable workspace is discretised and deconstructed into smaller patches, which are then stitched together creating a *global workspace roadmap*. Using the roadmap, path feasibility is obtained and local path planning is used to generate a complete trajectory. This method can determine a singularity-free path between any two connectible points in the configuration space, including assembly mode changes.

1. Introduction

Path planning for parallel manipulators is difficult due to their complicated and restrictive workspace. Parallel singularities in particular pose a significant challenge in path planning because of the high internal forces and torques upon encountering those regions, surmounting to the loss of control of the end effector. Naturally, singularities of any type should be avoided in path planning which often results in a reduction of the reachable workspace. As such, the parallel and serial singularity regions form the boundaries of each smaller disjoint workspace, which forms the basis of assembly and working modes [1, 2]. These small disjoint workspaces, if forcibly bound by their assembly and working modes, can significantly reduce the overall reachable workspace [3].

One solution is to plan around known singularity regions. One can determine all singularity-free regions [4] or constrain manipulator design [5] such that it never encounters singularity configurations. In [6], a variational approach for singularity-free path planning of parallel mechanisms was discussed where the authors opted for a dynamic-based path-finding. Utilising the Lagrangian and potential energies, the results are claimed to be almost optimal in distance and smoothness. Road map, cell decomposition and potential field methods were identified as feasible options for path planning. The same authors attempted to create singularity-free paths for the Stewart platform in [7]. They were successful in generating feasible paths, but the major drawback in their approach is that they assume disjoint workspaces are permanently disconnected. In light of [8], this assumption may be conservative. They have also stated that their results were very sensitive to intersecting singularity hyper-surfaces.

Workspace enlargement through passing of a controllable serial singularity was explored for the 5R planar manipulator [3, 9], but due to the nature of the mechanism, all parallel singularities were entirely uncontrollable and could not be used for workspace expansion or path planning. Much research has been conducted into the possibilities of expanding the overall workspace without having to pass singularities [8]. This application has been applied to the 3-R$\underline{P}$R mechanism, where assembly modes can be changed by encircling a cusp point [10], points of which there are

*Corresponding author.
Email addresses: wesley.au@monash.edu (W. Au), hoam.chung@monash.edu (H. Chung), chao.chen@monash.edu (C. Chen)

three direct kinematic solutions coinciding [11, 12]. However, the path existence depends on which cusp point and starting assembly mode is chosen. Not knowing this, the path feasibility (the success of the assembly mode-change) is unclear until the end effector has traversed partially around the proposed path around a chosen cusp point.

It was also shown that in certain manipulators, assembly mode changes can be made without encircling a cusp point [13]. A case was found where a *double point*, a point where two singularity loci intersect also allows for assembly mode changes when encircled in the configuration space. This phenomenon further complicates trajectory planning whose methods are cusp point-dependent.

Another challenge is path planning for higher DOF parallel manipulators, as time complexity of graph-based searches become insurmountable. Much research has been done in finding ways to accelerate the speed of path planning algorithms in higher dimensions. In [14], a combination of cost map planning and sample-based search is utilised in configuration space. With the use of multiple rapidly-exploring random tree algorithms coupled with heuristic-based guides, it was claimed that the algorithms used are general enough to be applicable to 6-dimensional cases. Sampling based motion planning used in [15] favours boundary-tracking and less-visited areas over a grid-discretised space for algorithm optimisation. Rapidly-exploring random tree (RRT) and probabilistic road map algorithms have been the focus robotics path planning over the last decade and were shown to perform well [16, 17]. Its merits and drawbacks are discussed in detail in [18].

Accelerating the task of characterising the workspace and storing it efficiently in memory by utilising non-uniform discretisation methods exist, such as algebraic cylindrical decomposition [16, 19]. However this requires the workspace boundary to be defined semi-algebraically, in which this condition can not be guaranteed for complex and higher DOF systems. Recursive discretisation such as octree [20] helps alleviate memory constraints of numerical analysis in high-dimension space by adaptively refining the discretisation grid around detailed areas. Interval analysis is a flexible algorithm for solving numerical problems [21]. It can be applied to a variety of problems, but demands high level expertise to formulate an efficient algorithm.

This paper will expand on the concept of path planning in the configuration space of the 3-R$\underline{\text{P}}$R using the hierarchical planning scheme. Path planning was specifically chosen in the configuration space to avoid parallel singularities, which are uncontrollable. Parallel singularities are naturally the boundaries in the map of the number of solutions in the configuration space. This feature makes the path planning easier in the configuration space rather than the task space.

Our proposed hierarchical path planning for the 3-R$\underline{\text{P}}$R will utilise a topological graph representing the overall reachable workspace which will be used for singularity-free trajectory generation, departing from the well-known concept of encircling cusp points. This graph of nodes and links is generated by means of deconstruction of the overall reachable workspace in the joint variables into smaller *workspace patches*, bounded by the parallel singularity locus in the reduced configuration space. However, the location of the singularity may not exist at all solutions of the same configuration, hence the boundaries of each patch are checked for singularities and continuity to an adjacent workspace patch. A boundary that allows passage to an adjacent workspace patch is called a *gate*. The collection of workspace patches and their links via their gates that represent the overall reachable workspace is called the *global workspace roadmap*. After finding this roadmap, we can determine path feasibility between two points or assembly modes in the joint space, and depending on the algorithm used, can explicitly guarantee a result [16] whether a path exists or not. Finally, local path planning will find a path within each workspace patch in the list of individual connected workspaces, generated by the global planning which is then stitched together to form the overall feasible path. The entire path planning process is summarised in Figure 1.

2. Workspace Discretisation and Separation

The entire path-planning scheme relies on the discretisation of the n-dimensional configuration space. We have chosen a simple uniformly-discretised space where each cell centre is equally spaced apart and represents a unique joint configuration. This will allow easy implementation of searching algorithms. We will use the minimal-connectivity scheme possible for any dimension of discretisation, hence in our 2-dimensional case, we use 4-connectivity for connection to each adjacent cell. This means only faces are connected and any diagonal movement through a vertex is not allowed. Because diagonal movement through a vertex is allowed in 8-connectivity, it allows the possibility of passage through an infinitesimally small point in the grid space, possibly represented by an obstacle as seen in Figure 3. Hence singularity-free movement in 8-connectivity cannot be guaranteed.

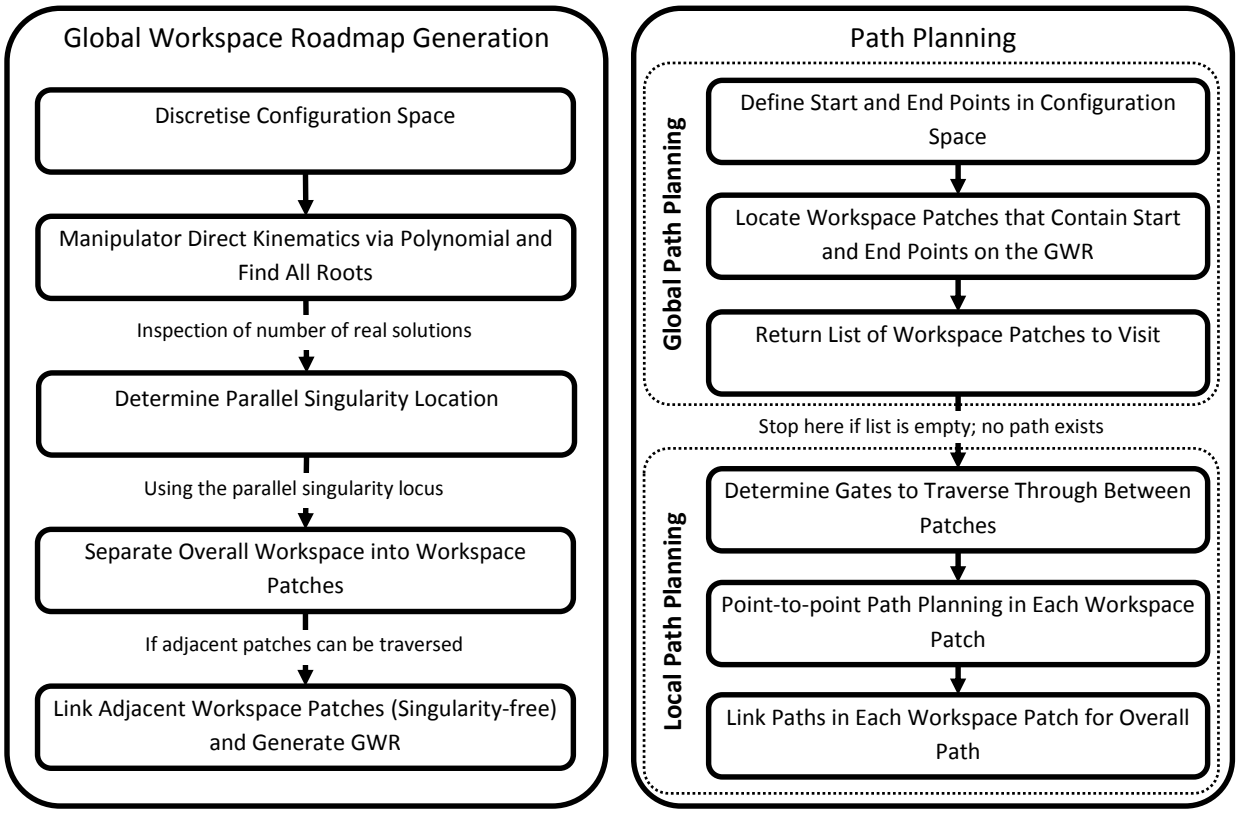


Figure 1: Workflow of the hierarchical path planning scheme.

2.1. Workspace Separation

Working in the configuration space, the parallel singularity locus was used to define the boundaries of our workspace patches for workspace deconstruction. This means that for a configuration space where multiple solutions exist, all solution surfaces and manifolds will have their workspace boundaries defined this way (see Figure 2). The main advantage of this method is that it guarantees each workspace patch can be projected in the dimensions equivalent to the DOF of the system. Hence this method does *not* introduce any further dimensions that the manipulator analysis was originally constrained to. The side effect is that it may introduce more fragmented workspaces than is needed for an efficient solution.

Once the boundaries are defined, each workspace patch can be treated as a workspace and link them together via the global workspace roadmap where a feasible path between patches exist.

2.2. Parallel Singularity Locus and the n-Solution Field

The parallel singularity locus in the joint space can be found in a number of ways. Given the analytical complexity of the determinant of the parallel Jacobian for high DOF manipulators [22], a simpler solution is generally sought after. Geometrical analysis of the manipulator can be used to derive a simpler solution [1], such as that used in [10] for the 3R$\underline{\text{P}}$R, but this method may not feasible for very complex, high DOF systems.

It was shown that a parallel singularity locus exists where there is a solution loss in transitioning between two adjacent configurations in the joint space [23]. In other words, the boundaries of areas of differing number of solutions in the joint space represents a singularity locus. Hence, given the direct kinematics solutions for a mechanism, we can use the *n-solution field* to find the parallel singularity locus without finding or solving any equations related to the Jacobian. The n-solution field is a map showing the number of direct kinematic solutions that exist at each discretised location in the joint space. Normally this can be generated by converting the kinematic equations of the manipulator

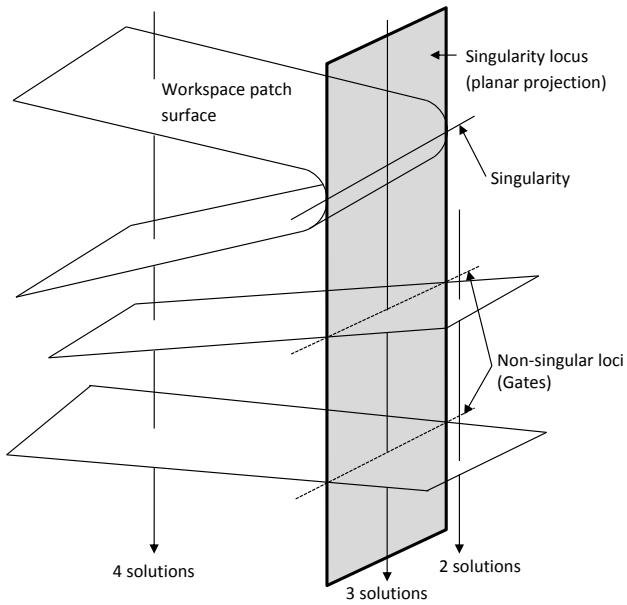


Figure 2: Singularity locus over multiple layers of workspace for a 2-dimensional case.

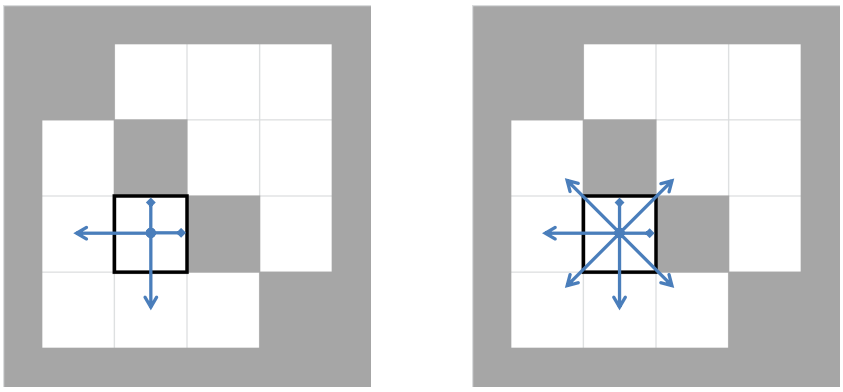
into a single univariate polynomial, and using numerical methods for finding number of real roots. Because the n-solution field is a map of the number of direct kinematic solutions in configuration space, the singularity locus can be easily identified and traced on it.

The location of the parallel singularity locus in the joint space does not necessarily indicate that there is a parallel singularity on all solutions for a particular configuration. A 2-dimensional example of this is shown in Figure 2 where the workspace surface represents the real roots of the univariate polynomial. The condition for the existence of a singularity for a certain layer of workspace is observed in polynomial's root multiplicity where it is greater than 1. This is observed as the workspace surface folding back over itself, where two real roots converge to the same value to cross into the complex plane. This feature gives the 3R$\underline{\text{P}}$R the ability to change assembly modes without encountering parallel singularities, which would not be possible if we have assumed the singularity locus projects its location of singularities to all solution surfaces. That means we must define for each workspace patch, what edge is a singularity and what edge is a gate. The concept of gates is discussed in Section 3.2.

2.3. Finding Connected Workspace

As each patch of workspace is analysed and separated, it is checked for connectivity in task and joint space variables $\mathbf{x}$ and $\mathbf{q}$ respectively where singularities and gates along the workspace patch boundary are identified and marked. The method employed for finding the connected workspace for each layer is the *rotary disk search* algorithm (RDS).

The rotary disk search is a graph-type algorithm that works like a *depth-first search*, but modified to suit the data structures we employed for the discretised configuration space and algorithm termination criteria. Each cell traversed must be adjacent and connected to the previous one, which reduces the chance of numerical connectivity errors compared to a striped search along each dimension. The end result of the search is a tree of connected workspace cells which defines a workspace patch. The cells that lie on the boundary of the workspace patch are assigned as gates or impassable boundaries according to the location of parallel singularities. This process is applied to each workspace patch in the configuration space and consequently, the entire network of workspace patches is created and represented by the global workspace roadmap.



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

Figure 3: Obstacle definitions with 4 and 8-connectivity in 2D. Dark regions indicate singular or obstacle and arrows indicate legal moves from the highlighted cell. 8-connectivity allows unwanted passage through a thinly-defined obstacle into a supposedly disconnected workspace.

2.4. Rotary Disk Search Algorithm

The rotary disk search is based on the *depth-first search*, or *Trémaux's algorithm* [27]. Given a finite, connected graph of vertices and edges, starting from an arbitrary vertex, the DFS will traverse along a connected edge from vertex to vertex, visiting all vertices and terminate. The goal of the algorithm is to scan an entire connected graph and guarantee that at termination of the algorithm that all vertices are visited once. The rotary disk search algorithm works in an identical way. The workspace represented by rotary disks (Figure 4(a)) and edges connecting each adjacent disk is analogous an undirected graph of vertices and edges, where each rotary disk represents a unique location in discretised configuration space. Thus the RDS algorithm is analogous to the DFS, with the exception that the order of the edges connected to each vertex is defined by the rotary disk.

It was proven for Trémaux's algorithm that upon termination, every edge of a graph has been traversed once in each direction [27]. Because Trémaux's algorithm is corollary to the depth first search to which the RDS is based on, the same proof is therefore applicable to the RDS.

The formal definition of the RDS below applies to the rotary disk layout as defined for the discretised workspace.

Let $Q \subset \mathbb{R}^d$ be a set defined by d intervals, that is, $Q \triangleq \{q \in \mathbb{R}^d | l_i \leq q_i \leq u_i,\ i = 1, \ldots, d\}$, where q_i denotes $i-$th element of q, and l_i's and u_i's are real numbers. We discretise Q with N intervals in each dimension, which gives us N^d cells. We now define a rotary disk, which represents a cell. Let us define the index vector $p \in \mathbb{R}^d$ such as

$$p = \begin{bmatrix} k_1 & k_2 & k_3 & \ldots & k_d \end{bmatrix}^T,$$

with positive integers $1 \leq k_1,\ k_2,\ k_3, \cdots, k_d \leq N$. A rotary disk r^p is defined as a set of $2d$ vertices, $r^p \triangleq \{s^p_{+1}, \ldots, s^p_{+d}, s^p_{-1}, \ldots, s^p_{-d}\}$. Each vertex represents a face of a cell in the corresponding direction as shown in Figure 4(a). It is obvious that all vertices in a rotary disk are perfectly connected. For compact notations, we let $\mathbf{1}_i \in \mathbb{R}^d$ be a vector whose $i-$th component is 1 and otherwise zero. Since we only allow face-to-face connections, there exists an undirected edge between s^p_{+i} and $s^{p+\mathbf{1}_i}_{-i}$. Similarly, there exists an edge between s^p_{-i} and $s^{p-\mathbf{1}_i}_{+i}$.

Algorithm 1 shows the code that makes up the rotary disk search. It is implemented as follows:

$G = (R, E)$, an undirected graph G represents a connected workspace where

- $R = \{r^1, r^2, ..., r^n\}$ are the n-number of rotary disks, or vertices in the graph that represents the workspace. Each disk represents a discrete location in d-dimensional space.
- E is the set of all edges. They are links between vertices v^p_{+i} and $v^{p+\mathbf{1}_i}_{-i}$, forming the links between adjacent rotary disks.

The following sets are utilised

- V^p_s, a stack[1] of disk subregions s visited on disk p.
- V_r, a stack of visited rotary disks r that satisfies the criterion $V^p_s \neq \varnothing$ and $V^p_s \neq r^p$.

V_i, a stack of branch point IDs associated to each element in stack V_r.

Functions

$Aspect()$ returns sign(det(J_x)). Evaluated at r, located at the centre of a cell.

$nSoln()$ returns an integer of the number of solutions where r^p is located in the configuration space. Evaluated at r, located at the centre of a cell.

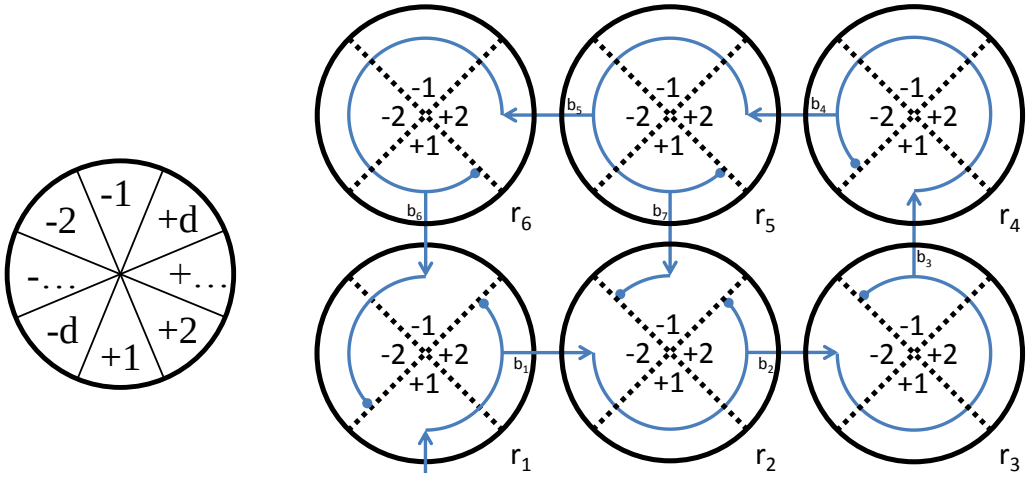
$PosIndex()$ returns the position vector of r^p.

Termination condition

$empty(V_r) = TRUE$.

2.4.1. RDS Work Flow

The general work flow for the RDS algorithm in a connected 2-dimensional workspace can be observed in Figure 4(b). The search begins arbitrarily at a rotary disk r_1 from vertex +1. The algorithm then tracks around the ordered vertices in the rotary disk until it encounters one of two scenarios: 1) a vertex that has a valid connection to a vertex in another rotary disk, and 2) a vertex already visited in the rotary disk. In Case 1, the algorithm will leave the current rotary disk r_1 and enter the next connected rotary disk r_2 via its vertex. At entering r_2, the branch point from the previous disk is recorded, shown as b_1. From r_2 the algorithm continues, branching at b_2 to b_6 until Case 2 is encountered in r_1. Upon reaching a visited vertex in rotary disk r_1, the algorithm will return to the last branch-off point in the previously rotary disk r_6, which is b_6. At b_6 in disk r_6, the algorithm continues to visit vertices in a anti-clockwise direction until Case 1 or 2 is encountered again. Because the next vertex from b_6 is already visited, Case 2 is encountered and the algorithm will return to b_5 in r_5 and continue to visit vertices. This process is repeated until all connected rotary disks are visited, where the algorithm will return to its beginning rotary disk r_1 and terminate.



(a) d-dimensional rotary disk.

(b) Search procedure of a set of 2-dimensional disks in a 2D workspace.

Figure 4: Rotary disk representation where each disk represents a cell and its subregions indicate each edge.

3. Hierarchical Path Planning

The path planning is split into two distinct processes: *global* and *local*. The global level path planning utilises the global workspace roadmap and is used to determine when configurations lie on a workspace patch and if they can be linked together by mechanism reconfiguration or other means. Local level path planning is implemented intra-workspace only, where a range of path-planning techniques can be used to find the best for our needs.

[1] As well-defined in computer science, a stack is a linear list whose addition and deletion happens only at the top of the list. For a given stack A, $pop(A)$ and $push(A, a)$ returns the resulting a stack, where $pop()$ removes the top element of A and $push()$ adds item a to the top of A. $peek(A)$ returns the top element in A. $exists(A, a)$ returns a boolean value whether item a exists in A, and $empty(A)$ returns a boolean value whether the stack is empty.

Algorithm 1: Rotary Disk Search

Data: Discretised configuration space represented by a set of rotary disks R.
p = arbitrary starting index in d-space where $det(J_x) \neq 0$ at this location.
$a = 0$
$W = \{r^p\}$
$S = \{-d, ..., -1, 1, .., d\}$, a set of integers representing all the indices for the vertices in each rotary disk. S_i denotes the i-th component in S. It is assumed that the order of indices is fixed in S.
Output:
W, a set of connected rotary disks found to be in the same workspace patch.
B, a set of rotary disks defined as a boundary of the workspace patch.

```
begin
    while (1) do
        a = (a mod 2d) + 1
        i = S_a
        V_r = push(V_r, r^p)
        V_s^p = push(V_s^p, s_i^p)
        b = (a mod 2d) + 1
        j = S_b
        if exists(V_s^p, s_j^p) then
            V_r = pop(V_r)
            if empty(V_r) then
                return W and B
            else
                p = PosIndex(peek(V_r))
                V_i = pop(V_i)
                i = peek(V_i)
        else
            i = j
            q = p + sign(i)1_|i|
            if r^q ∈ R then
                j = −i
                if !exists(V_s^q, s_j^q) then
                    if nSoln(r^p) = nSoln(r^q) and Aspect(r^p) = Aspect(r^q) then
                        V_i = push(V_i, i)
                        if r^p ∉ W then
                            W = W ∪ r^p
                        p = q
                        i = j
                    else
                        B = B ∪ r^p
```

3.1. Global Path Planning with the Global Workspace Roadmap

The global workspace roadmap (GWR) represents the entire overall reachable workspace in graphical form. Each node represents a single connected workspace patch and its links represent a gate to its adjacent workspace patch. The global path planning process uses this roadmap to generate a path between workspace patches that contain the starting and finishing configurations. Any path found on the global workspace roadmap is called the *global path*.

3.2. Gates

Because the existence of a singularity locus in the configuration space does not mean a singularity exists in all solutions for that configuration, we must define which boundaries of each workspace patch are actually singularities and which are not. The non-singular boundaries of each workspace patch are defined as gates, as observed in Figure 2. These boundaries segregate the overall reachable workspace and provide no hindrance in traversing between adjacent workspace patches. Gates play an important role in the global path planning as they will ensure the correct traversal between each workspace patch as dictated by the global path.

3.3. Local Path Planning and Costing

Local path planning covers path planning between gates or starting and ending configurations in a workspace patch. All local path planning performed in this paper utilises the A* search, which is a graph-based search algorithm with heuristics for robot motion planning for discrete data sets [24, 16]. As the global path planning returns an ordered list of workspaces to traverse (global path), the local path planning handles all planning to and from workspace boundaries and links them together to form an overall path from start to end configurations. The A* search can utilise costing to help find a reasonable path between these boundaries and as with all discrete graph-based searches, is guaranteed to return a pass or fail result [16].

Each data point in the workspace patch is given a cost according to their distance to a closest edge. There is no discrimination whether it is close to a singularity edge or gate, as this ensures that the trajectory generated by the A* search does not track close to any boundaries until passage through a gate is required. It also avoids the more serious problem of creating a trajectory that tracks closely to a singularity.

4. 3-RPR Mechanism

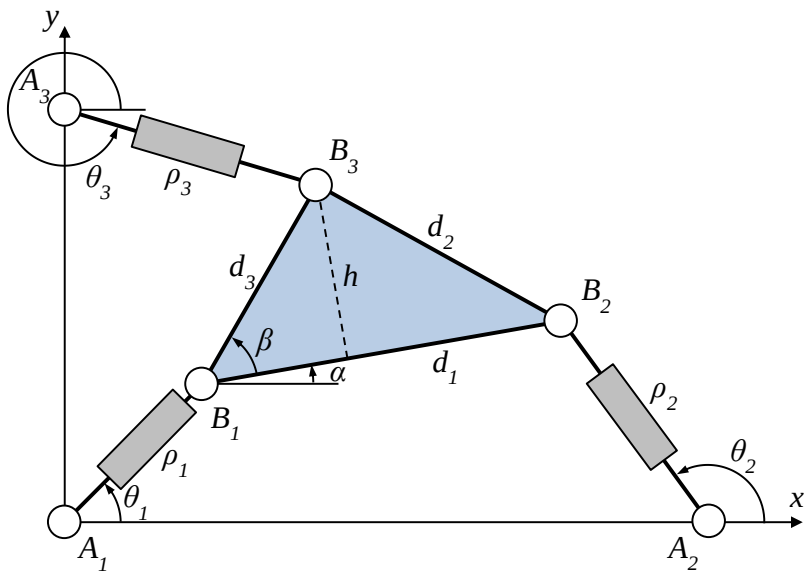


Figure 5: 3-RPR manipulator.

Figure 5 shows the 3-R$\underline{P}$R mechanism under study. It is a 3-DOF planar manipulator with active joint variables $\rho_{1,2,3}$ and passive joint variables $\theta_{1,2,3}$. The planar end effector frame is located at $B_1(x, y)$ with orientation α with respect to the x-axis. The base location for each leg is located on a plane at Cartesian locations $A_{1,2,3}(x, y)$. Constant lengths $d_{1,2,3}$ fully constrain the manipulator's geometry. The parameters are assumed to be $A_1(x, y) = (0, 0)$, $A_2 = (15.91, 0)$, $A_3 = (0, 10)$, $d_1 = 17.04$, $d_3 = 20.84$ arbitrary unit lengths and $\beta = 37°$.

4.1. Kinematics

The relationship between the joint variable $\mathbf{q}$ and the task space variable $\mathbf{x}$ can be written as a set of non-linear algebraic equations

$$\mathbf{F}(\mathbf{q}, \mathbf{x}) = 0, \tag{1}$$

where manipulation of the inverse kinematic equations, the constraint equations in the form (1) are

$$x^2 + y^2 - {\rho_1}^2 = 0 \tag{2}$$

$$(x + d_1 \cos\alpha - x_{a2})^2 + (y + d_1 \sin\alpha - y_{a2})^2 - {\rho_2}^2 = 0 \tag{3}$$

$$(x + d_3 \cos(\alpha + \beta) - x_{a3})^2 + (y + d_3 \sin(\alpha + \beta) - y_{a3})^2 - {\rho_3}^2 = 0. \tag{4}$$

4.1.1. Direct Kinematics

The direct kinematic equations are

$$x = \frac{EG - BH}{AE - BD} \tag{5}$$

$$y = \frac{AH - DG}{AE - BD} \tag{6}$$

$$(EG - BH)^2 + (AH - DG)^2 - (AE - BD)^2 {\rho_1}^2 = 0, \tag{7}$$

where

$$\begin{aligned}
A &= -2A_{2x} + 2d_1 \cos\alpha \\
B &= -2A_{2y} + 2d_1 \sin\alpha \\
C &= d_1^2 + {A_{2x}}^2 + {A_{2y}}^2 - 2d_1 A_{2x} \cos\alpha - 2d_1 A_{y2} \sin\alpha \\
D &= -2A_{3x} + 2d_3 \cos(\alpha + \beta) \\
E &= -2A_{3y} + 2d_3 \sin(\alpha + \beta) \\
F &= {A_{3x}}^2 + {A_{3y}}^2 + {d_3}^2 - 2d_3 A_{3x} \cos(\alpha + \beta) - d_3 A_{3y} 2 \sin(\alpha + \beta) \\
G &= {\rho_2}^2 - {\rho_1}^2 - C \\
H &= {\rho_3}^2 - {\rho_1}^2 - F.
\end{aligned}$$

For a given configuration $\mathbf{q}$ there exist up to six solutions for α as expressed in Equation (7) [22]. Therefore this equation was converted to a 6th order univariate polynomial which allows efficient root-finding algorithms to find all solutions of α.

4.1.2. Singularities and the n-Solution Field

Again starting from Equation (1), performing the time derivative leads to the velocity equation

$$J_q \dot{\mathbf{q}} - J_x \dot{\mathbf{x}} = 0. \tag{8}$$

By calculating the determinants J_q and J_x and equating them to 0, we obtain the parallel and serial singularity equations. This manipulator exhibits a special case for each singularity equation where they depend only on a single set of task or configuration space variables, $J_x = J_x(\mathbf{x})$ and $J_q = J_q(\mathbf{q})$. In particular, the determinant of the serial Jacobian matrix is equates to

$$\det(J_q) = \rho_1 \rho_2 \rho_3 = 0. \tag{9}$$

If we assume that all elements of $\mathbf{q}$ are greater than 0, we can discount serial singularities in the workspace calculations as this condition will never satisfy Equation (9). The parallel singularity equation depends on task space variables $\mathbf{x}$ only and forms hard boundaries in and around the entire workspace.

Figure 6 shows the variables ρ_2 and ρ_3 ($\rho_1 = 17$) with the number of solutions for each configuration shown. The dark lines indicated represent the parallel singularity locus.

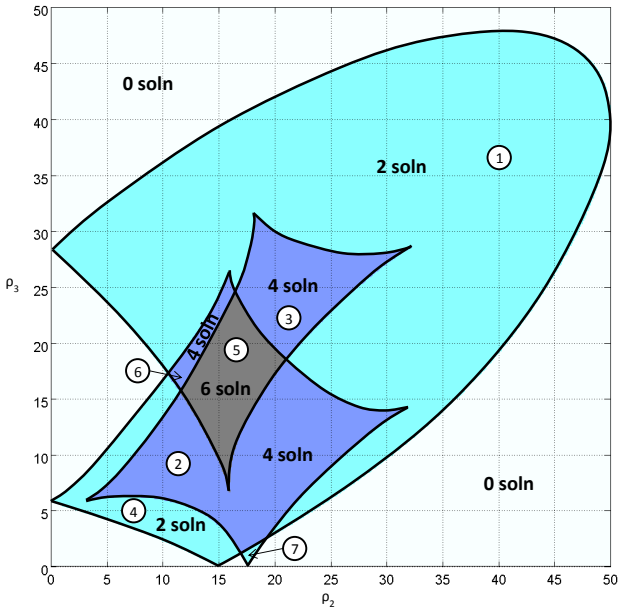


Figure 6: The n-solution field in the configuration space with number of solutions shown. Each field separated by the parallel singularity locus (indicated by the dark lines) is given a unique ID from 1 to 7. This figure is similar to the work seen in [11].

Note that in this example, the reduced configuration space is used, hence this map is valid only at $\rho_1 = 17$. Changing the value of this parameter will cause the workspace patches to grow or shrink, and hence a new map will need to be generated.

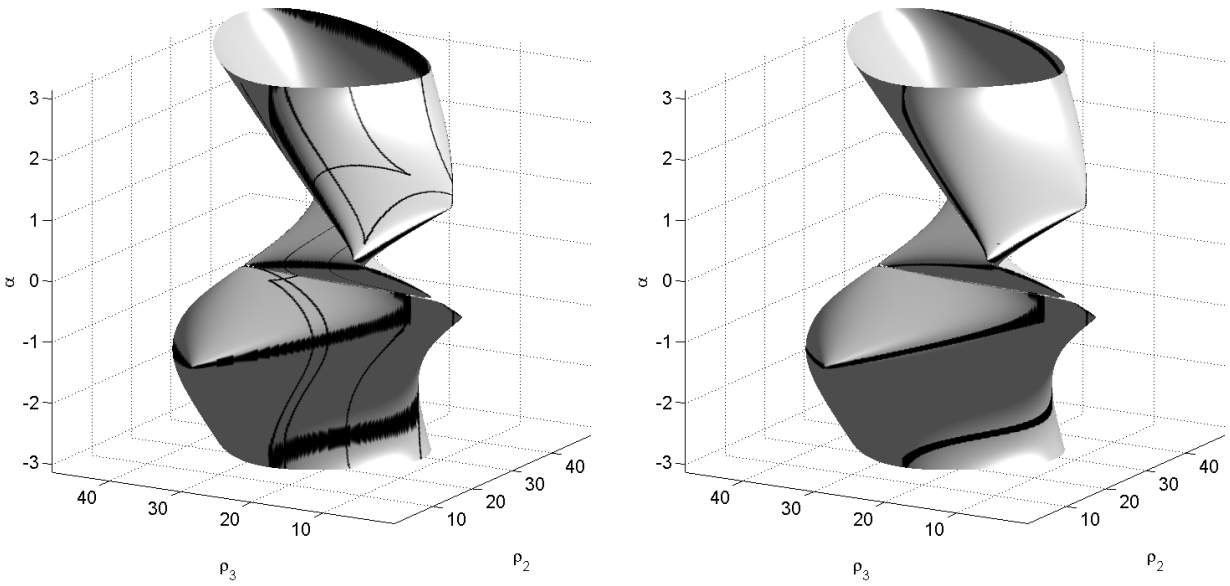
4.2. Reduced Configuration Workspace in (ρ_2, ρ_3, α)-Space

To better understand the (ρ_2, ρ_3) reduced configuration workspace and the paths we generate, we will visualise this in the (ρ_2, ρ_3, α)-space, showing multiple solutions per configuration of ρ. We want to generate a path in the (ρ_2, ρ_3) space, but the existence of up to 6 solutions of α for (ρ_2, ρ_3) means that we must take into account α as a variable in the third dimension in the (ρ_2, ρ_3, α)-space for generating a workspace roadmap, hence the α-axis is not discretised. Then the difficulty of this task is in the separation of the workspaces in an efficient and well-defined manner. The detection of singularity-free assembly mode changes is not well defined [11], therefore we must study the nature of the multiple-solution workspace in the joint variables in order to determine whether there are any other criteria for workspace separation.

Figure 7 shows the workspace in joint variables ρ_2, ρ_3 with the end effector rotation α, which represents the multiple solutions of α per (ρ_2, ρ_3). It is important to note that 3D visualisation of the multi-solution workspace is not needed for path planning purposes, but to demonstrate the theory behind the singularity locus and workspace separation principles.

4.2.1. Workspace Separation

Figure 7(a) shows the workspace with the proposed deconstruction boundaries marked as solid lines, determined by projecting the singularity locus from Figure 6 down the α-axis. Because the n-solution field is actually a 2D projection of the overall reachable workspace surfaces in 3D, splicing the workspace in this way ensures all resulting workspace surfaces remain flat and projectable in 2D. Each region in the n-solution field with a unique ID (shown in



(a) The multilayered and spiralling aspects, sliced according to the singularity locus shown in Figure 6.

(b) The same workspace, but without non-singular lines (gates).

Figure 7: (ρ_2, ρ_3, α)-space representation of the 3-R$\underline{\text{P}}$R workspace.

Figure 6 as encircled numbers) will spawn the number of workspace patches equal to the number of solutions present in that region. For example, Workspace ID 5 will spawn 6 unique 3D workspace patches of the same 2D profile. Because these workspace patches share the same 2D projection, path planning can be performed in 2D on this profile, then simply projected onto the 3D workspace patches.

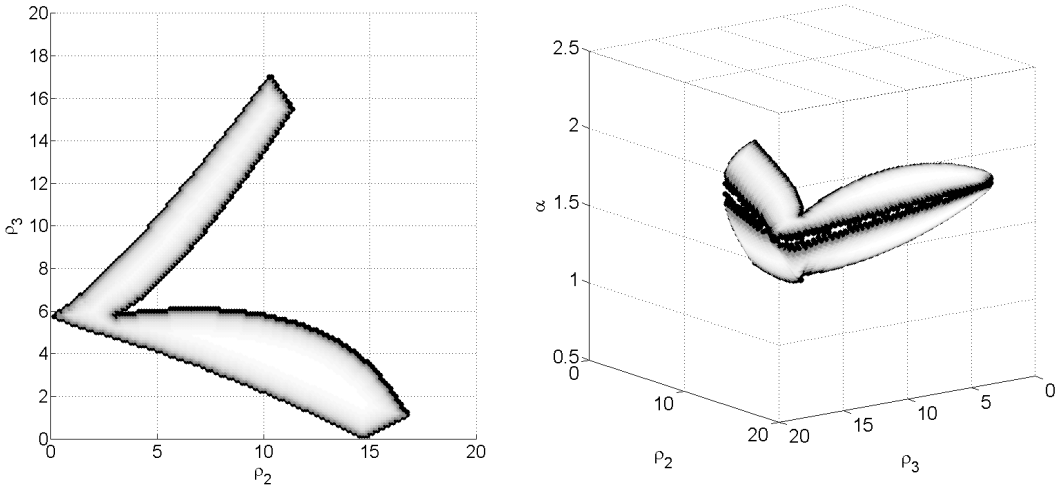
Workspace patch ID numbers are assigned by parent-child numbering system, where the first number represents the ID given in the n-solution field and the second number is given for each patch such that all workspace patches have a unique ID. For example, workspace patches 41 and 42 shown in Figure 8(b) have the same 2D projection as in Figure 8(a), hence local path planning can be performed in 2D, then the path can projected onto the 3D surface of either workspace patch. In total, there are 24 individual workspaces for this configuration of 3-R$\underline{\text{P}}$R.

4.2.2. Aspects

Figure 7(b) shows the identical workspace as in Figure 7(a), but with non-singular boundaries, or gates, removed. This shows that the singularity locus on Figure 6 is valid, but not for all layers of the workspace. This clearly demonstrates the feasibility of singularity-free assembly mode changes for this manipulator. It can also be seen that the regions of parallel singularity show the separation of two aspects, the positive and negative determinant of the parallel Jacobian matrix J_x [11]. It was proven that there exists only two aspects for the 3-R$\underline{\text{P}}$R manipulator in [25].

4.2.3. Degeneracy

The (ρ_2, ρ_3, α)-space alone cannot provide a full description of the end effector configuration. This is because there exist points of degeneracy, where two distinct assembly modes coincide on the same α value but are not singular points [10, 26]. We observe this as a self-intersection of the workspace in the same aspect in the (ρ_2, ρ_3, α)-space. At the point of intersection, Equation (7) degenerates to the case where there are an odd number of unique, real solutions, but there is more than one solution. This is only possible when there are repeated real roots, and these roots have different values of (x, y) in the task space. This can be seen in the root locus plot in Figure 9, which represents a slice of the (ρ_2, ρ_3, α)-space at $\rho_3 = 12$. In the current 3-R$\underline{\text{P}}$R configuration, the mechanism degenerates on two separate α values, at approximately 0.03 and 0.885 radians (1.7° and 50.7°). The equations to find these points were derived in [26]. As a result of this degeneracy, we must check the continuity of *all* task space variables in $\mathbf{x}$, which means



(a) Workspace patch 4 projected on the (ρ_2, ρ_3) plane. (b) Workspace patches 41 and 42 shown in 3D (ρ_2, ρ_3, α)-space.

Figure 8: Workspace ID 4 profile of the 3-R$\underline{\text{P}}$R. There are 2 solutions as stated by the n-solution field and hence 2 separate workspace patches are present with workspace boundaries represented by the dark regions.

utilising (x, y) along with the existing continuity condition of α. In practice for this manipulator, (x, y) is amalgamated into a single value of $\theta_1 = \text{Atan}\left(\frac{y}{x}\right)$ for checking continuity, which is the passive joint angle θ_1 of the first leg of the 3-R$\underline{\text{P}}$R. In fact, the $(\rho_2, \rho_3, \theta_1)$ space produces no degeneracy condition [26], hence it is possible in this case to accurately determine continuity of a workspace using θ_1 only.

Due to the possibility of degeneracy in certain spaces, it is always recommended to check the continuity of all variables ($\mathbf{x}$ and $\mathbf{q}$) to ensure the effects of degeneracy do not produce erroneous results in workspace separation. In fact, continuity-checking conditions can be performed in any co-ordinate system as long as it is accurate. Hence it does not matter which co-ordinate system is used, because the resultant separated workspaces will be the same but represented in a different co-ordinate system.

5. 3-R$\underline{\text{P}}$R Path Planning

In this application, the reduced configuration space is used where $\rho_1 = 17$ and other joint variables are free, limiting the example to a 2-DOF system. All plots in the configuration space have their x and y axes discretised by $\Delta\rho_2 = \Delta\rho_3 = 0.125$ between $[0, 50]$, making a 401×401 grid. For 3D plots where angle α exists, the α-axis is cyclic and bound between $[-\pi, \pi]$.

The hierarchical path planning scheme is applied to two different examples: planning between two separate configurations, and planning for an assembly mode-change for the same active joint configuration. Both local and global path planning use the A* search. Cost-weighting has been implemented for the local path planning for each individual workspace to ensure the path generated does not hug any boundary of a workspace or a singularity boundary and allows the correct gate to be reached. The global planning search is not weighted at this time, therefore the algorithm will complete at the first shortest path encountered in the GWR. At this stage, where multiple solutions are known for the global path, they will be manually acknowledged. Weightings based on the workspace links will be implemented in future works.

All results were generated on MATLAB R2010a except for local path planning (A* search) where it was generated using Visual Basic .Net 2010. The system ran on Windows 7 x64 with an Intel i7 2600K processor at 4.5 GHz with 16 GB of RAM.

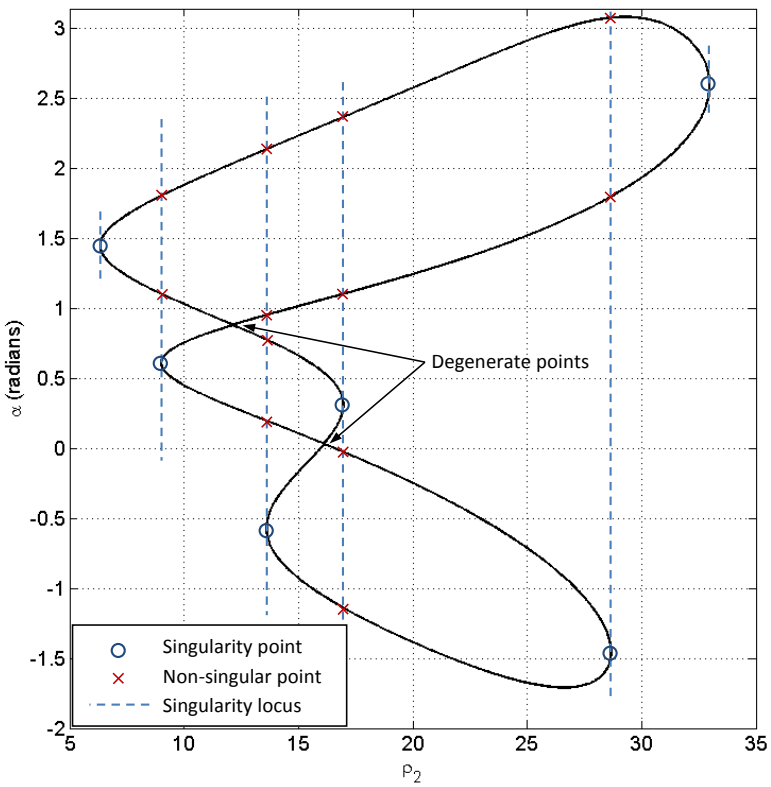


Figure 9: Root locus plot, varying ρ_2 at $\rho_1 = 17$ and $\rho_3 = 12$. Degenerate locations indicated at approximately 0.03 and 0.885 radians.

5.1. Global Workspace Roadmap

The GWR for a reduced configurations space is also only valid for the specific parameter. An example of the changing roadmap is observed in Figure 10 with different values of $\rho_1 = 13, 17, 30$. Note that the GWRs generated clearly show workspace patches are always grouped into two aspects, supporting the theory of this manipulator having only two aspects [25, 11].

5.2. Planning Between Two Separate Configurations

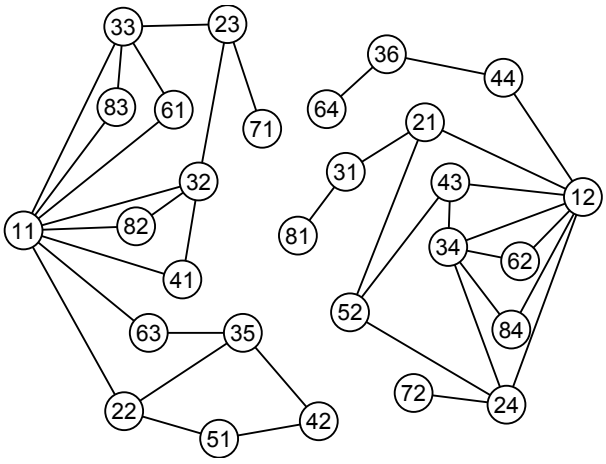
A path is to be planned between two completely unique configurations, starting from q_1 and ending at q_2, where

$$q_1(\rho_2, \rho_3, \alpha) = (12, 18, 122.82^\circ)$$
$$q_2(\rho_2, \rho_3, \alpha) = (10, 25, -24.24^\circ).$$

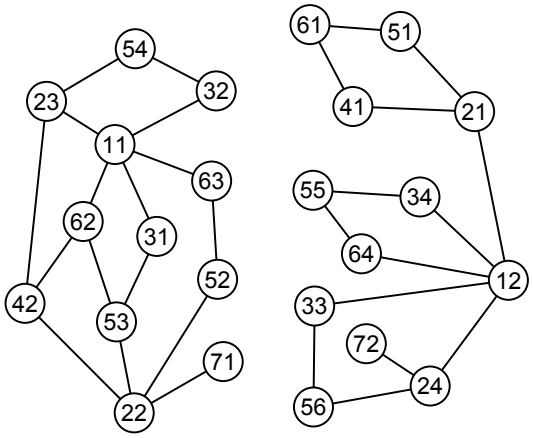
Configuration q_1 lies in workspace 61 and q_2 lies in workspace 12. The proposed global paths generated by the A* search on the GWR ($\rho_1 = 17$, Figure 10(b)) are

$$\text{Global Path 1: } 61 \rightarrow 41 \rightarrow 21 \rightarrow 12$$
$$\text{Global Path 2: } 61 \rightarrow 51 \rightarrow 21 \rightarrow 12.$$

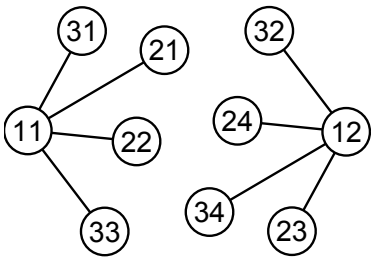
Figure 11 shows the resulting trajectory generated by the local path planning via Path 1, and Figure 12 from Path 2.



(a) $\rho_1 = 13$



(b) $\rho_1 = 17$



(c) $\rho_1 = 30$

Figure 10: GWRs for the 3-RPR manipulator. Generated using *Gephi*. [28].

5.2.1. Remarks

Two possible global paths were suggested by the search in the global workspace roadmap with both having 4 workspace changes. Path 2 creates the more direct path. In both cases, the paths generated by the local path planning are safe from singularities and unused gates, giving ample space to cross the correct gates planned in the GWR. The jagged path is a result of coarse and simplistic discretisation of the configuration space and balancing between costing and search heuristics in the A* search.

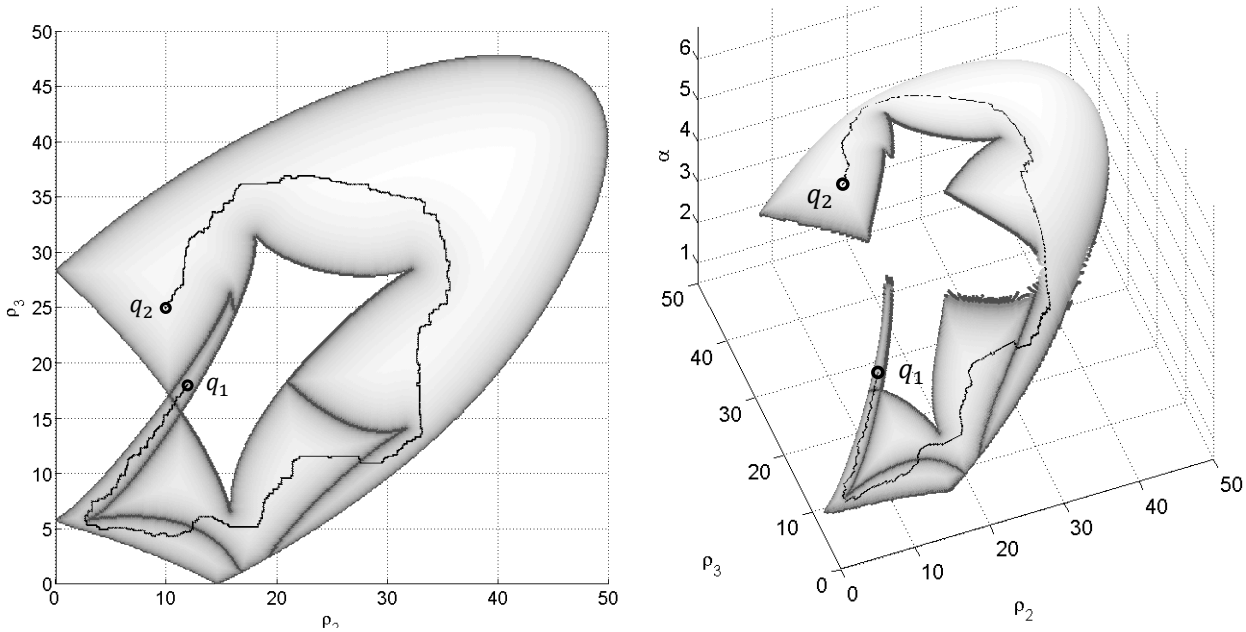
5.3. Planning For Assembly-Mode Change

A path is to be planned for an assembly mode change where the start and end active joint configurations remain the same. In this case,

$$q_1(\rho_2, \rho_3, \alpha) = (20, 25, 29.47^\circ)$$
$$q_2(\rho_2, \rho_3, \alpha) = (20, 25, 96.13^\circ).$$

Configuration q_1 lies in workspace 31 and q_2 lies in workspace 32. The proposed global path generated by the A* search on the GWR is

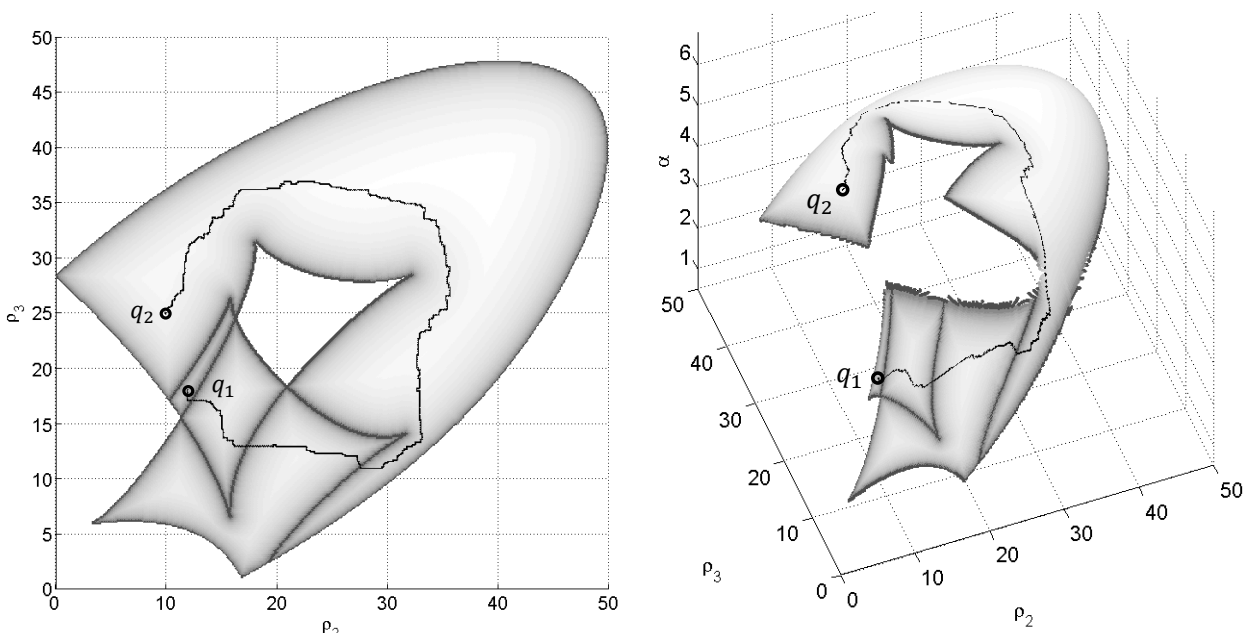
$$\text{Global Path: } 31 \rightarrow 11 \rightarrow 32.$$



(a) Planar view of the path plan.

(b) (ρ_2, ρ_3, α)-space representation of the path plan.

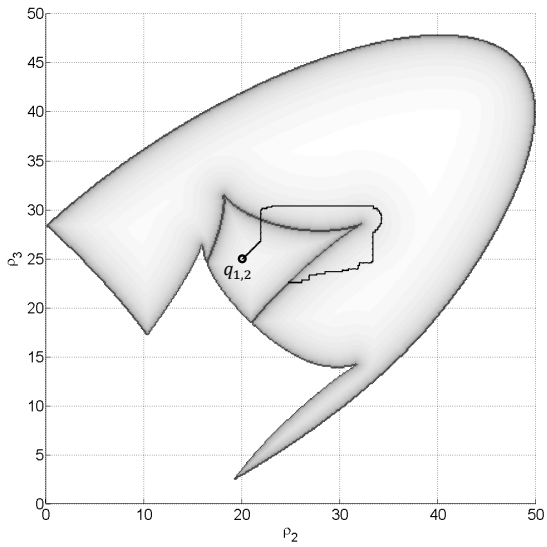
Figure 11: Point to point trajectory via Global Path 1.



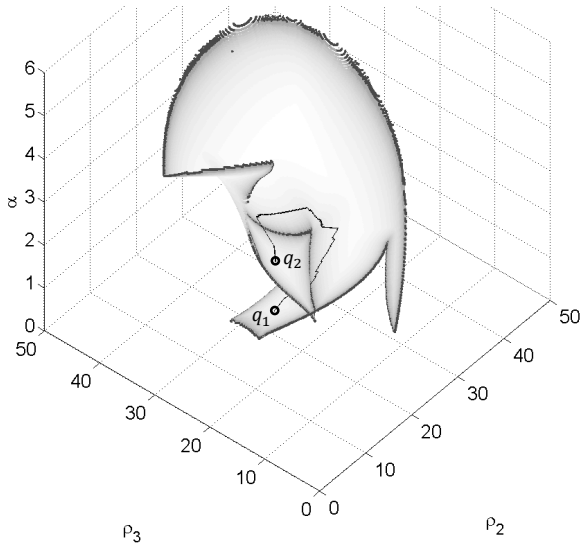
(a) Planar view of the path plan.

(b) (ρ_2, ρ_3, α)-space representation of the path plan.

Figure 12: Point to point trajectory via Global Path 2.



(a) Planar view of the path plan. Note the path loops beneath the top layer.



(b) (ρ_2, ρ_3, α)-space representation of the path plan.

Figure 13: Trajectory for an assembly mode change.

5.3.1. Remarks

A trajectory is successfully generated (Figure 13) that achieves an assembly mode change from its initial position in configuration space. It is observed that a cusp point is encircled and that the path successfully navigates through a degenerate region where the workspace intersects itself.

6. Conclusion and Future Work

The hierarchical path planning method was successfully demonstrated on the 3-R<u>P</u>R manipulator constrained to 2-DOF. The overall workspace decomposition was systematic and fast using the n-solution field, the global workspace roadmap was generated and utilised to find the list of workspaces to traverse. The local path planning scheme was able to generate paths within the workspaces and when combined together, would generate a feasible and reasonable trajectory, given the path constraints.

The methods described for workspace decomposition and path-finding were designed to be as general as possible, opening up the possibility of applying this scheme on similar parallel mechanisms. The n-solution field can be generated for a mechanism whose multiple solutions to its kinematic equations can be determined. This can be achieved by conversion to a univariate polynomial. Using this algebraical method, we can utilise fast root-finding algorithms in our numerical analysis.

We have used a fixed cell size for our discretisation, which could generate more cells than are needed to fully define a workspace. This may result in excessive memory usage particularly in high dimensions. We also risk introducing unwanted features such as discontinuity in the configuration space if the grid is too coarse. Because this paper focuses mainly on the feasible path planning method of the workspaces of parallel manipulators, the grid size was selected such that we know it will not introduce unwanted features. In future work, an adaptive discretisation scheme can be developed for use on higher DOF manipulators.

Improvements can be made upon this work such as the inclusion of cost-weighting the nodes the global workspace roadmap and the refining of the local path planning such as path smoothing. Performance-wise, the A* search algorithm's speed in the local path planning is highly dependent on relationship between the cost and heuristic functions. With the inclusion of higher dimensional data from high DOF mechanisms, code optimisations would have to be implemented.

Even without the local path planning, the information given by the GWR allows us to determine if a path is feasible between any two configurations in any space. This information is valuable for very complex high DOF systems if a quick solution is needed to determine path feasibility.

Acknowledgements

The authors would like to acknowledge the partial support by the ARC Discovery DP098512.

[1] C. Gosselin, J. Angeles, Singularity analysis of closed-loop kinematic chains, IEEE Transactions on Robotics and Automation 6, (June 1990) 281–290.

[2] L. Tsai, Robot Analysis, Wiley, (1999).

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

[4] Q. Jiang, C. M. Gosselin, The maximal singularity-free workspace of planar 3-RPR parallel mechanisms, in: Proc. International Conference on Mechatronics and Automation (2006).

[5] M. Z. Huang, J.-L. Thebert, A study of workspace and singularity characteristics for design of 3-dof planar parallel robots, International Journal of Advanced Manufacturing Technology 51 (2010) 789–797.

[6] S. Sen, B. Dasgupta, A. K. Mallik, Variational approach for singularity-free path-planning of parallel manipulators, Mechanism and Machine Theory 38 (2003) 1165–1183.

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

[8] 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.

[9] W. Au, C. Chen, H. Chung, Path planning of planar parallel mechanisms using global workspace road maps, in: Proc. ASME 2012 International Design Engineering Technical Conferences and CIE (2012).

[10] M. Zein, P. Wenger, D. Chablat, Non-singular assembly-mode changing motions for 3-RPR parallel manipulators, Mechanism and Machine Theory 43 (2008) 480–490.

[11] G. Moroz, F. Rouiller, D. Chablat, P. Wenger, On the determination of cusp points of the 3-RPR parallel manipulators, Mechanism and Machine Theory 45 (11) (2010) 1555–1567.

[12] 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.

[13] E. Macho, O. Alturazza, C. Pinto, A. Hernandez, Transitions between multiple solutions of the direct kinematic problem, in: Advances in Robot Kinematics: Analysis and Design (J. Lenarčič, P. Wenger, eds.), Springer Science+Business Media B.V. (2008) 301–310.

[14] L. Jaillet, J. Cortés, T. Siméon, Sampling-based path planning on configuration-space costmaps, IEEE Transactions on Robotics 26 (4) 2010 635–646.

[15] I. A. Şucan, L. E. Kavraki, A sampling-based tree planner for systems with complex dynamics, IEEE Transactions on Robotics 28 (2012) 116–131.

[16] S. M. LaValle, Planning Algorithms, Cambridge (2006).

[17] S. Karaman, E. Frazzoli, Incremental sampling-based algorithms for optimal motion planning, International Journal of Robotics Research (2010).

[18] D. Hsu, J.-C. Latombe, H. Kurniawati, On the probabilistic foundations of probabilistic roadmap planning, The International Journal of Robotics Research 25 (July 2006) 627–643.

[19] D. Chablat, E. Ottaviano, G. Moroz, A comparative study of 4-cable planar manipulators based on cylindrical algebraic decomposition, in: Proc. ASME IDETC & CIE (2011).

[20] J. Camata, A.L.G.A. Coutinho, Parallel implementation and performance analysis of linear octree finite element mesh generation scheme, Wiley Online Library. DOI: 10.1002/cpe.2869.

[21] D. Chablat, P. Wenger, F. Majou, J.P. Merlet, An interval analysis based study for the design and the comparison of three-degrees-of-freedom parallel kinematic machines, The International Journal of Robotics Research 23 (2004) 615–622.

[22] J. P. Merlet, Parallel Robots, Springer 2nd ed. (2006).

[23] A. K. Zaiter, P. Wenger, D. Chablat, A study of the singularity locus in the joint space of planar parallel manipulators: special focus on cusps and nodes, in: Proc. 4th International Congress Design and Modeling of Mechanical Systems (2011).

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

[25] M. Coste, A simple proof that generic 3-RPR manipulators have two aspects, Journal of Mechanisms and Robotics 4 (2012) 1–6.

[26] P. Wenger, D. Chablat, M. Zein, Degeneracy study of the forward kinematics of planar 3-RPR parallel manipulators, Journal of Mechanical Design 129 (2007) 1265–1268.

[27] S. Even, Graph algorithms 2nd ed., Cambridge (2012) 46–49.

[28] M. Bastian, S. Heymann, M. Jacomy, Gephi: An open source software for exploring and manipulating networks, in: Proc. International AAAI Conference on Weblogs and Social Media (2009).